

# Routing on Networks of Optical Crossbars\*

(Extended Abstract)

Friedhelm Meyer auf der Heide, Klaus Schröder, and Frank Schwarze

Heinz Nixdorf Institute and Computer Science Department, University Paderborn,  
33102 Paderborn, Germany

**Abstract.** We describe routing algorithms on networks composed of optical busses. Using networks with short busses and small degree we are able to give very fast routing algorithms. First, we describe a leveled optical network and routing algorithms for it. Next, we show how to simulate this network on high-dimensional meshes of optical busses (MOBs). We present algorithms, e.g., for  $h$ -relations with runtime being double-logarithmic in the size of the mesh, linear in  $h$ , and polynomial in the dimension. Previous results are exponential in the dimension. We use a novel type of protocol and analysis inspired by hashing based shared memory simulations with redundant storage representation from [MSS95].

## 1 Introduction

In recent years, the possibility of using optical devices to build very fast, high bandwidth communication networks has attracted many researchers, engineers as well as (theoretical) computer scientists. Anderson and Miller [AM88] were the first to consider routing algorithms for  $h$ -relations using the following model, later called the Completely Connected Optical Communication Parallel Computer (OCPC), or, as a routing device, the optical bus. It is motivated by possibilities and restrictions of optical communication technology.

An optical bus of length  $k$  connects  $k$  processors. In one step, each processor can try to send a message to an arbitrary other processor. The sending is successful, only if the receiving processor gets no other message in this step. For a discussion of this model see [AM88], [GJLR93] and [GJLR94]. Extensions are discussed in [DM93] and [MSS95].

In order to avoid too long optical busses, 2-dimensional meshes in which rows and columns are connected by optical busses are examined in [GJLR94]. These meshes are called mesh of optical busses (MOBs). Here, length  $k$  busses suffice to obtain an efficient routing device for  $k^2$  processors.

In this paper we examine the routing capability of this and other networks of optical busses. Formally such a network can be described by a hypergraph whose nodes are the processors and whose edges are the optical busses. The following parameters characterize the quality of such a network of size  $N$ :

- the maximum length  $k$  of the busses

---

\* supported by the DFG-Graduiertenkolleg "Parallele Rechnernetzwerke in der Produktionstechnik", ME 872/4-1 and by DFG-SFB 376 "Massive Parallelität"

- the maximum degree  $d$
- the routing time for permutations,  $h$ -relations, random functions etc.

Our main results are very fast routing algorithms on certain leveled networks of optical busses, the split&hash networks. They are inspired by high-ary, low depth versions of Butterfly networks or Multibutterfly networks [Upf89].

We derive a network of busses from such a network by replacing certain sub-graphs by optical busses. Furthermore we show how to simulate such split&hash networks on high-dimensional MOB, achieving the fastest known routing algorithms for them. Our techniques for permutation routing differ significantly from those previously derived for MOB of constant dimension as in [GJLR94]. They are inspired by hashing techniques to simulate shared memory on an OCPC as described in [MSS95].

## 1.1 Previous Results

Anderson and Miller [AM88] present an algorithm for realizing  $h$ -relations in  $O(h + \log N)$  expected time on a single OCPC.

In [GJLR93] Goldberg et al. present an algorithm for  $h$ -relations using  $O(h + \log \log N)$  time, with high probability,<sup>2</sup> if  $h < \log N$ .

The algorithm of Goldberg et al. works on a single OCPC. In [GJLR94] they give an algorithm on a 2-dimensional MOB using  $O(h + \log \log N)$  time, with high probability, if  $h < \log N$ . In the same paper they suggest an extension of their algorithm to higher dimensions still using  $O(h + \log \log N)$  time, if the dimension  $d$  is constant. Extensions to higher dimensions result in a runtime exponential in  $d$ .

## 1.2 New Results

We present a new Butterfly-like class of networks, the so called *split&hash networks*, and its realization using optical busses. On such networks we give algorithms for delivering packets from the  $N$  sources to the  $N$  sinks. Our algorithm for permutation routing requires  $O(d^2 \cdot \log d \cdot \log \log N)$  steps on a split&hash network of depth  $d$ , with high probability. The probability space is described by some random choices done for constructing the networks. The main building block of our algorithm is an adaption of a scheme presented in [MSS95] where it is used and analyzed for obtaining fast shared memory simulations on OCPCs and similar machines.

