COMPUTER SCIENCE PUBLICATION

EFFICIENT COMPUTATIONS ON MESHES WITH EXPRESS LINKS

Steven Cheung and Francis C.M. Lau

Technical Report TR-95-01

May 1995

DEPARTMENT OF COMPUTER SCIENCE
FACULTY OF ENGINEERING
UNIVERSITY OF HONG KONG
POKFULAM ROAD
HONG KONG

Efficient Computations on Meshes with Express Links

Steven Cheung
Department of Computer Science
University of California
Davis, CA 95616, USA

Francis C.M. Lau
Department of Computer Science
The University of Hong Kong
Hong Kong

May 2, 1995

Abstract

To overcome the diameter problem of meshes, enhanced mesh models equipped with various kinds of buses have been proposed. In this short note, we examine a weaker model that does not use any bus mechanism. In this model, we stretch some of the links in an ordinary mesh to connect non-neighboring nodes. Our model, which is just an ordinary mesh, is more attractive from the implementation point of view when compared with the enhanced models. The "express links" in our model turn out to be surprisingly useful: we show that, given N values, an N-processor two-dimensional mesh with express links can solve semigroup computations in $\Theta(N^{1/4})$ steps, prefix computations in $\Theta(N^{1/4})$ steps, the median row problem in $\Theta(N^{1/4})$ steps, median finding in $O(N^{1/4}\log N)$ steps, and the all points closest neighbor problem in $\Theta(N^{1/4})$ steps, whereas these problems require $\Omega(N^{1/2})$ steps on a two-dimensional mesh without express links.

Index Terms—Interconnection networks, mesh-connected computers, parallel algorithms, parallel processing, semigroup computation.

1 Introduction

One of the most frequently cited problems of low-dimensional meshes is that they have a large diameter. In other words, it may take a long time for a processor to send a packet to another processor. Various kinds of enhanced meshes equipped with faster means for long-distance data movements have been proposed to counteract this diameter problem. Examples of two-dimensional enhanced meshes include meshes with a global bus [2] [5] [18], meshes with multiple buses [3] [6] [12], meshes with hierarchies of buses [15], meshes with separable buses [13] [16], and reconfigurable meshes [4].[1] These models have been

[1] In reconfigurable meshes, mesh links can be connected together to form many different configurations of buses.

demonstrated to be powerful in solving a wide variety of problems. However, in terms of implementation, broadcast buses and reconfigurable buses have longer propagation delay and lower throughput than mesh links [11] [16]. Point-to-point links are easier to build than broadcast or reconfigurable buses (which must be capable of connecting a reasonable number of processors) because of the simplicity of the former in terms of hardware implementation. In fact, point-to-point links appear to be the dominant choice of connection for modern-day parallel computers, as is evident in many practical examples including Intel-CMU iWarp, Intel iPSC and Paragon, nCUBE 2, Thinking Machines CM-2, Caltech Mosaic C, Texas Instruments C40, Inmos T9000 transputer, Ametek 2010, MIT J-Machine, and Stanford DASH. In this paper, we show, by connecting some of the non-neighboring nodes through ordinary links in mesh-connected parallel computers, how we can solve the semigroup computation problem and other problems efficiently. We call these links that connect two non-neighboring nodes "express links". Our conclusion is that meshes with express links can perform competitively in solving certain basic problems when compared to meshes that are enhanced with buses. Two notes are in order. First, these express links are just ordinary mesh links, and are unlike the express channels in Dally's express cubes, which require some special interchange hardware for their operation [7]. Second, as we will see later on, such express links could be quite sparse in a given mesh and still the resulting configuration can solve our problems with the desired performance; given this fact, a mesh with express links retains its flavor of an ordinary mesh, and the additional cost due to express links, if any, is minimal.

In the next section, we present the mesh with express links model and and compare our results on semigroup computations using this model to related works. After that, we give the lower bounds for any non-trivial problem on meshes with express links. Then we present matching upper bounds for semigroup computations on one-, two-, and higher-dimensional meshes with express links as well as on some variants of the model. We have also considered other basic problems using this model; we state at the end our results for these problems.

