Parallel prefix computation on extended multi-mesh network

Prasanta K. Jana [a,*], B. Damodara Naidu [a], Shailendra Kumar [b], Monish Arora [b], Bhabani P. Sinha [c]

[a] *Department of Computer Science and Engineering, Indian School of Mines, Dhanbad 826 004, India*
[b] *CReWMaN Lab, 411 Woolf Hall, 500 First Street, Arlington, TX 76019, USA*
[c] *Advanced Computing and Microelectronics Unit, Indian Statistical Institute, 203 B.T. Road, Calcutta 700 035, India*

Received 25 April 2000; received in revised form 29 April 2002

Communicated by A. El Abbadi

Abstract

A parallel algorithm for prefix computation of $N = n^4$ elements on an $n \times n$ extended multi-mesh network is presented. The network is a modified version of an earlier multi-mesh network with a 4-regular structure. The algorithm takes $O(N^{1/4})$ time on N processors ($13N^{1/4} - 5$ communication steps and $\log N + 4$ arithmetic/logic steps).

Keywords: Prefix computation; Multi-mesh network; Parallel algorithm; Time complexity; Associative binary operation

1. Introduction

Given N data values $x_1, \ldots, x_N$ and the associative binary operation $\circ$, the prefix problem is to compute $P_i = x_1 \circ x_2 \circ x_3 \circ \cdots \circ x_i$, $1 \leqslant i \leqslant N$.

Parallel solutions to many real-life problems such as job scheduling and knapsack depend on the efficiency at which prefix computations can be carried out. Prefix computation is also extensively used in loop optimization, evaluation of polynomials, solution of linear equations, polynomial interpolation, etc. Several parallel algorithms for prefix computation involved have been reported in [6,7,10–12,16]. Owing to the broad applications of this computation, many prefix circuits have also appeared in the literature [8,13–15]. Paper [1] describes an $O(\log N)$ time algorithm on a specialized network with $O(N \log N)$ processors. Lin and Lin [9] present an algorithm on a fully connected message passing system of N processors with no more than $\lceil 1.44 \log N \rceil + 1$ communication steps. On a CRCW-PRAM model, an algorithm requiring $O(\log N / \log\log N)$ time using $(N \log\log N)/\log N$ processors appears in [6]. Egecioglu and Srinivasan [2] give a $2\tau\sqrt{N} + O(\log N)$ time algorithm on a $\sqrt{N} \times \sqrt{N}$ mesh, where τ is the time for a single routing step. They also give an algorithm with $\sqrt{2}\tau\sqrt{N} + O(\sqrt{\tau}N^{1/4})$ time on a disc of N processors.

The multi-mesh network of [4] has several interesting topological properties, e.g., low diameter, existence of a Hamiltonian cycle, 4-regularity. Parallel algorithms for several fundamental problems including summation, matrix-multiplication, sorting, polynomial interpolation, and DFT computation, have been efficiently mapped onto this architecture and reported in [3,5]. A parallel algorithm for finding polynomial roots on this architecture with a little topological modification appears in [17]. An extension of this network is proposed in [18] for its application in designing light wave networks.

Our parallel algorithm for prefix computation on an $N^{1/4} \times N^{1/4}$ extended multi-mesh network uses a modified version of the multi-mesh network of [4]. The algorithm, which is based on the optimal prefix computation mesh given in [2], runs in time $O(N^{1/4})$ for N data values using N processors. It takes $13N^{1/4} - 5$ communication steps and $4\log N^{1/4} + 4$ arithmetic/logic steps. This can be compared with $2\sqrt{N} + 1$ communication steps and $\log N + 1$ arithmetic/logic steps required by the algorithm of [2] on a square mesh using N processors.

2. The computational model

An $n \times n$ multi-mesh network [4] consists of n^2 meshes, each of size $n \times n$. These n^2 meshes are arranged to form an $n \times n$ lattice. Thus, it has $N = n^4$ processors. Each $n \times n$ mesh in this network is called a *block*. Fig. 1 shows a 4×4 multi-mesh network.

Let $BL(\alpha, \beta)$ denote the block in row α and column β. By $P(\alpha, \beta, x, y)$, we denote the processor in row xth row and column yth column of block $BL(\alpha, \beta)$. Within the block, processor $P(\alpha, \beta, x, y)$ is directly connected by to processors $P(\alpha, \beta, x \pm 1, y \pm 1)$, if they exist, by means of bi-directional, links, called *intra-block* links. Additional *inter-block* bi-directional links, which connect the boundary or corner processors between different blocks, are defined as follows:

(1) $P(\alpha, \beta, 1, y)$ is connected to $P(y, \beta, n, \alpha)$ for $1 \leqslant y, \alpha, \beta \leqslant n$. As a special case, for $\alpha = y$ these links connect processors within the same block $BL(\alpha, \beta)$.
(2) $P(\alpha, \beta, x, 1)$ is connected to $P(\alpha, x, \beta, n)$ for $1 \leqslant x, \alpha, \beta \leqslant n$. As a special case, for $\beta = x$ these links connect processors within the same block $BL(\alpha, \beta)$.

