I. J. Computer Network and Information Security, 2022, 1, 69-80
Published Online February 2022 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2022.01.06

Optimized Parallel Counting Sort Algorithm for Distinct Numeric Values on Biswapped Hyper Hexa-Cell Optoelectronic Network

Ashish Gupta
Institute of Engineering & Technology, Dr. Rammanohar Lohia Avadh University, Ayodhya, India
E-mail: ashish.parj@gmail.com

Received: 12 July 2021; Revised: 21 September 2021; Accepted: 03 November 2021; Published: 08 February 2022

Abstract: The optoelectronic Biswapped-Hyper Hexa-Cell is a recently reported recursive and a symmetrical architecture of Biswapped Family. This symmetrical network has claimed and proved to be advantageous in terms of network diameter, bisection width, minimum node-degree and network cost compared to its counterpart architecture of OTIS family named 'OTIS Hyper Hexa-Cell' and traditional grid-based architecture of Biswapped family named 'Biswapped-Mesh'. In this paper, we present a novel and efficient parallel algorithm for counting sort for sorting distinct numeric values on d_h-dimensional Biswapped-Hyper Hexa-Cell optoelectronic network. The parallel algorithm demands $10d_h + 12 + \log_2 S_A$ electronic and 10 optical moves, where S_A is the size of count array: $A_{cip}[S_A]$, and S_A equals to maximal minus minimal numeric value plus one. On the basis of analysis, it is concluded that proposed algorithm delivers better performance since speedup and efficiency improved for worst case scenario (difference between maximal and minimal data values becomes larger) with the increase of only few communication moves required for sorting.

Index Terms: Optoelectronic, Biswapped, Parallel Algorithm, Hyper Hexa-Cell, Counting Sort.

1. Introduction

The Biswapped network [3] is basically a family of optoelectronic 2-level hybrid structures that take any graph as modules and connect them in complete bi-partite pattern. Actually, Biswapped network gained much attention of researchers due to its symmetrical nature and large network size with node-degree identical to its counter-part OTIS network [2]. It may be noted that, both OTIS and Biswapped network follow the optoelectronic pattern of connectivity among processors, where optical (inter-cluster) connection connect the processors that lies in different clusters and electronic connections connect the processors that lies in same cluster. Unlike OTIS network, entire Biswapped network composed of two parts namely *part-0* and *part-1*. Generally, the total clusters in each part of all architectures of Biswapped family always equals to the size of each identical cluster. And, in both part, clusters connect to clusters of opposite part by complete bipartite connectivity. However, the condition also arises when in each part, total clusters in both parts of Biswapped network may not equals to cluster size. It may be noted that, power consumption and bandwidth utilization become optimized for the former case when total clusters in each part of Biswapped network equal to the cluster size.

In 2018, Gupta and Sarkar [9] suggested a recursive and node-symmetrical architecture of biswapped family named 'Biswapped-Hyper Hexa-Cell'. The Biswapped-Hyper Hexa-Cell has demonstrated some major benefits, such as 'small diameter', 'high bisection width' and 'high minimum node-degree', when compared to its counterpart architecture of OTIS family named 'OTIS Hyper Hexa-Cell' [4]. In terms of comparison performed with grid-based architecture of biswapped family named 'Biswapped-Mesh' [5], the Biswapped-Hyper Hexa-Cell demonstrated important benefits, such as 'small diameter' and 'high minimum node-degree'. Recently, Gupta and Sarkar [10] proposed another grid-based node-symmetrical variation of Biswapped family named 'Biswapped-Torus' and showed comparison with Biswapped-Mesh and Biswapped-Hyper Hexa-Cell. Based on the comparison with Biswapped-Mesh, it has been analyzed that Biswapped-Torus can perform diameter-based numerical problems faster. In addition, it also has architectural benefit of node-symmetry. Compared to the symmetrical counter-part of biswapped family named Biswapped Hyper Hexa-Cell, the Biswapped-Torus can deliver competitive performance in-terms of network diameter only up to the network size of 1250. Concerning network cost, Biswapped-Torus is advantageous. Gupta and Sarkar [11] also suggested a fast and economical variation of Biswapped family named BSN MOT recently and compared its performance with counterpart OTIS Mesh of Trees, Biswapped-Hyper Hexa-Cell and Biswapped-Mesh networks. In the present article, optimal parallel algorithm is proposed for sorting distinct numerical values using counting sort

methodology on Biswapped-Hyper Hexa-Cell optoelectronic network for the optimized case, when total clusters in each part equal to the cluster size.

Truly saying, sorting of numeric values is necessary for efficient lookup and search. In fact, many real time applications are based on the efficient sorting, such as in real time operating systems, scheduling of tasks according to their relevance (based on some properties) for execution. The adopted method for sorting in this article is counting sort, and computational model is Biswapped-Hyper Hexa-Cell. Counting sort is an optimal choice for sorting, when the difference between the size of the input array (contains data elements that need to be sorted) or output array (contains sorted data elements), and the count array is minimal. The primary advantage of Counting sort is its ability to performs sorting in linear time in worst case compared to counter-part sorting algorithms such as quick sort which require O(nlogn) time. The disadvantage of counting sort is the requirment of large auxiliary space for arrays. In fact, the performance of counting sort diminishes as difference between the input or output array, and count array getting large. It may be noted that, in the proposed algorithm, the computation of prefix sum on count array is suggested to map in parallel in O(logn) time compared to the linear time O(n). The performance evaluation is shown to be performed by analyzing and comparing communication moves, speedup and efficiency. In fact, based on the analysis, proposed approach claims to deliver optimal performance for mapping counting sort on Biswapped-Hyper Hexa-Cell, since speedup and efficiency improved for worst case scenario (difference between maximal and minimal data values becomes larger) with the increase of only few communication moves required for sorting.

As of now, a brief introduction concerning the present article is shown. Rest of the article is structured as follows. In Section 2, related literatures to the present work is presented. In Section 3, computational model (Biswapped Hyper Hexa –Cell) is illustrated in detail including intra-cluster connectivity for cluster (Hyper Hexa-Cell), and inter-cluster connectivity for the entire Biswapped Hyper Hexa-Cell network. In Section 4, methodology related to the counting sort is presented in detail. In Section 5, proposed parallel algorithm is presented for mapping counting sort on Biswaped Hyper Hexa-Cell Optoelectronic network for sorting distinct numeric values. Further, Section 6 covers the relevant portion of results & discussion includes evaluation of speed up and efficiency, and comparison of communication moves, speed-up and efficiency for different size of count array for One, Two and Three-Dimensional Biswapped Hyper Hexa-Cell interconnection network. Finally, in Section 7, the work is concluded.

