Efficient Load Balancing Algorithm for the Arrangement-Star Network

Ahmad M. Awwad
Computer Science Department
Petra University
Amman, Jordan

Jehad A. Al-Sadi
Information Technology and Computing Department
Arab Open University
Amman, Jordan

***Abstract*—The Arrangement-Star is a well-known network in the literature and it is one of the promising interconnection networks in the area of super computing, it is expected to be one of the attractive alternatives in the future for High Speed Parallel Computers. The Arrangement-Star network has many attractive topological properties such as small diameter, low degree, good connectivity, low broadcasting cost and flexibility in choosing the desired network size. Although, some of the research work has been done on Arrangement-Star network, it still needs more investigation and research efforts to explore these attractive topologies and utilize it to solve some real life applications. In this paper we attempt to fill this gap by proposing an efficient algorithm for load balancing among different processors of the Arrangement-Star network. The proposed algorithm is named as Arrangement Star Clustered Dimension Exchange Method ASCDEM presented and implemented on the Arrangement-Star network. The algorithm is based on the Clustered Dimension Exchange Method (CDEM). The ASCDEM algorithm is shown to be efficient in redistributing the load balancing among all different processors of the network as evenly as possible. A complete detail of this algorithm in addition to examples and discussions to explore the benefits of applying this distributed algorithm is presented in this paper. Furthermore an analytical study on the algorithm is presented and discussed to explore the attractive performance of the proposed algorithm.**

Keywords*—*Interconnection Networks; Topological Properties; Arrangement-Star; Load balancing

I. INTRODUCTION

The Arrangement-Star network as a case of study on vertex product networks [1, 2, 3, 4], it is constructed from the cross product of the Star and Arrangement graphs. It has shown to have superior topological properties over its constituents: the Star and Arrangement graphs [5, 6, 7]. Besides having a smaller diameter, node degree, and number of links, it has a lower broadcasting cost and more flexibility in choosing the desired network size.

Although some algorithms proposed for the Arrangement-Star graph such as distributed fault-tolerant routing algorithm [5], some of the important problems that the Arrangement-Star network still needs more efforts and researchers is the issue of load balancing among different processors of this network. Since there is no enough research work in literature for proposing efficient algorithms for load balancing on the Arrangement-Star network. In this research effort we move one more step in filling this gap by investigating and proposing the ASCDEM algorithm on the Arrangement-Star network, the proposed algorithm is based on the CDEM algorithm which was able to redistribute the load balance among all node of the networks on OTIS-Hypercube network as evenly as possible [8]. A reasonable and efficient implementation of the ASCDEM algorithm on the Arrangement-Star network will make it more attractive for the solving real life applications problem.

This paper is organized as follows: In the next section we present the related work on load balancing, section III introduces the necessary basic notations and definitions, section IV presents the implementation of the ASCDEM algorithm on the Arrangement-Star network, furthermore we present examples on SCDEM algorithm to explain and explore detailed transactions of the algorithm on different Arrangement-Star network sizes, section V presents an analytical study of the ASCEDM algorithm, finally section VI concludes this research work.

II. BACKGROUND AND RELATED WORK

Many attractive properties for the Arrangement-Star graph have been shown in the literature enabled it to be one of the candidate's networks for the High Speed Parallel Computers (HSPC) and a reasonable choice for any real life applications [5]. This outcome about Arrangement-Star network has motivated us to spend more time and do some research on it for some important class of algorithms such as: the load balancing because still this networks suffers from shortening in number of algorithms for the load balancing problem in general and for load balancing problem in specific. This algorithm has been studied and proposed for many HSPC infrastructures ranging from electronic networks [5] and also for Optoelectronic networks [9, 10].

The Load balancing algorithm is a famous type of problems that is needed by all HSPC infrastructures. The load balancing problem have been investigated from many angles and point views. As an example on the literature work this problem was investigated by the researchers Ranka, Won, and Sahni [11], As conclusion of their work they come out with an efficient algorithm to be implemented on HSPC called the Dimension Exchange Method (DEM) on the hypercube topology. This algorithm (DEM) constructed and developed by issuing and getting the average load of neighbors' nodes, where the symmetric degree of the hypercube is n, All adjacent nodes which are connected on the n^{th} dimension they will exchange their task loads to redistribute the task load and as evenly as possible, the processor with extra load will share

any extra amount of the load to its adjacent neighbor node. The DEM algorithm main advantage that it was able to redistribute the load balances of processors among all neighbors as evenly as possible. Furthermore Ranka and *et al* have enhance the load balance in the DEM algorithm in its worst case to achieve log2n on the cube network [11].