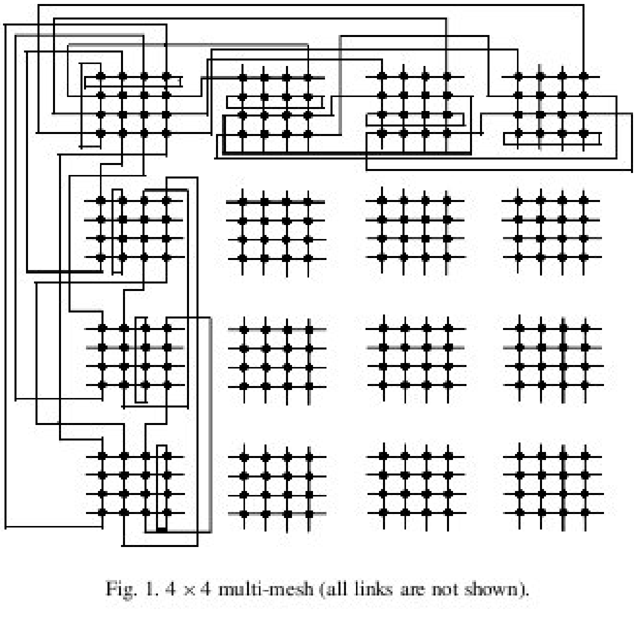

Fig. 1. 4×4 multi-mesh (all links are not shown).

For the sake of efficient mapping of prefix computation, we extend the multi-mesh with extra links as follows. We call such a multi-mesh an $n \times n$ *extended multi-mesh*.

(i) $P(\alpha, \beta, n/2+1, n)$ is connected to $P(\alpha+1, \beta, n/2+1, n)$ for $1 \leqslant \alpha < n$, $1 \leqslant \beta \leqslant n$.
(ii) $P(n/2, \beta, n/2+1, n)$ is connected to $P(n/2, \beta+1, n/2+1, n)$ for $1 \leqslant \alpha \leqslant n$, $1 \leqslant \beta < n$.
(iii) $P(n/2+1, \beta, n/2+1, n)$ is connected to $P(n/2+1, \beta+1, n/2+1, n)$ for $1 \leqslant \beta < n$.

3. The parallel prefix algorithm

The key idea in our parallel algorithm for prefix computation of $N = n^4$ data values on an $n \times n$ extended multi-mesh network is as follows. The computation is performed in two phases. In the first phase, the optimal prefix computation (Algorithm B in [2]) is applied on all the blocks in parallel, and the partial results are stored locally, in each block. In phase two, the prefix computation is computed on the whole architecture using the partial result of phase one. For this, each block is treated as a single virtual processor. To implement phase two, the links among different blocks are used to broadcast the partial results of phase one. For the sake of simplicity, we assume that n is a power of 2. However, the basic idea can easily be extended when this is not so.

The algorithm is described below stepwise. The important steps are illustrated by an example for $n = 4$ with the help of Figs. 2–6. In these figures, solid lines show the data routing using actual and virtual links. We consider, however, the actual (physical) links to calculate the time complexity required by each step of the algorithm. We assume that a communication step requires τ units of time and that a binary operation $\circ$ requires unit time.

Parallel Algorithm:

Initialization: Initially, the data elements are distributed among the processors $P(\alpha, \beta, i, j)$ such that $P(\alpha, \beta, i, j)$ contains the following data (see Fig. 2):

$$x_{(j-1)n+i+(\alpha-1)n^2+(\beta-1)n^3} \quad \text{for } \alpha \leqslant n/2 \text{ and } i \leqslant n/2,$$
$$x_{jn+n/2+1-i+(\alpha-1)n^2+(\beta-1)n^3} \quad \text{for } \alpha \leqslant n/2 \text{ and } i > n/2,$$
$$x_{(j-1)n+i+(3n/2-\alpha)n^2+(\beta-1)n^3} \quad \text{for } \alpha > n/2 \text{ and } i \leqslant n/2,$$
$$x_{jn+n/2+1-i+(3n/2-\alpha)n^2+(\beta-1)n^3} \quad \text{for } \alpha > n/2 \text{ and } i > n/2.$$

Step 1 (*the first phase*): Apply prefix computation on each block in parallel.

