HEURISTICS ALGORITHMS FOR BROADCASTING IN CACTUS GRAPHS

Neil Conlan

A thesis

in

The Department

of

Computer Science

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Computer Science

Concordia University

Montréal, Québec, Canada

September 2015

Rama Bhat, Ph.D.,ing., FEIC, FCSME, FASME, Interim Dean

Faculty of Engineering and Computer Science

Abstract

Heuristics Algorithms for Broadcasting in Cactus Graphs

Neil Conlan

This is the Abstract text.

Acknowledgments

This is the acknowledgments text.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since the early days of compting, computers have been used to solve problems in speeds that are unatainable by humans. Initially computers where designed using a single processor. While software executing using a single processor was great at solving problems in a sequantial way, the need for higher speeds was needed.

Multi-computer and multi-processor systems (often called distributed computing) was a solution to this speed problem. While designing software to run on a distributed system introduced more complexity to the software, thus making it more difficult to design software, the speedup of solving problems this way was substantial enough to accept the increase in complexity.

Distributed computing works by breaking down a large problem thats needs to be solved into smaller independant problems that can be solved in parallel and then merging the results to obtain the final solution to the large problem. In some cases the processors of the distributed system need to share information with each other. This can be accomplished using shared memory that all processors have access to or each processor can have it's own local address space (distributed memory). Shared memory systems have limitations on the number of processors that can be connected together, which makes it not practical for very large problems that require a large amount of processors.

Solving problems using distributed memory systems have some advantages over the shared memory model. One advantage is that there is no limitation on the number of processors that can be used to solve a problem. Another advantage is that each processor has it's own memory pool which can be used to fit the data of the smaller problem it is tasked to solve fully inside memory. Since each processor has it's own memory the total amount of memory can be larger than the shared memory model.

Processors of the interconnected network often have to share information with each other. This is accomplished by sending data over the network. It turns out that not only the power of the individual processors is required to solve a given problem but also the speed at which processors can disseminate information over the network. In recent years a lot of work has gone into studying properties of interconnected networks in order to find the best network structures for communication between processors of a network.

There are different types of communication primitives a network can use when data needs disseminated to other processors. These communication primitives are:

- **Routing** or one-to-one communication.
- **Broadcasting** or one-to-all communication.
- **Multicasting** or one-to-many communication.
- **Gossiping** or all-to-all communication.

One of the most fundemental and interesting dissemination problems is broadcasting. *Broadcasting* is when one node of a network, called the *originator*, has data that it wants to share with all other nodes of that network. This is accomlished by placing a series of calls over the communication lines of the given network. Once a node is informed, the informed node can help the originator in distributing the data. These calls are assumed be be performed in discrete time units. The broadcasting of this data should be finished as quickly as possible, subject to the following constraints:

- Each call involves one of the informed nodes with one of its uninformed neighbors nodes.
- Each call requires one unit of time.
- A node can only participate in one call per unit of time.
- In one unit of time each informed node can work in parallel.

Formally, any network can be modelled as a connected gragh $G = (V, E)$, where V is the set of vertices (or nodes) and E is the set of edges (or communication lines) between the vertices of the gragh G. Two vertices $u, v \in V$ are said to be *adjacent* (or neighbors) if there is an edge $e \in E$, such that $e = (u, v)$. The *degree* of a vertex u, $d(u)$ or $deg(u)$, is defined as the number of incident of that vertex. The *degree* of a graph G is the maximum degree of all vertices of the graph, $max\{deg(v)|v \in V\}$. The shortest path between a vertex u and a vertex v is called the *distance* between u and v, and is denoted by $dist(u, v)$. The *diameter* of a graph G is the maximum distance between and two vertices of the graph, $max\{dist(u, v)|u \in V, v \in V)\}$.

A *broadcast scheme* of a graph with origining vertex u is defined as the set of calls performed to complete the broadcasting in the network. The broadcast time from originating vertex u, $b(u, G)$ or just $b(u)$, is the minimum number of time units (or rounds) required to complete the broadcast from vertex u. From any originating vertex u it is clear that the minimum number of rounds required to complete the broadcast is $b(u) \geq \lceil \log n \rceil$ since at best the number of informed nodes each round can double. The maximum number of rounds required to broadcast on a network is $b(u) = n - 1$ since in the worst case there will be one newly informed verex. The broadcast time of a graph G, is defined as $max\{b(u)|u \in V\}$.

1.1 Section Title Goes Here

Paragraph 1 Paragraph 1 Paragraph 1 Paragraph 1 Paragraph 1 Paragraph 1 Paragraph 1 Paragraph 1 Paragraph 1 Paragraph 1 Paragraph 1 Paragraph 1

1.1.1 Classes and Objects

Sub section Sub section Sub section Sub section Sub section Sub section Sub section Sub section Sub section Sub section Sub section Sub section Sub section Sub section Sub section Sub section Sub section Sub section