Bioinformatische Anwendung von *Graphlets* zur Analyse von Proteinstrukturtopologien
Rohfassung

Ben Haladik

10. März 2016

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation

Die verschiedenen Funktionen, die Proteine erfüllen sind zu einem großen Teil durch ihre Struktur bestimmt. Diese Struktur ist ein bestimmender Faktor, für ihre Fähigkeit Liganden zu binden und chemische Reaktionen zu katalysieren. Daraus folgt, dass die Analyse von Proteinstrukturen zentral für ein tieferes Verständnis von zellulärem Leben ist. Die vergleichende Analyse von Proteinen liefert nicht nur wichtige Erkenntnise über ihre Funktion, sondern hilft auch dabei, evolutionäre Verwandschaften zu entdecken, die durch reine Sequenzanalysen nicht mehr nachvollziehbar sind.

In dieser Arbeit werden Proteinstrukturtopologien untersucht. Die Topologie eines Proteins ist als die Anordnung seiner Sekundärstrukturelemente (SSEs) zueinander definiert. Die Betrachtung von SSEs hat den Vorteil, dass diese auch über große evolutionäre Distanzen stark konserviert sind.

Zur Darstellung dieser Topologien werden hier Graphen verwendet. Graphen gehören zu den am stärksten untersuchten mathematischen Strukturen, da sich mit ihnen viele verschiedene komplexe Zusammenhänge darstellen lassen. Ihre Nutzung ist überall dort angebracht, wo Daten nicht als Zahlen oder Vektoren darstellbar sind, weil sie eine Menge von Objekten und ihren Beziehungen untereinander repräsentieren.

Graphen finden sich in der Erforschung von sozialen Netzwerken wieder und werden in der Chemie zur Darstellung von Molekülen genutzt. Auch in der Bioinformatik sind Graphen das zentrale Mittel zur Darstellung komplexer Beziehungen. Interaktionen von Proteinen werden genauso als Graphen modelliert, wie Signalewege in Zellen oder eben Proteine. Die Analyse von biologischen Netzwerken ist ein zentrales Mittel, um biologische Vorgänge besser zu verstehen [4].

Der Vergleich von Graphen ist jedoch keine triviale Aufgabe. Die Frage: "Ist dieser Graph in diesem anderen Graphen enthalten?" befriedigend schnell beantworten zu können, ist nicht möglich, denn das zugrunde liegende Entschei-

dungsproblem ist NP-vollständig [5]

Um dieses Problem zu umgehen werden hier *Graphlets* angewendet. *Graphlets* sind kleine, induzierte Teilgraphen, die für große Graphen immer noch schnell abzählbar sind. Mit dieser Technik lassen sich in polynomieller Zeit Teilstrukturen eines Graphen ermitteln, deren Vergleich einfacher durchzuführen ist, als ein direkter Vergleich der Graphen selbst. So lassen sich *Graphlets* zur vergleichenden Analyse von Proteinstrukturtopologien verwenden.

1.2 *State of the art*

Mit der wachsenden Anzahl von Struktureeinträgen in der *Protein Data Bank* (PDB) ist eine Vielzahl von Methoden entstanden, um diese Strukturen zu vergleichen.

Der wohl bekannteste Algorithmus zum Vergleich von dreidimensionalen Proteinstrukturen ist DALI von *Holm* und *Sander* [3]. Er führt ein globales Alignment durch, indem Distanzmatrizen verglichen werden. In diesen Matrizen sind die intramolekularen Distanzen der Cα-Atome der jeweiligen Proteine eingetragen.

In den 23 Jahren, die seit der Veröffentlichung von DALI vergangen sind, sind aber noch viele weitere Methoden mit unterschiedlichen Ansätzen entwickelt worden. Der Algorithmus von *Shindyalov* und *Bourne* [13] berechnet ein Alignment von Proteinstrukturen, indem er zunächst kleine Paare von Substrukturen aligniert und dann versucht dieses Alignment auf einen optimalen Pfad auszudehnen.

Der FATCAT-Algorithmus von *Ye* und *Godzik* [14] verwendet eine ähnliche Idee und versucht zusätzlich die Substrukturen flexibel zu alignieren, um auch gleiche Proteine mit veränderter Konformation erkennen zu können.

TM-align von *Zhang* und *Skolnick* [15] berechnet eine Rotationsmatrix und nutzt Dynamische Programmierung. Der Algorithmus benutzt als Bewertungsschema den sogenannten *TM-Score*, der besonders geeignet ist, um lokale Ähnlichkeiten zu erkennen.

Der SSM-Algorithmus von *Krissinel* und *Hendrick* [6] verwendet eine graphenbasierte Darstellung von Proteinen für ein erstes Alignment und verfeinert dieses dann durch die Berechnung der Distanzen äquivalenter Cα-Atome. Er wird im *PDBeFold-Web-Server* implementiert. Als einziger hier beschriebener Algorithmus ermöglicht er schnelle multiple Strukturvergleiche über einen *Web-Service*.

Die meisten dieser Methoden führen einen *Template*-basierten Vergleich durch. Das heißt, dass der Ähnlichkeitswert, der einem Paar zugewiesen wird zum einen von der Reihenfolge der Eingabe abhängt und zum anderen von den anderen Proteinen abhängt, die Teil des multiplen Vergleichs sind.

Graphlets wurden zuerst von *Pržulj et al.* auf biologische Daten angewandt [9], [10]. Sie nutzten den *Graphlet*-Algorithmus, um Ähnlichkeiten von Protein-Protein-Interaktionsnetzwerken zu berechnen.

N. Shervashidze war die erste, die *Graphlets* zur Analyse von Proteinen anwandte [12]. Sie nutzte *Support Vector Machines* auf *Graphlet*-Vektoren, um für Proteingraphen zu entscheiden, ob diese Enzyme darstellen oder nicht.

Graphlets wurden auch von *Tatiana Bakirova* verwendet, um Proteinstrukturen zu analysieren [1]. Sie hat das Programm `graphletAnalyser` verfasst, das in dieser Arbeit weiterentwickelt wird.

1.3 Ziele

Ziel dieser Arbeit war zunächst die Erweiterung der Funktionalität von `graphletAnalyser`. Hierzu gehört zunächst eine funktionierende Datenbankanbindung, um die berechneten Daten abzuspeichern. Die Suche nach markierten *Graphlets* sollte so implementiert werden, dass sie auf Graphen mit beliebigen Markierungen angewandt werden kann. Deshalb wurde ein Algorithmus entwickelt und implementiert, der aus einem Alphabet von Knotenmarkierungen alle Worte berechnet, die markierte 2- und 3-*Graphlets* repräsentieren - der *Graphlet*-Worte-Algorithmus. Die Suche nach diesen markierten *Graphlets* in einem Graphen wurde ebenfalls implementiert.

In Fallstudien wird überprüft, ob und inwiefern sich *Graphlets* eignen, um die Ähnlichkeit von Proteinstrukturtopologien zu untersuchen. Dies wurde mit unterschiedlichen Metriken getestet. Die hierbei errechneten Ähnlichkeitswerte wurden mit den Ergebnissen des Strukturalignment-Programms *PDBeFold* [6] verglichen.

Weiterhin wurde mit dem PDBTop500-Datensatz [7] untersucht, welche strukturellen Merkmale von Proteinen mit *Graphlets* ermittelt werden.

1.4 Aufbau der Arbeit

Zunächst wird im Kapitel *Materialien und Methoden* die *Protein Topology Graph Library* (PTGL) [8] vorgestellt, deren Idee die Grundlage für diese Arbeit liefert.

Es folgt eine Kurzbeschreibung von PLCC, der *Software*, die die Graphen der PTGL erstellt und diese verwaltet, sowie eine Beschreibung des *Graphlet*-Algorithmus.

Weiterhin wird das Programm `graphletAnalyser` vorgestellt, welches den *Graphlet*-Algorithmus implementiert. Anschließend wird die erste Metrik vorgestellt, mit denen die erhaltenen *Graphlet*-Vektoren verglichen werden. Es folgt eine Beschreibung der verwendeten Datensätze und von *PDBeFold*, dessen Ergebnisse mit denen von `graphletAnalyser` verglichen werden.

Im Ergebnisteil wird zunächst der modifizierte Jaccard-Index präsentiert, der sich mit leichten Änderungen am Tanimoto-Koeffizienten orientiert. Dann folgen Beschreibungen des neuen *Graphlet*-Worte-Algorithmus und der in den Fallstudien erhaltenen Ergebnisse.

Im abschließenden Teil *Diskussion und Ausblick* wird versucht, die Frage zu klären, ob sich *Graphlets* für multiplen Proteinstrukturvergleich und ähnliche

Anwendungen eignen. Weiterhin wird untersucht, ob Modifikationen der *Graphlet*-Vektoren, oder der Metriken nötig sein könnten, um bessere Ergebnisse zu erhalten.

Kapitel 2

Materialien und Methoden

2.1 PTGL

Die Protein Topology Graph Library entstand aus einer Idee von *Patrick May* und *Ina Koch* ([8]). Ausgehend von der Tatsache, dass sich Proteinstrukturtopologien als räumliche Beziehungen von SSEs untereinander definieren lassen, verwendet die PTGL *Graphen*, um Proteinstrukturtopologien darzustellen. Hierbei stellen die Knoten des Graphen die SSEs eines Proteins dar. Sie werden dem jeweiligen SSE entsprechend markiert. Knoten, die α-Helices repräsentieren, werden mit einem H markiert, β-Faltblätter mit einem E. Weiterhin ermöglicht die PTGL die Darstellung von Liganden, ([11]) denen mit L markierte Knoten zugeordnet werden. Um die räumliche Nachbarschaft von Sekundärstrukturen und Liganden mit- und untereinander darstellen zu können, werden ungerichtete Kanten zwischen Knoten gezogen, wenn die entsprechenden Elemente benachbart sind. Jede Polypeptidkette eines Proteins wird dann als *Proteingraph* dargestellt. Die Zusammenhangskomponenten eines Proteingraphen werden als Faltungsgraphen bezeichnet, weil sie typischerweise eine unabhängige Faltungseinheit darstellen.

Durch diese abstrahierte Darstellung können zentrale Charakteristika eines Proteins wie Motive und Domänen einfach visualisiert werden.

2.2 PLCC

PLCC ist die *Software*, die die Daten der PTGL generiert und verwaltet. Sie wird von *Tim Schäfer* geschrieben und verwaltet.

Die Berechnung der Graphen der PTGL erfolgt unter Verwendung der entsprechenden PDB und DSSP-Dateien. Um den Graphen für eine Polypeptidkette zu berechnen, werden aus der DSSP-Datei die SSEs des Proteins ausgelesen. Für jedes Paar von SSEs wird die Anzahl der räumlichen Kontakte ihrer Residuen in der PDB-Datei berechnet. Wenn die Anzahl dieser Kontakte einen

gewissen Grenzwert überschreitet, wird angenommen, dass diese SSEs räumlich benachbart sind und die jeweiligen Knoten werden durch eine Kante verbunden. So wird für jede Polypeptidkette ein Graph erstellt.

Komplexgraphen werden ebenfalls in dieser Arbeit untersucht. Ihre Berechnung erfolgt analog zur Berechnung der Proteingraphen. Der Unterschied besteht darin, dass ein Komplexgraph mehrere Polypeptidketten beschreibt.

Aminosäuregraphen werden analog zu Proteingraphen und Komplexgraphen berechnet. Der Unterschied zu den anderen Graphformaten ist, dass keine SSEs betrachtet werden. Stattdessen repräsentiert jeder Knoten eine Aminosäure eines Proteins. Die Knoten werden entsprechend der chemischen Eigenschaften der Aminosäuren markiert. Knoten, die saure oder basische Residuen darstellen, werden mit einem c markiert. Ein p markiert Knoten für polare Residuen, die weder sauer noch basisch sind. Für unpolare Aminosäuren wird ein h verwendet. Auch Liganden können in Aminosäuregraphen dargestellt werden. Ihre Knoten werden durch ein ? markiert. Aminosäuregraphen können Proteinkomplexe und einzelne Polypeptidketten darstellen.

Weiterhin ermöglicht PLCC den Vergleich von *Graphlet*-Vektoren, die im folgenden Teil vorgestellt werden.