Zaho, Xiao, and Qin have investigated and proposed hybrid structure of diffusion and dimension exchange called DED-X which worked in a perfect manner for the load balancing algorithm on Optoelectronic networks [12]. The DED-X problem main task was to redistribute the load balancing between different nodes of the network to three different phases. The achieved outcome on Optical Transpose Interconnection System networks proved that the redistribution of load balance between all nodes of the topology was efficient and mostly even. Furthermore the reached outcome and the issued results of the simulation from Zaho et al of the proposed algorithms on load balancing has shown a considerably big improvements in enhancement in redistribution the load balancing of the processors of the topology [12]. In a different literature and research done by Zaho and Xiao they investigated a different algorithm named t DED-X for load balancing on homogeneous optoelectronic technology and they proposed new algorithm framework, Generalized Diffusion-Exchange- Diffusion Method, this framework was efficient for the load balancing distribution on the Heterogeneous optoelectronic technology [11, 13, 14].

On the other hand Zaho, Xiao, and Qin have investigated and proved that the efficiency of the new investigated load balancing algorithms to be more effective than the X old load balancing algorithm [12].

The target of this research effort is to investigate a new algorithm for the load balancing among the nodes of the Arrangement-Star networks named Arrangement Star Clustered Dimension Exchange Method (ASCDEM). The algorithm is based on the Clustered Dimension Exchange Method (CDEM) [13].

III. Definitions and Topological Properties

During the last two decades a big number of interconnection networks for High Speed Parallel Computers (HSPC) investigations are proposed in literature [6, 13, 15]. As an example one of these networks was the hypercube interconnection network [8, 13]. Also a well know example is the Star network [6]. Some properties of this network have been studied in the literature including its basic topological properties, parallel path classification, node connectivity and embedding [17, 18, 19, 20]. The authors Akers and Krishnamurthy have proved that the Star graph has several advantages over the hypercube network including a lower degree for a fixed network size of the comparable network sizes, a smaller diameter, and smaller average diameter. Furthermore they showed that the Star graph is maximally fault tolerant edge, and vertex symmetric [6].

The major drawback of the Star network is related to its scalability problem [21]. The size of the Star network increases as a factorial function, and thus grows widely very rapidly; for example, the value of 5! is equal to 120 while the value of 6! is 720. Until today despite its attractive topological properties, the Star graph has not been used in practical systems yet because of this problem.

In an attempt to address this problem in the Star network, Day and Tripathi [7] have proposed the Arrangement graph as a generalization of the Star graph. The Arrangement graph is a family of undirected graphs that contains the Star graph family. It slightly brings a solution to the problem of the scalability, which the Star graph suffers from (i.e. the problem of growth of the number $n!$ of nodes in the n-Star). It also preserves all the nice qualities of the Star graph topology including, hierarchical structure, vertex and edge symmetric, simple shortest path routing and many fault tolerance properties [7]. Still a common drawback of the Star and Arrangement graphs is the restriction on the number of nodes: $n!$ for the Star graph and $m!/(m-k)!$ for the Arrangement graph. The set of values of $n!$ (or $m!/(m-k)!$) is spread widely over the set of integers; so, one will be faced with the choice of too few or too many available nodes.

However, there has been relatively a limited research efforts have been dedicated to design efficient algorithms for the Arrangement-Star graph including broadcasting [19], selection and sorting [20, 22], Fast Fourier Transform [23], and Matrix Multiplications [24] and load balancing. In an attempt to overcome the load balancing problem we present an efficient algorithm for load balancing problem on Arrangement-Star graph to redistribute the load balancing among all processors of the network as evenly as possible.

An Arrangement graph is specified by two parameters m and k, satisfying $1 \le k \le m$. For simplicity let $\langle m \rangle = \{1,2,\ldots,m\}$ and $\langle k \rangle = \{1,2,\ldots,k\}$.

Definition 1: The (m,k)-Arrangement graph $A_{m,k} = (V_1, E_1)$, $1 \le k \le m$-1 is defined as follows [7]:

$V_1 = \{p_1 p_2 \ldots p_k \mid p_i \in \langle m \rangle$ and $pi \ne p_j$ for $i \ne j\} = P_k^m$, and

$E_1 = (p,q) \mid p$ and q in V_1 and for some i in $\langle k \rangle$, $p_i \ne q_i$ and $p_j = q_j$ for $j \ne i\}$.

That is, the nodes of $A_{m,k}$ labelled with a unique Arrangements of k elements out of m symbols $\langle m \rangle$, and the edges of $A_{m,k}$ connect Arrangements which differ in exactly one of their k positions An edge of $A_{m,k}$ connecting two Arrangements which differ only in position i called an i-edge. In this case, p and q are i-adjacent and q is called (i, q_i)-neighbour of p. The (m,k)-Arrangement graph $A_{m,k}$ is regular of degree $k(m-k)$ and of size $m!/(m-k)!$, and diameter $\lfloor 3k/2 \rfloor$. The $(m, m\text{-}1)$-Arrangement graph $A_{m,m-1}$ is isomorphic to n-Star graph S_n [3, 7], and the $(m,1)$-Arrangement graph is isomorphic to the complete graph with m nodes [7].

