Context Based Bioinformatics

Gergely Csaba

München 2013

Context Based Bioinformatics

Gergely Csaba

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität
München

vorgelegt von
Gergely Csaba
aus Miskolc

München, den 21.05.2013

Erstgutachter: Prof. Dr. Ralf Zimmer
Zweitgutachter: Prof. Dr. Dmitrij Frishman
Tag der mündlichen Prüfung: 10.05.2013

Eidesstattliche Versicherung

(Siehe Promotionsordnung vom 12.07.11, § 8, Abs. 2 Pkt. .5.)

Hiermit erkläre ich an Eidesstatt, dass die Dissertation von mir selbstständig, ohne unerlaubte Beihilfe angefertigt ist.

Csaba, Gergely

Name, Vorname

München, 20.12.2012

Ort, Datum

Unterschrift Doktorand/in

Formular 3.2

Summary

The goal of bioinformatics is to develop innovative and practical methods and algorithms for biological questions. In many cases, these questions are driven by new biotechnological techniques, especially by genome and cell wide high throughput experiment studies.
In principle there are two approaches:

1. Reduction and abstraction of the question to a clearly defined optimization problem, which can be solved with appropriate and efficient algorithms.

2. Development of context based methods, incorporating as much contextual knowledge as possible in the algorithms, and derivation of practical solutions for relevant biological questions on the high-throughput data. These methods can be often supported by appropriate software tools and visualizations, allowing for interactive evaluation of the results by experts.

Context based methods are often much more complex and require more involved algorithmic techniques to get practical relevant and efficient solutions for real world problems, as in many cases already the simplified abstraction of problems result in NP-hard problem instances. In many cases, to solve these complex problems, one needs to employ efficient data structures and heuristic search methods to solve clearly defined sub-problems using efficient (polynomial) optimization (such as dynamic programming, greedy, path- or tree-algorithms).

In this thesis, we present new methods and analyses addressing open questions of bioinformatics from different contexts by incorporating the corresponding contextual knowledge.
The two main contexts in this thesis are the protein structure similarity context (Part I) and network based interpretation of high-throughput data (Part II).

For the protein structure similarity context **Part I** we analyze the consistency of gold standard structure classification systems and derive a **consistent benchmark set** usable for different applications. We introduce two methods (Vorolign, PPM) for the protein structure similarity recognition problem, based on different features of the structures.
Derived from the idea and results of Vorolign, we introduce the concept of **contact neighborhood potential**, aiming to improve the results of protein fold recognition and threading.
For the re-scoring problem of predicted structure models we introduce the method **Vorescore**, clearly improving the fold-recognition performance, and enabling the evaluation of the contact neighborhood potential for structure prediction methods in general.
We introduce a contact consistent Vorolign variant **ccVorolign** further improving the structure based fold recognition performance, and enabling direct optimization of the neighborhood potential in the future. Due to the enforcement of contact-consistence, the ccVorolign method has much higher computational complexity than the polynomial Vorolign method - the cost of computing interpretable and consistent alignments.
Finally, we introduce a novel structural alignment method (**PPM**) enabling the explicit modeling and handling of phenotypic plasticity in protein structures. We employ PPM for the analysis of

effects of alternative splicing on protein structures. With the help of PPM we test the hypothesis, whether splice isoforms of the same protein can lead to protein structures with different folds (fold transitions).

In **Part II** of the thesis we present methods generating and using context information for the interpretation of high-throughput experiments.

For the generation of context information of molecular regulations we introduce novel textmining approaches extracting relations automatically from scientific publications.
In addition to the fast NER (named entity recognition) method (**syngrep**) we also present a novel, fully ontology-based context-sensitive method (**SynTree**) allowing for the context-specific disambiguation of ambiguous synonyms and resulting in much better identification performance. This context information is important for the interpretation of high-throughput data, but often missing in current databases.
Despite all improvements, the results of automated text-mining methods are error prone. The **RelAnn** application presented in this thesis helps to curate the automatically extracted regulations enabling manual and ontology based curation and annotation.

For the usage of high-throughput data one needs additional methods for data processing, for example methods to map the hundreds of millions short DNA/RNA fragments (so called reads) on a reference genome or transcriptome. Such data (RNA-seq reads) are the output of next generation sequencing methods measured by sequencing machines, which are becoming more and more efficient and affordable.
Other than current state-of-the-art methods, our novel read-mapping method **ContextMap** resolves the occurring ambiguities at the final step of the mapping process, employing thereby the knowledge of the complete set of possible ambiguous mappings. This approach allows for higher precision, even if more nucleotide errors are tolerated in the read mappings in the first step.

The consistence between context information of molecular regulations stored in databases and extracted from textmining against measured data can be used to identify and score consistent regulations (**GGEA**). This method substantially extends the commonly used gene-set based methods such over-representation (ORA) and gene set enrichment analysis (GSEA).

Finally we introduce the novel method **RelExplain**, which uses the extracted contextual knowledge and generates network-based and testable hypotheses for the interpretation of high-throughput data.

Zusammenfassung

Bioinformatik befasst sich mit der Entwicklung innovativer und praktisch einsetzbarer Verfahren und Algorithmen für biologische Fragestellungen. Oft ergeben sich diese Fragestellungen aus neuen Beobachtungs- und Messverfahren, insbesondere neuen Hochdurchsatzverfahren und genom- und zellweiten Studien. Im Prinzip gibt es zwei Vorgehensweisen:

1. Reduktion und Abstraktion der Fragestellung auf ein klar definiertes Optimierungsproblem, das dann mit geeigneten möglichst effizienten Algorithmen gelöst wird.

2. Die Entwicklung von kontext-basierten Verfahren, die möglichst viel Kontextwissen und möglichst viele Randbedingungen in den Algorithmen nutzen, um praktisch relevante Lösungen für relevante biologische Fragestellungen und Hochdurchsatzdaten zu erhalten. Die Verfahren können oft durch geeignete Softwaretools und Visualisierungen unterstützt werden, um eine interaktive Auswertung der Ergebnisse durch Fachwissenschaftler zu ermöglichen.

Kontext-basierte Verfahren sind oft wesentlich aufwändiger und erfordern involviertere algorithmische Techniken um für reale Probleme, deren simplifizierende Abstraktionen schon NP-hart sind, noch praktisch relevante und effiziente Lösungen zu ermöglichen. Oft werden effiziente Datenstrukturen und heuristische Suchverfahren benötigt, die für klar umrissene Teilprobleme auf effiziente (polynomielle) Optimierungsverfahren (z.B. dynamische Programmierung, Greedy, Wege- und Baumverfahren) zurückgreifen und sie entsprechend für das Gesamtverfahren einsetzen.

In dieser Arbeit werden eine Reihe von neuen Methoden und Analysen vorgestellt um offene Fragen der Bioinformatik aus verschiedenen Kontexten durch Verwendung von entsprechendem Kontext-Wissen zu adressieren. Die zwei Hauptkontexte in dieser Arbeit sind (Teil 1) die Ähnlichkeiten von 3D Protein Strukturen und (Teil 2) auf die netzwerkbasierte Interpretation von Hochdurchsatzdaten.

Im Proteinstrukturkontext **Teil 1** analysieren wir die Konsistenz der heute verfügbaren Goldstandards für Proteinstruktur-Klassifikationen, und leiten ein vielseitig einsetzbares **konsistentes Benchmark-Set** ab.
Für eine genauere Bestimmung der Ähnlichkeit von Proteinstrukturen beschreiben wir zwei Methoden (**Vorolign, PPM**), die unterschiedliche Strukturmerkmale nutzen.
Ausgehend von den für Vorolign erzielten Ergebnissen, führen wir **Kontakt-Umgebungs-Potentiale** mit dem Ziel ein, Fold-Erkennung (auf Basis der vorhandenen Strukturen) und Threading (zur Proteinstrukturvorhersage) zu verbessern.
Für das Problem des Re-scorings von vorhergesagten Strukturmodellen beschreiben wir das **Vorescore** Verfahren ein, mit dem die Fold-Erkennung deutlich verbessert, aber auch die Anwendbarkeit von Potentialen im Allgemeinen getested werden kann.
Zur weiteren Verbesserung führen wir eine Kontakt-konsistente Vorolign Variante (**ccVorolign**) ein, die wegen der neuen Konsistenz-Randbedingung erheblich aufwndiger als das polynomielle

Vorolignverfahren ist, aber eben auch interpretierbare konsistente Alignments liefert.
Das neue Strukturalignment Verfahren (**PPM**) erlaubt es phänotypische Plastizität, explizit zu modellieren und zu berücksichtigen. PPM wird eingesetzt, um die Effekte von alternativem Splicing auf die Proteinstruktur zu untersuchen, insbesondere die Hypothese, ob Splice-Isoformen unterschiedliche Folds annehmen können (Fold-Transitionen).

Im **zweiten Teil** der Arbeit werden Verfahren zur Generierung von Kontextinformationen und zu ihrer Verwendung für die Interpretation von Hochdurchsatz-Daten vorgestellt.

Neue Textmining Verfahren extrahieren aus wissenschaftlichen Publikationen automatisch molekulare regulatorische Beziehungen und entsprechende Kontextinformation.
Neben schnellen NER (named entity recognition) Verfahren (wie **syngrep**) wird auch ein vollständig Ontologie-basiertes kontext-sensitives Verfahren (**SynTree**) eingeführt, das es erlaubt, mehrdeutige Synonyme kontext-spezifisch und damit wesentlich genauer aufzulösen. Diese für die Interpretation von Hochdurchsatzdaten wichtige Kontextinformation fehlt häufig in heutigen Datenbanken. Automatische Verfahren produzieren aber trotz aller Verbesserungen noch viele Fehler. Mithilfe unserer Applikation **RelAnn** können aus Texten extrahierte regulatorische Beziehungen ontologiebasiert manuell annotiert und kuriert werden.

Die Verwendung aktueller Hochdurchsatzdaten benötigt zusätzliche Ansätze für die Datenprozessierung, zum Beispiel für das Mapping von hunderten von Millionen kurzer DNA/RNA Fragmente (sog. reads) auf Genom oder Transkriptom. Diese Daten (RNA-seq) ergeben sich durch next generation sequencing Methoden, die derzeit mit immer leistungsfhigeren Gerten immer kostengnstiger gemessen werden knnen. In der **ContextMap** Methode werden im Gegensatz zu state-of-the-art Verfahren die auftretenden Mehrdeutigkeiten erst am Ende des Mappingprozesses aufgelöst, wenn die Gesamtheit der Mappinginformationen zur Verfügung steht. Dadurch könenn mehr Fehler beim Mapping zugelassen und trotzdem höhere Genauigkeit erreicht werden.

Die Konsistenz zwischen der Kontextinformation aus Textmining und Datenbanken sowie den gemessenen Daten kann dann für das Auffinden und Bewerten von konsistente Regulationen (**GGEA**) genutzt werden. Dieses Verfahren stellt eine wesentliche Erweiterung der häufig verwendeten Mengen-orientierten Verfahren wie overrepresentation (ORA) und gene set enrichment analysis (GSEA) dar.

Zuletzt stellen wir die Methode **RelExplain** vor, die aus dem extrahierten Kontextwissen netzwerkbasierte, testbare Hypothesen für die Erklärung von Hochdurchsatzdaten generiert.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"Is a^b transcendental, where a is algebraic and b is irrational? Yes."
Hilbert's 7-th problem from 1900, and the answer provided by Gelfond (1934) and Schneider (1935)

What is context based bioinformatics?

Bioinformatics uses informatics methods to solve problems inspired by biology or, in the best case, to solve biological problems. Many research areas of bioinformatics are dominated by biologically inspired problems, which are formally represented as search or optimization problems and then addressed by appropriate algorithmic solutions and complexity analyses. This often leads to efficient and elegant methods.
Unfortunately, these algorithms are also often not very relevant as solutions to real biological problems. The reason is that the formal problem specifications are simplifications, which ignore important context information. The context information is essential for the biological application and unavoidably complicates matters. On the other hand, taking complicating contexts into account induces even more challenging algorithmic problems and, if they are carefully and meaningfully addressed, can result in relevant results to biological questions.
This thesis investigates context based bioinformatics problems and methods for a range of biological problems. It proposes several innovative methods, algorithms and software programs, mainly heuristic and approximate, which demonstrate that context dependent problems can also result in elegant and highly efficient solutions to real biological applications.

In order to explain the principles indicated by the term "context based" we illustrate the difference between the context free and the context based problem solving approach.
Consider one of the most famous problems in bionformatics: the protein folding problem. Since Anfinsen 1961 ([9]) stated his 'thermodynamic hypothesis' and demonstrated it with his experiments on ribonuclease A in 1973 ([8]), we know that the 1D structure of a protein (the sequence of the amino acids) in many cases determines its 3D structure (under natural environmental or physiological conditions).

Formulating this observation as a mathematical and algorithmic problem, we obtain a question similar to one of the famous ten problems (quoted above) of Hilbert presented 1900 at the 2nd international congress on mathematics, the protein folding problem: "Is there a function f that returns for any given residue sequence of length n over the alphabet of the one-letter code of the amino acids (20 letters) n real-valued 3-tuples (x, y, z) representing the C-$alpha$ coordinates of the corresponding residue?".
The answer provided appears to be given by nature: yes, there is (at least as judged from the many 3D structures determined by e.g. x-ray crystallography). But in order to mathematically derive such an answer to the question (a theorem), we have to, for example, explicitly construct such a function f. From a computer science view this could mean to provide a computer program which effectively computes f, i.e. outputs given a protein sequence of length n the 3D coordinates of the corresponding atoms.
Let us first consider the "context free" approach to do so. In the 'mathematics way' we would take the problem, think about it and (maybe years or hundreds years later) one might come up with some clever way that shows the existence of such a function for all the 20^n sequences for all possible n. To do so one does not need any additional context information, just the problem definition and time. However, it is very unlikely that this approach will result in a practical solution to the original (protein folding) problem, but just a solution to the mathematical/combinatorial problem.
What about the 'physics way' to solve the problem? Using the laws of nature one can take the ordered set of the amino acids and describe all the possible forces for any given 3D conformation resulting in an overall energy for any given set of coordinates. Doing so the problem "simply" reduces to find an f (the conformation) leading to the minimal energy for the input sequence. Again, we do not need any additional information (in addition to the context information describing the atomic interaction forces). But, unfortunately, the proposed solution again is not very practical and relevant due to the so called Levinthal paradox: Levinthal estimates 3^{100} conformations for a sequence of length 100 [41] leading to very high, intractable computational costs. Even though such context free approaches (molecular dynamics simulations) are actually applied to the protein folding problem, but so far they were only successful for very few small problem instances and if used in combination with context based approaches.
It appears that without using additional information - the context of the problem - the problem is very hard in practice.
Therefore, the widely used approach is to analyze different aspects of the problem, the entities involved (in this case the proteins) in the problem and derive additional constraints etc. In our example of protein folding, the context of the problem has rapidly evolved over the last 50 years with increasing research interests on proteins and the respective techniques to analyze them: we now have large databases of experimentally derived protein structures, even larger databases of protein sequences and many publications leading to additional concepts and measurements. Thus, the context of the problem has extended a lot and cannot compactly described in a formal problem specification in the form of a Hilbert's problem. As an example the database with a large number of the protein structures gave rise to the concept of distinct folds and with it the concept of protein structure similarity as a possible information source to understand the folding process. Meanwhile, we also know that if the sequence similarity between two proteins is above

a certain threshold (say above 40% sequence identity, i.e. 40% of the n residues are identical in an appropriate alignment) the function f will provide very similar 3D coordinates for the two protein structures. Thus, the function f for such a protein sequence with unknown structure could actually be derived from an alignment with a protein sequence with experimentally determined coordinates (the homology-based or similarity-based approach). This in turn provides the context of evolutionary variances leading to the idea of using the large sequence databases as an information source. The collected information on proteins and their sequence and structure leads to context based variants of the original protein folding problem: for many applications it appears to be enough to - for the input sequence - find a similar protein with known structure in the database. This would solve the problem for any sequence with a similar enough experimentally determined protein structure. This modified (reduced) version of the problem is called the fold recognition problem. This is only one example, there are many more aspects of the protein folding problem, where appropriate contexts represent knowledge and concepts implying reduced versions of the original protein folding problem.
The common theme of all of these versions of the original problem and its sometimes solvable and computable solutions is that they are more practically relevant due to relaxing the problem for implied needs and based on additional prior knowledge. E.g. the requirement of assigning real-valued atom coordinates (see above) appears to be rather artificial as many atom coordinates cannot be determined with high precision and the atom coordinates are not fixed anyway. Rather, protein structures appear to be much more flexible under 'physiological' conditions and they are subject to change based on interactions with other molecules, ligands, or proteins. Thus, the very function of proteins implies such a flexibility of protein structures, and, therefore, the original formulation of the problems turns out to be inappropriate for many applications. In fact, it opens up the area of investigating protein sequence and structure similarities, the homology- and similarity-based approaches, which are build on huge amounts of empirical knowledge. Thus, many new interesting research problems can be defined. On the other hand, some of these problems can effectively be addressed and sometimes be solved. Moreover, the solutions appear to be much more relevant for subsequent biological research questions – and also for the very nature of protein structures and the protein folding problem.

Context based: how to solve it?

"Although this may seem a paradox, all exact science is dominated by the idea of approximation"
Bertrand Russel

In his book "How to Solve It: A New Aspect of Mathematical Method" [154] G. Polya describes a recipe for general problem solving: (i) understanding the problem, (ii) devising a plan, (iii) carrying out the plan, (iv) looking back.
In our setting of the context based problems and context based algorithms the first step would also include to find and define the corresponding contexts, the second - which is basically to build a model representing a problem - would include the representation of contextual knowledge in

the model. Due to the nature of such a model both (ii) and (iii) include some simplification (see the quote from Bertrand Russel [80]). As outlined above the simplification step is already one of the most problematic steps: if we oversimplify the problem to a well-defined mathematical or combinatorial optimization problem we may solve it exactly and provably optimal, but the solution might not be relevant to the original problem. In general, the only way out is to derive a model, which is as close to the original problem as possible and the solution (either optimal, or approximate, or heuristic) is relevant or useful for the biological problem. For many bioinformatics problems the derived models are very complex, and often the goal is to find a practical and adequate solution, not necessarily an optimal one. Thus, the common goals of the work described in this thesis is finding practical and adequate solutions: for most of the investigated problems we design new heuristic methods incorporating the corresponding context. We also focus on the essential step (iv) looking back - we describe appropriate (often non-trivial) benchmarking strategies and sets for evaluating whether the goals and relevant solutions have been achieved.
Besides using the contexts in the model we try to include them also in the optimization/solving step. More precisely, in most cases the method explicitly reflects the context based hypothesis to test. Often, the methods in some way automate the manual procedure for specific classes of context based problem instances. In our understanding this "no black box" behavior is very important for getting a deeper understanding biological problems and the corresponding solutions as well for deriving new contexts. As Polya observes in his book: "Trying to find the solution, we may repeatedly change or point of view, our way of looking at the problem. We have to shift our position again and again. Our conception of the problem is likely to be rather incomplete when we start the work; our outlook is different when we have made some progress; it is again different when we have almost obtained the solution.". In our view this iteration is a general pattern in biological/bioinformatics research, and every iteration creates a new context. Thus, appropriate bioinformatics methods and algorithm development should take this into account. More precisely, in this thesis we address different problems using the prior knowledge, find practical and adequate solutions and thereby - through the 'non black-box' behavior and explicit modeling of the problem - perform one or more iterations according to Polya's iterative problem solving paradigm.

Outline of the thesis

The thesis consists of two parts, both organized around a particular context. Below we describe these in more detail.

Part 1: The protein structure similarity context

As already partly discussed above the analysis of protein structures is a very active research field (e.g. searching for the term "protein structure" in PubMed results in about 500.000 hits). In this research many representations of proteins and many methods for comparing proteins have been investigated. The ultimate forces driving protein structure folding and, thus, underlying protein

structure similarities are still unknown. Many forces and features have already been used either to simulate protein folding or to quantify protein similarities. In addition, evolutionary constraints and evolution of protein function might provide further context information of structure comparison and protein folding.
Protein structures have been determined experimentally for several decades by now and many improvements in the respective techniques (X-ray, NMR) have lead to the collection of about 70.000 structurally resolved protein chains in a central, well-organized database, the Protein Data Base (PDB, [17, 18, 20]). Of course, the huge collection of protein structures is a useful resource to derive the **protein structure similarity context**. It appears to be natural to analyze the collection of known structures for establishing important contexts for structure comparison, structure prediction, and structure analysis in general. One obvious outcome are the plethora of so called homology-based protein structure prediction methods, which by exploiting this context in one or the other way are pretty successful in comparing and predicting protein structures. In fact, as has been demonstrated e.g. by the by now ten rounds of the CASP challenge over the last twenty years, exploiting this context appears to be the most effective if not the only effective way to computationally analyze protein structures.
Chapter 2 addresses this step: it analyzes the two gold standards for protein structure classification and similarities: the CATH and SCOP databases. Prior to the classification both databases divide the experimentally determined protein structures into re-occurring "folding units": the domains. The classification of these domains is then performed in both gold standards in a hierarchical manner, implying less structural similarity from leafs (the domains), via the inner nodes (structural subclasses) to the root of the hierarchy. In chapter 2 we analyze the consistency between the domain definitions as well as the structure of the respective hierarchies. Interestingly, we find that while both data sources have the same goal and similar principles there are severe differences. In the chapter the commonalities but also the differences of the two classifications are analyzed and highlighted. The implied definitions of large and comprehensive sets of structurally similar and non-similar domain pairs are based on a consistent classification creates the protein classification context for the analysis in the following chapters.
In the Vorolign (3) chapter we address the protein structure similarity problem in the context of the amino acid contacts. We use a amino acid based Voronoi tessellation of the 3D structure and analyze the similarities between the derived voronoi cells. Interestingly, we find that the 1D and 2D (secondary structure) sequence neighborhood of the cells is a very good descriptor: it has a state-of-the art separation power between similar and non-similar proteins. This finding leads to a new concept: the contact environment potential. There is a long history of using contact potentials or potentials of mean force for remote homology detection [121, 122, 125, 131, 138, 170, 175, 183] but typically these potentials score individual contacting pairs rather than the whole contact-neighborhood.
In the Vorescore-chapter 4 we analyze this novel concept of the contact environment potential in the context of the protein structure prediction (PSP). The main problem of the contact-neighborhood scoring is that it is unclear how to use and how to optimize it in the PSP setup. This chapter evaluates the current and the possible power of the concept with a heuristic approach: instead of directly optimizing the potential, we rescore predicted models using the environment potential thereby also employing knowledge about the different levels of similarity given by the

protein structure similarity context. The results show that the potential is able to improve the fold recognition quality over current approaches, but, theoretically, the performance could be much better if high-quality predicted models are available and used. This, and the high computational cost to generate the predicted models suggest that it is worthwhile to address the complex task of direct optimization of the contact environment potential.
The first step towards such a direct optimization is performed in the ccVorolign (*contact consistent Vorolign* chapter 5. Here, we review the Vorolign method [22] in the protein structure prediction context, and change the similarity optimization problem accordingly. In contrast to the original Vorolign similarity definition, which calculates the similarities between contact environments independent of the final correspondence between similar environments, ccVorolign optimizes the similarity consistent with the final alignment. This small change, however, leads to a much more involved computational problem, which is quite close to the protein threading problem. The chapter introduces a practical, heuristic approach to solve the problem, and the results indicate that a new concept for using potential is needed: the ϵ-inconsistency. The ccVorolign analysis suggests that the pairwise and neighborhood contacts are often not exactly conserved between corresponding amino acids in remotely homologous proteins, but more between corresponding regions. Taken together, the evidences from Vorolign, Vorescore and ccVorolign suggest that a block-based environment potential optimization could significantly improve the fold recognition (left to future work), thereby generalizing amino-acid contacts to the context of contacting regions.
In chapter 6, Protein Plasticity Measure, we again address the protein structure similarity problem, in this case based on the context of protein plasticity and develop an efficient structural alignment method (PPM). This concept accounts for the fact that proteins are not rigid objects and, thus, any particular set of 3D atom coordinates is only a simplified representation of the actual protein. Of course, experimental structures in the PDB as measured by X-ray crystallography are often represented as definitive fixed coordinates, but NMR structures also in the PDB often contain several alternative sets of coordinates indicating the flexibility or plasticity of the protein. Here, with plasticity of a protein structure, we mean the possible conformational space occupied by a protein in a particular physiological context, which we assume to be typically larger than a fixed rigid structure. Using the context information that proteins structures are itself dynamic entities - often in order to fulfill their functions - we model the similarity between two protein structures via the extent of plasticity needed to transform one protein structure into the other. This gets rid of the very restricted but widely used "sequence of 3D points" view of protein structures and protein structure similarity. The new view also tries to somewhat automate the visual assessments a human performs if analyzing the similarity. As our results on protein structure comparison show, this leads to a better capturing of remote homologies and to much better interpretable residue-based structural mappings. In addition, as the method mimics manual approaches to analyze similarities, it can be also used to obtain more insights into block-based environmental potential analysis prior to addressing the complex threading problem.
Chapter 7 on 'Splicing effects in Protein Structures' is another example of how PPM can be used for analyzes in different contexts. In this chapter, we analyze how alternative splicing affects protein structures. While alternative splicing leads to multiple transcripts and, thus, multiple proteins encoded by the same gene, not much is known about the structural similarity / differ-

ence of these so called “protein isoforms”. In this chapter we provide a comprehensive analysis of known isoforms with clear structural templates and categorize the implied structural changes using both intrinsic and context based measures. Surprisingly, we find many cases where alternative splicing affects core parts of the structures leading to structural so called “non-trivial” isoforms, where the resulting protein appears to have a definitive fold and a defined biological function. Using PPM we could identify possible alternative folds for splice isoforms different from the native fold of the unspliced protein. This suggests a novel hypothesis, namely that alternative splicing might change the protein fold.

Part 2: The high-throughput experimental data context

In the second part of the thesis we deal with the analysis of high-throughput experimental data and the respective contexts. During the last decades the progress in biotechnology produced the ability to perform a large number of experiments simultaneously; for example using microarray or sequencing (RNA-seq) technology one can take a snapshot of the level of transcription of all (or in case of microarrays of all known) mRNAs. Using this in a differential setup (measure the levels in two samples with distinct phenotypes) or in a time course (measuring the levels in the same samples over time) can provide valuable insights about the main factors (genes or transcripts) leading to the observed (phenotypic) differences, and in principle can help to identify possible drug targets. Due to the large number of measured entities and due to the complex nature of the results the main task of bioinformatics for such high throughput (HT) experiments is to provide help for the reasonable interpretation of the measurements (often by constructing appropriate models).

In this thesis we address such interpretation tasks for the differential transcription based experiments, i.e. where the mRNA levels are measured between a control sample and a sample of interest (condition). We term the result of such an experiment (the list of the genes where the mRNA level has significantly changed along with the direction and strength of the change) as an “experimental outcome”. This type of output is commonly used for microarray and RNA-seq experiments, but also for quantitative mass spectrometry data such as SILAC measurements.

The main problem for the interpretation of experimental outcomes is that the outcome is a result of a complex cell regulatory response on the experimental condition possibly including regulatory events on different layers such as for example response to metabolic concentration differences, external/internal signaling pathways, regulation of the transcription, post transcriptional/translational regulation, or in some cases the differences are driven even by genetic rearrangements (BCR-ABL [112]). In addition, most of the current techniques measure the changes between cell cultures leading to measurements averaging in between the cells, and in many cases the conditions reflect changes accumulated over time. From all these influences the experiment measures only one layer of the regulation difference: the final results of the transcriptional regulation.

Due to this relatively sparse information the power of interpretations is very limited without using context information: basically one can only directly operate over the experimental outcome and rank the genes according to the significance of their changes between samples. Using a little more context - the knowledge about sets of genes grouped by their function, or the biological

process they are putatively involved in - one can analyze the probability that these sets of genes are affected by the differential regulation. The latter - gene set over-representation/enrichment analysis became the standard evaluation strategy for high throughput experiments. However, this interpretation of the data does not directly yield an explanation how and why these changes occur and leaves the initial question ("what are the driving forces to get to this experimental outcome') unanswered. Although it is probably impossible to answer this question for every data set today, we are interested in methods providing at least some guidance for the experimentalist. Our approach is to incorporate as much prior (context) knowledge as possible to help in the interpretation and provide tools to infer/reject/restrict these - for example driven by expert knowledge or additional experimental evidence.

The main information needed in the interpretation step is the knowledge about the regulations on different layers taking place in the cell - a gene regulatory network (GRN). This immediately raises the question, how to obtain the required regulatory information. There are many large scale experiments which allow to derive possible interactions between proteins, e.g. yeast two hybrid (Y2H) approaches generating protein protein interaction networks or ChIP-chip and ChIP-Seq experiments yielding TF-target networks. However, PPI experiments are prone to a large number of false positives with respect to actual regulation or even just protein binding in vivo. ChIP-chip/-Seq experiments, in principle, provide information about possible transcriptional regulations (via providing transcription factor (TF) binding sites in promoter regions of genes (target)) but similar to PPIs it is unclear whether the binding found in vitro take place in vivo, and if so whether it has a real regulatory effect. In addition, due to the experimental technique, such TF-target relationships must be extracted for each TF independently and, thus, the number of such experimental results differs largely between organisms. For example while about 70% of all putative TFs in yeast were processed with ChIP-chip or ChIP-seq techniques, for human this fraction is only about 6-10%. On the other hand, numerous regulations have been examined in small-scale experiments often along with a relatively detailed characterization of the context of the regulation. Only a small part of this knowledge is organized in pathway databases such as KEGG [144] and BioCarta [1], whereas the gold standard gene ontology (GO [12]) provides information about sets of genes involved in processes but without providing detailed regulatory information. A much larger part of the current knowledge - in fact the complete knowledge - is captured in a completely different database, the database containing the biomedical scientific abstracts/publications: PubMed [196], unfortunately in free-text format only.

Chapter 8 deals with the problem of how to derive the context information needed for the interpretation of high throughput data automatically from the millions of free text articles. Obviously this cannot be done manually for the complete set of publications, so we developed und utilized text-mining techniques for the task. We will introduce a very fast dictionary based method *syngrep* used for the identification of entities (named entity recognition, NER) - in our case mostly protein and gene names - in texts. Such a program can be used to annotate and pre-filtering the text data for various applications. We have also improved and used RelEx - a method developed by Fundel et al. in 2007 [61] - to extract detailed regulatory relations between entities identified by syngrep and, thereby, derive a textmining based GRN. Finally, we introduce a web-based annotator tool enabling fast (semi-) manual annotation of complex regulations using syngrep and RelEx as prefilter.

There is much more regulatory knowledge contained in scientific publications as currently can be extracted by RelEx. Given methods for precise recognition of occurrences of named entities one could extract associations between proteins/genes and tissues, experimental conditions or complex phenotypes such as diseases. The named entity recognition task for such entities, however is much more complex than for proteins as terms describing for example diseases are usually more ambiguous and overlap more with common english words.
In Chapter 9 we present a more context based named entity recognition method and its application to such a complex task: we evaluate the recognition and identification of psychiatric diseases. In chapter 10 we switch to the experimental data context: here we address one of the first steps of interpretation of the RNA-seq experiments: the mappings of the short reads sequences. One of the differences between microarray and RNA-seq experiments is that while in microarray experiments the genomic source of the probes is clear, and they are designed such that they are unique, in RNA-seq experiments the source of the reads sequences has to be identified afterward. The fact that the reads may in principle come from any genomic locus offers several advantages: with RNA-seq, e.g., one can detect expressions of transcripts with lower abundance, one can find unknown or rare genetic variations, one obtains also more information for alternative splicing analysis. The first step for all these applications is to find the correct positions of genomic origin for all the reads. With the currently used read lengths of 50-100 bp and sequencing errors of about 1-2% per base this is not an altogether trivial task: many reads can be mapped to multiple genomic loci. The main idea in this chapter is to use the context knowledge of how the reads are generated: assuming an RNA-seq experiment the reads correspond to transcripts and at the same time, transcripts will produce multiple reads. This is, sets of reads correspond to transcripts and so to given genomic stretches and corresponding splice sites. The algorithm we present in this chapter exploits this idea - it uses the set of all sequenced reads to disambiguate between genomic locations where a given read can possibly come from.
The method presented in chapter GGEA (chapter 11) - mainly the work of Ludwig Geistlinger - is one step towards a more context aware gene set enrichment method. The main idea here is to incorporate direct regulatory information for the differently expressed genes (the experimental outcome). The assumption behind GGEA is that the experiment captures both the activity of the regulators and the regulated genes, and, if so, we can check for every regulatory edge whether it is consistent with the measured data. This admittedly strong assumption about capturing the activity of regulators does not necessarily hold for microarray or RNA-seq experiments - but the principle could also be used if combining different type of experiments such as differential transcription factor binding to estimate the activity of the regulators and their effects.
GGEA is not yet fully network based - it evaluates the consistency of the regulatory edges independently - but the idea of consistency evaluation is very close to the wanted mechanistic regulatory explanation. Therefore, we introduced a novel, fully network based type of gene set enrichment: the explainability of a predicted enriched gene set, an idea analyzed in more detail in chapter 12. In the final chapter “RelExplain” (chapter 12) of the thesis, we use the context from public databases as well as from the textmining techniques as described in chapter 8 and again address the goal to provide as much help in the interpretation of experimental outcome as possible. The underlying assumption here is that for a given set of significantly regulated genes and a biological process there is an implied pathway or sub-network connecting and thus explaining

the observed experimental outcomes. The RelExplain method presented in the chapter optimizes heuristically such an explanation for any relevant gene set, and a web based fronted provides means such that a user can prove/falsify/recalculate automatically derived explanations. Using the software iteratively the user can get closer to a better understanding of the experimental data under investigation and, thereby, easily can identify relevant prior knowledge (encoded in the publications leading to the explaining regulatory relations) for his/her research.

Almost all chapters presented in this thesis are based on manuscripts published, submitted or prepared for submission, thus, contains contributions, work and ideas from multiple people being co-authors of the papers and manuscripts. In most cases I am the main (or equal-main) contributor, for the other three cases Vorolign, (chapter 3), Analysis of effects on alternative splicing on protein structures (chapter 7), GGEA (chapter 11), I was more involved in methodical or evaluation details, and the main contributions are listed and appraised in the respective introduction part of the corresponding chapter (Fabian Birzele for chapter 3 and 7, Ludwig Geistlinger for chapter 11). The contents of the chapters are mostly identical to the corresponding manuscript, only minor (mostly formatting) changes are applied.

Part I

The protein structure similarity context

The protein structure prediction and folding problem is one of the long-standing problems in bioinformatics and computational biology. The first part of my thesis centers around context dependent problems and the respective methods (algorithms) for the classification, analysis, comparison, alignment and prediction of protein structures. For all of these research questions I present approaches which exploit context information and develop efficient methods, which are practically applicable to real-world problem instances.
This includes a new consistent protein structure classification and associated benchmark sets for protein structure comparison and prediction, a fast protein structure comparison method, a method and scoring function to re-rank predicted models, the analysis of this scoring function for threading (protein structure prediction, fold recognition) and a fast and flexible protein structure alignment algorithm. The developed methods have been applied to analyze structures induced by splicing events, which might contribute to the structural diversity in a cell.

Chapter 2

Systematic comparison of SCOP and CATH

Many protein structures have been experimentally determined. Some proteins have been solved several times at different resolution, in different contexts, and with and without ligands. Other proteins have been solved with more than one experimental technique. In order to get an overview of the more than 85.000 (75.000 X-ray and almost 10.000 NMR) structures currently (12/2012) deposited in the central structure repository (the PDB database, http://www.pdb.org) the structures have been curated, sorted, grouped and classified in various ways. Two of the best known protein structure classifications are SCOP and CATH, which classify the structures (respectively the over 170.000 structural domains) of the PDB in respective hierarchies, e.g. into 2626 superfamilies (CATH 12/2012). For SCOP (2/2012) the number are: 135.000 domains are classified into 4272 families, 1961 superfamilies and 1194 folds. This already shows two things: first a drastic reduction is achieved via the classification by SCOP an CATH but, second, the classification differs somewhat.

In order to get deeper understanding about the similarities between protein structures and to proceed in deriving rules about protein folding we need some standard-of-truth about what is similar and what is not. Fortunately, several research groups addressed this need in the last decades. In this chapter we analyze the similarities and the differences between the aforementioned most prominent protein structure classifications CATH and SCOP.

The main goal of this chapter is to derive a consistent classification from these two standards: thus, we try to map the levels of the two hierarchies in the best possible way and then map the respective classes such that the consistent overlap is as large as possible. I.e., for all corresponding similarity levels we aim at finding sets of domains, which are consistently defined as similar (or not similar) at the given level in both hierarchies. Surprisingly, although we work on the consistently defined domains (actually, in many cases, the partitioning of protein chains into structural domains is also inconsistent) and the two classifications share very similar classification principles, we observe large differences among the sets corresponding to the inner nodes of the two hierarchies. As a consequence, we defined and derived the comprehensive set of (in CATH an SCOP) consistently defined domain pairs enabling in-depth benchmarking for different applications.

The analysis described in this chapter is joint work with Fabian Birzele and the findings here were published in BMC Structural Biology in 2009 [39]. The current chapter reprints this paper in a re-formatted and type-setted way according to the layout of my thesis.
My contribution in this work was the development and implementation of the classification hierarchy comparison method, the evaluation of the effect of the consistent benchmark-set on the structural similarity measure TM-score.

2.1 Introduction

The classification and comparison of the more than 50000 protein structures deposited in the PDB [19] (January 2009) is an essential step to extract valuable knowledge from protein structure data. Today, the two most prominent protein structure classification schemes are SCOP [7] and CATH [77]. Both partition proteins into domains and classify them in a hierarchical manner. SCOP sorts protein domains into classes, folds, superfamilies and families while the four major levels of CATH are class, architecture, topology and homologous superfamily.
The SCOP database is mainly based on expert knowledge and on the first level of the hierarchy defines four major classes namely 'all α', 'all β', 'α/β' as well as '$\alpha + \beta$' describing the content of secondary structure elements in the domain. According to the SCOP authors, domains in a common fold have the same major secondary structures in the same arrangement with the same topological connections. In the same superfamily, domains share low sequence identities but their structures and, in many cases, functional features suggest that a common evolutionary origin is probable while domains clustered in the same family are likely to have a common evolutionary origin based on sequence similarity or functional evidence.
The building process of CATH contains more automatic steps and less human intervention compared to SCOP. Analogous to SCOP, CATH starts at the class level defining three major classes of secondary structure content ('all α', 'all β' and 'α/β'). The second layer, called 'Architecture', clusters domains with common general features with respect to the overall protein-fold shape but does not take connectivity into account. The 'Topology' level is analogous to the SCOP fold level and groups structures that have a similar number and arrangement of secondary structure elements with the same connectivity. The last (major) level, 'Homologous superfamily', clusters domains with a high structural similarity and similar functions, which suggest that they may have evolved from a common ancestor.
In the last years, SCOP and CATH have been used to address various open questions in structural biology and are further employed as training and gold-standard databases in various applications making them invaluable resources in structural bioinformatics. They have been used to study the interplay of protein structure and protein sequence evolution ([158], [180]), and - as we will show in chapter 7 - it can also be used to explore the connection between alternative splicing and protein structure evolution.
Besides those analyses, they are often used in the context of automatic protein structure classification and protein structure prediction when training and / or evaluating the respective methods.
Automatic protein structure classification (given the resolved structure) has become an important topic with the faster growing number of PDB structures in order to analyze structural and func-

tional features of proteins. Methods which are specifically suited for an accurate and automatic assignment of structures to their respective class often use SCOP or CATH as reference and template datasets or to evaluate their performance. Among those methods which heavily use SCOP or CATH for structure prediction from the sequence are AutoSCOP [70] and PFRES [32]. Examples for methods to compute protein structure alignments and to predict similarities between structures will be shown Vorolign (chapter 3), PPM (chapter 6), FatCat [200] or TM-Align [206]. In the context of machine learning, various methods to discriminate between structural classes defined by SCOP or CATH, e.g. using Support Vector Machines [115, 129], have been published. Also, various methods which aim at the prediction of a proteins structure from the sequence [185] have been developed in the last years, especially in the context of the CASP experiments reviewed e.g. in [134] (and references within) where reference databases such as SCOP and CATH are used in the prediction and the assessment phase. Of course, difference in the reference sets will inevitably lead to differences in the assessment reflecting the performance with respect to the criteria used to construct the reference sets.

Although the two hierarchies have become the gold standard in the field, their goals and the methods used to classify structures are not the same which leads to different classifications of the same protein. Differences are found with respect to the domain partitioning of the protein chain, as well as in the classification of a domain into its corresponding structure class. Some differences and similarities between SCOP and CATH have already been evaluated ([43],[81]) and those analyses allowed for valuable insights into the problems and challenges of classifying protein structures. Nevertheless, since the most recent study [43] the number of protein structures available in the PDB has more than doubled. This fact may have also increased the problem classifying all known structures in a consistent manner. Also, in contrast to previous studies, we will focus on the extraction of consensus classifications based on the detailed comparison of the two hierarchies which should be a useful resource for (machine learning) methods for protein structure classification and prediction.

In more detail, we propose a new approach to compare SCOP and CATH on the different levels of the two hierarchies using a similarity measure for sets of domains. Based on an initial mapping of individual domains defined in both hierarchies and on the similarity of two sets of domains, we identify for each set from one hierarchy the corresponding overlapping set(s) from the other hierarchy. This allows to map sets of domains on different levels of SCOP and CATH and to analyze the differences and similarities of the two hierarchies in detail.

SCOP and CATH are often used as 'standard of truth' datasets and inconsistencies and differences in the hierarchies unavoidably lead to problems in the training phase (since wrong or misleading concepts are learned) as well as in the testing / benchmarking phase. Proteins classified to be different by one hierarchy may indeed be similar according to the other classification leading to an overestimation of the errors made. To overcome those problems, we extract sets of pairs of protein domains from our SCOP-CATH mapping which are consistently classified in both hierarchies. Those pairs represent a novel and comprehensive benchmark (training) set which allows for a more consistent evaluation (and training) of protein structure comparison and protein structure prediction methods.

Finally, we utilize our mapping as orthogonal evidence in order to identify potential connections between different folds in one hierarchy which may be revealed via a connection of the two folds

suggested by the respective other hierarchy. Such connections between different folds (which are supposed not to be evolutionary related i.e. due to the SCOP definition) provide interesting starting points to further analyze the protein sequence-structure space.

2.2 Datasets

For our analysis we use the most the version 1.73 of SCOP (September 2007) and the CATH version 3.1.0 (January 2007) which contains a similar number of proteins. The mapping containing the more recent CATH version 3.2.0 can be found on the supplementary website at http://www.bio.ifi.lmu.de/SCOPCath. The website and the benchmark datasets will be updated regularly when new versions of SCOP and CATH are released. SCOP 1.73 contains 34495 proteins deposited in the PDB (97178 domains) which are classified into 11 classes, 1283 folds, 2034 superfamilies and 3751 families. CATH comprises 30028 PDB proteins which are partitioned into 93885 domains and sorted into 4 classes, 40 architectures, 1084 topologies and 2091 homologous superfamilies. The union set of the proteins in the two classification schemes contains 36970 proteins. 27553 PDB proteins are classified in both hierarchies. Please note that throughout this article we regard the following levels of SCOP and CATH to be 'equivalent': SCOP family / superfamily $\rightarrow$ CATH homologous superfamily, SCOP fold $\rightarrow$ CATH topology, SCOP class $\rightarrow$ CATH class.

2.3 Detailed comparison of SCOP and CATH

In the following we present the results of our analysis of similarities and differences between SCOP and CATH in more detail. We will first discuss the results of mapping the different domain definitions of SCOP and CATH onto each other, showing that there are (surprisingly) large differences between SCOP and CATH with respect to their domain definitions. We will then use the set of mappable domains (for which domain definitions largely agree), restrict the respective hierarchies to those domains and compute the mapping of inner nodes of the two restricted hierarchies. We then analyze this mapping of inner nodes in detail which turns out to be very complex indicating many inconsistencies between SCOP and CATH. The usefulness of the SCOP-CATH mapping is shown then by two applications.
Our analysis depends on whether we map SCOP to CATH or vice versa. We present the results of the mapping of SCOP $\rightarrow$ CATH in the following. The results for the mapping of CATH $\rightarrow$ SCOP are available in the supplementary material on http://www.bio.ifi.lmu.de/SCOPCath.

2.3.1 Domain mapping

In order to analyze the different domain definitions in SCOP and CATH, we keep a domain defined in one hierarchy fixed and count how often one or more domains from the respective other classification are mapped onto it. A domain is mapped *iff* the overlap o, as defined in the Methods section, is greater than 0.0, i.e. we map all domains which have at least one residue in

common with the query domain. The results are shown in Table 2.1 and confirm results from previous studies [81] that SCOP tends to define larger domains which may be represented by several, smaller domains in CATH.

For our final mapping of domains we use a much more restrictive overlap threshold of $T_o = 0.8$. This implies a unique and bijective mapping of domains onto each other but leaves many domains unmapped. Including protein domains which overlap to only a small extent would lead to additional problems when comparing the two hierarchies, especially since domains are also classified according to their secondary structure elements and content (see [111] for further discussion). Therefore, including secondary structure elements in the domain definition of one hierarchy while not including them in the other one is likely to lead to differing classifications. The rigid threshold of 0.8 assures that only domains which are defined as the same parts of the protein structures in SCOP and in CATH are contained in the final dataset.

As shown in the following, differing domain assignments have a large impact on the resulting classification. Out of the 27553 proteins which are classified in both hierarchies, for only 19266 (about 70%) the domain definitions are similar enough leading to 56104 domains in the final dataset (increasing up to 66128 mappable domains with an overlap threshold $T_o > 0.5$). In SCOP, those domains are classified into 11 classes, 754 folds, 1258 superfamilies and 2228 families which means that on the other hand, for 538 folds, 776 superfamilies and 1523 families already the domain definitions of SCOP and CATH differ to such a large extent that they can not be meaningfully mapped onto each other. According to CATH, the proteins belong to 4 classes, 38 architectures, 736 topologies and 1462 homologous superfamilies. Two architectures, 348 topologies and 629 superfamilies of CATH remain unmapped.

Those values show a surprisingly large number of domains in either of the two hierarchies which are defined in a very different manner in the respective other classification scheme according to their domain boundaries and result in the fact that for only 70% of the proteins the classifications can be compared. Moreover also only 70% of all SCOP families and CATH superfamilies are retained in the mapping due to differing domain assignments.

	1	2	3	4	5	6
SCOP	49251	17162	1885	435	130	29
CATH	68270	11018	492	3	0	0

Table 2.1: Mapping of the domain definitions of the two hierarchies. An overlap threshold > 0.0 is used, i.e. all domains which share at least one residue are mapped onto each other. The SCOP row shows the number of CATH domains mapped onto a single SCOP domain, while the CATH row describes the number of SCOP domains mapped onto one domain defined in CATH. A single domain in SCOP may be partitioned into up to 6 domains in CATH. Overall, about 20000 (out of 70000) SCOP domains map more than one CATH domain while about 11500 out of 80000 CATH single domains map to more than one SCOP domain.

2.3.2 Mapping of inner nodes

Given the set of mappable domains as discussed above, we computed the mapping of inner nodes of the two hierarchies as described in the Methods section. The results are shown in Table 2.2. Using the F-measure (as defined in the Methods section) we are able to identify for every inner node of SCOP the corresponding, i.e. best fitting, node in the CATH. In such a mapping one would e.g. expect that SCOP superfamilies (and families) map best to the CATH 'homologous superfamily' level.
Surprisingly, when using a F-measure threshold of 0.0 (we map every query SCOP node onto the CATH node with maximal F-measure), the mapping of inner nodes and, therefore, the partitioning of the fold space according to SCOP and CATH appears to be more complicated than expected and many inconsistencies can actually be observed.
When we require a certain quality for a mapping, i.e. setting the F-measure threshold to 0.8, a large number of inner nodes do not find a partner in the other hierarchy. SCOP and CATH therefore define their sets of domains on every level of the hierarchies and for many cases very differently and a large number of unexpected mappings (all the cases except for the green cells in Table 2.2) can be observed. For example 240 (178) homologous superfamilies in CATH can not be mapped to a corresponding SCOP superfamily or family for a F-measure threshold of 0 or 0.8, respectively.

F>0	Unmapped	C	A	T	H
fold class	0	4	2	1	4
fold	0	0	5	504	236
superfamily	0	0	2	32	1224
family	0	0	1	9	2218
F>0.8	Unmapped	C	A	T	H
fold class	8	2	0	0	1
fold	125	0	4	439	177
superfamily	236	0	1	24	997
family	1055	0	1	6	1166

Table 2.2: Number of inner nodes from a hierarchy level in SCOP mapping best to a node from some level in CATH. Two different F-measure thresholds of 0.0 and 0.8 are shown. For example, 504 SCOP folds map best to a CATH topology node given a threshold of 0.0 dropping down to 439 nodes for a F-measure threshold of 0.8.

2.3.3 Comparison of domain pairs

In order to analyze the surprisingly large number of inconsistencies between SCOP and CATH in more detail, we tested all pairs of domains in the set of mappable domains for their consistency in the respective other hierarchy. For example, we test if a pair from the same SCOP superfamily is also classified to be in the same 'homologous superfamily' level in CATH. The results of this

pairwise comparison of the two hierarchies are shown in Tables 2.3 and 2.4. This analysis reveals a very large number of domain pairs which are not classified consistently in the two hierarchies. Even though on the family level, where the evolutionary relationship of the proteins should be clear, 98% of the pairs are consistently defined on the one hand, more than 130000 pairs classified into 70 different folds and 102 superfamilies are not classified in a consistent manner. More than 700000 pairwise errors are observed on the superfamily and more than two million errors on the fold level. Table 2.4 allows for a more detailed analysis of the mapping between the different levels of SCOP and CATH and the errors that occur. For example 0.866% (corresponding to 70.188 pairs) of the domain pairs from the same SCOP family are classified to be in different topologies (of the same CATH class) in CATH.

Fortunately, many errors are contributed by a relatively small number of 'super folds' (Rossmann Folds, Immunoglobulin and some others). Those fold classes also build clusters of similar folds which are further discussed in the context of interfold similarities below.

Nevertheless, a large number of inconsistencies can not be explained by these well known superfolds. All inconsistent pairs can be interactively explored on http://www.bio.ifi.lmu.de/SCOPCath. An interesting example are d1bbxd_ and d1rhpa_ which are classified to belong to two different classes in SCOP (b.34.13.1 and d.9.1.1, respectively) and are indeed very different on the structure level, but belong to the same homology level according to CATH (2.40.50.40). A second example is the pair d1ku7a_ and d1j9ia_ (classified as a.6.1.5 and a.4.13.2). The two domains are indeed structurally similar (though they have a different number of helices). They are classified as different folds SCOP but belong to the same homology level in CATH (1.10.10.10).

All inconsistencies will lead to problems when benchmarking automatic structure classification methods. Also, they may lead to learning wrong concepts in the training phase of machine learning methods for protein structure classification as decision criteria are only learned with respect to one classification or correct criteria are ignored in the learning phase because of inconsistencies.

	consistent	inconsistent	folds	superfamilies
family	7.970.415	133.335	70	102
superfamily	8.208.965	713.181	121	159
fold	10.879.564	2.389.191	84	500
class	268.747.988	62.849.692	745	1258
other class	962.011.672	249.897.353	745	1258

Table 2.3: Shows the inconsistencies between SCOP and CATH with respect to the levels of the SCOP hierarchy. The second column displays the number of consistent pairs (pairs of proteins from folds, superfamilies and families in the green cells in Table 2.4) and the third column the number of inconsistent pairs. Columns four and five display the number of distinct folds and superfamilies which account for the inconsistencies observed.

	outer	class	fold	superfamily	family
outer	79.38%	8.31%	0.99%	0.40%	0.03%
class	18.16%	56.15%	2.55%	1.88%	0.87%
arch	2.42%	24.90%	2.80%	1.27%	0.09%
top	0.04%	10.50%	81.99%	4.44%	0.66%
hom	0.002%	0.14%	11.66%	92.01%	98.34%

Table 2.4: Displays the detailed mappings of domain pairs in percent from SCOP (columns) onto CATH (rows). Columns sum up to 100% and green table cells display consistent mappings. Please note that due to the very large number of pairs, even small percentage values correspond to many examples (see Table 2.3 for details).

2.3.4 Extraction of a novel benchmark set

The pairwise comparison also allows us to extract sets of domain pairs which are consistently defined across the hierarchies and which may be used as novel benchmark sets to train and evaluate structure comparison methods. In particular, we extracted two sets of domain pairs:

- domains which are consistently defined as being similar in both hierarchies (in the following denoted as the SCOP-CATH set) corresponding to the consistent fold, superfamily and family pairs in Table 2.3.

- One set of non-similar, negative domain pairs, i.e. domains in the same class, which are consistently classified into different folds.

Also, to avoid an overrepresentation of very similar domains in the dataset, we clustered the domains according to their sequence similarity. All domains with a pairwise sequence identity of more than 50% were clustered together. For each cluster we retained only one representative domain in the final benchmark set (SCOP-CATH50 set) which can be obtained at http://www.bio.ifi.lmu.de/SCOPCath. We also provide additional data, i.e. the details of the clustering process, which allows users to define their own benchmark sets using different sequence identity cutoffs in case that other sequence identity thresholds are appropriate for the specific application.
Redfern et al. also used a consistent set between SCOP and CATH in benchmarking their CATHEDRAL method [13]. Our approach is designed to contain all pairs of proteins which are consistently defined between the two databases. This is an important feature for benchmarking structure classification methods in very detail on a large set of different fold topologies. In contrast, the Redfern dataset, designed for a different purpose, focuses on consistently defined superfamilies whose members overlap to at least 80%. Extracting protein pairs from these consistent superfamilies would lead to a large number of pairs in the benchmark set (up to 20% of the proteins in a superfamily) which would be actually classified inconsistently between SCOP and CATH.

Our dataset can directly be employed for training and benchmarking novel methods developed in the field on different levels of the hierarchies and therefore different levels of structural similarity. In the following we show that this novel benchmark set allows for a much more consistent evaluation of structure comparison methods which is not biased by inconsistencies in the different gold standards.

2.4 Applications of the SCOP-CATH mapping

In the following, we discuss the results of two applications of our detailed SCOP-CATH comparison.

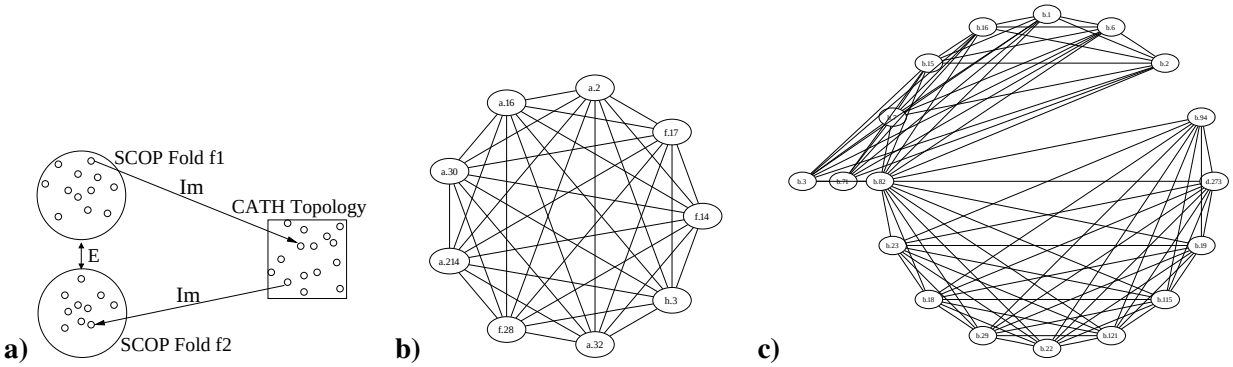


Figure 2.1: **a)** Shows the method of connecting different folds in i.e. SCOP via a link proposed by the mapping of SCOP and CATH. **c)**: Shows the interfold similarity of α-hairpin proteins in SCOP which are clustered in the same fold according to CATH (1.10.287). **c)**: Shows an more complicated fold graph clustering proteins of jelly-roll (2.60.40, Immunoglobulin-Like) and immunoglobulin topologies (2.60.120, Jelly-Roll) in a non-clique subgraph. All fold graphs may be interactively explored on http://www.bio.ifi.lmu.de/SCOPCath

2.4.1 Benchmarking structure-comparison methods

For benchmarking purposes, and as an examplary structure comparison method, we used the TM-align method which computes a structural alignment optimizing the TM-Score [207]. TM-Score measures the similarity of two structures by an optimized rigid body superposition and a TM-score of above 0.4 has been described to indicate structural similarity ([204], [205]). TM-align has been chosen for this study since the TM-Score has already been used to discriminate between similar and non-similar proteins ([204], [205]) and should therefore allow for a good discrimination of similar and non-similar protein domains. Furthermore, the method is quite fast allowing for the computation of the more than 5.000.000 structural alignments in reasonable time.

For our analysis, we compare the performance of TM-align on the complete benchmark set with the performance on the novel benchmark set proposed in this paper. The only difference between

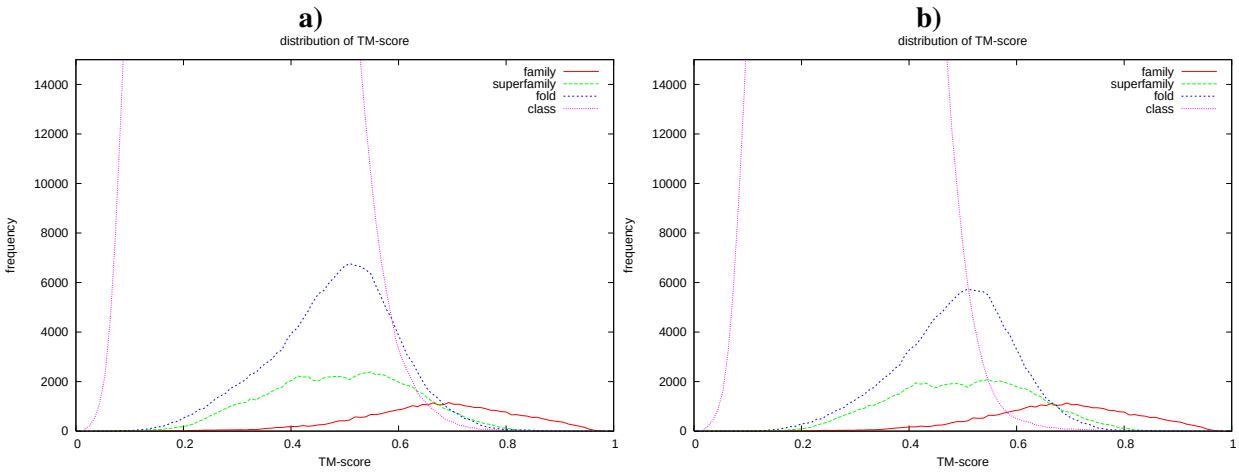


Figure 2.2: In detail evaluation of the performance of TM-align on the complete set of similarity relationships defined by SCOP (left column) and the performance on the novel benchmark set proposed in this study (right column). The plots are further discussed in section 2.4.1

the two sets are the pairs being evaluated. While all pairs which are similar according to SCOP are evaluated in the original setting, our novel benchmark set contains only those pairs which are consistently defined to be similar or different in both SCOP and CATH. Therefore, while the domains contained in the sets are the same, the number of pairs being compared is much smaller in our novel benchmark set than in the original set (16% of the positive pairs have been removed).

In the following we will discuss the plots shown in Figure 2.2, 2.3 and 2.4 which evaluate the performance of TM-align on the two benchmark sets in detail.

Plots **(a)** and **(b)** in 2.2 show the distribution of TM-Scores of domain pairs within the same class/fold /superfamily/family. The distributions of the scores are very similar between both sets indicating that the main properties of the benchmark sets are similar. There is no apparent bias in the benchmark set proposed here towards domains which are easier to classify and both sets appear to be equally difficult regarding their similarity relationships.

The plots **(c)**, **(d)** in the first row in Figure 2.3 as well as **(e)** and **(f)** in row two introduce a novel type of plot to benchmark the performance of structure comparison methods. The plots can be used for any structure comparison method to evaluate in detail the classification performance and in particular the errors made by a method. Especially, they allow to estimate the performance of a method given a template database where members of the family and superfamily are missing and analyze in detail the number of domains for which problems occur in a set of domains and also quantify the dimension of the problem. Plots **(c)** and **(d)** show the number of domains for which we observe problems according to the structural similarity detected by TM-Align. For every query domain, we show how many domains from a different fold have a higher similarity score than the highest scoring member of the domain's own family (red cross), superfamily (green x) or fold (blue star). On the x-axis we show all query domains for which we observe problems, while on the y-axis, the number of problematic cases for a query (i.e. the number of domains from different a different fold ranked higher than the own family/superfamily/fold) is plotted.

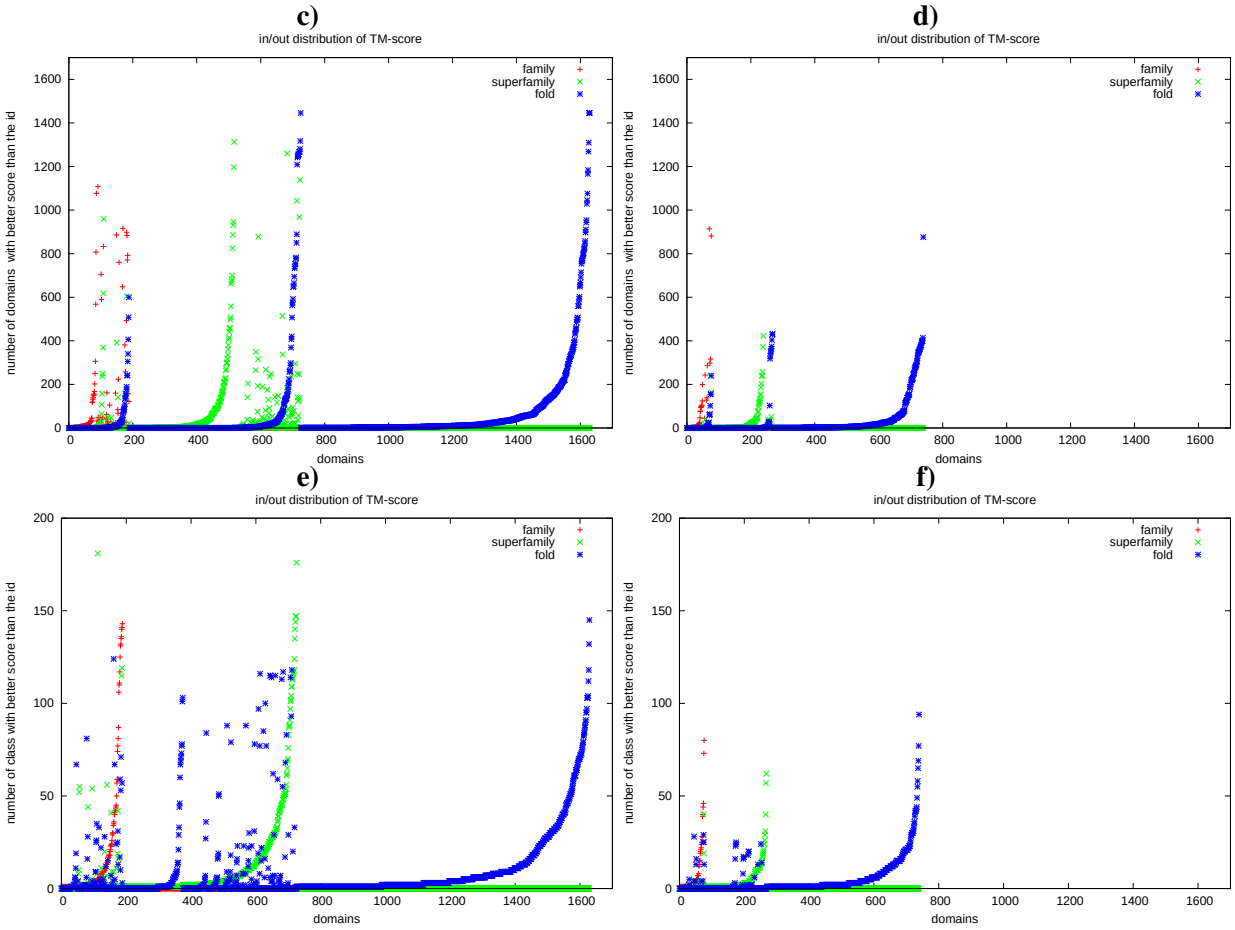


Figure 2.3: In/Out plots for TM-align-optimized TM-scores on the complete set of similarity relationships defined by SCOP (left column) and the performance on the novel benchmark set proposed in this study (right column). The plots are further discussed in section 2.4.1

For example if there are ten domains from a different fold scoring better than the most similar member from the domain's own family a red cross (at (x,10)) would be plotted. Similarly a blue dot is plotted if wrong proteins score better than a member of the querie's superfamily and a green x is plotted in the case of wrong domains scoring better than the own fold. Also, domains in columns which contain blue dots would not be assigned to their correct folds in the case of missing family and superfamily members since the best hit comes from a different fold.

Panels **(e)**-**(f)** in Figure 2.3 row three are similar to panels **(c)**-**(d)**, but instead of displaying the number of domains, they show the number of distinct folds (different from the domains own fold) which score better than the respective own family, superfamily or fold.

Comparing the plots that are computed based on the complete set of domain pairs (left column) with the plots computed on the benchmark set of consistent domain pairs (right column) we find that TM-Score / TM-align produces errors for only half of the domains tested and the dimension of the errors (i.e. the number of domains / folds which score better) also strongly decreases.

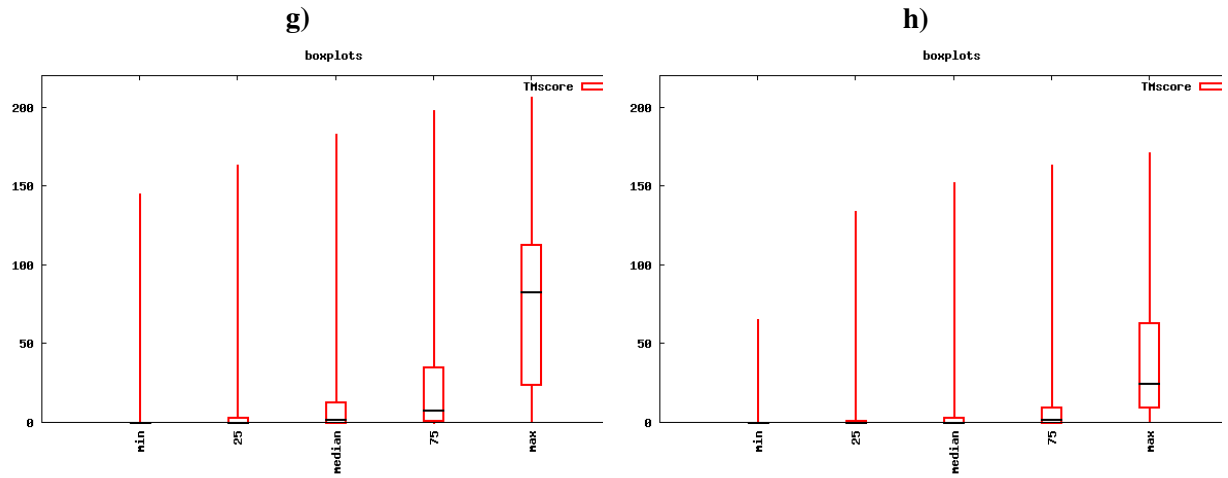


Figure 2.4: Distributions of ranks of similar proteins in TM-align ranked lists on SCOP (left column) and and the consistent mapping (right column). The plots are further discussed in section 2.4.1

The only difference between the two sets tested is the removal of pairs which are inconsistently defined between SCOP and CATH. While we remove about 16% of the positive pairs (in SCOP) to obtain the consistent set, the number of errors observed is reduced by 53% (compared to 16% error reduction which would be expected when removing arbitrary pairs). Therefore, the removal of pairs which are inconsistently defined in SCOP and CATH allows to over-proportionally reduce the number of errors. Our conclusions are twofold: many errors reported for protein structure classification methods originate from pairs of domains which are similar to one another, but are classified differently by SCOP or CATH. A different set of errors results from pairs that are e.g. classified in the same family but not similar enough to be distinguished from random pairs by a structure-based comparison method. Using only pairs of domains consistently defined in SCOP and CATH allows to reduce the amount of errors significantly and to separate erroneous behavior of a method (e.g. errors in the similarity model for protein structures implemented in a method) from problems arising due to pairs of domains for which even gold standards and experts disagree in their classification.

The plots **(g)** and **(h)** in Figure 2.4 completes and summarizes the analysis. To compute them, we sort the results obtained for every query domain according to their similarity scores. Then, we count for every member of the query fold, how many distinct other folds score better than the respective fold member (please note that every fold is counted only once even if multiple domains from a fold lead to errors). The boxplot in **(g)**-**(h)** shows the errors for five specific fold members: for the best and worst scoring fold members, as well as for the fold member placed at the 25%, median and 75% positions in the sorted list. As, unfortunately, correct fold members score quite differently, this allows to assess the overall performance of fold members by showing how often wrong members score better than the selected five fold member representatives. The boxplot now simply summarizes these numbers for all queries.