2.3 Der *Graphlet*-Algorithmus

Um diese Graphen vergleichen zu können, werden *Graphlets* verwendet. Diese sind im Gegensatz zu Graph-Isomorphismen in polynomieller Zeit berechenbar [12]. *Graphlets* sind kleine, induzierte Teilgraphen mit bis zu 5 Knoten. Die Abbildungen 5.1, 5.2 und 5.3 zeigen die *Graphlets* mit 3, 4 und 5 Knoten. Wir betrachten hierbei nur die zusammenhängenden *Graphlets*. Sie werden durch den Algorithmus gezählt und ihre jeweilige Anzahl wird in einen Vektor geschrieben, den wir als *Graphlet*-Vektor bezeichnen.

2.3.1 Beschreibung des Algorithmus

Um alle *Graphlets* der Größe $k \in \{3, 4, 5\}$ zu zählen, werden alle Euler-Wege der Länge $k - 1$ in dem gegebenen Graphen gesucht. Für jeden gefundenen Euler-Weg werden alle inzidenten Kanten aller Knoten des Weges überprüft. Wird hierbei ein *Graphlet* gefunden, wird der Zähler für das entsprechende *Graphlet* im *Graphlet*-Vektor um den Wert an der entsprechenden Stelle der unten stehenden Gewichtungsvektoren w_k erhöht.

Für *Graphlets* mit $k \geq 4$ existieren zusätzlich die sogenannten Stern-*Graphlets* (g_4 in 5.2, sowie g_{19},g_{20} und g_{21} in 5.3), die keinen solchen Euler-Weg enthalten.

Graphlet-Gewichtungsvektoren

$$w_2 := \left(\frac{1}{2}\right) \tag{2.1a}$$

$$w_3 := \left(\frac{1}{6}, \frac{1}{2}\right) \tag{2.1b}$$

$$w_4 := \left(\frac{1}{24}, \frac{1}{12}, \frac{1}{4}, 1, \frac{1}{8}, \frac{1}{2}\right) \tag{2.1c}$$

$$w_5 := \left(\frac{1}{120}, \frac{1}{72}, \frac{1}{48}, \frac{1}{36}, \frac{1}{28}, \frac{1}{20}, \frac{1}{14}, \frac{1}{10}, \frac{1}{12}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, \frac{1}{12}\frac{1}{12}, \frac{1}{4}\frac{1}{4}, \frac{1}{2}, 1, \frac{1}{2}, 1\right) \tag{2.1d}$$

$$\tag{2.1e}$$

Jede Stelle eines Gewichtungsvektors w_k ist mit einem *Graphlet* assoziiert. Die Nenner der Brüche in den Vektoren entsprechen den Anzahlen der Euler-Wege der Länge $k-1$ in den entsprechenden *Graphlets*. Wenn alle Euler-Wege eines *Graphlets* besucht wurden und der zugehörige induzierte Teilgraph ihm entspricht, wird sein Zähler im *Graphlet*-Vektor ganzzahlig und das *Graphlet* gilt als gefunden. Hierbei bilden die Stern-*Graphlets*, die keine Euler-Wege der Länge $k-1$ enthalten, natürlich die Ausnahme.

Da wir 29 verschiedene *Graphlets* betrachten, hat ein *Graphlet*-Vektor 29 Stellen. Für den Vergleich von Proteinstrukturen bedeutet dies, dass der Berechnugsaufwand in $O(1)$ liegt, da anstatt von Graphen, oder dreidimensionalen Körpern Vektoren im Vektorraum $\mathbb{R}^{29}$ verglichen werden.

TODO: Pseudocode in Anhang einfügen

2.4 `graphletAnalyser`

In seiner ursprünglichen Version wurde das Programm bereits 2013 von *Tatiana Bakirova* geschrieben. Im Sommer 2015 wurde es um einige Funktionen erweitert.

`graphletAnalyser` ist in C++ geschrieben. Zur internen Darstellung der Graphen wird die *Boost-Graph-Library* verwendet. Die Datenbankanbindug wird mittels der Bibliothek pqxx realisiert.

Als Input erhält das Programm eine oder mehrere GML-Dateien, die Graphen darstellen. Aus jeder GML-Datei wird mit der *Boost-Graph-Library* intern ein Graph erstellt, der für die Berechnungen verwendet wird.

Für den eingelesenen Graphen berechnet das Programm alle *Graphlets* der Größen 3,4 und 5. Die Implementierung entspricht der *Matlab*-Implementierung

von *N. Shervashidze* [12]. Zusätzlich berechnet es markierte *Graphlets* der Größen 2 und 3, wobei die Markierungen den SSE-Zuordnungen der PTGL entsprechen. Es gibt 35 verschiedene markierte *Graphlets* mit 2 oder 3 Knoten. Unter Berücksichtigung dieser Markierungen erhöht sich die Anzahl der Stellen der *Graphlet*-Vektoren auf 64. Die Knotengradverteilungen können ebenfalls berechnet werden.

Der Output des Programms umfasst zum einen die Ausgabe der *Graphlet*-Vektoren mit oder ohne markierte *Graphlets* in den Formaten csv, matlab und nova. Zusätzlich können die *Graphlet*-Vektoren in einer Datenbank, die zuvor mit PLCC erstellt wurde gespeichert werden. Die Knotengradverteilung kann zusammen mit anderen Daten eines Graphen als .csv-Datei ausgegeben werden.

2.5 Ähnlichkeitsmaß

Durch die Verwendung der *Graphlets* wird der direkte Vergleich von Graphen durch einen Vergleich von *Graphlet*-Vektoren ersetzt. Typischerweise wird eine eine Abstandsmessung durchgeführt, um Vektoren zu vergleichen. Hier verwenden wir eine Metrik von *Pržulj et al.*.

2.5.1 Relative *Graphlet*-Häufigkeiten-Distanz

Pržulj et al. haben *Graphlets* bereits auf Protein-Protein-Interaktionsnetzwerke ([9]) angewandt. Als Maß für die Ähnlichkeit von Netzwerken nutzen sie die Relative-*Graphlet*-Häufigkeiten-Distanz (RGF) . Diese Metrik berechnet den Abstand $D(G, H)$ zwischen zwei Graphen G und H als logarithmierte Differenz der normalisierten Anzahl der *Graphlets* in G und H. Sie ist folgendermaßen definiert:

Sei $N_i(G)$ die Anzahl der *Graphlets* von Typ $i \in 1, ..., 29$ und
$T(G) = \sum_{i=1}^{29} N_i(G)$ die Anzahl der *Graphlets* in G, beziehungsweise H.

Dann ist $D(G, H)$ für zwei Graphen G und H definert als:

$$D(G, H) := \sum_{i=1}^{29} |F_i(G) - F_i(H)| \tag{2.2a}$$

$$mit F_i(G) := -log(\frac{N_i(G)}{T(G)}) \tag{2.2b}$$

Diese Metrik lässt sich analog zur euklidischen Distanz auffassen, denn sie berechnet den Abstand für zwei Vektoren x, y als Differenz ihrer Einträge $x_i - y_i$. Sie verwendet die normalisierten *Graphlet*-Vektoren unter der Annahme, dass die Ähnlichkeit zweier Netzwerke sich aus der Ähnlichkeit lokaler Substrukturen ableiten lässt [9]. Somit können Netzwerke, die ähnliche Substrukturen habe,

sich aber in ihrer Größe stark unterschieden, immer noch als ähnlich erkannt werden. Zusätzlich soll durch diese Normalisierung ausgeschlossen werden, dass einzelne besonders häufige *Graphlets* in einem Graphen die Messung verzerren.

Da alle Einträge der Vektoren aus dem Intervall $[0, 1] \in \mathbb{R}$ stammen werden sie logarithmiert. Damit erhält man Abstände, die aus deutlich größeren Intervallen stammen. Die Ergebnisse können so leichter interpretiert werden und der Einfluss der von Rundungsfehlern verkleinert sich.

Weiterhin wurde gezeigt, ([9]) dass diese Metrik auch bei verrauschten Daten noch sehr gut funktioniert. Hierbei ist jedoch zu beachten, dass sie bisher vor allem für sehr große Netzwerke mit mehreren Tausend Knoten und Kanten verwendet wurde. Diese Größe kann von Aminosäuregraphen erreicht werden, wenn sie große Proteinkomplexe modellieren. Proteingraphen und Komplexgraphen sind aber deutlich kleiner.

2.6 *PDBeFold*

Im Rahmen der Fallstudien wird `graphletAnalyser` mit *PDBeFold* [6] verglichen. *PDBeFold* führt einen graphenbasierten Strukrutvergleich durch. Analog zur PTGL modellieren Knoten SSEs. Zusätzlich werden diese mit der Anzahl von Residuen der SSE markiert. Im Gegensatz zur PTGL verbindet PDBeFold jeden Knoten durch eine Kante mit jedem anderen Knoten. Die Kanten werden hierbei mit Vektoren markiert, in denen unter anderem die Distanzen und Winkel zwischen den SSEs eingetragen sind.

Die Proteine werden also als vollständige, ungerichtete Graphen modelliert, in denen jede Kante alle Informationen zur räumlichen Orientierung ihrer inzidenten Knoten enthält. Diese Graphen werden vorberechnet.

Zum Vergleich von Strukturen wird für diese Graphen zunächst ein Alignment durchgeführt, um eine Abschätzung der strukturellen Ähnlichkeit zu erhalten und die äquivalenten Residuen zu ermitteln. Im Anschluss führt PDBeFold ein Alignment der Cα-Atome durch und berechnet die Ähnlichkeit der Proteine als durchschnittlichen paarweisen Abstand äquivalenter Residuen. Als Abstandsmaß verwendet es die *Root-Mean-Square-Deviation*.

Da `graphletAnalyser` Vektoren berechnet, die verglichen werden, ist ein direkter Vergleich der Abstände aus beiden Programmen nicht möglich. Der Vergleich kann nur durch einen Vergleich relativer Abstände innerhalb der Datensätze erfolgen.

2.7 Datensätze

Im folgenden Teil sind die verwendeten Datensätze beschrieben. Es wurden zwei Fallstudien mit Abstandsmessungen durchgeführt, um diese zu testen.

Zusätzlich wurden *Graphlet*-Vektoren für einen Datensatz aus 500 nicht redundanten PDB-Einträgen berechnet. Für diesen Datensatz wurden keine

Abstandsmessungen durchgeführt, stattdessen wurden die *Graphlet*-Vektoren selbst analysiert.

2.7.1 Fallstudien - Datensatz 1

Dieser Datensatz enthält 15 verschiedene Proteine mit bekannten strukturellen Ähnlichkeiten. Die Proteine wurden so gewählt, dass ihre Einordnungen in strukturelle Klassen in den Datenbanken CATH und SCOPe äquivalent zueinander sind. Der Datensatz wurde also so zusammengestellt, dass bezüglich der strukturellen Einteilung der Proteine der größtmögliche Konsens besteht. Je 5 Proteine dieses Datensatzes befinden sich in der gleichen Klasse. Von diesen 5 besitzen je 4 die gleiche Topologie und von diesen entstammen wiederum 3 der gleichen homologen Superfamilie. Von diesen ist ein Paar direkt homolog mit einer Sequenzidentität von mehr als 95%. Das dritte Protein ist zu den anderen beiden entfernt homolog mit einer Sequenzidentität von weniger als 30%.

Es sind Proteine der Klassen α, β und α/β vertreten. Proteine mit wenigen SSEs wurden nicht betrachtet, da für diese die Proteingraphen und die Komplexgraphen in den meisten Fällen keine Kanten besitzen. Dadurch können keine *Graphlets* in diesen Graphen gefunden werden.

Weiterhin wurde darauf geachtet, dass jedes Protein dieses Datensatzes aus genau einer Polypeptidkette mit genau einer Domäne besteht. Dies hat mehrere Gründe.

Zum ersten ermöglicht diese Auswahl die beste Vergleichbarkeit der Messungen mit Proteingraphen, Komplexgraphen und Aminosäuregraphen, da Proteingraphen im Gegensatz zu Komplexgraphen und Aminsäuregraphen nur eine Polypeptidkette darstellen können. Durch die Beschränkung auf Proteine mit genau einer Polypeptidkette wird ein direkter Vergleich durch die Korrelation der paarweisen Distanzen in den unterschiedlichen Formaten möglich.

Ein weiterer Grund für die Beschränkung auf solche Proteine ist, dass man ein "klareres Signal" erhält. Die Struktur von Proteinen wird hauptsächlich anhand ihrer Domänen beschrieben. Wenn *Graphlets* sich wirklich für den Vergleich von Topologien eignen, dann müssten gleiche Domänen zu gleichen, oder zumindest ähnlichen *Graphlet*-Vektoren führen. Da diese Vektoren globale Struktureigenschaften des Proteins beschreiben erhielte man für ein Multidomänenprotein mit unterschiedlichen Domänen einen Vektor, der die Struktureigenschaften beider Domänen "vermischt" und das Signal verrauscht.