Definition 2: The n-Star graph, denoted by S_n, has $n!$ nodes each labelled with a unique permutation on $\langle n \rangle = \{1,\ldots,n\}$. Any two nodes are connected if, and only if, their corresponding permutations differ exactly in the first and one other position.

The diameter, δ, and the degree, α, of the Star graph are as follows [6]:

δ, of *n*-Star graph = ⌊1.5 (n-1)⌋

α, of the *n*-Star graph = *n*-1, where *n*>1.

Definition* 3*: The Arrangement-Star graph is the cross product of the *n*-Star graph and the (*m*, *k*)-Arrangement graph, and is given by $AS_{n,m,k} = A_{m,k} \otimes S_n$ such that $n>1$ and $1 \leq k \leq m$.

Note that if G_1 and G_2 are two undirected graphs then for any node $X = \langle x_1, x_2 \rangle$ in the cross product graph, $G = G_1 \otimes G_2$, has an address consisting of two parts, one coming from G_1 and the other coming from G_2. We will denote the earlier part by $lp(X)=x_1$ and the later part by $rp(X)=x_2$.

Figure 1 shows the topology of $AS_{2,3,2}$ that is obtained from the graph product of S_2 and $A_{3,2}$ networks. A node $X = \langle u, v \rangle$ in $AS_{2,3,2}$ consisting of two parts, left part coming from the Star graph and the right part coming from the Arrangement graph (*lp and rp*). Two nodes $X = \langle u, v \rangle$ and $Y = \langle u', v' \rangle$ are connected if, $lp(X) = lp(Y)$ and $rp(X)$ is connected $rp(Y)$ in $A_{m,k}$ (in this case *X* and *Y* are said Arrangement-connected) or $rp(X)= rp(Y)$ and $lp(X)$ is connected $lp(Y)$ in S_n (in this case *X* and *Y* are said Star-connected). For instance in Figure 1 the node *ab*13 is connected to the node *ab*12, and the node *ab*23 is connected to the node *ba*23.

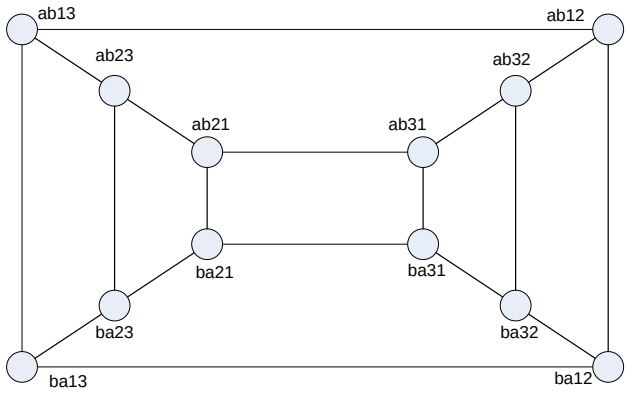

Fig. 1. Arrangement-Star graph, AS2,3,2

IV. The Implementation of the Ascdem Algorithm on the Arrangement-Star Network

The algorithm we present in this paper ASCDEM is based on the Clustered Dimension Exchange Method CDEM for load balancing for the Arrangement-Star Interconnection networks [8].

The main achievement of the new presented ASCDEM is to obtain even load balancing for the $AS_{n,m,k}$ network by redistributing the load size to reach an equal load size at each node within the whole network. The structure of the $AS_{n,m,k}$ network consists of S_n network as a first level structure of the hierarchal $AS_{n,m,k}$ network, the first level of S_n consists of *n*! Sub-graphs, each sub-graph represented by an $A_{m,k}$ Arrangement graph. The links and edges between the nodes of the whole graph have been identified and described in the above section.

The ASCDEM load balancing algorithm is based on the following two phases:

- Phase 1: Distributing the load balancing among all sub-graphs of the first level hierarchal S_n graph, we Start by balancing the load of every two nodes via the edges that connect these sub-graphs within the Star topology structure. By the end of this phase we guarantee that all sub-graphs will have almost the same total number of loads since each sub-graph is represented as if it is a single node of the Star network structure in the first level hierarchy. It worth to mention here, that the load within each sub-graph is not sorted at this stage. To complete this phase we need to make *n*!/2 parallel redistribution steps of load among every two nodes via a Star structure edge. But at each of these parallel steps, there will be an *n*-1 sequential exchanges for each node with its *n*-1 neighbors within the Star structure.

