Dynamic Load Balancing in Peer-to-Peer Networks

Dissertation of Miroslaw Korzeniowski

November 2005

Abstract

In the last few years Peer-to-Peer networks have become very popular both in everyday internet use and academic community. A very important concept in this area are so called Distributed Hash Tables which are used to publish and find information in a distributed way. They are usually based on a virtual space into which all data is hashed. The whole space is partitioned among the peers participating in the network and each peer is responsible for the items hashed into its piece of the space. In this thesis only Distributed Hash Tables based on a ring topology are considered.

In case when peers choose their places on the ring uniformly at random, the ratio between the longest and the shortest interval assigned to a peer is $\Theta(n \log n)$ with probability almost 1. The first part of this thesis gives algorithms which assure that all peers are assigned pieces of the ring of the same size up to a constant factor. The schemes work in distributed fashion and balance the load even in a dynamic environment. Two algorithms are given, one of which balances a static system within logarithmic time but has high communication cost and another one which works properly in a dynamic environment and has bounded communication cost but balances the system with some delays.

The second part shows how to use the existing concepts as a basis for distributed data structures. It is shown how to implement a distributed binary search tree so that it is both balanced and resistant to failures. A balanced binary search tree is embeded into a network which has both de Bruijn and hypercubic connections. The de Bruijn connections implement the tree edges, whereas the hypercubic connections are used in rotations to assure the balance of the tree. Through a special wake-up mechanism in the underlying network resilience to adversarial faults is achieved. In case of failures in the network, some data is lost but the tree rebuilds itself so that it is still a valid data structure. Such recovery works even if only one node survives the failure.