Apply the optimal prefix algorithm (given as Algorithm B) in [2] on each block in parallel. The contents of $P(\alpha, \beta, i, j)$ is shown below. Fig. 3 illustrates this step, where we use $i : j$ to denote $x_i \circ x_{i+1} \circ \cdots \circ x_j$. This step requires parallel time $2n\tau + 2\log n + \tau + 1$; it is step 1 of Algorithm B of [2].

$$x\left[1 + (\alpha-1)n^2 + (\beta-1)n^3 : (j-1)n + i + (\alpha-1)n^2 + (\beta-1)n^3\right]$$
for $\alpha \leqslant n/2$ and $i \leqslant n/2$,
$$x\left[1 + (\alpha-1)n^2 + (\beta-1)n^3 : jn + n/2 + 1 - i + (\alpha-1)n^2 + (\beta-1)n^3\right]$$
for $\alpha \leqslant n/2$ and $i > n/2$,
$$x\left[1 + (3n/2-\alpha)n^2 + (\beta-1)n^3 : (j-1)n + i + (3n/2-\alpha)n^2 + (\beta-1)n^3\right]$$
for $\alpha > n/2$ and $i \leqslant n/2$,
$$x\left[1 + (3n/2-\alpha)n^2 + (\beta-1)n^3 : jn + n/2 + 1 - i + (3n/2-\alpha)n^2 + (\beta-1)n^3\right]$$
for $\alpha > n/2$ and $i > n/2$.

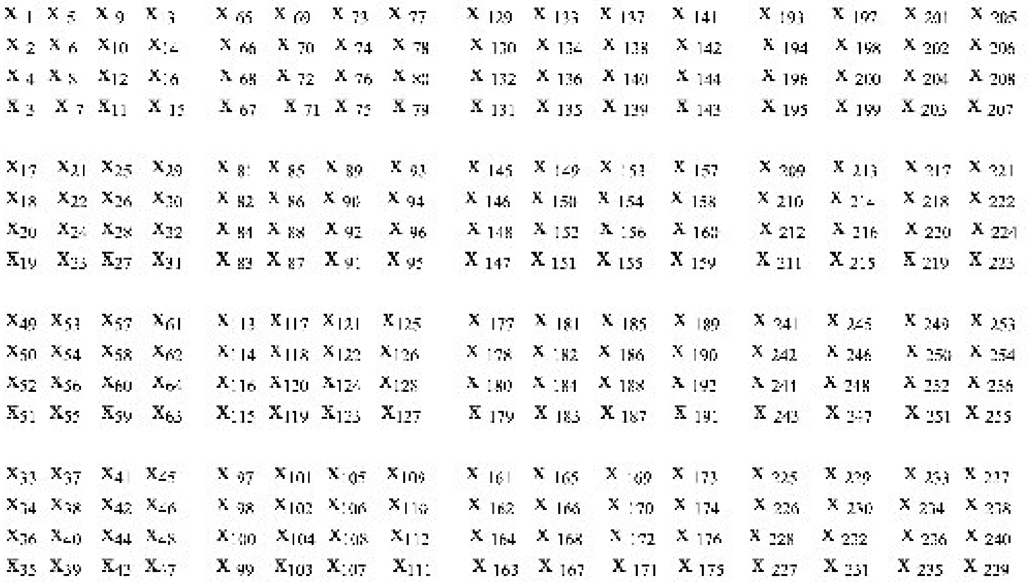

Fig. 2. Initialization.