- Phase 2: Distributing the load size within each sub-graph, this will the second level of the hierarchal $AS_{n,m,k}$ network, where each sub-graph is an Arrangement graph representation, by the end of the phase 1, all sub-graphs will have the same load size, then by redistributing the load sizes among these Arrangement graphs, the whole $AS_{n,m,k}$ network will have almost equal load sizes at each node. This phase requires *m*!/2(*m*-*k*)! parallel redistribution steps of load among every two nodes via an Arrangement structure edge. But at each of these parallel steps, there will be a *k**(*m*-*k*) sequential exchanges for each node with its *k**(*m*-*k*) neighbors within the Arrangement structure. By the end of this phase, all nodes will have almost the same load size, the following algorithm in Feg 2 describe the ASCDEM method of load balancing.

ASCDEM algorithm works on redistributing load balancing among all processors of the network, phase 1 is done in parallel among all nodes via the Star topological connections. Then Phase 2 is also done in parallel among all nodes via the Arrangement topological connections.

- Phase 1: The load balancing between the processors; sub-graphes; of Sn based on ASCDEM algorithm is exchanged as in steps 1 to 14 in parallel, at first step the load exchange will be between all the processors in which they differ in 1st position and 2nd position for all the factor networks of Sn i.e. S_{n-1}. Then the same process will be repeated continually until it reach the neighbours p_j that are *n* positions far away from p_i. By the end of this phase all sub-graphs will have almost the same total number of load sizes.

Note that *n*-1 is the number of neighbors of any processor in S_n:

1. *for* $p = 2; p \leq n; p{+}{+}$ // Start of phase#1
2.
3. *for* all neighbour nodes p_i and p_j which they differ in 1st and p position of S_n do in parallel
4. Give-and-take p_i and p_j total load sizes of the two nodes
5. *TheAverageLoad* $p_{i,j}$ = Floor (*Load* p_i + *Load* p_i)/2
6. *if* (*Totalload* p_i >= excess *AverageLoad* $p_{i,j}$)
7. *Send* excess load p_i to the neighbour node p_i
8. Load p_i = Load p_i – extra load
9. Load p_j = Load p_j + extra load
10. *else*

11. Receive extra load from neighbour p_j
12. Load p_i = Load p_i + extra load
13. Load p_j = Load p_j – extra load
14. Repeat steps (1 to 13) $\left\lfloor \frac{3}{2}(n-1) \right\rfloor$ times // diameter of the Star Topology -End of phase#1
15. *for* $p = 1$; $p \leq k$; p // *the* k^{th} *positions*- Start of phase#2
16. *for* $d = 1$; $d < k*(m-k+1)$; d++ // differ of k^{th} position
17. *for* all neighbor nodes p_{ki} and p_{kj} which they differ in exactly one k position of $A_{m,k}$ do in parallel
18. Give-and-take p_{ki} and p_{kj} total load sizes of the every two neighbor nodes where there are differ in exactly d in their k^{th} position // $|k_i - k_j| = d$ excluding the fixed positions
19. *TheAverageLoad* $p_{ki,kj}$ = Floor (*Load* p_{ki} + *Load* p_{ki})/2
20. *if* (*Totalload* p_{ki} >= excess *AverageLoad* $p_{ki,kj}$)
21. *Send* excess load p_{ki} to the neighbour node p_{ki}
22. Load p_{ki} = Load p_{ki} – extra load
23. Load p_{kj} = Load p_{kj} + extra load
24. *else*
25. Receive extra load from neighbour p_{kj}
26. Load p_{ki} = Load p_{ki} + extra load
27. Load p_{kj} = Load p_{kj} – extra load
28. Repeat steps (15 to 27) $\left\lfloor \frac{3}{2}k \right\rfloor$ times // diameter of the Arrangement topology -End of phase#2

Fig. 2. The ASCDEM load balancing Algorithm

• Phase 2: The load balancing within the processors of each sub-graph where each sub-graph is an $A_{m,k}$ network. The ASCDEM algorithm in steps 15 to 28 performed in parallel, at first step the load exchange will be between all the processors in which they differ in exactly one k position for any two neighboring nodes, which means they are connected via an Arrangement structure. Then the same process will be repeated continually all of the $\left\lfloor \frac{3}{2}k \right\rfloor$ neighbors. By the end of this phase all nodes of the network will have almost the same load size.