The algorithm can be extended to route a random function. The running time is  $O\left(\frac{\log N}{\log \log N}\right)$ , with high probability, if  $d = O(\log^\delta N)$ ,  $\delta < \frac{1}{2}$ . The last step of the algorithm makes use of an algorithm from [GJLR93]. The overall running time is optimal, since some sink gets  $\Theta\left(\frac{\log N}{\log \log N}\right)$  packets, with high probability.

Combining our methods with another technique from [GJLR93] and [AM88] we derive an algorithm for  $h$ -relations. The algorithm is randomized and requires  $O(d^3 \cdot \log d \cdot \log \log N + d^3 \cdot h)$  steps, with high probability, if  $h \leq \log N$ ,  $d = O(\log^\delta N)$  and  $\delta < \frac{1}{2}$ .

<sup>2</sup> With high probability means a probability of at least  $1 - N^{-\gamma}$ , where  $\gamma > 0$  can be chosen arbitrary.

Simulating a split&hash network of depth  $d$  we obtain algorithms for the  $d$ -dimensional MOB.

We get an algorithm using  $O(d^4 \cdot \log d \cdot \log \log N)$  steps, with high probability, to route permutations and an algorithm for random functions using  $O(d^4 \cdot \log d \cdot \log \log N + \frac{\log N}{\log \log N})$  steps, with high probability. This is optimal for  $d = O(\log^\delta N)$ ,  $\delta < \frac{1}{4}$ . The MOB version of our  $h$ -relation algorithm requires  $O(d^5 \cdot \log d \cdot \log \log N + d^3 \cdot h)$  steps, with high probability, for any  $h$ -relation, for  $d = O(\log^\delta N)$ ,  $\delta < \frac{1}{2}$ .

Our results show a tradeoff between running time for various routing problems, the size of the optical busses and the degree of the network. The size of the optical busses in a split&hash network of depth  $d$  is  $O(d \cdot N^{\frac{1}{d}})$  and exactly  $N^{\frac{1}{d}}$  in a  $d$ -dimensional MOB with  $N$  processors. The degree is  $O(d)$ , both in the split&hash network and in the MOB.

### 1.3 Organization of the Paper

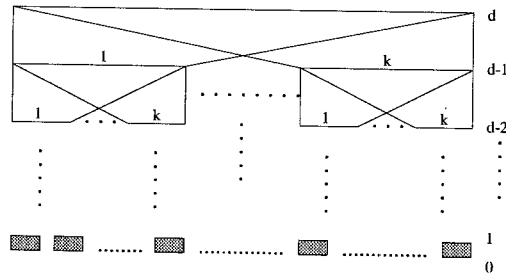
In the next section we describe the split&hash networks. Section 3 contains the algorithm for permutation routing on split&hash networks. The result is transferred to the MOB by simulation in Section 4. In Section 5 we give the extensions to random functions and  $h$ -relations. Most proofs are only sketched. Complete proofs are contained in a full version of this paper, available soon as technical report [MSS96]<sup>3</sup>.

## 2 The Split&Hash Network

A  $(N, d)$ -Butterfly-type network has the following structure. For  $d = 1$  it is the complete bipartite network with  $N$  sources and  $N$  sinks. For  $d > 1$ , it consists of  $k = N^{\frac{1}{d}}$  many  $(N^{\frac{d-1}{d}}, d-1)$ -Butterfly-type networks  $B_1, \dots, B_k$  and  $N$  new nodes, the sources. These sources are connected by  $k$  identical bipartite graphs, called *funnels*, to the sources of each  $B_j$ ,  $j = 1, \dots, k$ . These  $k$  funnels all share the same  $N$  sources. For an illustration see the picture below. The nodes on level 0 are the sinks, those on level  $d$  the sources.

(Note: Butterfly and Multibutterfly are  $(N, \log N)$ -Butterfly-type networks.)

In a Butterfly-type network a packet is sent from an input node to an output node by traveling along the unique sequence of sub-Butterfly-type networks. This sequence is called the *coarse path* of the packet.