In der folgenden Tabelle **??** sind die PDB-IDs der Proteine dieses Datensatzes zusammen mit ihrer CATH-ID und der Klassifizierung aus der PDB Datei aufgeführt.

Für diesen Datensatz wurden alle *Graphlet*-Vektoren berechnet. Die paarweisen Distanzen dieser Vektoren zueinander wurden mit der RGF und dem modifizierten Jaccard-Index berechnet. Für die berechneten Distanzmatrizen wurden die Korrelationen der Distanzen in den verschiedenen Formaten berechnet und sie wurden mit den von PDBeFold berechneten Distanzen verglichen.

PDB-ID	CATH-ID	Klassifizierung
1qpu	1.20.120.10	Elektronentransport
1qq3	1.20.120.10	Elektronentransport
1cgn	1.20.120.10	Elektronentransport
1he9	1.20.120.260	Toxin (Exoenzym)
3gf9	1.20.900.10	Endocytose
1exs	2.40.128.20	Lipid bindendes Protein
1ngl	2.40.128.20	Transport Protein
1qqs	2.40.128.20	Zucker bindendes Protein
3slo	2.40.128.130	Protein Transport
1wjx	2.40.280.10	RNA-bindendes Protein
5chy	3.40.50.2300	Signaltransduktion
2id9	3.40.50.2300	Signal Protein
3i42	3.40.50.2300	unbekannte Funktion
1d4o	3.40.50.1220	Oxidoreductase
2w0i	3.40.20.10	Transferase

Tabelle 2.1: PDB-Einträge der ersten Fallstudie. Die PDB-IDs sind mit dem CATH-Code der zugehörigen homologen Superfamilie und der Klassifizierung aus dem *Header* der PDB-Datei in der entsprechenden Zeile eingetragen

2.7.2 Der PDBTop500-Datensatz

Der PDBTop500-Datensatz wurde von *Lovell et al.* vorgestellt ([7]). Diese nutzten ihn zur Validierung von PDB-Strukturen mit Hilfe der Cα-Geometrie. Der Datensatz enthält 500 nicht redundante PDB-Einträge mit einer Auflösung von 1,8 Å
oder besser.

Er wurde ausgewählt, weil die *Graphlet*-Vektoren für ihn schnell berechenbar waren. Im Gegensatz zu anderen Datensätzen mit nicht redundanten PDB-Einträgen wie dem FATCAT- oder dem ASTRAL-Datensatz enthält dieser keine riesigen Proteinkomplexe, deren *Graphlets* auf durchschnittlichen Computern nicht in weniger als 24 Stunden berechnbar sind. Er ist aber immer noch groß genug, um Strukturvergleichsmethoden validieren zu können. *Zhang* und *Skolnick* verwendeten zur Validierung von *TM-align* einen Datensatz aus 200 PDB-Einträgen ([15]).

Da ein Vergleich der paarweisen Distanzen auf einem solchen Datensatz zu aufwändig war, wurde dieser Datensatz für eine statistische Analyse der *Graphlet*-Vektoren genutzt.

Für die relativen Häufigkeiten der einzelnen *Graphlets* wurden Minima, Maxima, Varianzen und Durchschnittswerte berechnet. Hierbei war das Ziel festzustellen, ob wirklich alle *Graphlets* für Strukturvergleiche notwendig sind.

Weiterhin wurde ein Datensatz aus Aldolasen zusammengestellt, um festzustellen, ob sich die Verteilungen der *Graphlet*-Häufigkeiten, zwischen dem großen Datensatz und den Aldolasen unterscheiden. Wenn der Unterschied der

Verteilungen zwischen diesen beiden Datensätzen so groß ist, dass eine klare Unterscheidung möglich ist, dann sollte auch eine maschinelle Klassifizierung der *Graphlets* für Aldolasen möglich sein. Aldolasen wurden gewählt, weil diese Superfamilie strukturell besonders divers ist [2].

Wenn die *Graphlet*-Verteilungen der Aldolasen und des PDBTop500-Datensatzes klar unterscheidbar sind, dann sollten sie auch für Superfamilien, die strukturell einheitlicher sind unterscheidbar sein.

Die PDB-IDs der untersuchten Aldolasen lauten: 1lbm, 1lor, 1mxs, 1nsj, 1v5x, 1vqt, 1wau, 3ceu und 7tim.

Kapitel 3

Ergebnisse

3.1 Der modifizierte Jaccard-Index

Der Jaccard-Index ist im eigentlichen Sinne ein Maß, um die Ähnlichkeit von gleichmächtigen Mengen zu bewerten. Für zwei Mengen A, B berechnet sich der Jaccard-Index $D_{Jac}(A, B)$ folgendermaßen:

$$D_{Jac}(A, B) := \frac{\sum_{x \ inA \wedge x \in B} 1}{\sum_{x \in A \vee x \in B} 1}$$

Dementsprechend sind zwei Mengen A, B gleich, wenn gilt $D_{Jac} = 1$ und disjunkt, wenn gilt $D_{Jac} = 0$. Mit ihm wird die relative Anzahl der Elemente beider Mengen berechnet. Um dieses Maß in sinnvoller Weise auf *Graphlet*-Vektoren zu übertragen wurde ein zusätzlicher Faktor $k \in \mathbb{R}$ mit $k \in [0, 1]$ eingeführt, der als Präzisionsfaktor zu verstehen ist. Die Defintion des modifizierten Jaccard-Index $D_{Jac-m}(v, w)$ für zwei Vektoren v, w lautet also:

$$D_{Jac-m}(v, w) := \frac{\sum_{i=1}^{n} x_i}{\sum_{x \in A \vee x \in B} 1}$$

Hierbei gilt:

$$x_i = \begin{cases} 1 & \text{if} \quad v_i >= w_i \times k \wedge w_i >= v_i \times k \\ 0 & \text{else} \end{cases}$$

In dieser modifizierten Variante werden zwei Vektoren v, w als gleich angesehen, wenn sich v_i, w_i für alle i höchstens um den Faktor k unterschieden. Dies steht im Gegensatz zur RGF, die - analog zum euklidischen Abstand - die Abstände zwischen zwei *Graphlet*-Vektoren misst.

3.2 Der *Graphlet*-Worte-Algorithmus

In der letzten Version von `graphletAnalyser` war es bereits möglich markierte *Graphlets* mit 2 und 3 Knoten in Proteingraphen zu zählen. Diese Funktionalität wurde im Rahmen dieser Arbeit verallgemeinert, so dass der Nutzer beliebige Alphabete angeben kann. Der Algorithmus erhält das Alphabet $\sum = \{\sigma_i : i \in \mathbb{N}\}$ der Knotenmarkierungen. Aus diesem Alphabet berechnet er Worte w, die zur Repräsentation der markierten *Graphlets* genutzt werden. Hierbei können verschiedene Worte das gleiche *Graphlet* repräsentieren. Im Falle von 2-*Graphlets* repräsentieren die zwei Worte (σ_i, σ_j) und (σ_j, σ_i) das gleiche markierte *Graphlet* mit den Knotenmarkierungen σ_i, σ_j. Worte, die das gleiche *Graphlet* repräsentieren, werden im Folgenden als *äquivalente Graphlet-Worte* bezeichnet.

Die Berechnung der äquivalenten *Graphlet*-Worte der Länge 2 ist trivial. Aus dem Alphabet $\sum$ werden alle Worte $w = (\sigma_i, \sigma_j)$ berechnet, wobei Spiegelungen nicht mit ausgegeben werden, da zwei Worte (σ_j, σ_i) und (σ_j, σ_i) äquivalente *Graphlet*-Worte sind.

Die Berechnung aller äquivalenten *Graphlet*-Worte der Länge 3 ist komplizierter, da sie für zwei verschiedene *Graphlets* berechnet werden müssen. Für das *Graphlet* g_1 sind alle Worte äquivalent zueinander, die zyklische Vertauschungen voneinander sind. Für das *Graphlet* g_2 sind Worte äquivalent zueinander, die Spiegelungen voneinander sind (siehe Abbildung 5.1).

Pseudocode Platzhalter
Pseudocode Platzhalter
Pseudocode Platzhalter
Pseudocode Platzhalter
Pseudocode Platzhalter
Pseudocode Platzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter
Pseudocode PLatzhalter

Der Algorithmus besteht aus 3 *for*-Schleifen, die über das Alphabet iterieren. In der äußersten Schleife werden Worte hinzugefügt, in denen alle Buchstaben gleich sind. In der zweiten Schleife wird jeweils der nächste Buchstabe des Alphabets betrachtet. Für jedes Paar a, b von Buchstaben über einem Alphabet $\sum$ sind die Mengen der äquivalenten Worte für das *Graphlet* g_1 $P_{3-Kreis} = \{\}$ und $P_{3-Weg} = \{aaa, aba, aab, abb\}$ für das *Graphlet* g_2

TODO: Beschreibung der Mengen, Beweis

Für das Alphabet der SSE- und Komplexgraphen $\sum_{SSE} := \{H, E, L\}$ und das Alphabet der Aminosäuregraphen $\sum_{AA} := \{h, p, c, ?\}$ gibt der oben beschriebene Algorithmus die folgenden Listen aus:

$$p_2 := (HH, HE, HL, EE, EL, LL) \tag{3.1a}$$

$$p_{3-Weg} := (HHH, HEH, HHE, HEE, EHE, \tag{3.1b}$$

$$HEL, LHE, ELH, HLH, HHL, \tag{3.1c}$$

$$HLL, LHL, EEE, ELE, EEL, \tag{3.1d}$$

$$ELL, LEL, LLL) \tag{3.1e}$$

$$p_{3-Kreis} := (HHH, HEH, HEE, HEL, LEH, HLH, HLL, EEE, ELE, ELL, LLL) \tag{3.1f}$$

(3.1g)

Die Vektoren a_2, a_{3-Weg} und $a_{3-Kreis}$ beschreiben die Worte für *Graphlets* in AA-Graphen

$$a_2 := (hh, hp, hc, h?, pp, pc, p?, cc, c?, ??) \tag{3.2a}$$

$$a_{3-Weg} := (hhh, hph, hhp, hpp, php, hpc, chp, pch, hp?, ?hp, \tag{3.2b}$$

$$p?h, hch, hhc, hcc, chc, hc?, ?hc, c?h, h?h, hh?, \tag{3.2c}$$

$$h??, ?h?, ppp, pcp, ppc, pcc, cpc, pc?, ?pc, c?p, \tag{3.2d}$$

$$p?p, pp?, p??, ?p?, ccc, c?c, cc?, c??, ?c?, ???) \tag{3.2e}$$

$$a_{3-Kreis} := (hhh, hph, hpp, hpc, cph, hp?, ?ph, hch, hcc, hc?, \tag{3.2f}$$

$$?ch, h?h, h??, ppp, pcp, pcc, pc?, ?cp, p?p, p??, \tag{3.2g}$$

$$ccc, c?c, c??, ???) \tag{3.2h}$$

Um die Korrektheit des Algorithmus zu überprüfen wurde die Ausgabe für Alphabete mit 3 und 4 Buchstaben händisch überprüft.

3.3 Erweiterung von `graphletAnalyser`

Das Einlesen von Komplexgraphen und Aminosäuregraphen ist implementiert worden. Die entsprechenden Alphabete sind im Programmcode vordefiniert und können vom Nutzer über Parameter ausgewählt werden oder in der Konfigurationsdatei festgelegt werden.

Nutzerdefinierte Knotenmarkierungen können nun in der Konfigurationsdatei angegeben werden. Der Nutzer kann ein Alphabet von Knotenmarkierungen und ein *Label*, unter dem diese Knotenmarkierungen in den GML-Dateien gespeichert sind, angeben. Für dieses Alphabet werden alle äquivalenten *Graphlet*-Worte durch den *Graphlet*-Worte-Algorithmus berechnet. Diese werden dann bei der Berechnung der markierten *Graphlets* im Graphen unter dem vorgegebenen *Label* gesucht und gezählt. Es können also beliebige Alphabete und *Labels* angegeben werden, so lange die Markierungen der Knoten nicht mehr als einen Buchstaben enthalten.

Die Datenbankanbindung wurde um Funktionen zum Speichern von Aminosäuregraphen und Komplexgraphen erweitert. Das Speichern von Vektoren markierter *Graphlets* wurde implementiert.

