Exposé
Finding Protein Complexes through Clustering of Protein-Protein Interaction Networks

Chris Roeseler
Humboldt Universität zu Berlin

15. Juni 2020

1 Einleitung und Motivation

Proteine und Proteinkomplexe sind wichtige Bestandteile in Zellen. Sie nehmen eine Vielzahl von Aufgaben wahr, vom Transport diverser Stoffe innerhalb eines Organismus bis zum Transkribieren weiterer Zellbestandteile aus DNS[1]. Schon vor 21 Jahren war bekannt, dass Proteine selten völlig eigenständig arbeiten, sondern meist Komplexe bilden. „[...]we now know that nearly every major process in a cell is carried out by assemblies of 10 or more protein molecules."[1] Informationen über diese sogenannte Quartärstruktur sind in der Biologie entsprechend wichtig. „The QS of a protein or a protein assembly is almost invariably essential to its function[...]."[2] Proteinkomplexe sorgen für eine Modularisierung in der Zelle. Somit kann eine Vielzahl von Reaktionen innerhalb der Zelle ablaufen und reguliert werden. Aufgrund dieser Modularität sind manche Komplexe in der Zelle nicht dauerhaft vorhanden. Sie werden erst bei Bedarf gebildet und zerfallen, nachdem sie ihre Funktion erfüllt haben. Diese Komplexe können schon nach wenigen Sekunden wieder aufteilen, wodurch eine Messung im Labor erschwert wird[3]. Eine Möglichkeit aus der Menge an Informationen aus PPI-Datenbanken Rückschlüsse über Proteinkomplexe zu ziehen, die z. B. im Labor geprüft werden können würde hierbei hilfreich sein.

2 Ziel

Ziel dieser Arbeit liegt in der heuristische Analyse der Parameterauswahl eines Quasi-Cliquen-Algorithmus am Beispiel eines Proteininteraktionsnetzwerkes.

3 Umsetzung

Ein Ziel dieser Arbeit soll auf der effizienten Umsetzung des Algorithmus sein um eine maximale Menge an Parameter Kombinationen zu analysieren. Hierfür soll komplett *In-Memory* gearbeitet werden. Aus der Menge von öffentlich zugänglichen PPI- und Komplexdatenbanken (siehe Tabelle 1) sollen einzelne Instanzen in ein Graphenobjekt überführt werden. Diese können für die Suche komplett im Hauptspeicher gehalten werden. Eine Instanz ist hierbei die kleinste Einteilung, die von der jeweiligen Datenbank angeboten wird, wie PPI's einzelner Organismen, Gattungen, Klassen, Typen, etc..

PPI	Komplex	Kombiniert
IntAct	IntAct: Compex Portal	Reactome
String	Corum	PCDq
BioGrid	FuzDB	
	3D Complex	

Tabelle 1: Auswahl möglicher Datenbanken

Alle genannten Datenbanken bieten ihre Daten zum freien Download an. Nach Akquirierung wird durch *pre-processing* das jeweilige Netzwerk in ein Graphenobjekt zu übertragen. Dies wird mit einem simplen Python Script realisiert.

Die Darstellung des Netzwerks (Graphenobjekt) wird mit einer *graph library* umgesetzt. Hierbei stehen diverse Möglichkeiten zur Auswahl, z. B. Pathpy, Networkx und Networkit. Da auf dem Graphen nur simple Aktionen ausgeführt werden ist die Geschwindigkeit Hauptauswahlkriterium.
Die Wahl für eine *graph library* gegen eine einfache Listen Struktur ist eine höhere Abstraktionsebene die das Arbeiten intuitiver gestaltet und die Anbindung an weitere Graphen Tools vereinfacht. Gleichzeitig sind besagte libraries hoch effizient implementiert. Im Fall von Networkit auf Basis von C++ was eine geringe Laufzeit bei genannter Abstraktion ermöglicht. Die Darstellung der Netzwerke wird auf ein Minimum beschränkt. Ein PPI wird dargestellt durch einen ungerichteten Graphen $G = (V, E)$.

- V: Menge aller Proteine im spezifischen PPI
- E: Menge aller paarweisen Interaktionen

Die Analyse der Ergebnisse soll heuristisch erfolgen. Hierfür werden die gefundenen Kandidaten aus jeder Parameterkombination mit den schon bekannten Proteinkomplexen verglichen. Ein ideales Ergebnis sollte dann alle bekannten Komplexe abdecken und damit die Korrektheit der weiteren Kandidaten bestärken. Die Abdeckung der Kandidaten mit bekannten Komplexen muss hierbei noch analysiert werden. Durch die recht flexible Definition der Quasi-Clique kann nicht von einer 100%-Überdeckung von Komplex und Kandidat ausgegangen werden. Denkbar ist eine prozentuale Angabe der Überdeckung.

