Scalable Computing: Practice and Experience ISSN 1895-1767
Volume 12, Number 2, pp. 275–282. http://www.scpe.org

SOME GEOMETRIC PROBLEMS ON OMTSE OPTOELECTRONIC COMPUTER

SATISH CH. PANIGRAHI* AND ASISH MUKHOPADHYAY†

Abstract. Optical Multi-Trees with Shuffle Exchange (OMTSE) architecture is an efficient model of an optoelectronic computer. The network has a total of $3n^3/2$ nodes. The diameter and bisection width of the network are $6\log n - 1$ and $n^3/4$ respectively. In this note, we present synchronous SIMD algorithms on an OMTSE optoelectronic computer for the following problems in computational geometry: Convex Hull, Smallest Enclosing Rectangle, All-Farthest/All-Nearest Neighbors, Closest/Farthest pair, Maximal Points. The strength of the proposed algorithms over the existing algorithms on OMULT has also been discussed.

Key words: Parallel Algorithms, Optoelectronic Computer, Computational Geometry, OTIS Mesh, OMULT

1. Introduction. Optical interconnections are superior in power, speed with less crosstalk properties as compared to electronic interconnections when the interconnection distance is more than a few millimeters [1, 6]. Motivated by these observations, some new hybrid optoelectronic computer architectures utilizing both optical and electronic technologies have been proposed and investigated by several researchers [8, 10, 13, 15]. In these architectures, both the electronic link and the optical link are done where the former is being considered within the same physical package (e.g. chip) where as the latter is for the pair of processors that are kept in different packages.

A number of parallel algorithms on these optoelectronic computers have been addressed and studied extensively [3, 4, 5, 7, 8, 9, 10, 14]. In this paper we present some computational geometry algorithms such as Convex Hull, Smallest Enclosing Rectangle, All-Farthest/All-Nearest Neighbor, Closest/Farthest pair, Maximal Points, on OMTSE optoelectronic computer [8, 10]. Irrespective of different factor network of OMTSE than OMULT, here in this paper we show that Convex hull and Smallest Enclosing Rectangle problem for n points can be solved on OMTSE in $O(\log n)$ time with the same time complexity as on OMULT [2]. Here it is worth noting that the total number of processors of OMULT and OMTSE respectively be $\delta 1 = n^2(2n-1)$ and $\delta 2 = n^2(\frac{3n}{2})$ (we have $\delta 1 < \delta 2$, as because of their topological nature we can assume that $n \geq 4$). Islam et al. in [2] stated that algorithm for empirical cumulative distribution, all nearest neighbor can be implemented on OMULT in $O(\log n)$ time for n number of points. In this paper we explore this line of work farther and implement the algorithms such as All-Farthest/All-Nearest Neighbor, Closest/Farthest pair among n^2 points in $O(n \log n)$ time and also provide an algorithm for maximal points among n^3 data points in $O(\log n)$ time.

The rest of the paper is organized as follows. In section 2 we briefly present the topological property of the OMTSE System. In section 3, we describe our propose algorithms and finally we conclude in section 4.

2. Topology of OMTSE. The factor network used in OMTSE topology constitutes two layer Trees with Shuffle Exchange (TSE) network. The TSE is nothing but an interconnection network containing a group of $2^k, k \geq 1$, complete binary trees of height one and the roots of these binary trees are connected with Shuffle-Exchange fashion. The OMTSE interconnection system consists of n^2 TSE networks, which are organized in the form of an $n \times n$ grid in matrix form. We denote the TSE network placed at i_{th} row and j_{th} column of this matrix by $G_{ij}, 1 \leq i, j \leq n$. Each TSE network has n nodes at layer 2 and $n/2$ nodes at layer 1 which results in N $= 3n^3/2$ processors in total. The nodes within each TSE network are interconnected by usual electronic links, while the nodes at layer 2 (i.e. the layer having leaf processors) of different TSE networks are interconnected by optical links according to the rules defined below. Let us label the nodes in each TSE network $G_{ij}, 1 \leq i, j \leq n$, by distinct integers from 1 to $3n/2$ in reverse order, i.e., the nodes at both layer 2 and 1 of TSE network are numbered from 1 to $3n/2$ in order from left to right. The node, k, in a TSE network G_{ij} will be referred as the processor $P(i, j, k), 1 \leq i, j \leq n, 1 \leq k \leq 3n/2$. We can now define the optical links interconnecting only leaf nodes in different TSE networks in the following way.

