行政院國家科學委員會專題研究計畫　成果報告

以代理機制為基礎之交換機制網路多廣播排程演算法

計畫類別：個別型計畫
計畫編號：NSC92-2213-E-002-060-
執行期間：92 年 08 月 01 日至 93 年 07 月 31 日
執行單位：國立臺灣大學資訊工程學系暨研究所

計畫主持人：劉邦鋒

計畫參與人員：吳貞貞，林藝芳

報告類型：精簡報告

處理方式：本計畫可公開查詢

中　華　民　國 93 年 12 月 20 日

Efficient Agent-based Multicast on Wormhole Switch-based Irregular Networks

Pangfeng Liu, Jan-Jan Wu and Yi-Feng Lin, "Efficient agent-based multicast on wormhole switch-based irregular networks," *the 17th International Parallel and Distributed Processing Symposium*, (IPDPS), 2003.

Abstract

This paper describes an agent-based approach for scheduling multiple multicasts on wormhole switch-based networks. Multicast/broadcast is an important communication pattern, with applications in collective communication operations such as barrier synchronization and global combining. Our approach assigns an agent to each subtree of switches such that the agents can exchange information efficiently and independently. The entire multicast problem is then recursively solved with each agent sending message o those switches that it is responsible for. In this way, communication is localized by the assignment of agents to subtrees. This idea can be easily generalized to multiple multicasts since the order of message passing among agents can be interleaved for different multicasts. We conduct experiments to demonstrate the efficiency of our approach by comparing the results with SPCCO, a highly efficient multicast algorithm. We found that SPCCO suffers link contention when the number of simultaneous multiple multicast becomes large. On the other hand, our agent-based approach achieves better performance in large cases.

摘要

本論文提出一以代理人機制在蟲洞通訊網路中處理多重一對多傳輸的演算法. 多重一對多傳輸是一種非常重要的通訊模式, 其應用範圍包括同步及全域資訊整合. 我們所提出的機制對美一通訊子樹指定一代理人, 並由代理人以遞迴方式負責將資訊作有效的轉送. 如此通訊將被侷限在子樹中已達成有效通訊之目的. 我們實作此演算法並與文獻中之SPCCO作比較. 實驗結果顯示SPCCO在多重一對多傳輸的數目增大時會出現嚴重網路連線壅塞, 而我們的代理人演算法則仍能發揮較佳的效能.

Keyword: multiple multicasts, wormhole switch-based networks, up-down routing

1 Introduction

Multicast/broadcast is commonly used in many scientific, industrial, and commercial applications [1]. Distributed memory parallel systems require efficient implementations of multicast and broadcast operations in order to support various applications. In recent years, with the speed of microprocessors increasing and cost decreasing and the availability of high bandwidth, low latency switches (such as Fast Ethernet switches, Myrinet switches, ATM switches, Servernet switches) at a reasonable cost, it is popular to interconnect workstations/PCs together with commodity switches. This makes clusters of workstations/PCs an appealing vehicle for cost-effective parallel computing. To reduce communication latency and buffer requirement, wormhole switching technique [4, 15] is often used in these switches. Systems with wormhole routing provide a very small buffer space at each hop and divide a message into small flits that travel through the network in a pipeline fashion. The main drawback of wormhole switching is that blocked messages hold up the links, prohibiting other messages from using the occupied links and buffers. In a multicast, the source node sends the same data to an arbitrary number of