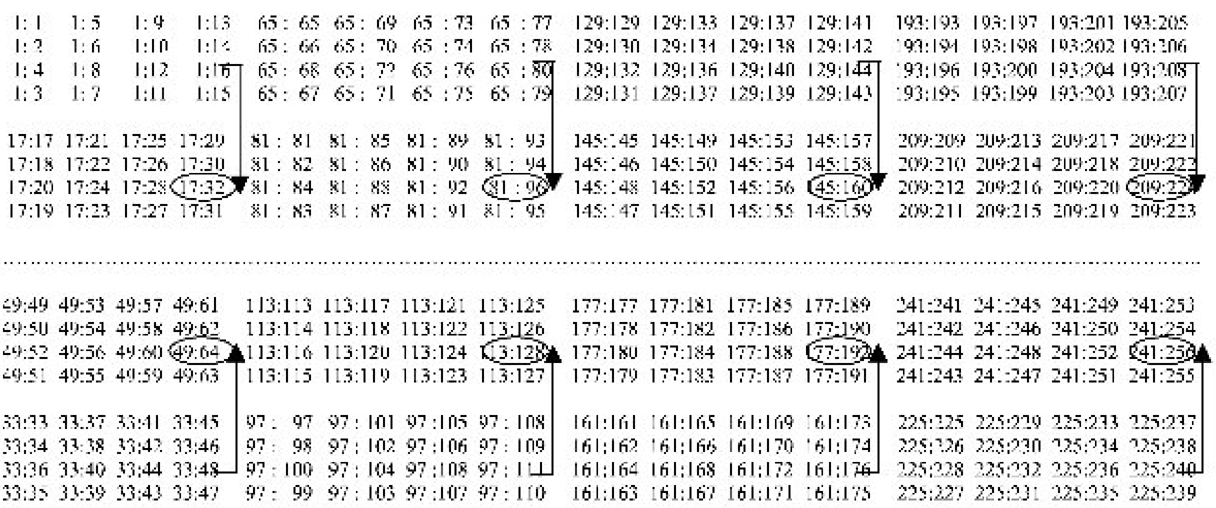

Fig. 3. Situation after step 1.

Step 2 (*start of the second phase*): Apply the same prefix computation on the whole extended multi-mesh network.

Do steps 2.1 and 2.2 in parallel:

2.1 Using the communication links between processors $P(\alpha, \beta, n/2 + 1, n)$ and $P(\alpha + 1, \beta, n/2 + 1, n)$, for $1 \leqslant \alpha < n/2$, $1 \leqslant \beta \leqslant n$, follow the steps similar to those in the algorithm of [16] to compute the prefix over all $n/2$ data values (the values computed after step 1) contained in $P(\alpha, \beta, n/2 + 1, n)$.

2.2 Using the communication links between processors $P(\alpha, \beta, n/2 + 1, n)$ and $P(\alpha + 1, \beta, n/2 + 1, n)$, for $n/2 + 1 < \alpha \leqslant n$, $1 \leqslant \beta \leqslant n$, follow the steps similar to those in the algorithm of [16] to compute the prefix over all $n/2$ data values (prefix values computed after step 1) stored in $P(\alpha, \beta, n/2 + 1, n)$.

Steps 2.1 and 2.2 use the $\log n/2$ arithmetic step algorithm of [16] to compute the prefix on $n/2$ values. The data communication paths are shown in Fig. 3. The number of routing steps required for each step is $n/2 - 1$. Thus, steps 2.1 and 2.2 require parallel time $(n/2 - 1)\tau + \log n/2$.

2.3 For all α, β, $1 < \alpha < n$, $1 \leqslant \beta \leqslant n$, broadcast the currently computed prefix value at $P(\alpha, \beta, n/2 + 1, n)$ to other processors in the same block.

Step 2.3 requires $2(n-1)\tau$ parallel time units.

2.4 For all β, j, $1 \leqslant \beta \leqslant n$, $1 \leqslant j \leqslant n$, $P(\alpha, \beta, i, j)$ computes the following:

$$x\big[(\beta-1)n^3 + 1 : (j-1)n + i + (\alpha-1)n^2 + (\beta-1)n^3\big]$$
for $\alpha \leqslant n/2$ and $i \leqslant n/2$,

$$x\big[(\beta-1)n^3 + 1 : jn + n/2 + 1 - i + (\alpha-1)n^2 + (\beta-1)n^3\big]$$
for $\alpha \leqslant n/2$ and $i > n/2$,

$$x\big[n^3/2 + (\beta-1)n^3 + 1 : (j-1)n + i + (3n/2 - \alpha)n^2 + (\beta-1)n^3\big]$$
for $\alpha > n/2$ and $i \leqslant n/2$,

$$x\big[n^3/2 + (\beta-1)n^3 + 1 : jn + n/2 + 1 - i + (3n/2 - \alpha)n^2 + (\beta-1)n^3\big]$$
for $\alpha > n/2$ and $i > n/2$.

The situation after step 2.4 is illustrated in Fig. 4. Step 2.4 requires one unit of time.

Step 3.

3.1 For all β, $1 \leqslant \beta \leqslant n$, send the values computed in step 2 from processor $P(n/2, \beta, n/2 + 1, n)$ to processor $P(n/2 + 1, \beta, 1, n/2)$ and from processor $P(n/2 + 1, \beta, n/2 + 1, n)$ to $P(n/2, \beta + 1, n, n/2 + 1)$.

The data routing for this step is shown in Fig. 4. This step requires $n\tau$ units of time.

