International Journal of Networking and Computing – www.ijnc.org
ISSN 2185-2839 (print) ISSN 2185-2847 (online)
Volume 1, Number 1, pages 36–48, January 2011

Impact of Reconfigurable Function on Meshes with Row/Column Buses

Susumu Matsumae
Department of Information Science, Saga University
Saga, 840-8502, Japan

Received: July 1, 2010
Revised: November 4, 2010
Accepted: December 9, 2010
Communicated by Akihiro Fijiwara

Abstract

This paper studies the difference in computational power between the mesh-connected parallel computers equipped with dynamically reconfigurable bus systems and those with static ones. The mesh with separable buses (MSB) is the mesh-connected computer with dynamically reconfigurable row/column buses. The broadcasting buses of the MSB can be dynamically sectioned into smaller bus segments by program control. We examine the impact of reconfigurable capability on the computational power of the MSB model, and investigate how computing power of the MSB decreases when we deprive the MSB of its reconfigurability. We show that any single step of the MSB of size $n \times n$ can be simulated in $O(\log n)$ time by the MSB without its reconfigurable function, which means that the MSB of size $n \times n$ can work with $O(\log n)$ step slowdown even if its dynamic reconfigurable function is disabled.

Keywords: Dynamically reconfigurable bus, Statically partitioned bus, Processor array, Polylogarithmic time simulation

1 Introduction

The mesh-connected parallel computer is a processor array that consists of processors arranged to a 2-dimensional grid. Each processor is connected via bi-directional unit-time communication links to its adjacent processors. Its natural structure is suitable for VLSI implementation and allows a high degree of integration. However, the mesh architecture has a crucial drawback that its communication diameter is large due to lack of broadcasting mechanism. To overcome this problem, many researchers have considered adding global buses (broadcasting buses) to the mesh. By using global buses, data broadcasting can be carried out: a processor can send data to a global bus, and those processors along the bus can receive it from the bus. Basically, the global buses are static, i.e., their connection topologies are statically fixed and can not be dynamically changed during the execution of programs.

Recently, the more powerful bus model called *reconfigurable bus system* have been intensively studied due to their strong computational powers [7, 14, 17, 18, 20]. The dynamic bus system can be used to dynamically obtain various interconnection patterns among the processors during the execution of programs. One typical technique to implement such a reconfigurable function of the bus system is to fuse/segment bus fragments dynamically [18]. The processors in the same interconnected bus fragment can be seen as the ones which own a unit-time broadcasting bus over the processors.

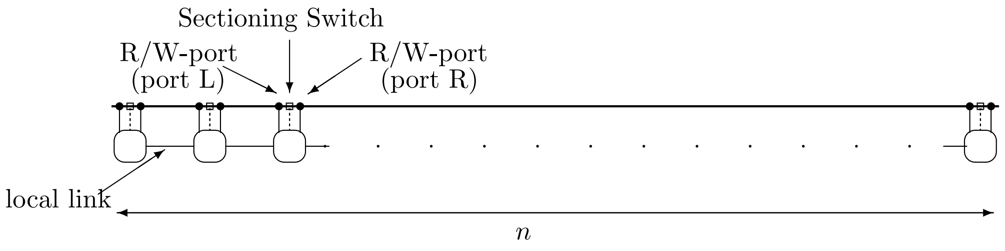


Figure 1: A separable bus along a row of the $n \times n$ MSB. Broadcasts are carried out in the following way: 1) several processors section the global bus by locally-controlled sectioning switches, 2) several processors send data to the bus through port L and/or R, and 3) several processors receive data from the bus through port L and/or R.

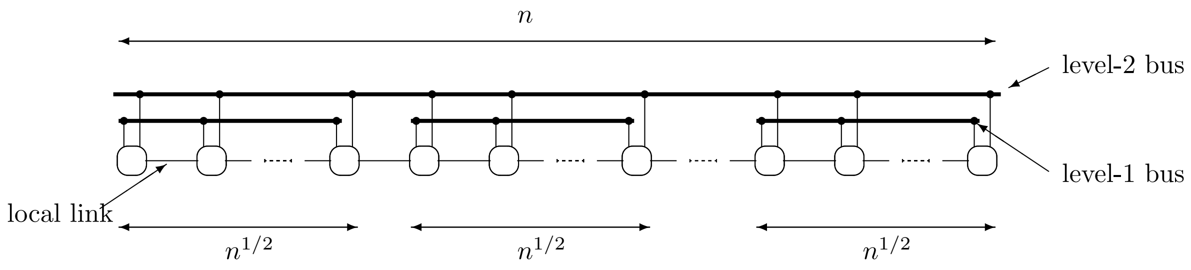


Figure 2: Partitioned buses along a row of the $n \times n$ MMPB. Here, $L = 2$, $\ell_1 = n^{1/2}$, and $\ell_2 = n$.

Dynamic reconfigurable function enables the models to make efficient use of broadcast buses, and to solve many important, fundamental problems (arithmetic, sorting, graph algorithms, etc.) efficiently, mostly in a constant or polylogarithmic time [14, 18]. Such reconfigurability, however, makes the bus systems complex and causes negative effects on the communication latency of global buses [5]. Hence, it is practically important to study the trade-off between such points quantitatively.

In this paper, we investigate the impact of reconfigurable capability on the computational power of mesh-connected computers with global buses. Here, we deal with the *meshes with separable buses* (MSB) [7, 17] and a variant of the meshes with partitioned buses called the *meshes with multiple partitioned buses* (MMPB) [8]. The MSB and the MMPB are the mesh-connected computers enhanced by the addition of broadcasting buses along every row and column. The broadcasting buses of the MSB, called *separable buses*, can be dynamically sectioned into smaller bus segments by program control, while those of the MMPB, called *partitioned buses*, are statically partitioned in advance and cannot be dynamically reconfigurable. In the MSB model, each row/column has only one separable bus, while in the MMPB model, each row/column has L partitioned buses ($L \geq 1$). See Figure 1 and 2. By comparing the relative power between these models, we have studied the difference in computational power between the parallel models equipped with reconfigurable bus systems and those with static ones.