Thus the boxplots give a summarized overview of the observed errors. By comparing the two boxplots for the comprehensive and the consensus sets we again find a substantial reduction of errors in the consensus set. While the number of errors for the best scoring fold member is generally small, the errors for the low-scoring fold members quickly increase in both sets, but much more drastically in the comprehensive set as compared to the consensus set. For example, if we look at the fold members scored in the lower quarter (75%) of the fold members, we find 10 different random folds before a correct domain in the original dataset and only one fold in the novel benchmark set. This again indicates, that the number of errors as well as their quantity are significantly lower in the novel benchmark set compared to the original benchmark set.
Overall, the novel benchmark set proposed here is much more consistent than the original pairwise relationships defined by SCOP. It results in a much smaller number of errors (less than half the amount of the errors in the original set). Due to the largely reduced inconsistencies the set should also be well suited for training novel machine learning algorithms for protein structure classification, since it may allow for learning more consistent concepts from the input data.
Furthermore, we expect that the new benchmark set and also the new type of plots allow for a more instructive and objective evaluation of other structure comparison methods as well. The benchmark set is also used in section 6, 4 and 5.

2.4.2 Inter-Fold similarities revealed by consistency checks

As a second application, we have used our mapping of the two hierarchies in order to identify similarities of different folds / topologies defined in one hierarchy which are implied by mapping them onto the same fold / topology in the respective other classification scheme. Methods to detect possible interfold similarities have already been described for example by Friedberg et al. [58] and the CATH developers [84]. Here, we do not propose a novel approach to analyze such similarities from a structural point of view but utilize the orthogonal criteria and knowledge from two curated classification schemes to identify them. More specific, we search for folds f_1 defined in SCOP which map to a topology level in CATH t while this topology level in CATH also maps to a second fold f_2 in SCOP (see also Figure 2a).
The identification of such similarities provides interesting insights into the differences and similarities of fold classifications in SCOP and CATH and further allows to identify interesting links in the fold space. In order to propose a link we currently require the existence of at least five domains, which do not share a sequence identity of more than 50%, to support the link.
This analysis reveals a large number of singletons, i.e. folds / topologies with no link to another fold. 1137 folds in SCOP as well as 904 topologies in CATH turn out to be singletons. For relatively few folds / topologies similarities with other folds are identified which are interesting cases for further analyses in the context of protein structure and sequence evolution.
For SCOP, we identified 29 subgraphs, i.e. groups of folds which are connected via a link in CATH to another fold. 18 of the groups represent graphs of size 2, i.e pairs of folds while the other 11 subgraphs connect up to 39 different folds in SCOP. The largest graph contains a cluster of SCOP folds representing domains which are classified as Rossmann Fold Topology (3.40.50) in CATH but are splitted into 38 different folds in SCOP. Another large cluster comprises β-sandwich proteins with Greek-key topology which represent a cluster of 7 folds.

Two further interesting examples are shown in Figure 2. Figure 2b shows the interfold similarity of α-hairpin proteins in SCOP which are clustered in the same fold according to CATH (1.10.287). Part 2c shows a more complicated fold graph clustering proteins of jelly-roll (2.60.40, Immunoglobulin-Like) and immunoglobulin topologies (2.60.120, Jelly-Roll) in a subgraph which also shows that those graphs do not necessarily form a clique.

2.5 Methods

In the following, we will handle the two hierarchies (SCOP and CATH) as labeled trees, where the leaves correspond to the domains classified in the corresponding hierarchy. Inner nodes represent sets of protein domains which are clustered together on a specific level of the hierarchy. For SCOP, inner nodes represent classes, folds, superfamiles or families. For CATH, inner nodes correspond to classes, architectures, topologies and homologous superfamilies. We denote the underlying sets of domains of the two hierarchies with D_1, D_2 and the hierarchy trees themselves as H_1, H_2. We further define $H_i = (V_i, E_i)$ where $D_i \subseteq V_i$ are the leaves of the tree. Since domain definitions in different hierarchies also may be different, we have to map the domains defined by SCOP and CATH ($D_1 \leftrightarrow D_2$) in a first step. In a second step we will define and compute a mapping between inner nodes of the hierarchies.

2.5.1 Mapping of domain assignments

A protein domain is defined as a set of segments within one protein, where a segment is defined as a consecutive part of one chain of the protein. Please note that this definition also allows to define discontinuous domains and domains spanning different chains of a protein. In order to compare different domain assignments of the same protein we have to compare sets of segments. To do so, we use the sets of residue positions $RP(d)$ for all segments of domain d and define the similarity of segments via the intersection of their RP sets. Of course, such a mapping is not necessarily unique, i.e. it is possible that a domain in D_1 maps to more than one domain in D_2 or, more generally, that n domains in D_1 correspond to m domains in D_2. In such cases the definitions of the domains may be very different and we exclude domains from D_1, D_2 if their overlap o (see below) is smaller than a specified threshold. For two domains d_1 and d_2 from D_1 or D_2, respectively, we now define the overlap o of two domains as:

$$\mathrm{o}(d_1, d_2) = \frac{|\{\mathrm{RP}(d1)\} \cap \{\mathrm{RP}(d2)\}|}{|\{\mathrm{RP}(d1)\} \cup \{\mathrm{RP}(d2)\}|}$$

If we use a threshold $T_o > 0.5$ the mapping will be unique (but not necessarily complete).

2.5.2 Mapping inner nodes of the hierarchies

While the mapping of domains is more or less trivial (except for cases where the domain definitions differ to a large extent), mapping inner nodes of the hierarchy appears to be more complicated. As already mentioned inner nodes represent sets of domains. The image of a set of

domains in the one hierarchy is the set of domains in the other hierarchy where T_o exceeds a given threshold, i.e. for $S_1 \subseteq D_1$ (equivalently for $S_2 \subseteq D_2$) the image of S_1 is defined as follows:

$$img(S_1) = \{d_2 \in D_2 | \exists d_1 \in S_1 \text{ with } o(d_1, d_2) > T_o\}$$

Further, we define the sensitivity, specificity and the F-measure of a domain mapping of two sets $S_1 \subseteq img(D_2) \subseteq D_1$ and $S_2 \subseteq img(D_1) \subseteq D_2$ on the restricted hierarchies as:

$$\text{sensitivity}(S_1, S_2) = \frac{|S_1 \cap img(S_2)|}{|S_1|}$$

$$\text{specificity}(S_1, S_2) = \frac{|S_1 \cap img(S_2)|}{|S_1 \cup img(S_2)|}$$

$$\text{F-measure}(S_1, S_2) = \frac{2 * \text{sensitivity}(S_1, S_2) * \text{specificity}(S_1, S_2)}{\text{sensitivity}(S_1, S_2) + \text{specificity}(S_1, S_2)}$$

In order to map sets of domains, we search for all inner nodes S_1 from hierarchy H_1 and S_2 from hierarchy H_2 where F-measure(S_1,S_2) > 0, i.e. there needs to be at least one domain which occurs in both sets. The F-measure is especially useful as it accounts for a tradeoff between sensitivity and specificity. This is necessary since, obviously, the most sensitive mapping will be always the root, the most specific one the direct parent nodes of two mappable domains.
Given the F-measure for every pair of nodes which have at least one mappable domain in common, we identify the nodes in H_2 which match best to a given node n_1 in H_1. From each path from the root to n_2 in H_2, only one node (the best one according to the F-measure) will be used in the mapping. Nevertheless, there may be different paths in H_2 containing nodes mapped to the query node from H_1. In those cases matches from different paths are sorted according to their F-measures.
Based on those definitions we calculate for each non-leaf node n_1 in H_1 a sorted (by their F-measures) set $MS(n_1)$ consisting of non-leaf nodes in H_2 where: $\forall a, b \in MS$ a is not descendant of b and b is not descendant of a.

2.6 Benchmarking strategy

We define now two type of tests on the consistently mapped domain pair set: the "recognition" and the "misclassification" tests, both types for different levels of similarities available in the set. The main difference between the two test types is whether we test if we can recognize the similarity for exactly at a given similarity level (misclassification test), or if we test that we can recognize the similarity for a given similarity level *or below*.
For a given similarity-ranking method M (both sequence or structure based) the procedure is the same for all tests:
For a given query structure x we do a library scan, and calculate the similarity of all templates in the database. Then we take the best hit - the one with the highest method M specific score - and evaluate the performance based on th ranking of the similarities between x and this best-hit. The

different evaluation strategies define which subset of the library is to be evaluated.
Therefore in each test we test a set of well-defined similar templates against the set of negatives (members of other folds), therefore we can measure of the performance of the methods on the different similarity levels. As we perform these tests on the consensus hierarchy between CATH and SCOP the similarity levels are well defined.
The recognitions tests correspond in fact to the classical fold recognition task, whereas the real challenging test is to recognize the correct fold in the absence of higher level similarities (superfamily or family).
Of course it may happen that a given test-type for some similarity levels the set of the similar templates is empty in the library, and thus the possibility is also not available. Therefor the absolute number of tests may differ for each test type and similarity level.
In the following we describe each of the possible tests and the possible outcomes in detail.

Family misclassification test: Figure 2.5.a

In the misclassification tests we test each similarity level alone against the set of negatives (members of other folds). In the family misclassification test we use only the similarities of the family level, and test a given query against the template library. So we have only two possibilities for the best hit:

- it may come from the same family (blue arrow) - this would be the good case
- or it may come from a different fold (red arrow) - the method reports that a domain in an other fold is more similar than the all members of the family

Family recognition test: Figure 2.5.b

For the family recognition test all similarities are available, this is the easiest test (and used in most publications).
Here we have basically four possibilities where the first hit reported by the tested method may come from :

- it may come from the same family (blue arrow) - this would be the good case
- or it may come from a different family but from the same superfamily (magenta arrow)- which means the family recognition fails, but we recognize at least the correct superfamily
- or it may come from a different superfamily but from the same fold (green arrow)- again family recognition fails, but we find at least the correct fold
- or the worst case - we assign a wrong fold (red arrow) - i.e. the method reports that a domain in an other fold is more similar than the all members of the same fold.

Superfamily misclassification test Figure 2.5.c

In the superfamily misclassification test we use only the similarities of the superfamily level (i.e. no family members - no very closely related proteins, but still closely related proteins) and test a given query against the template library. So we have only two possibilities for the best hit:

- it may come from the same superfamily (but different family) (magenta arrow) - this would be the good case
- or it may come from a different fold (red arrow) - the method reports that a domain in an other fold is more similar than the all members of the superfamily

Superfamily recognition test Figure 2.5.c

For the superfamily recognition test similarities coming from the family level (i.e. are templates having the same family annotation as the query domain) are excluded. This setup makes the detection of the correct similarities already difficult for sequence methods as the sequence similarities between superfamily but not family members are much lower than between family members.
Here we have basically three possibilities where the first hit reported by the tested method may come from :

- or it may come from a different family but from the same superfamily (magenta arrow)- this is the good case
- or it may come from a different superfamily but from the same fold (green arrow)- here the superfamily recognition fails, but we find at least the correct fold
- or the worst case - we assign a wrong fold (red arrow) - i.e. the method reports that a domain in an other fold is more similar than the all members of the same superfamily or members of other superfamilies in the same fold.

Fold misclassification/Fold recognition test Figure 2.5.e

For the fold level the misclassification test and fold recognition test is the same, as for both tests only members of the same fold but different superfamilies are available. This test is considered as the “true” fold recognition test, as shown in the following chapters the performance at this level is far away from satisfying if only the sequence (and no structure) of the query is known. Here we have basically two possibilities where the first hit reported by the tested method may come from:

- it may come from the same fold (but different superfamilies) (green arrow) - this would be the good case
- or it may come from a different fold (red arrow)- the method cannot tell the difference between the right and wrong fold using only the remote similarities.

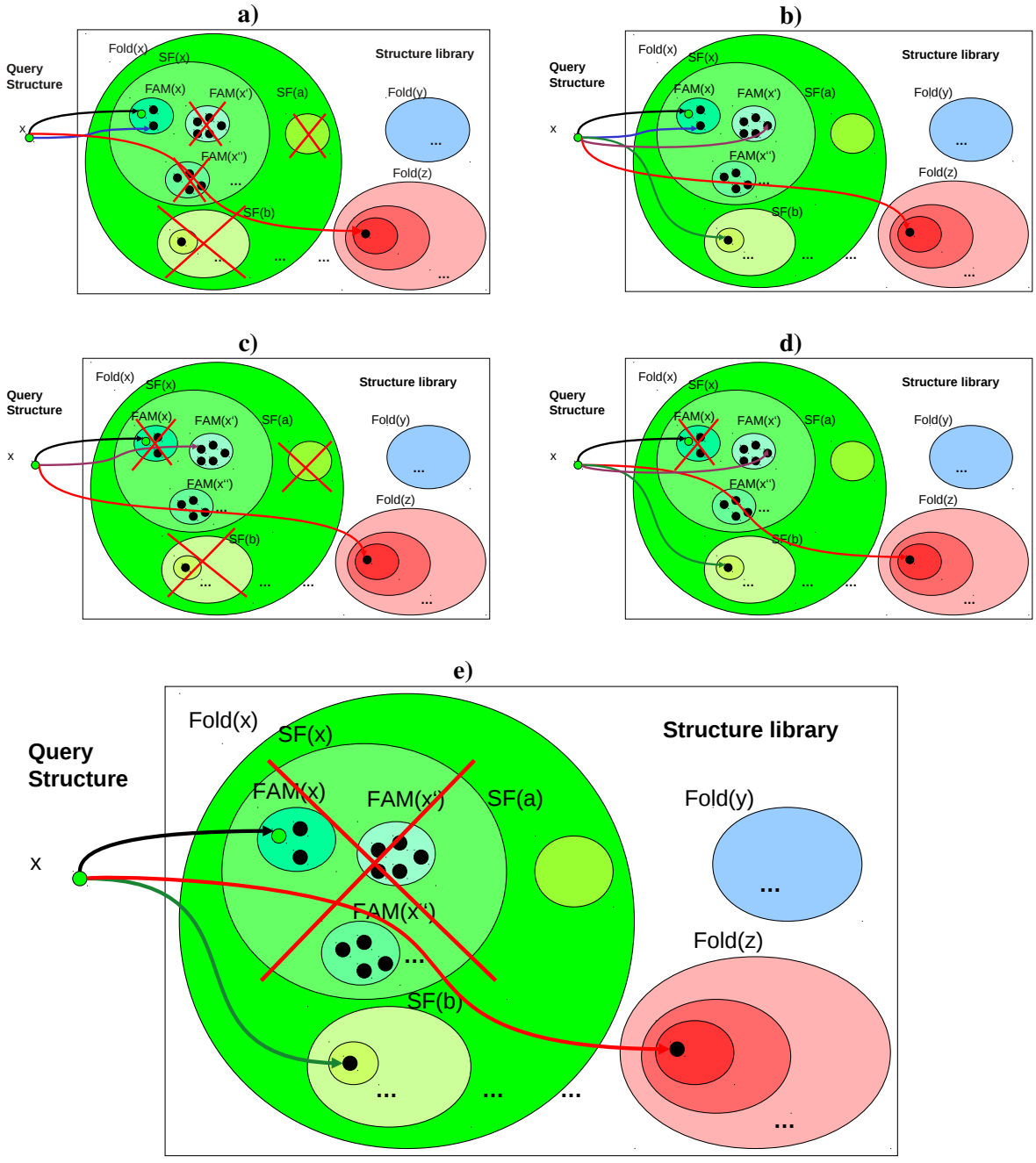


Figure 2.5: Visualization of Benchmarking Strategy based on the new gold standard

2.7 Conclusions

Protein structure classification is an essential step towards a deeper understanding into the interplay of protein structure and protein sequence evolution. Here, we have carried out a detailed study of the similarities and differences between the two most prominent databases, namely SCOP and CATH, which have become gold standards in the field and are used in various machine learning approaches and assessments of structure prediction and classification like the CASP experiments.
We find that there are essential differences between the two classification schemes due to their way of partitioning proteins into domains (which has already been described and discussed by [90]). SCOP tends to partition domains into fewer but larger components than CATH. In total, only about 70% of the domain definitions for proteins classified in SCOP and CATH agree (at an overlap threshold of 80%) and about one third of the families in SCOP and homologous superfamilies in CATH can not be mapped on domains of the respective other hierarchy.
For the remaining set of about 20000 proteins we then tested how well their classifications agreed with the classification in the respective other hierarchy. For this comparison we have used the F-measure to determine the similarity of two sets of domains on a specific level of two hierarchies. We find that both hierarchies show significant differences and often disagree in their way to partition the protein structure space also in cases of nearly identical domain definitions.
Given those findings and our mapping of SCOP and CATH hierarchy nodes, we extract a novel benchmark set of protein domain pairs which are defined consistently across both hierarchies. We show that benchmarking TM-align (as an exemplary structure classification method) on the novel benchmark set leads to a largely improved performance in comparison to the original set where the similarities as defined with respect to SCOP). This is due to the fact that errors (proteins which are similar according to one hierarchy but separated into different classes in the other one) which occur due to inconsistencies are removed from the novel benchmark set. Therefore, the benchmark set proposed here allows for a more objective evaluation of the performance of protein structure comparison methods as the remaining errors observed are more likely to be due to the method itself. Furthermore, this set should have advantages for both, training and testing all kinds of prediction methods, especially machine learning approaches to protein structure classification since more consistent concepts may be learned in the training phase.

The mapping between SCOP and CATH provides interesting, orthogonal knowledge on the topology of the protein structure space which allows to identify non-trivial links between different folds in e.g. SCOP via their connection observed in CATH. There are some very interesting and large (up to more than 30 folds) sets which may be clustered together in SCOP according to CATH. Among them are some known clusters of folds like the Rossmann Fold Topology. But there are also several other clusters of folds which may be interesting starting points for a further analysis of their sequence-structure properties and may help to further understand the interplay of protein sequence and protein structure evolution, also in the context of alternative splicing (see section 7), as different structure classifications reflect different viewpoints and criteria on structural and evolutionary similarity.

Finally using the well-defined similarities on different levels as well as the corresponding well-defined non-similar domains for a given query we defined corresponding benchmark strategies minimizing the influence of possible misannotations onto the performance of the method. In the following chapters we will extensively use the benchmark set defined in this chapter along with the benchmarking strategies.

Chapter 3

Vorolign - Fast structural alignment using Voronoi contacts

One way to get closer to an understanding of the big question "what makes proteins fold" is the analysis of similarities between known protein structures. To measure protein structure similarities in a meaningful way so call structural alignments have been used together with the associated alignment scores. Again a whole range of structural alignment algorithms, methods and scoring functions have been proposed. In this chapter we present another structural alignment method based on scoring of the amino acid contacts derived from Voronoi tessellations of the 3D coordinates.

The content of this chapter is a joint work with Fabian Birzele and Jan Gewehr and is based mainly on [22]. Vorolign is mainly the work of Fabian Birzele who developed the method during his PhD work and described the paper in his thesis.

The main finding in this chapter is the concept of the "contact environment potential", a novel way to describe similarities between proteins with similar structures based on their sequence and secondary structure. As this novel scoring concept is mainly sequence based, we want to know whether the environment potential could be used in a fold recognition or threading setup. This question motivates the following two chapters of my thesis: chapter 4 on VORESCORE and chapter 5) on ccVorolign, which both use the Vorolign method as a starting point and propose a new Re-scoring method and a new threading-style structure comparison method, respectively.

Therefore, for the sake of self sufficiency, we present the Vorolign approach here in full detail. Moreover, in the following chapters, we will evaluate Vorolign also on the benchmark set derived in chapter 2. The application of Vorolign to *multiple* structural alignments however is omitted for the purpose of this thesis.

My contribution to Vorolign is co-development and evaluation of the original method, the (faster) reimplementation of the algorithm.

3.1 Introduction

Protein structure comparison is an essential step to gain a deeper understanding of the interplay of protein sequence and structure evolution, to automatically classify proteins into families and to identify similarities that cannot be detected on the sequence level.
The problem has been heavily investigated in computational biology for more than two decades and numerous approaches have been developed to address different aspects of the problem. The methods proposed differ in the representation of protein structures in the algorithm, the procedure to identify structurally equivalent residues, the approach to combine similar regions to an alignment, as well as in the treatment of protein structures, either rigid or flexible. Also, while some of the methods address the problem of pairwise protein structure alignment, others align multiple structures. Multiple structure alignment is becoming more and more important with the growing number of protein structures in the PDB [19] for example in order to identify common structural cores of protein families.
Among the well known methods for pairwise comparison of protein structures are Dali [91] and CE [167], both treating proteins as rigid bodies, and FATCAT [200] that is able to align protein structures in a flexible way. Methods for multiple structure alignment are for example MultiProt [166], POSA [201] or CE-MC [78].
The focus of the different methods varies greatly, also due to the fact that it is not clear how exactly an ”optimal” solution to the structure comparison problem has to be defined [201]. Further, the 'best' structural alignment of two or more structures is hard to identify. For rigid body alignment algorithms, there is always a tradeoff between the number of residues that can be aligned below a distance threshold (so called number of equivalent residues, N_e) and the overall root mean square deviation ($RMSD$) of the superimposition of the aligned parts of the structures.
Proteins are dynamic entities and in many cases protein flexibility is essential for protein function. Therefore, treating proteins flexible instead of rigid can be an interesting and important feature in order to detect similarities between protein structures in different conformational states or between moved domains in multi-domain proteins. On the other hand, allowing for too many flexibilities in the protein chain and thus ignoring the overall structure of the protein can lead to an overestimation of the structural similarity.
In this paper, we propose a fast method to flexibly align two or more protein structures. The method is based on the assumption that the environments of two structurally equivalent residues are similar due to positive selection in order to ensure the structural integrity of the protein. We measure the similarity of the structural environments of two residues by their evolutionary relationship with respect to amino acid and secondary structure exchange scores and use this similarity function to align two protein structures using dynamic programming [139]. In contrast to other protein structure alignment methods that use mostly geometry-based similarity measurements we take the protein structure only implicitly into account, via the network of neighbouring residues (3D contacts), allowing a larger degree of flexibility of the protein structures being compared. Additionally, the more sequence-based similarity scoring function avoids certain common artifacts known from structural alignments where evolutionary equivalent residues are not aligned due to divergence of the structures (see Figure 3.1).
To represent the biochemical environment of a residue, we use its nearest-neighbours residues as

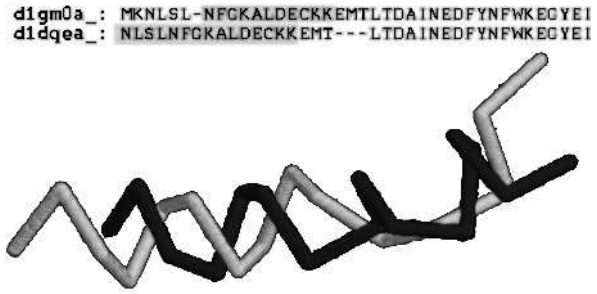


Figure 3.1: Shows part of the superimposition induced by the pairwise structural alignment of d1gm0a_ and d1dqea_ from SCOP class a.39.2.1 as aligned by CE. Though having almost identical sequences, identical residues are de-aligned due to structural divergence.

defined by the Voronoi tessellation [188] of the protein structure. Voronoi tessellation has been found to be useful for several tasks in structural bioinformatics including packing analysis [159], protein folding [65] and structure comparison ([95], [160]).
In our case, the major benefit of using the Voronoi tessellation, in contrast to distance-based contact definitions, is two-fold. First, we obtain a well defined set of nearest-neighbour contacts of a residue containing those residues that share a common face in the Voronoi diagram with the residue under consideration. Second, the contact set implicitly takes the geometry of the residue environment into account, since residues do not only have to be close in space but must also be direct neighbours without other residues between them.
The performance of our method, called Vorolign, is evaluated for various applications of protein structure comparison including pairwise and multiple structure alignment as well as the automated assignment of a protein to its corresponding protein family on a comprehensive set of difficult examples, derived from the difference set between the ASTRAL [28] versions 1.67 and 1.65.

3.2 Voronoi tessellation of protein structures

We use the C_β atoms (for Glycin the C_α atom) of the protein as the input set of points in the Euclidean space for the computation of the Voronoi decomposition. This decomposition partitions the space into convex polyhedra, called Voronoi cells. Each residue cell contains by definition all points that are closer to the corresponding C_β atom than to all other input nodes. All polyhedra, which are direct neighbours in space share a common face in the Voronoi decomposition corresponding to a contact of the two input points and, consequently, the two residues.
The Voronoi decomposition of a set of input points can be computed efficiently via standard computational geometry algorithms [147]. We use the quickhull algorithm as implemented in the QHULL program [14] to obtain the Voronoi tessellation of a given protein structure.
There is a problem, though, with a straightforward application of the standard procedures to obtain the Voronoi decomposition of proteins. Residues especially on the surface of the protein

will get unbounded, infinite Voronoi polyhedra and, which is even more unfavorable in our case, can share common faces with residues that are very distant in space. This effect can also be observed if the protein structure contains a larger cleft. Such residue pairs cannot be regarded as being nearest-neighbours in a native, aqueous environment and would lead to artifacts later on when scoring the evolutionary relationship of residue environments. In order to deal with this problem, we introduce explicit solvent atoms surrounding the protein using a method discussed by [209], which has been shown to be a fast and accurate method to approximate the water shell placed around the protein molecule. This method places solvent atoms in a regular three-dimensional grid with a distance between the grid nodes of d_{ll} Å within and around the protein structure. Then all solvent atoms that are closer than d_{pl} Å to any protein atom, are removed from the grid. We also remove all solvent molecules from the grid that are more than D_{pl} away from any protein site, with $D_{pl} = \sqrt{3 * d_{pl}^2}$. Following Zimmer et al. we set $d_{ll} = 3$ and $d_{pl} = 4$.

Properties of Voronoi cells

Our structural alignment routine employs the dynamic programming algorithm usually used to compare protein sequences [139] to align two protein structures. Standard sequence alignment routines use an amino acid exchange scoring matrix like PAM [44], to determine the similarity of two residues. We calculate the evolutionary similarity of two residues based on features of their corresponding Voronoi cells.

A priori, one could think of several, possibly conserved, properties that could be used to calculate the similarity of two cells. Examples are geometrical features like volume, shape or the surface area, biochemical properties of the faces or the nearest-neighbours of a cell, i.e. residues whose cells share a common face with the cell under consideration.

In this study we compute the evolutionary conservation of two Voronoi cells, i.e. their similarity, by using their nearest-neighbour environments and therefore the conservation of Voronoi contacts. Those features implicitly take the structure of the protein into account since they are derived from a structure-based process, i.e. the Voronoi tessellation. Nevertheless, they are sufficiently general to allow a certain degree of flexibility in the two protein structures being compared. This is an important feature to detect similarities across more diverse protein structures and for the comparison of multi-domain proteins with domain movements.

3.3 Similarity of Voronoi contacts

Given two proteins $X = x_1x_2...x_p$ and $Y = y_1y_2...y_q$ for which we want to calculate a structural alignment. The set of nearest-neighbours of a residue x_i, as defined by the Voronoi tessellation, is denoted as $N(x_i) = \{x_{i_1}, x_{i_2}, ..., x_{i_n}\}$. In order to measure the similarity of two residues x_i and y_j we will calculate the similarity of their corresponding nearest-neighbour sets $N(x_i)$ and $N(y_j)$.

3.3.1 Similarity of two nearest-neighbours

As a first step, we need to define the similarity of two residues x_{i_k} and y_{j_l} in the nearest-neighbour sets $N(x_i)$ and $N(y_i)$. This similarity will be scored by the weighted sum of two scores as given in the equation below:

$$Sim(x_{i_k}, y_{j_l}) = \omega_1 * AA(x_{i_k}, y_{j_l}) + \omega_2 * SSE(x_{i_k}, y_{j_l}). \tag{3.1}$$

In the equation, $AA(x_{i_k}, y_{j_l})$ corresponds to an amino acid exchange matrix score and $SSE(x_{i_k}, y_{j_l})$ scores the similarity of the corresponding secondary structure elements as defined by a secondary structure element scoring matrix.
We also tested the incorporation of other features like the Euclidean or sequential distance of the two residues to their central residue, the Euclidean distance of x_{i_k} and y_{j_l} with respect to a reference frame [30] as well as PSI-BLAST profiles [6], into the similarity scoring function. This did not lead to a significant change in the alignment accuracy indicating that our two features are sufficient to characterize the evolutionary relationship of the nearest-neighbour environments for our needs.

3.3.2 Similarity of nearest-neighbour sets

Having defined the similarity of two nearest-neighbour residues, we estimate the similarity of the two nearest-neighbour sets $N(x_i)$ and $N(y_i)$. For this we need a matching of the residues in the $N(x_i)$ and $N(y_j)$ sets. Once we have found such a correspondence, the final score (similarity) of the two nearest-neighbour environments is simply given by the sum of the similarities of the residues matched onto each other minus a penalty score for each unmatched residue.
In principle such a matching of the two nearest-neighbour sets could be calculated by different methods to solve matching problems in bipartite graphs like the Hungarian algorithm [110]. But since the possible matchings of the neighbours onto each other are constrained by their order in the protein sequence, the optimal solution can efficiently be computed using dynamic programming. The position of x_i (or y_j respectively) is taken into account by using two independent sets of nearest-neighbours, one containing all neighbours that are found left of the residue under consideration (x_i or y_i) in the protein sequence, the other set containing all residues found right of the residue in the sequence.
The similarity of two nearest-neighbour sets can be computed using dynamic programming with respect to the similarity function of two nearest-neighbour residues given in equation 3.1 and an additional penalty for unmatched residues p_u. This corresponds to calculating a global alignment with linear gap costs of the residues in the two nearest-neighbour sets. The entries $S(k, l)$ in the dynamic programming matrix S are filled using the following equation:

$$S(k,l) = max \begin{cases} S(k-1, l-1) + Sim(x_{i_k}, y_{j_l}) \\ S(k-1, l) - p_u \\ S(k, j-l) - p_u \end{cases} \tag{3.2}$$

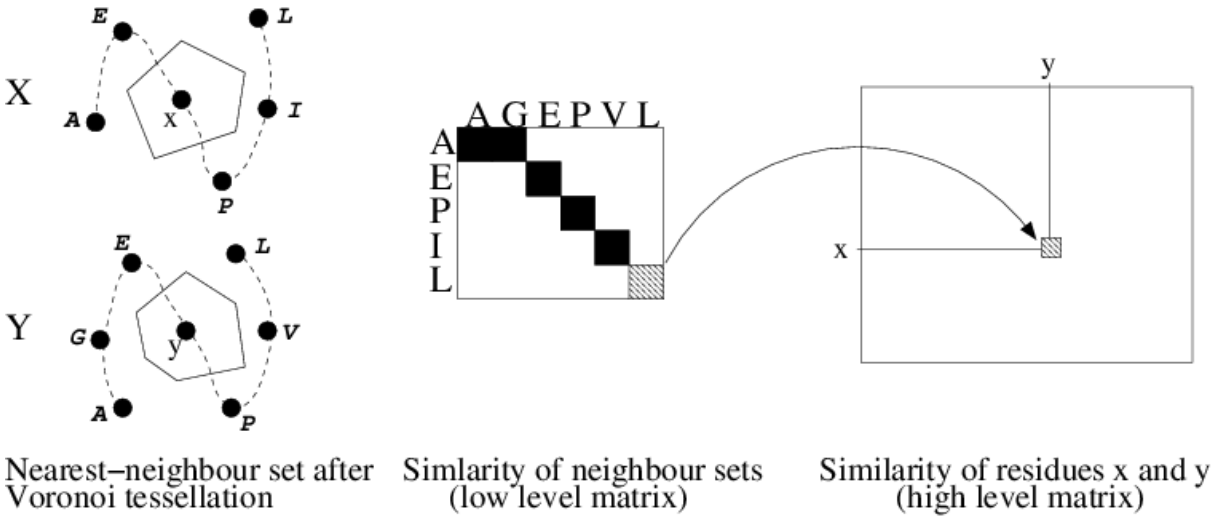


Figure 3.2: Shows the basic workflow of the Vorolign method. First, the neighbour sets of each residue x and y in the two structures X and Y are defined by Voronoi Tessellation. The similarity of two neighbour sets is calculated using dynamic programming (low level matrix) and equation 3.2. The score of the low level matrix is used to fill the high level dynamic programming matrix. Please notice: to keep the figure simple, we did not split up the neighbour sets as discussed in section 3.3.2.

The final score of that alignment, i.e. the entry in the last row and column of the matrix S, is used to score the similarity of the two environments of the residues x_i and y_i and is denoted as $Sim(N(x_i), N(y_i))$ in the following.

3.4 Pairwise alignment of protein structures

Having defined a function to measure the similarity of two residues x_i and y_j, we can align two protein structures using any dynamic programming algorithm and the similarity function $Sim(N(x_i), N(y_i))$. The method is summarized in figure 3.2. The choice of the algorithm (global, free-shift, local) as well as the gap penalty model (linear or affine) depends on the application and all standard methods are available in the Vorolign program. Other features that influence the alignment quality are the scoring matrices chosen to measure the amino acid and secondary structure element similarity of two nearest-neighbour residues together with their corresponding weights ω_1 and ω_2 (see equation 3.1) as well as gap penalties for the low level (p_u) and the high level (gap open: G_O and gap extend: G_E) dynamic programming steps.
All parameters have been optimized for four different amino acid exchange matrices using a genetic algorithm [48] as discussed in section 3.6.
The idea of using a low level dynamic programming step in order to fill a higher level dynamic programming matrix is also known as 'double dynamic programming' [177] and has already been used in the context of protein structure alignment ([176], [177]). Their way to calculate the similarity in the low level matrix differs substantially from ours.

For a pair of residues (i, j), the two structures are centered and superimposed with respect to the adjacent residues. With respect to that superimposition all pairs of residues are examined and their similarity is measured using mainly geometric features. The score of an alignment of all residues of the two protein structures is used to fill the high level matrix entry for the pair (i, j). In comparison, Vorolign concentrates on the nearest-neighbour residues to compute the similarity score by using amino acid and secondary structure element conservation. This significantly reduces the costs of each similarity computation and allows Vorolign to calculate flexible structural alignments.

3.5 Fast scan for family members

Protein structure alignment is often used for the automatic detection of structurally similar proteins in a database, given a structurally resolved, but still unclassified protein. A similar application would be the generation of all-against-all alignments of a set of proteins in the PDB in order to define protein families. For that application, not only the accuracy of the method in detecting proteins that belong to the same family, e.g. a SCOP [136] class, but also the speed of the database scan is important. Since Vorolign is a very fast method for generating structural alignments, we tested the performance of the method in detecting the correct SCOP family for a target protein in a database of 4358 family representatives from ASTRAL25. The results are discussed in section 3.7.
For all database proteins, the Voronoi tessellation and the corresponding nearest-neighbour sets can be pre-calculated in order to speed up the scan. After the calculation of the Voronoi tessellation for the target protein, the protein can be aligned against the database. To avoid the calculation of Vorolign alignments between protein structures that do not show a significant similarity on the secondary structure level, e.g. an all-β protein against an all-α protein, we employ secondary structure element alignment (SSEA) [127] to pre-filter the template database. In the current setup, we use the 250 highest scoring hits (~5% of the template database) found by SSEA which are subsequently aligned to the target using the Vorolign method. The results are then ranked with respect to their Vorolign alignment scores and for the best 20 alignments we compute structural superimpositions using the TMscore program [207].

3.6 Parameter optimization

Parameters used throughout the experiments have been optimized on a set of 20 randomly chosen protein pairs. None of the training protein pairs has been included in the test cases discussed later on. We used a genetic algorithm [48] with the average TM-Score of the structural alignments as fitness function. Proteins were repeatedly aligned by Vorolign using the values of the population member until the genetic algorithm converged. As convergence criterion we used 5 generations without a change of the fittest population member. The algorithm has been restarted 50 times.
Parameters were optimized for free-shift alignment with affine gap penalties (G_O and G_E) as well as four different structure-based amino acid exchange matrices: the Blake-Cohen matrix

Matrix	ω_1	ω_2	p_u	G_O	G_E
BC	0.3	2.52	1.23	-16.58	-6.15
P	2.03	2.42	1.78	-13.68	-4.83
T	0.58	0.71	5.04	-22.50	-6.50
N	0.26	0.96	4.45	-22.866	-8.91

Table 3.1: Optimized parameter settings for the three different amino acid exchange matrices as found by the genetic algorithm.

[24] (BC), a matrix by [155] (P), by [13] (N) and the SM_THREADER matrix [49] (T). As secondary structure scoring matrix we used the matrix described by [192]. The optimization results are shown in Table 3.1.

3.7 Family recognition and pairwise structural alignments

Method	Family	Superfamily	Fold	Wrong	TM-Score	$\%N_e$	Subset RMSD
Vorolign, T	**86.4%**	**92.4%**	**97.7%**	**2.3%**	0.74	76.3	**1.90** Å
Vorolign, N	83.7%	90.4%	96.1%	3.9%	0.71	74.1	1.93 Å
Vorolign, P	82.9%	88.9%	91.4%	8.6%	0.70	73.1	2.01 Å
Vorolign, BC	81.6%	87.6%	89.9%	10.1%	0.69	73.3	2.04 Å
CE_1	81.8%	88.4%	90.6%	9.4%	**0.78**	77.8	1.93 Å
CE_2	84.6%	91.9%	94.1%	5.9%	0.77	**78.2**	1.95 Å
C_β, T	82.9%	89.5%	94.4%	5.6%	0.70	73.2	1.97 Å
PPA	79.6%	85.7%	89.2%	10.8%	0.56	64.3	2.14 Å
Free-shift alignment	65.8%	68.0%	69.6%	30.4%	0.49	49.0	2.16 Å
SSEA	60.8%	68.9%	75.6%	24.4%	–	–	–
Blast	48.9%	52.5%	52.8%	47.2%	–	–	–

Table 3.2: Results of the family recognition scan and the corresponding average pairwise alignment quality of the methods. Methods are evaluated taking the best hit with respect to the method specific score into account. For Vorolign, C_β, SSEA, free-shift alignment and PPA those are the respective alignment scores, for Blast we use the best E-Value and for CE the best z-score. If CE returns more than one alignment for a target-template pair the z-Score of the longest alignment (CE_1) or the best z-Score of a pair (CE_2) is used. For SSEA no alignment quality is evaluated since the method does not compute residue-based alignments. A similar argument applies for Blast since it returns only short, local alignments.

In order to test the ability of Vorolign to detect the correct protein family in a database of representative protein structures, we used 979 proteins from the ASTRAL compendium that are contained in version 1.67 (February 2005) but not in the previous version 1.65 (December 2003).

From the 10039 domains contained in the difference set of version 1.67 and 1.65 (excluding genetic domains), we removed all domains for which the sequence identity using standard sequence alignment methods was found to be above 30%, with more than 30 identical residues. The final set of 979 proteins therefore comprises the non-trivial cases out of the 10039 targets and allows us to asses the 'blind test' performance of our and other methods in predicting the SCOP family for a given target.
Those proteins were scanned against the ASTRAL25 database (version 1.65) which includes 4358 proteins using the setup described in section 3.5. All four amino acid exchange matrices from section 3.6 are evaluated.
The performance of all methods will be measured as the fraction of proteins that can be assigned to their correct family, superfamily or fold with respect to the SCOP classification. For Vorolign, only the best template due to the Vorolign alignment score is taken into account for the evaluation. To account for the quality of the corresponding structural alignments, we use TM-Score, the percentage of equivalently aligned residues (N_e) and the average RMSD of the superimposition of the residues in the N_e subset given the alignment. The results can be found in Table 3.2.
The performance of Vorolign is compared against several other methods. SSEA, free-shift sequence alignment with affine gap costs (Pam250 matrix, $G_O = -12$, $G_E = -1$), Blast (standard options, against Astral25 database) [6], Profile-Profile alignment (PPA) as used in [186] as well as CE, which is faster than for example Dali and performed best in a recent study of protein fold comparison servers [143]. All methods (except for Blast) were given the same 250 best hits from the SSEA pre-filter and again only the best template with respect to the method specific quality score is taken into account.
To show the advantage of using Voronoi instead of C_β distance-based contacts, we test the performance of the method using neighbour-sets defined by C_β contacts (threshold 6.5 Å) together with the SM_THREADER matrix. Parameters have been optimized as discussed in 3.6 and are set to $G_O = -20.07, G_E = -8.76, \omega_1 = 0.71, \omega_2 = 1.47, p_u = 4.76$.
When comparing the performance of the four different amino acid exchange matrices in combination with the Vorolign method, we find the different matrices are differently well suited to score the evolutionary similarity of residue environments. The SM_THREADER matrix performs best with respect to all quality measurements applied. This finding is interesting, since this matrix is not computed with respect to evolutionary criteria but expresses the contribution of a residue to the total energy of the protein. This could be an interesting starting point for future research, using potentials instead of amino acid exchange scores to score the similarity of two residue environments in combination with the Vorolign method.
A comparison of the results of the SM_THREADER matrix in combination with nearest-neighbour sets defined by Voronoi or C_β distance finds Voronoi contacts to perform better than C_β-based contacts according to recognition rates as well as alignment quality. This difference cannot be expected to be large since the nearest-neighbour sets will, of course, overlap, but the advantage of the Voronoi-based contact definition gained in comparison to C_β contacts in our opinion justifies the small overhead of computing the Voronoi tessellation in order to retrieve the nearest-neighbour sets.
The results also show that standard sequence alignment methods, i.e. Blast and Smith-Waterman, are not able to make accurate family, superfamily or even fold assignments for our test set and

demonstrate that the set contains the more challenging cases. The structural quality of the alignments is also poor.
Methods applied in the field of fold recognition, namely SSEA and PPA, gain higher recognition rates. The results of secondary structure element alignment justify its application as a pre-filter to speed up computationally more expensive methods like Vorolign, CE and PPA. Profile-Profile alignment performs surprisingly well with respect to the family, superfamily and fold recognition rates. In almost 90% of the cases the method is able to identify the correct fold of the target. Nevertheless, the structural quality of the alignments is not comparable to Vorolign and CE with an average subset size of 64.3% and a TM-Score value of 0.56%.
Of course, a fair comparison of the abilities of Vorolign can only be made with another structural alignment method. With respect to the recognition rates Vorolign and CE outperform all other methods discussed above. Interestingly, Vorolign in combination with the T and the N matrices is more accurate in predicting the correct family, superfamily or fold of a target than CE, making only 2.3% wrong assignments in comparison to 5.9% wrong assignments made by CE. The cases where Vorolign is correct while CE is wrong can be assigned to two different types of errors. Often CE identifies several hits with high z-scores in the template database. From a structural point of view all those hits could be regarded as being "true" hits since their structures can be superimposed well spanning large parts of the proteins. But since the SCOP hierarchy is not solely based on structural but also other criteria like sequence and function the structure-related z-score of CE cannot distinguish between the true and the false hits with respect to the SCOP annotation. Here, the more sequence-based similarity score of Vorolign seems to be an advantage, taking not only structural but also sequential knowledge into account. Second, in some rare cases the CE alignment with the true family member is not optimal leading to a bad z-score for the true family. With respect to the alignment quality, CE performs slightly better than Vorolign according to TM-Score and the number of equivalently aligned residues. This result could have been expected since Vorolign does not attempt to optimize the superimposition of the two structures at any point in the algorithm.
A Vorolign scan of a target against the database (including Vorolign alignments against 250 template proteins) takes on average 1 minute on a single CPU. In comparison, the freely available implementation of CE takes on average 30 seconds per protein and therefore about 2 hour per scan in the same environment.

3.8 Conclusions

The presented method "Vorolign" addresses the protein structure alignment problem a novel way. The method is based on a similarity function to measure the similarity of two sets of nearest-neighbour residues using amino acid and secondary structure exchange matrices together with double dynamic programming. The nearest-neighbour environment of a residue is defined by the Voronoi tessellation of the corresponding protein structure leading to a well-defined set of nearest-neighbours while taking the geometry of the protein implicitly into account.

In this chapter, the performance of Vorolign is evaluated for automatic protein family assignment,

other evaluations will follow in the next chapters. The results indicate that the sequence and secondary structure information along with contact information is sufficient to recognize similarities without exact knowledge about the 3D structure (and without explicitly aligning the 2D concact maps as DALI [91] does). For this thesis, this is also the main result of Vorolign: it serves as starting point for further analysis and methods for the rescoring of fold-recognition models (Vorescore, see chapter 4) and even closer step for a threading-style application (cc-Vorolign see chapter 5).

Chapter 4

Vorescore - fold recognition improved by rescoring of protein structure models

In this chapter we work in the protein structure prediction context: the goal is to find the correct 3D-fold for a protein sequence with (yet) unknown structure. Moreover we will use both the context of the contact environment potential from chapter 3 and the properties of the protein structure similarity context from chapter 2.

The main idea in this chapter is to employ the Vorolign score for the rescoring problem: often fold recognition and alignment approaches are in principle able to identify the most similar protein structure templates - but not necessarily in the right order as ranked by the used score. Therefore, a better suited score might be applied to compute the new for the produced results and to re-rank them according to the new scores. The goal, of course, is to identify the best fitting closest templates to the native structure and to place it the first place of a prediction.

The main finding here is that the contact environment scoring is able to capture the "fitness" of a given protein sequence in a template protein structure and clearly improves the recognition of the correct fold. Besides using the contact environment potential for scoring, the presented method "Vorescore" also explicitly makes use of the protein structure similarity context: it uses the hierarchy of similarities to "normalize" this fitness between different folds. While the results in this chapter confirm that the contact environment potential could be useful in the protein structure prediction context, the practical time complexity and the need for higher quality models motivates the quest for an explicit and direct optimization of the potential. This problem is addressed in chapter 5 by ccVorolign.

The Vorescore method - a first extension of Vorolign towards protein structure prediction - and all results presented here have been published in 2010 [40] and presented at the ECCB'10 in Ghent. The following chapter is a reprinted and reformatted version of this paper and otherwise unchanged.

4.1 Introduction

Protein structure prediction is one of oldest and most investigated problems in bionformatics. Many methods have been proposed and for quite some time systematically assessed in the CASP experiment (bi-annually since 1994). Despite the hardness of the problem, homology-based methods appeared to be quite successful due to improved alignment methods and more and more available template structures.

These methods produce target-template alignments and associated alignment scores. The alignments can be employed to derive structural models for the target using the 3D coordinates of the template structure according to the alignment. The alignment score can be used to rank several candidate templates and the associated alignments to identify the best suited one.

A large range of methods have been proposed for both of the two related but not identical problems: the alignment problem and the model scoring problem. A consensus on the ultimate method have not been reached in both cases as can be seen from the results and discussion in the CASP experiments. Typically, different methods have their strengths for different targets.

Often computing the correct alignment is the crucial step in homology-based modeling, as wrong alignments typically cannot be corrected later on during the modeling.

On the other hand, the scoring system of the alignment methods is far from perfect. In some cases the method can recognize the best template with high confidence (for this usually a Z-score or p-value is used). But in the absence of highly similar templates, e.g. when only templates from the same fold but neither from the same family nor the same superfamily are available, the scoring is critical.

Even if the scoring system would be appropriate to identify the best alignment between the target and a single template this would not necessarily imply that the score can be used to compare and rank different alignments to different template structures or the resulting model structures. On the other hand, in many cases the alignment between the target and an appropriate template is good enough to build a good model, but the alignment method ranks this alignment lower than an alignment with a wrong template. This can be due to the fact that the alignment scoring is not comparable between different template structures.

Thus, many approaches have been proposed to **rescore** the produced **models** using more involved scoring functions which are not and in many cases cannot be directly optimized by the alignment method. A wide variety of those methods, so called model quality assessment programs (or MQAPs), have been used with good success in the CASP experiments, e.g. as so called MQAP metaservers employing some consensus of several MQAPs [152]. In principle, the rescoring approaches can be classified into physics-based systems used for model scoring in do novo folding approaches such as Rosetta and heuristic scoring systems using different physico-chemical features and empirically tuned functions.

In this paper we propose a particularly simple rescoring system used in the successful structure alignment program Vorolign [22]. Moreover, the Vorpsi and Vorescore methods show how to employ a structure comparison approach for structure prediction. The results presented demonstrate that conservation in structural neighborhoods is an important feature for sequence-structure relationships and a determining factor for protein structures.

4.2 Methods

The rescoring method we propose is based on the idea of scoring the compatibility of a (target) sequence with a proposed model structure derived form an alignment of the target sequence with a template. The (re-)scoring of the model structure is done via comparing the two structures, the native template structure and the target model structure via a structural alignment method. We use Vorolign for the latter as it has a very high accuracy on the family ($\sim 97\%$) and superfamily ($\sim 90\%$) but in contrast to other methods also on the fold level ($\sim 80\%$) (see [22] and section 4.9). Moreover, it is a very simple method which is especially targeted at exploiting sequence information in the context of structural contacts. Therefore, it is well suited for the task at hand. The rescoring approach shows how to make a successful structure comparison approach (Vorolign) available for structure prediction (Vorpsi and Vorescore).
In the following sections, we describe the principle of Voronoi contacts and the Vorolign structure alignment method. Then we discuss the quality measures we need for computing and assessing structural models (TM-align, TM-score, PPM). The rescoring method is introduced in two variants: Vorpsi using only a single Vorolign rescore for each computed model and Vorescore which takes advantage of additional Vorolign model scores, i.e. Vorpsi scores, computed for a set of similar templates. Finally, we describe the assessment setup, which allows to evaluate the performance of the method and its comparison with other prediction and rescoring methods on different levels of target difficulty (family, superfamily, and fold level).
All methods presented in this section are based on Vorolign introduced in chapter 3. In the following we will denote the Vorolign with standard parameters with VOROLIGN.

4.3 Homology-based protein structure prediction and model assessment

Homology-based protein structure prediction methods rely on an alignment of the target sequence with a template structure from which model coordinates are derived according to the alignment.
Here, we use simple pairwise alignment of sequences with a *Dayhoff*-like substitution matrix and affine gap costs. Optimal alignments can be efficiently computed with the Gotoh algorithm [73]. The Gotoh algorithm with the *Dayhoff* matrix [44] and gap open costs of -12 and gap extension costs of -1 computing global alignments is called GOTOH in the following.
In addition we use a sensitive alignment method based on the alignment of hidden markov models. For this we use the in the latest CASP experiments very successful method HHalign from the HHpred toolbox [171] with standard parameters and call it HHALIGN.
For the generation of candidate models we additionally used 123D [5] and profile-profile alignment (PPA) [187] also with standard parameters given in the papers.
As model quality assessment method we use the Rosetta scoring function (called ROSETTA) from the Rosetta package [156] with standard parameters. and the ProQ method. For ProQ we used two versions, one based on C_α atoms from the Pcons package [191] and the standalone

method ProQ(combined) = 5*ProQ-MaxSub + 1*ProQ-LG [190]. The latter ProQ version performs much better such that this one is called PROQ and used in the following.

4.4 Protein structure comparison measures

There is a wide range of methods to measure the structural similarity of protein structure and, related to that, the similarity of a model structure to a native structure. The latter problem is used to assess the quality of structure predictions (e.g. in the CASP experiment) and is somewhat easier as the involved protein sequences are the same and, thus, the alignment is given.
Well known current methods are Vorolign (see above), PPM [38], TM-align optimizing the TM-score [207], GDT_TS [203], LG-score [116], and MaxSub [168]. TM-score and GDT_TS are often used by assessors and predictors to evaluate predicted models in the CASP experiment.
Due to the importance of the problem e.g. for protein structure classification, structure analysis and structure prediction many more approaches have been proposed, classical ones such as CE [167]) and also recent ones (Mustang [106], STACCATO [165], FATCAT [200], both pairwise and multiple alignment methods.
Interestingly, the problem of protein structure comparison has not yet reached overall consensus and there is still room for discussion on both the best alignment and the best score. This is also true for the best model assessment method as all methods arguably have some difficulties.
As GDT is kind of 'official' CASP assessment protocol we focus on the TM-score as an independent measure for similarity of model and native structure and use it in the following as our main model quality measure. TM-score is also supposed to be quite independent from the protein lengths. E.g., in the results we evaluate the performance of our rescoring methods Vorpsi and Vorescore with respect to the TM-score and the increase of TM-score in comparison with models predicted by competing methods.

4.5 Rescoring of homology-based models

Rescoring methods predict 3D structural models for protein target sequences. They consist of the following steps (see figure 4.1):

1. a prediction method is used to propose model structures
2. if the prediction/alignment method produces a highly confident model and alignment with a template structure, we take this template-induced model as the prediction (i.e. Z-score > 7.0 for HHALIGN).
3. otherwise, the models are rescored to determine the best scoring model as the final structure prediction.

For the prediction step, typically, alignment or threading methods are used to align the target sequence against a library of protein templates. This yields a ranked list of alignments with associated scores (*prediction method scores* or *alignment scores*), which are used to rank the

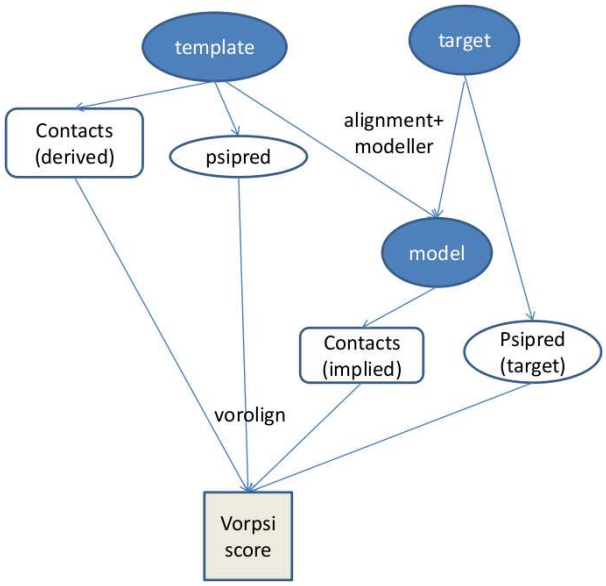


Figure 4.1: Overview of the Vorpsi method.

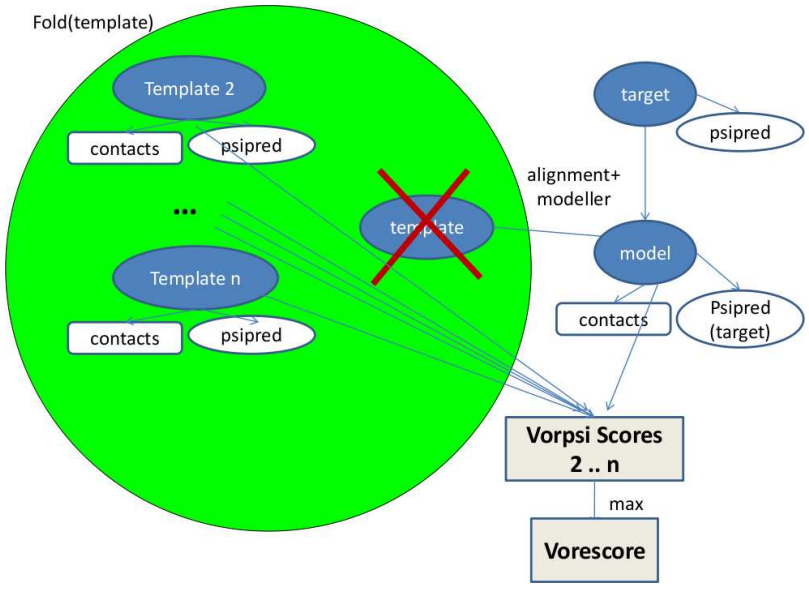


Figure 4.2: Overview of the Vorescore method exploiting structural classifications.

alignments and to select the best one. As prediction method many methods have been proposed such as pairwise and multiple sequence alignment [73, 74], profile [120] and hidden markov alignments [50, 51], profile-profile [187] and HMM-HMM [171] alignments, and structure based alignments (123D [5], RDP [179]).

The alignment is used to derive structure models for the target protein from the template coordinates. This can be done by copying the template coordinates according to the alignment or via modeling programs, e.g. the program modeller [54] can be used for this purpose.

4.6 Scoring of a model based on the template: Vorpsi

The rescoring method VORPSI needs for the second, (rescoring) step only the model and template structures. VORPSI tries to evaluate the compatibility of the target sequence with the proposed model structure. For this, it employs the VOROLIGN structural comparison between the model structure for the target with the native template structure used for the model. If alignment approaches are used to produce the model structures, also the alignments and in particular also the used templates are known and can directly be used. Otherwise templates need to be identified via structural search with the model structure against a template library.

The rescoring method computes new model scores by comparing the model structure with native template structures via VOROLIGN. The resulting model scores are sorted and the best scoring model is selected as the VORPSI structure prediction for the target.

Thus, the rescore score replaces the original prediction method (alignment) score to select the best model. This selection can be better or worse than the original one. This is assessed by comparing the respective models using the TM-score between the model and the native structure. The evaluation estimates how much and how often the TM-score is increased by the rescored model.

The rationale of this simple rescoring approach is as follows. Even simple prediction methods might be able to propose good models, but maybe are not able to score and rank them appropriately. Vorolign compares two proteins based on their structure, secondary structure and sequence. For every residue in a structure, VOROLIGN analyses the residue contacts with neighboring amino acids. In aligning two residues in the two structures, the match score is determined via aligning the contacting neighbors and scoring their similarity using a substitution matrix and their secondary structure state. Thereby, VOROLIGN, allowing for quite some structural flexibility, successfully scores the sequence-structure compatibility with fold recognition success rate of about 98%. As we will show in the following this approach appears to be very useful for model selection as well and, together with an appropriate alignment method, for protein structure prediction, especially fold recognition. Therefore, the VOROLIGN score likely is a better criterion to select good models for the target protein.

4.7 Scoring of models using knowledge of the structure space: Vorescore

In order to exploit knowledge on the sequence-structure space for the model selection problem we use the following simple approach. Instead of using the template structure of the model for the rescoring we take all structures in the fold of the template, apply VOROLIGN rescoring, and take the best score among these templates. This requires to compute additional alignments (VOROLIGN) for the selected templates in the fold.
In more detail, VORESCORE works as follows (see figure 4.2):

1. apply VORPSI for the target
2. if the best rescored template does not have additional structures beyond its family in the same fold, VORESCORE takes the best scored model.
3. otherwise, for every model from a template t, new alignments/models are computed for the target using all templates in the fold of t except for the family of t. The best of these templates according to the VOROLIGN score is selected as the VORESCORE prediction.

1. If the alignment method produces a highly confident alignment and model with a template structure, Vorescore takes this template induced model as its prediction (i.e. Z-score ¿ 7.0 for HHALIGN).
2. Otherwise, we rescore all the produced models with Vorolign against their respective templates and sort the models with respect to this Vorolign score. If the best rescored template does not have additional structures beyond its family in the same fold, Vorescore takes the best scored model.
3. Otherwise, for every model from a template t, new alignments/models are computed for the target using all templates in the fold of t except for the family of t. The best of these templates according to the Vorolign score is selected as the Vorescore prediction.

The rationale behind VORESCORE is the following: If the prediction method mistakenly ranks a certain template best, e.g. because of chance similarities with the wrong and missing sequence similarities with the correct template, chances are high that the Vorspi rescoring will also be mistaken. In these cases, we resort to alternative models taken from the same fold. Therefore, we explicitly remove all family members of the original template and take all its fold members as alternative candidates. By aligning and rescoring them with VORPSI, VORESCORE selects its predicted model in this case.

4.8 Results and discussion

We propose to employ our structure comparison method VOROLIGN for structure prediction by using it for rescoring structural models produced by some alignment method. The assessment

works for any alignment method, which aligns target sequences against structural templates. The rescoring then VOROLIGNs the model structure with the template structure used for the model building.
We have comprehensively evaluated the VORESCORE method for a range of alignment and threading methods. Here we focus on results for representative methods, a simple method and a highly sensitive and accurate method producing poor and high-quality alignments and models, respectively. We compare our new method Vorescore against a range of quality assessment and rescoring approaches and present results on a physics-based (ROSETTA) and an empirical PROQ method as examples.
Our results on VORESCORE from this assessment are as follows: First (section 4.9), we define an appropriate test-set of 410 target proteins for the performance evaluation. This set is difficult for sequence-based alignment methods, but allows for good structural template-based models with a TM-score above 0.3 (up to high scores of above 0.9). The test-set is derived from the CATHSCOP consensus set and allows for an unbiased, large-scale evaluation of the performance in different scenarios and different levels of difficulty, e.g. family, superfamily, and fold recognition.
Second (section 4.10), we apply different alignment methods such as pairwise sequence alignment (GOTOH) and hidden markov model alignment HHALIGN). From the alignments we build model structures using modeller [54] and rescore the resulting models with VOROLIGN and other rescoring methods such as ROSETTA und PROQ. The results show improvements of VORESCORE rescoring over the original alignments in about 40% of the cases, significant improvements of VOROLIGN over ROSETTA, which cannot really be used for successful rescoring of models and some improvement over the best state-of-the-art rescoring methods (such as PROQ) using involved scoring methods and meta-/consensus approaches.
Finally, in (section 4.11), we evaluate the potential of the rescoring if very good structural alignments and models are available. On one hand, VORESCORE can improve by another 50% of the cases as compared to the GOTOH or HHALIGN models, which again shows the ability of the simple VOROLIGN scoring to select good models. On the other hand, there is still much room for improvement: even if very good models are available the selection process misses the best models in many cases, and this margin is even larger for the rescoring of actual models predicted by sequence methods, which reinforces the necessity of producing good alignments (and models) in the first place.

4.9 Definition of an appropriate test-set

CATH [146] and SCOP [136] are the two most prominent hierarchical classifications of protein structure domains. Unfortunately, they are not completely consistent wrt. similarities and dissimilarities. Therefore, we defined a comprehensive and consistent set of similar (and maybe more importantly dissimilar) pairs of domains, the CATHSCOP-set [39]. The CATHSCOP-set contains lists of templates of consistently classified domains for 4859 target domains with pairwise maximal sequence similarities of 50%. As the CATHSCOP-set defines both similar and dissimilar pairs, we can perform tests on more distant similarity recognition. The CATHSCOP-

set contains 3919 and 2197 target domains, where superfamily (beyond the family similarities) or fold similarities (beyond the superfamily similarities) can be found, respectively.
As our tests are computationally costly due to the model building runs, we create a smaller, but similarly challenging test-set using the 'hard' targets for the simple sequence alignment method GOTOH. The *test-set* is defined as the set of the domains, where GOTOH scores an dissimilar domain (neither in the same SCOP fold nor in the same CATH architecture) better than a domain from its own SCOP family and CATH superfamily. We found 410 such targets, inducing 321.641 target-template pairs. In the test-set 338 and 181 targets can be used to perform superfamily and fold tests, respectively.
To investigate the properties of the consistent sets we checked the performances of different sequence alignment method on both the whole CATHSCOP-set (4891 targets 3.739.824 pairs (Table 4.1) and the test-set (410 targets, 321.641 pairs (Table 4.2)). We perform three *fold recognition tests*: 1.) *family level test*: all similarities are available 2.) *superfamily level test*: members of the target family are excluded 3.) *fold level test*: members of the target superfamily are excluded.

	maximum similarity in the CATHSCOP-set		
	family (4859)	superfamily (3919)	fold (2197)
GOTOH	84.07% (4085)	40.50% (1587)	23.81% (523)
123D	77.94% (3784)	30.75% (1205)	21.94% (482)
PPA	93.72% (4554)	73.92% (2897)	50.71% (1114)
HHALIGN	94.11% (4573)	76.24% (2988)	47.06% (1034)
VOROLIGN	**97.53% (4739)**	**90.25% (3537)**	**77.70% (1707)**

Table 4.1: **Fold recognition rates for the CATHSCOP-set.** We show for every test the number of targets involved in the given set in parenthesis. Recognizing the correct fold having all similarity levels (family column) available is an easy task for the current best alignment methods (PPA, HHALIGN), which achieve almost the performance of VOROLIGN. For the superfamily and fold level only more distant similarities are available. Thus, the recognition rates for sequence-based methods are much lower. On the fold level, even the best methods fail on the fold-recognition task on every second target. Also for this case, the use of structural neighborhoods exploited by VOROLIGN improves by more than 50% on the best sequence method (PPA) to over 75% fold recognition rate.

Due to its construction, GOTOH does not recognize the correct family for the targets in the test-set. This does not necessarily mean that GOTOH does not produce any meaningful alignments. In fact, as shown in Figure 4.3 a large number of acceptable models can be derived from the GOTOH alignments. Figure 4.3 also shows that the test-set allows for lots of improvements between the favored models of GOTOH ('+') and the best models possible ('best(all models)'). The optimum among the GOTOH alignments would be 'best(GOTOH)' and the actual improvement achieved by the VORESCORE rescoring is 'VORESCORE'.

	maximum similarity in the test-set		
	family (410)	superfamily (338)	fold (181)
GOTOH	22.68% (93)	27.51% (93)	16.57% (30)
123D	22.93% (94)	22.19% (75)	19.34% (35)
PPA	81.46% (334)	50.07% (176)	20.99% (38)
HHALIGN	89.76% (368)	66.57% (225)	37.02% (67)
VOROLIGN	**96.10% (394)**	**87.87% (297)**	**76.80% (139)**

Table 4.2: **Fold recognition rates on the test-set.** The test-set is a subset of the CATHSCOP-set with 410 query proteins. It is somewhat harder for the sequence methods, but the fold recognition performance of VOROLIGN on the test-set is about the same as on the comprehensive CATHSCOP-set.

The need for a smaller test-set results from the expensive tests we perform. We create models for each alignment computed with several sequence and structural methods resulting in about 2.000.000 models for the smaller test-set. On our machines this takes almost a year single-CPU time (modeller takes about 15 seconds pro alignment on average).
The defined test-set is representative and only slightly more challenging than the CATHSCOP-set and it allows for improvement via rescoring. We use the test-set for the evaluation in the following.

4.10 Model quality improvement over alignment methods

In this section we present the improvements achieved with VORESCORE when applied to various prediction methods. Due to space constraints, we restrict the results presented here to two alignment methods GOTOH and HHALIGN and to two rescoring methods, ROSETTA and PROQ. We also focus only on the performance of VORESCORE, which performs slightly better than the simpler VORPSI method. The assessment is based on a comprehensive evaluation of a large number of difficult targets (410). It involves building and scoring more than two million models.
Significant improvements can be observed for both simple (GOTOH) and the most sensitive alignment methods (HHALIGN) methods. The largest improvements of about 40% of the cases are found on the most difficult (fold) level, as should be expected for the sequence-based prediction methods. Surprisingly, the simple VORESCORE method outperforms both the involved ROSETTA (4 fold, the net improvement of ROSETTA on the fold level is -8% = (6.63 - 14.92) for GOTOH, +10% for HHALIGN, for VORESCORE +40% for both GOTOH and HHALIGN) and PROQ (by about 30%) scoring systems with respect to the net improvements of selecting better models (see Table 4.3).
Table 4.3 and Figure 4.4 show different aspects of the same data. The figures show that for all three levels VORESCORE achieves improvements over the whole TM-score range, i.e. for all

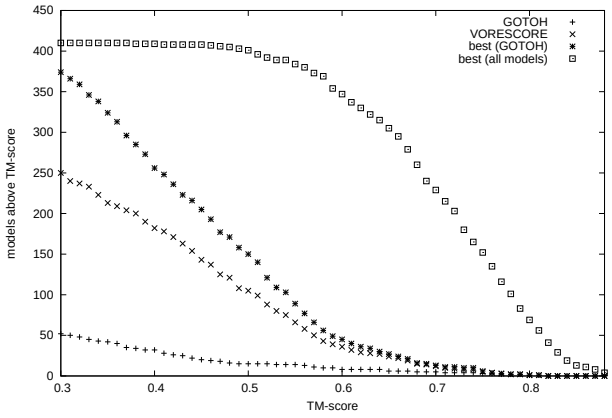


Figure 4.3: **Model quality of GOTOH alignments.** The figure shows the quality (measured with the TM-score) for the models build from GOTOH alignments for the family level recognition test. The y-axis shows the number of targets having a selected model with TM-score larger than the value on the x-axis. Due to the construction of the test-set the rates for the best model of GOTOH are rather low, but as shown by the 'best(GOTOH)' rates, there are a large number of high quality models which could be predicted via perfect rescoring of the GOTOH models. In fact, 'VORESCORE' can improve the quality of the selected models significantly. Of course, much better models beyond the GOTOH alignments are possible ('best(all models)').

levels of target difficulty. The more pronounced improvements are observed for the more difficult cases (TM-scores between 0.3 and 0.5). For high TM-scores the performances converge as due to the high similarities the alignment models often are already the best models possible and, thus, cannot be improved via rescoring.

Table 4.3 summarizes our results on the overall rescore success rates. VORESCORE selects worse models in only very few cases. For GOTOH it is able to find better models in 60% of the cases for family and 40% of the cases for superfamily and fold levels. The test-set is not easy: ROSETTA does much worse here, it selects worse models in 20-30% of the cases and better models only in rare cases 3-7% (best performance for the fold level). Results are interesting in that they show that improvements are possible even for very simple alignment methods. It appears that due to the low specificity of the method wrong models can be scored much better than good models and that good models nevertheless are actually be produced despite the overall low performance of the method. And VORESCORE is able to detect these models via a simple structure comparison of model and template structure.