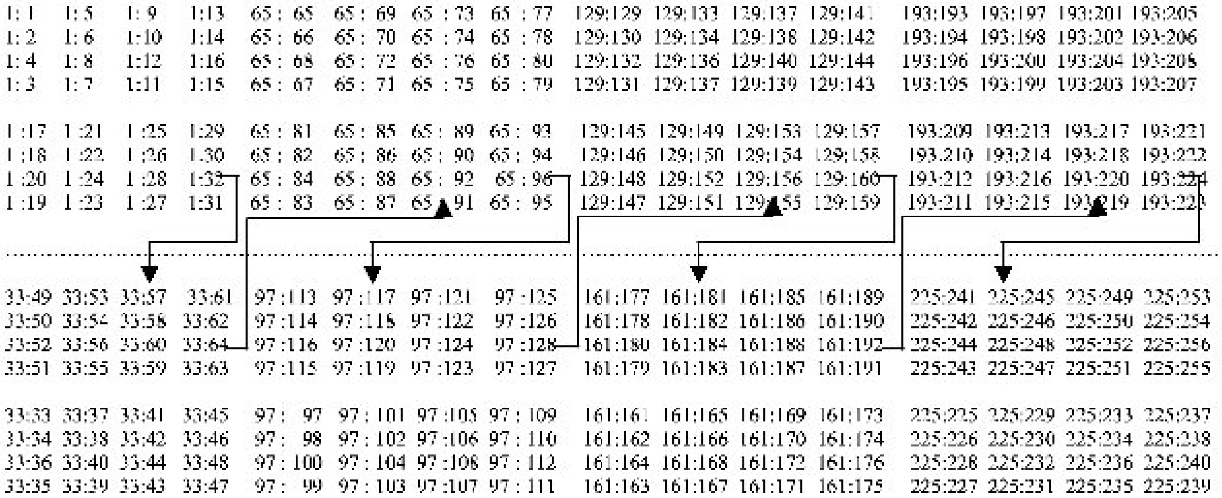

Fig. 4. After step 2.

3.2 Broadcast the data elements received in each block ($n \times n$ mesh) in step 3.1 to all other processors in that block.

Step 3.2 needs $2(n-1)\tau$ units of time.

3.3 For all j, $1 \leqslant j \leqslant n$, $P(n/2, \beta, i, j)$ computes:

$$x\left[n^3/2 + (\beta - 2)n^3 + 1 : (j-1)n + i + (\alpha - 1)n^2 + (\beta - 1)n^3\right]$$
for $2 \leqslant \beta \leqslant n$ and $i \leqslant n/2$,

$$x\left[n^3/2 + (\beta - 2)n^3 + 1 : jn + n/2 + 1 - i + (\alpha - 1)n^2 + (\beta - 1)n^3\right]$$
for $2 \leqslant \beta \leqslant n$ and $i > n/2$,

and $P(n/2+1, \beta, i, j)$ computes:

$$x\left[(\beta - 1)n^3 + 1 : (j-1)n + i + (3n/2 - \alpha)n^2 + (\beta - 1)n^3\right]$$
for $1 \leqslant \beta \leqslant n$ and $i \leqslant n/2$,

$$x\left[(\beta - 1)n^3 + 1 : jn + n/2 + 1 - i + (3n/2 - \alpha)n^2 + (\beta - 1)n^3\right]$$
for $1 \leqslant \beta \leqslant n$ and $i > n/2$.

Step 3.3 needs one unit of time. The content of each processor after step 3 is shown in Fig. 5.

Step 4. The processors in the blocks of row $n/2+1$, perform their prefix computations along this row. Similarly the processors in the blocks of row $n/2$ perform their prefix computations along this row as shown in Fig. 5.

Do steps 4.1 and 4.2 in parallel.

4.1 For all β, $1 \leqslant \beta < n$, using the communication links between processors $P(n/2, \beta, n/2+1, n)$ and $P(n/2, \beta+1, n/2+1, n)$, follow steps similar to those in the algorithm of [16] to compute the prefix over all n values (prefix values computed by step 3) contained in $P(n/2, \beta, n/2+1, n)$ and $P(n/2, n, n/2+1, n)$.

