Constant Thinning Protocol for Routing h-Relations in Complete Networks

Anssi Kautonen[1], Ville Leppänen[2], and Martti Penttonen[1]

[1] University of Joensuu, Department of Computer Science, P.O.Box 111, 80101 Joensuu, Finland, email {Anssi.Kautonen,Martti.Penttonen}@cs.joensuu.fi
[2] University of Turku, Department of Computer Science and Turku Centre for Computer Science, Lemminkäisenkatu 14 A, 20520 Turku, Finland, email Ville.Leppanen@cs.utu.fi

Abstract. We propose a simple protocol, called *constant thinning protocol*, for routing in a complete network under OCPC assumption, analyze it, and compare it with some other routing protocols.

1 Introduction

Parallel programmers would welcome the "flat" shared memory, because it would make PRAM [15] style programming possible and thus make easier to utilize the rich culture of parallel algorithms written for the PRAM model. As large memories with a large number of simultaneous accesses do not seem feasible, the only possible way to build a PRAM type parallel computer appears to be to build it from processors with local memories. The fine granularity of parallelism and global memory access, what makes the PRAM model so desirable for algorithm designers, sets very high demands for data communication. Fortunately, memory accesses at the rate of the processor clock are not necessary, but due to the parallel *slackness* principle [16], latency in access does not imply inefficiency. It is enough to route an *h-relation* efficiently. By definition, in h-relation there is a complete graph, where each node (processor) has at most h packets to send, and it is the target of at most h packets. We assume the *OCPC* (Optical Communication Parallel Computer) or *1-collision* assumption [1]: If two or more packets arrive at a node simultaneously, all fail. An implementation of an h-relation is *work-optimal* at *cost* c, if all packets arrive at targets in time ch.

The first attempt to implement an h-relation is to use a *greedy* routing algorithm. By greedy principle, one tries to send packets as fast as one can. The fatal drawback of the greedy algorithm is the *livelock*: packets can cause mutual failure of sending until eternity. In a situation, when each of two processors has one packet targeted to the same processor, they always fail.

Another routing algorithm was proposed by Anderson and Miller [1], and it was improved by Valiant [15]. They realize work-optimally an h-relation for $h \in \Omega(\log p)$, where p is the number of processors. Other algorithms with even lower latency were provided by [6, 7, 2, 3, 11]. Contrary to these, the h-relation algorithm of Geréb-Graus and Tsantilas [5], GGT for short, has the advantage of being *direct*, i.e. the packets go to their targets directly, without intermediate nodes. For other results related with direct or indirect routing, see also [4, 14, 8, 9, 12, 13]. Actually our new algorithm was inspired by [13], although the latter deals with the continuous routing problem, not the h-relation.

proc GGT(h,ϵ,α)
for $i = 0$ **to** $\log_{1/(1-\epsilon)} h$ **do** Transmit($(1-\epsilon)^i h$,ϵ,α)

proc Transmit(h,ϵ,α)
for all processors P **pardo**
  for $\frac{\epsilon}{1-\epsilon}(\epsilon h + max\{\sqrt{4\epsilon\alpha h \ln p}, 4\alpha \ln p\})$ times **do**
    choose an unsent packet x at random
    attempt to send x with probability $\frac{\#\ unsent\ packets}{h}$

2 Thinning protocols

The throughput of the greedy routing of randomly addressed packets is characterized by

$$(1 - \frac{1}{h})^{h-1} \geq \frac{1}{e}$$

where $1/h$ is the probability that one of the $h-1$ competing processors is sending to the same processor at the same time. (For all $x > 0$, $(1-1/x)^{x-1} \geq e^{-1}$.) This would be the throughput if all processors would create and send a new randomly addressed packet, which is not the case in routing an h-relation. When failed packets are resent again and again, the risk of livelock grows. A solution is to decrease the sending probability. In GGT algorithm, the sending probability of packets varies between 1 and $1 - \epsilon$ $(0 < \epsilon < 1)$. The transmission of packets is thus 'thinned' by factor 1 to $1/1 - \epsilon$, preventing the livelock. We now propose a very simple routing protocol, where thinning is more explicit.

proc CT(h,h_0,t,t_0) %$1 \leq h_0 \leq h, 1 \leq t_0 \leq t$
for all processors **do**
  while packets remain **do**
    Transmit h randomly chosen packets (if so many remain)
      within time window $[1..\lceil th \rceil]$
    $h := \max\{(1 - e^{-1/t_0})h, h_0\}$

A drawback of thinning is that a processor cannot successfully send a packet at those moments, when it does not even try to send. Thinning by factor t would thus imply inefficiency by factor t. But this is somewhat balanced by the success probability, which increases from $1/e$ to

$$(1 - \frac{1}{th})^{h-1} \geq \frac{1}{e^{1/t}}.$$

The expected throughput with thinning is thus characterized by the function $te^{1/t}$ whose growth is very modest: for $t = 1, 1.2, 1.5, 2.0$ and 3.0 the values of $te^{1/t}$ are 2.7, 2.8, 2.9, 3.3, and 4.2. Still, even small thinning increases greatly the robustness against livelock. For very small numbers of packets constant thinning alone does not guarantee high success probability. For that reason, CT has a minimum size $h_0 \in \Omega(\log p)$ for thinning window, to prevent repeated collisions.