In this paper, we study how much slowdown is required when we deprive the MSB of its reconfigurable function. Here, we show that the MSB of size $n \times n$ can work with $O(\log n)$ step slowdown without its reconfigurable function. Since we have shown that the MSB of size $n \times n$ can simulate the *reconfigurable mesh* [1, 14, 20] (or PARBS, the processor array with reconfigurable bus systems) of size $n \times n$ in $O(\log^2 n)$ steps [13], we can state that the reconfigurable mesh of size $n \times n$ can also work with $O\left(\log^3 n\right)$ step slowdown even if its reconfigurable function is unused.

This paper is organized as follows. Section 2 explains computational models, problem definition, and related works. Section 3 shows that the MSB of size $n \times n$ can work with $O(\log n)$ step slowdown even if its dynamic reconfigurable function is disabled. And finally, Section 4 offers concluding remarks.

2 Preliminaries

2.1 Models

A mesh-connected computer (mesh) is a parallel computational model in which $n \times n$ identical processors (PEs) are arranged in a 2-dimensional grid with n rows and n columns [4, 19]. See Figure 3. The PE located in row i and column j is denoted as PE$[i, j]$ $(1 \leq i, j, \leq n)$. Each PE is connected to its 4 adjacent PEs (if provided) via unit-time bi-directional communication links. The mesh structure is very natural and is convenient to be implemented in 2-D layout. However, the conventional mesh has a large communication diameter, and usually needs $\Omega(n)$ time for solving problems (e.g., prefix-sum computations of $n \times n$ values distributed one data per PE) [4, 19].

The *mesh with broadcasting buses* (MB) is a mesh enhanced by the addition of broadcasting buses along every row and column [16]. See Figure 4. Because of the broadcasting capability, the MB can solve the prefix-sum problem more efficiently in $O\left(n^{1/2}\right)$ time steps [16]. The *mesh with partitioned buses* (MPB) is a variant of the MB model, where each broadcasting bus is equally partitioned by a fixed length ℓ [2, 6]. The MPB can solve the prefix-sum problem in $O\left(n^{1/3}\right)$ steps when $\ell = \Theta(n^{2/3})$ [6]. The *mesh with multiple partitioned buses* ($\mathsf{MMPB}^{\langle L \rangle}$) is the mesh with multiple partitioned buses, where each row/column has L partitioned buses $(L \geq 1)$ [11]. See Figure 2. It should be noted that the MB and MPB models can be derived from the $\mathsf{MMPB}^{\langle L \rangle}$ with $L = 1$. Those L partitioned buses of the $\mathsf{MMPB}^{\langle L \rangle}$ are indexed as *level-1*, *level-2*, …, *level-L*, respectively. We assume that the partitioned buses of the $\mathsf{MMPB}^{\langle L \rangle}$ are equally partitioned by the same length if they belong to the same level. For each level-k, the value ℓ_k denotes the length of a bus segment of the level-k buses. The $\mathsf{MMPB}^{\langle L \rangle}$ can solve the prefix-sum problem in $O\left(Ln^{1/(2L+1)}\right)$ steps [11].

To obtain more powerful computational power, dynamically reconfigurable broadcasting buses have been studied [18]. One such model is the *mesh with separable buses* (MSB)[1], where each row/column broadcasting buses can be dynamically sectioned into smaller bus segments by locally controlled sectioning switches of PEs. Each PE has access to the global bus through local read/write ports at the sides of sectioning switch. See Figure 1. In the row separable bus shown in Figure 1, broadcasts are carried out in the following way: 1) several PEs section the global bus by locally-controlled sectioning switches, 2) several PEs send data to the bus through port L and/or R, and 3) several PEs receive data from the bus through port L and/or R. The MSB is such a powerful model that solves many problems in polylogarithmic time (mostly in $O(\log n)$ time, e.g., $O(\log n)$ steps for solving the prefix-sum problem) [7, 17, 18].

A single time step of the above-mentioned models is composed of the following three substeps:

1) Local communication substep:
Every PE communicates with its adjacent PEs via local links.

2) Broadcast substep:
Every PE changes its switch configurations by local decision (this operation is only for the MSB). Then, along each broadcasting bus segment, several of the PEs connected to the bus send data to the bus, and several of the PEs on the bus receive the data transmitted on the bus.

3) Compute substep:
Every PE executes some local computation.

[1]The MSB is essentially the same model as the *horizontal-vertical reconfigurable mesh (HV-RM)* described in [1, 18].

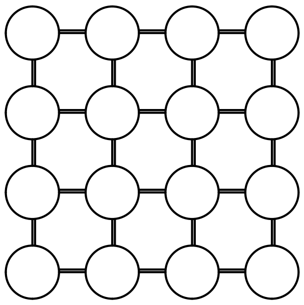


Figure 3: A conventional mesh of size 4×4. Each processor can communicate with its adjacent processors via bi-directional local communication links.

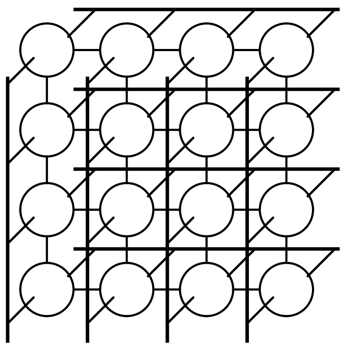


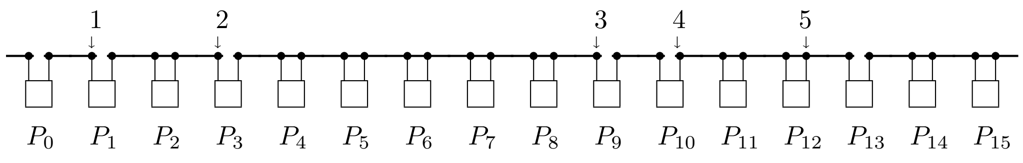
Figure 4: A mesh with broadcasting of size 4×4. In addition to the local communications, broadcasts along rows and columns can be executed by the broadcasting buses.