<sup>3</sup> <http://www.uni-paderborn.de/cs/ellern.html> or  
<http://www.uni-paderborn.de/cs/fmadh.html>

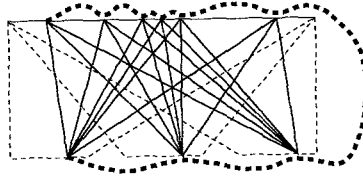
To specify an  $(N, d)$ -Butterfly-type network we have to specify the funnels.

Let  $a \in \mathbb{N}$  be fixed. The  $(N, d, a)$ -split&hash network is defined as follows. Between level  $i + 1$  and level  $i$  the following  $(k^{i+1}, k^i, a_i)$ -funnel is used. It is defined by  $a_i := \lceil a \cdot \frac{d}{i} \rceil$  functions  $h_j^i : [k^{i+1}] \rightarrow \left\{ \left\lfloor \frac{j k^i}{a_i} \right\rfloor + s \mid s \in \left[ \left\lfloor \frac{k^i}{a_i} \right\rfloor \right] \right\}$ ,  $j \in [a_i]^4$ . So the  $k^i$  bottom nodes of the funnel are split into  $a_i$  blocks of equal size, each  $h_j^i$ ,  $j \in [a_i]$ , maps to one block. We assume that each  $h_j^i$  is uniform, i.e.  $|(h_j^i)^{-1}(x)| = |(h_j^i)^{-1}(y)|$  for all  $x, y \in \text{image}(h_j^i)$ . So the funnel has the edges  $\{(l, h_j^i(l)) \mid l \in [k^{i+1}], j \in [a_i]\}$ .

A random  $(N, d, a)$ -split&hash network is obtained by choosing a random permutation  $\varphi : [k^{i+1}] \rightarrow [k^{i+1}]$  and putting  $h_j^i(l) := \varphi(l) \bmod \left\lfloor \frac{k^i}{a_i} \right\rfloor + j \cdot \left( \left\lfloor \frac{k^i}{a_i} \right\rfloor + 1 \right)$ . In the remainder of the paper we assume all split&hash networks to be *random* split&hash networks.

Since we want to deal with optical crossbars, we give the following optical crossbar version of the split&hash network. Note that every node of level  $d, \dots, 2$  is a top node of a group of  $k$  funnels. Every  $l$ -th bottom node of a funnel in the group is connected to the same set of top nodes. So we replace all edges adjacent to any  $l$ -th bottom node by one optical crossbar connecting the  $i$ -th bottom nodes with the top nodes adjacent to the  $l$ -th bottom nodes. Due to the choice of the functions  $h_j^i$  the optical crossbar has length  $k + k \cdot a_i$ . Further we replace the complete bipartite graphs between level 1 and 0 by optical crossbars of length  $2 \cdot k$ .

The picture shows the optical crossbar (thick dotted line) replacing the edges of two functions in three funnels in a split&hash network for  $k = 3$ .



**Remark A** A  $(N, d, a)$ -split&hash network has depth  $d$ , optical busses of length  $O(N^{\frac{1}{d}})$  and degree  $O(d)$ .

### 3 Permutation Routing

Our first result is about routing a permutation  $\pi : [N] \rightarrow [N]$  from the input nodes to the output nodes of a  $(N, d, a)$ -split&hash network, so the (unique) packet of the  $l$ -th source has to be delivered to the  $\pi(l)$ -th sink.

**Theorem 1.** Let  $k, d \in \mathbb{N}$ ,  $N := k^d$ . Any permutation can be routed from the sources to the sinks of a random  $(N, d, a)$ -split&hash network in  $O(d^2 \cdot \log d \cdot \log \log N)$  steps, with high probability, if  $a$  is a sufficiently large constant (independent of  $N$ ).

<sup>4</sup>  $[l]$  denotes the set  $\{0, \dots, l-1\}$

*Proof.* Fix permutation  $\pi$ . For  $\epsilon > 0$  (to be fixed later) we show how to route a batch of  $\epsilon \cdot N$  messages which contains the messages with destinations  $\pi(i)$  where  $\pi(i) \bmod (\frac{1}{\epsilon})$  has some fixed value. The complete routing consists of  $\frac{1}{\epsilon}$  batches.

The routing of a batch is performed in  $d$  rounds. At the beginning of round  $i$  all packets are on level  $d - i + 1$  in different processors. Round  $i$  delivers them to different processors of level  $d - i$  obeying their coarse path.