2. Related Works

Concerning many real time applications, efficient sorting of numeric values is desirable for look up, search, scheduling of tasks, etc. Many literatures have been proposed in last few years on optimal parallel mapping of different sorting methods on optoelectronic architectures. Most recent, Al-Adwan *et. al* [20] suggested parallel quicksort algorithm on OTIS Hyper Hexa-Cell network and measured performance analytically and by simulation in terms of run-time, speed-up and efficiency. In fact, the simulated outcomes of better performance in-terms of speedup and efficiency matched with the analytical outcomes.

Besides sorting, in last few years, relevant literatures have been reported regarding the proposal of parallel algorithm for optimal mapping of important numerical problems on optoelectronic OTIS architectures. *For Instance*, in 2017, Datta, De and Sinha [12] suggested parallel prefix algorithm on optoelectronic Multi-Mesh network and claimed to accomplish parallel execution for the same in only 7n electronic communication moves compared to the 13n + 5 electronic communication moves on the optoelectronic Extended Multi-Mesh network [13]. Al-Adwan *et al*. [18] proposed parallel Repetitive 2-Opt (PRTO) algorithm for solving symmetric Travelling salesman problem on optoelectronic OTIS Mesh and OTIS Hypercube networks. Besides, Parallel algorithm for heuristics local search using similar Parallel Repetitive 2-Opt (PRTO) is also suggested by Al-Adwan *et. al* [19] to solve symmetric Travelling salesman problem on members of OTIS family named OTIS mesh of trees and OTIS Hyper Hexa-Cell networks.

Compared to OTIS framework, symmetrical property of optoelectronic Biswapped framework has attracted researchers to study and explore different dimensions of this network. *For instance*, Zhao et al. [6] proposed efficient GPM algorithm for load balancing and claimed its effectiveness by analytical model. Ling and Chen [7] proposed algorithm for node-to-set disjoint paths problem in an arbitrary biswapped network with a connected cluster network. Chen and Ling [8] investigated the problem of embedding cycles of various lengths in a biswapped network based on the cycle of length (l) ($l \geq 3$) of cluster network. In addition, on member architectures of Biswapped family, optimal parallel mapping of numerical problems has also been reported in recent years. *For instance*, Gupta and Sarkar presented novel and efficient algorithm for computing prefix sum [14], LaGrange's interpolation [16] and Special Cases of binomial series [17] on traditional grid-based member of optoelectronic biswapped family named Biswapped-Mesh network. Besides Biswapped-mesh, optimal parallel mapping of prefix sum problem has also been reported on recursive and symmetrical member of biswapped family named Biswapped-Hyper Hexa-Cell [15]. Lytvynenko *et al.* [20] performed comparative analysis of self-organizing algorithms for forecasting economic parameters. Dash *et al.* [21] evaluate the performance on Open MP parallel platform on the basis of problem size. Zeng *et al.* [22] proposed parallel algorithms for Freezing Problems during Cryosurgery. In this article, an important problem of sorting numeric values using counting sort methodology is presented on recursive and symmetrical variation of Biswapped framework named 'Biswapped-Hyper Hexa-Cell'.

3. Computational Model

Basically, the Biswapped-Hyper Hexa-Cell composed of two parts: *part-0* and *part-1*, where each part may either contain N or $\frac{N}{2}$ clusters [9]. For bandwidth and power optimization, total clusters in each part should equal to the cluster size (N) that results in total N^2 processors in each part, and $2N^2$ processors in the entire Biswapped-Hyper Hexa-Cell network.

Intra-cluster (electronic) connections

The labelling of each identical cluster (d_h-Dimensional Hyper Hexa-Cell) require two parameters: subcluster-number and processor-number for intra-cluster communication. The subcluster-number identifies each subcluster (One-Dimensional Hyper Hexa-Cell), and processor-number identifies the processor within each subcluster. The labelling of subcluster-number requires d_h-1 bits, and labelling of processor-number require three bits. For instance, Two-Dimensional Hyper Hexa-Cell (cluster) composed of two One-Dimensional Hyper Hexa-Cell networks (subcluster). Therefore, only one bit (0 and 1) require for identify each subcluster. For labelling of processor-number, rightmost two bits are labelled as 00, 01 and 10 both in upper and lower triangle that initiates from the top most node, and proceed towards the right node from left node. The labelling of left most bit is 0 for the upper triangle and 1 for the lower triangle. For better understanding refer to Figures 1 and 2 that shows labelled network architectures of One and Two-Dimensional Hyper Hexa-Cell networks respectively.

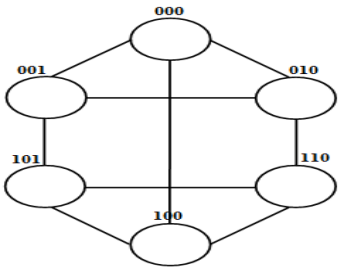

Fig.1. One-Dimensional Hyper Hexa-Cell

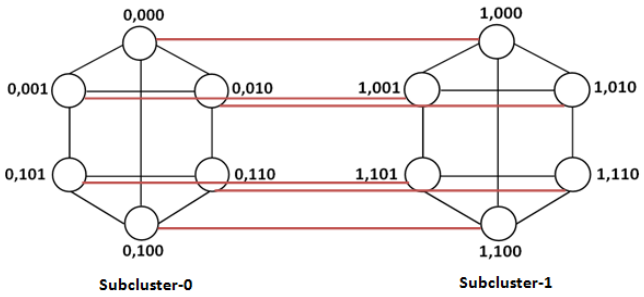

Fig.2. Two-Dimensional Hyper Hexa-Cell

Inter-cluster (optical) connections

The labelling of processors in Biswapped-Hyper Hexa-Cell required three parameters: c**,** i and p which represent cluster-number**,** processor-number**,** and part-number respectively for inter-cluster communication.

$$\forall c \text{ and } \forall p, \text{ Processor: } P(c, i, 0) \text{ is connected to the processor: } P(i, c, 1) \text{ and vice versa.}$$

For better understanding, refer to Figure 3, which shows labelled network architecture of One-Dimensional Biswapped-Hyper Hexa-Cell network.