*School of Computer Science, University of Windsor, Canada(panigra@uwindsor.ca).
†School of Computer Science, University of Windsor, Canada(asishm@cs.uwindsor.ca)

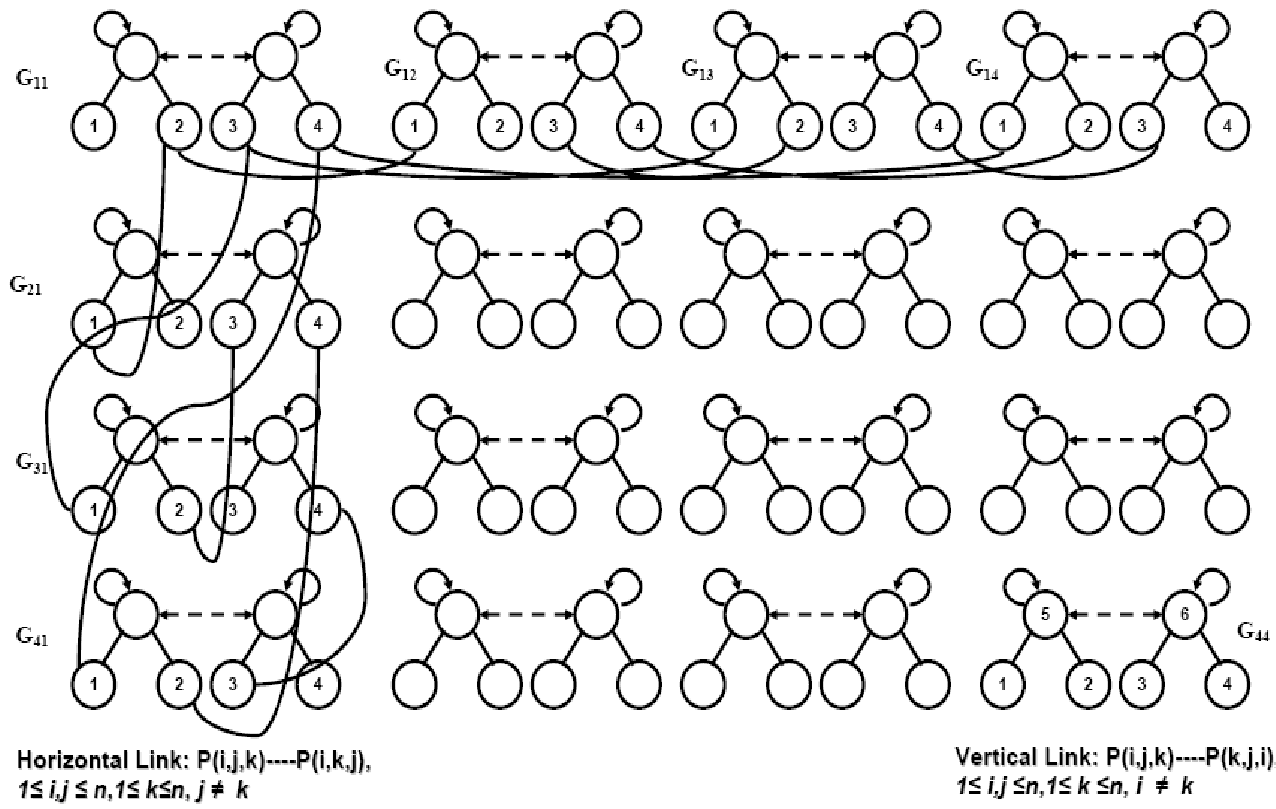

Fig. 2.1: An example of OMTSE topology with n = 4

(1) Processor $P(i,j,k), 1 \leq i,j,k \leq n, j \neq k$, is connected to the processor $P(i,k,j)$ by bi-directional optical link called horizontal inter-TSE link.

