Exposé zur Diplomarbeit
Hauptspeicherbasierte Bearbeitung von Pfadanfragen an große Graphen

André Koschmieder
Institut für Informatik, Humboldt-Universität zu Berlin
Unter den Linden 6, 10099 Berlin, Germany
koschmie@informatik.hu-berlin.de

Lehrstuhl: Wissensmanagement in der Bioinformatik
Betreuer: Prof. Leser

Oktober 2008

Zusammenfassung

Die Beantwortung von Pfadanfragen an große Graphen ist ein Thema, das in den letzten Jahren mehr und mehr an Bedeutung gewinnt. In der Vergangenheit stand die Datenbank-basierte Bearbeitung im Vordergrund, mit heutigen Hauptspeichergrößen ist dies allerdings keine Voraussetzung mehr.

Im Rahmen dieser Diplomarbeit soll herausgearbeitet werden, wie effizient Pfadanfragen an Graphen im Hauptspeicher beantwortet werden können. Hierbei steht die Suche nach Pfaden, die einen gegebenen regulären Ausdruck matchen, im Mittelpunkt.

1 Motivation

Die Bedeutung von Graphen in der Datenverarbeitung hat sich in den letzten Jahrzehnten stark erhöht. Mit den größer werdenden Fähigkeiten der Computer können durch leisungsfähigere Speichermedien und Prozessoren immer größere und komplexere Graphen verwaltet und verarbeitet werden. Graphen mit vielen Millionen Knoten und Kanten sind heutzutage in vielen Bereichen üblich [1, 5, 13].

Zu den wichtigsten Aufgaben bei der Arbeit mit Graphen gehören die Suche nach Pfaden im Graphen sowie die Frage nach der Erreichbarkeit zwischen Knoten. Die effiziente Suche nach möglichst kurzen Pfaden im Graph ist eine wichtige Aufgabe unter anderem für Navigationssysteme [6, 11] und Internet Routing Protokolle [7, 14]. Erreichbarkeitsanfragen zwischen zwei Knoten sind die zentrale Aufgabe bei der Arbeit mit hierarchischen Strukturen wie der Bearbeitung von XML-Queries [12].

In der Biologie werden Graphen verwendet, um Wechselwirkungen zwischen Stoffen in verschiedenen Prozessen zu modellieren. Die Knoten im Graphen stellen hierbei Gene, Proteine, Enzyme oder andere chemische Substanzen dar, während die Kanten die verschiedenen Reaktionen beschreiben. So können z.B. Gene miteinander in Beziehung gesetzt werden, wenn ein Gen die Aktivierung des anderen Gens verstärkt oder verringert; dies kann als Graph mit gerichteten Kanten dargestellt werden. Ein Pfad in diesem Graphen kann nun einen aktivierenden Gen-Pfad bilden, wenn die Kanten (Beziehungen zwischen den Genen) bestimmte Eigenschaften erfüllen. Um spezielle aktivierende Gen-Pfade im Graphen finden zu können, muss eine Suche nach einem Pfad durchgeführt werden, der bestimmte Bedingungen erfüllt. Diese lassen sich gut durch reguläre Ausdrücke beschreiben, die die Eigenschaften der Kanten erfüllen müssen, oder die bestimmte Reihenfolgen der Knoten oder Knotenklassen beschreiben. Die Suche nach solchen Pfaden ist daher eine wichtige Aufgabe der Bioinformatik.

Graphen, die anhand von biologischen Erkenntnissen erstellt werden, sind häufig sehr groß. So enthält die STRING Datenbank [13] Beziehungen zwischen mehr als 1,5 Millionen Proteinen, die BIND Datenbank [2] mehr als 200.000 Protein Interaktionen und die PubGene Datenbank [5] mehr als 6 Millionen Beziehungen in der Literatur. Daher wurden in der Vergangenheit vor allem Datenbank-bezogene Verfahren für die Suche im Graphen entwickelt; solche Datenmengen konnten bis vor wenigen Jahren nicht im Hauptspeicher abgelegt oder bearbeitet werden. Mit heutigen Hauptspeichergrößen von dutzenden Gigabytes schon bei mittleren Servern lassen sich aber auch sehr große Graphen im Hauptspeicher verwalten, was einen potentiellen Geschwindigkeitsgewinn mit sich bringt.

