Optimally Scaling Permutation Routing on Reconfigurable Linear Arrays with Optical Buses

Jerry L. Trahan*[†] Anu G. Bourgeois[†] Yi Pan*[§] Ramachandran Vaidyanathan*[†]

[†] Department of Electrical & Computer Engineering
Louisiana State University, Baton Rouge, Louisiana, USA

[§] Department of Computer Science
University of Dayton, Dayton, Ohio, USA

Abstract

We present an optimal and scalable permutation routing algorithm for three reconfigurable models based on linear arrays that allow pipelining of information through an optical bus. Specifically, for any $P \leq N$, our algorithm routes any permutation of N elements on a P-processor model optimally in $O\left(\frac{N}{P}\right)$ steps. This algorithm extends naturally to one for routing h-relations optimally in $O(h)$ steps.

We also establish the equivalence of the three models, LARPBS, LPB, and POB. This implies an automatic translation of algorithms (without loss of speed or efficiency) among these models.

1. Introduction

An optically pipelined bus offers advantages over an electronic bus system of unidirectional propagation and predictable delays. These two properties enable synchronized concurrent access to an optical bus in a pipelined fashion [2, 8]. Combined with the abilities of the bus structure to broadcast and multicast, this architecture suits many communication-intensive applications. Researchers have proposed several models based on pipelined optical buses as practical parallel computing platforms [2, 6, 7]. Many parallel algorithms exist for arrays with pipelined buses [3, 5, 6, 7, 9, 11], indicating that such systems are very efficient for parallel computation due to the high bandwidth available by pipelining messages.

Optical buses provide efficient communication in a parallel computer system. Increasing the length of an optical bus introduces many problems, such as longer bus cycle, power distribution, and signal degradation. Smaller sized arrays are an unavoidable necessity. This requires algorithms to remain efficient regardless of problem size and introduces the problem of scaling algorithms to run on smaller models.

In this paper we present an optimal and scalable permutation routing algorithm for three reconfigurable models based on linear arrays that allow pipelining of information through an optical bus. This algorithm readily translates to an optimal $O(h)$ step algorithm for routing h-relations. The three models are the *Linear Array with a Reconfigurable Pipelined Bus System* (LARPBS) [6, 11], the *Pipelined Optical Bus* (POB) [3], and the *Linear Pipelined Bus* (LPB) [5].

2. Contributions of This Work

We will show that the LARPBS, LPB, and POB are equivalent in the sense that any algorithm proposed for one of these models can be implemented on either of the others with the same number of processors and to within a constant factor of the same time (Theorem 2). The segmenting ability of the LARPBS simplifies algorithm design, yet, due to the equivalence of these models, it is not necessary to include segment switches.

Permutation routing is one of the most fundamental data movement algorithms in parallel systems. On a system with a pipelined optical bus, this problem has a straightforward solution [6]. We consider a "scaled" version of the problem involving permutation routing of N elements on a P-processor LARPBS, for any $P \leq N$; its solution is quite involved. Here each processor holds $\frac{N}{P}$ (possibly a non-constant number of) elements. A naive approach can create hot-spots in which several messages destined for the same processor are scheduled during the same step. Our result is a scalable algorithm that permutes N elements optimally on a P-processor LARPBS (and hence the LPB and POB as well) in $O\left(\frac{N}{P}\right)$ steps, for any $P \leq N$. This algorithm adapts to one for routing h-relations optimally.

*Supported in part by the National Science Foundation under Grant No. CCR-9503882.