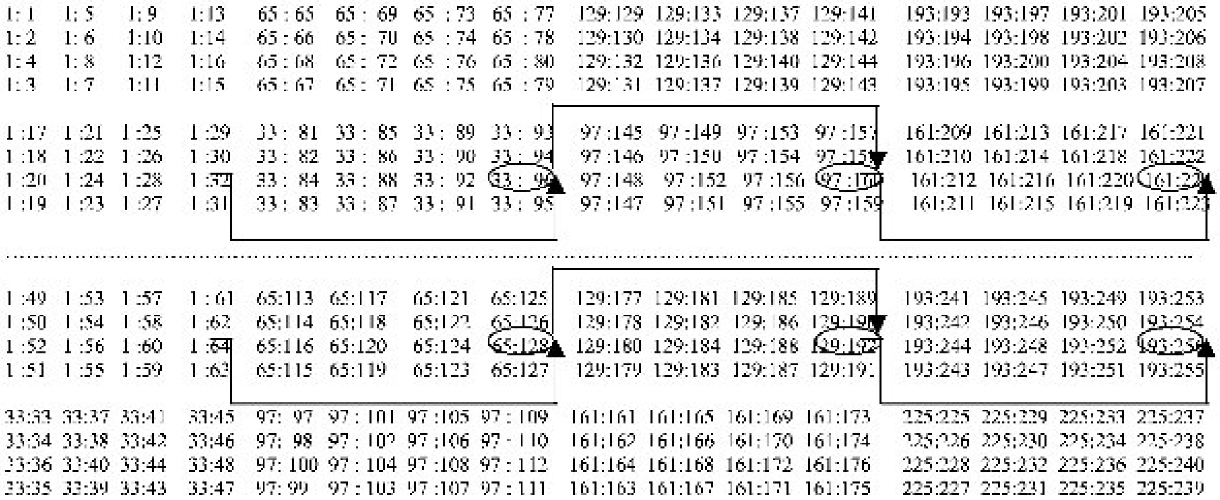

Fig. 5. After step 3.

4.2 For all β, $1 \leqslant \beta < n$, using the communication links between processors $P(n/2+1, \beta, n/2+1, n)$ and $P(n/2+1, \beta+1, n/2+1, n)$, follow steps similar to those in the algorithm of [16] to compute the prefix over all the n values (prefix values computed by step 3) contained in $P(n/2+1, \beta, n/2+1, n)$ and $P(n/2+1, n, n/2+1, n)$.

Steps 4.1 and 4.2 calculate the prefix operation as indicated by circles in Fig. 5 in $(n-1)$ communication steps and $\log n$ arithmetic steps [16], so these steps require parallel time $(n-1)\tau + \log n$.

4.3 For all β, $1 < \beta \leqslant n$, broadcast the computed prefix values in $P(n/2, \beta-1, n/2+1, n)$ and $P(n/2+1, \beta-1, n/2+1, n)$ to all processors in the blocks $BL(n/2, \beta)$ and $BL(n/2+1, \beta)$, respectively.

Step 4.3 needs $(1+(n-1)+n/2)\tau = 1.5n\tau$ units of time.

4.4 For all j, $1 \leqslant j \leqslant n$, $P(n/2, \beta, i, j)$ computes

$$x\left[1:(j-1)n+i+(n/2-1)n^2+(\beta-1)n^3\right]$$

for $2 \leqslant \beta \leqslant n$ and for $i \leqslant n/2$,

$$x\left[1:jn+n/2+1-i+(n/2-1)n^2+(\beta-1)n^3\right]$$

for $2 \leqslant \beta \leqslant n$ and for $i > n/2$,

and processor $P(n/2+1, \beta, i, j)$ computes

$$x\left[1:(j-1)n+i+(n-1)n^2+(\beta-1)n^3\right]$$

for $2 \leqslant \beta \leqslant n$ and for $i \leqslant n/2$,

$$x\left[1:jn+n/2+1-i+(n-1)n^2+(\beta-1)n^3\right]$$

for $2 \leqslant \beta \leqslant n$ and for $i > n/2$.

Step 4.4 requires one unit of time. Fig. 6 illustrates the resulting situation after Step 4.

Step 5. Do steps 5.1 and 5.2 in parallel.

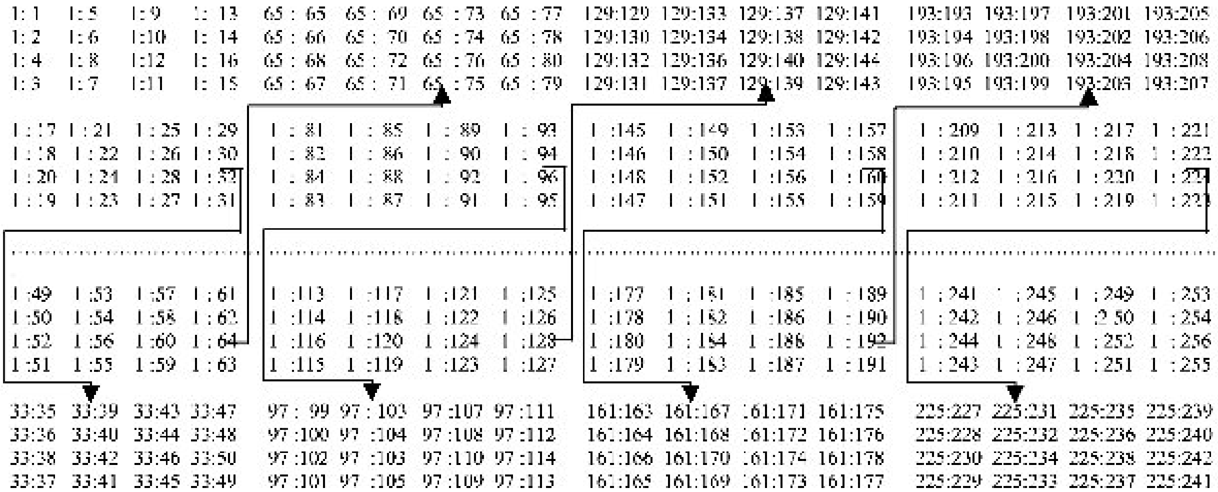

Fig. 6. Situation after step 4.