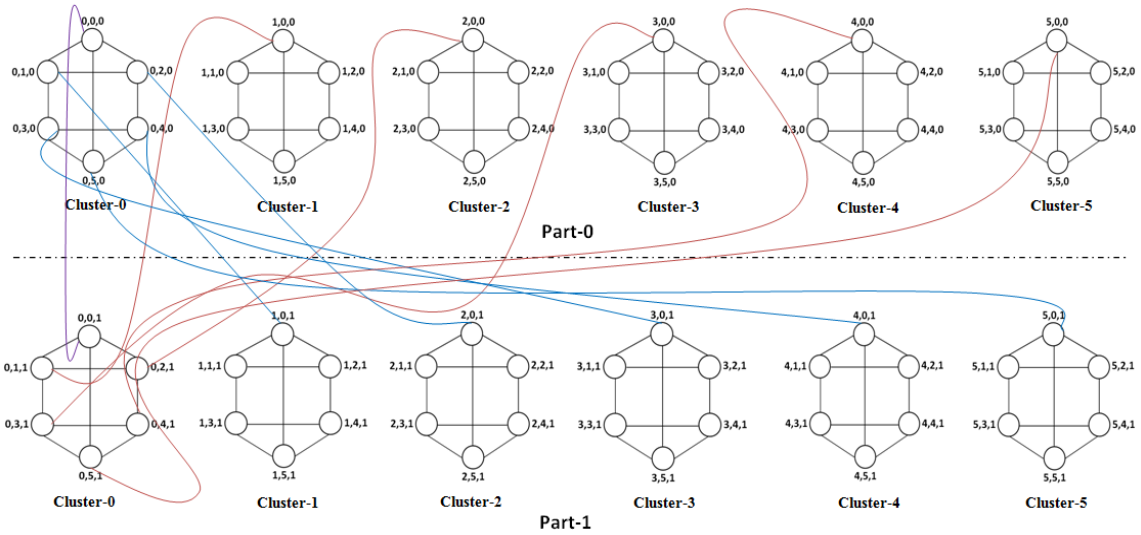

Fig.3. One-Dimensional Biswapped-Hyper Hexa-Cell

4. Methodology

In our adopted parallel approach for mapping counting sort on optimized case of d_h-Dimensional Biswapped-Hyper Hexa-Cell network, initially the primary aim is to find the minimal and the maximal data elements among all data elements that need to be sorted. This process is required to develop count array: $A_{cip}[S_A]$ (as displayed). The size (S_A) of this array: $A_{cip}[S_A]$ ranges from minimal to maximal data values. The labelling in the base of this array: $A_{cip}[]$ identifies the processor which hold this array, where labels: c, i and p represents cluster-number, processor-number and part-number respectively. The array: $A_{cip}[]$ contains value: 1, if index of this array is identical to the data element otherwise 0.

Thereafter, the prefix operation is applied to the data elements contained in count array: $A_{cip}[]$. This computed prefix values finally decide the index in array: $B_{cip}[]$, where the data element finally stored in sorted pattern. For better understanding, count array: A_{cip}[Max-Min+1] and output array: B_{cip} [$2N^2$] is displayed below, where Max and Min represents the maximal and the minimal data values in the entire architecture of Biswapped-Hyper Hexa-Cell network. It may be noted that, in our proposed approach, the size of sorted array: $B_{cip}[]$ will be equal to the size of computational network ($2N^2$), since we assume each processor holds a single data element. Note that, size of the cluster of Biswapped Hyper Hexa-Cell network can be computed by formula: $N = 6\times 2^{d_h-1}$.

Array: A_{cip} [Max-Min+1]-

Min	Min +1	Min +2	Min +3	…	Max -1	Max

Array: B_{cip} [$2N^2$] (sorted array)-

1	2	3	4	5	6	…	$2N^2$

Assume, data elements: $X_0, X_1, \ldots, X_{Z-1}$ are need to be sorted, where $X_0 = 12$, $X_1 = 4$, $X_2 = 15$, $X_3 = 6$, $X_4 = 1$, $X_5 = 14$ and $X_6 = 9$.

Array: A_{cip} [15]

1	2	3	4	…	14	15

Array B_{cip} [7]-

1	2	3	4	5	6	7
1	4	6	9	12	14	15

5. Parallel Algorithm

In this section, parallel algorithm is presented for mapping counting sort for sorting distinct numeric values on d_h-Dimensional Biswapped Hyper Hexa-Cell optoelectronic network. In Section 5.1, the primary parallel algorithm of counting sort on Biswapped Hyper Hexa-Cell is presented. Further, parallel algorithms for subroutines needed to execute Algorithm 5.1 are displayed in subsequent subsections. *For example*, in Algorithms 5.2 and 5.3, parallel mapping for finding maximal and minimal data elements on the Biswapped Hyper Hexa-Cell network is presented. In Algorithms 5.4 and 5.5, parallel mapping of finding maximal and minimal data elements in a cluster (Hyper Hexa-Cell) are presented that is basically required for the execution of Algorithms 5.2 and 5.3. In Algorithms 5.6 and 5.7, parallel algorithms for broadcast data element and broadcast of array of registers are presented. In Algorithm 5.8, parallel algorithm for performing data sum on array of registers is shown.

Algorithm 5.1. Algorithm of Counting Sort on Biswapped Hyper Hexa-Cell

Assuming, data elements: $X_0, X_1, X_2, \ldots, X_{2N^2-1}$ need to be sorted
Registers: W(c, i, p) and Y(c, i, p)
Array of Registers: $A_{cip}[S_A]$ and $B_{cip}[S_B]$,
S_A - size of array: $A_{cip}[]$, S_B -size of array: array: $B_{cip}[]$.

Data Initialization:

∀c, ∀i and ∀p,
$W(c, i, p) \longleftarrow X_{cN+i+pN^2}$.
$Y(c, i, p) \longleftarrow X_{cN+i+pN^2}$.
$A_{cip}[index_{array\text{-}A}] \longleftarrow 0$.
$B_{cip}[index_{array\text{-}B}] \longleftarrow 0$.

Step 1. Perform Steps 1.1 and 1.2 for finding minimal and maximal data elements in parallel.

1.1. *∀c, ∀i and ∀p,* Call Find_Biswapped_Max_W(0, 0, 0); *//Algorithm 5.2//*
1.2. *∀c, ∀i and ∀p,* Call Find_Biswapped_Min_Y(0, 0, 0); */Algorithm 5.3//*

Step 2. Perform Steps 2.1 and 2.2 for broadcast of minimal and maximal data elements in parallel.

2.1. Call Broadcast_W(0, 0, 0); *// Algorithm 5.6,* Assuming W(0, 0, 0) is source register//
2.2. Call Broadcast_Y(0, 0, 0); *// Algorithm 5.6,* **A**ssuming Y(0, 0, 0) is source register//

Step 3. *∀c, ∀i and ∀p,* $S_A = Y(0,0,0) - W(0,0,0) + 1$, $Y(0, 0, 0) <= index_{array\text{-}A} <= W(0,0,0)$.