(2) Processor $P(i,j,k), 1 \leq i,j \leq n, i \neq k$, is connected to the processor $P(k,j,i)$ by bi-directional optical link called vertical inter-TSE link.

The diameter of a network is defined as the maximum distance between any two processing nodes in the network. If we start from a node $P(i,j,k), 1 \leq i,j \leq n, 1 \leq k \leq 3n/2$, we can reach another node $P(i',j',k'), 1 \leq i',j' \leq n, 1 \leq k' \leq 3n/2$, of the OMTSE interconnection system by traversing the path

$P(i,j,k) \rightarrow P(i,j,j') \rightarrow P(i,j',j) \rightarrow P(i,j',i') \rightarrow P(i',j',i) \rightarrow P(i',j',k')$

It can easily be seen that the diameter of OMTSE topology is $6\log n - 1$ which is $O(\log n)$ comprising of $6\log n - 3$ electronic links and 2 optical links. Similarly we can find out the bisection width of OMTSE topology is equal to $n^3/4$. An Example of OMTSE topology for $n = 4$ with partial links is shown in FIG. 2.1.

3. Proposed Algorithms.

3.1. Convex Hull. The convex hull [11] of a set of points S in the plane is smallest convex polygon P that encloses S, smallest in the sense that there is no other polygon P' such that $P \supset P' \supseteq S$. To find the convex hull for a given set of points S on a plane we need to identify the extreme points, in particular, what constitutes constructing the boundary. Suppose $|S| = n$ and assume that no three points in S are collinear then our algorithm employs the result of the following theorem discussed in [14].

THEOREM 3.1. *For any point $p_i \in S$, let $p_{j0}, p_{j1}, ..., p_{jn-2}$ be the points in $S - p_i$ (i.e. $p_{jk} \neq p_i, 0 \leq k \leq n-2$), sorted by the polar angle made by the vector $\overrightarrow{p_i p_{jk}}, 0 \leq k \leq n-2$. The point p_i is an extreme point of S iff there is a $k, 0 \leq k \leq n-2$, such that counterclockwise angle between p_{ik} and $p_{i(k+1)mod(n-1)}$ is more than π.*

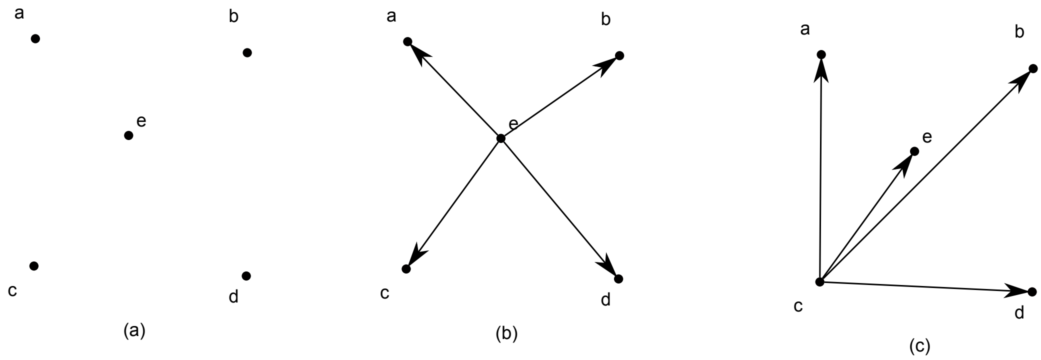

Fig. 3.1: Example for Theorem 3.1:(a) Original Layout, (b) $p_i = e$, (c) $p_i = c$

We assume that each leaf processor $P(i, j, k)(1 \leq i, j, k \leq n)$ has three registers, represent by $A(i, j, k)$, $B(i, j, k)$ and $C(i, j, k)$. We have a set of points $S = p_1, p_2, ..., p_n$ in which no three points are collinear. The coordinates of all n points are initially stored in the A-register of the leaf nodes G_{11}.