destination nodes. When multiple multicast operations occur at the same time, it is very likely that some messages may travel through the same link at the same time and thus content with each other, if they are not scheduled properly. Minimizing contention in collective communication has been extensively studied for systems with regular network topologies, such as mesh, torus and hypercubes [3, 5, 6, 10, 9, 11, 16]. Switch-based networks, on the other hand, typically have irregular topologies to allow the construction of scalable systems with incremental expansion capability. These irregular topologies lack many of the attractive mathematical properties of the regular topologies. This makes routing on such systems quite complicated. In the past few years, several deadlock-free routing algorithms have been proposed in the literature for irregular networks [2, 7, 12, 17]. These routing algorithms are quite complex and thus make implementation of contention-free multicast operations very difficult. The goal of this paper is to develop efficient (multiple) multicast algorithms for irregular wormhole switch-based networks. In [8], Fan and King proposed a unicast-based implementation of single multicast operation based on Eulerian trail routing. In this paper, we consider the widely used, commercially available deadlock-free routing strategy called "up-down" routing. Kesavan and Panda proposed a series of single and multiple multicast algorithms [13]. The basic idea is to order the destination processors into a sequence, then apply a binomial tree-based multicast [14] on these destinations. The chain concatenation ordering (CCO) algorithm first constructs as many partial order chains (POC) as possible from the network. A partial order chain is a sequence of destinations such that we can apply a binomial multicast on it without any contention. The CCO algorithm then concatenates these POCs into sequence where a binomial multicast is performed [13]. The sequence consists of fragments of processor sequences in which messages within the same fragment can be sent independently, therefore congestion is reduced. Based on the CCO algorithm, the source partitioned CCO (called SPCCO) performs multiple multicasts simultaneously. Each multicast produces its own sequence (consisting of POCs), and each resulting sequence is shifted until the source appears at the beginning of the sequence. By shifting these sequences, the communication is "interleaved" according to the source, and communication hot-spots are avoided. Both CCO and SPCCO use the idea of POC to reduce contention. Within a single POC different messages do not interfere with one another as long as they are from different sections within a POC. However, this POC structure may not always be preserved since the later binomial multicast is not aware of it. Our agent-based algorithm deals with this issue by localization and interleaving. For a single multicast, our algorithm uses a recursive construct to localize communication. We then generalize it to multiple multicasts by interleaving the communication tasks among different subnetworks.

Our agent-based approach starts with a recursive multicast algorithm. An agent for a multicast is chosen for each subtree of the up-down routing tree. An agent is responsible for relaying the multicast messages to all the destinations in that subtree. This task is divided into subtasks for each subtree, where they are performed recursively. We generalize this algorithm to multiple multicasts by choosing a primary agent for each multicast. The primary agents are chosen from the subtrees of the root of the routing tree, and are properly interleaved so that the tasks are distributed evenly. The primary agents for different multicasts exchange messages and then use the multicast algorithm to propagate messages. Depending on how primary agents are chosen and

how the information are exchanged among the primary agents, our agent-based multiple multicast algorithm has four variations and have be described in detail in IPDPS paper.

2 Model

The connectivity of switches in the network can be represented by a graph G = (V;E), where the set of nodes V represents switches, and the set of edges E represents the bidirectional connection channels among switches. The graph G can be highly irregular. In addition, each processor is connected to a unique switch.

2.1 Routing Mechanism

We now describe the up-down routing [7] used in our multiple multicast algorithm. The up-down routing mechanism first uses a breadth-first search to build a spanning tree T for the switch connection graph G = (V;E). Since T is a spanning tree of G, E is partitioned into two subsets – T and E _ T. Those edges in T are referred to as tree edges and those in E _ T as cross edges [13]. Since the tree is built with a BFS, the cross edges can only connect switches whose levels in the T differ by at most 1. A tree edge going up the tree, or a cross edge going from a processor with a higher processor id to a processor with a lower one, are referred to as up links. The communication channels going the other direction are down links. In up-down routing a message must travel all the up links before it travels any down links. Due to the acyclic nature of how the directions of links are defined, the up-down routing is deadlock-free.

3 Agent-Based Algorithms

We first introduce the algorithm for single multicast, and then generalize the idea to multiple multicasts. Note that our algorithms assume the up-down routing mechanism. The algorithms specify how to perform a single/multiple multicast by determining the source and destination of all the intermediate communications, but the actual route from source to destination is determined by the up-down routing.