For HHALIGN the situation is different as HHALIGN belongs to the most sensitive and most accurate sequence based methods. HHALIGN is supposed to produced better alignment scores and better alignments for the best scoring one - but also for the other candidates. So again the situation is not easy for a rescoring method. Again VORESCORE selects better models in many cases (10% on the family up to 50% on the fold level) and worse models in very few cases (5-6%). For every level the net improvement (better-worse) is significant (5%, 15%, and 40% for the different levels, respectively), with an overall improvement for 189 of the 410 targets as com-

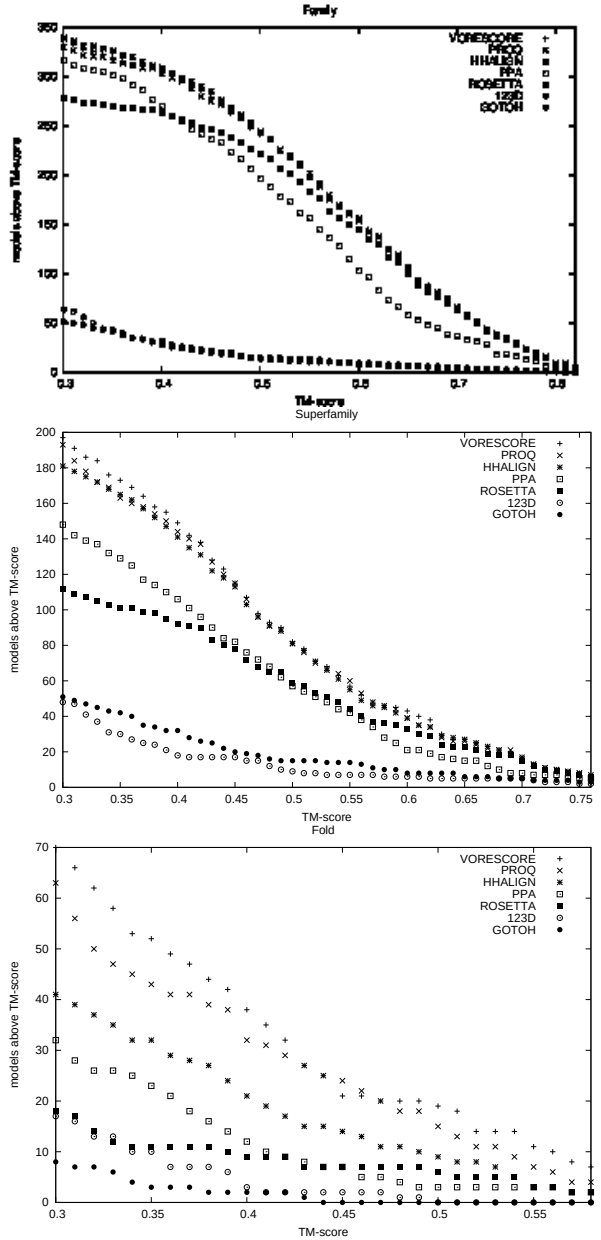


Figure 4.4: **Comparative recognition performances for alignment and rescoring methods** The three figures show the recognition rates for the family (top), superfamily (middle), and fold (bottom) levels for the relevant range of model qualities (TM-score above 0.3 up to convergence). The relative performance is similar for all levels and all model quality ranges: the pairwise alignments are clearly outperformed by profile alignment and these by rescoring methods.

pared to 41 targets, where the models are actually worse than the original ones. ROSETTA again performs very differently here: rescoring selects more worse than better models depending on the level resulting in a net improvement only on the fold level, resulting in an overall worse performance (127 worse, 114 better) as compared to the original HHALIGN. Thus, ROSETTA cannot be used as a rescoring method in e.g. the CASP competition. PROQ works quite well with similar performance as VORESCORE for HHALIGN on the family and superfamily levels, but 30% lower rates on the fold level (+30% = (40.91-10.23) as compared to +40% for VORESCORE). For the GOTOH alignments PROQ performs much worse as compared to VORESCORE on all three levels (39% vs 59%, 22% vs 38%, and 30% vs 41%).

(a) Rescore success rate for GOTOH

rescored model worse			
	family	superfamily	fold
ROSETTA	21.95% (90)	24.56% (83)	14.92% (27)
PROQ	9.02% (37)	11.83% (40)	6.08% (11)
VORESCORE	**3.17% (13)**	**5.03% (17)**	**2.76% (5)**
rescored model better			
	family	superfamily	fold
ROSETTA	4.88% (20)	4.14% (14)	6.63% (12)
PROQ	47.80% (196)	34.02% (115)	35.91% (65)
VORESCORE	**62.44% (256)**	**43.49% (147)**	**43.65% (79)**

(b) Rescore success rate for HHALIGN

rescored model worse			
	family	superfamily	fold
ROSETTA	11.22% (46)	17.46% (59)	20.44% (37)
PROQ	6.14% (25)	7.25% (24)	10.23% (18)
VORESCORE	**4.88% (20)**	**4.73% (16)**	**5.52% (10)**
rescored model better			
	family	superfamily	fold
ROSETTA	5.12% (21)	17.75% (60)	30.39% (55)
PROQ	8.35% (34)	**26.59% (88)**	40.91% (72)
VORESCORE	**9.02% (37)**	23.67% (80)	**44.20% (80)**

Table 4.3: **Rescore success rate for ROSETTA, PROQ and VORESCORE** on two alignment methods (a) GOTOH and (b) HHALIGN. If the alignment method predicts with high confidence, all three rescoring methods simply accept this production. Otherwise, the rescoring with ROSETTA, PROQ, and VORESCORE is based only on the models predicted by the respective alignment (GOTOH, HHALIGN) method. We term the rescoring neutral if the TM-score difference between the method first model and the rescored model is below 0.05. The net improvement of a method is given by the respective difference between the number of better and worse models.

The same data of Tables 4.3-4.4 is presented in more detail in Figure 4.4. In this figure the results are plotted against the TM-score of the models to show the dependency on the similarity of the target and template structure (as measured by the TM-score between the predicted model and the native structure). The results are shown in three plots for the family, superfamily and fold level, respectively.

For various methods, each figure shows the number of models produced by the method above a certain TM-score. Of course this number is highest for smaller TM-scores and is decreased by increasing the required TM-score. Overall, there are 410 targets (family level, 338 for the superfamily and 181 for the fold level) in the used test-set above the TM-score threshold of 0.3 (that is why the plot starts at TM-score 0.3).

Figure 4.4 shows seven plots for seven methods. The methods shown are the four alignment methods HHALIGN, PPA, 123D, and GOTOH and the three rescoring methods VORESCORE, PROQ, and ROSETTA applied to all models proposed by the alignment methods.

E.g. 'HHALIGN' shows the number of original HHALIGN ranked models (i.e. selecting the best scoring HHALIGN alignment) above the respective TM-score (as compared to the native structure of the target). Thus, this number approximates the HHALIGN performance in a CASP assessment for the 410 targets in our test-set. The other plots show the respective numbers for the other alignment and rescoring methods. The selected figures are just a small portion of the data we produced on the test-set.

The plots show several things. The same trend (with different amounts) and the same ranking is observed for all family, superfamily, and fold levels. First, the curves are clearly sorted for all TM-score levels, i.e. there is a consistent ranking for all TM-score values. The highest (i.e. best performing) curve are the two rescoring methods VORESCORE followed by PROQ. This is followed by HHALIGN, which on the family and superfamily level performs equally well but worse on the fold level. HHALIGN clearly outperforms PPA on all levels. the ROSETTA rescoring only works for the family and superfamily level where it is better than PPA for the easier targets but worse for the harder ones. Thus, ROSETTA appears to work for good models only, on the fold level its performance is almost down to the pairwise alignment methods 123D and GOTOH, which performs worst on all the levels on the hard targets of the test-set.

Overall, the effects are TM-score dependent in that the differences are largest for smaller TM-score thresholds and are converging for the higher TM-score thresholds as the models tend to be very similar (and thus the problem easier) for large TM-scores. The VORESCORE improvements are quite small on the family level, increase for the superfamily level to the quite drastic effects on the fold level (as hoped). On the fold level VORESCORE is able to produce almost twice as many good models based on the HHALIGN alignments (as compared to HHALIGN) for lower TM-scores (say < 0.5). E.g., on the fold level, GOTOH finds two, 123D three, ROSETTA nine, PPA 12, HHALIGN 21, PROQ 32, and VORESCORE 38 models with a TM-score of 0.4 or larger.

4.11 Model quality improvement using high quality alignments

The above results show the performance of the rescoring method in a realistic structure prediction setup similar to e.g. the CASP assessment. In this section, we evaluate the performance of the rescoring methods in a somewhat artificial setting where the best models are actually given and also subject to the rescoring. For this, we compute structural alignments with TM-align and PPM and add them to the models under investigation. Again VORESCORE is used to rescore the model structures against the native template structures and rank all the models. Table 4.4 and Figure 4.5 show the results.

In the Table 4.4 three numbers are given: how often a worse model is selected, how often a better model is selected, and as an intermediate neutral case, how often the original and the rescored models are of the same quality (TM-score difference less than 0.05). The first Table 4.4(a) shows these numbers for the family, superfamily, and fold levels for GOTOH alignments. VORESCORE selects worse models in very rare singular cases and better models in more than 80% of the cases (family level). The number and percentage of better models is somewhat smaller at 65% and 67% of the cases at the superfamily and fold levels. This reflects the fact that models for the family level are much better as compared to the more distant models on the fold level which appear to be more difficult also for rescoring.

For HHALIGN (Table 4.4(b)) the situation is similar: again rescoring performs extremely well and improves on the HHALIGN models in 15%, 38% and 68% of the cases for the respective levels (family, superfamily, fold). Of course, HHALIGN produces much better rankings and models in the first place, such that it is more difficult to select better models if at all (e.g. on the family level). So, on the family and superfamily level it is often not possible to improve on the HHALIGN alignment (in 80% and 55% of the cases). On the fold level, however, VORESCORE comes up with better models in 60% and with better or neutral models in more than 96% of the cases.

The Figure 4.5 is similar to Figure 4.4. The difference is that in the latter only predicted models, i.e. by sequence-based alignments, are used, whereas the former applies to the rescoring also to structure alignments from TM-align and PPM computed between the template and the native target structure. Moreover the figure contains the model qualities for the best and best predicted models. Thus, the 'best(all models)' (1) figure shows the number of the overall best possible model for the target at the given TM-score value - this is the theoretical optimum for the test-set given the template library independent from any alignment and rescoring method. The 'best(all predicted models)' (2) gives this figure for all actually proposed models, i.e. the performance of a perfect rescoring method. 'VORESCORE on all predicted models' (3) presents the actual performance achieved by VORESCORE on the proposed models and the difference to the former figure (2) indicates the shortcomings of the VORESCORE rescoring. 'VORESCORE on all models' (4) shows what the rescoring could do if the best (structure-based) models would be available, again the difference to the theoretical optimum (1) shows that the rescoring is not perfect. On the other hand, it shows that the rescoring could perform very well on good models (as the difference to the actual performance (3) is quite large). Finally, 'HHALIGN' (5) shows the HHALIGN performance in comparison. The difference of what is possible with the best alignment methods (5) to what can be done with rescoring (3) is large The difference to what

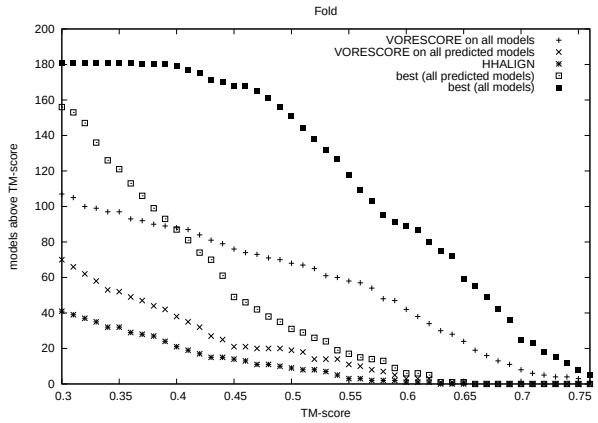


Figure 4.5: **Theoretical model quality for the test-set (fold level)** The Figure shows the number of models above a certain TM-score threshold for several methods as compared to the theoretical optimum. Here, structure alignment based models are also available for the rescoring. The number for the best possible template-based model, the best of all predicted models, and the best VORESCORE models (rescoring of *all* models) are compared to the actual VORESCORE (on *all predicted* models) and HHALIGN performance.

could be done if better models (4) and/or better rescoring (1)(2) would be available is even larger. As an example, at the TM-score threshold 0.4, HHALIGN finds 21 models, VORESCORE on all predicted models 38, VORESCORE on all models (also structural alignments) 88, whereas the best of all predicted models yields 87 and the best possible 179 models above TM-score 0.4. With respect to what is theoretically possible on the current test-set the results are somewhat disappointing: while on the family and superfamily levels VORESCORE as the best method can find about 75% and 60% of the best models [data not shown] this is not possible on the fold level, where VORESCORE only finds about 45% of the models and fails in well above 50% of the cases (see Figure 4.5). Also in comparison to the best structural models (TM-align and PPM models) the difference is very large, which indicates that sequence-based alignments fail to produce any reasonable model for many cases (at least at the fold level). Unfortunately, also on the superfamily level there is still a large margin [data not shown].
Last but not least, there is also quite some margin between the TM-align and PPM models and the best models possible, which indicates more intricate problems of scoring and assessing structures and structural models, again especially on the fold level as structures become more diverse.

4.12 Conclusions

Selecting the correct model is of great importance for homology-based modeling. The alignment score is not the best measure for identifying the best template and alignment. Even if the alignment scoring is far from perfect often good models are proposed and can be selected via appropriate rescoring, e.g. VORESCORE, which performs best among the methods evaluated

(a) Rescore success rate over GOTOH

model	family	superfamily	fold
worse	1.46% (6)	4.73% (16)	1.66% (3)
neutral	17.80% (73)	30.47% (103)	31.49% (57)
better	**80.73% (331)**	**64.79% (219)**	**66.85% (121)**

(b) Rescore success rate over HHALIGN

model	family	superfamily	fold
worse	5.85% (24)	5.62% (19)	3.87% (7)
neutral	**79.27% (325)**	**55.62% (188)**	27.62% (50)
better	14.88% (61)	38.76% (131)	**68.51 % (124)**

Table 4.4: **Rescore success rate of VORESCORE over GOTOH and HHALIGN on all models.** Rescoring is based on both GOTOH and HHALIGN predictions and, additionally, TM-align- and PPM-based models. The rescoring is *neutral* if the TM-score difference between model and rescored model is smaller than 0.05.

here.
From our results we draw the following conclusions:
1. We have shown that template-based protein structure prediction can be significantly improved by using structure information via rescoring of alignment-induced models. Many alignment methods often produce reasonable alignments which score worse than alignments with alternative templates. Rescoring the produced alignments using appropriate sequence-structure information helps to select those models which improve GDT- or TM-score, measures used in the CASP predictions to assess the model quality and to rank the predictions.
2. As might be clear beforehand scoring functions which do not directly take sequence-structure information into account such as ROSETTA do perform badly on the task of selecting the best alignment induced templates. This is especially the case in comparison with VORESCORE, which drastically outperforms these methods.
3. Maybe not very surprisingly, there is still much room for improvement of those methods. In particular, the method can only select the best of the proposed alignment-induced models. Thus, it is important that good models are actually proposed. If models from the best structural alignment methods are available, rescoring can improve by another large margin (on rescoring only actual sequence-based alignment models). Unfortunately, rescoring is not perfect even in these cases. It was and remains crucial to determine the best alignments with the best models.
It remains to be seen how far one can go with purely sequence-based methods, but rescoring is certainly one way in incorporate additional sequence-structure information into the structure prediction process. The proposed particularly simple rescoring system VORESCORE shows how to use structure comparison for structure prediction and the presented results demonstrate that conservation in structural neighborhoods is an important feature for sequence-structure relationships and a determining factor for protein structures.

Chapter 5

ccVorolign: Contact consistent neighborhood potential without gap scoring improves structural fold recognition

The ultimate goal of analysis of protein structure similarities is to find the sequence based features determining the fold of a protein. The results of the Vorolign method (chapter 3) have shown that the used neighborhood similarity potential is such a feature, and, therefore a good candidate to be used in the protein structure prediction (threading) setup. Moreover, both the Vorolign score and the Vorolign alignment should only be based on consistent contact mappings. This means, that only those contact mappings contribute to the score which are consistent, i.e. implied by the optimal alignment. This is not necessarily the case for the original Vorolign. On the other hand protein structures allow for some plasticity. The Vorolign scoring should account for that via a relaxed ϵ-contact consistency. This concept complicates the optimization but improves performance.

Vorolign cannot directly be applied in the threading setup, as it compares the amino-acid contacts of two known protein structures. In contrast, threading compares a known protein structure (the template) to a sequence with unknown structure (the target). The goal of threading is to find a sequence-structure alignment reflecting the structural similarity best among all possible alignments of the sequence with a template structure from a template library.

In order to use the Vorolign-potential for threading one has to **predict** the amino-acid contacts for the target. For this, there are two possible solutions: First, one can generate a large compendium of possible contact maps and utilize the potential to select the most promising one (analyzed in chapter 4). Second, optimize the target contact map based on the template contact map with respect to the Vorolign-similarity under the assumption that the target and the template have a similar structure.

As a result of the optimization one gets a sequence-structure alignment and the structural contacts from the known structure of the template can be transferred to the target sequence. This transfer of contacts in the threading scenario also implies that the similarity score incorporates

only similarities of neighborhoods on transferred (real to predicted) contacts. This is, due to the unique template residue to target residue mapping (the alignment) all mapping of contacting template residue pairs to contacting target residue pairs are clearly defined.
The final similarity score consists of terms scoring these mappings of real contacts to predicted contacts. Due to the bijective mapping of real and predicted contacts, the similarity score is contact-consistent.
Unfortunately, the original definition of the Vorolign-scoring does not guarantee contact-consistence: the similarity score of a resulting alignment may incorporate similarities of residue pairs not being part of the alignment. Actually, the lack of contact-consistency is a key feature for Vorolign's speed: without it the similarity of a given residue pair can be calculated independently of the final alignment and, thus, dynamic programming can be used to find the optimal Vorolign score.
In this chapter we will introduce a contact-consistent (cc-) Vorolign variant to make an an optimization possible in the threading context. Furthermore, we analyze the effects of enforcing contact-consistency on the protein structure comparison performance. The required contact-consistency increases the complexity of the problem, thus, we suggest a heuristic solution, i.e. we shift the heuristic from the model building step towards the optimization step. More precisely, in this chapter we present a generalized A*-based approach to derive optimal or heuristic solutions of threading-like problems, and we apply ccVorolign to the structure-based fold recognition problem.
The results we obtain indicate that the contact-consistent variant of Vorolign remains a good feature to recognize remote structural homologies, with the additional possibility to optimize the new potential in a threading setup. Moreover, while the introduced ccVorolign method does not use any (typically rather ad-hoc) gap scoring, it slightly outperforms the original Vorolign method. Using no gap penalties is an important advantage, as for threading there is no obvious theoretical justification for defining appropriate gap costs and their use during the optimization.
The second important finding of this chapter is that the Vorolign environmental potential has to account for a small amount of plasticity between the contact neighborhoods of corresponding target and template residues. This finding, which is in agreement with the observations from chapter 6 will likely have influence on the next steps towards using the potential in full threading scenarios.

5.1 Introduction

What makes proteins fold similar? Why do proteins belong to the same fold? What is the most likely fold for a given protein sequence? These questions are not easy to answer, not even for proteins for which two high-resolution protein structures are available. Even the best structure comparison methods (such as PPM see chapter 6) can determine the best fold only in about 83% of the cases. The best sequence based methods such as HHpred ([171]), i.e. methods predicting the correct fold for a given protein sequence based on sequence information alone, perform below 50% on this task. The fold recognition performance of Vorolign (chapter 3) is promising, as it uses only the sequence and secondary structure similarities for scoring (but of course implicitly also the contact information via the structural neighbors of residues). There is,

however, an important detail of the Vorolign scoring which speeds up the similarity calculation, but makes the optimization of the Vorolign score in the threading setup impossible: the possible inconsistency of the used neighborhood similarities to the resulting alignment (discussed in detail section 5.2).
Nevertheless, the results for Vorolign indicate that a fold recognition method optimizing the Vorolign-scoring schema with implied or transferred contacts can lead to much better fold recognition performance than the current best sequence based methods.
Vorescore (chapter 4) is a step in this direction: it also shows promising results, but it operates on a predefined set of alignments and is computationally expensive. Vorescore has been demonstrated to perform very well on selecting the right template alignments by rescoring the given alignment using the Vorolign score and the contacts implied by the template structure. Particularly good results for Vorescore are observed if the set of alignment to be rescored also contains high quality models (coming from structural alignments) again indicating that scoring the important conserved contacts is meaningful and beneficial.
Recently, Marks et al. [123] showed that evolutionary conserved residue couplings observed in multiple sequence alignments can be used to determine accurate 3D protein structures, again hinting to the importance of relatively few conserved contacts.
Can we design an algorithm directly optimizing the Vorolign score in the threading context? There are two issues to resolve: First, we have to create a contact-consistent scoring variant (discussed below) and adapt the gap costs for the threading-setup. Gap costs in general may help a lot, but they need fine tuning and are in general hard to interpret. An ideal similarity method would therefore somehow exploit the core regions and score them, without applying any gap costs (for different approaches to account only for core regions in threading see [179, 198, 199]).
The influence of gap costs on the performance of the Vorolign scoring is shown and discussed in figure 5.1.

The Goal: a gap free, consistent-contact based similarity measure

The main goal of this chapter is to design and evaluate an algorithm using the similarity scoring of Vorolign in a contact-consistent way, so that it can be applied in a threading scenario. Moreover, we will apply the algorithm in a "gap is free" manner, so that the scores will be independent of the length differences of the inputs.
The algorithm presented here is generally applicable for any threading method using some kind of contact scoring, i.e. it can be adapted to optimize alignments with respect to some contact potential.

5.2 Inconsistencies in the standard-Vorolign alignments

The general principle of Vorolign is to score similarities of amino acids according to their structural neighborhoods. As shown in equation 5.2 the similarity of a given residue pair does not depend on the resulting alignment. This in turn means that the score of the resulting alignment

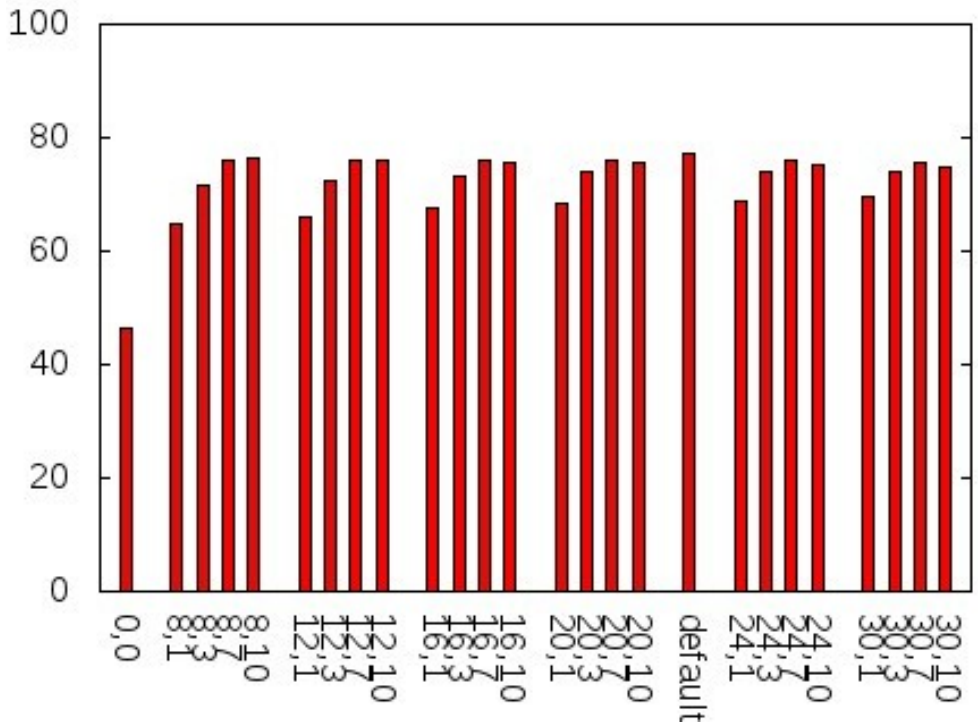


Figure 5.1: Gap cost dependency of Vorolign. The plot shows fold recognition performances of the standard Vorolign for different gap settings. On the x-axis, the gap cost settings (gap open, gap extend) settings are shown, on the y-axis the corresponding fold-recognition performance (only using the fold-level similarities) are shown. The gap costs have a great influence on the recognition rate, if no gap costs used, the performance drops about 30% as compared to the default (gap open 22.5, gap extend 6.5) settings.

(which is as usual the sum of the similarities of the matches + gap costs) may be based on similarities not contained in the alignment.

Figure 5.2 visualizes the inconsistencies for a given domain pair (1qk3C00 and 1qb7A00). In the upper part of the figure the used similarities of residue pairs are shown: the x-axis reflects the sequence and secondary structure assignment of the target protein, the y-axis the sequence and secondary structure assignment of the template protein. The residue pairs contained in the resulting Vorolign-score-optimal alignment are visualized as black dots. Additionally, all inconsistent residue pairs contributing to the alignment scores are visualized as red dots.

These inconsistent residue pairs are the result of the neighborhood similarity calculation done in low level dynamic programming. The score of an aligned residue pair (a black dot) is determined by the maximal similarity of the contacting residues (the contact neighborhood) of the residue pair. In fact, prefix and suffix contacts of the residue pair are analyzed separately, i.e. the contact neighborhood similarity of a residue pair is the sum of similarity of the prefix and suffix contacts. The low level dynamic programming calculates the maximal (prefix or suffix) neighborhood similarity.

The bottom part of figure 5.2 shows the low level dynamic matrix for the prefix contacts for a specific aligned position (156,169): the amino acid 156 from 1qk3C00 has 12 prefix contacts while the 169-th AA from 1qb7A00 has 8 prefix contacts.
The result of such a low-level dynamic programming is a score for the similarity of the (prefix or suffix) contact neighborhood, which is based on the similarities of the matches in the backtrace of the low level dynamic programming matrix.
This backtrace is highlighted with green and red colors in figure 5.2(b): the cells are colored green if the corresponding match is contained in the high-level alignment and red if it is not. The yellow cells highlight the matchings contained in the high-level alignment, which are not part of the low-level backtrace leading to the neighborhood similarity score in question. The red dots in the upper part of figure 5.2 originate from the red cells in the bottom part - and arise due to the independence of similarity scoring between the low and high level dynamic programming. This independence implies the prefix property of the Vorolign similarity, and allows for the double dynamic approach, but at the same time it is the cause for the red dots in 5.2(a).
The meaning of the inconsistencies is crucial for our goal: they imply that the similarity measured by the Vorolign neighborhood potential is not fully described by the resulting alignment, but also scores similarities between residues having no associated residue in the other structure. In the threading setup however we only have the contact information for one of the proteins, and by predicting a structure we basically copy the coordinates of the residues, and implicitly also the contacts of the template protein to the target sequence. As, however, the alignment not fully determines the Vorolign score, we can not optimize the alignment with respect to it.
If the score potential is fully determined by the alignment we can simultaneously optimize the target contact map along with the similarity score, as the contact map determines the similarity score. Figure 5.2 visualizes the resulting inconsistencies in the amino acid contact maps of the input proteins. The upper-right part of the figure shows the contact map for 1qk3C00, the lower part the contacts of 1qb7A00. The contacts not used for the similarity score are colored gray, the ones contained by the alignment (i.e. consistent) are colored green, while the inconsistent ones are colored red.
While, as explained above, the score schema of Vorolign does not enforce contact consistency, one would expect that in most cases the Vorolign-score optimal alignments are also contact-consistent. As both figure 5.2 and figure 5.3 indicate an unexpected amount of inconsistencies, we analyzed the inconsistencies of Vorolign alignments for a comprehensive set of similar proteins. We analyzed 130.000 alignments of similar proteins (within fold pairs from the CATH-SCOP set (see chapter 2), and calculated the percentages of inconsistent match-similarities used in the final similarity score. In addition to simply check for consistency/inconsistency we also calculate the distribution of different inconsistency levels. We quantify the inconsistency level with the help of ϵ-consistent mappings (see equation 5.4). The rationale behind inspecting inconsistent mappings is: for small ϵ values the inconsistent mappings lead to about the same structural correspondence than the alignment itself. This is of great importance for our goal: while the resulting contact map is not clearly defined for any inconsistency level above zero, for small inconsistency levels the threading optimization might lead to practical solutions, especially in fold recognition scenarios.
Figure 5.4 shows the inconsistency distributions of the non-empty alignments. Surprisingly,

we find that even in alignments implying high structural similarity (as indicated by high TM-score [207]) we observe a large percentage of inconsistency. For example, even if we allow $\epsilon = 1$ consistent mappings (green curve in the plot) and would tolerate <=20% mapped contacts with more inconsistency (i.e. x-axis value at 80%), only about 50% of the $\tilde{2}7.000$ high quality alignments fulfill these relatively relaxed criteria.
There are two possible explanations the contact neighborhoods are a very good way to describe and recognize structure similarities, the conservation of the contacts itself is much less important than expected or (ii) the observed inconsistencies are only an artifact of the Vorolign optimization and the sequence properties of the actually conserved contacts are also sufficient for recognition of structural similarities.
In this chapter we will introduce a method optimizing the Vorolign-scoring schema with explicitly enforced consistence.

5.3 Methods

First we will review the similarity measure used in the original Vorolign (VSM) method (chapter 3, [22]), then define the consistent Vorolign similarity (cVSM) measure, and the ϵ-consistent Vorolign similarity measure (ϵ-cVSM).
Then we will show that the ϵ-cVSM does not imply prefix optimality, optimality and, thus, dynamic programming can not be used to optimize the cVSM. We introduce a novel algorithm to optimize cVSM, which is also generally applicable for threading methods with amino-acid contact preferences.
In the following three sections we copy the similarity definitions used in Vorolign (chapter 3) in order to make the changes applied in the ccVorolign similarity definition clearer.

5.3.1 Similarity of Voronoi contacts

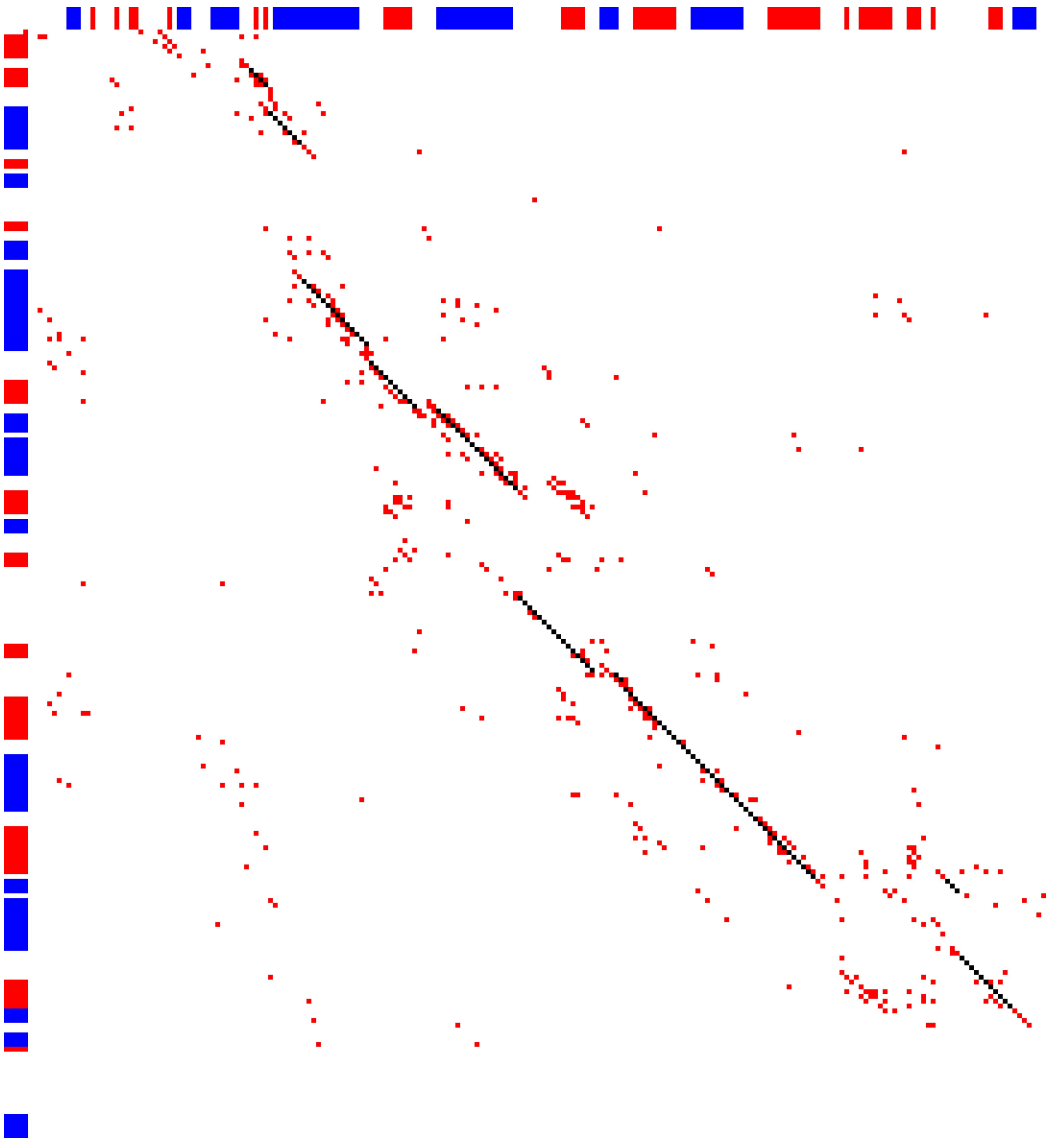
Given two proteins $X = x_1x_2...x_p$ and $Y = y_1y_2...y_q$ for which we want to calculate a structural alignment. The set of contact-neighbors of a residue x_i, as defined by the Voronoi tessellation, is denoted as $N(x_i) = \{x_{i_1}, x_{i_2}, ..., x_{i_n}\}$. In order to measure the similarity of two residues x_i and y_j we will calculate the similarity of their corresponding contact-neighbor sets $N(x_i)$ and $N(y_j)$.

Similarity of two nearest-neighbors

As a first step, we need to define the similarity of two residues x_{i_k} and y_{j_l} in the nearest-neighbor sets $N(x_i)$ and $N(y_i)$. This similarity will be scored by the weighted sum of two scores as given in the equation below:

$$Sim(x_{i_k}, y_{j_l}) = \omega_1 * AA(x_{i_k}, y_{j_l}) + \omega_2 * SSE(x_{i_k}, y_{j_l}). \tag{5.1}$$

In the equation, $AA(x_{i_k}, y_{j_l})$ corresponds to an amino acid exchange matrix score and $SSE(x_{i_k}, y_{j_l})$ scores the similarity of the corresponding secondary structure elements as defined by a secondary structure scoring matrix.



(a) Vorolign inconsistencies in the main dynamic programming matrix

	3:M:	4:A	5:S	6:K	11:Y	68:T	69:Y	127:V	128:L	129:I	154:S	155:M
mapped	**-1**	**-1**	**-1**	**-1**	**-1**	**63**	**64**	**140**	**141**	**142**	**167**	**168**
63:R	-3.04	-2.92	-2.8	-0.04	-2.86	**1.46**	-2.86	-5.52	-5.04	-5.94	-4.2	-4.44
64:Y	-0.34	-1.72	-2.74	-2.68	3.68	1.82	**3.68**	-1.92	-1.92	-1.8	-4.14	-1.74
67:M	8.24	3.74	1.76	2.0	3.98	-1.6	3.98	0.86	2.18	1.64	-2.44	4.04
140:V	1.52	0.5	-2.14	-2.8	0.26	-1.86	0.26	10.42	7.72	9.88	3.46	7.12
141:V	1.52	0.5	-2.14	-2.8	0.26	-1.86	0.26	10.42	7.72	9.88	3.46	7.12
142:L	3.02	-0.1	-2.56	-2.32	0.26	-2.46	0.26	7.84	10.3	8.68	3.04	8.62
167:V	1.52	0.5	-2.14	-2.8	0.26	-1.86	0.26	10.42	7.72	9.88	3.46	7.12
168:V	1.52	0.5	-2.14	-2.8	0.26	-1.86	0.26	10.42	7.72	9.88	3.46	7.12

(b) Vorolign inconsistencies in the second level dynamic matrix

Figure 5.2: Contact inconsistency in Vorolign scoring. For discussion see section 5.2. Black dots on (a) correspond to aligned residues in an optimal Vorolign-alignment, red dots correspond to inconsistent mappings used in the final optimal similarity score. Green cells on (b) highlight consistently used mappings between the high-level and low-level dynamic programming tracebacks for a specific position (156,169), red cells show inconsistent mappings used for the similarity score, yellow cells highlight consistent, but not used (as leading to suboptimal scores) mappings.

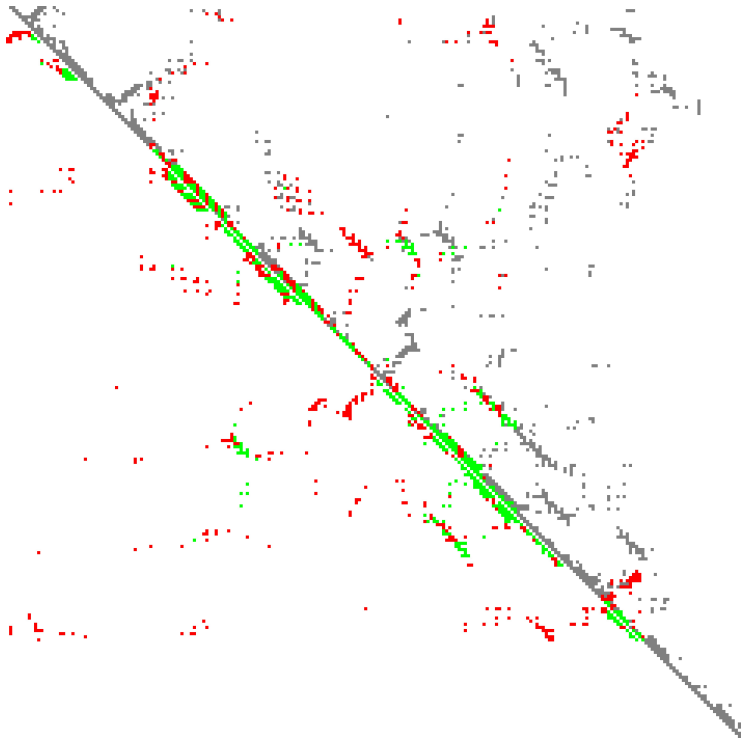
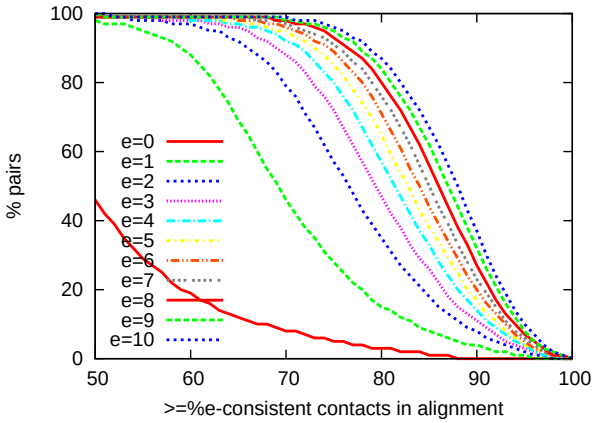
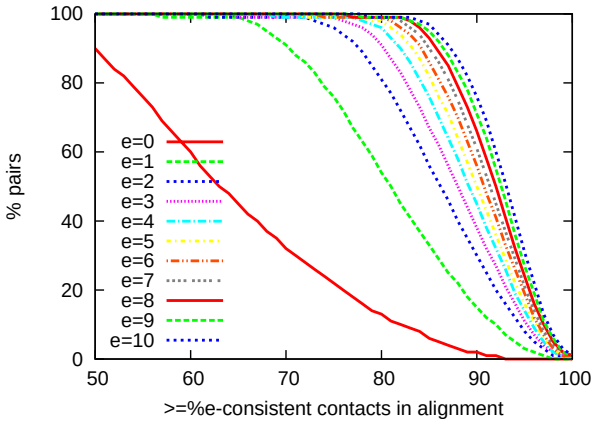


Figure 5.3: Contact inconsistency in Vorolign scoring visualized in the protein amino-acid contact maps. For discussion see section 5.2. Green dots highlight contacts consistent to the scoring of the final optimal alignment in the two structures (below and above the diagonal), red dots show inconsistently scored contacts.



(a) no TM-score threshold (120.000 alignments)



(b) TM-score $>=$ 0.5 (27.000 alignments)

Figure 5.4: Inconsistencies in the Vorolign similarity scoring. The upper figure shows the inconsistency distribution over all within-fold pairs with maximal 50% sequence identity from the CATHSCOP set. Surprisingly, the larger percentage of the used contacts are not in agreement with the optimal Vorolign alignment (red line), and even if smaller errors (+-1 green line, +-2 blue line) are tolerated more than 60% of all aligned pairs use less than 80% consistent contacts in the similarity score. This is even true for high-quality alignments shown in the lower pictures, where only alignments exceeding 0.5 TM-scores are shown. The plot indicates that structurally clearer alignments are in general more consistent, but still, only about 50% of the high quality alignments have more than 80% +-2 consistently used contacts in the similarity score (blue line).

Similarity of nearest-neighbor sets

Having defined the similarity of two nearest-neighbor residues, we estimate the similarity of the two nearest-neighbor sets $N(x_i)$ and $N(y_i)$. For this we need a matching of the residues in the $N(x_i)$ and $N(y_j)$ sets. Once we have found such a correspondence, the final score (similarity) of the two nearest-neighbor environments is simply given by the sum of the similarities of the

residues matched onto each other minus a penalty score for each unmatched residue.
The position of x_i (or y_j, respectively) is taken into account by using two independent sets of contacts: one containing all contacts in the prefix sequence of the residue under consideration (x_i or y_i) in the protein sequence, the other set containing all residues found right of the residue in the sequence.
The similarity of two nearest-neighbor sets can be computed using dynamic programming with respect to the similarity function of two nearest-neighbor residues given in equation 5.1 and an additional penalty for unmatched residues p_u. This corresponds to calculating a global alignment with linear gap costs of the residues in the two nearest-neighbor sets. The entries $S(k,l)$ in the dynamic programming matrix S are filled using the following equation:

$$S(k,l) = max \begin{cases} S(k-1,l-1) + Sim(x_{i_k}, y_{j_l}) \\ S(k-1,l) - p_u \\ S(k,j-l) - p_u \end{cases} \tag{5.2}$$

Consistent similarity score of nearest-neighbor sets

Assuming a set of consistent mappings, i.e. a set CM of two-tuples (x_{i_k}, y_{j_l}) we change the equation 5.2 to:

$$S(k,l) = max \begin{cases} S(k-1,l-1) + \begin{cases} -\infty \text{ if } (x_{i_k}, y_{j_l}) \notin CM \\ Sim(x_{i_k}, y_{j_l}) \text{ else} \end{cases} \\ S(k-1,l) - p_u \\ S(k,j-l) - p_u \end{cases} \tag{5.3}$$

ϵ-consistent mappings

Given an alignment A with a set of aligned positions ((x_{i_k}, y_{j_l})) we define the set of ϵ-consistent mappings of the alignment as follows:

$$CM = \{(x_i, y_j) | \exists (x_k, y_l) \in A, |x_k - x_i| <= \epsilon \wedge |y_l - y_j| <= \epsilon\} \tag{5.4}$$

Thus, if the alignment is represented as a path in a two dimensional matrix with the sizes of the aligned sequences, the ϵ-consistent mappings are the ϵ "hull" around the alignment-path. For small ϵ values all paths within this hull would result in a very similar alignment to A, and would imply a similar topological correspondence of aligned regions between target and template.
As a similarity scoring scheme using ϵ-consistent mappings will imply also ϵ-consistent contact mappings we term similarity scoring schemes using ϵ-consistent mappings for its underlying alignment ϵ**-contact-consistent similarity score** or ϵ**-cc similarity score**.

5.3.2 The complexity of the optimization problem of the ϵ-contact-consistent similarity score

As the ϵ-contact-consistent similarity score of an alignment depends on the alignment itself, the prefix property does not hold. Moreover, this scoring schema corresponds to a threading scenario with amino acid interaction preferences. This is the case as with a selection of a given (x_k, y_l) mapping for the final alignment we explicitly affect the similarity scoring of many mappings, namely the similarity scores of all inconsistent mappings to (x_k, y_l). In [114] Lathrop gives a proof that this property makes threading NP-complete, motivating a heuristic approach for the solution for the optimization of the ϵ-cc Vorolign similarity score.

5.3.3 General approach: A*

While the optimization problem is NP-complete, it is generally not unsolvable for practical problems, or at least suboptimal might be found, which are sufficient in a real-world scenario. However, it is important to look for a (sub-)optimal solution in an efficient way. The algorithm of choice for such scenarios is A* [162], providing a general way to avoid evaluation of large irrelevant parts of the search space.
The general principle of A* is to use partial solutions (i.e. in our case define some constraints about the final alignment, without fully defining it), and estimate an upper bound of the similarity score (termed h-score) achievable from the current constraint (this upper bound estimation is called *admissible heuristic*). Applying the admissible heuristic guarantees that every partial solution overestimates the score achievable from it and so we can always expand the partial solution with the highest h-score (by applying more and more constraints to the alignment) without missing the optimal solution. The efficiency of the A* algorithm depends on the upper-bound estimation - the closer to the real score the faster A* will find the optimal solution. As it is reasonable to expand the most promising partial solution A* uses a priority queue sorted by the h-scores. Initially, it contains only the empty partial solution (no constraints), then in every step the highest scoring (most promising) solution is expanded, i.e. new partial solutions based on the highest scoring one are placed into the queue. As the h-score drops or remains equal with every expansion, and the algorithm expands always the highest h-scoring partial solution, the estimated score gets closer to the real score in every step. In addition, whenever a full solution is encountered one can update the maximal real score seen, and partial solutions below or equal this score can be pruned from the queue. As the h-score will be calculated for every partial solution and upon expansion for every child, it is important to choose a higher bound estimation which can be calculated efficiently.

One has to keep in mind that partial solutions can be achieved on different expansion paths from the empty partial solution, eventually leading to many unnecessarily multiple calculated partial solutions. In order to avoid this one has to keep track about the already visited partial solutions, but as there are exponentially many partial solutions, this can lead to memory problems.

The key points of A* are (i) the definition of the partial solutions, (ii) the admissible heuristic

(h-score calculation) (iii) keeping track of the already visited partial solutions. In the following, we will introduce a generally applicable solution for threading methods using contact-scoring.

5.3.4 The partial solutions

As the goal of our optimization problem is to calculate an alignment between protein sequences, we choose a corresponding constraint in the partial solution definition. For this we recall the idea of the fix-point alignments, i.e. in every step we will fix some (i, j) mapping. While we use this idea, we apply a minor modification: instead of fixing an (i, j) mapping we only apply the constraint, that the alignment path in the main dynamic programming matrix will cross the x-axis at (i, j). The main difference is that we do not require that i is mapped to j, it may be also unaligned. However as we fix the alignment path we explicitly define invalid mappings: namely all mappings (sequentially) inconsistent to (i, j), i.e. $\{(k, l)|k < i \wedge l > j$ and $(k, l)|k > i \wedge l < j\}$. This minor modification has a great influence on the implementation: as we do not require to select a match for a given i we can apply the constraints in a very efficient way, similar to Hirschberg's linear space algorithm [89] adapted to dynamic programming: for the target protein with length n we will first create partial solutions where we calculate the h-scores for all possible j-s for $i = n/2$ then for the maximal scoring we expand it so that calculating the h-scores for the relevant j at $i = n/4$, then for $i = 3n/4$ and so on. More formally: we create a balanced binary tree with n nodes and iterate over the nodes in a breadth-first manner. Doing so we can unfold a full solution from the empty solution in quadratic time as shown for the Hirschberg algorithm. Figure 5.5 shows an example of the successively applied constraints and motivates the quadratic time needed in total.
Of course, in our case the maximal achievable similarity score changes with every additional constraint. The efficient calculation of these scores is discussed in 5.3.5.
The second important effect of the minor modification for the constraint used in the partial solution over the fixed point alignment and the balanced binary tree-style selection of the target side constraint points is that it automatically solves the problem of tracking of all visited partial solutions. With these we defined an ordering in the search space, in addition, no partial solution can be achieved via two different paths - the number of constraints directly defines the next x-coordinate to be constrained.

5.3.5 The core of A*: the admissible heuristic

First we note that due to it's definition the standard Vorolign similarity is a $max(n, m)$-cc similarity score for two proteins of lengths n and m. In addition it is clear that for any $\kappa >= \epsilon$ a κ-cc similarity score is an admissible heuristic for the ϵ-consistent heuristic score. This is the case, as $\epsilon - CM \subseteq \kappa - CM$ in equation 5.3, and so all similarities on the high-level matrix will be greater or equal for κ than for ϵ.
We will use the same principle in the admissible heuristic: using the constraints from the partial solutions we define CM as the set of not-yet inconsistent mappings. This is given by the areas between the constrained points in the partial solution and an ϵ-wide area around these. Figure 5.5 highlights the corresponding CM for every step as gray areas. With more and more aligned

points the size of the CM set decreases rapidly, leading to similar rapid drop of the corresponding h-score.

Efficient calculation of the heuristic score

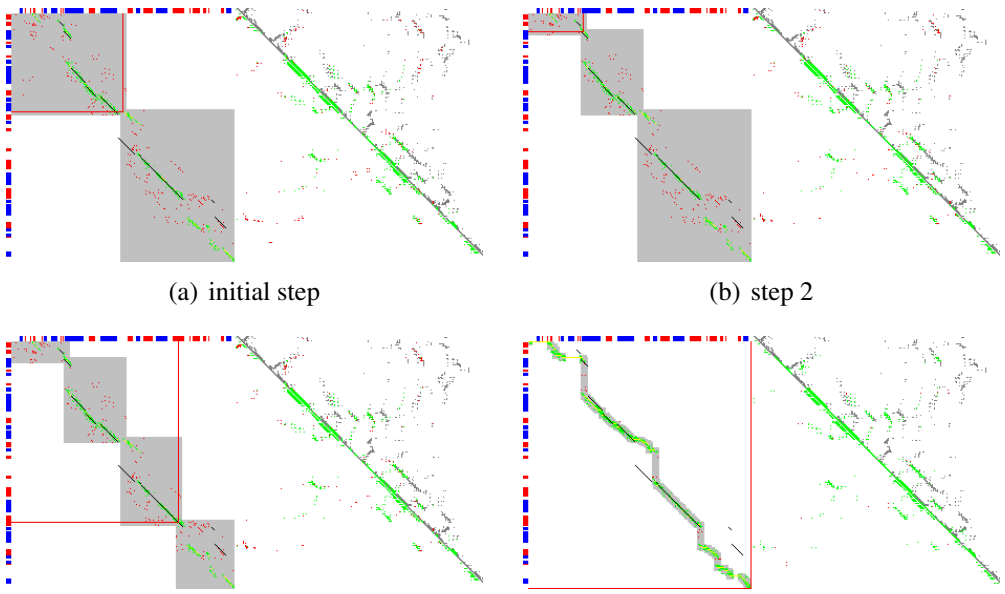
While the ccVorolign similarity induces an NP-hard problem, in practice one can use the admissible heuristic to efficiently retrieve relevant solutions. Therefore, the efficiency of the h-score calculation is crucial. Our goal is to keep Vorolign's quadratic runtime in the ccVorolign method for a set of constraints leading from an empty partial solution to a no more extendable alignment.

In each expansion-step we have to calculate the h-scores for all partial solutions resulting from the current constraint. This is: given a set of already defined set of constraints CS of size cs we calculate the best y constraint for the $cs + 1$-th constraint point for the x position of the target sequence (the sorting of the x points is explained in 5.3.4). The range of the possible y mappings is given by the constraints in CS - initially this is the length of the template protein. To calculate the h-score for all child partial solution we use the current CM and perform a forward and a backward alignment (as in Hirschberg's algorithm). With this we can calculate an upper bond of the yet achievable similarity score for all children in quadratic time. However, as we have to check for every contact in the low-level matrix used for the calculation of the similarities of the neighbor-sets (see section 5.3.1) whether it is contained in CM, the quadratic time requirement holds only if we can implement this check in constant time.
The trivial solution would be to calculate a full $n \times m$ boolean matrix and set the fields corresponding to the elements in CM to true. Unfortunately, this would mean that on the path from an empty solution to a full solution we had to update this matrix n times and, therefore would result in a cubic runtime.
To overcome this, we store a list of relevant regions from the current constraints as rectangular areas, where all coordinates within the region are elements of CM. As two subsequent constraints define such an area, and we have two default constraints (the upper left corner, the lower right corner in the main dynamic programming matrix) there are always $cs + 1$ such areas. As they are ordered by the corresponding sequence positions we can assign an area index a_i to each of them.
We then create two indexes: we store for each i and for each j the sorted list of the a_i-s containing the corresponding amino acid in the target or template, respectively. In addition, we store for each j the minimum and maximum i relevant for j. To create these indexes we only have to walk over the fields within the relevant areas once for the target and once for the template positions. The cost for the calculation of these indexes for a given set of constraints is $O(n + m)$. The path from an empty partial solution to a full partial solution is maximally n long (as we extend the constraint set CS for all target position at most once). So the full cost of calculating these indexes from an empty partial solution to a full one is at most $O((n + m) \times n)$.
To check now if a given mapping (x_k, y_l) is contained in CM we can first check if x_k is within the relevant range of y_l and if so, if they share an a_i. For this second check we can simultaneously walk over the two sorted lists of a_i. The computation cost of this operation depends only on the length of the a_i indexes. This in turn depends on ϵ: as the regions depend on the constraints,

and these are naturally ordered, a given mapping (x_i, y_i) can only be covered by $2\epsilon + 1$ regions. So by using these indexes we can implement the membership-test for CM in $|\epsilon|$ time and so the calculation of the heuristic score of all children costs $O(\epsilon nm)$. As ϵ is a small constant, the calculation of the indexes is of quadratic cost, and by using Hirschberg's strategy, the calculation of a full solution from the empty partial solution remains quadratic.

5.3.6 A* in a target versus template library scenario

For alignment methods using dynamic programming the standard way to find the the most similar protein for a query protein in a library of templates, is to calculate all query-template alignments and take the one with the highest score. While this is also an option for an A* optimization, a different approach offers an additional advantage: if partial solutions for all templates are added to the priority queue for a given target at the same time, one can omit optimization of templates having low h-scores, and spend more time on the calculation on relevant templates. In addition, if only the best prediction (fold recognition) is searched, one can stop optimizing if all entries in the queue correspond to the same template.



(a) initial step (b) step 2

(c) step 3 (d) final step

Figure 5.5: the three initial steps and the last step of a using the admissible heuristic in a greedy way. All figures consists of two parts - left side is the alignment part, the right side shows the contact usage in the aligned proteins (upper right triangle:target, lower left triangle: template). Color schema for the alignment side: black dots show the optimal standard Vorolign alignment, the gray areas show the currently not-yet-contact-inconsistent mappings, the green dots show the optimal alignment within the gray areas.

5.4 Results

5.4.1 Testset

The test-set is derived from the CATHSCOP consensus set (Chapter 2) and allows for an unbiased, large-scale evaluation of the performance in different scenarios and different levels of difficulty, e.g. family, superfamily, and fold recognition. In total, it contains more than 1.87 Mio pairs of domains so that each pair has less than 50% sequence identity.

5.4.2 Evaluation strategy

For the evaluation we perform fold-recognition in three scenarios: (i) in the Family scenario (4.859 targets) all templates are included in the library, (ii) in the Superfamily scenario (3.919 targets) only templates not in the family (i.e. same superfamily, same fold, or different fold) are included in the library, (iii) in the Fold scenario (2.917 targets) only templates in the same fold or a different fold (i.e. only remote or no homology at all) are included.
All ccVorolign predictions were made on AMD Opteron processors with 2600Mhz and 2Mb cache and had a maximal memory usage of 4Gb. For all runs we used the target versus template library scenario (see section 5.3.6), and applied two time thresholds: we stopped the queue expansion after 30 minutes (if the optimal solution was not found earlier) and evaluated the partial solutions using a greedy algorithm on the heuristic score for additional 60 minutes. In order to check the dependence of the correct secondary structure annotation used in the similarity score, we also performed a run with ϵ=2 using PSIPRED [99] annotations instead of DSSP[102] annotations.

5.4.3 Performance of ccVorolign

Table 5.1 shows the results of the evaluation. The evaluated methods are PPM (chapter 6), HHalign ([171]) Vorolign (chapter 3) with the standard settings, Vorolign without any gap costs (ng-Vorolign). The performance drop for HHalign for the fold level evaluation shows the complexity of the benchmark set for sequence based approaches, whereas the performance of the purely structure based method PPM yields an upper limit of detectable similarities. The difference between the performance of Vorolign and the ng-Vorolign shows the increasing importance of the gap costs in the similarity scoring - presumably the needed compensation to the wrongly detected neighborhood for non-structural homologous pairs due to inconsistent contact mappings.
Interestingly, flexibility is needed in scoring the contact conservation: if contact-consistency is strictly enforced (0-ccVorolign) the performance drops, mostly due the large number of resulting empty alignments (percentage given as "0-score"). If however, a small amount of contact-plasticity is allowed, (1,2,3)-ccVorolign the contact consistent variants can even outperform the standard Vorolign settings (2-ccVorolign) - although not using any gap costs at all. Apparently, this relaxed contact consistency contributes more to the recognition than the difference

between predicted and real secondary structure: even if using predicted secondary structures (by PSIPRED) the 2-cc-psipred-Vorolign slightly outperforms the standard Vorolign.

Method	**Family (4859)**	**Superfamily (3919)**	**Fold (2197)**
PPM	98.02%	93.01%	83.75%
HHalign	94.11%	76.24%	47.06%
Vorolign	97.53%	90.25%	77.70%
ng-Vorolign	89.79%	66.11%	42.24%
3-ccVorolign	97.88%	92.14%	79.79%
2-ccVorolign	**97.98%**	**92.37%**	**80.52%**
1-ccVorolign	97.14%	88.82%	73.92%
0-ccVorolign	86.79% (9.47% 0-score)	47.72% (37.66% 0-score)	23.58% (52.44% 0-score)
2-ccpsipred-Vorolign	**97.65%**	**90.94%**	**78.83%**

Table 5.1: Evaluation of fold recognition rates in the presence of the correct family (Family column), the presence of the correct superfamily, but not the correct family (Superfamily column) as well as in the absence of the correct family and superfamily but in the presence of the correct fold (Fold column). ng-Vorolign is the standard Vorolign variant with gap cost settings (0,0) (gap open, gap extend), the rows for (1,2,3)-ccVorolign refer to ϵ contact-consistent Vorolign runs with $\epsilon = 1, 2, 3$ respectively, 2-cc-psipred-Vorolign is the $\epsilon = 2$ contact-consistent Vorolign variant using PSIPRED secondary structure predictions instead of the DSSP annotation, which is always slightly worse than 2-ccVorolign but still in all cases better than the original Vorolign.

5.5 Conclusions

The performance of the structural alignment method Vorolign (chapter 3) and the predicted structure model rescoring method VORESCORE (chapter 4) suggest the usability of the contact neighborhood potential for threading. However, in the Vorolign setup the contact-consistency is not guaranteed, but we require it for the threading. In this chapter we investigated the effect of the enforced contact consistency in the Vorolign scoring. The main findings and achievements of this chapters are:

1. Enforcing contact consistency improves the fold recognition performance, but only if we account for a certain amount of plasticity. Despite the fact that the contact graphs between members of the same folds are approximately conserved (they are mappable with $\epsilon = 2$ consistency, see definition 5.4), most remote homologies cannot be detected, if strict contact consistency is enforced. This suggest that the contacts rather than exactly are conserved more in a "block-type" manner.

2. We presented the general algorithm to address the protein threading problem. Our results suggest that the sequence and (predicted) secondary structure information alone is sufficient to recognize fold-level similarities, without using any gap scores to penalize larger inserts or deletions. Although the usability of the presented method for real threading remains to be shown, we think that our findings may help to understand the rules of folding. Moreover, if combined with the current achievements in predicting contacts from evolutionary sequence variations [124], the number of protein sequences with predictable fold might be significantly increased.

Chapter 6

Protein structure alignment considering phenotypic plasitcity

In the previous chapters we focused on the similarity between protein structures based on their contacts and contact neighborhoods. While this appear to be a very informative and useful to find the driving force of protein structure similarity we also need methods to reliably identify all remote structure similarities as judged from the full set of atom coordinates. In this chapter we define a general concept describing structure similarities via virtual structure transformations from one structure to the other - similar to the edit distance to measure distances between sequences typically represented as appropriate alignments. Thus, the transformations used and scored by PPM constitute structural edit operations with associated costs. The corresponding algorithm mimicks the "manual-workflow" of visually assessing the similarity of between two protein structures.

The PPM method was also instrumental to assess possibly introduced fold changes by alternative splicing events in chapter 7. We also assisted in a more detailed alignment accuracy based evaluation of the PPM method, which was published in Rocha at al. [161] Beside the use of PPM for remote structure similarity and possible fold change transition detection, it can also be used to test sequence-driven similarities (such as the contact environment potential) in the structural context: Instead of using the purely-structure based Topological-Edges defined in the chapter, we can use more sequence-driven similarities to calculate the "structural mutation" between block pairs. For example, one can use the contact-consistent neighborhood potential variant from ccVorolign (5 chapter) to evaluate how one can select the correct block pairs from the set of possible block pairings. If doing so over the set of similar blocks we can concentrate on selecting the correct topological edges, thereby assuming that the step defining the blocks can be solved. This evaluation of the power of the contact potential derived from chapter 5 in the PPM framework is planned for future work.

PPM is again joint work with Fabian Birzele. The chapter is based on a paper published in 2008 ([38]) and presented at the ECCB'08 in Cagliary. My contribution on PPM is the main idea to exploit protein plasticity, the algorithm to find the best matches, the efficient implementation of the involved method as well as the benchmarking and visualization of the results.

6.1 Introduction

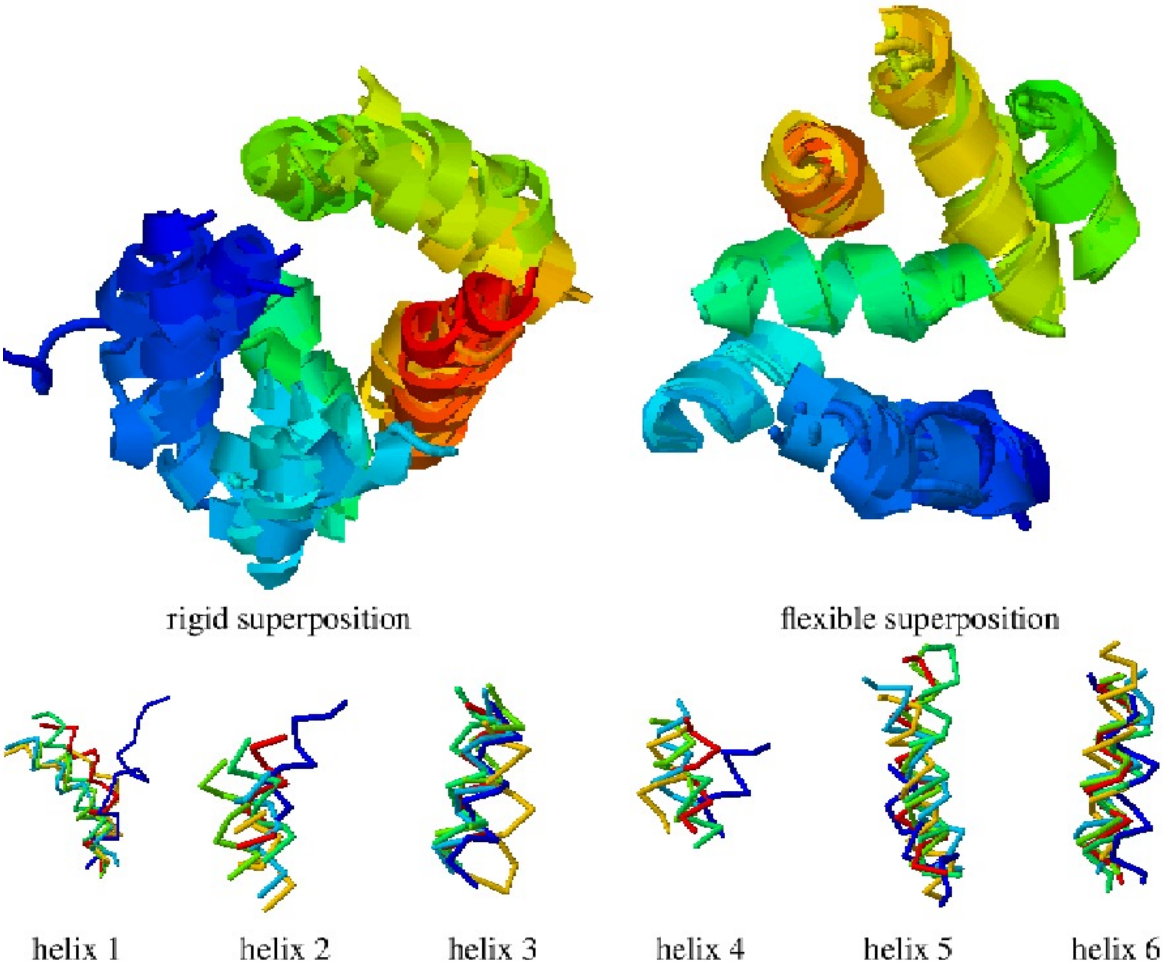


Figure 6.1: On the top left figure we show the overall multiple rigid superposition of members of the pheromone binding family (SCOP: a.39.2.1) implied by a manually curated alignment (guided by disulphide bridges and secondary structure elements). While the overall topology remains the same for all members of the family the exact position and orientation of the helices shows high plasticity (top left). As can be seen from a flexible multiple superposition (top-right) the helices are very well conserved. Structural variations in the different helices are shown for each of the six helices separately in the lower part of the figure.

The comparison of protein structures and the detection of their similarities and differences is one of the major steps in understanding the interplay of protein sequence and protein structure evolution.
To date, the gold standard for protein structure classification are the (manually) curated databases SCOP [7] and CATH [77]. Despite large efforts over many years, these 'gold standards' are known to disagree to some extent [81] which indicates that defining and identifying structural

similarity is difficult and still controversial. For benchmarking purposes of new measures and methods it is therefore suggestive to use consensus sets defined from both classifications. SCOP classifies structures into classes, folds, superfamilies and families in a hierarchic manner. But with the growing number of protein structures in the PDB [19] and the arrival of structural genomics projects [75] there has been a growing need to model expert knowledge in automatic methods for protein structure classification and protein structure analysis. Currently (February 2008), out of the 45.000 protein structures in the PDB, more than 10.000 are not classified in the most recent SCOP version 1.73.

A crucial step in structure analysis is the computation of a structural alignment of two or more protein structures and, in contrast to sequence alignment methods, structure alignment methods aim directly at optimizing the structural similarity of the input proteins.

Sequence alignments are commonly computed using amino acid exchange matrices as well as gap penalties. Those parameters explicitly model the process of sequence evolution by mutations as well as insertions and deletions. It is difficult to propose an equivalent model for protein structure evolution which could be used for structure alignment.

Therefore, to date, all structure alignment methods capture protein structure similarity not by the evolutionary cost of mutating ("morphing") one sequence and structural conformation into the other but by a similarity of the three-dimensional objects. The methods differ in their score for protein structure similarity as well as in the algorithms used to optimize this score. Different similarity measures may be differently well suited for specific applications.

Many methods measure structural similarity based on a rigid body superposition of the input structures (given the alignment) and for that purpose several scores have been proposed. Among them are CE [167] (optimizing the RMSD) and TM-Align [206] optimizing the TM-Score [207]. Though being a valid approach for many applications, rigid superposition neglects the fact that proteins are dynamic and flexible objects that allow for (significant) structural variation even within families and much more so within superfamilies and folds. Therefore, in recent years, also methods for flexible protein structure alignment have been published that account for large scale structural flexibilities and can align protein structures even in cases that show large conformational changes. A well known method is FATCAT [200].

While the two approaches discussed so far explicitly use the coordinates of the protein structure in the scoring and optimization process, other methods work on different representations like contact matrices [91], graphs [85] or Voronoi cells [22]. Those methods also allow for a certain amount of intrinsic flexibility of the protein structures as long as the contact networks are conserved.

The method proposed here is based on the observation that, despite conformational changes and large scale flexibilities, protein structure families exhibit a high flexibility on a smaller scale which we call *phenotypic plasticity* (see Figure 6.1 for an example). Phenotypic plasticity of a protein structure comprises the changes in the actual 3D-structure, which are to be expected for the given protein sequence (the "genotype") or within a given genotype population (i.e. a group of proteins related according to a family, superfamily or fold relationship). Further, phenotypic plasticity is grounded in evolutionary events that happen on the sequence level, namely mutations, insertions and deletions.

In contrast to existing approaches, our method, in the following called PPM (phenotypic plas-

ticity method), explicitly tries to model, score and optimize the changes which naturally occur in protein structure evolution. Under the hypothesis that two protein structures are similar, we assume that they share locally similar substructures (not necessarily restricted to secondary structure elements). For those, single amino acid exchanges (if they occurred) have not led to major structure rearrangements. Nevertheless, some point mutations as well as most insertions and deletions require certain topological rearrangements (movement of elements) on the structure level. In PPM we now try to measure the structural cost of mutating (or "morphing") one structure into the other one. During this process, we score the similarity of locally similar substructures as well as the conservation of their topological arrangement. This approach explicitly allows to map corresponding structural elements of the two structures onto each other and to observe their phenotypic plasticity in a population. This feature of PPM will be especially helpful to study the interplay of sequence and structure evolution in the future.
In the following we will first describe our model of protein structure similarity based on phenotypic plasticity and an algorithm to optimize this similarity score. We will then evaluate and discuss in detail the performance of PPM in recognizing similar structures on two benchmark sets and will compare PPM against other well known structure classification methods. We will conclude with an outlook on future work and extensions of PPM.

