Exposé

Indizierung von Graphen zur Beantwortung von Erreichbarkeits- und Distanzanfragen

Christoph Wagner

Betreuer: Silke Trißl
Lehrstuhl Wissensmanagement in der Bioinformatik

April 2006

1 Motivation

Die Aufklärung von Stoffwechselwegen ist ein zentrales Anliegen der Lebenswissenschaften. Die Modellierung und Analyse metabolischer Netzwerke ist Gegenstand eines eigenen Forschungszweiges, der Systembiologie. Datenbanken wie KEGG machen Netzwerke verschiedener Spezies öffentlich zugänglich. Stoffwechselwege werden von Molekülen gebildet, die durch chemische Reaktionen miteinander in Beziehung stehen. Metabolische Netzwerke sind also Graphen, deren Knotenmenge aus Proteinen, Enzymen, anderen chemischen Substanzen o. ä. gebildet werden. Die qualitative Suche nach Stoffwechselwegen mit spezifischen Eigenschaften entspricht damit in wesentlichen Teilen elementaren graphentheoretischen Aufgabenstellungen. Da die Graphen sehr groß sein können, ist es notwendig, effiziente Algorithmen und Speicherstrukturen zur Unterstützung der Suche zu verwenden. Potenziell läßt sich dazu die spezielle Struktur solcher Netzwerke ausnutzen.

2 Zielsetzung

Es sollen Algorithmen entwickelt und implementiert werden, die es ermöglichen, Pfadanfragen an metabolische Netzwerke bzw. beliebige gerichtete Graphen $G =$

(V, E) zu stellen. Varianten davon sind z. B.
Gegegen zwei Knoten A und B,

- gibt es einen Pfad von A nach B?
- welches ist der kürzeste Pfad von A nach B?
- gibt es einen Pfad von A nach B mit Länge k?
- zähle alle Pfade von A nach B auf
- etc.

Die Daten der Netzwerke liegen als Liste von Knoten und (gerichteten, ungewichteten) Kanten in einer Oracle-Datenbank vor. Die Pfadanfragen sollen auch auf großen Graphen mit mehreren Millionen Kanten noch durchführbar sein. Zu implementieren sind also nicht hauptspeicher-basierte Algorithmen, sondern solche, die auch die Kosten für I/O-Zugriffe berücksichtigen. Um eine schnelle Beantwortung der Anfragen zu ermöglichen, sollen Indexstrukturen angelegt werden, die die Beantwortung so gut wie möglich unterstützen und wenig Speicherplatz benötigen.

3 Vorgehensweise

Der limitierende Faktor, der die Anzahl der anwendbaren Algorithmen und Indexstrukturen einschränkt, ist der Speicherbedarf. So ist es bei großen Netzwerken unmöglich, die Transitive Hülle zu berechnen, d. h. für jedes Paar $(u, v) \in E \times E$ die Länge eines kürzesten Pfades von u nach v zu speichern, da diese im worstcase quadratisch in der Knotenzahl ist.

Es werden also Methoden verlangt, die Informationen über die Struktur des Graphen wesentlich effizienter abspeichern können. Zur Überprüfung der Erreichbarkeit genügt es, die starken Zusammenhangskomponenten des Graphen zu untersuchen. Starke Zusammenhangskomponenten in gerichteten Graphen bilden das Gegenstück von Zusammenhangskomponenten in ungerichteten Graphen. Damit reduziert sich der Rechenaufwand zur Entscheidung der Erreichbarkeit erheblich. Innerhalb einer Komponente ist die Eigenschaft „Erreichbarkeit“ trivial erfüllt. Die starken Zusammenhangskomponenten selbst bilden einen gerichteten azyklischen Graphen (DAG). Zur Entscheidung von Erreichbarkeit zwischen Knoten in verschiedenen Komponenten kann man entweder den DAG traversieren oder einen vorberechneten Index, beispielsweise die transitive Hülle über den Komponenten anfragen. Diese würde sehr viel weniger Knoten enthalten als der Ursprungsgraph. Starke Zusammenhangskomponenten sind in linearer Zeit berechenbar, z. B.

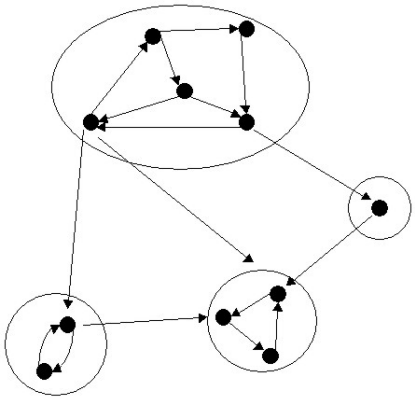

Abbildung 1: Starke Zusammenhangskomponenten

- durch zweimalige Tiefensuche (beschrieben z. B. in [**?**]).
- oder durch den Algorithmus von Tarjan

Zur Berechung der Distanz, also der kürzesten Entfernung zwischen zwei Knoten, reicht die Information der starken Zusammenhangskomponenten nicht aus. Die einfachste Möglichkeit, Distanzinformationen zur Verfügung zu stellen, ist, die Kanten in der transitiven Hülle um Distanzbeschriftungen zu erweitern. Diese Methode ist wegen des quadratischen Platzbedarfes jedoch für große Graphen nicht durchführbar. Es existiert jedoch ein Verfahren um die transitive Hülle stark zu komprimieren. Der 2-Hop-Cover Algorithmus ([**?**]) erlaubt es, alle in der transitiven Hülle repräsentierten Pfade als Konkatenation je zweier Pfadteilstücke zu repräsentieren. Diese Teilstücke werden an sogenannten „hop - nodes“ zusammengefügt. Statt der ganzen transitiven Hülle speichert man nur die Längen der Pfade zu den hop-nodes. Damit ist es möglich, alle Entfernungen zwischen Knoten im Graphen zu rekonstruieren. Ein Problem des Algorithmus besteht in der schlechten Laufzeit bei der Berechnung der hop-nodes. Ein anderes besteht darin, dass die transitive Hülle vor der Überdeckung durch hop-nodes zunächst vollständig materialisiert werden muss. Darum soll in der Arbeit geprüft werden, ob sich der Aufwand beschränken läßt, indem man sich bei der Berechnung auf kleinere Partitionen (z. B. starke Zusammenhangskomponenten) des Graphen beschränkt oder Distanzen nur bis zu einer gewissen Tiefe berücksichtigt werden sollen. Für Knotenpaare, deren Entfernung aufgrund dieser Beschränkung nicht effizient angefragt werden kann, muss dann eine Breitensuche durchgeführt werden.

Diese Verfahren und Algorithmen sollen in der Diplomarbeit für das relationale Datenbanksystem Oracle implementiert werden. Als Programmiersprachen bieten sich die native Oracle-Sprache PL/SQL an oder Java in Verbindung mit SQLJ

oder JDBC. Es ist einerseits zu prüfen, ob die beschriebenen Verfahren mit den theoretisch garantierten Laufzeitbeschränkungen implementierbar sind. Auf der anderen Seite sollen die Algorithmen skalierbar sein und gute Laufzeiten in der Praxis erbringen.

Literatur

[1] O. Krumke, H. Noltemeier. *Graphentheoretische Konzepte und Algorithmen*. B. G. Teubner. 152-156, Wiesbaden 2005.

[2] E. Cohen, E. Halperin, H. Kaplan, U. Zwick, Reachability and distance queries via 2-hop Labels. *SIAM J. Compt. 32:1338-1355, 2003*.