Algorithm: $ConvexHull()$

Input: $\forall k, 1 \leq k \leq n$
$A(1, 1, k) \leftarrow p_k$
Output: $\forall k, 1 \leq k \leq n$
Extreme points $\leftarrow$ B(1, 1, k)

Step 1: $\forall i, j; 1 \leq i, j \leq n$, do in parallel
Broadcast all these n points to the A-register of the respective leaf nodes of G_{ij} [9].
Step 2: $\forall i, j, k; 1 \leq i, j, k \leq n$, do in parallel
Broadcast the point in the A-register of $P(i, j, i)$ to all the $B(i, j, k)$ of G_{ij}.
Step 3: $\forall i, j, k; 1 \leq i, j, k \leq n$, do in parallel
Compute the polar angle of the vector $\overrightarrow{p_ip_{ik}}$ at $P(i, j, k)$ of G_{ij} and store in $C(i, j, k)$ along with the zero vector.
Step 4: $\forall i, j, k; 1 \leq i, j, k \leq n$, do in parallel
Sort the n vectors $\overrightarrow{p_ip_{ik}}$ stored in the C-register of the leaf nodes of each G_{ij}. After this step we assume the sorted order list given by each G_{ij} is $p_ip_{i1}, p_ip_{i2}, ..., p_ip_{in}$ (i.e. in each G_{ij} the vector $\overrightarrow{p_ip_{i1}}$ always represent the zero vector.)
Step 5: $\forall i, j, k; 1 \leq i, k \leq n$, and $2 \leq j \leq n$, do in parallel
Broadcast the content of $C(i, j, j)$ to $A(i, j, k)$.
Step 6: $\forall i; 1 \leq i \leq n$, do in parallel
i) $\forall j, 2 \leq j \leq n - 1$,
Calculate the counter clockwise angle between $\overrightarrow{p_ip_{ij}}$ and $\overrightarrow{p_ip_{i(j+1)}}$ at each G_{ij} and store the result in $C(i, j, j + 1)$
ii) $\forall j, j = n$,
Calculate the counter clockwise angle between $\overrightarrow{p_ip_{in}}$ and $\overrightarrow{p_ip_{i2}}$ at each G_{in} and store it in $C(i, n, 2)$.
Step 7: $\forall i; 1 \leq i \leq n$, do in parallel
i) $\forall j; j = 1$
$C(i, j, 1) \leftarrow 0$.

ii) $\forall j; 2 \leq j \leq n-1$
if $((C(i,j,j+1) > \pi))$
$C(i,j,1) \leftarrow 1.$
else
$C(i,j,1) \leftarrow 0.$
iii) $\forall j; j = n$
if $((C(i,n,2) > \pi))$
$C(i,n,1) \leftarrow 1.$
else
$C(i,n,1) \leftarrow 0.$
Step 8: $\forall i,j; 1 \leq i,j \leq n$, do in parallel
$C(i,1,j) \leftarrow C(i,j,1)$. /* through the horizontal optical link content of $C(i,j,1)$ is moved to $C(i,j,1)$ */
Step 9: $\forall i,k; 1 \leq i,k \leq n$, do in parallel
if $(C(i,1,k) == 0)$
$B(i,1,1) \leftarrow NULL$. /* if the content of C-register of all leaf nodes of G_{i1} is 0 then reset $B(i,1,1)$ to NULL value */
Step 10: $\forall i, 1 \leq i \leq n$, do in parallel
$B(1,1,i) \leftarrow B(i,1,1)$. /* through the vertical optical link content of $B(i,1,1)$ is moved to $B(1,1,i)$ */

Hence the extreme points of the convex hull can be taken from the B-register of all leaf nodes of G_{11} excluding the NULL entries. In order to analyze the time complexity of the above algorithm we also consider the data movements along the both electronic link and optical link. For the complete group broadcast [9] the step 1 needs $4\log n - 2$ electronic moves and 3 optical moves. For the required intra-group group broadcast [8] the step 2 and 5 need $2\log n - 1$ electronic move. For the basic assignment and geometry operations we can assume that the steps 3, 6, 7 and 9 need $O(1)$ time. The required sorting (see appendix) of n points at corresponding $G_{ij}, 1 \leq i,j \leq n$ the step 4 needs $7\log n - 1$ electronic move and 5 optical move. In addition, for the required inter-group data movement the step 8 and 10 need one optical move. Thus overall, we need $O(\log n)$ to compute the convex hull.