Here, we assume that a PE writes to only one bus at a time in the MMPB model. The bus accessing capability is of Common-Collision model. If there is a write-conflict on a bus, the PEs on the bus receive a special value $\perp$ (i.e., PEs can detect whether there is a write-conflict on a bus or not). If there is no data transmitted on a bus, the PEs on the bus receive a special value ϕ (i.e., PEs can know whether there is data transmitted on a bus or not).

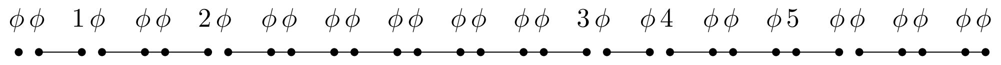
2.2 Simulation Problem

In this paper, we consider simulating any single step of the $n \times n$ MSB ($\mathcal{M}$) by using the mesh with statically partitioned buses ($\mathcal{M}'$). Here, we assume that $\mathcal{M}$' is the same size as $\mathcal{M}$. The processor mapping is a natural one: each PE$[i,j]$ of $\mathcal{M}$' simulates PE$[i,j]$ of the $\mathcal{M}$. Each PE of $\mathcal{M}$' is given the behaviour of its corresponding PE of $\mathcal{M}$. We assume that the computing power of PEs, the bandwidth of local links, and that of broadcasting buses are equivalent in both $\mathcal{M}$ and $\mathcal{M}$'.

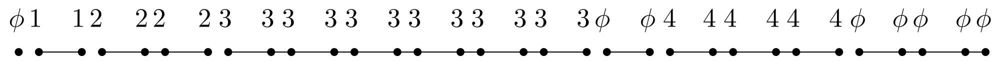
Since the only difference between $\mathcal{M}$ and $\mathcal{M}$' is the broadcasting capability, both local communication and compute substeps are easily simulated in a constant number of steps (each PE$[i,j]$ of $\mathcal{M}'$ simply executes the same operations as PE$[i,j]$ of $\mathcal{M}$). The broadcast substep of the $\mathcal{M}$ is simulated by solving connected-component labeling (CC-labeling) problem for a *port-connectivity graph* (pc-


1
2
3
4
5
P_0 P_1 P_2 P_3 P_4 P_5 P_6 P_7 P_8 P_9 P_{10} P_{11} P_{12} P_{13} P_{14} P_{15}

(a) Broadcasts to be simulated. The processors P_1,P_3, P_9, P_{10}, and P_{12} respectively send data


$\phi\phi$ 1ϕ $\phi\phi$ 2ϕ $\phi\phi$ $\phi\phi$ $\phi\phi$ $\phi\phi$ $\phi\phi$ 3ϕ $\phi 4$ $\phi\phi$ $\phi 5$ $\phi\phi$ $\phi\phi$ $\phi\phi$

(b) The corresponding port-connectivity graph G with initial labels


$\phi 1$ 1 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3ϕ $\phi 4$ 4 4 4 4 4ϕ $\phi\phi$ $\phi\phi$

(c) After connected-component labeling, vertices in each component C is labeled by the smallest initial label of all the vertices in C, with regarding ϕ as the greatest element

Figure 5: Broadcasts on a separable bus along a row of the $n \times n$ MSB are simulated by connected-component labeling of the port-connectivity graph. Here, $n = 16$

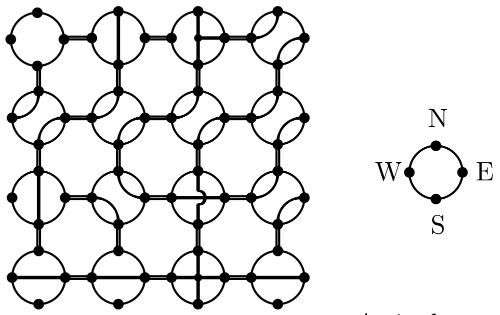
graph). See Figure 5 for an example. Vertices of the pc-graph correspond to the read/write-ports of PEs, and edges stand for the port-to-port connections. Each vertex is initially labeled by the value which is sent through the corresponding port by the PE at the broadcast substep. If there is no data sent through the port, the vertex is labeled by ϕ. The CC-labeling is done in such a way that vertices in each component C is labeled by the smallest initial label of all the vertices in C, with regarding ϕ as the greatest value. These labels are called *component labels*. Obviously, the simulation of the broadcast substep of $\mathcal{M}$ can be achieved in $O(T)$ steps on the $\mathcal{M}'$ if the CC-labeling of the corresponding port-connectivity graph can be executed in $O(T)$ steps by $\mathcal{M}'$.

2.3 Related Works

Dynamically reconfigurable capability is a very powerful function. For example, the most flexible model called *reconfigurable mesh* (RM, see Figure 6) [1, 14, 20] can solve many problems in a constant number of steps (e.g., sorting n elements in $O(1)$ time using the RM of size $n \times n$).

In this paper, we study how much slowdown is required when we deprive the MSB of its reconfigurable function. In [9, 11], we have shown that the MSB of size $n \times n$ can be simulated time-optimally in $O\left(n^{1/(2L+1)}\right)$ steps using the MMPB of size $n \times n$, where L is constant and the global buses are of *word-model*, i.e., the bus-width is the same as the number of bits in one word. From this result, it is natural to think that the slowdown may be at least of polynomial time. However, in [10], we successfully showed that we can suppress the slowdown to $O\left(\log^2 n\right)$ steps, by considering the relation between the word-model bus and the *bit-model* bus. In [10], we utilized the fact that each single word-model bus can be viewed as $\lceil\log n\rceil$ bit-model buses, where $\lceil\log n\rceil$ is the word-size of processor. Since we assumed that we can separately section each wire of a word-model bus with different lengths, the computational model in [10] is, strictly speaking, not the regular MMPB model. Later in [12], without putting any special assumption that we can separately section each wire of a word-model bus with different lengths, we showed that the MSB can work with $O\left(\log^3 n\right)$ step slowdown without its reconfigurability.