6.2 Methods

PPM scores the similarity of the topology of locally similar substructures (core blocks) and we determine possible core blocks and mappings in a first step. Having identified core blocks we need some measure to quantify their topological similarities/differences (i.e. the implied structural mutations). We address this problem in three steps: (i) we introduce a measure quantifying topological difference of core block pairs in two structures, (ii) we introduce the graph (the PPM-graph) of topological changes of pairs of core blocks, and define their topological difference on this graph, (iii) we introduce an algorithm that identifies the consistent subgraph with maximal score in the PPM-graph. The PPM-method is summarized in Figure 6.2. Please note that all similarity functions used are only presented in the form of examples in the text for space reasons. Details on those functions are available in the supplementary material on http://www.bio.ifi.lmu.de/PPM.

6.2.1 Phenotypic Plasticity Measure

Determining possible core blocks

In Figure 6.2a-b) we show the result of the core block detection and the subsequent filter step. Given two protein structures p_1 and p_2, let $B(p_1)$ and $B(p_2)$ be sets of blocks (subsequences of the respective proteins) which are structurally similar to at least one block in the other protein.

Definition (block): A subpart (subsequence) b_1 in protein p_1 is called a *block* if it has at least length T_l (set to 6 in our experiments) and is rigidly superposable to some subsequence b_2 in

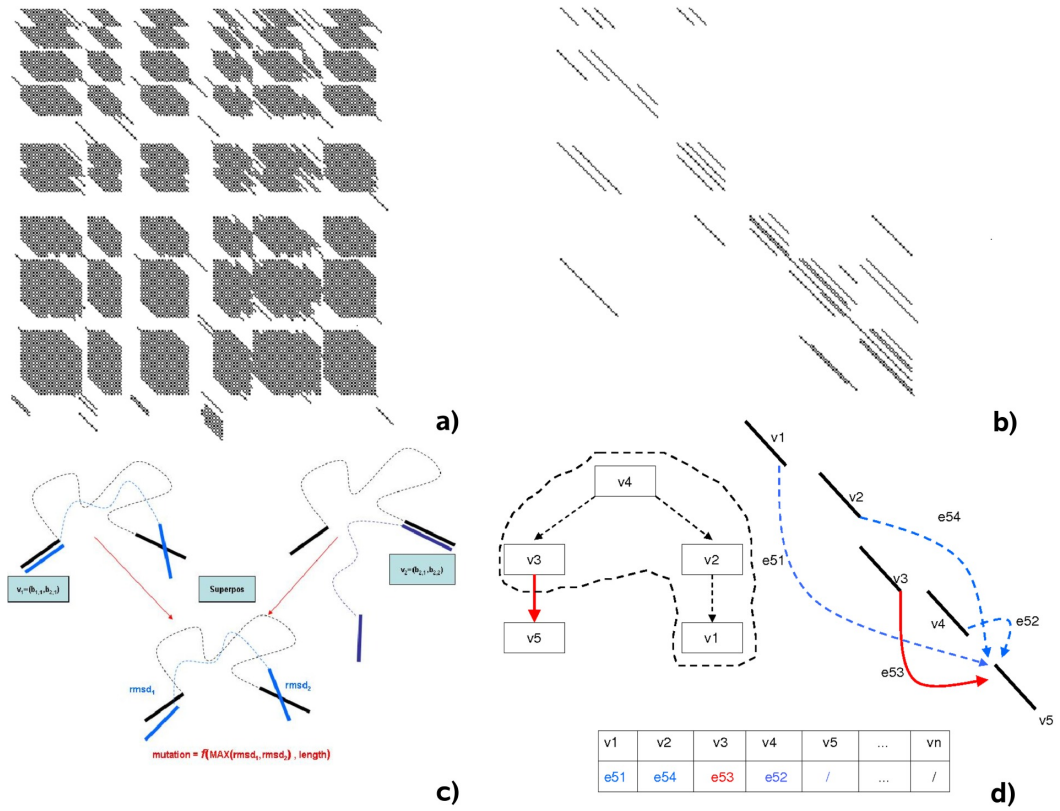


Figure 6.2: Shows an overview on the steps of PPM. **a)** In a first step, mappings of locally similar, ungapped blocks are identified in the two structures. **b)** The number of possible core block mappings is reduced due to the criterion described in section 6.2.1. **c)** Visualizes the computation of the similarity of pairs of core block mappings as described in section 6.2.1. **d)** Shows the computation of the final alignment. The cost of adding block b_5 to the alignment consisting of blocks $b_1, ..., b_4$. The edges $e_{51}, ..., e_{54}$ are sorted with respect to their weights, i.e. e_{51} represents the edge with the smallest, e_{54} the edge with the highest cost. Depending on which edge is chosen each step, adding a block becomes more or less expensive, allowing to adjust the degree of phenotypic plasticity in the score. We used level 3 throughout our experiments.

protein p_2, such that all associated distances in the superposition are below T_d Å.

The threshold T_d is chosen in a length dependent manner and represents a tradeoff between the length of the block pair and the the maximally allowed pairwise distance of the residues. The longer the fragment, the higher the variability allowed. In our experiments T_d is for example set to 1.24, 2.37, 2.91, 3.3 Å, for blocks of length 6, 12, 18, 24 respectively. Please note that using pairwise distances instead of e.g. RMSD results in highly similar local substructures. Each block mapping represents a local, gapless alignment with locally similar structures. In the following steps we will search for the maximal set of consistent blocks inducing similar topologies in the input protein structures.
In order to reduce the complexity of subsequent steps and to avoid shift errors in the core block alignment, we filter the core block mappings found using global topology information. Therefore, we calculate the superposition of the two proteins induced by a core block mapping and compute the maximum number of residue pairs under 6 Å. When applying this step to every residue of the query protein we can drop many incorrect core block mappings. An example for this approach is shown in Figure 6.3, while the effect of the filter step is shown in Figure 6.2b. The resulting set of possible core blocks is denoted by $V = \{v_j = (b_1, b_2) | b_i \in B(p_i)\} = B(p_1) \times B(p_2)$
The similarity of a mapping of two core blocks $sim(b_1, b_2) = sim(v_j)$ is determined by a empirical function representing a tradeoff between the length of the pairs and the corresponding RMSD. The score for a pair of blocks lies between 0 and length with higher scores indicating better pairs. Examples of the score are: length = 10, RMSD = 1.2 leads to a score of 10.0, length = 10, RMSD = 2.0 to a score of 5.46 and length = 20, RMSD = 2.0 results in a score of 17.05.

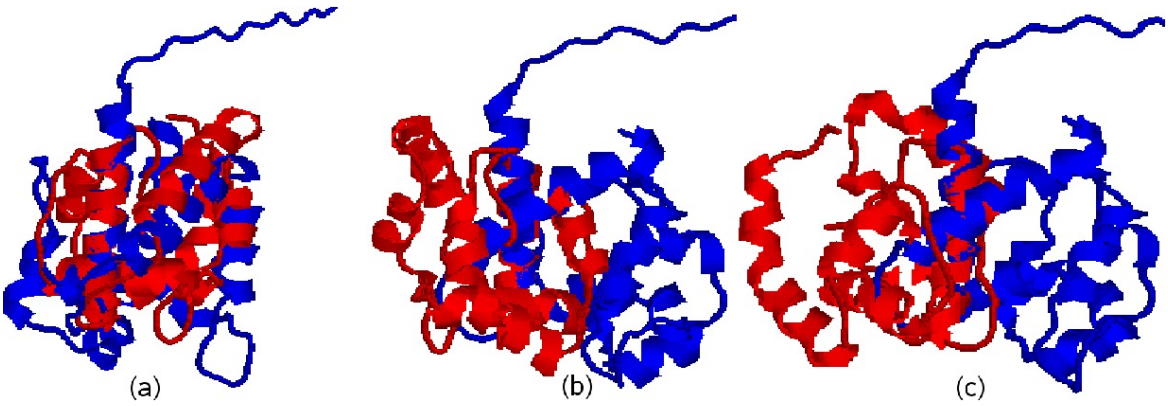


Figure 6.3: The effect of shift errors on global superposition. Domains d1qwva_ and d1r5ra_ (members of the same SCOP family) are locally superposed based on the mapping of the terminal helices. The first figure (a) aligns the helix correctly, and clearly all elements of the structures (the other 5 helices) are topologically close in space to their corresponding counterparts. The second (b) and third (c) figure align the terminal helix with a shift error of one and two, respectively. The effect is that based on the local match the other five helices are distant in space.

Similarity of topologies implied by pairs of core block mappings

Having identified possible mappings of core blocks, we can compare pairs of such mappings using their topological relationships. In general we represent the topology of a protein structure by the 3D-geometry of the blocks. In more detail, having assigned $b_{1,1}$ from protein one to $b_{2,1}$ in protein two and $b_{1,2}$ to $b_{2,2}$ we calculate the implied structural mutation assuming that the topology of $(b_{1,1}, b_{1,2})$ changed into the topology $(b_{2,1}, b_{2,2})$. For this we compute the superposition of the two mappings ($(b_{1,1}, b_{1,2})$ and $(b_{2,1}, b_{2,2})$) and calculate their RMSD in this superposition. The maximal value of the two RMSDs obtained ($b_{1,1}$ on $b_{2,1}$ and $b_{1,2}$ on $b_{2,2}$) is used as a "structural cost" of a mutation i.e. a cost of the topological change implied by the joint mapping of the blocks. The maximal RMSD is used in order to avoid effects of large blocks on the mutation cost. A schematic example of the procedure is shown in figure 6.2c).
Please note that while the similarity of a core block pair is measured by an RMSD-based score, the final cost of the structural mutations observed is measured on the PPM graph (see section 6.2.2). The PPM graph explicitly tries to score the topological costs of a structural mutation.
The function to convert the lengths of the two mappings and their maximal RMSD into a cost function for structural mutations is empirical and represents again a tradeoff between length and allowed RMSD, i.e. a maximal RMSD of 4.3Å across 33 residues leads to costs of -2.79 while an RMSD of 5.1 on 65 residues is less expensive (due to it's larger size) and costs -2.09.

6.2.2 The PPM-graph: similarity of the topologies

To measure the similarity of topologies we define the PPM-graph. The set of core blocks mappings (see section 6.2.1) defines the nodes V of the graph. In the first step, the graph is fully connected and the edges are weighted by the pairwise similarity of pairs of core blocks as defined in section 6.2.1. The schematic idea of the scoring process is shown in figure 6.2d).

Definition (PPM-graph G_{PPM}): Given two proteins p_1 and p_2, we define the phenotypic plasticity (PP) graph as a labeled graph $G_{PPM}(p_1, p_2) = (V, E)$ as follows: $V = B(p_1) \times B(p_2)$ and $E = V \times V$. Please note that nodes v_j represent aligned block pairs, i.e. mappings of two core blocks in an alignment of proteins one and two.

To reduce the number of edges we only use the "coordinating" set of edges in our experiment, i.e. we only use edges if the distance of the core block pairs is less than 30Å in both proteins .

Definition (Linear Phenotypic Plasticity Measure of an alignment LPPM(A): Having defined the similarity of nodes and edges, let A be an alignment of protein structures p_1 and p_2, $A = (v_1, e_1, ..., v_{n-1}, e_{n-1}, v_n)$, i.e. an alignment is simply a subset of core blocks connected by topological edges which form a spanning tree of the core blocks in the alignment. The linear LPMM(A) connects the blocks in a linear fashion from the N- to the C-terminus and is defined as: LPPM(A)= $\sum_{i=1}^{n} sim(v_i) + \sum_{j=1}^{n-1} weight(e_j)$

The LPPM(A) score defined above always links a block to the alignment by the edge connecting

it to the previous block in the sequence. Therefore, the edge weight does not take the overall topology of the protein into account. To overcome that problem we define a variant of score as follows (see also Figure 6.2d). For each aligned block node v_i, the graph G_{PPM} contains $n-1$ incident edges and each could be used to connect v_i to the alignment.

Consider the sorted list of edges weights $e_{i1}, e_{i2}, ..., e_{in}$, with e_{i1} having the smallest weight. A tree T where the nodes are the n block pairs connected by $n-1$ edges represents a structural alignment/superposition of p_1 and p_2, $T = (v_1, ..., v_n; e_1, ..., e_{n-1})$. The level allows to control the plasticity of the alignment: level 1 connects a node v_i via the edge of smallest weight e_{i1} to the tree, level k uses the edge e_{ik} to score connecting node v_i.

The higher the level the more edges are penalized in the alignment. Therefore level 1 allows for the largest plasticity of the protein structures, since adding a node (block pairs) to the alignment can always be done at the lowest possible cost. Using level $n-1$ enforces the largest rigidity onto the protein structures since always using the worst edge for connecting the nodes (the pair with the largest inconsistency) increases the cost of the alignment and therefore decreases the score.

Definition (Level k Tree Phenotypic Plasticity Measure of an alignment T_kPPM(A)): Let the tree T represent an alignment of protein structures p_1 and p_2, $T = (v_1, ..., v_n; e_1, ..., e_{n-1})$, where $e_j = e_{jk}$ (the k-th best edge connecting the nodes) T_kPPM(A)= $\sum_{i=1}^{n} sim(v_i) + \sum_{j=1}^{n-1} weight(e_j)$

Definition (Phenotypic Plasticity Measure of an alignment PPM(A)): In our experiments the level has been set to 3 which accounts for a reasonable tradeoff between flexibility and rigidity allowed for the input structures. Therefore we define PPM(A)= $T_3PPM(A)$

6.2.3 Optimizing and normalizing the PPM-score

Having calculated all possible core blocks and all topological edges, we need to identify the consistent subgraph with maximal score in the PPM-graph. This subgraph represents the structural alignment of the two input structures. We solve this problem by using the A* algorithm with an appropriate heuristic function. The A* algorithm searches for the path in the graph maximizing the similarity of the two structures. A* guarantees to find the optimal solution (if the corresponding heuristic function is "admissible" which means that the function always overestimates the best possible similarity resulting from a path through the graph). Since it is possible to find such a heuristic function, the algorithm is able to find the optimal solution to our problem in reasonable time (less than a second on average). Please note that the algorithm does not incorporate sequence gap costs at any time in the optimization procedure. However, for measuring the overall structural similarity it is useful to penalize non-aligned (na) residues and to normalize the score. In our experiment we simply penalize every non-aligned residue with a constant value of -0.1 and the similarity of a query and target protein is normalized by the length of the query protein. The normalized PPM-score is defined as:

Definition (Normalized Phenotypic Plasticity Measure of two proteins)

$$PPM_{norm}(P_1, P_2) = \frac{argmax_A(T_3 PPM(A(P_1, P_2))) - 0.1 * na}{length(P_1)}$$

6.3 Results and discussion

In the following we will evaluate the performance of PPM according to its capability to recognize structural similarities of a query protein in a template database of protein structures. Therefore, we will first show the results of PPM on the Vorolign testset [22] and will then perform a larger, more difficult and more detailed evaluation on a comprehensive set of domains from the most recent SCOP version 1.73. PPM is evaluated against the two state-of-the-art structure alignment methods Vorolign and TM-Align. While TM-Align represents a method optimizing a rigid-body similarity criterion, Vorolign represents a method which uses a different, more flexible and more sequence-based representation of the protein structure, namely Voronoi neighborhoods, and allows to compute alignments even in cases where large flexibilities exist.

6.3.1 Family recognition on the Vorolign set

In order to test the ability of PPM to detect the correct protein family in a database of representative protein structures, we first use the family recognition set used by [22]. This set contains 979 non-trivial test proteins from the ASTRAL compendium that are contained in version 1.67 (February 2005) but not in the previous version 1.65 (December 2003). Simulating the Vorolign test scenario, those proteins were scanned against the ASTRAL25 database (version 1.65).
The performance of all methods is measured as the fraction of proteins that can be assigned to their correct family, superfamily or fold according to the SCOP classification. Only the hit with the best score as computed by the respective method is taken into account for the evaluation.
PPM is compared against the structure alignment methods Vorolign, CE and TM-Align as well as a sequence-based method, profile-profile alignment [185]. The results are shown in Table 6.1 (the performances of further, sequence-based methods are additionally discussed in [22]).
In this test scenario, all structure-based methods clearly outperform sequence-based methods like profile-profile alignment. Further, PPM outperforms the second best method (Vorolign) by about 2% on the family and the superfamily level and performs similar to Vorolign on the fold level. The test shows that PPM is able to predict the SCOP classification for a protein structure with a very high accuracy and can improve the family prediction rate on this set by another 2% compared to Vorolign.

6.3.2 Large scale evaluation

To evaluate the performance of PPM, Vorolign and TM-Align (the three top-scoring methods on the Vorolign set) on a larger and also more recent test set, we used a comprehensive set of 4933 domains from SCOP 1.73. The domains are a representative subset of all domains classified in SCOP 1.73 which are similarly defined in CATH. We require an overlap of the

Method	Family	Superfamily	Fold
PPM	**88.3%**	**94.5%**	97.5%
Vorolign	86.4%	92.4%	**97.7%**
TM-align	83.8%	92.6%	95.9%
CE	84.6%	91.9%	94.1%
PPA	80.8%	87.5%	91.9%

Table 6.1: Evaluation of the family, superfamily and fold recognition rate. Methods are evaluated taking the best hit with respect to the method specific score into account. For Vorolign, TM-align, PPM and PPA those are the respective alignment scores and for CE the best z-Score is used. If CE returns more than one alignment for a target-template pair the best z-Score of a pair is used. Further results for sequence-based methods can be found in [22]

Method	Family	Superfamily	Fold
PPM	78.6% (3817)	91.4% (4442)	**98.0%** (4763)
Vorolign	**79.9%** (3881)	**91.9%** (4464)	97.5% (4739)
TM-align	78.5% (3815)	91.4% (4429)	97.4% (4735)

Table 6.2: Evaluation of the family, superfamily and fold recognition rate of PPM, TM-Align and Vorolign on a large set of almost 5000 SCOP domains. The absolute number of correct predictions accounting for the rates are shown in brackets.

Method	Family	Superfamily	Fold
PPM	1.07% (53)	**3.62%** (122)	**13.61%** (357)
Vorolign	**0.83%** (41)	6.29% (212)	18.68% (490)
TM-align	1.56% (77)	6.02% (203)	19.79% (519)

Table 6.3: Evaluation of the errors made by a method in the presence of the correct family (Family column), the presence of the correct superfamily, but not the correct family (Superfamily column) as well as in the absence of the correct family and superfamily but in the presence of the correct fold (Fold column). An error is defined as a domain from a different fold scoring better than the query's own family, superfamily or fold members, respectively.

residues in the two domain definitions for a protein of more than 80%. All domains share a sequence identity of less than 50%. Every domain is used as query and compared against all domains in the set which belong to the same family, superfamily, fold (positive examples) and the same class but not the same fold (negative examples). Additionally, we require a pair of domains to be classified consistently in the CATH database, i.e. pairs from the same family / superfamily in SCOP need to be classified into the same 'homologous superfamily' in CATH and pairs belonging to the same SCOP fold into the same CATH topology. This approach assures a high quality of the structural similarity relationships defined by the gold standards. Templates from other classes are not taken into account to limit the number of pairs to be evaluated. In total, we compute more than 3.6 million pairwise structure comparisons (the set can be obtained from

http://www.bio.ifi.lmu.de/PPM).
When evaluating the same scenario as above, namely the family, superfamily and fold recognition rate of the methods according to the first hit (shown in Table 6.2) the methods turn out to perform very similar. Vorolign slightly outperforms PPM and TM-Align on family and superfamily level while PPM performs better than the other two methods on fold level. The somewhat different results of the two benchmarks in Table 6.1 and 6.2 on family and superfamily as compared to fold level may be due to a smaller sequence identity allowed for pairs in the Vorolign set ($<30\%$) compared to the larger set (50% at maximum).
In contrast to Table 6.2, Table 6.3 evaluates the performance of the methods in a different and more difficult scenario. Here, the performance is evaluated in a situation where the family of the target has been removed from the benchmark set to test a methods ability to detect similarities in the absence of clear sequence similarity. This setup becomes for example important for folds which contain only one family and need to be extended by a new family ("protein structure threading").
Therefore, all hits from the query domain's own fold are regarded as correct hits while hits from the same class but from a different fold are regarded to be errors. The table shows the number of query proteins where templates from wrong folds score better than templates from the own family, superfamily or fold, respectively. The error rates obtained allow for a detailed analysis of the performance of a method in the absence of the true family or superfamily since in those cases wrong folds would be predicted. In this more detailed and also more difficult test scenario (compared to pure family recognition) PPM significantly improves on both, Vorolign and TM-Align, on the superfamily and fold level, significantly reducing the number of errors. On the family level, it performs slightly worse than Vorolign. On this level, Vorolign's more sequence based scoring function seems to be an advantage.

6.3.3 Detailed analysis of examples

We further analyzed the errors made by PPM on the family level (cases where family members are not identified correctly) on both sets. The results for all errors made can be visually inspected in 3D at http://www.bio.ifi.lmu.de/PPM. The errors may be partitioned into three different classes. For some cases, the true family and the other fold found by PPM can not be distinguished on the structure level. One such example is shown in Figure 6.4. The figure allows for a fast and detailed analysis of the differences of the two hits found. When analyzing the similarities (i.e. the aligned parts) and differences (i.e. the unaligned parts) of the proteins in the two superpositions one can see that both families share the same common core (though being classified in different folds) of three helices and differ in some other, mostly unstructured, parts. It will be difficult for any purely structure-based method to distinguish those two folds (if they turn out to be truly different).
Further, we find examples where PPM overestimates the cost of phenotypic plasticity in cases where an unexpectedly large plasticity can be observed. Those errors may be solved in future versions by a better understanding of the costs of structural mutations and, maybe, even family or fold specific cost functions.

The third class of errors comprises cases, where e.g. length differences of template and query lead to problems in the score normalization and in the ranking process.
To exemplify the ability of PPM to compute high quality alignments of core regions and to deal with phenotypic plasticity, we show the example of a.39.2.1 which has already been briefly discussed in Figure 6.1. This family consists of six helices and is stabilized by six cysteine bridges. The family shows a high level of phenotypic plasticity which makes it hard for a rigid body method to align the helices correctly, i.e. without shifts and without adding gaps to the helices. Also, TM-Align aligns a number of single residues which are not related but are close to each other in space in a rigid body superposition. In contrast, such errors are avoided by PPM and the core regions (the six helices) are correctly identified and aligned (w.r.t. to the cysteine bridges) by our PPM method as shown in Figure 6.5.

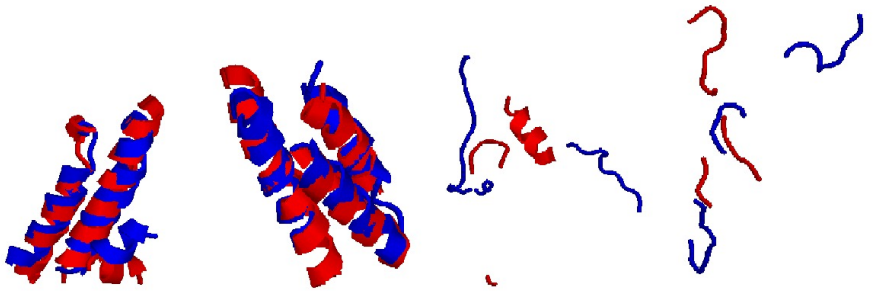


Figure 6.4: Example of a PPM misclassification where a member from a different fold scores better than the protein's own family member. The query and both template proteins consist of three helices in the same orientation and topology. Additionally, the member of the same family has a longer helix which slightly changes its orientation. The hit from the other fold on the other hand has no further secondary structure elements. Finally, the flexible superposition shows that the parts aligned by PPM fit very well.

6.4 Conclusion

We presented a novel idea and a new method to address the protein structure alignment and protein structure classification problem in the presence of phenotypic plasticity. In contrast to existing approaches PPM explicitly tries to model the evolutionary cost of the mutations, insertions and deletions that occurred on the structure level during the transformation of one structure to another as implied by the sequence changes. Thereby, it accounts for the natural variance observed in protein structure families, a phenomenon we call phenotypic plasticity. This plasticity can lead to problems and artifacts in the alignment process (e.g. shifting helices or random matches in space as shown in Figure 6.5) and those are avoided by PPM since it only aligns

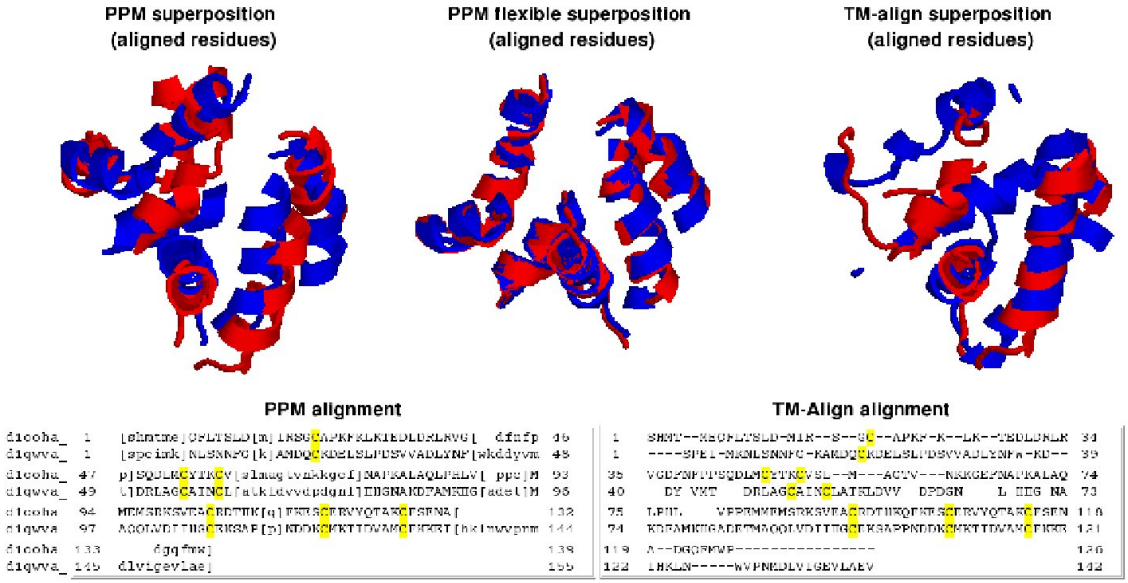


Figure 6.5: An example for the alignments and superpositions computed by PPM and TM-Align of two proteins ($d1ooha\_$, $d1qwva\_$) from family a.39.2.1 (also shown in Figure 6.1). The family has six conserved cysteine bridges which should be conserved in a good alignment of the two members of the family. Due to phenotypic plasticity of the corresponding helices, TM-Align produces shifts and breaks in the alignment which results in three (out of six) non-conserved cysteine bridges (right part). In contrast, all six bridges are conserved in the PPM alignment (left part) and the corresponding, locally similar helices are identified correctly (as shown in the flexible superposition based on the PPM alignment in the middle of the figure.

highly similar local substructures in a first step and then optimizes the alignment of those core blocks using the A* algorithm. The process takes the overall topology of the structures into account and avoids overestimating the structural similarity due to a too large level of flexibility.

The performance of PPM has been evaluated on two comprehensive benchmark sets. On the Vorolign-set, PPM outperforms all other methods in recognizing the correct family and superfamily of a query protein. On the more recent benchmark set of almost 5.000 SCOP domains and the corresponding 3.6 million structure pairs PPM performs comparable to Vorolign and TM-Align in the family / superfamily and fold recognition scenario but significantly outperforms the other methods in a different test setup where the ability to recognize at least a member from the correct fold in the absence of family or superfamily members is evaluated.

Future development of PPM will be into different directions. First, the method is completely modular according to which scoring functions are used to score the similarity of core blocks as well as their topologies. Our final goal of finding an evolutionary measure for structural mutations similar to the idea behind accepted mutations in matrices like PAM [44] is not reached yet. So far, the scoring functions are purely structure-based and due to our current lack of knowledge on the true costs of mutations and insertions/deletions on the structure level can only be optimized and parametrized empirically. It is an ongoing and iterative process to further refine those

parameters to capture the real costs of mutations and insertions/deletions on the structure level, maybe even in a family or fold specific manner.
Second, we would like to extend the currently purely structure-based scoring functions used in the method to a combination of sequence and structure scores to account for both criteria and further improve on the quality of the protein alignments. High quality alignments with respect to sequence and structure form the basis for a detailed analysis of the interplay of protein sequence and structure evolution.
Third, conserved pairwise core blocks identified by PPM will further be used to identify conserved cores of protein structure families with interesting applications in protein structure prediction and sequence-structure alignment (e.g. threading). For this task, PPM will be extended in order to compute multiple structure alignments or to combine sets of pairwise alignments to a multiple alignment.
Finally, the model of structure evolution underlying PPM may be an interesting starting point towards a novel definition of protein structure similarity and protein structure flexibility within and between groups of structures, a feature that will also help to detect similarities across different folds which may be explored in nature for example by alternative splicing [21] or during structure evolution.

Chapter 7

Effects of alternative splicing on the protein structures

This chapter investigates the structural effects of splicing events. Splicing events are introduced and mapped onto protein structures to identify evolutionary conserved splicing events with the help of multiple alignments of transcript sequences. Splicing events are defined as non-trivial if they affect structural elements in the core of a protein fold and if they are not evolutionary conserved. These non-trivial splicing events are comprehensively analyzed and using PPM structurally similar folds are searched in the template database for the spliced proteins. Interestingly, in several cases the spliced structures appear to be structurally similar to a protein with a different fold than the native (un-spliced) protein.

As discussed in the previous chapters, most of the knowledge about the protein structures and the successes in protein structure prediction are the consequence of the observation that the primary sequence of the protein defines the 3D structure. However, in higher eukaryotes there is a process operating on the transcript sequences of genes resulting in different transcripts transcribed from the same gene and, thus, sharing high sequence similarity: the alternative splicing. Alternative splicing is thought to be one of the major sources for functional diversity in higher eukaryotes. Interestingly, when mapping splicing events onto protein structures, about half of the events affect structured and even highly conserved regions i.e. are non-trivial on the structure level. The currently accepted hypothesis is that these non-trivial splice variants result in nonsense-mediated mRNA decay or non-functional, unstructured proteins, which do not contribute to the functional diversity of an organism. Here we show in a comprehensive study on alternative splicing that proteins appear to be much more tolerant to structural deletions, insertions and replacements than previously thought. We find literature evidence that such non-trivial splicing isoforms exhibit different functional properties compared to their native counterparts and allow for interesting regulatory patterns on the protein network level. We provide examples that splicing events may lead to transitions between different folds in the protein sequence-structure space, i.e. that different spliced isoforms of a protein might have different folds. Thus splicing, as a common genetic mechanism, could explain structural and functional diversity. Taken together, those findings hint to a more prominent role of splicing in protein structure evolution and to a different view of phenotypic plasticity of protein structures.

This content of this chapter is joint work with Fabian Birzele, and has been published in 2008 in Nucleic Acid Research [23]. My main contribution is on the definition and analysis of the structural splicing variants and the search for structure similarities with PPM.

7.1 Introduction

Alternative splicing is one of the major sources for functional diversity in the proteomes of multicellular organisms. It refers to assembling the exons of a gene in different ways during pre-mRNA splicing such that different mRNAs and, thus, proteins are produced from the same gene. Based on EST data it is estimated that up to 74% of all human multi-exon genes are alternatively spliced (1) which drastically increases the number of proteins in the human proteome and, together with time and tissue-specific regulation of alternative splicing, largely increases the functional complexity of an organism. Alternatively spliced proteins are involved in many biological processes such as apoptosis (2) or the control of synaptic function (3) and play important roles in human diseases like cancer where the influence of alternatively spliced genes on transcription factors of signaling pathways have been described (4). The effects of alternative splicing on the function of a single protein range from changes in substrate or interaction partner specificity to the regulation of DNA binding properties (5). In order to change the function of a protein by alternative splicing, its structure may be changed accordingly.
To date, the structures of less than 10 isoforms are available in the Protein Data Bank (PDB) (6) and known structural implications of splicing events on some of those proteins have been reviewed in (5). Those examples include a protein called Piccolo and the surprisingly drastic rearrangement of its C2-domain altered by a short insert of nine residues (7). Despite those few examples only little is known from experimental data about how and to what extend alternative splicing affects protein structures. Due to this lack of knowledge from biological data, several recent studies (8-10) have mapped alternative splicing events onto predicted protein structures and have analyzed features of the regions being affected. While they could link some structural properties like protein disorder (8) to a group of splicing events, the effects on many isoforms appear to be non-trivial. Based on this surprising complexity of alternative splicing on the proteome level, the most recent study by Tress et al. (9) even comes to the converse conclusion that it seems unlikely that the spectrum of conventional enzymatic or structural functions can be substantially extended through alternative splicing.
In this chapter we present the results from comprehensively mapping all splice variants annotated in Swissprot (11) onto known protein structures from the PDB leading to almost 500 isoforms annotated to more than 350 Swissprot entries whose structures can be modeled with a very high accuracy (see Materials and methods section). While about half of the events fall into variable regions of protein structures or affect complete domains, the effects on the other half appear to be non-trivial since they affect structured and well conserved regions of the corresponding protein family. The large number of such non-trivial events, which are also found to be conserved among different species, can be explained in two ways: Firstly, non-trivial splicing events are non-functional on the mRNA or protein level leading to nonsense-mediated mRNA decay or unstructured proteins which are degraded after translation. This would indeed allow only few

exons of an organism to be alternatively spliced, clearly questioning the importance of splicing on the proteome level. Secondly, they may represent evidence that non-trivial splice events may produce functional isoforms where the absence of highly conserved parts of the structure might even allow for new structural and new functional properties of the isoform.
Here we provide evidence for the second hypothesis. Having mapped a large set of splicing events onto protein structures we first explore the natural variation of the corresponding protein structure family, namely the evolutionary isoforms of the respective protein, which allow us to explain about 50% of the splicing events. The other half of the isoforms defines the set of non-trivial splicing events which can not be explained by the observed amount of variation in the respective protein family and which we examined in more detail. Based on evolutionary considerations of known fold changing events (12) we group non-trivial events into eight different categories comprising different effects to be expected on the structure level. We then show that an extensive search of the biological literature provides clear evidence of stable protein products originating from such isoforms as well as evidence for a well defined functional role of those proteins in the cell. The existence of such isoforms will help to sharpen our understanding of a proteins tolerance against major structural changes and additionally largely increases the importance of alternative splicing for generating functional and structural diversity. We will therefore review the function as well as the structural complexity of some of those isoforms. Based on those findings, we try to explain the tolerance of such isoforms against the splicing events, which can not be explained in their own fold, by members from different folds. We find evidence that such links between different folds in the sequence-structure space may indeed exist, and, for the first time, suggest a simple and common genetic mechanism, namely alternative splicing, for nature to explore them in vivo. Finally, we show that new experimental methods like Affymetrix exon array chips provide interesting data to prove or falsify our hypothesis in the future.

7.2 Results

Our study is based on 367 Swissprot proteins and 488 additional splicing isoforms which can be modeled on the structure level with a very high accuracy. As shown in Figure 7.1 about 50% of the events fall into variable, often terminal, regions of the corresponding protein superfamily (evolutionary isoforms) or affect complete domains. The other half (255 events) of the events are harder to explain since they affect regions conserved in all superfamily members including core secondary structure elements as well as highly conserved residues.
Different categories of non-trivial splicing events Based on evolutionary considerations about possibly fold changing events (12) we defined eight categories describing different types of those splicing events. Due to their hydrogen bonding patterns -strands being located at the edge of a larger -sheet are known to be more variable than internal -strands. Therefore, two categories describe events affecting peripheral or internal beta strands. Similarly, in proteins belonging to - fold classes, often -secondary structure motifs tend to be affected and, accordingly, we defined two classes comprising peripheral and internal -motifs. Additional categories contain conserved coil regions, conserved helices, large scale events (affecting more than 50% of the structure) as well as events affecting repetitive protein structure families whose repeat number is known to

vary in evolution. Table 1 shows the distribution of the splicing events in the different categories.
Literature search for experimentally verified and functionally characterized isoforms The biological literature annotated to the isoforms in Swissprot provides a valuable source of information. While Swissprot annotates proteins only if they have been verified experimentally, this must not be the case for their corresponding isoforms which may originate from large scale EST or cDNA experiments. Therefore, most isoforms have only been experimentally verified on the mRNA level, while the protein products of the isoform were not investigated in the corresponding study. Nevertheless, out of the 255 non-trivial isoforms, 43 (17%) have been experimentally validated on the protein level and for 26 isoforms (10%) the function of the spliced variant has been described. Surprisingly, and in contrast to previous findings, we find literature evidence for functionally important and well characterized isoforms in all of our eight categories (see Table 1) indicating that even large scale events may lead to functional and interesting protein products. A complete list of all literature references and isoforms is given in the supplementary material. In the following we will review some interesting isoforms from different categories.
Alternative splicing of a terminal region of a protein Many splicing events in our dataset change amino- or carboxy-terminal parts of a protein structure which are found to be more variable and differ significantly among the members of a protein family. One of the examples, where the splicing event can be explained by the variability observed in the corresponding protein family is found in the human protein p38 (Q16539), a member of the mitogen-activated kinase (MAPK) family. Those proteins are integral parts of several signal transduction pathways and known to play important roles e.g. in the stress response of the cell. For p38 several splice variants are annotated in public databases and among the well studied splice variants are two proteins known as Mxi2 (Q16539-3) and Exip (Q16539-4) which both differ in large parts of their carboxy-terminal ends compared to p38. Also, a similar splicing event is annotated for a homologous protein in mouse (P47811-2). The splicing event annotated for Exip (13) removes 46 residues from the protein structure, in addition 52 residues differ in their sequence due to a frameshift introduced by the event (see Figure 5a). It results in the loss of a well conserved interaction domain used to interact with upstream kinases and downstream substrates. This leads to the fact that the protein is not targeted by MKK6 anymore. Expression of the isoform in the cell leads to an earlier onset of apoptosis and seems to target signal transduction pathways which are different from those targeted by p38 (13). In the example of Exip, the splicing event indeed targets a more variable part of the protein family (SCOP superfamily d.144.1). Structures, which lack the carboxy-terminal part are known to fold into stable conformations.
Alternative splicing at the edges of -sheets Alternative splicing events that involve -strands naturally lead to the disruption of important hydrogen bonds in the protein structure. Nevertheless, such events are typical in the evolution of globular proteins if they affect peripheral -strands of a larger -sheet (12). Therefore, these events might be tolerable by a protein structure, even if they affect conserved regions of the proteins family. One such event is the removal of one peripheral -motif from LMPTP (P24666), a tyrosine phosphatase. The protein is known to be expressed in three different isoforms, all of which differ in a 38 amino acid long part corresponding to one -motif. The -strand represents a peripheral strand of a -sheet consisting of 4 strands in total. While the original sequence is replaced by another 38 residues in isoform 2, the corresponding part is removed in isoform 3 (LMPTP-C, P24666-3) (see also Figure 5b). Detailed analysis of

LMPTP-C (14) shows that the protein is lacking phosphatase activity and can also not be phosphorylated by Lck kinase indicating that the active center has been tackled by the splicing event. When being co-expressed with isoform 2, LMPTP-C is shown to act as an antagonist to its native variant. The proposed mechanism (14) is that LMPTP-C competitively associates with the cellular substrates or regulators of its native counterparts and thereby blocks dephosphorylation of their targets. LMPTP-C represents an example how a splicing event changes a proteins function by removing the active center of the protein. While this goes hand in hand with a loss of its native function, the isoform is still able to mimic features of the native structure which allows it act as antagonist of LMPTP.

Alternative splicing of internal strands of -sheets As shown above, the deletion of peripheral -motif can result in a functional protein revealing an interesting mechanism for the regulation of enzyme activity. The deletion of internal strands from a -sheet or a -barrel appears to be more problematic since this results in the loss of hydrogen bonds on both sides of the strand and requires the formation of several new ones to retain the native-like structure of the protein. Nevertheless, there are known examples for strand deletion events that occurred in structure evolution as discussed in (12). The p65 (Q04206) subunit of the NF-B transcriptional activator has one splice variant (Q04206-3) which exhibits such a removal event (15) as shown in Figure 5c, where nine residues, corresponding to one internal -strand, are removed. Again, for a homologous protein in mouse (Q04207-2) the same splicing event is annotated. While the splice variant lost its capability to bind to p50, the second subunit of the NF-B complex, it can form weak heterodimers with the native isoform of p65. Those heterodimers are found to be greatly reduced in their ability to bind DNA. This finding allows for two possible conclusions. Either, co-expression of the isoform and the native protein negatively regulates the NF-B function, again revealing a pattern where the inactivation of a protein feature may act as a antagonist for the native protein. Or, in case that the isoform is still able to bind IB (the inhibitor of the NF-B complex), it may act as a regulatory sink binding excess IB and allowing p65 or the p65/p50 complex to enter the nucleus (15).

Alternative splicing may affect large and conserved regions of the protein structure In the following, we give evidence for the fact that alternative splicing events may affect large and conserved regions of a protein structure and still can result in an isoform with unique functional features. CC3 (Q9BUP3) is known to be a metastasis suppressor inducing apoptosis in human cells which is not expressed in highly metastatic lines of small cell lung carcinoma. A variant, called TC3 (Q9BUP3-2) (16), undergoes an alternative splicing event which removes 107 residues from its carboxy-terminal end and further replaces 21 residues at the new terminus which do not share any sequence similarity with the original sequence. As shown in Figure 5d the splicing event affects more than 50% of the protein structure. It removes two peripheral -strands from a -sheet consisting of seven strands as well as several additional helices and strands which are not involved in the formation of the core -fold. Strikingly, TC3 has, in contrast to its native variant CC3, an anti-apoptotic function which seems to be located in its unique C-terminal part. Even though the protein lost several conserved elements of its fold it seems to be able to fold into a stable, functional isoform (see also Figure 3, rightmost column). It is shown to be short-lived due to a degradation signal located in the new carboxy-terminal end of the protein (16) which possibly represents another physiological feature. A second example which exhibits a similar splicing

event that removes an even larger part from the structure is an isoform of caspase 9 (P55211-2) (see Figure 5e). The isoform, named caspase 9b, is again shown to function as an endogenous apoptosis inhibitory molecule (17). The isoforms of CC3 and caspase 9 reveal a surprising tolerance of -fold proteins to large scale aberration events. This tolerance might originate in the evolutionary history of proteins of this fold class as they might have evolved by successively adding -motifs to the edges of the core sheets. This might result in the fact that they can also be removed from the structure without loosing capability to fold into a stable conformation.

Alternative splicing of highly repetitive protein structures Internal repetition of (super-) secondary structure elements resulting from intragenic duplication and recombination events has been observed for several protein structure families. Proteins that exhibit repetitive structure are involved in many different functions in the cell while an increase of the repeat number in general affords a protein enhanced evolutionary prospects due to an enlargement of its binding surface area (18). While repeats are found in all phyla they seem to be more common in eukaryotes which may be associated with an increasing complexity of the cellular functions that are readily available from assemblies of repeats. Several protein families contain repetitive elements but the major classes are -propellers (b.67, b.68, b.69, b.70), -trefoils (SCOP fold b.42), - superhelices (SCOP fold a.118), Leucine-rich repeats (SCOP fold c.10) as well as the -hairpin--hairpin repeats (SCOP fold d.211) (18). The fact that repeat duplication has been a successful strategy throughout protein structure evolution that changed functional features of the proteins without loosing the possibility to fold into a stable structure leads to the conclusion that such proteins should be highly tolerant against structural changes by alternative splicing. Indeed, we find that 4 out of the 5 repeat classes (all except for -trefoils) described above harbor alternative splicing events affecting complete sets of repetitive motifs. Two examples are shown in Figure 5f und 5g. For most repetitive protein structures large scale deletion events are likely to be tolerable by the structure though, unfortunately, experimental validation and a functional categorization of the splicing variants is missing for all isoforms in our dataset. The principle of changing the number of repeats in the course of evolution in order to evolve novel functional features seems also to be used frequently to increase the functional diversity by alternative splicing as indicated by the large number of repetitive protein folds with annotated splicing events.

Alternative splicing may support hypothesis on the origin of TIM-Barrels from Half-Barrels For a number of protein structures and protein structure families it is well known that they resulted from ancient gene duplication and/or fusion events. Such duplication events are not always obvious from sequence data since the two subdomains have possibly already evolved to an extent where sequence similarity is random. A well studied and recurrent motif in protein structures is the (/)8-barrel family (TIM-barrel, SCOP fold c.1). Proteins in this family adopt a large variety of different functions and based on sequence and structure analysis it has been proposed (19) for some members of that family that they originated from a gene duplication and fusion event of two ancestral half-barrel proteins. Those ancient half-barrels probably formed a homodimer consisting of two identical half-barrels (20). Our analysis now provides additional support for this hypothesis since it reveals two splicing isoforms (Q9BZP6-3 from CHIA HUMAN, and P27934-2 from AMY3E ORYSA) where one half of the barrel is removed by a large-scale removal event (see figure 5h). The isoform of the human chitinase gene (Q9BZP6-3) has been described by Saito et al. (21) to be specifically expressed in lung though experimental validation

of the existence of the stable protein product is lacking. In comparison to its native isoform the protein lacks a secretory signal sequence leading to the conclusion that it might be present in the cytoplasm instead of being secreted. It also lacks the amino-terminal active site essential for chitinase activity. So far, we have no experimental validation for a functional gene product and a stable protein resulting from those splicing events. Nevertheless, based on the proposed evolutionary mechanism of fusing two half-barrels by an ancient gene duplication and fusion event, the splicing isoforms possibly form a (homo-)dimer to reconstruct the complete barrel. The possibility to express proteins of the TIM-Barrel family as half-barrels might offer an increased functional variability by combining half-barrels containing different functional sites in heterodimeric complexes.

Indications for fold transitions caused by alternative splicing For 225 non-trivial isoforms (excluding repetitive and conserved coil cases as these splice events will presumably not result in a different fold) we searched for similar structures as described in the Materials and methods section. Applying the TM-Score criterion (TM-Score > 0.4) (22) alone we find for 139 (66%) isoform structures resulting from splicing events a similar structure from a different fold. Applying the more stringent criterion (secondary structure and isoform coverage) results in 49 isoforms (47 of which have a TM-Score larger than 0.4). For these, we superposed the spliced structure with the target fold and visually inspected the superpositions for conservation of core secondary structure elements as well as their connectivity, i.e. the topology of the core elements. We observe 21 (10%) highly confident superpositions, i.e. models for the spliced structures having a fold different from the one of the non-spliced protein. Thus, these different folds are probable structural models for the isoform, which could explain the drastic changes caused by the splicing event (four examples are shown and briefly discussed in Figure 3). Of course, proteins resulting from splicing events might not be able to fold at all into a stable structure and often this will be the case. In other cases the structure might be stable but will form a novel fold (so far not solved and deposited in the PDB). In rare cases, the modified structure might be similar to a known fold different from the native one. In the latter case we would observe links between different folds by defined genetic changes (alternative splicing) transforming one stable 3D structure into a different stable 3D structure. Despite many attempts and research on structure classifications and structural descriptions and features, which led to the well known structural resources such as SCOP and CATH (23), reliable and traceable links between fold classes are very rare. This is even more the case for evolutionary explanations of the observed similarities and events. Here we do not only observe a considerable number of such transformation events but also provide a simple genetic mechanism explaining them as all the events correspond to known observed transcripts.

Validation of non-trivial splicing event using exon array data Recently, Affymetrix released a novel type of chip which is capable of measuring most exons in Human as well as Mouse (Mus musculus) and Rat (Rattus norvegicus) by single probesets on the chip. The analysis of such chips allows measuring the expression of different transcripts of one gene in vivo, under different experimental conditions and in different tissues. The time and tissue specific expression of certain transcripts may be another indication of a functional role of the corresponding gene product in the cell and will therefore help to understand functional diversity resulting from alternative splicing. Additionally, this data will be a helpful resource to validate or falsify our hypothesis

that many, also non-trivial splicing events may play functional roles in the cell. For this reason we have mapped all probesets provided on the human exon chip onto human exons annotated in Ensembl (24). Additionally we have structurally modeled all human genes for which we can find reliable structural annotations in the PDB. The data can be accessed in the ProSAS database (25) at http://services.bio.ifi.lmu.de/ProSAS. In total, we are able to cover 80.1% of all human exons with at least one Affymetrix probeset. When concentrating on high quality structures (more than 40% sequence identity between template and human transcript) about 35% of the human genes are covered at least partly by protein structure while 17% are completely (more than 75% coverage) modeled by a protein structure. This indicates that the combination of exon chip experiments with structural data indeed has the potential to test our hypothesis since a large number of genes (and exons) is at the same time covered by structure and measured on the chip.

7.3 Discussion

Our study reveals a large number of functionally important, alternatively spliced proteins that harbor non-trivial splicing events and hints to a high degree of plasticity and a large tolerance against major rearrangements on the protein structure level. The possibility to express the antagonist of a protein as an isoform of the native variant represents an intuitive mechanism to increase the functional complexity of an organism by alternative splicing and has been discussed by several studies before. The structural explanation for this mechanism may be grounded in the removal of highly conserved parts, which are essential for the function of the native variant. If the isoform is still able to fold into a native-like structure, it can mimic native structural features and interact with native interaction partners without processing them further. Thus, alternative splicing immediately provides a mechanism for turning an activator into an effective inhibitor via a simple, possibly regulated, genetic mechanism.

The sequence-structure protein space tries to link different folds by appropriate similarities and differences but examples for fold transitions are rare and typically difficult to explain biologically. Our study provides examples for such links and explains them with a simple and common genetic mechanism. Thus, alternative splicing may be a new approach to chart the protein space and gain insights into mechanism of protein structure evolution. Future work will be on exploring the fold space as well as the changes that occur within and between folds in the context of alternative splicing in more detail. Therefore, structural analysis of alternative splicing events may help to identify common paths of protein fold evolution similar to the events discussed by (12) and to describe events that may be tolerated within protein families. This knowledge may have interesting applications in protein design.

Without experimental proof we can currently only speculate about the structures of isoforms resulting from non-trivial splicing events. Several facts indicate that at least some of those isoforms could have a well defined structure. They perform a well defined function in the cell and are able to mimic features of their native counterparts. Additionally, the identification of fold transitions exemplifies that they could adopt structures from different folds. Nevertheless they might also be unstructured or fold into yet unknown conformations. Therefore, this study will provide interesting starting points for experimentalists trying to gain a deeper understanding of

the non-trivial alterations of protein structure produced by alternative splicing and will lead to new insights into protein structure stability and the principles of protein fold evolution.
We also expect recently established experimental techniques like exon-level micro arrays to contribute significantly to our understanding of the functional and, in the context of this study more important, the structural effects of alternative splicing. NMR-techniques (26) and mass spectrometry (27) measuring complete proteomes will contribute to validate or falsify the importance of alternative splicing for the functional diversity of complex organisms.
As in principle alternative splicing is a mechanism to produce a combinatorial number of transcripts, even a small percentage of stable structures implies a very large number of new protein variants. Thus, we believe that evolution makes use of alternative splicing to produce structural and functional diversity and this diversity is due to the large structural plasticity of proteins. Understanding alternative splicing on the proteome level will be one of the major challenges of computational and experimental biology in the next years.

7.4 Materials and methods

In the following the methods and data sources used for our study are described in detail. The methods are summarized in the supplementary figure S2.

7.4.1 Alternative splicing and literature data

The data for alternatively spliced proteins used in this work was obtained from the Swissprot protein database (September 2006), which annotates splicing events for 9135 out of 231434 protein entries. Those 9135 entries harbor 20845 alternative splicing events where 56.6% of the events are deletion events and the other 43.3% of the events represent replacements. In 22.2% of the replacements the original sequence is shorter than the replacement sequence (insertions), in 27.7% the replacement sequence is shorter than the original sequence (deletions) while in 50.1% of the cases the original sequence and the replacement sequence are of the same length. Literature assignments to different isoforms are also provided by Swissprot. We have examined them manually for evidence for the experimental proof of a stable protein product as well as experimental validation of its function.

7.4.2 Protein structure assignment

To obtain protein structure data we ran BLAST (28) against all proteins in the PDB (August 2006) for all Swissprot proteins with annotated splicing events. For each alternatively spliced protein we then used free-shift alignment (29) (Pam250 matrix, gap open: 12, gap extend: 1) to compute full length sequence-structure alignments of the alternatively spliced Swissprot entries with their respective homologues identified by BLAST. From all alternatively spliced Swissprot proteins only those whose structure could be modeled with a very high sequence identity of at least 60% between template and target and whose sequence is covered to at least 75% by protein structure are used for further analysis.

7.4.3 Assignment of protein structures to families

The protein structures used to model the Swissprot proteins have been assigned to their corresponding SCOP (30) families as defined in SCOP version 1.71 (December 2006). All Swissprot entries that are modeled with structures not yet classified in the SCOP version 1.71 were assigned to their respective protein families using Vorolign (31). The final dataset contains 367 Swissprot proteins with 488 annotated isoforms which are classified into 166 different families, 134 superfamilies and 119 folds with respect to the SCOP hierarchy.

7.4.4 Multiple structure alignments and evolutionary isoforms

Multiple structure alignments were computed from multiple structure superpositions with STACCATO (32) which has been shown to compute accurate alignments with respect to both, sequence and structure. In order to guarantee enough variability within the set of evolutionary related protein structures, we use proteins from the same SCOP superfamily. On this level of the SCOP hierarchy, protein structures exhibit enough structural variance to allow the definition of conserved and variable regions without overestimating structure conservation due to too similar proteins. Each set must contain at least three members and their structural similarity is measured by the TM-Score (33). Proteins in a set have to be similar enough (indicated by a pairwise TM-Score of larger than 0.4) while still showing structural variability (TM-Score smaller than 0.8). Given a multiple structure alignment, a conserved region of a SCOP superfamily is defined as a block of at least 10 residues which are conserved among all members of the superfamily. Each protein in a block may contain two gaps at maximum to account for some small variability within blocks. All regions outside of the conserved blocks are defined as variable regions. The proteins in the set define what we call evolutionary isoforms which display insertions, deletions and substitutions and define the set of evolutionary events that are likely to be tolerable for a protein structure.

7.4.5 Alternative splicing and alternative structural models

In order to suggest alternative structures for non-trivial alternative splicing isoforms we applied the splicing event onto the structure (e.g. removed the structural parts belonging to a skipped exon). We then searched for reliable structural superpositions of the resulting structure model against all known folds (according to the SCOP classification). Such a search resulted in a number of structurally similar proteins from SCOP folds different than the proteins own fold. The soundness of such superpositions was measured by different criteria. First, as argued by Zhang and Skolnick (22) a TM-Score larger than 0.4 is a clear evidence for a structural similarity (criterion 1). Since we believe this criterion is not strict enough to claim such remote structural similarities we also used more stringent criteria. Therefore, we filtered the superpositions to those superposing at least 80% of the spliced structure and 60% of its secondary structure elements and additionally examined the resulting superpositions manually for the conservation of core secondary structure elements with the correct connectivity and topology (criterion 2).

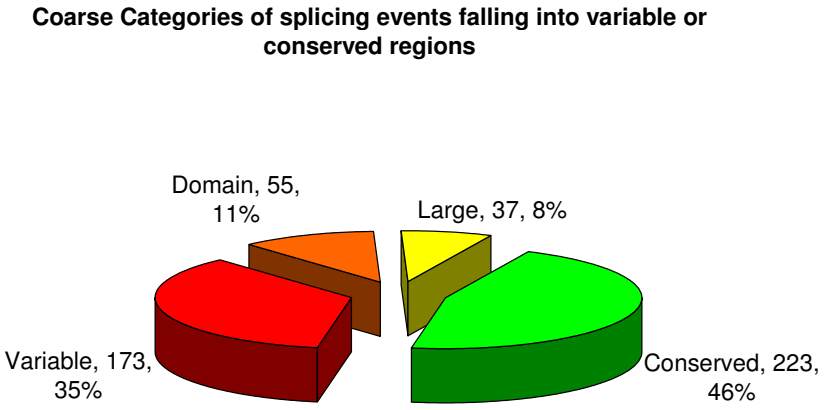


Figure 7.1: Distribution of 488 splicing events in the four major categories. 35% of the events fall into variable regions of the corresponding superfamily while 11% affect complete domains of multi-domain proteins. 8% of the isoforms affect larger regions (more than 50% of the structure) while 46% affect conserved regions of their corresponding superfamily which are present in all superfamily members.

7.5 Discussion

Our study reveals a large number of functionally important, alternatively spliced proteins that harbor non-trivial splicing events and hints to a high degree of plasticity and a large tolerance of protein structures and folds against major rearrangements. The possibility to express the antagonist of a protein as an isoform of the native variant represents an intuitive mechanism to increase the functional complexity of an organism by alternative splicing and has been discussed by several studies before. The structural explanation for this mechanism may be grounded in the removal of highly conserved parts, which are essential for the function of the native variant. If the isoform is still able to fold into a native-like structure, which often seems to be the case, it can mimic native structural features and e.g. interact with native interaction partners without processing them further. Thus, alternative splicing immediately provides a mechanism for turning an activator into an effective inhibitor via a simple, possibly regulated, genetic mechanism.

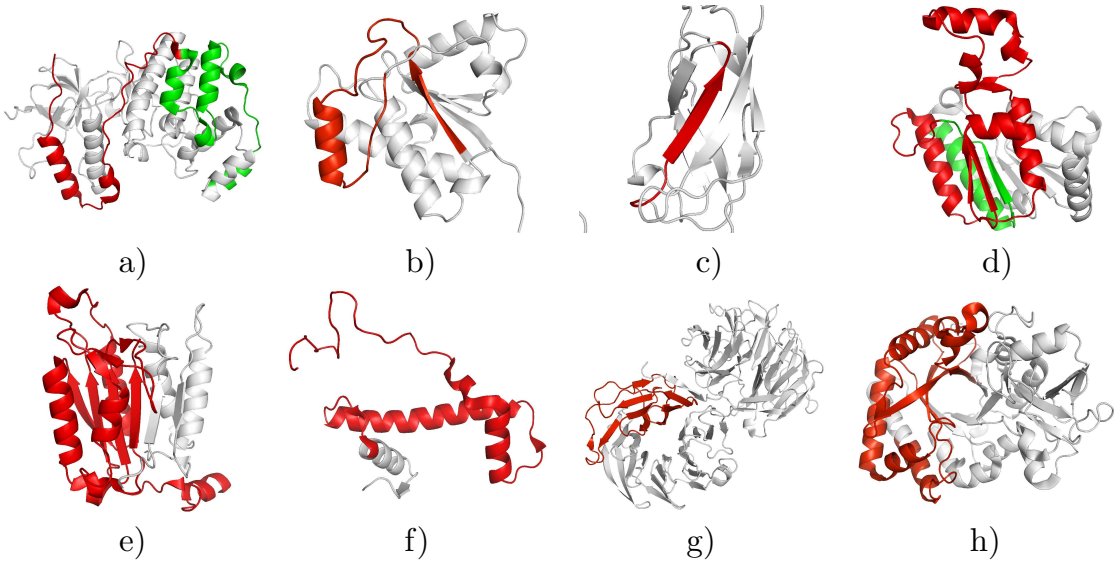


Figure 7.2: Visualization of alternative splicing events on the structure level. Substitutions are colored in green while deletions are colored in red. All figures have been created using PyMOL (http:www.pymol.org). a) shows the removal of the carboxy-terminal part from MK14_HUMAN (Q16539-4, pdb: 1zzlA), b) the removal of one external strand and helix motif in PPAC_HUMAN (P24666-3, pdb: 5pnt), c) the removal of an internal strand in TF65-HUMAN (Q04206-3, pdb 1nfi), d) the removal of a large part of the protein from TIP30_HUMAN (Q9BUP3-2, pdb: 2bkaA), e) the removal of several strands in CASP9_HUMAN (P55211-2, pdb: 1nw9B), f) and g) the removal of repetitive motifs from WDR1_CAEEL (Q11176-2: pdb: 1pevA) and CD2A1_HUMAN (P42771-2, pdb 2a5e) as well as h) the removal of one half of a TIM-barrel structure in CHIA_HUMAN (Q9BZP6-3, pdb: 1vf8A).

	Coil	α	β(p)	β(i)	$\alpha\beta$(p)	$\alpha\beta$(i)	Repeat	Large	Total
Region	14	50	29	25	49	35	21	37	260
Isoform confirmed	4	15	2	9	6	3	1	3	43
Function described	2	7	1	5	5	3	1	2	26

Table 7.1: This table displays the distribution of non-trivial isoforms in the eight categories defined based on evolutionary considerations with respect to different features. (p) and (i) indicate the position of the corresponding -strands either at internal or peripheral positions of the sheet. The Conserved region affected row displays the number isoforms which affect conserved regions of the corresponding superfamily. The Isoform confirmed row displays isoforms which have been confirmed in the literature on the protein level, while the Function described row references isoforms in the different categories which have been described in the literature to perform a well defined function. The Function / Class row contains the log odd ratios of functionally described isoforms in the different structural classes (third row) versus the background class distribution (first row). All Log odd ratios of the values given for Isoform confirmed and Function described against the background distribution of structural classes within isoforms with conserved regions affected can be found in the figure 7.4.

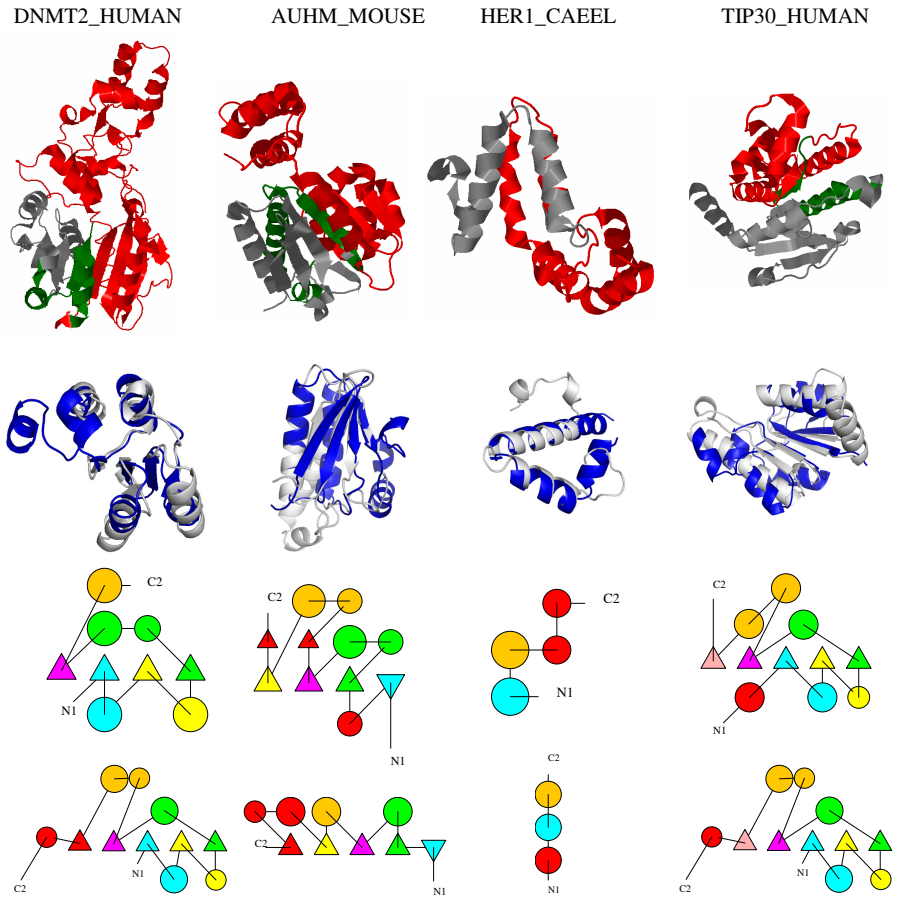


Figure 7.3: This figure shows four examples for possibly fold changing splicing events by non-trivial splicing events (i.e. those which cannot be accommodated in the native structure) and by superposing the spliced structure to a different SCOP fold. The examples can also be explored interactively following this link URL. Each column represents one example. In the first row the splicing event is visualized on the native protein structure of the Swissprot protein. In the second row the superposition of the spliced protein with the corresponding protein from a different fold is shown. Rows three and four display TOPS (34) diagrams of the spliced protein (row three) and the protein belonging to the different fold (row four). Corresponding secondary structure elements are colored the same, elements missing in the other protein are colored in red. Sometimes corresponding helices are split up which frequently results from breaks in the DSSP assignments. From left to right the following examples are shown: Column 1: DNMT2_HUMAN (O14717-6, Astral: d1g55a_, SCOP: c.66.1.26). The spliced protein superposes very well (TM-Score: 0.68) with d1gsoa2 (SCOP: c.30.1.1). Topologically the proteins are very similar, except for a very short strand (length 2) - helix (length 4) motif at the C-terminal end of d1gsoa2. Column 2: AUHM_MOUSE (Q9JLZ3-2, Astral: d1hzda_, SCOP: c.14.1.3) which superposes well (TM-Score: 0.56) with d1vc1a_ (SCOP: c.13.2.1). Topologically the proteins are similar, except for two small strands (both of length 2) and one short helix (length 3) missing in d1vc1a_. Additionally, the C-Terminal part of d1vc1a_ has an additional, short helix-strand motif. Column 3: HER1_CAEEL (P34704-2, Astral: d1szha_, SCOP: a.226.1.1) superposed with d1ni8a_ (SCOP: a.155.1.1, TM-Score 0.49). Only a small fragment (helix-turn-helix-motif) is left over by the splicing event. The two TOPS diagrams are similar with the two main helices being preserved while short helical parts are missing in either of the two proteins. Interestingly, d1ni8a_ is described to contribute to DNA binding after dimerization (35) which might also be the way how the isoform resulting from the HER1 splicing event is stabilized. Column 4: TIP30_HUMAN (Q9BUP3-2, PDB: 2bka, SCOP: c.2.1.2) which again superposes well with d1gsoa2 (see also DNMT2_HUMAN) from SCOP fold c.30.1.1 (TM-Score: 0.54). Topologically the two proteins are very similar according to TOPS except for two helices missing at the C- and N-Terminal ends. The function of the isoform is discussed in the text (isoform TC3). Images have been created using Jmol (http://www.jmol.org) and PyMol (http://www.pymol.org) .

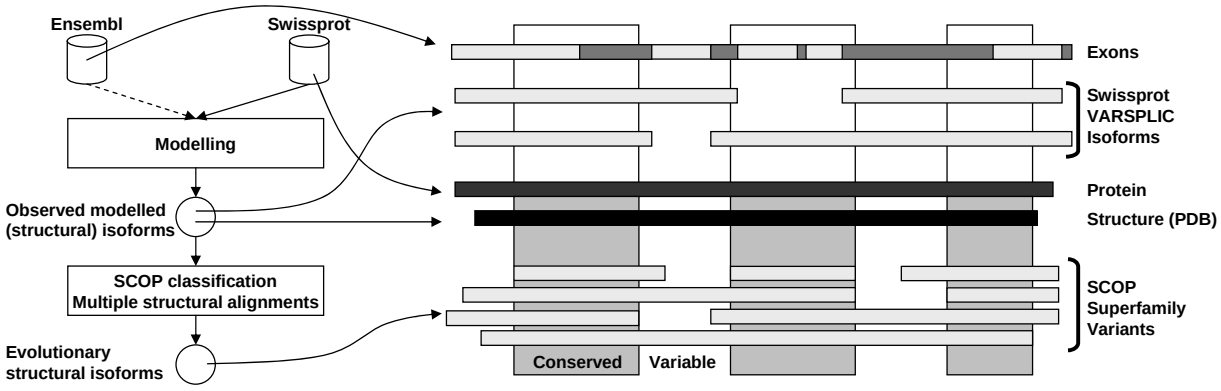


Figure 7.4: The figure shows the main modelling steps of our approach and the resulting gene products. Validated isoforms annotated in Swissprot are conservatively mapped onto 3D structures from the PDB resulting in structural models for the Swissprot isoforms. The SCOP protein classification and high quality multiple structural alignments with variability defines Conserved and Variable regions and thereby evolutionary structural isoforms observed in superfamilies. This allows assessing whether the validated isoforms affect only variable regions or structural cores of proteins.

.