THEOREM 3.2. *Algorithm PCH requires $O(\log n)$ time to compute the convex hull of n points.*

The above algorithm can be extended for the smallest enclosing rectangle of n points within $O(\log n)$ time as discussed in [2]. But it would be interesting to devise algorithm for convex hull and smallest enclosing rectangle among n^2 data points on both OMULT and OMTSE optoelectronic computer.

3.2. All-Nearest/All-Farthest Neighbor. All-Nearest(All-Farthest) Neighbor problem can be stated as follows: given a set $S = \{p_1, p_2, ..., p_q\}$ of q points, for each point $p_i \in S$ we wish to determine a point $p_j \in \{S - p_i\}$ such that the Euclidean distance $\|p_i - p_j\|$ is minimum(maximum).

In order to implement All-Nearest Neighbor (All-Farthest Neighbor can be dealt analogously) problem for n^2 points, we assume that each leaf processor $P(i,j,k), 1 \leq i,j,k \leq n$, has four registers A, B, C and D; where as each non-leaf processor $P(i,j,k), 1 \leq i,j \leq n, n+1 \leq k \leq \frac{3n}{2}$, has two registers A and B. Initially, the points $p_{(i-1)+k}$ is stored in the A(i, i, k)of all the diagonal leaf nodes of $G_{ii}, 1 \leq i \leq n$, where as all the D-registers of OMTSE system are set to zero. Set a counter variable c to zero at B-register of each non leaf processor of OMTSE optoelectronic system. Here we describe the algorithm in the following steps

Algorithm $AllNearestNeighbor()$
Input: $\forall i,k, 1 \leq i,k \leq n$
$A(i,i,k) \leftarrow p_{(i-1)+k}$
Output: $\forall i,k, 1 \leq i,k \leq n$
Nearest Neighbor of $p_{(i-1)+k} \leftarrow C(i,i,k)$

Step 1: Perform a column group broadcast [8].
Step 2: While $(c < n)$ do

Step 2.1: $\forall i, j, 1 \leq i, j \leq n$, do in parallel
if $(c == 0)$
Broadcast the content of $A(i, j, i)$ to B-registers of all leaf nodes of G_{ij}.
else
Broadcast the content of $B(i, j, 1 + (j\%n))$ to B-register of all leaf nodes of G_{ij}.
Step 2.2: $\forall i, j, k, 1 \leq i, j, k \leq n$, do in parallel
$D(i, j, k) \leftarrow \|A(i, j, k) - B(i, j, k)\|$
if $(D(i, j, k) == 0)$
$D(i, j, k) \leftarrow \infty$
Step 2.3: $\forall i, j, k, 1 \leq i, j, k \leq n$, do in parallel
Compute the minimum of values stored in each D-register of G_{ij} and store the result in $C(i, j, 1 + ((j + c - 1)\%n))$.
Step 2.4: $\forall i, j, k, 1 \leq i, j, k \leq n$,do in parallel
Perform horizontal optical move on the content of B-registers so that the data from each side move to the corresponding leaf nodes.
Step 2.5: $\forall i, j, k, 1 \leq i, j \leq n, n + 1 \leq k \leq \frac{3n}{2}$, do in parallel
$c = c + 1$.
Step 3: $\forall i, j, k, 1 \leq i, j, k \leq n$ and $j \neq k$, do in parallel
Perform horizontal optical move on the content of C-registers so that the data from each side move to the corresponding leaf nodes.
Step 4: $\forall i, k, 1 \leq i, k \leq n$, do in parallel
Compute the minimum of values stored in each C-register of G_{ki} and store the result in $C(k, i, i)$.
Step 5: $\forall i, k, 1 \leq i, k \leq n$ and $i \neq k$, do in parallel
$C(i, i, k) \leftarrow C(k, i, i) /*$ Vertical Optical Move $*/$

For the required column group broadcast the step 1 requires $2 \log n - 1$ electronic moves and 3 optical moves. The Step 2.1 requires $2 \log n - 1$ electronic moves for intergroup broadcast. To find the minimum in each group G_{ij}, the step 2.3 and step 4 require $O(\log n)$ time. Again the Step 2.4, Step 3 and Step 5 require one optical move each. For the basic increment and distance measure we can assume that the Step 2.2 and Step 2.5 require $O(1)$ time. Since we have n iterations of while loop in Step 2, the overall complexity of the algorithm is $O(n \log n)$ for n^2 points.

