

*Anna Katharina Dehof*  
Novel Approaches for Bond Order Assignment  
and NMR Shift Prediction



# **Novel Approaches for Bond Order Assignment and NMR Shift Prediction**

## **Dissertation**

zur Erlangung des Grades eines  
Doktors der Naturwissenschaften (Dr. rer. nat.)  
der Naturwissenschaftlich-Technischen Fakultät I  
– Mathematik und Informatik –  
der Universität des Saarlandes

vorgelegt von

**Anna Katharina Dehof, M.Sc.**

Saarbrücken  
2011

© 2011 Anna Katharina Dehof

Covergestaltung und Titelbild: © 2011 Anna Katharina Dehof, BALL developer team,

<http://www.ball-project.org>

Herstellung und Verlag: Books on Demand GmbH, Norderstedt

ISBN 978-3-8482-0773-2

Bibliografische Informationen der Deutschen Nationalbibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen

Nationalbibliographie; detaillierte bibliografische Daten sind im Internet über

<http://dnb.d-nadb.de> abrufbar.

Tag des Kolloquiums	15.03.2012
Dekan	Prof. Dr. Holger Hermanns
Berichterstatter	Prof. Dr. Hans-Peter Lenhof Prof. Dr. Sebastian Böcker
Vorsitz	Prof. Dr. Philipp Slusallek
Akad. Mitarbeiter	Dr. Oliver Müller



**Eidesstattliche Versicherung**

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form in einem Verfahren zur Erlangung eines akademischen Grades vorgelegt.

Saarbrücken, den 17. November 2011

---

(Anna Katharina Dehof)



## **Abstract**

Molecular modelling is one of the cornerstones of modern biological and pharmaceutical research. Accurate modelling approaches easily become computationally overwhelming and thus, different levels of approximations are typically employed. In this work, we develop such approximation approaches for problems arising in structural bioinformatics. A fundamental approximation of molecular physics is the classification of chemical bonds, usually in the form of integer bond orders. Many input data sets lack this information, but several problems render an automated bond order assignment highly challenging. For this task, we develop the BOA Constructor method which accounts for the non-uniqueness of solutions and allows simple extensibility. Testing our method on large evaluation sets, we demonstrate how it improves on the state of the art. Besides traditional applications, bond orders yield valuable input for the approximation of molecular quantities by statistical means. One such problem is the prediction of NMR chemical shifts of protein atoms. We present our pipeline NightShift for automated model generation, use it to create a new prediction model called Spinster, and demonstrate that it outperforms established manually developed approaches. Combining Spinster and BOA Constructor, we create the Liops-model that for the first time allows to efficiently include the influence of non-protein atoms. Finally, we describe our work on manual modelling techniques, including molecular visualization and novel input paradigms.

## **German Abstract**

Molekulare Modellierungsmethoden gehören zu den Grundpfeilern moderner biologischer und pharmazeutischer Forschung. Akkurate Modellierungsmethoden erfordern jedoch enormen Rechenaufwand, weshalb üblicherweise verschiedene Näherungsverfahren eingesetzt werden. In dieser Arbeit entwerfen wir solche Näherungen für verschiedene Probleme aus der strukturbasierten Bioinformatik. Eine fundamentale Näherung der molekularen Physik ist die Einteilung chemischer Bindungen in wenige Klassen, meist in der Form ganzzahliger Bindungsordnungen. In vielen Datensätzen ist diese Information nicht enthalten und eine automatische Zuweisung ist hochgradig schwierig. Für diese Problemstellung entwickeln wir die BOA Constructor-Methode, die sowohl mit uneindeutigen Lösungen umgehen kann als auch vom Benutzer leicht erweitert werden kann. In umfangreichen Tests zeigen wir, dass unsere Methode dem bisherigen Stand der Forschung überlegen ist. Neben klassischen Anwendungen liefern Bindungsordnungen wertvolle Informationen für die statistische Vorhersage molekularer Eigenschaften wie z.B. der chemischen Verschiebung von Proteinatomen. Mit der NightShift-Pipeline stellen wir ein Verfahren zur automatischen Generierung von Vorhersagemodellen vor, generieren damit das neuartige Spinster-Modell und zeigen, dass es bisherigen manuell entwickelten Verfahren überlegen ist. Die Kombination mit BOA Constructor führt zum sogenannten Liops-Modell, welches als erstes Modell die effiziente Berücksichtigung des Einflusses von nicht-Proteinatomen erlaubt. Abschließend stellen wir unsere Arbeiten zu manuellen Modellierungsmethoden vor, welche z.B. die molekulare Visualisierung oder neue Eingabemethoden beinhalten.

## German Summary

Methoden des „Molecular Modelling“ gehören heute zu den wichtigsten Grundpfeilern der modernen biologischen und pharmazeutischen Forschung. Dabei haben diese Techniken auch weit über die Lebenswissenschaften hinaus wichtige Anwendungsfelder, die z.B. die physikalische Chemie oder die Materialwissenschaften umfassen.

Grundlage der Modelling-Methoden ist eine Betrachtung der zu untersuchenden Systeme auf molekularer, meist sogar auf atomarer Ebene. Dies erfordert im Allgemeinen eine möglichst genaue Vorhersage molekularer und atomarer Wechselwirkungen, die als Basis zur Berechnung vieler weiterer Eigenschaften dienen können. Die genauesten Methoden zur Vorhersage solcher Wechselwirkungen basieren auf der Quantenmechanik und sind damit physikalisch fundiert. In der Praxis jedoch sind sie für viele biologische Anwendungen erheblich zu aufwändig: Die akkurate Simulation quantenmechanischer Systeme gehört zu den größten Herausforderungen des wissenschaftlichen Rechnens.

Erheblich erschwert wird die Situation durch die enorme Größe typischer biomolekularer Systeme, die oft mehrere Tausend bis mehrere Millionen Freiheitsgrade aufweisen. Da schließlich in vielen Anwendungen in der Praxis wiederum mehrere Tausend bis mehrere Millionen Auswertungen benötigt werden, müssen meist Näherungsverfahren bemüht werden. Dabei wird die *ab-initio* Simulation atomarer und molekularer Eigenschaften aus quantenmechanischen Gesetzmäßigkeiten durch approximative Modelle ersetzt, die entweder auf vereinfachten physikalischen Gesetzen, auf statistischen Inferenzmethoden oder auf einer hybriden Kombination beider beruhen. In dieser Arbeit werden wir solche Näherungsverfahren für verschiedene Probleme der strukturbasierten Bioinformatik entwickeln.

Zu den grundlegendsten Näherungen an die Molekülphysik gehört sicherlich die Klassifikation chemischer Bindungen in eine kleine Menge verschiedener Bindungstypen wie z.B. den Einfach-, Doppel-, Dreifach- oder Vierfach- sowie den delokalisierten oder aromatischen Bindungen. Obwohl diese Klassifikation die tatsächlichen Verhältnisse nicht perfekt beschreibt und viele Fälle nicht eindeutig in eine der Klassen eingeteilt werden können, hat sie sich in der Praxis hervorragend bewährt und ist in der gesamten Chemie weit verbreitet.

Beim Molecular Modelling wird diese Menge von Bindungstypen jedoch typischerweise weiter reduziert, so dass nur noch Einfach-, Doppel-, Dreifach- und manchmal noch Vierfachbindungen betrachtet werden. In diesem Fall wird jede Bindung durch eine natürliche Zahl beschrieben, die als *Bindungsordnung* bezeichnet wird und Werte zwischen eins und vier annehmen kann. Diese Bindungsordnungen liegen allen Kraftfeldmethoden zugrunde, die in vielerlei Hinsicht das Rückgrat des Molecular Modelling bilden. In der Praxis beschreiben Bindungsordnung eine Vielzahl an Phänomenen sehr gut, doch führen sie auch zu einer Reihe von Problemen. Das vielleicht schwerwiegendste dieser Probleme ist die Uneindeutigkeit der Zuordnung von Bindungsordnungen zu einem gegebenen Molekül. Da Bindungsordnungen nur eine recht grobe Approximation der tatsächlichen molekularen Eigenschaften darstellen, können verschiedene Effekte, wie z.B. delokalisierte oder aromatische Elektronen, nur durch Kombinationen verschiedener Bindungsordnungszuweisungen beschrieben werden. Hinzu kommt, dass die Vorhersage der Bindungsordnungen für ein gegebenes Molekül ein schwieriges Problem darstellt und im allgemeinen nur über komplexe Regelsysteme, die von chemischen Experten entworfen werden, gelöst werden kann. Durch diese Probleme bleibt die automatische Zuweisung von Bindungsordnungen eine hochgradig schwierige Herausforderung. Andererseits enthalten viele wichtige Eingabedaten für das Molecular Modelling keine, unvollständige oder falsche Bindungsordnungen, so dass deren automatische Bestimmung eine wichtige Komponente aller Modelling-Systeme darstellt.

In dieser Arbeit entwickeln wir neue Ansätze zur automatischen Zuweisung von Bindungsordnungen, die so genau wie möglich sind, gleichzeitig aber mit der inhärenten Uneindeutigkeit umzugehen wissen. Um sich den veränderlichen Anforderungen der Anwender und persönlichen Präferenzen anpassen zu können, erlaubt unser System die einfache Modifikation des zugrundeliegenden Regelsatzes, wenn gewünscht sogar zur Laufzeit. Die Grundlage unseres Zugangs ist dabei ein Ansatz, der ursprünglich für das „Generalized Amber Force Field (GAFF)“ entwickelt wurde. Dieser Ansatz wurde von uns stark erweitert und in Form eines exakt lösbaren diskreten Optimierungsproblems neu gefasst. In dieser

Dissertation werden wir sowohl dieses Optimierungsproblem als auch verschiedene Algorithmen zu seiner effizienten exakten Lösung vorstellen. Darüberhinaus besprechen wir die Implementierung der Methode, die wir als *BOA Constructor* bezeichnen, im Rahmen der BALL-Bibliothek und zeigen, wie diese in eigenen Programmen oder mit Hilfe der graphischen Benutzerschnittstelle verwendet werden kann. Schließlich demonstrieren wir anhand umfangreicher Evaluationsdatensätze, dass BOA Constructor dem bisherigen Stand der Forschung überlegen ist.

BOA Constructor liefert valide Eingaben für alle traditionellen Anwendungen der Bindungsordnungszuweisung. Dies beinhaltet z.B. die Atomtypisierungsprozeduren molekularmechanischer Kraftfelder, die u.a. für die energetische Bewertung, Strukturoptimierung und Molekulardynamiksimulationen eingesetzt werden, aber auch die Detektion flexibler Gruppen oder aromatischer Bindungen für QSAR-Applikationen oder die rechnergestützte kombinatorische Chemie. Doch über diese klassischen Anwendungen hinaus liefert die zuverlässige Zuweisung von Bindungsordnungen auch wertvolle Informationen für die Vorhersage molekularer Eigenschaften durch statistische Methoden.

Ein wichtiges Beispiel einer solchen Anwendung ist die rechnergestützte Vorhersage chemischer Verschiebungen von Proteinatomen im NMR-Experiment, die für die Aufklärung biomolekularer Strukturen immer wichtiger wird. Chemische Verschiebungen hängen sehr sensitiv von der chemischen und räumlichen Nachbarschaft der betrachteten Atome ab, weshalb sie wichtige Informationen über den räumlichen Aufbau des Moleküls liefern. Zur Vorhersage chemischer Verschiebungen für eine Kandidatenstruktur verwendet man üblicherweise eine Kombination effizienter semi-klassischer Approximationen der Physik des NMR-Experiments in Verbindung mit statistischen Regressionsmethoden.

Zur Entwicklung neuer Vorhersagemethoden für chemische Verschiebungen von Proteinatomen stellen wir die Pipeline *NightShift* vor, die die Erzeugung aktueller Trainings- und Evaluationsdatensätze, das Training neuer hybrider Vorhersagemodelle und deren Evaluierung umfasst. Während alternative Modelle einen enormen, größtenteils *manuellen* Arbeitsaufwand bei der Entwicklung und Anpassung neuer Daten erzeugen, erlaubt *NightShift* die automatische Generierung eines aktuellen Modells, basierend auf der aktuellen Datenlage. Die von uns durch *NightShift* automatisch erzeugten Modelle erzielen dabei eine teils erheblich höhere Genauigkeit als die etablierten manuell generierten Methoden. Das erfolgreichste unserer neuen Modelle basiert auf einer Kombination semi-klassischer Terme mit einem Random-Forest-Modell und wird von uns als *Spinster* bezeichnet.

Die Kombination von *NightShift* und *Spinster* mit BOA Constructor erlaubt uns schließlich, ein wichtiges ungeklärtes Problem des Molecular Modelling zu betrachten: die Vorhersage des Einflusses von nicht-Proteinatomen – z.B. aus einem eventuellen Liganden – auf die chemische Verschiebung der Atome im Protein. Abgesehen von enorm rechenaufwändigen quantenchemischen Modellen ignorieren alle derzeit vorhandenen Vorhersagemodelle diesen Einfluß völlig. In unserem neuartigen Modell, welches wir als *Liops* bezeichnen, verwenden wir Atomtypisierungen, die wiederum auf unseren Bindungsordnungen beruhen. Die resultierenden Typen dienen uns als Eingaben in statistische Regressionsverfahren für die Vorhersage der chemischen Verschiebung. Obwohl *Liops* derzeit eher als „proof-of-concept“ denn als fertiges Vorhersagemodell betrachtet werden sollte, zeigt es bereits jetzt den Wert bindungsordnungs- und atomtypbasierter Deskriptoren für schwierige Vorhersageprobleme.

Zusätzlich zu diesen Beiträgen zum automatisierten Molecular Modelling waren wir stark in mehrere Projekte auf dem Gebiet der manuellen Modelling-Verfahren involviert. Auch diese Projekte, die u.a. Arbeiten auf dem Gebiet der Molekülvisualisierung und der neuartigen Eingabemethoden umfassen, werden wir in dieser Arbeit kurz vorstellen.

## **Danksagung**

Meinen Dank für großartige Hilfe und Unterstützung während der Erstellung der vorliegenden Arbeit möchte ich an dieser Stelle vielen Menschen, vor allem meinen Betreuern, Kollegen, Freunden und meiner Familie aussprechen. In allen Phasen der Arbeit an dieser Dissertation standen mir sehr geschätzte, professionelle und hilfsbereite Menschen zur Seite. Diesen möchte ich für ihre Anregungen und ihre Unterstützung ganz herzlich danken.

In allererster Linie möchte ich hier meinem Doktorvater Herrn Prof. Dr. Hans-Peter Lenhof für die Heranführung an die Bioinformatik und die hervorragende wissenschaftliche Betreuung meinen Dank aussprechen. Ohne sein herausragendes Engagement hätte ich mich wohl nicht für die Bioinformatik entschieden.

Für eine großartige Kooperation und andauernde Unterstützung möchte ich als nächstes Herrn Prof. Dr. Sebastian Böcker und seinen Mitarbeitern am Lehrstuhl für Bioinformatik der Universität Jena, insbesondere Herrn Dr. Quang Anh Bui Bao, danken.

Eine ähnlich enge Zusammenarbeit bestand zum Lehrstuhl für Computergraphik an der Universität des Saarlandes, bzw. dem Deutschen Forschungszentrum für künstliche Intelligenz. Besonders bedanken möchte ich mich hier bei Herrn Prof. Dr. Philipp Slusallek, Lukas Marsalek, Iliyan Georgiev, Mike Phillips, Dr. Hilko Hoffmann, Georg Demme, Rainer Jochem und Michala Rehorova.

Auch Herr Prof. Dr. Oliver Kohlbacher und Herr Prof. Dr. Peter Bayer haben meine Arbeit wissenschaftlich begleitet und standen mir jederzeit mit Rat und Tat zur Seite. Auch dafür möchte ich mich herzlich bedanken. Die vielen fachlichen Diskussionen und die daraus entstandenen Ideen waren prägend für diese Arbeit.

Nicht fehlen dürfen an dieser Stelle meine Kolleginnen und Kollegen am Zentrum für Bioinformatik Saar, besonders das BALL-Team, mit dem ich unzählige Codingsessions, teils bis früh morgens, verbracht habe: Andreas Hildebrandt, Daniel Stöckel, Stefan Nickels, Sabine Müller, Wolfgang Herget, Simon Loew und Lara Schneider. Auch Benny Kneissl, Alexander Rurainski, Sophie Weggler, Jana Panning, Matthias Dietzen und Lars Hildebrandt haben wertvolle Beiträge zu meiner Arbeit geliefert. Ein ganz besonderer Dank gebührt meinen Freunden, deren Begleitung durch meine Promotionszeit, ihre Ratschläge und ihre mentale Unterstützung ich nicht missen möchte. Besonders beigetragen zu dieser Arbeit haben Beate Radics, Stefanie Collin, Linda Wolters, Judith Radics und der Chorfahrdienst Schreiber.

Herzlich danken möchte ich auch meiner Familie, die mich über die verschiedenen Stationen meiner Ausbildung hinweg langmütig mitgetragen und unterstützt und mir nicht nur orthopädisch den Rücken gestärkt hat!

Zum Entstehen dieser Arbeit hat in besonderem Maße auch Andreas beigetragen, der mich stets bestärkt und ermutigt hat. Bei ihm bedanke ich mich für seine Geduld, seinen Zuspruch und seine liebevolle Unterstützung während meiner Promotion, den Höhen und besonders den Tiefen. Danke.

# Contents

<b>1. Introduction</b>	<b>5</b>
<b>2. Optimal Bond Order Assignment</b>	<b>11</b>
2.1. Introduction	11
2.2. Problem Definition	13
2.3. Former Approaches	15
2.4. The Antechamber Approach	16
2.4.1. The Heuristic Trial-and-Error Approach of Antechamber	17
2.5. Aims of our Work	19
2.6. Solution Schemes	19
2.6.1. An A* Approach	20
2.6.2. An Integer Linear Program Approach	24
2.6.3. A Fixed Parameter Tractability Approach	27
2.7. Extensions of our Approach	33
2.7.1. Introducing Structural Information	34
2.7.2. Hybrid Penalty Score	35
2.7.3. Hydrogens	36
2.7.4. Selective Bond Order Assignment	37
2.8. New Penalty Table	38
2.9. Results	40
2.9.1. Data Sets for Validation	41
2.9.2. Comparison to Antechamber	41
2.9.3. Comparison to Reference Assignments	44
2.9.4. Comparison of Running Times	45
2.9.5. Comparison of Penalty Tables	45
2.9.6. Incorporating Structural Information	49
2.10. Summary	50
2.11. Outlook	50
<b>3. NMR Shift Prediction</b>	<b>53</b>
3.1. Introduction	53
3.2. NMR Spectroscopy	57
3.2.1. The Magnetic Field	57
3.2.2. The NMR Experiment	59
3.2.3. Chemical Shift Contributions	60
3.3. Former Approaches for Chemical Shift Prediction	64
3.4. Aims of our Work	67
3.5. Materials and Methods	68
3.5.1. Data Set Construction	69
3.5.2. A new Hybrid Model	73
3.6. Results	87
3.6.1. The NightShift Pipeline	88

3.6.2.	The Data Set . . . . .	94
3.6.3.	The Features . . . . .	96
3.6.4.	The Pure Protein Models . . . . .	97
3.7.	Application to Protein-Ligand Complexes: A Proof of Principle Study . . . . .	99
3.7.1.	General Methodology . . . . .	101
3.7.2.	The Underlying Protein Model . . . . .	101
3.7.3.	Towards a Model of the Ligand Influence . . . . .	101
3.7.4.	Features for the Influence of the Ligand on Protein Atoms . . . . .	102
3.7.5.	Performance Evaluations . . . . .	103
3.7.6.	Results . . . . .	103
3.7.7.	Discussion . . . . .	107
3.8.	Summary . . . . .	108
3.9.	Outlook . . . . .	109
<b>4.</b>	<b>The BALL Project: A Framework for Biomolecular Modelling</b>	<b>111</b>
4.1.	Introduction . . . . .	111
4.2.	The Implementation of BOA Constructor . . . . .	112
4.2.1.	The GUI . . . . .	115
4.2.2.	Python Interface . . . . .	115
4.3.	Implementation of NightShift, Spinster, and Liops . . . . .	115
4.3.1.	A Grammar for CIF . . . . .	115
4.3.2.	Features Required for the Models Spinster and Liops . . . . .	116
4.4.	Manual Molecular Modelling . . . . .	116
4.4.1.	Molecular Visualization . . . . .	117
4.4.2.	Stereoscopic Visualization . . . . .	118
4.4.3.	Multitouch Interaction Functionality . . . . .	119
4.5.	Application of Ray Tracing to Automated Molecular Modelling . . . . .	122
4.5.1.	Geometric Molecular Properties Through Ray Casting . . . . .	122
4.5.2.	Results . . . . .	123
<b>5.</b>	<b>Conclusion</b>	<b>127</b>
<b>6.</b>	<b>Authors Contributions</b>	<b>129</b>
6.1.	Bond Order Assignment . . . . .	129
6.2.	NMR Shift Prediction . . . . .	129
6.3.	BALL Project . . . . .	129
<b>A.</b>	<b>Supplementary Information on BOA Constructor</b>	<b>131</b>
A.1.	A New Penalty Table for Bond Order Assignment . . . . .	131
A.2.	Molecular Structures for the new Rules in the Penalty Table . . . . .	137
A.3.	New Heuristic Approaches for Bond Order Assignment . . . . .	139
A.3.1.	K-Greedy . . . . .	139
A.3.2.	Branch & Bound . . . . .	139
A.4.	Selected Aspects of the Implementation of BOA Constructor . . . . .	142
A.4.1.	Python Interface . . . . .	142
A.4.2.	Graphical User Interface . . . . .	143
A.4.3.	BOA Constructor Options . . . . .	146
A.4.4.	Distribution of Penalty Rules . . . . .	147
A.5.	A* Performance Measures . . . . .	148



<b>B. Supplementary Information on NightShift, Spinster, and Liops</b>	<b>155</b>
B.1. Parameters for Semi-Classical NMR Chemical Shift Predictors . . . . .	155
B.2. Lexer and Parser for CIF - Files . . . . .	155
B.2.1. A FLEX-based Lexer for CIF - Files . . . . .	155
B.2.2. A Grammar for CIF - Files . . . . .	161
<b>C. Copyrights of Figures and Quotations</b>	<b>163</b>
<b>D. Publications and Talks by the Author</b>	<b>165</b>
D.1. Journal Publications (peer-reviewed) . . . . .	165
D.2. Conference Proceedings (peer-reviewed) . . . . .	165
D.3. Conference Proceedings (not peer-reviewed) . . . . .	166
D.4. Posters . . . . .	166
D.5. Talks at International Conferences . . . . .	167
D.6. Talks at Research Institutions . . . . .	167
D.7. Technical Demonstrations at International Events . . . . .	167
D.8. Publications in Preparation . . . . .	167



# 1. Introduction

Counteracting the effects of diseases has always been one of the major driving forces behind the development of science. For thousands of years, humans have attempted to alleviate their ailments through the use of more or less effective substances, and today, medical spending easily belongs to the largest classes of expenditures. In the year 2010 alone, worldwide spending on prescription drugs arrived at a volume of \$865 billion, and it is expected to cross the \$1 trillion mark by 2014 [DFS<sup>+</sup>11, Ait11]. Consequently, vast investments are made annually to try and design novel therapeutics: according to recent estimates, the pharmaceutical industry spent \$67.4 billion on research and development in 2010, up from \$47.6 billion in 2004 [Pha11]. But despite these enormous global efforts, the tangible output is surprisingly small: the amount of new chemical entities approved by the American Food and Drug Administration per year has remained stable for decades at an average of roughly 20. Comparing these huge investments with the number of produced drugs, it becomes obvious that the development of a single drug is a very expensive enterprise [DGM<sup>+</sup>09]. Estimates vary widely, depending on which factors are taken into account. Including failed attempts and marketing costs, a recent survey computed an average cost of \$1.778 billion per drug [PMD<sup>+</sup>10]. The rates of failure are high – [AB03] states that roughly 1 in 1000 drugs pass the pre-clinical stage – and it is assumed that only three out of twenty approved drugs generate more revenue than the costs their development incurred. In addition, development times are exceedingly large, with typical estimates ranging between 10-15 years [DiM01, DHG03, DG04, DGM<sup>+</sup>09].

These numbers raise the question whether the amount of effort put into drug design is really worth its while for society as a whole. Obviously, the treatment of previously incurable diseases, as could be observed with the advent of the first medication against HIV, has a tremendous effect on the afflicted population. But there are strong indications that better drugs profit society in general: according to a recent study by Lichtenberg, life expectancy at birth in the United States of America increased by 2.37 years in the period from 1991 to 2004. Of these, advances in medical imaging technology were found responsible for 0.62-0.71 years, while newer drugs accounted for 1.44-1.8 years [Lic09] and some factors were strongly related with a decrease, i.e., prevented a further increase in life expectancy, such as growing obesity and declining quality in medical education.

The increase in life expectancy will impact the age class structure of society as well. The average life expectancy is assumed to increase from 68 in the years 2005-2010 to 76 in the years 2045-2050 and the number of people older than 60 years are assumed to increase from 667 million (2005-2010) to about two billion (2045-2050) [Uni09]. In addition, with the work force becoming increasingly older on average, special care will be needed to keep productivity high. Taking average development times into account, suitable medication targeting such age-related demands [LDH82, Col10] needs to be researched as soon as possible and will thus likely evolve into a main research focus in the near future.

To understand how Bioinformatics can help to develop novel therapies, it is important to analyse the inner workings of drugs in the body. The general mechanism behind the action of drugs has only been understood very recently. Today, we know that most drugs work by inhibiting the action of biological macromolecules, mostly proteins [Koc76]. The basic principle behind this inhibition was first described – albeit in a simpler fashion – by Emil Fischer in 1894 [Fis94a, Fis94b], who coined the term “Lock and Key” principle. This principle hinges upon the idea that molecules need to come into close physical contact if they are to exert an effect on each other. Sensitivity can be ensured if the contact area is maximized for the desired binding partner. Vice versa, sensitivity can be achieved by minimizing the contact area for other molecules. To this end, one of the two binding partners features

## 1. Introduction

a clearly structured binding site (the lock), the other one a negative imprint of the lock (the key), so that part of one molecule fits well into a part of the other. In drug design, a small molecule mimics the macromolecule's natural binding partner in a way that allows it to enter the binding pocket. It is designed to bind stronger than the natural ligand and to remain chemically stable in the bound conformation. Hence, it blocks the protein's binding site, inhibiting whatever action the protein would perform on its ligand (or vice versa) otherwise.

Since the days of Emil Fischer, the basic principle was complemented with further insights and additions, such as the "induced fit" model proposed by Koshland [Kos58]. Rational drug design thus tries to understand and predict these processes in order to guide the drug development instead of using a purely trial-and-error based approach. This, however, turns out to be very difficult. The history of Aspirin can serve as an illustrative example for the time scales involved in understanding drug mechanisms at an atomic level: Aspirin has been marketed since 1897, but its mechanism was only elucidated in 1971 by John Vane [Van71], for which he was awarded with a Nobel prize in 1982. Even with modern rational approaches, drug design is still a very difficult, lengthy, and expensive process. Recent predictions indicate that the speed and efficiency of the development process need to improve substantially in the future for drug design to remain profitable and for innovations to continue [PMD<sup>+</sup>10]: at the current rate of drug development, the revenue generated by novel drugs is insufficient to replace the amount lost due to patent expiration [Goo08]. This clearly highlights the need for more efficient and more successful drug design techniques. Previous experience in other fields of science and industry indicates that the most promising route towards a sufficient productivity boost lies in the consequent adoption of computer based techniques.

In the life sciences, the computer has already become an invaluable tool in handling and assessing the huge amount of data that is produced. The size and diversity of the data as well as the complexity of the related questions demand sophisticated, efficient, and accurate processing techniques that often push the frontiers of research in the computer sciences. This thesis is devoted to one particularly important application of computer science methods to medical research – the so-called field of molecular modelling.

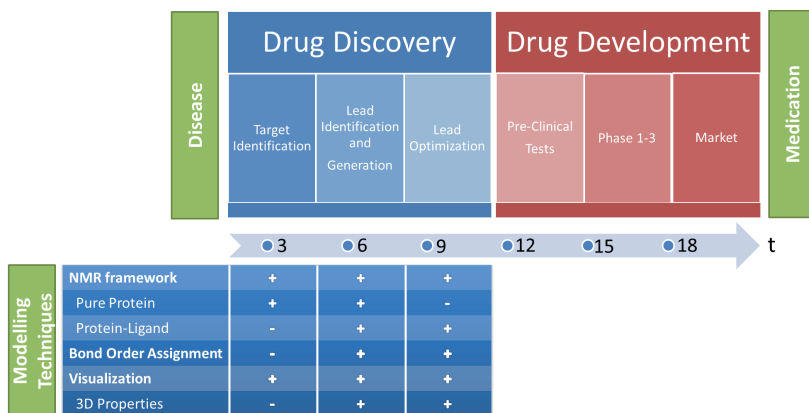


Figure 1.1.: The drug design pipeline and our molecular modelling techniques.

Molecular modelling is the art of visualizing, simulating, and editing molecular systems on a computer. Even though molecular modelling techniques have already evolved into a main cornerstone of many fields of science, such as computer aided drug design, theoretical chemistry, and materials science,

the current state of the art is still far from realizing the full potential of computer based modelling techniques.

If we compare this situation with other fields of science where computer aided techniques have had a significant impact in the past, we can see that improved molecular modelling techniques should lead to important breakthroughs. Before the advent of modern simulation techniques, designing, e.g., a new car or a new airplane required the physical generation of several prototypes that were then subjected to wind tunnel testing, crash tests, and similar experiments. The results of these were then used to optimize the design until the design goals were sufficiently met. Today, though, most of the design work happens on the computer, using sophisticated modelling and simulation packages. Usually starting from a model of a previous product, aerodynamics, crash test behaviour, and many additional properties are simulated and optimized for.

But while in computational engineering, computer aided techniques have greatly reduced and in many cases entirely replaced the need for physical experimentation in many stages of the development, molecular modelling is still often relegated to the role of a crude pre-experimental filtering technique. The main reason for this difference lies in the complexity of the underlying simulation tasks: in the engineering field, scientists are usually interested in macroscopic properties for which the underlying physics is well understood and highly accurate simulation techniques that scale well to many processor nodes have been researched extensively. In the molecular modelling field, on the other hand, we are interested in microscopic properties of the molecules under study for which we cannot make use of macroscopic averaging techniques. We thus need to describe the systems on an atomic level, leading to easily overwhelming numbers of degrees of freedom. To make matters worse, simulating the theory that describes the behaviour of molecular systems at the atomic level, quantum mechanics, are often to the most difficult fields of computational science. Finally, in biological applications, we are often required to apply the simulation techniques not only to tens or thousands of molecular systems, but to hundreds of millions.

The typical strategy to overcome these challenges is thus to replace the ab-initio simulation of atomic and molecular properties from the laws of quantum mechanics by approximate models, either based on simplified physics [Hil05], on statistical inferences [HTF09], or on a hybrid combination of both [NNZW03]. In this work, we will develop such approximation approaches for several different problems arising in structural bioinformatics.

One of the most basic and fundamental approximations of molecular physics is the classification of chemical bonds into several different types, such as single, double, triple, delocalized, or aromatic. While this classification is not perfect, in the sense that there are several cases that cannot unambiguously be classified into one of the common types, it works very well in practice, and is widely used throughout Chemistry.

In molecular modelling applications, this set of bond types is typically further reduced, and only the bond types single, double, triple, and possibly quadruple are taken into account. Each bond is thus described by an integer *bond order*, varying between one and four. These bond orders lie at the heart of all force field based molecular modelling approaches and describe a wealth of molecular phenomena very well, but also lead to several problems. Most importantly, since bond orders only approximate the true molecular situation, the assignment is not necessarily unique and several bond order combinations are needed to correctly capture the effects of, e.g., delocalized electrons. In addition, inferring bond orders for a given molecule is a difficult task and is usually guided by a number of complex rules designed by experts from chemistry. These problems render an automated bond order assignment highly challenging, but unfortunately, many input data sets for molecular modelling applications lack bond order information. Thus, automated assignment strategies are a very important component of modelling frameworks.

In this work, we develop new approaches for automated bond order assignment that are as accurate as possible, yet account for the inherent non-uniqueness. To address changing requirements in the chemical rules or individual preferences, the system allows simple modification of the underlying rule

## 1. Introduction

set, even at run time. As basis for our approach, we used an ansatz originally designed for the Generalized Amber Force Field (GAFF) [VWC<sup>+</sup>04], which we greatly extended by recasting it in the form of an exactly solvable optimization problem. Our method is to the best of our knowledge the only one that is provably correct even for input molecules with missing bond order information and incorrect coordinates. In Chapter 2, we will present both the optimization problem and several algorithms we developed for its efficient optimal solution. Furthermore, we present a framework that connects these methods with an interface for adapting the approximation rules by experts in chemistry and demonstrate on extensive evaluation data sets how our exact approaches improve on the state of the art.

Our bond order assignment strategy, which we call *BOA Constructor*, provides valid input for all traditional applications. Among many others, this includes the atom typing procedures of force fields used for energy estimation, structural optimization, and molecular dynamics simulations. Thus, BOA Constructor enabled us to implement the GAFF force field in a way that not only guarantees for the first time a provably exact typization, but also greatly improves extensibility. Other classical applications of bond order assignment that can be addressed using BOA Constructor include the detection of flexible groups or aromatic bonds as needed for QSAR applications, or computational combinatorial chemistry. In addition to these established application scenarios, we will demonstrate that reliable bond order assignment and the resulting reliable atom typization also build valuable input for the approximation of molecular quantities by statistical means.

As we will see in more detail later in this work, the type of approach that is used for approximating such properties typically differs for different kinds of molecules. In drug design, we are mainly interested in three different variants: small ligand-like molecules, proteins, and DNA or RNA strands. While each of these classes is very diverse in itself, there are several intrinsic properties that set them apart from the others.

Ligand molecules, for instance, are usually relatively small - much smaller, at least, than most proteins - but feature a very diverse chemistry. This diverse chemistry requires computationally expensive techniques for modelling, such as quantum mechanics or sophisticated force fields, which fortunately can often be afforded due to their small size. Proteins and DNA/RNA, on the other hand, are usually much larger than ligands, but are chemically very regular, since these molecules are made from a small number of well-defined building-blocks (neglecting post-translational modifications). Hence, for these molecules, the level of approximation of the physical effects is often much greater than for ligands. Combining these two fields - the diverse ligand chemistry with the more restricted protein and DNA/RNA chemistry - is a formidable task.

As we will demonstrate in this thesis, molecular descriptors based on bond- and atom types provide valuable information about the local chemical environment of atoms. This information can be used in statistical regression techniques or in a hybrid combination of such statistical approaches with the simulation of simplified physical laws. For systems composed of proteins and ligands, this hybrid approach allows to exploit the chemical regularity of proteins and at the same time include the non-protein atoms in statistical models based on bond- and atom types.

An important example of such an application, discussed in Chapter 3, is the computer-aided prediction of NMR chemical shifts, which becomes increasingly important for the resolution of biomolecular structures. One crucial attribute of NMR shifts is their strong dependence on the local chemical and spatial neighborhood which is then used to infer structural information about the system at hand. To predict chemical shifts from a putative structure, efficient semi-classical approximations of the physics of NMR are usually combined with statistical regression techniques.

To address NMR chemical shift prediction, we first build a fully automated pipeline called *NightShift* for the generation of training and test data sets and the training of hybrid shift prediction models. NightShift can automatically adapt to the availability of new experimental data. Thus, the tedious manual work of generating data sets and new models for chemical shift prediction that used to take months to years will be greatly simplified and can be reduced to calling a single shell script to re-train

the pipeline, or write a few lines of code to implement novel descriptors.

Using NightShift, we automatically generate models for protein shift prediction that outperform established manually generated approaches. The best of these models, which we call *Spinster*, uses a hybrid combination of simplified physical laws with a random forest model. Combining NightShift and Spinster with our work on bond order assignment then finally allows us to address an important unsolved problem in molecular modelling: the prediction of the influence of non-protein atoms on protein chemical shifts. Apart from computationally extremely expensive quantum chemical models, all currently available methods for shift prediction fail to include the influence such non-protein atoms, such as ligand atoms, entirely. In our new model, which we call *Liops*, we used bond types computed with BOA Constructor and the resulting atom types as features for regression techniques for NMR shift prediction. While Liops should currently be seen as a proof-of-concept rather than a final model, it demonstrates that features based on bond orders and atom types carry useful information for difficult prediction tasks.

Since the current state of automated modelling still makes manual preparation, assessment, and correction indispensable, we further investigated techniques to visualize and manipulate biomolecular structures and their properties. As shown in Chapter 4, this led to a number of developments, in particular in the field of real time ray tracing of molecular systems, which are tangential to this work and will only be covered briefly. In the course of these projects, we found that modern visualization techniques can also help in computing molecular properties in a very general, albeit very efficient way. A first application, the estimation of molecular areas, volumes, and cavities, will be introduced briefly in Section 4.5.

All the developments described above can benefit computer aided drug design, as was our main goal in this thesis. In Fig. 1.1, we show how our contributions map to the different stages of the drug discovery pipeline.

Creating such techniques for molecular modelling requires a powerful framework for solving routine recurring tasks efficiently and correctly, such as reading or writing common file formats, representing molecules in efficient data structures, or applying established molecular modelling techniques. Our work is thus based on the Biochemical Algorithms Library (BALL) [HDR<sup>+</sup>10], which provides hundreds of pre-coded solutions to common modelling problems. In the course of this thesis, we greatly extended BALL's functionality to support our research. In addition, all techniques developed in this thesis have been integrated into BALL, allowing automatic dissemination to thousands of users world wide.





## 2. Optimal Bond Order Assignment

### 2.1. Introduction

Many algorithms in Computational Biology and Chemistry are concerned with the analysis or manipulation of molecular structures. Often, mere knowledge about the connectivity of the molecule is insufficient and has to be supplemented with information about the bond orders: imagine, e.g., the detection of flexible groups in a molecule, or explicit (non-aromatic) bond orders needed for atom typization in force fields. Unfortunately, this information is often neglected in molecular databases and file formats. Even important molecular databases, such as the Protein Data Bank (PDB) [BWF<sup>+</sup>00, BHN03] and the Cambridge Structural Database [All02], are known to contain erroneous data for connectivity and bond order information [Lab05]. Hence, we cannot rely on its availability, explaining the need for accurate and efficient algorithms for bond order assignment.

For proteins and nucleic acids, bond orders can be easily deduced due to their building block nature, but this does not hold for other kinds of molecules such as ligands. The problem is made much worse by the fact that quite often, the bond order assignment for a given molecule is not unique, even when neglecting symmetries in the molecule. The chemical reasons for this effect are complex and out of scope of this work. Here, we just want to state that the concept of integer bond orders is only an approximation to a full quantum chemical treatment and cannot explain all effects occurring in molecules. Important examples are aromatic or delocalized bonds, leading to different resonance structures (an example is shown in Fig. 2.1). In addition, formal charges are often not contained in the input files, but atoms carrying a formal charge will obviously show a different bonding pattern.

One school of thought tries to overcome these problems and errors through hand-curation, which clearly provides the highest reliability. On the other hand, manual curation does not scale well to large numbers of molecules, and it does not help in situations where modifications are programmatically applied to the molecules, e.g., in computational combinatorial chemistry, where small molecules are automatically built up from certain templates by connecting molecular fragments to pre-defined positions.

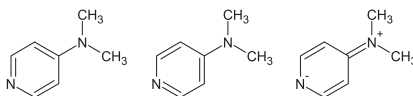










Figure 2.1.: Different resonance structures of 4-(N,N-dimethylamino)pyridine. A bond order assignment program should optimally be able to compute all of these configurations.

An illustrative example demonstrating the influence of different bond orders on the biological activity is given by the Fibromyalgia (FMS) inhibitors, whose inhibition potency has been studied in [MWC<sup>+</sup>08]. Selected investigated inhibitors are shown in Fig. 2.2. For example, cases 18g and 18h only differ in a single bond order, but yield significantly different IC<sub>50</sub> values. A comparison of cases 13a and 13g even shows a difference of one order of magnitude.

## 2. Optimal Bond Order Assignment



Compound	A	IC <sub>50</sub> <sup>a</sup> (μM)
13a		0.63
13b		0.89
13c		0.47
13d		0.94
13e		0.060
13f		0.018
13g		0.054








Compound	B	IC <sub>50</sub> <sup>a</sup> (μM)
18a		0.01
18b		0.038
18c		0.002
18d		0.015
18f		0.005
18g		0.0008
18h		0.0003

Figure 2.2.: Influence of bond orders on the IC<sub>50</sub> values of FMS inhibitors as presented in the work of Meegalla and coworkers [MWC<sup>+</sup>08].

Further examples that demonstrate the influence of bond orders on relevant properties are given in Fig. 2.5, which exemplarily shows two cases where wrong bond orders lead to a wrong interpretation of two important molecular properties: flexibility and aromaticity.

In this chapter, we describe a new combinatorial optimization approach for bond order assignment, which we call *BOA Constructor*. BOA Constructor was designed as the new bond order component for our BALL library (c.f. Chapter 4). As a general technique, it does not make any a-priori assumptions about the quality of the input data. From a theoretical point of view (see Section 2.2), there are several shades of the bond order assignment problem, depending on the characteristics of the input. In essence, there are two important types of information that may or may not be reliable: the *molecular topology*, which we define as the atoms of a molecule and the atom pairs that are covalently bonded to each other, as well as the *atomic coordinates*. If the topology is incorrect, only a coordinate based approach can usually succeed, and hence, reliable atomic positions are required in these cases. However, in most use-cases in practice, the molecular topology is assumed as given, but atomic coordinates may be unreliable (c.f. Fig. 2.3). In these cases, a connectivity based approach is much more stable than a position based one. Consequently, our bond order assigner is built upon an established connectivity based approach by Wang and collaborators [WWKC06]. On the other hand, to support cases where the molecular topology is incorrect, incomplete, or missing entirely, or where the user knows the degree of reliability of input coordinates, we extended the original approach to allow the user to continuously switch between a purely connectivity and a purely position based approach.

To ensure high computational efficiency and simple extensibility, we separated the theoretical formulation of bond order assignment as a combinatorial optimization problem from the solution strategies used. This allowed us to implement and test different solvers, each with their own advantages and disadvantages. Three of these solvers, an A-Star (A\*) algorithm, an Integer Linear Program (ILP),

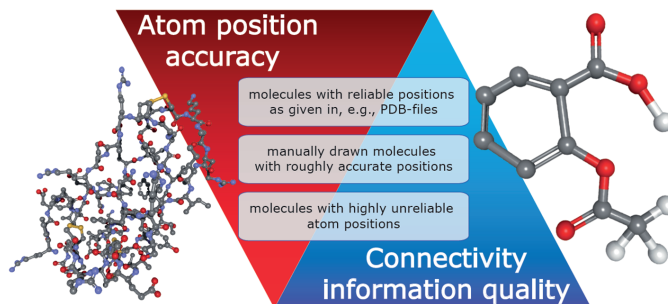


Figure 2.3.: Classes of input molecules for the bond order assignment problem.

and a Fixed-Parameter Tractability (FPT) approach, solve the problem to provable global optimality. Two additional heuristic optimization techniques, a K-Greedy and a Branch-and-Bound strategy, usually provide good approximations. As a great advantage compared to previous approaches, our exact solvers can not only enumerate all co-optimal solutions, but also sub-optimal ones, if so desired.

Finally, the separation between problem formulation and solution strategy allows to extend the approach in several directions, such as the inclusion of structural information, the addition of missing hydrogens, or even the inference of the molecular topology.

In Section 2.3 and Section 2.4, we present former approaches and research on automated bond order assignment. In Section 2.5 and Section 2.6, we describe our aims and our formulation of the bond order assignment problem and discuss provably exact algorithms to solve it. We then turn to possible extensions to our algorithms with respect to including structural information, adding missing hydrogens (c.f. Section 2.7), and a new penalty table (c.f. Section 2.8). Finally, Section 2.9 presents the results of our approaches on several evaluation data sets. While this chapter focusses mostly on the conceptual and algorithmic aspects of bond order assignment, details of the implementation of BOA constructor will be described in Section 4.2.

Most of the results presented in this section have been published in the Proceedings of GCB 2009 [DRLH09] as well as in a Bioinformatics publication [DRB<sup>+</sup>11].

## 2.2. Problem Definition

The concept of bond orders is only a rough, yet widely accepted, approximation to the quantum mechanical concept of a chemical bond. The idea behind this approach is to count the number of electrons shared between the two bonded atoms. In this approximation, the bond order is assumed to be an integer number between, typically, 1 and 4. But with this approximation, it is not possible to explain all effects occurring in molecules. An extension thus also allows cases where electrons are shared by more than two atoms, e.g., so-called delocalized systems like carboxyl groups, or aromatic rings. But here, electrons cannot be uniquely mapped to individual bonds. Hence, most modelling approaches use the approximation as integers. But this immediately poses the problem of assigning such approximated bond orders to a molecule.

The *connectivity based bond order assignment problem* is defined as follows: given a molecule with topology information, i.e., the connection between atoms in the molecule, compute its most probable bond order assignment. In the *coordinate or position based bond order assignment problem*, we are instead given the atomic coordinates and want to infer the bond orders from these. Due to the

## 2. Optimal Bond Order Assignment

individual chemical properties of atoms that are part of a molecule, not all theoretical possible bond order assignments are valid. Also, the valid bond order assignment is not necessarily unique, e.g. in aromatic ring systems, c.f. Fig. 2.1 and Fig. 2.5.

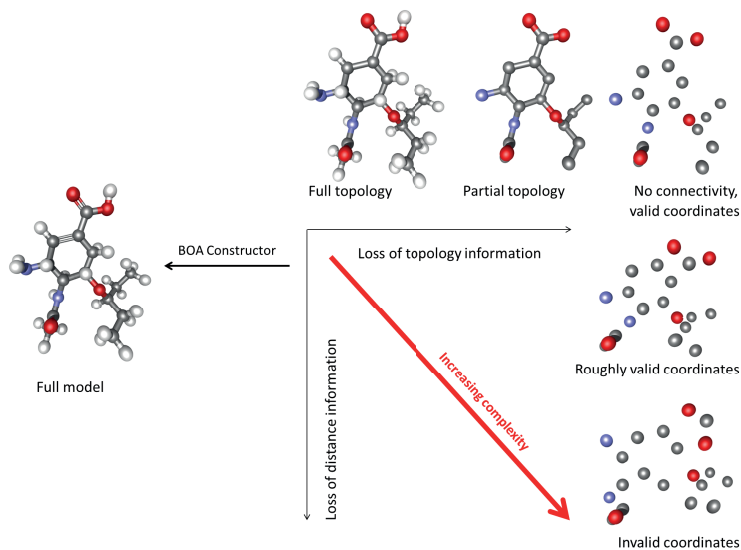


Figure 2.4.: Classes of the bond order assignment problem. Crucial pieces of information are topology (atoms and bonds) and atomic distances.

Unfortunately, several additional shades of the bond order assignment problem exist as shown in Fig. 2.4. This is due to the different possible levels of given connectivity information and reliability of the atomic positions for general molecular input. A number of former approaches for solving the bond order assignment problem thus further restrict the problem to require the correctness of given atomic positions.

For the general case, we can distinguish three major classes of input molecules:

1. molecules with reliable atom coordinates, bond lengths, and angles.
2. manually 'drawn' molecules with roughly accurate positions (e.g. 2D sketches).
3. molecules with highly unreliable atom positions.

For the first case, heuristic classifiers based on order dependent statistics about bond lengths and angles are known to work well in practice and are wide-spread (see, e.g., [ZCW07]). But extending this technique to molecules from the second class produces highly unreliable results, and application to the latter class is obviously bound to fail. Alternative approaches have thus focused on inferring bond orders from connectivity information alone. In cases where atom positions are known to be sufficiently accurate, though, position based approaches typically yield more accurate results. Consider, e.g., two

molecules differing only by their bond orders. A connectivity based approach can never distinguish between the two, while a position based approach employs the different lengths and angles induced by the different bond orders.

Unfortunately, given an arbitrary molecule from an arbitrary source, there is no way to reliably classify into which category it falls. In addition, in many application scenarios, the information how many hydrogen atoms are bonded to an atom is missing as well; common databases often lack this information since it is not always experimentally accessible.

In the worst case, we lack connectivity as well as distance information (see Fig. 2.4). In this case, the number of possible connections to create a single molecule for the given set of atoms is already exponential in the number of atoms and we cannot easily decide between the different solutions. If, on the other hand, the distance information is fairly correct, which is a reasonable assumption for typical molecular data as discussed above, connectivity information can be deduced. Thus, in this work we assume to be given at least the full molecular topology. Please note that we consider the information how many hydrogens are connected to an atom as connectivity information as well, but we will show how this additional problem can be elegantly addressed at additional computational cost in our approaches later. Finally, we will show how the approach can additionally use positional information if this is available.

## 2.3. Former Approaches

The problem of assigning bond orders has been addressed by different types of approaches since the advent of large molecular ligand databases made manual data curation infeasible. Very different strategies have been applied to derive bond order information in combination with a broad range of other structural information, such as atomic connectivity [BH92, ZCW07], hybridization states [HRB97], ionization states and charges [Lab05], and atom typing [WWKC06]. In the following, we briefly describe the most important of these approaches.

One of the first approaches for automated bond order assignment was developed by Baber and Hodgkin [BH92]. Based on reference bond lengths and valence angles, these authors assigned initial bond orders and coordination states (linear, trigonal, tetrahedral, and octahedral), each weighted with a confidence value. Conflicts between the atom's valence and the connectivity are solved iteratively in a greedy manner by adjusting the assignment with smallest confidence value.

The BALI program by Hendlich and coworkers [HRB97] extends this ansatz by additionally considering hybridization, aromaticity, and information that is independent of the geometric data: functional groups. BALI proceeds in a stepwise manner, first deriving bond orders based on connectivity information, then recognizing functional groups, followed by comparing the bonds against known bond length and angles, and recognizing aromatic rings. Finally, a heuristic conflict resolution is applied.

A completely different solution is proposed by Labute [Lab05], who represents the bond order assignment problem as a Maximum Graph Matching. Sophisticated geometric tests are applied to generate probable bond orders that are weighted based on statistics deduced from a large collection of organic molecules. The solution of the resulting combinatorial optimization problem leads to a consistent bond order assignment. However, this approach still strongly relies on the availability of correct three-dimensional atom positions.

The program I-interpret [ZCW07] focuses on correct file format conversion. Since in some file formats, the bond order information is missing, the authors instead rely on atomic coordinates to solve the bond order assignment problem. In a stepwise procedure, connectivity, hybridization states, functional groups, and aromatic rings are recognized, yielding an initial atom and bond type assignment. All remaining unsolved bonds are typed based on standard bond lengths and dihedral angles before, finally, a priority based conflict resolver is applied.

Another approach that formulates the bond order assignment problem as an abstract optimization

## 2. Optimal Bond Order Assignment

problem is presented in the work of Froeyen and Herdewijn [FH05]. The general idea here is to find assignments that satisfy the octett rule [Lan19], and hence represent a valid Lewis structure [Lew16]. Based on atom types and connectivity information, the octett rules for all atoms are formulated and combined into a linear programming problem that can be solved by an external solver. This approach follows a very elegant idea and can be used in principle without requiring three-dimensional coordinates. However, it still suffers from a number of drawbacks. First, it can only generate a single valid Lewis structure, even though in non-trivial cases, many valid solutions exist. While this could be changed in principle – even though at large computational expense – the problem is fundamentally much worse: valid Lewis structures fulfilling the octett rule are a nearly<sup>1</sup> necessary, but not a sufficient condition for a chemically sound bond order assignment. To illustrate this, imagine a six-ring of carbon atoms without attached hydrogens. Setting all ring bonds to be double bonds leads to a valid Lewis structure that fulfills the octett rule, but is chemically forbidden: such a system of double bonds would require bond angles of 180° each, which is clearly inconsistent with a ring topology. Hence, fulfilling the octett rule is insufficient, since it cannot distinguish between different isomers that are unequally likely due to their chemical environment.

In summary, all of the approaches discussed before suffer from several drawbacks: to the best of our knowledge none of the alternative approaches offers enumeration of all optimal or even sub-optimal bond order assignments for a given topology, i.e. equally likely or only slightly more unlikely solutions. The key algorithms often contain hard-coded heuristics for fragments and functional groups that cannot be easily changed or updated. In addition, the source code is often not available, while the heuristics are not described in the publications, rendering re-implementation impossible. Furthermore, most of these approaches strongly rely on the correctness of structural data, which is usually not given and thus cannot be assumed for general applications.

### 2.4. The Antechamber Approach

Instead of relying on the correctness of the atom coordinates as many of the previously presented approaches do, our work extends a connectivity based approach proposed by Wang et al. [WWKC06] that uses the flexible and extensible notion of *molecular penalty scores*: each atom in the molecule is assigned to one of several possible classes based on its element, degree of connectivity, and element and degree of its neighbours. For each atom, the sum over the degree of its bonds is defined as its *valence*, and for each class, each valence is assigned an individual *atomic penalty score*. The sum over all atomic penalty scores defines the *total penalty* of the molecule; minimization of this score is supposed to yield the most realistic assignment of bond orders. The approach has been implemented in the popular Antechamber package [WWKC01, Wan10]: Antechamber, now a part of the AmberTools suite, is a program used to prepare input structures for use with the Amber molecular mechanics package. In particular, it is used to generate input for the GAFF force field [WWC<sup>+</sup>04], a generalized version of Amber.

In Antechamber's bond typing approach, a chemically motivated, expert generated penalty function is used to score bond order assignments. This function is then heuristically optimized. This procedure works well for small molecules without, e.g., convoluted ring structures, where it usually finds one of the optimal assignments. Unfortunately, application to larger or more complex molecules suffers from two drawbacks: the score of the resulting assignment is not guaranteed to be optimal and the algorithm provides only one solution while there can be more than one assignment with optimal score. Fig. 2.5 exemplarily shows two cases where these two drawbacks may lead to a wrong interpretation of two molecular properties: flexibility and aromaticity. The top structures represent valid bond order combinations for the 2-amino-2,3-dihydroquinazolin-4(1H)-iminium molecule, yet rotation of an NH<sub>2</sub>

<sup>1</sup> The problem is aggravated by several exceptions from the octett rule which we do not discuss here. In this sense, it is not even a necessary condition.

group is restricted due to double bonds. The lower structures show valid bond order combinations for the 4-azido-1,2,5-thiadiazol-3-amine molecule, yet only one structure meets the AM1-BCC aromaticity criterion [JJB02]. Finally, the heuristic nature of the approach leads to a complex implementation. More formally, the bond order assignment problem as addressed in [WWKC06] is defined as finding the most probable consistent bond order assignment for a given molecule by minimizing a total penalty score  $tps$ , where each atom is assigned an atomic valence  $av$  that is defined as the sum over all bond orders  $bo$  of all bonds connected to the atom under consideration:

$$av = \sum_{i=1}^{con} bo_i.$$

Here,  $con$  denotes the number of bonded atoms. The distance of the calculated  $av$  to the atom's most desirable valence value is measured by the atomic penalty score  $aps$ . The possible valences of an atom and the corresponding distance penalty scores are stored in a penalty table that uses a rule based atom type classification as derived in [WWKC06]. The sum over all atomic penalty scores of a molecule now yields the total penalty score

$$tps = \sum_{i=1}^n aps_i$$

where  $n$  denotes the number of atoms. The smaller the  $tps$  of a given bond order assignment, the more reasonable it is. Finding the most probable consistent bond order assignment for a given molecule can thus be addressed by minimizing the total penalty score. Recently, Böcker and coworkers showed that exact minimization of the  $tps$  is NP-hard [BBST09]. Thus, in [WWKC06], minimization then proceeds in a heuristic and greedy manner.

### 2.4.1. The Heuristic Trial-and-Error Approach of Antechamber

The approach taken by Antechamber tries to solve the problem by separating the assignment of atomic valences from the assignment of bond orders. The rationale behind this idea is as follows: neglecting feasibility, the assignment of atomic valences can be done independently for each atom, and hence is trivial. From this point on, the problem reduces to finding a valid bond order assignment that leads to the chosen atomic valences. From a computational perspective, though, not much is gained: there are exponentially many atomic valence assignments of the molecule that have to be tested, and each bond order assignment for each valence state is still a hard problem as soon as ring systems are present in the molecule. However, this separation lends itself to being implemented a simple heuristic approach: first, for a given atomic valence assignment, all bond orders that can be immediately decided are directly inferred. Those that are still undetermined after this step are then addressed by a so-called trial-and-error test discussed below.

Since it is not immediately clear if a given atomic valence assignment will lead to a feasible bond order assignment, the program typically has to test a large number of these, in order of ascending total penalty score. However, due to the combinatorial explosion of atomic valence assignments, the program generates a list of such assignments sorted with respect to their  $tps$  and pruned after a fixed number of entries. With default settings, Antechamber tests 2000 assignments for feasibility.

The pseudo code in Alg. 1 describes how bond orders are assigned for a given atomic valence combination.

Unfortunately, the algorithm described above is not guaranteed to return an optimal solution. First, the restriction to a relatively small number (2000 in the default implementation) of atomic valence assignments leads to failures as soon as complex ring structures are involved. Second, the trial-and-error test as described above lacks backtracking capabilities. While these can be integrated into the

## 2. Optimal Bond Order Assignment

---

**Algorithm 1** Bond order assignment for given atomic valences of molecule  $M$

---

```
1:
2: def Antechamber( $M$ ):
3: while  $M$  contains unassigned bonds: do
4:   assignTrivialBonds( $M$ )
5:   if  $M$  contains unassigned bonds: then
6:     result  $\leftarrow$  trialAndError( $M$ )
7:     if result == "infeasible": then
8:       return "infeasible"
9:     end if
10:  end if
11: end while
12:
13: def trialAndError( $M$ ):
14:  $b \leftarrow$  first unassigned bond in  $M$ 
15: for all  $o \in \{1, 2, 3\}$ : do
16:   setOrder( $b, o$ )
17:   if assignTrivialBonds( $M$ ) != "infeasible": then
18:     return "feasible"
19:   end if
20: end for
21: return "infeasible"
22:
23: def assignTrivialBonds( $M$ ):
24: for all atoms  $a$  in  $M$ : do
25:   if ( $con(a) == 0$  and  $av(a) \neq 0$ ) or ( $con(a) \neq 0$  and  $av(a) == 0$ ): then
26:     return "infeasible"
27:   end if
28:   for all unassigned bonds  $b$  of  $a$ : do
29:     if  $con(a) == av(a)$ : then
30:       setOrder( $b, 1$ )
31:     else if  $con(a) == 1$ : then
32:       setOrder( $b, av(a)$ )
33:     end if
34:   end for
35: end for
36:
37: def setOrder( $b, o$ ):
38:  $order(b) \leftarrow o$ 
39: ( $a_1, a_2$ )  $\leftarrow$  atoms( $b$ )
40: for all  $i \in \{1, 2\}$ : do
41:    $av(a_i) \leftarrow av(a_i) - o$ 
42:    $con(a_i) \leftarrow con(a_i) - 1$ 
43: end for
```

---



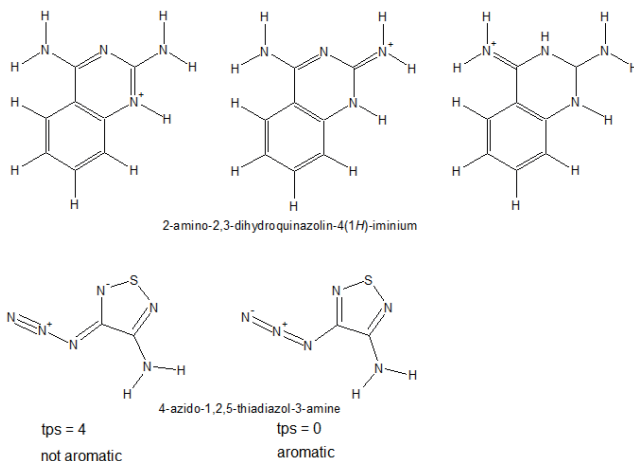


Figure 2.5.: **Top:** Different co-optimal bond order assignments. In the left structure, both  $\text{NH}_2$  groups are connected to the molecule via a single bond and are thus freely rotatable. In the middle and right structure, one  $\text{NH}_2$  group is connected via a double bond. **Bottom:** Heuristic and optimal bond order assignments. The left structure is the solution provided by Antechamber with a *tps* of 4. Its 5-ring does not fulfill the aromaticity criterion of AM1-BCC [JJB02]. The right structure is the solution computed with our exact solvers. Its *tps* is 0 and the 5-ring meets the AM1-BCC aromaticity criterion.

algorithm, the step increases the worst case run time exponentially. Table 2.1 shows cases where this procedure leads to sub-optimal solutions.

## 2.5. Aims of our Work

We found that all of the problems described in Section 2.4 can be circumvented by re-formulating the total penalty score minimization as a combinatorial optimization problem.

To this end, we have expressed the combinatorial optimization problem by a suitable objective function and designed heuristics to solve the assignment problem by an A\* approach. In addition, we designed an Integer Linear Program (ILP) for bond order assignment and collaborated closely with the group of Prof. Dr. Sebastian Böcker in their efforts to develop a Fixed-Parameter Tractability (FPT) solution. We then continued to extend the basic algorithms in several different directions, such as the addition of missing hydrogens, the inclusion of structural information into the otherwise purely connectivity based approach and a new penalty table.

## 2.6. Solution Schemes

Given the new formal definition of the bond order assignment problem, we now present our different solution schemes, each with its own set of advantages and shortcomings. In addition to the three exact solvers, we have also tested different heuristic strategies. To focus the presentation on the most important results, discussion of these heuristic approaches has been deferred to Appendix A.3.

## 2. Optimal Bond Order Assignment

### 2.6.1. An A\* Approach

In order to be able to efficiently enumerate *all* feasible solutions – optimal and sub-optimal ones alike – we formulated the bond order assignment problem as an A\* search algorithm. This allows enumeration of all assignments in order of increasing penalty and hence, for instance, to compute the assignments of all solutions for a given molecule up to a user defined penalty threshold.

As a combinatorial optimization problem, the bond order assignment problem can be represented by a tree, where each layer stands for one of the decisions that have to be made. In our case, the tree has  $k$  layers, where  $k$  is the number of bonds that have to be assigned. A node at layer  $i$  has  $\mu$  children, where  $\mu$  is the number of possible bond orders, typically 3, and each edge is labeled with its corresponding order. Hence, by tracing the path from the root to a node  $w$  at layer  $i$ , we can determine the values of the first  $i$  bonds in this particular partial assignment represented by the node  $w$ . Thus, the root node corresponds to a completely unassigned molecule with only unknown bond orders, while the leaf nodes correspond to complete bond order assignments. Adding a child node only if the resulting valence state is valid guarantees the leaf nodes to correspond to feasible bond order combinations. In order to further discriminate between the different combinations, each leaf is assigned its total penalty score describing the likelihood of the bond order assignment due to the heuristic penalty function.

