DIMACS Series in Discrete Mathematics
and Theoretical Computer Science
Volume 00, 0000

Routing in Optical Networks: The Problem of Contention

LESLIE ANN GOLDBERG

ABSTRACT. A Completely Connected Optical Communication Parallel Computer (OCPC) consists of n processors which are connected by a complete network. Communication on this network proceeds in a sequence of synchronous communication steps. During each communication step each processor can send one message to any other processor. If a processor is the destination of a single message during a communication step then it receives the message. However, if a processor is the destination of more than one message during a communication step then the messages are garbled and they are not delivered. We consider a class of message routing algorithms called *token routing* algorithms. An algorithm for routing messages on an optical network is said to be a *token routing* algorithm if it has the property that the only messages that are exchanged by the processors are the original messages to be routed, and these messages are treated as tokens. In this model each message is associated with a token which is passed on when the message is sent. Messages can be forwarded from one processor to another. However, a processor must have the token corresponding to a given message in order to send the message. We consider a contention resolution problem in which two processors (which are not known to each other) are given messages and the goal is to get both messages to processor 1. This problem can be solved in $O(\log\log n)$ steps by a deterministic token routing algorithm. We show that all deterministic token routing algorithms for solving this problem require $\Omega(\log\log n)$ steps. As a corollary, we find that every deterministic token-routing algorithm for routing h-relations on an n-processor OCPC requires $\Omega(h + \log\log n)$ communication steps.

1991 *Mathematics Subject Classification.* Primary 68Q22, 68Q25; Secondary 68R10.

This work was performed at Sandia National Laboratories and was supported by the U.S. Department of Energy under contract DE-AC04-76DP00789

1. Introduction

In massively parallel computers, processors communicate by exchanging messages over a network of communication links. In current machines the network is necessarily of low degree. Each processor can communicate directly with only a few other processors, and the remainder of the processors must be reached indirectly by routing messages along a sequence of links. Optical technology is likely to significantly speed up parallel computation because it makes it possible to build networks of high degree. In particular, the huge bandwidth of the optical medium can be divided so that each processor has its own channel for receiving messages and each processor can send on any channel[1]. Even though such an interconnection network is a complete graph, there remains the constraint that no processor can receive messages simultaneously from two other processors without corruption. That is, *contention* makes it difficult to route messages. Our goal is to understand the effect of contention on optical routing problems.

The mathematical model that we use for optical routing is the *Completely Connected Optical Communication Parallel Computer (OCPC)* model of Anderson and Miller [1]. In this model, n processors are connected by a complete network. Communication on this network consists of a sequence of synchronous communication steps. During each communication step each processor can send one message to any other processor. If a processor is the destination of a single message during a given communication step then it receives the message. However, if a processor is the destination of more than one message during a communication step then the messages are garbled and they are not delivered.

Previous papers related to the problem of routing on the OCPC include [1, 5, 6, 7, 8, 9, 11 and 13]. One problem that has been studied is that of routing an *h-relation* on an OCPC. An h-relation is a communication problem in which each processor has at most h messages to send and at most h messages to receive. Anderson and Miller [1] have observed that an h-relation can be routed in h communication steps if all of the processors are given total information about the h-relation to be realized. It is more usual to assume that initially each processor only knows about the messages that it wants to send and the processors learn about the h-relation only by receiving messages from other processors. In this case it is difficult to avoid contention and the problem of routing h-relations becomes (provably) more difficult. In particular, Goldberg, Jerrum and MacKenzie [8] have shown that (for $h > 1$) every randomized algorithm for routing arbitrary h-relations on an n-processor OCPC requires $\Omega(h + \sqrt{\log \log n})$ expected communication steps. The fastest known randomized algorithm for solving this problem (due to Goldberg, Jerrum, Leighton, and Rao [7]) takes $O(h + \log \log n)$ communication steps.