3 Analysis

The analyses for GGT and CT are rather similar. One can prove that

1. the number of unsent packets decreases geometrically from h to $\log p$
2. the rest of the packets can be routed in time $O(\log p \log\log p)$

Theorem 1. *For $h \in \Omega(\log p \log\log p)$, CT routes any h-relation in time $O(h)$ with high probability.*

Proof. Consider a round of the while loop, where $h' \geq k_0 \log p = h_0$ for a suitable constant k_0. We shall show that if in the beginning of a round the routing task is an h'-relation, it will be a ch'-relation after the round with high probability.

To ease the calculation of probabilities, we worsen the routing task a little by completing the initial h'-relation to a full h'-relation. We do this by adding 'dummy packets' with proper destination to those processors not having initially h' packets. We assume that the dummy packets participate in routing as the other packets. In the analysis below, the dummy packets can collide with normal packets as well as with other dummy packets, but in the actual algorithm attempting to route a dummy packet corresponds to an unallocated time slot (unable to cause any collisions). Thus the number of successful normal packets is always better in the actual situation.

We first show that, with high probability, the number of packets decreases to ch' or below. As a packet has the probability $1/th'$ of being sent at a given moment of time (from the interval $[1 \ldots \lceil th' \rceil]$), and by the h'-relation assumption, at most h' packets have the same target, the probability of success for a packet at a given moment of time is at least

$$\frac{1}{th'} \times (1 - \frac{1}{th'})^{h'-1} \geq \frac{1}{e^{1/t} \cdot th'},$$

since

$$(1 - \frac{1}{th'})^{m} \geq (1 - \frac{1}{th'})^{h'-1}$$

for any $m \leq h' - 1$ (consider m as the actual number of other packets with the same target) and

$$(1 - \frac{1}{th'})^{h'-1} = \left((1 - \frac{1}{th'})^{th'-t}\right)^{1/t} \geq \left((1 - \frac{1}{th'})^{th'-1}\right)^{1/t} \geq e^{-1/t}$$

for $h' > 1$ and $t \geq 1$. For each packet, the probability of success for a packet during the procedure call is at least

$$\frac{1}{e^{1/t} \cdot th'} \times th' = e^{-1/t}.$$

Hence, the expected number of successful packets of a processor within these th' units of time is $E_t = h'e^{-1/t}$. By applying Chernoff bound [10]

$$Pr(N < (1-\epsilon)E_t) \leq e^{-\frac{1}{2}\epsilon^2 E_t}$$

and choosing $(1-\epsilon)E_t = E_{t_0}$, we get $\epsilon = 1 - E_{t_0}/E_t \approx 1/t_0 - 1/t$, and

$$Pr(N < h'e^{-1/t_0}) \leq e^{-0.5 \times \epsilon^2 \times h' e^{-1/t}} \leq \frac{1}{p^{2.5}}$$

for $h' > h_0 = k_0 \log p$ for some constant k_0.

Hence, in a processor the number of outgoing packets decreases by compression factor $c = 1 - 1/e^{1/t_0}$ with probability $1 - 1/p^{2.5}$, and in all p processors and in all $\log_{1/c} p < \sqrt{p}$ rounds with probability $1 - 1/p$. Guaranteeing that the number of incoming packets is at most ch' for each processor is analyzed analogously. Clearly, the full h'-relation (completed with dummy packets) decreases to ch'-relation. Removing the dummy packets from the system can only decrease the degree of the relation. Within $\log_{1/c} h$ geometrically converging phases, or $O(ht)$ time, there remain no more than $O(\log p)$ packets in each processor.

Observe that by choosing a larger h_0 in the analysis above, we can easily show the same progression in the degree of the relation with probability $1 - p^{-\alpha}$ for any positive constant α.

For the rest of the algorithm, when $h' \leq h_0$ packets remain, the success probability of a packet is

$$\frac{h'}{th_0} \times (1 - \frac{1}{th_0})^{h_0 - 1} \geq \frac{h'}{h_0} \frac{1}{te^{1/t}}.$$

Thus the expectation of sending times for this packet is $te^{1/t}h_0/h'$. The sum of all these expectations, until all packets have been sent, is

$$te^{1/t}h_0(\frac{1}{h_0} + \frac{1}{h_0 - 1} + \ldots + \frac{1}{2} + 1) = E \in O(h_0 \log h_0).$$

By another form of Chernoff bound [10]

$$Pr(T > r) \leq \frac{1}{2^r} \quad \text{for } r > 6E$$

we see that

$$Pr(T > k_1 h_0 \log h_0) \leq \frac{1}{p^{2.5}}$$

for some k_0 and therefore all packets can be transmitted to their targets in time $O(\log p \log\log p)$ with high probability. By combining the two phases we see that all packets can be routed in time $O(h + \log p \log\log p)$ with high probability. By choosing $h \in \Omega(\log p \log\log p)$ we complete the proof. $\square$