3.1 Single Multicast

For a given irregular network, we first construct a routing tree as in up-down routing [7]. The routing tree has all the switches as the tree nodes, and the inter-switch communication channels as the tree edges. Every tree node is the root of a unique subtree in this routing tree, and for ease of notation we will not distinguish a tree node (a switch in the network) from the subtree where it is the root. For a given multicast message m and a switch v we will define two functions – an agent function A(m; v) that returns a processor within the subtree rooted at v and will be responsible for relaying multicast message m, and a cost function C(m; v) that estimates the total cost of sending m to all of its specified destinations within the subtree rooted at v. We define these agent and cost functions recursively. Let D(m; v) be the set of destination processors of message m that are connected to switch v. First we consider the case where v is a leaf in the routing tree, then A(m; v) is defined to be an arbitrary destination processor in D(m; v), and the cost function C(m; v) is log ｜D(m; v) ｜. If ｜D(m; v) ｜ is 0, that is, m does not have any destination attached to switch v, we define A(m; v) to be an empty set and C(m; v) = 0. We now consider the agent and the cost function for an internal node v in the routing tree. The agent function for v is defined as follows: If ｜D(m; v) ｜> 0, we pick an arbitrary destination of m in D(m; v) to be A(m; v). Otherwise we consider all the children of v that m must be sent to, and set A(m; v) to be the agent from these subtrees that has the highest cost. Formally, let S(m; v) be the set of children of v that have destinations of m in their subtrees, then A(m; v) = w such that w 2

S(v) and C(m;w) _ w0 for all w0 2 S(v) . Note that from this definition the agent of a switch is not necessarily connected to the switch itself.

The cost function for an internal node is defined as follows: For the purpose of recursion we assume that the agent of v knows the message m. If ｜D(m; v)｜is 0, the agents of tree nodes from S(v) will first perform a multicast among themselves using a binomial multicast [14], then as soon as an agent a from S(m; v) finishes receiving m, it recursively performs a multicast to all the destinations in the subtree where it is defined as the agent. The total communication cost is then defined as C(m; v). When ｜D(m; v)｜>0, the situation is more complicated since the agent of v can send m to other destinations in D(m; v), or to the agents of S(m; v). We apply a procedure ForwardInSwitch that determines the order for those in D(m; v) and S(m; v) to receive messages. After the schedule is fixed we compute the total cost C(m; v) for v.

When ｜D(m; v)｜> 0, v does have some destination processors for message m and one of them is the agent of v. When the agent sends messages to those destinations in D(m; v), the messages will not interfere with each other. Also when the agent of v sends messages to those agents in S(m; v), no contention is possible if no cross edges are involved. In addition, the message passing from one category will not contend with those in the other category. When ｜D(m; v)｜= 0, we use a single multicast to send the messages among all the agents of S(m; v), with one of them now being assigned as the agent of v. We conclude that these messages will not contend with each other unless cross edges are involved, since the agents of different subtrees in S(m; v) will not be in the same subtree. After guaranteeing low congestion, the algorithm ForwardInSwitch, which optimizes the schedule of the message-passing among agents, computes the total cost.

3.3 Multiple Multicasts

The primary agent sends its message m to a destination d in D(m; r) if any, and to the agents of S(m; v). 3. Each agent a of S(m; r) sends messages to its destinations by calling RAM, and a sends m to D(m; r) with a binomial multicast. We consider several alternatives in the first two steps of our multiple multicast algorithms. First we consider two alternatives in choosing the primary agent. It is now clear that if different multicasts select different primary agents, we can “interleave” the traffic in the second step and achieve good performance. On the other hand, we do not want to place the primary agents away from the original multicast source very often, which may cause large traffic through the root of the routing tree. As a result there is a tradeoff between good locality and interleaving. In our implementation we experimented two methods – we either choose the primary agent that is in the same subtree as the multicast source, or any agent of switches in S(m; v) at random. These two approaches will be denoted as SameTree and Random respectively. Secondly, we consider alternatives in implementing the second step of our multiple multicast algorithms. After the primary agent is chosen, it has to send the message to a processor in D(m; r) and all the agents of switch in S(m; r). This can be implemented in two different methods – the primary agent can send m to all the others with a binomial multicast, or it can work together with all the other primary agents to propagate information. In the second approach, we arrange the chosen processor in D(m; v) and all the primary agents as a ring. Each processor in the ring is responsible for relaying the information to the right side neighbor in the ring. Initially every primary agent places its message into this “circular track” and the message will be relayed to all the primary agents. We refer to these two approaches as Binomial and Cyclic respectively. Combined with the alternatives we have four

