Parallel Algorithm for Prefix Computation on OTIS k-Ary 3-Cube Parallel Computers

Keny T. Lucas
Department of Information Management, Xavier Institute of Social Service, Ranchi, Jharkhand, India
Email: kennylucas@xisss.ac

Abstract—The OTIS (Optical Transpose Interconnection System) has been a popular interconnection model for developing parallel processing systems. Various real-life problems including job scheduling, knapsack, loop optimization, evaluation of polynomials, solutions of linear equations, and polynomial interpolation depend on the time complexity of prefix computation for the efficiency of their respective solutions. In this paper we have proposed parallel algorithm for prefix computation on SIMD model of OTIS k-ary 3-cube. The prefix computation for N(\(8k^4\)) data elements requires \(O(k)\) electronic moves and \(O(1)\) OTIS move on \(k^3\) processors.

Index Terms—interconnection network, OTIS, parallel algorithm, prefix computation, time complexity

I. INTRODUCTION

The OTIS (Optical transpose Interconnection System) [1], [2] is a hybrid system that exploits the qualities of electronic links as well as optical links for developing multicomputers. The optical links are superior to electronic links in terms of power, speed and crosstalk properties if the connect distance between processors is more than a few millimeters. The electronic links are preferred to optical links for smaller distance between processors [3], [4]. In an OTIS model, the total number of processors in the network is divided into groups and each group can be assumed to be a microchip. All the processors within a group are connected through the electronic links whereas the processors of different groups are connected through the optical links. The number of groups in the network can be equal to the number of processors in each group for maximized bandwidth and minimized power consumption [1]. The interconnection pattern of processors within each group determines the overall model of such system, i.e. an OTIS-G has G interconnection pattern for all of its groups. Some of the OTIS models are OTIS-Ring, OTIS-Mesh, OTIS-Hypercube, OTIS-Torus, OTIS-Mesh of trees [2], [5]. In the recent years, many parallel algorithms have been proposed for various OTIS models that includes image processing [6], matrix multiplication [7], basic operations [8], BPC permutation [9], \(k\)-\(k\) sorting [10], randomized algorithm [11], extreme finding [12], decentralized consensus protocol [13], polynomial interpolation [14], [15], polynomial root finding [14], construction of conflict graph [16], gossiping [17], [18] etc.

Many real-life problems, such as job scheduling, knapsack, loop optimization, evaluation of polynomials, solutions of linear equations, and polynomial interpolation depend on the efficiency of prefix computation for their solutions. For a given set of data elements, i.e. \(x_1, x_2, x_3, \ldots, x_N\) belonging to a domain \(\mathcal{D}\), the prefix computation can be given as \(P_i = x_1 \oplus x_2 \oplus x_3 \oplus \ldots \oplus x_i\), \(1 \leq i \leq N\) where \(\oplus\) is an associative operator over the domain \(\mathcal{D}\). Many parallel algorithms for prefix computation have been proposed in [19], [20], [21], [22], [23], [24], [25], [26]. The algorithm proposed in [19] for extended multi-mesh network requires \(O(N^{\alpha})\) on \(N\) processors. The algorithm proposed in [25] on optical multi-trees requires \(O(\log n)\) electronic moves + 4 optical moves for \(n^2\) data elements on \(n^2 \times n^2\) processors. In this paper we propose a parallel algorithm for prefix computation on SIMD model of OTIS k-ary n-cube for \(n=3\). Our proposed algorithm requires \(O(k)\) electronic moves + \(O(1)\) optical moves.

The rest of the paper is organized as follows. Section 2 describes the topology of the OTIS k-ary n-cube. Our proposed algorithm is discussed in section 3 followed by conclusion in section 4.

II. TOPOLOGY OF OTIS K-ARY 3-CUBE

In an OTIS k-ary 3-cube network, there are \(N = k^6\) processors altogether in the network. These processors are divided into \(k^3\) groups and each group contains \(k^3\) processors within it. In this architecture, each group is k-ary 3-cube with three dimensional grid structures. The two most variant of k-ary n-cube are hypercube (\(k = 2\)) and torus (\(n = 2\) or 3). The hypercube has been used in iPSC/2 [27] and iPSC/860 [28] whereas the torus has been used in J-Machine [29], CRAY-3TD [30] and CRAY-3TE [31] parallel computers. All the intra-group processors are connected to their adjacent processors through the electronic links whereas the inter-group processors are connected through the optical links. Let each processor be denoted by \((G_{g_1,g_2,g_3}P_{p_1,p_2,p_3})\), then each optical link connects processor \((G_{g_1,g_2,g_3}P_{p_1,p_2,p_3})\) and \((G_{g_1,g_2,g_3}P_{p_1,p_2,p_3})\) where \(1 \leq g_1, g_2, g_3, p_1, p_2, p_3 \leq k\). The topology of OTIS k-ary n-cube for \(k = 3\); \(n = 3\) for \(g_1=1\) and \(p_1=1\) is shown in Fig. 1, where \(n\) represents the dimension of each group and \(k\) represents the number of processors in each dimension.