Here, we improve the result in [12], and show that the MSB of size $n \times n$ can work with $O(\log n)$ step slowdown without its reconfigurable function. Since we have shown that the MSB of size $n \times n$



A 4×4 RM

A single processor
and its 4 ports

Figure 6: A 4×4 RM and a single processor. Each processor can dynamically change the local connection pattern of its 4 ports (N, S, E, and W) during the execution of programs.

can simulate the RM of size $n \times n$ in $O\left(\log^2 n\right)$ steps [13], we can state that the reconfigurable mesh of size $n \times n$ can also work with $O\left(\log^3 n\right)$ step slowdown even if its reconfigurable function is unused.

3 Simulation of Dynamically Reconfigurable Buses by Statically Partitioned Buses

The difference between the MB, MPB, $\mathsf{MMPB}^{\langle L\rangle}$, and MSB models is only their broadcasting capabilities. As mentioned in Section 2.2, to simulate operations of the MSB by using other models, we focus on how to mimic the broadcast substep of the MSB, because the local communication and the compute substeps of the MSB can be easily simulated in a constant number of steps (each simulating PE simply executes the same operations as the simulated PE). Also, in what follows, we explain how to simulate the broadcasts along rows only, since those along columns can be simulated similarly.

To begin with, we introduce the following lemma:

Lemma 1 *[10] Any step of the* MSB *of size $n \times n$ can be simulated in $O\left(Ln^{1/(2L+1)}\right)$ steps by the* $\mathsf{MMPB}^{\langle L\rangle}$ *of size $n \times n$.* ■

In the algorithm proving this lemma, we let each $\ell_j = \Theta(n^{\alpha_j})$ where $\alpha_j = 2j/(2L+1)$. If we let $L = \log n$, then $O\left(Ln^{1/(2L+1)}\right)$ becomes $O(\log n)$. Hence, we have the following corollary.

Corollary 1 *Any step of the* MSB *of size $n \times n$ can be simulated in $O(\log n)$ steps by the* $\mathsf{MMPB}^{\langle \log n\rangle}$ *of size $n \times n$.* ■

Here, please note that $\ell_{j+1}/\ell_j = n^{2/(2L+1)}$ becomes a constant when $L = \log n$. The broadcast along each row (resp. column) of the MSB is simulated locally by the corresponding row (resp. column) of the $\mathsf{MMPB}^{\langle \log n\rangle}$. We briefly explain the algorithm proving Corollary 1 as follows. In [10], the algorithm is described as a recursive algorithm, but here we write it in non-recursive way. For simplify the exposition, we assume that $n \bmod \log n = 0$ and that $\log n$ is a positive integer. Also, we let $\ell_j = 2^j$. In the algorithm proving Corollary 1, the broadcasts along each row (resp. column) of the MSB is simulated locally by the corresponding row (resp. column) of the $\mathsf{MMPB}^{\langle \log n\rangle}$. The algorithm first simulates broadcasts along rows, and then those along columns. Since the simulation of the column broadcasts is essentially the same as that of the row broadcasting, we explain how to simulate a row of the MSB by using the corresponding row of the $\mathsf{MMPB}^{\langle \log n \rangle}$ only. Let G be the pc-graph correspond to the broadcast operation taken along a row of the simulated MSB. Then, the following algorithm solves the CC-labeling problem for G by using the corresponding row of the simulating $\mathsf{MMPB}^{\langle \log n \rangle}$.

```
Algorithm Alg_Cor1_on_MMPB$^{\langle \log n \rangle}$

Stage 1: { Label combination in Bottom-Up fashion }
    for $d = 1$ to $\log n$ do
        For each subgraph $G_k$ of $G$, compute local component labels within $G_k$
        for the leftmost and rightmost vertices of $G_k$, and check whether the two
        vertices are connected to each other or not. (Here, we divide the pc-graph
        $G$ into $n/\ell_d$ disjoint subgraphs $G_1, G_2, \ldots, G_{n/\ell_d}$ of width $\ell_d$.[2])

Stage 2: { Label propagation in Top-Down fashion }
    for $d = \log n$ downto 1 do
        For each subgraph $G_k$ of $G$, compute component labels within $G_k$ for the
        leftmost and rightmost vertices of $G_k$. (Here, $G_k$ is defined in the same
        fashion as in Stage 1.)

end of Alg_Cor1_on_MMPB$^{\langle \log n \rangle}$
```

The algorithm adopts the same divide-and-conquer strategy used in the component labeling algorithm for a binary image proposed by Maresca et al. in [3, 15]. In the for-loop body of Stage 1 and Stage 2 of Alg_Cor1_on_$\mathsf{MMPB}^{\langle \log n \rangle}$, the simulating $\mathsf{MMPB}^{\langle \log n \rangle}$ uses only the level-d bus and local links for data communication. The details are in [10].

Figure 7 illustrates the cost for simulating a row of the MSB by using the corresponding row of the $\mathsf{MMPB}^{\langle \log n \rangle}$. The broadcasting buses of the simulating $\mathsf{MMPB}^{\langle \log n \rangle}$ are partitioned by length $2, 4, 8, \ldots, n$.

Next, we consider a variant of the MPB ($\mathsf{MMPB}^{\langle 1 \rangle}$), called *mesh with partitioned buses of different lengths* (hereafter $\mathsf{MPB}_{\mathrm{dif}}$). In the original MPB model, row/column buses are partitioned equally with the same length if they belong to the same level, while in the $\mathsf{MPB}_{\mathrm{dif}}$ model, each row/column bus can be partitioned with different lengths. In the $\mathsf{MPB}_{\mathrm{dif}}$ model, each row/column has only one statically partitioned bus. Please note that the $\mathsf{MPB}_{\mathrm{dif}}$ model can be viewed as the MSB without its reconfigurable function. In the following, we show that the $\mathsf{MPB}_{\mathrm{dif}}$ can simulate the MSB efficiently in $O(\log n)$ time.