3.3. Closest-Pair/Farthest-Pair of Points. This problem can be defined as follows: given a set $S = \{p_1, p_2, ..., p_q\}$ of q points, $\exists \{p_i, p_j\} \in S$ such that euclidean distance $\|p_i - p_j\|$ is minimum(maximum). The closest pair of points can be found by first solving the All-Nearest neighbor problem and then determining the closest pair among the nearest problem of each point. Here we describe the basic algorithm for n^2 points in following steps

Algorithm: $ClosestPairPoints()$

Input: $\forall i, k, 1 \leq i, k \leq n$
$A(i, i, k) \leftarrow p_{(i-1)+k}$
Output: Closest-pair $\leftarrow C(1, 1, 1)$

Step 1: $AllNearestNeighbor()$
Step 2: $\forall i, 1 \leq i \leq n$
Compute the minimum at each G_{ii} and store the result in $C(i, i, 1)$
Step 3: $\forall i, 1 \leq i \leq n$
$C(1, i, i) \leftarrow C(i, i, 1)$
Step 4: $\forall i, 1 \leq i \leq n$
$C(1, i, 1) \leftarrow C(1, i, i)$
Step 5: Compute the minimum at G_{11} and store the result in $C(1, 1, 1)$

The algorithm *ClosestPairPoints* require additional $3\log n - 1$ electronic moves and 2 optical moves which will be subsumed by the $O(n\log n)$ of *AllNearestNeighbor* algorithm.

3.4. ECDF. In ECDF (empirical cumulative distribution function) problem [14], we are given a set $S = \{p_1, p_2, ..., p_q\}$ of q distinct points. For $\{p_i(x_i, y_i), p_j(x_j, y_j)\} \in S$, we will say p_i dominates p_j iff $x_i \geq x_j$ and $y_i \geq y_j$. For all $p_i \in S$, we are going to determine the number points it dominates in set S. In the FIG 3.2 we have illustrated a dominating relationship between three points p_1, p_2, and p_3. In this case, the number of points dominated by p_1, p_2 and p_3, respectively are 1, 1, and 0.

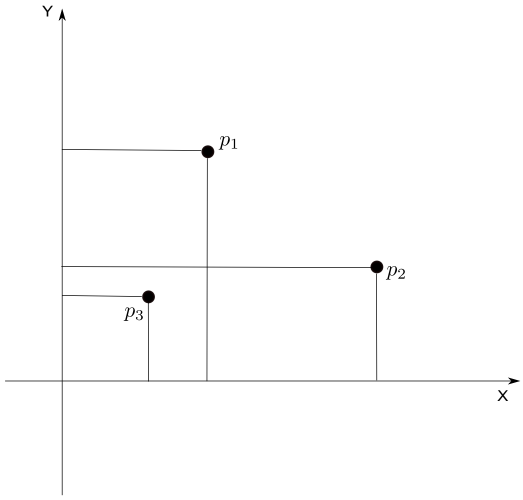

Fig. 3.2: Example of dominating relation

The algorithm to implement ECDF for n^2 is quite similar to the *AllNearestNeighbor* algorithm. Here in order to get dominating value of point $p_{(i-1)+k}$ at corresponding $C(i,i,k)$, in step 2.2 of *AllNearestNeighbor* algorithm the D-register value is set to 1 if B-register point dominates its A-register point. Then in Step 2.3, we need to compute the summation of all D-register value with in that group and store the result in $C(i, j, 1 + ((j + c - 1)\%n))$. After this the D-register is reset to zero and contiue the loop while $c < n$. Further, in Step 4 we need to compute the summation of values stored in each C-register of G_{ki} and store the result in $C(k,i,i)$. Finally in Step 5 we get the dominating value of point $p_{(i-1)+k}$ at corressponding $C(i,i,k)$.

To compute the summation in shuffle exchange network [12] takes the same complexity as to compute the minimum. Thus the time taken to implement ECDF for n^2 points is same as that of *AllNearestNeighbor* algorithm i. e. $O(n\log n)$.