$A_{cip}[index_{array\text{-}A}] \longleftarrow 1$, If $index_{array\text{-}A} == X_{cN+i+pN^2}$.

Step 4. *∀c, ∀i and ∀p,* Call Data_Sum_$A_{cip}[S_A]$; //Algorithm 5.8//
Step 5. Perform prefix sum on 2D array: $A_{000}[S_A]$. [1]
Step 6. Call Broadcast_$Array_{000}[S_A]$; *//Algorithm 5.7//*
Step 7. *∀c, ∀i and ∀p,* $S_B = 2N^2$, $1<=index_{array\text{-}B}<=2N^2$.

$B_{cip}[A_{cip}[Index_{array\text{-}A}]] \longleftarrow X_{cN+i+pN^2}$, If, $Index_{array\text{-}A}==X_{cN+i+pN^2}$.

Step 8. Call Data_Sum_$B_{cip}[S_B]$. *//Algorithm 5.8//*

//Final result obtained in 2D array: $B_{000}[S_B]$.//

Time Complexity (Algorithm 5.1)

The initial step (Step 1) demands $2d_h + 4$ electronic moves and 2 optical moves for finding maximal and minimal data elements in Biswapped-Hyper Hexa-Cell network. Further, Step 2 demands $2d_h + 2$ electronic moves and 2 optical moves for data broadcast in the entire Biswapped Hyper Hexa-Cell network. Thereafter a constant step required in Step 3. Further, Step 4 require $2d_h + 2$ electronic and 2 optical moves for performing data sum on the entire Biswapped-Hyper Hexa-Cell network. Step 5 demands $\log_2 S_A$ electronic moves for performing prefix sum on count array: $A_{cip}[S_A]$.

Again, Step 6 needs $2d_h + 2$ electronics and 2 optical moves data broadcast in Biswapped-Hyper Hexa-Cell network. Step 7 demands a constant step. Again data sum will be performed on Biswapped-Hyper Hexa-Cell in Step 8 that require total $2d_h + 2$ electronic moves and 2 optical moves. Therefore, total $10d_h + 12 + \log_2 S_A$ electronic and 10 optical moves, where, S_A is the size of array: $A_{cip}[S_A]$.

Algorithm 5.2: *Algorithm of Subroutine for finding Maximal data element in Biswapped Hyper hexa-Cell.*

Subroutine: Find_Biswapped_Max_P(0, 0, 0)
Assume, processor: P(0, 0, 0) to hold the maximal and minimal data value */
{
Step 1. ∀c, ∀i and ∀p, Call Cluster_Max_P(c, i, p) //Algorithm 5.4//
Step 2. ∀c, Perform Steps 2.1 and 2.2 in parallel.

2.1. P(0, c, 1) ⟵ P(c, 0, 0).
2.2. P(0, c, 0) ⟵ P(c, 0, 1).

Step 3. ∀i and ∀p, Call Cluster_Max_P(0, i, p) //Algorithm 5.4//
Step 4. Do P(0, 0, 0) ⟵ P(0, 0, 1), if, P(0, 0, 1) > P(0, 0, 0).
}

Algorithm 5.3: *Algorithm of Subroutine for finding Minimal data element in Biswapped Hyper hexa-Cell.*

Subroutine: Find_Biswapped_Min_P(0, 0, 0)
{
Step 1. ∀c, ∀i and ∀p, Call Cluster_Min_P(c, i, p). //Algorithm 5.5//
Step 2. ∀c, Perform Steps 2.1 and 2.2 in parallel.

2.1. P(0, c, 1) ⟵ P(c, 0, 0).
2.2. P(0, c, 0) ⟵ P(c, 0, 1).

Step 3. ∀i and ∀p, Call Cluster_Min_P(0, i, p) //Algorithm 5.5//
Step 4. Do P(0, 0, 0) ⟵ P(0, 0, 1), if, P(0, 0, 1) < P(0, 0, 0)
}

Time Complexity (Algorithm 5.2 & 5.3):

In Algorithms-5.2 and 5.3, Step 1 demands $d_h + 2$ electronic moves for finding minimal and maximal data value. Step 2 demands 1 optical move. Again, Step 3 demands $d_h + 2$ electronic moves for finding minimal and maximal data value. Step 4 demands 1 optical move. Therefore, total $2d_h + 4$ electronic moves and 2 optical moves is required to find maximal and minimal data element in Biswapped-Hyper Hexa-Cell.

Algorithm 5.4: Algorithm of finding maximal data element in a cluster, considering processor: P(c, 0, p) to hold the maximal data value.

Cluster_Max_P(c, i, p)
{
Step 1. ∀s, ∀c and ∀p, Perform Steps 1.1 and 1.2 in parallel. // s represent subcluster//

Step 1.1 ∀s, ∀c and ∀p, Do steps 1.1.1 and 1.1.2 in serial.

Step 1.1.1. P(c, 6s , p) ⟵ P(c, 6s + 1, p), if, P(c, 6s +1, p) > P(c, 6s, p).
Step 1.1.2. P(c, 6s, p) ⟵ P(c, 6s + 2, p), if, P(c, 6s + 2, p) > P(c, 6s, p).

Step 1.2 Do steps 1.2.1 and 1.2.2 in serial.

Step 1.2.1. P(c, 6s + 5, p) ⟵ P(c, 6s + 3, p), if, P(c, 6s + 3, p) > P(c, 6s + 5, p).
Step 1.2.2. P(c, 6s + 5, p) ⟵ P(c, 6s + 4, p), if, P(c, 6s + 4, p) > P(c, 6s + 5, p).

In Figure 5 and Figure 6, communication performed in substeps-1.1.1 and 1.2.1 are shown by label-1 and communication performed in substeps-1.1.2 and 1.2.2 are shown by label-2.

Step 2. ∀s, ∀c and ∀p, do in parallel.

$P(c, 6s, p) \longleftarrow P(c, 6s + 5, p)$, if, $P(c, 6s + 5, p) > P(c, 6s, p)$.

In Figure 5 and Figure 6, communication performed in step 2 is shown by label-3.

Step 3. ∀c, Compare obtained P(c, 0, p), P(c, 6, p), … $P(c, 6\times(2^{d_h-1}-1), p)$ and store outcome at processor: P(c, 0, p)
}

In Figure 6, communication performed step 3 is displayed by label-4.