Visiting all nodes in the tree, the optimal bond order assignment can be found in a brute-force manner with exponential running time. A heuristically good (but not necessarily optimal) result can be found in linear time in a greedy manner if all intermediate nodes are assigned the atomic penalty score of the partial bond order assignment they represent. Greedily expanding the locally optimal solution will then yield the heuristic assignment. An exact solution can be found in *worst-case* exponential, but much better *expected* running time, if at each intermediate node, more information is provided. This leads to the popular A\*-search algorithm [HNR68], which employs a search heuristic to guide the algorithm in descending the tree.

More formally, the algorithm associates with each node  $w$  a function  $f(w) = g^*(w) + h^*(w)$  where  $g^*(w)$  describes the score corresponding to the decisions already made and  $h^*(w)$  is the so-called search heuristic. For the purposes of the A\*-search algorithm, the search heuristic must be an admissible estimate of the score of the best leaf that can be reached starting from node  $w$  and descending further down the tree. Here, admissible means that it needs to be 'optimistic': for all nodes  $w$ , the estimated cost  $h^*(w)$  may never be greater than the lowest real cost to reach a goal node when starting at  $w$ . Given the additional information provided by  $h^*$ , the A\*-search algorithm always expands one of the nodes with the most promising score, ensuring that the first leaf reached is optimal. Roughly speaking, if the algorithm would visit a leaf with worse score first, the search-heuristic would have overestimated the penalty of the real optimal solution, which an admissible heuristic never does.

Alg. 2 shows the general A\*-search algorithm, where  $r$  denotes the root node,  $f(w)$  the search heuristic of a node  $w$ , which returns the sum of the score corresponding to the partial solution represented by  $w$  and the admissible heuristic estimate of the score corresponding to the upcoming decisions.

In practice, the A\*-search algorithm is typically implemented with a priority queue  $\mathcal{PQ}$ , so that line 3 is in  $\mathcal{O}(1)$ . As long as  $\mathcal{PQ}$  is not empty, the algorithm can compute further solutions in the order of optimality. When replacing line 6 with:

```
print  $\bar{w}$ 
```

the algorithm will return all solutions in the order of increasing score.

Alg. 3 shows the adaption of the A\*-search algorithm to the bond order assignment problem. To this end, we represent a molecule  $M$  by a set of its bonds, which are visited in an arbitrary but fixed order. Let  $r$  denote an artificial root node, corresponding to the empty assignment. During the algorithm, bonds not denoted as free will keep their original bond orders. As long as the priority queue is not empty, the algorithm computes further solutions in the order of optimality. When replacing line 10 with:

```
store bond order configuration as denoted in  $\bar{w}$ 
```

**Algorithm 2** A\*-search algorithm

---

```

1:  $\mathcal{PQ} \leftarrow \{r\}$ 
2: while  $\mathcal{PQ} \neq \emptyset$  do
3:    $\bar{w} \leftarrow \arg \min_{w \in \mathcal{PQ}} f(w)$ 
4:    $\mathcal{PQ} \leftarrow \mathcal{PQ} \setminus \{\bar{w}\}$ 
5:   if  $\bar{w}$  is leaf then
6:     return  $\bar{w}$ 
7:   else
8:     for all  $c \in \text{children}(\bar{w})$ : do
9:        $\mathcal{PQ} \leftarrow \mathcal{PQ} \cup \{c\}$ 
10:    end for
11:  end if
12: end while

```

---

the algorithm will return all bond order configurations in order of increasing penalty.

Obviously, the order in which the bonds are considered during the A\*-search has a strong influence upon the number of nodes that have to be expanded before finding the first solution. Considering bonds in the order of decreasing certainty about the most probable order will keep the expanded tree slim, whereas considering bonds in the order of increasing bond order certainty leads to a very broad tree.

A schematic sketch of the A\*-algorithm for bond order assignment is given in Fig. 2.6. To map the bond order assignment problem formally to an A\*-search, we need further notations that are adapted to the partial bond order assignments corresponding to each node  $w$  in the search tree.

We denote the set of all assigned bonds in the node  $w$  by  $W(B)$ , the assigned bonds connected to atom  $a$  in node  $w$  by  $W(a)$ , and the set of atoms for which all bonds are already assigned a bond order by  $K$ . We further define the valence of atom  $a$  in node  $w$  as

$$v_w(a) := \sum_{b \in W(a)} bo(b).$$

Then, the functions  $g^*$  and  $h^*$  can be defined as:

$$g^* = \sum_{a \in K} P(a, v_w(a)) \quad (2.1)$$

$$h^* = \sum_{a \in A \setminus K} \min_{i \in V(a)} \{P(a, i)\} \quad (2.2)$$

where  $V(a) \subset \mathbb{N}$  contains the possible valences of atom  $a$  according to the penalty table  $P$ .

The function  $g^*$  sums the atomic penalties of all completely assigned atoms in the partial bond order assignment represented by node  $w$ , whereas  $h^*$  considers all atoms with bonds of unassigned bond order. For the atoms in this set, we compute the minimal atomic penalty possible under the current partial assignment independently of the other atoms in the set: each atom can choose its preferred value for each unassigned bond without considering overall consistency. Obviously,  $h^*$  is optimistic.

Indeed, the search heuristic given in (2.2) is far too optimistic and can be tightened significantly. Thus, we define the bond order of an assigned bond by  $bo(b)$ . A partial bond order assignment induces a simple lower bound  $v_w(a)$  for the valence of atom  $a$ .

## 2. Optimal Bond Order Assignment

---

**Algorithm 3** A\*-bond order algorithm (molecule  $M$  with  $n$  bonds)

---

```

1:
2: def A*-BOA( $M$ ):
3:  $\mathcal{PQ} \leftarrow \{r\}$ 
4:  $b_1 :=$  first free bond in  $M$ 
5: for all bond orders  $i$  do
6:    $\mathcal{PQ} \leftarrow \mathcal{PQ} \cup \text{bondOrders}(\{b_1 \leftarrow i\})$ 
7: end for
8: while  $\mathcal{PQ} \neq \emptyset$  do
9:    $\bar{w} \leftarrow \arg \min_{w \in \mathcal{PQ}} f(w)$ 
10:   $\mathcal{PQ} \leftarrow \mathcal{PQ} \setminus \{\bar{w}\}$ 
11:  if  $\bar{w}$  is leaf then
12:    assign bond orders as denoted in  $\bar{w}$ 
13:  else
14:     $b_x :=$  is next free bond in  $M$ 
15:    for all bond orders  $i$  do
16:       $\mathcal{PQ} \leftarrow \mathcal{PQ} \cup \text{bondOrders}(\bar{w}, \{b_x \leftarrow i\})$ 
17:    end for
18:  end if
19: end while
20:
21: def bondOrders( $\bar{w}, t$ ):
22:  $\mathcal{O} \in \{0, \dots, \mu\}^n$ 
23: for all  $i \in \{1, \dots, n\}$  do
24:    $\mathcal{O}_i \leftarrow \begin{cases} 0 & \text{bond } i \text{ unassigned in } \bar{w} \text{ and } t \\ k & \text{bond } i \text{ assigned order } k \text{ in } \bar{w} \text{ or } t \end{cases}$ 
25: end for
26: return  $\mathcal{O}$ 

```

---

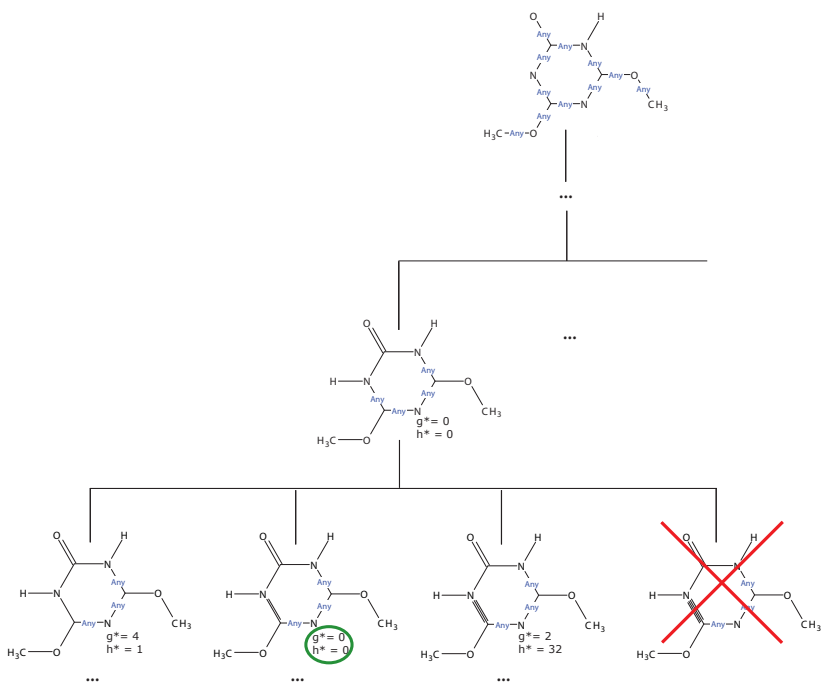


Figure 2.6.: Schematic sketch of the A\*-bond order algorithm.

## 2. Optimal Bond Order Assignment

Assuming at least a single bond for each unassigned bond of atom  $a$ , a tighter lower bound for the valence is given by

$$lo(a) := v_w(a) + \sum_{b \in B(a) \setminus W(a)} 1 = v_w(a) + |B(a) \setminus W(a)|$$

Thus, we can formulate a tighter search heuristic by

$$g^* = \sum_{a \in K} P(a, v_w(a)) \quad (2.3)$$

$$h^* = \sum_{a \in A \setminus K} \min_{lo(a) \leq i \leq \mathcal{V}_{\max}(a)} \{P(a, i)\} \quad (2.4)$$

where  $\mathcal{V}_{\max}(a)$  denotes the maximal allowed valence of  $a$  according to penalty table  $P$ .

An even tighter version of the search heuristic would also take the already assigned bond orders of the neighbouring atoms of  $a$  in  $w$  into account. The maximum order of an unassigned bond with respect to atom  $a$  is given by

$$t(a) := \mathcal{V}_{\max}(a) - lo(a) + 1$$

Denoting by  $a_1, a_2$  the atoms connected by an unassigned bond  $b$ , its maximum bond order equals

$$bo_{max}(b) := \min\{t(a_1), t(a_2)\},$$

yielding an upper bound for the atomic valence of an atom  $a$

$$up(a) := \min \left\{ \mathcal{V}_{\max}(a), v_w(a) + \sum_{b \in B(a) \setminus W(a)} bo_{max}(b) \right\}$$

Thus, a tighter version of the search heuristic is given by:

$$h^* = \sum_{a \in A \setminus K} \min_{lo(a) \leq i \leq up(a)} \{P(a, i)\} \quad (2.5)$$

We implemented all three heuristics in our code, where we default to the last since it is the tightest of the three. A detailed comparison of the performance of the different heuristics can be found in Appendix A.5.

### 2.6.2. An Integer Linear Program Approach

While the tree based approaches described before are very flexible, simple to implement and to extend (as we will see in Section 2.7) the problem can be solved even more efficiently by using Integer Linear Programming (ILP) techniques [PS98].

The key idea of Integer Linear Programming is to describe and solve an optimization problem by integer variables and linear relationships between those variables. These relationships take the form of equality or inequality constraints which the feasible solutions to the problem have to fulfill. Due to the linearity, the function to be optimized in an ILP can be written as  $c^T x + \beta$  where  $c \in \mathbb{R}^n$  denotes a vector of coefficients which is called the "cost vector" and  $\beta \in \mathbb{R}$  is an additive constant. This linear construct is called the *objective function*. The linear relationships between the variables are then modelled by a set of linear equality or inequality constraints.

Two particular forms of ILPs – the *canonical* and the *standard* form – are important in theory and practice, and it can be shown that each ILP can be cast into either of them. In the canonical case,

which was introduced by Danzig, the ILP takes on the following form:

$$\left. \begin{array}{l} \min_x \quad c^T x \\ \text{s.t.} \quad Ax \geq b \\ \quad \quad x \geq 0 \\ \quad \quad x \in \mathbb{Z}^n \end{array} \right\} \quad (2.6)$$

where  $A \in \mathbb{R}^{m \times n}$  is a matrix of coefficients and  $b \in \mathbb{R}^m$  a vector, while in the standard form, the ILP looks as follows

$$\left. \begin{array}{l} \min_x \quad c^T x \\ \text{s.t.} \quad Ax = b \\ \quad \quad x \geq 0 \\ \quad \quad x \in \mathbb{Z}^n \end{array} \right\} \quad (2.7)$$

To convert one form into the other, or to convert an arbitrary ILP into standard or canonical form, a combination of several standard transformations has to be performed. For instance, switching between minimization and maximization, or 'greater-than' and 'less-than' can be achieved through a simple change of sign. Equality and inequality constraints can be converted by the use of so-called 'slack variables', and the positivity constraint on  $x$  can be avoided by choosing from a set of new variables which can be either positive or negative only.

An intuitive picture for solving ILPs use a geometric perspective: the variables  $x$  span a multi-dimensional space. The objective function represents all valid combinations of the variables and defines a set of planes within the full domain of the variables. Each constraint given in  $A$  defines a half space, and thus further limits the variable space by a plane. If the ILP is feasible, i.e., if it is possible to fulfill all constraints simultaneously, the combination of all constraints builds a convex polyhedron.

For a linear program without the integer constraint, it is easy to see that an optimal solution can always be found at a "best intersection" of the constraint polyhedron and the set of objective functions: from any point inside the polyhedron, we can always decrease the value of the objective function by walking along  $-c$  until we hit a constraint. Then, we can traverse this constraint by walking along the component of  $-c$  perpendicular to the constraint's normal (if this is parallel to  $-c$ , we can proceed in any direction along the constraint). This procedure will at some point terminate in a corner of the polyhedron, and since the polyhedron is convex, we will have reached the lowest feasible objective value.

In the integer case, however, the corners of the polyhedron will not necessarily be feasible solutions, since they might not be integer themselves. This complication renders the problem considerably harder in the worst case. Thus, while Linear Problems can be solved efficiently by traversing the corners of the solution polyhedron in a suitable manner, solving an Integer Linear Problem is NP-hard.

Fortunately, several sophisticated strategies for the solution of ILPs have been developed. While none of these can guarantee polynomial running time in the worst case, these techniques often work surprisingly well in practice. For instance, in the branch and cut - approach, the problem is first relaxed to a linear program without the integer constraint. If the solution of this problem is integer, the optimal ILP solution has been found. If it is fractional, instead, the next step consists in searching for an additional constraint that is fulfilled by all feasible integer points, but not by the fractional solution just found. Then, the relaxed problem, augmented by this constraint, is solved again in the hope that the solution becomes "less fractional" with each iteration and finally converges to the integer one in a finite number of steps. If no such constraint can be found, the problem is split into two independent subproblems instead which split the solution space in half. For instance, if the relaxed LP solution determined an optimal  $x_1 = 12.4$ , two independent problems will be set up with

## 2. Optimal Bond Order Assignment

$x_1 \leq 12$  and  $x_1 > 12$ , respectively, and the procedure will be run recursively in each of them. This leads to a binary tree of sub-problems, and will finally yield the desired solution. This combination of a branch-and-bound approach with a cutting-plane technique has proven particularly useful in practice and is realized in many open-source and commercial solvers.

We will now continue with our formulation of the bond order assignment problem as an Integer Linear Problem. Since this work was done in collaboration with Dr. Alexander Rurainski, some of these discussions can also be found in his dissertation [Rur10]. Hence, we will focus on the higher level concepts behind our approach instead of the details of its implementation. We will start our discussion with a formal introduction of our Integer Linear Program.

We use the following notations: Let  $P$  be the penalty table, then

- $A$  is the set of all atoms of the molecule under consideration.
- $B(a)$  is the set of bonds of atom  $a \in A$  and  $B$  denotes the set of all bonds of the molecule.
- $V(a) \subset \mathbb{N}$  contains the possible valences of atom  $a \in A$ .
- $P(a, v)$  is the penalty value for atom  $a \in A$  and valence  $v \in V(a)$ .

Our approach uses two different classes of variables. For each bond  $b \in B$  we introduce indicator variables  $x_{b_i} \in \{0, 1\}$ , symbolizing whether the corresponding bond  $b$  is set to the bond order  $i \in \{1, \dots, \mu\}$ . Here,  $\mu$  is the maximum bond order considered. In the following, we will set  $\mu$  to 3, allowing single, double, and triple bonds.

To ensure that each bond  $b \in B$  is assigned exactly one bond order, we add the linear constraints

$$\sum_{i=1}^{\mu} x_{b_i} = 1$$

for all  $b \in B$ . Then, the sum over all bond orders of all bonds  $b \in B(a)$  of an atom  $a$  can be computed as

$$\sum_{b \in B(a)} \left( \sum_{i=1}^{\mu} x_{b_i} \cdot i \right)$$

The second class of variables focuses on the atomic valences: for all atoms  $a$  and corresponding possible valences  $v$  according to the penalty table  $P$ , we introduce indicator variables  $y_{a,v} \in \{0, 1\}$ . Each  $y_{a,v}$  symbolizes whether the corresponding valence is chosen or not, i.e., penalty  $P(a, v)$  contributes to the score if and only if  $y_{a,v} = 1$ . Thus, the objective function of our score minimization problem can be formulated as a linear function in  $y$  with penalty prefactors:

$$\min_y \sum_{a \in A} \sum_{v \in V(a)} P(a, v) \cdot y_{a,v}.$$

To ensure that each atom is assigned exactly one valence state, we add the additional linear constraints

$$\sum_{v \in V(a)} y_{a,v} = 1$$

for all  $a \in A$ . In addition, we have to ensure that for each atom, the sum of its bond orders equals its chosen valence. These constraints can be formulated as

$$\sum_{v \in V(a)} y_{a,v} \cdot v = \sum_{b \in B(a)} \left( \sum_{i=1}^{\mu} x_{b_i} \cdot i \right)$$



for all  $a \in A$ , because the left hand side evaluates to valence  $v$  if and only if  $y_{a,v} = 1$ .

In summary, the score minimization problem can be formulated as the following integer linear program:

$$\begin{aligned}
 \min_{\mathbf{x}, \mathbf{y}} \quad & \sum_{a \in A} \sum_{v \in V(a)} P(a, v) \cdot y_{a,v} \\
 \text{s.t.} \quad & \sum_{v \in V(a)} y_{a,v} \cdot v = \sum_{b \in B(a)} \left( \sum_{i=1}^{\mu} x_{b_i} \cdot i \right) \quad \forall a \in A, \\
 & \sum_{v \in V(a)} y_{a,v} = 1 \quad \forall a \in A, \\
 & \sum_{i=1}^{\mu} x_{b_i} = 1 \quad \forall b \in B, \\
 & y_{a,v} \in \{0, 1\} \quad \forall a \in A, \forall v \in V(a), \\
 & x_{b_i} \in \{0, 1\} \quad i \in \{1, \dots, \mu\}, \forall b \in B.
 \end{aligned}$$

Additional solutions can be found if for each bond order assignment  $s = \{x_{b_i} | x_{b_i} = 1\}$  computed so far we add the constraint(s)

$$\sum_{x_{b_i} \in s} x_{b_i} < k \quad \forall s$$

where  $k$  denotes the number of bonds in the molecule. Please note that this is the reason that we used  $x_{b_i} \in \{0, 1\}$  instead of  $x_b \in \{1, \dots, \mu\}$  to model the bond orders, since it simplifies the enumeration of additional solutions.

For the solution of ILPs to provable global optimality, several strategies can be chosen, such as the popular pure branch & bound approaches or branch & cut methods [PS98]. We employed the open source solver `Ip_solve` [BEN04] which uses a simplex-algorithm based branch & bound approach [PS98]. In our experiments, we have seen a drastic increase in running time if more than one solution is computed. Thus, the ILP approach is not well suited for obtaining co-optimal or sub-optimal bond order assignments.

### 2.6.3. A Fixed Parameter Tractability Approach

In [BBST09], Böcker and coworkers were able to prove the NP-hardness of the bond order assignment problem in the form discussed in this work by a reduction of the 3-SAT\* problem [BKS07]. And even though the hypothesis that  $P \neq NP$  has still not been proven, NP-hardness implies that we should not expect to find a polynomial-time exact solver for bond order assignment. To make matters worse, [BBST09] also shows that the problem is inapproximable, meaning that every guaranteed  $\epsilon$ -approximation to the problem will again be NP-hard. Thus, it seems as if we could not expect to find an exact solution strategy with a runtime that can compete with the simple, albeit heuristic and inexact, Antechamber approach.

For some classes of NP-complete problems, however, efficient solutions can be found, given that the problem instances occurring in practice fulfill some formal restrictions on the input. These experiences lead to the concept of fixed-parameter tractability [DF99, FG06, Nie06, Bui10].

The general idea behind fixed-parameter tractability is the following: the NP-completeness of the problem suggests that each exact algorithm will have at least exponential runtime. If we can formulate the algorithm in such a way that the exponential terms in the runtime are exponential only in some parameter  $k$  of the input, and if this parameter  $k$  can be chosen in such a way that it will be bounded for all expected problem instances, the algorithm will indeed behave polynomial, not exponential. As an example, if we might find an algorithm for the bond order assignment problem that would be

## 2. Optimal Bond Order Assignment

exponential only in the number of rings contained in the input solution, and if we would restrict ourselves to input molecules with less than, e.g., 10 rings, we would have found a polynomial time algorithm, albeit with a very large constant prefactor.

In the following, we will briefly describe such a fixed-parameter approach for the bond order assignment problem which Prof. Dr. Sebastian Böcker and coworkers developed in collaboration with us. The parameter  $k$  of this approach will be significantly more complex than the number of rings used in the example above. On the other hand, the actual parameter (the width of a tree-decomposition of the molecule) will lead to a typically very small bound for realistic input molecules.

The idea behind the approach is as follows: many NP-hard graph based problems can be solved in polynomial time if the underlying graph is a tree. In general, graph algorithms often perform much faster on trees than on cyclic graphs because the deletion of an inner node partitions the tree into independent subgraphs. Algorithms for solving the independent, smaller, and simpler problems can be applied, and in the end, the solutions of all subgraphs have to be combined, which is often efficiently possible. These observations lead to the concepts of Divide & Conquer approaches and Dynamic Programming.

A popular approach to transform a cyclic graph into a tree is to perform a tree-decomposition [RS86]. In the following, we will describe how we transform a molecule into a tree and how we applied dynamic programming to compute a bond order assignment.

In our FPT approach, each molecule is considered as a *molecule graph*  $G = (U, E)$ , where each vertex represents an atom and each edge represents a bond. A *tree decomposition* of a molecule graph  $G = (U, E)$  consists of an index set  $I$ , a set of *bags*  $X_i \subseteq U$  for  $i \in I$ , and a tree  $T$  with node set  $I$  such that:

1. every vertex  $u \in U$  is contained in at least one bag  $X_i$ ;
2. for every edge  $\{u, v\} \in E$ , there is at least one bag  $X_i$  such that  $u, v \in X_i$ ;
3. for two nodes  $i, k$  of the tree  $T$ , if  $u \in X_i$  and  $u \in X_k$ , then  $u \in X_j$  also holds for all nodes  $j$  of the tree along the path from  $i$  to  $k$  in  $T$ .

Fig. 2.8 shows a molecule, its graph representation, and a corresponding tree decomposition. The reason why a tree decomposition helps with applying ideas from Divide & Conquer strategies can be intuitively understood from the fact that, if all atoms contained in one bag were removed from a molecule, it would be partitioned into independent parts, just as a tree is partitioned into independent parts by removing one inner node. Hence, the bags in the tree decomposition allow to distinguish between an 'up' and a 'down' direction for the Divide & Conquer strategy by using one of the resulting parts as the 'upper', the other parts as the 'lower' set. By definition of a tree decomposition, in particular, by property (3), this separation into 'upper' and 'lower' will always be consistent.

One important property of a tree decomposition is its width: the *width* of a tree decomposition is a measure of how far a graph differs from a tree and equals  $\omega - 1$  for  $\omega := \max\{|X_i| \mid i \in I\}$ . The *treewidth* of a graph  $G$  is the minimum width of any tree decomposition of  $G$ . With this definition, the treewidth of a tree equals one, while the treewidth of a clique of size  $n$  equals  $n-1$ .

Given a molecule graph  $G$ , we first compute a tree decomposition. We will see below that the running time and the required space of our algorithm grow exponentially with the width of the decomposition. Unfortunately, computing a tree decomposition with minimum width is again an NP-hard problem [ACP87]. But fortunately, heuristic and exact algorithms to compute such tree decompositions efficiently in practice exist [GD04, BFK<sup>+</sup>06].

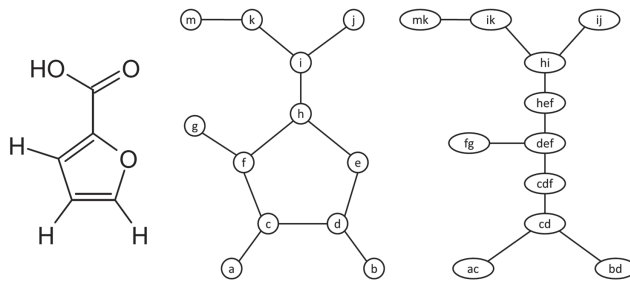


Figure 2.7.: A molecule (left), its graph representation (middle) and a corresponding tree decomposition (right) with width 2.

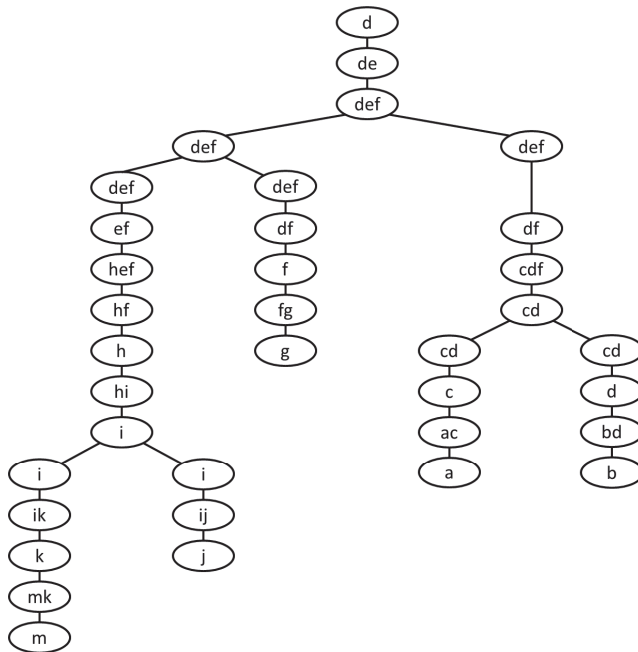


Figure 2.8.: The corresponding nice tree decomposition.

## 2. Optimal Bond Order Assignment

To simplify the description of our algorithm, we use the special case of *nice* tree decompositions: here, we assume the tree  $T$  to be rooted. A *nice tree decomposition* is a tree decomposition satisfying the following additional requirements:

1. Every node of  $T$  has at most two children.
2. If a node  $i$  has two children  $j$  and  $k$ , then  $X_i = X_j = X_k$ ; in this case,  $i$  is called a *join node*.
3. If a node  $i$  has one child  $j$ , then one of the following two conditions must hold:
  - a)  $|X_i| = |X_j| + 1$  and  $X_j \subset X_i$ ; in this case,  $X_i$  is called an *introduce node*.
  - b)  $|X_i| = |X_j| - 1$  and  $X_i \subset X_j$ ; in this case,  $X_i$  is called a *forget node*.

Here, introduce nodes and forget nodes are viewed as moving bottom-up from the leaves to the root. We can easily transform a tree decomposition into a nice tree decomposition, in time linear in the size of the tree decomposition. Let  $T$  denote a tree decomposition of a graph  $G$  with bags  $X_i$ . As a first step, we root the decomposition  $T$  at an arbitrary node  $X_r$ . Now, we need to ensure that each node is either a join, introduce, or forget node. To create suitable join nodes, we consider every node  $X_i$  with more than one child. A join node has exactly two children with the same content as itself. Thus, we first remove all  $n$  children of  $X_i$  and store them for later usage. Then, we add a binary tree at  $X_i$  with  $n$  leaves rooted at  $X_i$ , where every node has the same content as  $X_i$ . Finally, we reconnect one of the stored original children with each leaf of the binary tree. After this binarization, every inner node of  $T$  has at most two children and two children of an inner node contain the same vertices as their parent node. The only violation of the niceness that can still occur is to forget or to introduce more than one atom in a node. This can be fixed by replacing the node by a path of introduce or forget nodes. The result will be a nice tree decomposition.

Interestingly, it can be shown [Bui10] that by this algorithm, the size of  $T$  increases by a factor of at most  $d_{max} \times \omega$ , where  $d_{max}$  is the maximum degree in the original tree decomposition. Also, the width of  $T$  remains unchanged.

Fig. 2.8 illustrates a tree decomposition and a corresponding nice tree decomposition of a graph. It can be easily verified that the union of all bags in the tree decomposition, as well as of all bags in the nice tree decomposition, contain every vertex of the graph, and every edge of the graph exists in at least one bag of the tree decompositions. Furthermore, all bags sharing a common vertex induce a connected subgraph in the tree decomposition.

In addition to the conversion into a nice tree decomposition, we apply further extensions: the tree  $T$  is rooted at an arbitrary bag. Above this root, we add additional forget nodes, such that the new root contains a single vertex. Let  $X_r$  denote the new root of the tree decomposition and  $v_r$  denote the single vertex contained in  $X_r$ . Analogously, we add additional introduce nodes under every leaf of  $T$ , such that the new leaf also contains a single vertex.

As stated above, with a tree decomposition we now know how to define at each node in the graph an 'up' and 'down' subgraph, where we can apply the ideas of dynamic programming. A crucial idea is to delay the scoring of each atom until it is forgotten. On the path from the root to any leaf, this can happen only once for any atom because of property 3. Hence, the decision is unique for every atom on each path.

To use the above facts efficiently, we introduce for every node  $i$  in the (extended nice) tree decomposition a scoring matrix  $D_i$ . For bag  $i$  with atoms  $\{a_1, \dots, a_k\}$ , the scoring matrix will have the form  $D_i[v_1, \dots, v_k; b_{1,2}, \dots, b_{k-1,k}]$ , i.e., we assign one index for each atom in the bag, and one index for each of the bonds between atoms in the bag.

Intuitively, the scoring matrix will answer for every node (i.e., bag)  $i$  the following question: what is the best total penalty score that can be reached if all previous decisions lead to the valences  $v_1, \dots, v_k$  for the atoms in bag  $i$ , and if the atoms in bag  $i$  form bond orders  $b_{1,2}, \dots, b_{k-1,k}$  between them. The optimization in this node thus occurs over all possible previous decisions, where previous is to

be understood in the sense of our idea of a decomposition into 'lower' and 'upper' (see Fig. 2.9): all atoms that have been forgotten in the subtree  $T_i$  rooted at node  $i$  have already been scored, i.e., their best possible scores for all their valence combinations have already been computed and stored in the corresponding score matrices in the subtree. Hence, in computing the new scoring matrix, we can simply refer to these values and combine them in a suitable manner.

With this definition, it is immediately clear how to initialize the values of the leaves: in each leaf, we only have one atom  $a_1$ . Thus, we have no bonds to optimize over, and the valence so far assigned to the atom is 0. Thus, the matrix  $D_i[v_1; \cdot]$  will be 0 if  $v_1$  equals 0 and  $\infty$  otherwise.

Now, we can deduce the recursion formulas. Here, using a nice tree decomposition finally comes in handy: we only need to distinguish between three possible cases, depending on the type of the current node.

If this node is an introduce node, it contains one atom more than its only child. In this case, we only have to carry over the scores from the subtree for all feasible combinations: since, according to proposition 3, the new atom  $a_k$  never appeared in the subtree below it, the new score matrix will contain a value of  $\infty$  for all combinations with a non-zero  $v_k$ . For zero  $v_k$ , we can simply copy over the scores from the subtree.

If the node is a forget node, we will have to perform the mentioned optimization over the bond orders and resulting valences for the atom that is to be forgotten, i.e., the atom that is to be scored.

Finally, if the father is a join node, it has two children with different sets of forgotten atoms. For both children, we have already computed the optimal ways to achieve valences  $v_1, \dots, v_k$  for every feasible valence combination through bonds to their respective sets of forgotten atoms. If we want to determine the best way to assign, for example, a valence of 2 to atom  $a_1$ , we could either take a full valence of 2 from the first child, and a valence of 0 from the second, or a valence of 1 from first and second, or a valence of 0 from the first and 2 from the second child. The optimization thus takes the minimal resulting score over all these combinations.

In the following, we give the corresponding formulas in full mathematical detail. For this, a few definitions are required.

Let  $\{a_{i,1}, a_{i,2}, \dots, a_{i,k}\}$  be the atoms inside bag  $X_i$ , where  $k \leq \omega$ . In our presentation below, we want to avoid double indices, so we refer to the atoms inside bag  $X_i$  as  $a_1, a_2, \dots, a_k$ . It should be understood that these are different atoms for each bag. For simplicity of presentation, we also assume that the molecular subgraph induced by  $a_1, a_2, \dots, a_k$  is fully connected and, thus, contains all bonds  $a_1a_2, a_1a_3, \dots, a_{k-1}a_k$ . In our implementation, of course, we do not make this assumption.

Let  $Y_i$  denote the atoms in the molecule graph  $G$  that are contained in the bags of the subtree  $T_i$  of  $T$  below bag  $X_i$ :  $Y_i = \bigcup_{j \in T_i} X_j$  (see Fig. 2.9). To simplify the formulation of the dynamic programming below, we will not use the bond order  $\tilde{b}_{i,j}$  between atoms  $a_i, a_j$  but, instead, the *free bond order*

$$b_{i,j} := \tilde{b}_{i,j} - 1 \in \{0, 1, 2\}$$

Then, the *valence* of an atom  $a$  is the sum of free bond orders over all incident bonds, plus the degree  $\deg(a)$  of the atom in the molecule graph, i.e., the number of bonds it participates in. We assign a score matrix  $D_i$  to each bag  $X_i$  of the tree decomposition: let  $D_i[v_1, \dots, v_k; b_{1,2}, \dots, b_{k-1,k}]$  be the minimum score over all valence assignments to the vertices (i.e., the atoms) in  $Y_i \setminus X_i$  if for every  $l = 1, \dots, k$ ,  $v_l$  valences of atom  $a_l$  have been consumed by the atoms in  $Y_i \setminus X_i$ , and free bond orders  $b_{1,2}, \dots, b_{k-1,k}$  are assigned to bonds  $a_1a_2, a_1a_3, \dots, a_{k-1}a_k$ . Using this definition, we delay the scoring of any vertex to the forget node where it is removed from a bag. We can compute the minimum score among all assignments using the root bag  $X_r = \{a_r\}$  as

$$\min_{v_r \in V(a_r)} \{P(a_r, v_r) + D_r[v_r]\}.$$

The algorithm starts at the leaves of the tree decomposition and computes the score matrix  $D_i$  for

## 2. Optimal Bond Order Assignment

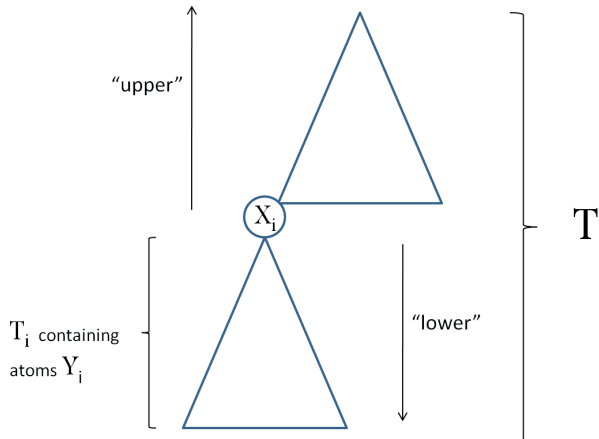


Figure 2.9.: Definition of a subtree: the subtree  $T_i$  of a tree decomposition  $T$  contains all bags below node  $i$ ;  $Y_i$  denotes the set of all atoms within the subtree  $T_i$ .

every node  $X_i$  when score matrices of its children nodes have been computed. We initialize the matrix  $D_j$  of each leaf  $X_j = \{a_1\}$  with  $D_j[v_1; \cdot] = 0$  if  $v_1 = 0$ , and  $D_j[v_1; \cdot] = \infty$  otherwise. During the bottom-up traversal, the algorithm then distinguishes whether  $X_i$  is a forget node, an introduce node, or a join node, and computes  $D_i$  as follows:

**Introduce nodes.** Let  $X_i$  be the parent node of  $X_j$  such that  $X_j = \{a_1, \dots, a_{k-1}\}$  and  $X_i = \{a_1, \dots, a_k\}$ . Then

$$D_i[v_1, \dots, v_k; b_{1,2}, \dots, b_{k-1,k}] = \begin{cases} D_j[v_1, \dots, v_{k-1}; b_{1,2}, \dots, b_{k-2,k-1}] & \text{if } v_k = 0, \\ \infty & \text{otherwise.} \end{cases}$$

**Forget nodes.** Let  $X_i$  be the parent node of  $X_j$  such that  $X_j = \{a_1, \dots, a_k\}$  and  $X_i = \{a_1, \dots, a_{k-1}\}$ . Then

$$D_i[v_1, \dots, v_{k-1}; b_{1,2}, \dots, b_{k-2,k-1}] = \min_{\substack{b_{1,k}, \dots, b_{k-1,k} \in \{0,1,2\} \\ v_k \in \{0, \dots, \max\{V(a_k)\} - \deg(a_k)\}}} \left\{ P\left(a_k, v_k + \deg(a_k) + \sum_{l=1}^k b_{l,k}\right) + D_j[v_1 - b_{1,k}, \dots, v_{k-1} - b_{k-1,k}, v_k; b_{1,2}, \dots, b_{k-1,k}] \right\}$$

where  $\deg(a_k)$  denotes the degree of vertex  $a_k$ .

**Join nodes.** Let  $X_i$  be the parent node of  $X_j$  and  $X_h$  such that  $X_i = X_j = X_h$ . Then

$$D_i[v_1, \dots, v_k; b_{1,2}, \dots, b_{k-1,k}] = \min_{\substack{v'_l=0, \dots, v_l \\ \text{for } l=1, \dots, k}} \left\{ D_j[v'_1, \dots, v'_k; b_{1,2}, \dots, b_{k-1,k}] \right. \\ \left. + D_h[v_1 - v'_1, \dots, v_k - v'_k; b_{1,2}, \dots, b_{k-1,k}] \right\}$$

For simplicity of the presentation of our algorithm, we assumed above that every two vertices in each bag of the tree decomposition are connected by an edge, but in reality, the degree of a vertex in a molecule graph cannot exceed the maximum valence  $d \leq 7$  of an atom in the molecule graph. Therefore, the number of edges in a bag is upper-bounded by  $\omega d$ . Given a nice tree decomposition of a molecule graph  $G$ , the algorithm described above computes an optimal assignment for the bond order assignment problem on  $G$  in time  $\mathcal{O}(\alpha^{2\omega} \cdot 3^\beta \cdot \omega \cdot m)$ , where

$$\alpha = 1 + \max_{a \in A} \{\max V(a)\}$$

is the maximum (open) valence of an atom plus one,  $m$  and  $\omega - 1$  are size and width of the tree decomposition,  $d$  is the maximum degree in the molecule graph, and  $\beta := \min\{\binom{\omega}{2}, \omega d\}$  [BBST09]. Obviously, the exponential behaviour is restricted to the treewidth parameter  $\omega$ .

The implementation of this approach, created in the group of Prof. Dr. Sebastian Böcker, was done in Java. For the optimal tree decomposition of molecular graphs, it used the method QuickBB in the library LibTW implemented by van Dijk *et al.* (<http://www.treewidth.com>). The result is then transformed into a nice tree decomposition in linear time. Running times reported for the FPT algorithm (c.f. table 2.4) include the running times of computing the optimal nice tree decomposition. Full details of the implementation – such as the use of hash maps instead of arrays – can be found in [BBST09]. Here, we only want to note one interesting common approach to reduce the runtime in practice: the program initializes an integer  $u = 0$  and does not store matrix entries with score exceeding  $u$ . If the score of the optimal solution is at most  $u$ , this optimal solution will be found. Otherwise, the algorithm is called again with increasing  $u$ , until an optimal solution is found. If not only the optimal solutions but also a certain number of sub-optimal solutions are required, the algorithm is called repeatedly with increasing  $u$ , until all required sub-optimal solutions are found or  $u$  arrives at its upper bound

$$\sum_{a \in A} \max_{v \in V(a)} P(a, v).$$

Recently, we have converted the initial Java implementation to C++, based on an initial conversion by Kai Dührkop. Details on this implementation can be found in Section 4.2.

## 2.7. Extensions of our Approach

In addition to solving the basic bond order assignment problem as defined in Section 2.2, we further augmented BOA Constructor to approach related problems, i.e., scoring by structural information instead of connectivity, combining structural and topological information for scoring, deducing additionally missing hydrogens, and generalizing the scoring scheme for partial bond order assignment. All but the last represent more difficult problems and add further complexity. We exemplarily present and discuss the extensions for the A\* scoring scheme.

In principle, most extensions discussed here can be adapted to the ILP or FPT as well. However, the simplicity of the A\* algorithm greatly facilitates such extensions and hence, it is of great use as a sounding board for novel ideas.

## 2. Optimal Bond Order Assignment

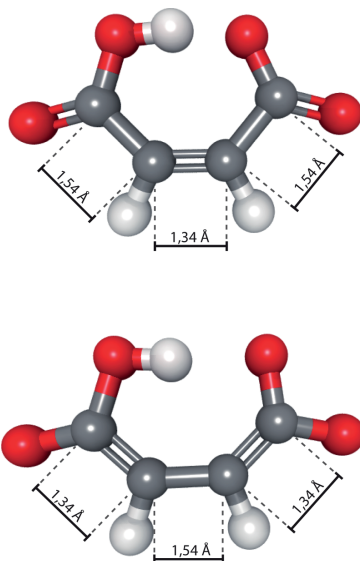


Figure 2.10.: Including bond length information into bond order assignment.

In addition to these extensions, we also improved the penalty table. This, however, is completely independent from our solvers and will be discussed in Section 2.8.

### 2.7.1. Introducing Structural Information

So far, the search algorithms neglect structural information such as bond lengths or angles. But for input molecules with rather reliable positions, using structure derived information may further assist in improving the bond order assignment (see Fig. 2.10).

To show how this can be achieved, we extend the penalty scoring of a molecule  $M$ 's bond order configuration  $c_i$  by a *bond length penalty* term.

Similarly, bond angles or hybridization states can be taken into account. The overall *bond length deviation* of all bonds in configuration  $c_i$  compared to averaged bond lengths, computed for all orders and atom pairs, is given by

$$bd = \sum_{b \in W(B)} (\text{len}(b) - \bar{l}_{t(b),o})^2$$

where  $\text{len}(b)$  denotes the bond length of bond  $b$ , and  $\bar{l}_{t(b),o}$  denotes the averaged bond length of reference bonds connecting the same atom types and having the same bond order  $o$  as bond  $b$  in configuration  $c_i$ . We already introduced the notation of  $W(B)$  to give the set of all assigned bonds of molecule  $M$  in configuration  $c_i$ , i.e., node  $w$ , that are already assigned a bond order.

The average values  $\bar{\text{len}}(b)$  can either be statistically derived from a data set, or taken from alternative sources. Here, we opted for the bond length definitions of the MMFF94 force field [Hal96a].

In order to compare two bond length penalties of two bond order configurations of the same molecule  $M$ , we normalize the bond length deviation by the sum over all its atoms' maximally possible bond



length deviations. Thus, the *normalization factor* for the bond length penalty of molecule  $M$  is given by

$$\text{bd}_{\max} = \sum_{b \in B} \max_{o \in \{1, \dots, \mu\}} (\text{len}(b) - \bar{l}_{t(b),o})^2.$$

If a bond  $b$  is assigned to a bond order for which no averaged length is available, the deviation is set to the maximally possible bond length deviation over all other bond orders for this bond type. If there is no such other bond order set, the bond is excluded from computation both in the bond length deviation and the normalization factor. In all other cases, the normalized bond length penalty can then be defined as

$$\text{bp} = \frac{\text{bd}}{\text{bd}_{\max}}.$$

The problem of finding a most probable bond order assignment for a molecule based on its atom connectivity can be translated into the problem of finding a bond order assignment with minimal bond length penalty, which can be further mapped onto an A\*-search algorithm in a very similar manner as for the atom-penalty approach presented in Section 2.6.1: a node  $w$  in layer  $l$  of the search tree represents a partial bond order assignment of the first  $l - 1$  bonds. It has  $m_l$  children, where  $m_l$  is the number of possible bond orders of bond  $l$ . For the evaluation of a node  $w$  in the search tree that represents a partial bond order conformation  $c_i$ , we define a function  $f = g_4^* + h_4^*$  as follows:

$$g_4^* = \text{bp} \tag{2.8}$$

$$h_4^* = \sum_{b \in B \setminus W(B)} \min_{o \in \{1, \dots, \mu\}} (\text{len}(b) - \bar{l}_{t(b),o})^2 \tag{2.9}$$

where  $B$  again denotes the total set of bonds of molecule  $M$ .  $W(B)$  denotes the subset of bonds in  $B$  contained in configuration  $c_i$  (corresponding to node  $w$ ) that are already assigned a bond order.  $t(b)$  denotes the type of bond  $b$ , and  $\bar{l}_{t(b),o}$  denotes the average length of all bonds of type  $t$  and order  $o$  in the data set.

Given this heuristic, the first leaf reached by an A\* search algorithm has minimal bond length penalty, i.e., the proposed bond orders have minimal deviation from comparable reference bond orders.

The heuristic (2.9) can be further tightened by considering the atom's minimal and maximal valences, which leads to the following evaluation function:

$$f = \begin{cases} g_4^* + h_4^* & \forall a \in A: \sum_{b \in B(a)} v \in [\mathcal{V}_{\min}(a), \mathcal{V}_{\max}(a)] \\ \infty & \text{else} \end{cases} \tag{2.10}$$

where  $B(a)$  denotes the set of bonds of atom  $a \in A$  and  $\mathcal{V}_{\min}(a)$  ( $\mathcal{V}_{\max}(a)$ ) denotes the minimal (maximal) possible valences for atom  $a$ , which we deduced from the given atom penalty table  $P$ .

In our experiments we found that pure bond length information does not suffice to clearly distinguish between reasonable bond order assignments in the A\* approach. We thus developed a strategy to combine structural and connectivity information which we call *fine penalty* (fp): whenever two nodes in the atom penalty based A\* bond order search tree (see Alg. 3) are evaluated by the function  $f$  to the same value, we additionally applied the search heuristics (2.8), (2.9), (2.10) to decide which node to expand first. This strategy has the advantage that it does not change the optimality of the solutions, but simply re-orders co-optimal ones according to decreasing likelihood.

### 2.7.2. Hybrid Penalty Score

The fine penalty corresponds to a subsequent application of connectivity and structure based penalty scores. Including both, connectivity and structural information, in a more general fashion than a

## 2. Optimal Bond Order Assignment

subsequent application can be done by combining both the atom penalty and the bond length penalty approach to a linear function, providing a tunable hybrid score. To this end, a normalization of the atom type penalty is necessary as well.

We define the *atom type normalization factor* of a molecule  $M$  as

$$\text{ap}_{\max} = \sum_{a \in A} \max_{v \in V(a)} \{P(a, v)\}.$$

Thus, the *normalized atom type penalty* can be expressed by

$$\text{ap} = \frac{1}{\text{ap}_{\max}} \left( \sum_{a \in A} P \left( a, \sum_{b \in B(a)} \text{bo}(b) \right) \right).$$

Combining the normalized atom type penalty with the normalized bond length penalty gives the hybrid penalty function  $\alpha (\text{bp}) + (1 - \alpha) (\text{ap})$ , where  $\alpha \in [0, 1]$  denotes the tuning factor.

Applying the idea of a hybrid penalty also to the search heuristic, we end up with the following hybrid evaluation function  $f^* = g_{\text{hybrid}}^* + h_{\text{hybrid}}^*$ :

$$g_{\text{hybrid}}^* = \frac{\alpha}{\text{bd}_{\max}} \left( \sum_{b \in W(B)} (\text{len}(b) - \bar{l}_{t(b),o})^2 \right) \quad (2.11)$$
$$+ \frac{(1 - \alpha)}{\text{ap}_{\max}} \left( \sum_{a \in K} P \left( a, \sum_{b \in B(a)} \text{bo}(b) \right) \right)$$

$$h_{\text{hybrid}}^* = \frac{\alpha}{\text{bd}_{\max}} \left( \sum_{b \in B \setminus W(B)} \min_{o \in \{1, \dots, \mu\}} (\text{len}(b) - \bar{l}_{t(b),o})^2 \right) \quad (2.12)$$
$$+ \frac{(1 - \alpha)}{\text{ap}_{\max}} \left( \sum_{a \in A \setminus K} \min_{l(a) \leq i \leq V_{\max}(a)} \{P(a, i)\} \right)$$

In cases where structural information is included ( $\alpha > 0$ ), further speed up can be gained by splitting the search tree at trusted bonds into a number of *independent* smaller search trees, computing for each subtree the optimal solution(s), and finally combinatorially combining those independent optimal solutions to yield the set of all optimal full solutions. Such trusted bonds can, e.g., be single bonds between two carbons that can be identified relatively reliably by their characteristic tetrahedral bond angles.

As shown in the previous section, further speed up can be reached by applying the tighter heuristic (2.5) to the atom type based term.

Very similar to the aforementioned inclusion of bond length information, torsional angle information can be included into our algorithm in order to additionally guide to the most probable bond order assignment. This, however, has not yet been implemented and is the subject of future work.

### 2.7.3. Hydrogens

Experimental structure elucidation is often blind to hydrogen positions. Consequently, many structural databases such as PDB [BWF<sup>+</sup>00, BHN03], often lack hydrogens, which renders, e.g., the discrimination between hydroxyl oxygens and carbonyl oxygens difficult. Due to the diversity of possible input molecules, our A\*-search algorithm offers the functionality to add additional hydrogens. To this end, we extended the standard A\*-bond order algorithm by adding further *virtual* hydrogen bonds whenever

the maximal valence is not reached. These bonds have the special property that they are restricted to having bond orders 0 or 1. The resulting algorithm is presented in Alg. 4.

Let  $v_a$  denote the current atomic valence of atom  $a$ ,  $\mathcal{V}_{\max}(a)$  the maximal allowed atomic valence according to the atom penalty table  $P$  (see penalty table in [WWKC06]), and  $f(w)$  the search heuristic of a node  $w$ . Please note that in principle all search heuristics (2.2), (2.4), (2.5), (2.9), (2.10), or (2.12) can be used.

Line 10 of the algorithm iterates over all possible bond orders, i.e., for the virtual hydrogen atoms the possible bond orders are 0 and 1. The solutions of Alg. 3 are a subset of the solutions produced by Alg. 4.

---

**Algorithm 4** A\*-bond order algorithm with adding of hydrogen atoms (molecule  $M$  with  $n$  bonds)

---

```

1: for all  $a \in \text{atoms}(M)$  do
2:    $v_a \leftarrow \sum_{b \in \text{bonds}(a)} 1$ 
3:   while  $v_a < \mathcal{V}_{\max}(a)$  do
4:     create virtual bond  $b_{\text{new}}$  for atom  $a$ 
5:      $v_a \leftarrow v + 1$ 
6:   end while
7: end for
8:  $\mathcal{PQ} \leftarrow \{r\}$ 
9:  $b_1 :=$  is the first free bond in  $M$ 
10: for all bond orders  $i$  do
11:    $\mathcal{PQ} \leftarrow \mathcal{PQ} \cup \text{bondOrders}(\{b_1 \leftarrow i\})$ 
12: end for
13: while  $\mathcal{PQ} \neq \emptyset$  do
14:    $\bar{w} \leftarrow \arg \min_{w \in \mathcal{PQ}} f(w)$ 
15:    $\mathcal{PQ} \leftarrow \mathcal{PQ} \setminus \{\bar{w}\}$ 
16:   if  $\bar{w}$  is leaf then
17:     store or assign bond orders as denoted in  $\bar{w}$ 
18:   else
19:      $b_x :=$  is next free bond in  $M$ 
20:     for all bond orders  $i$  do
21:        $\mathcal{PQ} \leftarrow \mathcal{PQ} \cup \text{bondOrders}(\bar{w} \cup \{b_x \leftarrow i\})$ 
22:     end for
23:   end if
24: end while

```

---

#### 2.7.4. Selective Bond Order Assignment

For some applications, it makes sense to consider not all bonds of a molecule, but rather to keep some trusted bond orders fixed. In a combinatorial chemistry application, e.g., the core fragment's bond orders should remain unchanged while the newly added parts of the molecule need to be assigned reasonable bond orders. All aforementioned algorithms work perfectly with BALL's sophisticated selection mechanisms. Selection filters can be defined by BALL predicates, SMILES, and SMARTS strings to restrict BOA Constructor's scope. Thus, bond order assignment restricted to selected parts of a molecule is easily possible.

## 2. Optimal Bond Order Assignment

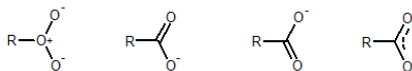


Figure 2.11.: Resonance structures of a carboxylic group.

## 2.8. New Penalty Table

The original heuristic implementation of the bond order assigner as presented in Wang et al. [WWKC06] used a fixed, hard-coded set of 35 atom classes. Adding even a single class, or splitting one existing class into two, for instance, would require large-scale changes in the code base. However, changing the classes would often be desirable. An important example is given by the carboxylic group (see Fig. 2.11), for which the original approach proposed two double bonds, i.e. a charged carbon. For most purposes, though, one would prefer a single-bond/double-bond combination. To this end, the corresponding rule for carbon allowing valence 5 (former Antechamber rule 8) has to be deleted and an additional atomic valence has to be introduced for oxygens, allowing one oxygen in a carboxylic group to have valence 1 instead of being mapped to former rule 20.

Fortunately, BALL provides us with a simple, yet very flexible, way of grouping atoms dynamically into classes: the expression mechanism, which supports simple predicates as well as complex ones like SMARTS [SMA] expressions. These predicates can be combined with boolean operators to form powerful selection patterns. Hence, we decided to harness this functionality for bond order assignment as well. In our implementation, rules and atom classes are not hard coded, but rather read from an XML-file that contains BALL-expressions for selecting the atoms corresponding to their penalty classes and the corresponding penalty data. The full table of expressions and penalty values can be found in table A.1 in Appendix A.1. Here, we will exemplarily discuss the meaning of one entry of the penalty table, namely rule 12, which looks as follows:

```
SMARTS([#7D3] (~[#8D1, #16D1]) (~[!#8#!#16, !D1]) (~[!#8#!#16, !D1]))
```

The keyword SMARTS indicates to BALL's expression parser that the expression in parentheses should be forwarded to the SMARTS matcher. A full description of the SMARTS language would be well out of scope of this work. The interested reader is referred to the official SMARTS homepage [SMA] instead. Here, we briefly describe the SMARTS subset used for our penalty table. In a SMARTS expression, individual atoms can be described in several different ways, such as their element type, connectivity, aromaticity, and bonds. Individual properties can be combined by logical operators, such as **and**, which is denoted by **&**, **or**, which is denoted by **,**, and **not**, which is denoted by **!**. The element of an atom can be denoted by its symbol, where lower- and upper case are used to discriminate between aromatic and non-aromatic atoms, or by its atomic number prefixed with a hash symbol **#**, which matches irrespective of aromaticity. The degree of an atom, i.e., the number of bonds it participates in, is denoted by the symbol **D**, followed by the desired number.

Thus, in the expression above, the first subexpression in square brackets describes a single atom of atomic number 7 – a nitrogen – that is bound to three other atoms. The following terms in parentheses describe the environment of this atom further. In our case, we describe its three neighbour atoms. In this description, the symbol **~** denotes a bond of arbitrary order. Since we do not know the correct bond orders yet, this is the only connection type we can employ in our rules. Considering that the rules for the second and third partner in the above expression are the logical complement of the rule for the first, we can finally see that the rule requires exactly one of the binding partners to be an oxygen or

a sulfur participating in only one bond. In total, the rule thus matches only for nitrogen atoms with three bonds, where exactly one of these bonds connects to a singly-bonded oxygen or sulfur.

Given the editable configuration files, adjusting the former penalty table to cover also previously unrecognized or wrongly assigned molecules is simple. E.g., the MMFF94 validation suite includes a file DIGCOL (see Fig. A.1(g)) containing a charged CSS<sup>-</sup>-group, which was treated in a very unconventional fashion by the Antechamber table. Instead of a single/double bond combination, Antechamber used two double bonds. Hence, using default parameters will not produce a suitable solution in this case.

Extending the rule set via our user-adaptable penalty table framework for sulphur (S) atoms within such a functional group can be simply achieved by adding the entry

```
<entry id="37">
  <elementstring>S</elementstring>
  <smartstring>SMARTS([$([#16D4] (~ [#8D1-, #16D1-])
                                (~ [#8D1 , #16D1])
                                (~ [#8D1 , #16D1])
                                (~ [#8 & !#16, !D1]))))
  </smartstring>
  <penalty valence="6">0</penalty>
</entry>.
```

This is a good example of the power provided by our user-adaptable configuration files, since a single additional rule will lead to the desired result.

Similarly, 4 additional molecules of the MMFF94 validation suite (H3OPW1, FE2PW3, FE3PW3, VIMHII see Fig. A.2(c), Fig. A.2(a), Fig. A.2(b) and Fig. A.2(d)) are not well covered by the original Antechamber rules either, but can be simply treated by adding rules for ions and oxygen with three neighbours (rule 22, rule 45, and rule 54 in table A.1).

Please note that rules 16, 23, and 40 do not make sense chemically, but have been included as fallback default rules or to handle some degenerate cases sometimes occurring in the input when the molecules are automatically generated. Depending on the application scenario, it may be preferable to disable these rules and abort if one of these cases is encountered.

The original Antechamber set contained 35 atom classes. We adapted this set by extending existing rules and adding rules for non-covered atomic bond situations, ions, and charged atoms, yielding a new penalty class set containing 54 atom classes. Three rules of the original Antechamber set (new rule 12, 15, 45) were adapted by simply extending the atomic valences. In addition, we added 21 rules covering additional molecular fragments as denoted in the column "description" of table A.1. Eight rules out of these 21 were added to cover unbound ions that might occur in molecular files. Five rules (1, 10, 26, 37, and 44) were added to cover charged atoms. Handling of charged atoms does not always need the creation of a new rule. E.g., rule 18 simply allows two valences av1 and av2 with assigned penalty scores of 0.

In addition to the aforementioned extensions, we simplified a number of Antechamber rules in cases where the atom class definition implied restrictions to the reachability of valence states, but these valence states were nevertheless allowed. Column "description" in table A.1 denotes the corresponding rules (2, 13, 15, 21,25, 29, 34). The former rules 33 and 34 stated equal allowed valence states for two molecular sets, where one was a subset of the other one. Since in the new rule 38, we can forgo the distinguishing SMARTS expression, additional speed up in the SMARTS matching was gained.

Please note that the definition of the former Antechamber rule 30 in the respective publication differs from its implementation in the Antechamber package. For our work, we chose the version found in the implementation.

## 2.9. Results

Of these algorithms, the A\* is clearly the simplest and most straight-forward approach to implement, and is highly efficient for small to medium sized input data. Its most striking advantage, however, is the ease with which it can be modified or extended. Hence, we use the A\* algorithm as our test bed for novel ideas, such as the introduction of structural features.

In contrast, the ILP approach requires the availability and integration of an ILP solver such as CPLEX, Gurobi (<http://www.gurobi.com/>) or `lp_solve` [BEN04], and is difficult to extend. On the other hand, its computational efficiency, in particular for larger molecules, is much higher than that of the A\*.

Finally, the FPT approach requires a complex implementation and is the hardest of the three to extend. On the other hand, it has unparalleled efficiency for an exact approach and is typically the fastest of our algorithms.

Apart from the guaranteed optimality of the solution returned by our solver, we found a second advantage of our more formal approach to be just as important: all of the exact solvers are able to not only return one, but rather all optimal solutions for a given molecule<sup>2</sup>. This allows downstream techniques to work with ensembles of bond order assignments, e.g., different resonances instead of a single fixed representation. If even more information about potential bond order assignments is required, sub-optimal solutions can be enumerated as well.

All algorithms were extensively tested on large data sets of small molecular entities, with greatly improved results over earlier approaches. At this point, we also introduced a novel penalty table that increases the accuracy of the method even further, while it simultaneously extends the applicability to further classes of molecules.

Having thus established the superiority of exact optimization techniques for bond order assignment over earlier heuristic approaches, we continued to extend the basic algorithm in several different directions, such as the addition of missing hydrogens, the inclusion of structural information into the otherwise purely connectivity based approach, and the proper handling of charges. All of these extensions were implemented and tested on the basis of our A\* method.

For proteins and DNA, bond orders can be simply inferred by matching the given state to a database containing the bond orders for all amino acids and nucleotides. Hence, we focus the evaluation of our algorithms on small and medium-sized molecules, e.g., drug-like molecules. Such molecules can be found in large numbers in several established ligand databases, such as ZINC [IS05], ASTEX [NMH<sup>+</sup>02], KEGG Ligand and KEGG Drug Database [GOH<sup>+</sup>02], the MMFF94 validation suite [Hal96b], or the Cambridge Structural Database [All02]. But evaluating the correctness of our bond order algorithms poses certain constraints: we need ligand structures that contain not only the connectivity information, but also pre-assigned bond orders and explicit hydrogens. As a further requirement, aromatic bonds should be given in kekulized form, i.e., replaced by a suitable pattern of single and double bonds. In contrast to structure based bond assignment approaches, however, we can use databases that contain three-dimensional ligand structures as well as those only storing structure diagrams or SMILES expressions. To provide a diverse test data set fulfilling those constraints, we chose the MMFF94 validation suite and the KEGG Drug set to provide our ground truth.

The following sections are organized as follows: Section 2.9.1 describes the data sets used for evaluation. In Section 2.9.2, we compare the total penalty score *tps* of the results of our exact solvers with that of the results of the original Antechamber approach. In Section 2.9.3, we compare the results of the different approaches to the expert generated, hand-crafted reference assignments, while Section 2.9.4 shows typical running times on realistic input examples. The different penalty tables are evaluated against each other in Section 2.9.5 before we end the section with a discussion of a first integration of structural information in Section 2.9.6.

<sup>2</sup> For the ILP solver, computing all optimal solutions greatly spoils the running times, though.