3.5. Two-Set Dominance. The two set dominance problem can stated in this way: We have given two sets $S_1 = \{p_1, p_2, ..., p_p\}$ and $S_2 = \{q_1, q_2, ..., q_q\}$, for each point $p_i \in S_1 (or\ q_j \in S_2)$ we wish to determine the number points in $S_2 (or\ S_1)$ is dominated by $p_i (or\ q_j)$. This is quite similar to the ECDF and can be achieved with $O(n\log n)$ for $\|S_1 + S_2\| = n^2$ points.

3.6. Maximal Points. A point $p \in S$ is maximal iff it dominates all the points in S. This is quite simple and can be achieved by $O(\log n)$ time for $\|S\| = n^3$ points as follows.

Algorithm: *MaximalPoint*
Input: Arbitrarily assign the n^3 points to n^3 leaf processors of OMTSE optoelectronic computer.
Output: Maximal point $\leftarrow A(1,1,1)$
Step 1: $\forall i, j, 1 \leq i, j \leq n$,
Each group G_{ij} determine the maximal point with in that group and store the result in $A(i,j,1)$
Step 2: $\forall i, j, 1 \leq i, j \leq n$,

$A(i, 1, j) \leftarrow A(i, j, 1)$

Step 3: $\forall i, 1 \leq i \leq n$,
Each group G_{i1} determine the maximal point with in that group and store the result in $A(i, 1, 1)$

Step 4: $\forall i, 1 \leq i \leq n$,
$A(1, 1, i) \leftarrow A(i, 1, 1)$

Step 5: The group G_{11} determine the maximal point with in that group and store the result in $A(1, 1, 1)$

For finding the local maximal points with in a group, the Step 1, 3 and 4 requires $O(\log n)$ electronic moves each. Further, for the inter group communication we require one optical move each for the Step 2 and 4. Thus overall we have $O(\log n)$ algorithm with exactly $3 \log n$ electronic moves and 2 optical moves. Now if we define minimal points analogous to maximal points, the above algorithm can be improved slightly to get both the maximal and minimal points out of $n(n-1)^2$ points with $4 \log n + 4$ electronic moves and 3 optical moves as discussed in [10].

4. Conclusion. We have shown that several computational geometry problems can be solved on OMTSE optoelectronic computer efficiently. It would be interesting to devise the discussed algorithms for n^3 number of points on OMTSE and OMULT system.

Acknowledgments. This research is supported by an NSERC Individual Discovery Grant to second author.

REFERENCES

[1] M. R. Feldman, S. C. Esener, C. C. Guest, and S. H. Lee, *Comparison between optical and electrical interconnects based on power and speed considerations*, Appl. Opt., 27 (1988), pp. 1742–1751.

[2] R. Islam, N. Afroz, S. Bandyopadhyay, and B. P. Sinha, *Computational geometry on optical multi-trees (OMULT) computer system*, in CCCG, 2005, pp. 150–154.

[3] P. K. Jana, *Improved parallel prefix computation on optical multi-trees*, in India Annual Conference, 2004. Proceedings of the IEEE INDICON 2004. First, 20-22 2004, pp. 414 – 418.

[4] P. K. Jana, *Polynomial interpolation and polynomial root finding on otis-mesh*, Parallel Computing, 32 (2006), pp. 301–312.

[5] P. K. Jana and K. Sinha, *Permutation algorithms on optical multi-trees*, Comput. Math. Appl., 56 (2008), pp. 2656–2665.

[6] A. V. Krishnamoorthy, P. J. Marchand, F. E. Kiamilev, and S. C. Esener, *Grain-size considerations for optoelectronic multistage interconnection networks*, Appl. Opt., 31 (1992), pp. 5480–5507.

[7] D. K. Mallick and P. K. Jana, *Parallel prefix on mesh of trees and otis mesh of trees*, in PDPTA, H. R. Arabnia and Y. Mun, eds., CSREA Press, 2008, pp. 359–.

[8] S. C. Panigrahi, S. Paul, and G. Sahoo, *OMTSE - an optical interconnection system for parallel computing*, in Advanced Computing and Communications, 2006. ADCOM 2006. International Conference on, 20-23 Dec 2006, pp. 626 –627.