We believe that every randomized algorithm for routing h-relations on an n-

[1]The division of the bandwidth is typically done by *wavelength division multiplexing*. For more information about the implementation of such networks see [10], [3] and [4] and the references in [6] and [12].

processor OCPC requires $\Omega(h + \log\log n)$ communication steps. However, this conjecture has not been proved. More is known for specific classes of optical routing algorithms. In particular, several authors have studied a class of algorithms called *direct* algorithms for routing h-relations. An OCPC algorithm for realizing h-relations is said to be *direct* if it has the property that the only messages that are exchanged by the processors are the original messages of the h-relation and these messages are sent directly to their destinations with no forwarding. Geréb-Graus and Tsantilas [6] have shown that h-relations can be routed in $O(h + \log h \log n)$ expected communication steps by a randomized direct algorithm. MacKenzie, Plaxton and Rajaraman [11] have shown that every such algorithm requires $\Omega(h + (\log h/\log\log h)\log n)$ expected communication steps. MacKenzie, Plaxton and Rajaraman have also studied the problem of routing h-relations with deterministic direct algorithms. They showed that $\Omega(h \log n/\log h)$ steps are required and that $O(h \log n \log h)$ steps suffice.

The lower bounds of MacKenzie, Plaxton and Rajaraman demonstrate that direct algorithms are not very good at dealing with contention in optical networks. The goal of this work is to investigate the contention-resolution capability of a wider class of algorithms, which we call *token routing* algorithms. An algorithm for routing messages on an optical network is said to be a *token routing* algorithm if it has the property that the only messages that are exchanged by the processors are the original messages to be routed and these messages are treated as tokens. In this model, messages can be forwarded from one processor to another. However, a processor must have the token corresponding to a given message in order to send the message. (The token is passed on when the message is sent.) The motivation for considering token routing algorithms is that copying messages is expensive, so one would like to have message-routing algorithms which do not make many copies of the messages to be delivered. Token routing algorithms do not make copies of messages. The routing algorithm of Goldberg, Jerrum, Leighton, and Rao, by contrast, could make up to $\log n$ copies of any given message.

In order to investigate the contention-resolution capability of token-routing algorithms, we study a simple contention resolution problem, which we call the *2-contention resolution problem.* In this problem, two processors (which are not known to each other) are given messages and the goal is to get both messages to processor 1[2]. In this paper we confine our attention to deterministic algorithms. The 2-contention resolution problem can be solved in $O(\log\log n)$ steps by a deterministic token routing algorithm on an n-processor OCPC. We show that all such algorithms require $\Omega(\log\log n)$ steps[3]. As a corollary, we find that every

[2]The 2-contention resolution problem is similar to the 2-station "control tower problem" studied in [11]. The only difference is that the control tower problem requires algorithms to be direct. (A similar problem has been studied in a different context in [2].)

[3]In fact, we show that for any $x \geq 2$, the x-contention resolution problem, in which x processors are given messages to deliver to processor 1, requires $\Omega(x+\log\log n)$ communication steps. Surprisingly, this result is not implied by the fact that $\Omega(\log\log n)$ steps are required to

deterministic token-routing algorithm for routing h-relations on an n-processor OCPC requires $\Omega(h + \log\log n)$ communication steps.

2. Contention Resolution by Token Routing Algorithms

We begin this section by observing that the 2-contention resolution problem can be solved in $O(\log\log n)$ steps by a token routing algorithm on an n-processor OCPC. The algorithm, which was suggested to us by Satish Rao, is as follows. Suppose that the two messages are given to the ith processor and the jth processor. Let $i_1, \ldots, i_{\log n}$ be the bits in the binary representation of i and let $j_1, \ldots, j_{\log n}$ be the bits in the binary representation of j. Processors i and j use binary search to find an integer ℓ such that $i_\ell \neq j_\ell$. (The bits i_ℓ and j_ℓ will be used to break the contention.) To start the binary search, processor i sends its message to the processor whose binary representation is $i_1, \ldots, i_{(\log n)/2}, 0, \ldots, 0$ and processor j sends its message to the processor whose binary representation is $j_1, \ldots, j_{(\log n)/2}, 0, \ldots, 0$. If there is a collision then the two processors concentrate their search on the second half of their bits. Otherwise, the messages are returned, and the two processors concentrate their search on the first half of their bits.

In this section we will prove the following theorem.

THEOREM 1. *Every token routing algorithm requires $\Omega(\log\log n)$ steps to solve the 2-contention resolution problem on an n-processor* OCPC.

Since the 2-contention resolution problem is a particular example of a 2-relation, we obtain the following corollary.

COROLLARY 2. *Every token routing algorithm for routing h-relations on an n-processor* OCPC *requires $\Omega(h + \log\log n)$ communication steps.*

In order to get a result which is slightly stronger than Theorem 1 we will generalize the 2-contention resolution problem and we will consider the x-contention resolution problem (for $x \geq 2$), in which x processors (which are not known to each other) are given messages. The goal is to deliver all of the messages to processor 1. We will prove the following.