### 2.9.1. Data Sets for Validation

The MMFF94 validation suite [Hal96b] provides 761 small drug-like molecules, mainly derived from the Cambridge Structural Database [All02]. The molecules were thoroughly prepared by the authors of the MMFF94 force field by assigning bond orders, adding hydrogens where valences had to be completed, and minimizing the resulting complexes. The MMFF94 validation suite was originally designed to test the MMFF94 force field parameters, and thus yields a diverse set of molecules with hand-curated connectivity information, hydrogens, and bond order assignment, and 3D positions that we found very reasonable for testing bond order perception.

The KEGG Drug Database [GOH<sup>+</sup>02], provided by the Kanehisa Laboratories, offers a remarkable number of drug-like molecules for diverse applications in bioinformatics. The molecular coordinates are two dimensional, which is suitable for representation by structure diagrams, but is unsuited for structure based bond order assignment as performed by most of the former approaches. It thus represents a perfect test scenario for bond order assignment from topology alone. Unfortunately, hydrogens are missing in the KEGG Databases, and were added for our tests using standard rules for completing free valences as performed by OpenBabel [GHH<sup>+</sup>06]. Furthermore, 2550 files of the KEGG Drug set contain more than one molecule, and each molecule may appear in more than one file. To prevent a skewed analysis, we split up the data set into unique connected components. Ignoring molecules with less than 4 atoms (e.g. water), this preparation led to a test set of 7424 molecules from the KEGG Drug set.

All exact algorithms – A\*, ILP, FPT – and the inexact Antechamber approach were applied to the two test sets – MMFF94 and KEGG Drug – and the resulting assignments compared to the reference state. Please note that a comparison to other bond order assignment programs is not easily possible, since most are commercial and none of them was available to us.

### 2.9.2. Comparison to Antechamber

In order to evaluate whether solving the optimization exactly makes a difference in practice, we first focus on the following properties:

1. how often do manual, heuristic, and exact approaches produce an optimally scored solution;
2. how often do the exact approaches find a solution with a smaller *tps* than the heuristic;
3. how often does each approach fail to find a feasible solution.

Evaluation on the MMFF94 validation suite (761 molecules in total) was done as follows: the Antechamber bond perception algorithm as well as our own algorithms – A\*, ILP, and FPT – were run for each input molecule. Note that all exact algorithms will in principle compute the same solutions, and only the order of co-optimal solutions can differ. If both Antechamber and our algorithms computed bond order assignments (i.e., none of the approaches failed), we compared these to test if the Antechamber assignment is optimal.

For 734 molecules (96.45%) the solution found by the heuristic Antechamber approach is optimal. For 9 molecules (1.18%) as shown in table 2.1, the exact algorithms indeed find bond order assignments with a total penalty score less than that of the solution provided by Antechamber. For 14 cases (1.83%), our algorithms computed an optimal bond order assignment whereas the heuristic Antechamber bailed out. In 4 cases (0.53%), neither Antechamber nor our algorithms computed a bond order assignment, due to missing atom types in the penalty table. In no case, Antechamber computed a solution but our algorithms did not. In total, Antechamber bailed out in 18 cases (2.30%), and in 23 cases (3.02%) we improved upon Antechamber (no solution by Antechamber, or better solution by our algorithms).

## 2. Optimal Bond Order Assignment

molecule	score		number of optimal solutions
	heuristic	exact solvers	
DAKCEX	1	0	2
GETFIU	1	0	1
GIDMEL	2	1	7
KEWJIF	4	0	1
SAFKAL	1	0	1
JECYIZ	4	0	1
FENYIG	1	0	1
GETFOA	1	0	1
VIRBON	1	0	1

Table 2.1.: Comparison of the total penalty score (*tps*) for selected molecules of the MMFF94 validation suite where our algorithms computed bond order assignments with smaller *tps* than the assignment heuristically computed by Antechamber.

	MMFF94	KEGG
Heuristic solution is optimal	734 (96.45%)	7202 (97.01%)
Heuristic solution is sub-optimal	9 (1.18%)	15 (0.20%)
Heuristic found no feasible solution	18 (2.37%)	207 (2.79%)
Exact solvers found no feasible solution	4 (0.53%)	180 (2.42%)
Reference assignment is optimal	599 (78.71%)	6326 (85.21%)

Table 2.2.: Comparison of our exact solvers with the original heuristic implementation of Antechamber and the expert generated solutions ("reference solution") for the molecules of the MMFF94 validation set and the KEGG Drug set.



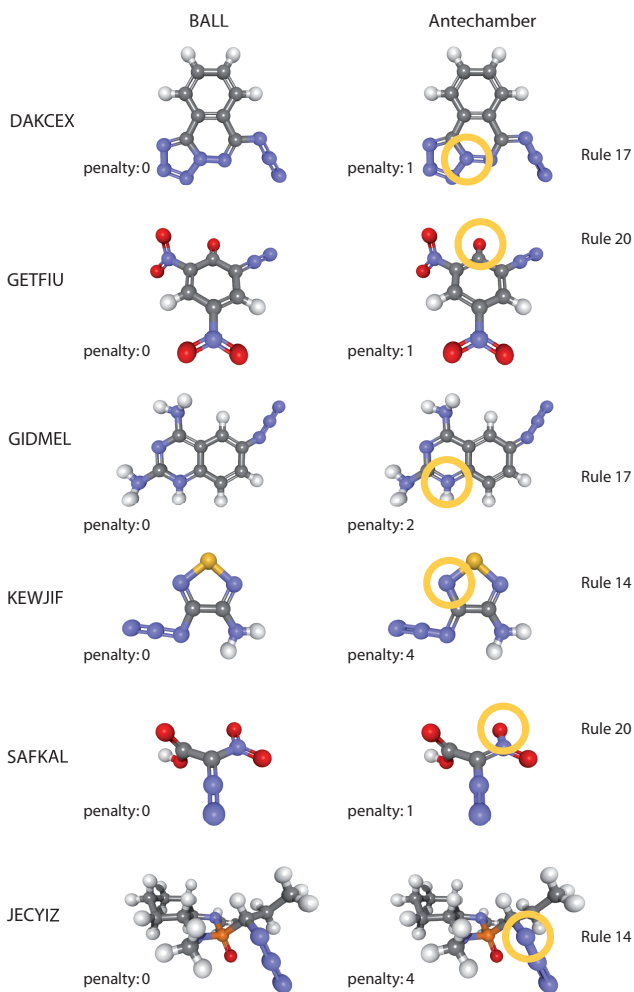


Figure 2.12.: Comparison of the bond order assignments for molecules of the MMFF94 validation suite where the first solution found by BALL has lower penalty score than the solution found by Antechamber.

## 2. Optimal Bond Order Assignment

The comparison of our algorithms to the Antechamber approach on the KEGG Drug set (7424 molecules in total) looks very similar. For 7202 molecules (97.01%), the bond order assignment found by Antechamber is optimal. For 13 molecules (0.18%) containing PO<sub>4</sub>, Antechamber reproducibly provided infeasible solutions, whereas our algorithms computed optimal assignments. For 27 cases (0.36%) our algorithms computed an optimal assignment but Antechamber bailed out. In 180 cases (2.42%), both approaches bailed out, as not all atom types are contained in the original penalty table given in [WWKC06]. In total, Antechamber bailed out in 207 cases (2.79%), and we improved upon Antechamber in 40 cases (0.54%).

A complete comparison for both test sets is given in table 2.2. Please note that the test data sets were chosen such that they are relatively well-suited to the heuristic Antechamber approach, e.g., they contain relatively few large or complex ring systems.

### 2.9.3. Comparison to Reference Assignments

As a second step in the analysis, we compare the results produced by all approaches to the reference assignment. For our own solvers, which are able to enumerate all optimal (FTP, ILP) or even all feasible solutions (A\*), we only recorded the first one for this test.

Since the MMFF94 validation suite contains 316 molecules with more than one optimal bond order assignment we also compared all optimal solutions computed by our algorithms with the "reference" bond order assignment to handle cases where our algorithms reproduced the "reference" bond order assignment, yet returned another equally scored bond order assignment on the first position.

As can be seen in table 2.3, our methods are able to reproduce significantly more bond order assignments of the MMFF94 validation suite than the original Antechamber approach. While Antechamber correctly recomputed 37.05% of the molecules, the exact methods reconstructed between 53.88% and 61.89% of the reference bond order assignments as the first solution. Similar results can be seen on the KEGG Drug set: Antechamber correctly reproduced 41.96% of the bond order assignments, compared to 49.95% and 56.9% for the exact methods. Obviously, all results returned by the exact solvers are optimal and hence, the deviations in these numbers are due to systematic differences in the order in which each algorithm enumerates the solutions. In the case of the A\* algorithm, this order can easily be tweaked by adapting the heuristic part of the scoring functions (c.f. Section 2.7.1). By design, our A\* heuristics tend to avoid the occurrences of larger bond orders, but this strategy could be further fine tuned. Note that the FPT algorithm can easily be modified to simulate this behaviour as long as the number of optimal solutions remains small, as computing all optimal solutions does not significantly increase running times. For the ILP approach, in contrast, running times would increase considerably. In the future, we plan to sort co-optimal solutions with respect to another objective function before writing the output. This might possibly further increase the quality of our results, and is the topic of ongoing research.

Considering that bond order assignments need not be unique, it makes sense to provide the user not only with the first solution, but with all optimal ones, or even sub-optimal ones. Taking all optimal solutions into account, we find that our algorithms find the reference solution in 78.71% of the cases on the MMFF94 validation suite and in 85.21% on the KEGG Drug set. A complete comparison is given in table 2.3.

Obviously, the performance of all approaches is limited by the quality of the penalty table: the definition of the atom classes, their allowed valence states, and the choice of the valence state's penalties have a significant influence on the performance. As can be seen in table 2.2, the current penalty table does not cover all molecules in the reference data sets – for four molecules in the MMFF94 set and for 180 in the KEGG Drug set, the required atom classes are missing. Hence, in our own implementations, we use SMARTS expressions stored in an XML file to define the penalty classes, which allows a user to easily add atom types, or tune the rule set to his needs. To guarantee a fair comparison between the solvers, we ensured that for all tests in this thesis, our implementation used exactly the same penalty

data set	method	reference is 1st solution	solver reproduces reference
MMFF94	Antechamber	282 (37.05%)	282 (37.05%)
	ILP	471 (61.89%)	599 (78.71%)
	A*	455 (59.79%)	
	FPT	410 (53.88%)	
KEGG	Antechamber	3115 (41.96%)	3115 (41.96%)
	ILP	4224 (56.90%)	6326 (85.21%)
	A*	3708 (49.95%)	
	FPT	3777 (50.88%)	

Table 2.3.: Performance of the original Antechamber implementation and our exact solvers on the test sets using the penalty table as defined in [WWKC06]. The third column denotes the number of molecules for which the algorithms return the original bond order assignment as first solution, the fourth column the number of molecules for which the algorithms return it at as any of their co-optimal solutions.

file	# bonds	ILP	A*	FPT	Antechamber
DEBMOM01	10	0.026	0.028	0.005	0.0045
COTMON	20	0.165	0.138	0.016	0.0055
DEDSIO	30	0.165	0.151	0.017	0.0060
DEGRIQ	40	0.469	0.422	0.115	0.0090
DUYSES	48	1.163	0.569	0.025	0.0055
BEWCUB	61	2.152	0.780	0.039	0.0075
all files	17658	252.048	227.070	24.883	7.85

Table 2.4.: Running times in seconds for computing *all* optimal solutions of selected molecules of the MMFF94 validation suite. Molecules were chosen according to their number of bonds.

classes as Antechamber where not explicitly mentioned otherwise. Improvements to the penalty table, and a systematic study of their influence, will be discussed in Section 2.9.5.

#### 2.9.4. Comparison of Running Times

As can be seen in table 2.4, computing all optimal solutions for all 761 molecules of the MMFF94 data set, the total running time was 252.0s for the ILP, 227.1s for the A\* algorithm, and 24.9s for the FPT algorithm. The Antechamber heuristic took 7.9s to compute one solution for all molecules (c.f. table 2.4). All reported running times were averaged over 20 repetitions and were measured on an Intel® Pentium® E5200 (2M Cache, 2.50 GHz, 800 MHz FSB) dual core processor with 2Gb of RAM. Thus, the ability to provide all optimal exact solutions and to use user-editable SMARTS strings for penalty class assignment takes its toll: the heuristic Antechamber approach is the fastest of the methods, about an order of magnitude faster than ILP and A\*. On the other hand, it is comparable to the running time of the FPT. Still, all running times are sufficiently small to allow the routine usage in high-throughput applications.

#### 2.9.5. Comparison of Penalty Tables

As pointed out in Section 2.4, the penalty table defines the atom classes between which the algorithm can distinguish, the allowed valence states for each such atom class, and the penalties associated with

## 2. Optimal Bond Order Assignment

data set	penalty table	method	reference is		no solution
			1st solution	optimal	
MMFF94	Wang	Antechamber	282 (37.05%)	282 (37.05%)	18 (2.36%)
		ILP	471 (61.89%)	599 (78.71%)	4 (0.53%)
		A*	455 (59.79%)		
	improved	ILP	467 (61.37%)	639 (83.97%)	0 (0.00%)
		A*	459 (60.32%)		
KEGG	Wang	Antechamber	3115 (41.96%)	3115 (41.96%)	207 (2.79%)
		ILP	4224 (56.90%)	6326 (85.21%)	191 (2.57%)
		A*	3708 (49.95%)		
	improved	ILP	4255 (57.31%)		
		A*	4391 (59.15%)		

Table 2.5.: Performance of the original Antechamber implementation, our ILP formulation, and our A\* search algorithm on the MMFF94 validation suite (top) and the KEGG Drug set (bottom) using the penalty table as defined in Wang et al. [WWKC06] and the improved BALL penalty table (see table A.1.)

each classes' valence states. Thus, the quality of the penalty table strongly influences the performance of our algorithms.

The quality of a penalty table manifests itself in its ability to reproduce natural bond order assignments. We approximate being a natural bond order assignment by its similarity to reference assignments from high quality manually curated molecular databases. To this end, we use the MMFF94 validation suite and the KEGG Drug set and compare all optimal solutions with the reference bond order assignment. The quality of a penalty table can be further specified by the following properties:

1. the number of reference molecules that are assigned optimal penalty (should be large).
2. the number of co-optimal bond order assignments that differ from the reference assignment (should be small).
3. the number of reference molecules whose reference bond order configuration was assigned a non-optimal total penalty score (should be small).
4. the bailing out rate (should be small).

Note that even in a perfect penalty table, the number of co-optimal solutions can never be zero in general, since there will be a significant amount of bond order assignments that denote the same 'true' solution due to, e.g., spatial symmetry or kekulization of aromatic groups. We thus first discuss the other three quality measures given above. An overview of these quantities, measured for the original and our novel penalty table, is given in tables 2.5 and 2.6.

It is immediately apparent that the new penalty table improves all three quality measures, even though the table was not fitted against these quantities. Instead, we merely added rules that make individual sense from a chemical perspective (c.f. Section 2.8). With these new rules, the number of sub-optimally scored reference assignments drops considerably on both data sets and consequently, the number of optimally scored ones increases. Also, the new rules allow to solve problem instances for which all solvers – including Antechamber – previously failed since penalty values were missing. To give an example of how this was achieved, Fig. A.2(a) to Fig. A.2(d) show the four molecules of the MMFF94 validation suite on which all solvers failed due to missing types, namely FE2PW3, FE3PW3, H3OPW1, and VIMHII.

table	data set	reference is		bailing out
		optimal	sub-optimal	
Wang	MMFF94	599 (78.71%)	153 (20.10%)	4 (0.53%)
	KEGG	6326 (85.21%)	850 (11.54%)	191 (2.57%)
improved	MMFF94	639 (83.97%)	117 (15.37%)	0 (0.00%)
	KEGG	7167 (96.54%)	74 (1.00%)	180 (2.42%)

Table 2.6.: Comparison between the penalty table as defined in Wang et al. [WWKC06] and the improved BALL penalty table (compare table A.1). Column three denotes the number of reference molecules that are assigned optimal penalty and column four denotes the number of reference assignments with sub-optimal score.

The remaining causes for bailing out on the KEGG Drug set are the atom types Co, Cr, Cs, Ba, As, \*, Au, Ag, Pt, Ag, Sb, La, Hg, Kr, In, Gd, Ir, Tl, Tc, Ti, Ga, Xe, B, and R (an arbitrary fragment). In principle, most of these could be easily remedied (apart from \* and R), by introducing penalty values for these types as well. However, since the rule matching of atoms to penalty classes is a time-consuming step, we decided not to include rules for these cases, which are highly unusual in drug design situations.

Having established that the new penalty tables allows application to more molecule classes than previously possible, and that it improves the quality of the solutions, we now turn to evaluate the number of co-optimal solutions.

Fig. 2.13 shows a histogram of the distribution of the number of optimal solutions on the KEGG Drug set using the penalty table as defined in Wang et al. and our new penalty table (see table A.1). As can be seen from the graph,  $\approx 60\%$  of the input structures in the test set have more than two optimal solutions. Large numbers of optimal solutions are relatively rare, however, with only  $\approx 6\%$  of the structures featuring more than five and  $\approx 1\%$  featuring more than 10 optimal assignments. Fig. 2.14 shows six molecules of the KEGG Drug set for which both penalty tables - the one by Wang et al. and our improved penalty table, produce more than 100 co-optimal bond order assignments. This is due to the fact that these structures contain a large number of aromatic rings. Each such ring has in general two possible bond order assignments due to the alternating single-double bonds. Thus, each aromatic ring multiplies - in the worst case - the number of optimal solutions by a factor of 2. Again, the new penalty table improves the situation by resolving previously degenerated cases into individual penalties, hence reducing the number of co-optimal solutions. But obviously, the combinatorial aspects cannot be avoided and the number of co-optimal solutions is still large as soon as ring systems occur.

Another interesting aspect is the distribution of rules. Fig. A.8 in Appendix A.4 shows percentage and absolute number of the most frequently used rules. Obviously, the hydrogen rule is used most often, the second rank is non-carboxylic carbon, followed by oxygen. All rules are covered at least once in the KEGG Drug set (data not shown), indicating that the tables could not be pruned further without spoiling the quality.

In summary, we can conclude that the new penalty table indeed improves the quality of bond order assignment considerably without introducing any negative side-effects. Every single quality measure improved, and the incurred cost in run-time for the small number of new rules is insignificant. We thus unconditionally recommend the new penalty table over the original one in all applications.

## 2. Optimal Bond Order Assignment

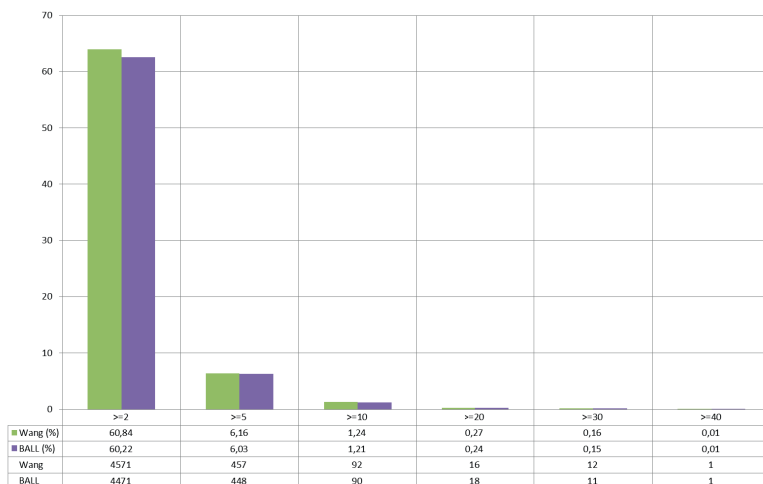


Figure 2.13.: Histogram of the distribution of the number of optimal solutions, given as absolute values and percentages, for the KEGG Drug set using the improved penalty table A.1 and the penalty table as defined in [WWKC06].

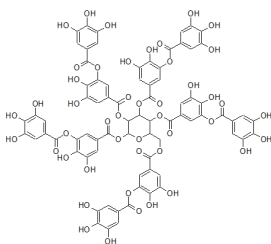


Figure 2.14.: Molecule in the KEGG Drug set with more than 100 co-optimal bond order assignments.

table	method	reference is		no solution
		1st solution	optimal	
Wang	Antechamber	282 (37.05%)	282 (37.05%)	18 (2.36%)
	A* (no FP)	455 (59.78%)	599 (78.71%)	4 (0.53%)
	A* (with FP)	472 (62.02%)		
improved	A* (no FP)	459 (60.32%)	639 (83.97%)	0 (0.00%)
	A* (with FP)	493 (64.78%)		

Table 2.7.: Influence of the fine penalty on the performance of our A\*-search algorithm on the MMFF94 validation suite using the penalty table as defined in Wang et al. [WWKC06] and the improved BALL penalty table (see table A.1).

table	method	reference is		no solution
		1st solution	optimal	
Wang	Antechamber	3115 (41.96%)	3115 (41.96%)	207 (2.79%)
	A* (no FP)	3708 (49.95%)	6326 (85.21%)	191 (2.57%)
	A* (with FP)	3561 (47.97%)		
improved	A* (no FP)	4391 (59.15%)	7167 (96.54%)	180 (2.42%)
	A* (with FP)	4168 (56.14%)		

Table 2.8.: Influence of the fine penalty on the performance of our A\*-search algorithm on the KEGG Drug set using the penalty table as defined in Wang et al. [WWKC06] and the improved BALL penalty table (see table A.1).

## 2.9.6. Incorporating Structural Information

### Fine Penalty

As a first step towards integrating bond length information into the penalty scoring scheme, we used the bond length information to reorder partial bond order assignments with equal total atomic penalty score  $tps$  in the priority queue of the A\* algorithm (c.f. Section 2.7.1). For the evaluation, we applied the A\* algorithm with and without fine penalty to the MMFF94 set. The KEGG Drug set was not used for this first test, since it does not provide 3D positions. To evaluate the impact of the fine penalty, we counted how often the reference assignment was returned as first solution. Since we assume the bond lengths in the MMFF94 set to be fairly accurate, the fine penalty should help the prediction quality in this case.

Table 2.7 shows that, for data sets with reliable atomic positions, performance improvement can be gained taking also bond length deviations into account: for the MMFF94 validation suite we see an increase from 455 (59.78%) references assignments on first position to 472 (62.02%) when using the former penalty table of Wang et al. Using our improved penalty table, we see an increase from 459 (60.32%) to 493 (64.78%).

In a second step, we tested the influence of the fine penalty on the KEGG Drug set, which does not provide reasonable atom coordinates. Our assumption, as stated in the beginning of this chapter, was that in these cases, using a structure based approach will spoil the performance. This is clearly supported by table 2.8: even with the non-intrusive addition of structural information in the form of the fine penalty, i.e., as a simple re-ordering of co-optimal solutions, performance on the KEGG Drug set decreases.

### 2.10. Summary

Automated bond order assignment is an important problem when working with user generated molecules, molecular data bases, or computational combinatorial chemistry. Especially fully automated pipelines in high throughput applications depend on reliable bond order assignments. The modern and extensible approach realized in Antechamber is based on sound chemical principles and has proven to be a very valuable tool. In this work, we have shown three different exact solvers as alternatives to the heuristic approach pursued by Wang et al. [WWKC06]: an A\* algorithm, an integer linear program formulation, and a fixed-parameter approach. Appendix A.3 also discusses two systematic approximate solvers. While we found in our evaluations that Wang's heuristic solver works surprisingly well – in roughly 97% of all cases in our tests, Antechamber computed a solution with optimal score – it still can be significantly improved using exact techniques. If we keep in mind that bond order assignments are in many cases non-unique – different resonance structures, for instance, might have the same probability to occur – the ability to further systematically enumerate all solutions becomes an invaluable tool. When bond order assignments are important, it might be worthwhile to enumerate all optimal assignments, run whatever procedure is supposed to work with the results in the next step, and average over the results.

Comparing the three different exact strategies, each of them has its advantages and disadvantages. If computational efficiency is required, the best choice is clearly the fixed parameter approach, where running times are almost on par with the Antechamber heuristic. The A\* algorithm, on the other hand, is even simpler to implement than the heuristic and can be very easily extended through the heuristic cost function. Both approaches can compute co-optimal and sub-optimal solutions without greatly increasing running times in our experiments, and geometric information can be employed to provide a more sensible ordering of the results. Such an inclusion has been prototypically performed and evaluated with the fine penalty, which provides a re-ordering of co-optimal solutions during the exploration phase of the A\* algorithm. This strategy was found to aid prediction significantly for molecules with accurate input positions, but to spoil the results otherwise. The ILP approach, finally, is relatively simple to implement when external solvers can be used. However, in our experiments, enumerating all solutions typically spoiled the running time. An additional advantage of our methods is their easy extensibility. For example, adding missing hydrogens or even bonds is possible but will require more elaborate, e.g. structure based, scoring to handle the exponential number of combinations. Such a scoring scheme only requires modifications of the *tps* definition.

Obviously, the quality of the results strongly depends on the quality of the atom type classification and the choice of the corresponding valence penalties. Expert-determined values for the scores were presented in [WWKC06] and are of great practical value, but extending the system for special applications would be highly desirable. Here, the use of BALL in our implementation proves to be very helpful: using its internal SMARTS parser, the user can easily extend the penalty table and atomic classification in order to include additional expert knowledge about the molecules of interest.

Similarly, further extensions to the algorithm are easily possible. It has been designed in a modular fashion, allowing to improve or replace each of its individual components. The implementation has been made available as open source in BALL version 1.3 [HDR<sup>+</sup>10].

### 2.11. Outlook

The achievements described in this chapter allow to address a number of new questions and interesting extensions in the field of bond order assignment.

Besides solving the bond order assignment problem efficiently and optimally, the underlying penalty table can be further investigated. In this chapter, we already described a new penalty table which we designed manually. However, training a penalty table using machine learning techniques for a given



data set promises new insights and improved performance. In addition, extending the penalty table to estimate charges and the number of attached hydrogens might further improve the performance of our approach for less reliable input structures and can be addressed by the same learning methods.

On the other hand, hydrogens pose a challenging problem that will not be entirely solvable by new penalty tables: since "missing" valences can simply be filled by adding additional hydrogen atoms, any non-optimal assignment that differs from an optimal one by smaller valences only can be made optimal through hydrogen addition. This obviously leads to an explosion of the space of co-optimal solutions and hence, probability measures for the expected number of hydrogens will be needed. One source of these could be structural information that leads to an inference of hybridization states, if atom positions are sufficiently reliable. A more promising source of probability estimates, even for structures with inaccurate or entirely incorrect atom coordinates, would be to use predicted pKa values. In an ongoing project, we investigate such a combination of a simple pKa predictor with BOA Constructor. A different area of future work concerns the structural scoring terms, for which we so far introduced a simple bond length criterion. This simple function can be extended to encompass angles as well. These might allow to reliably distinguish hybridization states, e.g.,  $109^\circ$  of  $sp^3$  and  $120^\circ$  of  $sp^2$ , which would be highly useful information. To fully integrate the new score, the weighting function allowing a linear switch between pure and mixed scores would have to be adapted as well.

A fascinating future perspective for molecular modelling that is enabled by our work is the use of ensembles of resonance structures: since in many cases, all assignments of optimal penalty are equally likely, not only according to our penalty table but also to the expert, returning a single representative is dubious at best. In such situations, working with an ensemble of co-optimal solutions might lead to more stable and more accurate results, e.g., in a docking context, where the flexibility of individual groups as determined by the bond order assignment has a strong influence on the outcome.

Aside from the bond assignment problem itself, we believe that the methodology introduced in this work might help in entirely different fields of structural bioinformatics through the use of our tree decomposition. While such decomposition algorithms have been available for some time – indeed, our own implementation is based on the QuickBB library – BALL greatly simplifies their application to molecular systems. In addition, the nice tree decomposition returned by our approach is often much simpler to use than a more general one.

Algorithmically, the bond order assignment problem bears close resemblance to the side chain optimization problem, where similar solution strategies have been developed ([AKLM02, LL98, XJB05]). Future work will study whether modern probabilistic approaches (see, e.g. [YSFW08]) for this problem will also be appropriate for bond order assignment.



## 3. NMR Shift Prediction

### 3.1. Introduction

Nuclear Magnetic Resonance (NMR) chemical shift prediction has developed into a valuable tool for computational structural biology and biomolecular NMR spectrometry. Out of all structures in the Protein Data Bank (PDB), about 88% were resolved by an X-ray experiment and about 11% by solution NMR, counted at the time of writing (Q1 of 2011). While the number of NMR resolved structures is significantly smaller than the number of X-ray resolved ones, many of these structures could not have been resolved by any other method but NMR (e.g. [BAM<sup>+</sup>98]). While X-ray crystallography requires crystallization of the protein of interest, NMR experiments allow to study protein structures as well as their dynamics under nearly physiological conditions. NMR is routine for small proteins up to 15kDa, yet problems arise with increasing protein size, and NMR typically yields lower resolution compared to X-ray.

The experimental details of NMR will be introduced in Section 3.2. Here, we will just present a short description. The idea of NMR spectroscopic experiments is to measure magnetic properties of atom nuclei to gain information about the topology (which atom is bound to which other atom) and the structure (which atom is close to which other atom) of a molecule. The experiment actually measures the properties of only a selection of atoms, all at the same time. The nature of this selection will be covered later in Section 3.2.2. An individual atom's response in this experiment is denoted its "chemical shift", the collection of all chemical shifts plus some experimental noise the "NMR spectrum". In practice, usually combinations of NMR responses of different atom types are recorded, leading to multi-dimensional NMR measurements, such as the  $^1H$ - $^{15}N$  HSQC spectrum shown in Fig. 3.3. However, for the purposes of chemical shift prediction, the differences between one and higher dimensional NMR are irrelevant. Hence, we will not discuss multi-dimensional NMR in this work.

The prediction of the chemical shifts based on structural information is known as the *shift prediction problem* and the work described in this chapter aims at protein chemical shift prediction. A related problem is the *shift assignment* where we are given an experimentally measured NMR spectrum and are interested in mapping individual atoms to the spectrum's chemical shifts. Fig. 3.1 illustrates the terms.

The chemical shifts depend very sensitively on the three-dimensional structural details. While this renders their prediction a formidable task, it simultaneously makes them a very valuable source of structural information. Hence, historically<sup>1</sup>, the first aim of chemical shift prediction was structure elucidation: if sufficiently accurate and complete shift prediction rules were known, this would mean that the physical effects influencing the spectra would be understood. This would then allow for a full theoretical interpretation of NMR spectra. For example, a certain shift value might be understood to be related to a particular torsion angle, so that from its value, the torsion angle might be directly inferred. Alternatively, the structure could be solved by optimizing its positions such that simulated and measured shifts were in optimal agreement.

---

<sup>1</sup> For a more complete discussion of the history of chemical shift prediction, the interested reader is referred to the review by Wishart [Wis11].

### 3. NMR Shift Prediction

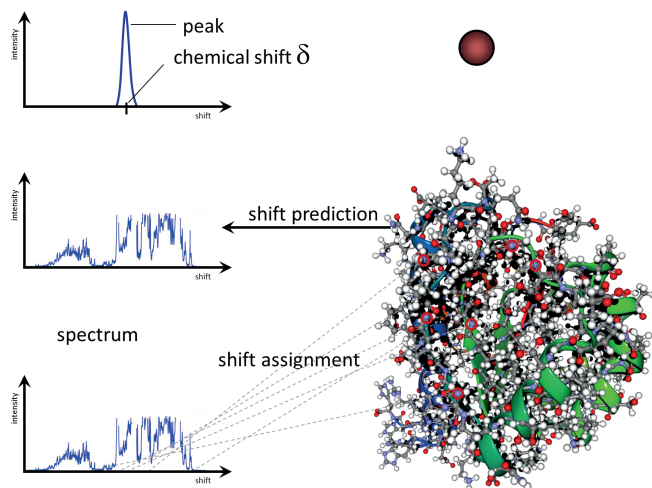


Figure 3.1.: The NMR chemical shift of a single atom and a molecule, the resulting NMR spectra, and the chemical shift prediction and assignment problem. The protein spectrum was taken from [RSH<sup>+</sup>10] with kind permission from Springer (c.f. C) and resulted in PDB entry 2KLB.

But those efforts towards structure elucidation by NMR shift prediction were overtaken by methods based on the Nuclear Overhauser Effect<sup>2</sup> (NOE). This effect allows to derive distance constraints from multi-dimensional NMR spectra, which can be used to solve protein structures more effectively in a distance geometry setting.

However, NOE-based approaches turned out to have some drawbacks as well: distance constraints offer only indirect information, the NOE experiments are error prone and time consuming, and they impose a size constraint, as for proteins larger than 200 amino acids, the complexity is usually too high to be solvable. The most obvious disadvantage, however, is that NOE-based methods are not fully automated and are useless without shift assignment. On the other hand, such shift assignment can greatly profit from accurate shift prediction. Hence, even NOE-based methods rely to some degree on the availability of shift prediction methods.

Although chemical shift prediction is thus currently not the method of choice for direct structure elucidation, the great sensitivity of the shifts to structural variations has led to a variety of applications in different contexts that require structural information. In the past, shift prediction has been successfully employed in fields as diverse as structure determination [Wil90, WSR91, KCB04, BTL<sup>+</sup>09], structure optimization [LdDO93], shift referencing [WBY<sup>+</sup>95, ZNW03, Wan10], molecular flexibility [BW05, BW06, BW08], and protein-protein docking [KBM<sup>+</sup>01, MCS<sup>+</sup>08, CMV11]. Particularly important was the recent discovery that chemical shifts can be used to produce highly resolved structures of globular proteins [CSDV07, SLD<sup>+</sup>08, WAB<sup>+</sup>08]. Chemical shift prediction also enables a

<sup>2</sup> A further description of the Nuclear Overhauser Effect would require a lengthy discussion of the theory behind NMR without any further crucial insights for the purposes of this work.

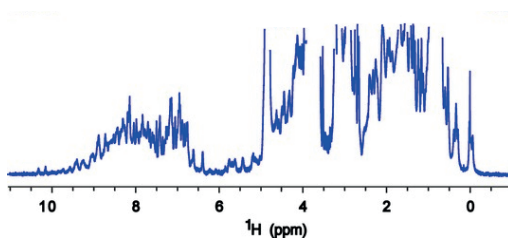


Figure 3.2.: 1D  $^1\text{H}$  NMR spectra of NsR431C as published in [RSH<sup>+</sup>10] with kind permission from Springer (c.f. C).

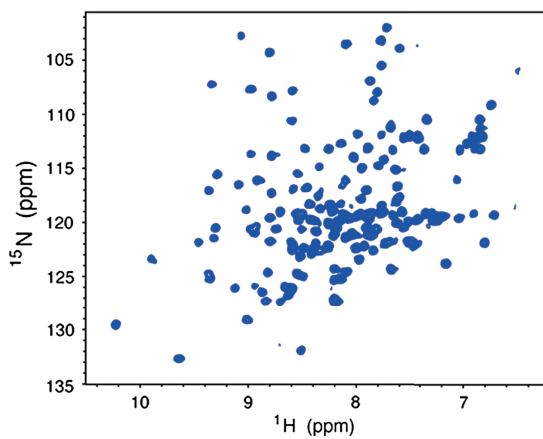


Figure 3.3.: 2D  $^1\text{H}$ - $^{15}\text{N}$  HSQC spectrum of NsR431C as published in [RSH<sup>+</sup>10] with kind permission from Springer (c.f. C).

### 3. NMR Shift Prediction

technique known as chemical shift perturbation [LMP04], where a ligand is titrated with a protein and a series of NMR spectra is recorded. Predicting the protein's chemical shifts and assigning them to the spectra, the atoms which are influenced by the ligand can be determined, yielding the interface of the protein-ligand complex.

The application we are mostly interested in this work is protein-ligand docking. In 2001, Kohlbacher and coworkers developed a scheme for using experimentally obtained one-dimensional  $^1\text{H}$  NMR spectra in a protein-protein docking context [KBM<sup>+</sup>01]: chemical shifts for the putative complexes generated by the docking procedure were predicted and converted into a spectrum using a simple sum of Gaussian or Lorentzian curves. The spectral similarity to an experimentally obtained spectrum was then used as a scoring function in the docking process. In a proof-of-concept study, it was shown that the method is in principle able to distinguish between low- ( $< 5\text{\AA}$ ) and high-rmsd ( $> 10\text{\AA}$ ) structures. In [Deh07], we extended this scheme by implementing shift prediction for  $^1\text{H}$ ,  $^{13}\text{C}$ , and  $^{15}\text{N}$  and by testing on further scenarios.

Montalvao et al. [MCS<sup>+</sup>08, CMV11] used a strategy very similar to [KBM<sup>+</sup>01] albeit with the more recent CamShift shift prediction program [KRC<sup>+</sup>09].

However, none of the previous approaches is applicable to the field of protein-ligand docking: for reasons we will describe in more detail later in this chapter, shift prediction usually works on either small molecules such as ligands, or pure protein- or DNA-molecules, but not on complexes of these. In this work, we lay the foundation for extending NMR-based scoring to protein-ligand docking. Our idea is to measure the influence of the ligand atoms onto the protein atom shifts. This influence is due to the ligand atoms as well as their bonds. However, the effects of ligand atoms on the protein shifts are subtle, and hence, we need to solve two problems: first, we need to improve the quality of the pure protein models themselves, and second, find a way to include ligand information into the prediction.

The development of novel NMR chemical shift prediction techniques is a challenging task. Previous approaches either focus on full quantum mechanical ab-initio models (e.g. [HK64, XC01, OKK04, FOME11]) which are computationally very expensive, or settle for approximations borrowed from classical physics [ÖC94, WA93, KBM<sup>+</sup>01, NNZW03]. As a third option, prediction techniques can use statistical models based on semi-classical, structural, or sequential features of the proteins (e.g. [Mei03, NNZW03, AL06, KRC<sup>+</sup>09, AAFA10]). For medium- to high-throughput applications, the most successful approaches today offer good prediction accuracy with relatively low computational cost by combining semi-classical and statistical approaches. These techniques are known as *hybrid* methods.

Developing a new hybrid method, or extending an existing one, is a hard and complex task for which three questions have to be addressed: (a) which data set can be used to train a model, (b) which features should be included into it, and (c) which statistical technique should be employed.

The question of the data set in particular is a very difficult one. The required information for creating such a data set is spread over several data bases, such as the Biological Magnetic Resonance Bank (BMRB) [UAD<sup>+</sup>08] and the Protein Data Bank (PDB) [BWF<sup>+</sup>00, BHN03] and is stored in different, notoriously hard-to-parse, file formats. To make matters worse, real-life data sets often contain serious syntactical, semantical, and logical errors or inconsistencies [ZNW03, GGCH07, RV10, Wan10, HLGW11, Wis11].

Due to these complications, most former approaches rely on hand-curated data sets created by the application of non-standardized sequences of restriction and correction methods.

Another challenge when training prediction models is the choice and computation of the semi-classical terms and the structural and sequential features to learn from. Computing these terms and molecular features correctly, reliably, and efficiently requires complex molecular data structures and algorithms. In this work, we present an extensible automated pipeline, called *NightShift – NMR shift Inference by General Hybrid model Training* for data set generation and training of hybrid NMR chemical shift prediction methods. Several semi-classical terms for shift prediction are implemented and readily available. As of now, we include random coil contributions, aromatic ring current effects, electric

field contributions, and hydrogen bonding effects. In addition, the feature set for the training of the statistical term encompasses sequential, structural (angles, surface, and density), force-field based, and experimental properties. All features are computed using our open source library BALL [HDR<sup>+</sup>10], and can be easily extended.

Based on recent research [AL06, HLGW11], we propose a random forest model for the statistical contribution which in our experiments has demonstrated to yield very accurate and stable results. In general, however, the pipeline is model-agnostic and can be used with any regression technique implemented in R [R D11].

The first result of NightShift is a pure protein NMR shift prediction model called *Spinster - Single Protein NMR Shift deTERmination*.

We further investigated a first model for protein shift prediction in the presence of ligands called *Liops - Ligand Influence On Protein Shifts*. To this end NightShift itself was extended to handle a combination of a pure protein model, our Spinster model, and several ligand-related features. For the definition of such features that capture relevant information about the diverse chemistry of possible protein ligands, we relied on our work on atom- and bond-typing, as described in Chapter 2. In particular, we made use of the information encoded in the GAFF atom types computed by the procedures described earlier. The organization of this chapter is as follows: in Section 3.2, we give a brief introduction into NMR spectroscopy. Section 3.3 introduces the state of the art in chemical shift prediction, while Section 3.5 discusses a number of relevant issues that guided us while creating our pipeline and designing our models. Finally, in Section 3.6, we present the pipeline NightShift, the data set, and the evaluation of our Spinster model. A first proof-of-principle application to the protein-ligand case is described in Section 3.7 before we close this chapter with questions that can be further addressed given our research.

Most of the results of this chapter have been disseminated in peer-refereed publications: our work on the NightShift pipeline and the pure protein model Spinster has been presented at the 25<sup>th</sup> Molecular Modelling Workshop 2011 in Erlangen, Germany. A manuscript on this work has also been submitted. The protein-ligand model Liops was presented at the German Conference on Computational Biology (GCB) 2011 in Weihenstephan, Germany [DLH11]. Further, our work has been the topic of a scientific poster at the joint 19<sup>th</sup> Annual International Conference on Intelligent Systems for Molecular Biology and 10<sup>th</sup> European Conference on Computational Biology (ISMB/ECCB) 2011 in Vienna, Austria.

## 3.2. NMR Spectroscopy

Nuclear magnetic resonance (NMR) experiments manipulate the nuclear spin of a molecular system in order to deduce information about its chemical composition and three-dimensional arrangement. The nuclear spin is a magnetic property and to understand its meaning we need to familiarize ourselves with the principles of magnetic fields.

### 3.2.1. The Magnetic Field

A number of charged particles moving in direction  $\frac{\mathbf{I}}{|\mathbf{I}|}$ , are called an electric current  $\mathbf{I}$ , where the magnitude  $|\mathbf{I}|$  of  $\mathbf{I}$  describes the amount of charge traversing a plane perpendicular to  $\mathbf{I}$ . For the remainder of this section, we use following notation  $I := |\mathbf{I}|$ .

The current  $\mathbf{I}$  always creates a magnetic field  $\mathbf{B}$ , a vector field that is perpendicular everywhere to  $\mathbf{I}$ , i.e., it is located in the plane with normal  $\mathbf{I}/I$ . If the current follows a straight line, for instance, if we consider a long straight piece of wire, the field lines, i.e., the tangent curves, to  $\mathbf{B}$  are circles around  $\mathbf{I}$ .

### 3. NMR Shift Prediction

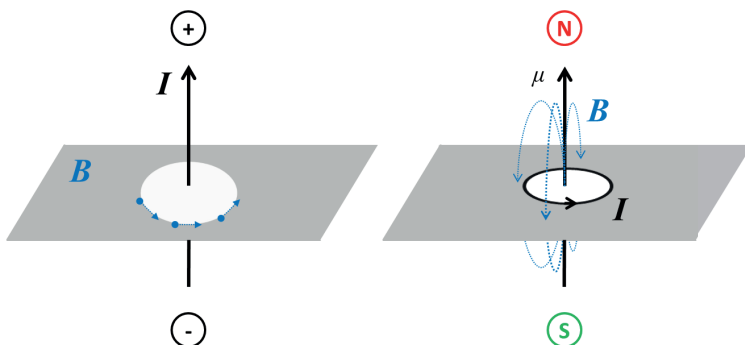


Figure 3.4.: **Left:** Linear current  $I$  induces a magnetic field  $B$  perpendicular to  $I$ .

**Right:** Circular current  $I$  induces a magnetic dipole field  $B$  with magnetic moment  $\mu$ . Note that  $A$  and  $\mu$  point into the same direction.

At a point with distance  $r$  from the wire, the strength  $B$  of  $B$  will be given by  $B = \frac{\mu_0 I}{2\pi r}$ , where  $\mu_0 = 4\pi \times 10^{-7} T \frac{m}{A}$  is the vacuum permeability, and where  $B$  is measured in Tesla (T).

If we could force the current to run in a circle itself, the superposition of the magnetic fields created in each point of this circle would lead to the well-known field of a magnetic dipole (c.f. Fig. 3.4 right). This magnetic dipole can be described by a vector  $\mu$ , the so-called magnetic dipole moment or, in brief, magnetic moment. To define  $\mu$ , we imagine a vector  $A$  with its origin in the center of the circular current and a direction perpendicular to the circle. Whether the vector points upwards or downwards is decided by the right hand rule, i.e., the thumb of the right hand shows the direction of  $A$  if the other four fingers point into the direction of the circular current. As magnitude of  $A$ , we use the area enclosed by the circular current,  $\pi r^2$ . Then, the magnetic moment  $\mu$  is given by  $\mu := IA$ . If a magnetic dipole is placed in an external magnetic field of strength  $B$ , it feels a torque  $\tau = \mu \times B$ . This torque vanishes only if the dipole moment  $\mu$  is parallel or antiparallel to  $B$ , i.e., the torque will align the magnetic dipole with the magnetic field. The antiparallel alignment is energetically unstable, i.e., if it occurs, even small thermal distortions will lead to a torque that will finally yield a parallel alignment.

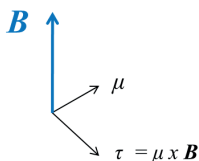


Figure 3.5.: Torque  $\tau$  of a magnetic field  $B$  and magnetic moment  $\mu$ .



If two magnetic dipoles are located close to each other, they will feel their respective magnetic fields. Thus, the influence of each magnetic dipole on the other is given by a torque so that both dipoles will rotate until they are aligned.

### 3.2.2. The NMR Experiment

Experimentally, it has been shown that subatomic particles such as electrons or protons show behaviour that is consistent with that of small magnetic dipoles, i.e., they create a corresponding magnetic field and they react to magnetic fields just as a dipole would. Intuitively, we could imagine these particles to move on a circle incessantly. While this picture is physically impossible, the real reason for the existence of these magnetic moments lies deep in the combination of quantum physics with the theory of relativity, and cannot be discussed here. Instead we will just presume its existence. This magnetic moment of a subatomic particle – in the above picture, the moment introduced by the circular motion of the particle – is known as its *spin*. Not only electrons feature spins, but also nucleons.

In nuclear magnetic resonance (NMR) spectroscopy, the behaviour of these nuclear spins is measured to infer knowledge about their chemical environment. But not all atoms have a total nuclear spin: according to Pauli's principle (which also holds for nucleons), nuclear particles try to pair in antiparallel spin direction. Thus, the total nuclear spin can only differ from zero (each pair of parallel and antiparallel spins adds up to zero) if the nucleus contains an odd number of protons. For  $^1\text{H}$ , this obviously holds, but for  $^{12}\text{C}$ , for instance, it does not. Hence, NMR on carbon atoms can only succeed if the carbon is exchanged with its  $^{13}\text{C}$  isotope. Similarly,  $^{14}\text{N}$  will be exchanged by  $^{15}\text{N}$ .

In principle, each NMR experiment follows the same scheme: using a strong uniform external magnetic field  $\mathbf{B}$ , the nuclear spins are aligned into the same direction. In practice, only a small fraction will align due to random thermal movements. Then, energy is injected into the system in the form of an electromagnetic wave, which will excite some of the spins: they will flip from the energetically favorable parallel alignment to the unstable antiparallel one if the energy required for the flip equals the energy provided by the wave. Since this state is unstable, the spins will flip back again. The energy that is emitted during this process is measured as the response of the system and converted into a difference, the so-called *chemical shift*  $\delta$ , from the responses in a well-defined standard chemical environment. Since the energy of the electromagnetic wave  $E = h\nu$  is proportional to the frequency  $\nu$ , the shift can be computed as

$$\delta = \frac{\nu_{\text{probe}} - \nu_{\text{standard}}}{\nu_{\text{standard}}} \quad (3.1)$$

where  $\nu_{\text{standard}}$  denotes the responses in the standard chemical environment. All shifts of a sample yield the NMR spectrum, which shows the measured intensity as a function of the chemical shift (in practice, two-dimensional spectra are often constructed by exciting different atoms at the same time; however, this is out of scope of our current treatment). In a real-world spectrum, each of these shifts leads to a curve of finite width, a so-called peak. In addition to these peaks – which correspond to the useful information content – the spectrum contains so-called 'parasitics' such as high-frequent noise or low-frequent baseline that do not correspond to the chemical shift of any atom in the system. Instead, they occur due to imperfections in the measurement process.

In the case of NMR based protein structure prediction, which has to cope with such imperfect real-world spectra, the next step is thus to identify peak positions. The shift values corresponding to these peaks (in the simplest case, the median value of the peak) are then mapped to atoms in the protein (the assignment), and these mapped shifts are then stored in a so-called NMRStar file [HC95] and uploaded to the Biological Magnetic Resonance Data Bank (BMRB) [UAD<sup>+</sup>08].

#### 3.2.3. Chemical Shift Contributions

The chemical shifts strongly depend on the chemical environment in a three-dimensional neighbourhood of each atom, and hence, they can be used in turn to infer information about this neighbourhood and all derived quantities. In particular, it is possible to derive distance constraints from the relationships between different shifts, and this can be used to solve the structures of proteins.

The chemical environment influences the shift for a number of reasons: as mentioned before, other subatomic particles have the magnetic spin property as well. Thus, in addition to nuclear spins, all atoms have electronic spins as well. Since the electrons are far away from their nucleus, the direct effect of their magnetic moment on the nuclear spins is relatively small. However, the nucleons experience magnetic fields and magnetic moments of electrons as well. If two atoms form a chemical bond (covalent or ionic alike) they share certain electrons with their binding partner. In contrast to 'regular' electrons, these bonding electrons change their spatial distribution relative to their nuclei. Thus, the resulting magnetic field seen by a bonded nucleon differs from that of an unbound one.

The presence of ions or large partial charges in close spatial proximity to a bond, or electronegative or electropositive substituents of the bonding atoms, influence the bonding electrons further and yield an unequal electron distribution between both bond partner atoms: positive charges attract electrons while negative charges repel them. For carbon atoms, the hybridization state affects the chemical shift as well, since for different hybridization states, the electrons distribute differently over the valence shells.

Similar to chemical charges, hydrogen bonds affect the nuclear spin. In a hydrogen bond, a proton is shared between a hydrogen donor and an acceptor atom, e.g., a nitrogen of a peptide bond might share its hydrogen with a second peptide bond's oxygen. Unfortunately, while it is intuitively clear (and supported by experiment) that such hydrogen bonds have an impact on the nuclear spin, this effect seems less well understood than most of the others.

Another common source for magnetic fields that often occurs in proteins are aromatic rings. The carbon atoms of an aromatic ring have delocalized  $\pi$ -electrons that form circular clouds parallel to the ring carbon atom plane and that induce an aromatic ring current. The ring current's magnetic field is directed perpendicular to the carbon atom plane and may change the influence of the external magnetic field onto the ring atoms and close neighbors.

In the following, we want to discuss some of the models that have been developed in the past to describe individual contributions to chemical shift values in proteins. The models will be at the semi-classical level, which means that they are approximate classical models of effects that are known from a full quantum-mechanical treatment. At this point, we want to address an aspect of the terminology that commonly leads to confusion: in chemical shift prediction, the term "ab-initio" refers to methods computing shifts from physical theories, including the semi-classical ones. In quantum mechanics, on the other hand, the term "ab-initio" refers to methods that directly solve the Schrödinger-equation without resorting to heuristics or empirically derived parameters, such as semi-empirical methods. In this thesis, we will try to minimize confusion by using the term "semi-classical" instead of "ab-initio" exclusively.

#### Semi-Classical Models for NMR Chemical Shift Contributions

Although the concept of magnetic spin is far from being completely understood at a molecular level, some quantum mechanical phenomena are well-characterized by classical equations. The semi-classical terms for NMR chemical shifts encompass the following: random coil contributions, the ring current, electric field, magnetic anisotropy, and hydrogen bond effects. In the description of these semi-classical terms, we closely follow the presentation given in our previous work [Deh07].

**Random Coil** The *random coil contribution* or *reference* chemical shift  $\delta_{\text{coil}}$  can be understood as an intrinsic shift value mainly dependent on the amino acid type. In [WBH<sup>+</sup>95], it is defined as “the experimentally measured chemical shift of an amino acid residue within a peptide, which is free to access all sterically allowed regions of its conformational space”. Traditionally, peptides with sequence Gly–Gly–X–Ala or Gly–Gly–X–Gly–Gly or short random coil sequences are chosen, as they remain unstructured under varying solvent conditions and prevent neighboring residues from generating steric perturbations in the measured residue X.

**Ring Current** In aromatic rings, found in the amino acids as found in the amino acids Phe, Tyr, His, and Trp (Trp contains two aromatic rings, Trp<sub>1</sub> and Trp<sub>2</sub>), the overlapping delocalized p-orbitals form so-called  $\pi$ -bonds above and below the ring. In general, movement of the electrons through these  $\pi$ -bonds will have no preferred direction and hence, the total current through the ring will be zero. If a magnetic field is applied, however, the electrons will tend to rotate in a clockwise or counter-clockwise direction, where the strength and direction of the rotation are determined by the component of the magnetic field normal to the ring plane. This then gives rise to circular current, called the *ring-current*, which induces its own magnetic field in the direction of the axis of rotation, according to the principles laid down in Section 3.2.1.

Due to the small number of electrons involved, the ring current is small in magnitude and hence, the induced magnetic field is much smaller than the external field that created the circular motion. Thus, it is only of local relevance and decays faster than many of the other terms: it only affects hydrogen, carbon, and nitrogen atoms (e.g. H <sup>$\alpha$</sup> , H <sup>$N$</sup> , C <sup>$\alpha$</sup> , C <sup>$\beta$</sup> , C', and N atoms), the so-called target atoms, in close spatial proximity.

In the following, we focus on the approaches that have been used by chemical shift prediction models (c.f. Section 3.3): the method of Haigh and Mallion [HM72, HM79], used in the ShiftX approach, the Johnson–Bovey model [JJB58] used by Kohlbacher and coworkers, and a classic point–dipole model [Pop58] used by the CamShift approach.

As presented in [KRC<sup>+</sup>09], the CamShift model for the ring current contribution can be written as:

$$\delta_{\text{ring}} = \sum_i \left( \alpha_i \sum_{k \in \text{rings}(i)} \left[ \frac{1 - 3 \cos^2(\theta_k)}{r_k^3} \right] \right) \quad (3.2)$$

where  $i$  iterates over the five different ring types (Phe, Tyr, His, Trp<sub>1</sub>, and Trp<sub>2</sub>),  $\alpha_i$  is a fitting parameter of the CamShift model,  $\text{rings}(i)$  denotes the set of all rings of type  $i$  in a protein,  $\theta_k$  denotes the angle between the vector perpendicular to the ring plane and the vector connecting the ring center and the target atom,  $r_k$  the distance between target atom and the ring center.

In its functional form, this approach is similar to the Haigh–Mallion method, which in turn is closely related to the Johnson–Bovey model. These latter two models can be formalized as:

$$\delta_{\text{ring}} = \sum_{k \in \text{rings}} I_k B G(\mathbf{R}_k) \quad (3.3)$$

where  $I_k$  denotes a ring specific intensity factor of the  $k$ -th aromatic ring,  $B$  a specific constant of the target atom's type,  $\mathbf{R}_k$  gives the distance of the target atom from the ring center, and  $G(\mathbf{R}_k)$  is a geometric factor. The difference between the Haigh–Mallion and the Johnson–Bovey model lies in the definition of the geometric factor  $G(\mathbf{R}_k)$ . In the Haigh–Mallion model, the factor is derived from an approximate solution of the quantum mechanical treatment, leading to

$$G(\mathbf{R}) = \sum_{i < j} S_{i,j} \left( \frac{1}{r_i^3} + \frac{1}{r_j^3} \right) \quad (3.4)$$

### 3. NMR Shift Prediction

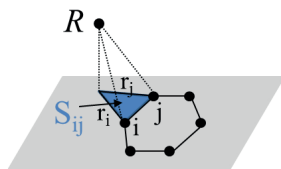


Figure 3.6.: Geometric parameters of the geometric factor  $G$  as defined in the Haigh–Mallion model.

where  $i$  and  $j$  iterate over all ring members in the sum,  $r_i$  and  $r_j$  give the distances of a target atom - projected into the ring plane - from two neighboring atoms  $i$  and  $j$  of the aromatic ring, and  $S_{ij}$  denotes the area of the triangle (projected target atom, atom  $i$ , atom  $j$ ). Fig. 3.6 shows the geometric parameters needed in more detail, table B.1 gives the intensity factors and table B.2 the constant factor values per target atom type.

The Johnson–Bovey model, on the other hand, follows a semi-classical approach: here, quantum mechanics is used to derive the intensity of the ring current, but the result is then inserted into classical electrodynamics, leading to

$$G(\mathbf{R}) = \frac{1}{\sqrt{(a^2 + \rho^2)^2 + z^2}} \left\{ K + \frac{a^2 - \rho^2 - z^2}{(a - \rho)^2 + z^2} E \right\} \quad (3.5)$$

Here,  $\mathbf{R}$  is the distance of the target atom from the ring center,  $\rho$  and  $z$  are the components of  $\mathbf{R}$  in cylindrical coordinates,  $a$  is the radius of the aromatic ring,  $K$  is the complete elliptic integral of the first and  $E$  that of the second kind (c.f. [AS64]).

**Electric Field** As described in Section 3.2.3, polar groups in the spatial proximity of a nucleus influence the magnetic field experienced by that nucleus, a phenomenon that is called *electric field effect*. Roughly speaking, the electrons in a bond can move along the bond vector and thus come closer to or move farther away from the nucleus we are currently interested in. Positive charges in the area will attract the electrons while negative ones will repel them. A charged atom influencing an atom in this way is called the *source atom*, the influenced atom is called the *target atom* of the electric field effect. For each charge  $q$  in the vicinity, we can compute the resulting electric field at the position  $\mathbf{R}$  of the target nucleus as (remember that quantities denoted in bold face are vector valued)

$$\mathbf{E}(\mathbf{R}) = \alpha \frac{q}{d^2} \frac{\mathbf{d}}{d} \quad (3.6)$$

where  $\mathbf{d}$  is the distance vector between charge and target nucleus, and  $\alpha$  is a constant prefactor. The effect on each bond of the target atom can now be computed independently of each other. Assuming without loss of generality that the currently considered bond is aligned with the z-axis, projection of the electrostatic field onto the bond vector yields

$$E_z = |E| \cos(\vartheta) \quad (3.7)$$

where  $\vartheta$  is the angle between field and bond. This then gives a measure for the direction into which the electrons will want to move.

Using this formula, Buckingham [Buc60] approximated the effect on the chemical shift of an atom due to one of its bonds as follows:

$$\delta_{\text{EF}} = \varepsilon_1 E_z + \varepsilon_2 \|\mathbf{E}\|^2 \quad (3.8)$$

For each atom, the effects of each of its bonds are then simply added up to yield to total electric field contribution.

Often, the fields involved are quite small, and hence, the quadratic term in the above equation is often neglected by setting  $\varepsilon_2 = 0$ , yielding in total

$$\delta_{\text{EF}} = \frac{q \varepsilon_1 \cos(\theta)}{d^2} \times 10^{22} \quad (3.9)$$

where  $\varepsilon_1$  equals  $1 \times 10^{-12}$ ,  $q$  denotes the partial charge associated with the source atom,  $\theta$  denotes the angle formed by each triple (source atom, target atom, target atom partner), and  $d$  denotes the distance between source and target.

While most authors keep the functional form due to Buckingham, different parameterizations have been proposed, e.g. Kohlbacher et al. [KBM<sup>+</sup>01] mentions [OC94, WA93]. As a result, Kohlbacher and coworkers and Wishart and coworkers [NNZW03] use the same formula for the electric field effect, but with different values for the charges  $q$ : Kohlbacher et al. use values taken from the Amber 94 force field [CCB<sup>+</sup>95], and Wishart et al. use  $-0.9612 \times 10^{-10}$  esu for O, OD, and OE nuclei,  $1.3937 \times 10^{-10}$  esu for all C nuclei and  $0.7209 \times 10^{-10}$  esu for N nuclei, where the target atoms are all C<sup>α</sup> and all H atoms.

**Magnetic Anisotropy** Some, but not all, approaches try to model the *magnetic anisotropy* of certain chemical groups as well. Magnetic anisotropy describes the situation where the magnetic field is not only shielded but also rotated due to the presence of a certain chemical group, e.g., the peptide group. While the ShiftX method neglects anisotropy, Kohlbacher and coworkers used the model proposed by McConnell [McC57], which uses an approximation of the peptide bond's magnetic susceptibility tensor  $\chi$  to yield

$$\delta_A = \frac{1}{3 N_A r^3} \sum_{i=x,y,z} \chi_{ii} (3 \cos^2(\theta_i) - 1) \quad (3.10)$$

where  $r$  denotes the distance between the target atom (in principle all nuclei are affected, but the Kohlbacher approach is limited to hydrogens) and the anisotropic bond,  $N_A$  is Avogadro's constant, and  $\theta_i$  is the angle between the distance vector  $r$  and the  $i$ -th axis. The parameters used by Kohlbacher et al. were those proposed by Williamson and Asakura [WA93].

**Hydrogen Bond Effect** Similar to the anisotropy term, the effects of *hydrogen bonds* are often neglected as well. A hydrogen bond is a bond-like phenomenon, where the proton of a hydrogen bond to a so-called donor atom tunnels between the potential wells formed by the electrons of its donor and an acceptor atom in close spatial neighbourhood. Most of this effect is already captured by the electric field effect. However, some authors still prefer to include an implicit hydrogen bond contribution. For instance, Wishart and collaborators [NNZW03] use the model by Wagner et al. [WPW83] and Wishart et al. [WSR91] to improve the prediction for the influence on H<sup>α</sup> and H<sup>N</sup> atoms in their ShiftX program. For H<sup>N</sup>, the resulting formula reads

$$\delta_{\text{HB}} = \frac{0.75}{r^3} - 0.99 \quad (3.11)$$

where  $r$  is the distance between hydrogen and oxygen. For H<sup>α</sup> atoms, the corresponding formula is

$$\delta_{\text{HB}} = \frac{15.69}{r^3} - 0.67 \quad (3.12)$$

### 3. NMR Shift Prediction


complexity	model	approach
high	Quantum Mechanics	ShiftS
	Hybrid	ShiftX, ShiftX2, Sparta, CamShift, Arun et al.
	Semi-classical	Kohlbacher et al.
	Statistical	Proshift, BioShift
	low	Sequence Homology

Figure 3.7.: Comparison of model complexities for selected NMR chemical shift prediction methods.