5.1 For all β, $1 \leqslant \beta < n$, send the value computed in step 4 from processor $P(n/2+1, \beta, n/2+1, n)$ to $P(n/2-1, \beta+1, n, n/2+1)$.

5.2 For all β, $1 \leqslant \beta \leqslant n$, send the values computed in step 4 from the processor $P(n/2, \beta, n/2+1, n)$ to $P(n/2+2, \beta, 1, n/2)$.

The data routing for steps 5.1 and 5.2 is shown in Fig. 6. Step 5 requires parallel time $(n/2+1)\tau$.

Step 6. For all β, $1 < \beta \leqslant n$, processors $P(n/2-1, \beta+1, n, n/2+1)$ and $P(n/2+2, \beta, 1, n/2)$ broadcast their computed prefix values to any one processor in each of the blocks in their corresponding columns (i.e., blocks with the same β). (This can be done by first broadcasting the data along a row (row n of block $BL(n/2-1, \beta+1)$ or the first row of block $BL(n/2+2, \beta)$ and then sending the data to one processor of each of the other blocks with the same β by following one more link of the multi-mesh network.) Then, broadcast these values to all other processors in the corresponding blocks.

In step 6, sending data to any one processor of each of the other blocks requires $(n/2+1)\tau$ time units. Following that, broadcasting data within a block needs time $2(n-1)\tau$.

Step 7. Each processor calculates its target prefix value in unit time, so that $P(\alpha, \beta, i, j)$ contains

$$x\left[1 : (j-1)n + i + (\alpha-1)n^2 + (\beta-1)n^3\right]$$

for $\alpha \leqslant n/2$ and $i \leqslant n/2$,

$$x\left[1 : jn + n/2 + 1 - i + (\alpha-1)n^2 + (\beta-1)n^3\right]$$

for $\alpha \leqslant n/2$ and $i > n/2$,

$$x\left[1 : (j-1)n + i + (3n/2-\alpha)n^2 + (\beta-1)n^3\right]$$

for $\alpha > n/2$ and $i \leqslant n/2$,

$$x\left[1 : jn + n/2 + 1 - i + (3n/2-\alpha)n^2 + (\beta-1)n^3\right]$$

for $\alpha > n/2$ and $i > n/2$.

The final result is shown in Fig. 7. The total computation time required by all the above steps is $(13n-5)\tau + 4\log n + 4$.

1: 1	1: 5	1: 9	1: 13	1: 65	1: 69	1: 73	1: 77	1: 129	1: 133	1: 137	1: 141	1: 193	1: 197	1: 201	1: 205
1: 2	1: 6	1: 10	1: 14	1: 66	1: 70	1: 74	1: 78	1: 130	1: 134	1: 138	1: 142	1: 194	1: 198	1: 202	1: 206
1: 4	1: 8	1: 12	1: 16	1: 68	1: 72	1: 76	1: 80	1: 132	1: 136	1: 140	1: 144	1: 196	1: 200	1: 204	1: 208
1: 3	1: 7	1: 11	1: 15	1: 67	1: 71	1: 75	1: 79	1: 131	1: 137	1: 139	1: 143	1: 195	1: 199	1: 203	1: 207
1: 17	1: 21	1: 25	1: 29	1: 81	1: 85	1: 89	1: 93	1: 145	1: 149	1: 153	1: 157	1: 209	1: 213	1: 217	1: 221
1: 18	1: 22	1: 26	1: 30	1: 82	1: 86	1: 90	1: 94	1: 146	1: 150	1: 154	1: 158	1: 210	1: 214	1: 218	1: 222
1: 20	1: 24	1: 28	1: 32	1: 84	1: 88	1: 92	1: 96	1: 148	1: 152	1: 156	1: 160	1: 212	1: 216	1: 220	1: 224
1: 19	1: 23	1: 27	1: 31	1: 83	1: 87	1: 91	1: 95	1: 147	1: 151	1: 155	1: 159	1: 211	1: 215	1: 219	1: 223
1: 49	1: 53	1: 57	1: 61	1: 113	1: 117	1: 121	1: 125	1: 177	1: 181	1: 185	1: 189	1: 241	1: 245	1: 249	1: 253
1: 50	1: 54	1: 58	1: 62	1: 114	1: 118	1: 122	1: 126	1: 178	1: 182	1: 186	1: 190	1: 242	1: 246	1: 250	1: 254
1: 52	1: 56	1: 60	1: 64	1: 116	1: 120	1: 124	1: 128	1: 180	1: 184	1: 188	1: 192	1: 244	1: 248	1: 252	1: 256
1: 51	1: 55	1: 59	1: 63	1: 115	1: 119	1: 123	1: 127	1: 179	1: 183	1: 187	1: 191	1: 243	1: 247	1: 251	1: 255
1: 35	1: 39	1: 43	1: 47	1: 99	1: 103	1: 107	1: 111	1: 163	1: 167	1: 171	1: 175	1: 227	1: 231	1: 235	1: 239
1: 36	1: 40	1: 44	1: 48	1: 100	1: 104	1: 108	1: 112	1: 164	1: 168	1: 172	1: 176	1: 228	1: 232	1: 236	1: 240
1: 38	1: 42	1: 46	1: 50	1: 102	1: 103	1: 110	1: 114	1: 166	1: 170	1: 174	1: 178	1: 230	1: 234	1: 238	1: 242
1: 37	1: 41	1: 45	1: 49	1: 101	1: 105	1: 109	1: 113	1: 165	1: 169	1: 173	1: 177	1: 229	1: 233	1: 237	1: 241