By using L consecutive rows of the $\mathsf{MPB}_{\mathrm{dif}}$, the L partitioned buses attached to a single row of the $\mathsf{MMPB}^{\langle L \rangle}$ can be simulated in $O(L)$ time. The idea is as follows: Each one of the L partitioned buses along a row of the $\mathsf{MMPB}^{\langle L \rangle}$ is simulated by a distinct row of the simulating $\mathsf{MPB}_{\mathrm{dif}}$. See Figure 8. The L partitioned buses attached to the row of the $\mathsf{MMPB}^{\langle L \rangle}$ are simulated by row $1, 2, \ldots, L$ of the $\mathsf{MPB}_{\mathrm{dif}}$. Each broadcasting bus along row i of the $\mathsf{MPB}_{\mathrm{dif}}$ is partitioned by length ℓ_i so that the level-i bus of the $\mathsf{MMPB}^{\langle L \rangle}$ can be simulated in a constant time by the row i of the $\mathsf{MPB}_{\mathrm{dif}}$. Here, the behavior of each PE in the simulated row of the $\mathsf{MMPB}^{\langle L \rangle}$ is initially given to the corresponding PE in the topmost row (row 1) of the L rows of the $\mathsf{MPB}_{\mathrm{dif}}$. The simulation consists of two stages. In Stage 1, data is moved downward, row by row, and the broadcasting on the level-i bus of the $\mathsf{MMPB}^{\langle L \rangle}$ is simulated when the data is at row i. In Stage 2, the data (simulation result) is moved upward to the topmost row. See Figure 9. It is not difficult to confirm that this simulation requires $O(L)$ steps, which is mainly from the cost for the data-transfer vertically along each column of the L rows over the $\mathsf{MPB}_{\mathrm{dif}}$ (here, the data-transfer uses only local links if it is vertical one). Formally, we describe the algorithm as follows:

[2]We say that a subgraph of pc-graph is of width w if it contains $2w$ vertices corresponding to the read/write-ports of w consecutive PEs.

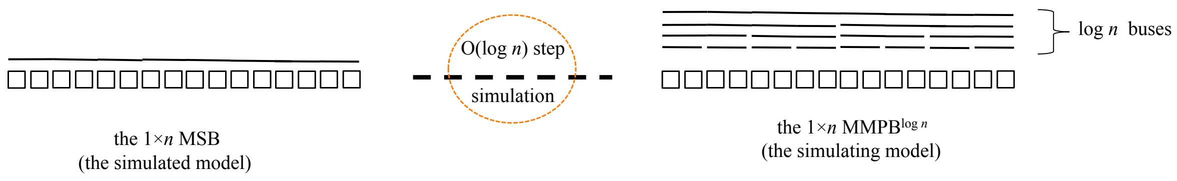


Figure 7: A row of the MSB of size $n \times n$ is simulated by the corresponding row of $\mathsf{MMPB}^{\langle \log n \rangle}$. Here, $n = 16$.

```
Algorithm ALG_LEM2_ON_MPB$_{dif}$

Stage 1: { Simulation and Vertical downward data shifting }
    for $d = 1$ to $L$ do
        (1-1) The row $d$ of the $\mathsf{MPB}_{\rm dif}$ simulates the broadcasts taken on level-$d$
              bus of the $\mathsf{MMPB}^{\langle L \rangle}$.
        (1-2) Each PE in the row $d$ of the $\mathsf{MPB}_{\rm dif}$ transfers the simulation results
              obtained so far and the behavior information of the simulated row of
              the $\mathsf{MMPB}^{\langle L \rangle}$ to the PE below in the row $(d+1)$ via vertical local link.
              $(d \neq L)$

Stage 2: { Vertical upward data shifting }
    for $d = L$ downto 1 do
        (2-1) Each PE in the row $d$ of the $\mathsf{MPB}_{\rm dif}$ transfers the complete simulation
              results to the PE above in the row $(d-1)$ via vertical local link. $(d \neq 1)$

end of ALG_LEM2_ON_MPB$_{dif}$
```

Obviously, (1-2) and (2-1) of ALG_LEM2_ON_$\mathsf{MPB}_{\rm dif}$ can be executed in $O(1)$ steps. As for (1-1), it takes only $O(1)$ steps as well, since each partitioned bus along row d of the $\mathsf{MPB}_{\rm dif}$ is sectioned by every ℓ_d.

Hence, we can prove the following lemma.

Lemma 2 *Any step of a single row (resp. column) of the* $\mathsf{MMPB}^{\langle L \rangle}$ *of size* $n \times n$ *can be simulated in* $O(L)$ *steps by the* L *consecutive rows (resp. columns) of the* $\mathsf{MPB}_{\rm dif}$ *of size* $n \times n$. ■

Figure 10 illustrates the cost for simulating a row of the $\mathsf{MMPB}^{\langle L \rangle}$ by using consecutive L rows of the $\mathsf{MPB}_{\rm dif}$.

Next, we consider how to simulate L consecutive rows, not a single row, of the $\mathsf{MMPB}^{\langle L \rangle}$ using the L rows of the $\mathsf{MPB}_{\rm dif}$. Here, we execute ALG_LEM2_ON_$\mathsf{MPB}_{\rm dif}$ in a pipeline fashion on the $\mathsf{MPB}_{\rm dif}$. For example, consider the case where the L consecutive rows of the $\mathsf{MMPB}^{\langle L \rangle}$ are simulated by the corresponding L rows of the $\mathsf{MPB}_{\rm dif}$. The simulation of row 1 of the $\mathsf{MMPB}^{\langle L \rangle}$ can start immediately as in Figure 9, while that of row i, $i \neq 1$, can start only when the data-movement from row i to the topmost row via local-links is completed. Formally, we can describe the algorithm as follows:

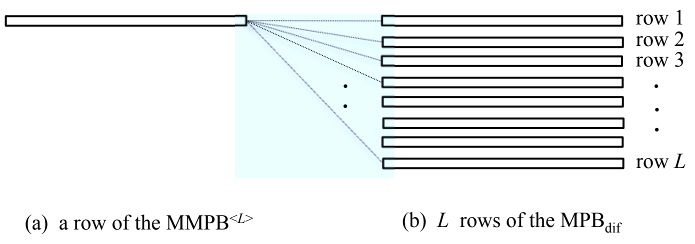

row 1
row 2
row 3
row L
(a) a row of the MMPB$^{\langle L\rangle}$
(b) L rows of the MPB$_{\rm dif}$

Figure 8: A row of the $\mathsf{MMPB}^{\langle L\rangle}$ is simulated by the L consecutive rows of the $\mathsf{MPB}_{\rm dif}$. Each broadcasting bus along row i of the $\mathsf{MPB}_{\rm dif}$ is partitioned by length ℓ_i so that the level-i bus of the $\mathsf{MMPB}^{\langle L\rangle}$ can be simulated in a constant time by the row i of the $\mathsf{MPB}_{\rm dif}$.