Interestingly, the term implemented in ShiftX differs significantly from the original ShiftX model as described in the ShiftX publication [NNZW03]. Instead, the implementation uses

$$\delta_{\text{HB}} = \begin{cases} \frac{c_1}{r^3} + c_2 & \text{for H atoms} \\ \frac{c_3}{r} + \frac{c_4}{r^{\frac{3}{2}}} + \frac{c_5}{r^2} + \frac{c_6}{r^{\frac{5}{2}}} + \frac{c_7}{r^3} + c_8 & \text{for H}^\alpha \text{ atoms} \end{cases} \quad (3.13)$$

where  $r$  denotes the distance between acceptor and hydrogen, and  $c_i, i = 1 \dots 8$  are given in table B.4. A more complex model has been proposed by Morozov and coworkers [MKTB04] and is used, e.g., by CamShift. However, this model is correlated with the electrostatic term used in ShiftX and by Kohlbacher and coworkers and thus should not be simply used in combination with these terms.

With these insights into NMR spectroscopy, and into models for some of the contributions to chemical shifts, we can now turn to the question of how to predict chemical shifts for atoms of a given molecule. If required, an NMR spectrum can then be simulated from the individual shift values by placing Gaussians of a fixed width around each measured chemical shift, and potentially adding noise.

### 3.3. Former Approaches for Chemical Shift Prediction

The field of NMR chemical shift prediction has been addressed by a variety of diverse approaches. Thus, it is difficult to present an exhaustive overview of former approaches. In this section, we present the most important ones instead. For an overview on protein chemical shift prediction, the interested reader is referred to an excellent review by Wishart [Wis11].

As mentioned in Section 3.2.2, NMR spectroscopy is sensitive to  $^1\text{H}$ ,  $^{13}\text{C}$ , and  $^{15}\text{N}$  atoms. However, proteins contain several atoms of these types and in the remainder of this chapter, we use the following nomenclature borrowed from the PDB file format conventions as defined in table B.3.

As customary, we distinguish between the following classes of prediction approaches: Full Quantum Mechanics, Sequence Homology, Semi-Classical Models, Statistical Models, and Mixed or Hybrid Models (see Fig. 3.7). These terms will now be explained in more detail.

**Quantum Mechanics Approaches** If computational complexity would be irrelevant, full quantum mechanics would always be the method of choice. From the wave functions, all chemical shifts can be predicted. However, for proteins in solution, this approach is usually too costly. Still, a number of quantum mechanical approaches targeting proteins has been proposed, each using different approximations (e.g. [HK64, KS65, PB82, HGPF88, OKK04, GYK<sup>+</sup>07, KO07, BKO11, FOME11]). For example, the SHIFTS [XC01] approach from the group of Case predicts  $^{15}\text{N}$ ,  $^{13}\text{C}^\alpha$ ,  $^{13}\text{C}^\beta$ , and  $^{13}\text{C}'$  chemical shifts by using backbone torsional angle patterns to query a database of pre-computed

density functional theory (DFT) calculations. The underlying database was trained on more than 2000 peptides.

**Sequence Homology Models** The other extreme on the computational scale are homology based methods. Here, chemical shifts are inferred from known shifts of homologous proteins. A typical example is the program ShiftY [WWBS97], which collects homologous sequences from the BMRB [UAD<sup>+</sup>08] to derive a set of similar fragments and to deduce chemical shifts for the query protein. Other examples for homology based shift prediction are the approach of Gronwald and coworkers [GBS<sup>+</sup>97], the TALOS program [CDB99], and the SPARTA program [SB07, SB10].

**Statistical Models** Similar in spirit to homology based models, but more general, are the Statistical Models: here, techniques from statistical learning are used to train a predictive model for the chemical shift from a large set of known training data values. Statistical learning is a very wide field with a diverse set of methods, and hence, many different approaches for shift prediction have been attempted. Most notably, Meiler proposed the artificial neural network approach PROSHIFT [Mei03], Arun and Langmead presented a random forest model [AL06], and Atieh and coworkers used a polynomial expansion in their BioShift program [AAFA10].

A great advantage of the PROSHIFT approach is that, in contrast to most prediction models, it does not restrict spatial proximity to sequential vicinity. It is also worth mentioning that PROSHIFT is one of the few publications besides ShiftX [NNZW03] in this field that explicitly described the preparation of its training data set and made it freely available. In contrast to most data sets, the Meiler set also contains protein structures that have been resolved by NMR. In 2003, Meiler had access to 322 BMRB–PDB pairs for training and evaluation. At the time of writing of this thesis (Q1 of 2011), we already had access to 2029 PDB–BMRB pairs with NMR resolved PDB files.

A statistical learning method that tries to remedy the overfitting problem of neural networks is the work of Arun and Langmead. The authors employed random forests, a modern non-linear regression ensemble machine learning method that is provably robust against overfitting (a detailed description can be found in Section 3.5.2).

The BioShift model, on the other hand, follows a more general goal, namely the chemical shift prediction for atoms of any type of biological molecules: proteins, DNA, RNA, polyamines, etc. alike. To provide a simple and computationally fast model, its parameters depend only on Amber [CCB<sup>+</sup>95] atom types, bond lengths, angles, dihedrals, and non-bonded terms using generic polynomial functions. Although the generic functions seem complex at first glance, the number of parameters in practice is small (see table 3.3). Unfortunately, the parameter training was limited to shift information of 5 proteins and 4 polyamines only.

The method, we would like to mention is the Preceding Residue Specific Individual (PRSI) method by Wang et al. [WWC<sup>+</sup>04], which uses a hypersurface similar to the one of the ShiftX approach described in the next paragraph. The PRSI model was trained for backbone nitrogen shift prediction, and only uses features derived from the preceding residue in the sequence.

**Semi-Classical Models** The idea of combining several semi-classical approximations to the full quantum mechanics problem into an additive model for chemical shift prediction leads to the so-called *Semi-Classical Models*. Here, different effects known from full quantum mechanics are approximated using simpler classical models such as ring-current effects or hydrogen bonding. Typical semi-classical terms are discussed in Section 3.2.3.

Important examples of these models are the approaches by Ösabay and Case [ÖC94], Williamson and Asakura [WA93], and Kohlbacher and coworkers [KBM<sup>+</sup>01]. The models differ in the effects taken into account, the exact approximations used, and the parameters used to combine them. For example,

### 3. NMR Shift Prediction

the Kohlbacher approach computes the shift as

$$\delta = \delta_{\text{coil}} + \delta_{\text{A}} + \delta_{\text{JB}} + \delta_{\text{EF}} \quad (3.14)$$

where  $\delta_{\text{coil}}$  denotes the random coil shift,  $\delta_{\text{A}}$  the magnetic anisotropy contribution,  $\delta_{\text{JB}}$  gives the ring current effect in Johnson–Bovey approximation, and  $\delta_{\text{EF}}$  is the effect of the electric field (these terms are discussed in Section 3.2.3). Please note that Kohlbacher et al. modelled chemical shifts for hydrogen atoms only.

Unfortunately, semi-classical approximations do not reflect the full nature of chemical shifts, which leads to the idea of hybrid models.

**Mixed- or Hybrid Approaches** Over the years, experience has shown that some of the effects governing chemical shifts can be well-represented by semi-classical terms while others cannot. Hence, good semi-classical models are today often combined – usually in a simple, additive form – with a specially trained statistical model to improve the prediction with only small increase in computational complexity. And indeed, most modern approaches for chemical shift prediction in proteins seem to use such a mixed- or hybrid approach. In this way, those effects that are well-explained by semi-classical approximations will be well captured, while the remaining effects that cannot easily be translated to analytical formulae or predicted via classical means can be estimated by a statistical model.

The most prominent example of such an approach is the ShiftX model by Wishart and collaborators [NNZW03] for predicting  $^1\text{H}$ ,  $^{13}\text{C}$ , and  $^{15}\text{N}$  chemical shifts. ShiftX combines semi-classical equations for ring current, electric field, and hydrogen bonds with empirically derived hypersurfaces. These chemical shift hypersurfaces were pre-calculated and capture dihedral angle, side chain orientation, secondary structure, and nearest neighbor effects. Recently, a novel approach called ShiftX2 [HLGW11] has been presented, a combination of two improved versions of former programs: ShiftY+ and ShiftX+. The improvement of ShiftY+ over the earlier ShiftY was achieved by employing a local alignment method instead of global alignment.

The second component, ShiftX+, differs from the earlier ShiftX model mainly in the choice of the statistical component, which was originally a hypersurface. The new model is still a hybrid model, but not in the classical sense: while semi-classical and statistical prediction are usually just added together to form the final value, ShiftX+ now uses the values of the semi-classical terms as additional features for the statistical model. In total, the approach uses 63 features, trains separate tree-based predictors on these, and uses bagging and boosting methods to combine the individual predictions into the final outcome.

As customary, ShiftX2 trains different models for each atom type it supports, leading to a total of 6 backbone and 34 side chain models.

A model that is very similar to ShiftX2 is the SPARTA method of Shen and Bax [SB07, SB10]. The key idea of the SPARTA method is to use ring current and hydrogen bonding effects to correct the result of a homology search, which employs local sequence and structure similarity of a residue and its sequential nearest neighbors to predict its backbone chemical shifts. A database is searched to find and average over the 20 best matches for a given sequence triple and torsion nine-tuple. Statistical optimization was employed to determine the weighting factors for the torsional angles and for sequence similarity. The backend database contains  $^1\text{H}^N$ ,  $^1\text{H}^\alpha$ ,  $^{13}\text{C}^\alpha$ ,  $^{13}\text{C}^\beta$ ,  $^{13}\text{C}'$ , and  $^{15}\text{N}$  chemical shifts for 200 proteins for which a high resolution X-ray structure ( $< 2.4\text{\AA}$ ) was available at the time of construction. The recent new development SPARTA+ [SB10] additionally uses an artificial neural network.

The CamShift prediction model of Kohlhoff and coworkers [KRC<sup>+</sup>09] strongly focuses on conformational information as well. The approach combines a random coil term with a polynomial expansion in terms of interatomic distances and angles. The parameters of this expansion were trained on a data set extracted from the RefDB [ZNW03] and depend on atom type, residue type, and hybridization state. The resulting function is differentiable and, as the authors state, can hence be used as restraint



in molecular dynamic simulations [KRC<sup>+</sup>09]. The resulting shift prediction can be applied to H<sup>α</sup>, H<sup>N</sup>, C<sup>α</sup>, C<sup>β</sup>, C', and N backbone atoms.

An extended version of CamShift additionally offers distance and angle dependent polynomial terms for hydrogen bonds, ring currents, and disulfide bridges. Note that while the simple version is a nearly linear model, the extended version involves more complex functions, e.g., cosines of the dihedral angles.

**Non-Protein Models** Until now, we have described the state of the art in protein chemical shift prediction. But the importance of NMR extends to other kinds of molecules as well, such as DNA, RNA, and ligands. As we have seen repeatedly in this thesis, the computational approaches that can be applied to the different molecular classes differ most fundamentally between the cases of proteins, DNA, and RNA on the one side and ligands on the other. Again, the typically small size of ligand molecules allows to use computationally expensive approaches from quantum mechanics, while their diverse chemistry prevents the use of most simpler techniques. For proteins, DNA, and RNA, on the other hands, their regular composition and building block nature favors fragment based approaches. For the sake of completeness, we will restrict ourselves to naming the most important approaches for DNA and ligand chemical shift prediction here.

In the field of DNA shift prediction, the web server of Lam [Lam07] and the methods of Wijmenga et al. [WKH97] and Altona and collaborators [AFWH00] are available. The BioShift [AAFA10] model mentioned earlier is also applicable to DNA.

For ligand molecules, a detailed review of prediction methods is given in [SBC<sup>+</sup>08]. Popular examples are NMRPredict [NMR], the ACD/CNMR Predictor suites, the approach by Kuhn and coworkers [KENS08], the databases NMRShiftDB [SKK03, SK04] and SpecInfo [BG91], and the approach by Abraham and coworkers [ABGP06, ABGK05]. An exhaustive analysis of statistical methods for metabolite hydrogen chemical shift prediction was performed in the work of Kuhn and coworkers [KENS08].

### 3.4. Aims of our Work

Our work on NMR chemical shift prediction described in this thesis has two major goals: establishing a fully automated pipeline for training and evaluating new shift prediction models (including the generation of the underlying data sets), and extending protein shift prediction to the case of protein ligand complexes. The reasoning behind our interest in these goals will be described in the following. When we started our work, our initial motivation was to extend the NMR-guided protein-protein docking approach by Kohlbacher et al. [KBM<sup>+</sup>01] to the case of protein-ligand docking. But after evaluating the former approaches described in the last section, and considering our own experiences described in [Deh07], we realized that the current state of protein-only shift prediction does not yet allow this extension. The reason for this shortcoming is not so much the prediction quality per se, but rather problems with consistency, applicability, and extensibility. For example, as discussed later in this thesis, previous prediction approaches concentrate on input that was derived from X-ray crystal structures instead of using NMR structures for the training and test sets. We will later demonstrate that the performance of state-of-the-art methods drops considerably when applied to non-X-ray structures, but in the scenario we have in mind, the characteristics of the input are more consistent with the NMR than with the X-ray case: the results of a docking run usually have imperfectly placed atoms, which is more consistent with a lower-resolution structure than with highly resolved X-Ray. In addition, the docking is simulated in solution rather than in a crystal.

A second problem with former approaches concerns the fact that most of them have been trained on data sets that include protein-ligand complexes, but the ligand has merely been ignored during training. To avoid overfitting, we would thus have to take particular care to exclude any protein-ligand complex that is homologous to one of those contained in the original training set of the protein model

### 3. NMR Shift Prediction

at hand. This, however, would greatly reduce the amount of available training data for protein-ligand shift prediction.

We thus wanted to train a protein shift prediction model on a ligand-free data set of NMR resolved structures in order to base protein chemical shift prediction on NMR structures with a prediction quality that is sufficient to later on detect the sometimes minute influences of ligand atoms on the protein.

In a first step, we evaluated whether one of the established programs could form the basis of our extension. Unfortunately, this turned out to be impossible. Current shift prediction code is not written with extensibility or adaptation in mind, and indeed, re-training the existing models to account for new features – if such can even be implemented into the system – is a cumbersome and time-consuming manual process. Similarly, data set generation is usually performed by hand in order to guarantee optimal input quality of the training and test sets.

As a consequence, previous approaches for NMR protein chemical shift prediction are usually only very rarely improved upon or trained on new data sets. Instead, a more or less complete rewrite is often necessary.

We thus decided to not only create a new prediction method, but to also attack the problem at its roots instead: we want to develop a fully automated pipeline that can generate up-to-date data sets, perform the training of a new hybrid model using a choice of statistical learning techniques, and evaluate the model without the need of manual intervention. The pipeline also encapsulates a large number of necessary pre-processing steps, which will be described in detail in Section 3.6.1. Using this pipeline, we created two different hybrid models, which will be discussed in Section 3.6.4 and Section 3.7

In the process of creating the pipeline, we noticed that many of the solutions for the pure protein NMR chemical shift prediction case can easily be applied to the protein-ligand and protein-DNA case as well. To this end, we designed additional features based on our work in atom- and bond-typing (see Chapter 2), and thus extended the pipeline to the protein-ligand case. Finally, we performed a proof-of-principle study that demonstrates its potential.

## 3.5. Materials and Methods

In this chapter, we present the materials and methods for the development of a pipeline for automatically generating and evaluating data sets and prediction models for new hybrid protein and protein-ligand chemical shift prediction. The latter, NMR chemical shift prediction for protein-ligand complexes, has, to the best of our knowledge, not been investigated so far.

Typical hybrid models combine semi-classical terms with a statistical model. We, however, use the semi-classical terms as features in addition to molecular features in a fully statistical model. Another innovation of the presented work is the size and breadth of features (semi-classical, sequential, structural, force field based, and experimental) provided in the pipeline by implementing known features and introducing new ones. Independently from our approach, the recently published ShiftX2 [HLGW11] program extended its set of features considerably as well. However, we invented and implemented many additional features that are not covered by the ShiftX2 model. These include, e.g., rotamers, GAFF atom types, density and packing related properties.

Finally, the pipeline allows to analyse different statistical models based on an automatically generated data set. In contrast to alternative prediction methods, our automatically generated data set considers NMR resolved PDB structures as opposed to the X-ray resolved structures. As we will repeatedly see in this chapter, this choice guarantees consistency between NMR shift data and three dimensional conformations: roughly speaking, the structures have been resolved by the same experiment that was used to generate the NMR data, and thus guarantee the same physico-chemical conditions and complete sequence identity between PDB and BMRB files.

In the remainder of this chapter, we present the construction of the data set, our new hybrid model combining the semi-classical terms, molecular data derived features, the statistical models used in our pipeline, and a grammar for the CIF file format used by the BMRB.

### 3.5.1. Data Set Construction

Careful selection and design of a proper training data set are crucial to finally yield a reasonable statistical model, in our case even more so than usually: since we want to extend shift prediction to cases that were previously ignored, we cannot just rely on previous data sets or conventional rules of thumb. In this section, we thus want to discuss the reasoning behind the choices we made in this crucial step.

For training NMR shift prediction methods, we need in essence three different kinds of data as input: (a) recorded chemical shifts and the corresponding physico-chemical conditions of the experiment, (b) three-dimensional protein- and protein-ligand structures for the recorded shifts, and (c) values for the different kinds of features we decide to offer to our statistical models. The term “data set” is often used ambiguously in NMR shift prediction and can, in general, refer to any of the three, or a combination thereof.

Once we have access to the data described in (a) and (b), the third kind, i.e., the features for the training of the predictors, can be computed completely. This will be discussed in detail in Section 3.5.2. But unfortunately, the first two kinds of data are spread over several different databases: the three-dimensional atomic data for our protein- and protein-ligand systems is provided by the Protein Data Bank (PDB) [BWF<sup>+</sup>00, BHN03], where it is stored in the PDB file format. The NMR shift data, and information about the experimental conditions leading to these shifts, is collected in the Biological Magnetic Resonance Bank (BMRB) [UAD<sup>+</sup>08], where it is stored in the NMRStarFile format (often referred to simply as “BMRB files”), a version of the Crystallographic Information File (CIF) format specialized to NMR data. Section 4.3.1 will discuss how we parse these in our implementation. Neither of these two data sources typically contains a reference to the other, but the BMRB offers a database of PDB–BMRB pairs where the PDB structure has supposedly<sup>3</sup> been resolved from the experiment described in the BMRB file. This data set contains several homologous proteins, which might be an issue for training statistical or hybrid models based on it and hence, the mapping is often pruned. Our own choices for the mapping between PDB and BMRB will be discussed in detail later in this section. Unfortunately, both data sources are known to sometimes contain serious errors and inconsistencies. This is particularly problematic for the BMRB data, which not only often contains clear syntactic errors, but also much less obvious logical ones. Probably the most prominent example is the so called “referencing problem”, which will be discussed in more detail later in this section. In brief, the chemical shift is measured relative to a certain reference value but this choice is not always consistent: unfortunately, and somewhat unexpectedly, this important step is often incorrectly reported in the BMRB files, leading to shift values that are constantly biased away from the true values. While referencing errors might in principle be detected and corrected for by trying to estimate this offset, another common kind of error is harder to address: many files contain mis-assigned atoms, leading to errors that are even harder to track. A recent study indicates that as much as 20% of the data files in the BMRB are mis-referenced, and up to 40% contain assignment errors. Consequently, a number of publications [ZNW03, GGCH07, RV10, Wan10, HLGW11, Wis11] hint at the necessity of checking and correcting the given data, BMRB NMR data and PDB coordinates alike.

From what has been said above, we see that the creation of an NMR chemical shift prediction data set is not a trivial straight-forward enterprise but rather has to address several critical issues, such as the completeness and quality of the PDB files, the chemical re-referencing problem for NMR data, and

<sup>3</sup> A certain amount of pairs in this official mapping contain obviously inconsistent information (e.g., different amino acid sequences), which introduces problems that will be discussed later.

### 3. NMR Shift Prediction

the treatment of homologs within the data set. Most former approaches thus rely on manual, hand-curated data sets created by the application of non-standardized sequences of restriction and correction methods. But this manually driven approach leads to a number of problems, most importantly, a potential bias and the enormous amount of effort involved. We thus decided that in our approach, we want to automatically prepare such data sets instead. To this end, we need to answer several open questions arising from the above discussion:

- How can we automatically create valid mappings between PDB and BMRB entities?
- Should we use NMR or X-ray resolved PDB files and how do we obtain them automatically?
- Should we perform correction of BMRB and PDB files and if so, how?
- Should we exclude homologous proteins and if so, to what extent and how?
- Which features should we consider for training the statistical model?

In the following, we will address each question in detail.

**PDB-BMRB Mapping** A PDB to BMRB mapping solves the problem of assigning shifts that were recorded in an NMRStarFile and deposited to the BMRB to the three-dimensional representation of atoms stored in a PDB file. The problem is typically decomposed into two sub-problems: the mapping of PDB entries to BMRB entries, i.e., which creates a relationship between whole molecules, and the mapping of PDB atoms to BMRB shifts, which works on the level of individual atoms. In the following, we will refer to the first sub-problem as the “PDB-BMRB mapping problem”, to the second as the “PDB-atom-to-BMRB-atom mapping problem”.

A number of PDB to BMRB mappings have been previously published. The most important of these are the very recent ShiftX2 [HLGW11] training and test set, the RefDB [ZNW03], the PROSHIFT set [Mei03], the TALOS+ set [SDCB09], or the general PDB to BMRB mapping of the BMRB [UAD<sup>+</sup>08] itself. All but the RefDB and the BMRB lack regular updates. To the best of our knowledge, all alternative prediction approaches first created a data set by hand, applying many manually chosen restriction criteria. This, however, has three major disadvantages: first, it is a very cumbersome and time-consuming process that is usually not repeated when new experimental information becomes available. Second, the manual curation of the data set imposes a certain bias into the final models. And third, cutting away much of the input space might remove valuable information. The first problem could be circumvented by using one of the available data sets (some information about these data sets can be found in table 3.5).

However, all of these data sets suffer from the other two mentioned problems. Thus, we decided to approach the problem differently: we want to make use of all available data with the ability to easily retrain the model. To obtain the largest possible data set currently available, we make use of the official mapping between PDB and BMRB entries provided by the BMRB. Simple alignment with, e.g., ClustalW [THG94, CSK<sup>+</sup>03] provides BMRB-atom to PDB-atom mapping information. The corresponding atoms can then be identified by an atom naming converter linking BMRB to PDB atom naming conventions.

**NMR versus X-ray Resolved Structures** A number of approaches like ShiftX [NNZW03], Sparta [SB07, SB10], and CamShift [KRC<sup>+</sup>09] focus on X-ray resolved PDB structures, arguing that X-ray experiments provide higher coordinate accuracy than NMR resolved structures.

We, however, decided to base our data set on the official BMRB to PDB mapping mostly because of its consistency: the mapping usually connects chemical shift data with the structures that have been resolved based on the very same NMR experiment. Thus, physical conditions, protein modifications,

and similar variables will in most cases be identical between BMRB and PDB entry. As a further advantage, the most recent version of the mapping can be automatically obtained from the BMRB web site.

In addition, taking NMR resolved structures into account relieves us from deducing missing hydrogen positions by other software. Hydrogen placement not only affects and biases the hydrogen shift prediction, but also the predictions for other atom types since hydrogen atoms are present everywhere and form hydrogen bonds, an important feature in most prediction approaches. Instead of deducing this information from other algorithms, NMR resolved structures directly contain this information.

On the other hand, X-ray resolved structures are assumed to have in general higher resolution than their NMR resolved counterparts and consequently, most shift prediction methods rely on them as input to the training. In our opinion, though, the advantages of using NMR resolved data as input, most importantly the greater consistency, outweigh their disadvantages (resolution and typical structure size).

Furthermore, the application we have in mind is a docking scoring scenario where we want to compare our prediction explicitly against an NMR resolved structure. Thus, a dedicated NMR based prediction method is needed.

To the best of our knowledge, only one prediction model was trained on a mixture of NMR and X-ray resolved structure, the PROSHIFT [Mei03] method, but none was trained on NMR structures only. Our approach fills this gap.

**Correction of BMRB and PDB files** An alternative source for the shift data with similar advantages as the BMRB would be the RefDB, which is essentially a re-referenced version of the BMRB. As discussed previously, there is sufficient reason to believe that a non-negligible percentage of chemical shifts in the BMRB have been misreferenced, which can obviously lead to spoiled shift prediction. For our study, however, we decided to base our predictors on the official data instead of the re-referenced ones. The most obvious reason for this choice is the fact that the RefDB has been re-referenced using ShiftX as a predictor, which might bias the process, a bias that we wanted to exclude for our study. As a second reason, we wanted to assess how good automated predictor generation from non-referenced data can perform as a lower bound for the achievable accuracy.

Unfortunately, due to the complex nature of the experimental procedure, chemical shift information is prone to numerous kinds of errors, where BMRB as well as PDB information is affected.

We will first consider errors in the BMRB files, and then discuss errors in PDB files. Typical sources of errors within the BMRB are mis-assignments, typographical errors, and chemical referencing errors. Clearly, typographical errors are the simplest case and can be taken care of by, e.g., BALL's `normalizeNames()` procedure. The other two types of errors are more complex and we first need a clear definition. An *assignment error* occurs, if the chemical shift is assigned to the wrong atom. A *referencing error* is defined as a general reference offset between two NMR experiments with the same molecular entity. There are two reasons for this difference [Wis11]: the use of an incorrect reference standard and the use of wrong solvent conditions.

According to Wishart [WBY<sup>+</sup>95], the errors induced by choosing the wrong reference can be as large as 4–40 ppm.

In a number of publications [ZNW03, GGCH07, RV10, Wan10, HLGW11, Wis11], the necessity of checking and correcting the given data, BMRB NMR data and PDB coordinates alike is discussed. For instance, Wang and coworkers [Wan10] state that more than 20% of the proteins in the BMRB are improperly referenced and nearly 40% of the protein entries are hampered by assignment errors [ZNW03]. For BMRB files, a number of approaches (e.g. [ZNW03, GGCH07, RV10, Wan10]) has been developed so far to detect and correct assignment and referencing errors.

In the following, we give short descriptions of each method.

The ShiftCor/RefDB approach [ZNW03] was among the first methods for solving these problems. Here, the chemical shift information is corrected with respect to the solvent (RefDB assumes DSS)

### 3. NMR Shift Prediction

nucleus	solvent compound	ratio
$^1\text{H}$	DSS	1.000 000 000
$^{13}\text{C}$	DSS	0.251 449 530
$^{15}\text{N}$	$\text{NH}_3$	0.101 329 118

Table 3.1.: IUPAC recommended ratios for chemical shift referencing, taken from [Wis11].

method	year	detects mis-assignment	detects referencing	structure-based	basis
RefDB (ShiftCor)	2003	Y	Y	Y	ShiftX prediction
CheckShift	2007	N	Y	N	sequence based secondary structure prediction
Vasco	2010	Y	Y	Y	solvent accessible surface and secondary structure
PANAV	2010	Y	Y	N	residue type and secondary structure classes

Table 3.2.: Summary and comparison of chemical shift validation programs.

based on a protein-wide delta to its ShiftX predictions. RefDB was used for data set generation in many shift prediction approaches, e.g. [NNZW03, AL06, HLGW11, KRC<sup>+</sup>09].

The CheckShift method [GGCH07] performs referencing error correction based on a protein wide distribution comparison between a target distribution (the queried NMR file) and a prepared reference distribution.

The Vasco (“Validation of Archived chemical Shifts through atomic COordinates”) approach [RV10] performs detection and correction based on solvent accessible surface and secondary structure information.

The “Probabilistic Approach for protein NMR Assignment Validation” (PANAV) [Wan10] identifies and corrects assignment and referencing errors based on fragment-wise comparison of residue type and typical secondary structure dependent chemical shift distributions.

A detailed review on NMR referencing techniques is given in [Wis11]. IUPAC recommended ratios for chemical shift referencing correction relative to DSS are shown in table 3.1.

For PDB files originating from X-ray experiments, a number of methods for quality checks is available [WRZ<sup>+</sup>03, WS07, BLZ<sup>+</sup>10].

ShiftX2 applied these approaches extensively for the generation of their data set. We, however, decided to use NMR resolved structures in our approach, thus relieving us from X-ray correction.

**Homologous Proteins** Another important aspect of designing a data set is the problem of homology within the data set.

In protein chemical shift prediction, the use of homologous proteins in the training and test data sets is debated controversially. One body of opinion states that chemical shifts are so strongly structure dependent that using homologous proteins can only be beneficial for prediction accuracy, while the other argues that the presence of a protein in the test set homologous to a protein in the training set will severely skew the statistics and might lead to an underestimation of the real error rates. Since we want to ensure a rigorous evaluation of our model, we trade in the potential benefits of using homologous structures for an increased robustness and hence use a homology filter on the BMRB to PDB mapping.

To this end, we cull the mapping by sequence similarity with cutoff 10% by applying the standalone PISCES package provided by the Dunbrack group [WDJ05].

It is our belief that for NMR shift prediction the homology bias problem does not arise, at least not as strongly as in the other fields, since NMR is extremely sensitive to structural changes. On the contrary, a predictive model will gain additional insight since the small deviations in structure as they can be found within homologs lead to strong changes in the chemical shifts. This information might

be crucial when targeting structure validation and docking scenarios alike.

However, excluding homologs from the training and test data set allows evaluation as it has been performed in the field before and thus comparison to other methods. An interesting question for future research is if and how the exclusion of homologs influences the weighting of the features in the statistical model and impacts the test errors.

Data set generation is part of our automated pipeline, and the techniques used for its construction as well as the resulting data set will be described in Section 3.6.2. Having thus addressed the problem of the data set, we can now discuss the new hybrid model.

### 3.5.2. A new Hybrid Model

As discussed previously, we decided for a hybrid approach, combining semi-classical terms with a statistical model for NMR chemical shift prediction. In the following, we first describe the semi-classical terms we used for our implementation, we then describe the other features reflecting the structural environment, and then focus on the requirements for a suitable statistical model.

#### Semi-Classical Terms

For our work, we decided to employ models for the contributions discussed in Section 3.2.3. We only excluded the magnetic anisotropy, since predictions for this term are typically insufficiently accurate. For the random coil contributions, we used parameters as defined in [WBH<sup>+</sup>95], since these are well-established and current. For the ring current model, we employ the Haigh–Mallion method [HM72, HM79], for the electric field effect the model by Buckingham [Buc60], with more recent parameters taken from the ShiftX implementation [NNZW03]. The hydrogen bond term was also modeled as in ShiftX.

The chosen models all have recently developed parameter sets reflecting the current state of the art, can be evaluated efficiently and numerically accurate, and – most importantly – have been shown to be compatible with each other so that arbitrary combinations can be combined for the full model.

Besides semi-classical terms, our new hybrid model uses a large number of molecular properties and experimentally derived features, which will be discussed in the next section.

#### Features for NMR Chemical Shift Prediction

A comparison of former approaches shows that the features used for the statistical models are very similar. To construct our set of putative features, we developed new ideas and combined them with known features that may have been already considered by different former approaches. Considering the application we have in mind, namely the scoring of docking results in a high-throughput manner, we need to carefully balance potential accuracy and computational efficiency. We decided to address both goals by implementing a large number of features, where each feature can be computed efficiently and accurately as opposed to using fewer computationally expensive ones. The idea behind this broad ansatz is to make use of as many different points of view and to capture as many influences and dependencies as possible to support the prediction algorithm.

The large size of the feature set has another advantage: during the process of training a statistical model, we will implicitly perform a feature selection or importance estimation. Offering many alternative features to the subset selection may turn up important features that might otherwise have been missed, or only approximately taken into account through a combination of other features. Hence, our approach increases our chances for gaining more insight into the driving forces or key players for NMR chemical shifts.

All features have been implemented in our library BALL [HDR<sup>+</sup>10], which combines the implementational efforts of years of coding and community testing to serve as a standard for algorithmic performance, and not to rely on a rag rug of secondary programs. This not only greatly facilitates the

### 3. NMR Shift Prediction

	ShiftX	CamShift	BioShift	ShiftX2	BALL
HN	608	235	15	63	111 (49)
HA	1310	300	20	63	111 (49)

Table 3.3.: Comparison of the number of parameters between the ShiftX [NNZW03], CamShift [KRC<sup>+</sup>09], and BioShift [AAFA10] model as performed in [AAFA10]. BioShift tries to accomplish similar overall prediction accuracy as ShiftX2 despite using only a small number of parameters. For better comparison, we also added the number of features offered by ShiftX2 and our new protein model, for which we show the number of features offered for training and, in parentheses, the final number of features actually used.

implementation but also leads to improved stability and correctness as compared to alternative approaches. The resulting set is to the best of our knowledge unique with respect to its size and diversity. Please note that the large feature size does not increase the risk of overfitting our statistical models significantly, since we have a very data rich situation: even in our approach, the final number of features is still several orders of magnitude smaller than the number of data points (c.f. table 3.9).

The opposite strategy to our broad ansatz was performed during the design of the BioShift model of Atieh and collaborators, who were interested in the smallest number of features necessary to still reach comparable root mean square deviations (rmsds) and conducted research into the number of parameters required (see table 3.3). In their study, Atieh et al. found that for H<sup>N</sup> and H<sup>α</sup> atoms, the BioShift chemical shift prediction achieved similar rmsd values compared to ShiftX and CamShift although it uses significantly fewer parameters. For convenience, we added columns for ShiftX2 and our models, as well.

In addition to the semi-classical contributions presented in Section 3.2.3, our NightShift pipeline offers sequential features, structural features, force field-based features, and experimental features.

Please note that some of these features are not directly used as input to our final statistical models: a small number of them is used as quality filters instead, while some others have been excluded for different reasons (see below). However, to keep the pipeline as general as possible and to improve extensibility, we retain them for later use.

In the following, we will describe our features in more detail.

**Sequential Features** Very common properties for NMR shift prediction are the atomic *element*, the amino acid type *aa*, and the information whether the current residue is a C- or N-*terminal* residue. Traditionally, (compare, e.g., ShiftX or PROSHIFT) the amino acid type information is also evaluated for the sequential vicinity of a residue *i*, i.e. residues *i* + 2, *i* + 1, *i* - 1 and *i* - 2. A very simple sequential feature is the amino acid sequence length (*protein\_size*) of the protein under consideration.

**Structural Features** By far the most features of our putative set are of structural character. Our feature set encompasses secondary structure-, hydrogen bond and disulfid bond-, torsional angle and distance-, rotamer-, surface-, as well as density related features.

The influence of backbone torsional angles upon chemical shifts has been reported since the early days of NMR shift prediction research and intensively investigated since then by, e.g., [AST<sup>+</sup>84, ÖC94, ZALL10]. Important applications of this work include NMR resonance assignment (e.g. [GSBW00]), the determination of backbone torsional angles (e.g. [CDB99, SDCB09]), or secondary structure assignment (e.g. [WSR92, MK09, ZALL10]). For more details, the interested reader is referred to the work of Cornilescu and coworkers [CDB99].



Thus, secondary structure information and backbone torsional angles belong to the core features of chemical shift prediction, e.g., in the ShiftX model, the PROSHIFT method, and the homology based SPARTA approach.

We used the general *secondary structure* classification types helix, sheet, and loop and the explicit torsional angles  $\psi$ ,  $\phi$ ,  $\chi_1$  and  $\chi_2$  as features. For secondary structure assignment, we use the BALL implementation of the DSSP algorithm [KS83]. Similar to the amino acid type, the torsional angles of the neighbouring residues of a residue  $i$ , i.e. residue  $i + 1$  and  $i - 1$ , are added to our feature set as well.

In addition to the torsional angles, we add a number of distances to our set, namely the distance of an atom to the closest (not necessary within the atom's residue)  $C^\alpha$  ( $dist\_CA$ ),  $C^\beta$  ( $dist\_CB$ ), backbone nitrogen ( $dist\_N$ ), and to the backbone oxygen ( $dist\_O$ ). These or similar features are also employed in, e.g., ShiftX or the BioShift model.

As explained in Section 3.2.3, *hydrogen bonds* are sensitive to hydrogen (foremost  $H^\alpha$ ,  $H^N$ ) chemical shift prediction and are thus covered by semi-classical terms in the ShiftX and the CamShift model.

In addition to the semi-classical contribution accounting for hydrogen bonds, we followed the ShiftX approach in adding a number of features describing a hydrogen bond in more detail. For alpha and amide hydrogen atoms ( $H^\alpha$  and  $H^N$ ), and hydrogen bond acceptors, i.e. backbone carbonyl oxygens (O) and side chain oxygens ( $O_\delta^*$ ,  $O_\epsilon^*$ ,  $O_\gamma^*$ ,  $O_\eta^*$ ), we compute the features  $hbond$ ,  $hbond\_ang$ ,  $hbond\_len$ . These features are indicator, angle, and length properties for the involvement of the current atom's residue in a hydrogen bond.

Potential hydrogen bond donors are carbon, oxygen, and nitrogen atoms. For these, we first compute the indicator feature  $hbond\_donor$ . If the atom takes part in a hydrogen bond, we also compute its length  $hbond\_donor\_len$ , and the angle  $hbond\_donor\_ang$ . Otherwise, the last two are set to a special value, indicating that the feature is not present.

Splitting hydrogen bond features in this manner allows us to distinguish between hydrogen bond donors and acceptors, which in our opinion differ significantly in their influence on the chemical shifts.

From the ShiftX model, we inherited features describing each type of possible hydrogen bond separately for a given residue, e.g. hydrogen bonds via the atoms HA1, HA2, HN, or the backbone oxygen as acceptor. For these hydrogen bond types, we compute an indicator, the length, and the angle yielding the additional features  $hbond\_X$ ,  $hbond\_X\_len$ , and  $hbond\_X\_ang$  with  $X \in \{HA, HA2, HN, OH\}$ . For hydrogen bond detection of both, side chain and backbone atoms, we used the BALL implementation of the ShiftX approach, since the more common DSSP method of Kabsch and Sanders only works for the backbone.

Similar to hydrogen bonds, disulfide bridges stabilize the structure of a protein since they link two thiol groups as found in the amino acid cysteine. We thus added an indicator feature *disulfide*, the disulfide bond's length ( $disulfide\_len$ ) and the dihedral angle  $C^\beta-S^\gamma-S^\gamma-C^\beta$  ( $disulfide\_ang$ ).

To include further knowledge about the protein structure, we do not only store the features "secondary structure elements" and "torsional angles", but also add rotamer information to our feature set. We used the BALL implementation of the backbone dependent rotamer library bbind02.May.lib of Dunbrack [DK93, DK94, DC97] to determine the closest rotamer of a residue *rotamer* as well as the root mean squared deviation (rmsd) from the closest rotamer  $rotamer\_X\_dist$ . In practice, the feature *rotamer* leads to a categorical predictor with 340 levels, much more than can typically be handled by statistical models. Hence, we introduce 340 indicator variables  $rotamer\_X$ , one for each rotamer, instead. These features are then added to all atoms of the corresponding residue.

Atoms on the molecular surface are located in a very different chemical environment from those in the protein's core. In addition to their exposure to the solvent, the shifts of surface atoms are obviously much more sensitive to intermolecular interactions than buried ones. We thus added features to our set that capture the degree of exposure, or the proximity to the surface.

The Solvent Accessible Surface (SAS) area is often calculated in a per-residue fashion [VR09]. However, it has been argued that per-atom SAS values "provide a more meaningful and precise measure

### 3. NMR Shift Prediction

for use in analysis and structure prediction, especially for residues with longer side chains" [SGSA06].

As stated in [VR09], the per-atom solvent accessible surface area correlates with chemical shift information and the ShiftX2 approach takes the solvent accessibility into account as well.

For the computation of the surface area, we used the PARSE radius parameter set [SSH94] with the solvent accessible surface area computation of BALL in a per-atom and per-residue fashion with two different radii for the solvent rolled over the surface:  $r_1 = 1.5\text{\AA}$  and  $r_2 = 0.5\text{\AA}$ . This yields features named *atom\_sas*, *atom\_sas2*, *residue\_sas*, and *residue\_sas2*) that denote whether, and if so, to what extent, an atom and its residue contribute to the protein's surface.

Accounting for the surface area directly leads to the *solvent* as experimental feature. According to [Wis11] "protein chemical shifts have been found to vary considerably depending on the solvent in which they are measured". As typical solvents, Wishart names water, trifluoroethanol, DMSO, and chloroform/methanol, however, all of these but water seem to be very seldom reported (0.4%) [Wis11].

The idea of using the density within the neighborhood of an atom as feature was already introduced by Meiler [Mei03], who used an average reciprocal distance. We further elaborated this idea and created features accounting for the density and packing in spatial vicinity, weighted and simple summation (*atom\_density*, *atom\_w\_density*, *atom\_pack*, *atom\_w\_pack*), and the distance to the protein's center of mass (*dist\_com*).

The feature *atom\_density* simply counts the atoms within a radius of  $5\text{\AA}$ , whereas *atom\_w\_density* weights an atom's contribution with the reciprocal distance to the atom under consideration, also within a  $5\text{\AA}$  radius.

In order to account for the different radii of different atom types we designed 'packing' features. The feature *atom\_pack* adds up the van-der-Waals volume of neighboring atoms within a radius of  $5\text{\AA}$  with a simple spherical capping approximation at the boundary. The feature *atom\_w\_pack* is computed analogously, but additionally weights the volumes with the distance to the atom under consideration. For both features, atoms of the same residue are excluded since the corresponding effect is already encoded in the amino acid and the rotamer features.

The feature *dist\_com*, distance to the center of mass, is considered to account for the buriedness of an atom in combination with the SAS contribution and the size of the protein.

The ShiftX2 and PROSHIFT models also employ hydrophobicity as prediction feature. Several hydrophobicity scales exist, but these usually assign a fixed value per amino acid type. Thus, a hydrophobicity feature would correlate perfectly with the amino acid feature, which should be avoided when training statistical models [HTF09].

**Force Field based Features** Molecular force fields conveniently categorize atoms according to their spatial and topological environment into predefined atom types as part of their typization procedure. We borrow this idea of using an encoded local environment by adding PDB atom types (*atom\_name*) as well as force field energies and energetic contributions to our putative feature set.

For protein atoms, we added the Amber [PC03, CCD+05] force field atom types generated by BALL's `normalizeNames()` procedure to our putative feature set.

The Amber energy itself is computed as

$$\begin{aligned}
 E_{\text{total}} = & \sum_{\text{bonds}} K_r (r - r_{eq})^2 + \sum_{\text{angles}} K_\theta (\theta - \theta_{eq})^2 \\
 & + \sum_{d \in \text{dihedrals}} \frac{V_{n(d)}}{2} [1 + \cos(n(d)\phi - \gamma)] \\
 & + \sum_{i < j} \left[ \frac{A_{ij}}{R_{ij}^{12}} - \frac{B_{ij}}{R_{ij}^6} + \frac{1}{4\pi\epsilon_0\epsilon} \frac{q_i q_j}{R_{ij}} \right] \\
 & + \sum_{\text{H-bonds}} \left[ \frac{C_{ij}}{R_{ij}^{12}} - \frac{D_{ij}}{R_{ij}^{10}} \right] \tag{3.15}
 \end{aligned}$$

and reflects the spatial arrangement of the atoms of a molecule. Here,  $r$  denotes the bond length,  $\theta$  the bond angle,  $n(d)$  gives the number of conformations with minimal energy for a dihedral,  $V_{n(d)}$  denotes the energy barrier height of the current dihedral,  $\phi$  gives the torsion angle,  $\epsilon_0$  the vacuum permittivity,  $\epsilon$  the dielectric constant,  $R_{ij}$  the distance between two atoms  $i$  and  $j$ , and  $K_r$ ,  $r_{eq}$ ,  $K_\theta$ ,  $\theta_{eq}$ ,  $\gamma$ ,  $A_{ij}$ ,  $B_{ij}$ ,  $q_i$ ,  $q_j$ ,  $C_{ij}$ ,  $D_{ij}$  are parameters of the force field.

Since we use the BALL library, we can employ its force field implementation to compute the Amber energies. We use the Amber overall energy  $E_{\text{total}}$  to restrict our data set to valid 3D structures (in our experiments, we found a threshold of  $E_{\text{total}} \leq 1000 \text{kJ/mol}$  to work well).

In addition, we also added the values of the single contributions bend (*AmberBend*), stretch (*AmberStretch*), torsion (*AmberTorsion*), van-der-Waals (*AmberVDW*), and electrostatics (*AmberES*) for each atom. To avoid strong correlation with the rotamer features, we subtract for each atom the energetic contribution due to interactions with its own residue from the overall energy.

In addition to the force field atom types, we also added the atom's *charge* as defined in the PARSE parameter set to our feature set. As pointed out in Section 3.2.3, charges influence the local distribution of bonding electrons and thus change the influence of the external magnetic field during the NMR experiment.

**Experimental Features** NMR experiments vary with respect to a number of experimental conditions. For example, a molecule is solved in a certain *solution* with a certain *pH* value and the experiment is performed under a certain *temperature* and *pressure*.

In some experiments, not only the protein itself but also non-protein ligands are present. This is particularly interesting for our protein-ligand prediction model. We thus added the indicator features *has\_ion*, *has\_ligand*, and *has\_DNA*.

The NMR experiment can be further varied by making use of scalar coupling effects and adding a second (or third) dimension to the experiment. This is achieved by measuring the spin of a second atom type within the same experiment, e.g., COSY (correlation spectroscopy), DOSY (diffusion ordered spectroscopy), or TOCSY (total correlated spectroscopy). We expressed this fact by adding indicator features *has\_H\_shifts*, *has\_C\_shifts*, and *has\_N\_shifts*.

The NMR spectrometer itself can cause experimental differences as well: the *field strength* applied, the type of *spectrometer*, or the *manufacturer*.

In the following, we discuss some experimental features in more detail and argue whether or not these should be included into our putative feature set.

Taking the solvent into account is - to the best of our knowledge - new to the field, but we assume this feature to carry important information. In the work of Abraham and colleagues [ABGP06], for instance, the influence of the solvent solution to the chemical shifts of hydrogens in organic molecules was analysed and an important dependency detected. As the surface of a protein consists to a large degree of hydrogen atoms, the shifts of these surface hydrogens are more sensitive to

### 3. NMR Shift Prediction

intermolecular interactions than that of the buried atoms. Therefore, the solvent effects cannot be neglected [AAFA10] and we add the NMR *solvent* to our list of putative features. In contrast, ShiftX2 used the program SHIFTCOR [ZNW03] to re-reference all experimental observed shifts to DSS (2,2-dimethyl-2-silapentane-5-sulfonic acid) before the actual training. The problem of re-referencing error has been discussed already in Section 3.5.1.

However, most approaches consider solely X-ray crystallography derived PDB files, where the solvent is either not present at all or occurs in a diluted and crystallized state that is very different state from the fluid state in NMR experiments. A number of approaches, like ShiftX or CamShift, take only BMRB–PDB entries of the RefDB [ZNW03] into account, where a ShiftX model prediction based correction for the solvent was already performed.

The *alignment\_score* originates from the mapping between BMRB and PDB files. We employ Clustal W [THG94, CSK<sup>+</sup>03] to determine the mapping between the residues of the PDB file and the BMRB file and use the corresponding score for restricting the data set to BMRB–PDB pairs with 100% sequence identity.

In summary, our set of putative features contains 111 protein features. Some of these, such as the alignment score protein size, pH, pressure, and temperature or field strength are not usually employed as features in the true sense of the meaning, but are rather used as quality filters in different stages of model creation. Many of the features are new and have been first introduced in this work, such as the rotamers, density and packing, distance and contribution to the solvent accessible surface, the total force field energy and selected force field energy contributions, experimental properties like the solvent solution (as opposed to re-referencing), indicators for availability of N, C, and H shifts, and for the protein-ligand case the presence of ion(s), DNA, and ligands in the experimental setup. Table 3.9 gives the number of features computed by the pipeline. Table 3.4 gives a short definition for each feature.

Several of the features presented above are of categorical nature, which renders problems for most statistical models. We postpone the discussion of this aspect until after the introduction of the models used in this work.

#### Issues for the Choice of a Statistical Model

In the previous sections, we have described the different semi-classical components as well as the features that serve as input for our models. Here, we discuss how the two types of information can be combined into a powerful model for NMR chemical shift prediction. Our task now is thus to decide on the statistical model.

In general, our automated pipeline NightShift allows to exhaustively evaluate the space of statistical models. However, not all statistical models fit well to our particular situation, and most models require special treatment of the input. Hence, we decided to limit the model space manually to a few promising models and evaluate the pipeline on those. However, NightShift can be easily extended to handle further models.

The choice of candidate models for this evaluation was guided by a number of questions and issues. For example, the model should be able to handle categorical input like secondary structure and amino acid type, as well as quantitative measurements. In addition, the statistical model should be accurate, robust to overfitting, and efficiently evaluable to be applicable to high-throughput situations.

The set of questions that guided our decisions were as follows:

- How do we combine semi-classical and statistical contributions into a single hybrid model?
- Which statistical models should we consider?
- How should we treat categorical features?
- Do we train a single general model or different models for each common atom type?

feature	definition
atom_name	PDB atom type
element	atomic element
aa	amino acid type
aa_prev	amino acid type of the previous residue (according to PDB indexing)
aa_next	amino acid type of the next residue (according to PDB indexing)
$\phi$	backbone torsional angle involving the backbone atoms C'-N-C $^{\alpha}$ -C'
$\psi$	backbone torsional angle involving the backbone atoms N-C $^{\alpha}$ -C'-N
$\chi$	side chain dihedral angle involving N-C $^{\alpha}$ -C $^{\beta}$ -C $^{\gamma}$
$\chi_2$	side chain dihedral angle involving C $^{\alpha}$ -C $^{\beta}$ -C $^{\gamma}$ -C $^{\delta}$
$\phi_{next}$	$\phi$ angle of the next residue
$\psi_{next}$	$\psi$ angle of the next residue
$\chi_{next}$	$\chi$ angle of the next residue
$\chi_2_{next}$	$\chi_2$ angle of the next residue
$\phi_{prev}$	$\phi$ angle of the previous residue
$\psi_{prev}$	$\psi$ angle of the previous residue
$\chi_{prev}$	$\chi$ angle of the previous residue
$\chi_2_{prev}$	$\chi_2$ angle of the previous residue
secondary_structure	secondary structure of the atom's residue
protein_size	the protein's size in number of amino acids
atom_sas	the atom's contribution to the numerical solvent accessible surface of its molecule ( $r=1.5\text{\AA}$ )
atom_sas2	the atom's contribution to the numerical solvent accessible surface of its molecule ( $r=0.5\text{\AA}$ )
residue_sas	the residue's contribution to the protein's numerical solvent accessible surface ( $r=1.5\text{\AA}$ )
residue_sas2	the residue's contribution to the protein's numerical solvent accessible surface ( $r=0.5\text{\AA}$ )
atom_density	number of atoms within a radius of 5Å
atom_w_density	atom_density weighted with the reciprocal distance to the atoms within a 5Å radius
atom_pack	van-der-Waals volume of atoms within a 5Å radius while excluding atoms of the atom's own residue using a simple spherical capping approximation for atoms crossing the 5Å boundary
atom_w_pack	atom_pack with additional weights for the volumes with the distance to the atom under consideration
dist_com	distance of the atom to the center of mass of its protein
Amber	total Amber energy
AmberVDW	van-der-Waals energy contribution
AmberES	electrostatic energy contribution
AmberTorsion	torsional Amber energy contribution
AmberStretch	Amber bond stretch energy contribution
charge	atom's charge as defined by the PARSE parameter set [SSH94]
random_coil	random coil parameter set as used in [WBH <sup>†</sup> 95]
ring_current	ring current shift contribution as formulated in the Haigh-Mallion model [HM72, HM79]
electric_field	shift contribution due to electric field according to [Buc60] model with parameter set from [NNZW03]
hbond_effect	shift contributions due to hydrogen bonds according to [NNZW03]
dist_O	distance to the next oxygen
dist_N	distance to the next nitrogen
dist_CB	distance to the next C $^{\beta}$
dist_CA	distance to the next C $^{\alpha}$
C-terminal	indicator for the C terminal residue
N-terminal	indicator for the N terminal residue
hbond	indicator for the involvement of the current atom in a hydrogen bond as acceptor or hydrogen
hbond_len	length of the hydrogen bond of the current atom
hbond_ang	angle of the hydrogen bond of the current atom
hbond_donor	indicator for the atom as hydrogen bond donor (carbon, oxygen, and nitrogen)
hbond_donor_len	length of a hydrogen bond where this atom acts as donor
hbond_donor_ang	angle of a hydrogen bond where this atom acts as donor
hbond_X	indicator for the presence of a hydrogen bond of type X (HA, HA2, HN, OH) in the atom's residue
hbond_X_len	length of a hydrogen bond of type X (HA, HA2, HN, OH) in the atom's residue
hbond_X_ang	angle of a hydrogen bond of type X (HA, HA2, HN, OH) in the atom's residue
disulfide	flag indicating the presence of disulfide bridge
disulfide_len	length of the disulfid bridge (distance between involved sulfur atoms )
disulfide_ang	torsional angle of the disulfid bridge (C $^{\beta}$ -S $^{\gamma}$ -S $^{\gamma}$ -C $^{\beta}$ )
rotamer_X	closest rotamer to the atom's residue
rotamer_dist_X	root mean squared distance to the closest rotamer of the atom's residue
pH	pH value as stored in the NMRStar file
temperature	temperature value as stored in the NMRStar file
pressure	pressure value as stored in the NMRStar file
solvent_solution	solvent solution value as stored in the NMRStar file
field_strength	field strength of the NMR spectrometer as stored in the NMRStar file
spectrometer	NMR spectrometer as stored in the NMRStar file
manufacturer	manufacturer of the NMR spectrometer as stored in the NMRStar file
alignment_score	alignment score of PDB amino acid sequence and BMRB residue sequence computed by Clustal W [THG04, CSK <sup>†</sup> 03]
has_H_shifts	indicator for the presence of hydrogen shifts in an NMRStar file
has_C_shifts	indicator for the presence of carbon shifts in an NMRStar file
has_N_shifts	indicator for the presence of nitrogen shifts in an NMRStar file
has_ion	indicator for the presence of an ion in the PDB file
has_ligand	indicator for the presence of a ligand in the PDB file
has_DNA	indicator for the presence of DNA in the PDB file

Table 3.4.: Feature definitions.

### 3. NMR Shift Prediction

- How can we evaluate our model(s)?
- How can we extend the model(s) to protein-ligand complexes?

In principle, an additional question would ask for the technique used to estimate the generalization error of the model. This, however, can be directly answered in our case: our training problem is very unusual in computational biology in the sense that we are given much more training data points than possible features of interest. Indeed, we are in so heavily data-rich situation, that for the evaluation we do not need to perform cross validation and a simple splitting into training and test set suffices.

The use of notation in this field is often ambiguous and imprecise. To avoid confusion, we will thus first introduce a formal nomenclature. Then we will address each of the above questions in more detail.

**Nomenclature for Chemical Shift Prediction** The basis for all of our models are the experimentally measured chemical shifts. To denote these in a more formal setting, we use the following definitions: let  $\mathcal{N} = \{N_1, \dots, N_l\}$  denote the set of nitrogen atoms,  $\mathcal{C} = \{C_1, \dots, C_m\}$  the set of carbon atoms, and  $\mathcal{H} = \{H_1, \dots, H_n\}$  the set of hydrogen atoms in the data set. By  $\delta_{N,i}^{\text{exp}}$  ( $\delta_{C,i}^{\text{exp}}$ ,  $\delta_{H,i}^{\text{exp}}$ ), we denote the experimentally measured shift for the  $i$ -th nitrogen (carbon, hydrogen) atom. Finally, we denote the properties evaluated for the  $i$ -th atom (the features), by  $x_{ij}, j = 1, \dots, p$ .

**A Nonconventional Hybrid Model: the Use of Semi-Classical Terms as Features** Classical hybrid models for NMR chemical shift prediction use a linear combination of semi-classical terms and a statistical component for the residual shift. If we denote by  $\delta_{N,i}^{\text{sc}}$  ( $\delta_{C,i}^{\text{sc}}$ ,  $\delta_{H,i}^{\text{sc}}$ ) the shift predicted for the  $i$ -th nitrogen (carbon, hydrogen) atom by the semi-classical components of our model, the classical hybrid approach models the shift as

$$\delta_{e,i}^{\text{hybrid}} := \delta_{e,i}^{\text{sc}} + \delta_{e,i}^{\text{stat}} \quad (3.16)$$

with  $e \in \{N, C, H\}$ . In this setting, the statistical component  $\delta_{e,i}^{\text{stat}}$  has been trained to predict the *residual shift*  $\delta_{e,i}^{\text{res}}$  as a function of the features  $x_{ij}$  of the  $i$ -th atom (the dependency on  $x_{ij}$  is usually suppressed in the notation):

$$\delta_{e,i}^{\text{res}} = \delta_{e,i}^{\text{exp}} - \delta_{e,i}^{\text{sc}} \quad (3.17)$$

However, in our opinion, a linear combination is too simple a model to account for the complex dependencies between the terms: as explained previously, the semi-classical terms are only approximations with a certain amount of fit parameters. If we combine several of these terms that have been developed independently of each other, these fitted constants may not be optimal. To give an example, the hydrogen bond effect and the electric field effect are clearly not orthogonal and hence should be used with different prefactors when used alone than when used in combination. We thus decided to treat the semi-classical values as features of our statistical model, instead of subtracting them from the experimentally measured shift. The only exception here is the random coil term, which can be directly observed experimentally. Hence, we augment our set of features by the semi-classical terms except the random coil term and try to predict

$$\delta_{e,i}^{\text{res}} = \delta_{e,i}^{\text{exp}} - \delta_{e,i}^{\text{coil}} \quad (3.18)$$

In this way, the statistical model implicitly performs a weighting of the different semi-classical terms. Please note that independently of and simultaneously with our introduction of this unconventional hybrid approach, Wishart and coworkers decided to use the same in their new ShiftX2 model.

**Putative Statistical Models** As discussed in Section 3.3, several statistical models have been previously employed for NMR shift prediction. The approaches proposed in the literature include spline hypersurfaces [NNZW03], ensemble machine learning methods [HLGW11], random forests [AL06], polynomial regression [KRC<sup>+</sup>09, AAFA10], and artificial neural networks [Mei03, SB10].

One of the first statistical models employed were the spline hypersurfaces used by ShiftX [NNZW03]. Here, the input features are used to compute a spline for each feature and a number of spline surfaces for selected pairs of features. However, experience with ShiftX shows that this approach leads to a number of problems. First, the method suffers numerical problems because it is difficult to guarantee continuity at periodic boundaries as, e.g., required by torsional angles. Maybe even more importantly, fitting the splines was performed without controlling the complexity of the surface (the degree of noise or the number of high-frequency components) so that oscillations between training data points lead to a high variance of the result. As such, the method seems prone to overfitting to the training data set.

From a statistical point of view, artificial neural networks are clearly a superior choice to such a noisy spline hypersurface. However, neural networks are becoming increasingly replaced with other modern statistical learning techniques such as support vector machines or random forests since they require a very careful modelling and tuning to achieve good prediction quality without serious overfitting.

General linear or polynomial expansions, on the other hand, as used in the CamShift [KRC<sup>+</sup>09] or the BioShift model [AAFA10] imply a relationship, which might not always be correct or meaningful. In addition, categorical features like amino acid type or secondary structure are difficult to handle in such a model.

Further learning techniques for protein chemical shift prediction based on sequential and structural features have been studied by Wishart and coworkers and Arun and Langmead. Both suggested independently additive bagging learners based on regression trees [AL06, HLGW11].

With this background in mind, we started considering candidates for our new hybrid model. As a first step, we decided to train a simple linear least squares model. The linear model is arguably the most simple statistical regression technique. As suggested by the name, the model uses a linear combination of features to predict the outcome. Training the model requires to estimate the coefficients of this linear function from the data. In general, linear models have small variance, but may lead to a large bias if the real dependency of the outcome on the features is non-linear. While this inflexibility often precludes the use of linear models, their low variance and high stability significantly reduce the risk of overfitting. This makes linear models particularly useful in establishing a trustworthy lower bound of prediction accuracy. To provide a baseline for the quality of automatically derived hybrid models, we thus decided to first train a linear model as a candidate for the statistical component.

As a next candidate, we planned to use support vector regression to yield a method that can adapt better to the data at hand. However, the computational demand for fitting a support vector machine against the large number of training data instances turned out prohibitively large. While we could prune our data set significantly until training becomes possible, we instead decided to use another modern learning technique that is similarly powerful but computationally more efficient, namely a random forest predictor that is similar in spirit to the bagging method employed by the recently published ShiftX+ approach [HLGW11]. Compared to most other regression techniques, random forests have the additional advantage that they can directly include categorical features as well as quantitative ones.

In the following, we describe general features of linear models and random forests, and discuss how we can apply them to the shift prediction problem.

**Linear Model** As explained above, the function we want to predict is the residual shift

$$\delta_{e,i}^{\text{res}} = \delta_{e,i}^{\text{exp}} - \delta_{e,i}^{\text{coil}} \quad (3.19)$$

### 3. NMR Shift Prediction

The linear model assumes that the true functional form of the  $\delta_{e,i}^{\text{res}}$  is given by

$$\delta_{e,i}^{\text{res}} = \beta_0 + \sum_{j=1}^p \beta_j x_{ij} + \varepsilon_i \quad (3.20)$$

where  $\varepsilon_i$  is an error term added for completeness. For NMR prediction, the constant bias term is typically ignored, i.e.,  $\beta_0 = 0$ . One reason for this choice is the fact that NMR shifts are differential quantities where any constant measurement bias will occur in each term of the difference, and hence vanish in total. And even if such a bias would survive building the difference, it would be an individual quantity for one particular spectrometer, and maybe one set of experimental conditions. In our data sets, however, we have a mixture of shifts derived from a diverse set of spectrometers under different experimental conditions. Assuming that a single constant terms works equally well for all of these is highly unrealistic at best. Finally, it is well known that the shifts for a given element type fall into different categories, depending at least on the type of amino acid. Hence, any bias term should be considered not as a constant over the whole data set, but rather be adapted for each of these categories. Since our feature set includes suitable categorical features, the linear model will be free to train an individual bias for each category. Hence, in accordance with typical convention, we also set  $\beta_0 = 0$  in the following.

Collecting all residuals in a vector  $\delta_e^{\text{res}} = (\delta_{e,1}^{\text{res}}, \dots, \delta_{e,n}^{\text{res}})^t$ , all error terms  $\varepsilon_i$  in a vector  $\varepsilon$ , and all features evaluated for all atoms in a matrix  $\underline{\underline{\mathbf{X}}}$ , eq. (3.20) takes the form

$$\delta_e^{\text{res}} = \underline{\underline{\mathbf{X}}}\beta_e + \varepsilon. \quad (3.21)$$

If the error terms have zero mean, are uncorrelated with the features, and have finite variance, a suitable estimate for the  $\beta_i$  values can be found from the ordinary least squares method, which minimizes the sum of the squared deviations of predicted from observed shifts:

$$\hat{\beta}_e := \arg \min_{\beta_e} \|\underline{\underline{\mathbf{X}}}\beta_e - \delta_e^{\text{res}}\|^2 \quad (3.22)$$

with solution [HTF09]

$$\hat{\beta}_e = (\underline{\underline{\mathbf{X}}}^t \underline{\underline{\mathbf{X}}})^{-1} \underline{\underline{\mathbf{X}}}^t \delta_e^{\text{res}}. \quad (3.23)$$

With the so determined  $\hat{\beta}_e$  we can then predict the response of an atom  $a$  with unknown shift and features  $\mathbf{x}_a$  from

$$\hat{\delta}_a = \delta_a^{\text{coil}} + \mathbf{x}_a^t \hat{\beta}_e.$$

**Random Forests** The random forest method was originally developed by Leo Breiman and Adele Cutler [Bre01]. Random Forest have been previously employed in the context of NMR chemical shift prediction by, e.g., Arun and Langmead [AL06], however with a significantly smaller feature and training set based on protein structures solely resolved by X-ray experiments.