Mögliche Erweiterungen bei ausreichender Zeit könnten das Inkorporieren des Zelltyps sein um die Suchmenge weiter zu verkleinern.

4 Algorithmus Wahl

Die Arbeit „On effectively finding maximal quasi-cliques in graphs“[5] aus dem Jahr 2007 bietet eine Erweiterung der Maximum-Cliquen-Eigenschaft auf Quasi-Cliquen an. Diese λ,γ-Cliquen sollen für diese Arbeit gesucht und ausgewertet werden. Hierfür gilt folgende Definition:

Definition 1. *Given an undirected graph (V, E), and two parameters λ and γ with $0 \leq \lambda \leq \gamma \leq 1$, the subgraph induced by a subset of the node set $V' \subseteq V$ is a (λ, γ)-quasi-clique if, and only if, the following two conditions hold:*

- $\forall v \in V' : deg_{V'}(v) \geq \lambda \cdot (|V'| - 1)$
- $|E'| \geq \gamma \cdot \binom{|V'|}{2}$

where $E' = E \cap (V' \times V')$ and $deg_{V'}(v)$ is the number of elements of V' connected to v.

Für λ=γ=1 ergibt sich die klassische Maximal-Cliquen-Suche. Somit ist die Quasi-Cliquen-Suche für beliebige λ,γ ein Unterproblem der Maximal-Cliquen-Suche, welches NP hart ist. Aus diesem Grund soll ein heuristisches Suchverfahren verwendet werden. Neben der Definition bietet das genannte Paper Erweiterungen von MAX-clique Algorithmen für die Quasi-Cliquen Suche an. Diese wurde für Reactive Local Search(RLS) und Dynamic Local Search(DLS) umgesetzt und getestet. Für diese Arbeit wurde DLS gewählt.

DLS ist ein *stochastic local search algorithm* der mit zwei alternierenden Phasen arbeitet: einer Expand Phase und einer Plateau Phase. In der Expand Phase werden benachbarte Knoten aufgenommen, solange sie die Cliquen Eigenschaft erfüllen. Kann nicht weiter expandiert werden, beginnt die Plateau Phase. In dieser Phase werden Nachbarknoten gesucht, die die Cliquen Eigenschaft erfüllen, sofern ein bereits eingeführter Knoten wieder entfernt wird. Dies wird solange wiederholt, bis ein Expand möglich ist. Beide Phasen wechseln sich ab, bis die Clique die Zielgröße erreicht hat oder die maximale Schrittzahl erreicht ist.

Zur Suche von Quasi-Cliquen müssen hierbei drei Mengen konstruiert werden: hinzufügbare, kritische und entfernbare Knoten. Hinzufügbare und entfernbare Knoten bezeichnen hierbei Knoten, die in die Clique eingefügt oder entfernt werden können, ohne die Quasi-Cliquen Eigenschaft zu verletzen. Kritische Knoten bezeichnen Knoten, deren Grad so niedrig ist, dass bei entfallen einer weiteren Kante zu diesem Knoten die Quasi-Cliquen Eigenschaft verletzt wird. Problematisch in der Implementation ist das jeder Knoten eine eigene kritische Menge besitzt. Ausserdem müssen für Expand und Plateau Phasen separate Mengen

organisiert werden.

Erste Versuche haben gezeigt das die Neuberechnung dieser Mengen in jedem Schritt nicht realisierbar ist, ohne die Laufzeit exponentiell zu erhöhen. Aus diesem Grund muss eine Datenstruktur entwickelt werden, welche eine Änderung in der Clique mit minimalem Aufwand in die Hilfsmengen überführt. In dem Paper „On effectively finding maximal quasi-cliques in graphs“ werden hierfür Strukturen vorgeschlagen, die angepasst werden sollen.

Literatur

[1] B. Alberts, "The cell as a collection of protein machines: preparing the next generation of molecular biologists," *Cell*, vol. 92, no. 3, pp. 291–294, 1998.

[2] J. Janin, R. P. Bahadur, and P. Chakrabarti, "Protein–protein interaction and quaternary structure," *Quarterly reviews of biophysics*, vol. 41, no. 2, pp. 133–180, 2008.

[3] J. B. Pereira-Leal, E. D. Levy, and S. A. Teichmann, "The origins and evolution of functional modules: lessons from protein complexes," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 361, no. 1467, pp. 507–517, 2006.

[4] D. Szklarczyk and L. J. Jensen, *Protein-Protein Interaction Databases*, pp. 39–56. New York, NY: Springer New York, 2015.

[5] M. Brunato, H. H. Hoos, and R. Battiti, "On effectively finding maximal quasi-cliques in graphs," in *International conference on learning and intelligent optimization*, pp. 41–55, Springer, 2007.