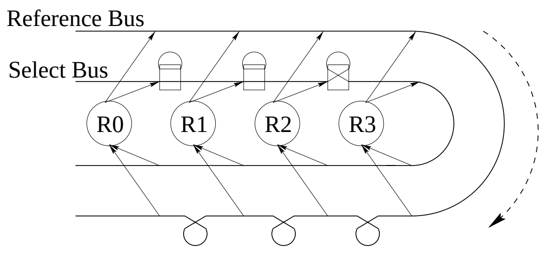

Figure 1. Structure of an LARPBS

The only method so far with an optimal deterministic solution to these problems uses the *Linear Array with Reconfigurable Optical Buses* (LAROB)[7]. The LAROB allows its processors to use counters and bit polling within a bus cycle and extra hardware capabilities beyond the more restricted models used in this paper. Rajasekaran and Sahni [9] designed an optimal randomized algorithm. In contrast, our algorithm is deterministic.

3. Model Descriptions

Let an optically pipelined bus have the same length of fiber between consecutive processors, so propagation delays between consecutive processors are the same. Let a *bus cycle* be the end-to-end propagation delay on the bus. We specify time complexity in terms of a step comprising one bus cycle and one local computation. For more details on the time complexity issue, see Guo *et al.* [2] and Pan and Li [6].

3.1. Model Structures

In the LARPBS, as described by Pan and Li [6], the optical bus is composed of three waveguides, one for carrying data (the *data waveguide*) and the other two (the *reference* and *select waveguides*) for carrying address information (see Figure 1). (For simplicity, the figure omits the data waveguide, as it resembles the reference waveguide.) Each processor connects to the bus through two directional couplers, one for transmitting and the other for receiving [2, 8]. The receiving segments of the reference and data waveguides contain an extra segment of fiber of one unit pulse-length, Δ, between each pair of consecutive processors (shown as a delay loop in Figure 1). The transmitting segment of the select waveguide also has a switch-controlled conditional delay loop of length Δ between processors R_i and R_{i+1}, for each $0 \leq i \leq N-2$ (Figure 1).

To allow segmenting, the LARPBS has optical switches on the transmitting and receiving segments of each bus for each processor. With all switches set to *straight*, the bus system operates as a regular pipelined bus system. Setting the switches at R_i to *cross* segments the whole bus system into two separate pipelined bus systems, one consisting of processors $R_0, R_1, \cdots, R_i$ and the other consisting of R_{i+1}, $R_{i+2}, \cdots, R_{N-1}$.

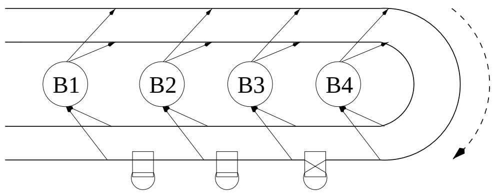

Figure 2. Structure of a POB

The *Linear Pipelined Bus* (LPB) [5] is identical to the LARPBS with the exception that it does not have any segment switches. The *Pipelined Optical Bus* (POB) [3] is a similar model. Conditional delay switches, however, are positioned on the receiving side of the reference and data waveguides, rather than on the transmitting side of the select line (see Figure 2). The POB contains no segment switches, so it is not able to segment its bus.

3.2. Addressing Techniques

The models use the *coincident pulse technique* [8] to route messages by manipulating the relative time delay of *select* and *reference* pulses on separate buses so that they will coincide only at the desired receiver, signaling the processor to read the corresponding data frame. Each processor has a *select frame* of N bits (*slots*), of which it can inject a pulse into a subset of the N slots. The coincident pulse technique admits broadcasting and multicasting of a single message by appropriately introducing multiple select pulses within a select frame.

When multiple messages arrive at the same processor in the same bus cycle, it receives only the first message and disregards subsequent messages that have coinciding pulses at the processor. (This corresponds to priority concurrent write.) Denote the processor that has a select pulse injected in its slot in a select frame for a particular message as the *selected destination.* The *actual destination* denotes the processor that detects the coinciding reference and select pulses. (The selected and actual destinations may differ due to conditional delay loops and segmenting.)