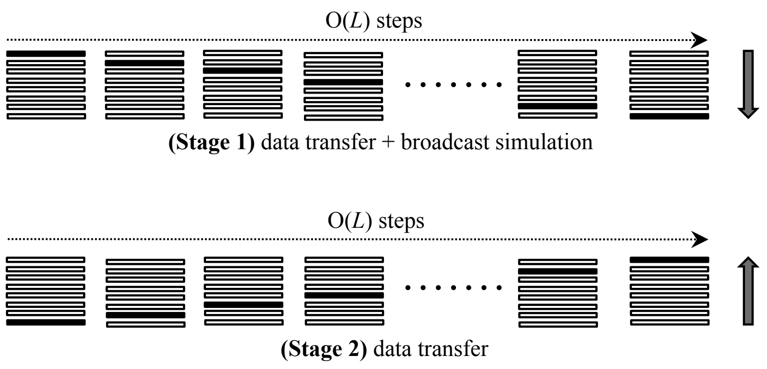


Figure 9: The active region is shifted, row by row, during the simulation of a row of the $\mathsf{MMPB}^{\langle L\rangle}$ by the L consecutive rows of the $\mathsf{MPB}_{\rm dif}$.

```
Algorithm ALG_LEM3_ON_MPB$_{\rm dif}$

for $e = 1$ to $4L$ do
    (1) { Transfer of initial data upward to the topmost row }
        PE$[i, j]$ of the MPB$_{\rm dif}$ transfer the behavior information of PE$[i+e-1, j]$
        of the MMPB$^{\langle L\rangle}$ to PE$[i-1, j]$ of the MPB$_{\rm dif}$ via vertical local link.
        $(2 \le i \le L-e+1,$ and $1 \le j \le n)$
    (2) { Stage 1 of ALG_LEM2_ON_MPB$_{\rm dif}$}
        Each row $i$ of the MPB$_{\rm dif}$ executes the for-loop body of Stage 1 of ALG_LEM2_ON_MPB$_{\rm dif}$
        with $d = e$ for the simulation of row $(e-i+1)$ of the MMPB$^{\langle L\rangle}$.
        $(1 \le i \le L,\ e-L+1 \le i \le e,$ and $1 \le j \le n)$
    (3) { Stage 2 of ALG_LEM2_ON_MPB$_{\rm dif}$}
        Each row $i$ of the MPB$_{\rm dif}$ executes the for-loop body of Stage 2 of ALG_LEM2_ON_MPB$_{\rm dif}$
        with $d = e$ for the simulation of row $(e+i-2L)$ of the MMPB$^{\langle L\rangle}$.
        $(1 \le i \le L,\ 2L-e+1 \le i \le 3L-e,$ and $1 \le j \le n)$
    (4) { Transfer of results downward to the original position }
        PE$[i, j]$ of the MPB$_{\rm dif}$ transfer the simulation results of PE$[e-i-2L+1, j]$
        of the MMPB$^{\langle L\rangle}$ to PE$[i+1, j]$ of the MPB$_{\rm dif}$ via vertical local link.
        $(1 \le i \le L,\ e-3L+1 \le i \le e-2L,\ i < e-i-2L+1,$ and $1 \le j \le n)$
```

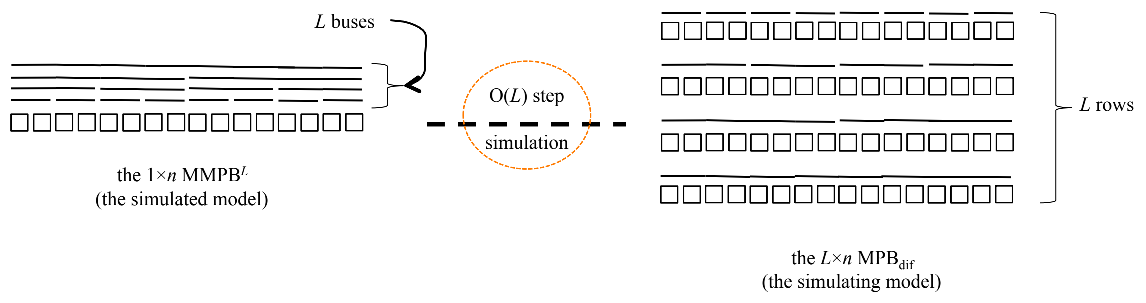


Figure 10: A row of the $\mathsf{MMPB}^{\langle L\rangle}$ of size $n \times n$ is simulated by the consecutive L rows of the $\mathsf{MPB}_{\mathrm{dif}}$ of size $n \times n$. Here, $n = 16$ and $L = \log n$.

end of ALG_LEM3_ON_$\mathsf{MPB}_{\mathrm{dif}}$

Thus, we can prove that the L consecutive rows of the $\mathsf{MMPB}^{\langle L\rangle}$ can be simulated in $O(L)$ steps by the corresponding L consecutive rows of the $\mathsf{MPB}_{\mathrm{dif}}$. Since the mesh of size $n \times n$ is composed of n/L submeshes of size $L \times n$, we obtain the following lemma.