We have to show how a round can be done. The last round delivering the packets from level 1 to their destinations on level 0 takes at most one step, since only partial permutations on complete bipartite graphs have to be executed.

It remains to show how the other rounds can be realized. We make use of the following *Funnel Algorithm* which is a modification of an algorithm given in [MSS95]. The algorithm is executed in all  $(k^{i+1}, k^i, a_i)$ -funnels, i.e. all funnels of level  $i$ .

**Funnel Algorithm:**

set all processors holding a packet of the actual batch as active  
while there are active processors do

for  $k := 1$  to  $a_i$  do  
    if processor  $l \in [k^{i+1}]$  is active: try to send the packet to  $h_k^i(l)$   
    if sending was successful (i.e. only processor  $l$  tried to send a packet to  $h_k^i(l)$ ): set processor  $l$  as inactive

Every run through the ‘for’ loop is called a *round* of the algorithm.

**Main Lemma.** Let  $i \in \{2, \dots, d\}$ ,  $\beta > 1$ , and  $\epsilon > 0$  such that  $\epsilon\beta a_i < 1$ . The following holds for  $k$  large enough. (a) A set of  $\epsilon \cdot m := \epsilon \cdot k^{i+1}$  packets given on arbitrary distinct sources of a random  $(k^{i+1}, k^i, a_i)$ -funnel can be delivered to distinct sinks of the funnel using  $t_i := \frac{\log \log(k^i)}{\log(a_i - 1)} - 1$  rounds, with probability at least  $1 - m^{-\gamma \cdot \lceil \frac{d}{i} \rceil}$  for a constant  $\gamma > 0$  depending on  $a$  and  $\frac{1}{\epsilon}$ .

(b) A constant fraction of the  $\epsilon \cdot m$  packets is delivered to distinct bottom nodes after a constant number of rounds, with high probability.

(c) Each round contains  $a_i$  steps.

The *proof* is omitted due to space limitations. It is based on a proof given in [MSS95] for the running time of the so called  $(N, \epsilon, a, b, c)$ -scheme which is similar to the Funnel Algorithm given here. Our proof is much simpler than the one given in [MSS95] and can also be used in their situation.

Using this lemma we can bound the running time of our routing algorithm.

For any  $\gamma > 0$  the Main Lemma yields a probability of at most  $k^{-(i+1) \cdot \gamma \cdot \lceil \frac{d}{i} \rceil} \leq N^{-\gamma}$  for a packet to be undelivered after  $t_i$  rounds. So the probability for an undelivered packet in any funnel of level  $i$  is almost  $k^i \cdot N^{-\gamma}$ . Hence the probability for an undelivered packet in any funnel of the split&hash network is at most  $N^{-\gamma} \cdot \sum_{i=1}^{d-1} k^i \leq N^{-\gamma} \cdot k^d \leq N^{-\gamma+1}$ .

Using the Main Lemma the running time per batch of  $\epsilon \cdot N$  packets is at most

$$\begin{aligned} \sum_{i=1}^{d-1} t_i \cdot a_i &\leq d \cdot \frac{\log \log N}{\log(a-1)} \cdot \log d \\ &= O(d \cdot \log d \cdot \log \log N). \end{aligned}$$

As it suffices to chose  $\frac{1}{\epsilon} = O(d)$  Theorem 1 follows.  $\square$

## 4 Simulation on the Mesh of Busses

**Theorem 2.** *Let  $k, d, a \in \mathbb{N}$ ,  $N := k^d$ . Any permutation can be routed on the  $d$ -dimensional MOB of edge-size  $k$  using  $O\left(d^4 \cdot \log d \cdot \frac{\log \log N}{\log(a-1)}\right)$  steps, with high probability, if  $a$  is a sufficiently large constant.*

*Proof.* The proof is done by simulating the split&hash network on the MOB. We embed a  $(N, d, a)$ -split&hash network into the  $d$ -dimensional MOB of edge-size  $k = N^{\frac{1}{d}}$  in the natural way, embedding the  $l$ -th node of every level into node  $(l_1, \dots, l_d)$  of the MOB where  $\sum_{\nu=1}^d l_\nu \cdot k^{\nu-1} = l$ .