Random forests combine many so-called “regression trees” into an ensemble to yield a final output (c.f. Fig. 3.8). Each of these regression trees codifies a number of binary decisions made on the basis of the values of some of the features. The decisions are represented in a tree, where the final result is stored in the leaves. The regression function is then modelled as a piecewise constant function.

While evaluating such a tree is trivial, training a tree requires to find the suitable features on which to perform a split, the values at which a split is performed, the precise sequence of splits on variables (i.e., the topology of the tree), and the constants for each final region. Remember that we want to predict

$$\delta_{e,i}^{\text{res}} = \delta_{e,i}^{\text{exp}} - \delta_{e,i}^{\text{coil}} \quad (3.24)$$



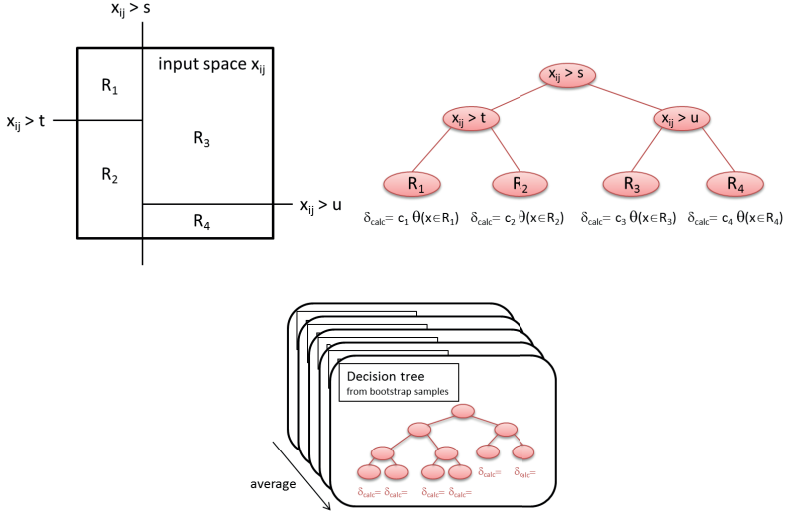


Figure 3.8.: Partition of the input space (top left) implies a regression tree (top right). The combination of regression trees yields a random forest (bottom).

with  $e \in \{N, C, H\}$ , based on the properties  $x_{ij}, j = 1, \dots, p$ , where  $i$  is the index of the training sample and  $j$  the index of the property. These can be combined into a vector of response/feature tuples of the form

$$(\delta_{e,i}^{\text{res}}, \mathbf{x}_i) \quad (3.25)$$

where  $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})$ . The splits performed on each variable partition the input space into a number of distinct regions  $R_1, \dots, R_M$ , so that the final result can be modelled as a piecewise constant function over these:

$$\hat{\delta}_e^{\text{res}}(\mathbf{x}) = \sum_{m=1}^M c_m \theta(\mathbf{x} \in R_m) \quad (3.26)$$

where  $\theta$  is the indicator or Heaviside function, which evaluates to 1 if  $\mathbf{x}$  is contained in region  $R_m$  and to 0 otherwise. If we assume that the regions have already been determined, and if we use an ordinary squared loss function

$$\mathcal{L} := \sum_i \|\delta_{e,i}^{\text{res}} - \hat{\delta}_e^{\text{res}}(\mathbf{x}_i)\|^2 \quad (3.27)$$

the optimal constants  $c_m$  for each region can be easily determined by minimizing the loss:

$$\frac{\partial \mathcal{L}}{\partial c_m} = 2 \left\{ \sum_{i: \mathbf{x}_i \in R_m} (\delta_{e,i}^{\text{res}} - c_m) \right\} \stackrel{!}{=} 0 \quad (3.28)$$

$$\Rightarrow c_m = \frac{1}{n_m} \sum_{i: \mathbf{x}_i \in R_m} \delta_{e,i}^{\text{res}} \quad (3.29)$$

where  $n_m$  is the number of observation points  $\mathbf{x}_i$  falling into region  $m$ . Intuitively speaking, this

### 3. NMR Shift Prediction

result means that the optimal choice of the constant for each region is the average residual shift of all observations falling into this region.

The problem of finding the optimal splits turns out to be more difficult, and is in general not computationally feasible [HTF09]. One option to compute sub-optimal, but nonetheless useful, trees is to use a greedy strategy, where in each round, one variable and one split point is selected such that from all possible choices, the squared loss function decreases the most. In each generated region, the process is iterated until some threshold of the algorithm has been reached to provide the final tree. Several strategies to terminate the splitting process exist, where a larger tree leads to a potentially smaller bias but larger variance, and a smaller tree vice versa. In practice, trees are often first grown relatively large, and then pruned using some node evaluation criteria [HTF09].

For large noisy data sets, as they occur in Bioinformatics, such a single regression tree often yields a relatively weak predictor. In an ensemble method, many individual predictors are thus combined to provide a more accurate outcome. In the random forest method, large numbers of regression trees are built using a random choice of split variables without node pruning instead of a greedy approach. The training subsets for the trees are randomly sampled with replacement from the training data set, a procedure called *bootstrapping*. The final estimation is then computed from the average over all results from all trees in the forest (*aggregating*). Due to its bagging nature (bootstrapping and aggregating), random forests feature variance reduction, and are hence considered very robust with respect to overfitting. In our work, we used the R implementation of random forests by Liaw and Wiener [LW02]. For regression, only two parameters have to be set in this implementation: the number of trees to grow (*ntrees*), and the number of features considered for each tree (*mtry*). It is well-established that both parameters usually only influence the prediction performance mildly, if the number of trees is not set much too small. For most regression applications, it is recommended to set the value of *mtry* to one third of the number of features [LW02].

**The Role of the Element Type** Most former approaches for shift prediction offer a model for each atom type separately (e.g.  $H^\alpha$ ,  $H^N$ ,  $C^\alpha$ ,  $C^\beta$ ,  $C'$ , and N backbone atoms). With the advent of modern statistical models, this specialization is in principle no longer necessary. Ensemble methods like random forests will simply add splits according to atom type, or create different trees for each element type. But for linear models, no such simple loophole exists. Since the typical range of shift values differs by one order of magnitude between hydrogen and nitrogen atoms, combination within one linear model is difficult. For linear models, we thus have to train separate linear models for each atom or element type separately. However, due to the enormous amount of data (see table 3.6) we can split the data into separate training sets for each backbone atom type and even the side chain atom types for both models, linear and random forest alike.

**How to Handle Categorical Features** In regression scenarios such as the presented NMR prediction problems, special care has to be taken when some of the features are not numeric in nature. In our case, this problem occurs, for example, for the amino acid or the secondary structure type of a given residue. To integrate such values into a regression model one might encode these properties with numbers. This, however, leads to the question of how to determine an order within these features.

In general, linear models and random forests in R are able to handle categorical features, but for two of our features – the GAFF atom types and the rotamers – the number of distinct feature values (so-called levels) exceed the maximal number of categories that can be handled by R's random forest implementation. In these cases, the solution is binarization, i.e., the creation of a binary feature for each category, where a value of '1' indicates the presence of the property while '0' denotes its absence. This elegantly relieves us from defining an ordering between the different categorical values. For reasons of efficiency, we only applied binarization to categorical variables that exceeded R's threshold of 32 values.

In general, linear models often suffer from another problem: feature values that are comparably high, e.g., the year when the experiment was performed or the atomic element number. In such cases, one usually has to perform a normalization of the values, since otherwise the weights might suffer from numerical instabilities. In our final models, however, this case does not occur.

**Performance Assessment** If sufficient observations are available, the preferred method of evaluating statistical models consists in assessing the error within the training data set and an independent test data set to measure the generalization error. For NMR shift prediction methods, the training and test errors are typically measured as the root mean squared error (rmse) and by Pearson's Correlation Coefficient (corr).

$$\text{rmse} = \sqrt{\frac{\sum_{i=1}^n (\delta_i^{\text{exp}} - \delta_i^{\text{pred}})^2}{n}} \quad (3.30)$$

$$\text{corr} = \frac{1}{n-1} \sum_{i=1}^n \left( \frac{\delta_i^{\text{exp}} - \hat{\delta}_i^{\text{exp}}}{s_{\delta^{\text{exp}}}} \right) \left( \frac{\delta_i^{\text{pred}} - \hat{\delta}_i^{\text{pred}}}{s_{\delta^{\text{pred}}}} \right) \quad (3.31)$$

with  $\delta^{\text{exp}}$  denoting the experimentally measured chemical shift of an atom,  $\delta^{\text{pred}}$  the predicted chemical shift, and  $n$  the number of predictions made.  $\hat{\delta}^{\text{exp}}$  ( $\hat{\delta}^{\text{pred}}$ ) denotes the mean of the experimentally measured (predicted) shifts and  $s_{\delta^{\text{exp}}}$  ( $s_{\delta^{\text{pred}}}$ ) its standard deviation.

Based on the rmse, the importance of a given feature for a statistical predictor can be estimated by the 'Increase in %rmse' measure, which computes the percentage by which the rmse of a given classifier increases if the values of the feature are randomly perturbed or permuted. If a feature carries significant information about the outcome, exchanging its true values by essentially random ones should hurt prediction performance noticeably and hence, should be reflected in a large 'Increase in %rmse' value.

To provide a meaningful performance estimation, great care has to be taken that model training and assessment are carried out on distinct data sets. Often, this is achieved by estimating the generalization error, which can be approximated using k-fold cross validation or bootstrapping techniques. In k-fold cross validation, the data set is divided into  $k$  subsets of equal size after random permutation. In an iterative fashion, each of the  $k$  subsets is taken aside once while the remaining subsets are used for training a statistical model. The final models are then evaluated on the subsets that were separated in the beginning. The bootstrapping method extends this idea by constructing a large number ( $\ll k$ ) of training sets via random sampling with replacement.

In strongly data rich situations, however, so-called holdout validation is usually preferred over cross validation or bootstrapping [HTF09]. In this technique, the data set is initially split into a training- and test data set. Due to the huge amount of available chemical shifts and the comparatively small number of features (c.f. table 3.9), we thus split the data into such independent training and test sets of user-defined ratio.

In a statistical learning context, the error can be further decomposed into a bias and a variance contribution:  $\text{mse} = \text{bias}^2 + \text{variance}$ , where  $\text{mse} = \text{rmse}^2$ .

Here, the bias measures a general error that is present in all predictions, while the variance denotes the instability or uncertainty of a prediction. Thus, the bias is also known as the systematic, the variance as the random error. High error variance is often a sign of overfitting the statistical model. Such overfitted models will perform poorly for unknown test instances and the generalization error will thus be high. The goal of any statistical model is to reduce both quantities. However, in general a trade-off between both has to be made, e.g., allowing some bias can lead to a variance reduction and vice versa.

### 3. NMR Shift Prediction

**The Functional Form of the Final Models** In an idealized NMR chemical shift spectrum  $\mathcal{S}$ , the peaks are just sharp sticks and the spectrum has the following form

$$\mathcal{S} = \sum_{a \in MS} \delta(a) + \epsilon$$

where  $\epsilon$  denotes the noise,  $\delta(a)$  the chemical shift for atom  $a$ , and  $MS$  the molecular system. If the system under consideration is a protein, the spectrum can be further divided into protein and solvent contributions:

$$\mathcal{S} = \sum_{a \in P} \delta(a) + \sum_{a \in S} \delta(a) + \epsilon$$

where  $P$  denotes the set of all protein atoms and  $S$  the solvent. The chemical shifts within a protein-ligand complex thus are:

$$\mathcal{S} = \sum_{a \in P} \delta(a) + \sum_{a \in L} \delta(a) + \sum_{a \in S} \delta(a) + \epsilon$$

where  $L$  denotes the set of ligand atoms.

If we take the influence of protein, ligand, and solvent onto each other into account as well, we find:

$$\begin{aligned} \mathcal{S} &= \sum_{a \in P} \delta^P(a) + \delta^L(a) + \delta^S(a) \\ &+ \sum_{a \in L} \delta^P(a) + \delta^L(a) + \delta^S(a) \\ &+ \sum_{a \in S} \delta(a) + \epsilon \end{aligned}$$

where  $\delta^P(a)$  ( $\delta^L(a)$ ,  $\delta^S(a)$ ) denotes the chemical shift contribution induced by the protein (ligand, solvent) onto an atom  $a$ .

The chemical shift information stored in a BMRB file is often restricted to protein atoms only. Thus, our model, and the typical content in the BMRB, is limited to

$$\mathcal{S} = \sum_{a \in P} \delta^P(a) + \delta^L(a) + \delta^S(a) + \epsilon$$

Since the solvent is assumed to be always present, and to be distributed evenly around the protein, the solvent influence is usually not modelled explicitly. Instead, it is contracted into  $\delta^P(a)$  and only implicitly represented by, e.g., features like the solvent exposure of an atom.

For a pure protein shift prediction, we thus have to model

$$\mathcal{S} = \sum_{a \in P} \delta^P(a) + \epsilon$$

and for a protein-ligand shift prediction, we have to consider

$$\mathcal{S} = \sum_{a \in P} \delta^P(a) + \delta^L(a) + \epsilon$$

where  $\delta^P$  comprises now the influence of solvent and protein on a protein's chemical shift and  $\delta^L$  the ligands influence.

Given these considerations, we decided for an additive model, given by

$$\mathcal{S} = \sum_{a \in P} \mathcal{M}_{PL}(a) = \sum_{a \in P} \{\mathcal{M}_P(a) + \mathcal{M}_L(a)\}.$$

The pure protein chemical shift prediction model is given by

$$\mathcal{M}_P(a) = \delta_{\text{coil}}(a) + \hat{\delta}^P(a, \delta_{\text{ring}}(a), \delta_{\text{EF}}(a), \delta_{\text{HB}}(a), f_1(a), \dots, f_i(a))$$

where  $\hat{\delta}^P$  is either a linear model or a random forest model,  $\delta_{\text{coil}}(a)$ ,  $\delta_{\text{ring}}(a)$ ,  $\delta_{\text{EF}}(a)$ ,  $\delta_{\text{HB}}(a)$  denote random coil, ring current effect, electric field contribution, and hydrogen bond effect, and  $f_k(a)$  denotes a feature  $k$ . Please note that the random coil contribution is treated separately.

Based on this or any other pure protein model  $\mathcal{M}_P$ , we can then build an additive protein-ligand shift prediction model  $\mathcal{M}_L$ . In pure protein prediction, typical models are based on approximations to semi-classical terms, statistical models, or a combination of both. Since to our knowledge, no reliable approximations of classical terms covering the protein-ligand interface are available, we decided for a purely statistical ligand model

$$\mathcal{M}_L(a) = \hat{\delta}^L(a, \text{lf}_1(a), \dots, \text{lf}_j(a))$$

where  $\hat{\delta}^L$  is a random forest model, and  $\text{lf}_1, \dots, \text{lf}_j$  denote the ligand related features (mainly based on GAFF atom types) as described in Section 3.7.4.

So far, we have fully specified our data set and the statistical models we want to apply for training our hybrid models. The next step consists in parsing the NMR data files provided by the BMRB, which turned out to be a non-trivial challenge. However, to focus the presentation in this chapter on issues directly related to NMR shift prediction, the more technical details of handling these files have been deferred to Section B.2.

## 3.6. Results

In this section, we present the results of our work on protein NMR chemical shift prediction: (a) a fully automated pipeline for the creation of data sets and the training and evaluation of statistical models, which we call *NightShift – NMR shift Inference by General Hybrid model Training*, (b) the actual data set created by the pipeline that uses all current non-homologous entries of the BMRB, (c) our new feature set, and, finally, (d) linear and random forest models trained on these data sets. Results on shift prediction in the presence of ligands will be discussed in Section 3.7.

We will demonstrate that even a simple linear model that has been generated automatically can yield a comparable performance to state-of-the-art programs, while the performance of the more sophisticated random forest-based model, which we call *Spinster*, not only exceeds former approaches for most atom types but also has many crucial advantages to alternative models. The whole training has been performed fully on NMR resolved structures, which is notoriously difficult as described previously.

In addition, we will show that the full automation of the process allows to easily retrain the models as soon as new data is available on the BMRB web site, while alternative techniques will require a cumbersome manual development.

Even though our NightShift pipeline already offers many features, known and novel alike, the set can easily be extended. Similarly, training alternative statistical models requires only small adaptations. These and other possible further extensions will be presented as well.

At this point, we want to stress that we took great care to ensure that none of the differences of our approach to alternative ones will lead to an unfair advantage of our new models when compared to other approaches in the statistical evaluation. For example, the lower resolution of NMR structures compared to high-quality X-ray ones implies that our prediction problem is harder in principle than that of alternative approaches.

On the other hand, we have previously described that the use of NMR structures improves consistency and removes potential bias from the data set. Also, the ability to simply retrain the models and to

### 3. NMR Shift Prediction

use much more data than all alternative approaches counterbalances the resolution problem. As we will later see, the great robustness of the models trained on our data sets supports this argument.

#### 3.6.1. The NightShift Pipeline

In Section 3.5, we described the necessary components to design a new hybrid method. However, if we expect to have to rebuild the model later on, e.g. due to new ideas for features or significant improvements in the number and type of PDB–BMRB pairs, it makes sense to build the components in a modular fashion and to combine them in a ready-to-use pipeline.

Our key goals in developing such a general pipeline are automatizability, flexibility, robustness, and simple extensibility – goals that can easily become contradictory if they are not carefully addressed. By comparing several manual NMR chemical shift prediction approaches, we identified the following steps:

1. creation of an initial mapping between NMR and structural data,
2. filtering and restriction to a high-quality and non-homologous subset,
3. linking the NMR information to individual atoms,
4. computing the proposed features,
5. storing them in a format that can be easily queried, and
6. training and evaluating the proposed statistical models.

A fully automated data set generation and training and evaluation of statistical models poses many challenges. Many of these are of a technical nature, such as the correct parsing of the difficult file formats or the efficient handling of chemical shift data in suitable data structures. While the Biochemical Algorithms Library (BALL) already offered several useful classes for this field, we had to greatly extend its functionality in the course of this thesis. For instance, we added stable and efficient parsers for the CIF- and NMRStar file formats, which were discussed previously in Section 4.3.1. Our extensions to BALL were too numerous to be mentioned here in full detail (they encompassed several thousand lines of code that have all been included into BALL 1.4.1). Instead of discussing each new C++ class or method, such as the semi-classical terms we implemented, we want to focus our attention in this section on the design and implementation of the final pipeline that integrates data set creation and preparation with training and evaluating the models. A schematic diagram of the NightShift pipeline for both, the protein only and the protein–ligand case, can be found in Fig. 3.9. The protein–ligand case will be discussed in more detail in Section 3.7.

NightShift is implemented as a bash script that can run without any manual intervention. In the following, we describe the individual steps of the pipeline. For clarity of presentation, we separate NightShift into logically distinct steps, however, in the actual implementation, some steps, such as creating and filtering the mapping, are performed in the same programs for reasons of computational efficiency.

**Creating an Initial PDB-to-BMRB Mapping** As defined in Section 3.5.1, the problem of mapping experimentally obtained shifts to atoms in a PDB file has two parts: the mapping of PDB to BMRB files, and the mapping of PDB-atoms to BMRB-atoms. There, we have described that we decided to use the official mapping generated by the BMRB for the first sub-problem. Thus, as the very first task in our pipeline, a Python script automatically queries the BMRB for the mapping, downloads it, and parses the results to yield a PDB-ID to BMRB-accession number mapping. Parsing is trivial, since the file merely lists each tuple on a separate line in simple ASCII format. This initial mapping now forms the input for the next step of our pipeline.

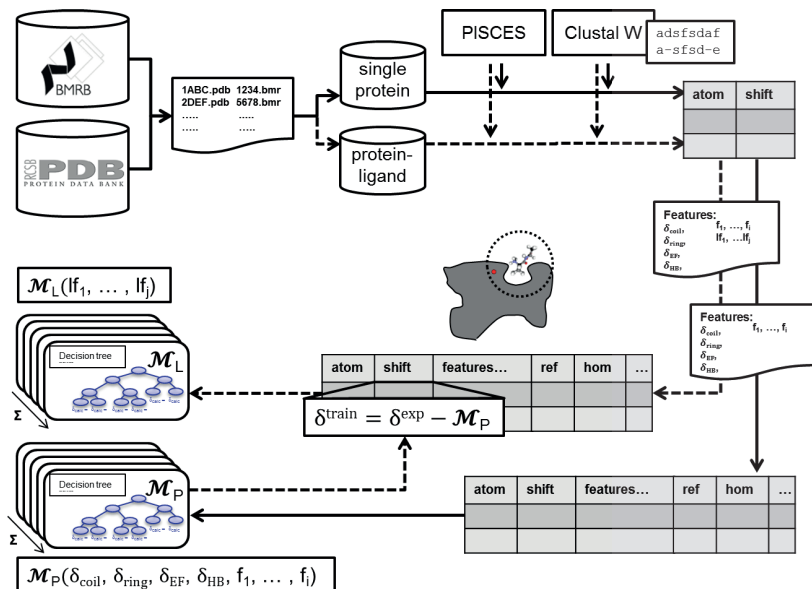


Figure 3.9.: Our NightShift pipeline for data set generation and training of our NMR chemical shift prediction models. The solid lines symbolize the creation of the pure protein model  $\mathcal{M}_P$ , called Spinster. The dotted lines symbolize the pipeline for training the Liops model  $\mathcal{M}_L$  which covers the ligand influence upon the proteins atoms.

### 3. NMR Shift Prediction

**Filtering the PDB-to-BMRB Mapping** The initial PDB-to-BMRB mapping needs to be further restricted with respect to the exclusion of homologs, the application of quality criteria, and the limitation to single protein entities or protein-ligand complexes.

In our pipeline, we use two different filters. The first of these filters considers the types of molecules contained in the system: when creating a protein-only model, we want to skip all PDB entries containing additional ligands or DNA. For the protein-ligand model described later in this thesis, we skip all pure protein PDB entries and entries containing DNA.

To reliably decide whether a PDB entry contains a single protein, a ligand, or DNA before downloading and parsing the file itself, the pipeline includes a Python script to query the PDB RESTful web service (<http://www.rcsb.org/pdb/software/rest.do>) and to parse its results. The RESTful web service uses the principle of REpresentational State Transfer Architecture [Fie00], a simple protocol for communicating with a program over the world wide web. The web service allows to query the PDB for diverse kinds of information, such as the composition of all ligands included in a PDB file. The desired operations and operands can be simply encoded in the HTTP protocol, results are returned as XML. As an example, the following listing shows the result of querying the PDB for the ligand information of PDB-Id 4HHB:

```
<?xml version='1.0' standalone='no' ?>
<structureId id="4HHB">
  <ligandInfo>
    <ligand structureId="4HHB" chemicalID="HEM" type="NON-POLYMER">
      <chemicalName>PROTOPORPHYRIN IX CONTAINING FE</chemicalName>
      <formula>C34 H32 FE N4 O4</formula>
      <smiles>
        Cc1c2/cc/3\nc(/cc\4/c(c(/c(/[nH]4)c/c5n/c(c\c(c1CCC(=O)O)
        [nH]2)/C(=C5)CCC(=O)O)C=C)C)C(=C3C)C=C
      </smiles>
    </ligand>
    <ligand structureId="4HHB" chemicalID="PO4" type="NON-POLYMER">
      <chemicalName>PHOSPHATE ION</chemicalName>
      <formula>O4 P -3</formula>
      <smiles>[O-]P(=O) ([O-]) [O-]</smiles>
    </ligand>
  </ligandInfo>
</structureId>
```

The second filter we decided to employ is a homology criterion as discussed in Section 3.5.1. We thus added a script to our NightShift pipeline that automatically culls the mapping by sequence similarity with a user defined similarity cutoff (in our experiments, we used 10%). To this end, we employed the standalone PISCES package provided by the Dunbrack group [WDJ05]. PISCES, which combines PSI-BLAST and a structural alignment score for homology detection, has a number of advantages for our use-case: first, it restricts a given list of PDB entries to a subset with low paired sequence similarity while preferring structures with higher quality. Thus, information about structural quality is used to decide which entries are removed. Also, it is freely available as a standalone version that can be readily integrated into our pipeline. If a homology filter is not desired by the user, the step can simply be skipped by setting a flag in our pipeline script.

Similarly, it is easily possible to extend the filtering procedure by user generated scripts. These can either directly prune the data set, or add special quality filter columns to the database which can be employed by the statistical models. In later stages, additional filters can be employed at the atomic level, in order to exclude unreasonable shifts or atoms with unrealistic positions.

The input for the next step is now the resulting filtered mapping file between PDB and BMRB entries.

**Download of PDB and BMRB Entries and Creation of PDB-to-BMRB Entry PDB-to-BMRB Atom Mapping Tables** The goals of this step are as follows: (a) to automatically download all PDB and BMRB files specified in the filtered mapping created by the previous step, (b) to compute an assignment between PDB atoms and BMRB atoms, (c) to perform referencing error correction, if



so desired, and, (d) to store both mappings (PDB-Id to BMRB-accession number and PDB-atom to BMRB-atom) in separate tables. This functionality is covered in our NightShift pipeline by another Python script, which starts by automatically downloading all files (PDB and BMRB) listed in the filtered entry-level mapping from the respective web interfaces.

The next step is then to compute the assignment from PDB to BMRB atoms. Our general approach was discussed in Section 3.5.1. In practice, several technical problems complicate matters further: both databases use different chain identifiers, different residue indices, and different atom naming conventions. Here, we discuss how our implementation circumvents these difficulties.

To unify the chain identifiers between both files, we compute a pairwise alignment of each chain of the PDB with each chain of the BMRB file by calling ClustalW and re-index the chains so that the alignment scores of chains with equal index are maximized. If multiple chains can be matched with equal best alignment score, we simply choose the first match. This situation often occurs for homo-multimeric proteins, where the chemical shifts are only stored once in the BMRB file, but the PDB file contains the full multimer. If we would match the shifts with each of the identical domains in the PDB file, we would introduce highly redundant data with a severe risk of overfitting our models and hence, retaining only one match improves the stability of our method.

The alignments of the mapped chains are then used to re-index the sequences in PDB and BMRB file so that we arrive at a mapping at the residue level.

To resolve the different atom naming conventions between PDB and BMRB, we supplemented BALL with a PDB to NMRStar atom name converter, that stores a bijection between all names in both formats for each amino acid. Iterating over the atoms of each amino acid, and converting its name to NMRStar format yields the desired atom-level assignment.

When reading the BMRB files, we face some technical problems since BMRB files are hard to parse correctly and in some cases, the files contained serious syntax errors or inconsistencies. We thus designed a fault-tolerant NMRStarFile parser which we included into the BALL library, as well as data structures and algorithms for mapping and assignment of NMR chemical shift data.

For convenient handling of the data and simplified exchange between the different components of our pipeline, we store all information accumulated at run time that is needed for the training of our chemical shift prediction model in later stages, namely the experimental shifts of the atoms, the corresponding atomic features, and the filter or quality scores, in a database.

We decided to use SQLite for this step, since it is easily accessible from the main languages used by our pipeline, Python, C++, and R [R D11], on a variety of different operating systems.

At this stage, we store the residue alignment and its alignment score, which can be later used for quality filtering, in a PDB-to-BMRB mapping-related table of the database. In addition, for each PDB-BMRB chain pair, we store general information that is related solely to the PDB or BMRB entity: NMR experimental information, such as availability of  $^1\text{H}$ ,  $^{13}\text{C}$ , or  $^{15}\text{N}$  NMR chemical shifts, the NMR spectrometer used in the experiment, the experimental conditions like solution, pH, temperature, and pressure.

The NMR information is accessed using BALL's Python interface to our NMRStarFile parser, while PDB information can be obtained from the PDB RESTful web service.

Now, finally having access to the NMR file Id, shift re-referencing can be performed, if so desired. As explained previously, in this work we decided to skip this step to remove a potential source of bias. However, it can be easily included in the model by processing the stored BMRB files, or by downloading the data from a corrected source, such as the RefDB, directly.

The input for the next step is a PDB-to-BMRB atom mapping.

**Creation of an Atom-to-Features Table** For each matched atom, the experimental NMR chemical shift has to be linked to the corresponding structural atom data and to its features. To this end, we need to parse the previously downloaded BMRB and PDB files to access the relevant information, to

### 3. NMR Shift Prediction

assign the experimental chemical shifts to PDB atom entities, to compute and store all features and the semi-classical contributions, and to allow fast access to this data. At the same time, we want the approach to be easily extensible if new features are conceived.

Based on the given mapping table, the corresponding PDB and BMRB files (in PDB and NMRStar format, respectively) are read in and chemical shifts are assigned.

At this point, we can finally compute the features described in Section 3.5.2 for each of the atoms in our input. For PDB files that contain several connected components like additional ligands or water we face some technical problems. On the one hand we often face incomplete ligand information while on the other hand the ligands often result in a failure of the Amber force field computations, since Amber is only parametrized for amino acids, DNA, and RNA. We thus split a given molecular system into several molecules for which Amber force field calculations can be performed separately without loss of information.

All features have been implemented in BALL, which greatly simplified matters by offering data structures and pre-implemented functionality. To achieve simple extensibility of our approach, each feature is implemented as a separate function that is identified by a hard-coded string. A C++ class then takes a list of such feature names and calls the corresponding functions, which have been stored in a `StringHashMap`. This mechanism allows on the one hand to individualize the collection of features that should be computed at run time by modifying the list of enabled feature names. On the other, the names of the features are used to store the result in a corresponding column in the results table. To add a new feature, a user only has to add a function for its computation, described by a unique string, and insert it into the hash map. If the feature's name is included in the list of enabled features, it will be computed for each atom and stored in the results table, where it will be available to the statistical models.

Since this step can be very time-consuming, we developed a parallelized bash script that creates the Atom-to-Features table, spawns a number of independent processes to compute the desired features, and fills the result table with the experimental shift values and the corresponding feature values. After this step, the database has been finalized and will not be changed further. Thus, suitable indices can now be created for each table without adversely affecting running times.

The output of this step is a database with two tables, one for the PDB chain to BMRB mapping related information, the other for the atoms and their shifts and properties.

**Post-Processing and Curation of the Database** For training the prediction models, we decided to use the de-facto standard for open source statistical computing, R [R D11]. But before we can continue with actually training the models, our experiments showed that we need to pre-process and curate the data: the quality and correctness of many input files – the BMRB files in particular – is dubious at best, and syntactic or logical errors sometimes are carried through the different stages into the databases. The goal of this stage is thus to (a) determine problematic entries, (b) try to repair the information if possible and (c) prune the data point from the set if repairing fails.

To this end, we supplemented our pipeline with a collection of R methods that are applied before the procedures for training and evaluating the statistical models are called. Support for reading the tables into R is provided by the RSQLite package.

The most persistent problem in practice arose due to R's dynamic type system: variable types are typically inferred at run time from the values assigned to each variable. Incorrect feature values in the BMRB file often lead to a mis-interpretation of the feature type. For instance, BMRB files sometimes contained syntactically incorrect floating point expressions. A single of these then leads to a wrongly inferred variable type, and the whole feature becomes unusable for the statistical learning procedures. Determining the correct types automatically in an error-tolerant fashion is a very difficult problem. In our case, we finally decided to enforce a hard-coded type for those features for which typization errors occurred.

For some of the categorical features, the situation is even worse: in some cases, the categories (called 'levels' in R) are not well-defined. In these cases, we have to exclude the whole feature. The NMR spectrometer name is a typical example of this effect. The levels Unity+, Unity-plus, UnityPlus, Unity-Plus, UNITYplus, and UNITY-plus probably all refer to the same NMR spectrometer type, but no automated procedure can decide this reliably. To automatize the exclusion of columns during the training, we implemented a filtering method that eliminates user specified features.

Since categorical features can be problematic for statistical regression techniques, our pipeline offers a binarization function that automatically creates and computes decision variables for each of the feature's allowed values<sup>4</sup>.

A further source of complications is incomplete data, where a feature has not or can not be assigned a value for some atoms of one element type. Since we are in a data rich situation, we decided not to impute NA-values from the remaining data but rather use the following simple strategy: if a feature column contains too many NA-values – more than 1% in our current implementation – we omit this feature from training, while otherwise, the feature is retained and only the data points that contain an NA are removed.

Finally, features aiming solely at quality checks are used to restrict the data set to, e.g., sensible Amber energies or high alignment scores, and then pruned from the feature list. This can be achieved in R directly during the reading of the table by performing a simple SQL select statement that can be adapted easily.

To allow for comparison with other chemical shift prediction methods, the pipeline offers a Python script that calls an alternative prediction program, stores its results, opens the database, parses the results, and adds them as a new column into the database. We did this exemplarily for ShiftX and ShiftX2. To prohibit that these columns enter the model training, we adopt the convention to use a prefix of *CMP\_* for the names of features that should not enter the training and filter these out automatically.

The input for the next step is now a well-prepared and curated R-dataframe.

**Training a Pure Protein Model** Our pipeline offers a framework to train any possible statistical model supported by R. However, as discussed in Section 3.5.2, we decided to exemplarily train two statistical models, a linear model and a random forest model. Especially the linear model aims at determining the lower bounds of accuracy, robustness, and performance that could be reached – without sophisticated manual intervention – by automatically training models to an automatically generated data set made up of NMR resolved structures for each atom class.

To comfortably switch between statistical models while guaranteeing the same preparation of the data and selection of columns, we integrated the presented data preparation functionality into a collection of R methods that can be called by the main R script, and implemented for each statistical model a separate R function for training. Switching between the different statistical models can thus be controlled easily via command line arguments to the main R script. The main R script randomly splits the provided shift data set into a training and test set according to a user defined ratio<sup>5</sup> and calls the specific training method of the chosen statistical model.

A training method is given the training data subset, i.e., the remaining feature columns and the column to train against and returns a vector of models, one for each atom super class. For integrating a new statistical model, a method with the same interface has to be implemented.

For training the linear model we use the MASS package [VR02], while for training the random forest models we use the R package randomForest [LW02] with 500 trees to grow (parameter *ntrees*) and a third of available variables sampled as candidate at each split (parameter *mtry*).

<sup>4</sup> This, e.g., helps to overcome a restriction in R's random forest implementation, which is limited to categorical features with 32 levels.

<sup>5</sup> For the pure protein model, we use a ratio of 60:40

### 3. NMR Shift Prediction

Finally, our script automatically stores the created models and convenience information like column variable types and the choice of training and test set in "R.data" format.

**Evaluation** For the evaluation of the resulting prediction model, NightShift automatically applies the model to the test set and computes, for each atom super class, the root mean squared error (rmse) and Pearson's Correlation Coefficient (corr). For random forest models, we also return the 'Increase in %rmse' measure of feature significance<sup>6</sup>. Based on the *CMP\_* prefix, the script computes rmse and corr of other methods on the test set for each atom super class as well.

In the next section, we present the data sets created by our NightShift pipeline.

#### 3.6.2. The Data Set

As described in Section 3.5.1, all previously proposed approaches to NMR chemical shift prediction rely on laborious manual data set preparation due to the various obstacles that render this process highly challenging. In the previous Section 3.6.1, we have explained how our new pipeline can automatically generate training and test data sets to circumvent this time-consuming chore. In this section, we will now discuss the data sets that resulted from running the pipeline at the time of writing this thesis (March 2011). An overview and comparison to established data sets can be found in table 3.5.

approach	year	size of training set files (shifts)	size of test set files (shifts)	X-ray / NMR resolved	homolog exclusion	re-referencing
Meiler [Mei03]	2003	292 (n.a.)	30 (n.a.)	both	N	N
Sparta [SB07]	2007	200 (n.a.)	25 (n.a.)	X-ray	N	Y
Sparta+ [SB10]	2010	387 (n.a.)	193 (n.a.)	X-ray	Y	Y
ShiftX [NNZW03]	2003	37 (n.a.)	31 (n.a.)	X-ray	N	Y
ShiftX2 [HLGW11]	2010	197 (206,903)	61 (n.a.)	X-ray	Y	Y
CamShift [KRC <sup>+</sup> 09]	2009	n.a. (224,036)	35 (n.a.)	X-ray	N	Y
BALL (pure protein)	2011	515 (326,652)	344 (217,768)	NMR	Y	N
BALL (protein-ligand)	2011	106 (27,460)	45 (11,770)	NMR	Y	N
BMRB (incl. DNA)	regularly updated		2977	NMR	N	N
RefDB (incl. DNA)	regularly updated		2426	both	N	Y

Table 3.5.: Summary and comparison of data sets used by different hybrid shift prediction approaches.

The third and fourth column show the size of the data set in the number of proteins. The CamShift publication [KRC<sup>+</sup>09] reported only the overall sum of shifts which we denote in parentheses if numbers are available. Note that for the BALL sets, we already excluded homologous proteins restricted to pairs of identical sequence.

#### Pure Protein Data Set

The original BMRB-to-PDB mapping downloaded from the BMRB at the time of writing contained 2029 NMR resolved PDB to BMRB pairs describing single proteins. The discrepancy to table 3.5 is due to the small number of X-ray resolved structures in the BMRB and due to cases where multiple proteins, DNA, or ligands were contained. Within this set, some PDB files contained multiple chains, in total 236, yielding 2265 PDB chain-BMRB pairs. After performing a 10% homology restriction via PISCES [WDJ05], we arrived at 890 different PDB-to-BMRB mappings accounting for 898 PDB chain-BMRB pairs in our database. Within the homology restricted set, 8 PDBs contained multiple chains.

Our goal in building our data set from NMR- instead of X-ray resolved structures was to improve internal consistency. This goal was clearly achieved: for X-ray resolved PDB files, the corresponding sequence in the BMRB file often differs considerably from the one in the PDB file, while in our NMR data set, this was only rarely the case. In fact, after pruning the non-identical pairs, the data set still contained 859 PDB chain-BMRB pairs, with a minimal protein size of 40 residues, a maximum of

<sup>6</sup> For the definition of these performance measures, c.f. Section 3.5.2

atom super class	members	number of shifts in data set
N	backbone nitrogen	65,246
CA	alpha backbone carbon ( $C^{\alpha}$ )	66,579
CB	beta backbone carbon ( $C^{\beta}$ )	60,353
C	backbone carboxy carbon ( $C^{\gamma}$ )	48,442
H	backbone hydrogens attached to the backbone nitrogen ( $H^N$ )	68,461
HA	side chain hydrogens on alpha positions (1HA, 1HA, 2HA) ( $H^{\alpha}$ )	71,066
HB	hydrogens on beta positions (1HB, 2HB)	62,106
HD	hydrogens on delta positions (2HD, HD1, HD2, 1HD1, 1HD2, and 2HD2)	37,514
HG	hydrogens on gamma positions (HG, 1HG1, 1HG2, 2HG, 2HG1, HG1)	43,221
HEHZ	remaining hydrogens (HE, HE1, HE2, HE3, 2HE, 1HE, 1HE2, 2HE2, HH2, HZ, 1HZ, HZ2, HZ3)	21,532

Table 3.6.: Definition of our atom super classes using notations borrowed from the Amber force field.

370 residues, and a total of 544,520 shifts. The distribution of these shifts over the different atom classes can be found in table 3.6. This also shows that the assumption of a data rich scenario was correct: the final number of shifts is several orders of magnitudes larger than the number of features (c.f. table 3.9).

Comparing our new, automatically generated, data set to the most recent alternative one, the data set of ShiftX2, we can see that we did not only increase the consistency by using NMR instead of X-ray, but also by restricting the set to pure protein instances. Interestingly, the ShiftX2 test set contains only 16 single protein instances, and 45 protein-ligand cases according to information obtained from the PDB RESTful server information. Despite this restriction, our pure protein data set is still significantly larger than that of ShiftX2, which contains  $197 + 61 = 258$  PDB files with 206,903 shifts in the training set. This can be seen as a great advantage of the automated approach taken in this work: building the data set by hand is so cumbersome a task that typically, only a small fraction of the possible input cases are taken into account, while our own approach can just as easily make use of the whole data set as of a sub set.

### Separation of Models by Atom Type

A crucial decision before training the models concerns the handling of different atom types. As extreme cases, all atom types could be collected into one large model that contains the type as one additional feature, or different models could be trained for each atom type for which shifts are experimentally available.

The first option has, to the best of our knowledge, never been used in chemical shift prediction: the physico-chemical processes that govern the reaction of an atom's shift to features such as torsion angles or sequence neighbourhood differ too widely at least for different chemical elements to combine them into a single formula. The other extreme, on the other hand, might lead to the risk of overfitting the models, in particular for the hydrogens which can be classified into many different atom types. Using Amber atom types, for instance, we would end up with 32 individual models for all protein H, N, and C atoms. For comparison, the ShiftX2 approach employs 6 backbone and 34 side chain models. In our study, we decided for an intermediate approach by classifying similar Amber types into 10 atom super classes. The classes were chosen such that the number of experimental shifts available for the training of each did not vary too strongly. For each of these 10 types, we then trained and evaluated a separate model. The final atom super classes are defined in table 3.6, which also shows the available shifts per atom super class.

### Protein–Ligand Data Set

In addition to the pure protein set, NightShift directly produces a second data set in which proteins occur together with ligands, both small-molecular ligands and ions. Please note that in the following, the term "ligand" will refer to both. This protein-ligand data set will later be used for a proof-of-concept study in Section 3.7. To the best of our knowledge, no such protein-ligand NMR chemical

### 3. NMR Shift Prediction

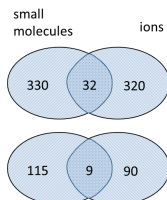


Figure 3.10.: Type and number of ligands in our data set before (top) and after (bottom) applying our filtering criteria.

shift data set or prediction model has been published previously.

To create the data set, the initial PDB to BMRB mapping downloaded from the BMRB was filtered to restrict to those instances that contain a ligand. The presence of a ligand was again determined by querying the PDB RESTful web service. The resulting mapping of protein-ligand complexes consisted of 618 PDB–BMRB pairs with ions or small molecules present. Of this set, 121 PDB files included multiple chains, 320 contained ions, and 330 small molecule ligands. Of these, 32 contained both types of ligands, ions as well as small molecules. Unfortunately, 37 BMRB files in this set contained grave syntax errors that could not be uniquely resolved, and hence, we excluded them from further consideration. This results in a number of 581 PDB–BMRB mappings for our protein-ligand database.

num small ligands	0	1	2	3	$\geq 4$
num PDB entries	288	236	62	19	13

Table 3.7.: Distribution of the number of small molecules per PDB entry.

num ions	0	1	$\geq 2$
num PDB entries	298	305	15

Table 3.8.: Distribution of the number of ions per PDB entry.

Applying a 10% homology restriction yields 196 PDB chain to BMRB mappings (115 ions, 90 small molecules, 9 both) as shown in Fig. 3.10. Further demanding identical sequences in PDB and BMRB and removing covalently bound ligands results in 151 mappings, our final data set. In total, our protein-ligand set contains 102,427 chemical shifts. Retaining only those protein atoms within a 10Å sphere around any ligand atom leaves us with a final set of 39,230 shifts for training and evaluating our models.

#### 3.6.3. The Features

We supplemented our NightShift pipeline with a number of already known features, improved some of the existing ones, and added new ones as well. A detailed description of each individual feature is given in Section 3.5.2 and all definitions are summarized in table 3.4.

In the course of developing NightShift and training the models, not all of the provided features turned out to be meaningful. For instance, the features *field\_strength*, *spectrometer*, *manufacturer* are available in the BMRB files, however, for the spectrometer and the manufacturer further analysis yields that they are often not properly set. In addition, the spectrometer type may strongly correlate with the lab in general and thus with the proteins the lab performs research on. Of these experimental features, we thus finally considered only the field strength. We also briefly considered a number of additional experimental features, but finally decided against these: the values of these features were

	N	CA	CB	C	H	HA	HB	HD	HEHZ	HG
orig num shifts	65,440	66,870	60,761	48,686	68,496	71,243	62,116	37,523	21,548	43,227
orig num features	111	111	111	111	111	111	111	111	111	111
final num shifts train	39,147	39,947	36,211	29,065	41,076	42,639	37,263	22,508	12,919	25,932
final num shifts test	26,099	26,632	24,142	19,377	27,385	28,427	24,843	15,006	8,613	17,289
final num features	44	40	39	44	44	44	38	43	38	38

Table 3.9.: Number of shifts and features of the *pure protein* data set per atom super class before and after pruning missing values. The first two lines show the numbers for the raw data set before applying the training procedure, lines three and four show the number of shifts, and the last line shows the number of features actually used by the models.

method	N correlation (rmse)	CA correlation (rmse)	CB correlation (rmse)	C correlation (rmse)	H correlation (rmse)	HA correlation (rmse)
SHIFTX	0.7073 (0.5190)	0.8820 (2.6593)	0.9758 (1.0746)	0.9957 (1.1733)	0.8384 (1.1724)	0.8875 (0.2533)
SPARTA	0.5960 (0.5845)	0.8985 (2.5141)	0.9814 (0.9418)	0.9968 (1.0107)	0.8763 (1.0222)	0.8012 (0.3336)
SPARTA+	0.5133 (0.6357)	0.8864 (2.7054)	0.9774 (1.0893)	0.9962 (1.0975)	0.8497 (1.1795)	0.8472 (0.3124)
CamShift	0.7143 (0.5060)	0.8636 (2.8236)	0.9744 (1.1035)	0.9959 (1.1442)	0.8632 (1.0697)	0.8926 (0.2474)
SHIFTS	0.5127 (0.6301)	0.7622 (4.4087)	0.9659 (1.2849)	0.9937 (1.4285)	0.6928 (1.7439)	0.8413 (0.2989)
PROSHIFT	0.5742 (0.5928)	0.8273 (3.1527)	0.9368 (2.5713)	0.9900 (2.6842)	0.7941 (2.3260)	0.7847 (0.3439)
SHIFTX2	0.9714 (0.1711)	0.9800 (1.1169)	0.9959 (0.4412)	0.9992 (0.5163)	0.9676 (0.5330)	0.9744 (0.1231)
SHIFTX+	0.9058 (2.3829)	0.9826 (0.9098)	0.9968 (1.0102)	0.8844 (0.9900)	0.7896 (0.4432)	0.9122 (0.2239)

Table 3.10.: Summary of the performance (correlation coefficients and rmse) for predicted *backbone* shifts for seven recent chemical shift predictors using the ShiftX2 test set of 61 proteins. These values were taken from the ShiftX2 publication [HLGW11].

surprisingly often incorrectly set, e.g., negative pressures or a temperature of 0K. Automated or even manual correction of these features seems impossible and we can only hope that future curation efforts at the BMRB will resolve this problem. Hence, we decided to exclude these features from our current models, but keep them in the pipeline so that they can be easily included in future work, as soon as curated data sets become available.

Relying on the BALL library allows improved computation of some already known features. In the original ShiftX2 implementation, e.g., the secondary structure information is assumed to be given. This has two major drawbacks: first, if no secondary structure information is given, the contribution due to secondary structure in the prediction model cannot be evaluated and second, if the algorithm used for assigning secondary structures differs from the algorithm that was used during training, this information might be inconsistent. BALL already contains a variety of routines for problems such as these, e.g., secondary structure assignment, hydrogen bond detection, Amber energies, and many others.

The pure protein data set contained - after filtering - up to 452 features, including 342 rotamer levels (required because of R's random forest 32 level constraint), yielding a total of 111 features.

### 3.6.4. The Pure Protein Models

The final statistical models are evaluated on the randomly chosen test set created by our pipeline. Comparison to state-of-the-art techniques was performed by applying the stand-alone versions of ShiftX and ShiftX2 (Version Q1 2011) to our test data sets as well. The results are shown in table 3.11.

In the work of Wishart et al. [HLGW11], alternative NMR chemical shift prediction methods have been evaluated on the ShiftX2 test data set and the published results are summarized in table 3.10.

Please note that on our test set, ShiftX2 performs significantly worse than on its own validation set, and ShiftX outperforms ShiftX2 for all atom super classes. While part of this effect may be due to mis-referencing, we suspect that the main reason is the lower resolution of the (NMR) structures in our set as compared to the (X-ray) ones in the ShiftX2 training and test sets.

### 3. NMR Shift Prediction

method	N correlation (rmse)	CA correlation (rmse)	CB correlation (rmse)	C correlation (rmse)	H correlation (rmse)
BALL (linear model)	0.756 (3.385)	0.946 (1.582)	0.991 (1.698)	0.710 (1.589)	0.517 (0.537)
BALL (random forest)	0.817 (2.977)	0.956 (1.425)	0.992 (1.582)	0.731 (1.524)	0.593 (0.505)
ShiftX	0.556 (5.603)	0.959 (1.382)	0.989 (1.880)	0.765 (1.452)	0.596 (0.532)
ShiftX2	0.554 (5.606)	0.953 (1.475)	0.984 (2.238)	0.711 (1.650)	0.534 (0.583)
training / test size	39,147 / 26,099	39,947 / 26,632	36,211 / 24,142	29,065 / 19,377	41,076 / 27,365

method	HA correlation (rmse)	HB correlation (rmse)	HD correlation (rmse)	HEHZ correlation (rmse)	HG correlation (rmse)
BALL (linear model)	0.996 (3.381)	0.732 (0.707)	0.987 (0.462)	0.980 (0.385)	0.880 (0.336)
BALL (random forest)	0.997 (2.889)	0.820 (0.559)	0.994 (0.324)	0.981 (0.375)	0.860 (0.365)
ShiftX	0.524 (5.901)	0.725 (1.026)	0.305 (0.342)	0.894 (0.383)	0.375 (0.319)
ShiftX2	0.517 (5.998)	0.721 (1.033)	0.012 (0.706)	0.816 (0.505)	-0.147 (0.967)
size of training / test set	42,639 / 28,427	37,263 / 24,843	22,508 / 15,006	12,919 / 8,613	25,932 / 17,289

Table 3.11.: Summary of the performance (correlation coefficients and rmse) of our models in comparison to ShiftX and ShiftX2 on our test set. The sizes are measured in the number of available atomic shifts.

#### Linear Model

A linear model is arguably one of the most simple regression technique available for the task of chemical shift prediction. In our case, where we have significantly less features than training examples (depending on the atom type, by a factor of 500 – 1000), it is highly unlikely that a linear model would lead to overfitting. Hence, the goal of our linear model is to determine the lower performance bounds of simple statistical models that are trained automatically on an automatically created test sets. Considering the enormous complexity of the task of NMR chemical shift prediction, the automatically generated linear models work surprisingly well on our test set. Except CA, C, and H which perform slightly poorer than ShiftX2, with a correlation (rmse) of 0.946 (1.582) compared to 0.953 (1.475) of ShiftX2 for CA, 0.71 (1.589) compared to 0.711 (1.65) of ShiftX2 for C, and 0.517 (0.537) compared to 0.534 (0.583) of ShiftX2 for H, our models perform better. Table 3.11 shows the results on our test set.

#### Random Forest Model - Spinster

The second model created by NightShift is the random forest model, which we call *Spinster – Single Proteln NMR Shift deTERmination*. In our first experiments, we used a tuning procedure for the parameter *mtry* of the random forest, which denotes how many features are considered for each tree. In general, the random forest method is supposed to be very stable with respect to variation in this parameter, if sufficiently many trees are grown. This was completely consistent with our experience - in all cases, we saw a variation of not more than 5% in rmse when varying *mtry* between the two extremes of 1 and the total number of features. Hence, we decided not to tune *mtry* in the following. Since varying *mtry* does not significantly influence the prediction performance, we instead used the recommended technique [LW02] of setting *mtry* to a third of the number of features. Since we have 111 features in our model, this resulted in an *mtry* value of 37. Similarly, the number of trees to grow did not have a strong influence on the outcome, unless the value was set significantly too small. On the other hand, this value does have a strong influence on running time and memory requirement of the training procedure, we kept the recommended default of 500 [LW02].

With this choice of parameters, we trained Spinster, our random forest models using our NightShift pipeline. The performance of these automatically generated models on the test sets is consistently better than that of the recently published ShiftX2 model, but of similar order. Table 3.11 shows the results on our test set.

Comparing the random forest model Spinster to the linear model, we see an improvement for all atom super classes. Interestingly, the model also performs better than ShiftX2 in all cases. In addition, it is better than ShiftX1 but for the atom class CA, where we achieve a correlation (rmse) of 0.956 (1.425)



and ShiftX1 resulted in 0.959 (1.382). During its training, the random forest implicitly performs a feature selection. We now focus on the feature performance in the random forest model.

While building a random forest, meaningful features for the splitting of the bootstrap sample are determined. To this end, a random forest evaluates the individual features with respect to the “permutation accuracy importance” measure. This value records by how many percent the root mean squared error (rmse) increases if the values of this feature are randomly permuted. The importance measure is given in a ‘% increase in MSE’ scale.

For the pure protein model Spinster, up to 44 features have been used by the random forest. Detailed information on the number of features provided and actually chosen by the random forest models for each atom super class is given in table 3.9. Table 3.12 shows the importance values for each feature employed in our random forest models. Traditional features, such as amino acid type and torsional angles, are within the top ten scored features as expected. However, our newly proposed features seem to be important as well: first, it is striking that for each atom type, at least one of our new features belonged to the top ten most significant ones. In the case of HEHZ, the number even went as high as six. Also, some features seem to be consistently important for all - or at least, many - atom types. The related features *dist\_com*, *residue\_sas*, *residue\_sas2*, *atom\_sas*, and *atom\_sas2* occur at least four times in the top ten, the *sas* even features at least six times. Another feature that is loosely connected with those is the Amber Van-der-Waals - term (*AmberVDW*), which of course also distinguishes between atoms on the surface and in the center. Hence, the particular aspect of ‘buriedness’ seems to play an important role, consistent with our expectation. Another feature that seems to be valuable for prediction is *AmberTorsion*, which also appears four times in the top ten list. This is consistent with the established connection between chemical shifts and backbone torsional angles, but, since the Amber torsion energies are not restricted to the backbone, it extends this concept to arbitrary dihedrals.

As a final note, we want to mention the feature *charge* for HG, which has a surprisingly large value for this atom type, more than six times larger than the importance of this feature for any other atom type. This may indicate that our semi-classical terms do not treat hydrogen bonds for these kinds of atoms correctly, but might equally well be an artifact. We will further investigate this effect in the future.

## 3.7. Application to Protein-Ligand Complexes: A Proof of Principle Study

In [NNZW03] Wishart and coworkers stated “A more serious limitation of SHIFTX, however, is in fact that the program (...) does not account for the presence of organic ligands (heme rings, aromatic substrates, etc.)”. They further state that “The inclusion of rare or unique organic ligands (i.e., drug leads, specially developed inhibitors, etc) will present some challenges. . .”. In this section, we describe our first steps towards approaching these challenges by finally turning our attention to the influence of ligand atoms on protein chemical shifts.

The basic idea behind our approach – an additive model – was already described in Section 3.5.2. In brief, idea is to first identify those atoms in the protein in close proximity to the ligand, to evaluate the protein model on these atoms, and to try and explain the remaining deviation from the experimentally observed shift. While this method does not allow to compute the shift for the ligand atoms themselves, the ligand structure is indirectly reflected in the response of the protein atoms close to parts of the ligand. This dependency can lead to a first step along the way towards a pipeline for including protein-ligand NMR into the scoring of docking results, similar in spirit to [KBM<sup>+</sup>01, MCS<sup>+</sup>08, CMV11], where the protein-protein case has been considered.

The residual shift value, i.e., the deviation of the predicted protein shift from the experimental one, has two contributions: one from the inaccuracies and plain errors in our protein model, and the second

### 3. NMR Shift Prediction

feature	N	CA	CB	C	H	HA	HB	HD	HEHZ	HG
aa	<b>110.82</b>	<b>74.21</b>	12.15	<b>66.07</b>	<b>53.68</b>	<b>100.68</b>	<b>7.71</b>	<b>35.47</b>	<b>62.12</b>	4.42
aa_next	39.74	<b>53.35</b>	8.73	22.37	37.60	10.56	2.61	3.90	8.23	6.03
aa_prev	<b>150.84</b>	11.08	2.81	15.58	<b>46.38</b>	<b>58.26</b>	2.51	9.62	10.49	0.00
Amber	11.45	15.17	4.89	5.32	13.27	9.40	<b>6.77</b>	<b>15.19</b>	4.02	1.05
AmberES	10.06	10.43	5.72	2.29	12.37	5.33	0.17	4.38	5.79	1.17
AmberStretch	8.19	0.00	0.00	4.72	0.00	28.11	0.00	1.05	0.00	0.00
AmberTorsion	<b>49.31</b>	20.22	<b>18.98</b>	<b>28.19</b>	3.60	<b>48.96</b>	2.60	0.00	0.00	0.00
AmberVDW	33.47	<b>27.14</b>	9.78	<b>25.26</b>	<b>40.22</b>	13.62	1.63	10.76	<b>18.26</b>	8.19
atom_density	40.59	17.98	12.23	21.16	27.85	33.54	1.45	2.14	10.96	8.25
atom_sas	15.25	17.26	2.47	18.80	26.25	15.98	<b>6.95</b>	1.64	7.90	<b>12.93</b>
atom_sas2	36.01	25.14	7.09	<b>24.21</b>	<b>45.52</b>	16.40	1.77	2.33	9.10	<b>11.19</b>
atom_name	0.00	0.00	0.00	0.00	0.00	<b>123.99</b>	<b>5.40</b>	<b>20.52</b>	<b>29.70</b>	2.09
atom_pack	29.84	9.63	13.23	<b>24.12</b>	11.68	6.74	<b>5.72</b>	2.03	14.69	8.18
atom_w_density	26.29	14.79	11.76	14.95	15.62	18.77	1.83	<b>15.43</b>	<b>15.88</b>	<b>18.68</b>
atom_w_pack	22.50	12.58	8.77	23.11	14.31	19.91	2.16	2.21	<b>15.24</b>	7.04
charge	8.67	17.44	3.90	10.88	5.19	11.88	3.31	<b>23.52</b>	<b>25.15</b>	<b>164.33</b>
χ	41.59	<b>27.92</b>	3.25	14.89	18.25	<b>34.13</b>	1.14	8.30	5.29	2.97
χ2	22.89	25.79	10.93	13.82	16.23	13.95	<b>5.18</b>	<b>24.41</b>	10.73	6.02
χ2_next	14.07	13.05	9.16	6.67	10.61	4.71	0.00	0.95	5.90	<b>14.64</b>
χ2_prev	30.72	0.00	0.00	5.52	10.96	14.94	0.00	8.34	0.00	0.00
χ_next	17.07	12.46	6.92	5.95	12.85	11.98	0.97	1.93	4.11	3.40
χ_prev	43.30	0.00	0.00	3.86	22.26	<b>40.81</b>	0.00	<b>18.64</b>	0.00	0.00
dist_com	38.89	<b>27.82</b>	<b>18.03</b>	4.74	<b>53.67</b>	10.20	1.05	8.53	8.73	<b>8.78</b>
dist_CA	9.48	14.27	11.07	7.42	8.06	13.85	<b>5.36</b>	5.67	<b>18.63</b>	<b>19.24</b>
dist_CB	<b>44.40</b>	25.21	<b>16.44</b>	11.97	26.39	<b>37.84</b>	<b>6.45</b>	2.56	15.16	<b>38.95</b>
dist_N	22.13	17.02	<b>13.47</b>	23.39	29.16	21.17	1.58	11.94	<b>18.08</b>	6.47
dist_O	7.26	6.49	<b>23.59</b>	<b>27.36</b>	34.12	22.40	<b>6.62</b>	<b>12.02</b>	5.45	<b>12.67</b>
disulfide	0.00	<b>46.32</b>	<b>154.34</b>	0.00	0.00	0.00	5.05	0.00	0.00	0.00
electric_field	0.00	17.20	0.00	0.00	17.37	0.00	0.00	0.00	0.00	0.00
element	0.00	0.00	0.00	0.00	0.00	8.38	0.00	0.00	0.00	0.00
hbond_HA	4.46	3.17	4.33	3.59	7.37	0.00	2.30	5.31	0.00	1.68
hbond	0.00	0.00	0.00	0.00	14.37	0.00	0.00	0.00	0.00	0.00
hbond_donor	13.66	3.88	0.00	0.00	0.00	16.75	0.00	0.00	0.00	0.00
hbond_effect	0.00	0.00	0.00	0.00	23.91	0.00	0.00	0.00	0.00	0.00
hbond_HN	14.76	22.52	<b>17.88</b>	13.01	14.76	14.35	0.13	1.19	6.54	7.62
hbond_OH_len	0.00	0.00	0.00	14.63	0.00	0.00	0.00	2.10	0.00	0.00
φ	<b>80.23</b>	0.00	0.00	23.14	<b>51.62</b>	30.71	0.00	3.28	0.00	0.00
φ_next	<b>49.49</b>	20.02	8.02	23.45	25.94	15.97	2.51	5.20	6.19	4.82
φ_prev	<b>51.14</b>	0.00	0.00	0.00	37.84	12.50	0.00	0.00	0.00	0.00
protein_size	9.88	16.10	3.70	12.23	9.26	3.61	0.69	1.96	1.92	1.50
ψ	<b>64.53</b>	<b>118.21</b>	<b>30.08</b>	<b>34.65</b>	<b>55.26</b>	<b>38.06</b>	0.00	5.89	4.90	8.75
ψ_next	<b>59.97</b>	<b>56.26</b>	<b>15.77</b>	<b>26.93</b>	39.51	<b>42.16</b>	1.04	0.00	8.46	3.97
ψ_prev	<b>212.21</b>	0.00	0.00	23.15	<b>138.20</b>	<b>151.83</b>	0.00	0.00	0.00	0.00
residue_sas	37.95	14.04	12.67	<b>48.43</b>	39.12	21.92	<b>8.23</b>	7.42	<b>29.60</b>	8.75
residue_sas2	25.30	<b>27.92</b>	8.84	20.47	26.09	32.06	4.89	<b>27.65</b>	<b>18.41</b>	7.46
ring_current	17.70	5.07	10.74	13.75	<b>52.83</b>	14.71	0.00	0.00	0.00	0.00
secondary_structure	33.03	<b>81.27</b>	<b>43.71</b>	<b>24.71</b>	<b>51.75</b>	18.45	2.68	<b>15.05</b>	4.72	<b>8.77</b>

Table 3.12.: Importance of features in the *pure protein* random forest models per atom super class. The values are given in the usual '% increase in MSE' scale. The new features are printed in bold, the top ten scored features are printed in bold as well.

from the actual ligand effect. Obviously, in order to succeed, the protein model must be sufficiently accurate, and the ligand effect sufficiently pronounced so that a statistical effect can be detected. In this section, we perform a proof-of-concept study that attempts to demonstrate that this is indeed the case. We will thus need a number of ligand-related features, for which we will make heavy use of our work on atom- and bond-typization and the GAFF force field. The study presented here will thus also investigate whether features derived from GAFF carry information that correlates with chemical shift deviations.