Algorithm 5.5: Algorithm of finding minimal data element in a cluster, considering processor: P(c, 0, p) to hold the minimal data value.

Cluster_Min_P(c, i, p)
{
Step 1. ∀s, ∀c and ∀p, Perform Steps 1.1 and 1.2 in parallel. //s represent subcluster//

Step 1.1. Do steps 1.1.1 and 1.1.2 in serial.

Step 1.1.1. $P(c, 6s , p) \longleftarrow P(c, 6s + 1, p)$, if, $P(c, 6s +1, p) < P(c, 6s, p)$.
Step 1.1.2. $P(c, 6s, p) \longleftarrow P(c, 6s + 2, p)$, if, $P(c, 6s + 2, p) < P(c, 6s, p)$.

Step 1.2. Do steps 1.2.1 and 1.2.2 in serial.

Step 1.2.1. $P(c, 6s + 5, p) \longleftarrow P(c, 6s + 3, p)$, if, $P(c, 6s + 3, p) < P(s, 6s + 5, p)$.
Step 1.2.2. $P(c, 6s + 5, p) \longleftarrow P(c, 6s + 4, p)$, if, $P(c, 6s + 4, p) < P(s, 6s + 5, p)$.

In Figure 5 and Figure 6, communication performed in substeps-1.1.1 and 1.2.1 are shown by label-1 and communication performed in substeps-1.1.2 and 1.2.2 are shown by label-2.

Step 2. ∀s, ∀c and ∀p, do in parallel.

$P(c, 6s, p) \longleftarrow P(c, 6s + 5, p)$, if, $P(c, 6s + 5, p) < P(c, 6s, p)$.

In Figure 5 and Figure 6, communication performed in step-2 is shown by label-3.

Step 3. ∀c, Compare obtained P(c, 0, p), P(c, 6, p) ,…, $P(c, 6\times(2^{d_h-1}-1), p)$, and store outcome at processor: P(c, 0, p)

In Figure 6, communication performed step-3 is displayed by label-4.

Time Complexity (Algorithm 5.4 & 5.5):

In Algorithms-5.4 and 5.5, Step1 demands 2 electronic moves. Step 2 demands one electronic move. Step 3 demands d_h -1 electronic move for comparing top most node: P(c, 0, p) of upper triangle of each subcluster and collecting minimal and maximal data element at processor: P(c, 0, p). Therefore, total $d_h + 2$ electronic moves to find maximal and minimal data element in each cluster.

Algorithm 5.6: Algorithm of broadcast of data element stored in processor register.

Broadcast_P(c, i, p)
{
Step 1. Perform cluster broadcast from source processor: P(c, i, p).
Step 2. ∀c and ∀i, perform optical moves in parallel.

$P(i, c, p') \longleftarrow P(c, i, p)$, p' is opposite part.

Step 3. ∀c, perform cluster broadcast of data element stored in processor: P(i, c, p').

Step 4. ∀c and ∀i, perform optical moves in parallel.

$P(i, c, p) \longleftarrow P(c, i, p')$.

}

Algorithm 5.7: Algorithm of broadcast of array of registers.

Broadcast_Array$_{cip}$[S]

{

Step 1. Perform cluster broadcast of $Array_{cip}[S]$.

Step 2. ∀c and ∀i, perform optical moves in parallel.

$Array_{icp'}[S] \longleftarrow Array_{cip}[S]$. //p' is opposite part//

Step 3. ∀c, perform cluster broadcast of $Array_{icp'}[S]$ stored in processor: P(i, c, p').

Step 4. ∀c and ∀i, perform optical moves in parallel.

$Array_{icp}[S] \longleftarrow Array_{cip'}[S]$.

}

Time Complexity (Algorithm 5.6 & 5.7)

In Algorithms 5.6 and 5.7, Step 1 demands $d_h + 1$ electronic moves. Step 2 needs one optical move. Step 3 demands $d_h + 1$ electronic moves. Step 4 needs one optical move. Therefore, total $2d_h + 2$ electronics and 2 optical moves.

Algorithm 5.8: Algorithm for performing sum of array of registers.

Data_Sum _$Array_{cip}$ [S]

{

Step 1. ∀c, ∀i and ∀p, perform data element sum of array of each subcluster and store partial results at $Array_{c,0,p}(S)$, $Array_{c,6,p}(S)$,…, $Array_{c,\ 6\times(2^{d_h-1}-1),p}(S)$.

Step 2. ∀c, and ∀p, combine the partial results as data sum obtained at array: $Array_{c,0,p}(S)$, $Array_{c,6,\ p}(S)$,…, $Array_{c,\ 6\times(2^{d_h-1}-1),p}(S)$ and store final result at processor's array: $A_{c0p}(S)$.

Step 3. ∀c, Perform steps 3.1 and 3.2 in parallel.

3.1. $Array_{0c1}[S] \longleftarrow Array_{c00}[S]$.

3.2. $Array_{0c0}[S] \longleftarrow Array_{c01}[S]$.

Step 4. ∀i and ∀p, perform data sum of array elements of each subcluster of cluster-0, and store partial results at $Array_{0,\ 0,\ p}[S]$, $Array_{0,\ 6,\ p}[S]$,…, $Array_{0,\ 6\times(2^{d_h-1}-1),p}[S]$.

Step 5. ∀p, Combine partial results as data sum obtained at array: $Array_{0,\ 0,\ p}(S)$, $Array_{0,\ 6,\ p}(S)$,…, $Array_{0,\ 6\times(2^{d_h-1}-1),\ p}(S)$ and store final result at processor's array: $A_{00p}(S)$.

Step 6. $Array_{000}[S] \longleftarrow Array_{001}[S]$.

Time Complexity (Algorithm 5.8):

In Algorithm 5.8, Step 1 demands 2 electronic steps for performing data sum on each subcluster (One-Dimensional Hyper Hexa-Cell). Step 2 demands $d_h - 1$ electronic steps for combine partial results obtained at each subcluster. Step 3 demands one optical move. Again, Step 4 demands 2 electronic steps for data sum in each subcluster. Step 5 demands $d_h - 1$ electronic moves for combine partial results. Step 6 needs one optical move. Therefore, total $2d_h + 2$ electronic moves and 2 optical moves are required to perform parallel data sum on d_h –Dimensional Biswapped-Hyper Hexa-Cell network.