Wenn die Option `--useDatabase` ausgewählt wird, prüft das Programm, ob der entsprechende Graph bereits in der Datenbank vorhanden ist. Falls der Graph gefunden wurde, wird der *Graphlet*-Vektor für den entsprechenden Graphen in die Datenbank eingetragen.

3.4 Fallstudien - Datensatz 1

Die paarweisen RGF-Distanzen der Proteine befinden sich in den Tabellen **??** **??** und **??**. Die paarweisen Jaccard-Indizes befinden sich in den Tabellen **??**, **??** und **??**. Hierbei wurden die Zellen, die die 4 besten Bewertungen für das Protein der entsprechenden Zeile enthalten grün eingefärbt. Hierbei gilt, dass das Grün umso dunkler ist, je stärker die Ähnlichkeit bewertet wird.

Der Vergleich mittels RGF zeigt für die Aminosäuregraphen die stärksten Ähnlichkeiten immer innerhalb der entsprechenden CATH-Klassen. Sowohl die Proteine aus der *mainly-alpha*-Klasse, als auch die Proteine der *mainly-beta*-Klasse haben die besten Ähnlichkeitswerte mit Proteinen der gleichen Klasse. Innerhalb der Klasse der *alpha-beta*-Proteine, gibt es mit 2id9 und 1d4o zwei Proteine, denen eine größere Ähnlichkeit zu Proteinen der *mainly-alpha*-Klasse attestiert wurde. Bei 1d4o fällt auf, dass der niedrigste Wert mit 7,091 deutlich höher ist, als die besten Werte aller anderen Proteine. 2id9 hat laut der RGF-Distanz die größte Ähnlichkeit zu 1he9. Bis auf diese beiden Außnahmen lässt sich jedoch eine starke Korrelation mit den CATH-Klassen erkennen. Alle anderen Proteine haben mindestens die zwei kleinsten zwei RGF-Distanzen zu Vertretern aus der selben CATH-Klasse.

Dies gilt für die Aminosäuregraphen. Die RGF-Distanzen der Proteingraphen zueinander zeigen ein weniger klares Bild. Für die *mainly-alpha*-Klasse und die der *mainly-beta*-Klasse befinden sich die Proteine mit den kürzesten Distanzen immer noch in der selben Klasse. Dies lässt sich für die Proteine der *alpha-beta*-Klasse aber nicht mehr behaupten. Hier haben 2ID9, 3I42 und 2w0I die kürzesten Distanzen zu Proteinen anderer Klassen.

Bei den Komplexgraphen ist die Korrelation zwischen der RGF-Distanz und der Zugehörigkeit zur CATH-Klasse noch geringer. Die Tabelle **??** zeigt nur für die Proteine 1QQ3, 1HE9, 1EXS und 1QQS die kürzeste Distanz zu einem Vertreter der gleichen Klasse. Es fällt jedoch auf, dass besonders häufig die Proteine der *alpha-beta*-Klasse 5CHY, 2ID9 und 3I42 als ähnlich zu anderen bewertet werden.

Der Vergleich der Jaccard-Indizes zeigt ein ähnliches Bild, wie der Vergleich der RGF-Distanzen. Bei den Aminosäuregraphen zeigt sich, dass innerhalb der *mainly-alpha*-Klasse wieder die paarweisen Ähnlichkeiten der *mainly-alpha*-Proteine am größten sind. Dies gilt bis auf eine Ausnahme auch für die *mainly-beta*-Proteine. Das Protein mit der PDB-ID 1NGL wird als strukturell ähnlichstes Protein zu 2W0I bewertet. In der Klasse der *alpha-beta*-Proteine gibt es mit 2ID9 und 1D4O wieder zwei Ausreißer, die die größten paarweisen Ähnlichkeiten nicht zu Vertretern der eigenen Klasse haben. Für 2ID9 wird 1HE9 als ähnlichstes Protein angegeben und 1D4O wird 1QQS zugeordnet.

Die paarweisen Tanimoto-Koeffizienten der Proteingraphen zeigen - wie schon bei den RGF-Distanzen - eine geringere Korrelation mit der Zugehörigkeit zu den CATH-Klassen, als die Koeffizienten der Aminosäuregraphen. Es haben zwar wieder mindestens 3 Vertreter jeder Klasse ihren nächsten Nachbarn in der gleichen Klasse, aber es gibt auch einige Proteine, die ihren nächsten Nachbarn außerhalb der eigenen Klasse haben. Hierzu gehören 1D4O, 2W0I (beide *alpha-beta*) und 3GF9 (*mainly-alpha*). Es fällt auf, dass wieder die Proteine mit den PDB-DIs 2ID9 und 3I42 besonders häufig als strukturell ähnlich zu vielen anderen Proteinen bewertet werden.

Für die Komplexgraphen zeigt die Tabelle wieder eine hohe Ähnlichkeite unter den ersten 3 Proteinen 1QPU, 1QQ3 und 1CGN. Auch innerhalb der Klasse *alpha-beta* sind 3 Proteine mit der höchsten paarweisen Ähnlichkeit bewertet worden. Die geringe Anzahl von stark bewerteten Ähnlichkeiten innerhalb der *mainly-beta*-Klasse ist sehr auffällig. 1QQS und 3SLO sind das einzige Paar mit *beta*-Topologie, dessen Ähnlichkeit als groß bewertet wurde.

3.5 PDBTop500-Datensatz

Für den PDBTop500-Datensatz wurden *Graphlet*-Vektoren der drei verschiedenen Graphenformate berechnet. Diese Vektoren wurden für jedes Format in einer .csv-Datei zusammengefasst und für jede dieser Dateien wurden *Boxplots* erstellt.

In diese Boxplots wurden zusätzlich die relativen Häufigkeiten der bereits beschriebenen Aldolasen eingetragen. Hiermit kann festgestellt werden, ob die Superfamilie der Aldolasen anhand ihrer *Graphlets* von anderen Superfamilien unterscheidbar ist. Der Boxplot **??** stellt diese Daten für die Aminosäuregraphen dar. Da die relative Häufigkeiten der *Graphlets* in den Aminosäuren deutlich kleiner sind, als bei Proteingraphen und Komplexgraphen, weil es mehr *Graphlet*-Worte gibt, wurde die Skala der y-Achse logarithmiert.

Unter den *Graphlets* ohne Markierungen in Abbildung **??** gibt es nur zwei bei denen sich die relativen Häufigkeiten der Aldolasen nicht mit denen des Datensatzes überlappen. Dies sind die *Graphlets* g_8 und g_{15} in 5.3, die in den Aldolasen nicht gefunden werden.

Unter den markierten *Graphlets* gibt es deutlich mehr Unterschiede. Sieben verschiedene *Graphlet*-Verteilungen überlappen sich nicht. Die ersten drei hiervon entsprechen den *Graphlet*-Worten h?, c? und ??. Diese stehen für Kanten zwischen unpolaren Residuen und Liganden, Kanten zwischen sauren oder basischen Residuen und Liganden und Kanten zwischen Liganden,

Die nächsten drei Worte für die keine Überlappungen gefunden wurden sind die Worte hhh, hph und hhp. Diese repräsentieren Wege der Länge 2 zwischen drei unpolaren Residuen, zwei unpolaren Residuen mit einem polaren Residuum als inneren Knoten und zwei unpolaren Residuen mit einer polaren Aminosäure als Endknoten.

Bei den Proteingraphen in Abbildung **??** sind die *Graphlet*-Verteilungen ein wenig stärker voneinander getrennt. Dies gilt für das 4-*Graphlet* g_6, das 5-*Graphlet* g_13. Unter den markierten *Graphlets* ist die Trennung für die am stärksten, die durch die Worte HL und HLL markiert werden. Ersteres repräsentiert den Kontakt einer α-Helix mit einem Liganden. Letzteres markiert einen Kreis, in dem zwei Liganden mit einer Helix in Kontakt stehen.

Für die Komplexgraphen in Abbildung **??** finden wir die stärksten Unterschiede zwischen den Verteilungen der Aldolasen und denen des Referenzdatensatzes. Für die unmarkierten *Graphlets* g_1 und g_2 in 5.1, g_2, g_3 und g_6 in 5.2 sowie g_6, g_{10}, g_{12}, g_{13}, $g16$, g_{18}, g_{20} und g_{21} in 5.3. Die stärksten Abweichungen der Verteilungen finden wir bei den markierten *Graphlets* für jene, die durch die Worte HL, LLL, und EEE repräsentiert sind. Für das Wort LL finden wir diese Abweichungen sowohl für das markierte *Graphlet*, das g_1 in 5.1 entspricht, als auch für jenes, das g_2 repräsentiert. Damit sind die Unterschiede der *Graphlet*-Verteilungen zwischen dem Referenzdatensatz und den Aldolasen in den Komplexgraphen am größten.

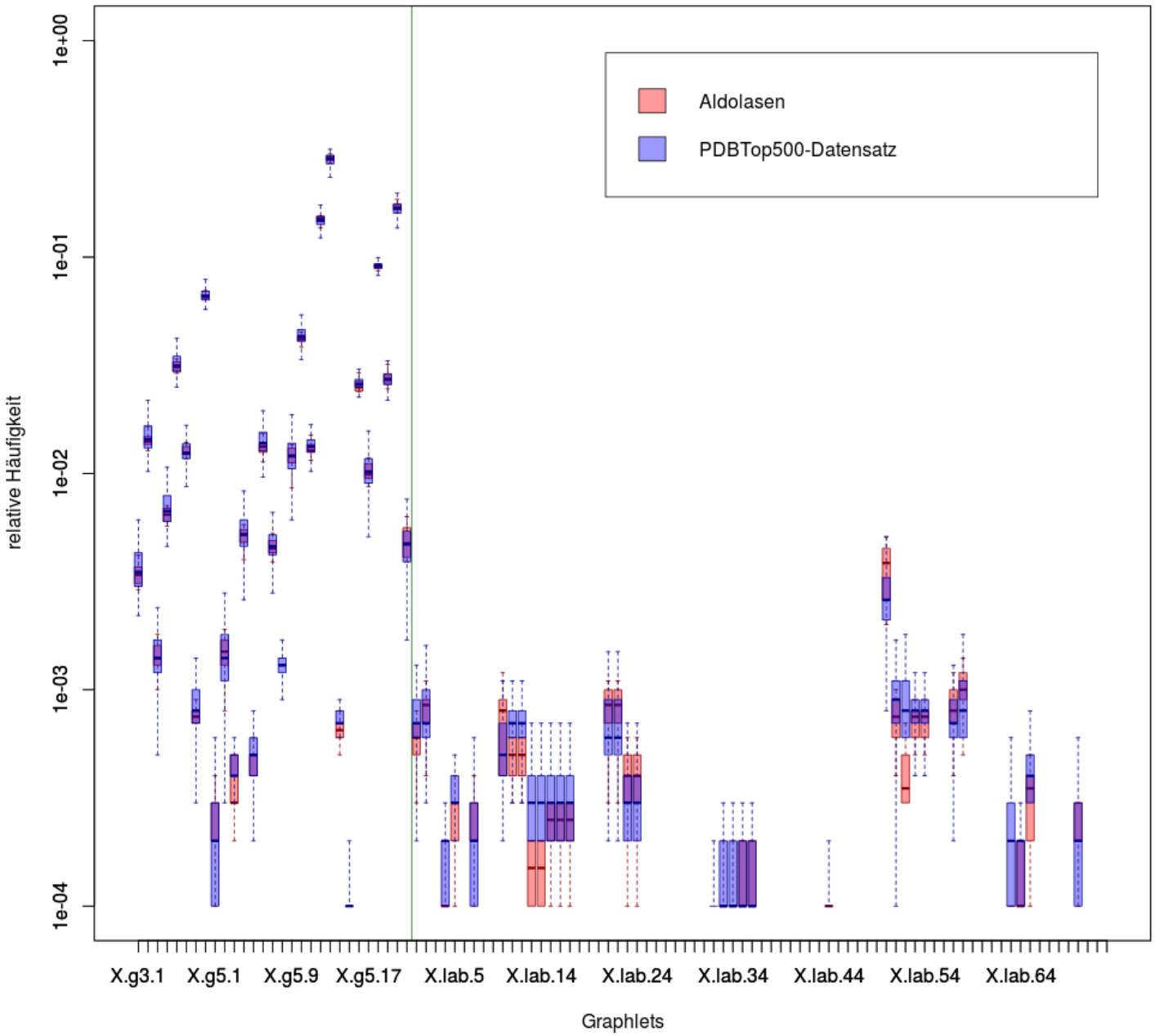


Abbildung 3.1: Verteilung der relativen Häufigkeiten der *Graphlets* in Aminosäuregraphen. Jeder Punkt auf der x-Achse entspricht einem *Graphlet*. Die y-Achse ist logarithmiert und beschreibt die relative Häufigkeiten. Die Häufigkeiten der Proteine des PDBTop500-Datensatzes sind in blau, die der Aldolasen in rot angegeben. Links von der vertikalen Linie sind *Graphlets* ohne Markierungen, rechts davon sind *Graphlets* mit Markierungen. Ausreißer sind nicht eingetragen.