THEOREM 3. *Suppose that $x \geq 2$. Every token routing algorithm requires $\Omega(\log\log n)$ steps to solve the x-contention resolution problem on an n-processor* OCPC.

It is clear that Theorem 1 follows from Theorem 3. It is not obvious that the converse is true. In fact, Theorem 1 is somewhat easier to prove than Theorem 3 because in Theorem 1 one can make use of the fact that, since there are only two messages in the 2-contention resolution problem, either both messages collide on a given step, or neither collides. In the remainder of the section we will prove Theorem 3.

---

solve the 2-contention resolution problem.

We will say that an algorithm for the x-contention resolution problem is *weakly successful* on a given input in t steps if, when it is run with that input for t steps, at least one of the following is true.

1. There is some processor that sees more than one of the x messages, or
2. Every message visits a processor in $\{1, \ldots, \log n\}$.

We will prove the following lemma.

LEMMA 4. *Suppose that $x \geq 2$ and that $T \leq \log\log n/2$. Suppose that $\mathcal{A}$ is a token-routing algorithm that allegedly solves the x-contention resolution problem on an n-processor* OCPC. *Then there is an input on which $\mathcal{A}$ is not weakly successful in T steps.*

Clearly, Theorem 3 follows from Lemma 4. To prove Lemma 4 we use the following notation. A *t-step history* will be defined to be a pair (i, C) in which i is an integer in the range $1 \leq i \leq n$ and C is a subset of $\{1, \ldots, t\}$. (The t-step history (i, C) will be used in the proof to denote the fact that in a certain run of a certain algorithm a token starting at processor i had collisions on the steps in C (and not on the steps in $\{1, \ldots, t\} - C$).

We will use the following claim.

CLAIM 5. *Suppose that $t \geq 0$ and suppose that a token routing algorithm is run on an input on which it is not weakly successful in t steps. Then for any integer t' in the range $1 \leq t' \leq t + 1$, and for any token in the input, the t-step history of the token contains enough information to deduce*

1. *which processor held the token at the start of step t',*
2. *whether (and where) the token was sent on step t', and*
3. *which processor held the token at the end of step t'.*

PROOF OF CLAIM 5. Claim 5 is proved by induction on t. The case $t = 0$ is trivial. Suppose that $t > 0$ and suppose that a token routing algorithm is run on an input on which it is not weakly successful in t steps. Consider any token in the input. By induction, the $(t-1)$-step history of the token contains enough information for us to deduce which processor (call it p) held the token at the start of step $t + 1$. Since the algorithm is not weakly successful in t steps p did not see any other tokens during steps $1, \ldots, t$. Thus, all information available to p at the start of step $t + 1$ is encoded in the t-step history of the token. □

It follows from Claim 5 that if a token routing algorithm is not weakly successful on a given input in t steps then the $t+1$st step of the algorithm is a function of the t-step histories of the tokens. Thus, for the purposes of proving Lemma 4, we can encode a T-step algorithm as a sequence of functions $f_1, \ldots, f_T$. Function f_t will map every $(t-1)$-step history to an integer in the range $\{0, ..., n\}$. If (i, C) is the $(t-1)$-step history of a given token and $f_t(i, C) = 0$ then the token will be held at step t. Otherwise, it will be sent to $f_t(i, C)$.

Suppose that $x \geq 2$ and that $T \leq \log\log n/2$. Fix a T-step token routing algorithm $\mathcal{A} = f_1, \ldots, f_T$. We wish to show that there is an input on which $\mathcal{A}$

is not weakly successful in T steps.

For each t in the range $\{0, \ldots, T\}$ we will define a hypergraph H_t. Let the vertices of H_t be all possible t-step histories. For each size x subset $\{i_1, \ldots, i_x\}$ of $\{1, \ldots, n\}$ we will add up to one hyperedge to H_t as follows: Consider a run of t steps of $\mathcal{A}$ when the x tokens start at processors $i_1, \ldots, i_x$. If during these t steps some processor sees more than one of the x messages then do not add a hyperedge corresponding to $\{i_1, \ldots, i_x\}$. Otherwise, let (i_j, C_{i_j}) denote the t-step history of the token that started at i_j in this run of the algorithm and put the hyperedge containing the x vertices (i_j, C_{i_j}) in H_t.