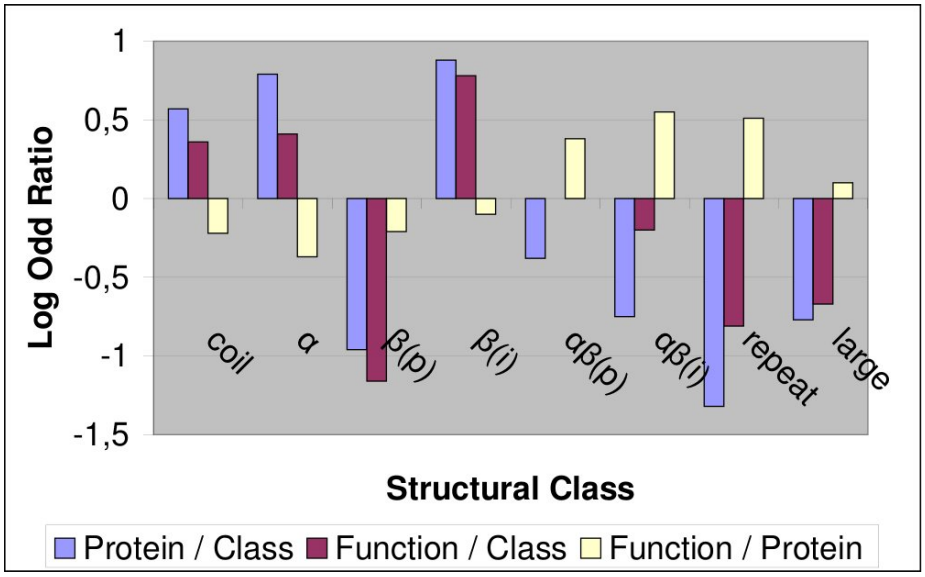


Figure 7.5: This figure shows the log odd ratios of the values discussed in Table 7.4.5. Protein / Class represents the log odd ratios of the confirmation of the protein product in the literature against the background of the representation of this class in the 255 isoforms harboring non-trivial events. Function / Class are the log odd ratios of functionally described isoforms in the different structural classes versus the background class distribution. Finally, the Function / Protein values are the log odd ratios of the functional description of the protein in the literature against the confirmation of the protein product in the literature. Surprisingly, literature evidence for events affecting internal beta strands as well as alpha helices is overrepresented (even more than coil affecting events) while literature evidence for peripheral strand events is clearly underrepresented in the data. Also, literature evidence for events affecting repetitive elements is underrepresented, even though one would expect that such events may be easily tolerated by the structure and may lead to functional diversity of the isoforms.

Part II

The high-throughput experimental data context

Currently, many bioinformatics problems are implied by new experimental biomolecular techniques, most of them high-throughput measurements. Several of these techniques such as microarrays, next generation sequencing or quantitative proteomics allow to measure gene or protein expression on a genome-wide level. Differential measurements of two or more states of a system yield so called experimental outcomes, (long) lists of significantly and differentially expressed genes and proteins (possibly with associated fold change and significance values). Our goal is to develop new methods to help with the interpretation of such data in practice. Again prior knowledge and context information and constraints need to be employed in such analysis tools.

We have developed new methods for textmining based on context-dependent ontologies, a context based approach to more reliably map short reads from high-throughput sequencing experiments, a new method generalizing gene set to gene network enrichment analysis, and a new method to exploit networks to explain experimental outcomes.

Chapter 8

Extracting knowledge from texts

One of the key steps in interpreting the experimental outcome from high throughput experiments is to collect available knowledge and map the observed changes on this contextual knowledge. Due to the nature of how knowledge is generated in research, this knowledge is hidden in thousands of scientific publications. Therefore, in order to get the context for interpretation in a structured fashion we have to identify and extract the objects of our interest and their relations. This chapter addresses this task: we introduce textmining techniques and an annotation tool to define the context for the interpretation used in chapter 12. The main results presented in this chapter are of technical nature: the focus is to retrieve a context usable in practice effectively. The method presented for named entity recognition (syngrep) is able to identify and output different types of entities or their possible relationships. It is very fast, it can scan all PubMed abstracts in only a couple of minutes against hundreds of thousands of known entity synonyms. This efficiency allows for more flexibility: missing or non-specific names for proteins or other entities can be simply added or filtered out in an iterative way, or specific questions based on co-occurrences can be answered fast. The availability of this co-occurrence searches enables also the efficient usage of the more time-consuming RELEX ([61]) algorithm. In this thesis we developed a fully automated version of RELEX based on the method presented in [61] with the extensions opposed in the Master's thesis of Theresa Niederberger [142]. As a result of this and using the efficient three-occurrence prefiltering given by syngrep we can derive or update large in-depth characterized regulation networks for example for human within a couple of hours on our computer cluster. Although the derived context may contain errors, the regulations contained in the network provide valuable information, and most importantly, as the source of any regulation is linked to the corresponding publication all the regulations are testable. Of course it is not feasible to check all extracted regulations, but given experimental data, one can focus on the relevant ones (e.g. on the regulations corresponding to genes significantly changed in some experiment) and, thereby ensure the full correctness of the context as proposed in chapter 12. In addition, in this chapter we present a tool for in-depth analysis of a given context of interest: the relation annotator web-tool RelAnn. Exemplified with the biological process "diauxic shift" in yeast, RelAnn uses syngrep and RELEX as a prefilter, and provides an integrative and unified way to annotate and organize the relevant regulations in this context.

8.1 Finding occurrences of biological entities in texts with syngrep

The first step of extracting knowledge from publications about relations between biological entities such as genes or proteins is to find occurrences where these are mentioned in scientific texts.
The task of identifying text parts corresponding to some entity of interest and assigning the correct entity to the occurrence is called named entity recognition (NER).
In the context of biological entities the main challenges for NER come from two factors: (i) the size of the text database (e.g. the free scientific texts available such as all abstracts from PUBMED) and (ii) the non-standardized naming of the entities.
The naming of entities often reflects the evolving history of knowledge: genes are named according their phenotypic effects for the knock-out genotype (especially for fly genes) resulting in names overlapping with common English words such as 'dwarf', 'early' or 'brief', or in other cases at the beginning only the putative position of the genes were known resulting in names such as 'orf1', 'orf2' etc, or the knowledge is derived due to sequence homology resulting in names such as 'A domain-containing protein similar to matrilin and collagen' 'Brain-specific chordin-like protein'. For a deeper overview of the nomenclature issues for genes and protein see [62].
While challenge (i) is a matter of computation power, (ii) requires algorithms to disambiguate between false and true matches and also to disambiguate between the highly overlapping names of different biological entities.
Accordingly, one can divide the NER methods into two classes: dictionary based or dictionary free. While for the dictionary based methods the entities and their names and spelling variants of the names are predefined, dictionary free methods try to identify previously unknown tags as additional biological entities. Our primary goal is to extract relationships between biological entities, and so we rely on dictionary based named entity recognition. One of the best-performing tools for this is ProMiner [82]. As the occurrence detection of ProMiner is a very time consuming process we developed and alternative simple but ultra-fast occurrence detection algorithm: syngrep. The input and output for syngrep is the same as for ProMiner: the two inputs are (1) the biomedical text to perform the search on (usually PubMed abstracts) and (2) the pre-processed synonym dictionaries, and the outputs are the detected occurrences of the entities. While the outputs of syngrep are practically identical to the results or ProMiner (evaluated by Katrin Fundel, one of the authors of ProMiner) it is more than 100 times faster.
Next, briefly describe the underlying algorithmic steps for syngrep, and the built-in filtering/disambiguation rules.

8.2 The syngrep method

The core of the syngrep matching is a PATRICIA trie [132] where all names and spelling variants of all entities are stored (see figure: 8.2). To allow approximate matching to some extent, we consider only two character classes: informative (small-case alphanumeric characters)

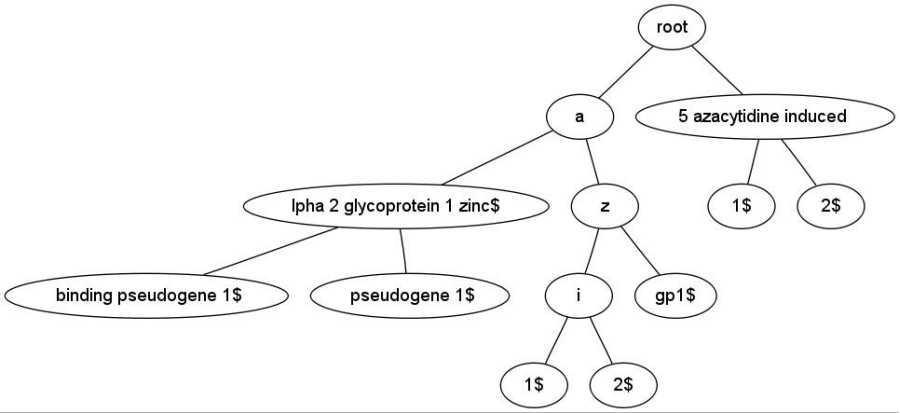


Figure 8.1: A simple PATRICIA trie using only some synonyms for four human genes. While for many abbreviated symbols PATRICIA ends up in similar structures as the Aho-Corasick trie, for the long name variants the PATRICIA representation saves a lot of memory and yields to typical trie-depths between 20-30 (see table 8.2).

and non-informative characters (whitespaces, various delimiters). First, every input synonym is converted into a reduced-representation string by mapping the input characters either to small-case variants or to a whitespace (the mapping is redefinable). Implied whitespace segments are reduced to a single whitespace. The set of normalized strings is then used to construct the PATRICIA-trie, where all entries are linked to their defining synonym and to the set of synonyms and identifiers (ids) describing the same entities (this information is used in optional subsequent filter/disambiguation steps). Now, given the memory-efficient lookup in the PATRICIA trie, we can apply the Aho-Corasick algorithm [4] to find all occurrences in a large text database. In syngrep we apply two minor modifications: (i) we do not use failure links and (ii) we only start lookups at word boundaries - both modifications are implied by the nature of the application: there are no valid “within words” hits and as usually the depth trie is very small and only word boundary failure links would be accepted, there are very few cases where a failure link would speed up the matching. As the search trie is not modified upon matching and the limiting factor of the matching is the I/O traffic, it improves the runtime when the matching is performed in a highly parallel mode using shared memory for the trie among the processes - even on a single processor. In table 8.2 we provide some information about the typical properties of the PATRICIA-tries constructed from synonym lists, and about the speed of the matching via syngrep on a current computer (16 core, AMD Opteron, 2.6 Mhz) with 80 parallel threads against $\tilde{2}$1.5 Mio PubMed abstracts ($\tilde{3}$55 Mio sentences, 25 Gb text). Given this speed, syngrep can be used as a quick prefilter for more sophisticated algorithms (for example SynTree in chapter 9) and via the built-in simple filters to provide the input for relation extraction steps.

Simple built-in filters/disambiguation in syngrep

Although the main purpose of syngrep is a fast occurrence matcher, it also provides some simple and general filters/disambiguation strategies and also allows for some generalized matchings.

Synfile	Entries	Depth	Inner Nodes	Leafs	Terminals	runtime
Human	34842	23	150345	319908	373528	11m 18 sec
Bovine	19939	20	47671	93361	107277	8m 37 sec
Fly	30459	19	90404	179583	223267	10 m 53 sec
Mouse	41173	22	146154	319631	360727	10 m 42 sec
Rat	39613	22	95477	171858	215751	9m 28 sec
Yeast	7787	17	56905	90647	111837	7m 12 sec
All	173813	25	408038	853053	1017994	19m 10 sec
RelEx input(*)	173889	25	410099	857364	1022782	14m 48 sec

Table 8.1: Properties of typical syngrep-PATRICIA-tries. Typically, a synfile corresponds to a given context describing entities with multiple synonyms - here organisms. The column entries define the number of entities defined in the synonym file, the columns depth, inner nodes, leafs and terminals describe properties of the PATRICIA trie build up in memory for searching, runtime is the real-time usage for syngrep matching the trie against the whole PubMed. The settings for RelEx input are described in section 8.3. The main speed-limiting factor is still I/O: even though for RelEx input there are more keys and more rules to check, it is faster than 'All' as the output size is much smaller (1.1 Gb vs 7.1 Gb).

Here we give a short overview of these options.

Filters based on the matched text

Disallowed matchings: As syngrep performs the matching in an ignore-case and approximate matter (any sequence of non-alphanumeric characters can be matched to any other sequence of non-alphanumeric characters) one can specify unallowed unspecific character matchings by defining unallowed character mappings. An example of such a non-allowed matching is to match any of "<>=" to "-." so that for example the protein named "PU.1" cannot be matched to a string "PU>1".
Length specific case sensitivity: as many short protein names are abbreviations one can specify an integer value for the so called "trusted word length". Matches having a length below this threshold will only be accepted if they match case sensitive.
Excluded words: Many gene names are derived from the description of the knock-out phenotype and are also common English words. Naturally, such synonyms lead to a large number of detected occurrences, most of them not representing the gene. As a solution one can force the use of a list of excluded words for a given synonym file, and prune hits overlapping with these excluded words if the matched text is written in small case (usually gene names overlapping with common english words are written in an all-capital manner if in the right context).
Inline abbreviation pruning: Another general pattern in gene/protein naming is the use of abbreviations. In the cases where the abbreviation is defined "inline", i.e. in the form "long name variant (abbreviation)" we can check whether the long name variant describes the same entity as the abbreviation. To test this, syngrep aligns every long name variant from the dictionary consist-

ing of at least two words against the prefix of the abbreviation matched (the part of the text before the starting parenthesis) via dynamic programming. If an alignment with at least 80% identity is found, syngrep assumes that it is a novel spelling variant of the entity of interest, otherwise the hit will be discarded.
Negative abbreviations: In contrast to the previous case, there are many cases where the genes only use the abbreviated forms (no long form is mentioned in the text), disabling the strategy of inline abbreviation pruning. On the other hand there are several expressions abbreviated the same way, for example the Histo-blood group ABO system transferase protein is named "ABO" after the ABO blood group system, but the abbreviation "ABO" is also used for "air breathing organ" or for "alpha benzoin oxime". Syngrep therefore can handle negative abbreviation definitions, where for every abbreviation invalid long forms are listed. Both the short and long forms will then be included in the PATRICIA-trie. Upon matching of an abbreviation syngrep will check whether an inconsistent long-form is also matched within the same text corpus. If this happens, the hit will be pruned.

Abbreviation detection and generalized matching

Inline abbreviation detection: The inverse problem of inline abbreviation pruning is the detection of inline abbreviations, applying for the cases where we find a hit to the long name variant in the text in the form "long name variant (abbreviation)". In these cases the abbreviation may be ad-hoc, and not widely used, so it is not in the dictionary, but in the remaining part of the text the entity is referred to by the abbreviation. Therefore, syngrep dynamically defines the abbreviation as a synonym for the entity for the remaining part of the text, and assigns the entity to all occurrences of the abbreviation accordingly.
Generalized matchings: In some use cases one would like to detect several variants of a given entity, e.g. while looking for the general disease "cancer" one would like to get the information of the specific cancer variant mentioned in the text. To achieve this, syngrep provides the possibility to define "expanding" identifiers, so that one can define entities in the form:

```
cancer@SYNGREP_BACKWARD_EXPAND:glioma|melanoma|cancer|lymphoma
malignant@SYNGREP_FORWARD_EXPAND:malignant
```

Accordingly, syngrep expands the matched text by the next word in the required direction, so that the entity "cancer" from the example above would match "breast cancer" or "skin cancer", and the entity "malignant" would report matches like 'malignant tumor" or "malignant neoplasm".

Disambiguation strategies and tuple filters

Although the main focus of syngrep is to act as a fast occurrence detector and as a prefilter to more involved methods, it offers two optional, simple strategies to resolve ambiguities based on the input synonym dictionaries: (i) "long name" pruning and (ii) context based pruning. In both cases the general principle is to only resolve ambiguities with sufficient evidences, and report all ambiguous mappings otherwise.

long name disambiguation: This strategy is used for overlapping short synonyms within one dictionary. If there are also defined longer name variants for the corresponding entities, and syngrep detected those somewhere else in the abstract for only one of the entities it assumes that the matched shorter variant also refers to the same entity.
context based disambiguation: syngrep threats every synonym file as a context of its own, and accepts an additional ”context“ definition where synonyms describing the context itself can be defined. For example if using the organism specific synonym files from table 8.2 one can use a context file containing terms for describing the organisms such as ”human“, or ”h. sapiens“ for the human synonym file or ”mus musculus“ or ”mice“ for the mouse synonym file. These synonyms will then be added to the PATRICIA-trie and in the frequent case (due to the homologous genes) of cross-synonymfile ambiguities syngrep can use the information for which context terms were matched. If the text clearly describes entities of a given context (i.e. organism) syngrep can discard the overlapping hits from all other contexts. In addition one can also require the matching of context-specific terms to report any entity occurrence - for example in the case of the fly synonyms one can further reduce the putative false positive matches in articles not mentioning fly at all.
Tuple filters: One of the most frequently used filters are the so called 'tuple' filters. With this option one can specify sets of synonym files sets to be matched within one abstract or sentence in order to report them. Doing so one can focus on co-occurrences of synonym classes, for example if one is interested in articles describing regulations of proteins/genes induced by some metabolite in the context of diseases one can create a three-tuple search with three synonym file sets: (1) synonym files for proteins, (2) synonym files for metabolites and (3) synonym file of diseases. In many cases, using this option with specific synonym files in combination with the built-in HTML output of syngrep results in a fast and directly usable textmining solution.

8.3 From named entities to relations

After the identification of entities in scientific tests, we can proceed to extracting relations between them. The simplest information one can extract is co-occurrence. In most cases co-occurrences describe cases no regulatory relationships, best one can use them as a document/sentence prefilter for more involved methods, or manual annotation. As described in the previous section, syngrep directly provides the possibility to define “tuple” searches and so look for co-occurrences of given entity types within abstracts or sentences. As one can define synonymfiles for any context, the co-occurrence search option of syngrep is often used in different projects to get an initial impression about the available knowledge (e.g. [137]).
One of the best performing methods [101] of relation extraction from sentences with already identified entities within is RelEx [61]. In the course of this Phd thesis we reorganized the implementation of the RelEx workflow (see figure 8.3) so that the installation and use of RelEx became much easier and parallelizable. As RelEx uses so called “action” keywords in addition to the named entities, we also compiled a dictionary containing all spelling variants of these actions for syngrep. As RelEx works on sentence bases, a syngrep 3-tuple search with (at least) two entities of interest and an action keyword is a fast and powerful prefilter for RelEx. The standard

settings for syngrep as a prefilter (see last row in table 8.2) are the following:

- 3-tuple sentence based search (2 x entity + action keyword)
- inline abbreviation detection
- case sensitive for hits with length $<= 3$
- excluding hits overlapping with common english words for the entities (but not for actions)
- synfile-context based disambiguation, fly gene names only valid if context also matched.

Table 8.3 gives an overview about the power of the co-occurrence based syngrep prefilter, and the final outcome of a RelEx run.

filter	number of found PubMed abstracts	number of PubMed sentences
abstract tuple(2)	3.221.960	14.674.690
abstract tuple(3)	908.345	6.045.458
sentence tuple(2)	2.935.722	8.496.472
sentence tuple(3)	553.175	975.150
RelEx hits	71925	84850

Table 8.2: Number of sentence protein-protein (-action) co-occurrence hits detected by syngrep. Syngrep can look for tuples on different levels: hits are reported if there are hits found for all sets specified in the tuple definition within the abstract or within one sentence. The number in the filter column (2) or (3) refers to the number of sets used in the tuple definition.
The columns 2 and 3 show the resulting number of positive PubMed abstracts and sentences respectively. As expected, tuple filter based on sentences are more stringent as on abstracts, and similarly, a three-occurrence filter is more stringent than two-occurrence filter. As a result the filter setting for the RelEx run (sentence tuple(3), for settings see section 8.3) can reduce the 355 Mio PubMed sentences to less than one million. As shown in the last row, in more than 90% of them RelEx cannot identify a relation, resulting in less than 90.000 sentences putatively describing biological regulations between the identified entities (in this case genes).

8.4 The best available relation extraction: human annotation

Although RelEx has an appreciable performance for the detection of regulatory relations in general, the characterization of a relation is more error prone and limited to the predefined RelEx schema. In addition, the relations detected by RelEx are binary: a regulator and a regulatee, i.e. it is unable to detect relations where several regulators have to act in concert or the regulation takes place under certain conditions only. These limitations hold not only for RelEx but for most methods extracting regulations, therefore to describe knowledge in a much higher detail usually manual curation/annotation is needed. On the other hand however, the number of publications

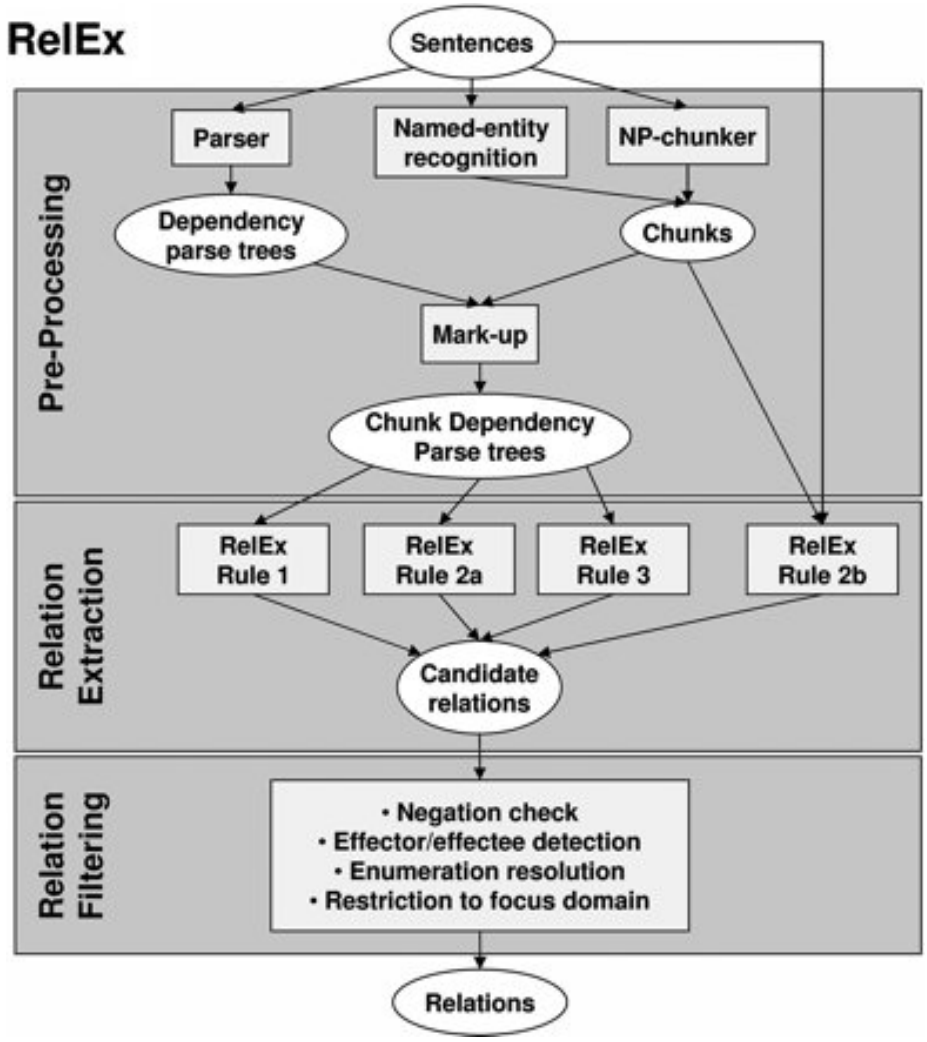


Figure 8.2: The RelEx workflow, figure and legend copied from [61]. The work-flow of RelEx is subdivided into preprocessing, relation extraction and relation filtering leading from the original free-text sentences to directed, qualified relations. Preprocessing is based on publicly available tools and named entity identification. Candidate relations are extracted according to rules applied on chunk dependency trees and original sentences, and subjected to filtering steps.

putatively containing knowledge about biological regulations is huge, and the manual annotation is a time consuming process, so we need tools to focus and accelerate this.
In the course of this thesis we developed a system called `RelAnn` for human annotation targeting these issues. The tool integrates syngrep and RelEx as a starting point for annotation, but allows for editing the entity definitions or adding new entities change the synonyms and re-run the entity recognition or relation detection step. The relations are represented as parametric Petri net transitions optionally saving additional knowledge on evidence (mostly the biological methods leading to the discovery of the relations) for the transition. It also provides many features accel-

erating the annotation such as click-based quick filters to find abstracts containing an annotated entity/relation, or click-based import of a (pre-)annotated entity as an input/output/evidence for a relation. In addition, all entities, relations or relation parameters can be represented as ontologies providing the possibility to annotate more general or specialized relations depending of the available knowledge.
The tool can handle multiple projects and users, and is currently used in different projects, here we give only a short overview of the used features of `RelAnn` as described in the prepared manuscript about the comprehensive annotation of yeast gene regulatory network for diauxic shift (the annotation itself is mainly the work of Ludwig Geistlinger). The focus of this manuscript is to annotate the regulations between transcription factors (TF) and target genes (TG) in the context of diauxic shift depending on environmental or metabolic changes (signals). An application note describing all features of `RelAnn` is in preparation (joint work with Simone Wolf).

Representation of relations in the yeast diauxic-shift GRN study

To represent literature-curated GRIs, we propose discrete regulation models in which discrete states of the regulators (TFs and signals) result in discrete quantity states of the regulated gene (e.g. a low, medium, or high expression) depending on the regulation type. As an instance, we use Petri net models to efficiently represent the information typically available in the literature. Petri net models are well established in informatics (see for review [135]) and have been extensively applied to biochemical processes, like metabolic pathways e.g. [108] and gene regulatory networks reviewed in ([31]). The extension of Petri net models with fuzzy logic [202] in the PNFL approach [197] allows a more detailed semi-quantitative representation of in- and output of the Petri net transitions, which are defined by simple rule sets according to the regulation type [68, 109]. The parametrization of such transcriptional transitions is based on a differential regulation setting, where the presence or absence of a signal induces a reaction of specific transcription factors (e.g. an enhanced or reduced activity), which in turn regulate their TGs differentially (up or down, as compared to the corresponding opposite signal state).
Input The input of a transcriptional transition is composed on the one hand by the signal, which triggers the regulation, and, on the other hand, by the TFs, which perform the actual regulation of the TG under investigation. For example, at the diauxic shift the depletion of glucose (the signal) triggers the derepression of enzymes involved in non-fermentative metabolism (the TGs) by specific TFs.
Signals can be extra- or intracellular messenger molecules (e.g. cAMP), nutritional compositions (e.g. glucose lacking growth medium), environmental and experimental conditions (e.g. high pH or heat stress) and even cellular states (e.g. retrograde regulation depending on the functional state of the mitochondria). In general, the signal determines the context in which the transition is enabled and for most GRIs, a boolean model (presence or absence of the signal) appears to be sufficient to represent the information given in the literature.
The TFs are classified as *up*- or *down*-regulated in discrete quantity states *weak*, *medium* or *strong* and the special states *over-expression* (for *up*) and *knockout* (for *down*).
Output Analogously, TGs are classified as *up*- or *down*-regulated by a certain transition with

a *weak*, *medium* or *strong* effect strength. Intuitively, this models the fold change in the transcription of the TG. According to empirical standards, *weak* regulation refers to expression changes below 2-fold, *medium* between 2- and 5-fold, and *strong* above 5-fold. The transition type results immediately from a given in- and output configuration, e.g. a TF knockout, resulting in a weak up-regulation of the TG, indicates a weak inhibition.

Main features of RelAnn used for the yeast diauxic-shift GRN study

The main design principles of `RelAnn` are:

- Pre-indexing of defined biological entities (genes, proteins, etc.) in the literature
- Simple, click-based annotations to relate the entities to each other
- Representation of relations as Petri net transitions

As illustrated in Figure 8.4, we employed `RelAnn` for the transformation of literature knowledge to the representation of GRIs as semi-quantitative Petri net transitions (as described in the previous Section).
Subsequent to the pre-indexing of the relevant text using a synonym search, occurrences of defined entities are used for the definition of input (regulators, i.e. TFs and signals), output (regulatees, i.e. TGs) and experimental evidence for a regulatory transition. Thus, every part of the transition (regulatory, regulatees, evidence) is linked to some phrase in a scientific article of the `PUBMED` database, thereby making the source of the knowledge traceable. In addition, in- and output specification allow the assignment of the semi-quantitative type of needed (input) or induced (output) change associated to the regulation, i.e. *up* or *down*, with *weak*, *medium*, or *strong* effect strength (bottom right of Figure 8.4b).
A special feature of `RelAnn` is the organization of all components (gene, signal, evidence, regulation and parameter types) in ontologies enabling powerful queries and specifications using generalization and specialization.

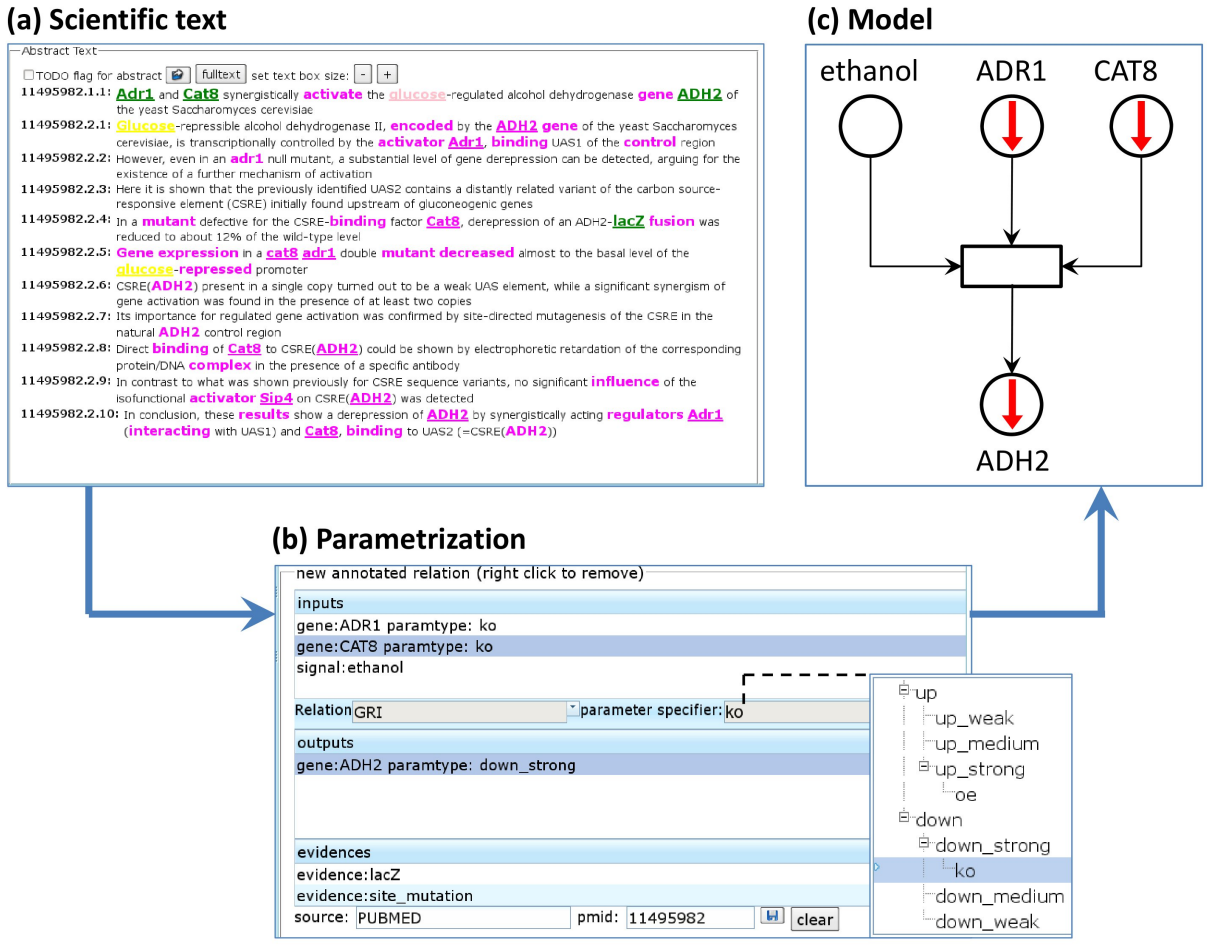


Figure 8.3: From pure text to semi-quantitative models of gene regulatory interactions. Within our annotation framework, the pre-indexed regulatory entities in **(a)** can be easily selected and used for the parametrization in **(b)** of in- and output of the Petri net model of the gene regulatory interaction in **(c)**. In the same manner, experimental evidence is seamlessly assigned to the model.

Chapter 9

SynTree

Textmining is one option to extract knowledge from large collections (e.g. PUBMED) of free texts, either full texts or abstracts from scientific publications. Knowledge described in these texts can be represented by pairwise or higher order relations (transitions) between biological entities in respective contexts.
Of course, such knowledge is also available in structured databases or can be obtained by manual curation. Unfortunately, databases are incomplete and manual curation is very expensive and time-consuming. Thus, for many applications, automatic textmining is the only practical option. Thus, a large body of research focused on the identification of biological entities in free texts (named entities recognition, NER) and the extraction of relations. In our lab several such tools have been developed (Prominer, [60, 82, 83], RelEx [61, 63])
We have also developed syngrep as a software tool, which is able to rapidly scan large text collections, e.g. all the about 20 million PUBMED abstracts within minutes on a single CPU. Thus, textmining with syngrep is routinely applicable to real-world applications, e.g. in pharmaceutical companies and research.
This chapter reconsiders the named entity recognition problem discussed in chapter 8 and proposes a more context aware and context dependent approach. The context sensitivity complicates the problem and the possible solutions. But still, the goal is to end up with efficient and practical methods to address it. The main idea of exploiting context information (the whole sentence, paragraph, section, etc.) of the terms matched in the synonym dictionary is not new, it is employed in several machine learning approaches. Here we present a new method called SynTree: it uses the containing text as an information source, but the real context for the identification and disambiguation comes from simple rules and from an ontology describing the domain of the biological entities to be recognized. This however does not come for free: the rules have to be defined manually (but are of very simple structure). While it seems to be a disadvantage over machine learning techniques, one has to keep in mind that for machine learning techniques a large number of texts have to be annotated - simply to obtain a corpus where the method can learn on. In the process of annotation the human curator usually can derive generalized rules - the input needed for the presented method SynTree. In this sense, the required rule definition for SynTree is only a higher-level annotation of the texts.
In fact an expert in some research field can even define rules based on his knowledge without any

annotation. In our view, the transparency of the decisions made by the method is more important than being fully automated (after having annotated the large corpus) - as thereby possibly wrong or missing rules can be changed/added accordingly. Correspondingly, we propose an iterative way to check, correct and add such rules on a large corpus in this chapter.
The chapter is mainly based on a submitted manuscript and is joint work with Simone Wolf. In her Master's thesis she has investigated psychiatric diseases and collected the context knowledge for them - so she became a pilot user for Syntree. For the definition of the ontology for psychiatric disorders she had the support of the research group at the Max-Planck-Institute for Psychiatry in Munich: for this we thank Prof. Chris Turck, Philipp Gormans, and MD Claudia Ditzen. Accordingly, I developed tools to create and change the ontology and rules and designed and implemented the method, she filled these structures with the psychiatric disease specific knowledge. Similarly, for the evaluation I provided tools (within the RelAnn framework presented in chapter 8) for efficient annotation of the identified entities by the evaluated methods, and she used these to perform the annotation work for the psychiatric disease ontology.

9.1 Introduction

Exploiting the large wealth of scientific knowledge represented in the scientific literature automatic extraction methods are required to make it applicable for a wide range of applications from knowledge and data mining to network based interpretation of high-throughput data via involved bioinformatics methods. Often text mining is used for the extraction and the formal representation of facts and relations on biological entities such as genes, proteins, metabolites, drugs, species, experimental conditions, cell lines, and diseases. One of the first and often crucial steps in text mining involves the identification of the relevant biological entities from the free texts. For most of these entities large nomenclatures and ontologies are in use resulting in an even larger set of terms used in scientific descriptions. Often, terms and their usage in scientific texts are highly ambiguous and overlapping with other named entities as well as common natural language terms. in many cases terms can only be resolved to unique named entities of the correct domain area if the context of the text is considered. There are relatively few approaches which deal with the full complexity of context-dependent ambiguity resolution. One type of approaches relies on general machine learning methods able to deal with context-dependencies such as conditional random fields (CRFs, for an introduction see [174]). The other type tries to exploit explicit representations of the domain ontology and explicit rules for the definition and/or exclusion of terms as named entities.

In this paper, we introduce a new explicit method called SynTree. SynTree is based on explicit domain ontologies named entity trees (so called NE-trees) and explicit human defined rules together defining the context-dependent named entities. It tries to simulate the entity recognition of human experts during reading a scientific text: Based on expert knowledge (and maybe access to a domain ontology), candidate named entity occurrences are evaluated in the context of the overall text and either accepted as a valid named entity, discarded as something else, or kept for further identification after more careful reading of the text. SynTree therefore uses the domain ontology and a set of context-dependent rules and applies them iteratively in the dynamic context defined by the already found entities in the current text until no new named entities are identified. The approach employs hierarchically defined entities, dynamic contexts and possibly scopes of these contexts for disambiguation of occurrences.

SynTree is very fast and nevertheless can identify the named entities in a complex domain with both high sensitivity and specificity. The paper describes the SynTree method, describes a large annotated ontology for psychiatric disorders and associated rules, introduces a detailed benchmarking protocol for NER performance assessment, and uses it for the in-depth performance analysis of SynTree and some competing methods. The NE-tree for psychiatric disorders has been developed with experts from the MPG Psychiatry in Munich. It serves as an example for the advantages of the explicit SynTree text mining, is available for further work in psychiatry research and as a prototype for the construction of more NE-trees for other domains.

The approach tries to mimic human behavior in contrast to implicit machine learning approaches. This is also reflected by our training and benchmarking protocol. Relevant texts are annotated by human experts and the NE-tree on psychiatric disorders is extended by context and scope information and maybe additional rules are introduced. Text mining with SynTree using the refined NE-tree and rules results in new sets of texts with identified and approved occurrences of NEs.

These sets can iteratively be used to define new annotations and rules. Of course, these annotated sets can also be used to train and re-train machine learning models such as CRMs. We use the two context-dependent ML methods ABNER and STANFORD in our benchmark protocol to compare the performances of implicit and explicit approaches with respect to correctness and speed.
Overall, SynTree can identify occurrences of terms describing psychiatric disorders and assign named entities to them in PubMed abstracts at estimated specificity and sensitivity of both over 90% outperforming context-dependent ML methods. SynTree can process the complete PubMed abstracts in about 1.5 CPU hours and, thus, is about 50 times faster than competing approaches.

9.1.1 Background on psychiatric diseases

Psychiatric disorders are an active research area in medicine. Since ancient times abnormalities in behavior and thinking in patients are studied with the aim to find or improve the diagnosis and treatment. Psychiatry as a serious research focus in biology started in the 20th century by Emil Kraeppelin and Alois Alzheimer [93]. During this time the idea of biological causes for the outbreak of disturbances in the psyche took roots. The pathobiology of psychiatric disorders today is still poorly understood. Problems in understanding the mechanisms of mental disorders result from the wide variety of symptoms, a high comorbidity and the high fraction of subjective judgments in the course of diagnosis. Diagnosing in psychiatry is done by clustering the signs, symptoms and course of the acute syndrome in self assessments and doctor-patient sessions [46]. A more precise categorizing and distinction of psychiatric disorders is not possible because no molecular biomarkers and only very few medical tests are available [169]. Hence the classification of a syndrome is mainly based on the experience of the medical doctor. Major efforts have been made to unify the classification in order to be able to achieve some standards in the diagnosis. Two main systems are used for classification: *Chapter V of the International Classification of Diseases (ICD-10)* [2] provided by the World Health Organization (WHO) and the *Diagnostic and Statistical Manual of Mental Disorders (DSM-IV)* [10] created by the American Psychiatric Association (APA). Both systems try to dissect diseases whereas the ICD-10 is more a categorical approach and the DSM-IV follows the dimensional definition in psychiatry.

The ultimate goal of the approach presented here is to automatically detect the occurrences of psychiatric diseases in biomedical publications and identify their subtype. Given a fast method able to identify psychiatric disorders in large databases like PubMed, one can move to the challenging problem to find possible indicators (biomarkers) exclusively for one disease subtype. Such a biomarker could help to improve diagnostics [169], have a personalized treatment plan later on and it would be a possible drug target as well.
In this work we (i) introduce a general context-dependent named entity recognition (NER) approach and (ii) apply and evaluate it on the psychological disease detection problem. For the application we focus on dementia, schizophrenia, mood disorders and phobic disorders.
For these four diseases we created a hierarchy describing the more and more specific subtypes. We applied mainly the classifications provided by ICD-10 schemata, the categorizations described in ”Psychiatrie systematisch” [46] as well as the expertise of a medical doctor. This

hierarchy contains terms used for describing the diseases as well as a binary characterization of the terms, either ”defining” or ”context-dependent”. Defining terms are unique for a given disease, i.e. wherever such a term occurs it is clear which disease is meant. Context dependent terms in contrast are not unique or general terms which may describe a disease but only if they occur in the right context. This hierarchy and classification serves as main input for the presented approach.

9.1.2 Related work

For extracting relations between proteins, genes or other entities with disease entities co-occurrence analysis of named entities often is the method of choice as computer linguistic and natural language processing methods need a lot of customizations and resources [53]. Anyway, NER is the basis of most text mining methods, and, thus many approaches have been developed for it, but NER for diseases has attracted less attention than NER for genes and proteins.

Literature mining is used to derive context knowledge for the interpretation of experimental (high throughput) data [97]. In studies of psychiatric disorders like major depressive disorder so far no clear candidate genes/molecular biomarkers [29, 97, 169] have been found. Burmeister et al. [29] gives an detailed overview for psychiatric disorders and the different approaches for figuring out the genetics behind those complex disorders including linkages analysis, GWAS etc. with and without additional text mining.

There is a plethora of text mining approaches and systems exploiting NER, which can roughly be can be classified into three main categories: rule-based, machine learning-based and hybrid solutions. [3, 107]. Often their problem is not the detection of possible candidates for named entities but the so-called disambiguation between different possible candidate entities.

Hettne et al. [87] improve their constructed synonym dictionary in terms of disambiguation and completeness by applying rules similar to the rules used in SynTree. The 'Miscellaneous rule' is similar to our negative prefix sets on the global level (for all dictionary entries). SynTree allows to define such rules for each entry/node/subset of the named entity ontology with different (dynamically validated) scopes in order to make SynTree more flexible and to reduce FNs and FPs.

Chiticariu et al. [3] investigate whether explicit, rule-based NER systems can compete with state-of-the art, implicit machine learning (ML) methods. The developed NER rule language (NERL) is competitive with ML methods but only after involved customization for the three corpora they used. They also rely on globally defined rules similar to the rules used by Hettne et al.[87]. Chiticariu et al. [3] state the usefulness of rule-based systems because of their transparency and relativly small number of operations needed for acceptable results even if they are limited for a specific domain. SynTree has the advantage that it is very fast, thus customizing rules and modifiers e.g. changing the scope or add/delete/move entries/sets can efficiently be done in an iterative manner.

Another study (Jimeno et al. [98]) evaluated three NER methods (statistcal-based, dictionary-based and MetaMap [11]) for disease entities and observed that the dictionary-based system outperformed the other two methods in terms of F-measure.

9.2 Materials and methods

9.2.1 Basic concepts in SynTree

SynTree is applied to a collection of texts. A text can be any sequence of sentences such as a PubMed abstract. An occurrence of a term at a particular text position is in several scopes such as a sentence or the abstract and has two more associated scopes the prefix and postfix of the term occurrence at the respective position. In addition to the text, SynTree uses two more inputs: a) the named entity tree (NE-tree) defining the entities of the domain under consideration, e.g. (subsets of) genes, proteins, or diseases, e.g. psychiatric disorders and b) a set of rules modifying the entities of term occurrences. In such an NE-tree nodes correspond to both contexts and named entities and have associated lists of synonyms: (unconditional) synonyms, which define an context/NE and context-dependent syns, which are only indicative of the context/NE iff the context of the occurrence of the synonym is valid. The nodes can also be annotated with a context scope (ABSTRACT by default, i.e. the whole text, or SENTENCE, or PREFIX, or SUFFIX)
For the construction of the NE-tree for four main psychiatric disorders we define subsets based on the ICD-10 categorization and the expertise of a medical doctor. An example section of an NE-tree corresponding to the disease 'mood disorder' of the psychiatric disorder ontology is shown in Figure 9.1. Mood disorder's most prominent symptom is the shift of mood in either depressive or manic direction. The top categories are unipolar (shift of mood only in one direction) and bipolar (mood shifts in both directions) disorders. In all subcategories synonyms describing mania and depression need to be included. The example highlights two major issues of NER approaches:

(i) entity groups can be nested in each other

(ii) terms may be context-dependent.

An example for (i) is that the term "mood disorder" is valid both for unipolar and bipolar disorders. If a NER approach simply uses a dictionary of valid terms for every entity "mood disorder" has to occur in NE ontology for both terms. Once such a term occurs the hit is ambiguous and it is unclear which entity is meant. This issue can be resolved trivially by encoding the dictionary in a hierarchical manner as shown in Figure 9.1. It simply encodes the valid "is a subtype of" type relationship.
The context-dependent term issue (ii) also leads to ambiguous or false positive hits. Terms such as "depressive" or "depression" are only valid if they occur in a psychiatric disease context. A false positive occurrence would be for example "depression of signal level x". The disease meant by "depression" **depends** on a much more restricted context: it could be bipolar or unipolar depression and a corresponding contexts needs to be established first in order for the term to be a valid disease NE.
SynTree consists of two layers. The first layer is the input synonym tree and its set of modifiers for each node desribed above. Furthermore the first layer includes rules needed for a correct context matching (further details see section 9.2.2). The second layer is a rule processor system which allows to compile simple human defined rules into modifier sets. The rules used for the psychiatric disease NER are listed in the Appendix.

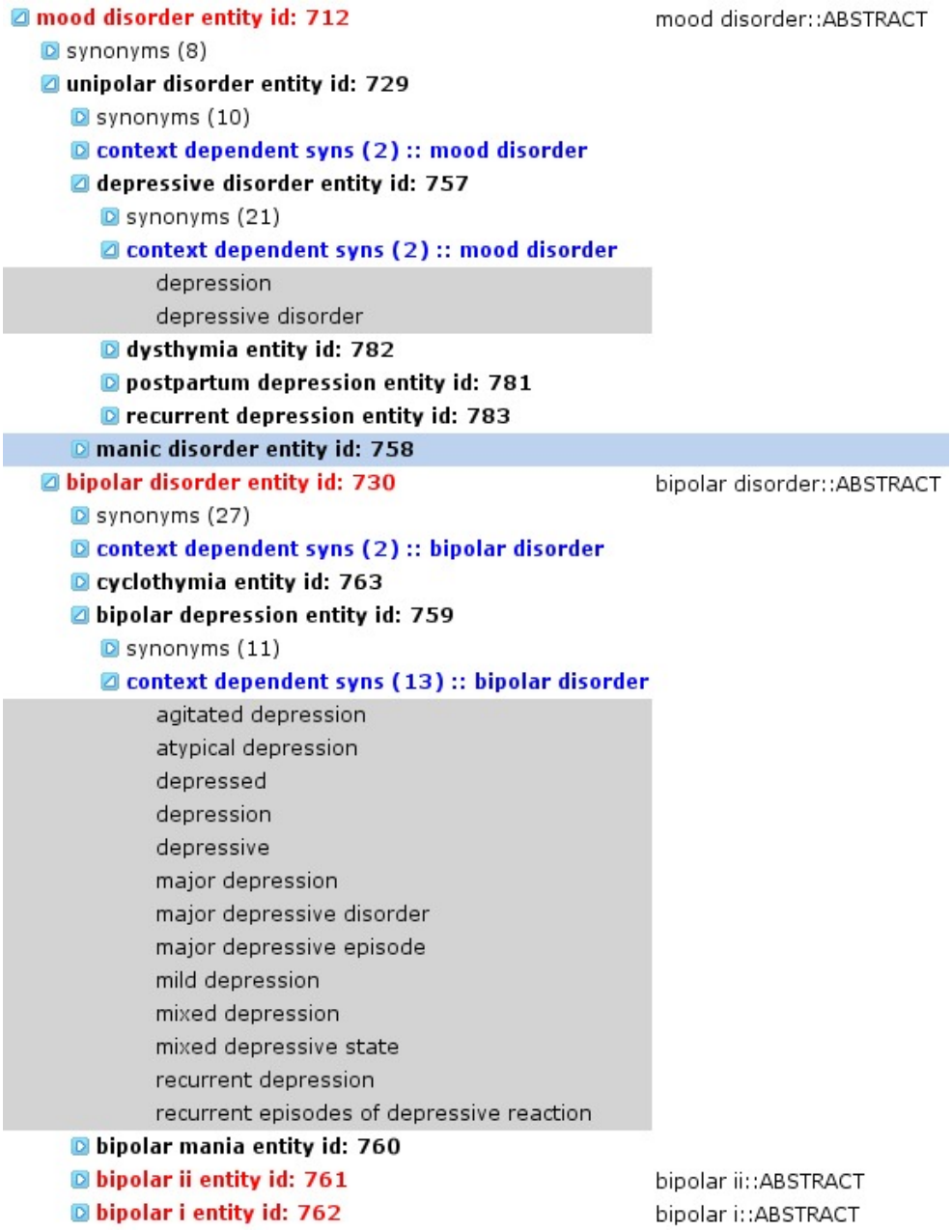


Figure 9.1: Mood disorder hierarchy. Ambiguous synonyms (depression, depressive disorder) which appear in multiple disease subtypes are define as context-dependent (gray highlighted).

SynTree uses an iterative named entity identification procedure which is based on dynamically defined contexts and context-dependent rules. The idea is illustrated in Figure 9.2. Starting with an initial matching of potential named entity occurrences and modifiers, SynTree iteratively assigns validated named entities based on the current context defined by the already identified NEs, Invalidated candidates are removed, all the other candidates are kept (Figure 9.2 (1)-(3)).

SynTree stops (Figure 9.2 (3)) if no new NE could be assigned and, thus, the context remains the same. All the remaining candidates are discarded.

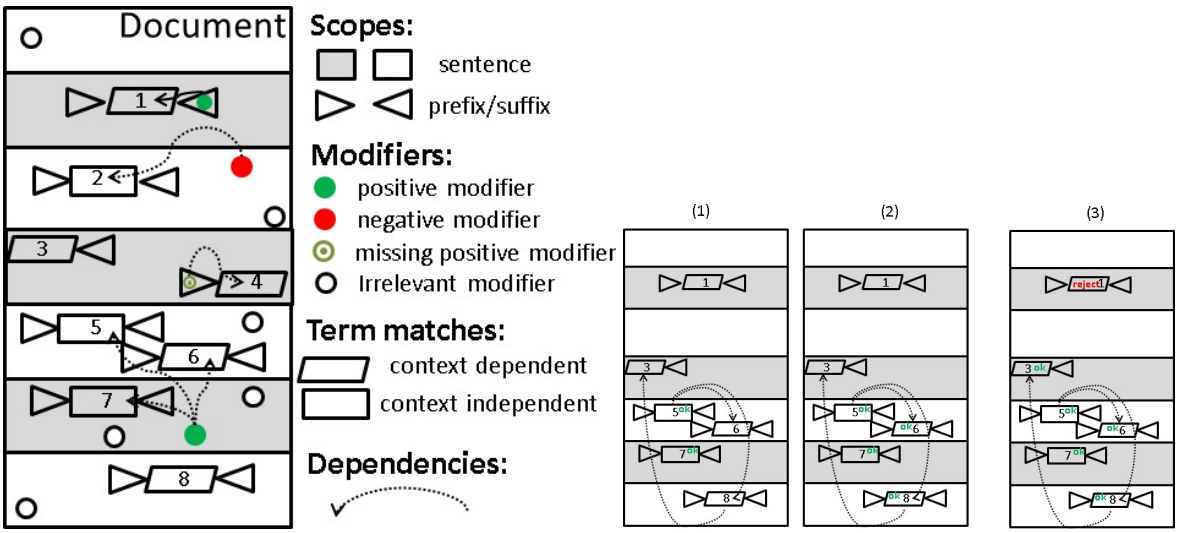


Figure 9.2: Illustration of the SynTree modifier and context evaluation strategy. Syntree uses the synonyms in the ontology to match potential NEs and modifiers in the text, thus hits of respective types are identified together with there scopes and (dynamic) dependencies. SynTree iteratively assigns new contexts and tries to validate candidates within the new contexts and scopes, e.g. in step (1) the candidates 2 and 4 are invalidated due to a (-) modifier and a missing (+) modifier (and removed), candidates 5 and 7 are validated in the current context, the others are kept open. In the next rounds, in step (2) 6 and 8 are validated (via the dependent context 5) and, finally, in step (3) named entity 3 (via the new context 8), whereas the open occurrence 1 finally becomes invalid (due to missing contexts) and discarded. SynTree then terminates as no new contexts have been assigned.

9.2.2 Input tree and modifier set definitions

The following definitions describe the internal structure of SynTree. For the rule and context definitions we use the SynTree Rule Processor described in section 9.2.3 which compiles human defined general rules into the internal representation. The modular implementation makes it (fairly) easy to define other rules in addition to the ones available in the current implementation of SynTree.
The SynTree algorithm takes a tree structure of entity nodes as input. Each entity node has following properties:

1. a (optionally empty) list of identifiers
2. a (optionally empty) list of synonyms
3. a (optionally empty) list of Entity Nodes (children)

4. a list of modifier sets of any subset of the types defined in table 9.2.

In Figure 9.1 "mood disorder" is an entity node with entity id 712 (NE=712) and two child entity nodes (unipolar disorder, bipolar disorder). The modifier sets with their scope are shown on the righthand side.

Psychiatric disorder input tree:

The disease dictionary has four main nodes representing the four main diseases (schizophrenia, dementia, mood disorder and anxiety disorder). For each of those four disease we created a hierarchical structure (up to four levels depth) in order to represent the subtypes properly. Dementia is the largest tree in our data with 51 subtypes and 985 synonyms. This overrepresentation of dementia is due to the wide variety of causes for this illness. Phobic anxiety disorder is the smallest part in the hierarchy. It comprises 6 disease subtypes with together 149 synonyms. An overview on the numbers of disease subtypes, defined contexts and the number of synonyms is given in table 9.1.

disease	#subtype	#cont	#syn
Mood disorder	20	5	514
Schizophrenia	7	2	171
Phobic disorder	7	4	149
Dementia	52	21	985
total	86	32	1819

Table 9.1: Overview of disease subtypes and synonyms in the final psychiatric disorder ontology.

Modifier sets and scope control on the validity of synonym hits:

All modifier sets can be defined for positive (+) as well as for negative (-) matches. For a given scope SC and modifier term set list $s_1, \ldots, s_n$ we treat s_i as a boolean function which becomes true if at least one term in s_i is found in context SC.
For (+) modifier set a match of a synonym is valid if $\bigwedge_{s_i}$ is true, for a (-) modifier set only $\bigvee_{\neg s_i}$ has to be true. In other words: for (+) modifier sets at least one term has to be matched from each defined set in the modifier set, for (-) modifier set a synonym hit is only invalid if at least one term is matched in one of all defined sets in the modifier set. The PREFIX and SUFFIX modifier sets can consist only of a single set - simply because there is no possibility to match multiple terms from different sets immediately in front/after synonym hit.
As an example for the usage of (+) multiple modifiers consider the following setup: we look for publications containing knowledge about psychiatric disorder due to some metabolic disorder in childhood. Assuming we compiled the term sets describing the context for childhood (for example: s_1 =(child, children, childhood, infant, ...) and for metabolic disorder s_2=(metabolic, ...) and we have a synonym list for one psychiatric disorder. We can create a positive (+) modifier set for the abstract context: (s_1, s_2) applied on the synonym list. Thus, occurrences of psychiatric disorders are only valid if both the childhood-terms and the metabolic-terms are found in the

same context (e.g. ABSTRACT or SENTENCE). If we additionally want to discard hits of some specific disorder (e.g. anxiety) and the context of alcohol consumption we can define term sets ns_1, ns_2 regarding these two contexts and create a negative (-) modifier set using these sets. If either one term of the two sets is found the hit becomes invalid.

Inheritance in SynTree:

The hierarchical order of the nodes in SynTree offers two advantages. Firstly, we can hierarchically define entities consisting of several more specialized entities. This is even more an advantage for entities like genes or proteins. For example an abstract might contain knowledge about the 'Ribonuclease H' protein. There are however eight human proteins in SWISSPROT, seven of which (starting with 'POK') referring to "provirus ancestral Pol protein" and one "provirus ancestral Gag protein" having this as a descriptive name (POK5, POK6, POK7, POK10, POK12, POK17, POK20, GAK2). So the hit 'Ribonuclease H' is highly ambiguous but if encoded in a hierarchical structure it is still clear that we can extract knowledge about "provirus ancestral proteins" in general. In addition, we could make distinct between Pol and Gag proteins by using the corresponding contexts.
This way the tree structure helps in disambiguation in general: if the term for a hit can be found in several entity nodes we can look for a more general entity node having all these nodes as descendants. SynTree can assign the entity corresponding to this general node for this hit rather than report one or all possible entities. Moreover, if more specified contexts are given SynTree can resolve the ambiguity more precisely.
Secondly, the tree structure allows for inheritance and redefinition of properties: contexts and rules are simply transferred from parent to child entity nodes except for the contexts defined in the child entity itself. This way we can easily define general and scoped contexts and rules which should be applied on a large part of the hits (corresponding to subtrees of the ontology), but change them specifically for some cases. For example, the first two rules in the rule list in section 9.5 are generally applied on all hits, but are extended with rule 5 for the synonym "depressed".

Nodes with empty lists of identifiers/NEs:

For a matched synonym in the text SynTree reports the identifier/id/NE of the entity node where the synonym is listed. However, as defined in section 9.2.2 the identifier list of a node may be empty. If so, SynTree searches higher up in the hierarchy until a node with specified NEs are found. This feature allows for specific context requirements for synonyms of the same entity. The idea behind this feature is simple and explained in Figure 9.1: there may be specific synonyms not needing any further contexts as well as synonyms which are only valid in specific contexts. One example is the term "depressed", which could describe the disease, but is used in many different contexts including general english language. Empty NE lists for nodes and inheritance of contexts along the tree allows to resolve ambiguities (e.g. rule 1, 2 and 5).

scope	sets	(+) rule	(-) rule
prefix	s_1	s_1	$\neg s_i$
suffix	s_1	s_1	$\neg s_i$
sentence	$s_1 \dots s_n$	$\bigwedge_{s_i}$	$\bigvee_{\neg s_i}$
abstract	$s_1 \dots s_n$	$\bigwedge_{s_i}$	$\bigvee_{\neg s_i}$

Table 9.2: Modifier set types. SynTree uses four scopes (rows in the table). All of them can be used both as positive and negative constraint. The cells in the table specify the validity of a list of set, by defining the corresponding boolean variables s_i. A variable s_i is true if and only if at least one term of the set i is matched in the given context.

Nodes with an empty list of synonyms:

Similarly, nodes having child nodes may have no synonyms. This implies, that matches for them can only happen due to ambiguity resolution from the child nodes as described in section 9.2.2.

Defining modifier sets:

The rules described in section 9.2.2 use term sets to decide if a given context is valid. To make the definition of such term sets more intuitive and systematic, SynTree offers different operations on (elementary) modifier sets leading to new (combined) modifier sets. The elementary modifier sets can be either (i) a list of terms, or (ii) all valid hits of an entity node. For (ii) SynTree will apply the rules specified for the entity node and will only report the term matched if these are satisfied. Furthermore, to create a new modifier set one can use the existing sets and add or subtract operations, terms, or scopes.

9.2.3 The SynTree rule processor

The modifier sets and the hierarchical order of the synonyms enable a very flexible configuration of SynTree by the user. On the other side one would like to avoid the need to check which modifier set should apply on which synonym. The preferred way to define the rules for synonyms is to (i) define the general characteristics of the synonyms and (ii) specify the additional rule needed.
For (i) the characteristics are something like: “context-dependent synonyms of the node X” or “synonyms that are common english words” or “synonyms containing some term” or any combinations. The additional rule (ii) specification is something like: extend or subtract some terms from the modifier set active on the corresponding node or make the rule more specialized by adding a new set to the modifier set. It also offers possibilities to easily specify the sets of nodes where it should be applied: (i) one can simply give a list of node names or (ii) specify whether only the nodes containing context-dependent synonyms should be targeted.
The SynTree rule processor takes care of this and translates convenient rules into modifier sets for the subsequent hit finding using ACT matching. During the compilation the rule processor introduces new entity nodes and modifier sets and assigns modifiers set to existing nodes in the

Method	run time	factor
SynTree	1:22:13.132	1
STANFORD	61:21:47.390	47
ABNER	385:05:26.589	293

Table 9.3: Run times for each used methods to process all PUBMED abstracts (20 Mio abstracts, 340 Mio sentences, 24 Gb text) on a single 2.6 GHz processor. The times are "user" times without the time needed for I/O operations. Depending on the number of occurrences and the reporting level, I/O time can be considerable, but using parallelized I/O the SynTree runtime remains tractable at about 2h CPU time.

tree.

9.2.4 Matching rules in SynTree

The match engine (ACT) in SynTree is based on a modified Aho-Corasick/PATRICIA trie [36, 132] (see also chapter 8 which makes processing of large text corpora very fast (see Table 9.3). The algorithm uses semi-exact matching: we turn the alphabet into a smaller one (containing only small case english letters and numbers and whitespace) and map every capital letter to it's non-capital variant, and every special character to whitespace.
First, the input text is matched against a trie containing the synonyms. If there are matches the terms in the corresponding modifier sets are matched. At the synonym matching step we may have additional rules specifying which special character matches are allowed. SynTree performs additionally an inline-abbreviation resolution: whenever a synonym is matched and a '(' character follows it checks if the words in the parenthesis may be an in-place defined abbreviation for the match. If this is the case it dynamically defines the abbreviation as a synonym for the corresponding node and applies rules defined on this node on further occurrences of the abbreviation within one abstract. For the check if it is an abbreviation we have restrictive rules: the synonym term has to consist of multiple words, the abbreviation must not contain numbers and must contain at least one capital letter.

9.2.5 Benchmark sets

As to the best of our knowledge there is no benchmark set for psychiatric disorder named entity recognition we compiled one for SynTree. This benchmark should also serve as a compilation of knowledge for PDs. Towards this end, we want to examine the applicability of SynTree in an iterative process of human learning consisting of two subsequent steps pro iteration: (1) specifying (additional) rules and synonyms and (2) apply SynTree and assessing the results.
In order to compare SynTree to other context-sensitive textmining approaches we used two additional methods: A Biomedical Named Entity Recognizer (ABNER) [164] and the Stanford Named Entity Recognizer (STANFORD) [96]. Both methods are machine learning methods and based on conditional random fields [174]. Therefore our benchmarking will compare the two main approaches of biomedical text mining: (i) machine learning trained on examples and (ii)

human knowledge driven (based on entity synonyms or ontologies). The benchmark set will be extended iteratively, such that we can analyze the knowledge gain obtained via increasing the training set size against more derived human rules for context dependency.
For all iterations we annotated to all occurrences of psychiatric disease entities the exact position (sentence, start and end in the sentence) and the corresponding disease identifier. This annotation serves also as input for the machine learning methods, i.e. ABNER and STANFORD are trained on more and more training data in each iteration. The "human learning" for SynTree is done by changing or extending rules or new synonyms (for details see section 9.3). In addition we also performed two "baseline" predictions based on the synonym dictionary of SynTree only: (i) a specific one, where only non-context specific synonyms are included and (ii) where all synonyms (but without context checking) are included). We evaluate the predictions on two levels: (i) as a binary (2-class) prediction, i.e. whether the methods can correctly detect the occurrence of a disease term, and (ii) as a multi-class (NER) prediction, i.e. the methods have to identify the mentioned disease via the correct named entity. Our main application is application is multi-class NER prediction. Unfortunately, for the multi-class prediction STANFORD could only be trained with small amount of annotations (it reported a non-sufficient memory error on our 8 GB machines), so we evaluated all methods on the 2-class problem and compared only ABNER and SynTree for the real NER application.
In summary, we annotated a total of 1615 abstracts (1415 and 200 abstracts used for the final training and test, respectively) from which 815 contained at least one occurrence of a psychiatric disease (positives) and 800 negatives. The 815 positive abstracts contain in total 10552 named entity disease (psychiatric disorders) occurrences. The disease ontology for psychiatric disorders contains 86 different diseases of which 79 actually occur in our 1615 annotated abstracts.

9.2.6 Benchmark structure and measures

To evaluate the performance of SynTree, Abner and STANFORD we use well known measures: true positives (TP), false positives (FP) and false negatives (FN) and the derived measures precision (Pr), recall (R) and F-measure (F):

$$Pr = \frac{TP}{TP + FP} \qquad R = \frac{TP}{FN + TP} \qquad F = \frac{2 * Pr * R}{Pr + R} \tag{9.1}$$

.
As we precisely annotate the exact position of each disease occurrence in the abstracts we can classify the predictions based on their positions. In order to tolerate shorter/longer terms (compared to the annotation) reported by the methods we call a hit positive if it overlaps with only one annotated occurrence (2-class) and if the reported named entity is the same as the annotated one (NER).

9.2.7 Iterative benchmark procedure

For the evaluation in the construction of a comprehensive knowledge base we performed iterations simulating the learning process. We start with a manually selected and annotated set

set	SynTree	Abner	STANFORD
I(1)	0.96	0.72	**0.97**
I(2)	**0.67**	0.02	0.01
I(3,E)	**0.79**	0.04	0.00
I(3,A)	**0.94**	0.81	0.86
I(3,C)	**0.81 (0.93)**	0.74	0.66
I(4)	**0.90**	0.47	0.47

Table 9.4: The table show the F-measure on the different sets if evaluating the two-class prediction (predicting if a term corresponds to a psychiatric disease). For the descriptions of the sets see 9.3. On the easier sets the machine learning approaches perform closer to the SynTree performance, but - except for the I(1) set where STANFORD wins by 0.01 - SynTree outperforms both methods. On the sets where the predictions of the methods do not overlap (I(2) and I(3,E)) the margin becomes great: if SynTree does not report any disease in the abstract, there are virtually no true positive predictions reported by the ML methods. Additional modifier rules can drastically improve the performance, e.g. on the I(3,C) set from 0.81 to 0.93 F-measure.

of abstracts as the initial knowledge base. In the iteration, we use a given knowledge base to predict (with our three methods) disease occurrences in the PubMed and extend the knowledge base. The extension differs between human and machine learning: For the machine learning approaches we simply use the larger set of annotations to re-train them. For human learning we encode the gained knowledge in the SynTree tree and rules. In the Results section we describe the iteration step, how we transform the new insights into Syntree, and the observed results.

9.3 Results

The results for all methods on all sets used in the iterations are shown in tables 9.4 and 9.5. For the final training and test set (I(4), last row) of our psychiatric disorder corpus (1415/200 abstracts, 86 disease entities) SynTree achieves an combined precision/recall (F-measure) performance of 0.9 and 0.86 for the 2-class and NER problem, respectively, thereby outperforming the context-sensitive machine learning approaches by a very large margin (0.47 and 0.29, respectively).

Iteration procedure

First Iteration I(0) (initial annotated set): 115 abstracts with occurrences of PD named entity occurrences were selected form the PubMed and annotated with NERs at respective text positions. This initial set was used as the first training set for the machine learning approaches ABNER and STANFORD. We run them against the whole PubMed and analyzed the predicted annotations in the 115 abstracts. Additionally, we selected 100 putative negative abstracts from the predictions, annotated them and re-trained the machine learning methods to correct the most

set	SynTree	Abner	SynTree final
I(1)	0.90	0.62	0.89
I(2)	0.64	0.02	0.86
I(3,E)	0.77	0.03	0.79
I(3,A)	0.76	0.71	0.75
I(3,C)	0.69	0.65	0.74
I(4)	0.86	0.29	0.86

Table 9.5: Performance of recognizing the correct psychiatric disease type. Here we can only evaluate Abner and SynTree. The column SynTree shows the results with the knowledge encoded in the synonyms and rules at the time the set was first analyzed (corresponds to Abner performances as Abner was trained with all previously existing annotations for each set), "SynTree final" shows the performance of the final version of synonyms and rules. The table shows the results on all sets (see 9.3. On the clearly positive abstracts sets I(3,A), I(3,C) Abner and SynTree perform similarly, while on the harder sets I(2), I(3,E), I(4) and the manual constructed set I(1) SynTree clearly outperforms Abner.

prominent false positive predictions.

Second Iteration I(1) (test set): select another 100 positive and 100 negative abstracts.

Third Iteration I(2) (checking mutual exclusive hits): As we are interested in constructing a comprehensive knowledge base (ontology and set of abstracts) for PD, we do not stop here. Instead we exploit the difference and intersection sets of predicted abstracts from the three methods (Figure 9.3(a). Set I(2) is defined by selecting 300 abstracts, 100 abstracts from the sets predicted by one method only. Here, the 100 abstracts are taken from a ranked list of abstracts with the largest number of NE predictions. This results in a difficult benchmark set as the only way to perform well on such a set for a given method is that the other methods have many false positive predictions and the method itself achieves almost only true positives.
As the corresponding row (I(2)) in the 2-class evaluation table 9.4 shows, this is almost the case here: STANFORD and ABNER report virtually no positive hits while SynTree has a performance of F=0.67. The (relatively poor) performance of SynTree comes from a low precision (57.6) and a good recall (96.6) (data not shown). We analyzed the annotations leading to the low precision, which was caused by missing contexts for ubiquitous synonyms such as *"depression"* and *"bipolar"*. We incorporated this information in our NE-tree by checking and reclassifying our candidate context-dependent synonyms and extended the global negative suffix list by some terms (e.g. *"index", "diagnostic"*).