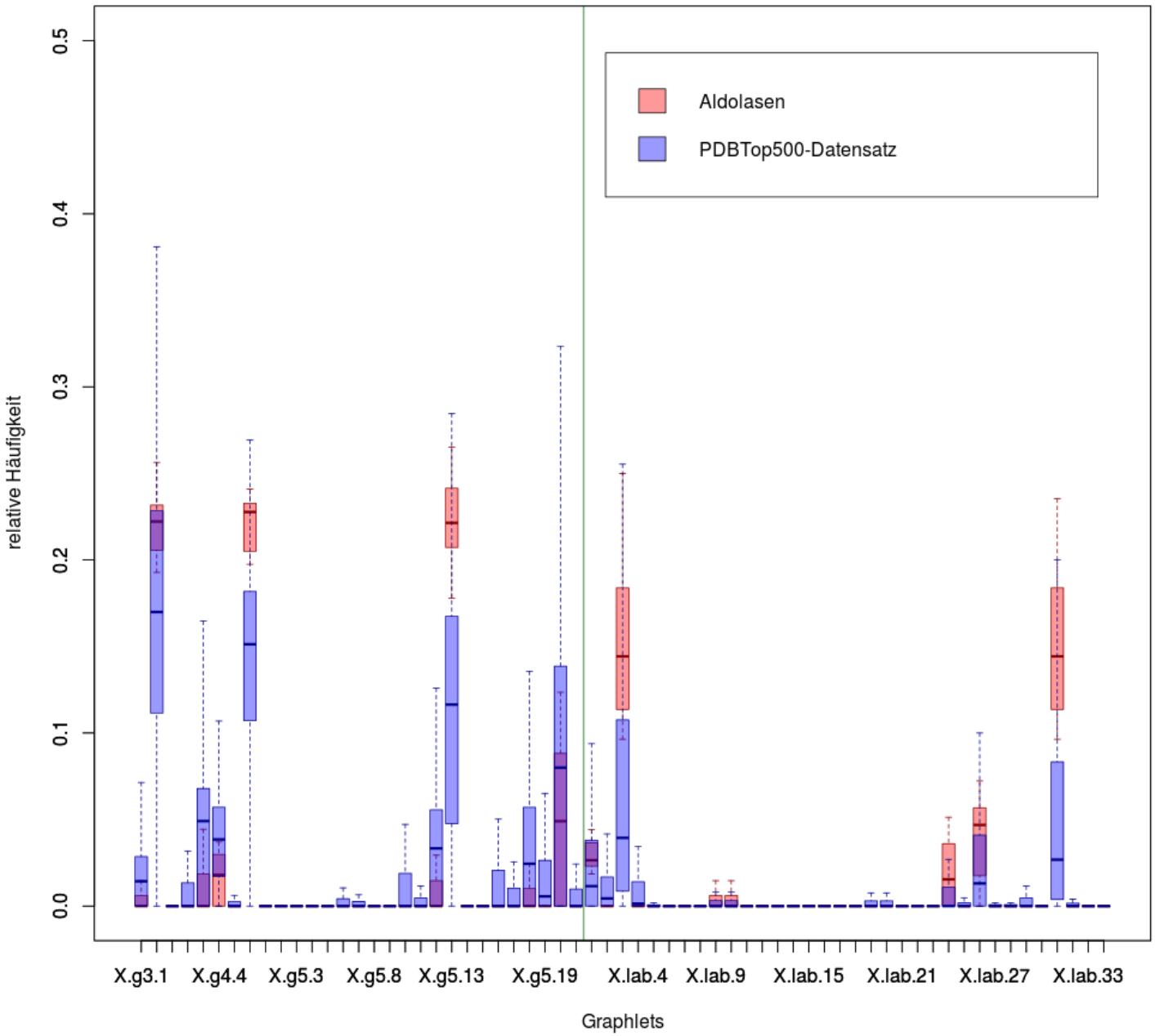


Abbildung 3.2: Verteilung der relative Häufigkeiten der *Graphlets* für Proteingraphen. Auch hier entspricht jeder Punkt auf der x-Achse einem *Graphlet* und die y-Achse zeigt die relativen Häufigkeiten an. Wieder trennt die dunkelgrüne Linie markierte und nicht markierte *Graphlets*. Solche ohne Markierungen sind links, mit Markierungen Versehene befinden sich rechts.

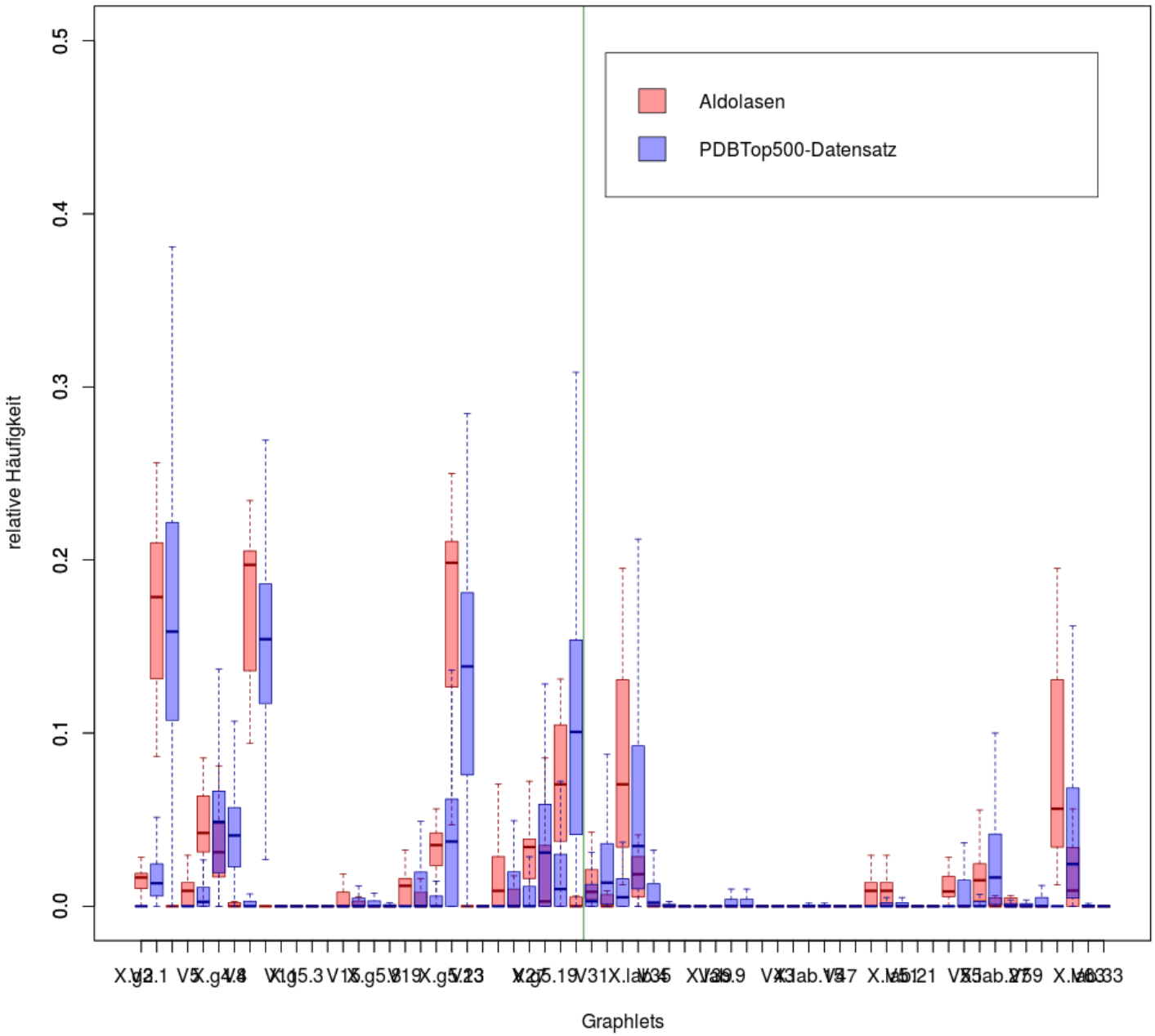


Abbildung 3.3: Verteilung der relative Häufigkeiten der *Graphlets* für Komplexgraphen. Auch hier entspricht jeder Punkt auf der x-Achse einem *Graphlet* und die y-Achse zeigt die relativen Häufigkeiten an. Wieder trennt die dunkelgrüne Linie markierte und nicht markierte *Graphlets*. Solche ohne Markierungen sind links, mit Markierungen versehene befinden sich rechts.

Kapitel 4

Diskussion und Ausblick

4.1 Diskussion

Die folgende Diskussion der Fallstudien widmet sich vor allem der Frage, wieso die Ergebnisse der Ähnlichkeitsvergleiche, sich so stark zwischen den jeweiligen Graphendarstellungen unterscheiden. Des weiteren wird der Zusammenhang zwischen dem Jaccard-Index und der RGF untersucht.

4.1.1 Datensatz 1

Vergleich der Graphformate

Wie schon im Ergebnisteil dargestellt, zeigen die Vergleiche der Aminosäuregraphen den höhsten Konsens mit der Einteilung der Strukturen durch CATH und SCOPe. Eine mögliche Erklärung hierfür ist die Gestalt der Graphen. Bisher wurden *Graphlets* zur Analyse von zusammenhängenden Graphen verwendet ([12], [10]). Proteingraphen und Komplexgraphen sind jedoch nicht immer zusammenhängend. es kommt häufig vor, dass einzelne Knoten keine Verbindungen zum Rest des Graphen aufweisen. Das unten stehende Bild zeigt ein Beispiel.

Dadurch, dass dies in den zusammenhängenden *Graphlets* nicht berücksichtigt werden kann, geht Information verloren. Unabhängig von der Wahl des Ähnlichkeitsmaß würde dieser Graph mit einem anderen Graphen, dem die beiden Helix-Knoten mit einem Grad von 0 fehlen, als gleich bewertet werden, obwohl dieser zwei SSEs weniger aufwiese. Diese SSEs können jedoch biologisch von zentraler Bedeutung sein.

Im Gegensatz hierzu sind die Aminosäuregraphen dieser Fallstudie zusammenhängend. Dies erklärt die höhere Genauigkeit.

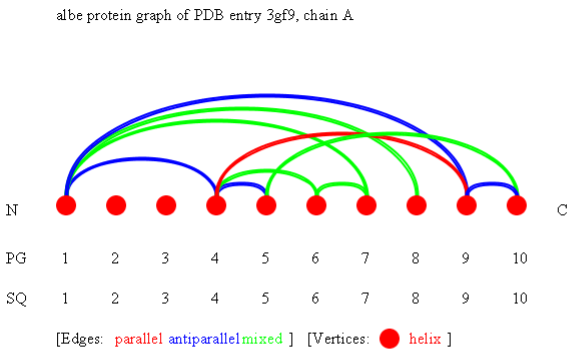


Abbildung 4.1: Proteingraph von 3GF9 - Datensatz 1

Datensatz	Korrelation
Aminosäuregraphen	0.6052
Proteingraphen	0.3102
Komplexgraphen	0.1256

Tabelle 4.1: Korrelationen der RGF mit den Jaccard-Indizes für die verschiedenen Graphformate

Vergleich der Distanzmaße

Wirft man einen Blick in die Tabellen, sieht es zunächst so aus, als würden sich Jaccard-Index und RGF ähnlich gut eignen, um die Ähnlichkeit der *Graphlet*-Vektoren zu bewerten. Die RGF stellt eine *Distanz* zwischen zwei Vektoren dar. Dementsprechend steht ein RGF-Wert von 0 für die Gleichheit zweier Vektoren, je höher der Wert ist, desto höher ist der Abstand zwischen den beiden Vektoren. Der modifizierte Jaccard-Index, der hier Verwendung findet, zählt Elemente, die sich um höchstens einen Faktor k unterscheiden. Ein RGF-Wert von 1 bedeutete, dass alle Elemente beider Vektoren sich höchstens um den Faktor k unterscheiden. Ein Wert von 0 bedeutet, dass alle Elemente sich um mehr als den Faktor k unterscheiden. Dementsprechend würde man erwarten, dass die *Pearson*-Korrelation von RGF und Jaccard-Index negativ ist. Die Tabelle 4.1 zeigt jedoch, dass diese für die Messungen der Fallstudie positiv ausfällt.