In the following, we will present the methodology behind our approach, including the novel ligand features, describe our extension of NightShift, the pipeline presented in the last section, and discuss the training and evaluation of a preliminary model.

### 3.7.1. General Methodology

Casting the setup described in the previous section in a more formal notation, we decompose the total shift model  $\mathcal{S}$  into a model for the intra-protein effects ( $\mathcal{M}_P$ ) and a model for the influence of the ligand onto the protein ( $\mathcal{M}_L$ ) that are related by:

$$\mathcal{S} = \sum_{a \in P} \mathcal{M}_{PL}(a) = \sum_{a \in P} \{\mathcal{M}_P(a) + \mathcal{M}_L(a)\}$$

This section will be concerned with the automated generation of the protein-ligand model  $\mathcal{M}_L$  through an extended version of the NightShift pipeline discussed in Section 3.6.1. The resulting model is called *Liops - Ligand Influence On Protein Shifts*.

As described previously, pure protein prediction models are typically based on approximations to semi-classical terms, statistical models, or a combination of both. Since to our knowledge, no reliable approximations of classical terms covering the protein-ligand interface are available, we decided to use a purely statistical ligand model, i.e.,

$$\mathcal{M}_L(a) = \hat{\delta}^L(a, \text{lf}_1(a), \dots, \text{lf}_j(a))$$

where  $\hat{\delta}^L$  is a random forest model, and  $\text{lf}_1, \dots, \text{lf}_j$  denote ligand related features.

### 3.7.2. The Underlying Protein Model

For this study, we decided to employ our automatically generated random forest protein shift prediction model Spinster discussed in Section 3.6.4. In principle, any protein chemical shift prediction method can be used as basic model. However, Spinster has several advantages: it was built exclusively on unreferenced pure protein data, PDB entries with additional ions, ligands, or DNA have been excluded, and it was built using NMR resolved PDB structures for high consistency between the PDB data and the underlying NMR experiment.

### 3.7.3. Towards a Model of the Ligand Influence

From our experience on protein-only models, we have reason to expect that force field related features, such as contributions to interaction energies between ligand and protein atoms, will form valuable features for chemical shift prediction in the protein-ligand case as well. However, in contrast to the protein-only case, where the chemistry is uniform and hence force fields are relatively simple, treating protein-ligand interactions correctly is considerably more difficult. Roughly speaking, we will need a force field that is equally suited to protein and ligand atoms. While there are specialized force fields for the treatment of ligands, such as MMFF94 [Hal96b], these usually have drawbacks in correctly capturing the protein chemistry. Hence, we decided to base our study on the GAFF force field [WWC<sup>+</sup>04], which for protein atoms equals the well-known Amber force field, a fact that ensures

### 3. NMR Shift Prediction

consistency with our protein models. For non-protein (or non-DNA) atoms, GAFF offers a wide range of force field parameters and a simple scheme to extrapolate these to atom types that are not fully covered.

However, fully integrating the GAFF force field into our chemical shift prediction is a non-trivial task. Instead, in this study we first determine whether such an approach is worthwhile at all by addressing two different questions: (a) do chemical shift predictions really fare significantly worse in the presence of ligand atoms and (b) is there a non-trivial correlation between the chemical ligand environment encoded in GAFF and the deviation of the chemical shifts close to a ligand from their predicted values? To answer the first question, we have collected a data set of 151 PDB-BMRB pairs of protein-ligand complexes (c.f. Section 3.6.2) with 39,230 shifts of atoms that were located less than 10Å from a ligand atom (c.f. Section 3.6.2). A detailed description of the set can be found in table 3.15.

To address the second one, we created a set of new features that capture information about the chemistry of the ligand atoms close to the protein. Such information has been intelligently encoded into the GAFF atom types, which are based on the local molecular topology of the ligand. In contrast to other atom typing techniques, GAFF not only relies on pure connectivity information, but also uses the orders of the bonds connecting the atoms to deduce the final atom type. For this to work, however, correct bond order information for the ligand is required.

Unfortunately, the ligand information provided by the Protein Data Bank (PDB) [BWF<sup>+</sup>00, BHN03] is often incomplete, hydrogen information is missing and/or bond order information is not provided. In cases where only one of these is missing, the other can be deduced by either filling up free valences with hydrogens or by distributing free valences over bonds. For the latter case, we use BALL's bond order assigner BOA Constructor [DRB<sup>+</sup>11], which we discussed in Chapter 2.

In principle, we could now try to proceed by computing GAFF interaction energies between all pairs of ligand- and protein atoms. However, this turns out to be a non-trivial problem: first, the input structures need to be carefully prepared: charges have to be assigned, e.g., with the help of a quantum chemical procedure, force field parameters have to be extrapolated, etc. Then, we might need to optimize the structures with respect to the GAFF energy in order to relax potentially non-optimal configurations. The parameters for the minimization will have to be carefully controlled, and quite possibly, structural restraints will have to be employed to keep the structure from deviating from the NMR experiment. Finally, we need to be able to decompose the energies into the intra-protein, intra-ligand and protein-ligand components for each term in the force field's Hamiltonian. Not all of these requirements can be easily satisfied using the commercial implementation of the GAFF force field, but on the other hand, implementing GAFF is a challenging task. Thus, while we would want to use force field energies to encode the interactions between protein atoms and the ligand, we cannot currently do so. Hence, we decided to instead start with a simpler set of features for a proof-of-concept study to determine whether such a force field based approach will be likely to succeed: instead of computing GAFF energies, we determine the local chemical neighbourhood from the composition of non-protein GAFF types we find in the vicinity of each protein atom. As cutoff for the neighborhood of protein and ligand atoms, we use 10Å. Careful analysis of the influence and the predictive power of these features will then serve as an indicator whether implementing GAFF will be worth its while for our purposes.

#### 3.7.4. Features for the Influence of the Ligand on Protein Atoms

As the first and most obvious feature we use the total number of ligand atoms within a 10Å radius from the protein target atom, summed over all atom types (*num\_het\_atoms*). In addition, we use the atomic element (*cl\_het\_element*), the GAFF type (*cl\_gaff\_type*), and the distance of the closest ligand atom (*cl\_het\_dist*), and for each GAFF type *i* separately the number and closest distances (*gaff\_type\_i* and *gaff\_type\_dist\_i*) for all ligand atoms within the 10Å radius as features. As mentioned in Chapter 2, our work on bond- and atom-typing allows to assign these types efficiently and correctly.

The feature list is complemented with the number of ligand atoms (*ligand\_size*) and an indicator variable for the presence of ions (*has\_ion*).

name	definition
<i>gaff_type_X</i>	indicator for the presence of GAFF type X within a 10Å radius
<i>gaff_type_dist_X</i>	distance to the closest atom of GAFF type X within a 10Å radius
<i>num_het_atoms</i>	number of ligand atoms within a 10Å radius
<i>ligand_size</i>	number of ligand atoms
<i>cl_het_element</i>	atomic element of the closest ligand atom
<i>cl_gaff_type</i>	GAFF type of the closest ligand atom
<i>cl_het_dist</i>	distance of the closest ligand atom
<i>has_ion</i>	indicator for the presence of ions in the system

Table 3.14.: Ligand related feature definitions.

### 3.7.5. Performance Evaluations

We evaluated the model on the randomly chosen test sets of protein-ligand complexes created by NightShift. In accordance with the pure protein models, comparison to state-of-the-art techniques was performed by applying the stand-alone versions of ShiftX and ShiftX2 to our test data sets. The performance of our models can be estimated from the root mean squared error (rmse) and Pearson's Correlation Coefficient (*corr*) on the test set as defined in Section 3.5.2.

### 3.7.6. Results

#### Adaptation of the Pipeline for the Protein-Ligand Model

Training protein-ligand models equals to a large extent the steps described in the pure protein case (see NightShift pipeline shown in Fig. 3.9), and thus the protein-ligand data set is prepared and handled by the same R methods.

Based on the information provided by the PDB RESTful web service (<http://www.rcsb.org/pdb/software/rest.do>), PDB entries with ion(s) or small molecule ligand(s) present are selected to build the protein-ligand mapping data set in the pipeline's first step.

The resulting mapping is then restricted by quality checks and homology filtering as known from the pure protein case, but then further restricted to protein atoms close to, and thus probably affected by, a ligand. Here, NightShift applies a 10Å distance cut-off.

For the resulting data set, NightShift then computes the protein and ligand features, splits the data set randomly into a training and test set using a ratio of 70:30, loads and applies the pure protein prediction Spinster, and finally trains Liops, the new ligand random forest model. Unfortunately, the data to feature ratio in the protein-ligand set is less favorable than in the pure protein case. We thus used only 3 atom classes to retain sufficient numbers of observation for a sound statistical analysis. The resulting training and test sizes of each atom super class model are shown in table 3.17.

When predicting the pure protein model's shift contribution, we face some technical problems: to be able to apply the pure protein models to the new ligand data, we have to ensure the same variable types and factor levels for the protein-ligand data set as we used for the pure protein set. Thus, we have to prepare the data in the very same way as the pure protein set, e.g., in the case of binarization, the same levels have to be present. If new levels occur, we cannot apply the prediction model and thus have to omit these atoms.

### 3. NMR Shift Prediction

Given the pure protein shift prediction  $\hat{\delta}^P$  of Spinster, we compute a residual shift  $\delta^{\text{res}}(a)$  for each atom  $a$  by subtracting the random coil and the pure protein prediction from the atom's experimental chemical shift:

$$\begin{aligned}\delta^{\text{res}}(a) &= \delta^{\text{exp}}(a) - \mathcal{M}_P(a) \\ &= \delta^{\text{exp}}(a) - \delta_a^{\text{coil}} - \hat{\delta}^P(a, \delta_a^{\text{ring}}, \delta_a^{\text{EF}}, \delta_a^{\text{HB}}, f_1(a), \dots, f_i(a))\end{aligned}$$

NightShift uses a variable `prediction_column` to comfortably switch between the columns to train against (experimental shift, re-referenced shift or residual) while using the same R methods as for the pure protein model.

	<sup>15</sup> N	<sup>13</sup> C	<sup>1</sup> H
ori num shifts	24,523	60,135	118,481
ori num features	202	202	202
final num shifts train	3,873	9,188	14,399
final num shifts test	1,660	3,938	6,172
num features	49	49	50

Table 3.15.: Number of shifts and features of the *protein-ligand* data set per atom super class. The first two lines show the numbers for the raw data set, the last line shows the number of features actually used by the models.

#### Random Forest Model for the Influence of Ligands - Liops

To answer the first question mentioned above, i.e., to determine whether shift prediction really works significantly worse in the presence of ligand atoms as compared to the bulk, we compared the prediction of an established protein shift prediction package, the recently published ShiftX2 program, on the set of protein atoms in the spatial neighbourhood of a ligand atom. The results can be found in table 3.17. As can be seen, the performance of shift prediction close to ligand atoms indeed breaks down considerably. This is in line with our expectations: the ligand atoms are part of the chemical environment, which is what determines the chemical shift values. Since the ligand atoms are not covered by any of the previous shift prediction models we have found in the literature, the estimate of the chemical environment does not match that realized in nature and hence, the prediction can be expected to strongly deviate from the true values. This motivates that additional steps should be taken to model ligand atoms in shift prediction procedures.

To now address the second question, i.e., to decide whether force field based features contain useful information for shift prediction, we started by plotting each feature against the absolute residual shift value, i.e., against the magnitude of the shift not explained by the protein model (of course, this number includes errors in the protein model as well). Not all features show a significant correlation, but some do: from Fig. 3.11 for the feature *ligand\_size*, e.g., we see that larger ligands tend to have a greater influence on shift prediction than smaller ones. The largest ligands in our data set, however, tended to be modified peptides so that the deviation from a pure protein model is not as pronounced as the ligand size might suggest.

Another example of an interesting dependency between residual shift value magnitude and feature can be found in Fig. 3.12, which shows the distances to the closest h4 and n3 atoms. Here, we see that the residual shift tends to decrease with increasing distance from the next h4 or n3 atom, since the interactions that influence the chemical shift are strongly decaying with distance.

In both figures, the results agree with our expectations. However, plotting the residual itself as opposed to its magnitude as a function of the individual features (c.f. Fig. 3.13), we found a less clear trend.

### 3.7. Application to Protein-Ligand Complexes: A Proof of Principle Study

feature	N	C	H
cl_het_dist	<b>57.21</b>	<b>22.34</b>	<b>14.45</b>
GAFFTYPE_11_Cu	6.99	9.12	0.64
GAFFTYPE_12_DU	2.41	4.95	3.93
GAFFTYPE_13_Fe	14.36	0.00	<b>10.31</b>
GAFFTYPE_14_Ga	4.88	1.90	3.27
GAFFTYPE_19_Mg	2.86	0.00	6.31
GAFFTYPE_30_Si	1.70	6.39	0.00
GAFFTYPE_35_Zn	<b>18.34</b>	<b>19.88</b>	6.34
GAFFTYPE_36_br	7.07	3.52	1.13
GAFFTYPE_37_c	<b>20.24</b>	<b>14.78</b>	2.07
GAFFTYPE_38_c1	0.28	10.83	6.77
GAFFTYPE_39_c2	<b>16.27</b>	<b>17.83</b>	<b>20.17</b>
GAFFTYPE_40_c3	<b>15.81</b>	11.52	<b>12.81</b>
GAFFTYPE_41_ca	12.98	12.35	6.34
GAFFTYPE_42_cc	7.91	1.40	3.17
GAFFTYPE_43_ce	4.98	0.00	0.00
GAFFTYPE_44_cg	3.80	7.38	0.00
GAFFTYPE_45_cl	2.26	4.48	4.71
GAFFTYPE_46_cp	1.90	3.11	3.54
GAFFTYPE_49_cx	3.18	0.00	0.00
GAFFTYPE_52_f	2.75	8.06	0.00
GAFFTYPE_53_h1	14.80	<b>15.60</b>	5.79
GAFFTYPE_54_h2	5.37	4.78	6.25
GAFFTYPE_56_h4	1.31	1.21	0.68
GAFFTYPE_57_h5	1.66	4.35	7.89
GAFFTYPE_58_ha	<b>16.30</b>	13.72	<b>9.86</b>
GAFFTYPE_59_hc	<b>17.05</b>	<b>16.57</b>	<b>8.68</b>
GAFFTYPE_60_hn	9.69	8.64	<b>13.56</b>
GAFFTYPE_61_ho	10.85	<b>14.64</b>	4.77
GAFFTYPE_67_ip	2.57	5.11	1.07
GAFFTYPE_68_n	10.46	9.22	6.40
GAFFTYPE_70_n2	3.80	3.34	0.00
GAFFTYPE_71_n3	1.02	4.83	1.77
GAFFTYPE_72_n4	0.00	0.87	1.05
GAFFTYPE_73_na	7.41	8.08	<b>16.49</b>
GAFFTYPE_74_nb	8.46	5.39	3.86
GAFFTYPE_75_nc	11.93	10.49	7.36
GAFFTYPE_76_ne	0.00	1.15	3.72
GAFFTYPE_77_nh	7.58	7.44	2.80
GAFFTYPE_78_no	1.75	0.78	0.33
GAFFTYPE_79_o	13.72	14.07	5.46
GAFFTYPE_7_Ca	<b>15.35</b>	<b>21.85</b>	2.35
GAFFTYPE_80_oh	11.99	13.39	4.95
GAFFTYPE_81_os	8.63	9.79	5.33
GAFFTYPE_85_p5	7.68	6.96	4.25
GAFFTYPE_96_ss	3.69	2.42	5.06
GAFFTYPE_98_sy	5.02	5.85	1.55
ligand_size	<b>29.38</b>	<b>25.47</b>	<b>24.72</b>
num_het_atoms	<b>20.06</b>	<b>21.85</b>	<b>15.02</b>

Table 3.16.: Importance of features in the *protein-ligand* random forest model. The top ten scored features are printed in bold face.

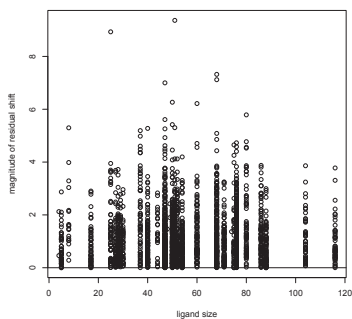


Figure 3.11.: Exemplary plot of the residual shift magnitude as a function of ligand size for carbon atoms.

### 3. NMR Shift Prediction

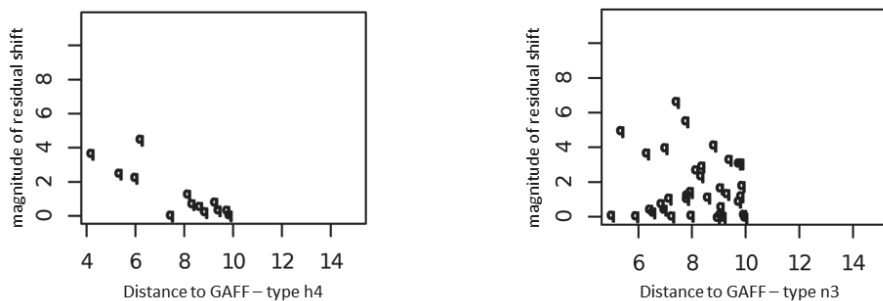


Figure 3.12.: Exemplary plot of the residual shift magnitude as a function of distance to the next atoms of type h4 (left) and n3 (right).

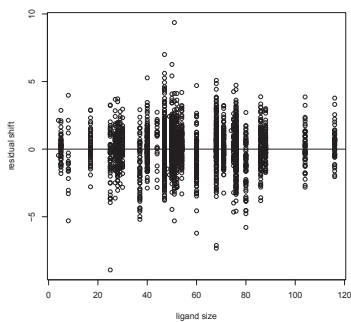


Figure 3.13.: Exemplary plot of the residual shift as a function of ligand size for carbon atoms.



Roughly speaking, the presence of several GAFF atom types in the vicinity of a protein atom seems to induce a strongly deviating shift, but the direction and strength of the deviation can in most cases not simply be read of from the atomic composition of the neighbourhood.

An intuitive reasoning thus seems to suggest that simple atom type features as described above can indeed help in improving prediction performance, but that their influence will be limited, and more sophisticated features such as GAFF energy components will be needed to further improve the quality. And indeed, this is in line with our results when we proceed with our pipeline and train random forest models to the residual shift values. Again, after finding that the model hardly varies with the two parameters, we set the number of trees to the recommended 500, the parameter *mtry* to the recommended number of features divided by three. But since we have significantly fewer data points at our disposal as in pure protein shift prediction, we only train three different models, one for C, N, and H atoms each. The results on the independent, non-homolog test set, which comprises 30% of the initial data set, can be seen in table 3.17. Obviously, shift prediction is assisted by the new ligand model – prediction accuracy increases in all cases – but the increase is moderate in nature.

method	<sup>15</sup> N correlation (rmse)	<sup>13</sup> C correlation (rmse)	<sup>1</sup> H correlation (rmse)
BALL (Spinster - $\mathcal{M}_P$ )	0.798 (2.796)	0.999 (1.643)	0.96 (0.876)
BALL (Liops - $\mathcal{M}_{PL}$ )	0.811 (2.712)	0.999 (1.592)	0.961 (0.867)
ShiftX2	0.72 (3.606)	0.999 (1.804)	0.939 (0.887)
ShiftX1	0.72 (3.606)	0.999 (1.804)	0.943 (0.944)
ShiftX2 (reported)	0.98 (1.117)	0.987 (0.497)	0.973 (0.147)
size (training/test)	3,873 / 1,660	9,188 / 3,938	14,399 / 6,172
num features	49	49	50

Table 3.17.: Performance of ShiftX2 as reported, as measured on our new data set of protein atoms close to a ligand, and of our new ligand model on this set. Values shown are Pearson correlation and rmse (in parentheses), respectively, per atom type. The reported values of ShiftX2 have been averaged over all atom types for each element. The last two lines show the size of our data set in terms of shifts and the number of used features in our random forest model.

### 3.7.7. Discussion

As expected from the feature/residual correlations, the inclusion of GAFF atom type features improves the prediction, but only moderately. The limited improvement is probably due to two effects: for some of the atoms, the errors of the protein model are relatively large and these of course do not correlate with any ligand feature. Secondly and more importantly, the dependency between protein shifts and ligand atoms is complex and cannot be modelled well by just counting atom types. In the terminology of a hybrid shift prediction model, we need semi-classical terms (i.e., models of the underlying physico-chemical processes in contrast to a mere collection of molecular properties) to greatly improve the prediction performance.

Still, considering the complexity of the task – for a similarly complex problem, imagine predicting force field energies using a purely statistical model from the composition of the chemical neighbourhood of a given atom – the slight but consistent increase in model performance strongly indicates that force field based descriptors are a promising ingredient for protein-ligand chemical shift prediction.

Here, we expect to improve matters by using GAFF energies in future work. Even though this might sound like a trivial extension at first glance, it is a highly non-trivial task that involves questions of a

### 3. NMR Shift Prediction

technical (implementation, integration, energy decomposition, ...) as well as of a more fundamental (parametrization, treatment of singularities, treatment of missing atoms or non-optimal input structures) nature. However, from the results of this study we are convinced that the effort will be worth its while.

## 3.8. Summary

In this chapter, we introduced NightShift, an automated pipeline for developing new protein chemical shift prediction methods for pure proteins as well as for proteins under the influence of a bound ligand. NightShift is complemented with a large number of features, established as well as novel ones. Our approach is realized in a modular fashion to offer a ready-to-reuse pipeline in case the available data amount improves in the future, new semi-classical models become available or new ideas for predictive features or statistical models arise. Thus, no time-consuming and error-prone reimplementation is necessary. Instead, only simple adaption of the code is required. Using NightShift, we created a pure protein model, called Spinster, and a protein-ligand model, called Liops.

NightShift can be easily extended by the user in several different directions:

- *Re-training the models:* Given new data in the BMRB, only the PDB to BMRB mapping is needed as input to generate a new linear or random forest model.
- *Using re-referenced data:* Re-referenced shift data can be used by either downloading the NMRStar files from a re-referenced database, such as the RefDB, or by applying a re-referencing tool to the downloaded data as a step in the pipeline.
- *Adding a model:* For testing a new statistical model, the user only has to include the corresponding R package and to wrap the correct prediction method to meet our interface definition of training methods.
- *Adding a feature:* For adding a feature, the user has to choose a feature name string, to add the feature name string to the list of features to compute, and to add a method that computes the feature from the protein structure information. The results will then automatically be included in the corresponding feature column and be made available to the R-based training procedures.
- *Adding a method to compare to:* Adding an approach to compare to is achieved by adding a corresponding *CMP\_* column to the table and registering this column with the validation methods of the NightShift pipeline.

We further showed that our fully automatically generated models not only perform comparably with the state of the art methods in chemical shift prediction but also have many crucial advantages compared to alternative approaches: first, our models are fast to evaluate and robust, making them applicable to high-throughput situations. They are generated automatically and can be easily retrained and adapted to new data. More fundamentally, Liops is the first model to consider the influence of ligand atoms as well.

NightShift has been designed in a way that it can be easily extended by new features and new statistical models, or by adding different homology removal strategies, or shift re-referencing solutions. By default, NightShift uses NMR resolved structures, and is the first approach to do so consistently. However, if X-ray structures are desired, exchanging a single module in the pipeline will allow their use.

The tight integration of NightShift, Spinster, and Liops with the BALL library, allows to easily combine NMR shift prediction with tasks like hydrogen addition, or Docking within the same framework. Creating new computational approaches that use the shifts as part of their scoring function is thus greatly simplified. In fact, in the context of developing the NightShift pipeline we have significantly

extended the functionality of the BALL library, by designing and implementing a toolbox for handling experimental NMR chemical shift data.

In a second study, we showed that extension to protein-ligand NMR shift prediction is indeed possible and presented a protein-ligand NMR prediction model called Liops. We were able to show that features including information about the chemical environment of ligand atoms indeed help in understanding protein chemical shifts and are thus convinced that with further work in this direction, in particular with including GAFF energies instead of atom types only, the goal of using NMR shift prediction in a protein-ligand docking context is indeed a realistic one.

### 3.9. Outlook

With the accomplishments described in this chapter, some of the remaining challenges in NMR chemical shift prediction can now be addressed.

For instance, in NMR resolved PDB structures, more than one model of the structures is usually available. While we currently use only the first model, it would be interesting to compare the prediction of single models and, e.g., to average and weight the input feature values for the training set. However, the detailed decision of averaging and weighting techniques must be carefully made for each feature separately.

One demand on our model is numeric stability for the prediction model. So far, we did not analyse this for our model or the existing approaches, but a scenario would be set up by creating test sets with randomly disturbed atomic coordinates to different extents. The preparation of such test sets, however, requires careful design to prohibit, e.g., steric clashes and access to such experimental data is difficult to obtain. However, this information is important to estimate the significance of observed chemical shift differences, e.g., between different structure models.

Having shown the NightShift pipeline to be valuable as a proof of concept, it can now easily be used to investigate new statistical models and to invent and test new features and semi-classical terms for NMR chemical shift prediction beyond those that were implemented in the scope of this thesis.

Similar to the extension of our pure protein model Spinster to protein-ligand cases, NightShift can also be extended to create hybrid protein-DNA/RNA models. Everything is in place – we only need to apply the pipeline for data set generation, training, and evaluating. However, we currently lack access to sufficient amounts of training and evaluation data for this kind of systems so that we have to postpone this study to future work.

Finally, we want to extend the protein-ligand model Liops in the future by using additional features, such as GAFF energies, that will help in capturing the intricate influence of the ligand atoms upon the protein. Then, the model can be employed as a scoring function for protein ligand docking. A necessary, yet difficult, prerequisite here is the creation of a reasonable data set which contains experimental NMR shift data for different docking poses and exchanging the exact 10Å cut-off for the influence of ligand atoms upon protein chemical shifts by a smooth transition function, e.g.,  $\tanh$ . If the predicted shifts can be shown to separate true positives from false positives, as done for protein-protein docking by Kohlbacher and coworkers, we would want to go one step further and compare unassigned raw spectra to simulated ones.



## 4. The BALL Project: A Framework for Biomolecular Modelling

An important cornerstone of this dissertation were our contributions to the BALL project, i.e., the Biochemical Algorithms Library [KL00, HDR<sup>+</sup>10] and its viewer BALLView [MHLK05, MHLK06]. The BALL project provides an open source C++ framework for structural bioinformatics applications, with a particular focus on molecular modelling and computer aided drug design. It has been developed continuously since its inception as part of the PhD thesis of Oliver Kohlbacher in 1996. In the following, we want to first motivate the importance of the BALL framework before we turn to a brief description of its functionality. This description is based on our argumentation in a recently published application note in the journal *Bioinformatics* [HDR<sup>+</sup>10]. Since this publication perfectly describes several aspects important for this chapter, we re-used parts of our text from that manuscript with kind permission of Oxford University Press (c.f. App. C). In addition, this chapter will introduce some of the components of BALL that were developed by the author of this thesis which were not or only briefly covered by the application note.

The main results of this thesis, as they pertain to BALL, will be discussed in Section 4.2, where we present our implementation of BOA Constructor, and in Section 4.3, where we describe the implementation of our NMR framework. As opposed to the remainder of this work, the presentation in this chapter will focus mostly on the technical rather than the conceptual level and on implementation rather than on algorithmics. For a more thorough description of the respective contexts, the reader is referred to the corresponding sections in the chapters on bond typization and NMR shift prediction. Furthermore, in Section 4.4 and Section 4.5 we summarize our work on integrating real-time ray tracing into the BALL project for visualization as well as for the computation of molecular properties.

### 4.1. Introduction

Developing programs for structural bioinformatics is a difficult and often tedious task. Even if the algorithms have been carefully designed, the programmer has to solve a variety of complex and recurring problems not fundamentally related to the algorithm at hand, but necessary for real-world applications. Not only do more advanced tasks like inferring missing atoms or bonds, energy evaluations, or structural minimization require considerable programming effort that can hardly be repeated for every new project, but also the most basic and mundane steps. For example, many molecular file formats are as hard to parse correctly as they are to write. To avoid costly and error-prone re-inventing of the wheel for any new structural bioinformatics application, two approaches can be imagined: a collection of loosely coupled tools and utilities for recurring sub-tasks, or powerful libraries and frameworks for rapid application development (RAD). Obviously, the second approach encompasses the first, i.e., creating small, specialized tools for a pipeline concept is trivial when relying on such a library. In addition, it allows its users simple access to the molecular data structures and algorithms that form building blocks of many algorithmic approaches and that often require complex implementations.

The Biochemical Algorithms Library (BALL) is a versatile C++ class library for structural bioinformatics that is supplemented with a Python interface for scripting functionality and a number of applications, such as the molecular modeling frontend BALLView. BALL has been used successfully for a large number of projects by the BALL developers (e.g. [DRLH09, DRB<sup>+</sup>11, DLH11, KLHT09, RRK10, CTK11, PGD<sup>+</sup>10, MGD<sup>+</sup>10, ZST<sup>+</sup>11, KMTH11]) as well as by external groups (for a small

#### 4. The BALL Project: A Framework for Biomolecular Modelling

selection of recent publications, see, e.g., [XB06, XJB07, SWBG08, MSP09, SP09, BS10, MTK<sup>+</sup>10]). In recent years, BALL has seen a significant increase in functionality and substantial usability improvements. It has been ported to further operating systems; indeed, it currently supports all major brands. Moreover, BALL has evolved from a commercial product into a free-of-charge, open source software licensed under the Lesser GNU Public License (LGPL).

To the best of our knowledge, BALL offers the widest range of functionality for rapidly and robustly developing applications in structural bioinformatics, it is growing fast and can be easily extended. It addresses users of the implemented techniques as well as designers of completely new approaches. A full description of BALL's functionality would fall well outside of the scope of this thesis; the current version (1.4.1 at the time of writing) contains more than 730 classes and several hundred thousands lines of code. A comprehensive overview can be found in the online documentation at <http://www.ball-project.org>.

BALL has been carefully designed to address programming experts as well as novices. Users can take advantage of BALL's rich functionality and are offered an extensive framework of data structures and algorithms through both, C++ and the python scripting interface. A variety of standard structural bioinformatics algorithms and data structures are offered, including molecular container classes and iterators, support for various file formats, selection of interesting molecular subgraphs by different kinds of chemical expression languages, different force fields for force- and energy estimation, structural minimization and simulation, docking techniques, molecular visualization and editing interfaces, and many more. Additional functionality can be easily added by the user.

The author of this thesis has extended the functionality of BALL and BALLView significantly. An overview of the main contributions to BALL and the people involved in these projects is given in 6.3. In the following, we will focus only on those contributions that were relevant to the remainder of this thesis.

### 4.2. The Implementation of BOA Constructor

BOA Constructor, our novel method for accurate, efficient, and extensible bond order assignment, has been implemented entirely within the BALL library and is included in the official release since version 1.4.1. To this end, we first converted the atom type descriptions as presented in [WWKC06] into SMARTS expressions. Our new penalty table that covers further atom classes (c.f. Section 2.8) uses the same mechanism. Since we wanted to support different optimization strategies, and wanted to easily support further extensions of each algorithm, we modelled our approach according to the *strategy* design pattern. The strategy pattern is defined as follows [GHJV94]:

"Define a family of algorithms, encapsulate each one, and make them interchangeable.  
Strategy lets the algorithm vary independently from clients that use it."

Represented in the Unified Modelling Language (UML), the strategy pattern takes the form shown in Fig. 4.1. Since we want to allow modification, extension, and exchange of the solution scheme at runtime, we designed a general interface `BondOrderAssignmentStrategy`, which prescribes a number of functions for communication with the rest of the surrounding context. This interface is realized<sup>1</sup> by one class per solution scheme. The interface of the strategy is slim and only consists of the following functions: a constructor that accepts a pointer to the context, an initialization routine, a function to compute and return the next solution, a function to read and one to set default options, and a function to clear the current state of the algorithm.

Intermediate solutions can be stored in the `PartialBondOrderAssignment` class, which also evaluates the different heuristic and exact penalty functions. The resulting full bond order assignments are stored

<sup>1</sup> Please note that C++ does not distinguish clearly between interfaces and classes and hence, all interfaces have been implemented as regular classes, and realization has been replaced by derivation.

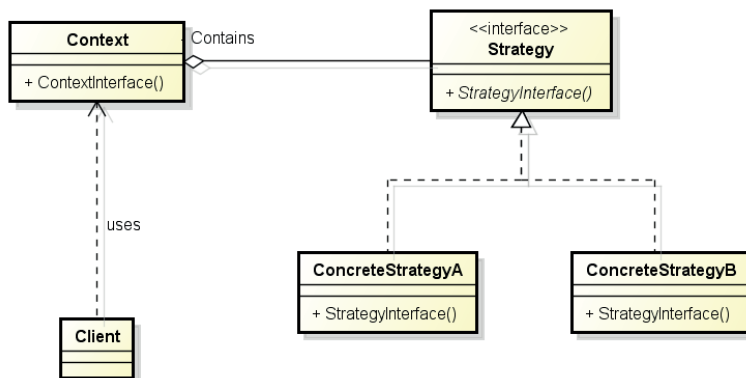


Figure 4.1.: The general strategy pattern denoted in UML.

as instances of the `BondOrderAssignment` class. The whole assignment process is controlled by the `AssignBondOrderProcessor`, which is derived from BALL's `UnaryProcessor<AtomContainer>`, allowing simple application to the different types of molecular containers of interest (`System`, `Molecule`, `Chain`, ...).

Through the strategy pattern, the user is completely insulated from the details of the implementation of the particular bond order strategy chosen. To choose between the different versions, one simply has to communicate one's choice to the `AssignBondOrderProcessor`, which then chooses the correct solution instance at runtime. The resulting scheme – with simplified interfaces – is shown in Fig. 4.2.

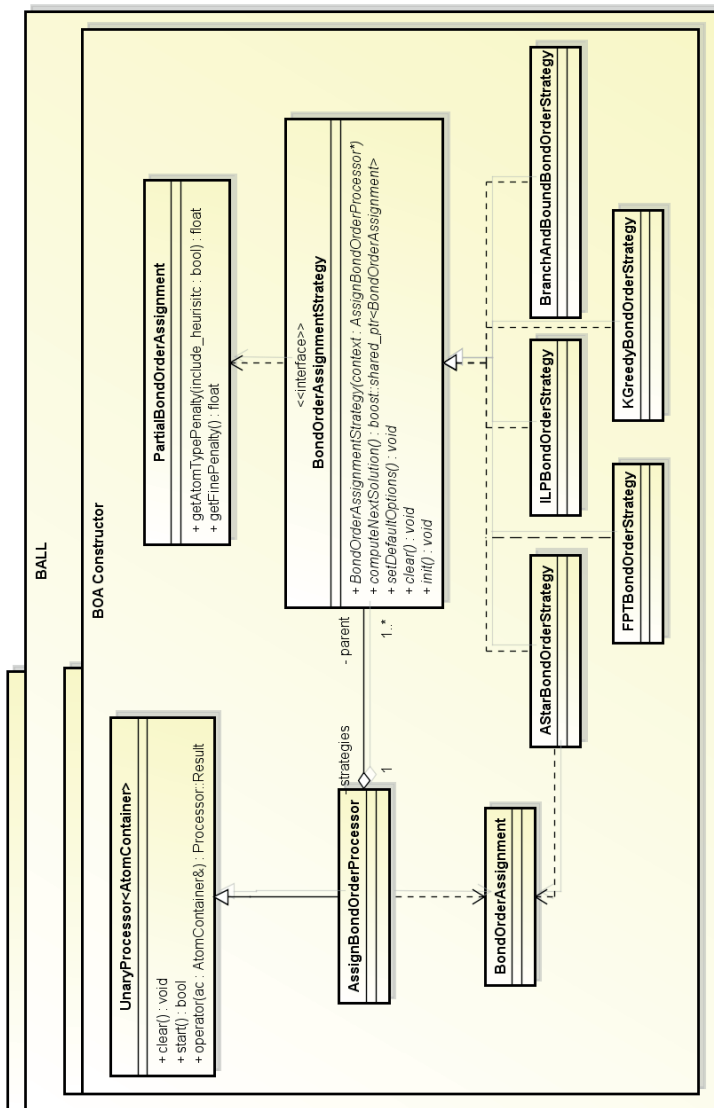


Figure 4.2.: The class model of BOA Constructor.



In general, great care has been taken to implement the approach in such a way that the new bond order assigner is as simple to use as possible for novice users while being very flexible and adaptable for the expert with more advanced needs. For instance, a variety of options (see table A.2) can be used to fine-tune the behaviour. The following code snippet shows how BOA Constructor can be used in a C++ application or library:

```
System sys;
...
AssignBondOrderProcessor bop;
bop.options.setBool(
    AssignBondOrderProcessor::Option::COMPUTE_ALSO_NON_OPTIMAL_SOLUTIONS,
    true);
...
sys.apply(bop);
Position i = bop.getNumberOfComputedSolutions();
bop.apply(0);
...
while (bop.computeNextSolution())
{
    i++;
    bop.apply(i);
}
```

#### 4.2.1. The GUI

All bond order assignment algorithms are fully integrated into BALL's graphical user interface BALL-View. All relevant options (c.f. Appendix A.2) can be edited in dialog forms before starting the processor. The computed solutions are presented in a list with 2D sketches and corresponding penalty values. The user can choose between either applying a selected assignment to the given molecule or creating a new molecule with the corresponding bond order assignment. This allows for the manual creation of molecule sets with different bond order assignments. An exemplary BALLView session showing the usage of BOA Constructor can be found in Appendix A.4.2.

#### 4.2.2. Python Interface

All functionality presented in this work is also available from the BALL Python bindings. An exemplary Python program using this functionality can be found in Appendix A.4.1.

### 4.3. Implementation of *NightShift*, *Spinster*, and *Liops*

To support our efforts towards automated NMR chemical shift prediction, we extended BALL in several ways within the course of this dissertation, in addition to the introduction of BOA Constructor mentioned above. The first functionality that had to be provided was a stable and efficient parser for the file format used to store experimental chemical shift information.

#### 4.3.1. A Grammar for CIF

Experimental NMR chemical shift data is commonly stored in the NMRStar file format [HC95]. It stores the chemical shift data as well as information about the molecular systems under consideration, the NMR spectrometer, or other experimental details. NMRStar is based on the Crystallographic

#### 4. The BALL Project: A Framework for Biomolecular Modelling

Information File (CIF [HAB91, BM02]) format, where the relation of NMRStar to CIF is equivalent to that of PDBML to XML.

The CIF format was not designed as a context free grammar in the first place and is thus hard to parse correctly. To the best of our knowledge, no complete C++ implementation of a flexible CIF parser is currently available on which we could base our implementation of NMRStar files. Thus, we decided to build our own. Our aim was to create a grammar for CIF, implemented using BISON and FLEX, that is as general as possible. In particular, it should be able to import all NMRStar files provided by the BMRB [UAD<sup>+</sup>08].

Unfortunately, the CIF format cannot be fully represented in a BISON grammar without advanced features of the lexer. Roughly speaking, some aspects of CIF, for example the white space handling and quotes, are context sensitive and require joint efforts from lexer and parser. To this end, we use the state mechanism of the flex lexer. Due to its large number of states, it is not illustrative to show the lexer as an automaton. Instead, we reproduce the final full ".l" file we developed in Appendix B.2. Given this lexer, we can finally formulate a grammar for parsing CIF. But instead of reproducing the full ".y" file for the parser generator BISON, it is much more instructive to represent the grammar in Backus-Naur form (BNF), which we do in Appendix B.2 as well. The resulting parser and lexer for reading the BMRB file format were fully integrated into BALL 1.4.1.

##### 4.3.2. Features Required for the Models Spinster and Liops

For our NMR shift prediction models Spinster and Liops, we added a variety of classes for the computation of the diverse input features. The most challenging step here was clearly the implementation and extension of the different semi-classical terms described in Section 3.2.3. Similarly difficult was the computation of the GAFF atom types employed by the Liops model, which required a complete implementation of GAFF's bond and atom typization mechanism. While the bond typing was handled by BOA Constructor, the atom typing used an implementation of GAFF's Chemical Environment and Atom Property string concepts [WWKC06], two molecular description languages similar in spirit to SMARTS. This implementation was mainly created by Sabine Müller in her Bachelor's Thesis [Mue08]. However, because GAFF and Antechamber had been extended and modified since the original implementation, we had to significantly adapt this component in BALL as well. Finally, most of the remaining features (c.f. Section 3.5.2) were simple to implement with the functionality already contained in BALL, such as Solvent Accessible Surface computation and fast geometric queries supported by BALL's hash grid implementation.

The features themselves are implemented in a strategy-like pattern, such that a driving class can trigger the computation of each feature value using a unified interface. In this way, the end user can choose the input features that should be included at run time.

#### 4.4. Manual Molecular Modelling

In the previous chapters 2 and 3, we have presented our research on automatizing several molecular modelling techniques that previously led to tedious manual work, such as the annotation or correction of bond orders, or the creation of NMR shift prediction models. But while such automated techniques become increasingly important in molecular modelling, in particular for high-throughput situations, there are still many cases where manual intervention is required. One reason for this deficiency is the quality of today's force fields and scoring functions: many molecular effects are either not yet well understood theoretically and thus not yet implemented – or are far too complex to ever be – but are known to experienced researchers. For example, the conformational changes due to allosteric effects are currently usually not covered by typical molecular modelling methods. Thus, the current state of the art usually demands a manual inspection of the results of structure prediction or docking methods. In the course of this thesis, we were also involved in several efforts towards improving the state of

the art of manual molecular modelling, in particular through improved interactive visualization. A full description of all these projects, however, would require significant amounts of space. Hence, in the interest of readability, we decided to focus the text on the automated aspects, and to only briefly give an overview on the work on manual techniques. This overview will be the topic of this section, and will be based on a publication on our work on molecular visualization which is currently in preparation and on our manuscript [MGD<sup>+</sup>10] published in the Proceedings of the 14<sup>th</sup> International Conference on Information Visualization in Biomedical Informatics (IVBI) in London, UK.

#### 4.4.1. Molecular Visualization

The comprehension of the three-dimensional geometry of individual molecules, their complexes, and their physico-chemical properties is often key for understanding biomolecular processes. For instance, rational drug design often involves the development of small molecules that are tailored to fit and fill a certain binding pocket and to interact favorably with the target molecule. Thus, visualizing complex molecules and their properties of interest, e.g. electrostatic potentials, has always been one of the cornerstones of molecular biology and related fields.

Significant attention has been paid to create a faithful and intuitive representation of three-dimensional arrangements on a two-dimensional computer screen. Apart from the use of high-end stereoscopic displays to simulate three-dimensional vision, research in molecular visualization has focused on providing the user with visual cues to improve the understanding of structural relationships. These consist, on the one hand, of a variety of different models or visualization modes of biomolecular entities (e.g. cartoon representations, ball-and-stick models, or surface renderings), each one highlighting specific aspects of the structure under consideration. On the other hand, our perception of three-dimensional objects relies on their interaction with light and simulating such an interaction significantly aids the brain in interpreting a two-dimensional picture as a representation of a three-dimensional entity. Obvious examples of such effects are shadows, light attenuation, and reflections.

Conventionally, the interactive display of molecular scenes is performed using a technique known as *rasterization*, as it is implemented, e.g., in the well-established OpenGL framework. Unfortunately, rasterization does not naturally support all of the visual effects mentioned above, even though many of them can be approximated for known types of geometry using sophisticated shading techniques. If interactivity is not an issue, but high-quality images are required, ray tracing methods are often employed, but these typically take minutes to hours to render a single picture, completely destroying interactivity. Also, they require the use of an external offline program.

Recently, real-time ray tracing evolved to combine the interactivity of rasterization based approaches with the superb image quality of ray tracing techniques. A description of the ray tracing method, and of the algorithmic advances that allowed its computation in real time, is clearly out of scope of this work. Harnessing such real-time ray tracing technology for molecular visualization greatly enhances the visual quality of molecular visualization and allows for a greatly improved spatial perception of molecular scenes, without spoiling the required interactivity. Binding pockets, for instance, can be understood much more intuitively if an accurate simulation of lighting properties, such as shadows, can be employed. During the course of this thesis, we have created the first preliminary integration of the real-time ray tracing library RTfact [GS08] into BALLView, which to the best of our knowledge comprised the first seamless integration of real-time ray tracing with molecular modelling functionality. RTfact has been developed in the group of Prof. Dr. Philipp Slusallek, one of the main experts on ray tracing technology whose research has greatly contributed towards its efficient computation.

Together with Lukas Marsalek, we designed the class interface that allows the coupling in an interactive fashion, and have worked on exposing the functionality through BALLView's user interface. This preliminary integration was greatly enhanced and extended, mostly by Stefan Nickels, Prof. Dr. Andreas Hildebrandt, Lukas Marsalek, and Iliyan Georgiev and the author of this thesis, but the main structure still remains in place.

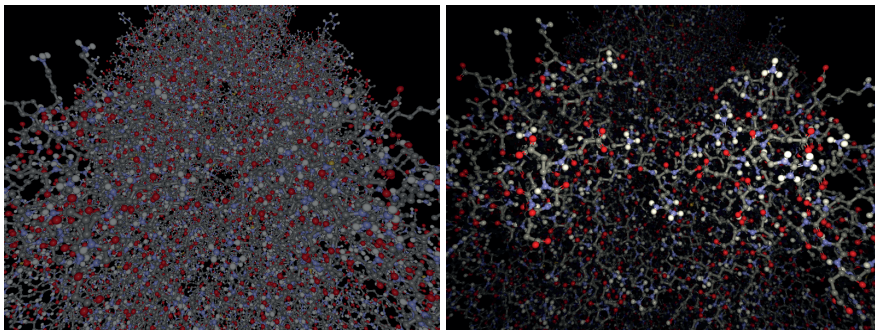


Figure 4.3.: Light attenuation improves depth perception. The **left** image shows a representation of PDB entry *1pma* rendered using only direct illumination without shadows and light attenuation. The **right** image was interactively ray traced with shadows and correct light attenuation.

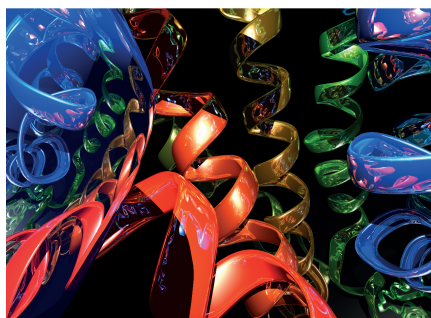


Figure 4.4.: Ray tracing enables accurate complex multiple inter-reflections and reflections from curved objects. The image shows a backbone ribbon model for PDB entry 3eml. This image was awarded with the Arts and Science Award of ISMB/ECCB 2011.

As a result of the integration, today, real-time ray tracing can be routinely used as a full replacement for the older OpenGL - based visualization. In particular, it allows to render all the standard representations and their arbitrary combinations, including the cartoon model. The interactive ray tracing is used in the standard workflow, making its use as simple as that of classical rasterization methods.

The importance of advanced visual effects as offered by RTfact is demonstrated by Fig. 4.3, which compares a typical application of rasterization with the visual result generated by RTfact. Further examples are shown in Figs. 4.4, 4.5, 4.6, 4.7, and 4.8.

#### 4.4.2. Stereoscopic Visualization

Three-dimensional vision is crucial for many applications in molecular modelling and drug design. The integration of the RTfact engine already helped in improving depth-perception significantly, but best results are achieved if it is coupled with techniques for stereoscopic visualization. While BALLView has

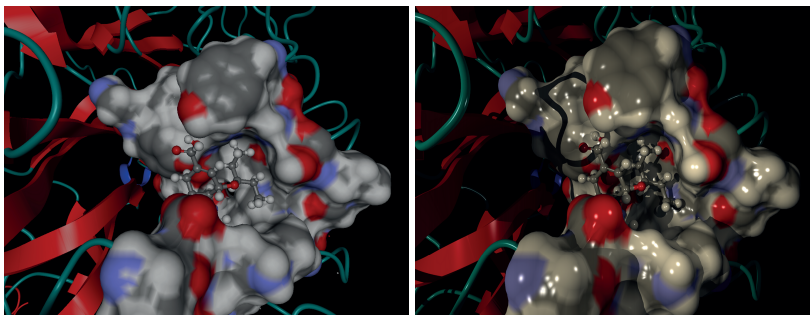


Figure 4.5.: Shadows support understanding of structural relationships.

**Left:** N1 Neuraminidase in complex with oseltamivir/Tamiflu (2hu4.pdb), using direct illumination only.

**Right:** 2hu4.pdb visualized using interactive ray tracing with realistic shadows. Please note the relative position of the coil and the SES surface, distance of the ball-and-stick model and the depth of the binding site, all of which are not apparent under the direct illumination model.

offered functionality for such stereoscopic visualization for several years, we were involved in several projects by Stefan Nickels which greatly improved this functionality. The improved functionality was also exposed through an intuitive graphical user interface, which we designed and implemented for this purpose.

#### 4.4.3. Multitouch Interaction Functionality

As discussed above, molecular visualization forms one important part of manual molecular modelling. Another aspect that is unfortunately often overlooked even though it plays an important role are the user input paradigms offered by the software. Traditionally, molecular modelling heavily relies on the mouse and keyboard as the main interfacing methods. But experience has taught that these are clearly non-optimal interfaces when working with inherently three-dimensional data. In a cooperation with Dr. Hilko Hoffmann and Prof. Dr. Philipp Slusallek, we have designed a protocol for the communication between BALLView and a home-built multitouch system of the Slusallek group that is to the best of our knowledge the very first integration of the multitouch paradigm into molecular modelling. The multitouch system created in the project is very easy to use and offers some clear advantages when compared to classical input strategies. For instance, a user standing in front of a stereo screen can easily use the multitouch table to navigate the molecules while being able to focus on the stereo screen the whole time.

The protocol that was designed to drive the multitouch project was implemented in the form of a plugin that can be used by other input methodologies as well. For instance, the plugin has formed the core of an experimental Android mobile application for controlling BALLView through the touch screen of a cell phone or an integration of BALLView into the OpenSimulator system. More details on the author's involvement in these projects, and on the researchers involved in them, can be found in Appendix 6.3.

#### 4. The BALL Project: A Framework for Biomolecular Modelling

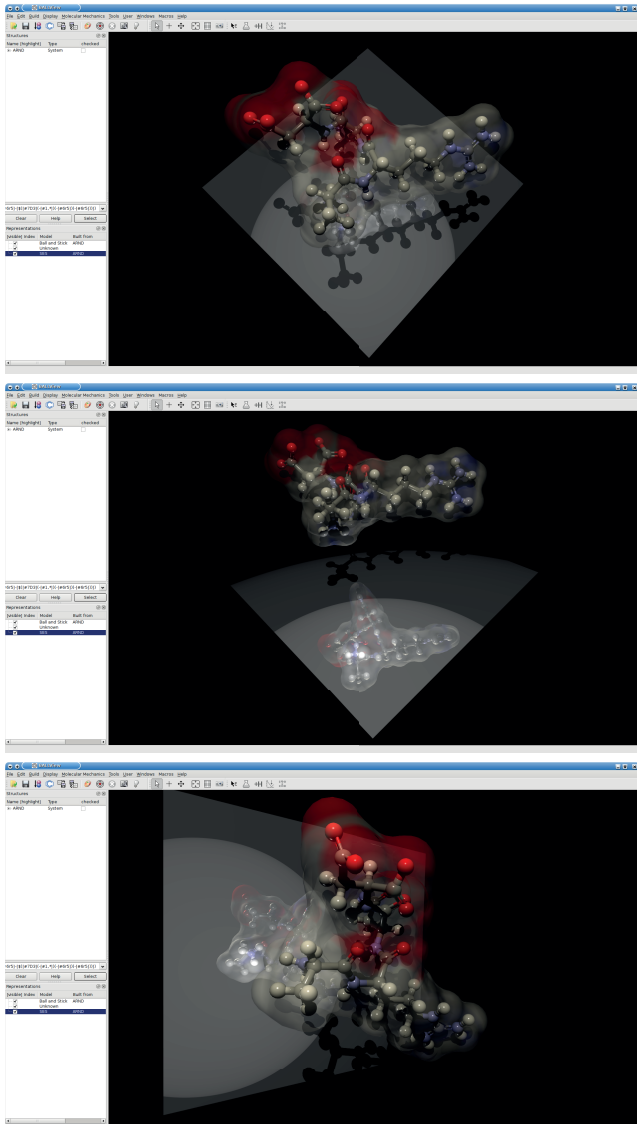


Figure 4.6.: Ray tracing easily allows artistic depictions, even with highly complex scenes.

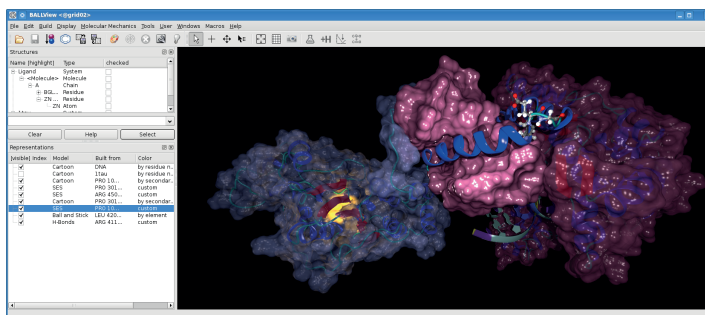


Figure 4.7.: The integration of the ray tracer into BALLView is done tightly and transparently, allowing the advanced effects reflection, shadows, light attenuation, or transparency to work seamlessly with any combination of available representations, including the cartoon models.

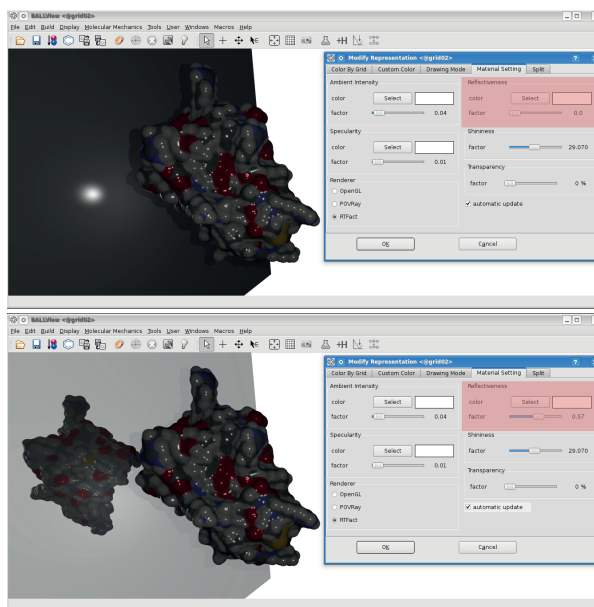


Figure 4.8.: Ray tracing enables accurate reflections, including multiple inter-reflections in real-time. Adding a reflection is as simple as dragging a material reflectivity slider between 0 and 1, giving a smooth transition between diffuse material and perfect mirror.

#### 4. The BALL Project: A Framework for Biomolecular Modelling

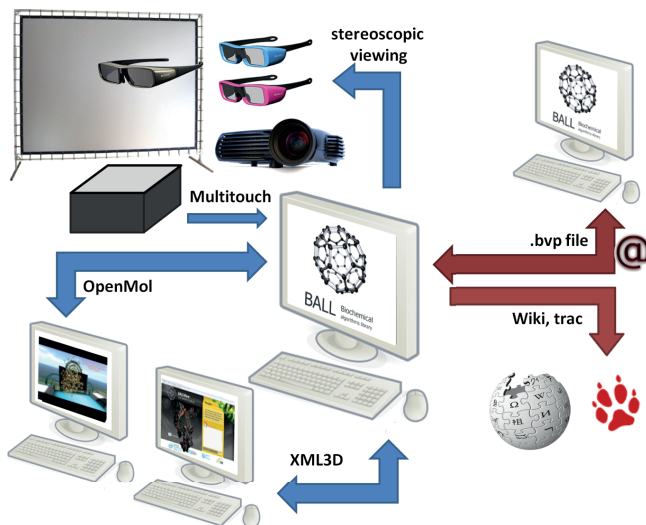


Figure 4.9.: Collaborative modelling functionality of BALLView. Red arrows indicate former approaches, blue arrows represent efforts implemented in our work.

### 4.5. Application of Ray Tracing to Automated Molecular Modelling

During our work with the Slusallek group on integrating real time ray tracing capabilities into BALLView, we noticed that the ray tracing method would be able to compute a variety of geometrical quantities required by many applications on the fly as a by-product of the visualization. In particular, we have set up a technique for computing the volumes and surface areas of proteins and their cavities that uses the ray tracer directly. The method is only based on the collection of triangles forming the molecular representation, which allows to generalize the procedure very easily to all different representations available for proteins, e.g. different surface models. It also enables more sophisticated features, such as restricting the volume or area computation to a certain region in space, e.g., close to the binding pocket. In this project, we were mainly responsible for the bioinformatical aspects, such as a clear problem definition, an interface to BALL, and the validation of the approach. More details on the author's role in the project and the researchers involved can be found in 6.3.

This method nicely combines manual molecular visualization techniques with automated modelling techniques, but the algorithmic details are out of scope of this work. Here, we will only briefly discuss the idea and the results of the approach by summarizing the relevant parts of our publication [PGD<sup>+</sup>10], to which the interested reader is referred.

#### 4.5.1. Geometric Molecular Properties Through Ray Casting

As discussed above, the interactions between biomolecules are to a large extent determined by the three-dimensional structures of the component molecules. Consequently, the efficient and accurate computation of molecular geometric properties has long since been studied extensively in Bio- and Cheminformatics.

One typical use-case for simple geometric properties is the estimation of binding free energies in the



presence of a solvent – water in the biomolecular case. To occur in a certain shape, each biomolecule has to displace a number of water molecules to form a cavity for itself. This loss of freedom of the solvent leads to an entropic effect which is a function of the molecular volume. Similarly, the surface tension of the water surrounding two individual molecules will differ from the tension around their complex, leading to a term in the free energy of binding which is a function of the solvent-exposed surface area. Other important geometric properties often focus on the presence, volume, and surface area of possible internal cavities, pockets, or tunnels inside the molecule of interest.

In [PGD<sup>+</sup>10], we proposed to use ray casting techniques, as known from computer graphics, to accurately estimate such geometric properties for arbitrary molecular surface definitions. Ray casting methods are known to parallelize very well and are able – using suitable acceleration structures – to handle even huge geometric models at interactive speeds. The only requirement posed on the molecular representation is the ability to efficiently intersect it with a ray of arbitrary direction, which is trivially possible for all tessellated surfaces, but also for more general representations.

In principle, the method employs the intersection points of rays with the molecular geometry. Using statistical sampling techniques for surface integrals, molecular surface areas can be easily estimated from these intersections. Intuitively speaking, if the normal of a triangle would be parallel to the directions of the rays, we would expect the ratio of rays hitting this triangle – and hence, the number of intersection points in this triangle – versus those missing the triangle to converge to the ratio of the area of the triangle versus the area in which rays are cast. If the direction of the triangle's normal deviates from the direction of the rays, we instead expect this ratio to be smaller by an amount that can be easily computed from elementary geometry. Thus, from the ratio of rays hitting a triangle, and from its orientation with respect to the ray, we can easily estimate the total surface area. Assuming that the molecular surface has no holes, any ray entering the molecule will leave it again. Statistically integrating the distance between these intersection points for all rays gives an estimate of the molecular volume, where care has to be taken to handle cases where the ray enters more than once. In practice, the algorithm also needs to handle holes in the surface triangulation. The details can be found in [PGD<sup>+</sup>10]. We used the real-time ray tracer RTfact [GS08] to implement our ray casting based methods.

Finally, the algorithm is able to take molecular cavities into account, i.e., to detect their presence, surface areas, and volumes. The rough idea behind cavity detection is to discretize the volume into a uniform three-dimensional grid which stores whether a given voxel is inside or outside the protein. Using a three-dimensional flood fill algorithm then gives the desired information.

#### 4.5.2. Results

As described in [PGD<sup>+</sup>10], we have chosen several molecules of the data set described in [SC08] for evaluating our results. Here, we compare the results to the established MSMS [SOS96] program, which is used to produce both reference estimates for volumes and areas, and the triangulated surfaces we test our methods with. Preprocessing was done in the following manner: we downloaded the molecules from PDB [BWF<sup>+</sup>00], checked the structure against BALL's Fragment database, added missing hydrogens, and deleted ligand, cofactors, and water molecules using BALL. We then ran MSMS to compute the surface area, volume, and triangulation of the SES surfaces with all contained cavities.

In general, we have found that our volume estimates match the MSMS results very closely, even when using a low sampling density, as can be seen in table 4.3. For the surface areas, the method still works well, but is less stable in general, as can be seen in table 4.2. The larger errors in the surface estimates as compared to the volumes are most probably a result of inconsistent input surface triangulations and a numerical problem known as "cosine clamping", as discussed in [PGD<sup>+</sup>10]. Both of these influences can be corrected for in the algorithm, and will be taken into account in future work.

Similar to the area estimation, our implementation of cavity detection is also a proof of concept in the sense that it is not yet optimized algorithmically and numerically. Nonetheless, we found that it works

#### 4. The BALL Project: A Framework for Biomolecular Modelling

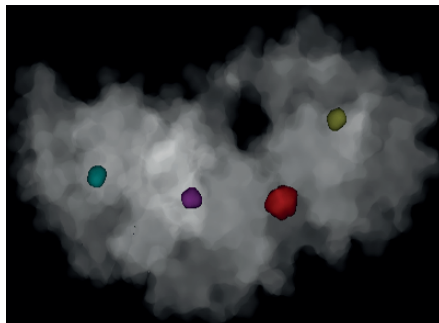


Figure 4.10.: The volume of molecule with PDB id 1g2a computed per pixel. Four cavities have been detected.

very reliably, given that the input triangulations of the cavity regions are correct and well formed. Exemplary results of the method for selected molecules can be found in table 4.1. Finally, Fig. 4.10 shows the volume of PDB entry 1g2a with four cavities.

In our experiments, we found our method to be computationally very efficient, even though it has not been optimized yet. Its performance is clearly on-par with highly optimized but more specialized methods, such as MSMS. In contrast to these, our new method is general enough that a user will be able to apply it to any kind of molecular model, even in conjunction with more sophisticated concepts like clipping regions, without the need to modify the program. In addition, it is fast enough that it will not be the bottleneck in any of its application scenarios. In summary, we conclude that ray casting methods are very suitable to address problems in structural bioinformatics, apart from the visualization context they were originally designed for. Our future work will focus on numerical stabilization of the above described methods, and on exploring additional geometric features. Further work will be aimed at other kinds of surface representations, as our method is applicable to any representation that can be intersected with a ray. This, for instance, holds for many implicit molecular surface definitions (implicit SES, skin surfaces, etc.), where triangulation can be altogether avoided.