Fourth Iteration I(3,E) , I(3,A), I(3,C) (checking mutual exclusive and overlapping hits): For the fourth iteration we re-trained the machine learning approaches with all available annotations (515 abstracts) and applied the changed rules from the previous iterations for SynTree. We

then applied all methods again on PubMed resulting in about 1.1 Mio abstracts with at least one prediction by one of the methods (see Figure 9.3(b)).
In this iteration we analyzed 100 abstracts for all seven intersections (by selecting the abstracts with the most predictions again). We created three sets from these 700 abstracts: I(3,E), the set of 300 abstracts containing only abstracts predicted by a single method (similarly to I(2)), I(3,A), the 100 abstracts predicted by all methods consistently, and I(3,C), the complete set of the 700 abstracts. The results for these three sets are shown in Table 9.4. Abstracts predicted by all three methods (set I(3,A) are correctly predicted, and all methods have a reasonable 2-class performance, where SynTree and STANFORD perform similarly with 0.96/0.97 F-measure. Abstracts not predicted by SynTree are nearly all incorrect as the results on set I(3,E) indicate: STANFORD and ABNER have performance close to zero, i.e. they identify almost no true positives, whereas SynTree still achieves a performance of almost 0.8 F-measure. On the complete set of 700 abstract (I(3,C)) the performance of SynTree is much lower (about 0.80 F-measure) than on the I(3,A) set (about 0.94 F-measure). This results from many false negatives from the set where both STANFORD and ABNER predicted hits. After taking a closer look on the those hits, we found that more than 99% (1742 annotations out of 1752) are variants of the term “depression” or “anxious” (i.e. “depression”, “depressed”, “depressive” etc, 12 distinct terms altogether). The reason for not recognizing these terms is that we classified these terms as context-dependent in the last iteration. Unfortunately, SynTree found no other context for more specific disease descriptions in these abstracts. To solve this problem we added two positive modifiers context sets for the nodes ”depressive disorder” and ”phobic anxiety disorder” as all 12 distinct terms belong to one of the two. One modifier describes a ”patient” context at the SENTENCE level, the second hints to a disease context (e.g. “symptom”, “illness”) at the SUFFIX level. The 2-class performance of SynTree jumps from 0.81 to 0.93 F-measure on I(3,C) while the performance on I(3,A) and I(3,E) remains the same (Table 9.4).

Fifth Iteration I(4) (final validation): As validation for the overall system we applied ABNER trained on 1415 abstracts and our final SynTree configuration to all PubMed abstracts. Again we selected 200 abstracts for the assessment: we ignored occurrences of the terms ”depression“ and ”anxious“, ranked the abstracts found by SynTree and ABNER and selected the 200 top-ranked ones (those with the most occurrences). Thereby we obtained a set of abstracts with a more diverse set of named entity diseases. STANFORD was excluded here for this selection, but also evaluated on the selected 200 abstracts.
As shown in Figure 9.3(c) most of the abstracts predicted by SynTree are also predicted by ABNER, but ABNER predicted more than 370.000 additional abstracts. Almost all of this 370.000 ABNER-predicted abstracts are false positives (see Table 9.6), as SynTree has only 3 false negative abstracts. Moreover, in order to reliably find abstracts containing PD term occurrences, it is sufficient to use the specific synonyms as the specific baseline has an abstract F-measure of 0.95, only slightly under the SynTree performance (0.97).
This is not the case any more on the 2-class level: while the specific dictionary still clearly outperforms the machine learning methods (F=0.77 against 0.47 from the best ML method), the margin between the specific baseline performance (F=0.77) and SynTree performance (F=0.9) is

much higher.
Table 9.8 shows the results for the NER problem . Although the conditional random fields in ABNER and STANFORD are also context-dependent and have been trained on a quite large training set containing 1415 abstracts and almost 10.000 (9.613) annotations, the performance of ABNER is very poor: F=0.29. Note that as the 2-class performance is much higher (F=0.47), this implies incorrect disease entity assignments. SynTree on the other hand not only can do a good 2-class prediction (F=0.9), but also identifies the correct named entity in most cases with a performance F=0.86.

I(4) set: Abstract level prediction results

	ABNER(2-class)	ABNER(Multi-class)	Stanford(2-class)	SynTree	BL(SENS)	BL(SPEC)
TP	**98**	96	97	97	99	94
FP	97	95	99	**5**	61	4
FN	**2**	4	3	3	1	6
100*Pr	50.3	50.3	49.5	**95.1**	61.9	95.9
100*R	**98.0**	96.0	97.0	97.0	99.0	94.0
F	0.66	0.66	0.66	**0.96**	0.76	0.95

Table 9.6: Evaluation of the abstract level predictions on the I(4) set. An abstract is classified as positive if there is at least one psychiatric disease mentioned. On the right side of the tables the baseline (BL) predictions using synonym dictionary hits are shown. The results indicate that for the abstract classification problem it is sufficient to the context unspecific synonyms as BL(SPEC) performs about the same as SynTree. The machine learning methods however predict many incorrect abstracts leading to much lower F values. The training of the ML methods was performed on all previously annotated abstracts (1415) having 9613 annotated entities.

I(4) set: 2-class level prediction results

	ABNER(2-class)	ABNER(Multi-class)	STANFORD (2-class)	SynTree	BL(SENS)	BL(SPEC)
TP	556	640	653	**840**	871	638
FP	1274	1157	1217	**101**	929	88
FN	367	283	270	**83**	52	285
100*Pr	30.4	35.6	34.9	**89.3**	48.4	87.9
100*R	60.2	69.3	70.8	**91.0**	94.4	69.1
F	0.40	0.47	0.47	**0.90**	0.64	0.77

Table 9.7: Evaluation of the 2-class predictions on the I(4) set. A term occurrence is classified as positive if it overlaps with a manually annotated psychiatric disease (regardless of a possible difference in the predicted and annotated disease entity). The test set of 200 abstracts is difficult (see description of I(4) in section 9.3). SynTree clearly outperforms the other methods with all Precision (Pr), Recall (R), and F around 0.9.

I(4) set: NER prediction results

	ABNER	SynTree
TP	359	**775**
FP	1438	**166**
FN	283	**83**
100*Pr	20.0	**82.4**
100*R	55.9	**90.3**
F	0.29	**0.86**

Table 9.8: Evaluation of NER prediction, i.e. identifying the correct disease entities in the I(4) set. For this task only ABNER and SynTree could be evaluated. Predictions reporting wrong disease entities as compared to the annotated entities are treated as false positives (FP). The results indicate that SynTree has more general (Recall (R) is about 35 points higher) and more specific rules (Precision (Pr) is more than 60 points higher) than ABNER trained on 1415 abstracts with 9613 annotations.

9.4 Discussion and outlook

The presented results, especially from Table 9.5, indicate that the features used in even context-dependent machine learning methods such as the conditional random fields in ABNER and STANFORD are not sufficient to distinguish between the disease named entities even if they trained on comprehensive sets of well annotated abstracts. In contrast, the simply and eailsy extendable system SynTree, which is based on contexts defined by the named entity ontology and simple human understandable rules can identify named entities at both high sensitivity and specificity. SynTree is also very fast (almost a factor 50 and 300 times faster than ABNER and STANFOTRD, respectively) and therefore, allows for an application of large datasets such as the whole PubMed abstracts or even full text collections and for an interactive use for constructing training sets, entity ontologies and rules. The paper shows that in order to obtain reliable results with good performance even with context-dependent disambiguation methods quite large and well-annotated gold standard sets are needed. These can for example be constructed via a the iterative benchmark procedure describe above. This also can be used to efficiently construct respectively extend entity ontologies and to derive SynTree disambiguation rules.
SynTree can be applied to all kinds of ontologies. As SynTree is fast (table 9.3) one can efficiently perform several iterations with changed ontology contexts and rules against PubMed in a day. We think that SynTree this way can be used to derive context information for the interpretation of high-throughput data.

9.5 Rule definitions for the psychiatric disorders

In this section we present the list of rules we used in the final configuration of SynTree for the psychiatric disease recognition. Most of the rules were defined along with the creation of the initial synonyms, for the small set of changes during the evaluations see section 9.3.

1. **basic rule**: for every context dependent synonym in the tree apply the positive modifier set from the closest parent with the defined scope (see Figure 9.1).

2. **general rejected prefixes**: if one term from this set occurs immediately before a hit, the match becomes invalid. Example terms in the set for this rule: *except in, not for, without, ...*

3. **general rejected suffixes**: if term from this set occurs immediately after a hit the match becomes invalid. Example terms in the set for this rule: *score, scale, questionnaire ...*

4. **positive suffix rules on context sensitive synonyms of depression and phobic disorders**: Hits from the synonyms "depressive", "depressed", "phobic" of the entity node "depressive disorder" are only valid if they have a specific (+) suffix context. Example terms for this set: *phase, episode, individual, patient, ...*

5. **negative prefix/suffix rules on context sensitive depression synonyms** : Hits from the synonyms "depressive", "depressed", *"phobic"* of the entity node "depressive disorder" are only valid if not describing the change of some measure, i.e if not followed by the terms *level, level of*, and not preceded by the terms *long-term, rapid, slow* .

6. **rejected suffix on synonyms ending on "thyroid"** Hits regarding the thyroid hormone should be rejected, so for synonyms ending on "thyroid" we extend the negative suffix modifier set by the terms *"hormone", "-hormone"*.

7. **rejected suffixes on context specific synonyms of schizophrenia** The context specific synonyms of schizophrenia are also often used as symptom description of other psychiatric diseases (e.g. psychotic symptoms). Therefore the negative suffix modifier set is extended for all (recursively) context specific synonyms of schizophrenia hits the by the terms *symptom, behavior*.

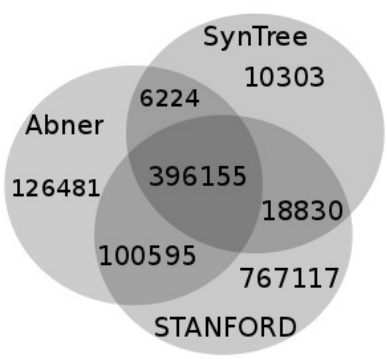
8. **rejected sentence context on context specific synonyms of paranoid schizophrenia**

   In order to avoid false matches for paranoid schizophrenia due to overlapping synonyms with the synonyms of paranoid disorder, we extend the negative sentence context modifier set by the term *paranoid disorder*.

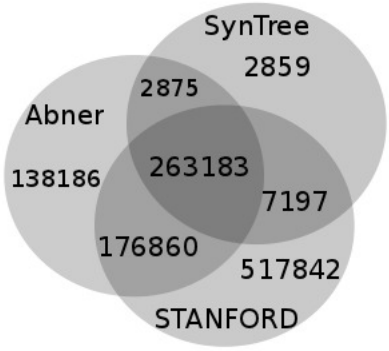
9. **required sentence context for hypovitaminosis** To ensure that the synonyms (vitamin names etc.) only match in a context regarding decreasing levels of the vitamins, we replace the positive sentence context by terms describing this: *decrease, decreased, deficiency, low, ...*

10. **rejected suffixes for seasonal affective disorders**: To find occurrences we use context-specific terms such as *"summer", "winter"*. In order to avoid false hits in the context where these terms are used for time frame description we extend the negative suffix context by the terms: *month, season*.

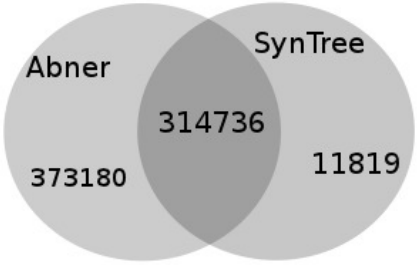
11. **allow prefix "negative" for schizophrenia**: For the special case of schizophrenia we remove the term *negative* from the inherited negative prefix modifier set.



(a) I(2)



(b) I(3,A)



(c) I(4)

Figure 9.3: Distribution of abstracts having at least one predicted occurrence of a psychiatric disease. We sampled from each of the seven disjunct sets 100 abstracts by selecting the abstracts with the most predictions. The light-gray parts show the mutual exclusively predicted abstracts, which were used in I(2) (see section 9.3) and I(3,E) (see section 9.3). Using the mutual exclusive predictions we can create hard benchmark sets by checking if (i) hits reported by a single method are correct and (ii) which true hits are only recognized by one of the methods. In I(3,A) and I(3,C) we also analyzed the abstracts where all three methods predicted disease occurrences (the darkest gray section, 100 abstracts) and all sets together (700 abstracts) respectively. For the results see tables 9.4 and 9.5.

Chapter 10

A context based approach to identify the most likely mapping for RNA-seq experiments

The context for method presented in this chapter is given by the experimental setup generating the data - the next generation sequencing, in particular the RNA-seq technique.

RNA-seq experiments fragment and amplify the transcribed mRNA of a cell, cells, a cell culture, or a tissue under investigation and, then, determine the nucleotide sequences of a sample of short fragments (typically several millions of so called reads). The reads originate from different parts of the mRNA transcripts. The goal is to find their origin (the genetic locus, i.e. the position in the gnome) for them. The ContextMap method presented in this chapter includes and explicitly uses the important features implied by this context: (i) the source of all reads should be explained by the mapping (ii) for expressed transcripts usually multiple reads will exist (iii) reads may correspond to exon-exon boundaries overlapping part of a transcript (junction reads) (iv) there may be variances between the genomes of the sample and the genome used for mapping and thus transcriptome and the genome used for mapping (resulting in mismatches in mapped reads even if no sequencing error happens), and the (v) sequencing may introduce single nucleotide errors (real sequencing errors).

The novel idea here is to use the context of all reads to resolve ambiguous read positions. At the same time, the approach is more flexible as more mismatches during read matching can be allowed - accounting for feature (iv). In addition, we show that contexts can be exploited for another purpose: handling of several genomes at the same time for application of (possibly) contamined samples or metagenomics. This is not an artificial setup: samples may be contaminated, or infected by microbes or viruses which might constitute important diagnostic issues.

The content of the chapter is mainly based on joint work with Thomas Bonfert and the publication presented at RECOMB-seq 2012 ([25]). My contribution here is the idea of the method, the first implementation (prototyping) and evaluation, and the idea to use additional genomes for mapping to extend the set of explainable reads.

10.1 Background

Deep sequencing of mRNA using next-generation sequencing technologies (RNA-seq) offers novel opportunities to profile and quantify whole transcriptomes. The nucleotide-level resolution of RNA-Seq experiments provides new insights for researchers into the complexity of alternative splicing and isoform expression [88, 92, 148, 173, 193]. One major challenge in RNA-seq is the identification (mapping) of the origin of each sequenced read, i.e. which part of which transcript it corresponds to. As this requires the alignment of a huge number (in the order of millions) of relatively short sequence reads against reference sequences, such as e.g. a genome or transcriptome, a number of specialized alignment algorithms have been developed. Here, algorithms based on the FM-index – a compressed, searchable suffix array-like structure [57] – have been most successful due to their reduced time and memory requirements. The most widely used of these algorithms is Bowtie [113].

The complex structure of a spliced transcriptome compared to the genome limits the applicability of simple alignment-based approaches for RNA-seq experiments. While an alignment against the genome can successfully map reads from an unspliced region of the transcriptome, such as individual exons or introns, reads from spliced regions are missed. To some degree, this problem can be addressed by an alignment against a known transcriptome or a database containing exons and all possible junctions between these exons [35, 133]. However, in the first case, the sensitivity of this approach depends strongly on the completeness of the annotated transcriptome and novel splice junctions will be missed. In the second case, the number and complexity of potential junction reads increases dramatically with increasing read length resulting in forbiddingly large databases.

As a consequence, a large number of more sophisticated bioinformatics approaches have been developed for the identification of junction reads. One of the first tools was TopHat [181] which is able to discover splice sites without the use of a transcriptomic annotation. In a first phase, reads are aligned to a reference genome using Bowtie and the mapped reads are then assembled to so-called islands. Subsequently, potential splice sites are annotated based on canonical splice signals and reads spanning these splice sites are identified. In contrast, two more recently published methods, RUM [76] and RNASEQR [33], start with read alignments to both the reference transcriptome and genome. Novel splice junctions are then identified by aligning unmapped reads to the genome using BLAT [104] which determines gapped local alignments. A gapped alignment only becomes feasible here because it is applied to the much smaller number of unaligned reads. Thus, all of these approaches rely heavily on additional information such as canonical splice sites or reference transcriptome annotations. A more sophisticated approach which is independent of the availability of such additional information is MapSplice [194]. The general idea of MapSplice is that it first generates candidate alignments for each read based on alignments of short fragments of the read against the genome. Splice junctions are then predicted based on all candidate alignments for different reads containing the respective splice site. Thus, in contrast to previous approaches MapSplice takes into account the context of a splice site, i.e. how many reads support the predicted splice site.

While this approach is interesting, there are still a few limitations. First, the context of other read mappings is taken only into account for potential splice sites. For reads mapping to an unspliced

region, the context is ignored. Second, only spliced read alignments are considered but not unspliced reads within this range. In particular, unspliced reads ending or starting at the potential splice junction might lend additional support to the splice site. Third, for each fragment only the alignments with the minimum number of mismatches is considered ignoring the possibility that the alignment to the correct origin may have more mismatches due to sequencing errors. Finally, although non-unique alignments are outputted by MapSplice, no effort is made to resolve them. While some downstream analysis methods have been developed which can deal with this uncertainty (e.g. for the estimation of transcript levels [117, 141, 151]), often these multi-maps are simply discarded and, thus, lost for the analysis.
In this chapter, we present a novel method (ContextMap) which extends the idea of using the context of other read alignments for identifying the correct position for each read. This method is more general than MapSplice as it uses the context not only for splice junctions but also non-spliced read alignments and always uses both spliced and unspliced read alignments. The general idea of the approach is not to assign each read to the position where it can be aligned with the fewest mismatches, but to the position where it fits best in the context with its surrounding and all other reads. For this purpose, we allow a high degree of ambiguity during the mapping stage which is eventually resolved in the final mapping. Thus, ContextMap is robust against biases caused by sequencing errors and may also be used for correction of sequencing errors.
ContextMap can be used in stand-alone and incremental mode. The incremental mode starts with an initial alignment of a mapping method of choice and refines this initial mapping. In the stand-alone mode, it obtains this initial mapping by first aligning the reads to the genome, the transcriptome or both. In the implementation we present here, ContextMap uses as initial context the unique mapping identified by MapSplice or TopHat. Evaluation on simulated reads for human and mouse showed a significant improvement in mapping quality of the ContextMap refinements compared to both MapSplice and TopHat. This highlights the importance of using the read context for obtaining the final mapping.

10.2 Methods

10.2.1 Outline

The central concept of ContextMap is a *context* of reads. A context is a set of reads which all originate from the same expressed stretch of the genome. In general, such a context corresponds to an individual gene but may also correspond to a few overlapping or closely located genes. At any step of ContextMap, only reads within the same context will be considered for assigning a position to the read within the *specific* context considered. At least initially, however, reads may be assigned to different contexts and in this case are assigned a position for each context. This ambiguous assignment of reads to contexts is eventually resolved in the final step of ContextMap. ContextMap then consists of the following three steps (see Figure 10.1): (1) the definition of initial contexts using a preliminary mapping of reads; (2) the extension of the preliminary mapping by re-aligning reads within their respective contexts; (3) the resolution of ambiguous mappings both within and between contexts to obtain a final unique mapping. Here, the advantage of using

contexts is that we can allow a much larger degree of ambiguity during the second step, as ambiguities are limited to individual contexts and not the whole genome and the contexts allow us to resolve the ambiguities successfully. In the following, the individual steps are explained in more detail.

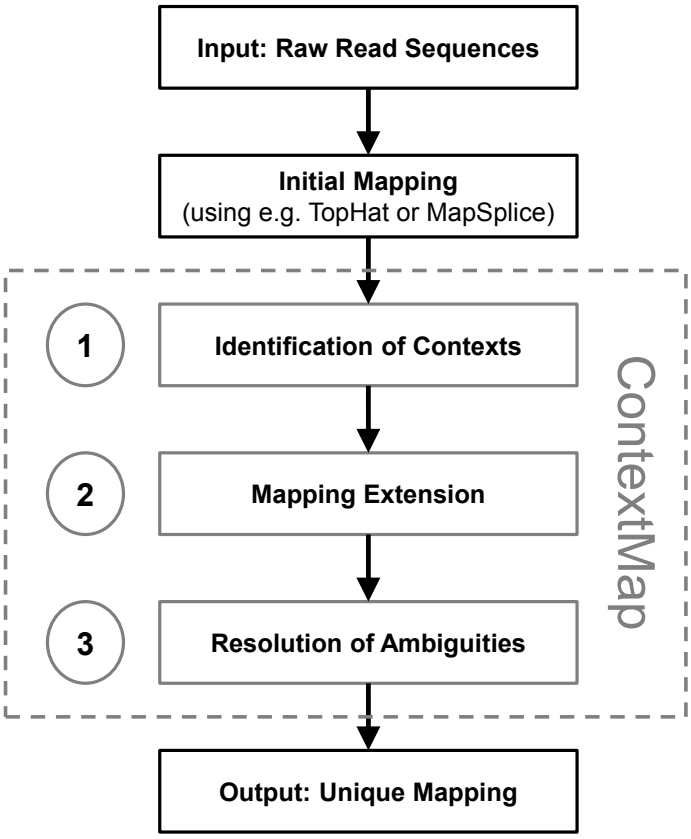


Figure 10.1: **Outline of ContextMap**. ContextMap takes as input initial alignments provided either by other mapping tools such as e.g. TopHat or MapSplice or by a genome or transcriptome alignment. As output unique locations are provided for all reads that can be mapped. ContextMap involves three steps which are explained in more detail in Figures 10.2-3: (1) Identification of contexts (see Figure 10.2 A); (2) Extension of initial alignments (Figure 10.2 B-C); (3) Resolution of ambiguities (Figure 10.3).

10.2.2 Identification of contexts

The initial step of ContextMap requires the definition of contexts, i.e. the definition of reads from the same expressed region of the genome (Figure 10.2 A). Here, the preliminary context does not have to be fully correct as it only roughly defines putatively expressed regions. In particular, the

precise alignment of the reads is of lesser importance, as reads will be re-aligned in a subsequent step to identify a larger number of candidate alignments.
There are several ways in which a context can be identified. The solution we implemented for this study is to use the alignment of other mapping algorithms such as MapSplice or TopHat which we aim to improve upon. In this case, all unique mappings determined by the used method are included. Alternatively, ContextMap can identify contexts itself by an initial alignment of reads to the genome using the forward and backward approach described in the following section. Potential splice junction reads are then identified by searching for easily detectable (balanced) splits. Instead of an alignment to the genome, an alignment to the transcriptome may also be used to identify the initial mapping.
The contexts are then defined based on the genomic distance between aligned reads. Reads with a maximum distance of d_{min} between start or end positions are collected into the same context. Accordingly, the minimum distance between contexts is d_{min}. Note that contexts may contain regions without mapped reads larger than d_{min} if these regions are contained within the spliced part of a read, i.e. an intron. The distance threshold can be adjusted if smaller or larger contexts are desired. In this study, d_{min} was set to 10kb. Alternatively, gene annotations may be used to define contexts, which then limits the mapping to known genes.

10.2.3 Extension of candidate mappings

The first part of this step is a re-alignment of previously unmapped or non-uniquely mapped reads to each context (Figure 10.2 B). In the following, alignments containing no splice site are denoted as full, whereas alignments to splice sites are denoted as split alignments/reads. In this step, all full and split alignments fulfilling a maximum mismatch criterion are generated using Bowtie in the following way. For each context, both a forward and backward Bowtie index is created. Using these indices, reads are then aligned to the contexts both in forward direction starting from the read start and in reverse direction starting from the read end. For both alignments a seed of 40% of the read (but at most 40bp) is used allowing a maximum number of 1 mismatch in the seed region. Again, these are parameters which can be adjusted by the user depending on read length and expected error rates. Forward and backward alignments are then combined if the maximum number of mismatches in the alignment does not exceed the predefined threshold.
These alignments as well as the initial alignments provide a set of potential splice sites for each context. Each splice site is characterized by two sequence positions (s_1, s_2) such that any read covering the splice site will first align on the genome upstream of and up to s_1 and then continue to align at s_2 and downstream of this position. Using the potential splice sites for each context, additional alignments are generated in the following way (Figure 10.2 C). For reads which have a full alignment crossing either s_1 or s_2 for a splice site, any alignment respecting the splice site and fulfilling the maximum mismatch criterion is generated. The same is done for partial read alignments in which the read could be aligned except for a small fragment at the start or the end, but the fragment was to small to obtain a meaningful alignment. As the number of potential splice sites suggested by alignments of other reads is limited, all of these splice sites can be tested whether they allow an alignment of the unaligned fragment fulfilling the maximum mismatch criterion.

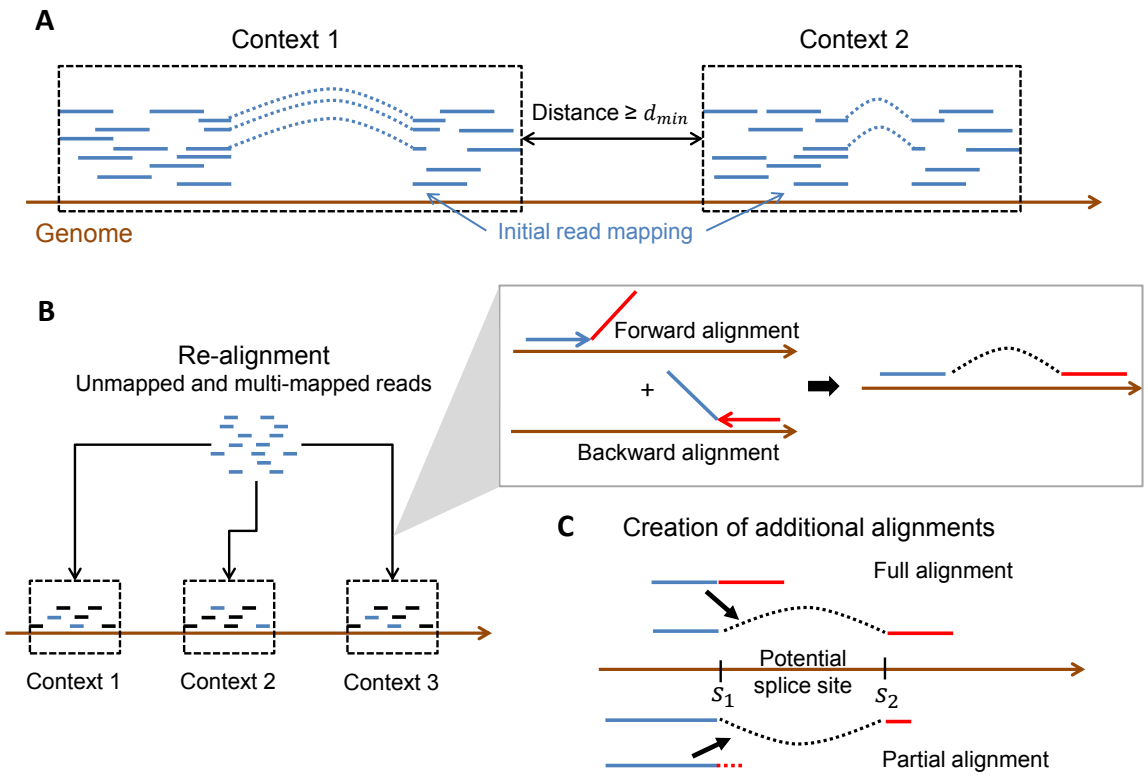


Figure 10.2: **Definition of contexts and extension of alignments**. A) A context is defined as a set of reads with overlapping mapping locations such that the minimum distance between any pair of reads from different contexts is d_{min}. B) The initial alignments used as input to ContextMap are extended by re-aligning unmapped or multi-mapped reads to each context using a forward alignment from the read start towards its end and a backward alignment from the read end towards its start. Forward and backward alignments are combined to a full or split read alignment if the maximum mismatch criterion is met. (C) All alignments obtained at this step provide a list of potential splice sites. Additional alignments respecting the maximum mismatch criterion are then created for full and partial read alignments based on these potential splice sites. Accordingly, for any read aligning up to the first position s_1 of a splice site it is investigated if the rest of the read can be aligned starting at the second position s_2 of the splice such that only a maximum number of mismatches are created. These additional alignments are then added to the set of candidate alignments

10.2.4 Resolution of ambiguities

Redundancy filtering

Having generated a large number of ambiguous mappings for each read and context which respect the maximum mismatch criterion, the next step is then to resolve the ambiguities within each context before addressing the ambiguities between contexts (Figure 10.3). To not unnecessarily bias this step by reads which will be later removed anyway as they are assigned to a different context, we use an additional parameter which specifies the maximum mismatch difference (δ_{mm}) for a read in any context to the best-matching context. For this purpose, we calculate for each read and each context the minimum number of mismatches in any of the candidate alignments. If the best match of a read in a context c requires $> \delta_{mm}$ mismatches more than in the best-matching context, the read is removed from c.
To further simplify the resolution process, full and split mappings with either the same start or end position are merged for this step. The final configuration for these reads (full or split) is then determined at the end of this step. In addition, we identify the best supported splice site among any pair of overlapping splice sites (Figure 10.3 A). Two splice sites (s_{11}, s_{12}) and (s_{21}, s_{22}) are overlapping if $|s_{11} - s_{21}| <$ read length and $|s_{12} - s_{22}| <$ read length. The idea behind this approach is to eliminate splice sites which are too close to each other and, thus, make no biological sense or are suggested by alternative split alignments of the same read. For this purpose, we calculate the number of supporting reads (full, split or partial) for each splice site and the corresponding mismatch cost and check which of the splice sites has a known splice signal. In the following, let n_i be the number of reads with i mismatches supporting the splice site and m the maximum number of mismatches allowed. Furthermore, $\lambda = 2$ if the splice site contains a known splice signal and $\lambda = 1$ otherwise. The evidence score is then calculated as

$$evidence = \lambda \cdot \sum_{i=0}^{m} \left(0.3^i \cdot n_i\right), \tag{10.1}$$

Thus, the score is the weighted sum of the number of reads with the weight decreasing exponentially with the number of mismatches. For each set of pairwise overlapping splice sites, the one with the largest evidence score is then chosen.

Calculation of read coverage scores

To obtain the unique mapping of each read within each context, we calculate a coverage score for this read in the following way (Figure 10.3 B). First we calculate for each position the number of reads mapping to this position (c_m). If there are ambiguous mappings for a read, it is counted for all positions in any of the ambiguous mappings. We then define 4 regions within and around the mapped read. Region 1 contains the positions the read is aligned to. Region 2 contains all positions 200 bp either upstream of the read start or downstream of the read end. Region 3 corresponds to the positions > 200 but ≤ 500 bp from read start or end and, finally, Region 4 to positions > 500 but ≤ 1000 bp from read start or end. The score for each region, $score_i$, is then defined as the maximum c_m within the corresponding region i. Region sizes and numbers may be adjusted depending on the user needs.

Finally, the coverage of a read is calculated as

$$coverage = \sum_{i=1}^{4} 2^{4-i} \cdot \lfloor \ln(score_i) \rfloor. \tag{10.2}$$

This calculation involves three aspects. First, the definition of regions for which the maximum c_m is calculated allows to take into account reads mapping very far from the actual read alignment without assigning too much weight to distant reads. The larger the region, the less weight is given to reads falling in this region. Second, the individual region scores are additionally weighted depending on the distance from the actual read alignment. Reads overlapping the alignment region have the largest weight and each region has a weight twice as high as the next more distant region. Finally, instead of the actual read counts logarithms of the counts are used, reflecting their order of magnitude. In this way, this coverage definition is related to the geometric mean of the scores which is less biased by large outliers than the arithmetic mean.

Priority queue and ambiguity graph

The ambiguous reads are then put into a priority queue sorted by the difference in coverage score between the best and second-best ambiguous mapping (Figure 10.3 C). Furthermore, a graph structure is created for each context in which each ambiguous read is connected with the positions in this context to which it maps (ambiguity graph). Thus, the ambiguity graph provides the information on which coverage scores have to be updated if a read is assigned to a unique position.

We then iteratively remove the entry with the highest score from the queue, i.e. the entry for which the ambiguity resolution is the most straightforward. Subsequently, we update the coverage scores based on the dependencies identified by the graph structure and then update the keys in the queue. Finally, when the queue is empty, all ambiguous reads have been assigned a unique location within the respective context. For a speed-up, the costly coverage updates were omitted in this study and ambiguities are resolved only based on the highest (initial) coverage score.

In the last part of this step, the assignment for the merged full and split mappings with either same read start or end is resolved. For this purpose, coverage scores are again calculated but this time only for the positions which are differently mapped by the alignment and the positions upstream of the read starts, in case of differential start points, or downstream of the read end, in case of differential end points. The alignment with the largest coverage is then chosen.

To resolve the between-context ambiguities, we basically perform the same steps as for each individual context by calculating coverage scores for each context and using a priority queue based on the coverage differences and a global dependency graph between reads and contexts. Finally, resolved read mappings are outputted if the relevant region suggests a real expression. Currently, we predict reads with at least 100 other reads mapped in a region within 1000 bp up- or downstream of the read. These are again user-defined thresholds.

10.2.5 Standalone ContextMap

The standalone version of ContextMap does not require an initial mapping from an external program as input but creates this initial mapping by itself using a modified version of Bowtie. This initial mapping is then refined using the same methods as in the original implementation with only few modifications. The major changes compared to the original implementation are outlined in the following.

Defining the initial mapping using sliding contexts

The procedure for determining the initial mapping consists of two steps. First, candidate alignments are defined resulting in either full or incomplete alignments. Here, we apply the same forward/backward alignment approach based on Bowtie used in the incremental mode. If a read can be aligned completely with at most the maximum number of mismatches allowed, it is classified as a full alignment. If the seed can be aligned, but the total number of mismatches is too large, it is an incomplete alignment. Incomplete alignments are then classified as either partial alignments – if the last valid mismatch is $\leq$ 10 bp away from the read end – or candidate split alignments.
The second step tries to extend the candidate split alignments to complete split alignments (Figure 10.2.5). For this purpose, we use a so-called sliding context approach which essentially consists in sliding a variable-sized window with a pre-defined maximum length across regions of the genome with read alignments. Starting from an alignment, the window is extended by any overlapping alignment until the maximum length is exceeded. This maximum value is chosen such that it corresponds approximately to the expected maximum length of a pre-mRNA. For each sliding context a Bowtie index is then built dynamically while the index for the previous sliding context is discarded. Using this index, completing alignments are then determined for each candidate split alignment within the sliding context. This restricts the search space to a region covering only one or very few genes. Three alignment searches are then performed with increasing seed length and increasing number of allowed mismatches as illustrated in Figure 10.2.5. This step is very fast for each sliding context as the context and consequently the index is very small, very strict rules are used for the search and only a small number of reads have to be aligned. All possible split alignments obtained for this sliding context are then included in the initial mapping in addition to the full and partial alignments. After alignments for a sliding context are completed, we proceed to the next candidate split alignment not completely contained in the previous context and extend this to obtain the next sliding context.

Combining partial alignments

In the incremental implementation, we investigated for each partial alignment whether it overlapped with a potential splice site suggested by other split alignments and whether a split alignment respecting the maximum number of mismatches could be obtained for the partial alignment based on this splice site. While this step is also included in the standalone version, an additional step is included to address partial alignments which cannot be resolved via candidate splice sites. In this step, we try to combine partial alignments for different reads to larger split alignments.

For this purpose, each pair consisting of a partial forward and a partial backward alignment for two different reads within a certain window is investigated. The window size is the same as the maximum size of the sliding contexts. To reduce runtime, only potential splice sites within the last 40% of a forward alignment and the first 40% of a backward alignment are considered. For splice sites closer to the read start or end, the corresponding alignments would be identified as split alignments in a previous steps, thus, no alignments are lost due to this restriction.

Extension to transcriptome annotation

Finally, standalone ContextMap can include knowledge on splice junctions provided by transcriptome annotations. This information is used at two points. First, when extending full and partial alignments to split alignments, not only candidate splice sites suggested by other split alignments are considered but also splice junctions indicated by the transcriptome annotation. Second, the scoring for overlapping splice sites was modified in the following way. Given a set of pairwise overlapping splice sites, it is first investigated whether any of these are supported by the annotation. If this is the case, all other splice sites in the set are discarded. If not, only splice sites in the set supported by a known splice signal are retained. Only if none of the splice sites in the overlapping set are supported by annotation or splice signals, the score described for the incremental version is used.

10.2.6 Identifying sequencing reads from multiple sources

One of the main difference of ContextMap compared to previously published methods is the large degree of ambiguity allowed in intermediate steps both between and within contexts. For each read not only the best alignment or context but any alignment or context fulfilling the constraints is investigated. The unique mapping for the read to only one context is determined only after ContextMap identified the optimal location of the read for each of its alternative contexts. Until the final step, contexts are treated independently of each other.

The advantage of this approach is that it allows investigating many alternative sources of reads in parallel. These alternative sources may include e.g. rRNA sequences, which are generally not included in genomic sequences, and viral and microbial genomes to account for infections or contamination of cell lines. Contexts can then be identified separately for each genome including optimal locations for each read for any context it is associated with. The final step is then used to decide for each read which of these contexts in any of the genomes considered fits best based also on the number of other reads aligned to the respective contexts.

This approach also allows circumventing the limitations of Bowtie to 2^{32}-1 characters per index as context identification and ambiguity resolution within contexts can be performed separately for different genome indices until the last integrating step. This is relevant as the human genome alone makes up 73% of the maximum index size. Moreover, all microbial genomes from the NCBI database taken together require 134% of the maximum index size. Thus, previous mapping approaches cannot simply use one index combining human and microbial genomes.

10.3 Results

10.3.1 Simulation of RNA-seq data sets

All approaches were evaluated on simulated human (hg19) and mouse (mm9) reads. For the human reads a 74bp single end RNA-Seq data set was generated with the FLUX simulator [178]. Since FLUX was too slow for simulating reads for the whole human genome we restricted the simulation to chromosome 1 and obtained a final set of $446,398$ reads. We then randomly introduced sequencing errors into this original read set to obtain two data sets with 1% and 2% uniformly distributed errors, respectively (Figure 10.3.1). Since sequencing quality generally deteriorates towards the end of the read, we also simulated reads with error probability increasing along the read length (Figure 10.3.1 A). In this case, the overall error rate was again fixed at 1% or 2%, respectively, but the position of the error in a read was drawn from a polynomial distribution with cumulative distribution function $F(x) = \frac{1}{l^3}x^3$, where l is the read length.
For the mouse read data, we used an evaluation set published by Grant *et al.* in the RUM algorithm [76], restricted to reads mapping to Ensembl transcripts. It contains 17,301,982 single end 100bp reads. In the original RUM simulation model, two types of errors were simulated: random base and tail errors. Base errors were uniformly distributed across the whole read and tail errors only in the tail of a random fraction of the reads. For our purposes, we used the first test set with a base error of 0.5% and no additional tail error. The second test set with an additional tail error of 50% in the last 10 bases of 25% of reads was not used. In this case, trimming of reads by the last 10 bases would always result in a significantly improved performance. To test the performance for higher error rates, we also introduced sequencing errors in the mouse data set to obtain two sets with uniformly distributed 1% and 2% error rate, respectively.
The objective behind using two different error rates each for each data set was to have both a relatively easy data set with few sequencing errors as well as a more challenging set with a larger error rate. In the first case, we expect little reduction of the performance due to error rates. In the second case, a large influence is expected.

10.3.2 Baseline mapping algorithms

To show that our approach is able to further improve the results of widely used mapping and junction discovery tools, we applied ContextMap to mappings which were produced with MapSplice and TopHat. Both of them are popular programs that are able to map RNA-Seq reads to a reference genome without using a transcriptome annotation. TopHat mappings were obtained allowing at most 2 mismatches per segment and read. For MapSplice we used default parameter settings except for the spliced mismatch parameter. For this we used 4 as otherwise MapSplice performed poorly for the data sets with a larger error rate. ContextMap was then applied both on the TopHat and MapSplice mappings.
In addition, we also evaluated the standalone ContextMap using transcriptome annotation from Ensembl and compared its performance against TopHat using transcriptome annotation and RNASEQR, which performs a transcriptome alignment in the first step. As the strategy of RUM (including a transcriptome alignment and a subsequent alignment of unaligned reads using

BLAT) is similar to RNASEQR, it was not evaluated.

10.3.3 Read mapping accuracy

To evaluate the accuracy of a mapping, we calculated the accuracy of read mapping, i.e. the fraction of reads aligned to the correct positions. This was done separately for reads which were simulated as complete reads as well as for reads spanning a splice junction. A uniquely mapped read is counted as an *exact* true positive (TP) if it was mapped correctly at every base. In case a read was either an exact match or at least the start or the end position on the genome was correctly predicted, we counted it as a *relaxed* true positive. Reads not (uniquely) mapped or mapped to wrong positions were treated as false negatives (FN). Accuracy was then calculated as

$$Accuracy = \frac{TP}{TP + FN}. \tag{10.3}$$

10.3.4 Results of the incremental variant

The evaluation results for the incremental variant over initial alignments derived from TopHat and MapSplice are shown in Tables 10.1 and 10.2 for the human and mouse data sets, respectively. As expected, all approaches performed best for complete read mappings and performance deteriorated considerably for the junction read mappings. However, if also we include the cases in which at least either the start or the end position of the read was predicted correctly, we see that in many cases at least part of the read could be mapped correctly.

Data set	Program	Complete Read Mapping		Junction Read Mapping	
		exact	relaxed	exact	relaxed
Human, 1%	TopHat	93.85	93.85	72.18	80.73
	ContextMap[1]	95.76	95.76	**73.12**	**86.21**
	MapSplice	78.81	89.64	70.16	78.78
	ContextMap[2]	**95.97**	**95.98**	73.07	86.05
Human, 2%	TopHat	90.30	90.30	70.38	78.05
	ContextMap[1]	**95.51**	**95.52**	**71.53**	**83.78**
	MapSplice	53.92	81.45	58.73	68.14
	ContextMap[2]	95.45	95.48	64.13	78.68

Table 10.1: **Evaluation results on human data sets** Accuracy of read mapping is reported for the two human data sets with 1% and 2% error rate, respectively. Here, ContextMap[1] denotes the refinement of TopHat initial mappings by ContextMap, ContextMap[2] the refinement of MapSplice initial mappings.

Our results clearly show that for both MapSplice and TopHat, ContextMap could improve significantly upon the predictive performance of the original mapping. In almost all cases, both variants

Data set	Program	Complete Read Mapping		Junction Read Mapping	
		exact	relaxed	exact	relaxed
Mouse, 1%	TopHat	94.43	94.43	80.10	86.21
	ContextMap[1]	**97.03**	**97.05**	82.95	91.62
	MapSplice	95.21	95.22	84.49	89.12
	ContextMap[2]	96.97	96.99	**87.77**	**94.91**
Mouse, 2%	TopHat	88.87	88.88	77.44	82.87
	ContextMap[1]	95.82	95.87	80.45	88.52
	MapSplice	92.89	92.89	77.89	81.95
	ContextMap[2]	**96.31**	**96.36**	**82.87**	**90.57**

Table 10.2: Accuracy of read mapping is reported for the two mouse data sets with 1% and 2% error rate, respectively. Notation is the same as in Table 10.1.

of ContextMap outperformed both MapSplice and TopHat and in all cases at least one variant performed best. Interestingly, although MapSplice performed significantly worse than TopHat on the human data sets, ContextMap on MapSplice outperformed ContextMap on TopHat for the complete reads on the 1% error rate set by a small margin and was only worse by a similar small margin on the 2% error rate set. This was particularly impressive as in these cases MapSplice was > 15 percentage points worse than TopHat on the complete reads. Although MapSplice is usually a good mapping algorithm, we found that determining the optimal parameter settings for data sets with high error rates is rather difficult. In this case, ContextMap provides a useful alternative to parameter tuning as predictions can be refined considerably even if the parameter settings for MapSplice are not optimal.

To analyze whether alignment accuracy depended on read coverage, genes were partitioned into four approximately equal-sized groups based on the average number of reads per base. For each group, accuracy values were then calculated separately. This analysis showed that performance for complete reads without splice sites was mostly independent of read coverage (Figure 10.3.4 A). In contrast, for junction reads, accuracy depended strongly on read coverage (Figure 10.3.4 B). Generally, the higher the coverage was for a gene, the higher was the accuracy of the corresponding read alignments. Thus, read coverage is important for the identification of (novel) splice junctions, likely due to the much larger number of possible alignments, but not for alignment of reads originating from unspliced transcript regions. Nevertheless the relative ranking of the evaluated mapping approaches was generally only little influenced by read coverage. Interestingly, on the mouse 2% error data set (Figure 10.3.4 B), ContextMap on MapSplice only outperformed ContextMap on TopHat for splice junctions in genes with highest read coverage. Since these genes contribute the largest set of reads to the overall read set, they mostly determined overall accuracy values and, accordingly, overall performance of ContextMap on MapSplice was also superior.

Interestingly, we found that alignment accuracy for both complete and junction reads depended

little on the distribution of errors along the reads. Alignment accuracy on the human data set was almost identical no matter whether we used uniform or polynomial error distributions along the read length. The only method that suffered consistently from a non-uniform error distribution was TopHat with a decrease of around 4 percentage points for both types of reads on the 2% error data set with polynomial error distribution. Remarkably, however, accuracy of ContextMap on TopHat mappings was not reduced for the complete reads on this set despite the reduced quality of the initial mapping. Only accuracy for junction reads suffered but still was significantly higher than for TopHat and MapSplice, respectively.
Finally, we evaluated parameter sensitivity of ContextMap by running it with different seed lengths (values of 10, 20, 30, 40) and different minimum expression values for contexts (0-200 in steps of 20) on the human data sets. For both TopHat and MapSplice original mappings, we generally observed only minimal variation in accuracy with standard deviations of < 0.51 and < 0.21 percentage points for the seed (on complete reads) and minimum expression value parameters, respectively. The largest variation was observed on junction reads for the seed length parameter with a standard deviation of 1.2-2.9 percentage points. Remarkably, alignment accuracy was actually increased by using shorter seed lengths due to a larger number of junction reads correctly aligned. Unfortunately, it also resulted in a slight increase in the number of junction reads aligned incorrectly instead of not at all as in the case of longer seed lengths.

10.3.5 Results for the standalone variant

The evaluation results of the standalone ContextMap with and without using transcriptome annotation from Ensembl are shown in Tables 10.4 and 10.6 (with using annotation) and Tables 10.3 and 10.5 (without annotations).
Remarkably, usage of transcriptome annotation for TopHat and ContextMap had no or only very little influence on the accuracy for complete reads. In contrast, we observed a substantial improvement for junction reads that was much more pronounced for the human data sets than for the mouse sets. One possible explanation for this difference might be that coverage of splice junctions by reads is often much higher in the mouse data than in the human data, thus allowing good prediction of splice sites already without annotation. An alternative explanation is that the human data sets were simulated based on exactly the same transcriptome annotation as used for the mappings while the mouse data set was simulated using 11 different sets of annotations. Accordingly, for the human data the annotation provided to ContextMap covers all splice junctions, thus improving accuracy substantially. In contrast, in the mouse data many splice junctions are novel compared to the Ensembl annotation. This may also provide an explanation why RNASEQR performed significantly worse on the mouse data sets compared to the human data. Furthermore, performance of RNASEQR was substantially influenced by the sequencing error rate in the data set. Both for mouse and human, accuracy of RNASEQR decreased by at least 5 percentage points when the average error rate was increased from 1 to 2%. A likely explanation for this is that RNASEQR uses a mode of Bowtie in which the maximum number of mismatches in the complete alignment is fixed. In this mode, at most 3 mismatches are allowed which reduces runtime but at the same time also reduces tolerance to sequencing errors. While the probability for more than 3 errors is <2% for the 1% error rate sets, it increased to 6 and 14%

for the human and mouse 2% error sets, respectively. Although in theory, RNASEQR should be capable of aligning these reads in the last BLAT-based step, its performance nevertheless is affected. In contrast, ContextMap alignment accuracy showed only little dependency on error rates. As a consequence, although ContextMap generally outperformed all other methods we evaluated, the relative improvement in performance was largest for the 2% error rate sets.

Error	Program	Complete reads		Junction reads	
		exact	relaxed	exact	relaxed
1%	TopHat	94.07	94.07	70.61	81.18
	MapSplice	86.52	86.52	73.63	83.10
	ContextMap	**95.92**	**95.99**	**77.02**	**83.51**
2%	TopHat	91.92	91.92	69.44	78.37
	MapSplice	73.39	73.41	61.47	71.91
	ContextMap	**95.96**	**96.05**	**76.05**	**81.99**

Table 10.3: Mapping results on **human** data sets (without annotation). Evaluation was performed separately for unspliced (complete) reads and reads crossing exon-exon junctions (junction reads). Furthermore, we distinguished between exact true positives that were correctly mapped at all base positions and relaxed true positives for which either the start or end position had to be identified correctly.

Conclusions

In this chapter, we presented a novel approach for the mapping of sequencing reads from RNA-seq experiments. In contrast to previous approaches, ContextMap fully exploits the information on the context of a read, i.e. reads mapping in the vicinity of a read considered. Accordingly, ContextMap consists of three steps. First, the contexts are determined based on a preliminary mapping. Second, reads unmapped in the first step are aligned to the context and existing alignments are extended based on potential splice sites suggested by other alignments. In this step, all alignments satisfying the maximum mismatch criterion are taken into account for each context. Finally, the best alignment for each read is determined first for each context and then the optimal context is determined for each read.

By addressing the problem of finding the best position for each gene separately for each context, the problem size is reduced considerably. This allows evaluating a much larger number of possible alignments and accordingly positions for each read within each context. Thus, instead of considering the context only for the prediction of splice sites as done by MapSplice, we can take it into account for both complete and junction reads.

Although ContextMap can also be applied to preliminary mappings from genome or transcriptome alignments, one major application is the refinement of mappings provided by other map-

Error	Program	Complete reads		Junction reads	
		exact	relaxed	exact	relaxed
1%	TopHat	94.07	94.07	77.75	88.14
	RNASEQR	94.83	94.86	**92.33**	**92.56**
	ContextMap	**96.99**	**96.88**	90.91	92.18
2%	TopHat	91.92	91.92	76.71	85.48
	RNASEQR	89.95	90.00	87.34	87.58
	ContextMap	**96.39**	**96.83**	**89.25**	**90.48**

Table 10.4: Mapping results on **human** data sets with support of an annotation. See Table 10.3 for a description of the notation.

Error	Program	Complete reads		Junction reads	
		exact	relaxed	exact	relaxed
1%	TopHat	95.08	95.08	79.09	86.29
	MapSplice	95.21	95.21	84.49	**89.12**
	ContextMap	**98.43**	**98.58**	**85.31**	88.65
2%	TopHat	90.98	90.98	76.93	82.92
	MapSplice	92.89	92.89	77.89	81.95
	ContextMap	**97.95**	**98.21**	**81.89**	**85.04**

Table 10.5: Mapping results on **mouse** data sets (without annotation). See Table 10.3 for a description of the notation.

ping algorithms such as TopHat and MapSplice. Based on the analysis on simulated RNA-seq experiments for human and mouse, we could show that our refinement using ContextMap could improve considerably upon the accuracy of both TopHat and MapSplice predictions. In most cases, refinements of either mappings by ContextMap outperformed both original mappings. Thus, even if large fractions of reads are already correctly mapped by both approaches, mapping quality can still be improved for a substantial number of reads by evaluating the support of alternative mappings in the context of all other reads. Furthermore, if finding the optimal parameter setting is difficult as for MapSplice on the 2% error data sets, ContextMap provides a useful alternative as it obtains high accuracy even if the original mapping quality was relatively low.

Error	Program	Complete reads		Junction reads	
		exact	relaxed	exact	relaxed
1%	TopHat	95.08	95.08	79.40	86.59
	RNASEQR	88.78	88.95	83.12	85.67
	ContextMap	**98.45**	**98.61**	**86.97**	**89.29**
2%	TopHat	90.98	90.98	77.24	83.23
	RNASEQR	80.39	80.57	73.82	76.46
	ContextMap	**97.89**	**98.16**	**83.25**	**85.40**

Table 10.6: Mapping results on **mouse** data sets with support of an annotation. See Table 10.3 for a description of the notation.

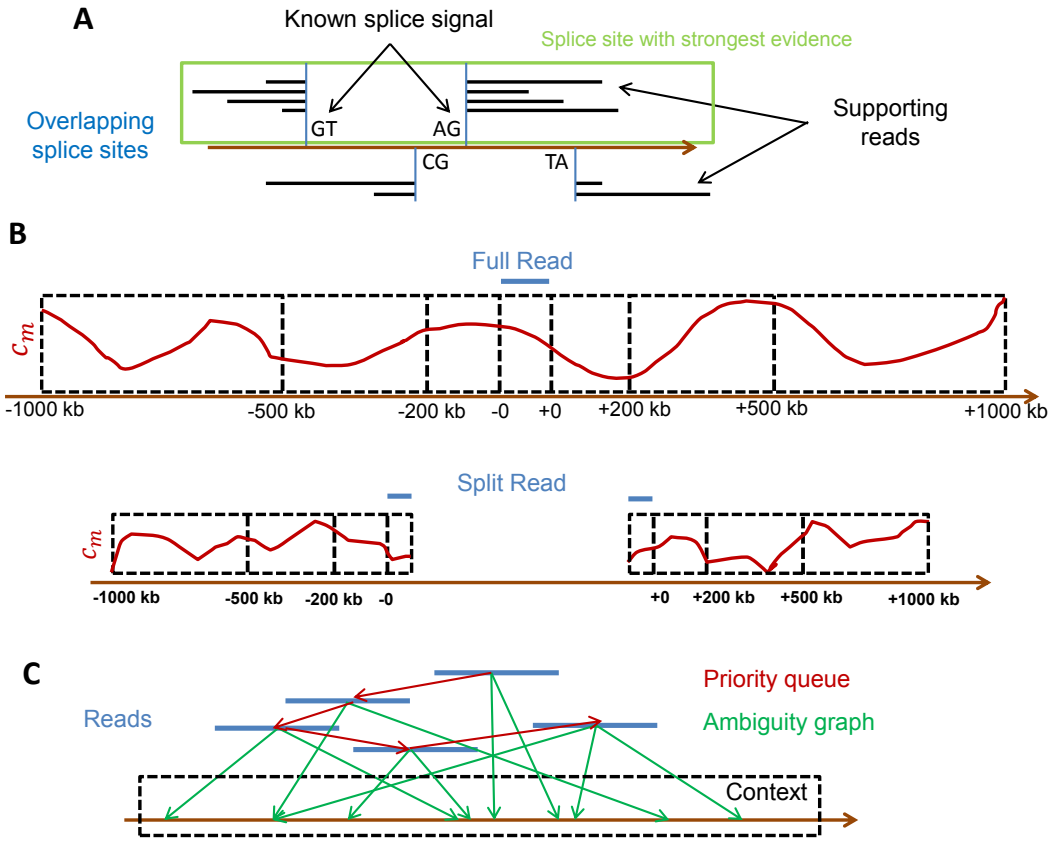


Figure 10.3: A) Among overlapping splice sites, the one with the best evidence is chosen. Here, two potential splice sites are shown (blue) with distance less than the read length. Evidence is provided by reads supporting the splice site (black lines) as well as known splice signals. In this example, the left-most splice site has more reads supporting it (4 compared for 2 for the right-most splice site) as well as the canonical splice signal GTAG. Thus, this splice site is chosen and the other one is discarded. B) For ambiguous mappings of the same read within a context, coverage scores are calculated based on the number of reads mapping per position (c_m, red curves) within the read alignment region and increasingly large windows around the read alignment. Here, -0 indicates the aligned position of the read start and +0 the position of the read end. A negative sign indicates positions upstream of the alignment and a positive sign positions downstream. Region 1 includes the positions from -200 to -0 and +0 to +200 bp, region 2 the positions from -500 to -200 and +200 to +500 bp, and so on. For split alignments only aligned read positions as well as positions upstream of the read start and downstream of the read end are included in the coverage calculation. (C) For the final ambiguity resolution for each context, reads (blue) are connected in a graph (green) to all possible locations in the context they can be mapped to. Furthermore, a priority queue (red) is created in which the reads are sorted according to the difference in coverage scores between the best and second-best alignment. Iteratively, the read with the currently largest difference is removed from the queue, its position is fixed for the context and coverage scores in the queue are updated.

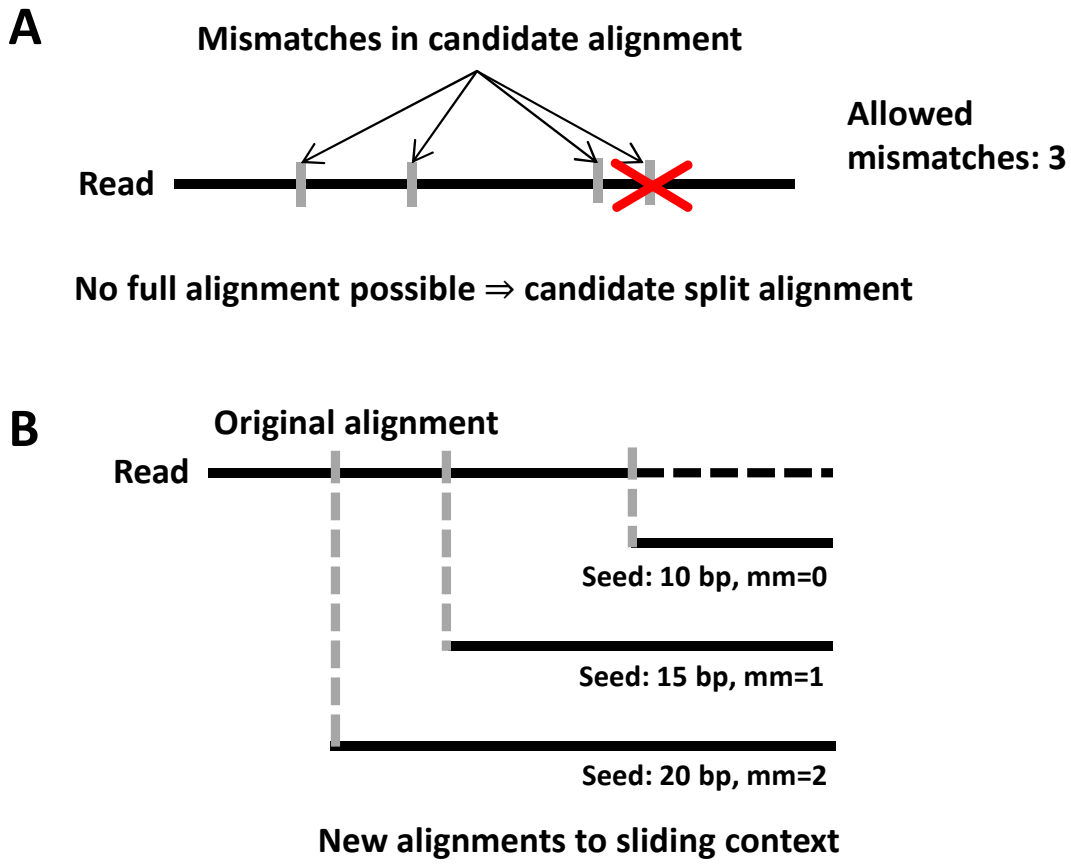


Figure 10.4: Identification of split alignments. Alignments exceeding the maximum mismatch criterion (A) are tested for a possible extension to a split alignment (B). For this purpose, three types of alignments starting from the other end of the read are performed within the same sliding context: alignments with a seed of (i) 10 and no mismatch (mm) up to the last valid mismatch position in the read according to the original alignment, (ii) 15 and at most 1 mismatch up to the second to last valid mismatch position, (iii) 20 and at most 2 mismatches up to the third to last valid mismatch position. If none of these alignments can be performed because the remaining length of the read is shorter than the seed length for all three alternatives, the alignment is classified as partial.

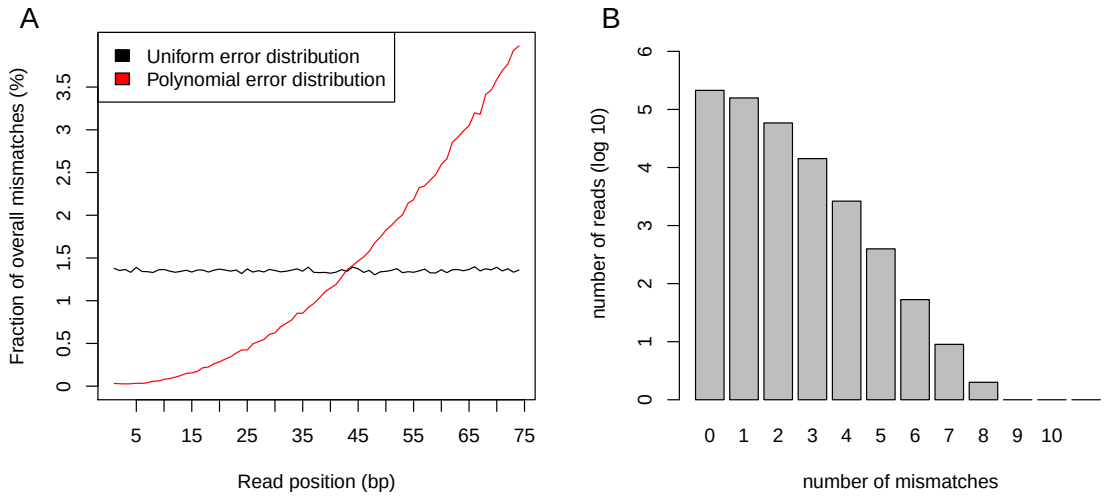


Figure 10.5: (A) Distribution of mismatches along the read length are shown for the uniform and polynomial mismatch distributions. (B) Distributions of the number of mismatches per read are shown for the human data set with 1% error rate.

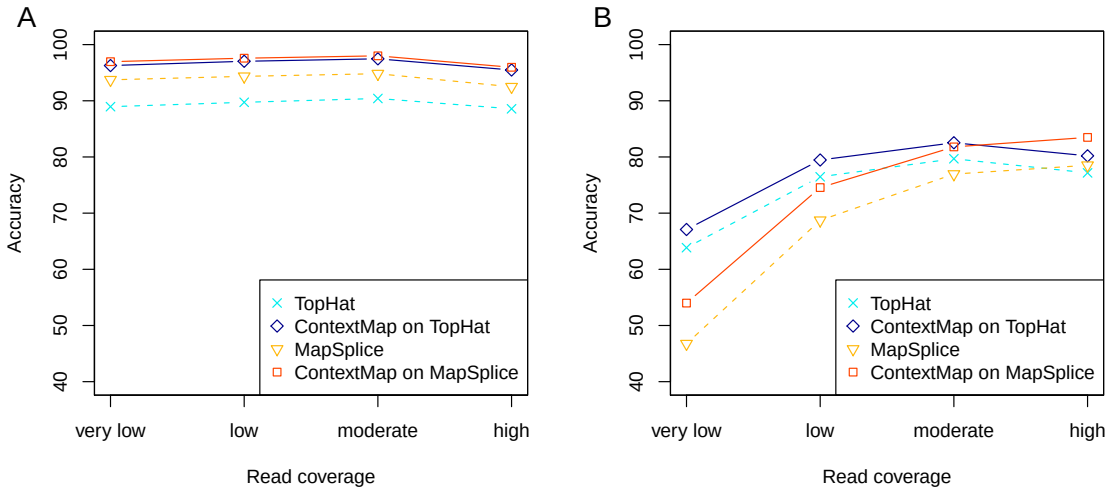


Figure 10.6: **Dependency of alignment accuracy on read coverage**. Alignment accuracy with respect to read coverage is shown for the mouse data set with 1% error rate: (A) complete reads and (B) junction reads. For this purpose, genes were binned into four approximately equal-sized groups based on the average read count per base and accuracy was calculated selectively for each group. For complete reads, no dependency was observed, whereas for junction reads, accuracy is strongly dependent on the read coverage.

Chapter 11

From sets to graphs: towards a realistic enrichment analysis of transcriptomic systems

The current standard for evaluation of high throughput transcriptomical experiments is: derive the set of significantly (differentially) regulated genes (the experimental outcome) and run a gene set enrichment method to find out which function or process is overrepresented (OverRepresentation analysis, ORA, GO gene set enrichment analysis, GSEA) in the outcome. Meanwhile, there are approaches to deal with the statistical problems due to the multiple testing and the dependencies between genes. In [72] a method is proposed to calculate the significances for for a meaningful question namely, how likely it is that a biological replicate would lead to the same results. Still, the concept of gene sets by design cannot model the regulation - to do so one needs relations and interactions between the entities (regulators and regulatees), a network.

In this chapter we present the gene graph enrichment analysis (GGEA) method explicitly shifting the analysis from sets to networks of genes. To do so we take human annotated gold standard gene regulatory networks (GRN) and test the *consistency* between regulator and regulatee in the experimental outcome. The underlying - strong - assumption is, that experiments both capture the activity of the transcription factors (the regulators) and the target genes. Still, the idea of consistency evaluation gets very close to mechanistic regulatory explanations. The extension of this idea from consistent edges to explanatory networks is presented in the RelExplain method (chapter 12).

The content of this chapter is mainly the work of Ludwig Geistlinger. It has been published in 2011 [68] and was presented on the ISMB conference in 2011. My main contribution focused on the evaluation of the method.

For this we also introduced a novel evaluation type based on the idea of the explanation power - the property optimized by the RelExplain method.

11.1 Introduction

Transcriptomic studies measure gene expression in different conditions. Striking genes, which are differentially regulated between the conditions, are of primary interest and investigated for common features and membership in group of genes, which have the same function or belong to the same biochemical pathway.
A first impression of similar behavior of genes can be achieved via clustering of genes [52]. The usually more effective overrepresentation analysis (ORA) tests the overlap of a predefined group of genes and the set of differentially genes assuming the hypergeometrical distribution under the null hypothesis [27]. The method is widely accepted and has been subject to modifications of diverse visual and model related features (see [105] for an overview), though the basic statistical principle remained unchanged. However, [72] criticize that the sampling procedure of ORA is statistically invalid and leads to a hazardous interpretation of the resulting p-value. Furthermore, the concentration on the usually small group of significantly differentially expressed genes, compared to the set of all the other, usually thousands of genes analyzed in the study that are ignored, is not suitable for an investigation on a global scale.
Both points of criticism are resolved in *Gene Set Enrichment Analysis* (GSEA) as it uses a valid sampling procedure and computes over the whole scope of genes [172]. A Kolmogorov-Smirnov test statistic is applied to test whether the ranks of the p-values of the genes in the gene set can be a sample from a uniform distribution. Several modifications of GSEA have been published (see [47] for an overview).
Though ORA and GSEA are convenient in the analysis of genes that are independently expressed, a serious problem arises when these methods are applied to gene set definitions extracted from regulatory networks and metabolic pathways. The assumption of independence among set members does not hold anymore; genes are found to be correlated due to mechanisms of co-regulation and co-expression. Initial steps to deal with that problem include implicit accounting for the correlation structure e.g. [15] and integration of network topology of undirected interaction networks e.g. [182]. Based on these first efforts, [119] have proposed *Gene Network Enrichment Analysis* (GNEA) that uses ORA to test for overrepresentation of gene sets in transcriptionally affected subnetworks of a global interaction network.
As the sign of gene expression changes and the direction of regulatory interactions are so far not taken into account, substantial features of the data are still ignored and the dynamics of the transcriptomic system are not realistically reflected. Activation and inhibition are essential regulatory mechanisms in the transcriptional machinery of the cell and are causes for up- and down-regulation of particular genes. Hence, we explain positive correlation in gene expression with activating edges of the transcriptional network. Vice versa, we assume inhibition to cause observed anti-correlation in gene expression patterns. In our following definition of *Gene Graph Enrichment Analysis* (GGEA), we exploit both fundamental regulation types in a novel enrichment framework for signed and directed gene regulatory networks, to judge whether the topology of the network is well fitted by the expression data.

11.2 Methods

11.2.1 Gene Graph Enrichment Analysis (GGEA)

Given gene regulatory information, for example extracted from biochemical pathways or a global transcriptional network, a gene set under investigation and gene expression data sampling different conditions, GGEA performs three essential steps: First, the gene set is mapped onto the underlying regulatory network, yielding an induced subnetwork. That is the affected part of the network, which consists of edges that involve members of the gene set. Second, each edge of the induced network is scored for consistency with the expression data, i.e. the signs of the expression changes of two interaction partners are evaluated for agreement with the regulation type (activation/inhibition) of the link that connects both genes. Third, the edge consistencies are summed up over the induced network, normalized and estimated for significance using a permutation procedure.

11.2.2 Experimental setup

In the following, we consider the classical setup of a transcriptomic study. This incorporates a set G of usually several thousand genes g_i ($i = 1, \ldots, n$) measured for differential expression between two conditions, each represented by a group of samples $S_1 = \{s_1, \ldots, s_k\}$ and $S_2 = \{s_{k+1}, \ldots, s_m\}$, respectively. The function

$$\text{expr} : G \times (S_1 \cup S_2) \to \mathbb{R} \tag{11.1}$$

returns the expression value for a gene and a sample at a time.