multiple multicast algorithms as follows – SameTree-Binomial, SameTree-Cyclic, Random-Binomial and Random-Cyclic.

References

[1] In Message Passing Interface Forum, MPI: A Message-Passing Interface Standard, *Mar. 1994.*

[2] N. J. Boden, D. Cohen, R. F. Felderman, A. E. Kulawik, C. L. Seitz, J. Seizovic, and W. Su. Myrinet - a gigabit per second local area network. IEEE Micro, *pages 29–36, Feb. 1995.*

[3] W. Dally. Deadlock-free message routing in multiprocessor interconnection networks. IEEE Trans. Comput.*, C-36(5):547–553, May 1987.*

[4] W. J. Dally and C. L. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. IEEE Transactions on Computers, *C-36(5):547–553, May 1987.*

[5] J. Duato. On the design of deadlock-free adaptive routing algorithms for multicomputers. In Proceedings of Parallel Architectures and Languages Europe 91*, June 1991.*

[6] J. Duato. A necessary and sufficient condition for deadlockfree adaptive routing in wormhole networks. In Proceedings of the 1994 International Conference on Parallel Proceeding*, August 1994.*

[7] M. D. S. et. al. Autonet: A high-speed, self-configuring local area network using point-to-point links. Technical Repor SRC research report 59, DEC, April 1990.

[8] K.-P. Fan and C.-T. King. Efficient multicast on wormhole switch-based irregular networks of workstations and processor clusters. In Proceedings of the Internationl Conference on High Performance Computing Systems*, 1997.*

[9] P. T. Gaughan and S. Yalamanchili. Adaptive routing protocols for hypercube interconnection networks. IEEE Computer*,26(5):12–23, May 1993.*

[10] C. Glass and L. Ni. The turn model for adaptive routing. J. ACM*, 41:847–902, Sept. 1994.*

[11] G. Gravano, G. D. Pifarre, P. E. Berman, and J. L. C. Sanz. Adaptive deadlock- and livelock-free routing with all minima paths in torus networks. IEEE Trans. Parallel and Distributed Systems*, 5(12):1233–1251, Dec. 1994.*

[12] R. Horst. Servernet deadlock avoidance and fractahedral topologies. In Proceedings of the International Parallel Processing Symposium*, pages 274–280, April 1996.*

[13] R. Kesavan and D. K. Panda. Efficient multicast on irregular switch-based cut-through networks with up-down routing In IEEE Trans. Parallel and Distributed Systems*, volume 12, August 2001.*

[14] F. T. Leighton. Introduction to Parallel Algorithms and Architectures: Arrays, Trees, hypercubes*. Morgan Kaufmann.*

[15] L. Ni and P. McKinley. A survey of wormhole routing techniques in direct networks. IEEE Computer*, 26(2):62–76, February 1993.*

[16] A.-H. E. P.K. McKinley, H. Xu and L. Ni. Unicastbased multicast communication in wormhole-routed networks IEEE Transactions on Parallel and Distributed Systems*, 5(12):1252–1265, December 1994.*

[17] W. Qiao and L. Ni. Adaptive routing in irregular networks using cut-through switches. In Proceedings of the 1996 International Conference on Parallel Proceeding*, pages I:52–60, August 1996.*