Lemma 3 *Any step of the* $\mathsf{MMPB}^{\langle L\rangle}$ *of size $n \times n$ can be simulated in $O(L)$ steps by the* $\mathsf{MPB}_{\mathrm{dif}}$ *of size $n \times n$.* ■

Figure 11 illustrates the cost for simulating consecutive L rows of the $\mathsf{MMPB}^{\langle L\rangle}$ by using the corresponding rows of the $\mathsf{MPB}_{\mathrm{dif}}$.

From Corollary 1 and Lemma 3, we can say that any step of the MSB of size $n \times n$ can be simulated in $O(\log^2 n)$ steps by the $\mathsf{MPB}_{\mathrm{dif}}$ of size $n \times n$. In what follows, we show that the time-cost can be further improved to $O(\log n)$ steps.

As described earlier, ALG_COR1_ON_$\mathsf{MMPB}^{\langle \log n\rangle}$ consists of Stage 1 and Stage 2. Both Stage 1 and Stage 2 have $\log n$ iterations, and in each iteration the simulating $\mathsf{MMPB}^{\langle \log n\rangle}$ uses only level-d bus of the $\mathsf{MMPB}^{\langle \log n\rangle}$. That is, as ALG_COR1_ON_$\mathsf{MMPB}^{\langle \log n\rangle}$ proceeds, the level number of involved buses of the $\mathsf{MMPB}^{\langle \log n\rangle}$ monotonically increases and then decreases. Here, please note that the pattern of increase and decrease of the level number of involved buses of ALG_COR1_ON_$\mathsf{MMPB}^{\langle \log n\rangle}$ is the same as that of the row number of active region in ALG_LEM2_ON_$\mathsf{MPB}_{\mathrm{dif}}$ (Figure 9). Hence, by the similar argument used for proving Lemma 2, a single execution of ALG_COR1_ON_$\mathsf{MMPB}^{\langle \log n\rangle}$ (solving CC-labeling problem for a pc-graph of a single row) can be performed in $O(\log n)$ steps by the consecutive $\log n$ rows of the $\mathsf{MPB}_{\mathrm{dif}}$ of size $n \times n$. Then, again, by the similar argument used for proving Lemma 3, $\log n$ parallel executions of ALG_COR1_ON_$\mathsf{MMPB}^{\langle \log n\rangle}$ can be done in $O(\log n)$ steps in a pipeline fashion by the consecutive $\log n$ rows of the $\mathsf{MPB}_{\mathrm{dif}}$. Hence, we obtain the following lemma:

Lemma 4 *The* MMPB *algorithm proving Corollary 1 can be executed in $O(\log n)$ steps by the* $\mathsf{MPB}_{\mathrm{dif}}$ *of size $n \times n$.* ■

Figure 12 illustrates the cost for simulating consecutive $\log n$ rows of the MSB by using the corresponding rows of the $\mathsf{MPB}_{\mathrm{dif}}$.

Now, from Lemma 4, we can state the main theorem of the paper as follows:

Theorem 1 *Any single step of the* MSB *of size $n \times n$ can be simulated in $O(\log n)$ steps by the* $\mathsf{MPB}_{\mathrm{dif}}$ *of size $n \times n$.* ■

Since the $\mathsf{MPB}_{\mathrm{dif}}$ can be obtained from the MSB without using its dynamically reconfigurable function, we can say that the MSB of size $n \times n$ can work with $O(\log n)$ step slowdown even if its reconfigurable capability is disabled.

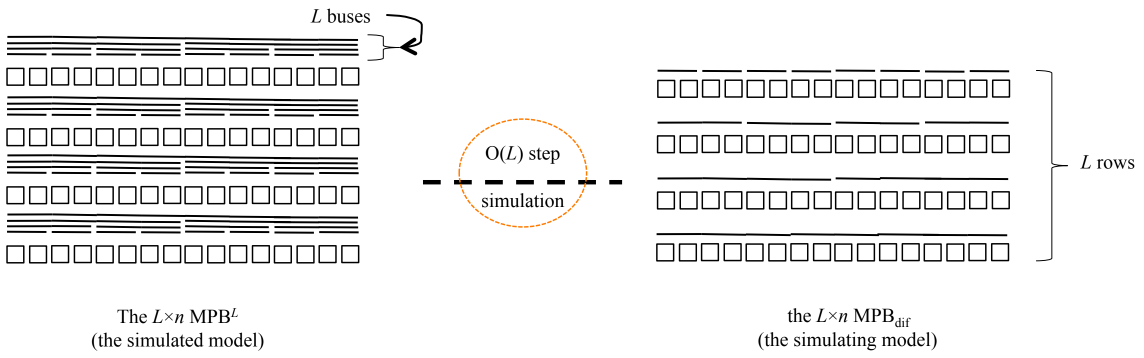


Figure 11: The L consecutive rows of the MMPB$^{\langle L \rangle}$ of size $n \times n$ is simulated by the corresponding rows of the MPB$_{\text{dif}}$ of size $n \times n$. Here, $n = 16$ and $L = \log n$.

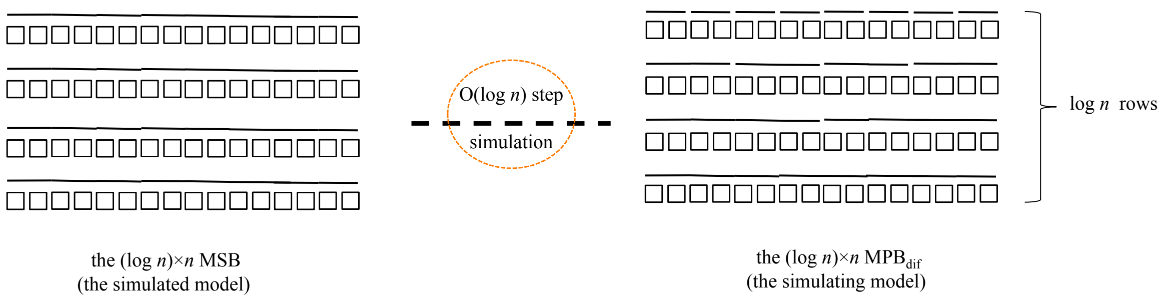


Figure 12: The $\log n$ consecutive rows of the MSB of size $n \times n$ is simulated by the corresponding rows of the MPB$_{\text{dif}}$ of size $n \times n$. Here, $n = 16$.