11.2.3 Measures of differential expression

The most intuitive measure for expression changes of a single gene between two conditions is the fold change

$$\text{fc} : G \to \mathbb{R}, \tag{11.2}$$

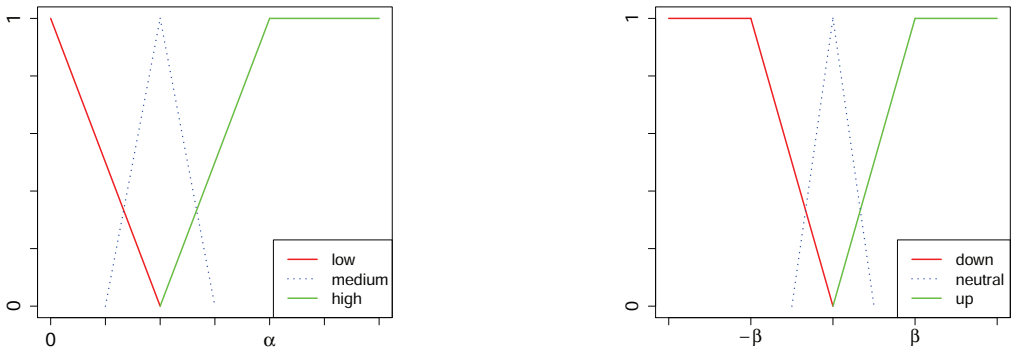
defined as the ratio of the mean expression values of a particular gene in both sample groups

$$\text{fc}(g_i) = \frac{\langle \text{expr}(g_i) \rangle_{S_1}}{\langle \text{expr}(g_i) \rangle_{S_2}}, \tag{11.3}$$

where $\langle \text{expr}(g) \rangle_S$ denotes the mean expression level of gene g in condition S. We compute t-test derived p-values to assign the statistical significance of the expression changes [149]. For simplification, we transform both measures to a logarithmic scale

$$\tilde{\text{fc}} := \log_2(\text{fc}), \quad \tilde{p} := -\log_{10}(p), \tag{11.4}$$

and set the significance thresholds α and β for $\tilde{p}$ and $\tilde{\text{fc}}$, respectively. Default values are $\alpha = -\log(0.05)$ and $\beta = 1$. In addition, we divide the range of both measures into two main categories and smooth the border via introduction of a degree of uncertainty according to the concept



(a) p-value (b) fold change

Figure 11.1: **Fuzzyfication of p-value and fold change.** Both measures are mapped onto two main categories, each having a membership function to express the uncertainty of the mapping. An optional third category is introduced in case of noisy data.

of *fuzzyfication* proposed by [109]. For the fold change, we map

$$(\tilde{\text{fc}} < 0,\ \tilde{\text{fc}} > 0) \mapsto (\text{down},\ \text{up}), \tag{11.5}$$

and compute membership values for both categories via the weighting functions $w : \tilde{\text{fc}} \mapsto [0, 1]$ (displayed in Fig. 11.1), resulting in a pair

$$\langle \tilde{\text{fc}} \rangle \ := \ \langle\ w_{\text{down}}(\tilde{\text{fc}}),\ w_{\text{up}}(\tilde{\text{fc}})\ \rangle. \tag{11.6}$$

Analogously, we map $\tilde{p}$ to areas of *low* and *high* significance. A third category can optionally be introduced to filter out unspecific signals in case of very noisy data. Table 11.1 shows how the fold change and p-value categories are combined to a single measure of differential expression

$$\text{de} : (\text{down},\ \text{up}) \times (\text{low},\ \text{high}) \mapsto (\text{reduced},\ \text{enhanced}),$$

in order to simultaneously summarize and express whether the transcriptional activity of a particular gene is reduced or enhanced in one sample group, as compared to the other.

	down	up
low	neutral	neutral
high	reduced	enhanced

Table 11.1: Combination of $\langle \tilde{\text{fc}} \rangle$ and $\langle \tilde{p} \rangle$.

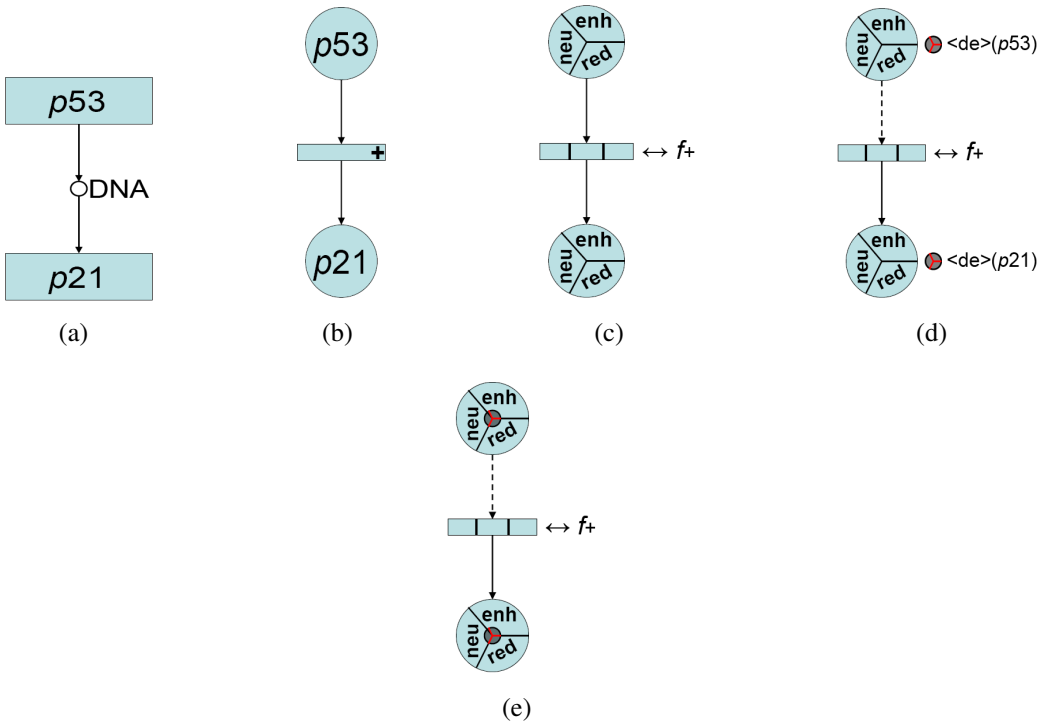


Figure 11.2: **Transformation of a gene regulatory relation to a Petri net (PN) transition with fuzzy logic.** The KEGG-like illustration of the transcriptional activation of $p21$ via $p53$ in (a) is first reconstructed in (b) as a basic PN transition. The regulation type is remembered. Subsequently, places and transition are fuzzified in (c) and the transition is associated with the fuzzy activation function f_+. Fuzzy $\langle \text{de} \rangle$ tokens are computed for both genes in (d) and, respectively to places, outgoing arcs are replaced by effect arcs (dashed). Eventually, the places are marked by their corresponding tokens in (e).

Application of the above rule set to a given fuzzyfied fold change and p-value is carried out via product-inference, i.e. each categorial weight of the $\langle \tilde{\text{fc}} \rangle$ instance is multiplied by each entry of the $\langle \tilde{p} \rangle$ instance.

This yields a 2×2 output table. Weights for the same output category, e.g. *enhanced*, are summed up, resulting in the desired output 3-tuple

$$\langle \text{de} \rangle \;=\; \langle\, w_{\text{reduced}},\; w_{\text{neutral}},\; w_{\text{enhanced}} \,\rangle. \tag{11.7}$$

which is, in conclusion, a measure for the direction and the significance of a expression change at the same time.

Induced gene regulatory networks

Enrichment analysis is the determination of significant gene sets out of a predefined universe of gene sets U, s.t. result sets accumulate striking features of the gene expression data. GGEA maps each gene set $u \in U$ onto a network of pairwise gene regulatory information I extracted from regulatory pathways or a global transcriptional network (see the below study setup section for an example). Each regulatory link $i \in I$ between two genes $g_1, g_2 \in G$ is a pair

$$i := (g_1, g_2)^* \quad * \in \{+, -\} \tag{11.8}$$

associated with a sign, which indicates whether g_1 activates or inhibits g_2. For a gene set $u \in U$ we construct the *induced* local gene regulatory network

$$I(u) := \{i = (g_1, g_2) \in I | (g_1 \in u) \vee (g_2 \in u)\}, \tag{11.9}$$

s.t. for each gene g of the gene set u all regulatory information is extracted, where g is either the regulating or the regulated gene.

Gene regulatory networks as Petri nets

Petri net models are well established in information theory [135] and have been extensively applied to biochemical processes, like metabolic pathways [69] and gene regulatory networks [31]. Based on that previous work, we construct a model for our problem as follows (Fig. 11.2). Given a gene regulatory network (GRN) under investigation, which consists of a set G_{GRN} of genes and a set E_{GRN} of regulatory edges between them, we construct the corresponding Petri net (PN) by reserving a place node for each gene $g \in G_{\text{GRN}}$, and establishing for each egde $e \in E_{\text{GRN}}$ a corresponding link between two places, which is mediated by a transition node that properly reflects the regulatory nature of the edge, i.e. activation or inhibition (Fig. 11.2(b)).
Given further a gene expression dataset, which satisfies the experimental setup described above, we calculate for each gene $g \in G \cap G_{\text{GRN}}$ the fold change and corresponding p-value, as it has been indicated above and exhaustively described elsewhere .
Following the work of [109], we transform the initial basic PN to a PN with fuzzy logic. This

	reduced	neutral	enhanced
f_+	reduced	neutral	enhanced
f_-	enhanced	neutral	reduced

Table 11.2: Firing rules for activation and inhibition.

incorporates the replacement of all simple place nodes by three-dimensional fuzzy place nodes

and similarity, the replacement of all simple transition nodes by corresponding three-dimensional fuzzy transitions (Fig. 11.2(c)). In addition, each transition t is associated with a fuzzy function f_t, which is either f_+ or f_- (as defined in Table 11.2), depending on the remembered regulation type of the transition at a time.
Subsequently, we fuzzify for each g its fold change and p-value, combine both quantities to the diff. exp. measure $\langle \text{de} \rangle$ (Figure 11.2(d)), and mark the corresponding fuzzy place nodes with the fuzzy $\langle \text{de} \rangle$ values (Figure 11.2(e)).
In a last adjustment step, we replace each link that points from a place to a transition by an effect arc, s.t. the firing of a transition keeps the marking of the corresponding input place unchanged.

11.2.4 Consistency of network constraints

The major problem of set enrichment strategies, when applied to GRN based gene sets, is, that they accumulate evidence for differential expression of single genes to express the enrichment of the whole set. Interfering and potentially contrary constraints of the underlying GRN are ignored. For example, two significantly up-regulated genes increase the enrichment of the set even if one gene inhibits the other. For that reason, we introduce the concept of consistency.

> **Definition** (*consistent edge*): An edge between two genes of a GRN is called *consistent* with the given expression data, if the expression behavior of both genes agrees with the interaction type of that edge.

This means for the above example that an up-regulated inhibitor should result in reduced expression of the affected gene.
Applied to the PN constructed above, a *consistent* transition t with fuzzy regulation function f_t between an input place A and an output place B satisfies

$$\langle \text{de} \rangle(B) = f_t(\langle \text{de} \rangle(A)). \tag{11.10}$$

Proof of consistency requires t to fire exactly once, yielding an output token composed according to the rules of f_t and the sum-product logic that has led to equation (11.7).

Penalizing and scoring

To determine if, or better, to which extent t is consistent with the given expression data, we calculate the consistency penalty

$$P(t) := |\delta\,[\langle \text{de} \rangle(B)] - \delta\,[f_t(\langle \text{de} \rangle(A))]| \tag{11.11}$$

as the absolute difference between the produced token $f_t(\langle \text{de} \rangle(A))$ and the token $\langle \text{de} \rangle(B)$ that has marked place B alone, before t was fired. According to [109] both tokens are previously defuzzyfied by center of gravity defuzzyfication δ, i.e. the categorial 3-tuples are backtransformed to numeric scalars.
We compute the GGEA consistency score for a GRN via transfer of these considerations for single transitions to the whole GRN-PN with the set of transitions T:

1. Firing of all $t \in T_a \subseteq T$, the set of all *active* transitions. These are all transitions having a non-empty input place, i.e. a place assigned to a gene $g \in G \cap G_{\text{GRN}}$.

2. Computation of the raw consistency score

$$S := \left(\sum_{t \in T_a} P(t) \right)^{-1}. \tag{11.12}$$

3. Normalization of the raw consistency score by the number of active transitions $|T_a|$

$$\bar{S} := \frac{S}{|T_a|}. \tag{11.13}$$

Significance and ranking

According to the recommendations of [72] and [67], statistical significance of the consistency score is estimated via a permutation approach based on subject sampling, which is defined in a self-contained way:

1. Permute group assignment of samples N times.

2. Recalculate differential expression measures for each permutation.

3. Recalculate consistency score for each permutation.

4. Find the consistency p-value as the proportion of permutation scores that are larger than the observed score.

We compute the consistency p-value for each gene set $u \in U$ and rank the gene sets by the adjusted p-values, i.e. p-values corrected for multiple testing. Gene sets below the chosen significance niveau are classified as *significantly and consistently* enriched.

Extensions

To apply to regulation processes involving multiple regulators and transcription complexes composed of several genes, we allow a transition t to have an arbitrary number of inputs $I_t = \{i_t^1, \ldots, i_t^k\}$ and outputs $O_t = \{o_t^1, \ldots, o_t^l\}$. This is accomplished via generalization of equation (11.10) to

$$(\langle \text{de} \rangle(o_t^1), \ldots, \langle \text{de} \rangle(o_t^l)) = f_t \left[(\langle \text{de} \rangle(i_t^1), \ldots, \langle \text{de} \rangle(i_t^k)) \right]. \tag{11.14}$$

Following the explanations of Figure 5 in [109], we model the combined effect by the minimum (*AND*), the maximum (*OR*) or the average (*MEAN*) of the effects, depending on the particular case.

Missing data, i.e. genes $g \in G_{\text{GRN}} \setminus G$ of the GRN, which are not spotted onto the chip, is resolved using transitivity. By going up and down, respectively, the regulation path until a non-empty place is reached, an empty origin is filled with the found token, which is adjusted to path length of the transitive relation. The adjustment is due to the fact that the evidence for regulation weakens, as the path length increases.

Benchmark setup

Data sampling and network construction

Gene expression data of *E. coli* was collected and sampled from the M3D database (Many Microbe Microarrays Database [56]). 1000 datasets were designed in a two-class fashion, s.t. each class contained 15 samples. It was assured that real-world distributions of fold changes and differential expression p-values were matched. A global gene regulatory network for *E. coli* was constructed using the regulatory interactions provided in the RegulonDB database [64]. From the union of all stored TF/gene, TF/operon, TF/TF, σ/gene and σ/TU regulatory interactions (TF stands for *transcription factor* and TU for *transcriptional unit*), we removed duplicated and ambiguous edges. The final network connected 2097 unique nodes by 5784 edges, which were clearly annotated as either activating or inhibiting.

Tool collection and gene set definitions

For each dataset, we applied the standard hypergeometrical overrepresentation test ORA1, and a collection of array resampling methods that have been claimed by [67] to correctly control false positive rates and gene correlation patterns. These are the modified resampling overrepresentation test ORA2 [72], SAFE [15], GSEA [172] and SAM-GS [47]. The gene set catalogue for analysis were defined on the one hand according to the KEGG pathway annotation [144] for *E. coli*, and, on the other hand, according to the GO classifications [12] of *E. coli*.

Consistency benchmark

For each tool, we collected for all datasets with statistical significant outcome ($p < 0.05$) the top ranked gene sets, chose 700 sets at random from them and computed the percentage of consistent relations in the corresponding induced regulatory networks. Regulation direction and strength were distinguished. Activating relations required both interactions partners to be expressed in the same direction to be consistent, while inhibiting relations required them to be expressed in the opposite direction. Regulation strength was categorized as weak and strong, depending on the differential expression p-value of the regulator. We chose 0.5 and 0.05, respectively, as the threshold for both categories. To estimate the null distribution in each category, one gene set out of the catalogue were randomly chosen for each of the 1000 datasets. We sampled 700 from them at random and computed the corresponding consistencies.

Case study setup

FiDePa and local GGEA

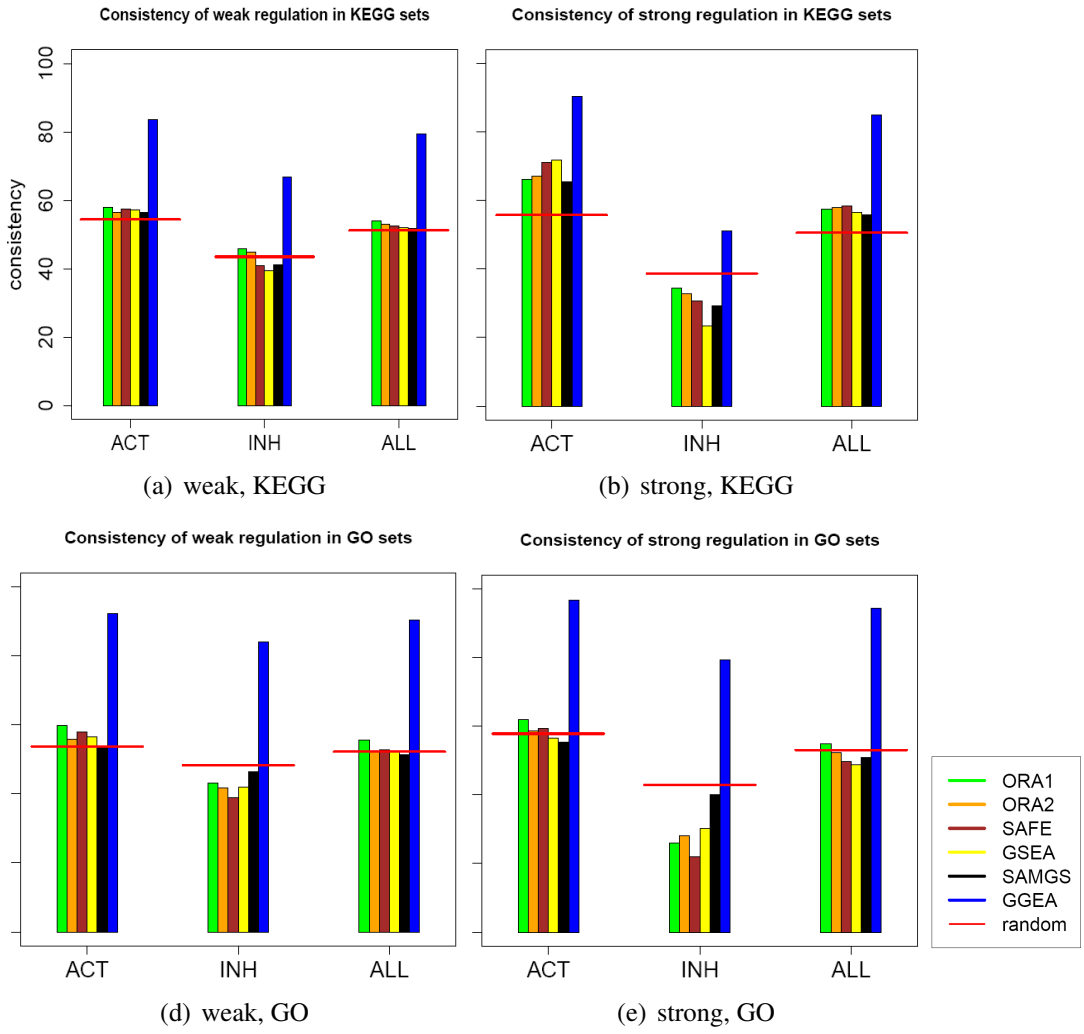
We applied GGEA to the glioma dataset that has been investigated before with the method FiDePa [103]. The method exploits GSEA first to determine striking paths of a particular length and uses resulting differentially regulated paths for the construction of a consensus network,

which is subsequently tested for overrepresentation of gene sets. In a similar approach, we computed consistency scores of regulatory links in all human non-metabolic KEGG pathways (gene regulatory and signaling pathways) and the ten edges with the highest consistency score were extracted from each of them. Duplicated edges were removed and the consensus graph was further reduced via application of a high pass consistency filter using the mean consistency score as threshold. That yielded a total of 378 edges connecting 342 unique nodes, which were tested, as in FiDePa, for overrepresentation.

11.3 Results

11.3.1 Consistency study

We investigated the degree to which regulatory interactions are consistent within results of gene set enrichment methods. We conducted a meta-analysis of 1000 *E. coli* datasets and evaluated the consistency of activatory and inhibitory relations found in the transcriptional network of *E. coli*. The study setup, incorporating data sampling, network construction, tool collection and gene set definition, is described in METHODS. The consistency benchmark and the classification of interaction strength as *weak* and *strong* is defined there as well. The results are shown in Fig. 11.3. We observe that the set enrichment tools systematically neglect mutual regulation among set members. For KEGG gene sets, weak regulations (Fig. 11.3a) are only randomly consistent and the gene set with maximal consistency is frequently not reported by the set enrichment tools. This holds for both, activatory and inhibitory links, and thus, for the total consistency too. Strong activators, which had an individual expression change of high statistical significance, and the effects on their targets are more consistently aligned (Fig. 11.3b). All five set enrichment methods are more than ten percent better than random, with GSEA achieving the highest percentage among them. However, the consistency gained in strong activations is lost for strong inhibitions in almost equal amount. This phenomena will be discussed in the next section.
The results for weak regulation in KEGG sets are nearly replicated in GO gene sets (Fig. 11.3c). Yet, the positive effect seen for higher signals of activators is not observed. Strong inhibitions are again highly inconsistent, with only SAM-GS performing around the null value (Fig. 11.3d). In contrast, GGEA, which has been optimized for this criterion, yields the most consistent gene sets. The optimum is almost achieved in all categories for both, KEGG and GO gene set definitions. Activating and inhibiting regulations links are nearly equally consistent, if adjusted to background distributions of both regulation types (see next section), and stronger signals are properly weighted in order to preserve the regulation kinetics. Although stronger signals have an higher impact on the GGEA score, also weak regulation processes are highly consistent in the sets found by GGEA. In general, these findings are more pronounced for GO sets, as compared to KEGG gene set definitions.



(a) weak, KEGG (b) strong, KEGG

(d) weak, GO (e) strong, GO

Figure 11.3: **Consistency of Regulatory Interactions in Top Ranked Sets.** Each of the tools ORA1 (green), ORA2 (orange), SAFE (brown), GSEA (yellow) and SAM-GS (black) were applied to 1000 *E. coli* datasets using KEGG and GO gene set definitions, respectively. From datasets with statistical significant outcome, the top ranked gene sets were collected and investigated for consistency of weak and strong regulations. Consistency was determined for activating (left bar plot panel) and inhibiting (middle) relations, as well as for relations of both directions in total (right). Equally computed GGEA results are displayed in blue. The null consistencies were estimated via randomly chosen sets and are indicated as horizontal red lines.

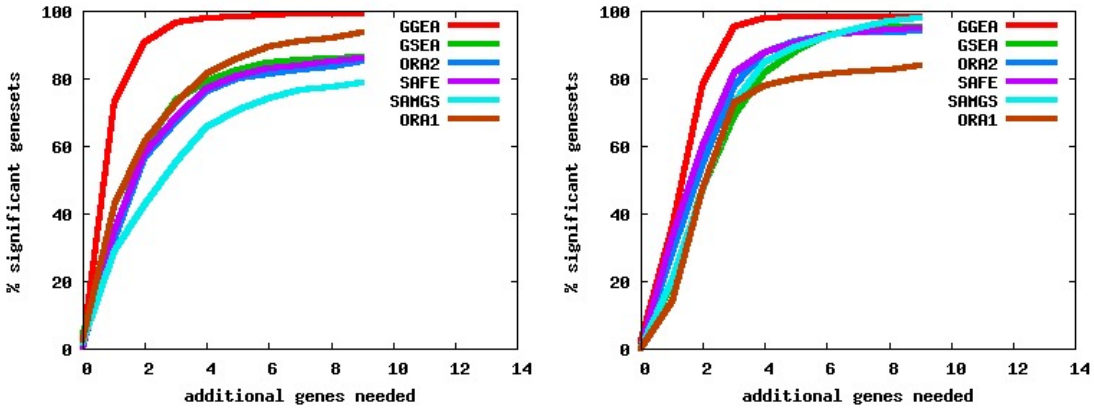
11.3.2 Explainability study

As GGEA incorporates the consistency in the set significance score, the consistency study only shows that using this information in fact leads to better consistency. Therefore we performed a second, more independent benchmark: the explainability study.
If a gene set is reported to be significant for an experiment by some method, it implies that the biological process or pathway corresponding to this set is regulated differently between the experimental conditions. This in turn implies, that there exists a regulatory network connecting the significantly differentially expressed genes in this gene set. So to interpret a geneset in the context of the experiment one would like to investigate the already known regulatory network (a subgraph of RegulonDB in this case) connecting the differentially expressed genes. In the best case the genes in question are connected within the reported gene set, so it is clear show the genes relate to each other in the given biological process.
Sometimes - actually very often - however the defined gene sets do not contain one or more important regulators or mediator genes and so are not connected to each other. In this evaluation we calculate the minimal number of additional genes not contained in the significant gene set which make the differentially expressed genes in the set connected, and make so the process interpretable. The less additional gene is needed the easier the interpretation: a single additional gene could possibly imply a regulator or mediator not contained by the set, but playing a role in it's regulation, while if several additional genes are needed one have to make more complex assumptions to make result interpretable. For the explainabilty study we explicitly make the input regulatory network undirected - this way we generalize the edges so that also possibly unknown inverse regulations are allowed.

To evaluate this quality we used the 700 most significant sets for all tools for both the KEGG and GO sets. As a measure of explainability we calculated for given needed additional gene number x the percentage of the reported gene sets where these can be explained by $\leq x$ additional genes. To make the For the explainability study we used only the knowledge that a given pair of gene interact in some form, but not the type of the interaction As the existence of some interaction between a pair of genes is the directions of the edges

The results are shown in Fig. 11.4. While the performance and ranking of the other methods depends on the set definitions, GGEA consistently reports easier explainable sets for both KEGG and GO than all other methods. Similar to the results of the consistency study the gap is much bigger between the performance of GGEA and the other methods when using GO sets definitions. For example GGEA needs a single additional gene to make the differential expressed genes in the set connected in 73% of the cases where ORA can explain only 42% of the sets with a single addition gene (SAFE 2: 35%, GSEA:34%, SAFE1:32%, SAMGS:29%). If allowing two additional genes GGEA can explain more than 90% of all reported genesets, while all other methods produce results only around 60% or below.

We find this evaluation is particularly interesting as it tries to approximate the process of the



(a) GO (b) KEGG

Figure 11.4: **Explainability of Expression Changes in Top Ranked Sets.** The 700 top ranked gene sets (introduced in the consistency study above) of each tool were restricted to genes with expression changes of statistical significance. For each top-ranked gene set of each tools, we computed the minimal needed genes that are not in the gene set, but are needed to connect the significantly regulated genes to a regulation network. The plot shows the percentage of the genesets for each tool for which x or less additional no-set-member genes are needed to create a connected regulatory network. So for example in case of the GO sets a single additional gene makes 73,14% of the significant gene sets for GGEA connected and so interpretable, while for example in case of ORA or SAMGS a single genes makes only 42.43% or 28.86% of the gene sets interpretable.

human interpretation and neither GGEA nor the other methods optimize the connectivity of significantly differential expressed genes.

11.3.3 Case study

In a final case study, we have transferred GGEA to regulatory constraints occurring in human non-metabolic pathways. We investigated two expression datasets of human neuronal tumors and compared results of GGEA to results of set enrichment strategies. In a first analysis, we applied GGEA to the glioma dataset that was investigated before by [103] with the method FiDePa (described in METHODS). We observe large agreement in the result lists of both methods; 17 pathways listed in the FiDePa result do also occur in the top 25 of the GGEA ranking. The positive control Glioma is better ranked (and has higher significance) by GGEA. Further, several unspecific and disease unrelated pathways detected by FiDePa (Type I/II diabetes mellitus, Epithelial cell sig. in *Heliobacter pylori*, Adipocytokine signaling pathway, Cell cycle, Maturity onset diabetes of the young) are discarded by GGEA and replaced by specific, cancer related pathways (Renal cell carcinoma, Endometrial cancer, Non-small cell lung cancer, Acute myeloid leukemia). Regarding the the first place of both rankings, GGEA (Pathways in Cancer; not detected by FiDePa) gives a clear, first disease related hint, while FiDePa (MAPK signaling path-

Pathway	ORA p (GGEA)	ORA p (FiDePa)	Rank (FiDePa)
↑ Pathways in cancer	1.8e-24	–	–
↑ Focal adhesion	1.4e-18	2.5e-06	5
↑ T cell receptor signaling	1.2e-17	1.5e-05	7
↑ Neurotrophin signaling	5.5e-15	–	–
↑ Colorectal cancer	1.1e-14	9.4e-05	11
↑ Pancreatic cancer	3.8e-14	0.0001	12
↑ Renal cell carcinoma	1.3e-13	–	–
↑ VEGF signaling	1.5e-13	0.006	22
↔ Fc epsilon RI signaling	4.1e-13	1.9e-05	9
↓ Chronic myeloid leukemia	6.3e-13	1.65e-05	8
↑ ErbB signaling	8.9e-13	–	–
↑ B cell receptor signaling	4.2e-12	0.001	17
↑ Glioma	5.1e-12	0.003	20
↑ Insulin signaling	3.2e-11	0.001	18
↑ Leukocyte trans. migration	3.9e-11	0.01	24
↓ Adherens junction	4.9e-11	1.4e-05	6
↓ GnRH signaling	6.5e-11	0.0003	16
↓ Nat. killer cell med. cytotox.	6.5e-11	1.4e-11	2
↑ Wnt signaling	1.2e-10	–	–
↓ Toll-like receptor signal.	1.2e-09	5.5e-05	10
↑ Endometrial Cancer	1.6e-07	–	–
↑ Non-small cell lung cancer	3.4e-07	–	–
↑ Acute myeloid leukemia	3.9e-07	–	–
↓ mTOR signaling	1.2e-06	0.0002	15
↓ MAPK signaling	4.4e-06	1.6e-25	1
...	...	...	...
↓ Apoptosis	0.04	9.3e-11	3

Table 11.3: Result comparison of GGEA and FiDePa application to the glioma dataset. Arrows in the first column denote whether a pathway is ranked higher or lower by GGEA, as compared to FiDePa.

way) reports a general signaling process, which is part of the normal response to extra-cellular stimuli. In addition, the Neurotrophin signaling pathway, which promotes neuronal tumors via modulation of neuronal apoptosis [130], is also not identified by FiDePa, but highly ranked (rank 4) by GGEA. Instead, FiDePa detects general Apoptosis with high significance, what is indeed commonly down-regulated in most cancers, yet other pathways are found to have higher influence in this particular case.

In another evaluation study, we used neuroblastoma expression data that was investigated for en-

Pathway	p-value
Neurotrophin signaling	7.5e-06
Chemokine signaling	0.0004
Cell adhesion molecules (CAMs)	0.0021
Regulation of actin cytoskeleton	0.0068
Focal adhesion	0.0091
Nat. killer cell med. cytotox.	0.0092
Leukocyte trans. migration	0.0099
Pathways in cancer	0.01
T cell receptor signaling	0.016
Fc epsilon RI signaling	0.019
Long-term depression	0.023
Axon guidance	0.033
Vasc. smooth muscle contraction	0.035
p53 signaling pathway	0.035
Melanogenesis	0.039
MAPK signaling	0.043
Thyroid cancer	0.05

Table 11.4: Result of GGEA application to the neuroblastoma dataset.

richment of metabolic pathways before [163]. However, enrichment analysis of non-metabolic pathways with a suitable graph enrichment method remained to be done. The application of GGEA to the neuroblastoma dataset identified 17 significantly and consistently enriched pathways (Table 11.4). Best ranked is the Neurotrophin signaling pathway, which was already detected in the glioma study to play an essential role in the development of neuronal tumors. Determination of edges with highest consistency in that pathway allows a deeper insight into the disease causing dynamics, and displays dominant regulation features. The high affinity nerve growth factor receptor, which in humans is encoded by the NTRK1 gene, is up-regulated in neuroblastoma cells and activates the adaptor protein SH2B3, the growth factor receptor-bound protein 2 (GRB2), the Abelson murine leukemia viral oncogene homolog 1 (ABL1), the phospholipase gamma 2 (PLCG2) and the SHC-transforming protein 1 (SHC1). A literature inquiry revealed that all of the activated and associated proteins are proliferating, oncogenic and/or apoptosis influencing and thus, of cancer promoting importance e.g. [145] or [26]. In addition, the up-regulation of the whole NTRK1 proliferation module in neuroblastoma was experimentally validated [55] some years ago. In a similar sensitive manner, GGEA detects the Chemokine signaling pathway, as neuroblastoma impairs chemokine-mediated dendritic cell migration [189] and chemokines strongly promote neuroblastoma primary tumor and metastatic growth [128]. Determination of relations with the highest consistence displays a throughout consistently activated path of remarkable length starting at the up-regulated set of chemokine receptors (XCR1, CXCRs, CCRs).

The detection of the Fc epsilon RI signaling pathway builds a bridge to the results for metabolic pathways. In [163], only moderate attention (discussed in their supplement) is paid to the extremely significant findings for Phosphatidylinositol metabolism (p = 9e-12) and for several pathways concerning the metabolism of lipids and fatty acids, e.g. Fatty acid metabolism (p = 1.7e-9) and Glycerophospholipid metabolism (p = 3.9e-7), which are listed in Table 1 of that publication. Fc epsilon RI signaling has a regulatory impact on both - the Phosphatidylinositol metabolism via modulation of the phospholipase (affected by the Neurotrophin pathway); and the metabolism of lipids in general via stimulation of arachidonic acid synthesis. Arachidonic acid is a polyunsaturated fatty acid that is required for membrane phospholipid synthesis. It is also involved in cellular signaling and known to activate syntaxin-3, which causes cell membrane expansion of neuronal cells [42]. [163] explain the several revealed signals in lipid related metabolisms with TCA based energy production; the above findings imply that the observed activation of biosynthesis of fatty acids and lipids (which is based on the latter) is rather due to the increased requirement of neuronal membrane material (i.e. specific lipids) in the fast growing and dividing neuroblastoma cells.
Additionally, GGEA detects cancer related pathways (e.g. p53 signaling pathway) and pathways involved in responses to cellular malformations (e.g. Natural killer cell mediated cytotoxicity), which were also detected in the glioma study. These general pathways were also partly identified via standard GSEA performed by the PathWave developers see Supplement Tables S5a and S5b in [163].

11.4 Discussion

The hypergeometrical overrepresentation test is the most frequently used way to enrich the list of significantly differentially expressed genes with biological knowledge of groups of genes, which have the same function or work together in the same biochemical pathway. As there are several methodological issues of ORA, which have been recently criticized, array resampling procedures like SAFE, GSEA and SAM-GS have been developed to better control the fraction of false positives and to better reflect implicit correlation patterns found in the gene expression data.
We wanted to know to which extent experimentally validated regulatory interactions are *per se* consistent with the expression data in results of the established set enrichment strategies. We conducted a large-scale consistency study in 1000 *E. coli* chips and used the *E. coli* `RegulonDB`, which is currently the major electronically-encoded regulatory network of any free living organism, for the proof of consistency in significant result sets of the tools. As set enrichment strategies ignore the sophisticated dependency structure implied by the gene regulatory network, we found that the network constraints are only randomly consistent with the gene expression in these result sets. Even strong causal signals in pairwise directed regulations were not properly reflected in a crucial amount and inhibiting constraints were more seriously violated than activating constraints. This is on the one hand due to the incompleteness and bias introduced during the annotation of the regulation type, as there are more activating than inhibiting regulations in the database. Regulation pairs are found to be annotated as inhibiting, if there was at least one validating experiment, although it might be found to be activating in other conditions. Addition-

ally, inhibition is inferior to activation concerning the biochemical kinetics, as it requires more energetic effort and there are usually interfering activating regulators as well. On the other hand, set enrichment methods tend to best rank gene sets, wherein members are mainly expressed in the same direction.

One might argue that certain networks constraints are not effective in the analyzed gene expression data, as they might be inaccurately annotated and particular genes might be systematically contrary expressed. For example, a regulatory link is annotated as activating, but found to be rather inhibiting in the expression pattern of both connected genes over all expression measurements. For that reason, we have performed an explainability study where we ignored regulation type and differential expression direction, but only take into account whether two genes where connected in the global network and whether a gene is significantly differentially regulated. We observed that for all set enrichment tools only a small amount of genes could be explained by other set members in a significant result. We found that causing signals for strong expression changes are partly far away and showed that set enrichment can only tell that there is *something* significant happen in the result sets, but not if the observed expression changes are *reasonable* in respect to regulation kinetics.

GGEA resolved the above problems of set enrichment strategies in both studies. We found that significant result sets of GGEA properly brought the regulation constraints in line with the expression behavior of the genes under investigation. Activating links were nearly optimally consistent and inhibiting links were preserved in a large amount of regulations. In addition, significant expression changes could be mostly explained with clear signals of other set members or nearby mediators. Besides, we emphasize on the sensitivity of GGEA shown in the consistency study. Weak signals, which are usually ignored by set enrichment methods, were also consistently aligned by GGEA. In general, coherently consistent weak regulations are preferred over strong, but inconsistent signals, as this is expected to better reflect the *in vivo* situation.

In a final case study, we have transferred GGEA to regulatory constraints occurring in human non-metabolic pathways. We investigated two expression datasets of human neuronal tumors and compared our results to GSEA.

GGEA discovered positive controls to be differentially expressed without relying on the set enrichment paradigm, but by consistency scoring alone. As for other pathways that have been detected in common on these datasets, GGEA does not only support the GSEA hypothesis - that genes of these pathways show striking behavior - it additionally backs up the findings by stating that the interplay between these genes displays consistent dynamics. Furthermore, graph enrichment can rule out pathways that involve striking genes, but indicate regulatory links between them which are contrary to their expression. In that case, other pathways might be more suitable to explain the observed variance. On the glioma dataset, GGEA discovered throughout specific and disease related pathways. Tightly connected and induced by increasing specificity, the fraction of false positives decreases. Hence, unspecific and inconsistent pathways are replaced by more appropriate and biologically reasonable pathways in the respective result lists. An example is the detection of the Neurotrophin signaling pathway that modulates neuronal apoptosis (a very specific finding), while general apoptosis is downgraded. The Neurotrophin signaling pathway also has a major influence on the development of neuroblastoma, another neuronal tumor type. The experimentally verified connection was detected by GGEA with high significance, while

GSEA failed to detect it. The discovery of such false negatives of the set enrichment analysis is due to improved sensitivity already observed in the consistency study. Interestingly, the findings of GGEA for non-metabolic pathways in the neuroblastoma study work together with the PathWave outcome for metabolic pathways. It has been demonstrated via the Fc epsilon signaling pathway example that both tools complement one another and that their interplay allows for deeper analysis in a broader context. In combination, they provide an enrichment utility for all types of pathways. However, it is surprising that only GGEA is sensitive enough to detect the Neurotrophin signaling pathway, the Chemokine signaling pathway and the Fc epsilon RI signaling pathway - all of which have been shown to be of crucial importance in neuroblastoma formation - while standard GSEA does not detect them at all. Best ranked pathways are here: Cell cycle, Ribosom and Olfactory transduction. The connection to the disease is incomprehensible and explanations are almost arbitrary.

11.5 Conclusion

In this work, we have presented a novel algorithmic framework to detect enriched consistency between gene expression data and network constraints of gene regulatory networks. The method, called *Gene Graph Enrichment Analysis* (GGEA), performs a rigorous shift from the inappropriate set paradigm to a proper *graph* paradigm, i.e. it accounts for directed regulatory links between genes and scores them for consistent expression changes. We have shown by a variety of clinical studies that GGEA can confirm, support and enhance the confidence in set enrichment analysis, yet can also resolve results which are contrary to the biological situation. A large amount of evidence has repeatedly shown that GGEA better reflects pathway related dynamics and that it comes with higher specificity and sensitivity than current tools. As a result the fraction of false positive and false negative classifications decreases and the enrichment analysis quality improves.

Future directions

Due to the shown advantages, we strongly argue to combine gene expression data and network information whenever both is available at the same time. Emerging tools like PathWave and GGEA are first steps towards a more specific and realistic secondary analysis of microarray data. As it has been discussed in [47], it is frequently observed in enrichment analysis that only parts or certain functional modules of canonical pathways are differentially regulated, and not the pathway as a whole. That has given rise to the field of *gene set reduction* and other approaches subsequent to the enrichment analysis. We do consider local GGEA that extracts regulation chains of highest consistency and GGEA analysis for subnets as novel contributions to that research area. Development of an accurate reduction algorithm on gene graphs will be an issue for our future work.

The rapidly evolving field of *deep sequencing* analysis [195] is expected to replace the error-prone microarrays. Expression levels, i.e. quantities of certain mRNA molecules, are not mea-

sured by noisy binding of small representative pieces anymore, but by total sequencing of the cellular mRNA. Challenges for the future will include the adaption of existing enrichment strategies to this new kind of transcriptomic data.
Finally, combination and integration of other types of high-throughput data (e.g. proteomic data) into the enrichment analysis framework are desirable and required for an improved knowledge about the physiological state of an organism. In the particular case of metabolism and diverse signaling processes that incorporate small metabolic messengers, quantitative measurements extracted from metabolomic studies could be of high value too. Consistency expressions would then not be restricted to gene-gene relations alone, but could also be applied to the variety of gene-compound relations, and even to compound-compound relations. Recent studies have already combined genetic and metabolomic information e.g. [71] with promising results, which motivate a similar approach also in the field of enrichment analysis.

Chapter 12

RelExplain: Explaining data via networks

In this chapter we address the ultimate goal of the second part of this thesis: the development of a tool helping to explain the experimental output of high-throughput experiments. We utilize the context derived using the techniques discussed in chapter 8: the context of the currently known regulations, extracted from scientific publications. In addition, we add (less traceable) knowledge from large scale experiments and public databases resulting in a network containing regulatory information about different regulation layers. Even though we collect thousands of regulations, many of them are itself context dependent and, in most cases, the full "signature" of the regulation is not known. Therefore, to get closer to our ambitious goal we have to deal with the sparseness of knowledge and provide additional possibilities for the user to influence the automatic explanation.

The main idea of the RelExplain method presented here is to act iteratively and interactively. Each iteration consists of an automated and a manual step: first the algorithm calculates explanations given the experimental outcome and context knowledge (the network and partitions of the networks corresponding to biological processes), then the user may influence the algorithm by excluding/including edges or by focusing on specific biological processes. For the manual step the tool provides means to examine the evidence (such as the corresponding publication) of the used regulations - this way the large amount of regulations extracted with textmining are reduced to the ones relevant to the experiment in question. After the final iteration the user may end up either with an explanation with associated evidence for every single regulation, or - and this may be due to the sparseness of the current knowledge the likely result - no appropriate explanation. This means: the goal of the method presented here is to help concentrating the knowledge relevant for the experiment in respective explanations and let the user decide. Accordingly, the presented method to find an explanation is heuristic - the heuristic used is motivated by our current understanding. Due to the lack of the appropriate gold standards, it is a hard task to evaluate methods in general. In our case the task is even more complex: the human action is an integrative part of the method. Accordingly, instead of presenting a new ranking of relevant biological processes for a given experimental outcome, we motivate the usefulness by our method: by showing the need of human input for current state-of-the art methods as well for one of the best analyzed experiments from the literature.

The content of this chapter is based on a submitted manuscript, and is joint work with Evi Berch-

told, who in her masters thesis helped to extend the RelExplain prototype to its current version.

12.1 Introduction

High-throughput experiments yield measurements on thousands of genes or proteins. Often, such measurements are used to compare two or more sets of conditions of a system. Typically, differential objects between conditions are determined via fold changes and associated statistical significance estimates (we call this an *experimental outcome, EO)*. Of course, the idea is the larger the fold change and the higher the significance the more important is the respective object for the difference and the understanding of the conditions. Thus, the result of such an analysis is a sorted list of potentially relevant objects. The problem is that this list can be long and incomprehensible. Moreover, the approach has a major flaw: the assumption that genes can be relevant individually. A biological function is realized by several genes working together in a pathway. This led to the idea that a set of relevant genes is a meaningful solution explaining the differences between conditions, where not every gene in the set individually has the largest fold change or the highest significance. Since the invention of pathway scores by [208], which computed scores for pathways and ranked the members of a pathway library to find the one most compatible with the data (i.e. EOs), a lot of effort has been spent on developing gene set scoring, so called gene set enrichment analysis (GSEA). GSEA methods (such as ORA [72], SAM-GS [47], SAFE [15], GSEA [172], and GGEA (Chapter 11, [68])) take an EO and a list of gene set definitions and compute the ranked list of sets most compatible with the EO. Despite the many different methods for set enrichment, GSEA became a de-facto standard, as it resolves the issues outlined in [72] yielding more meaningful p-values for the gene sets. GSEA results in ranked lists of gene sets, but no explanations on the regulatory mechanism within this sets for the EO at hand. Of course, here the question is in order whether arbitrarily defined gene sets are appropriate to resolve such mechanisms. E.g., GO biological processes appear to be much better suited for this purpose than GO molecular function or GO cellular component.
Recently, a step forward has been taken with gene graph enrichment analysis (GGEA, Chapter 11, [68]) which extends the approach to gene sets with accompanying edges. It introduces *consistent* edges, which are edges with direction and effect compatible with the measured values for the genes incident to the edge. In order to evaluate GGEA the notion of *explainability* has been introduced, which is important here. It is clear that GGEA does not employ a holistic network or pathway view but still relies on relevant genes and consistent edges, which are assumed to be independent.
Here, we propose a new Steiner tree approach, which tries to connect the relevant genes within a set of gene contexts in order to derive a sparse and cost optimal subset of nodes and edges explaining parts of the data, i.e. the EO. Such explanations are minimal subsets thereby reducing the effects of incompletely or not meaningfully defined gene sets. The RelExplain approach thereby extends the GGEA method to a holistic network/pathway-based approach which aims a mechanistic explanations for the data compatible with the directions and effects of edges in the identified results. Note that the proposed Steiner tree approach is only one particular way to come up with mechanistic models as explanations, other meaningful options can easily be

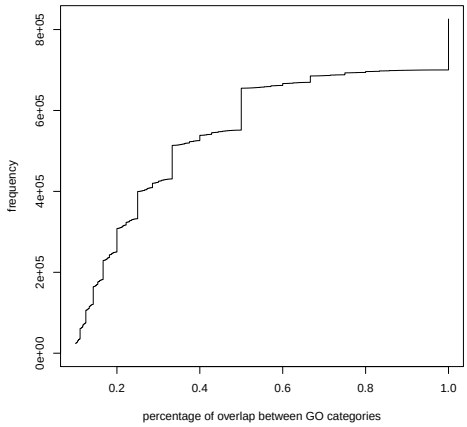


Figure 12.1: Overlap distribution of GO-classes, discussed in section 12.2.2.

imagined.

There is much more knowledge available than list of sets of genes involved in the same biological process (GO class), or the limited collection of well annotated pathways (KEGG). However, the facts and relations described in scientific publications are not easily accessible. Quite sophisticated text mining methods are required to extract biological entities and relations between such entities from free text. Many different methods have been proposed from named entity recognition, co-occurrence extraction, to relation recognition using dictionary-based, machine learning, or natural language processing approaches. We use syngrep (Chapter 8), SynTree (Chapter 9), and RELEX ([61]) for our purposes here. The output of this pipeline is a huge network of 455.280 annotated regulations (524.455 for the 'all' network including database entries). Unfortunately, mining relations is error prone, different benchmarks show that the accuracy of the mined relations is in a range of 50-80% for the best methods ([101]). The size of the network makes it unfeasible to check relations manually and it makes underlying algorithmic problems intractable.

The main goal of the RelExplain method is to integrate these two sources of information (gene set definitions and mined relations) for deriving mechanistic regulation models (so called explanations) for given data and given context knowledge.

Additionally, we provide a web interface where these explanations can be browsed and the source of the mined relations (PubMed sentences) can be validated in the context of the experiment. This way the huge text-mining network is pre-filtered by the experimental data. The interface provides the option to manually exclude invalid/unclear edges and re-calculate the explanations in order to iteratively arrive at validated hypotheses consistent with the context knowledge and the data.

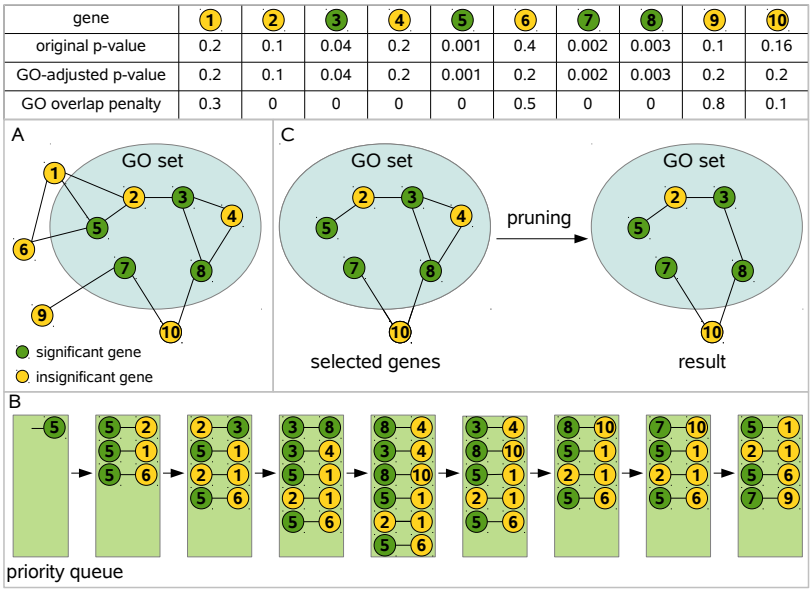


Figure 12.2: Shows the relevant network finding step of the RelExplain method discussed in section 12.2.2.

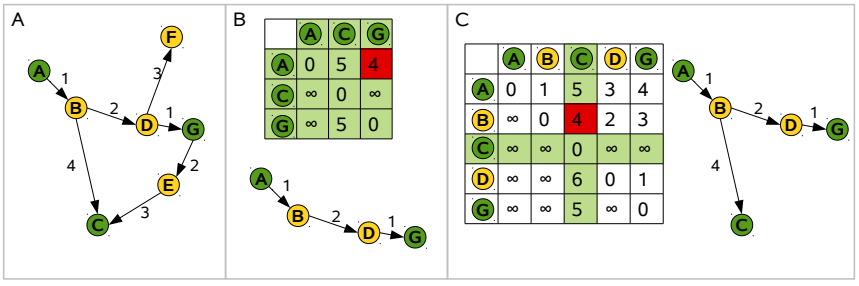


Figure 12.3: Shows Steiner-Tree determination step discussed in section 12.2.2.

12.2 The RelExplain method

RelExplain relies on a predefined sets of biological processes, e.g. as defined via KEGG pathways or GO biological process classes. Despite its obvious shortcomings to serve as biological processes, which can be used as interpretation and explanation for experimental data, we use the GO biological classes in this paper. RelExplain works on given (pre-processed) quantitative differential measurements, so called experimental outcomes (EOs), which consist of foldchange and significance values for genes. Again, despite lots of problems and options to obtain such data from actual raw measurements, we accept the significance values as p-values for the differential expression (fc) of the genes. The goal of RelExplain is to determine plausible explanations for EOs. An explanation is a set of genes together with connecting relations, which can provide hypotheses for the measured changes between conditions. A biological process bp is tested for containing an explanation: RelExplain tries to connect the significant genes in the bp via regulatory paths if possible within the process (i.e. necessary paths should not travel too far outside the bp). Thus, RelExplain is given a biological process, a background gene network and a distinguished set of regulated genes and returns the best-scoring Steiner tree of the significantly regulated genes within the network and the process pb. The score of a graph prefers genes within the biological process bp and penalizes necessary detours outside the process.

12.2.1 RelExplain input

Formally, the input of the RelExplain method is

1. a set of biological processes BP to be used as candidate explanations,
2. a candidate biological process bp,
3. a (background) network of genes $N = (ngs, nge)$ ($bp \subseteq ngs$),
4. data representing an experimental outcome (EO) $D = \{(fc_g, sig_g) \mid g \in ngs)\}$
5. a set of significantly regulated genes $srgs$ ($srgs \subseteq ngs$),
6. and two thresholds, 1) a gene significance t and 2) a maximum allowed distance dt.

The background network N represents the knowledge used to "explain" the data. It is typically derived via database extraction and literature mining. Here we use the relations derived from the literature via the RELEX approach ([61]), and, in addition, a RELEX network extended with edges extracted from databases ('all'). The data is given by the EO, which consists of differential foldchanges fc and associated significance values sig for all genes (of course, missing values are possible and common and has to be dealt with appropriately). The set of biological processes BP and the current candidate process bp are sets of genes, which can be extended to networks via the induced edges of the background network N. bp is the candidate hypothesis to be used to explain the data. Typically, this is a GO class ([12], i.e. a GO biological process), a KEGG pathway ([144]), or the like. As such classes are not necessarily defined in a meaningful way for

our purpose, RelExplain seeks for relevant subnetworks induced by the biological process. The significantly regulated genes can be pre-defined by the user or selected via the gene significance threshold t from the EO data D. The maximum distance threshold dt is a user parameter, which controls the allowed length of paths between regulated genes either in or outside the biological process in question.

12.2.2 RelExplain algorithm

The RelExplain algorithm consists of two phases: The first phase limits the network to a smaller relevant subnetwork. The second phase determines an (approximate) best-scoring Steiner tree in the resulting subnetwork.

RelExplain score penalties based on GO-set overlaps

The RelExplain score is based on the given set of gene (fc, sig) values especially the significance values. In addition, it defines a set of penalty scores based on the underlying set of biological processes. RelExplain computes Steiner trees to connect significant genes within a given pre-defined biological process by using additional nodes from within and outside the process with a minimum overall score. If such a connected Steiner tree is not possible, RelExplain allows detours through the background network. The allowed detours are limited via the maximum distance of any two regulated genes (a user defined threshold dt) and the score penalties for leaving the current process bp. This score penalty is defined in RelExplain via an analysis of the used set BP, here the GO biological process classes. RelExplain defines a penalty pf_g for every gene $g \notin bp$. pf_g has to be larger then the threshold t (for significant genes) and it depends on the GO class of gene and its overlap with bp (of course, the larger the overlap the smaller the penalty).
We precalculate the GO-set overlap for all GO-set pairs, which are not ancestors of each other in the GO hierarchy. Figure 12.1 shows the overlaps, which can be quite large. For every $g \notin bp$ we find its best overlapping annotated GO-class ,i.e. the class containing g having the largest overlap ov_g (defined as the size of its intersection with bp divided by the size of the smaller of the two sets). As the score penalty we use $pf_g = t + (1 - t) * (1 - ov_g)$, such that a perfect overlap yields a score of t and a zero-overlap yields a score of 1.

Finding the relevant network

The Steiner tree problem is known to be NP-complete [66]. The heuristic to find Steiner trees used in RelExplain scales with the network size. Therefore, RelExplain first identifies a (small) relevant subnetwork, if there are more than one significant gene in the bp in question (for zero and one significant gene in bp there is nothing to connect with a Steiner tree, in this case we say *bp cannot explain the data*. Figure 12.2 shows a sketch of this step: we start from the most significantly regulated gene and put its neighbors into a priority queue sorted by the edge score (for edge scores see section 12.2.3). In each subsequent step we take the node from the top of the priority queue, calculate the edge scores and put its neighbors back to the queue, until we either

exceed the maximal distance threshold on each path, or succeed in connecting the significantly regulated genes.
If not all significantly regulated genes are reachable the procedure is repeated starting from the most significantly regulated gene that was not yet connected. In the end the subnetwork that contains most of the significantly regulated genes is selected. If no subnetwork contains at least two significantly regulated genes the procedure terminates and bp is said to not be able to explain the data.

Finding the best-explaining Steiner-Tree

After limiting the network to the relevant area in the first phase, we find the Steiner tree in the remaining subnetwork with a shortest-path based heuristic, see Figure 12.3. We calculate the shortest paths (using the edge scores) between all significantly regulated genes in the relevant network. We initialize the resulting Steiner tree with the empty set and keep adding paths (the nodes and edges of the path) to the Steiner tree until all significant nodes are contained in the final Steiner tree. Iteratively, the best-scoring path is chosen and added to the growing Steiner tree. The shortest path matrix is updated by calculating shortest path scores between the newly added nodes to all significant nodes not yet in the tree (thereby excluding scores for node pairs already in the Steiner tree). This path extension strategy is similar to the well-known approach for approximating arbitrary Steiner trees by metric Steiner trees, where edge scores are replaced by scores of shortest path connecting the endpoints of the edge.

12.2.3 Relation scoring

Currently, we score the relations simply by taking the mean of the gene scores (p-value plus the penalty if needed, i.e. if the gene is not in bp). This means, that the score of the Steiner tree only depends on the genes in the determined tree and the scores (and penalties) associated with the genes. The penalty accounts for the path connecting the Steiner tree leaving the current hypothesis (i.e. biological process) under investigation. A more involved scoring of the added edge, e.g. based on the consistency of the edge (Chapter 11, [68]) could easily be used. Such an edge scoring could be based on the experimental evidence associated with the edge in the RELEX network (direction, effect, type and strength of the relation) and the experimentally determined expression changes in the data for the edge.

12.2.4 Handling the GO-hierarchy

RelExplain in the version described here uses the GO biological processes for the explanation. The GO classification is essentially used in RelExplain in two important ways: RelExplain exploits overall features of the GO biological process classification for finding the best explanations and most relevant biological processes. This is achieved via the score penalty, which is small if a very similar class has to be used to connect the Steiner tree of significant genes and is large for a very dissimilar class.

RelExplain treats the GO hierarchy in a bottom up fashion starting with the leaf nodes of the GO tree. Thus, the set of nodes to be processed is initialized with these leaf nodes. Internal nodes will be added to the set only if a node cannot explain the data, in which case the parents of the node are added to the set of nodes to be processed. This means that a node will never be processed if all its child classes can explain the data. This keeps the procedure efficient as large GO classes are only considered if necessary and in many cases small GO classes are already sufficient to explain the data. This avoids considering large unspecific classes (which likely can explain everything) in the first place and helps in a Occam's razor like way to focus the process to relevant parsimonious explanations.

12.2.5 Application of RelExplain

RelExplain can be used (1) for explaining predefined sets (for example to analyze results of gene set enrichment methods such as ORA, GSEA, or even GGEA), or (2) as a gene set enrichment/ranking method itself. In the first case, RelExplain provides explanations of the already identfied and ranked gene sets and allows to investigate the results in quite some detail by checking edges and the associated evidence in a user-friendly way (see also section 12.4).
In the second case, a ranking of competing biological processes (or gene sets) is computed based on a modified F-measure. Of course, using the modified F-measure score, RelExplain can also be employed as a re-ranking method in the first application case.

12.2.6 Modified F-measure to rank processes

To rank the explanations identified by RelExplain, the following score f is proposed:

$$f = 0.5 * \frac{s}{e + w * g} * \frac{s}{r} \tag{12.1}$$

Here, s is the number of significantly regulated genes that are contained in the explanation, e is the size of the explanation, w is a small weight that controls the influence of the number of insignificant genes within the analyzed GO category g and r is the overall number of significantly regulated genes. This score is high if the explanation contains few insignificant genes but many of the significant genes. Furthermore it favors small GO categories with many significant genes.

12.3 Results

12.3.1 Sample experiments

Here we report applying RelExplain to two different well-studied cancer data sets, one on Neuroblastoma and the other on ALL.
We used the acute lymphoblastic leukemia (ALL) data set of [34] in which samples of 37 patients with a BCR/ABL fusion gene are compared to 42 controls. This data set has already been analyzed several times and it is also the running example of the Bioconductor differential gene

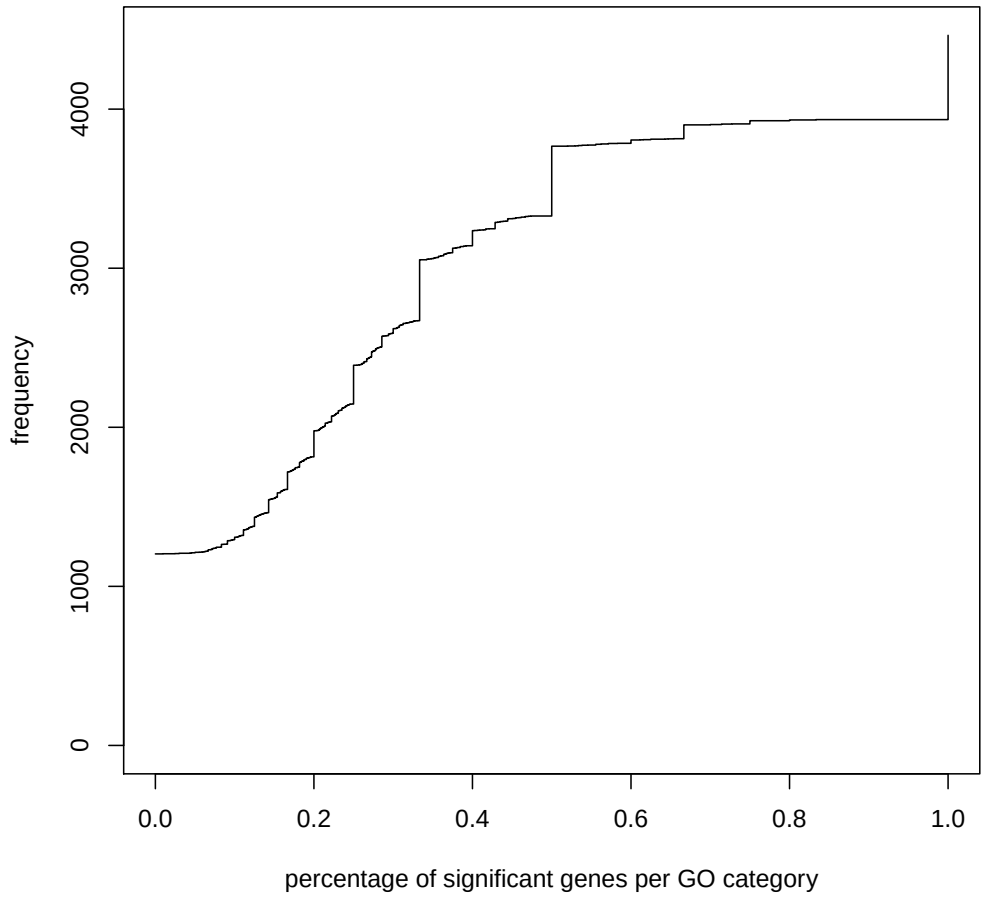


Figure 12.4: Distribution of significantly regulated Genes in GO biological processes

DAVID Bioinformatics Resources 6.7
National Institute of Allergy and Infectious Diseases (NIAID), NIH

Functional Annotation Clustering

Help and Manual

Current Gene List: all_sigf
Current Background: Homo sapiens
156 DAVID IDs

Options Classification Stringency Medium
Rerun using options Create Sublist

51 Cluster(s) Download File

Annotation Cluster 1	**Enrichment Score: 5.04**				**Count**	**P_Value**	**Benjamini**
GOTERM_BP_ALL	regulation of apoptosis	RT			24	2.4E-6	3.4E-3
GOTERM_BP_ALL	induction of apoptosis	RT			15	2.5E-6	1.8E-3
GOTERM_BP_ALL	induction of programmed cell death	RT			15	2.6E-6	1.2E-3
GOTERM_BP_ALL	regulation of programmed cell death	RT			24	2.8E-6	1.0E-3
GOTERM_BP_ALL	regulation of cell death	RT			24	3.0E-6	8.5E-4
GOTERM_BP_ALL	positive regulation of apoptosis	RT			17	3.8E-6	9.0E-4
GOTERM_BP_ALL	positive regulation of programmed cell death	RT			17	4.2E-6	8.5E-4
GOTERM_BP_ALL	positive regulation of cell death	RT			17	4.4E-6	7.8E-4
GOTERM_BP_ALL	positive regulation of biological process	RT			36	3.4E-4	3.2E-2
GOTERM_BP_ALL	positive regulation of cellular process	RT			32	1.3E-3	7.8E-2
Annotation Cluster 2	**Enrichment Score: 3.95**				**Count**	**P_Value**	**Benjamini**
GOTERM_BP_ALL	regulation of apoptosis	RT			24	2.4E-6	3.4E-3
GOTERM_BP_ALL	regulation of programmed cell death	RT			24	2.8E-6	1.0E-3
GOTERM_BP_ALL	regulation of cell death	RT			24	3.0E-6	8.5E-4
GOTERM_BP_ALL	cell death	RT			17	1.5E-3	8.9E-2
GOTERM_BP_ALL	death	RT			17	1.6E-3	8.8E-2
GOTERM_BP_ALL	apoptosis	RT			15	2.0E-3	9.6E-2
GOTERM_BP_ALL	programmed cell death	RT			15	2.3E-3	9.7E-2
Annotation Cluster 3	**Enrichment Score: 2.84**				**Count**	**P_Value**	**Benjamini**
GOTERM_BP_ALL	biological regulation	RT			93	3.5E-4	3.1E-2
GOTERM_BP_ALL	regulation of biological process	RT			86	2.7E-3	1.1E-1
GOTERM_BP_ALL	regulation of cellular process	RT			83	3.2E-3	1.1E-1
Annotation Cluster 4	**Enrichment Score: 2.67**				**Count**	**P_Value**	**Benjamini**
GOTERM_BP_ALL	regulation of oxidoreductase activity	RT			5	5.8E-4	4.4E-2
GOTERM_BP_ALL	regulation of monooxygenase activity	RT			4	1.7E-3	9.0E-2
GOTERM_BP_ALL	regulation of nitric-oxide synthase activity	RT			3	1.0E-2	2.5E-1
Annotation Cluster 5	**Enrichment Score: 2.43**				**Count**	**P_Value**	**Benjamini**
GOTERM_BP_ALL	actin cytoskeleton organization	RT			10	3.4E-4	3.4E-2
GOTERM_BP_ALL	actin filament-based process	RT			10	5.4E-4	4.4E-2
GOTERM_BP_ALL	cytoskeleton organization	RT			12	3.3E-3	1.2E-1
GOTERM_BP_ALL	organelle organization	RT			16	3.0E-1	8.8E-1
Annotation Cluster 6	**Enrichment Score: 2.19**				**Count**	**P_Value**	**Benjamini**
GOTERM_BP_ALL	anatomical structure formation involved in morphogenesis	RT			11	2.2E-3	9.7E-2
GOTERM_BP_ALL	angiogenesis	RT			7	3.1E-3	1.2E-1
GOTERM_BP_ALL	blood vessel morphogenesis	RT			8	4.3E-3	1.5E-1
GOTERM_BP_ALL	anatomical structure morphogenesis	RT			22	5.2E-3	1.6E-1
GOTERM_BP_ALL	blood vessel development	RT			8	9.6E-3	2.4E-1

Figure 12.5: Output of gene set enrichment analysis of the ALL dataset using DAVID [94]. The result clearly hints to two highly enriched clusters (enrichment both around or higher than 4): 'positive regulation of apoptosis' at first rank and 'regulation of apoptosis' at rank 2.

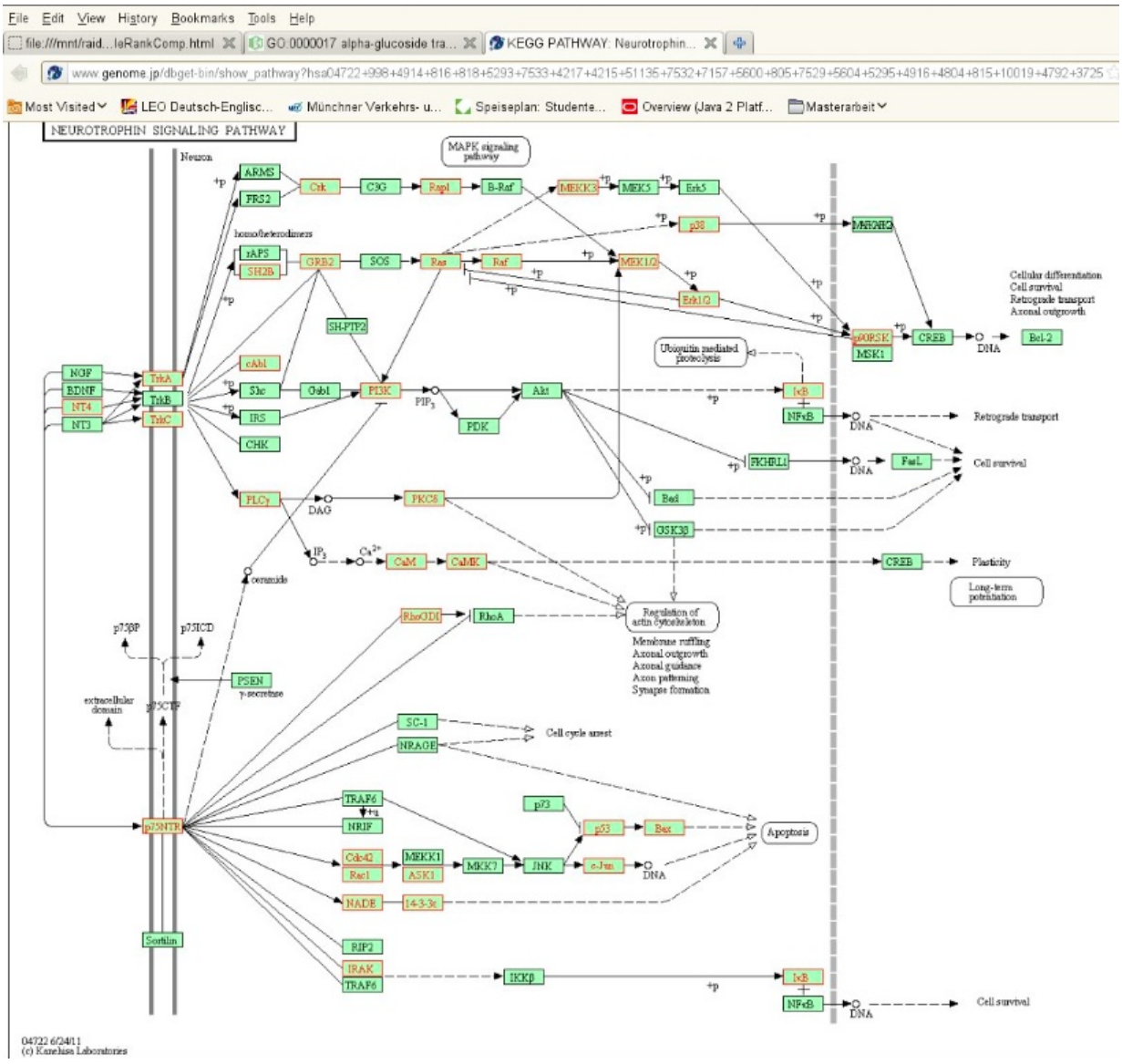


Figure 12.6: The pathway identified by our GGEA method for the Neuroblastoma data. The pathways hints to both cell survival and apoptosis. See 12.2.2.

expression tutorial ([184]). We followed the steps described in the tutorial to calculate multiple testing corrected p-values with the only difference that we did not apply any filters and used the mean of all probes to get gene level intensities for 8.799 genes. This resulted in 156 significantly differentially expressed genes (p-value< 0.05) between ALL and control cells in the experiment. They are distributed over 2.355 GO classes. When analyzing the significant differential genes via a GO over-representation analysis with the software tool DAVID [94], we obtain a signature which hints to apoptosis related processes, which appears to make sense for a cancer cell-line experiment (see Figure 12.5). Obviously, the results appear meaningful at a first glance, but the explanatory power is weak. Apparently, however, about 126 genes are actually down-regulated in the ALL cells and the remaining 30 genes are up-regulated. When using these up- and down-regulated genes (to somewhat naively account for the direction of the regulation) in both cases again apoptosis related processes with slightly changed p-values are obtained (output not shown). Again, one cannot learn much from this analysis apart from apoptosis being relevant (a fact which is apparent already from the phenotype of the cell lines). Therefore, more involved methods are needed taking the direction of the expression change and the significance as well as the directions and type of effect in the underlying network into account. Therefore, we also applied RelExplain to this data.
As the second data set we re-analyzed the neuroblastoma data set we already used in for GGEA (Chapter 11, [68]). In this data, in cancer and control cells, 5.500 human genes have been measured of which 3.110 appear to be significantly regulated. This large fraction of regulated genes alone makes over-representation and gene set enrichment analyzes problematic. Figure 12.4 shows the distribution of significantly regulated genes (t-test p-value $<$0.001) over the GO biological process classes. As expected, there are hundreds of GO classes with more than 50% of their genes being significantly regulated, and thousands of GO classes with a significant fraction of significant genes.

