Exposé zur Studienarbeit

Kürzeste Pfade mit GRIPP

Leonid Snurnikov

26. November 2007

Betreuung: Prof. Dr. Leser

Institut für Informatik
Humboldt Universität zu Berlin

1 Einleitung

Viele Probleme im Zusammenhang mit Netzwerken, Ontologien, Taxonomien und anderen Arten von Systematiken können auf Problemstellungen auf Graphen abgebildet werden. In der Bioinformatik findet man zum Teil sehr große Graphen, die von Datenbank Management Systemen verwaltet werden und viele Tausende Knoten umfassen. Sehr häufige Problemstellungen in diesem Zusammenhang sind die Fragen nach der Erreichbarkeit eines Knotens B von einem anderen Knoten A sowie nach dem kürzesten Pfad zwischen diesen beiden. Um effizient solche Anfragen beantworten zu können, werden spezielle Indexstrukturen mit entsprechend optimierten Anfragealgorithmen benötigt. Zu den sich teils widersprechenden Anforderungen zählen dabei eine möglichst kleine Indexgröße, schnelle Erzeugung der Indexstruktur, schnelle Ausführung von Anfragen sowie eine gute Skalierbarkeit mit der Größe des Graphen. Von besonderem Interesse sind dabei Algorithmen für allgemeine gerichtete Graphen.
Es existieren bereits einige Lösungen für die oben genannten Fragestellungen [1]. Eine davon ist GRIPP (graph indexing based on pre-/postorder numbering). GRIPP wendet das von Bäumen bekannte Konzept der Pre- und Postorder Nummerierung auf allgemeine gerichtete Graphen an, um effizient Erreichbarkeitsanfragen beantworten zu können. Eine große Stärke von GRIPP liegt darin, dass der Index auch für sehr große Graphen in vertretbarer Zeit erzeugt werden kann, wobei die durchschnittliche Anfragedauer konkurrenzfähig bleibt.

2 Ziel

Ziel dieser Studienarbeit ist Anwendung der GRIPP-Indexstruktur für die effiziente Beantwortung von Anfragen nach dem kürzesten Pfad zwischen zwei gegebenen Knoten A und B in einem gerichteten Graph. Wenn es mehrere solche Pfade gibt, sind alle von Interesse. Der Schwerpunkt liegt jedoch auf dem Auffinden eines Vertreters. Um die Suchtiefe zu begrenzen und damit die Suche zu beschleunigen wird zusätzlich eine obere Schranke für die Länge des kürzesten Pfades benötigt. Dazu soll eine weitere Indexstruktur erzeugt werden, mit deren Hilfe möglichst schnell eine *gute* Schranke ermittelt werden soll. *Gut* bedeutet dabei entweder gleich die Länge des kürzesten Pfades oder zumindest eines kurzen Pfades.