Fig. 7. Final result.

Acknowledgements

The first version of this paper appeared in the proceedings of the 7th Internat. Conf. on Advanced Computing and Comm. (ADCOM'99), December 20–22, 1999, Roorkee, India. The authors thank the anonymous referees for their valuable comments and suggestions, which greatly helped in improving the results.

References

[1] S.G. Akl, The Design and Analysis of Parallel Algorithms, Prentice-Hall, Englewood Cliffs, NJ, 1989.
[2] O. Egecioglu, A. Srinivasan, Optimal parallel prefix on mesh architecture, Parallel Algorithms Appl. 1 (1993) 191–209.
[3] M. De, D. Das, B.P. Sinha, An new network topology with multiple meshes, Tech. Rep. No. T/Rep / E-94/01, Indian Statistical Institute, Calcutta, 1994.
[4] D. Das, M. Dey, B.P. Sinha, A new network topology with multiple meshes, IEEE Trans. Comput. 68 (5) (1999) 536–551.
[5] M. De, D. Das, M. Ghosh, B.P. Sinha, An efficient sorting algorithm on multi-mesh network, IEEE Trans. Comput. 46 (10) (1997) 1132–1137.
[6] R. Cole, U. Vishkin, Faster optimal parallel prefix sum and list ranking, Inform. and Control 4 (1989) 334–352.
[7] C.P. Krushkal, T. Madege, L. Rudolph, Parallel prefix on fully connected direct connection machines, in: Proc. Internat. Conf. Parallel Process., 1986, pp. 278–284.
[8] S. Lakshivarahan, S.K. Dhal, Parallel Computing Using the Prefix Problem, Oxford University Press, Oxford, 1994.
[9] Y.C. Lin, C.M. Lin, Efficient parallel prefix algorithms on fully connected message passing computers, in: Proc. 3rd Int. Conf. on High Performance Computing (HiPC), Trivandrum, India, December 19–22, 1996.
[10] A. Nicolan, H. Hwang, Optimal schedule for parallel prefix computation with bounded resources, in: Proc. 3rd ACM SIGPLAN Symp. Principles and Practice Parallel Programming, 1991, pp. 1–10.
[11] P.M. Kogge, Parallel solution of recurrence problems, IBM J. Res. Development (1974) 138–148.
[12] R. Lander, M.J. Fisher, Parallel prefix computation, J. ACM 27 (1980) 831–839.
[13] D.A. Carlson, B. Sugla, Limited width parallel prefix circuits, J. Supercomput. 4 (1990) 107–129.
[14] F.E. Fich, New bounds for parallel prefix circuits, in: Proc. 15th Symp. Theory of Computing, 1993, pp. 100–109.
[15] M. Snir, Depth-size trade-offs for parallel prefix computation, J. Algorithms 7 (1986) 185–201.
[16] C.P. Krushkal, L. Rudolh, M. Snir, The power of parallel prefix, IEEE Trans. Comput. C-34 (10) (1985) 965.
[17] P.K. Jana, Finding polynomial zeros on a multi-mesh of trees (MMT), in: Proc. 2nd Internat. Conf. on Information Technology, Bhubaneshwar, India, December 20–22, 1999, pp. 203–206.
[18] A. Sen, S. Bandyopadhyay, B.P. Sinha, A new architecture and a new metric for lightwave networks, IEEE J. Lightwave Technology 19 (2001) 913–925.