2 Meshes with Express Links

In a two-dimensional mesh with express links, each row (resp. column) has some terminals (processors at which express links end). Adjacent terminals, separated by a certain fixed distance as required by the algorithm, are connected together by express links. Each processor is capable of computing some arithmetic or boolean operations, and sending at most one packet and receiving at most one packet in a time step—this is a one-port model [9]. Links are unidirectional in the sense that a link can transmit in one direction only during

any particular time step.[2] We assume that the propagation delay of the express links is one time step, independent of their length. This is supported by some empirical findings that node delays dominate wire delays. In other words, we can stretch links to span a moderate distance without significantly affecting the link speed [7] [1] [8]. The constant propagation delay assumption is also used in meshes with broadcast or reconfigurable buses [3] [4] [6] [10] [12] [16] [18], which, however, is less reasonable. In fact, the propagation delay of buses is likely to be proportional to $\log C$, where C is the number of processors connected to the bus [5] [16]. For more justifications of our model based on links only, readers are referred to [7] [11]. Figure 1 shows examples of one- and two-dimensional meshes with express links.

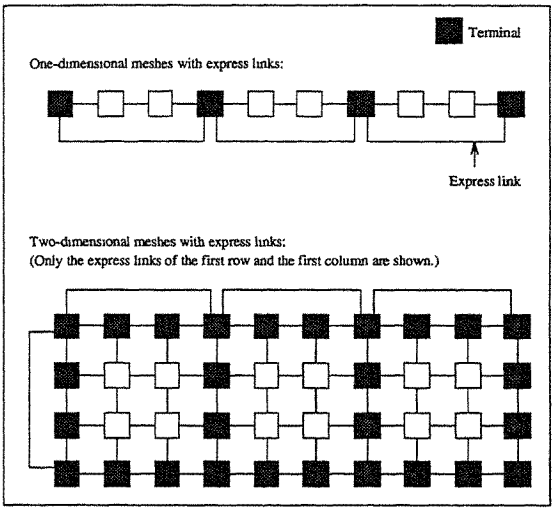


Figure 1: Meshes with express links.

3 Semigroup Computations

A semigroup computation (finding maximum/minimum, parity, and sum being special cases) can be described by a pair $(\oplus, S)$ where $\oplus$ is an associative binary operator and

[2]For the algorithms in this paper, unidirectional links are sufficient. In practice, if the links are bidirectional, the unused bandwidth could be used for another computation such as one of our algorithms but running in the reversed orientation.

Model	Dimension	Time
Ordinary meshes	$N^{1/2} \times N^{1/2}$	$\Theta(N^{1/2})$
Meshes with a global bus	$N^{1/2} \times N^{1/2}$	$\Theta(N^{1/3})$ [5] [18] [2]
Ordinary meshes with express links	$N^{1/2} \times N^{1/2}$	$\Theta(N^{1/4})$ [This paper]
Meshes with row and column buses	$N^{1/2} \times N^{1/2}$	$\Theta(N^{1/6})$ [12]
Meshes with row and column buses	$N^{3/8} \times N^{5/8}$	$\Theta(N^{1/8})$ [3] [6]
Meshes with separable row and column buses (one bus for every $N^{1/8}$ rows/columns)	$N^{3/8} \times N^{5/8}$	$\Theta(N^{1/8})$ [16]
Meshes with separable row and column buses (one bus per row/column)	$N^{1/2} \times N^{1/2}$	$O(\log N)$ [13]
Meshes with hierarchies of buses	$N^{1/2} \times N^{1/2}$	$O(\log N)$ [15]

Table 1: Comparison of our model to enhanced meshes on semigroup computations.

$S = \{a_0, a_1, ..., a_{N-1}\}$. The problem is to compute $a_0 \oplus a_1 \oplus ... \oplus a_{N-1}$.

Table 1 summarizes the results for semigroup computations on meshes with express links and on enhanced mesh models. For a practical range of N (from a few hundred to tens of thousand), meshes with express links represent a significant improvement over ordinary meshes that use links to connect only nearest neighbors, and have comparable performance to various enhanced mesh models.

The diameter of a mesh with express links, which is a lower bound for any non-trivial problem—one in which some processor must receive information from some other arbitrary processor—is determined by the maximum of the following two quantities:

- The maximum terminal-to-terminal distance.
- The maximum distance between a processor and the nearest terminal.