The dotted lines represent optical links and the solid lines...
represent the electronic links. The topology of OTIS k-ary n-cube for k=3 and n=3 can be constructed in the similar fashion.

![Figure 1. Topology of OTIS k-ary n-cube for k = 3 and n = 3](image)

The indices $g_1$, $g_2$, $g_3$ and $p_1$, $p_2$, $p_3$ represent the location of the group in the overall OTIS architecture and the location of the processor within the group respectively. The diameter of an OTIS k-ary n-cube network is $2n\lfloor k/2 \rfloor + 1$. The Hamiltonian cycle can also be easily embedded on OTIS k-ary n-cube model. We have represented each data movement through electronic link as electronic move and that through optical link as OTIS move. We also assume that each data link is bidirectional.

III. PROPOSED ALGORITHM

In our proposed algorithm, we assume that the data set is populated in row-major order within the group as well as outside the group (group-wise). We will count the electronic moves and the OTIS moves separately to analyze the time complexity of our proposed algorithm. In our proposed algorithm, four registers can be used to compute the prefix sum on the given network. The basic idea of our proposed algorithm is as follows: In the first stage, prefix sum is computed in $p_1$ dimension within all the groups in parallel using register $A$. The content of the last processor ($P_{p_1,p_2,k}$) is sent to all its previous processors using register $B$ in the same dimension. In the second stage, the content of $B$ register of each processor is sent to next processor in $p_2$ dimension using register $C$ to add with the value in $B$. This process also continues until the last processor in $p_2$ dimension gets the data from all its previous processors. This stage gives the prefix sum in the row-major order in $p_2$ and $p_3$ dimensions.

Then, the data of processor ($P_{p_1,k,k}$) is sent to all the processors in $p_2$, $p_3$ dimension. This data is used to compute prefix sum in $p_1$ dimension and this stage gives the complete prefix sum within each group. After, intra-group prefix computation is done, inter-group prefix sum must be performed. To achieve this, the content of processor ($P_{k,k,k}$) of each group is populated in all the processors within the group and must be sent to all the groups of higher coordinates through optical links. All these optically received data within each group must be added with that of all the processors within that group to obtain the final prefix sum. The proposed algorithm is discussed below.

A. Algorithm Prefix-Computation

Data initialization: The initial data set is populated in the row-major order within the group as well as outside the group.

Step 1: For all groups, do in parallel
Compute prefix sum for processors in dimension $p_3$, i.e. row-wise

Step 2: For all groups, do in parallel
Broadcast the data of processor ($G_{g_1,g_2,g_3}B_{p_1,p_2,k}$) obtained in step 1 row-wise within each group

Step 3: For all groups, do in parallel
Move the contents obtained from step 2 to the successive rows to compute the prefix sum of data in row-major order, i.e. in each $p_2p_3$ plane within each group

Step 4: For all groups, do in parallel
Broadcast the data of processor ($G_{g_1,g_2,g_3}B_{p_1,k,k}$) obtained in step 3 in $p_2$ and $p_3$ dimensions

Step 5: For all groups, do in parallel
Send the data from each processor to the respective processors in $p_1$ dimension to get the prefix sum group-wise

Step 6: For all groups, do in parallel
Broadcast the data of processor ($G_{g_1,g_2,g_3}A_{k,k,k}$) within the group obtained in step 5

Step 7: For all groups, do in parallel
Perform OTIS move, unidirectional from lower-coordinate groups to higher-coordinate groups, on data obtained in step 6

Step 8: For all groups, do in parallel
Add all the data obtained in step 7 and broadcast within the group

Step 9: For all groups, do in parallel
Add the data obtained from steps 5 and 8 in each processor to obtain the final prefix-sum

Time Complexity: Steps 1, 3 and 5, each requires $k-1$ electronic moves. Steps 2, 4, 6 and 8 require $\lfloor k/2 \rfloor \leq \lfloor k/2 \rfloor$. 
and $\lfloor k/2 \rfloor$ respectively. One OTIS is needed in step 7 whereas step 9 takes constant time. Thus, the overall time complexity of our proposed parallel algorithm for prefix computation of $k^2$ data elements on $k^2$-processor $k$-ary $n$-network is $O(k)$ electronic moves + $O(1)$ OTIS move.

IV. CONCLUSIONS

We have proposed a parallel algorithm for computing prefix sum on an SIMD model of OTIS $k$-ary 3-cube parallel computer. The time complexity of our proposed algorithm is $O(k)$ electronic moves and $O(1)$ OTIS move.

REFERENCES