4. Equivalence of the LARPBS, LPB, and POB

In this section, we prove that the LARPBS, LPB, and POB are equivalent. That is, each model can simulate a step of either of the two other models in constant time, using the same number of processors. (For a more detailed description on the equivalence of models see Trahan *et al.* [10].) We prove the equivalence of the three optical models by a cycle of simulations, one of which is sketched below. Each simulation consists of the following three phases: (i) determine the actual destinations for all messages, (ii) create the select and message frames, and (iii) send the messages.

Lemma 1 *Each step of an N processor LARPBS can be simulated by an N processor LPB in $O(1)$ steps.*

Sketch of proof: By combining the location of the select pulses within the select frame (selected destinations), the information on set segment switches, and the number of set conditional delay switches, each processor can determine the actual destination processors for its message. Each processor then adjusts its select frame to reflect the destinations computed in the previous step. At this point, processors set all delay switches to straight and transmit their messages. The simulation properly handles any concurrent-read or concurrent-write step of the LARPBS. $\blacksquare$

We have also constructed similar simulations of an LPB by a POB and of a POB by an LARPBS, establishing the equivalence of these models.

Theorem 2 *The LARPBS, LPB, and POB are equivalent models. Each one can simulate any step of one of the other models in $O(1)$ steps with the same number of processors.*

5. Permutation and h-Relations Routing

Permutation routing is one of the most important data movement algorithms in parallel systems. Let $\mathcal{N} = \{0, 1, \cdots, N-1\}$ and let $\pi : \mathcal{N} \longrightarrow \mathcal{N}$ be a bijection. Permutation routing of N elements on an N-processor system refers to sending information from processor i to processor $\pi(i)$, for each $i \in \mathcal{N}$. We consider a "scalable" version of the problem involving permutation routing of N elements on a P-processor LARPBS, for any $P \leq N$. An h-relation is a routing problem in which each processor sends and receives at most h pieces of information [12].

For a given algorithm and problem size, let $T(N)$ be the time to run the algorithm with N processors on model $\mathcal{M}$, where N is the number of processors required to run the algorithm as fast as possible. For the same problem size, let $T(P)$ be the time to run the algorithm adapted for P processors, where P is independent of N, except that $P < N$. In general, $T(P) = T(N) \cdot \frac{N}{P} \cdot g(N, P)$. If $g(N, P)$ is a constant, then the algorithm scales *optimally.* We present a permutation routing algorithm that scales optimally. We then extend this algorithm to route an h-relation on a P-processor LARPBS optimally in $O(h)$ steps. Although we present the algorithms for the LARPBS, the equivalence established in Theorem 2 immediately extends these results to the LPB and POB as well.

We treat scaling permutation routing from N to P processors as a simulation of the operation by an N-processor LARPBS, $\mathcal{Q}$, on a P-processor LARPBS, $\mathcal{R}$. Let Q_i denote the i^{th} processor of $\mathcal{Q}$, and let R_j denote the j^{th} processor of $\mathcal{R}$. To perform this simulation on $\mathcal{R}$, map $\frac{N}{P}$ processors from $\mathcal{Q}$ to each processor of $\mathcal{R}$ using block mapping [11]. The main challenge to the simulation is the possibility of message conflicts if multiple processors of $\mathcal{R}$ attempt to simultaneously send messages to destination processors of $\mathcal{Q}$ that are all simulated on the same processor of $\mathcal{R}$.

We present two scaling simulations to cover different relative values of P and N. Our first simulation uses as a base the key steps of an algorithm from Valiant [12] for routing an h-relation: sort messages according to their destinations, then use information from the sorting to route them without further contention. This simulation applies to the case $P \leq \sqrt{N}$. For the case $P > \sqrt{N}$, our second simulation adapts to the LARPBS a scaling simulation of the Passive Optical Star model by Berthomé *et al.* [1]. As a consequence of these simulations, permutation routing scales optimally on the LARPBS. We extend each scaling simulation to routing h-relations for appropriate values of P with respect to h.