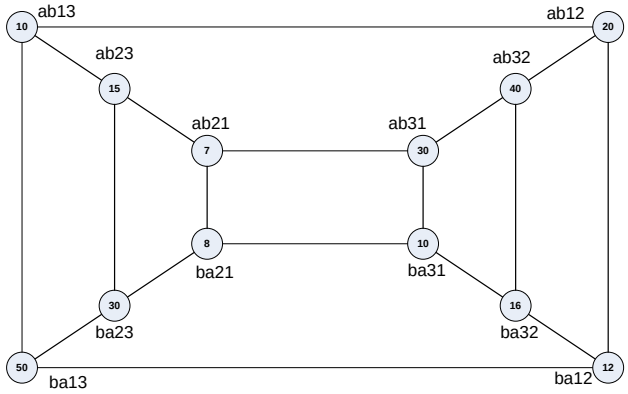

Fig. 3. Arrangement-Star graph, $AS_{2,3,2}$ with initial loads

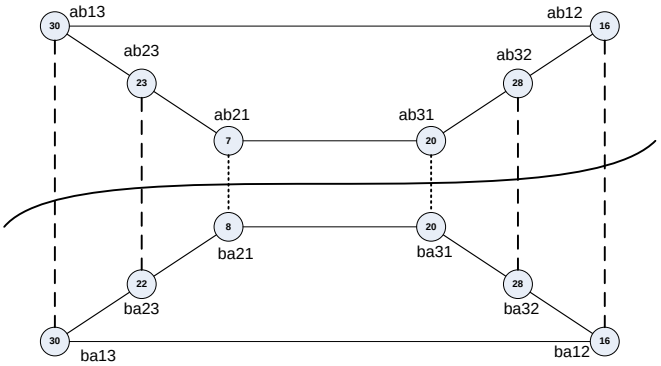

Fig. 4. Arrangement-Star graph, $AS_{2,3,2}$ after performing ASCDEM phase 1

Example 1: - To explain the ASCDEM algorithm presented in Fig. 2, the following example implements the load balancing algorithm on a $AS_{n,m,k}$ network where n=2, m=3, and k=2.

Fig. 3 shows the $AS_{2,3,2}$ network, it consists of two sub-graphs connected to each other via a Star stricture, each sub-graph is represented as an Arrangement graph; $A_{3,2}$ which has 6 processors. The total number of processors in the $AS_{2,3,2}$ network is twelve. Each node has an original load size assigned to it and it is represented in the figure inside every node. Since the degree of $AS_{2,3,2}$ is 3, it follows that each node connected to three other direct nodes, two of nodes via the Arrangement graph edges, and one node via the Star graph edge.

First we Start by implanting phase 1 of the algorithm by following the steps 2-12. Fig. 4 shows the new load size for each node after completing the first phase, edges in bold and dash lines represent the Star graph structure links, the curve line is to distinguish between the two Arrangement graphs in the figure. By the end of this phase, the total load sizes for each of the two sub-graphs are almost equal.

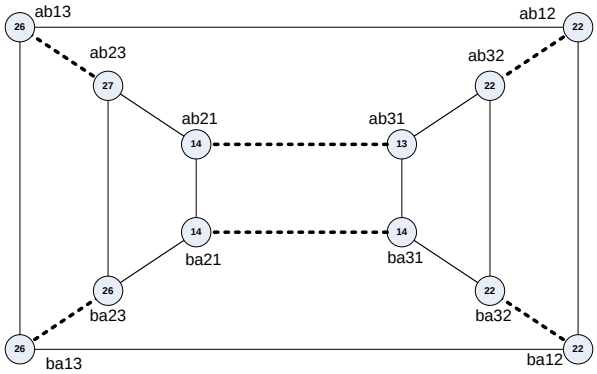

Fig. 5. Arrangement-Star graph, $AS_{2,3,2}$ after performing phase 2 where k=1

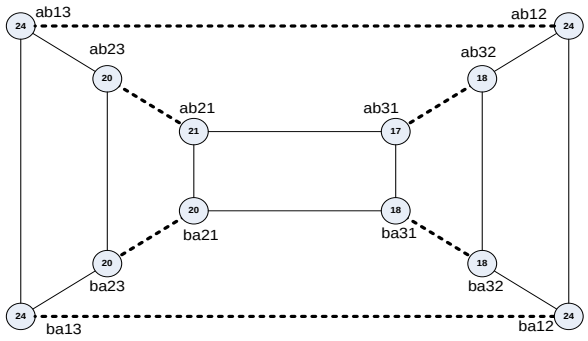

Fig. 6. Arrangement-Star graph, $AS_{2,3,2}$ after performing phase 2 where k=2

In phase 2 of the algorithm all adjacent nodes which differ in one and only one of their k position will redistribute their load balancing equally, this phase is done for each sub-graph separately and in parallel, at each parallel step, each node do this redistribution with its neighbors via an Arrangement graph edges. Figure 5 shows the parallel redistribution of loads within each Arrangement sub-graph for the first position where k=1, the dashed lines represent these exchanges between every two pair of nodes. Figure 6, shows the results for the k position=2. Figure 7 shows the final redistributed load size of every node, noting that the second phase of the algorithm is repeated 3 times; $m!/2(m-k)!$; to guarantee that equal distribution is done across the whole network to reach nodes at diameter distance from each other. Furthermore, all node exchanges at the same k position are done in parallel. The final results prove the efficiency of our algorithm where all nodes' loads are almost equal.