Consider a level  $i$  of the split&hash network. Since we define the functions  $h_j^i$  as folded permutations we can find partial permutations  $\psi_j^{i,\iota}$ ,  $\iota \in [a_i]$ , so that  $(\psi_j^{i,\iota}(l), h_j^i(l))$  is an edge of the MOB for one  $\iota \in [a_i]$  if  $(l, h_j^i(l))$  is an edge of the split&hash network.

In a preprocessing phase, we initialize the MOB such that all these  $\sum_{i=1}^{d-1} a_i^2 = O(d^3)$  permutations can be routed in time  $O(d)$ . (Note that the choice of the permutations does not depend on the input permutation.) So a step of the Funnel Algorithm can be replaced by  $a_i$  phases, each routing one of the permutations described above and then sending the packet to the destination sink of the funnel. If a node gets more than one packet while simulating a step it rejects all received packets, thus we ensure that the packets are delivered to distinct nodes. Since  $a_i = O(d)$ , simulating a step of the Funnel Algorithm requires  $O(d^2)$  steps on the MOB.  $\square$

## 5 Other Routing Problems

In Section 3 we showed how to realize a permutation on the split&hash network. In this section we will extend the results to  $h$ -relations and random functions. The result for random functions is a corollary of our Theorems 1 and 2:

**Corollary 3.** *A random function can be routed*

- (a) *from the input nodes to the output nodes of a  $(N, d, a)$ -split&hash network in  $O\left(\frac{\log N}{\log \log N}\right)$  steps,*
- (b) *on a  $d$ -dimensional MOB of edge-size  $k$  using  $O(d^4 \cdot \log d \cdot \log \log N + \frac{\log N}{\log \log N})$  steps,*

*with high probability, if  $d < \log^{\frac{1}{2}} N$ .*

*Proof.* We just have to assure that each funnel on each level gets no more than  $\epsilon \cdot k^{i+1}$  packets.

Since even the smallest funnels have a size of  $k = N^{\frac{1}{d}} = \Omega(\log N)$ , a random choice of  $\epsilon \frac{1}{\rho^\epsilon}$  packets assures for any  $\rho > 1$  that at most  $\epsilon \cdot k^i$  packets have to

travel through a certain funnel, with high probability, if  $N^{\frac{1}{d}}$  is large enough. So we choose batches of size  $N \cdot \frac{\epsilon}{\rho \epsilon}$  and make use of the Funnel Algorithm to deliver the packets from level  $d$  to level 1.

After delivering all packets of all batches to level 1 we have to deliver the packets to their destinations on level 0. This may be done using the  $h$ -relation algorithm of [GJLR93] on the complete bipartite graphs for  $h = \frac{\log N}{\log \log N}$ , since each node of level 1 gets at most  $\frac{\rho \epsilon}{\epsilon} = O(d) = O\left(\frac{\log N}{\log \log N}\right)$  packets, and each sink is the destination of at most  $O\left(\frac{\log N}{\log \log N}\right)$  packets, with high probability. The  $h$ -relation algorithm is repeated  $O(d)$  times, so the failure probability is sufficiently small.  $\square$

**Theorem 4.** *Any  $h$ -relation can be routed*

- (a) *from the input nodes to the output nodes of a  $(N, d, a)$ -split&hash network in  $O(d^3 \cdot \log d \cdot \log \log N + d^3 h)$  steps,*
- (b) *on the  $d$ -dimensional MOB of edge-size  $k$  in  $OO(d^5 \cdot \log d \cdot \log \log N + d^3 h)$  steps,*

*with high probability, if  $h \leq \log N$ ,  $d < \log^{\frac{1}{2}} N$  and  $N$  large enough.*

*Proof.* (SKETCH)

(a) Consider  $\epsilon \cdot k^{i+1} \cdot h$  packets on the input nodes of a funnel. Each node to contain at most  $h$  packets. We deliver the main part of the packets to the output nodes of the funnel by a modified version of the so called *Thinning Phase* introduced in [AM88] and also used in [GJLR93].

A node is called *active* if it contains at least one packet.

Let  $T \in \mathbb{N}$  and  $t_i = O\left(\frac{h}{2^i} + \log h\right)$ .