5.1. Permutation Routing: Case $P \leq N^{1/2}$

Assign one of $\frac{N}{P}$ *colors* to each processor of $\mathcal{Q}$ such that each processor of $\mathcal{R}$ simulates one processor of $\mathcal{Q}$ of each color. For example, since processors of $\mathcal{Q}$ are mapped to processors of $\mathcal{R}$ in a block mapping, then assign color $i \bmod \frac{N}{P}$ to processor Q_i, for $0 \leq i < N$.

Assume that each processor of $\mathcal{R}$ initially holds the selected destination of the message to be sent by each of the $\frac{N}{P}$ processors it simulates. $\mathcal{R}$ now determines the actual destination of each message in $O(\frac{N}{P})$ steps in a way similar to that sketched in the proof of Lemma 1. Recall that the actual destination processor is the processor of $\mathcal{Q}$ at which a select pulse actually coincides with the reference pulse. Assign to each message the color of its actual destination. Observe for any R_j that there are no constraints on the collection of colors of messages sent by processors of $\mathcal{Q}$ that it simulates.

The strategy is to sort messages by destination color in $O(\frac{N}{P})$ steps such that each processor holds one message of each color, then write all messages with the same color at the same time, hence, without conflict. As in Berthomé *et al.* [1], we adapt the mesh sorting algorithm of Marberg and Gafni [4]. This algorithm sorts mn elements on an $m \times n$ mesh in $O(m+n)$ time, in a constant number of phases, where each phase is a row or a column sort. We view the N elements on P processors as a $P \times \frac{N}{P}$ array, so that each processor holds a row of the array. A processor can perform a row sort in $O(\frac{N}{P})$ steps, as there are $\frac{N}{P}$ elements, each in the range $0, \ldots, \frac{N}{P} - 1$. To perform a column sort, transpose the array [11], sort multiple columns locally in a processor, then transpose back in place. Altogether, a column sort takes $O(\frac{N}{P})$ steps. Since the algorithm comprises a constant number of row and column sorts, the LARPBS $\mathcal{R}$ can sort the messages by color in $O(\frac{N}{P})$ steps. At the end of the sorting algorithm, messages are sorted in row major order, so transpose them, leaving each processor with one message of each color.

Finally, in $\frac{N}{P}$ rounds, processors write all messages with the same color in the same round. Because each processor of $\mathcal{R}$ simulates exactly one processor of $\mathcal{Q}$ of each color and because each processor of $\mathcal{Q}$ is the destination of at most one message, each round is an exclusive-read, exclusive-write communication for $\mathcal{R}$, so all messages reach their destinations in $O(\frac{N}{P})$ steps.

Lemma 3 *If $P \leq \sqrt{N}$, a permutation routing step of an N-processor LARPBS scales optimally to a P-processor LARPBS in $O(\frac{N}{P})$ steps.*

5.2. Permutation Routing: Case $P > N^{1/2}$

We next present a simulation that adapts to the LARPBS a scaling simulation of the Passive Optical Star (POS) model by Berthomé *et al.* [1]. Our algorithm is deterministic, in contrast to the randomized scaling simulation of Berthomé *et al.* Similarities between the two optical models allow us to build upon the structure of the POS simulation, while differences between the models require us to construct alternative approaches to various phases of the simulation.

The simulation works in the following phases.

1. **Balance writers:** As above, we assign colors to messages. Determine an arrangement of messages such that colors are "balanced" at each processor, that is, such that each processor holds at most a constant number of messages of each color. Refer to these processors as (balanced) *intermediate writers*. Route messages from original writers to intermediate writers.

2. **Route to final destination:** Route messages from intermediate writers to their final destinations. Because the intermediate writers are balanced and so are the receivers, that is, each is the destination of at most a constant number of messages of each color, this step is easy to implement in $O(\frac{N}{P})$ steps.