4 Experiments

We ran some experiments to get practical experience of the new algorithm, see Table 1. The results are averages of 500 experiments.

p	$t = 1.2$	$t = 1.5$	$t = 2.0$	GGT
128	3.74	3.82	4.20	4.86
256	3.73	3.77	4.16	4.92
512	3.76	3.77	4.13	4.95
1024	3.81	3.74	4.10	4.97
2048	3.78	3.72	4.07	4.99
4096	3.79	3.70	4.05	5.00

Table 1. Routing cost of the CT algorithm with $h = \log_2^2 p$ and $h_0 = \log_2 p$. Thinning parameters $t = 1.2, 1.5, 2.0$ were tried. For comparison, results of GGT with $h = \log_2^2 p$, $\epsilon = 0.25$, $\alpha = 0.1$ are presented.

By the results in Table 1, the constant thinning algorithm CT achieves the cost 3.7 with slackness $h = \log^2 p$. With a suitable combination of ϵ and α GGT comes below 5 but is clearly slower than CT. Note that $e = 2.7$ is the lower bound of the cost.

In addition to mere numbers, our routing simulator shows the progress of routing graphically (see Figure 1). In CT the throughput of packets is constant for most of the time, while GGT follows a saw blade pattern. The "tail" of the graph is very critical. Too little thinning or too small h_0 grows the tail, as the algorithm approaches to the greedy algorithm.

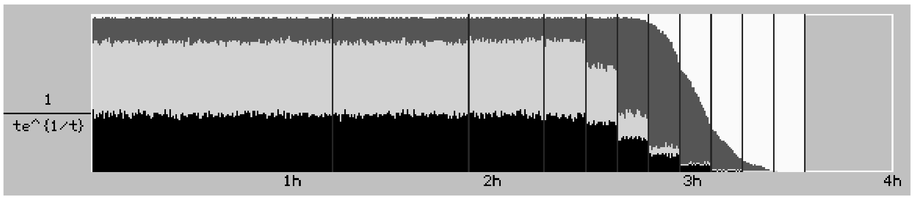

Fig. 1. Graphical output of a CT simulation in case $p = 1024$, $h = 128$, $h_0 = 16$, $t = t_0 = 1.2$. The picture is divided in three horizontal bands by ragged borderlines at $1/t = 83\%$ and at $1/te^{1/t} = 36\%$ of all processors. The bottom band represents successful processors, the middle band failed processors, and the top band passive processors. The vertical lines at intervals of cth, c^2th, ..., until $c^ith < h_0 = 16$ ($c = 1 - 1/e^{1/t_0}$) separate the phases of the algorithm. The total time in picture is $456 = 3.56h$.

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*, LA, 1988.
2. A. Czumaj and F. Meyer auf der Heide, and V. Stemann. Shared memory simulations with triple-logarithmic delay. *Proc. ESA95*, 46–59, 1995.
3. M. Dietzfelbinger and and F. Meyer auf der Heide. Simple, efficient shared memory simulations. *Proc. SPAA93*, 110–119.
4. J. Håstad, T. Leighton, and B. Rogoff. Analysis of backoff protocols for multiple access channels. *SIAM J. Comput.* 25:740–774, 1996
5. M. Geréb-Graus and T. Tsantilas. Efficient optical communication in parallel computers. In *Proc. SPAA'92*, pp. 41 – 48, June 1992.
6. L.A. Goldberg, M. Jerrum, T. Leighton, and S. Rao. A doubly logarithmic communication algorithm for the completely connected optical communication parallel computer. In *Proc. SPAA'93*, pp. 300 – 309, June 1993.
7. L.A. Goldberg, Y. Matias, and S. Rao. An optical simulation of shared memory. In *Proc. SPAA'94*, pp. 257 – 267, June 1994.
8. L.A. Goldberg and P.D. MacKenzie, Analysis of Practical Backoff Protocols for Contention Resolution with Multiple Servers. *Proc. SODA'96*, pp. 554–563.
9. L.A. Goldberg and P.D. MacKenzie, Contention Resolution with Guaranteed Constant Expected Delay, *Proc. FOCS'97*, pp. 213–222.
10. T. Hagerup, C. Rüb. A guided tour of Chernoff bounds. *Information Processing Letters* 33:305–308, 1989.
11. R.M. Karp, M. Luby, and F. Meyer auf der Heide. Efficient PRAM simulation on a distributed memory machine. *Proc. STOC'92*, pp. 318–326.
12. A. Kautonen and V. Leppänen and M. Penttonen. Simulations of PRAM on Complete Optical Networks. In *Proc. EuroPar'96*, LNCS 1124:307–310.
13. M. Paterson and A. Srinivasan. Contention resolution with bounded delay. In *Proc. FOCS'95*, pp. 104–113.
14. P. Raghavan, and E. Upfal. Stochastic Contention Resolution With Short Delays. In *Proc. STOC'95*, pp. 229–237.
15. L.G. Valiant. General purpose parallel architectures. In *Handbook of Theoretical Computer Science, Vol. A*, 943–971, 1990.
16. L.G. Valiant. A bridging model for parallel computation. *Communications of the ACM*, 33:103-111, 1990.