[9] ———, *Parallel prefix computation, sorting and reduction operation on OMTSE architecture*, in ICACC 2007 International Conference, 9-10 Feb 2007, pp. 616 –622.

[10] S. C. Panigrahi and G. Sahoo, *An MIMD algorithm for finding maximum and minimum on OMTSE architecture*, Scalable Computing: Practice and Experience, 9 (2008), pp. 69–75.

[11] F. P. Preparata and M. I. Shamos, *Computational geometry: an introduction*, Springer-Verlag, New York, 1985.

[12] M. J. Quinn, *Parallel computing (2nd ed.): theory and practice*, McGraw-Hill, Inc., New York, NY, USA, 1994.

[13] B. P. Sinha and S. Bandyopadhyay, *OMULT: An optical interconnection system for parallel computing*, in Euro-Par, M. Danelutto, M. Vanneschi, and D. Laforenza, eds., vol. 3149 of Lecture Notes in Computer Science, Springer, 2004, pp. 856–863.

[14] C.-F. Wang and S. Sahni, *Computational geometry on the OTIS-mesh optoelectronic computer*, in ICPP, IEEE Computer Society, 2002, pp. 501–.

[15] F. Zane, P. Marchand, R. Paturi, and S. Esener, *Scalable network architectures using the optical transpose interconnection system (OTIS)*, J. Parallel Distrib. Comput., 60 (2000), pp. 521–538.

Appendix

For the sake of explaining the basic idea, in this appendix we discuss how the sorting of n distinct elements can be performed in OMTSE optoelectronic computer. Let's assume that each processor $P(i, j, k), 1 \leq i, j, k \leq n$, has two registers $R1(i, j, k)$ and $R2(i, j, k)$. Initially we have n distinct elements $\{a_1, a_2, a_3, ..., a_n\}$ stored in R1-register of n leaf nodes of G_{11}. We can sort these elements by finding rank of each element in the list. Thus the objective of the algorithm is to place the element of rank $r, 1 \leq r \leq n$ in the processor $P(1, 1, r)$.

Algorithm *Sort*()

Step 1: Perform a column broadcast [8] so that the list of elements stored in the leaf nodes of G_{11} broad casted to the corresponding leaf nodes all $G_{i1}, 1 \leq n$.
Step 2: $\forall i, 1 \leq i \leq n$, do in parallel
Broadcast the element a_i to R2-register of all leaf nodes of G_{i1}. Set a Flag as 1 if a_i greater than other element in R1-register of same leaf node. Otherwise set Flag as zero. The value of the Flag variable can be kept in R2-register which may overwrite previous entries.
Step 3: $\forall i, 1 \leq i \leq n$, do in parallel
Compute the summation of all Flag values stored on each leaf nodes of G_{i1}, which is the rank(r) of the element a_i in the given list.
Remark: As a result of summation in the shuffle exchange network [12] the rank value will reflect in all nodes of shuffle exchange layer of G_{i1}.
Step 4: $\forall i, 1 \leq n$, do in parallel
if the rank of a_i is r then the element a_i is moved to $R1(i, 1, r)$.
Step 5: $\forall i, 1 \leq n$, do in parallel
$R1(r, 1, i) \leftarrow R1(i, 1, r)$ /* Vertical optical link */
Step 6: $\forall i, 1 \leq n$, do in parallel
$R1(r, 1, 1) \leftarrow R1(r, 1, i)$
Step 7: $\forall r, 1 \leq r \leq n$, do in parallel
$R1(1, 1, r) \leftarrow R1(1, 1, r)$ /* Vertical optical link */

For the complexity analysis of the above algorithm we also consider the data movement along the electronic and optical link. The above algorithm needs $(7 \log n - 1)$ communication steps along electronic links and 5 communication steps along optical links [8] giving overall $O(\log n)$ time algorithm. The idea can be extended to sort n^2 data values in $O(n \log n)$ time but this is beyond the scope of this paper.

Edited by: Dana Petcu and Marcin Paprzycki
Received: May 1, 2011
Accepted: May 31, 2011