PDB Id	Cavities	Vol. Diff.(%)	SA Diff.(%)
1g2a	4	-1.01	-14.75
1qjp	4	2.40	-7.52
1e02	5	-4.67	-12.84
256l	3	0.05	0.04
2ihl	1	2.51	-1.16
1ton	4	-4.29	-14.70
1gar	3	2.94	-1.13

Table 4.1.: Number of cavities along with volume and surface area differences for selected molecules compared to MSMS [SOS96].

Resolution	Min Diff.(%)	Max Diff.(%)	Avg. Diff.(%)
50 × 50	0.31	13.68	5.37
100 × 100	1.75	10.35	4.79
200 × 200	1.78	11.74	4.92
400 × 400	2.53	15.28	5.04

Table 4.2.: Minimum, maximum, and average differences between surface areas measured by our method and MSMS [SOS96] at different uniform sampling resolutions, averaged over 107 molecules.

Resolution	Min Diff.(%)	Max Diff.(%)	Avg. Diff.(%)
50 × 50	0.003	0.442	0.135
100 × 100	0.002	0.366	0.128
200 × 200	0.002	0.376	0.127
400 × 400	0.008	0.364	0.118

Table 4.3.: Minimum, maximum, and average differences between volumes measured by our method and MSMS [SOS96] at different uniform sampling resolutions, averaged over 107 molecules.



## 5. Conclusion

The aim of this work, as outlined in the introduction, was the development of methods for the efficient and accurate approximation of complex atomic and molecular properties. As long as full quantum approaches are computationally too costly for many applications on biomolecular systems, approximation schemes are often the only feasible approach.

Among those approximations, the classification of chemical bonds into a small set of bond types is one of the most crucial for molecular modelling. If a sufficiently fine-grained set of classes is used, this approximation is known to work very well in practice. In force field based molecular modelling approaches, however, several important bond types, such as delocalized or aromatic bonds, can typically not be correctly represented. Instead, these schemes work with integer bond orders, such that every bond has to be described as a single, double, triple, or possibly quadruple bond. If a molecule contains bonds that are, e.g., predominantly aromatic, the mapping to an integer type will typically not describe the bond's properties very well. In addition, the mapping to integer bond orders in these cases is ambiguous. Since bond order information is often missing, we decided to address the problem of automated bond order assignment.

Our contribution in this area is the development of the efficient, exact, and easily extensible method *BOA Constructor* for the computation of bond order assignments from the molecular topology. If, additionally, three-dimensional atomic coordinates are known, the BOA Constructor can integrate these into its scoring procedure to further improve the results. Our method uses a formulation of the bond order assignment problem developed by Wang et al., and is, to the best of our knowledge, the only technique that produces provably optimal results. At the heart of BOA Constructor lies the reformulation of the bond order assignment problem as a combinatorial optimization problem, for which we developed three solution strategies: an A-Star (A\*) algorithm, an Integer Linear Program (ILP), and a Fixed-Parameter Tractability (FPT) approach. Finally, the method is easily extensible due to our user definable scoring function.

As demonstrated by our experiments, the efforts put into bond order assignment clearly proved worthwhile. First and foremost, a comparison of our results with established heuristic programs shows that the guaranteed optimality of our approach improves the results in practice. Similarly important is the extensibility of the approach. In its present formulation, the method will work well for proteins, RNA, DNA, and common ligand atoms alike. If a user wants to apply the technique to molecules with a currently not covered chemistry, as they might occur in unusual ligands, e.g., he will only need to add penalty values for the new chemical combinations into an XML file. Due to the fundamental importance of bond orders for modelling approaches, BOA Constructor has many important applications. One of these is the implementation of atom typization techniques as used in force field development. Such atom types allow to concisely represent the chemical neighbourhood of an atom, and thus subsume large amounts of useful chemical information. We thus combined BOA Constructor with an implementation of the chemical description languages used by the GAFF force field. This resulted in a very efficient, provably exact, and easily extensible implementation of a very general atom typization strategy. Based on these atom types, we proceeded to implement the GAFF force field.

Apart from these classical application scenarios of bond order assignment, we found that reliable bond- and atom typization can also be used to support the application of statistical learning techniques to molecular modelling problems. As in other fields of Bioinformatics, statistical learning becomes increasingly important at the structural level, e.g., in the prediction of binding free energies. A particularly interesting development are hybrid strategies, where partial knowledge about the physical

## 5. Conclusion

and chemical laws governing the system of interest is combined with a statistical regression technique. Ideally, this yields a predictor that performs noticeably better than either of the two methods would perform individually. Statistical learning techniques strongly rely on the availability of informative, efficiently computable features that somehow correlate with the desired molecular or atomic property. Since bond- and atom types efficiently encode important chemical information, they are optimally suited to derive such features. Thus, the availability of reliable bond- and atom typing techniques allowed us to address an important unsolved problem in structural bioinformatics: the prediction of protein chemical shifts in the presence of ligands. In our approach, we use atom types to describe the chemistry of the ligand atoms, and combine them with additional molecular and atomic features to compute a statistical model. In practice, however, we found that the development of such a model first required us to develop a suitable model for NMR chemical shifts of proteins in the absence of ligands. While many such models have been proposed in the literature, none of the currently available techniques could be used in conjunction with our new ligand-based features. Since the development of such models up to now required large manual effort, we decided to solve the problem more generally by creating an automated pipeline – called *NightShift* – for data set generation and model training for protein chemical shift prediction. Using the pipeline, we exemplarily trained two prediction models as a proof of concept, one based on linear regression and one based on a random forest. The models, however, turned out to outperform established, manually developed, techniques even though they had been automatically generated. The random forest model – which we call *Spinster* – in particular turned out to be very accurate.

Using the *NightShift* pipeline, we finally combined the *Spinster* model with our atom type-derived features to produce a model for protein shifts in the influence of ligands. This model – called *Liops* – indeed demonstrates the value of using atom- and bond types for solving such a complex problem, but still leaves room for improvement.

In summary, with this work, we rendered possible the computation of bond orders in an exact, efficient, and extensible manner, used them for reliable atom typization, and showed that the resulting types are perfectly applicable to problems apart from the molecular force fields for which they had initially been derived. Due to their generality, these features are optimally suited for problems that contain different types of molecules like the protein ligand cases, as, e.g., known from drug design.

With the availability of these features, and with the proof-of-concept application to protein-ligand NMR chemical shift prediction, we see several possible routes of exciting future development. Apart from the possibilities for further improvement of the individual techniques as laid down in the respective chapters of this work, we particularly want to mention the application of chemical shift prediction to the field of RNA structure elucidation. Here, NOE constraints can often not be derived, and chemical shift information is all that is obtained from the NMR experiments. The ability to accurately predict such RNA chemical shifts for a variety of putative conformations would be of great value as an experimentally derived scoring function. A second application scenario which we believe to be of great interest would be the development of modern scoring functions for protein-ligand docking. Here, we expect representative atom types to also carry a wealth of information for statistical learning techniques.

Thus, while the problem of approximating complex atomic and molecular properties, such as chemical shifts, in a systematic and efficient way is still far from being solved, we believe that the results of this thesis can serve as a further step along the way towards that goal. Chemical information as encoded in atom types is, in our opinion, one of the most promising kinds of predictors for many applications of statistical learning in molecular modelling, and the results described here allow not only to compute this information in a very general fashion, but also allow the user or developer to easily define his own atom types – a process that typically required the complete rewrite of complex atom typing procedures previously. Since all of our results have been made publicly available in the open source framework BALL, we hope that they will be of use in many exciting research projects in the future.

## 6. Authors Contributions

### 6.1. Bond Order Assignment

Large parts of our work on *BOA Constructor* have been previously published in the proceedings of GCB 2009 [DRLH09] and in the journal *Bioinformatics* [DRB<sup>+</sup>11].

The A\* approach, the heuristics, the new penalty table, and the integration of structural information were developed and implemented by myself. I also developed and performed the test and validation scenarios.

In addition, I helped with the formulation of the fixed parameter approach, the Java implementation of which was performed in the group of Sebastian Böcker. Kai Dührkop from this group also provided a preliminary C++ implementation, which I redesigned and integrated into BALL.

Furthermore, I developed the initial integer linear program together with Prof. Dr. Andreas Hildebrandt and Dr. Alexander Rurainski, who then performed an initial implementation and further reformulations for increased computational efficiency. The integration into BALL was again performed by myself.

The GAFF atom typization was initially implemented by Sabine Müller. Since GAFF had been developed further since the initial implementation in BALL, I had to extend the typization mechanism to handle additional syntactic elements in the environment strings. Also, I implemented the bond order types, parameter estimation and charge correction according to the AM1BCC scheme as well as the force field parameter inference which allows to use GAFF in BALL.

### 6.2. NMR Shift Prediction

Our work on the NMR framework and NMR hybrid shift prediction has so far been partially published [DLH11], and more publications on the subject are currently in preparation.

The pipeline *NightShift*, which allows automated data set generation and model training will be presented in a manuscript that is currently finalized for publication. This publication will also discuss the integration of *NightShift* into the Galaxy workflow system [GNTT10] which further improves the user experience. This integration was also performed by myself, but we considered it out of scope of the topic of this thesis. Similarly, a publication on the pure protein model *Spinster* is currently in preparation, and will also be extended with a Galaxy integration. In addition, both *NightShift* and *Spinster* have been presented in a talk I gave at the 25<sup>th</sup> Molecular Modelling Workshop 2011 in Erlangen, Germany.

Our Protein-Ligand model *Liops*, finally, has been presented in a talk I gave at the German Conference on Bioinformatics (GCB) 2011.

All three components, *NightShift*, *Spinster*, and *Liops*, were developed and implemented by myself. The CIF parser used by the pipelines was developed in collaboration with Prof. Dr. Andreas Hildebrandt. Furthermore, additions to the pipeline (rotamer and packing features) were the topic of the Bachelor thesis of Simon Loew, which I supervised.

### 6.3. BALL Project

In the course of this thesis, I extended the functionality of BALL and BALLView significantly. In addition to work related to NMR shift prediction and automated bond order assignment, the added

## 6. Authors Contributions

features encompassed important core components, such as the peptide builder, which can be used to create peptides from a given sequence, molecular editing functionality, prediction of hydrogen bonds, and secondary structure prediction methods. Further enhancements were generated in a number of smaller projects and Bachelor Theses supervised by myself, such as an INCHI-processor (Bachelor Thesis of Alexander Zapp), alignment functionality (Bachelor Thesis of Nikola Koch), atom propensity estimation (in collaboration with Roche Diagnostics GmbH), learning the penalty table for automated bond order assignment (Bachelor Thesis of Alexander Gress), and the pKa computation of small ligand molecules (Bachelor Thesis of Adrian Fritz).

The design of the initial integration of ray tracing into BALLView was performed by Lukas Marsalek and myself. This project was then further pursued together with Prof. Dr. Andreas Hildebrandt, Lukas Marsalek, Ilyan Gregoriev, Stefan Nickels, and Prof. Dr. Philipp Slusallek, and led to several conference posters, talks, and publications [DGM<sup>+</sup>09, MGD<sup>+</sup>10, HDS<sup>+</sup>11] as well as a Technology Track demonstration at ISMB/ECCB 2009 in Stockholm, Sweden.

Another project that evolved from this work is the improvement of stereoscopic visualization functionality in BALLView. The stereoscopic integration was done by Stefan Nickels, Prof. Dr. Andreas Hildebrandt, and myself.

This work finally led to the Kiosk-Project displayed at "Woche der Wissenschaften 2011" in Saarbrücken, on the MS Wissenschaft 2011 and an exhibition "Abenteuer Wissenschaft 4: Der Mensch (Version1.0)" in the "Haus der Wissenschaften" of the Graz university. In this project, I was mainly responsible for the design and choice of presented content. The entire project was a cooperation of Sabine Müller, Stefan Nickels, Daniel Stöckel, Prof. Dr. Andreas Hildebrandt, and myself.

In the "Measuring 3D geometric properties"-project, I was mainly responsible for the bioinformatical aspects, such as a clear problem definition, an interface to BALL, and the validation of the approach. In addition, I worked with Mike Phillips, Lukas Marsalek, Iliyan Georgiev, Prof. Dr. Andreas Hildebrandt, Stefan Nickels, and Prof. Dr. Philipp Slusallek on the formulation of the algorithm.

In a cooperation with Dr. Hilko Hoffmann and Prof. Dr. Philipp Slusallek, I was involved in the design and implementation of a protocol for the communication between BALLView and a home-built multitouch system of the Slusallek group. The setup was presented on June 6<sup>th</sup> 2009 at the closing conference of the Foresight Process in Bonn 2009. This project further evolved into the OpenMol-Project, a preliminary integration of BALL and the OpenSimulator project, where I was involved in the implementation of the core components as well as parts of the interface on the BALLView side together with Stefan Nickels and Sabine Müller. In addition, we also designed the test case of modelling Aspirin. The work on the OpenSimulator side was performed by Kugamoorthy Gajananan, Dr. Arturo Nakasone, Stefan Nickels, Prof. Dr. Andreas Hildebrandt, and Prof. Dr. Helmut Prendinger.



## A. Supplementary Information on BOA Constructor

This appendix contains supplementary or advanced information on our framework for automated bond order assignment BOA Constructor that we deemed out of scope or tangential to the main text. This includes the details of the new penalty table A.1, selected molecular examples for the penalty rules A.2, heuristic solvers for the bond order assignment problem A.3, implementational details A.4, and further performance measurements A.5.

### A.1. A New Penalty Table for Bond Order Assignment

Here, we reproduce the full definition of our new penalty table for bond order assignment.

id	atom	SMARTS expression								description
		penalties								
		av0	av1	av2	av3	av4	av5	av6	av7	
1	H	SMARTS([#1+])								new, covers positively charged hydrogens
		0	-	-	-	-	-	-	-	
2	H	SMARTS([#1])								taken from Antechamber (rule 1), but prohibited av0 and av2
		-	0	-	-	-	-	-	-	
3	C	SMARTS([\$([#6D1](~[#7D2]))])								taken from Antechamber (rule 6)
		-	-	-	0	1	32	-	-	
4	C	SMARTS([#6D1])								taken from Antechamber (rule 7)
		-	-	-	1	0	32	-	-	
5	C	SMARTS([#6])								taken from Antechamber (rule 9)
		-	-	64	32	0	32	64	-	

A. Supplementary Information on BOA Constructor

id	atom	SMARTS expression							description	
		penalties								
		av0	av1	av2	av3	av4	av5	av6	av7	
6	N	SMARTS([\$([#7D1](~[#7D2]))])							taken from Antechamber (rule 11)	
		-	-	0	0	-	-	-		-
7	N	SMARTS([#7D1])							taken from Antechamber (rule 12)	
		-	-	3	0	32	-	-		-
8	N	SMARTS([\$([#7D2](~[#7D1]))])							taken from Antechamber (rule 13)	
		-	-	-	1	0	-	-		-
9	N	SMARTS([#7D2])							taken from Antechamber (rule 14)	
		-	-	4	0	2	-	-		-
10	N	SMARTS([\$([#7D3](~[#8D1-,#16D1-])~[#8D1,#16D1]))])							new, covers charged nitrogen as shown in Fig. A.1(a)	
		-	-	-	32	0	32	64		-
11	N	SMARTS([\$([#7D3](~[#8D1,#16D1])~[#8D1,#16D1]))])							taken from Antechamber (rule 15)	
		-	-	-	64	32	0	32		-
12	N	SMARTS([\$([#7D3](~[#8D1,#16D1])(~[!#8&!#16,!D1])(~[!#8&!#16,!D1]))])							taken from Antechamber (rule 16), added av5=0 covering e.g. pyrimidin-1-oxide	
		-	-	-	1	0	0	-		-
13	N	SMARTS([#7D3])							taken from Antechamber (rule 17) but prohibited av2	
		-	-	-	0	1	2	-		-
14	N	SMARTS([#7D4+])							new, covers charged nitrogen as shown in Fig. A.1(b)	
		-	-	-	-	0	-	-		-
15	N	SMARTS([#7D4])							taken from Antechamber (rule 18) but prohibited av2 and av3, and set av4 to 0, and av5 to 1	
		-	-	-	-	0	1	-		-
16	N	SMARTS([#7])							new default fallback	
		-	-	0	0	0	0	-		-

A.1. A New Penalty Table for Bond Order Assignment

id	atom	SMARTS expression							description
		penalties							
		av0	av1	av2	av3	av4	av5	av6	av7
17	O	SMARTS([#8D1-])							new, covers charged oxygens as shown in Fig. A.1(c)
		-	0	32	-	-	-	-	
18	O	SMARTS([\$([#8D1 (~[#6D3~[#8D1,#16D1]))])])							new, covers oxygens in carboxylic groups as shown in Fig. A.1(d)
		-	0	0	-	-	-	-	
19	O	SMARTS([#7D3 (~[#8D1,#16D1]) (~[#8&!#16,!D1]) (~[#8&!#16,!D1]) AND element(0))							taken from Antechamber (rule 19)
		-	0	1	-	-	-	-	
20	O	SMARTS([#8D1])							taken from Antechamber (rule 20)
		-	1	0	64	-	-	-	
21	O	SMARTS([#8D2])							taken from Antechamber (rule 21) but prohibited av1
		-	-	0	64	-	-	-	
22	O	SMARTS([#8D3])							new, covers oxygens as shown in Fig. A.1(e)
		-	-	-	0	-	-	-	
23	P	SMARTS(#15D1)							taken from Antechamber (rule 22)
		-	-	2	0	32	-	-	
24	P	SMARTS(#15D2)							taken from Antechamber (rule 23)
		-	-	4	0	2	-	-	
25	P	SMARTS(#15D3)							taken from Antechamber (rule 24) but prohibited av2
		-	-	-	0	1	2	-	
26	P	SMARTS([\$([#15D4 (~[#8D1,#16D1]) (~[#8 &!#16,!D1]) (~[#8 &!#16,!D1,#8D1-,#16D1-]) (~[#8D1-,#16D1-]))])])							new, covers phosphorus with 4 bonds as shown in Fig. A.1(f)
		-	-	-	-	-	0	32	
27	P	SMARTS([\$([#15D4 (~[#8D1,#16D1]) (~[#8 &!#16,!D1]) (~[#8 &!#16,!D1]) (~[#8D1,#16D1]))])])							taken from Antechamber (rule 25)
		-	-	-	-	-	32	0	

A. Supplementary Information on BOA Constructor

id	atom	SMARTS expression								description
		penalties								
		av0	av1	av2	av3	av4	av5	av6	av7	
28	P	SMARTS ([\$( [#15D4] (~[#8D1,#16D1]) (~[#8D1,#16D1]) (~[#8D1,#16D1]) (~[!#8 & !#16,!D1]))])								taken from Antechamber (rule 26)
		-	-	-	-	-	-	32	0	
29	P	SMARTS(#15D4)								taken from Antechamber (rule 27) but prohibited av3
		-	-	-	-	1	0	32	-	
30	S	SMARTS([#7D3] (~[#16D1]) (~[!#8 & !#16,!D1]) (~[!#8 & !#16,!D1])) AND element(S)								taken from Antechamber (rule 28)
		-	0	1	-	-	-	-	-	
31	S	SMARTS ([\$( [#16D1] (~[#6D3] ~[#8D1,#16D1]))])								new, covers -CSS functional groups as shown in Fig. A.1(g)
		-	0	0	-	-	-	-	-	
32	S	SMARTS([#16D1])								taken from Antechamber (rule 29)
		-	2	0	64	-	-	-	-	
33	S	SMARTS ([\$( [#16D2] (~[#8D1,#16D1]) ~[#8D1,#16D1]))])								new, covers sulfur dioxide as shown in Fig. A.1(h)
		-	-	-	-	0	-	-	-	
34	S	SMARTS([#16D2])								taken from Antechamber (rule 30) but prohibited av1
		-	-	0	64	-	-	-	-	
35	S	SMARTS([#16D3])								taken from Antechamber (rule 31)
		-	-	-	1	0	2	2	-	

A.1. A New Penalty Table for Bond Order Assignment

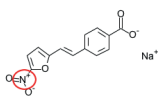
id	atom	SMARTS expression							description
		penalties							
		av0	av1	av2	av3	av4	av5	av6	av7
36	S	SMARTS([\\$([#16D4] (~[#8D1, #16D1]) (~[#8D1, #16D1]) (~[!#8&!#16, !D1]) (~[!#8&!#16, !D1])]))							taken from Antechamber (rule 32)
		-	-	-	-	-	-	0	
37	S	SMARTS([\\$([#16D4] (~[#8D1-, #16D1-]) (~[#8D1, #16D1]) (~[#8D1, #16D1]) (~[!#8&!#16, !D1])]))							new, charged $\text{NSO}_3^-$ group as shown in Fig. A.1(i)
		-	-	-	-	-	-	0	
38	S	SMARTS([\\$([#16D4] (~[#8D1, #16D1]) (~[#8D1, #16D1]) (~[#8D1, #16D1])]))							taken from Antechamber (merged rule 33 and 34)
		-	-	-	-	-	-	32	
39	S	SMARTS([\#16D4])							taken from Antechamber (rule 35)
		-	-	-	-	4	2	0	
40	S	SMARTS([\#16D5])							new default fallback
		-	-	-	-	-	2	0	
41	F	SMARTS([\#9])							taken from Antechamber (rule 2)
		64	0	64	-	-	-	-	
42	Br	SMARTS([\#35])							taken from Antechamber (rule 4)
		64	0	64	-	-	-	-	
43	I	SMARTS([\#53])							taken from Antechamber (rule 5)
		64	0	64	-	-	-	-	
44	Cl	SMARTS([\#17-])							new, covering charged chlorine
		0	-	-	-	-	-	-	
45	Cl	SMARTS([\#17])							taken from Antechamber (rule 3), added av3, av4 covering ions
		64	0	64	128	128	-	-	

A. Supplementary Information on BOA Constructor

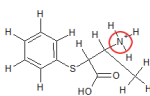
id	atom	SMARTS expression							description	
		penalties								
		av0	av1	av2	av3	av4	av5	av6	av7	
46	Si	SMARTS ([#14])							taken from GAFF (rule 10)	
		-	-	-	-	0	-	-		-
47	Li	SMARTS ([#3])							new, covering ions	
		0	-	-	-	-	-	-		-
48	Na	SMARTS ([#11])							new, covering ions	
		0	-	-	-	-	-	-		-
49	Mg	SMARTS ([#12])							new, covering ions [Nac13]	
		0	-	-	-	-	-	-		-
50	K	SMARTS ([#19])							new, covering ions	
		0	-	-	-	-	-	-		-
51	Ca	SMARTS ([#20])							new, covering ions	
		0	-	-	-	-	-	-		-
52	Cu	SMARTS ([#29])							new, covering ions	
		0	-	-	-	-	-	-		-
53	Zn	SMARTS ([#30])							new, covering ions	
		0	-	-	-	-	-	-		-
54	Fe	SMARTS ([#26])							new, covering ions	
		0	-	-	-	-	-	-		-

Table A.1.: New atom penalty classes. We converted the atom type descriptions as denoted in [WWKC06] to SMARTS expressions and extended the list to cover more atomic elements and to handle charged atoms.

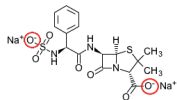
## A.2. Molecular Structures for the new Rules in the Penalty Table



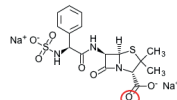
(a) Rule 10



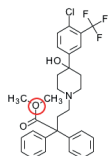
(b) Rule 14



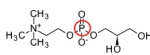
(c) Rule 17



(d) Rule 18



(e) Rule 22



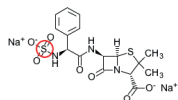
(f) Rule 26



(g) Rule 31



(h) Rule 33



(i) Rule 37

Figure A.1.: Example molecules for different rules in the penalty table.

A. Supplementary Information on BOA Constructor

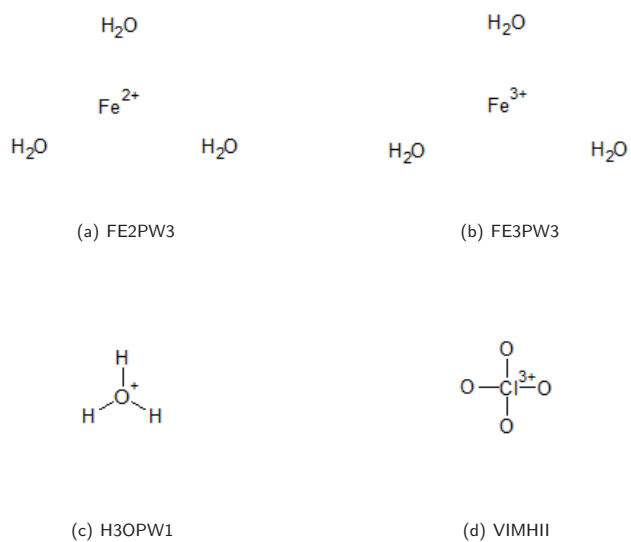


Figure A.2.: Molecules from the MMFF94 test set not covered by the original Antechamber penalty table.



### A.3. New Heuristic Approaches for Bond Order Assignment

In this appendix, we discuss two heuristic approaches to the bond order assignment problem that we developed as potential alternatives to the optimal ones. The main reason for studying these approaches is speed: as described in Section 2.4, the heuristic Antechamber approach often fails to compute the correct solution but is very computationally efficient. In this section, we test systematic approximation schemes as opposed to the very ad-hoc heuristic of Antechamber. Due to the great computational efficiency of our optimal solvers, the heuristic approaches were not needed in our experiments, but might be of interest for application to extremely large systems or as the basis for further extensions to the method. Hence, we reproduce them here for completeness.

#### A.3.1. K-Greedy

A greedy algorithm splits an optimization problem into a sequence of decisions and favours at each step a locally optimal choice, hoping that this strategy will lead to a global optimum. Taking at each step not only the single best but the  $k$  best choices into account leads to the  $k$ -greedy algorithm. In general, greedy algorithms can not guarantee to find an optimal solution, but often lead to acceptable estimates.

Mapping the bond order assignment problem onto a  $k$ -greedy algorithm is straightforward: while descending the bond order assignment tree (compare Section 2.6.1), a  $k$ -greedy algorithm expands at each layer  $i$  only the best  $k$  nodes, representing the  $k$  partial bond order assignments with smallest atomic penalty score *aps*.

Our implementation is denoted in Alg. 5 with  $k$  denoting the number of nodes that will be expanded in the next layer. For convenience, we assume  $k > \mu$ . Furthermore, let  $\mathcal{GPQ}$  denote a priority queue that stores the set of the  $k$  best nodes (partial bond order assignment) in each layer. It is typically implemented as a priority queue of fixed length.

Obviously, the order in which the bonds are considered has an influence upon the bond order assignments found by the  $k$ -greedy algorithm. Imagine two conformations differing in the orders of two bonds  $i$  and  $j$ , where the first conformation's order of bond  $i$  is slightly more penalized than the second conformation's order of bond  $i$ , whereas the first conformation's order of bond  $j$  has a significantly smaller penalty as the second conformation's order of bond  $j$ . Considering bond  $i$  before  $j$  during the algorithm favours the second conformation over the first one and might lead to an exclusion of the first conformation from further expansions, although the first conformation finally would yield the better total penalty score.

In total, the  $k$ -greedy bond order algorithm computes and stores a set of not necessarily optimal but hopefully "good" bond order assignments.

#### A.3.2. Branch & Bound

The main disadvantage of the  $k$ -greedy heuristic is that it cannot guarantee optimality, while the A\*-search suffers from complicated and expensive heuristic evaluations and in some cases an enormous search tree size. Combining A\*-search and  $k$ -greedy yields a branch & bound algorithm, that on the one hand is able to assure optimality while on the other hand dispenses complicated search heuristics and reduces the search tree size. To guarantee optimality, however, the algorithm needs to explore every leaf of the tree that is not cut away by the bound. This is usually much too expensive, and hence, it is often used in a heuristic fashion: the first leaf that is encountered is returned as an estimate of the optimum. This leads to a usually very efficient heuristic, that is known to work well if the bound is tight, but does not guarantee optimality.

In the course of a branch & bound algorithm (see Alg. 6), a search tree is generated by splitting the optimization problem into smaller subproblems. This step is denoted as branching. For the resulting

---

**Algorithm 5**  $k$ -greedy – bond order algorithm (molecule  $M$  with  $n$  bonds,  $k$ )

---

```

1:  $\mathcal{GPQ} \leftarrow \{r\}$ 
2:  $b_1 :=$  is the first free bond in  $M$ 
3: for all bond orders  $i$  do
4:    $\mathcal{GPQ} \leftarrow \mathcal{GPQ} \cup \text{bondOrders}(\{b_1 \leftarrow i\})$ 
5: end for
6: for all bonds  $b \neq b_1$  do
7:   if  $b$  has fixed bond order  $o$  then
8:     for all  $w \in \mathcal{GPQ}$  do
9:        $\text{addNode}(\mathcal{GPQ}, \text{bondOrders}(w, \{b \leftarrow o\}), k)$ 
10:    end for
11:   else
12:     for all bond orders  $i$  do
13:       for all  $w \in \mathcal{GPQ}$  do
14:          $\text{addNode}(\mathcal{GPQ}, \text{bondOrders}(w, \{b \leftarrow i\}), k)$ 
15:       end for
16:     end for
17:   end if
18: end for
19: for all  $w \in \mathcal{GPQ}$  do
20:   store bond order configuration as denoted in  $w$ 
21: end for
22: return
23:
24: def  $\text{addNode}(\bar{w}, \mathcal{GPQ}, k)$ :
25: if  $\text{len}(\mathcal{GPQ}) < k$  then
26:    $\mathcal{GPQ} \leftarrow \mathcal{GPQ} \cup \{\bar{w}\}$ 
27: else
28:   if  $g(\bar{w}) < \max_{w \in \mathcal{GPQ}} g(w)$  then
29:      $\mathcal{GPQ} \leftarrow \mathcal{GPQ} \setminus \{\arg \max_{w \in \mathcal{GPQ}} g(w)\}$ 
30:      $\mathcal{GPQ} \leftarrow \mathcal{GPQ} \cup \{\bar{w}\}$ 
31:   end if
32: end if
33: return  $\mathcal{GPQ}$ 

```

---

subproblems, an upper and lower bound is computed and the search tree is pruned at a node  $w$  if the subproblem represented by the node  $w$  exceeds the boundaries.

For mapping the bond order assignment problem onto the branch & bound algorithm we understand the bond order assignment tree (see Section 2.6.1) to represent a branching, the result of a first run of the  $k$ -greedy algorithm as an upper bound, and the value of the search heuristic  $h^*(\bar{w})$  as a lower bound. To be more flexible, we offer an additional pruning factor  $p$  that is multiplied with the upper boundary. This will allow to generate solutions that are sub-optimal by a factor of  $p$ . The final algorithm is shown in Alg. 6.

Let  $k$  denote the size of the greedy set providing the upper bound for pruning and  $p$  the pruning factor that is multiplied with the upper boundary. Please note that both parameters  $p$  and  $k$  have an influence upon the size of the search tree and the number of solutions provided by the algorithm. The function  $f(\bar{w})$  plays the role of a lower bound, where  $f(\bar{w})$  returns the sum of the current and the heuristic atomic penalty score of the partial solution represented by node  $\bar{w}$ :  $f(\bar{w}) = g^*(\bar{w}) + h^*(\bar{w})$ .

---

**Algorithm 6** branch & bound – bond order algorithm (molecule  $M$  with  $n$  bonds,  $k, p$ )

---

```

1: cutoff =  $f(k\text{-greedy}(B, k)) * p$ 
2:  $\mathcal{PQ} \leftarrow \{r\}$ 
3:  $b_1 :=$  is the first free bond in  $M$ 
4: for all bond orders  $i$  do
5:    $\mathcal{PQ} \leftarrow \mathcal{PQ} \cup \text{bondOrders}(\{b_1 \leftarrow i\})$ 
6: end for
7: while  $\mathcal{PQ} \neq \emptyset$  do
8:    $\bar{w} \leftarrow \arg \min_{w \in \mathcal{PQ}} f(w)$ 
9:    $\mathcal{PQ} \leftarrow \mathcal{PQ} \setminus \{\bar{w}\}$ 
10:  if  $\bar{w}$  is leaf then
11:    store bond orders as denoted in  $\bar{w}$ 
12:  else
13:     $b_x :=$  is next free bond in  $\mathcal{B}$ 
14:    for all bond orders  $i$  do
15:       $\bar{w} \leftarrow \text{bondOrders}(\bar{w}, \{b_x \leftarrow i\})$ 
16:      if  $f(\bar{w}) \leq \text{cutoff}$  then
17:         $\mathcal{PQ} \leftarrow \mathcal{PQ} \cup \{\bar{w}\}$ 
18:      end if
19:    end for
20:  end if
21: end while

```

---

## A.4. Selected Aspects of the Implementation of BOA Constructor

### A.4.1. Python Interface

```
import BALL

# get the first system
system = getSystems()[0]

# create a bond order assignment processor
abop = BALL.AssignBondOrderProcessor()
abop.options.setBool(
    BALL.AssignBondOrderProcessor.Option.KEKULIZE_RINGS, True)
abop.options.setBool(
    BALL.AssignBondOrderProcessor.Option.OVERWRITE_SINGLE_BOND_ORDERS,
    True)
abop.options.setBool(
    BALL.AssignBondOrderProcessor.Option.OVERWRITE_DOUBLE_BOND_ORDERS,
    True)
abop.options.setBool(
    BALL.AssignBondOrderProcessor.Option.OVERWRITE_TRIPLE_BOND_ORDERS,
    True)
abop.options.set(
    BALL.AssignBondOrderProcessor.Option.ALGORITHM,
    BALL.AssignBondOrderProcessor.Algorithm.A_STAR)
abop.options.setReal(
    BALL.AssignBondOrderProcessor.Option.BOND_LENGTH_WEIGHTING, 0)
abop.options.setInteger(
    BALL.AssignBondOrderProcessor.Option.MAX_NUMBER_OF_SOLUTIONS, 10)
abop.options.setBool(
    BALL.AssignBondOrderProcessor.Option.COMPUTE_ALSO_NON_OPTIMAL_SOLUTIONS,
    True)
abop.options.setBool(BALL.AssignBondOrderProcessor.Option.ADD_HYDROGENS, True)

# print the current atomic penalty
print abop.evaluatePenalty(system)

# apply the assignment processor
system.apply(abop)

# print all solutions
for i in range(abop.getNumberOfComputedSolutions()):
    print "solution ", str(i), ": penalty ", str(abop.getTotalPenalty(i)),
          ", ", abop.getNumberOfAddedHydrogens(i), " added hydrogens."

# apply the last solution
abop.apply(abop.getNumberOfComputedSolutions()-1)

# update the system
getMainControl().update(system)
```

### A.4.2. Graphical User Interface

BOA Constructor has been fully integrated into BALLView. Figs. A.3-A.7 show an exemplary BALLView session, where a caffeine molecule without bond order information is corrected using BOA Constructor.

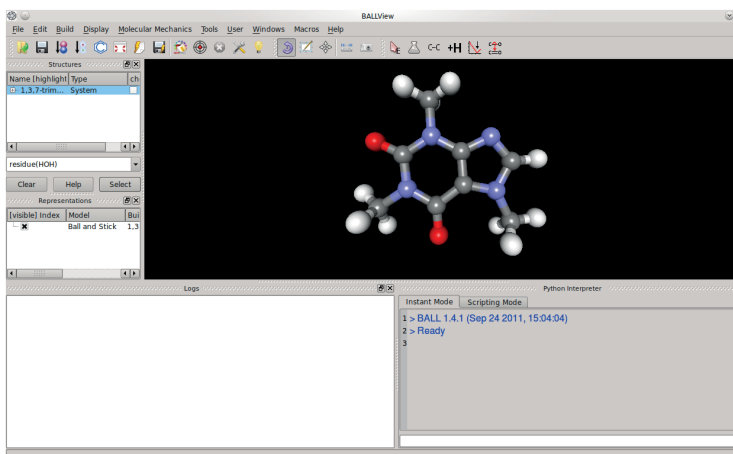


Figure A.3.: Caffeine without bond order information

## A. Supplementary Information on BOA Constructor

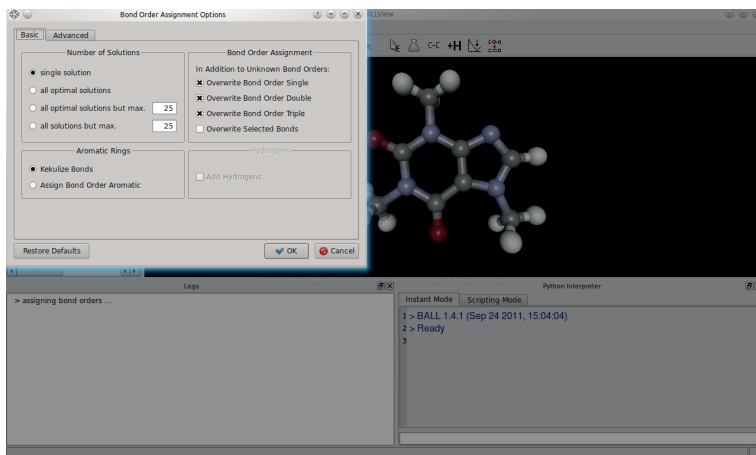


Figure A.4.: The basic options page for BOA Constructor. On the “Advanced” tab, the user can additionally include a structural score or the fine penalty, choose the solution strategy, or a different atom penalty table.

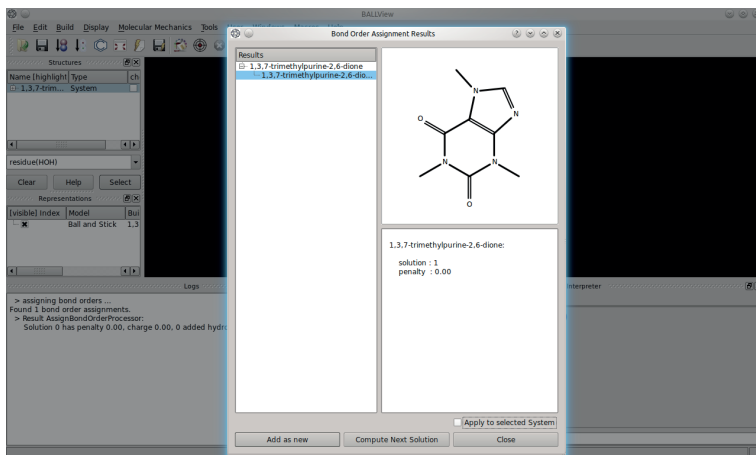


Figure A.5.: The first optimal solution of BOA Constructor for caffeine. Note that this is indeed the correct assignment, and has been scored with a penalty of 0.

#### A.4. Selected Aspects of the Implementation of BOA Constructor

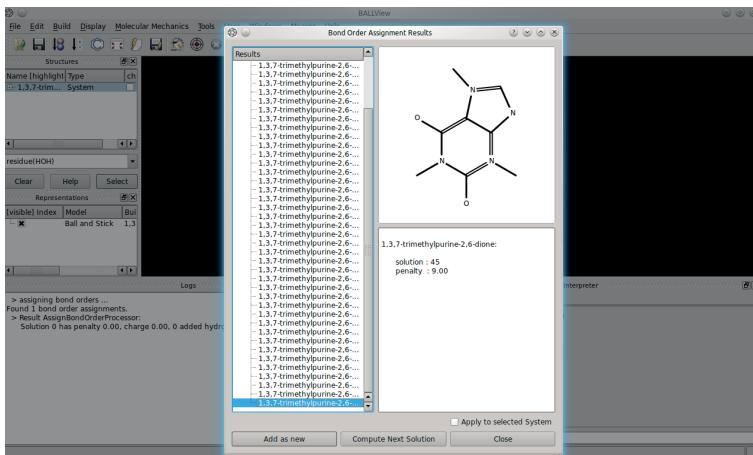


Figure A.6.: The 45th solution of BOA Constructor for caffeine. This solution is obviously non-optimal and has been scored with a penalty of 9.

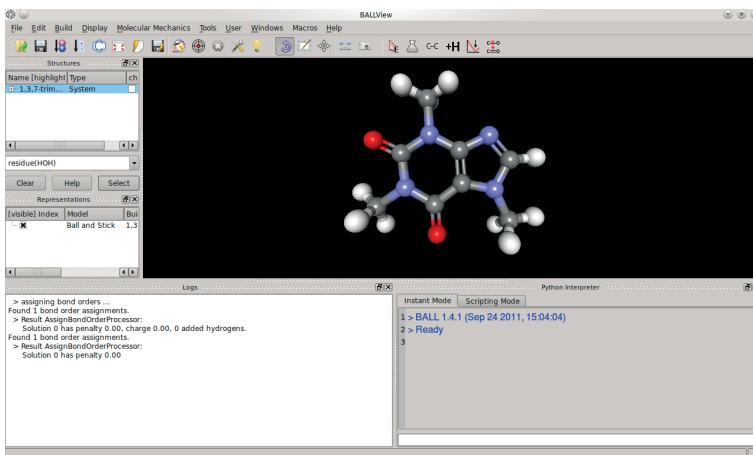


Figure A.7.: Caffeine after application of the first optimal solution of BOA Constructor.

## A.4.3. BOA Constructor Options

option	description	default
OVERWRITE_SINGLE_BOND_ORDERS	compute bond orders for all bonds with current type single bond order	true
OVERWRITE_DOUBLE_BOND_ORDERS	compute bond orders for all bonds with current type double bond order	true
OVERWRITE_TRIPLE_BOND_ORDERS	compute bond orders for all bonds with current type triple bond order	true
OVERWRITE_SELECTED_BONDS	compute bond orders for all selected bonds	false
ADD_HYDROGENS	add hydrogens based on free valences	false
USE_FINE_PENALTY	resolve penalties based on structural information	true
KEKULIZE_RINGS	try to kekulize rings	true
MAX_BOND_ORDER	the maximal possible bond order	3
MAX_NUMBER_OF_SOLUTIONS	the maximal number of solutions to compute.	10
COMPUTE_ALSO_NON_OPTIMAL_SOLUTIONS	compute also non-optimal solutions but not more than MAX_NUMBER_OF_SOLUTIONS	false
ALGORITHM	technique to compute the assignments: A_STAR, ILP, K_GREEDY, BRANCH_AND_BOUND	A_STAR
HEURISTIC	heuristic defining the tightness of the search criteria: Heuristic::{SIMPLE, MEDIUM, TIGHT}	Heuristic::TIGHT
BOND_LENGTH_WEIGHTING	weighting of bond length penalties wrt valence penalties	0.0
APPLY_FIRST_SOLUTION	apply the first solution directly on the given system	true
GREEDY_K_SIZE	the size of priority queue for the greedy algorithm	5
BRANCH_AND_BOUND_CUTOFF	the percentage cutoff for keeping PQ-Entries in the branch and bound algorithm	1.2

Table A.2.: Options for BOA Constructor.



#### A.4.4. Distribution of Penalty Rules

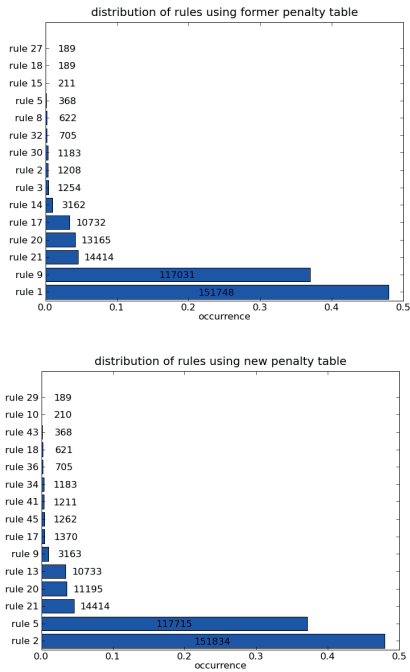


Figure A.8.: Distribution of matching rules using the former penalty table of [WWKC06] (upper figure) and the new penalty table as defined in table A.1 (lower figure).

## A.5. A\* Performance Measures

The performance of the A\* search heuristics manifests in two different ways: computational efficiency and order of the results. As all proposed heuristics are admissible, each will be able to compute all optimal solutions. This is confirmed empirically by comparing the number of optimally scored reference assignments of each heuristic (column four of table A.4 and table A.5) with the number of reproduced reference assignments of the other exact solvers (column four of table 2.3). Thus, we focus on the following properties for further evaluation

1. the number of reference assignments returned as first solution (should be large).
2. the number of steps necessary to find a first/all optimal solutions (should be small).
3. the costs per step (run time for the evaluation of all child nodes and correct insertion into the A\* priority queue) (should be small).

The first property naturally depends on the ability of the penalty table to assign an optimal penalty score to the reference molecule's bond order assignment. On the other hand, the heuristic algorithmically implies a certain order on all optimal solutions. In a general setting, one might expect the algorithm to return the most probable assignment first. Table A.4 shows that on the MMFF94 validation suite, for both penalty tables, search heuristic simple performs best. This may be due to its preference for smaller bond orders. However, this only measures the quality of the results of the A\* search. Similarly important may be the runtime properties. To understand the runtime behaviour, we have plotted a histogram of the number of steps needed by the different heuristics (see table A.9 and table A.10): heuristic "simple" needs noticeably more A\* search steps than heuristics "medium" and "tight", which perform very similar. Notably, the choice of penalty table does not seem to have a significant influence on the number of steps. The use of the fine penalty score, on the other hand, improves the performance of all three heuristics.

Taking only the first solution returned by each heuristic into account, all three heuristics perform comparably well. On the MMFF94 validation suite, the simple heuristic is able to reproduce 59.78%, whereas both, heuristic "medium" and "tight", achieve 56.89% of the reference solutions. On the KEGG Drug set, a reversed order can be seen: heuristic "medium" and "tight" slightly outperform heuristic "simple" with 50.58% versus 49.95% (see table A.5). These differences are due to a different ordering of partial solutions within the A\* priority queue.

Comparing the number of steps required for the A\* to compute a solution, heuristics "medium" and "tight" perform surprisingly similar. The additional tightening of the upper boundary does not influence the number of steps to compute a complete first bond order assignment as the A\* algorithms tries the bond orders with increasing order. Heuristic "simple", on the other hand, needs significantly more A\* search steps than the other two, at least for molecules with more than 30 bonds (c.f. table A.3).

Obviously the number of steps in the A\* search heuristics correlates with the dimension of the problem, which can be measured as the number of bonds. For this evaluation, we did not count bonds connecting hydrogens, since these are automatically fixed as single bonds. This relation is shown in table A.11 and table A.12.

file	# steps			# bonds	# optimal sol
	simple	medium	tight		
DEBMOM01	13	11	11	7 + 3 = 10	1
COTMON	48	34	34	13 + 7 = 20	2
DEDSIO	50	30	30	17 + 13 = 30	1
DEGRIQ	330	117	117	23 + 17 = 40	4
DUYPEs	165	119	119	26 + 22 = 48	8
BEWCUB	165	100	100	38 + 23 = 61	2
TAJSUS	301	83	83	20 + 11 = 31	8
DIYPOQ	825	87	87	18 + 17 = 35	2
TACLEO	1076	54	54	16 + 12 = 28	2

Table A.3.: Performance of the A\*-search heuristics on selected molecules of the MMFF94 validation suite, measured as the number of steps to find a first solution. Column five denotes the number of bonds (non-hydrogen bonds + hydrogens).

table	heuristic	reference is		no solution
		1st solution	optimal	
Wang	simple	455 (59.78%)	599 (78.71%)	4 (0.53%)
	medium	433 (56.89%)		
	tight			
improved	simple	458 (60.18%)	639 (83.96%)	0 (0.00%)
	medium	449 (59.00%)		
	tight			

Table A.4.: Performance of the A\*-search heuristics on the MMFF94 validation suite using the penalty table as defined in Wang et al. [WWKC06] and the improved BALL penalty table (see table A.1), measured as the number of reproduced reference bond order assignments.

table	heuristic	reference is		no solution
		1st solution	optimal	
Wang	simple	3708 (49.95%)	6326 (85.21%)	191 (2.57%)
	medium	3755 (50.58%)		
	tight	3754 (50.57%)		
improved	simple	4391 (59.15%)	7167 (96.54%)	180 (2.42%)
	medium	4022 (54.18%)		
	tight			

Table A.5.: Performance of the A\*-search heuristics on the KEGG Ligand Drug set using the penalty table as defined in Wang et al. [WWKC06] and the improved BALL penalty table (see table A.1) measured as the number of reproduced reference bond order assignments.

A. Supplementary Information on BOA Constructor

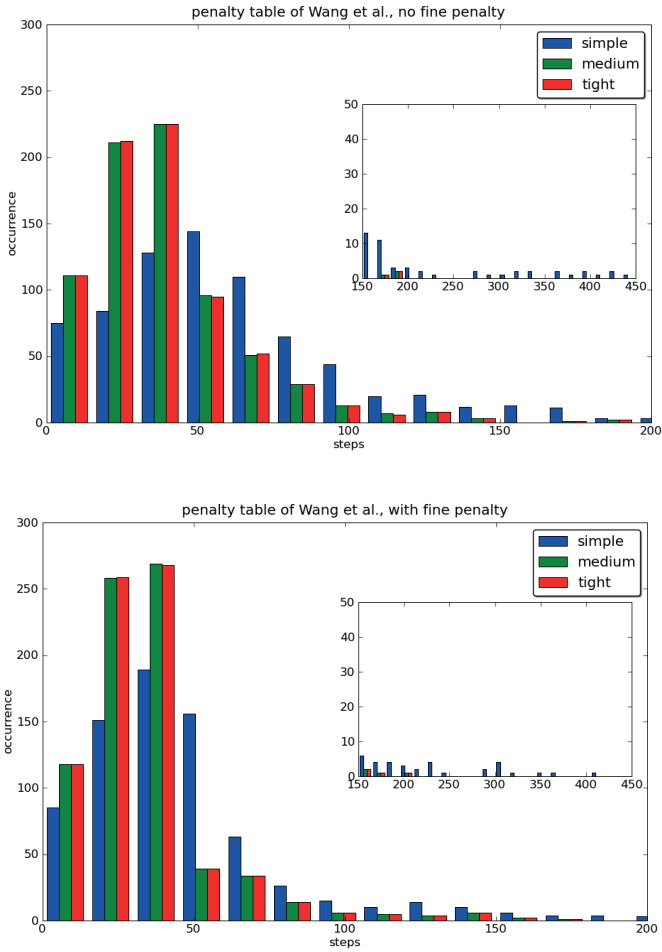


Figure A.9.: Comparison of the performance of the A\* search heuristics as a function of the number of steps (MMFF94 validation suite, using the penalty table as defined in Wang et al. [WWKC06]).

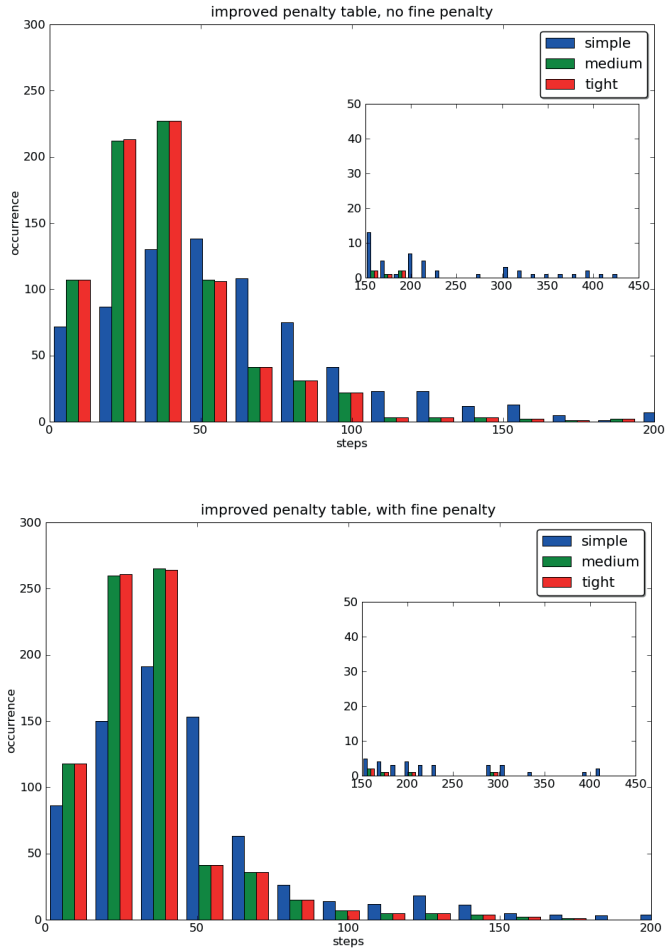


Figure A.10.: Comparison of the performance of the A\* search heuristics as a function of the number of steps (MMFF94 validation suite, using the improved BALL penalty table (see table A.1)).

A. Supplementary Information on BOA Constructor

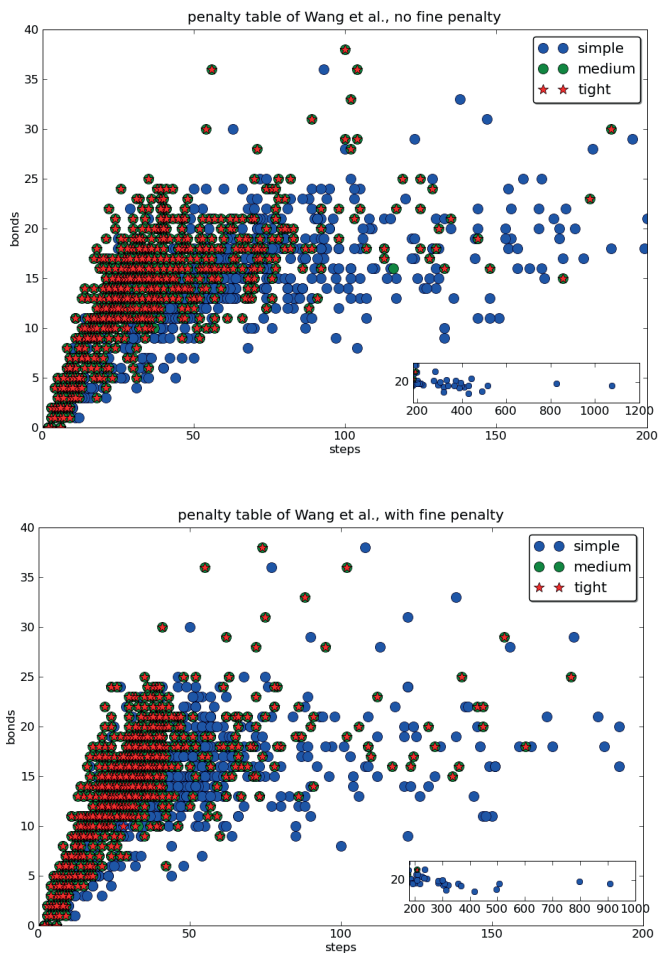


Figure A.11.: Performance of the different A\* search heuristics as a function of problem dimension (number of non-hydrogen containing bonds) on the MMFF94 validation suite, using the penalty table as defined in Wang et al. [WWKC06].

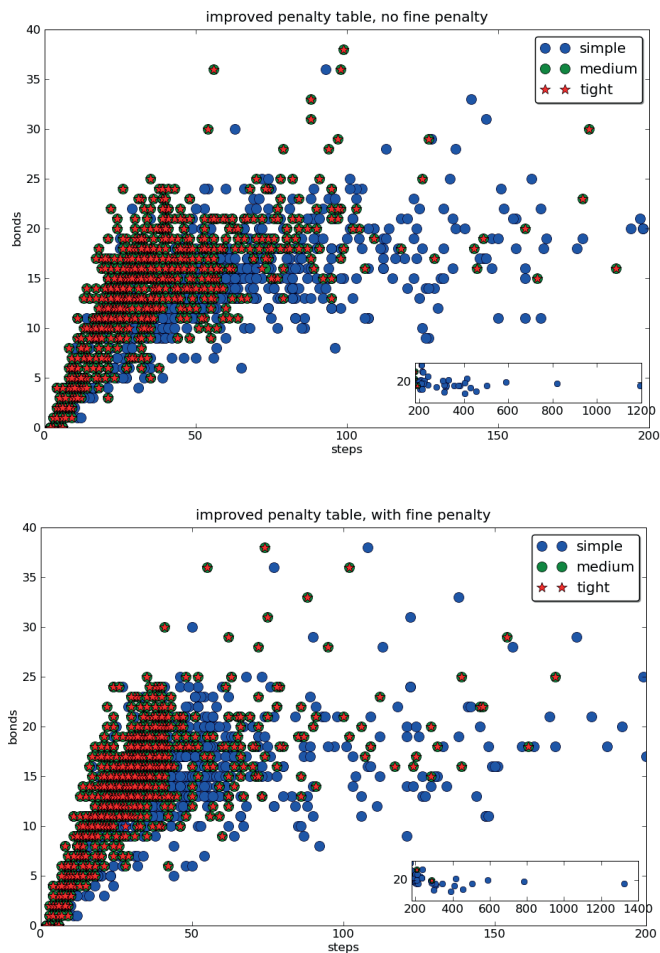


Figure A.12.: Performance of the different A\* search heuristics as a function of problem dimension (number of non-hydrogen containing bonds) on the MMFF94 validation suite, using the improved BALL penalty table (see table A.1).





## B. Supplementary Information on NightShift, Spinster, and Liops

This appendix contains supplementary or advanced information on our pipeline for chemical shift prediction NightShift and the protein- and protein-ligand models Spinster and Liops. This includes parameter values for the semi-classical terms in the models (c.f. App. B.1) and on the lexing and parsing of files in NMRStar format (c.f. App. B.2).

### B.1. Parameters for Semi-Classical NMR Chemical Shift Predictors

residue type $k$	intensity factor $I_k$	ring atoms
PHE	1.05	$C^\gamma C_2^\delta C_2^\epsilon C^\zeta C_1^\epsilon C_1^\delta$
TYR	0.92	$C^\gamma C_2^\delta C_2^\epsilon C^\zeta C_1^\epsilon C_1^\delta$
TRP1	1.04	$C_2^\delta C_3^\epsilon C_3^\zeta C_2^\eta C_2^\epsilon C_2^\delta$
TRP2	0.90	$C^\gamma C_2^\delta C_2^\epsilon N_1^\epsilon C_1^\delta$
HIS	0.43	$C^\gamma N_1^\delta C_1^\epsilon N_2^\epsilon C_2^\delta$

Table B.1.: List of ring current effectors considered in the ShiftX implementation of the Haigh-Mallion model, empirically determined values for their intensity factors  $I_k$ , and the involved ring atoms.

atom type	target nucleus factor
$H^\alpha$	5.13
$H^N$	7.06
$C^\alpha$	1.5
$C^\beta$	1.0
C	1.0
N	1.0

Table B.2.: List of ring current targets considered in the ShiftX-implementation of the Haigh-Mallion model and empirically determined values for the constant  $B$ .

### B.2. Lexer and Parser for CIF - Files

#### B.2.1. A FLEX-based Lexer for CIF - Files

Here, we reproduce the full content of the ".l" - file used to build a lexer for the Crystallographic Information File (CIF) [HAB91, BM02]. Please note that, due to the peculiarities of the CIF format, the lexer has to perform some of the work classically assigned to the parser. This requires extensive use of FLEX's state mechanism.

B. Supplementary Information on NightShift, Spinster, and Liops

location	type label	members
backbone	$H^N$	hydrogen bonded to the peptid bond's nitrogen
	$H^\alpha$	$\alpha$ -hydrogen bonded to the $\alpha$ -carbon
	$C'$	peptide backbone carbon also called carbonyl carbon
	$C^\alpha$	backbone carbon next to the carbonyl carbon
	$C^\beta$	carbon at position $\beta$ relative to the carbonyl carbon
	N	backbone nitrogen
side chain hydrogens	HA*	HA, 1HA, 2HA ( $H_*^\alpha$ )
	HB*	HB, 1HB, 2HB, 3HB ( $H_*^\beta$ )
	HD*	1HD, 2HD, HD1, HD2, 1HD1, 1HD2, 2HD2, 2HD1, 3HD1, 3HD2 ( $H_*^\delta$ )
	HE*	HE, HE1, HE2, HE3, 1HE, 2HE, 3HE, 1HE2, 2HE2 ( $H_*^\epsilon$ )
	HG*	HG, 1HG, 2HG, HG1, 1HG1, 2HG1, 3HG1, 1HG2, 2HG1, 3HG1 ( $H_*^\gamma$ )
	HH*	HH, HH2, 1HH1, 1HH2, 2HH1, 2HH2 ( $H_*^\eta$ )
	HZ*	HZ, 1HZ, 2HZ, 3HZ ( $H_*^\zeta$ )
	HEHZ*	HE*, HH*, HZ*
side chain nitrogens	ND*	ND1, ND2 ( $N_*^\delta$ )
	NE*	NE, NE1, NE2 ( $N_*^\epsilon$ )
	NH*	NH1, NH2 ( $N_*^\eta$ )
	NZ*	NZ ( $N_*^\zeta$ )
side chain carbons	CD*	CD, CD1, CD2 ( $C_*^\delta$ )
	CE*	CE, CE1, CE2, CE3 ( $C_*^\epsilon$ )
	CG*	CG, CG1, CG2 ( $C_*^\gamma$ )
	CH*	CH2 ( $C_*^\eta$ )
	CZ*	CZ, CZ2, CZ3 ( $C_*^\zeta$ )
side chain oxygens	OD*	OD1, OD2 ( $O_*^\delta$ )
	OE*	OE1, OE2 ( $O_*^\epsilon$ )
	OG*	OG, OG1 ( $O_*^\gamma$ )
side chain sulfur	SD	$S_*^\delta$
	SG	$S_*^\gamma$

Table B.3.: Atom type nomenclature, borrowed from the PDB file format conventions. The asterisk represents further subtypes like "HE\*" for HE1, HE2, and HE3.