3 Vorgehensweise

GRIPP baut zunächst einen spannenden Baum auf dem Graph G=(V,E) auf. Dabei bleiben einige Kanten (Nichtbaumkanten) zunächst außer Betrachtung. Sie werden anschliessend dadurch berücksichtigt, dass der spannende Baum um virtuelle Knoten (Nichtbauminstanzen) erweitert wird. Die Nichtbauminstanzen bilden Nichtbaumkanten ab und können als Verweise (*hop nodes*) auf die eigentlichen Knoten (Bauminstanzen) angesehen werden.
Der so erzeugte GRIPP-Indexbaum wird im Rahmen einer Tiefensuche mit Pre- und Postnummern versehen und als Relation in der Datenbank abgelegt. Bei der Beantwortung von Erreichbarkeitsanfragen unter Verwendung des GRIPP-Indexbaums werden unter anderem sogenannte RIS-Mengen (*reachable instance set*) betrachtet. Die RIS-Menge zu einem Knoten A ist die Menge der Nachfahren des Knotens A im GRIPP-Indexbaums. Wenn sich der gesuchte Knoten B in der RIS-Menge des Knotens A befindet, kann das leicht anhand der Pre-/Postnummerninklusion festgestellt werden, da dann gilt: $pre(A) < pre(B) < post(A)$. Falls sich der Knoten B nicht in der RIS-Menge des Knotens A befindet, die RIS-Menge jedoch Nichtbauminstanzen (*hop nodes*) enthält, so muss rekursiv den Verweisen gefolgt werden. In dem GRIPP-Algorithmus wird dies in der Reihenfolge der Tiefensuche vollzogen [1]. Da es vor allem in großen Graphen sehr viele Pfade von A nach B geben kann, wobei viele davon auch noch wesentlich länger als der kürzeste Pfad sind, wäre zumindest eine ungefähre Abschätzung für die Länge des kürzesten Pfades hilfreich, um die Rekursion vorzeitig abbrechen zu können. Mit einer dem Algorithmus übergebenen oberen Schranke r für die Länge des kürzesten Pfades müsste der GRIPP-Anfragealgorithmus alle Rekursionen bis maximal Länge r durchlaufen, um dann aus allen gefundenen Pfaden den kürzesten auszuwählen. Als Optimierung könnte er beim Fund eines kürzeren Pfades als r die obere Schranke für die restliche Suche entsprechend anpassen.
Alternativ könnte der GRIPP-Algorithmus ohne Angabe einer oberen Schranke die Suche (auch über die aktuelle RIS-Menge hinaus) in Reihenfolge der Breitensuche durchführen.

Es bleibt zu klären, wie die Abschätzung einer oberen Schranke für die Länge des kürzesten Pfades realisiert wird. Dazu könnten zunächst zufällig k Knoten (Menge $K \subseteq V$ mit $|K| << |V|$) aus der Gesamtmenge ausgewählt werden. Für alle berechnet man zum Beispiel durch Breitensuche paarweise die kürzesten Pfade und speichert diese in der Datenbank in Form einer Relation. Bei der Anfrage nach dem kürzesten Pfad von A nach B könnten nun folgende Fälle unterschieden werden:

- A und B sind beide Teil des Index. Somit ist der kürzeste Pfad bereits gefunden.
- $(A, B) \in E$. Dann ist der kürzeste Pfad ebenfalls klar.
- A, B oder beide sind nicht Teil des Index. Dann könnte in der Nachbarschaft nach $A', B' \in V$ gesucht werden mit $(A, A') \in E$ bzw. $(B', B) \in E$. Wenn
  - $(A', B') \in E$, dann ist die obere Schranke für GRIPP $s = 3$.
  - $A', B' \in K$, dann ist die obere Schranke für GRIPP $s = 2 + d(A', B')$, wobei $d(A', B')$ als die Länge des kürzesten Pfades zwischen A', B' dem Index entnommen werden kann.

Wenn es keine direkten Nachfolger A', B' mit $A', B' \in K$ oder $(A', B') \in E$ gibt, könnte man die Nachfolger mit einer Entfernung von 2 betrachten oder alternativ mit GRIPP wie oben bereits erwähnt ohne Einschränkung der Rekursionstiefe eine Breitensuche durchführen.
Um die Größe des Index zu verringern, können statt der kürzesten Pfade zwischen den Knoten aus K lediglich die wechselseitig kürzesten Abstände (Länge kürzester Pfade) gespeichert werden. Dabei muss allerdings zur Abfragezeit im Fall $A, B \in K$ trotzdem die GRIPP-Suche durchgeführt werden, um den eigentlichen Pfad zu ermitteln.
Die Routine zur Indexerstellung sowie der Algorithmus zum Finden einer oberen Schranke für die Länge des kürzesten Pfades sollen als Stored Procedures für das Oracle DBMS implementiert werden. Dazu würde sich PL/SQL oder Java eignen. Die Indexstruktur selber wird in Form einer Relation gespeichert. Beides ermöglicht das Ausnutzen der Speicherverwaltungsinstrumente von Oracle und das Arbeiten mit großen Graphen.

Literatur

[1] Silke Trißl and Ulf Leser. Fast and practical indexing and querying of very large graphs. *SIGMOD*, 2007.