Modified Thinning Phase

for  $g := 1$  to  $\log h$  do

  for  $j := 1$  to  $t_i$  do

    every active input node chooses a random  $l \in \{1, \dots, \frac{h}{2^{g-1}}\}$ .

    if the node contains at least  $l$  packets it takes part at the Funnel

    Algorithm using functions  $h_0^j, \dots, h_{a-1}^j$  for  $T$  steps using the  $l$ -th packet.

  if a node contains more than  $\frac{h}{2^g}$  packets it becomes inactive.

Part (b) of our Main Lemma assures that a certain packet has a constant probability to be delivered in a constant number of steps of the Funnel Algorithm. So the analysis for the Thinning Phase as given by Goldberg et al. can be modified yielding: With probability  $1 - \left(\frac{1}{k^i}\right)^{\gamma \log k^i}$ , for some  $\gamma > 0$  depending only on the choice of the  $T, \epsilon$  and  $\beta$  it holds: (i) there are at most  $k^i$  undelivered packets, (ii) at most  $\frac{k^{i+1}}{h^{2^i}}$  nodes of the splitter become inactive in round  $k$  of the outer loop. The running time is obviously  $t := O(T \cdot a_i \cdot \log^2 h + T \cdot a_i \cdot h)$ . To ensure that at most  $h$  packets are on the node of the next level we use  $s \cdot a_i$  functions with distinct images using each group of  $a_i$  functions at most  $h$  times. The routing is performed for each of  $\frac{1}{\epsilon}$  batches chosen as in Theorem 1. For each batch the Modified Thinning Phase is applied for  $d$  rounds. Round  $i$  delivers a

main part of the packets at level  $d - i + 1$  to level  $d - i$ . After that the packets are delivered from level 1 to their destination on level 0 using the algorithm of Goldberg et al.  $O(d)$  times.

Now each funnel of level  $i$  contains at most  $k^i$  packets, with high probability. These remaining packets are sent using the random function algorithm. To apply this algorithm we distribute the packets of each level, so that each node contains at most two packets, with high probability. This may be done by partitioning the top node sets into distinct sets of size  $O(\log^3 k^{i+1})$ . With high probability, each set contains at most 2 · size packets, which can be distributed using a parallel prefix algorithm on each set. After that the remaining packets are delivered to their destinations by applying the random function algorithm once for each level. To prove part (b): There are only  $O(d)$  additional permutations needed to bring together the sets of the partitions used for the redistribution.  $\square$

## References

- [AM88] Richard J. Anderson and Gary L. Miller. Optical communication for pointer based algorithms. Technical Report CA 90089-0782 USA, Computer Science Department, University of Southern California, 1988.
- [DM93] Martin Dietzfelbinger and Friedhelm Meyer auf der Heide. Simple efficient shared memory simulations. In *Proceedings of the 5th Ann. Symp. on Parallel Algorithms and Architectures*, 1993.
- [GJLR93] Leslie Ann Goldberg, Mark Jerrum, Tom Leighton, and Satish Rao. A doubly logarithmic communication algorithm for the completely connected optical communication parallel computer. In *Proceedings of the 5th Ann. ACM Symp. on Parallel Algorithms and Architectures*, pages 300–309, 1993. Preprint.
- [GJLR94] Leslie Ann Goldberg, Mark Jerrum, Tom Leighton, and Satish Rao. Doubly logarithmic communication algorithms for the completely connected optical communication parallel computers. Preprint, 1994.
- [McD89] C. McDiarmid. On the method of bounded differences. *Combinatorics, London Math. Soc. Lecture Notes Series*, (141):148–188, 1989.
- [MSS95] Friedhelm Meyer auf der Heide, Christian Scheideler, and Volker Stemann. Exploiting storage redundancy to speed up randomized shared memory simulations. In *Annual Symposium on Theoretical Aspects of Computer Science*, to appear in *Theoretical Computer Science*, 1995.
- [MSS96] Friedhelm Meyer auf der Heide, Klaus Schröder, and Frank Schwarze. Routing on networks of optical crossbars (Technical Report). Technical report, Heinz Nixdorf Institute and Computer Science Department, University Paderborn, 1996. to appear.
- [Upf89] Eli Upfal. An  $O(\log N)$  deterministic packet routing scheme. In *Symposium on Theory of Computing*. ACM, 1989.