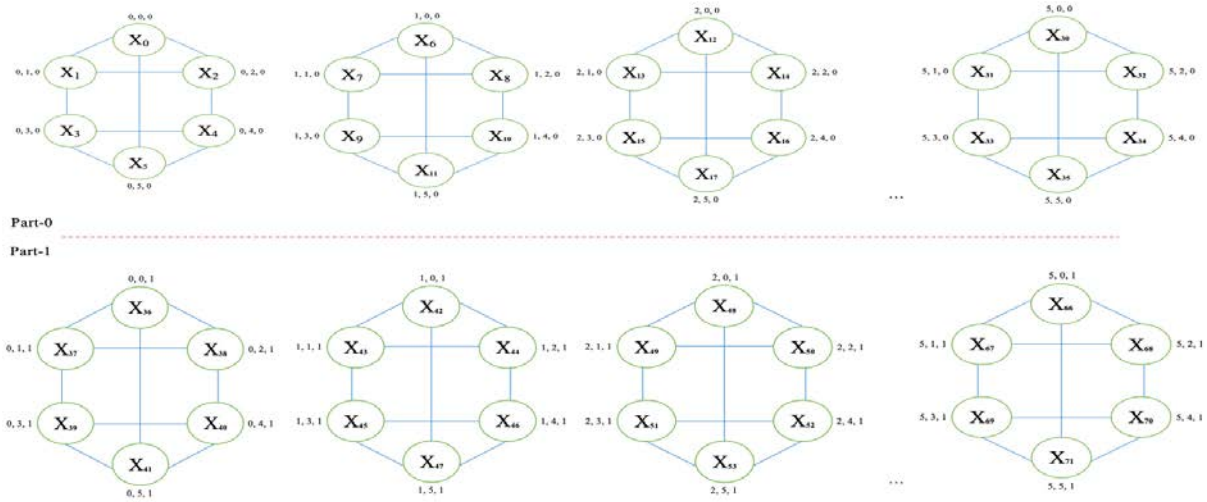

Fig.4. Data Initialization (One-Dimensional Biswapped-Hyper Hexa-Cell)

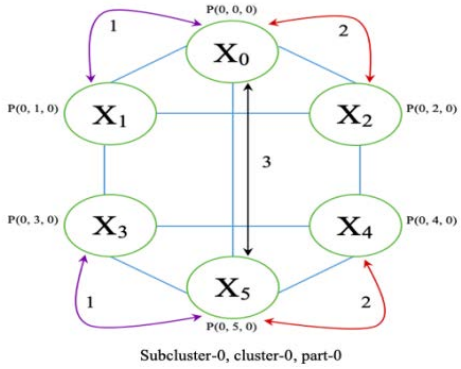

Fig.5. Communication Steps for Computing Min and Max (1-Dimensional Hyper Hexa-Cell)

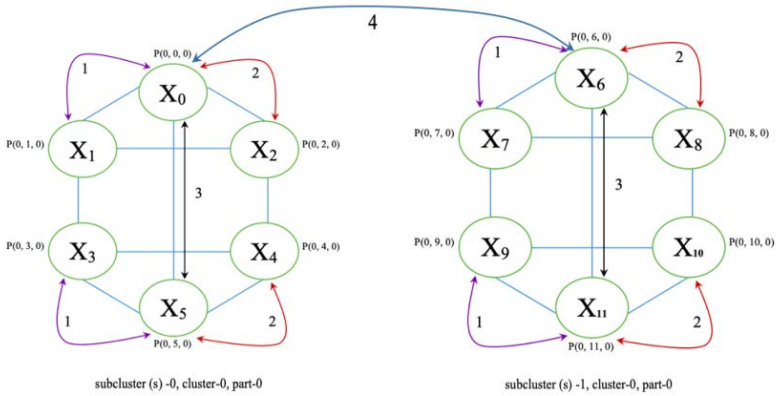

Fig.6. Communication Steps for Computing Min and Max (2-Dimensional Hyper Hexa-Cell)

6. Results & Discussion

The proposed parallel counting sort algorithm sorts $2N^2$ (network size) distinct data elements in $10d_h + 12 + \log_2 S_A$ electronic and 10 optical moves, where S_A is the size of count array: $A_{cip}[S_A]$. The counting sort delivers optimal performance, if difference between the sizes of array: $A_{cip}[S_A]$ and array: $B_{cip}[S_B]$ is least, where S_B is the size of network or the size of output array: $B_{cip}[S_B]$, where final sorted data elements will be stored. To measure performance of our proposed parallel counting sort approach respective to the serial approach of counting sort, performance matrices, such as speedup and efficiency is evaluated. The speedup and efficiency as performance matrices for our proposed parallel counting sort methodology is in Eqs. (1) and (2).

$$\text{Speedup} = \frac{S_A - 1 + 2S_B}{10d_h + 22 + \log_2 S_A} \quad (1)$$

$$\text{Efficiency} = \frac{\frac{S_A - 1 + 2S_B}{10d_h + 22 + \log_2 S_A}}{S_B} \quad (2)$$

Case 1. Assuming the optimal case, when size of count array: $A_{cip}[S_A]$ equals to the size of output array: $B_{cip}[S_B]$. In Tables 1, 2 and 3, speedup and efficiency are displayed for optimal cases. For example, speedup and efficiency are 5.634 and 0.078 respectively for optimal case of One-Dimensional Biswapped Hyper Hexa-Cell network (S_A=72). Similarly, for optimal case of Two-Dimensional Biswapped Hyper Hexa-Cell network (S_A=144), speedup and efficiency are 8.767 and 0.060 respectively. Further, speedup and efficiency are 14.345 and 0.049 respectively for optimal case of Three-Dimensional Biswapped Hyper Hexa-Cell network (S_A=288).

Case 2. Assuming the case when difference between the sizes of array: $A_{cip}[S_A]$ and array: $B_{cip}[S_B]$ is large. In other words, difference between maximal and minimal data values is large. In Tables 1, 2 and 3, such cases are displayed for one, two and three-dimensional Biswapped Hyper Hexa-Cell, when difference between maximal and minimal data values is large. Assuming minimal data value is 5 and maximal data values is 128 respectively among 72 distinctive data values stored at their respective processor of One-Dimensional Biswapped-Hyper Hexa-Cell network, then speedup and efficiency will be 6.854 and 0.095 respectively (as displayed in Table 1). Even speedup and efficiency will become better in case difference between maximal and minimal data values are comparatively becoming larger. *For example*, assuming maximal and minimal data values are 102 and 356 respectively among 72 distinctive data values stored at their respective processor of One-Dimensional Biswapped-Hyper Hexa-Cell network, then the speedup and efficiency will be 9.952 and 0.138 respectively (as displayed in Table 1). Let's take another example when the difference between maximal and minimal data values is comparatively more. *For example*, assuming maximal and minimal data values are 55 and 888 respectively among 72 distinctive data values stored at their respective processor of One-Dimensional Biswapped-Hyper Hexa-Cell network, then the speedup and efficiency will be 23.429 and 0.325 respectively (as displayed in Table 1). It can be observed by analysis of earlier examples that when the difference between maximal and minimal data values are getting larger, speedup and efficiency also increasing.