12.3.2 The GO hierarchy

We use a current GO version with about 24.000 GO classes in the GO biological process category, which is organized as a directed acyclic graph with depth 10. We find 11.529 of these classes, which contain at least one human gene. Overall, we find 15.861 genes annotated to these GO classes, with a gene annotated to 4.3 classes on average.
As already mentioned (see Figure 12.1) the GO classes overlap (contain the same genes) to quite some extent. We exploit this fact for assigning gene and class specific penalties when trying to explain the data with a particular process (see Section Methods above). In addition, using this large number of GO classes together with a possibly also large number of regulated genes, we have to expect many enriched and/or significant GO classes, which will be of limited help (see Figure 12.4) in understanding the data at hand or in proposing validation experiments.
Of course, for a data set with a small number of interesting genes (such as the ALL data set) the situation is much more promising, but as mentioned above, actually the results again are not very helpful.
There are several obvious problems: First, methods such as DAVID require to provide a list of genes of interest without further specification. Thus, it is a problem to distinguish the direction

of the changes and thus to identify the relevant GO classes or KEGG pathways. However, for understanding the data it is of course helpful to know, whether apoptosis or anti-apoptosis/cell survival related processes are actually involved in the cancer cell line. Second, GO classes and KEGG pathways often contain several (quite different) explanations. E.g. for the Neuroblastoma data, the GGEA method identified the 'Neurotrophin signalling pathway' from KEGG as containing a highly significant number of regulated genes and consistent edges (see Figure 12.6). As can be seen, the pathway is triggered by two regulated receptors, which both lead via a chain of regulated genes and consistent edges to an outcome. The first one (TrkA and TrkC) leads via the MAPK signaling (p38, ERK, NFkb) to the outcome 'cell survival' and 'cellular differentiation'. The second one (p75NTR) leads via p53 and c-Jun to 'Regulation of apoptosis', a quite contrary result of the involved process.

12.3.3 RelExplain results

As explained above, the interpretation even of well studied data sets is not easy, even if we do not have a ridiculously large number of regulated genes. The problems are that state-of-the-art methods do not exploit the direction of the regulation and the direction and effect of the edges in the network. Even doing that would be an edge-centered approach not accounting for the overall network structure of a GO class or KEGG pathway. Moreover, the sets are somewhat promiscuous with respect to their possible outcomes and explanations. Therefore, a more detailed analysis is necessary, which can be done with RelExplain. The main results are summarized in Table 12.1, which also contains a detailed comparison with other method in order to examine the results of the different GO set enrichment methods on four hypotheses that we derived from the literature. The BCR/ABL fusion gene has tyrosine kinase activity (keyword: "tyrosine kinase") and is known to cause uncontrolled proliferation[37, 118, 126, 153] (keyword: "proliferation"), inhibit apoptosis[45, 150, 157] (keyword: "apoptosis—cell death") and alter adhesion[45, 59, 153] (keyword: "adhesion"). We included those GO categories that match a keyword in their name and either contained at least one significant gene or had a rank of 150 or better in one of the applied enrichment methods. The full analysis can be found on the supplementary web site at http://services.bio.ifi.lmu.de/relexplain/supplement/, Table 12.1 contains the results for the 'inhibition of apoptosis' hypothesis. Quite a number of GO classes deal with the relevant processes and contain at least one significant gene or are ranked high by at least one method. These GO classes are color coded in the table, green: inhibition of apoptosis, white: apoptosis, red: induction of apoptosis.

Several observations can be made: First, the results on these GO classes are very diverse between the various methods (columns of the table). Not only are the p-values computed very different, but also the ranking of the classes varies widely (if rank intervals are given the respective p-values are the same within the interval). No obvious consensus between methods can be observed. Also the number of genes in the set varies a lot: from 6 to 1762, the same is true for the number of significant genes in the respective sets, which varies from 1 to 38. Thus, the results are quite inconclusive. Applying RelExplain to these sets in order to extract explanations from them, which might help the further interpretation, we obtain no explanation in several cases (thereby falsifying the hit) and small hypotheses in other cases. In some cases interesting hypotheses are

derived, e.g. for 'regulation of executive phase of apoptosis', where from a GO class of 182 genes, 6 are significant and 4 of them could be connected by a Steiner tree completely within the class (by using 4 additional nodes, which appear not to be regulated significantly). Another example is 'regulation of anti-apoptosis', where RelExplain finds 3 out of 53 genes regulated, from which an explanation with 1 additional node in the same class and 2 outside class nodes could be derived. These hypotheses are small enough to provide guidance for further validation and experiments. There are a handful of more hypotheses which appear meaningful, but which are way more complex and require a number of outside nodes for completing the respective Steiner trees.

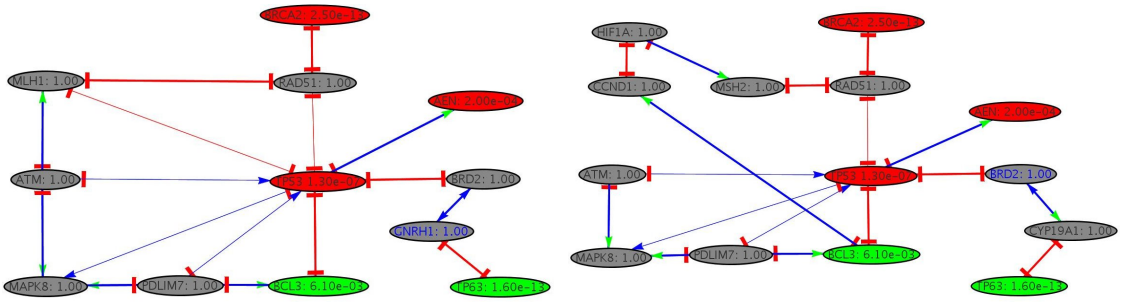
12.4 The RelExplain system

12.4.1 The RelExplain graphical user interface

RelExplain comes with a sophisticated web-based graphical user interface, which allows to apply the different modes of RelExplain (re-analysis, re-scoring and search for explanations) and to inspect the results in various ways. It allows to view network visualizations of explanations, to inspect the associated mapped data, the features of the network entities such as nodes and edges, and to browse the available experimental and literature evidence via directly linking the entities and the explanations to the mined literature. The features of the GUI will be described elsewhere (see the documentation on the server web site http://services.bio.ifi.lmu.de/relexplain). They include: a web-browseable textmining network several hundred-thousand nodes and relations, up-/down-load of EOs, searching and filtering by GO and/or entity, definition and upload of user defined genesets, upload and visualization of rankings from other methods, running RelExplain algorithm interactively, inspect and manual editing of (non-relevant) edges from explanations and re-run of RelExplain iteratively.

12.4.2 Worked out example explanations

To illustrate the interactive working with the RelExplain system, Figure 12.7 shows an example RelExplain output for the GO class "GO:0072332". ("p53 signal transduction by p53 class mediator resulting in induction of apoptosis") for the Neuroblastoma data.
The figure shows two network explanations and the corresponding literature evidence (bottom table) for the edges for validation purposes. The two Steiner trees illustrate a RelExplain explanation (top), which is inspected by the user based on the evidence for the edges (top half of the table). As some of the used evidences and thus edges appear to be weak (marked yellow in the top part of the table), these edges are excluded from a valid explanations. The second Steiner tree shows the result of re-running RelExplain on the edited network, this time providing a modified explanation, which can completely be validated as judged from supporting literature (lower half of the bottom table). The networks illustrate the Steiner trees constructed by RelExplain (thick edges) and additional relations connecting the nodes in the tree (thin edges) supporting the proposed hypothesis. The nodes in the network are colored green and red corresponding



(a) initial result network

(b) recalculated result network

edge	pubmed	sentence	confirmed
RAD51<=> MLH1	14704162	**MLH1**, in the *suppression* of homeologous recombination were similar in **rad51** strains	unclear
MLH1 <=> ATM	12447371	**MLH1** *associates* with **ATM**	yes
BRCA2 <=> RAD51	17515904	**BRCA2** *interacts* with **RAD51**	yes
PDLIM7 <=>MAPK8	16446357	**LMP1** *facilitates* the assembly of this *complex* and *enhances activation* of **JNK**	yes
BCL3 <=> PDLIM7	17881446	*induction* of p50/p50/**Bcl-3** complexes by **LMP1**	yes
TP63 <=> GNRH1	15255840	**AIs** must be combined with a gonadotropin *releasing* hormone (**GnRH**) agonis	unclear
BRD2 <=> TP53	17437340	*Suppressive* effect of **FSH** and LH on **p53** expression	yes
BCL3 <=> TP53	16384933	**Bcl-3** proto-oncogene *suppresses* **p53** activation	yes
AEN <=> TP53	18264133	(**AEN**) is a novel direct *target gene* of **p53**	yes
ATM <=> MAPK8	17084707	**ATM**, which *inhibits* **JNK**	yes
BRD2 <=> GNRH1	3109880	**FSH** may *interact* directly with **GnRH**	yes

TP63<=>CYP19A1	18695261	**AIs** *bind* noncovalently and reversibly to the **aromatase** protein	yes
CYP19A1 <=> BRD2	6423903	*Augmentation* of **aromatase** activity by **FSH**	yes
MSH2 <=> RAD51	12687013	**hMSH2** and hMSH6 *coimmunoprecipitated* with BLM, p53, and **RAD51**.	yes
MSH2 <=> HIF1A	15780936	**HIF-1alpha** is responsible for genetic instability at the nucleotide level by *inhibiting* **MSH2**	yes
CCND1 <=> HIF1A	20179204	*Suppression* of **cyclin D1** by **hypoxia-inducible factor-1**	yes
BCL3 <=> CCND1	11713278	**Bcl-3** *induces* **cyclin D1**	yes

(c) checked edges

Figure 12.7: Steiner-Tree explaining the GO-category GO:0072332 ('p53 signal transduction by p53 class mediator resulting in induction of apoptosis') for the Neuroblastoma data. Top: network explanation before checking the evidence of the explaining edges, middle: explanation after excluding unclear edges and rerun of RelExplain on the edited network, bottom: table containing the supporting publications for the edges (upper part for the first, lower part for the second network). Nodes are colored green/red depending on up/down regulation and the associated gene p-values are indicated, gray nodes are not significant and also show p-values (with penalties if not belonging to the current biological process). Arrows (activation) and bars (inhibition) indicate direction and effect of the edge, which are colored green/red if the annotation is the same (activate/inhibit) in both directions, and blue otherwise.

to up and down regulation of the genes as given in the data. Grey nodes are not significant and show the associated gene p-value (plus additional penalty if not part of the current biological process). The direction of the edges is indicated as well as the associated effect (activating=green, inhibiting=red) as given in the background network.
Note that in the version described here, RelExplain only uses a simple scoring of nodes, a simple consistency notion for edges, and a quite straightforward approach to network properties (Steiner tree of significant genes). Nevertheless, with RelExplain it is possible to derive understandable explanations, which are consistent with the data and the underlying network relations using a small number of interactive steps. We claim that this is pragmatically useful (either for identifying hypotheses or for falsifying them) and that it is likely the best you can achieve given the situation on the necessary prerequisites for a truly integrative data analysis.

12.5 Discussion and conclusion

State-of-the-art gene set over-representation and enrichment approaches have severe shortcomings and even major flaws, which can lead to over- and mis-interpretation of large scale high-throughput data sets. Nevertheless, these approaches are commonly applied on ever more of these ever faster generated data. We presented RelExplain, which aims at providing understandable explanations derived from context knowledge and the data. It provides means to validate and check the explanations by making the available facts available and by linking them with the actual data. In order to derive explanations, RelExplain takes a more holistic approach to biological networks and pathways as a means for data analysis by computing Steiner trees connecting the regulated genes within a particular functional context. Of course this does not come for free, several prerequisites are needed: First, detailed networks need to be available in order to hope for meaningful explanations. Second, the direction and effect of edges in these networks have to be considered to check edge consistency against the data. Third, as a pathway is more than its gene set or its set of edges, but a connected structure of entities working together, network properties have to be taken into account, which involve hard algorithmic problems.
We have re-analyzed well-studied data sets, which are plagued with various problems prohibiting explanations: So called p-values are often used and often mis-interpreted. A large fraction of genes might appear to be 'significantly' regulated. Reasonable contexts such as functional gene sets, biological processes, and pathways are not available or designed for different purposes. Thus, methods are overly optimistic in deriving relevant gene sets as is indicated by highly significant p-values. Using several methods with slightly different assumptions and different ways to calculate the p-values used for ranking the gene sets, we obtain grossly different results. RelExplain allows to check such results and to validate them against available knowledge. Thus, reasonable hypothesis for further experiments can be derived. Still, quite some progress in algorithmically exploiting network properties, in defining appropriate scores (not to speak of p-values), and in making available detailed mechanistic functional contexts is needed in order to routinely derive explanations from data.

GO term	RELEXplain	ORA	GGEA	GSEA	SAMGS	DAVID	size	#sig
negative regulation of cell death	8/6/5 494-495 (415)	1159 (387)	NA	NA	1034-1087 (10)	NA	666	10
positive regulation of anti-apoptosis	NA	396-400 (133)	376-5201 (4)	971 (334)	1-205 (1)	NA	42	2
negative regulation of programmed cell death	8/6/5 483-484 (407)	1071 (353)	NA	NA	1034-1087 (10)	NA	646	10
cell death	22/17/3 256 (221)	NA	NA	NA	NA	25 (23)	1762	38
regulation of programmed cell death	18/13/2 187 (162)	NA	NA	NA	NA	6 (6)	1263	30
programmed cell death	22/17/3 233 (202)	NA	NA	NA	NA	26 (24)	1607	36
execution phase of apoptosis	4/3/1 215 (186)	729-735 (233)	NA	252-253 (15)	377-512 (3)	NA	240	6
cellular component disassembly involved in execution phase of apoptosis	NA	1656-1659 (625)	376-5201 (4)	1717-1720 (699)	1526-1549 (22)	NA	76	1
regulation of cell death	19/14/2 175 (154)	NA	NA	NA	NA	4 (4)	1294	31
regulation of execution phase of apoptosis	4/3/1 145-146 (132)	204-209 (79)	NA	241-251 (14)	377-512 (3)	NA	182	6
regulation of anti-apoptosis	3/1/2 41 (37)	125-127 (45)	376-5201 (4)	656-659 (179)	1-205 (1)	NA	53	3
developmental programmed cell death	NA	886-887 (293)	376-5201 (4)	NA	1701-1722 (28)	NA	27	1
anti-apoptosis	7/4/3 124 (116)	132-134 (48)	NA	977 (338)	206-376 (2)	NA	286	8
positive regulation of programmed cell death	11/6/4 121 (113)	7-11 (2)	NA	1-185 (1)	206-376 (2)	3 (3)	543	21
induction of programmed cell death	7/3/3 77 (70)	1-6 (1)	NA	233-238 (12)	206-376 (2)	2 (2)	304	15
positive regulation of cell death	12/7/4 111 (103)	7-11 (2)	NA	1-185 (1)	206-376 (2)	1 (1)	556	22
induction of apoptosis	7/3/3 75 (68)	1-6 (1)	NA	287-291 (33)	206-376 (2)	11 (10)	301	15
induction of apoptosis by extracellular signals	2/0/2 202-203 (175)	29-30 (8)	NA	1028 (367)	206-376 (2)	28 (26)	112	6
negative regulation of anti-apoptosis	NA	201-203 (78)	376-5201 (4)	1384-1385 (582)	1679-1700 (27)	NA	6	1

Table 12.1: Summary of RelExplain results and comparison with other methods.

Chapter 13

Conclusion and Outlook

"Fast alles ist leichter begonnen, als beendet.
Almost everything is easier to start than to finish."
Johann Wolfgang von Goethe

With the ever increasing spectrum of biological methods revealing complex molecular mechanisms in the cell and the increasing number of data derived from these techniques it becomes more and more important to integrate the available knowledge for any scientific question: to create a context. In this thesis we introduced several novel context based methods for a variety of current, real-world bioinformatics problems. The investigated bioinformatics questions broadly are organized into two research areas: (i) protein structure similarities and (ii) interpretation of data from high-throughput experiments.
In the first part of the thesis we introduced central concepts within the context of protein structure similarities: (i) the concept of a contact environment potential and (ii) the plasticity of protein structures and its usage as a measure for protein structure comparison and alignment as well as for the tolerance of proteins to structural changes implied by alternative splicing.
The idea of contact environment potential is first tested with the Vorolign method. We showed that, given proper residue based contact information, protein structure similarity can be recognized by comparing the sequence and secondary structure similarities of 3D-contact neighborhoods. In the Vorescore chapter we analyzed the possible usage of this potential as a prototypic rescoring method for the protein structure prediction (PSP) problem. Finally, in the CC-Vorolign chapter we further analyzed the properties of this potential especially for the direct optimization for the PSP problem. Besides finding evidence for the need to handle some plasticity for the residue contact network conservation, the method described here also provides a general approach to directly optimize complex residue pair-potentials.
In the PPM chapter 6 we introduced a new approach to recognize protein structure similarities via the number of required 3D mutations to "morph" one structure to the other. The principle of this method allowed us to analyze the effects of alternative splicing of protein structures in chapter 7. In this analysis we found interesting cases of alternative splicing "isomorphs" likely changing their fold via alternative splicing events. In collaboration with experimental groups we are currently working on the experimental validation of our predictions.

Part II addresses one of the most recent challenges of bioinformatics: to provide analysis tools for the interpretation of outcomes of high-throughput experiments, especially next generation sequencing experiments. In general, the analysis of high-throughput data consists of two major steps: (i) transform the raw measurements into some experimental outcome representing the changes of the underlying biological entities between the experimental conditions and (ii) analysis of the experimental outcomes in order to get more insight about the underlying mechanisms leading to the measured outcome.
Chapter 10 deals with the first step for NGS experiments: to find the correct genomic positions (the mapping) of the sequenced short RNA reads. This step is crucial for the subsequent steps as the mapping directly influences the estimated level of gene or transcript abundance. The idea of our ContextMap method - to integrate the context of expressed transcripts already in the such as meta-transcriptomics analysis.
Chapter 8 and 9 provide methods and tools to generate the context of current knowledge described in publicly available scientific texts for the interpretation step. We introduced the fast dictionary-based named entity recognition (NER) tool *syngrep* acting as a powerful prefilter for the entity relation extraction method *RelEx* thereby establishing a solid basis for effective human annotation with the novel annotation tool *RelAnn*. While the number of associations extracted from all three tools above decreases rapidly the confidence in the extracted relations in turn increases.
In chapter 9 we introduce a novel strategy to increase the quality of automatic annotations by providing human insights encoded in entity ontologies and in rules acting on these ontologies. We show that with this strategy one can achieve up to 90% sensitivity and specificity for named entity recognition if applied to the difficult area of psychiatric disease entity recognition.
In chapter 11 we address the interpretation step using the context of knowledge encoded in a network and introducing the idea of consistency check between knowledge represented by the edges of regulatory networks and the measured data. This idea is then extended in the RelExplain method in chapter 12 for the contexts provided by the tools and methods of chapter 8 for the graph level, enabling an iterative and interactive way of data interpretation and analysis. The main advantages of RelExplain are: (i) it provides possibilities to integrate knowledge for different levels of biological regulation, (ii) it allows for checking of context-dependent relevance of current knowledge by focusing on biological processes relevant for the measurements via a combined holistic analysis prior regulatory information and the experimental outcomes.

Directions for future research

As stated in the above quote from J.W. Goethe, most of the approaches described in this thesis imply several options and possibilities for further analysis, improvement and extensions.
The results for the methods on contact-neighborhood potential together with the current development of contact prediction (see [100, 123, 124]) provide a promising starting point for both the analysis of the principles of protein folding. Using *deep multiple alignments* evolutionary conservation can be investigated of these neighborhood, as well as to extend the amount of predictable protein structures by combining contact prediction with template identification with contact-neighborhood potential. To create high quality deep multiple alignments containing fold-

level similarities the extension of the PPM method for multiple block-based structural alignment is promising.
For the analysis of the effects of alternative splicing on the structures the challenge shifts to support the experimental side with the context knowledge: the main issue is to find biological samples with splicing isoforms of interest and high abundance. In order to achieve this we currently develop methods to transfer isoform annotations between organisms and predict candidates for targeted experiments by integrating RNA-seq and mass spectrometry data for available samples.
For the interpretation of high-throughput experiments we see two major future directions: (i) a "software direction" consisting of the complete integration of raw data analysis into an interpretation pipeline to allow the redefinition of experimental outcome for different scientific questions, the integration of iterative knowledge acquisition by providing a graphical interface for the SynTree rule definition and iteratively integrate human annotated knowledge for the interpretation and (ii) methods to integrate new types of measurements - e.g. for example digital genomic footprinting (DGF), measurement of methylation and histone modification states, ribosomal profiling, binding site measurements (for both proteins, transcription factors and non-coding RNAs such as miRNAs and lncRNAs) [79, 86, 140] into the RelExplain algorithm. For the latter, we already investigated and evaluated some extensions in Evi Berchtold's master thesis [16].
The original goal of the thesis was to develop context based methods addressing complex problems according to Polya's recipe of problem solving. The first part of the thesis represents such an instance: using the insights obtained from the analyzes in chapter 2 we could first devise a plan to use and evaluate the concept of contact neighborhoods in chapter 3, then, by looking back re-formulate the problem for the protein structure prediction context and again devise a plan (evaluate the resulting method) in chapter 4, leading to the next iteration of problem solving in chapter 5. In each phase we also derived new contexts and shifted the complexity of the problem by the definition of this context from relatively simple (in our case solvable with dynamic programming) to complex - needing an appropriate heuristic to solve it practically.
The methods presented in the second part show the other side of the general principle "solve it with appropriate heuristics": the text-mining methods presented in chapter 8 and 9 do not aim to find the "optimum" for the objective of understanding the semantics ("real meaning") of the written text, but rather have the objective to extract usable knowledge pieces as parts of a context, which allow for a manual check if needed. Similarly, it is clear that the method introduced in 12 can not infer the "biological truth" from the data - the current understanding, the current experimental techniques, and the currently available measurement data are still insufficient for reaching this goal.
Therefore, it is simply neither necessary nor helpful to find the optimal solution of a derived formal optimization problem (in this case the finding of best-scoring Steiner-tree) - a heuristic suboptimal solutions will probably lead to the same insights. Similarly to the case described in the ccVorolign chapter (5) the algorithmic complexity comes from the incorporation of more context information, but in contrast to the ccVorolign case, for RelExplain the choice of the heuristic is already part of the step "devising a plan": here the background context estimates the uncertainty of the information the algorithm operates on.
We think, this characterizes the concept of context based bioinformatics: try to incorporate as

much knowledge as possible in the model - even if doing so makes the model and its solutions complex - as than the model stays close to the original problem. Then use this knowledge - the context - again to develop an appropriate heuristic, not necessarily solving the model optimal, but still serving a practical solution over the problem itself.

Acknowledgements

This thesis reflects only a part of the ideas and projects I had the opportunity to worked on in the last years, and is a result of uncounted hours of exciting discussions and debates with several people. Here I would like to thank to all of them for their valuable contributions.
First of all, I would like to express my deep gratitude to Professor Ralf Zimmer for his valuable and constructive suggestions during the planning and development of this thesis, for sharing his ideas with me, and for his open mind for all of my - sometimes premature - ideas.
I also wish to express my very great appreciation to Dr. Fabian Birzele, my long-time co-warrior against the secrets of protein structures. I also thank all members of Professor Zimmer's lab for creating an inspiring environment for scientific work.
I thank also Prof. Dmitrij Frishman for his willingness to serve as second reviewer on my PhD committee. I would also like to thank to all my co-authors and express special thanks for long and thought-provoking in-depth discussions to Dr. Robert Küffner, Florian Erhard, Tobias Petri, Evi Berchtold and Lukas Windhager. I thank Franziska Schneider and Frank Steiner to provide the valuable social and technical background in our group.
Finally I would like to express my great love and thank to my wife Orsolya and my three children Simon, Jázmin and Kornél for - among so many things - training me to improve my multitasking skills and helping me to see things from different perspectives.

Bibliography

[1] BioCarta - Charting Pathways of Life. URL `http://www.biocarta.com/genes/index.asp`.

[2] WHO. International statistical classification of diseases and related health problems, 10th revision. 2007. URL `http://http://apps.who.int/classifications/apps/icd/icd10online/`.

[3] *Domain Adaptation of Rule-based Annotators for Named-Entity Recognition Tasks*, 2010.

[4] Alfred V. Aho and Margaret J. Corasick. Efficient string matching: An aid to bibliographic search. *Communications of the ACM*, 18(6):333–340, June 1975.

[5] N. N. Alexandrov, R. Nussinov, and R. M. Zimmer. Fast protein fold recognition via sequence to structure alignment and contact capacity potentials. *Pac Symp Biocomput*, pages 53–72, 1996.

[6] S F Altschul, T L Madden, A A Schaffer, J Zhang, Z Zhang, W Miller, and D J Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*, 25(17):3389–3402, Sep 1997.

[7] A. Andreeva, D. Howorth, J.M. Chandonia, S.E. Brenner, T.J. Hubbard, C. Chothia, and A.G. Murzin. Data growth and its impact on the SCOP database: new developments. *Nucleic Acids Res.*, 36:D419–425, Jan 2008.

[8] C. B. Anfinsen. Principles that govern the folding of protein chains. *Science*, 181(4096): 223–230, Jul 1973.

[9] C. B. Anfinsen, E. Haber, M. Sela, and F. H. White. The kinetics of formation of native ribonuclease during oxidation of the reduced polypeptide chain. *Proc. Natl. Acad. Sci. U.S.A.*, 47:1309–1314, Sep 1961.

[10] APA. *Diagnostic and Statistical Manual of Mental Disorders, Fourth Edition-Text Revision (DSMIV-TR)*. American Psychiatric Association (APA), 4th edition, 2000.

[11] Alan R. Aronson and Franois-Michel Lang. An overview of metamap: historical perspective and recent advances. *J Am Med Inform Assoc*, 17(3):229–236, 2010.

[12] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat. Genet.*, 25(1):25–29, May 2000.

[13] E Azarya-Sprinzak, D Naor, H J Wolfson, and R Nussinov. Interchanges of spatially neighbouring residues in structurally conserved environments. *Protein Eng*, 10(10):1109–1122, Oct 1997.

[14] C. Bradford Barber, David P. Dobkin, and Hannu T. Huhdanpaa. The Quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483, December 1996.

[15] W. T. Barry, A. B. Nobel, and F. A. Wright. Significance analysis of functional categories in gene expression studies: a structured permutation approach. *Bioinformatics*, 21(9): 1943–1949, May 2005.

[16] Evi Berchtold. RELEXplain - explaining biological processes in the context of networks. Master thesis, LFE Bioinformatik, LMU München, September 2012.

[17] H. Berman, K. Henrick, and H. Nakamura. Announcing the worldwide Protein Data Bank. *Nat. Struct. Biol.*, 10(12):980, Dec 2003.

[18] H. Berman, K. Henrick, H. Nakamura, and J. L. Markley. The worldwide Protein Data Bank (wwPDB): ensuring a single, uniform archive of PDB data. *Nucleic Acids Res.*, 35 (Database issue):D301–303, Jan 2007.

[19] HM Berman, J Westbrook, Z Feng, G Gilliland, TN Bhat, H Weissig, IN Shindyalov, and PE Bourne. The protein data bank. *Nucleic Acids Research*, 28:235–242, 2000.

[20] F. C. Bernstein, T. F. Koetzle, G. J. Williams, E. F. Meyer, M. D. Brice, J. R. Rodgers, O. Kennard, T. Shimanouchi, and M. Tasumi. The Protein Data Bank: a computer-based archival file for macromolecular structures. *J. Mol. Biol.*, 112(3):535–542, May 1977.

[21] F. Birzele, G. Csaba, and R. Zimmer. Alternative splicing and protein structure evolution. *Nucleic Acids Res.*, 36:550–558, Feb 2008.

[22] F. Birzele, J. E. Gewehr, G. Csaba, and R. Zimmer. Vorolign–fast structural alignment using voronoi contacts. *Bioinformatics*, 23(2):e205–11, 2007.

[23] F. Birzele, J. E. Gewehr, and R. Zimmer. Autopsi: a database for automatic structural classification of protein sequences and structures. *Nucleic Acids Res*, 36(Database issue): D398–401, 2008.

[24] J D Blake and F E Cohen. Pairwise sequence alignment below the twilight zone. *J Mol Biol*, 307(2):721–735, Mar 2001.

[25] T. Bonfert, G. Csaba, R. Zimmer, and C. C. Friedel. A context-based approach to identify the most likely mapping for RNA-seq experiments. *BMC Bioinformatics*, 13 Suppl 6:S9, 2012.

[26] M. G. Borrello, G. Pelicci, E. Arighi, L. De Filippis, A. Greco, I. Bongarzone, M. Rizzetti, P. G. Pelicci, and M. A. Pierotti. The oncogenic versions of the Ret and Trk tyrosine kinases bind Shc and Grb2 adaptor proteins. *Oncogene*, 9(6):1661–1668, Jun 1994.

[27] R. Breitling, A. Amtmann, and P. Herzyk. Iterative Group Analysis (iGA): a simple tool to enhance sensitivity and facilitate interpretation of microarray experiments. *BMC Bioinformatics*, 5:34, Mar 2004.

[28] S E Brenner, P Koehl, and M Levitt. The ASTRAL compendium for protein structure and sequence analysis. *Nucleic Acids Res*, 28(1):254–256, Jan 2000.

[29] Margit Burmeister, Melvin G. McInnis, and Sebastian Zllner. Psychiatric genetics: progress amid controversy. *Nat Rev Genet*, 9(7):527–540, Jul 2008.

[30] A Califano, I Rigoutsos, and H J Wolson, editors. *An $O(n^2)$ algorithm for 3D substructure matching of proteins.* Plenum Publishing, 1994.

[31] C. Chaouiya. Petri net modelling of biological networks. *Brief. Bioinformatics*, 8(4): 210–219, Jul 2007.

[32] K. Chen and L. Kurgan. PFRES: protein fold classification by using evolutionary information and predicted secondary structure. *Bioinformatics*, 23:2843–2850, Nov 2007.

[33] Leslie Y. Chen et al. RNASEQR–a streamlined and accurate RNA-seq sequence analysis program. *Nucleic Acids Res*, Dec 2011.

[34] Sabina Chiaretti, Xiaochun Li, Robert Gentleman, Antonella Vitale, Kathy S Wang, Franco Mandelli, Robin Fo, and Jerome Ritz. Gene expression profiles of B-lineage adult acute lymphocytic leukemia reveal genetic patterns that identify lineage derivation and distinct mechanisms of transformation. *Clin Cancer Res*, 11(20):7209–7219, Oct 2005.

[35] Nicole Cloonan, Qinying Xu, Geoffrey J. Faulkner, Darrin F. Taylor, Dave T P. Tang, Gabriel Kolle, and Sean M. Grimmond. RNA-MATE: a recursive mapping strategy for high-throughput RNA-sequencing data. *Bioinformatics*, 25(19):2615–2616, Oct 2009.

[36] Thomas Cormen, Charles Leiserson, Ronald Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 3rd, 2009 edition, 1990. 1292 pp.

[37] D. Cortez, G. Stoica, J. H. Pierce, and A. M. Pendergast. The BCR-ABL tyrosine kinase inhibits apoptosis by activating a Ras-dependent signaling pathway. *Oncogene*, 13(12): 2589–2594, Dec 1996.

[38] G. Csaba, F. Birzele, and R. Zimmer. Protein structure alignment considering phenotypic plasticity. *Bioinformatics*, 24(16):i98–104, 2008.

[39] G. Csaba, F. Birzele, and R. Zimmer. Systematic comparison of SCOP and CATH: a new gold standard for protein structure analysis. *BMC Struct. Biol.*, 9:23, 2009.

[40] G. Csaba and R. Zimmer. Vorescore–fold recognition improved by rescoring of protein structure models. *Bioinformatics*, 26(18):i474–481, Sep 2010.

[41] Levinthal Cyrus. How to fold graciously. *Mossbauer Spectroscopy in Biological Systems: Proceedings of a meeting held at Allerton House, Monticello, Illinois. University of Illinois Press*, pages 22–24, 1969.

[42] F. Darios and B. Davletov. Omega-3 and omega-6 fatty acids stimulate cell membrane expansion by acting on syntaxin 3. *Nature*, 440(7085):813–817, Apr 2006.

[43] R. Day, D.A. Beck, R.S. Armen, and V. Daggett. A consensus view of fold space: combining SCOP, CATH, and the Dali Domain Dictionary. *Protein Sci.*, 12:2150–2160, Oct 2003.

[44] M O Dayhoff, R Schwartz, and B C Orcutt. A model of evolutionary change in proteins. *Atlas of Protein Sequence and Structure*, 5:345–352, 1978.

[45] M. W. Deininger, S. A. Vieira, Y. Parada, L. Banerji, E. W. Lam, G. Peters, F. X. Mahon, T. Khler, J. M. Goldman, and J. V. Melo. Direct relation between BCR-ABL tyrosine kinase activity and cyclin D2 expression in lymphoblasts. *Cancer Res*, 61(21):8005–8013, Nov 2001.

[46] Thomas Loew Dieter Ebert. *Psychiatrie systematisch*. UNI-MED-Verlag, 2005.

[47] I. Dinu, J. D. Potter, T. Mueller, Q. Liu, A. J. Adewale, G. S. Jhangri, G. Einecke, K. S. Famulski, P. Halloran, and Y. Yasui. Gene-set analysis and reduction. *Brief. Bioinformatics*, 10(1):24–34, Jan 2009.

[48] Aleksandra B Djurisic, Jovan M Elazar, and A D Rakic. Genetic algorithms for continuous optimization problems - a concept of parameter-space size adjustment. *Journal of Physics A: Mathematical and General*, 30(22):7849–7861, 1997.

[49] Z Dosztanyi and A E Torda. Amino acid similarity matrices based on force fields. *Bioinformatics*, 17(8):686–699, Aug 2001.

[50] S. R. Eddy. Hidden markov models. *Curr Opin Struct Biol*, 6(3):361–5, 1996.

[51] S. R. Eddy. Profile hidden markov models. *Bioinformatics*, 14(9):755–63, 1998.

[52] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. U.S.A.*, 95(25):14863–14868, Dec 1998.

[53] Ramon Erhardt, Reinhard Schneider, and Christian Blaschke. Status of text-mining techniques applied to biomedical text. *Drug Discov Today*, 11(7-8):315–325, Apr 2006.

[54] N. Eswar, D. Eramian, B. Webb, M. Y. Shen, and A. Sali. Protein structure modeling with modeller. *Methods Mol Biol*, 426:145–59, 2008.

[55] M. E. Evangelopoulos, J. Weis, and A. Kruttgen. Neurotrophin effects on neuroblastoma cells: correlation with trk and p75NTR expression and influence of Trk receptor bodies. *J. Neurooncol.*, 66(1-2):101–110, Jan 2004.

[56] J. J. Faith, M. E. Driscoll, V. A. Fusaro, E. J. Cosgrove, B. Hayete, F. S. Juhn, S. J. Schneider, and T. S. Gardner. Many Microbe Microarrays Database: uniformly normalized Affymetrix compendia with structured experimental metadata. *Nucleic Acids Res.*, 36(Database issue):D866–870, Jan 2008.

[57] P. Ferragina and G. Manzini. Opportunistic data structures with applications. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, page 390. IEEE Computer Society, 2000. ISBN 0-7695-0850-2. ACM ID: 796543.

[58] I. Friedberg and A. Godzik. Fragnostic: walking through protein structure space. *Nucleic Acids Res.*, 33:W249–251, Jul 2005.

[59] Keiji Funayama, Yumi Saito-Kurimoto, Yasuhiro Ebihara, Miyuki Shimane, Hitoshi Nomura, Ko ichiro Tsuji, and Shigetaka Asano. Adhesion-mediated self-renewal abilities of Ph+ blastoma cells. *Biochem Biophys Res Commun*, 396(2):193–198, May 2010.

[60] K. Fundel, D. Guttler, R. Zimmer, and J. Apostolakis. A simple approach for protein name identification: prospects and limits. *BMC Bioinformatics*, 6 Suppl 1:S15, 2005.

[61] K. Fundel, R. Küffner, and R. Zimmer. Relex–relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–71, 2007.

[62] K. Fundel and R. Zimmer. Gene and protein nomenclature in public databases. *BMC Bioinformatics*, 7:372, 2006.

[63] Katrin Fundel. *Text Mining and Gene Expression Analysis - Towards Combined Interpretation of High Throughput Data*. Phd thesis, LMU München, September 2007.

[64] S. Gama-Castro, V. Jimenez-Jacinto, M. Peralta-Gil, A. Santos-Zavaleta, M. I. Penaloza-Spinola, B. Contreras-Moreira, J. Segura-Salazar, L. Muniz-Rascado, I. Martinez-Flores, H. Salgado, C. Bonavides-Martinez, C. Abreu-Goodger, C. Rodriguez-Penagos, J. Miranda-Rios, E. Morett, E. Merino, A. M. Huerta, L. Trevino-Quintanilla, and J. Collado-Vides. RegulonDB (version 6.0): gene regulation model of Escherichia coli K-12 beyond transcription, active (experimental) annotated promoters and Textpresso navigation. *Nucleic Acids Res.*, 36(Database issue):D120–124, Jan 2008.

[65] H H Gan, A Tropsha, and T Schlick. Lattice protein folding with two and four-body statistical potentials. *Proteins*, 43(2):161–174, May 2001.

[66] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman, 1979.

[67] D. M. Gatti, W. T. Barry, A. B. Nobel, I. Rusyn, and F. A. Wright. Heading down the wrong pathway: on the influence of correlation within gene sets. *BMC Genomics*, 11:574, 2010.

[68] L Geistlinger, G. Csaba, R Küffner, N. Mulder, and R Zimmer. From sets to graphs: Towards a realistic enrichment analysis of transcriptomic systems. *Bioinformatics*, 27 (ISMB 2011, Wien):i366–i373, 2011.

[69] Hartmann J. Genrich, Robert Küffner, and Klaus Voss. Executable petri net models for the analysis of metabolic pathways. *STTT*, 3(4):394–404, 2001.

[70] J.E. Gewehr, V. Hintermair, and R. Zimmer. AutoSCOP: automated prediction of SCOP classifications using unique pattern-class mappings. *Bioinformatics*, 23:1203–1210, May 2007.

[71] C. Gieger, L. Geistlinger, E. Altmaier, M. Hrabe de Angelis, F. Kronenberg, T. Meitinger, H. W. Mewes, H. E. Wichmann, K. M. Weinberger, J. Adamski, T. Illig, and K. Suhre. Genetics meets metabolomics: a genome-wide association study of metabolite profiles in human serum. *PLoS Genet.*, 4(11):e1000282, Nov 2008.

[72] J. J. Goeman and P. Buhlmann. Analyzing gene expression data in terms of gene sets: methodological issues. *Bioinformatics*, 23(8):980–987, Apr 2007.

[73] O. Gotoh. An improved algorithm for matching biological sequences. *J Mol Biol*, 162(3): 705–8, 1982.

[74] O. Gotoh. Significant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments. *J Mol Biol*, 264 (4):823–38, 1996.

[75] M. Grabowski, A. Joachimiak, Z. Otwinowski, and W. Minor. Structural genomics: keeping up with expanding knowledge of the protein universe. *Curr. Opin. Struct. Biol.*, 17: 347–353, Jun 2007.

[76] Gregory R. Grant et al. Comparative analysis of RNA-Seq alignment algorithms and the RNA-Seq unified mapper (RUM). *Bioinformatics*, 27(18):2518–2528, Sep 2011.

[77] Lesley H Greene, Tony E Lewis, Sarah Addou, Alison Cuff, Tim Dallman, Mark Dibley, Oliver Redfern, Frances Pearl, Rekha Nambudiry, Adam Reid, Ian Sillitoe, Corin Yeats, Janet M Thornton, and Christine A Orengo. The cath domain structure database: new protocols and classification levels give a more comprehensive resource for exploring evolution. *Nucleic acids research*, 35(Database issue):D291–7, 2007.

[78] Chittibabu Guda, Sifang Lu, Eric D Scheeff, Philip E Bourne, and Ilya N Shindyalov. CE-MC: a multiple protein structure alignment server. *Nucleic Acids Res*, 32(Web Server issue):100–103, Jul 2004.

[79] M. J. Guertin, A. L. Martins, A. Siepel, and J. T. Lis. Accurate prediction of inducible transcription factor binding intensities in vivo. *PLoS Genet.*, 8(3):e1002610, 2012.

[80] Auden W. H. and Louis Kronenberger. *The Viking Book of Aphorisms, A Personal Selection*. New York: Dorset Press, 2nd edition, November 01, 1981. ISBN 0880290560.

[81] C. Hadley and D.T. Jones. A systematic comparison of protein structure classifications: SCOP, CATH and FSSP. *Structure*, 7:1099–1112, Sep 1999.

[82] D. Hanisch, J. Fluck, H. T. Mevissen, and R. Zimmer. Playing biology's name game: identifying protein names in scientific text. *Pac Symp Biocomput*, pages 403–14, 2003.

[83] D. Hanisch, K. Fundel, H. T. Mevissen, R. Zimmer, and J. Fluck. ProMiner: rule-based protein and gene entity recognition. *BMC Bioinformatics*, 6 Suppl 1:S14, 2005.

[84] A. Harrison, F. Pearl, R. Mott, J. Thornton, and C. Orengo. Quantifying the similarities within fold space. *J. Mol. Biol.*, 323:909–926, Nov 2002.

[85] A. Harrison, F. Pearl, I. Sillitoe, T. Slidel, R. Mott, J. Thornton, and C. Orengo. Recognizing the fold of a protein structure. *Bioinformatics*, 19:1748–1759, Sep 2003.

[86] J. R. Hesselberth, X. Chen, Z. Zhang, P. J. Sabo, R. Sandstrom, A. P. Reynolds, R. E. Thurman, S. Neph, M. S. Kuehn, W. S. Noble, S. Fields, and J. A. Stamatoyannopoulos. Global mapping of protein-DNA interactions in vivo by digital genomic footprinting. *Nat. Methods*, 6(4):283–289, Apr 2009.

[87] Kristina M Hettne, Rob H Stierum, Martijn J Schuemie, Peter J M Hendriksen, Bob J A Schijvenaars, Erik M van Mulligen, Jos Kleinjans, and Jan A Kors. A dictionary to identify small molecules and drugs in free text. *Bioinformatics*, 25(22):2983–2991, Nov 2009.

[88] Ladeana W. Hillier et al. Massively parallel sequencing of the polyadenylated transcriptome of C. elegans. *Genome Res*, 19(4):657–666, Apr 2009.

[89] Daniel S. Hirschberg. A linear space algorithm for computing maximal common subsequences. *Communications of the ACM*, 18(6):341–343, 1975.

[90] T.A. Holland, S. Veretnik, I.N. Shindyalov, and P.E. Bourne. Partitioning protein structures into domains: why is it so difficult? *J. Mol. Biol.*, 361:562–590, Aug 2006.

[91] L Holm and C Sander. Protein structure comparison by alignment of distance matrices. *J Mol Biol*, 233(1):123–138, Sep 1993.

[92] Brian E. Howard and Steffen Heber. Towards reliable isoform quantification using RNA-SEQ data. *BMC Bioinformatics*, 11 Suppl 3:S6, 2010.

[93] M. Berger (Hrsg.). *PSYCHISCHE ERKRANKUNGEN Klinik und Therapie*. Verlag Urban & Fischer, Muenchen-Jena, 2. auflage edition, 2004. 1264 pp.

[94] W. Huang da, B. T. Sherman, and R. A. Lempicki. Systematic and integrative analysis of large gene lists using david bioinformatics resources. *Nature protocols*, 4(1):44–57, 2009.

[95] Valentin A Ilyin, Alexej Abyzov, and Chesley M Leslin. Structural alignment of proteins by a novel TOPOFIT method, as a superimposition of common volumes at a topomax point. *Protein Sci*, 13(7):1865–1874, Jul 2004.

[96] Trond Grenager Jenny Rose Finkel and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, 1:pp. 363–370, 2005.

[97] Peilin Jia, Chung-Feng Kao, Po-Hsiu Kuo, and Zhongming Zhao. A comprehensive network and pathway analysis of candidate genes in major depressive disorder. *BMC Systems Biology*, 5:1–13, 2011.

[98] Antonio Jimeno, Ernesto Jimenez-Ruiz, Vivian Lee, Sylvain Gaudan, Rafael Berlanga, and Dietrich Rebholz-Schuhmann. Assessment of disease named entity recognition on a corpus of annotated sentences. *BMC Bioinformatics*, 9 Suppl 3:S3, 2008.

[99] D. T. Jones. Protein secondary structure prediction based on position-specific scoring matrices. *J. Mol. Biol.*, 292(2):195–202, Sep 1999.

[100] D. T. Jones, D. W. Buchan, D. Cozzetto, and M. Pontil. PSICOV: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments. *Bioinformatics*, 28(2):184–190, Jan 2012.

[101] R. Kabiljo, A. B. Clegg, and A. J. Shepherd. A realistic assessment of methods for extracting gene/protein interactions from free text. *BMC Bioinformatics*, 10:233, 2009.

[102] W. Kabsch and C. Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12):2577–2637, Dec 1983.

[103] A. Keller, C. Backes, A. Gerasch, M. Kaufmann, O. Kohlbacher, E. Meese, and H. P. Lenhof. A novel algorithm for detecting differentially regulated paths based on gene set enrichment analysis. *Bioinformatics*, 25(21):2787–2794, Nov 2009.

[104] W James Kent. BLAT–the BLAST-like alignment tool. *Genome Res*, 12(4):656–664, Apr 2002.

[105] P. Khatri and S. Draghici. Ontological analysis of gene expression data: current tools, limitations, and open problems. *Bioinformatics*, 21(18):3587–3595, Sep 2005.

[106] A. S. Konagurthu, J. C. Whisstock, P. J. Stuckey, and A. M. Lesk. Mustang: a multiple structural alignment algorithm. *Proteins*, 64(3):559–74, 2006.

[107] Martin Krallinger and Alfonso Valencia. Text-mining and information-retrieval services for molecular biology. *Genome Biol*, 6(7):224, 2005.

[108] R. Küffner, R. Zimmer, and T. Lengauer. Pathway analysis in metabolic databases via differential metabolic display (DMD). *Bioinformatics*, 16(9):825–836, Sep 2000.

[109] Küffner, R. and Petri, T. and Windhager, L. and Zimmer, R. Petri Nets with Fuzzy Logic (PNFL): reverse engineering and parametrization. *PLoS ONE*, 5(9), 2010.

[110] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Res. Logist. Quart.*, 2:83–97, 1955.

[111] L.A. Kurgan, T. Zhang, H. Zhang, S. Shen, and J. Ruan. Secondary structure-based assignment of the protein structural classes. *Amino Acids*, 35:551–564, Oct 2008.

[112] R. Kurzrock, H. M. Kantarjian, B. J. Druker, and M. Talpaz. Philadelphia chromosome-positive leukemias: from basic mechanisms to molecular therapeutics. *Ann. Intern. Med.*, 138(10):819–830, May 2003.

[113] Ben Langmead, Cole Trapnell, Mihai Pop, and Steven L. Salzberg. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol*, 10(3): R25, 2009.

[114] R. H. Lathrop. The protein threading problem with sequence amino acid interaction preferences is NP-complete. *Protein Eng.*, 7(9):1059–1068, Sep 1994.

[115] C.S. Leslie, E. Eskin, A. Cohen, J. Weston, and W.S. Noble. Mismatch string kernels for discriminative protein classification. *Bioinformatics*, 20:467–476, Mar 2004.

[116] M. Levitt and M. Gerstein. A unified statistical framework for sequence comparison and structure comparison. *PNAS*, 95(11):5913–20, 1998.

[117] Bo Li, Victor Ruotti, Ron M. Stewart, James A. Thomson, and Colin N. Dewey. Rna-seq gene expression estimation with read mapping uncertainty. *Bioinformatics*, 26(4): 493–500, Feb 2010.

[118] J. M. Lionberger, M. B. Wilson, and T. E. Smithgall. Transformation of myeloid leukemia cells to cytokine independence by Bcr-Abl is suppressed by kinase-defective Hck. *J Biol Chem*, 275(24):18581–18585, Jun 2000.

[119] M. Liu, A. Liberzon, S. W. Kong, W. R. Lai, P. J. Park, I. S. Kohane, and S. Kasif. Network-based analysis of affected biological processes in type 2 diabetes models. *PLoS Genet.*, 3(6):e96, Jun 2007.

[120] R. Luthy, J. U. Bowie, and D. Eisenberg. Assessment of protein models with three-dimensional profiles. *Nature*, 356(6364):83–5, 1992.

[121] A. D. MacKerell, D. Bashford, Bellott, R. L. Dunbrack, J. D. Evanseck, M. J. Field, S. Fischer, J. Gao, H. Guo, S. Ha, D. Joseph-McCarthy, L. Kuchnir, K. Kuczera, F. T. K. Lau, C. Mattos, S. Michnick, T. Ngo, D. T. Nguyen, B. Prodhom, W. E. Reiher, B. Roux, M. Schlenkrich, J. C. Smith, R. Stote, J. Straub, M. Watanabe, J. Wiórkiewicz-Kuczera, D. Yin, and M. Karplus. All-Atom Empirical Potential for Molecular Modeling and Dynamics Studies of Proteins†. *J. Phys. Chem. B*, 102(18):3586–3616, April 1, 1998.

[122] V. N. Maiorov and G. M. Crippen. Contact potential that recognizes the correct folding of globular proteins. *J. Mol. Biol.*, 227(3):876–888, Oct 1992.

[123] D. S. Marks, L. J. Colwell, R. Sheridan, T. A. Hopf, A. Pagnani, R. Zecchina, and C. Sander. Protein 3D structure computed from evolutionary sequence variation. *PLoS ONE*, 6(12):e28766, 2011.

[124] Debora S. Marks, Thomas A. Hopf, and Chris Sander. Protein structure prediction from sequence variation. *Nat Biotech*, 30(11):1072–1080, Nov 2012. ISSN 1087-0156.

[125] S. Mayewski. A multibody, whole-residue potential for protein structures, with testing by Monte Carlo simulated annealing. *Proteins*, 59(2):152–169, May 2005.

[126] Lynn McCallum, Susan Price, Nathalie Planque, Bernard Perbal, Andrew Pierce, Anthony D Whetton, and Alexandra E Irvine. A novel mechanism for BCR-ABL action: stimulated secretion of CCN3 is involved in growth and differentiation regulation. *Blood*, 108(5):1716–1723, Sep 2006.

[127] L J McGuffin, K Bryson, and D T Jones. What are the baselines for protein fold recognition? *Bioinformatics*, 17(1):63–72, Jan 2001.

[128] R. Meier, A. Mhlethaler-Mottet, M. Flahaut, A. Coulon, C. Fusco, F. Louache, K. Auderset, K.B. Bourloud, E. Daudigeos, C. Ruegg, G. Vassal, N. Gross, and J. Joseph. The chemokine receptor cxcr4 strongly promotes neuroblastoma primary tumour and metastatic growth, but not invasion. *PLoS ONE*, 2(10):e1016, 2007.

[129] I. Melvin, E. Ie, R. Kuang, J. Weston, W.N. Stafford, and C. Leslie. SVM-Fold: a tool for discriminative multi-class protein fold and superfamily recognition. *BMC Bioinformatics*, 8 Suppl 4:S2, 2007.

[130] F. D. Miller and D. R. Kaplan. Neurotrophin signalling pathways regulating neuronal apoptosis. *Cell. Mol. Life Sci.*, 58(8):1045–1053, Jul 2001.

[131] S. Miyazawa and R. L. Jernigan. Residue-residue potentials with a favorable contact pair term and an unfavorable high packing density term, for simulation and threading. *J. Mol. Biol.*, 256(3):623–644, Mar 1996.

[132] D. R. Morrision. PATRICIA-Practical algorithm to retrieve information coded in Alphanumeric. *JACM*, 15(4):514–534, Oct 1968.

[133] Ali Mortazavi, Brian A. Williams, Kenneth McCue, Lorian Schaeffer, and Barbara Wold. Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nat Methods*, 5(7): 621–628, Jul 2008.

[134] J. Moult. A decade of CASP: progress, bottlenecks and prognosis in protein structure prediction. *Curr. Opin. Struct. Biol.*, 15:285–289, Jun 2005.

[135] Tadeo Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, April 1989. ISSN 00189219.

[136] A G Murzin, S E Brenner, T Hubbard, and C Chothia. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J Mol Biol*, 247(4): 536–540, Apr 1995.

[137] Naeem, H. and Küffner, R. and Csaba, G. and Zimmer, R. . miRSel: automated extraction of associations between microRNAs and genes from the biomedical literature. *BMC Bioinformatics*, 11:135, 2010.

[138] S. V. Narayana and P. Argos. Residue contacts in protein structures and implications for protein folding. *Int. J. Pept. Protein Res.*, 24(1):25–39, Jul 1984.

[139] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol*, 48(3):443–453, Mar 1970.

[140] S. Neph, J. Vierstra, A. B. Stergachis, A. P. Reynolds, E. Haugen, B. Vernot, R. E. Thurman, S. John, R. Sandstrom, A. K. Johnson, M. T. Maurano, R. Humbert, E. Rynes, H. Wang, S. Vong, K. Lee, D. Bates, M. Diegel, V. Roach, D. Dunn, J. Neri, A. Schafer, R. S. Hansen, T. Kutyavin, E. Giste, M. Weaver, T. Canfield, P. Sabo, M. Zhang, G. Balasundaram, R. Byron, M. J. MacCoss, J. M. Akey, M. A. Bender, M. Groudine, R. Kaul, and J. A. Stamatoyannopoulos. An expansive human regulatory lexicon encoded in transcription factor footprints. *Nature*, 489(7414):83–90, Sep 2012.

[141] Marius Nicolae, Serghei Mangul, Ion I. Mandoiu, and Alex Zelikovsky. Estimation of alternative splicing isoform frequencies from rna-seq data. *Algorithms Mol Biol*, 6(1):9, 2011.

[142] Theresa Niederberger. Regelbasierte Extraktion von Proteinwechselwirkungen, July 2008.

[143] Marian Novotny, Dennis Madsen, and Gerard J Kleywegt. Evaluation of protein fold comparison servers. *Proteins*, 54(2):260–270, Feb 2004. Evaluation Studies.

[144] H. Ogata, S. Goto, K. Sato, W. Fujibuchi, H. Bono, and M. Kanehisa. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Res.*, 27(1):29–34, Jan 1999.

[145] M. Ohmichi, S. J. Decker, L. Pang, and A. R. Saltiel. Nerve growth factor binds to the 140 kd trk proto-oncogene product and stimulates its association with the src homology domain of phospholipase C gamma 1. *Biochem. Biophys. Res. Commun.*, 179(1):217–223, Aug 1991.

[146] C. A. Orengo, A. D. Michie, S. Jones, D. T. Jones, M. B. Swindells, and J. M. Thornton. Cath–a hierarchic classification of protein domain structures. *Structure*, 5(8):1093–108, 1997.

[147] Joseph O'Rourke. *Computational Geometry in C*. Cambridge University Press, 1993. ISBN 0-521-44034-3.

[148] Qun Pan et al. Deep surveying of alternative splicing complexity in the human transcriptome by high-throughput sequencing. *Nat Genet*, 40(12):1413–1415, Dec 2008.

[149] W. Pan. A comparative review of statistical methods for discovering differentially expressed genes in replicated microarray experiments. *Bioinformatics*, 18(4):546–554, Apr 2002.

[150] R. Parameswaran, M. Yu, M. Lim, J. Groffen, and N. Heisterkamp. Combination of drug therapy in acute lymphoblastic leukemia with a CXCR4 antagonist. *Leukemia*, 25(8): 1314–1323, Aug 2011.

[151] Bogdan Pasaniuc, Noah Zaitlen, and Eran Halperin. Accurate estimation of expression levels of homologous genes in rna-seq experiments. *J Comput Biol*, 18(3):459–468, Mar 2011.

[152] M. Pawlowski, M. J. Gajda, R. Matlak, and J. M. Bujnicki. MetaMQAP: a meta-server for the quality assessment of protein models. *BMC Bioinformatics*, 9:403, 2008.

[153] Pier Paolo Piccaluga, Stefania Paolini, and Giovanni Martinelli. Tyrosine kinase inhibitors for the treatment of Philadelphia chromosome-positive adult acute lymphoblastic leukemia. *Cancer*, 110(6):1178–1186, Sep 2007.

[154] G. Polya. *How to Solve It: A New Aspect of Mathematical Method*. Princeton University Press, 2nd edition, November 01, 1971. ISBN 0691023565.

[155] A Prlic, F S Domingues, and M J Sippl. Structure-derived substitution matrices for alignment of distantly related sequences. *Protein Eng*, 13(8):545–550, Aug 2000.

[156] S. Raman, R. Vernon, J. Thompson, M. Tyka, R. Sadreyev, J. Pei, D. Kim, E. Kellogg, F. DiMaio, O. Lange, L. Kinch, W. Sheffler, B. H. Kim, R. Das, N. V. Grishin, and D. Baker. Structure prediction for casp8 with all-atom refinement using rosetta. *Proteins*, 77 Suppl 9:89–99, 2009.

[157] Amanda J Redig, Eliza Vakana, and Leonidas C Platanias. Regulation of mammalian target of rapamycin and mitogen activated protein kinase pathways by BCR-ABL. *Leuk Lymphoma*, 52 Suppl 1:45–53, Feb 2011.

[158] G.A. Reeves, T.J. Dallman, O.C. Redfern, A. Akpor, and C.A. Orengo. Structural diversity of domain superfamilies in the CATH database. *J. Mol. Biol.*, 360:725–741, Jul 2006.

[159] F M Richards. The interpretation of protein structures: total volume, group volume distributions and packing density. *J Mol Biol*, 82(1):1–14, Jan 1974.

[160] Jeffrey Roach, Shantanu Sharma, Maryna Kapustina, and Charles W Jr Carter. Structure alignment via Delaunay tetrahedralization. *Proteins*, 60(1):66–81, Jul 2005.

[161] J. Rocha, J. Segura, R. C. Wilson, and S. Dasgupta. Flexible structural protein alignment by a sequence of local transformations. *Bioinformatics*, 25(13):1625–1631, Jul 2009.

[162] S. Russell. Efficient memory-bounded search methods. *ECAI*, pages 1–5, 1992.

[163] G. Schramm, S. Wiesberg, N. Diessl, A. L. Kranz, V. Sagulenko, M. Oswald, G. Reinelt, F. Westermann, R. Eils, and R. Konig. PathWave: discovering patterns of differentially regulated enzymes in metabolic pathways. *Bioinformatics*, 26(9):1225–1231, May 2010.

[164] Burr Settles. Abner: an open source tool for automatically tagging genes, proteins and other entity names in text. *Bioinformatics*, 21(14):3191–3192, Jul 2005.

[165] M. Shatsky, R. Nussinov, and H. J. Wolfson. Optimization of multiple-sequence alignment based on multiple-structure alignment. *Proteins*, 62(1):209–17, 2006.

[166] Maxim Shatsky, Ruth Nussinov, and Haim J Wolfson. A method for simultaneous alignment of multiple protein structures. *Proteins*, 56(1):143–156, Jul 2004.

[167] I. N. Shindyalov and P. E. Bourne. Protein structure alignment by incremental combinatorial extension (ce) of the optimal path. *Protein Eng*, 11(9):739–47, 1998.

[168] N. Siew, A. Elofsson, L. Rychlewski, and D. Fischer. Maxsub: an automated measure for the assessment of protein structure prediction quality. *Bioinformatics*, 16(9):776–85, 2000.

[169] Ilina Singh and Nikolas Rose. Biomarkers in psychiatry. *Nature*, 460(7252):202–207, Jul 2009.

[170] M. J. Sippl. Boltzmann's principle, knowledge-based mean fields and protein folding. An approach to the computational determination of protein structures. *J. Comput. Aided Mol. Des.*, 7(4):473–501, Aug 1993.

[171] J. Söding. Protein homology detection by hmm-hmm comparison. *Bioinformatics*, 21(7): 951–60, 2005.

[172] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, and J. P. Mesirov. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proc. Natl. Acad. Sci. U.S.A.*, 102(43):15545–15550, Oct 2005.

[173] Marc Sultan et al. A global view of gene activity and alternative splicing by deep sequencing of the human transcriptome. *Science*, 321(5891):956–960, Aug 2008.

[174] C. Sutton and A. McCallum. An introduction to conditional random fields for relational learning. In Ben Taskar (Eds.) Lise Getoor, editor, *Introduction to Statistical Relational Learning*, 2006.

[175] S. Tanaka and H. A. Scheraga. Medium- and long-range interaction parameters between amino acids for predicting three-dimensional structures of proteins. *Macromolecules*, 9 (6):945–950, 1976.

[176] W R Taylor. Protein structure comparison using iterated double dynamic programming. *Protein Sci*, 8(3):654–665, Mar 1999.

[177] W R Taylor and C A Orengo. Protein structure alignment. *J Mol Biol*, 208(1):1–22, Jul 1989.

[178] The Flux Project. Flux simulator version 1.0-rc4, 2011. URL `http://flux.sammeth.net`.

[179] R. Thiele, R. Zimmer, and T. Lengauer. Protein threading by recursive dynamic programming. *J Mol Biol*, 290(3):757–79, 1999.

[180] A.E. Todd, C.A. Orengo, and J.M. Thornton. Evolution of function in protein superfamilies, from a structural perspective. *J. Mol. Biol.*, 307:1113–1143, Apr 2001.

[181] Cole Trapnell, Lior Pachter, and Steven L. Salzberg. TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics*, 25(9):1105–1111, May 2009.

[182] I. Ulitsky and R. Shamir. Identification of functional modules using network topology and high-throughput data. *BMC Syst Biol*, 1:8, 2007.

[183] S. Vajda, M. Sippl, and J. Novotny. Empirical potentials and functions for protein folding and binding. *Curr. Opin. Struct. Biol.*, 7(2):222–228, Apr 1997.

[184] Anja von Heydebreck, Wolfgang Huber, and Robert Gentleman. Differential Expression with the Bioconductor Project. *Bioconductor Project Working Papers*, 7, 2004.

[185] Niklas von Ohsen, Ingolf Sommer, and Ralf Zimmer. Profile-profile alignment: a powerful tool for protein structure prediction. *Pac Symp Biocomput*, pages 252–263, 2003.

[186] Niklas von Ohsen, Ingolf Sommer, Ralf Zimmer, and Thomas Lengauer. Arby: automatic protein structure prediction using profile-profile alignment and confidence measures. *Bioinformatics*, 20(14):2228–2235, Sep 2004. Evaluation Studies.

[187] Niklas von Öhsen and Ralf Zimmer. Improving profile-profile alignments via log average scoring. In *WABI*, pages 11–26, 2001.

[188] G. Voronoi. Nouvelles applications des parametres continus a la theorie des formes quadratiques. *J. f. d. Reine und Angewandte Mathematik*, 134:198–287, 1908.

[189] S. R. Walker, P. D. Ogagan, D. DeAlmeida, A. M. Aboka, and E. M. Barksdale. Neuroblastoma impairs chemokine-mediated dendritic cell migration in vitro. *J. Pediatr. Surg.*, 41(1):260–265, Jan 2006.

[190] B. Wallner. ProQ parameters, 2010. Personal communication.

[191] B. Wallner and A. Elofsson. Can correct protein models be identified? *Protein Sci*, 12(5): 1073–86, 2003.

[192] A Wallqvist, Y Fukunishi, L R Murphy, A Fadel, and R M Levy. Iterative sequence/secondary structure search for protein homologs: comparison with amino acid sequence alignments and application to fold recognition in genome databases. *Bioinformatics*, 16(11):988–1002, Nov 2000.

[193] Eric T. Wang et al. Alternative isoform regulation in human tissue transcriptomes. *Nature*, 456(7221):470–476, Nov 2008.

[194] Kai Wang et al. MapSplice: accurate mapping of RNA-seq reads for splice junction discovery. *Nucleic Acids Res*, 38(18):e178, Oct 2010.

[195] Z. Wang, M. Gerstein, and M. Snyder. RNA-Seq: a revolutionary tool for transcriptomics. *Nat. Rev. Genet.*, 10(1):57–63, Jan 2009.

[196] D. L. Wheeler, C. Chappey, A. E. Lash, D. D. Leipe, T. L. Madden, G. D. Schuler, T. A. Tatusova, and B. A. Rapp. Database resources of the National Center for Biotechnology Information. *Nucleic Acids Res*, 28(1):10–14, Jan 2000.

[197] Lukas Windhager, Florian Erhard, and Ralf Zimmer. Fuzzy modeling. In Ina Koch, Wolfgang Reisig, and Falk Schreiber, editors, *Modeling in Systems Biology: The Petri Net Approach*. Springer, 2010.

[198] S. Wu and Y. Zhang. Recognizing protein substructure similarity using segmental threading. *Structure*, 18(7):858–867, Jul 2010.

[199] J. Xu, M. Li, D. Kim, and Y. Xu. RAPTOR: optimal protein threading by linear programming. *J Bioinform Comput Biol*, 1(1):95–117, Apr 2003.

[200] Yuzhen Ye and Adam Godzik. Flexible structure alignment by chaining aligned fragment pairs allowing twists. *Bioinformatics*, 19 Suppl 2:II246–II255, Oct 2003.

[201] Yuzhen Ye and Adam Godzik. Multiple flexible structure alignment using partial order graphs. *Bioinformatics*, 21(10):2362–2369, May 2005. Evaluation Studies.

[202] Lotfali Askar Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[203] A. Zemla, C. Venclovas, J. Moult, and K. Fidelis. Processing and analysis of casp3 protein structure predictions. *Proteins*, Suppl 3:22–9, 1999.

[204] Y. Zhang, I.A. Hubner, A.K. Arakaki, E. Shakhnovich, and J. Skolnick. On the origin and highly likely completeness of single-domain protein structures. *Proc. Natl. Acad. Sci. U.S.A.*, 103:2605–2610, Feb 2006.

[205] Y. Zhang and J. Skolnick. The protein structure prediction problem could be solved using the current PDB library. *Proc. Natl. Acad. Sci. U.S.A.*, 102:1029–1034, Jan 2005.

[206] Y. Zhang and J. Skolnick. TM-align: a protein structure alignment algorithm based on the TM-score. *Nucleic Acids Res.*, 33:2302–2309, 2005.

[207] Yang Zhang and Jeffrey Skolnick. Scoring function for automated assessment of protein structure template quality. *Proteins*, 57(4):702–710, Dec 2004.

[208] A. Zien, R. Kueffner, R. Zimmer, and T. Lengauer. Analysis of gene expression data with pathway scores. *Proceedings ISMB2000*, 8:407–17, 2000.

[209] R Zimmer, M Wohler, and R Thiele. New scoring schemes for protein fold recognition based on Voronoi contacts. *Bioinformatics*, 14(3):295–308, 1998.