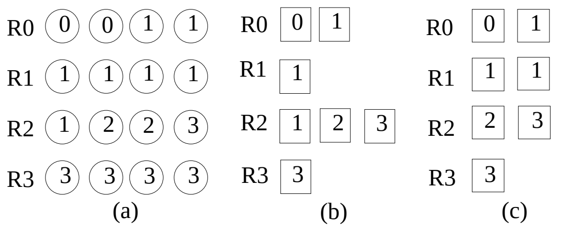

Figure 3. Balancing Writers Phase for $\frac{N}{P} = 4$: (a) Sorting in Small Groups; (b) Packing Supertokens; (c) Balance within Groups.

Balancing Writers Phase

As above, assign one of $\frac{N}{P}$ *colors* to each processor of $\mathcal{Q}$ such that each processor of $\mathcal{R}$ simulates one processor of $\mathcal{Q}$ of each color.

Assume that each processor of $\mathcal{R}$ initially holds the selected destination of the message to be sent by each of the $\frac{N}{P}$ processors it simulates. $\mathcal{R}$ now determines the actual destination of each message in $O(\frac{N}{P})$ steps. Assign to each message the color of its actual destination. Observe for any R_j that the colors can be arbitrary and there are no constraints on the collection of colors of the selected (or actual) destinations of messages sent by processors of $\mathcal{Q}$ that it simulates.

Each R_j creates a *token* for each message written by a processor Q_i that it simulates. A token contains the identity of the writing processor in $\mathcal{Q}$, the identity of the actual destination processor in $\mathcal{Q}$, the color (of the destination), and the message. Observe that $\mathcal{R}$ holds up to N tokens colored by $\frac{N}{P}$ colors, and with each of its P processors containing up to $\frac{N}{P}$ tokens.

Consider an example as shown in Figure 3, where $\frac{N}{P} = 4$ and the colors are numbered as $0, 1, 2,$ and 3. We will refer to this example in the following steps.

Sorting in Small Groups: Partition the processors into groups of $\frac{N}{P}$ consecutive processors each. Each group sorts tokens by color in $O(\frac{N}{P})$ steps.

Referring to the example, at the end of this step, Figure 3(a) shows the colors in sorted order. After sorting, each processor may hold tokens of the same color or it could hold tokens of multiple colors.

Packing Supertokens: Each processor of $\mathcal{R}$ now creates a *supertoken* representing all its tokens of the same color, for each color it holds. Each group of $\frac{N}{P}$ processors creates at most $2\frac{N}{P}$ supertokens, so $\mathcal{R}$ creates at most $2P$ supertokens overall [1]. This runs in $O(\frac{N}{P})$ steps.

Balance within groups: Each group balances the supertokens such that each processor holds at most two.

Sort supertokens: Sort the supertokens by color using radix sort [11] in $O(\log \frac{N}{P})$ steps.

Send back information: $\mathcal{R}$ sends back the location of each supertoken (after the sort) to the creator of the supertoken. To allow this reverse communication, during the previous steps, processors kept track of the sequence of the destinations of their messages.

Unpacking: In this stage, $\mathcal{R}$ sends tokens from the processor that created the supertoken to the holder of the supertoken after the sort.

Transpose: At this point, each processor of $\mathcal{R}$ holds all tokens corresponding to (up to) two supertokens. Each group can have at most two "light" supertokens of each color, that is, supertokens representing fewer than $\frac{N}{P}$ messages of a certain color. Each processor creates "dummy" tokens so that each supertoken corresponds to $\frac{N}{P}$ (real and dummy) tokens. There are $\frac{P^2}{N}$ small groups, so $\frac{P^2}{N} \cdot 2 \cdot (\frac{N}{P} - 1) < 2P$ dummy tokens are created for any particular color.

All tokens of the same color are in consecutive locations in consecutive processors. $\mathcal{R}$ now "transposes" all tokens from a block to a cyclic mapping in $O(\frac{N}{P})$ steps. Now, each processor holds a constant number of tokens (real or dummy) of each color, so the tokens are balanced.