$c_i$	value
$c_1$	9.7464
$c_2$	-0.9887
$c_3$	0.147521
$c_4$	$-1.65458 \times 10^{-05}$
$c_5$	-0.000134668
$c_6$	0.0598561
$c_7$	15.6855
$c_8$	-0.673905

Table B.4.: Parameters for the hydrogen bond effect as used in the ShiftX implementation.

```

%s in_data_heading
%s in_save_heading
%x in_single_quote
%x in_double_quote
%x in_loop
%x in_save_frame
%x in_textfield

%option noyywrap
%option array

EOL          [\n\r]
AnyPrintChar ({OrdinaryChar}|["\'#$_%\t[:blank:];\[\]])

SingleQuote  \'
DoubleQuote  \"
COMMENT      (#({AnyPrintChar})*{EOL})+
TOKENIZED_COMMENT [[[:blank:]\n\r]*{COMMENT}
WHITESPACE   [[[:blank:]\n\r]+
WS_OR_COMMENT ([[:blank:]\n\r]|{TOKENIZED_COMMENT})+
DATA_        [dD][aA][tT][aA]_
SAVE_        [sS][aA][vV][eE]_
NBC          [^[:blank:]\n\r]
LOOP_        [lL][oO][oO][pP]_
STOP_        [sS][tT][oO][pP]_
TAG          [[[:blank:]]*_{NBC}+

OrdinaryChar [0-9a-zA-Z!%&$()*+, \-./:<=>?@\\`'{}|\[\]~]
UnquotedString {OrdinaryChar}({OrdinaryChar}|\\;)*
UnderScore    _

TEXTLEADCHAR {OrdinaryChar}|{SingleQuote}|{DoubleQuote}
              |#|$_|_ |[:blank:]]|\\[|\\]
CHARSTRING   {UnquotedString}

DIGIT        [0-9]
UnsignedInteger {DIGIT}+
EXPONENT     [eE]([+\\ -]?) {UnsignedInteger}
INT          [+\\ -]?[0-9]+
FLOAT        [+\\ -]?(( [0-9] + " . " [0-9] + ) | {EXPONENT})
NUMBER       ({INT}|{FLOAT})
NUMERIC      ({NUMBER}|{NUMBER}({UnsignedInteger}))
VALUE        [ . ? ] | ( {NUMERIC} | {CHARSTRING} )

%%

<in_textfield>{
  ~;/{WS_OR_COMMENT}* {
    BEGIN(textfield_state_buffer);
    return TK_TEXTFIELD;

```

*B. Supplementary Information on NightShift, Spinster, and Liops*

```
}

{WS_OR_COMMENT}* {
    strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
    return TK_TEXTFIELD_LINE;
}

{TEXTLEADCHAR}{AnyPrintChar}*{EOL} {
    strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
    return TK_TEXTFIELD_LINE;
}
}

<*>~/{WS_OR_COMMENT}* {
    textfield_state_buffer = YY_START;
    BEGIN(in_textfield);
    return TK_TEXTFIELD;
}

<in_single_quote>{
    {SingleQuote}/{WS_OR_COMMENT}+{
        BEGIN(state_buffer);
        return TK_CLOSE_SINGLE_QUOTE;
    }
}

# {
    strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
    return TK_VALUE;
}

; {
    strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
    return TK_VALUE;
}

{VALUE} {
    strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
    return TK_VALUE;
}

{WHITESPACE} {
    strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
    return TK_WHITESPACE;
}
}

<in_data_heading,in_save_heading,in_loop,in_save_frame,INITIAL>
{WHITESPACE}+{SingleQuote} {
    state_buffer = YY_START;
    BEGIN(in_single_quote);
}
```

```

    return TK_OPEN_SINGLE_QUOTE;
}

<*>{SingleQuote} {
    return TK_SINGLE_QUOTE;
}

<in_double_quote>{
    {DoubleQuote}/{WS_OR_COMMENT}+ {
        BEGIN(state_buffer);
        return TK_CLOSE_DOUBLE_QUOTE;
    }

    # {
        strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
        return TK_VALUE;
    }

    ; {
        strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
        return TK_VALUE;
    }

    {VALUE} {
        strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
        return TK_VALUE;
    }

    {WHITESPACE} {
        strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
        return TK_WHITESPACE;
    }
}

<in_data_heading,in_save_heading,in_loop,in_save_frame,INITIAL>
{WHITESPACE}+{DoubleQuote} {
    state_buffer = YY_START;
    BEGIN(in_double_quote);
    return TK_OPEN_DOUBLE_QUOTE;
}

<*>{DoubleQuote} {
    return TK_DOUBLE_QUOTE;
}

<in_save_heading>{
    {NBC}+ {
        BEGIN(INITIAL);
        strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
        return TK_SAVE_HEADING;
    }
}

```

*B. Supplementary Information on NightShift, Spinster, and Liops*

```
}

{WS_OR_COMMENT} {
  BEGIN(INITIAL);
  strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
  return TK_WHITESPACE;
}
}

{SAVE_}{
  BEGIN(in_save_heading);
  return TK_SAVE;
}

<in_data_heading>{
  {NBC}+ {
    BEGIN(INITIAL);
    strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
    return TK_DATA_HEADING;
  }
}

~{DATA_} | {WS_OR_COMMENT}{DATA_} {
  BEGIN(in_data_heading);
}

<in_loop>{
  {STOP_} {
    BEGIN(INITIAL);
    return TK_STOP;
  }

  {VALUE} {
    strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
    return TK_VALUE;
  }

  {WS_OR_COMMENT} {
    strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
    return TK_WHITESPACE;
  }
}

{LOOP_} {
  BEGIN(in_loop);
  return TK_LOOP;
}
```

```

{VALUE} {
    strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
    return TK_VALUE;
}

<*>{UnderScore} {
    return TK_UNDERSCORE;
}

{WS_OR_COMMENT} {
    strncpy(CIFParserlval.text, yytext, CIFPARSER_LINE_LENGTH);
    return TK_WHITESPACE;
}
%%

```

### B.2.2. A Grammar for CIF - Files

```

<CIF> ::= <optional_whitespace> <optional_datablock>

<optional_datablock> ::= <empty>
    | <datablock_list> <optional_whitespace>

<datablock_list> ::= <datablock>
    | <datablock_list> <optional_whitespace> <datablock>

<datablock> ::= <data_heading> TK_WHITESPACE
    | <data_heading> TK_WHITESPACE <datablock_content> TK_WHITESPACE

<data_heading> ::= TK_DATA_HEADING

<datablock_content> ::= <data_items>
    | <saveframe>
    | <datablock_content> TK_WHITESPACE <data_items>
    | <datablock_content> TK_WHITESPACE <saveframe>

<data_items> ::= <tag> <optional_whitespace> <value>
    | <tag> TK_WHITESPACE
    | <loop> <tag_list> <optional_whitespace> <loop_body> <optional_whitespace>
    TK_STOP

<loop> ::= TK_LOOP

<tag_list> ::= TK_WHITESPACE <tag>
    | <tag_list> <optional_whitespace> <tag>

<loop_body> ::= <value>
    | <loop_body> <optional_whitespace> <value>

<saveframe> ::= TK_SAVE <save_heading> TK_WHITESPACE
    <save_frame_content> <optional_whitespace> TK_SAVE

```

## B. Supplementary Information on NightShift, Spinster, and Liops

$\langle \text{save\_heading} \rangle ::= \text{TK\_SAVE\_HEADING}$

$\langle \text{save\_frame\_content} \rangle ::= \langle \text{data\_items} \rangle$   
|  $\langle \text{save\_frame\_content} \rangle \text{TK\_WHITESPACE} \langle \text{data\_items} \rangle$   
|  $\langle \text{save\_frame\_content} \rangle \langle \text{data\_items} \rangle$

$\langle \text{optional\_whitespace} \rangle ::= \langle \text{empty} \rangle$   
|  $\text{TK\_WHITESPACE} \langle \text{optional\_whitespace} \rangle$

$\langle \text{tag} \rangle ::= \text{TK\_UNDERSCORE} \langle \text{value} \rangle$

$\langle \text{value} \rangle ::= \text{TK\_VALUE}$   
|  $\text{TK\_VALUE} \langle \text{single\_quote\_helper} \rangle \langle \text{value\_helper} \rangle$   
|  $\text{TK\_VALUE} \langle \text{single\_quote\_helper} \rangle$   
|  $\text{TK\_VALUE} \text{TK\_UNDERSCORE} \langle \text{value\_helper} \rangle$   
|  $\text{TK\_OPEN\_SINGLE\_QUOTE} \langle \text{single\_quoted\_string} \rangle$   
|  $\text{TK\_OPEN\_DOUBLE\_QUOTE} \langle \text{double\_quoted\_string} \rangle$   
|  $\text{TK\_TEXTFIELD} \langle \text{textfield\_line} \rangle \text{TK\_TEXTFIELD}$

$\langle \text{value\_helper} \rangle ::= \text{TK\_VALUE}$   
|  $\text{TK\_VALUE} \langle \text{single\_quote\_helper} \rangle \langle \text{value\_helper} \rangle$   
|  $\text{TK\_VALUE} \langle \text{single\_quote\_helper} \rangle$   
|  $\text{TK\_VALUE} \text{TK\_UNDERSCORE} \langle \text{value\_helper} \rangle$

$\langle \text{single\_quote\_helper} \rangle ::= \text{TK\_SINGLE\_QUOTE}$

$\langle \text{single\_quoted\_string} \rangle ::= \text{TK\_CLOSE\_SINGLE\_QUOTE}$   
|  $\text{TK\_VALUE} \langle \text{single\_quoted\_string} \rangle$   
|  $\text{TK\_WHITESPACE} \langle \text{single\_quoted\_string} \rangle$   
|  $\text{TK\_VALUE} \text{TK\_UNDERSCORE} \text{single\_quoted\_string}_i$   
|  $\text{TK\_SINGLE\_QUOTE} \langle \text{single\_quoted\_string} \rangle$   
|  $\text{TK\_DOUBLE\_QUOTE} \langle \text{single\_quoted\_string} \rangle$

$\langle \text{double\_quoted\_string} \rangle ::= \text{TK\_CLOSE\_DOUBLE\_QUOTE}$   
|  $\text{TK\_VALUE} \langle \text{double\_quoted\_string} \rangle$   
|  $\text{TK\_VALUE} \text{TK\_UNDERSCORE} \langle \text{double\_quoted\_string} \rangle$   
|  $\text{TK\_WHITESPACE} \langle \text{double\_quoted\_string} \rangle$   
|  $\text{TK\_DOUBLE\_QUOTE} \langle \text{double\_quoted\_string} \rangle$   
|  $\text{TK\_SINGLE\_QUOTE} \langle \text{double\_quoted\_string} \rangle$

$\langle \text{textfield\_line} \rangle ::= \langle \text{empty} \rangle$   
|  $\text{TK\_TEXTFIELD\_LINE} \langle \text{textfield\_line} \rangle$



## C. Copyrights of Figures and Quotations

All figures in this thesis are protected by copyright and may not be published without permission of the copyright owner. License numbers for previously published content were acquired through the Copyright Clearance Center. In the following, we describe the original sources of reproduced content and the licensing information where appropriate.

- Fig. 2.2 is used with kind permission from PERGAMON: Bioorganic & medicinal chemistry letters, Structure-based optimization of a potent class of arylamide FMS inhibitors, 18, 3632, 2008, Sanath K. Meegalla, license number 2901430976472.
- Fig. 2.1 has been previously published in Proceedings of GCB 2009 [DRLH09].
- Fig. 2.5, as well as some of the explanatory text, are used with kind permission from Oxford University Press: Bioinformatics, Automated Bond Order Assignment as an Optimization Problem, 27, 619, 2011, Anna Katharina Dehof, license number 2901431393493.
- Fig. 3.3 and Fig. 3.1 contain images that are used with kind permission from Springer Science+Business Media: Journal of Biomolecular NMR, A microscale protein NMR sample screening pipeline, 46, 2009, Paolo Rossi, license number 2780880807590.
- Figs. 3.9, 3.11, 3.12, and 3.13 were previously published in our own manuscript in the Proceedings of GCB 2011 [DLH11].
- Fig. 4.10 was published in our own manuscript [PGD<sup>+</sup>10].
- Fig. 4.3, 4.4, 4.5,4.7,4.8 were published in our own manuscript [MGD<sup>+</sup>10]. With Fig. 4.4, we additionally won the Arts and Science Award ISMB/ECCB 2011, for which the figure was published as the winning entry on several websites.



## D. Publications and Talks by the Author

### D.1. Journal Publications (peer-reviewed)

A.K. Dehof, A. Rurainski, Q.B.A. Bui, S. Böcker, H.-P. Lenhof, and A. Hildebrandt: (2011) **Automated Bond Order Assignment as an Optimization Problem**. *Bioinformatics*, 2011, 27, 619-625.

K.K. Singh, S. Erkelenz, S. Rattay, A.K. Dehof, A. Hildebrandt, K. Schulze-Osthoff, H. Schaal, and C. Schwerk: (2010) **Human SAP18 mediates assembly of a splicing regulatory multiprotein complex via its ubiquitin-like fold**. *RNA*, 2010, 16(12), 2442-2454.

A. Hildebrandt, A.K. Dehof, A. Rurainski, A. Bertsch, M. Schumann, N.C. Toussaint, A. Moll, D. Stoeckel, S. Nickels, S.C. Müller, H.-P. Lenhof, and O. Kohlbacher: (2010) **BALL - Biochemical Algorithms Library 1.3**. *BMC Bioinformatics*, 2010, 11:531.

K.H. Brämshwieg, A.B. Riemer, E. Förster-Waldl, A.K. Dehof, D. Neumann, H.N. Lode, C. Zielinski, O. Scheiner, H. Pehamberger, and E. Jensen-Jarolim: (2005) **Generation of Peptide mimics of Melanoma Antigen GD2** *Journal of Investigative Dermatology* 125(S1:A35)

E. Förster-Waldl, A.B. Riemer, A.K. Dehof, D. Neumann, K.H. Brämshwieg, G. Boltz-Nitulescu, H. Pehamberger, C. Zielinski, O. Scheiner, A. Pilloak, H.N. Lode, and E. Jensen-Jarolim: (2004) **Isolation and structural analysis of peptide mimotopes for the disialoganglioside GD2, a neuroblastoma tumor antigen** *Molecular Immunology* 42

### D.2. Conference Proceedings (peer-reviewed)

A.K. Dehof, H.-P. Lenhof, and A. Hildebrandt: (2011) **Predicting Protein NMR Chemical Shifts in the Presence of Ligands and Ions using Force Field-based Features**. *Proceedings of the German Conference on Bioinformatics (GCB 2011)*.

L. Marsalek, A.K. Dehof, I. Georgiev, H.-P. Lenhof, P. Slusallek and A. Hildebrandt: (2010) **Real-time Ray Tracing of Complex Molecular Scenes**. *Proceedings of the IVbm 10 - Information Visualization in Biomedical Informatics*

M. Phillips, I. Georgiev, A.K. Dehof, L. Marsalek, H.-P. Lenhof, A. Hildebrandt, and P. Slusallek: (2010) **Measuring Properties of Molecular Surfaces Using Ray Casting**. *HiCOMB 2010, Proceedings of 9th International Workshop on High Performance Computational Biology*

A.K. Dehof, A. Rurainski, H.-P. Lenhof and A. Hildebrandt: (2009) **Automated Bond Order Assignment as an Optimization Problem**. *Proceedings of GCB 2009 (German Conference on Bioinformatics, Halle (Saale), Germany, 28-30 Sep 2009)*, *Lecture Notes in Informatics*, 2009

A.K. Dehof, D. Stoeckel, S. Nickels, S.C. Müller, M. Schumann, H.-P. Lenhof, O. Kohlbacher, A. Hildebrandt: **The BALL project: The Biochemical Algorithms Library (BALL) for Rapid Application Development in Structural Bioinformatics and its graphical user interface BALLView**, *BOSC 2011 Vienna (SIG at ISMB ECCB 2011, Vienna, Austria, July 15-July 16, 2011)*

### D.3. Conference Proceedings (not peer-reviewed)

A.K. Dehof, I. Georgiev, L. Marsalek, D. Stoeckel, S. Nickels, H.-P. Lenhof, P. Slusallek, and A. Hildebrandt: (2009) **Visual Computing in Computer Aided Drug Design**. *1st. Visual Computing Research Conference, December 2009, Saarbrücken*

### D.4. Posters

A.K. Dehof, H.-P. Lenhof, A. Hildebrandt: **Pipeline for the training of NMR chemical shift prediction models**, *ISMB ECCB 2011 Vienna (19th annual international conference on Intelligent Systems for Molecular Biology and 10th European conference on Computational Biology, Vienna, Austria, July 17-July 19, 2011)*

A.K. Dehof, D. Stoeckel, S. Nickels, S.C. Müller, M. Schumann, H.-P. Lenhof, O. Kohlbacher, A. Hildebrandt: **The Biochemical Algorithms Library (BALL) - Rapid Application Development in Structural Bioinformatics**, *BOSC 2011 Vienna (SIG at ISMB ECCB 2011, Vienna, Austria, July 15-July 16, 2011)*

A.K. Dehof, D. Stoeckel, S. Nickels, S.C. Müller, M. Schumann, H.-P. Lenhof, O. Kohlbacher, A. Hildebrandt: **The Biochemical Algorithms Library (BALL) - Rapid Application Development in Structural Bioinformatics**, *MMW 2011 (25th Molecular Modelling Workshop 2011, Erlangen, Germany, 4-6 Apr 2011)*

A.K. Dehof, L. Marsalek, I. Georgiev, D. Stoeckel, S. Nickels, H.-P. Lenhof, P. Slusallek and A. Hildebrandt: **Interactive real time ray tracing in molecular visualization**, *GCB 2009 (German Conference on Bioinformatics, Halle (Saale), Germany, 28-30 Sep 2009)*

A.K. Dehof, L. Marsalek, I. Georgiev, D. Stoeckel, S. Nickels, H.-P. Lenhof, P. Slusallek and A. Hildebrandt: **Real-time ray tracing of complex molecular scenes with BALLView and RTfact**. *ISMB ECCB 2009 Stockholm (17th annual international conference on Intelligent Systems for Molecular Biology and 8th European conference on Computational Biology, Stockholm, Sweden, June 29-July 2, 2009)*

L. Marsalek, A.K. Dehof, I. Georgiev, D. Stoeckel, S. Nickels, H.-P. Lenhof, P. Slusallek and A. Hildebrandt: **Real-time Volume Ray Tracing For Bioinformatics Applications**. *ISMB ECCB 2009 Stockholm (17th annual international conference on Intelligent Systems for Molecular Biology and 8th European conference on Computational Biology, Stockholm, Sweden, June 29-July 2, 2009)*

A.K. Dehof, H.-P. Lenhof, H. Hoffmann, R. Jochem, A. Hildebrandt: **BALLView-VR: Wirkstoffentwicklung als Virtual Reality Anwendung mit Multi-Touch Display**. *Closing conference of the Foresight Process, 19.-20. June, 2009*

A.K. Dehof, A. Rurainski, S.C. Müller, H.-P. Lenhof: **Discrete optimization techniques for optimal bond order assignment** *ECCB 2008 (7th European conference on Computational Biology, Cagliari, Italy, 22- 26 September, 2008)*

## D.5. Talks at International Conferences

*"Predicting Protein NMR Chemical Shifts in the Presence of Ligands and Ions using Force Field-based Features"*. At: **German Conference on Bioinformatics**, 8. September 2011, Weihenstephan, Germany.

*"A Pipeline for the training of NMR chemical shift prediction models"*. At: **25th Molecular Modelling Workshop 2011**, 4. April 2011, Erlangen, Germany

*"Automated Bond Order Assignment as an Optimization Problem"*. At: **German Conference on Bioinformatics**, 30. September 2009, Halle, Germany

## D.6. Talks at Research Institutions

*"Discrete Optimization Techniques for Optimal Bond Order Assignment"*. At: **JCB Seminar, Chair of Prof. Sebastian Böcker**, 4. December 2008, Jena, Germany

## D.7. Technical Demonstrations at International Events

Demonstration **Real time ray tracing in molecular modelling** at the 21st Eurographics Symposium on Rendering, June 2010, Saarbrücken, Germany

Demonstration **Distributed collaborative molecular modelling** at Research@Intel, March 2010, Brussels, Belgium

Demonstration **BALLView: a molecular viewer and modelling tool with real-time ray tracing capabilities** at the technology track of ECCB/ISMB 2009, June 2009, Stockholm, Sweden

Demonstration of **BALLView with multitouch support** at the closing conference of the Foresight Process, June 2009, Bonn, Germany

Technical support of demonstration **Real time ray tracing in computational biology with BALLView** at Intel Press Conference, Cebit 2009, March 2009, Hannover, Germany

Demonstration of **BALLView** at Saarland University booth at CeBit 2009, March 2009, Hannover, Germany

## D.8. Publications in Preparation

A.K. Dehof, L. Marsalek, I. Georgiev, D. Stoeckel, S. Nickels, H.-P. Lenhof, P. Slusallek and A. Hildebrandt: **Interactive real time ray tracing in molecular visualization**

A.K. Dehof, S. Loew, H.-P. Lenhof, and A. Hildebrandt: **NightShift – An automated pipeline for training and evaluating NMR shift prediction models**

A.K. Dehof, S. Loew, H.-P. Lenhof, and A. Hildebrandt: **Spinster and Liops – New hybrid models for NMR chemical shift predictions for proteins with and without ligands**

A.K. Dehof, Q.B.A. Bui, K. Dührkop, H.-P. Lenhof, S. Böcker, and A. Hildebrandt: **BOA Constructor – A new program for accurate, efficient, and extensible bond order assignment**



## Bibliography

- [AAFA10] Z. Atieh, M. Aubert-Frecon, and A.-R. Allouche. Rapid, Accurate and Simple Model to Predict NMR Chemical Shifts for Biological Molecules. *The Journal of Physical Chemistry B*, 114(49):16388–16392, 2010.
- [AB03] C. P. Adams and V. V. Brantner. *New drug development: Estimating entry from human clinical trials*. Bureau of Economics, Federal Trade Commission, 2003.
- [ABGK05] R. J. Abraham, J. J. Byrne, L. Griffiths, and R. Koniotou.  $^1\text{H}$  chemical shifts in NMR: Part 22 - prediction of the  $^1\text{H}$  chemical shifts of alcohols, diols and inositols in solution, a conformational and solvation investigation. *Magnetic Resonance in Chemistry*, 43(8):611–624, 2005.
- [ABGP06] R. J. Abraham, J. J. Byrne, L. Griffiths, and M. Perez.  $^1\text{H}$  chemical shifts in NMR: Part 23 - the effect of dimethyl sulphoxide versus chloroform solvent on  $^1\text{H}$  chemical shifts. *Magnetic Resonance in Chemistry*, 44(5):491–509, 2006.
- [ACP87] S. Arnborg, D. G. Corneil, and A. Proskurowski. Complexity of finding embedding in a  $k$ -tree. *Journal on Algebraic and Discrete Methods*, 8:277–284, 1987.
- [AFWH00] C. Altona, D. H. Faber, and A. J. A. Westra Hoekzema. Double-helical DNA  $^1\text{H}$  chemical shifts: An accurate and balanced predictive empirical scheme. *Magnetic Resonance in Chemistry*, 38(2):95–107, 2000.
- [Ait11] M. Aitken. *The Global Use of Medicines: Outlook Through 2015*. Technical Report May, 2011.
- [AKLM02] E. Althaus, O. Kohlbacher, H.-P. Lenhof, and P. Müller. A combinatorial approach to protein docking with flexible side chains. *Journal of Computational Biology*, 9(4):597–612, 2002.
- [AL06] K. Arun and C. J. Langmead. Structure based chemical shift prediction using Random Forests non-linear regression. *Proceedings of The Forth Asia-Pacific Bioinformatics Conference, (APBC)*, pages 317–326, 2006.
- [All02] F. H. Allen. The Cambridge Structural Database: a quarter of a million crystal structures and rising. *Acta Crystallogr B*, 58(Pt 3 Pt 1):380–388, Jun 2002.
- [AS64] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover, New York, Ninth Dover printing, Tenth GPO printing edition, 1964.
- [AST<sup>+</sup>84] I. Ando, H. Saito, R. Tabeta, A. Shoji, and T. Ozaki. Conformation-dependent  $^{13}\text{C}$  NMR chemical shifts of poly(L-alanine) in the solid state: FPT INDO calculation of N-Acetyl-N'-methyl-L-alanine amide as a model compound of poly(L-alanine). *Macromolecules*, 17(3):457–461, 1984.

## Bibliography

- [BAM<sup>+</sup>98] P. Bayer, A. Arndt, S. Metzger, R. Mahajan, F. Melchior, R. Jaenicke, and J. Becker. Structure determination of the small ubiquitin-related modifier SUMO-1. *Journal of Molecular Biology*, 280(2):275–286, 1998.
- [BBST09] S. Böcker, Q. B. A. Bui, P. Seeber, and A. Truss. Computing bond types in molecule graphs. In *Proc. of Computing and Combinatorics Conference (COCOON 2009)*, volume 5609 of *Lecture Notes in Computer Science*, pages 297–306. Springer, 2009.
- [BEN04] M. Berkelaar, K. Eikland, and P. Notebaert. Ip\_solve 5.5, Open source (Mixed-Integer) Linear Programming system. Software, May 1 2004. Available at <http://lpsolve.sourceforge.net/5.5/>.
- [BFK<sup>+</sup>06] H. L. Bodlaender, F. V. Fomin, A. M. Koster, D. Kratsch, and D. M. Thilikos. On exact algorithms for treewidth. Technical Report UU-CS-2006-032, Institute of Information and Computing Sciences, Utrecht University, 2006.
- [BG91] W. Bremser and M. Grzonka. SpecInfo—A multidimensional spectroscopic interpretation system. *Microchimica Acta*, 104:483–491, 1991. 10.1007/BF01245533.
- [BH92] J. C. Baber and E. E. Hodgkin. Automatic assignment of chemical connectivity to organic molecules in the Cambridge Structural Database. *Journal of Chemical Information and Computer Sciences*, 32:401–406, 1992.
- [BHN03] H. Berman, K. Henrick, and H. Nakamura. Announcing the worldwide Protein Data Bank. *Nature Structural Biology*, 10(12):980, Dec 2003.
- [BKO11] M. Beer, J. Kussmann, and C. Ochsenfeld. Nuclei-Selected NMR Shielding Calculations: A Sublinear-Scaling Quantum-Chemical Method. *Journal of Chemical Physics*, 134(7), 2011.
- [BKS07] P. Berman, M. Karpinski, and A. D. Scott. Computational complexity of some restricted instances of 3-SAT. *Discrete Applied Mathematics*, 155(5):649–653, 2007.
- [BLZ<sup>+</sup>10] M. Berjanskii, Y. Liang, J. Zhou, P. Tang, P. Stothard, Y. Zhou, J. Cruz, C. MacDonell, G. Lin, P. Lu, and D. S. Wishart. PROSESS: a protein structure evaluation suite and server. *Nucleic Acids Research*, 38(SUPPL. 2):W633–W640, Jul 2010.
- [BM02] I. Brown and B. McMahon. CIF: The computer language of crystallography. *Acta Crystallographica Section B: Structural Science*, 58(3 PART 1):317–324, 2002.
- [Bre01] L. Breiman. Random forests. *Machine Learning*, (1):5–32, 2001.
- [BS10] M. Brylinski and J. Skolnick. Comparison of structure-based and threading-based approaches to protein functional annotation. *Proteins*, 78(1):118–134, Jan 2010.
- [BTL<sup>+</sup>09] M. Berjanskii, P. Tang, J. Liang, J. A. Cruz, J. Zhou, Y. Zhou, E. Bassett, C. MacDonell, P. Lu, G. Lin, and D. S. Wishart. GeNMR: a web server for rapid NMR-based protein structure determination. *Nucleic Acids Research*, 37(Web Server issue):W670–W677, Jul 2009.
- [Buc60] A. D. Buckingham. Chemical shifts in the nuclear magnetic resonance spectra of molecules containing polar groups. *Canadian Journal of Chemistry*, 38:300 – 307, 1960.
- [Bui10] Q. B. A. Bui. *Fixed-Parameter Algorithms for some Combinatorial Problems in Bioinformatics*. PhD thesis, Friedrich Schiller Universität Jena, 2010.



- [BW05] M. Berjanskii and D. S. Wishart. A simple method to predict protein flexibility using secondary chemical shifts. *Journal for the American Chemical Society*, 127(43):14970–14971, Nov 2005.
- [BW06] M. Berjanskii and D. S. Wishart. NMR: prediction of protein flexibility. *Nature Protocols*, 1(2):683–688, 2006.
- [BW08] M. Berjanskii and D. S. Wishart. Application of the random coil index to studying protein flexibility. *Journal of Biomolecular NMR*, 40(1):31–48, Jan 2008.
- [BWF<sup>+</sup>00] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28:235–242, 2000.
- [CCB<sup>+</sup>95] W. D. Cornell, P. Cieplak, C. I. Bayly, I. R. Gould, K. M. Merz Jr., D. M. Ferguson, D. C. Spellmeyer, T. Fox, J. W. Caldwell, and P. A. Kollman. A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. *Journal of the American Chemical Society*, 117(19):5179–5197, 1995.
- [CCD<sup>+</sup>05] D. A. Case, T. E. Cheatham, T. Darden, H. Gohlke, R. Luo, K. M. Merz, A. Onufriev, C. Simmerling, B. Wang, and R. J. Woods. The Amber biomolecular simulation programs. *Journal of Computational Chemistry*, 26(16):1668–1688, December 2005.
- [CDB99] G. Cornilescu, F. Delaglio, and A. Bax. Protein backbone angle restraints from searching a database for chemical shift and sequence homology. *Journal of Biomolecular NMR*, 13(3):289–302, 1999.
- [CMV11] A. Cavalli, R. W. Montalvao, and M. Vendruscolo. Using Chemical Shifts to Determine Structural Changes in Proteins upon Complex Formation. *The Journal of Physical Chemistry. B*, June 2011.
- [Col10] S. Collin. *Leistungsdiagnostik im Schwimmsport. Vergleich zweier spiroergometrischer Belastungsprotokolle- Laufbandrampentest vs. Schwimmbanktest*. PhD thesis, Universität zu Köln, 2010.
- [CSDV07] A. Cavalli, X. Salvatella, C. M. Dobson, and M. Vendruscolo. Protein structure determination from NMR chemical shifts. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 104(23):9615–9620, Jun 2007.
- [CSK<sup>+</sup>03] R. Chenna, H. Sugawara, T. Koike, R. Lopez, T. J. Gibson, D. G. Higgins, and J. D. Thompson. Multiple sequence alignment with the Clustal series of programs. *Nucleic Acids Research*, 31(13):3497–3500, 2003.
- [CTK11] S. Canzar, N. C. Toussaint, and G. W. Klau. An exact algorithm for side-chain placement in protein design. *Optimization Letters*, 5(3):393–406, 2011.
- [DC97] R. L. Dunbrack and F. E. Cohen. Bayesian statistical analysis of protein side-chain rotamer preferences. *Protein Science*, 6(8):1661–1681, August 1997.
- [Deh07] A. K. Dehof. Development and Implementation of an NMR-based Scoring Function for the Evaluation of Docking Results. Master's thesis, Saarland University, Germany, 2007.
- [DF99] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.

## Bibliography

- [DFS<sup>+</sup>11] F. Doloresco, C. Fominaya, G. T. Schumock, L. C. Vermeulen, L. Matusiak, R. J. Hunkler, N. D. Shah, and J. M. Hoffman. Projecting future drug expenditures: 2011. *American Journal of Health-System Pharmacy*, 68(10):921–32, May 2011.
- [DG04] M. Dickson and J. P. Gagnon. Key factors in the rising cost of new drug discovery and development. *Nature Reviews Drug Discovery*, 3(5):417–429, May 2004.
- [DGM<sup>+</sup>09] A. K. Dehof, I. Georgiev, L. Marsalek, S. Nickels, H.-P. Lenhof, P. Slusallek, and A. Hildebrandt. Visual Computing in Computer Aided Drug Design. In *Visual Computing Research Conference*, Saarbrücken, 2009.
- [DHG03] J. A. DiMasi, R. W. Hansen, and H. G. Grabowski. The price of innovation: new estimates of drug development costs. *Journal of Health Economics*, 22(2):151–85, March 2003.
- [DiM01] J. A. DiMasi. New drug development in the United States from 1963 to 1999. *Clinical Pharmacology and Therapeutics*, 69(5):286–296, May 2001.
- [DK93] R. L. Dunbrack and M. Karplus. Backbone-dependent Rotamer Library for Proteins Application to Side-chain Prediction. *Journal of Molecular Biology*, 230(2):543 – 574, 1993.
- [DK94] R. L. Dunbrack and M. Karplus. Conformational analysis of the backbone-dependent rotamer preferences of protein sidechains. *Nature Structural Biology*, 1(5):334–340, May 1994.
- [DLH11] A. K. Dehof, H.-P. Lenhof, and A. Hildebrandt. Predicting Protein NMR Chemical Shifts in the Presence of Ligands and Ions using Force Field-based Features. In *Proceedings of the German Conference on Bioinformatics (GCB)*, 2011.
- [DRB<sup>+</sup>11] A. K. Dehof, A. Rurainski, Q. B. A. Bui, S. Böcker, H.-P. Lenhof, and A. Hildebrandt. Automated bond order assignment as an optimization problem. *Bioinformatics*, 27(5):619–25, March 2011.
- [DRLH09] A. K. Dehof, A. Rurainski, H.-P. Lenhof, and A. Hildebrandt. Automated bond order assignment as an optimization problem. In *Proceedings of the German Conference on Bioinformatics (GCB)*, pages 201–209, 2009.
- [FG06] J. Flum and M. Grohe. *Parameterized Complexity Theory*, volume 7. Springer, 2006.
- [FH05] M. Froeyen and P. Herdewijn. Correct bond order assignment in a molecular framework using integer linear programming with application to molecules where only non-hydrogen atom coordinates are available. *Journal of Chemical Information and Modelling*, 45(5):1267–1274, 2005.
- [Fie00] R. T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.
- [Fis94a] E. Fischer. Einfluss der Configuration auf die Wirkung der Enzyme. *Berichte der deutschen chemischen Gesellschaft*, 27(3):2985–2993, 1894.
- [Fis94b] E. Fischer. Synthesen in der Zuckergruppe II. *Berichte der deutschen chemischen Gesellschaft*, 27(3):3189–3232, 1894.
- [FOME11] A. Frank, I. Onila, H. M. Möller, and T. E. Exner. Toward the quantum chemical calculation of nuclear magnetic resonance chemical shifts of proteins. *Proteins: Structure, Function, and Bioinformatics*, 2011.

- [GBS<sup>+</sup>97] W. Gronwald, R. F. Boyko, F. D. Sönnichsen, D. S. Wishart, and B. D. Sykes. ORB, a homology-based program for the prediction of protein NMR chemical shifts. *Journal of Biomolecular NMR*, 10(2), 1997.
- [GD04] V. Gogate and R. Dechter. A complete anytime algorithm for treewidth. In *UAI '04: Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 201–208. AUAI Press, 2004.
- [GGCH07] S. W. Ginzinger, F. Gerick, M. Coles, and V. Heun. CheckShift: Automatic correction of inconsistent chemical shift referencing. *Journal of Biomolecular NMR*, 39(3):223–227, 2007.
- [GHH<sup>+</sup>06] R. Guha, M. T. Howard, G. R. Hutchison, P. Murray-Rust, H. Rzepa, C. Steinbeck, J. Wegner, and E. L. Willighagen. The Blue Obelisk-interoperability in chemical informatics. *Journal of Chemical Information and Modelling*, 46(3):991–998, 2006.
- [GHJV94] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design patterns: elements of reusable object-oriented software*. Addison-Wesley Professional, 1st edition, 1994.
- [GNTT10] J. Goecks, A. Nekrutenko, J. Taylor, and T. G. Team. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biology*, 11(8):R86, 2010.
- [GOH<sup>+</sup>02] S. Goto, Y. Okuno, M. Hattori, T. Nishioka, and M. Kanehisa. LIGAND: database of chemical compounds and reactions in biological pathways. *Nucleic Acids Research*, 30(1):402–404, Jan 2002.
- [Goo08] M. Goodman. Market Watch: Pharma industry performance metrics: 2007–2012E. *Nature Reviews Drug Discovery*, 7(10):795, October 2008.
- [GS08] I. Georgiev and P. Slusallek. RTfact: Generic Concepts for Flexible and High Performance Ray Tracing. In *IEEE/Eurographics Symposium on Interactive Ray Tracing 2008*, August 2008.
- [GSBW00] P. Guntert, M. Salzmann, D. Braun, and K. Wuthrich. Sequence-specific NMR assignment of proteins by global fragment mapping with the program MAPPER. *Journal of Biomolecular NMR*, 18(2):129–137, 2000.
- [GYK<sup>+</sup>07] Q. Gao, S. Yokojima, T. Kohno, T. Ishida, D. G. Fedorov, K. Kitaura, M. Fujihira, and S. Nakamura. Ab initio NMR chemical shift calculations on proteins using fragment molecular orbitals with electrostatic environment. *Chemical Physics Letters*, 445(4-6):331–339, 2007.
- [HAB91] S. R. Hall, F. H. Allen, and I. D. Brown. The crystallographic information file (CIF): a new standard archive file for crystallography. *Acta Crystallographica Section A*, 47(6):655–685, Nov 1991.
- [Hal96a] T. A. Halgren. Merck molecular force field. I. Basis, form, scope, parameterization, and performance of MMFF94. *Journal of Computational Chemistry*, 17(5-6):490–519, 1996.
- [Hal96b] T. A. Halgren. MMFF VI. MMFF94s option for energy minimization studies. *Journal of Computational Chemistry*, 17:490–519, 1996.
- [HC95] S. R. Hall and A. P. F. Cook. STAR dictionary definition language: Initial specification. *Journal of Chemical Information and Computer Sciences*, 35(5):819–825, 1995.

## Bibliography

- [HDR<sup>+</sup>10] A. Hildebrandt, A. K. Dehof, A. Rurainski, A. Bertsch, M. Schumann, N. Toussaint, A. Moll, D. Stockel, S. Nickels, S. Mueller, H.-P. Lenhof, and O. Kohlbacher. BALL - Biochemical Algorithms Library 1.3. *BMC Bioinformatics*, 11(1):531, 2010.
- [HDS<sup>+</sup>11] A. Hildebrandt, A. K. Dehof, D. Stöckel, S. Nickels, S. Mueller, M. Schumann, H.-P. Lenhof, and O. Kohlbacher. The BALL project: The Biochemical Algorithms Library (BALL) for Rapid Application Development in Structural Bioinformatics and its graphical user interface BALLView. In *Proceedings of the 12th Annual Bioinformatics Open Source Conference BOSC 2011*, pages 24, 29, 2011.
- [HGPF88] M. Head-Gordon, J. A. Pople, and M. J. Frisch. MP2 energy evaluation by direct methods. *Chemical Physics Letters*, 153(6):503–506, 1988.
- [Hil05] A. Hildebrandt. *Biomolecules in a structured solvent: a novel formulation of nonlocal electrostatics and its numerical solution*. Rhombos-Verl., 2005.
- [HK64] P. Hohenberg and W. Kohn. Inhomogeneous electron gas. *Physical Review*, 136(3B):B864–B871, 1964.
- [HLGW11] B. Han, Y. Liu, S. Ginzinger, and D. Wishart. SHIFTX2: significantly improved protein chemical shift prediction. *Journal of Biomolecular NMR*, pages 1–15, 2011.
- [HM72] C. Haigh and R. Mallion. New tables of ring current shielding in proton magnetic resonance. *Organic Magnetic Resonance*, 4(2):203–228, 1972.
- [HM79] C. Haigh and R. Mallion. Ring current theories in nuclear magnetic resonance. *Progress in Nuclear Magnetic Resonance Spectroscopy*, 13(4):303–344, 1979.
- [HNR68] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics SSC4*, 4:100–107, 1968.
- [HRB97] M. Hendlich, F. Rippmann, and G. Barnickel. BALL: Automatic assignment of bond and atom types for protein ligands in the brookhaven protein databank. *Journal of Chemical Information and Computer Sciences*, 37:774–778, 1997.
- [HTF09] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. Springer Series in Statistics. Springer, 5th Printing edition, September 2009.
- [IS05] J. J. Irwin and B. K. Shoichet. ZINC—a free database of commercially available compounds for virtual screening. *Journal of Chemical Information and Modelling*, 45(1):177–182, 2005.
- [JJB58] C. Johnson Jr. and F. Bovey. Calculation of nuclear magnetic resonance spectra of aromatic hydrocarbons. *The Journal of Chemical Physics*, 29(5):1012–1014, 1958.
- [JJB02] A. Jakalian, D. B. Jack, and C. I. Bayly. Fast, efficient generation of high-quality atomic charges. AM1-BCC model: II. parameterization and validation. *Journal of Computational Chemistry*, 23(16):1623–1641, Dec 2002.
- [KBM<sup>+</sup>01] O. Kohlbacher, A. Burchardt, A. Moll, A. Hildebrandt, P. Bayer, and H.-P. Lenhof. Structure prediction of protein complexes by an NMR-based protein docking algorithm. *Journal of Biomolecular NMR*, 20(1):15–21, May 2001.

- [KCB04] D. E. Kim, D. Chivian, and D. Baker. Protein structure prediction and analysis using the rosetta server. *Nucleic Acids Research*, 32(Web Server issue):W526–W531, Jul 2004.
- [KENS08] S. Kuhn, B. Egert, S. Neumann, and C. Steinbeck. Building blocks for automated elucidation of metabolites: Machine learning methods for NMR prediction. *BMC Bioinformatics*, 9(1):400, 2008.
- [KL00] O. Kohlbacher and H. P. Lenhof. BALL—rapid software prototyping in computational molecular biology. Biochemicals Algorithms Library. *Bioinformatics*, 16(9):815–824, Sep 2000.
- [KLHT09] B. Kneissl, B. Leonhardt, A. Hildebrandt, and C. S. Tautermann. Revisiting Automated G-Protein Coupled Receptor Modeling: The Benefit of Additional Template Structures for a Neurokinin-1 Receptor Model. *Journal of Medicinal Chemistry*, 52(10):3166–3173, 2009.
- [KMT11] B. Kneissl, S. Mueller, C. S. Tautermann, and A. Hildebrandt. String Kernels and High-Quality Data Set for Improved Prediction of Kinked Helices in  $\alpha$ -Helical Membrane Proteins. *Journal of Chemical Information and Modeling*, October 2011.
- [KO07] J. Kussmann and C. Ochsenfeld. Linear-Scaling Method for Calculating Nuclear Magnetic Resonance Chemical Shifts Using Gauge-Including Atomic Orbitals within Hartree-Fock and Density-Functional Theory. *Journal of Chemical Physics*, 127(5), 2007.
- [Koc76] B. Koch. *Darstellung und Charakterisierung von multiplen Formen der Alpha-N-Acetylgalaktosaminidase*. PhD thesis, Universität Münster, 1976.
- [Kos58] D. E. Koshland. Application of a Theory of Enzyme Specificity to Protein Synthesis. *Proceedings of the National Academy of Sciences of the United States of America*, 44(2):98–104, February 1958.
- [KRC<sup>+</sup>09] K. J. Kohlhoff, P. Robustelli, A. Cavalli, X. Salvatella, and M. Vendruscolo. Fast and Accurate Predictions of Protein NMR Chemical Shifts from Interatomic Distances. *Journal of the American Chemical Society*, 131(39):13894–13895, 2009.
- [KS65] W. Kohn and L. Sham. Self-consistent equations including exchange and correlation effects. *Physical Review*, 140(4A):A1133–A1138, 1965.
- [KS83] W. Kabsch and C. Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12):2577–2637, 1983.
- [Lab05] P. Labute. On the perception of molecules from 3D atomic coordinates. *Journal of Chemical Information and Modelling*, 45(2):215–221, 2005.
- [Lam07] S. L. Lam. DSHIFT: a web server for predicting DNA chemical shifts. *Nucleic Acids Research*, 35(suppl 2):W713–W717, 2007.
- [Lan19] I. Langmuir. The arrangement of electrons in atoms and molecules. *Journal of the American Chemical Society*, 41(6):868–934, 1919.
- [LdDO93] D. Laws, A. de Dios, and E. Oldfield. NMR chemical shifts and structure refinement in proteins. *Journal of Biomolecular NMR*, 3(5):607–612, 1993.
- [LDH82] K. L. D. Hof. *Untersuchungen zur Reaktionslage des sympathischen Nervensystems nach Meditation (TM)*. PhD thesis, Universität zu Köln, 1982.

## Bibliography

- [Lew16] G. N. Lewis. The atom and the molecule. *Journal of the American Chemical Society*, 38(4):762–785, 1916.
- [Lic09] F. R. Lichtenberg. The Quality of Medical Care, Behavioral Risk Factors, and Longevity Growth. *National Bureau of Economic Research Working Paper Series*, No. 15068, 2009.
- [LL98] A. R. Leach and A. P. Lemon. Exploring the conformational space of protein side chains using dead-end elimination and the A\* algorithm. *Proteins*, 33(2):227–239, Nov 1998.
- [LMP04] C. A. Lepre, J. M. Moore, and J. W. Peng. Theory and applications of NMR-based screening in pharmaceutical research. *Chemical Reviews*, 104(8):3641–76, August 2004.
- [LW02] A. Liaw and M. Wiener. Classification and regression by randomForest. *R News*, 2(3):18–22, 2002.
- [McC57] H. McConnell. Theory of Nuclear Magnetic Shielding in Molecules. I. Long-Range Dipolar Shielding of Protons. *Journal of Chemical Physics*, 27:226–229, 1957.
- [MCS<sup>+</sup>08] R. W. Montalvo, A. Cavalli, X. Salvatella, T. L. Blundell, and M. Vendruscolo. Structure determination of protein-protein complexes using NMR chemical shifts: Case of an endonuclease colicin-immunity protein complex. *Journal of the American Chemical Society*, 130(47):15990–15996, 2008.
- [Mei03] J. Meiler. PROSHIFT: protein chemical shift prediction using artificial neural networks. *Journal of Biomolecular NMR*, 26(1):25–37, May 2003.
- [MGD<sup>+</sup>10] L. Marsalek, I. Georgiev, A. K. Dehof, H.-P. Lenhof, P. Slusallek, and A. Hildebrandt. Real-time ray tracing of complex molecular scenes. In *Proceedings of the International Conference on Information Visualisation*, pages 239–245, 2010.
- [MHLK05] A. Moll, A. Hildebrandt, H.-P. Lenhof, and O. Kohlbacher. BALLView: an object-oriented molecular visualization and modeling framework. *Journal of Computer-Aided Molecular Design*, 19(11):791–800, November 2005.
- [MHLK06] A. Moll, A. Hildebrandt, H.-P. Lenhof, and O. Kohlbacher. BALLView: a tool for research and education in molecular modeling. *Bioinformatics*, 22(3):365–366, 2006.
- [MK09] S. Mielke and V. Krishnan. Characterization of protein secondary structure from nmr chemical shifts. *Progress in Nuclear Magnetic Resonance Spectroscopy*, 54(3-4):141–165, 2009.
- [MKTB04] A. Morozov, T. Kortemme, K. Tsemekhman, and D. Baker. Close agreement between the orientation dependence of hydrogen bonds observed in protein structures and quantum mechanical calculations. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 101(18):6946–6951, 2004.
- [MSP09] C. K. Materese, A. Savelyev, and G. A. Papoian. Counterion atmosphere and hydration patterns near a nucleosome core particle. *Journal for the American Chemical Society*, 131(41):15005–15013, Oct 2009.
- [MTK<sup>+</sup>10] N. Maghsoudi, N. K. Tafreshi, F. Khodaghali, Z. Zakeri, M. Esfandiari, H. Hadi-Alijanvand, M. Sabbaghian, A. H. Maghsoudi, M. Sajadi, M. Zohri, M. Moosavi, and M. Zeinodini. Targeting enteroviral 2A protease by a 16-mer synthetic peptide: inhibition of 2Apro-induced apoptosis in a stable Tet-on HeLa cell line. *Virology*, 399(1):39–45, Mar 2010.

- [Mue08] S. Mueller. Bond- and Atomtyping for the BALL-library. Bachelor Thesis Saarland University, Center for Bioinformatics, Bachelor's Program in Bioinformatics, 2008.
- [MWC<sup>+</sup>08] S. K. Meegalla, M. J. Wall, J. Chen, K. J. Wilson, S. K. Ballentine, R. L. Desjarlais, C. Schubert, C. S. Crysler, Y. Chen, C. J. Molloy, M. A. Chaikin, C. L. Manthey, M. R. Player, B. E. Tomczuk, and C. R. Illig. Structure-based optimization of a potent class of arylamide FMS inhibitors. *Bioorganic & Medicinal Chemistry Letters*, 18(12):3632–3637, Jun 2008.
- [Nac13] A. Nacken. *Über Messungen im Magnesium-Spektrum nach internationalen Normalen*. PhD thesis, Universität Bonn, 1913.
- [Nie06] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.
- [NMH<sup>+</sup>02] J. W. M. Nissink, C. Murray, M. Hartshorn, M. L. Verdonk, J. C. Cole, and R. Taylor. A new test set for validating predictions of protein-ligand interaction. *Proteins*, 49(4):457–471, Dec 2002.
- [NMR] NMRPredict. [http://www.modgraph.co.uk/product\\_nmr.htm](http://www.modgraph.co.uk/product_nmr.htm).
- [NNZW03] S. Neal, A. M. Nip, H. Zhang, and D. S. Wishart. Rapid and accurate calculation of protein 1H, 13C and 15N chemical shifts. *Journal of Biomolecular NMR*, 26(3):215–240, Jul 2003.
- [ÖC94] K. Ösapay and D. A. Case. Analysis of proton chemical shifts in regular secondary structure of proteins. *Journal of Biomolecular NMR*, 4(2):215–230, 1994.
- [OKK04] C. Ochsenfeld, J. Kussmann, and F. Koziol. Ab initio NMR spectra for molecular systems with a thousand and more atoms: A linear-scaling method. *Angewandte Chemie - International Edition*, 43(34):4485–4489, 2004.
- [PB82] G. Purvis and R. Bartlett. A full coupled-cluster singles and doubles model: The inclusion of disconnected triples. *The Journal of Chemical Physics*, 76(4):1910–1918, 1982.
- [PC03] J. W. Ponder and D. A. Case. Force Fields for Protein Simulation. *Advantages in Protein Chemistry*, 66:27–85, 2003.
- [PGD<sup>+</sup>10] M. Phillips, I. Georgiev, A. K. Dehof, S. Nickels, L. Marsalek, H.-P. Lenhof, A. Hildebrandt, and P. Slusallek. Measuring properties of molecular surfaces using ray casting. In *2010 IEEE International Symposium on High Performance Computational Biology*, pages 1–7. IEEE, 2010.
- [Pha11] Pharmaceutical Research and Manufacturers of America. PhRMA 2011 profile. Technical report, 2011.
- [PMD<sup>+</sup>10] S. M. Paul, D. S. Mytelka, C. T. Dunwiddie, C. C. Persinger, B. H. Munos, S. R. Lindborg, and A. L. Schacht. How to improve R&D productivity: the pharmaceutical industry's grand challenge. *Nature Reviews Drug Discovery*, 9(3):203–14, March 2010.
- [Pop58] J. A. Pople. Molecular orbital theory of aromatic ring currents. *Molecular Physics*, 1(2):175 – 180, 1958.
- [PS98] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Courier Dover Publications, 1998.

## Bibliography

- [R D11] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2011. ISBN 3-900051-07-0.
- [RRK10] M. Roettig, C. Rausch, and O. Kohlbacher. Combining structure and sequence information allows automated prediction of substrate specificities within enzyme families. *PLoS Computational Biology*, 6(1), 2010.
- [RS86] N. Robertson and P. D. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *Journal of Algorithms*, 7(3):309 – 322, 1986.
- [RSH<sup>+</sup>10] P. Rossi, G. Swapna, Y. Huang, J. Aramini, C. Anklin, K. Conover, K. Hamilton, R. Xiao, T. Acton, A. Ertekin, J. Everett, and G. Montelione. A microscale protein NMR sample screening pipeline. *Journal of Biomolecular NMR*, 46:11–22, 2010.
- [Rur10] A. Rurainski. *Optimization in Bioinformatics*. PhD thesis, Universität des Saarlandes, Saarbrücken, 2010.
- [RV10] W. Rieping and W. F. Vranken. Validation of archived chemical shifts through atomic coordinates. *Proteins*, 78(11):2482–2489, Aug 2010.
- [SB07] Y. Shen and A. Bax. Protein backbone chemical shifts predicted from searching a database for torsion angle and sequence homology. *Journal of Biomolecular NMR*, 38(4):289–302, Aug 2007.
- [SB10] Y. Shen and A. Bax. SPARTA+: a modest improvement in empirical NMR chemical shift prediction by means of an artificial neural network. *Journal of Biomolecular NMR*, (48):13–22, 2010.
- [SBC<sup>+</sup>08] Y. D. Smurnyy, K. A. Blinov, T. S. Churanova, M. E. Elyashberg, and A. J. Williams. Toward More Reliable <sup>13</sup>C and <sup>1</sup>H Chemical Shift Prediction: A Systematic Comparison of Neural-Network and Least-Squares Regression Based Approaches. *Journal of Chemical Information and Modeling*, 48(1):128–134, 2008.
- [SC08] S. Sonavane and P. Chakrabarti. Cavities and atomic packing in protein structures and interfaces. *PLoS Comput Biol*, 4(9):e1000188, 09 2008.
- [SDCB09] Y. Shen, F. Delaglio, G. Cornilescu, and A. Bax. TALOS+: A hybrid method for predicting protein backbone torsion angles from NMR chemical shifts. *Journal of Biomolecular NMR*, (44):213–223, 2009.
- [SGSA06] Y. H. Singh, M. M. Gromiha, A. Sarai, and S. Ahmad. Atom-wise statistics and prediction of solvent accessibility in proteins. *Biophysical Chemistry*, 124(2):145 – 154, 2006.
- [SK04] C. Steinbeck and S. Kuhn. NMRShiftDB – compound identification and structure elucidation support through a free community-built web database. *Phytochemistry*, 65(19):2711 – 2717, 2004.
- [SKK03] C. Steinbeck, S. Krause, and S. Kuhn. NMRShiftDB – Constructing a free chemical information system with open-source components. *Journal of Chemical Information and Computer Sciences*, 43(6), NOV 2003.
- [SLD<sup>+</sup>08] Y. Shen, O. Lange, F. Delaglio, P. Rossi, J. M. Aramini, G. Liu, A. Eletsy, Y. Wu, K. K. Singarapu, A. Lemak, A. Ignatchenko, C. H. Arrowsmith, T. Szyperski, G. T. Montelione, D. Baker, and A. Bax. Consistent blind protein structure generation from NMR chemical shift data. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 105(12):4685–4690, Mar 2008.



- [SMA] SMARTS – A Language for Describing Molecular Patterns. <http://www.daylight.com/dayhtml/doc/theory/theory.smarts.html>.
- [SOS96] M. F. Sanner, A. J. Olson, and J. C. Spehner. Reduced surface: an efficient way to compute molecular surfaces. *Biopolymers*, 38(3):305–320, Mar 1996.
- [SP09] A. Savel'yev and G. A. Papoian. Molecular renormalization group coarse-graining of polymer chains: application to double-stranded DNA. *Biophysical Journal*, 96(10):4044–4052, May 2009.
- [SSH94] D. Sitkoff, K. Sharp, and B. Honig. Accurate calculation of hydration free energies using macroscopic solvent models. *Journal of Physical Chemistry*, 98(7):1978–1988, 1994.
- [SWBG08] E. Segev, T. Wyttenbach, M. T. Bowers, and R. B. Gerber. Conformational evolution of ubiquitin ions in electrospray mass spectrometry: molecular dynamics simulations at gradually increasing temperatures. *Physical Chemistry Chemical Physics*, 10(21):3077–3082, Jun 2008.
- [THG94] J. D. Thompson, D. G. Higgins, and T. J. Gibson. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22(22):4673–4680, 1994.
- [UAD<sup>+</sup>08] E. Ulrich, H. Akutsu, J. Doreleijers, Y. Harano, Y. Ioannidis, J. Lin, M. Livny, S. Mading, D. Maziuk, Z. Miller, E. Nakatani, C. Schulte, D. Tolmie, R. K. Wenger, H. Yao, and J. Markley. BioMagResBank. *Nucleic Acids Research*, 36(SUPPL. 1):D402–D408, 2008.
- [Uni09] United Nations, Department of Economic and Social Affairs, Population Division. World Population Prospects: The 2008 Revision. Technical report, New York, 2009.
- [Van71] J. R. Vane. Inhibition of prostaglandin synthesis as a mechanism of action for aspirin-like drugs. *Nature New Biology*, 231(25):232–235, June 1971.
- [VR02] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. ISBN 0-387-95457-0.
- [VR09] W. F. Vranken and W. Rieping. Relationship between chemical shift value and accessible surface area for all amino acid atoms. *BMC Structural Biology*, 9:20, 2009.
- [WA93] M. P. Williamson and T. Asakura. Empirical comparisons of models for chemical-shift calculation in proteins. *Journal of Magnetic Resonance, Series B*, 101(1):63–71, 1993.
- [WAB<sup>+</sup>08] D. S. Wishart, D. Arndt, M. Berjanskii, P. Tang, J. Zhou, and G. Lin. CS23D: a web server for rapid protein structure generation using NMR chemical shifts and sequence data. *Nucleic Acids Research*, 36(Web Server issue):W496–W502, Jul 2008.
- [Wan10] J. Wang. *AmberTools Users' Manual*. University of Texas, Southwestern Medical Center, April 2010.
- [WBH<sup>+</sup>95] D. S. Wishart, C. G. Bigam, A. Holm, R. Hodges, and B. D. Sykes. <sup>1</sup>H, <sup>13</sup>C and <sup>15</sup>N random coil NMR chemical shifts of the common amino acids. I. Investigations of nearest-neighbor effects. *Journal of Biomolecular NMR*, 5(3):332, 1995.
- [WBY<sup>+</sup>95] D. S. Wishart, C. G. Bigam, J. Yao, F. Abildgaard, H. J. Dyson, E. Oldfield, J. L. Markley, and B. D. Sykes. <sup>1</sup>H, <sup>13</sup>C and <sup>15</sup>N chemical shift referencing in biomolecular NMR. *Journal of Biomolecular NMR*, 6:135–140, 1995.

## Bibliography

- [WDJ05] G. Wang and R. Dunbrack Jr. PISCES: Recent improvements to a PDB sequence culling server. *Nucleic Acids Research*, 33(SUPPL. 2):W94–W98, 2005.
- [Wil90] M. P. Williamson. Secondary-structure dependent chemical shifts in proteins. *Biopolymers*, 29(10-11):1423–1431, 1990.
- [Wis11] D. S. Wishart. Interpreting protein chemical shift data. *Progress in Nuclear Magnetic Resonance Spectroscopy*, 58(1):62–87, 2011.
- [WKH97] S. S. Wijmenga, M. Kruihof, and C. W. Hilbers. Analysis of  $^1\text{H}$  chemical shifts in DNA: Assessment of the reliability of  $^1\text{H}$  chemical shift calculations for use in structure refinement. *Journal of Biomolecular NMR*, 10:337–350, 1997.
- [WPW83] G. Wagner, A. Pardi, and K. Wüthrich. Hydrogen bond length and  $^1\text{H}$  NMR chemical shifts in proteins. *Journal of the American Chemical Society*, 105(18):5948–5949, 1983.
- [WRZ<sup>+</sup>03] L. Willard, A. Ranjan, H. Zhang, H. Monzavi, R. Boyko, B. D. Sykes, and D. S. Wishart. VADAR: A web server for quantitative evaluation of protein structure quality. *Nucleic Acids Research*, 31(13):3316–3319, 2003.
- [WS07] M. Wiederstein and M. Sippl. ProSA-web: interactive web service for the recognition of errors in three-dimensional structures of proteins. *Nucleic Acids Research*, 35(Web Server issue):W407–410, 2007.
- [WSR91] D. S. Wishart, B. D. Sykes, and F. M. Richards. Relationship between nuclear magnetic resonance chemical shift and protein secondary structure. *Journal of Molecular Biology*, 222(2):311–333, Nov 1991.
- [WSR92] D. S. Wishart, B. D. Sykes, and F. M. Richards. The chemical shift index: a fast and simple method for the assignment of protein secondary structure through NMR spectroscopy. *Biochemistry*, 31(6):1647–1651, Feb 1992.
- [WWBS97] D. S. Wishart, M. Watson, R. Boyko, and B. D. Sykes. Automated  $^1\text{H}$  and  $^{13}\text{C}$  chemical shift prediction using the BioMagResBank. *Journal of Biomolecular NMR*, 10(4):329–336, 1997.
- [WWC<sup>+</sup>04] J. Wang, R. M. Wolf, J. W. Caldwell, P. A. Kollman, and D. A. Case. Development and testing of a general amber force field. *Journal of Computational Chemistry*, 25(9):1157–1174, Jul 2004.
- [WWKC01] J. Wang, W. Wang, P. A. Kollman, and D. A. Case. Antechamber, An Accessory Software Package For Molecular Mechanical Calculations. *Molecules*, 222(2):U403–U403, 2001.
- [WWKC06] J. Wang, W. Wang, P. A. Kollman, and D. A. Case. Automatic atom type and bond type perception in molecular mechanical calculations. *Journal of Molecular Graphics and Modelling*, 25(2):247–260, Oct 2006.
- [XB06] J. Xu and B. Berger. Fast and accurate algorithms for protein side-chain packing. *Journal of ACM*, 53:533–557, 2006.
- [XC01] X. P. Xu and D. A. Case. Automated prediction of  $^{15}\text{N}$ ,  $^{13}\text{C}^\alpha$ ,  $^{13}\text{C}^\beta$  and  $^{13}\text{C}'$  chemical shifts in proteins using a density functional database. *Journal of Biomolecular NMR*, 21(4):321–333, Dec 2001.
- [XJB05] J. Xu, F. Jiao, and B. Berger. A tree-decomposition approach to protein structure prediction. *Proc IEEE Comput Syst Bioinform Conf*, pages 247–256, 2005.

- [XJB07] J. Xu, F. Jiao, and B. Berger. A parameterized algorithm for protein structure alignment. *Journal of Computational Biology*, 14(5):564–577, Jun 2007.
- [YSFW08] C. Yanover, O. Schueler-Furman, and Y. Weiss. Minimizing and learning energy functions for side-chain prediction. *Journal of Computational Biology*, 15(7):899–911, Sep 2008.
- [ZALL10] Y. Zhao, B. Alipanahi, S. Li, and M. Li. Protein secondary structure prediction using NMR chemical shift data. *Journal of Bioinformatics and Computational Biology*, 8(5):867–884, 2010.
- [ZCW07] Y. Zhao, T. Cheng, and R. Wang. Automatic perception of organic molecules based on essential structural information. *Journal of Chemical Information and Modelling*, 47(4):1379–1385, 2007.
- [ZNW03] H. Zhang, S. Neal, and D. S. Wishart. RefDB: A database of uniformly referenced protein chemical shifts. *Journal of Biomolecular NMR*, 25:173–195, 2003. 10.1023/A:1022836027055.
- [ZST<sup>+</sup>11] H. Zellner, M. Staudigel, T. Trenner, M. Bittkowski, V. Wolowski, C. Icking, and R. Merkl. Prescont: Predicting protein-protein interfaces utilizing four residue properties. *Proteins: Structure, Function, and Bioinformatics*, 2011.