Thus, the best choice of the length of the express links is $N^{1/2}$ in the one-dimensional meshes; hence a lower bound of $\Omega(N^{1/2})$ for the 1D case. By the same token, the best choice for the length on the two-dimensional meshes is $N^{1/4}$; hence a lower bound of $\Omega(N^{1/4})$ for the 2D case.[3]

[3]Unlike meshes with multiple broadcast buses, skewed rectangular meshes with express links are inferior to square meshes with express links because the former have a larger diameter.

4 Semigroup Computations in One-Dimensional Meshes

Figure 2 depicts the basic idea of our algorithm (Algorithm 1) for solving the semigroup computation on one-dimensional meshes. This solution will be used in the next section as a building block for the higher-dimensional cases. The main technique we use is pipelining. To emulate sending multiple values over a long broadcast bus, we send them through a chain of express links in a pipelining fashion.

Algorithm 1 *Suppose we have an N-processor one-dimensional mesh with express links, and each express link is of length $N^{1/2}$. Each processor P_i has the value a_i initially, $0 \leq i \leq (N-1)$. There are two phases in the semigroup computation algorithm:*

1. *Every non-terminal processor P_i sends a_i to the processor on its left. When a non-terminal receives a packet from its right neighbor, it forwards the packet to its left neighbor. For each terminal, it performs the semigroup computation on the new value it receives in each step and the partial answer it has computed just before this step. Initially, the partial answer for a terminal is equal to the a_i assigned to it in the beginning.*

2. *All the partial answers are now stored in the terminals. The terminals send their partial answers to the leftmost terminal, which computes the final answer, using only the express links.*

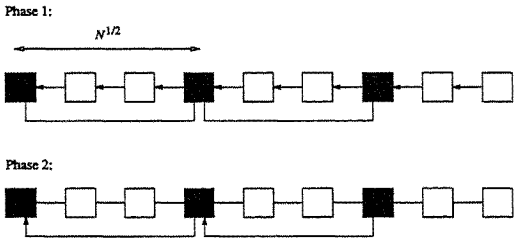


Figure 2: Semigroup computations on a 1D mesh with express links.

Time complexity: Both Phases 1 and 2 require $O(N^{1/2})$ steps. Thus the entire algorithm runs in $O(N^{1/2})$ steps.

5 Semigroup Computations in Two- and Higher-Dimensional Meshes

Algorithm 2 *Suppose we have an $n \times n$ mesh with express links, and each express link is of length $n^{1/2}$. $N(= n^2)$ is the total number of processors in the mesh. The a_i's are arranged in row-major order in the mesh.*[4] *That is, each processor P_i has the value a_i originally, where $i = (r \times n + c)$, and r (resp. c) is the row (resp. column) number of the processor and $0 \leq r, c \leq (n-1)$. The algorithm for solving the semigroup problem is as follows:*

1. *Apply Algorithm 1 to each row of the mesh. After that, the partial answer for each row resides in the leftmost processor of the corresponding row.*
2. *Apply Algorithm 1 to the leftmost column of the mesh (from bottom to top). At the end, the answer resides in the upper-leftmost terminal.*

Time complexity: Both Phases 1 and 2 require $O(n^{1/2})$ steps. Thus the entire algorithm runs in $O(n^{1/2})$ steps, or $O(N^{1/4})$ steps

Our results can be generalized to $n \times n \ldots \times n$ r-dimensional meshes with express links, each link of length $n^{1/2}$. For any given (fixed) r, we can prove that $\Theta(N^{1/2r})$ is the tight time bound for semigroup computations. Obviously, the diameter bound is $\Omega(n^{1/2})$. For the upper bound, we can apply our 1D result to each of the r dimensions successively.

To speed up the computation further, we can use the standard technique (*e.g.*, [14]) that puts multiple numbers in each processor. Suppose we want to perform a semigroup computation of N values on a P-processor 2D square mesh with express links, $P < N$. We can put (N/P) values in each processor, and have the processors compute the semigroup computation on its (N/P) values before invoking Algorithm 2. The running time of the entire computation is of order $\max((N/P), P^{1/4})$. By choosing P as $N^{4/5}$, we have the total running time of $O(N^{1/5})$.