To present the Arrangement-Star network clearly, figure 8 is an example of this network where we refer to it as $AS_{3,3,2}$. The size of the Star is 6 and the each node of the Star is presented by an Arrangement network of $A_{3,2.}$ The size of each Arrangement network is also 6 nodes. The total size of the whole network is 36 nodes as it is obvious from the figure below that each node is connected other neighboring nodes based of the properties of the Star and the Arrangement networks. For example the node abc13 is connected to the two neighboring nodes; abc12 and abc23; via the Arrangement graph properties and also it is connected to the two neighboring nodes; bac13 and cba13; via the Star network properties.

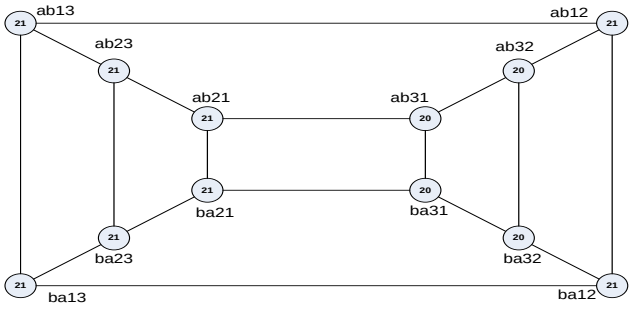

Fig. 7. Arrangement-Star graph, $AS_{2,3,2}$ after performing the ASCDEM algorithm

In the following example we will present the ASCDEM algorithm behavior on $AS_{3,3,2}$ described above to give more details about this algorithm:

Example 2: - To explore the ASCDEM algorithm presented in Fig. 2 in more details, we present another Arrangement-Star network to implement the load balancing algorithm on it. This network is denoted as $AS_{n,m,k}$ network where n=3, m=3, and k=2.

Fig. 9 shows the Arrangement-Star graph of $AS_{3,3,2}$ interconnection network, where each node in of the 6 nodes Star graph is represented by a complete Arrangement network of $A_{3,2,}$ which consists of 6 nodes. Since the degree of $AS_{3,3,2}$ is n-1 + $k(m-k)$ which is equal to 4, it follows that each node connected to four other direct nodes, two of nodes via the Arrangement graph edges, and two nodes via the Star graph edge.

First we Start by implanting phase 1 of the algorithm by following the steps 2-12. Fig. 10 shows the new load size for each node after completing the first phase. By the end of this phase, the total load sizes for each of the six sub-graphs are almost equal. Where each sub-graph is represented by an Arrangement network of $A_{3,2}$.

Fig. 11 shows the final load size for each node after completing the second phase which is also the final phase of the algorithm. By the end of this phase, the total load size for each node is almost equal. This proves that our algorithm works properly and performs the load balancing accurately.