Im Rahmen dieser Diplomarbeit soll daher der mögliche Geschwindigkeitsgewinn Hauptspeicher-basierter Verfahren gegenüber der Datenbankimplementierung bewertet werden. Es soll auch der Frage nachgegangen werden, bis zu welcher Größe sich die Graphen im Hauptspeicher verwalten lassen, und wann Datenbank-basierte Verfahren sinnvoller einsetzbar sind.

2 Verwandte Arbeiten

Wie Mendelzon und Wood zeigen, ist die Suche nach kreisfreien Pfaden in Graphen, die regulären Ausdrücken entsprechen, NP-vollständig [9]. Sie führen weiter aus, dass eine Lösung in polynomialer Zeit möglich ist, wenn der Graph oder der reguläre Ausdruck kreisfrei sind, oder der reguläre Ausdruck bestimmten Beschränkungen unterliegt, oder der Graph und der reguläre Ausruck konfliktfrei sind (kein Knoten kann zweimal erreicht werden bei Traversierung des Schnittautomaten). Es gibt daher eine Reihe von Publikationen, die sich mit praktischen Verfahren zur Optimierung dieser Suche beschäftigen.

Nestorov et al. schlagen einen Index in Form von *Representative Objects* [10] vor, die Strukturinformationen über die Daten im Graphen speichern und damit Suchanfragen optimieren können. Allerdings wird hier davon ausgegangen, dass immer ein Wurzelknoten existiert, bei dem die Suche begonnen werden soll. Eine praktische Anwendung davon sind *DataGuides* von Goldman und Widom [3], bei denen sie einen minimalen Index erstellen ähnlich der Vorgehensweise der Überführung eines NFAs in einen DFA. Diese Vorgehensweise ist allerdings nur in kreisfreien Graphen sinnvoll anwendbar. Auch die vorgeschlagene approximative Erweiterung [4] ist für große zyklische Graphen nicht gut geeignet.

Liske stellt in seiner Diplomarbeit [8] das Verfahren der *k-Minimierung* vor, bei dem der Graph zunächst durch heuristische Verfahren minimiert wird; hierbei werden Knoten aufgrund lokaler Eigenschaften zusammengefasst und der reguläre Ausdruck zunächst auf dem kleineren Indexgraphen ausgewertet. Dazu wird der Graph zunächst in einen DFA überführt und minimiert, und anschließend der Schnittautomat berechnet und ausgewertet. Die Implementierung ist für die Datenbank-bezogene Ausführung ausgelegt.

3 Ziele der Arbeit

Im Rahmen der Diplomarbeit soll untersucht werden, welche Vor- und Nachteile eine Hauptspeicher-basierte Implementierung der Suche nach Pfaden im Graphen, die durch reguläre Ausdrücke beschrieben werden, gegenüber Datenbank-basierten Lösungen bietet. Da zu erwarten ist, dass die Suche im Hauptspeicher deutlich schneller abläuft als in der Datenbank, soll der Frage nachgegangen werden, bis zu welcher Größe Graphen auf diese Art sinnvoll im Hauptspeicher zu handhaben sind.

Zu diesem Zweck sollen Hauptspeicher-basierte Algorithmen in JAVA implementiert werden, die auf einem gegebenen Graphen mit Knoten- oder Kantenlabeln Anfragen beantworten können, bei denen Pfade im Graphen gesucht werden sollen, die anhand von regulären Ausdrücken beschrieben werden. Da für die Vorgehensweise kein grundsätzlicher Unterschied zwischen durch Knoten- und Kantenlabeln beschriebenen Pfaden besteht, wird in der Arbeit nur von Kantenlabeln ausgegangen. Folgende Anfragen an den Graphen sollen beantwortet werden können: Sei R ein gegebener regulärer Ausdruck.

