Dynamisch verteilte Wörterbücher

Author(s):
Kröll, Brigitte

Publication Date:
1997

Permanent Link:

https://doi.org/10.3929/ethz-a-001744553 →

DISS. ETH Nr. 12026

Dynamisch verteilte Wörterbücher

ABHANDLUNG
zur Erlangung des Titels
Doktor der Technischen Wissenschaften
der
EIDGENÖSSISCHEN TECHNISCHEN HOCHSCHULE
ZÜRICH

vorgelegt von
BRIGITTE KRÖLL
Diplom–Mathematikerin Universität Freiburg i. Br.
geboren am 4. März 1967
in Heilbronn

angenommen auf Antrag von
Prof. Dr. Peter Widmayer, Referent
Prof. Dr. Eljas Soisalon Soininen, Korreferent
1997

Kurzfassung

Ein Kernproblem der Informatik ist die Verwaltung von Daten und die Manipulation derselben. Eine prominente Variante davon ist das Wörterbuchproblem. Dabei sollen linear geordnete Daten so verwaltet werden, daß die Operationen einfügen, suchen und entfernen effizient durchführbar sind. Die zunehmende Vernetzung von Computern und das Anwachsen großer Datenbestände erfordert, dieses Problem auch in verteilten Systemen zu lösen. Die vorliegende Arbeit stellt mehrere Datenstrukturen für das verteilte Wörterbuchproblem vor, untersucht ihren theoretischen Hintergrund und wertet die Implementierung einer dieser Datenstrukturen aus.

Das Wörterbuchproblem wurde in den letzten Jahrzehnten gründlich untersucht. Dabei ging man zunächst von Daten im Internspeicher und dann auch von Daten im Externspeicher aus, die immer durch einen Prozessor verwaltet wurden. Heute, im Zeitalter der immer schnelleren Netze, würden wir die Daten eines großen Wörterbuches gerne auf mehrere Server verteilen. Ein Schema für solch eine Verteilung sollte möglichst skalierbar sein, um sich an die Größe des Datenbestandes anpassen zu können. Da wir voraussetzen werden, daß jeder Server eine feste Anzahl Speicherplätze zur Verfügung stellt, variiert beim Wachsen und Schrumpfen des Datenbestandes die Anzahl der beteiligten Server über die Zeit. Diese Dynamik macht das Problem interessant, da die anfragenden Rechner in irgendeiner Form und mit möglichst wenig Kommunikation über Veränderungen informiert werden sollen. Dabei möchte man eine zentrale Organisation der Daten jedoch vermeiden. In dieser Arbeit werden mehrere verteilte Datenstrukturen für dieses Problem entwickelt und analysiert.

Zunächst beschreiben wir das von Litwin et al. vorgestellte *verteilte lineare Hashing*, das das verteilte Wörterbuchproblem löst. Dann entwickeln wir Datenstrukturen, die zusätzlich die Ordnung der Daten erhalten, und dadurch auch Anfragen erlauben, die sich auf diese Ordnung beziehen, wie etwa Bereichsanfragen. Dazu untersuchen wir *verteilte Suchbäume*. Aus dem natürlichen Binärbaum im klassischen Fall entwickeln wir eine verteilte Datenstruktur, den *verteilten natürlichen Binärbaum*. Diese Datenstruktur werden wir genau untersuchen.

Wie der klassische natürliche Baum kann auch der verteilte natürliche Baum entarten. Für den klassischen Fall wissen wir, daß Balancierung der Bäume solch eine Entartung verhindern kann und logarithmische Höhe der Bäume garantiert. In dieser Arbeit wird gezeigt, daß solch eine Balancierung für verteilte Bäume nicht genauso möglich ist. In der Klasse der *stabilen* verteilten Bäume, einer Verallgemeinerung der klassischen Bäume, kann man die

Höhe der verteilten Bäume im schlechtesten Fall nicht bestenfalls durch eine Funktion in der Größenordnung der Wurzelfunktion beschränken. Die untere Schranke für die Höhe ist scharf, denn wir entwickeln einen stabilen verteilten Baum, der diese Schranke auch als obere Schranke für die Höhe besitzt.

Über dieses Ergebnis hinaus werden wir *verteilte B-Bäume* studieren, die nicht in die Klasse der stabilen Bäume gehören. Diesen verteilten B-Bäumen fehlen gewisse Dezentralitätseigenschaften, sie garantieren aber einen logarithmischen schlechtesten Fall für die Wörterbuchoperationen.

Abschließend werten wir eine Simulation des verteilten natürlichen Binärbaumes experimentell aus. Diese Experimente ergänzen unsere theoretischen Analysen. Sie bestätigen die Effizienz des verteilten natürlichen Baumes.

Abstract

A fundamental problem in computer science is the storage and handling of data. One famous version of this problem is the dictionary problem: Linearly ordered data need to be stored in a way that the operations insert, search and delete can be performed efficiently. Since the connectivity of computers increases and the amount of data to be stored grows, we should solve this problem also in distributed systems. This thesis proposes several data structures for the distributed dictionary problem, investigates their theoretical background and evaluates an implementation of one of these data structures.

The dictionary problem has been well-studied in the last couple of decades. It was first solved for data stored in internal memory and later for data stored in external memory. In both cases data is managed by a single processor. Today's faster networks allow to distribute the data of a very big dictionary among several servers. How can we do that? A scheme for such a distribution should be as scalable as possible, so that it is adaptable to the amount of stored data. We will assume that memory for a fixed number of data elements is available at each server. This means that the number of participating servers changes with the amount of data. This dynamic process is the interesting part of the problem, because the sites requesting data have to be informed about the changes in the distribution. Site updates should involve as little communication as possible and not rely on a central entity. This thesis proposes and evaluates several distributed data structures for the problem.

First, we describe the *distributed linear hashing*, which solves the dictionary problem, as proposed by Litwin et al. We then propose data structures which, in addition, maintain the order of the data. They allow requests referring to this order, such as range queries. In particular, we consider *distributed trees* to which we apply the classic concept of random trees to obtain the *distributed random tree*, which we then describe in detail.

Like the classic random tree the distributed random tree can degenerate. For the classic case we can prevent the degeneration by balancing the tree and at the same time guarantee a logarithmic height. However, we will prove that the analogous result does not hold in the distributed case. In the class of *stable* distributed trees, a generalisation of classic trees, we prove that in the worst case the height of a distributed tree cannot be bounded more tightly than the square root of the number of nodes in the tree.

In order to guarantee a logarithmic worst case for the dictionary operations we developed *distributed B-trees*. However, they are not stable and lack

Höhe der verteilten Bäume im schlechtesten Fall nicht bestenfalls durch eine Funktion in der Größenordnung der Wurzelfunktion beschränken. Die untere Schranke für die Höhe ist scharf, denn wir entwickeln einen stabilen verteilten Baum, der diese Schranke auch als obere Schranke für die Höhe besitzt.

Über dieses Ergebnis hinaus werden wir *verteilte B-Bäume* studieren, die nicht in die Klasse der stabilen Bäume gehören. Diesen verteilten B-Bäumen fehlen gewisse Dezentralitätseigenschaften, sie garantieren aber einen logarithmischen schlechtesten Fall für die Wörterbuchoperationen.

Abschließend werten wir eine Simulation des verteilten natürlichen Binärbaumes experimentell aus. Diese Experimente ergänzen unsere theoretischen Analysen. Sie bestätigen die Effizienz des verteilten natürlichen Baumes.

some decentrality properties.

Finally we evaluate a simulation of the distributed random tree. These experiments round off our theoretical analysis. They confirm the efficiency of the distributed random tree.