Note that the set of hyperedges of H_t can be constructed from f_t and from the set of hyperedges of H_{t-1}. Suppose that H_t has a hyperedge corresponding to $\{i_1, \ldots, i_x\}$. Then H_{t-1} will also have a hyperedge corresponding to $\{i_1, \ldots, i_x\}$. Now, suppose that H_{t-1} contains a hyperedge consisting of the vertices $(i_1, C_{i_1}), \ldots, (i_x, C_{i_x})$. To see whether H_t contains a hyperedge corresponding to $\{i_1, \ldots, i_x\}$ and, if so, to determine which such hyperedge is in H_t, we proceed as follows. For each vertex (i_j, C_{i_j}) in the hyperedge in H_{t-1}: If there is another vertex (i_k, C_{i_k}) in the hyperedge in H_{t-1} such that $f_t(i_j, C_{i,j}) = f_t(i_k, C_{i_k})$ and this quantity is non-zero then let $C'_{i_j} = C_{i_j} \cup \{t\}$. Otherwise, let $C'_{i,j} = C_{i,j}$. Now look at the t-step histories $(i_1, C'_{i_1}), \ldots, (i_x, C'_{i_x})$. By Claim 5 we can use these t-step histories to determine whether, during the t steps of a run of $\mathcal{A}$ with tokens starting at $\{i_1, \ldots, i_x\}$, some processor sees more than one of the x messages. If not, the hyperedge consisting of vertices $(i_1, C'_{i_1}), \ldots, (i_x, C'_{i_x})$ is put in H_t.

A *clique* in H_t is defined to be a subset S of the vertices such that every size x subset of S is a hyperedge of H_t. Consider the size of the largest clique in H_t.

CLAIM 6. *H_0 contains the n-clique consisting of the vertices $(1, \emptyset), \ldots, (n, \emptyset)$.*

PROOF. Straightforward.

CLAIM 7. *Suppose that $\mathcal{A}$ is weakly successful on every input in T steps. Then the largest clique in H_T has size at most $\log n$.*

PROOF. To prove Claim 7 note that every hyperedge in H_T consists of x vertices $(i_1, C_{i_1}), \ldots, (i_x, C_{i_x})$ such that when the algorithm is run for T steps with the tokens starting at $i_1, \ldots, i_x$ every token visits a processor in $\{1, \ldots, \log n\}$ but no processor sees more than one of the x tokens. □

CLAIM 8. *If H_t has a clique of size m for $m > \log n$ then H_{t+1} has a clique of size $m^{1/3}$.*

PROOF OF CLAIM 8. The proof of Claim 8 is based on an argument which is sometimes known as the "Dedekind box" argument. Suppose that H_t has a clique containing m vertices. Suppose that at least $m^{1/3}$ of these vertices (call them $(j_1, C_{j_1}), \ldots, (j_{m^{1/3}}, C_{j_{m^{1/3}}})$) are mapped to the same point by f_{t+1}. If this point is 0 then these same vertices form a clique in H_{t+1}. If it is non-zero then the vertices $(j_1, C_{j_1} \cup \{t\}), \ldots, (j_{m^{1/3}}, C_{j_{m^{1/3}}} \cup \{t\})$ form a clique in H_{t+1}.

Otherwise, at least $m^{2/3}$ of the m vertices are mapped to distinct points by f_{t+1} (call the $m^{2/3}$ vertices $(j_1, C_{j_1}), \ldots, (j_{m^{2/3}}, C_{j_{m^{2/3}}})$). We will identify a large subset S of these vertices which forms a clique in H_{t+1}. To identify S, we construct a graph G on vertex set $(j_1, C_{j_1}), \ldots, (j_{m^{2/3}}, C_{j_{m^{2/3}}})$. For each vertex (j_i, C_{j_i}) add up to one edge to the graph as follows: If $f_{t+1}(j_i, C_{j_i}) > 0$ and there is another vertex (j_ℓ, C_{j_ℓ}) such that the t-step history (j_ℓ, C_{j_ℓ}) indicates that the token starting at j_ℓ visited $f_{t+1}(j_i, C_{j_i})$ in the first t steps then add the edge $((j_i, C_{j_i}), (j_\ell, C_{j_\ell}))$ to G. (There can be at most one such vertex (j_ℓ, C_{j_ℓ}) because the vertices form a clique in H_t.) It is clear that if we choose as S any set of vertices that is an independent set in G then the vertices in S will form a clique in H_{t+1}. Note that the maximum degree of a vertex in G is $t+2$. By Turán's theorem, G has an independent set of size $m^{2/3}/(t+3) \geq m^{1/3}$. □

Claim 6 and Claim 8 imply that H_T has a clique of size at least $n^{(1/3)^T}$ which is greater than $\log n$, as long as n is sufficiently large. Therefore, Lemma 4 follows from Claim 7.