6 Using Fewer Express Links

In this section, we show how to reduce the number of express links of a two-dimensional mesh without increasing the total running time of the semigroup computations.

[4] By symmetry, our algorithm can be easily adapted to handle the case of column-major order.

If the N values are in the submesh-row-major ordering,[5] or the operator $\oplus$ is commutative (*e.g.*, maximum, parity, sum, *etc.*), we can compute a semigroup computation on a mesh with express links, in which one row (resp. column) out of every $N^{1/4}$ rows (resp. columns) has express links, in $\Theta(N^{1/4})$ time steps. Figure 3 depicts this two-dimensional mesh with fewer express links.

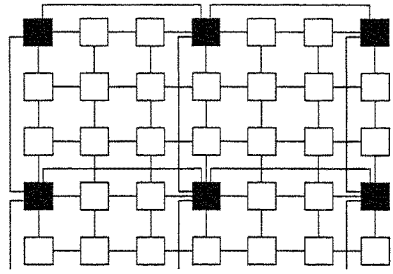


Figure 3: Meshes with fewer express links.

Algorithm 3 *(submesh-row-major order) Suppose we have an $n \times n$ mesh with express links. $N(= n^2)$ is the total number of processors in the mesh. Each express link is of length $n^{1/2}$, and only one row (resp. column) out of every $n^{1/2}$ rows (resp. columns) has express links. The mesh is partitioned into submeshes of size $n^{1/2} \times n^{1/2}$. The upper-leftmost processor in each submesh is the only terminal in that submesh. The semigroup computation algorithm has three phases:*

1. *Each submesh performs the semigroup computation on all the values in that submesh using ordinary links. The partial answer is stored in the terminal of that submesh.*

2. *The rest of this algorithm only uses express links for communication. The rightmost terminal of each row (that has express links) sends its partial answer to the second rightmost terminal. The second rightmost terminal performs the semigroup computation using its own partial answer and the one received from the rightmost terminal, and then sends the answer to its left neighboring terminal, and so on. After that, there are only $n^{1/2}$ partial answers left, one in each leftmost terminals.*

[5]Suppose we partition an $n \times n$ mesh into submeshes of size $n^{1/2} \times n^{1/2}$, where $n = N^{1/2}$, and number the submeshes in row-major order. The distribution of the N values is said to be in submesh-row-major order if the first n values (*i.e.*, $a_0,...,a_{n-1}$) are distributed in the first submesh in row-major order, and the next n values are distributed in the second submesh in row-major order, and so on.

3. *The leftmost terminals use a procedure similar to Phase 2 to compute the final answer, but instead of going from right to left, we computation goes from bottom to top. At the end, the final answer resides in the upper-leftmost terminal.*

Time complexity: Each phase completes in $O(n^{1/2})$ steps. Thus the entire algorithm runs in $O(n^{1/2})$ steps, or $O(N^{1/4})$ steps.

Algorithm 3 can be adapted to handle the case in which the N values are distributed in row-major order as well; the operator $\oplus$ is assumed to be non-commutative in this case.

Algorithm 4 *(row-major order)*

1. *Every row in each submesh carries out the semigroup computation, and the partial answer is stored in the leftmost processor of that row within the submesh. Then, all the partial answers of a submesh are sent to the terminal of that submesh.*

2. *Apply Phase 2 of Algorithm 3 to the $n^{1/2}$ rows of terminals. For each of these rows, we overlap the computation of the $n^{1/2}$ sets of partial answers (corresponding to the $n^{1/2}$ rows in a submesh): In step 1, the rightmost terminal starts the computation which goes from right to left for the first set of partial results. In step 2, the previous computation has come to the second rightmost terminal; at this time, the rightmost terminal starts the computation for the second set of partial answers; and so forth, until all the partial answers have been computed and stored in the leftmost terminals.*

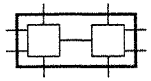
3. *Similar to Phase 2, except that the computations go from bottom to top. The final answer is stored in the upper-leftmost terminal.*

Time complexity: Although there are $n^{1/2}$ sets of answers to compute in both Phase 2 and Phase 3, because of the overlapping, the complexity of each of these phases is still $O(n^{1/2})$ steps, and hence the the algorithm runs in $O(n^{1/2})$ steps.