Eine mögliche Erklärung hierfür ist, dass die *Graphlet*-Häufigkeiten bei der RGF logarithmiert werden, dadurch werden die größten Werte zu den kleinsten

PDB	1cgn	2id9	3i42	2w0i	1qq3
1cgn		1.357	1.497	3.554	1.757
2id9	1.357			2.982	1.651
3i42	1.497	0.405		3.066	1.866
2w0i	3.554	2.982	3.066		3.109
1qq3	1.757	1.651	1.866	3.109	

Tabelle 4.2: Alignmenttabelle aus *PDBeFold*. In den Zellen sind die *Root-mean-square deviations* für die einzelnen Paare eingetragen.

und andersrum. So tragen Werte in ähnlichen Größenordnungen immer zu einer kürzeren Distanz bei, obwohl sich die tatsächlichen absoluten Häufigkeiten stark unterscheiden. Damit werden selten auftretende *Graphlets* stärker für die Distanz gewichtet als solche, die häufig auftreten.

Beim Betrachten der relative *Graphlet*-Häufigkeiten haben wir gesehen, dass für die *Graphlets*, die besonders häufig sind, auch die Varianz dieser Häufigkeiten besonders groß ist, Während gleichzeitig einige *Graphlets* fast nie auftauchen.

All diese *Graphlets*, die nieauftauchen sorgen dafür, dass der Jaccard-Index wahrscheinlic nie für ein Paar aus zwei beliebigen Proteinen den Wert Null annimmt. Damit wird bereits eine gewisse Mindestähnlichkeit vorausgesetzt. Gleichzeitig ist kann der Jaccard-Index aber *Graphlets*, die selten auftreten nicht stärker gewichten, denn wenn ein seltenes *Graphlet* in einem Graphen einmal auftaucht und in einem anderen zweimal ist dieser Unterschied zu groß, um in eine positive Bewertung einzufließen, obwohl dies eine starke Ähnlichkeit bedeuten könnte.

Vergleich mit *PDBeFold* - die Ausreißer

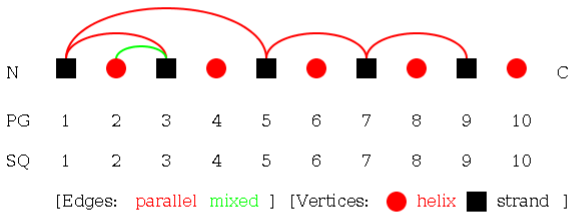
Da *PDBeFold* keine 15 Proteine auf einmal alignieren kann und ein *Template*-basiertes Alignment durchführt wurden für den Vergleich mit den *Graphlet*-Abständen eine Teilmenge der Proteine ausgewählt.

Beim Vergleich der Komplexgraphen in Tabelle **??** fällt auf, dass die Proteine 2id9, 3i42 und 2w0i die die geringsten Distanzen zu 1cgn haben, obwohl 1qq3 sein Ähnlichstes sein sollte. Ein Alignment dieser Proteine mit *PDBeFold* liefert die Tabelle **??** :

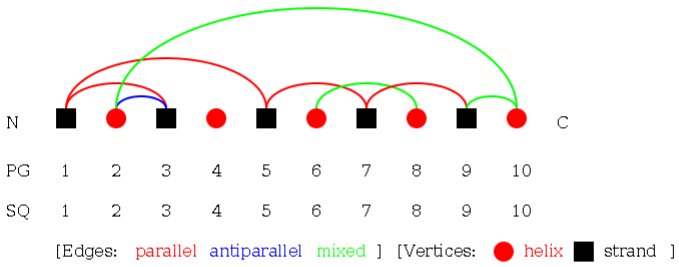
Sie zeigt, dass *PDBeFold* die Ähnlichkeit des α-Proteins 1cgn zu den α/β-Proteinen 2id9 und 3i42 höher bewertet, als die Ähnlichkeit zu dem α-Protein 1qq3, obwohl diese in der gleichen Superfamilie einzuordnen sind. Somit stimmen die Bewertungen der *Graphlet*-Vergleiche teilweise mit denen durch *PDBeFold* überein. Der einzige Unterschied in diesem Alignment ist, dass das Paar 1qq3, 1cgn als drittähnlichstes bewertet wurde, während es beim *Graphlet*-Vergleich als viertähnlichstes eingeordnet wurde.

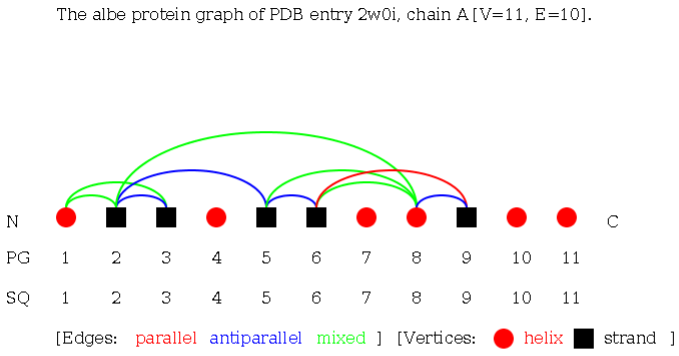
Die starke Ähnlichkeit der β-Proteine zu den α-β-Proteinen lässt sich erklären, wenn man die entsprechenden Graphen aus der PTGL betrachtet. Diese sind in den folgenden Abbildungen dargestellt:

The albe protein graph of PDB entry 2id9, chain A [V=10, E=5].



The albe protein graph of PDB entry 3i42, chain A [V=10, E=8].





In all diesen Graphen haben die meisten Knoten, die α-Helices repräsentieren keine adjazenten Knoten. Damit fließen für keinen dieser graphen mehr als 2 α-Helices in die *Graphlet*-Vektoren ein. Stattdessen finden sich nur die zusammenhängenden Teilgraphen dieser Graphen in den *Graphlet*-Vektoren wieder und diese Teilgraphen bestehen fast nur aus Knoten die β-Faltblrätter repräsentieren.

4.1.2 PDBTop500-Datensatz und Aldolasen

Der Vergleich der Proteine aus dem PDBTop500-Datensatz mit den Aldolasen hat gezeigt, dass sich die Verteilungen der *Graphlets* für einige *Graphlets* so stark unterscheiden, dass eine klare Unterscheidung der beiden anhand ihrer Verteilungen möglich zu sein scheint.

Bei den Aminosäuregraphen scheint diese Unterscheidung dabei am schlechtesten möglich zu sein. Da sich die Verteilungen für nicht-markierte *Graphlets* durch die Abwesenheit der *Graphlets* g_8 und g_{15} in den Aldolasen unterscheiden. Es ist nicht klar, ob diese *Graphlets* wirklich in allen anderen Proteinen vorkommen, oder diese Unterscheidung nur durch einige wenige Topologien, die im Datensatz vertreten sind, zustande kommt. Bei den markierten *Graphlets* unterscheiden sich die Verteilungen am stärksten für jene, die die räumliche Nähe von SSEs und Liganden bzw. Liganden untereinander repräsentieren. Auch hier ist es fraglich, ob dieses Merkmal zur Unterscheidung von Aldolasen von allen anderen Superfamilien wirklich geeignet ist, oder ob Liganden in dem Datensatz unterrepräsentiert sind.

Es ist interessant zu sehen, dass die Unterschiede in den *Graphlet*-Verteilungen

für Proteingraphen und Komplexgraphen am stärksten ist, obwohl diese im multiplen Vergleich der Fallstudie die geringeren Zusammenhänge mit der strukturellen Einteilung durch CATH zeigten. Die Ursache hierfür werden die verwendeten Metriken sein. Die hohen Unterschiede der *Graphlet*-Häufigkeiten der Komplexgraphen im Gegensatz zu den Proteingraphen sind wahrscheinlich auf die Kontakte der Polypeptidketten untereinander zurückzuführen.

4.2 Ausblick

4.2.1 Optimierung der Laufzeit von `graphletAnalyser`

Vor allem bei der Berechnung der *Graphlets* auf Aminosäurgraphen großer Proteine besteht Verbesserungspotential. Aktuell zählt das Programm bei einem Aufruf automatische alle *Graphlets* mit 3,4 und 5 Knoten jeweils getrennt voneinander.

Da für die Ähnlichkeitsberechnung in ihrer aktuellen Form alle *Graphlets* einbezogen werden, ist es sinnvoll diese Berechnungen in einem Algorithmus zusammenzufassen. Für Graphen mit n Knoten und einem maximalen Knotengrad d ließe sich die Laufzeit damit von $O(nd^4 + nd^3 + nd^2)$ auf $O(nd^4)$ reduzieren.

Diese Zusammenfassung ließe sich dadurch bewerkstelligen, dass man die Funktionen zum Überprüfen der *Graphlets* aus den Algorithmen zum Zählen der 3- und 4-*Graphlets* in den Algorithmus zum Zählen der 5-*Graphlets* in die *for*-Schleifen der entsprechenden Tiefen einfügt.

4.2.2 Neue Bewertungsschemata

Da die *Graphlet*-Verteilungen beim Vergleich des PDBTop500-Datensatzes mit den Aldolasen nur für einige wenige *Graphlets* stark genug waren, um beide anhand der Verteilungen unterschieden zu können ist es ratsam eine Bewertungsschema einzuführen, dass die verschiedenen *Graphlets* unterschiedlich stark gewichtet. Um die entsprechenden *Graphlets* zu identifizeren könnte man eine Faktorenanalyse durchführen. Durch diese erhält man Gewichtungen der einzelnen *Graphlets* die in das neue Bewertungsschema einfließen lassen kann.

Eine Alternative wäre die entsprechenden Gewichte anhand von bereits vorhandenen strukturellen Einteilungen zu berechnen - also *Graphlets*, die in einer Topologie oder Superfamilie besonders stark vertreten sind, stärker zu gewichten. Damit wäre allerdings nur ein Klassifikationsschema möglich, das unterscheiden kann, ob ein Eingabeprotein zu dieser strukturellen Klasse gehört oder nicht. Hierfür könnte man *Support-Vector-Machines* verwenden, wie sie von *Shervashidze et al.* [12] bereits zur Klassifizierung von Enzymen verwendet worden sind.

4.2.3 Weiterentwicklung des *Graphlet*-Worte-Algorithmus

Der *Graphlet*-Worte-Algorithmus ist so erweiterbar, dass Worte hinzugefügt werden können, die größere *Graphlets* repräsentieren können. Allerdings gilt dies

zunächst nicht für die Stern-*Graphlets*. Da in ihnen keine Euler-Wege, die mehr als 2 Knoten enthalten, sind, kann man sie nicht durch Worte repräsentieren.

Auch für die anderen *Graphlets* mit vier und fünf Knoten ist das Finden von repräsentierenden Worten nicht trivial, da es für sie deutlich mehr Euler-Wege gibt, als für die *Graphlets* der Größe 3. Dementsprechend wäre eine schnelle Implementierung nur für die *Graphlets* möglich, die einfache Wege oder Kreise sind.

4.2.4 Zusammenhängende Graphen

Viele Probleme bei der Ähnlichkeitsbewertung durch *Graphlets* scheinen daher zu rühren, dass Proteingraphen und Komplexgraphn nicht immer zusammenhängend sind. Um dieses Problem in der Zukunft umgehen zu können, gibt es zwei Möglichkeiten. Erstens kann man die Graphenberechnung durch PLCC so anpassen, dass ungeordnete Regionen der Polypeptidkette einbezogen werden. Dadurch vergrößert man aber wieder die Darstellung und verschlechtert deren intuitive Verständlichkeit. Die zweite Möglichkeit ist den *Graphlet*-Algorithmus so anzupassen, dass auch *Graphlets* erkannt werden, die nicht zusammenhängend sind. Dies ist zweifelsfrei möglich. Es ist jedoch fraglich, ob die zusätzlichen Dimensionen, die der *Graphlet*-Vektor dadurch erhält, wirklich für eine präzisere Unterscheidung sorgen können. Möglicherweise verschlechtert sich dadurch auch das Signal.