- Finde den kürzesten Pfad, der durch R beschrieben wird, zwischen beliebigen Knoten im Graphen.

- Finde den kürzesten Pfad, der durch R beschrieben wird, beginnend (endend) beim Knoten K (einem Knoten der Knotenklasse K).
- Finde den kürzesten Pfad zwischen den Knoten A und B (zwischen beliebigen Knoten der Knotenklassen A und B), der durch R beschrieben wird.

Die Algorithmen sollen mit den Datenbank-basierten Implementierungen bzgl. Laufzeitverhalten und Speicherverbrauch verglichen werden. Es soll zudem herausgearbeitet werden, bei welcher Größenordnung der Anzahl von Knoten und Kanten die Möglichkeiten heutiger Hauptspeichergrößen erschöpft sind.

Literatur

[1] 9th DIMACS Implementation Challenge: Shortest Paths, 2006. http://www.dis.uniroma1.it/ challenge9.

[2] G. D. Bader, I. Donaldson, C. Wolting, B. F. Ouellette, T. Pawson, and C. W. Hogue. Bind–the biomolecular interaction network database. *Nucleic Acids Res*, 29(1):242–245, January 2001.

[3] R. Goldman and J. Widom. Dataguides: Enabling query formulation and optimization in semistructured databases. In *Proceedings of 23rd International Conference on Very Large Data Bases*, pages 436–445, 1997.

[4] R. Goldman and J. Widom. Approximate dataguides. In *Proceedings of the Workshop on Query Processing for Semistructured Data and Non-Standard Data Formats*, 1999.

[5] T. Jenssen, A. Laegreid, J. Komorowski, and E. Hovig. A literature network of human genes for high-throughput analysis of gene expression. *Nat Genet*, 28(1):21–8, 2001.

[6] E. Köhler, R. H. Möhring, and H. Schilling. Acceleration of shortest path and constrained shortest path computation. In *Experimental and Efficient Algorithms, 4th International Workshop*, pages 126–138, 2005.

[7] D. Krioukov, K. Fall, and X. Yang. Compact routing on internet-like graphs. *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, 1:–219, March 2004.

[8] P. Liske. Auswertung regulärer Ausdrücke in Graphen. Master's thesis, Humboldt-Universtät zu Berlin, Germany, November 2007.

[9] A. O. Mendelzon and P. T. Wood. Finding regular simple paths in graph databases. *SIAM Journal on Computing*, 24(6):1235–1258, 1995.

[10] S. Nestorov, J. Ullman, J. Wiener, and S. Chawathe. Representative objects: Concise representations of semistructured, hierarchical data. In *Proceedings of the Thirteenth International Conference on Data Engineering*, pages 79–90, 1997.

[11] P. Sanders and D. Schultes. Engineering fast route planning algorithms. In C. Demetrescu, editor, *WEA*, volume 4525 of *Lecture Notes in Computer Science*, pages 23–36. Springer, 2007.

[12] R. Schenkel, A. Theobald, and G. Weikum. HOPI: An efficient connection index for complex XML document collections. In *EDBT*, pages 237–255, 2004.

[13] C. von Mering, M. Huynen, D. Jaeggi, S. Schmidt, P. Bork, and B. Snel. STRING: a database of predicted functional associations between proteins. *Nucleic Acids Res*, 31(1):258–61, 2003.

[14] E. W. Zegura, K. L. Calvert, and M. J. Donahoo. A quantitative comparison of graph-based models for internet topology. *IEEE/ACM Trans. Netw.*, 5(6):770–783, 1997.