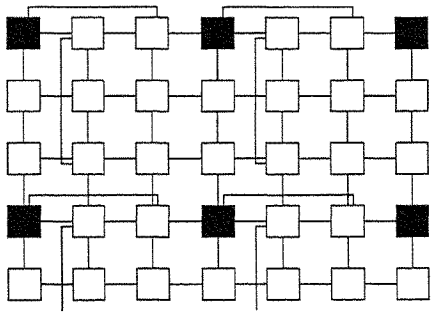
Our technique of overlapping semigroup computations for multiple rows can be applied to [16] to give an $O(N^{1/8})$ time algorithm for semigroup computations in which the values are distributed in row-major order on an $N^{5/8} \times N^{3/8}$ mesh with separable row/column buses. The algorithm in [16] requires that the values be distributed in submesh-row-major order.

7 Minimizing Node Degrees

There are two motivations for minimizing the number of links incident on each processor. First, as pointed out by Dally [7] and Agarwal [1], the channel width[6] is often limited by a node's pin count rather than by the wire bisection.[7] Thus reducing the node degree can increase channel width. Second, we may want to implement meshes with express links by using off-the-shelf homogeneous nodes of low degrees, such as TI C40 [17] or Inmos T9000 transputer [8]. Each C40 has 6 links and each T9000 has 4 links. We note that our solutions in the previous section require a configuration in which the terminals (except the ones along the edges) are of degree 8. To implement our solutions using say degree-5 nodes, we could combine multiple nodes to form a compound node. Figure 4(a) shows a compound node of degree 8 which is made from two degree-5 nodes. Such a node can serve as a terminal in our solutions. Alternatively, we could shorten or shift the express links a little so that they do not concentrate on the same nodes. Figure 4(b) shows such a variant which has a maximum node degree of 5. The algorithms in Section 6 can be easily adapted to this architecture with the same asymptotic time complexities.



(a) a compound node of degree 8



(b)

Figure 4: Using degree-5 nodes only.

[6]The channel width is the number of wires per link.

[7]The wire bisection is the minimum number of wires that, when cut, separate the network into two equal halves.

8 Hierarchies of Express Links

In the mesh with express links model, express links can be viewed as imposing a second level of mesh links on top of a first level of mesh links. When given a very large N, we can generalize the idea to yield a hierarchy of express links.

In the two-dimensional case, we have a two-dimensional mesh of submeshes with express links. The strategy is to compute the partial answer for each submesh in parallel, and then to combine the partial results of the submeshes to obtain the answer for the whole mesh. By choosing the submesh size of $n^{2/3} \times n^{2/3}$ and having express links of length $n^{1/3}$, we can perform semigroup computations in $O(n^{1/3})$ steps for each submesh. There are $n^{1/3} \times n^{1/3}$ submeshes totally. We can combine all the partial results of the submeshes using inter-submesh express links which have length $n^{2/3}$. This phase takes $O(n^{1/3})$ steps. Thus the total time is $O(n^{1/3})$ steps, or $O(N^{1/6})$ steps.

Each submesh can be further partitioned in sub-submeshes. The total time for an r-level scheme is $O(n^{1/r+1})$. For example, the total time is equal to $O(n^{1/4})$ or $O(N^{1/8})$ when $r = 3$.

9 Other Problems

Here, we state the time bounds for several other problems—prefix computations, the median row problem, median finding, and the all points closest neighbor problem—on an $N^{1/2} \times N^{1/2}$ mesh with express links (each of length $N^{1/4}$). Because the algorithms for these problems do not reveal techniques very different from those in [12] [16] and those we have presented above, we omit their details. The results are summarized in Table 2.

References

[1] A. Agarwal, "Limits on Interconnection Network Performance". *IEEE Transactions on Parallel and Distributed Systems*, Vol. 2, No. 4, October 1991, pp. 398–412.

[2] A. Aggarwal, "Optimal Bounds for Finding Maximum on Array of Processors with k Global Buses". *IEEE Transactions on Computers*, Vol. C-35, No. 1, January 1986, pp. 62–64.

[3] A. Bar-Noy and D. Peleg, "Square Meshes are Not Always Optimal". *IEEE Transactions on Computers*, Vol. 40, No. 2, February 1991, pp. 196–203.