Similarly, in table 2, 144 distinct numeric values have considered for sorting on 2-Dimensional Biswapped-Hyper Hexa-Cell, assuming each processor stores a single numeric value. Here also we show optimal case for counting sort when the difference between the maximal and minimal data values is least or equal to the size of network. *For example*, assuming maximal and minimal data values are 45 and 188 respectively among 144 distinctive data values stored at their respective processor of Two-Dimensional Biswapped-Hyper Hexa-Cell network, then the speedup and efficiency will be 8.767 and 0.060 respectively (as displayed in Table 2). Let's take another example where difference between maximal and minimal data values is comparatively larger. *For example*, assuming maximal and minimal data values are 12 and 2356 respectively among 144 distinctive data values stored at their respective processor of Two-Dimensional Biswapped-Hyper Hexa-Cell network, then the speedup and efficiency will be 49.482 and 0.343 respectively (as displayed in Table 2). Thus here also it is concluded that for worst case scenario when the difference between maximal and minimal data values is large, proposed algorithm performs comparatively better.

It may also be noted that as difference between maximal and minimal data values is getting larger, more communication moves required for parallel execution. However, only few communication moves (electronic and optical) has been observed as increasing in parallel sorting. *For example*, when the difference between the maximal and minimal data values is 144 for Two-Dimensional Biswapped-Hyper Hexa-Cell network, then total 49.16 communication moves required for sorting. Whereas, total 53.19 communication moves are required for sorting when the difference between maximal and minimal data values is 2344. Therefore, comparatively approx. four communication moves more required for parallel sorting of distinct numeric values.

Table 1. Communication Moves, Speedup & Efficiency for 1-Dimensional Biswapped Hyper Hexa-Cell

Network Size	Size of Array: A[S_A]	Parallel Communication Moves	Speedup	Efficiency
72 (Optimal Case)	72	38.16	5.634	0.078
72	124	38.95	6.854	0.095
72	255	39.99	9.952	0.138
72	538	41.07	16.581	0.230
72	834	41.70	23.429	0.325
72	1521	42.57	39.088	0.542

In Table 3, communication moves (electronic plus optical), speedup and efficiency is displayed for sorting 288 distinct numeric values on Three-Dimensional Biswapped-Hyper Hexa-Cell, assuming each processor stores a single numeric value. For the optimal case, when difference between maximal and minimal data values is least, 60.16 communication moves are required with the speedup and efficiency of 14.345 and 0.049 respectively. Likewise, One and Two-Dimensional Biswapped-Hyper Hexa-Cell, the performance of proposed algorithm on Three-Dimensional Biswapped-Hyper Hexa-Cell also becomes better as difference between maximal and minimal data value is getting

large. *For example*, when the difference between the maximal and minimal data values is 4214 for Three-Dimensional Biswapped-Hyper Hexa-Cell network, then total 64.04 communication moves required for sorting with the speedup and efficiency of 74.797 and 0.259 respectively. Therefore, a significant improvement in speedup and efficiency is observed compared to the optimal case with only near about four-unit increase in the communication moves.

Table 2. Communication Moves, Speedup & Efficiency for 2-Dimensional Biswapped Hyper Hexa-Cell

Network Size	Size of Array: A[S_A]	Parallel Communication Moves	Speedup	Efficiency
144 (Optimal Case)	144	49.16	8.767	0.060
144	267	50.08	11.062	0.076
144	456	50.83	14.617	0.101
144	789	51.62	20.844	0.144
144	1544	52.59	34.816	0.241
144	2345	53.19	49.482	0.343

Table 3. Communication Moves, Speedup & Efficiency for 3-Dimensional Biswapped Hyper Hexa-Cell

Network Size	Size of Array: A[S_A]	Parallel Communication Moves	Speedup	Efficiency
288 (Optimal Case)	288	60.16	14.345	0.049
288	578	61.17	18.849	0.065
288	1024	62.00	25.304	0.087
288	2345	63.19	46.209	0.160
288	3054	63.57	57.086	0.198
288	4214	64.04	74.797	0.259

7. Conclusion

In the present article, optimized parallel counting sort algorithm was proposed for sorting distinct numeric values on optoelectronic d_h-dimensional Biswapped-Hyper Hexa-Cell network. The proposed parallel algorithm demanded total $10d_h + 12 + \log_2 S_A$ electronic and 10 optical moves, where S_A is the size of count array: $A_{cip}[S_A]$.

The performance of the proposed algorithm was measured based on the performance matrices: speedup and efficiency. In fact, in Tables 1, 2 and 3, speedup and efficiency for both cases (Case 1 and Case 2) are displayed. In Case 1, optimal conditions were analyzed, when difference between the maximal and minimal data values is least. In Case 2, such conditions were analyzed when difference between the maximal and minimal data values is large. Based on the analyses in Table 1, 2 and 3, it was observed that the performance of the proposed algorithm (in-terms of speedup and efficiency) was constantly improving for the case when difference between the maximal and minimal data values was getting larger. *For example*, speedup and efficiency was 23.429 and 0.325 for One-Dimensional Biswapped-Hyper Hexa-Cell when the difference between maximal and minimal data values plus one (size of array: $A[S_A]$) was 834. Whereas, speedup and efficiency was 39.088 and 0.542 for One-Dimensional Biswapped-Hyper Hexa-Cell when the difference between maximal and minimal data values plus one (size of array: $A[S_A]$) was 1521. It was also observed that as difference between maximal and minimal data values was getting larger, only few communication moves more required for parallel execution. *For example*, when the difference between maximal and minimal data values was 834 for One-Dimensional Biswapped-Hyper Hexa-Cell network, then total 41.70 communication moves required for sorting. Whereas, total 42.57 communication moves were required for sorting when the difference between maximal and minimal data values was 1521.

Thus based on the analysis it is concluded that the proposed parallel algorithm delivers better performance since speedup and efficiency increases for the worst case scenario as difference between maximal and minimal data values increases, with only few moves increase in required communication moves.

In future, extensive effort required in the direction to develop a novel and efficient parallel algorithm for sorting non-distinctive numeric values on Biswapped-Hyper Hexa-Cell and other competitive optoelectronic networks, such as Biswapped-Torus. Besides, other well-known sorting methods, such as insertion sort, selection sort, merge sort, etc. may be adopted for sorting numeric values on Biswapped-Hyper Hexa-Cell network.