4.2.5 Schluss

Noch nicht fertig. kommt noch

Kapitel 5

Anhang

5.1 Bildverzeichnis

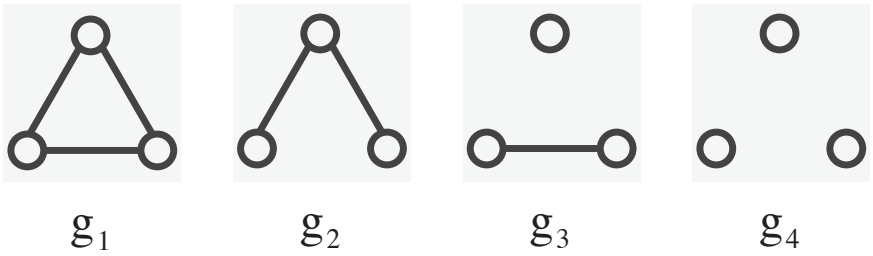


Abbildung 5.1: Graphlets der Größe 3 (*Shervashidze et al.*)

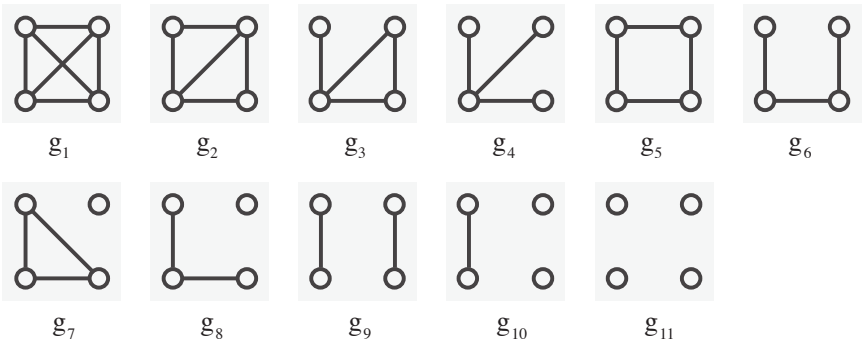


Abbildung 5.2: Graphlets der Größe 4 (*Shervashidze et al.*)

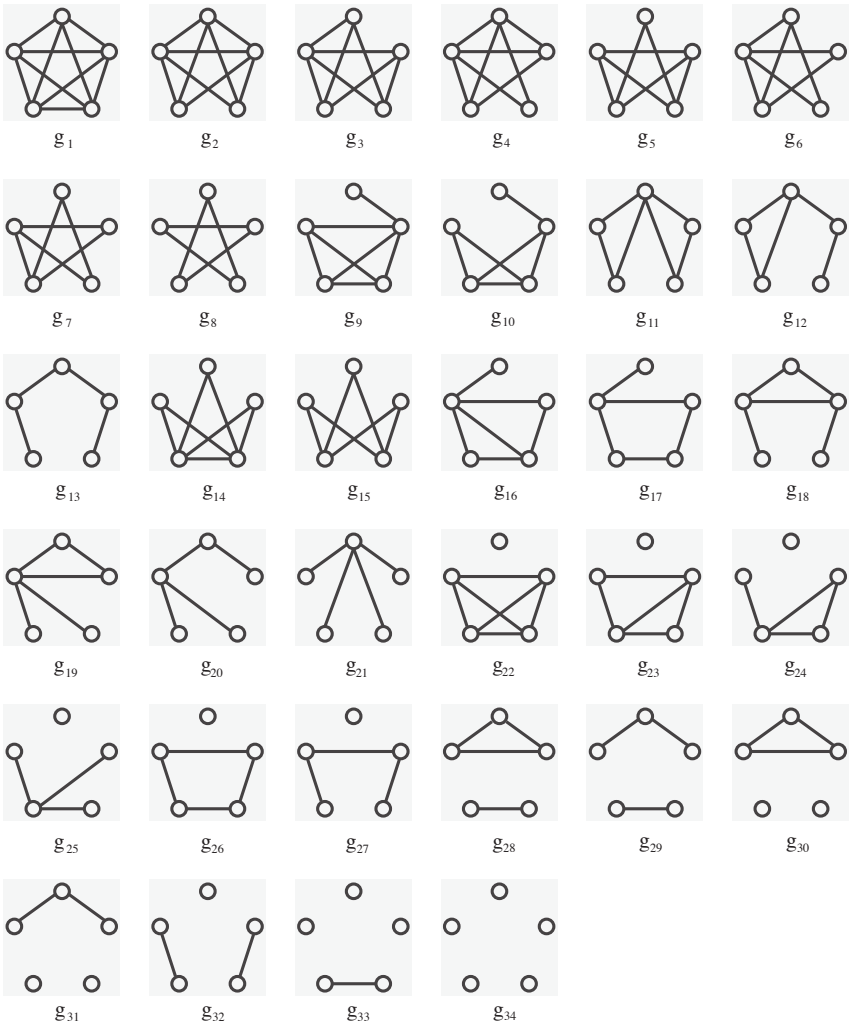


Abbildung 5.3: Graphlets der Größe 5 (*Shervashidze et al.*)

PDB-ID	1qpu	1qq3	1cgn	1he9	3gf9	1exs	1ngl	1qqs	3slo	1wjx	5chy	2id9	3i42	1d4o	2w0i
1qpu	X	0.806	0.606	0.697	3.131	1.159	2.195	2.197	1.226	1.049	1.014	0.811	1.098	0.628	0.972
1qq3	0.806	X	0.405	1.504	1.567	1.703	0.883	0.883	1.396	1.270	0.869	0.405	0.693	0.988	1.779
1cgn	0.606	0.405	X	0.810	1.432	1.633	1.193	1.193	1.513	1.339	1.060	0.811	1.098	1.011	1.516
1he9	0.697	1.504	0.810	X	4.492	3.253	1.386	1.386	4.357	1.291	1.135	1.099	1.386	3.599	2.506
3gf9	3.131	1.567	1.432	4.492	X	2.523	2.815	2.817	1.596	2.056	1.816	1.917	2.034	3.122	4.130
1exs	1.159	1.703	1.633	3.253	2.523	X	1.324	1.324	2.968	0.696	0.885	1.492	1.610	3.102	3.820
1ngl	2.195	0.883	1.193	1.386	2.815	1.324	X	0.002	0.787	2.629	2.279	0.788	0.286	0.980	2.722
1qqs	2.197	0.883	1.193	1.386	2.817	1.324	0.002	X	0.787	2.631	2.281	0.788	0.285	0.980	2.722
3slo	1.226	1.396	1.513	4.357	1.596	2.968	0.787	0.787	X	0.257	0.705	1.024	1.073	3.137	6.154
1wjx	1.049	1.270	1.339	1.291	2.056	0.696	2.629	2.631	0.257	X	1.043	0.933	0.914	0.644	2.093
5chy	1.014	0.869	1.060	1.135	1.816	0.885	2.279	2.281	0.705	1.043	X	0.607	0.800	0.275	2.282
2id9	0.811	0.405	0.811	1.099	1.917	1.492	0.788	0.788	1.024	0.933	0.607	X	0.692	0.783	2.890
3i42	1.098	0.693	1.098	1.386	2.034	1.610	0.286	0.285	1.073	0.914	0.800	0.692	X	1.076	3.008
1d4o	0.628	0.988	1.011	3.599	3.122	3.102	0.980	0.980	3.137	0.644	0.275	0.783	1.076	X	4.406
2w0i	0.972	1.779	1.516	2.506	4.130	3.820	2.722	2.722	6.154	2.093	2.282	2.890	3.008	4.406	X

Tabelle 5.2: Distanzen der Proteingraphen, gemessen mit RGF. In den Zellen der Tabelle stehen die RGF-Distanzen für die entsprechenden PDB-Dateien. Für jede Zeile (jedes Protein) sind die 4 niedrigsten Distanzen grün unterlegt. Je dunkler das grün ist, desto kürzer ist die Distanz

5.2 Tabellenverzeichnis

PDB-ID	1qpu	1qq3	1cgn	1he9	3gf9	1exs	1ngl	1qqs	3slo	1wjx	5chy	2id9	3i42	1d4o	2w0i
1qpu	X	1.139	1.296	6.159	5.656	10.52	11.00	12.35	12.45	11.66	7.517	6.128	6.625	7.326	8.572
1qq3	1.139	X	1.310	5.980	5.645	9.964	10.38	11.61	11.84	11.10	6.967	5.776	6.039	7.491	8.059
1cgn	1.296	1.310	X	6.093	5.737	9.987	10.44	11.55	11.81	11.16	6.801	5.663	5.929	7.091	7.918
1he9	6.159	5.980	6.093	X	2.289	8.032	6.813	11.03	10.87	8.405	3.974	2.445	3.452	10.93	4.807
3gf9	5.656	5.645	5.737	2.289	X	9.544	8.479	12.36	12.19	10.03	5.135	3.370	4.359	12.08	6.018
1exs	10.52	9.964	9.987	8.032	9.544	X	4.463	4.041	3.138	2.802	4.773	7.086	5.541	7.374	4.530
1ngl	11.00	10.38	10.44	6.813	8.479	4.463	X	7.716	6.865	3.764	4.718	6.054	5.065	9.755	4.031
1qqs	12.35	11.61	11.55	11.03	12.36	4.041	7.716	X	1.674	5.340	8.148	10.20	8.612	7.226	7.404
3slo	12.45	11.84	11.81	10.87	12.19	3.138	6.865	1.674	X	4.750	7.488	9.994	8.600	7.680	6.997
1wjx	11.66	11.10	11.16	8.405	10.03	2.802	3.764	5.340	4.750	X	5.264	7.522	5.699	8.450	4.384
5chy	7.517	6.967	6.801	3.974	5.135	4.773	4.718	8.148	7.488	5.264	X	2.600	2.817	8.667	1.497
2id9	6.128	5.776	5.663	2.445	3.370	7.086	6.054	10.20	9.994	7.522	2.600	X	2.447	10.24	3.657
3i42	6.625	6.039	5.929	3.452	4.359	5.541	5.065	8.612	8.600	5.699	2.817	2.447	X	9.544	2.817
1d4o	7.326	7.491	7.091	10.93	12.08	7.374	9.755	7.226	7.680	8.450	8.667	10.24	9.544	X	8.970
2w0i	8.572	8.059	7.918	4.807	6.018	4.530	4.031	7.404	6.997	4.384	1.497	3.657	2.817	8.970	X

Tabelle 5.1: Distanzen der Aminosäuregraphen, gemessen mit RGF. In den Zellen der Tabelle stehen die RGF-Distanzen für die entsprechenden PDB-Dateien. Für jede Zeile (jedes Protein) sind die 4 niedrigsten Distanzen grün unterlegt. Je dunkler das grün ist, desto kürzer ist die Distanz

PDB-ID	1qpu	1qq3	1cgn	1he9	3gf9	1exs	1ngl	1qqs	3slo	1wjx	5chy	2id9	3i42	1d4o	2w0i
1qpu	X	4.405	5.056	2.271	10.75	3.567	2.309	13.66	14.10	2.128	1.933	2.165	1.877	8.791	4.422
1qq3	4.405	X	1.906	1.714	10.03	3.860	3.457	6.851	12.70	2.587	2.431	2.557	2.269	7.907	3.002
1cgn	5.056	1.906	X	0.985	9.469	1.712	1.979	7.333	12.20	1.954	1.120	0.249	0.037	6.501	0.980
1he9	2.271	1.714	0.985	X	5.881	3.456	1.386	4.069	5.432	1.466	1.291	1.099	1.386	9.950	2.506
3gf9	10.75	10.03	9.469	5.881	X	5.473	4.230	14.86	15.99	2.516	1.971	2.575	2.693	8.811	11.61
1exs	3.567	3.860	1.712	3.456	5.473	X	1.290	3.217	4.391	0.611	0.662	1.459	1.576	9.191	4.226
1ngl	2.309	3.457	1.979	1.386	4.230	1.290	X	3.637	2.620	2.697	2.644	0.788	0.286	4.373	2.722
1qqs	13.66	6.851	7.333	4.069	14.86	3.217	3.637	X	12.90	5.645	4.202	3.394	3.512	16.22	4.907
3slo	14.10	12.70	12.20	5.432	15.99	4.391	2.620	12.90	X	4.974	3.863	2.351	2.412	14.09	7.116
1wjx	2.128	2.587	1.954	1.466	2.516	0.611	2.697	5.645	4.974	X	0.360	0.961	0.965	2.225	2.042
5chy	1.933	2.431	1.120	1.291	1.971	0.662	2.644	4.202	3.863	0.360	X	0.797	0.914	1.728	2.093
2id9	2.165	2.557	0.249	1.099	2.575	1.459	0.788	3.394	2.351	0.961	0.797	X	0.692	1.921	2.890
3i42	1.877	2.269	0.037	1.386	2.693	1.576	0.286	3.512	2.412	0.965	0.914	0.692	X	2.039	3.008
1d4o	8.791	7.907	6.501	9.950	8.811	9.191	4.373	16.22	14.09	2.225	1.728	1.921	2.039	X	10.76
2w0i	4.422	3.002	0.980	2.506	11.61	4.226	2.722	4.907	7.116	2.042	2.093	2.890	3.008	10.76	X