Lemma 4 *If $P > \sqrt{N}$, a permutation routing step of an N-processor LARPBS scales optimally to a P-processor LARPBS, in $O(\frac{N}{P})$ steps.*

5.3. Results

Theorem 5 *Permutation routing on an N-processor LARPBS, LPB, or POB scales optimally to a P-processor LARPBS, LPB, or POB in $O(\frac{N}{P})$ steps.*

Theorem 6 *An h-relation can be optimally routed by a P-processor LARPBS, LPB, or POB in $O(h)$ steps.*

We have considered two cases for the permutation routing and h-relations algorithms. Neither approach to scaling a permutation routing step extends to the other case. In the case $P \le \frac{N}{P}$, the approach of finding balanced intermediate writers runs into the obstacle that the number of available processors (P) is smaller than the size of a small group ($\frac{N}{P}$), so the sorting in small groups does not apply. Also, the number of supertokens is at least $\frac{N}{P}$ because of having $\frac{N}{P}$ colors, but this can be superlinear in P. (Note: Berthomé *et al.* [1] do not seem to have accounted for this case.) In the case $P > \frac{N}{P}$, the current approach based on sorting takes time $\omega(\frac{N}{P})$ when $P = \omega(N^{\frac{k}{k+1}})$ for constant k.

References

[1] P. Berthomé, T. Duboux, T. Hagerup, I. Newman, and A. Schuster, "Self-Simulation for the Passive Optical Star Model," *Proc. European Symp. on Algs.*, (1995), pp. 369–380.

[2] Z. Guo, R. Melhem, R. Hall, D. Chiarulli, and S. Levitan, "Array Processors with Pipelined Optical Busses," *J. Parallel Distrib. Comput.*, vol. 12, (1991), pp. 269–282.

[3] Y. Li, Y. Pan, and S. Q. Zheng, "Pipelined TDM Optical Bus with Conditional Delays," *Optical Engineering*, vol. 36, (1997), pp. 2417–2424.

[4] J. M. Marberg and E. Gafni, "Sorting in Constant Number of Row and Column Phases on a Mesh," *Algorithmica*, vol. 3, (1988), pp. 561–572.

[5] Y. Pan, "Order Statistics on Optically Interconnected Multiprocessor Systems," *Optics and Laser Technology*, vol. 26, (1994), pp. 281–287.

[6] Y. Pan and K. Li, "Linear Array with a Reconfigurable Pipelined Bus System: Concepts and Applications," *Informations Sciences – An International Journal*, vol. 106, (1998), pp. 237–258.

[7] S. Pavel and S. G. Akl, "Integer Sorting and Routing in Arrays with Reconfigurable Optical Buses," *Proc. Int'l. Conf. Par. Processing*, (1996), pp. III-90–III-94.

[8] C. Qiao and R. Melhem, "Time-Division Optical Communications in Multiprocessor Arrays," *IEEE Trans. Comput.*, vol. 42, (1993), pp. 577–590.

[9] S. Rajasekaran and S. Sahni, "Sorting, Selection and Routing on the Arrays with Reconfigurable Optical Buses," *IEEE Trans. Parallel Distrib. Systems*, vol. 8, (1997), pp. 1123–1132.

[10] J. L. Trahan, A. G. Bourgeois, and R. Vaidyanathan, "Tighter and Broader Complexity Results for Reconfigurable Models," *Parallel Proc. Letters*, vol. 8, (1998), pp. 271–282.

[11] J. L. Trahan, Y. Pan, R. Vaidyanathan, and A. G. Bourgeois, "Scalable Basic Algorithms on a Linear Array with a Reconfigurable Pipelined Bus System," *Proc. Int'l. Conf. on Parallel and Distributed Computing Systems*, (1997), pp. 564–569.

[12] L. G. Valiant, "General Purpose Parallel Architectures," in *Handbook of Theoretical Computer Science*, J. van Leeuwen, ed., MIT Press, (1990), pp. 943–972.