3. Open Problems

This paper does not address the problem of solving the 2-contention resolution problem with *randomized* token routing algorithms. The obvious direct randomized algorithm for solving this problem runs in 3 expected steps, but the probability that more steps are required is large. A related problem is that of deriving a lower bound for the problem of routing h-relations with randomized token routing algorithms. It is possible that the technique of Goldberg, Jerrum and MacKenzie [8] could be used to convert Theorem 1 (or Theorem 3) into a randomized lower bound for h-relation routing. A much more interesting open problem is to extend our result to obtain an $\Omega(h + \log\log n)$ lower bound for routing h-relations with *general* OCPC algorithms. A first step towards proving this result might be to extend the proof of Theorem 1 to a similar theorem for a model which allows limited copying of messages.

Acknowledgments

I would like to thank Mark Jerrum, James Park and Jon Atkins for useful discussions concerning this problem.

References

1. R. J. Anderson and G. L. Miller, *Optical communication for pointer based algorithms*, Technical Report CRI 88-14, Computer Science Department, University of Southern California, Los Angeles, CA 90089-0782 USA (1988).
2. Robert Cypher and Smaragda Konstantinidou, *Bounds on the efficiency of message-passing protocols for parallel computers*, Proceedings of the 5th ACM Symposium On Parallel Algorithms and Architectures, ACM Press, 1993, pp. 173–181.
3. Patrick W. Dowd, *High performance interprocessor communication through optical wavelength division multiple access channels*, Proceedings of the 18th International Symposium on Computer Architecture, 1991, pp. 96–105.

4. Mary Mehrnoosh Eshaghian, *Parallel algorithms for image processing on OMC*, IEEE Transactions on Computers **40** (1991), no. 7, 827–833.
5. A. V. Gerbessiotis and L. G. Valiant, *Direct bulk-synchronous parallel algorithms*, Proceedings of the 3rd Scandinavian Workshop on Algorithm Theory, 1992.
6. M. Geréb-Graus and T. Tsantilas, *Efficient optical communication in parallel computers*, Proceedings of the 4th ACM Symposium On Parallel Algorithms and Architectures, ACM Press, 1992, pp. 41–48.
7. Leslie Ann Goldberg, Mark Jerrum, Tom Leighton and Satish Rao, *Doubly logarithmic communication algorithms for optical communication parallel computers*, pre-print (1994), (A preliminary version of this paper appeared in the Proceedings of the 5th ACM Symposium On Parallel Algorithms and Architectures in 1993.).
8. Leslie Ann Goldberg, Mark Jerrum and Philip D. MacKenzie, *An* $\Omega(\sqrt{\log \log n})$ *lower bound for routing in optical networks*, Proceedings of the 6th ACM Symposium on Parallel Algorithms and Architectures, ACM Press, 1994.
9. Leslie Ann Goldberg, Yossi Matias and Satish Rao, *An optical simulation of shared memory*, Proceedings of the 6th ACM Symposium on Parallel Algorithms and Architectures, ACM Press, 1994.
10. Alfred Hartmann and Steve Redfield, *Design sketches for optical crossbar switches intended for large-scale parallel processing applications*, Optical Engineering **28** (1989), no. 4, 315–327.
11. P.D. MacKenzie, C.G. Plaxton and R. Rajaraman, *On contention resolution protocols and associated probabilistic phenomena*, Proceedings of the 26th ACM Symposium on Theory of Computing, ACM Press, 1994.
12. W. F. McColl, *General purpose parallel computing*, Lectures on Parallel Computation, Proceedings of the 1991 ALCOM Spring School on Parallel Computation (A.M. Gibbons and P. Spirakis, ed.), Cambridge University Press, 1993, pp. 337–391.
13. L. G. Valiant, *General purpose parallel architectures*, Handbook of Theoretical Computer Science (J. van Leeuwen, ed.), Elsevier, 1990, (see especially p. 967).

Department of Algorithms and Discrete Mathematics, Sandia National Laboratories, MS 1110, PO Box 5800, Albuquerque, New Mexico 87185-1110

E-mail address: lagoldb@cs.sandia.gov