4 Concluding Remarks

In this paper, we showed that the MSB of size $n \times n$ can work with $O(\log n)$ step slowdown even if its reconfigurable capability is unused. As a corollary, since we have shown that the MSB of size $n \times n$ can simulate the RM of size $n \times n$ in $O\left(\log^2 n\right)$ steps [13], we can state that the RM of size $n \times n$ can also work with $O\left(\log^3 n\right)$ step slowdown even if its reconfigurable function is unused.

Although the MPB$_{\text{dif}}$ model has only statically partitioned broadcasting buses, it can solve problems mostly in polylogarithmic time, for it can simulate the MSB in $O(\log n)$ steps. In addition, compared to other major models such as the hyper-cube, mesh of trees, and CCC (Cube Connected Cycles), the MPB$_{\text{dif}}$ is simple and is suitable for 2D layout. Furthermore, the MSB model can be viewed as a virtual programming platform for the MPB$_{\text{dif}}$ with only $O(\log n)$ step overhead. Hence, we think that the MPB$_{\text{dif}}$ may be one of the realistic mesh-based parallel computational models.

Finally, it should be mentioned that our simulation algorithm can simulate the MSB in which the concurrent write is resolved by the MIN rule [13] where the minimum among the sent values is received when a write-conflict occurs. This is because our algorithm simulates the broadcast operation of the MSB by connected-component labelling of the corresponding *port-connectivity graph* [13]. This fact may make up the slowdown required for the simulation because the MSB with MIN-bus model is very powerful, for example, it can find the minimum among the values distributed over the mesh in a constant time.

Acknowledgements

This work was partly supported by the MEXT Grant-in-Aid for Young Scientists (B) (20700014).

References

[1] Y. Ben-Asher, D. Gordon, and A. Schuster. Efficient self-simulation algorithms for reconfigurable arrays. *J. of Parallel and Distributed Computing*, 30(1):1–22, October 1995.

[2] K. L. Chung. Prefix computations on a generalized mesh-connected computer with multiple buses. *IEEE Trans. Parallel and Distributed Systems*, 6(2):196–199, February 1995.

[3] H.Li and Q.F.Stout, editor. *Reconfigurable massively parallel computers*, pages 50–52. Prentice-Hall, Inc., 1991.

[4] F. T. Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes.* Morgan Kaufmann Publishers, San Mateo, 1992.

[5] T. Maeba, M. Sugaya, S. Tatsumi, and K. Abe. An influence of propagation delays on the computing performance in a processor array with separable buses. *IEICE Trans. A*, J78-A(4):523–526, 1995.

[6] T. Maeba, M. Sugaya, S. Tatsumi, and K. Abe. Semigroup computations on a processor array with partitioned buses. *IEICE Trans. A*, J80-A(2):410–413, 1997.

[7] T. Maeba, S. Tatsumi, and M. Sugaya. Algorithms for finding maximum and selecting median on a processor array with separable global buses. *IEICE Trans. A*, J72-A(6):950–958, 1989.

[8] S. Matsumae. Simulation of meshes with separable buses by meshes with multiple partitioned buses. In *Proc. of the 17th International Parallel and Distributed Processing Symposium (IPDPS2003), IEEE CS press*, 2003.

[9] S. Matsumae. Optimal simulation of meshes with dynamically separable buses by meshes with statically partitioned buses. In *Proc. of the 2004 International Symposium on Parallel Architectures, Algorithms and Networks (I-SPAN'04) IEEE CS Press*, pages 475–481, 2004.

[10] S. Matsumae. Polylogarithmic gap between meshes with dynamically separable buses and meshes with statically partitioned buses. In *Proc. of the 19th International Conference on Parallel and Distributed Computing Systems (PDCS-2006)*, 2006.

[11] S. Matsumae. Tight bounds on the simulation of meshes with dynamically reconfigurable row/column buses by meshes with statically partitioned buses. *J. of Parallel and Distributed Computing*, 66(10):1338–1346, 2006.

[12] S. Matsumae. Impact of reconfigurability on meshes with separable row/column buses. In *Proc. of the 2008 International Conference on Parallel and Distributed Processing Techniques and Applications (PDCCS 2008)*, 2008.

[13] S. Matsumae and N. Tokura. Simulation algorithms among enhanced mesh models. *IEICE Transactions on Information and Systems*, E82–D(10):1324–1337, 1999.

[14] R. Miller, V. K. Prasanna-Kumar, D. Reisis, and Q. F. Stout. Meshes with reconfigurable buses. In *Proc. of the fifth MIT Conference on Advanced Research in VLSI*, pages 163–178, Boston, 1988.

[15] P. Baglietto, M. Maresca, and M. Migliardi. Image Labeling by Transitive Closure on the Polymorphic Processor Array. Technical report, DIST University of Genoa, March 1994. ftp://rigel.dist.unige.it/pub/docs/smimp_reports/smimp-03.ps.gz.

[16] V. K. Prasanna-Kumar and C. S. Raghavendra. Array processor with multiple broadcasting. *J. of Parallel Distributed Computing*, 4:173–190, 1987.

[17] M. J. Serrano and B. Parhami. Optimal architectures and algorithms for mesh-connected parallel computers with separable row/column buses. *IEEE Trans. Parallel and Distributed Systems*, 4(10):1073–1080, October 1993.

[18] R. Vaidyanathan and J. L. Trahan. *Dynamic Reconfiguration.* Kluwer Academic/Plenum Publishers, 2004.

[19] Vipin Kumar, Ananth Grama, Anshul Gupta, and George Karypis. *Introduction to Parallel Computing: Design and Analysis of Algorithms.* The Benjamin/Cummings Publishing Company, Inc., CA, 1994.

[20] B. Wang and G. Chen. Constant time algorithms for the transitive closure and some related graph problems on processor arrays with reconfigurable bus systems. *IEEE Trans. Parallel and Distributed Systems*, 1(4):500–507, October 1990.