Tabelle 5.3: Distanzen der Komplexgraphen, gemessen mit RGF. In den Zellen der Tabelle stehen die RGF-Distanzen für die entsprechenden PDB-Dateien. Für jede Zeile (jedes Protein) sind die 4 niedrigsten Distanzen grün unterlegt. Je dunkler das grün ist, desto kürzer ist die Distanz

PDB-ID	1qpu	1qq3	1cgn	1he9	3gf9	1exs	1ngl	1qqs	3slo	1wjx	5chy	2id9	3i42	1d4o	2w0i
1qpu	X	1.0	1.0	0.621	0.666	0.276	0.25	0.25	0.224	0.25	0.538	0.714	0.5	0.363	0.333
1qq3	1.0	X	1.0	0.621	0.621	0.304	0.276	0.304	0.224	0.25	0.578	0.666	0.666	0.428	0.463
1cgn	1.0	1.0	X	0.621	0.714	0.333	0.224	0.224	0.224	0.276	0.5	0.666	0.578	0.333	0.428
1he9	0.621	0.621	0.621	X	0.935	0.463	0.428	0.176	0.276	0.304	0.621	0.875	0.764	0.333	0.463
3gf9	0.666	0.621	0.714	0.935	X	0.276	0.333	0.2	0.2	0.224	0.621	0.764	0.621	0.25	0.463
1exs	0.276	0.304	0.333	0.463	0.276	X	0.621	0.621	0.764	0.875	0.666	0.5	0.578	0.463	0.818
1ngl	0.25	0.276	0.224	0.428	0.333	0.621	X	0.276	0.428	0.666	0.621	0.538	0.578	0.463	0.666
1qqs	0.25	0.304	0.224	0.176	0.2	0.621	0.276	X	1.0	0.621	0.5	0.276	0.395	0.463	0.463
3slo	0.224	0.224	0.224	0.276	0.2	0.764	0.428	1.0	X	0.538	0.463	0.333	0.395	0.463	0.5
1wjx	0.25	0.25	0.276	0.304	0.224	0.875	0.666	0.621	0.538	X	0.538	0.363	0.5	0.395	0.621
5chy	0.538	0.578	0.5	0.621	0.621	0.666	0.621	0.5	0.463	0.538	X	0.818	0.764	0.428	1.0
2id9	0.714	0.666	0.666	0.875	0.764	0.5	0.538	0.276	0.333	0.363	0.818	X	0.818	0.304	0.621
3i42	0.5	0.666	0.578	0.764	0.621	0.578	0.578	0.395	0.395	0.5	0.764	0.818	X	0.333	0.714
1d4o	0.363	0.428	0.333	0.333	0.25	0.463	0.463	0.463	0.463	0.395	0.428	0.304	0.333	X	0.333
2w0i	0.333	0.463	0.428	0.463	0.463	0.818	0.666	0.463	0.5	0.621	1.0	0.621	0.714	0.333	X

Tabelle 5.4: Jaccard-Indizes der Aminosäuregraphen. In den Zellen der Tabelle stehen die Jaccard-Indizes für die entsprechenden PDB-Dateien. Für jede Zeile (jedes Protein) sind die 4 größten Indizes grün unterlegt. Je dunkler das grün ist, desto größer der Index

PDB-ID	1qpu	1qq3	1cgn	1he9	3gf9	1exs	1ngl	1qqs	3slo	1wjx	5chy	2id9	3i42	1d4o	2w0i
1qpu	X	0.764	0.764	0.578	0.463	0.463	0.621	0.621	0.428	0.578	0.666	0.666	0.666	0.538	0.395
1qq3	0.764	X	0.935	0.764	0.463	0.621	0.714	0.714	0.538	0.714	0.764	0.875	0.818	0.666	0.5
1cgn	0.764	0.935	X	0.764	0.428	0.578	0.714	0.714	0.538	0.666	0.714	0.875	0.818	0.621	0.463
1he9	0.578	0.764	0.764	X	0.463	0.578	0.621	0.621	0.578	0.538	0.621	0.714	0.714	0.666	0.538
3gf9	0.463	0.463	0.428	0.463	X	0.428	0.463	0.463	0.463	0.463	0.538	0.428	0.463	0.5	0.333
1exs	0.463	0.621	0.578	0.578	0.428	X	0.5	0.5	0.714	0.5	0.578	0.666	0.578	0.714	0.621
1ngl	0.621	0.714	0.714	0.621	0.463	0.5	X	1.0	0.538	0.818	0.818	0.818	0.875	0.578	0.395
1qqs	0.621	0.714	0.714	0.621	0.463	0.5	1.0	X	0.538	0.818	0.818	0.818	0.875	0.578	0.395
3slo	0.428	0.538	0.538	0.578	0.463	0.714	0.538	0.538	X	0.538	0.5	0.578	0.621	0.578	0.463
1wjx	0.578	0.714	0.666	0.538	0.463	0.5	0.818	0.818	0.538	X	0.764	0.714	0.818	0.538	0.395
5chy	0.666	0.764	0.714	0.621	0.538	0.578	0.818	0.818	0.5	0.764	X	0.875	0.818	0.621	0.395
2id9	0.666	0.875	0.875	0.714	0.428	0.666	0.818	0.818	0.578	0.714	0.875	X	0.935	0.714	0.463
3i42	0.666	0.818	0.818	0.714	0.463	0.578	0.875	0.875	0.621	0.818	0.818	0.935	X	0.666	0.463
1d4o	0.538	0.666	0.621	0.666	0.5	0.714	0.578	0.578	0.578	0.538	0.621	0.714	0.666	X	0.5
2w0i	0.395	0.5	0.463	0.538	0.333	0.621	0.395	0.395	0.463	0.395	0.395	0.463	0.463	0.5	X

Tabelle 5.5: Jaccard-Indizes der Proteingraphen. In den Zellen der Tabelle stehen die Jaccard-Indizes für die entsprechenden PDB-Dateien. Für jede Zeile (jedes Protein) sind die 4 größten Indizes grün unterlegt. Je dunkler das grün ist, desto größer der Index

PDB-ID	1qpu	1qq3	1cgn	1he9	3gf9	1exs	1ngl	1qqs	3slo	1wjx	5chy	2id9	3i42	1d4o	2w0i
1qpu	X	0.428	0.395	0.428	0.153	0.304	0.395	0.132	0.111	0.363	0.395	0.428	0.428	0.276	0.276
1qq3	0.428	X	0.666	0.621	0.2	0.395	0.5	0.2	0.090	0.5	0.5	0.538	0.538	0.333	0.428
1cgn	0.395	0.666	X	0.621	0.224	0.428	0.5	0.224	0.111	0.463	0.5	0.538	0.538	0.363	0.363
1he9	0.428	0.621	0.621	X	0.25	0.578	0.621	0.153	0.090	0.538	0.621	0.714	0.714	0.428	0.538
3gf9	0.153	0.2	0.224	0.25	X	0.276	0.224	0.153	0.153	0.276	0.304	0.25	0.224	0.276	0.224
1exs	0.304	0.395	0.428	0.578	0.276	X	0.5	0.2	0.153	0.578	0.578	0.666	0.578	0.5	0.621
1ngl	0.395	0.5	0.5	0.621	0.224	0.5	X	0.132	0.132	0.818	0.818	0.818	0.875	0.428	0.395
1qqs	0.132	0.2	0.224	0.153	0.153	0.2	0.132	X	0.224	0.132	0.132	0.132	0.132	0.132	0.2
3slo	0.111	0.090	0.111	0.090	0.153	0.153	0.132	0.224	X	0.111	0.090	0.090	0.111	0.111	0.153
1wjx	0.363	0.5	0.463	0.538	0.276	0.578	0.818	0.132	0.111	X	0.935	0.714	0.818	0.5	0.395
5chy	0.395	0.5	0.5	0.621	0.304	0.578	0.818	0.132	0.090	0.935	X	0.875	0.818	0.5	0.395
2id9	0.428	0.538	0.538	0.714	0.25	0.666	0.818	0.132	0.090	0.714	0.875	X	0.935	0.5	0.463
3i42	0.428	0.538	0.538	0.714	0.224	0.578	0.875	0.132	0.111	0.818	0.818	0.935	X	0.428	0.463
1d4o	0.276	0.333	0.363	0.428	0.276	0.5	0.428	0.132	0.111	0.5	0.5	0.5	0.428	X	0.333
2w0i	0.276	0.428	0.363	0.538	0.224	0.621	0.395	0.2	0.153	0.395	0.395	0.463	0.463	0.333	X

Tabelle 5.6: Jaccard-Indizes der Komplexgraphen. In den Zellen der Tabelle stehen die Jaccard-Indizes für die entsprechenden PDB-Dateien. Für jede Zeile (jedes Protein) sind die 4 größten Indizes grün unterlegt. Je dunkler das grün ist, desto größer der Index

Literaturverzeichnis

[1] Tatiana Bakirova, Tim Schäfer, and Ina Koch. Comparison of protein topology graphs using graphlet-based methods. *GCB 2013 Göttingen-Poster Abstracts*.

[2] Sayoni Das, Natalie L Dawson, and Christine A Orengo. Diversity in protein domain superfamilies. *Current opinion in genetics & development*, 35:40–49, 2015.

[3] Liisa Holm and Chris Sander. Protein structure comparison by alignment of distance matrices. *Journal of molecular biology*, 233(1):123–138, 1993.

[4] Björn H Junker and Falk Schreiber. *Analysis of biological networks*, volume 2. John Wiley & Sons, 2011.

[5] Richard M Karp. *Reducibility among combinatorial problems*. Springer, 1972.

[6] E Krissinel and K Henrick. Secondary-structure matching (ssm), a new tool for fast protein structure alignment in three dimensions. *Acta Crystallographica Section D: Biological Crystallography*, 60(12):2256–2268, 2004.

[7] Simon C Lovell, Ian W Davis, W Bryan Arendall, Paul IW de Bakker, J Michael Word, Michael G Prisant, Jane S Richardson, and David C Richardson. Structure validation by cα geometry: ϕ, ψ and cβ deviation. *Proteins: Structure, Function, and Bioinformatics*, 50(3):437–450, 2003.

[8] Patrick May, Annika Kreuchwig, Thomas Steinke, and Ina Koch. Ptgl: a database for secondary structure-based protein topologies. *Nucleic acids research*, 38(suppl 1):D326–D330, 2010.

[9] Natasa Pržulj, Derek G Corneil, and Igor Jurisica. Modeling interactome: scale-free or geometric? *Bioinformatics*, 20(18):3508–3515, 2004.

[10] Natasa Pržulj, Derek G Corneil, and Igor Jurisica. Efficient estimation of graphlet frequency distributions in protein–protein interaction networks. *Bioinformatics*, 22(8):974–980, 2006.

[11] Tim Schäfer, Patrick May, and Ina Koch. Computation and Visualization of Protein Topology Graphs Including Ligand Information. In Sebastian Böcker, Franziska Hufsky, Kerstin Scheubert, Jana Schleicher, and Stefan Schuster, editors, *German Conference on Bioinformatics 2012*, volume 26 of *OpenAccess Series in Informatics (OASIcs)*, pages 108–118, Dagstuhl, Germany, 2012. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[12] Nino Shervashidze, Tobias Petri, Kurt Mehlhorn, Karsten M Borgwardt, and Svn Vishwanathan. Efficient graphlet kernels for large graph comparison. In *International conference on artificial intelligence and statistics*, pages 488–495, 2009.

[13] Ilya N Shindyalov and Philip E Bourne. Protein structure alignment by incremental combinatorial extension (ce) of the optimal path. *Protein engineering*, 11(9):739–747, 1998.

[14] Yuzhen Ye and Adam Godzik. Flexible structure alignment by chaining aligned fragment pairs allowing twists. *Bioinformatics*, 19(suppl 2):ii246–ii255, 2003.

[15] Yang Zhang and Jeffrey Skolnick. Tm-align: a protein structure alignment algorithm based on the tm-score. *Nucleic acids research*, 33(7):2302–2309, 2005.