References

[1] W. D. Hills, G. L. Steele, Data parallel algorithm, Communications of the ACM, vol. 29, no. 12, pp. 1170-1183.
[2] G. Marsden, P. Marchand, P. Harvey, S. Esener, Optical transpose interconnection system architecture, Opt. Lett. vol. 18, no.13, pp.1083–1085, 1993.
[3] W. Xiao, B. Parhami, W. Chena, Ma Hea and W. Wei. Biswapped networks: a family of interconnection architectures with

advantages over swapped or OTIS networks, International Journal of Computer Mathematics, vol. 88, no. 13, pp. 2669–2684, 2012.
[4] B. A. Mahafzah, A. Sleit, A. Nesreen. E. F. Hamad. Ahmad, Tasneem M, Abu-Kabeer. The OTIS hyper hexa-cell optoelectronic architecture, Journal of computing, vol. 94, pp. 411–432, 2012.
[5] W. Wei, Q. Li and M. Tao. BSN-mesh and its basic parallel algorithms. International Journal of Grid and Utility Computing, vol. 6, no. 3-4, pp. 213-220, 2015.
[6] C. Zhao. Efficient load balancing on biswapped networks. Cluster Computing, vol. 17, no. 2, pp. 403-411, 2014.
[7] S. Ling and W. Chen. Node-to-Set Disjoint Paths in Biswapped Networks. The computer journal. vol. 57, no. 7, pp. 953-967, 2014.
[8] W. Chen and S. Ling, Node-pancyclic properties of biswapped networks based on cycles in their factor networks. The Computer Journal. vol. 60, no. 1, pp.1-12, 2017.
[9] A. Gupta A and B. K. Sarkar, The recursive and symmetrical optoelectronic network architecture: Biswapped network hyper hexacell. Arabian Journal of Science and Engineering. vol. 43, no. 12, pp. 7635-7653, 2018.
[10] A. Gupta and B. K. Sarkar. Biswapped-Torus Network-A new efficient Node-Symmetrical Optoelectronic Network. National Academy Science Letters. 2020; Online First.
[11] A. Gupta and B. K. Sarkar. BSN MOT: A Fast and Cost-efficient Optoelectronic Architecture. IETE Technical Review. 2010; Online First.
[12] A. Datta, M. De and B. P. Sinha, Fast Parallel Algorithm for Prefix Computation in Multi-Mesh Architecture, Parallel Processing Letters. vol. 27, pp. 1750009-1-1750009-12, 2017.
[13] P.K. Jana, B. D. Naidu, S. Kumar, M. Arora and B. P. Sinha, Parallel Prefix Computation on Extended Multi-Mesh Network. Information Processing Letters. vol. 84, pp. 295-303, 2002.
[14] Ashish Gupta and Bikash Kanti Sarkar, A new parallel approach for prefix sum on BSN mesh, Proc. 2nd Int. Conf. on Next Generation Computing Technologies (Dehradun, Uttrakhand, 2016).
[15] Ashish Gupta, Bikash Kanti Sarkar,"Parallel Prefix Sum Algorithm on Optoelectronic Biswapped Network Hyper Hexa-cell", International Journal of Computer Network and Information Security, Vol.10, No.8, pp.27-35, 2018.
[16] A. Gupta and B. K. Sarkar, Parallel Algorithm for LaGrange's Interpolation on BSN-Mesh, In Proc. Int. Conf. Intelligent Communication, Control and Devices, (Dehradun, 2016), pp. 673-682.
[17] A. Gupta and B. K. Sarkar, An efficient parallel approach for mapping binomial series (special cases) on biswapped network mesh. In Proc. Smart and Innovative trends in next generation computing technologies (Dehradun, 2018), pp. 263-275.
[18] A. Al-Adwan, B.A. Mahafzah, and A. Sharieh, Solving traveling salesman problem using parallel repetitive nearest neighbor algorithm on OTIS-Hypercube and OTIS-Mesh optoelectronic architectures. J Supercomput. vol. 74, pp. 1–36, 2018.
[19] A. Al-Adwan, A. Sharieh and B. A Mahafzah, Parallel heuristic local search algorithm on OTIS hyper hexa-cell and OTIS mesh of trees optoelectronic architectures. Appl Intell. vol 49, pp. 661–688, 2019.
[20] A. Al-Adwan, R. Zaghloul, B. A. Mahafzah, A. Sharieh, Parallel quicksort algorithm on OTIS hyper hexa-cell optoelectronic architecture. Journal of Parallel and Distributed Computing. vol. 141, pp. 61-73, 2020.
[21] Volodymyr Lytvynenko, Olena Kryvoruchko, Irina Lurie, Nataliia Savina, Oleksandr Naumov, Mariia Voronenko, "Comparative Studies of Self-organizing Algorithms for Forecasting Economic Parameters", International Journal of Modern Education and Computer Science, Vol.12, No.6, pp. 1-15, 2020.
[22] Yajnaseni Dash, Sanjay Kumar, V.K. Patle,"Evaluation of Performance on Open MP Parallel Platform based on Problem Size", International Journal of Modern Education and Computer Science, Vol.8, No.6, pp.35-40, 2016.
[23] Peng Zeng,Zhong-Shan Deng,Jing Liu,"Parallel Algorithms for Freezing Problems during Cryosurgery", International Journal of Information Engineering and Electronic Business, vol.3, no.2, pp.11-19, 2011.

Authors' Profiles

Dr. Ashish Gupta did his M. Tech from IIT Dhanbad in Computer Science, and Ph.D. from BIT, Mesra, Ranchi, in Computer Science, India. Presently, he is serving as an Assistant Professor at Dr. Rammanohar Lohia Avadh University, Ayodhya in the Deptt. of Computer Science & Engineering Department for World Bank funded project under Technical Quality Improvement Plan (TEQIP-III) of Ministry of Education, Govt. of India. He is the author of SCIE and Scopus indexed research papers in reputed IEEE, Springer, Taylor & Francis Journal, Conferences and Book chapters. His research interests include Optoelectronic Interconnection Networks, Parallel Algorithm Design and IOT.

How to cite this paper: Ashish Gupta, "Optimized Parallel Counting Sort Algorithm for Distinct Numeric Values on Biswapped Hyper Hexa-Cell Optoelectronic Network", International Journal of Computer Network and Information Security(IJCNIS), Vol.14, No.1, pp.69-80, 2022. DOI: 10.5815/ijcnis.2022.01.06