Problem	Description	Time
Prefix computations	A generalization of semigroup computations. Find out $a_0 \oplus a_1 \oplus ... \oplus a_i$, for all i in the range $0 \leq i \leq (N-1)$.	$\Theta(N^{1/4})$
Median row	Each processor initially has a bit. Find the row where about half the 1's are above it and about half are below it.	$\Theta(N^{1/4})$
Median finding	Find the median of N numbers.	$O(N^{1/4} \log N)$
All points closest neighbor	Each processor initially has a bit. For each processor that has a 1, find the closest processor containing a 1.	$\Theta(N^{1/4})$

Table 2: Summary of results for other problems.

[4] Y. Ben-Asher, D. Peleg, R. Ramaswami, and A. Schuster, "The Power of Reconfiguration". *Journal of Parallel and Distributed Computing*, Vol. 13, No. 2, October 1991, pp. 139–153.

[5] S.H. Bokhari, "Finding Maximum on an Array Processor with a Global Bus". *IEEE Transactions on Computers*, Vol. C-33, No. 2, February 1984, pp. 133–139.

[6] Y.C. Chen, W.T. Chen, G.H. Chen, and J.P. Sheu, "Designing Efficient Parallel Algorithms on Mesh-Connected Computers with Multiple Broadcasting". *IEEE Transactions on Parallel and Distributed Systems*, Vol. 1, No. 2, April 1990, pp. 241–245.

[7] W.J. Dally, "Express Cubes: Improving the Performance of k-ary n-cube Interconnection Networks". *IEEE Transactions on Computers*, Vol. 40, No. 9, September 1991, pp. 1016–1023.

[8] M.D. May, P.W. Thompson, and P.H. Welch (eds.), *Networks, Routers and Transputers: Function, Performance, and Application*, IOS Press, 1993.

[9] V. Kumar, A. Grama, A. Gupta, and G. Karypis, *Introduction to Parallel Algorithms: Design and Analysis of Algorithms*, Benjamin/Cummings, 1994.

[10] J.Y.T. Leung and S.M. Shende, "On Multi-Dimensional Packet Routing for Meshes with Buses". *Journal of Parallel and Distributed Computing*, Vol. 20, No. 2, February 1994, pp. 187–197.

[11] Y.W. Lu, J.B. Burr, and A.M. Peterson. "Permutation on the Mesh with Reconfigurable Bus: Algorithms and Practical Considerations". *Proceedings of the* 7^{th} *International Parallel Processing Symposium*, April 1993. pp. 298–308.

[12] V.K. Prasanna Kumar and C.S. Raghavendra, "Array Processor with Multiple Broadcasting". *Journal of Parallel and Distributed Computing*, Vol. 4, No. 2, April 1987, pp. 173–190.

[13] T. Maeba, S. Tatsumi, and M. Sugaya, "Algorithms for Finding Maximum and Selecting Median on a Processor Array with Separable Global Buses". *Electronics and Communications in Japan*, Part 3. Vol. 73, No. 6, June 1990, pp. 39–48.

[14] R. Miller and Q.F. Stout, "Varying Diameter and Problem Size in Mesh Connected Computers". *Proceedings of the 1985 International Conference on Parallel Processing*, June 1985, pp. 697–699.

[15] C.S. Raghavendra, "HMESH: A VLSI Architecture for Parallel Processing". *Proceedings of the* 2^{nd} *Conference on Algorithms and Hardware for Parallel Processing (CONPAR 86)*, September 1986, pp. 76–83.

[16] M.J. Serrano and B. Parhami, "Optimal Architectures and Algorithms for Mesh-Connected Parallel Computers with Separable Row/Column Buses". *IEEE Transactions on Parallel and Distributed Systems*, Vol. 4, No. 10, October 1993, pp. 1073–1079.

[17] R. Simar, Jr. *et al.*, "Floating-Point Processors Join Forces in Parallel Processing Architectures". *IEEE Micro*, Vol. 12, No. 4, August 1992, pp. 60–69.

[18] Q.F. Stout, "Mesh-Connected Computers with Broadcasting". *IEEE Transactions on Computers*, Vol. C-32, No. 9, September 1983, pp. 826–830.