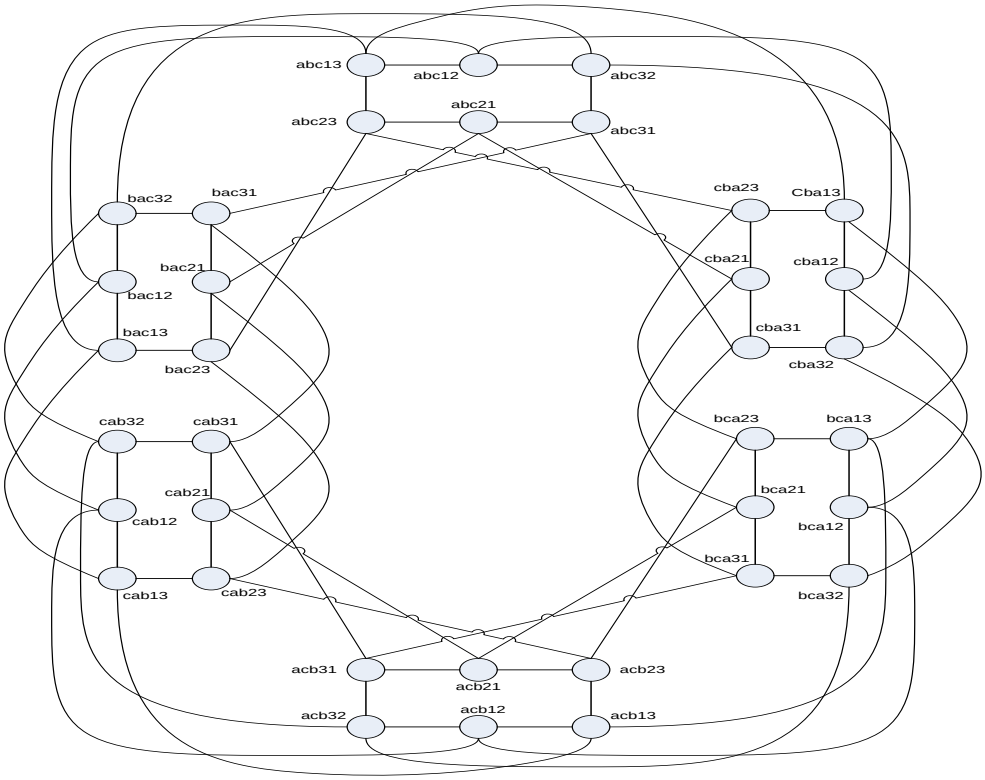

Fig. 8. Arrangement-Star graph, $AS_{3,3,2}$

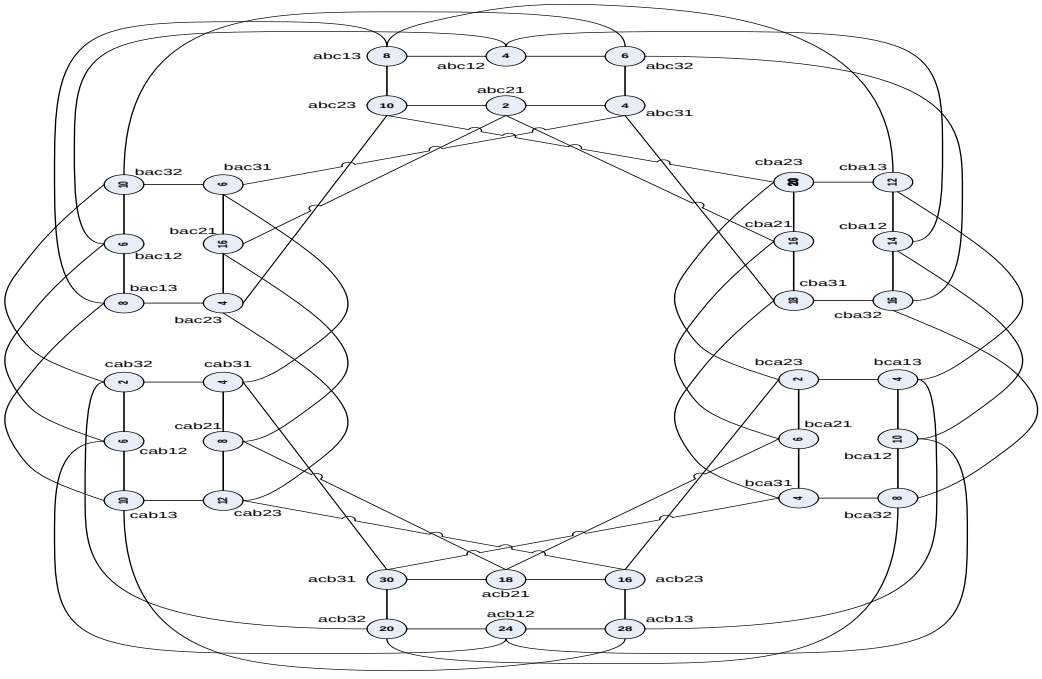

Fig. 9. Arrangement-Star graph, $AS_{3,3,2}$ –Initial state

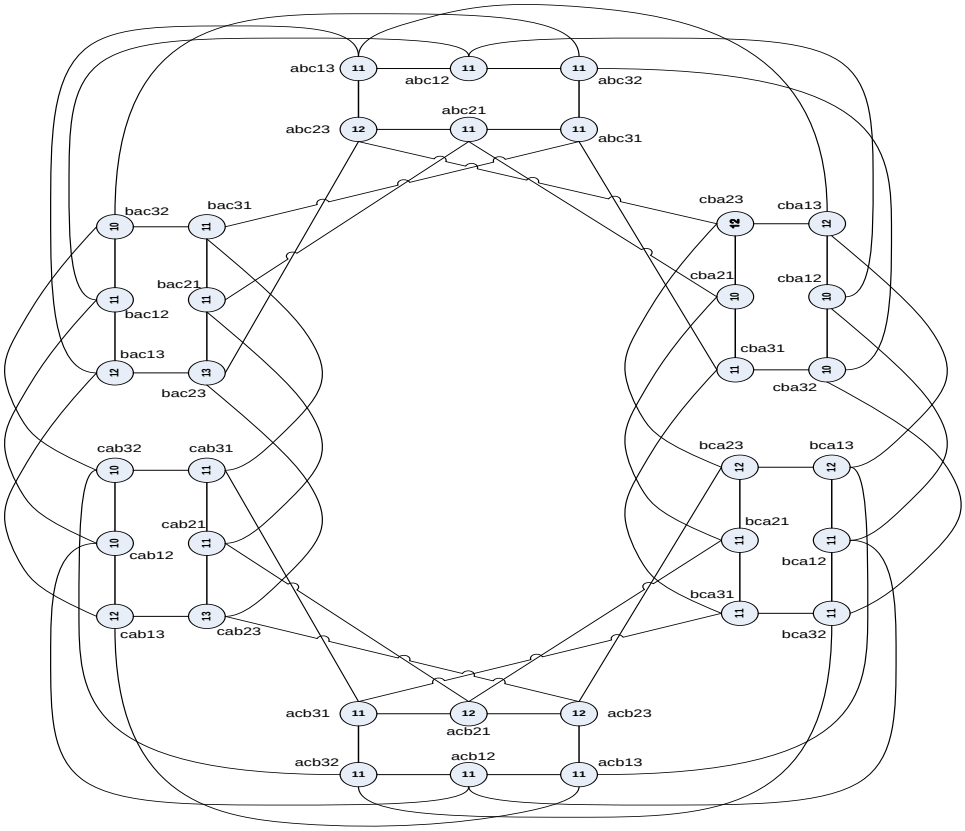

Fig. 10. Arrangement-Star graph, $AS_{3,3,2}$ – End of phase 1

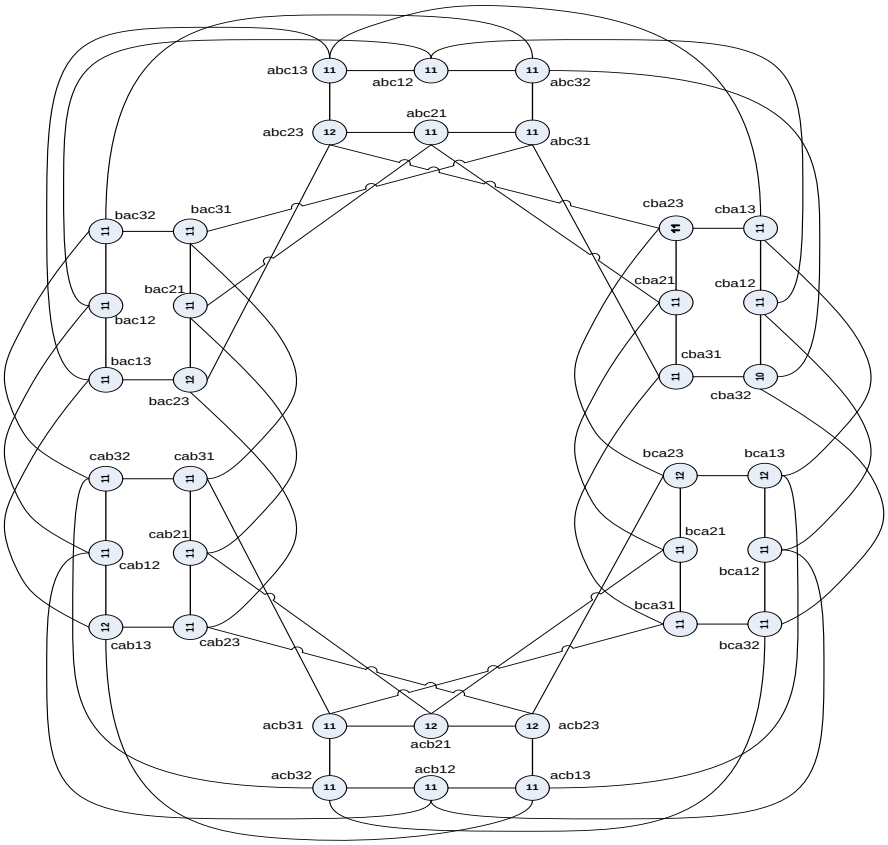

Fig. 11. Arrangement-Star graph, $AS_{3,3,2}$ End of phase 2

V. Analytical Study

In this section we introduce analytical results of the ASCDEM load balancing algorithm behavior on the Arrangement-Star network. The following propositions are summarizing the outcome of this analytical study.

Proposition #1: By performing the first phase of ASCDEM algorithm, the total number of sequential steps is equal to the diameter of the Star network which is $\left\lfloor \frac{3}{2}(n-1) \right\rfloor$.

Since the structure of the Arrangement-Star graph is based on representing an Arrangement graph as a node within the Star graph with parallel connectivity within the nodes of the whole network, ASCDEM algorithm utilizes this feature to distribute the load sizes among the sub-graphs of the graph; Arrangement networks; by utilizing the properties of the Star graph to distribute the loads where the longest length between any two sub-graphs is equal to the diameter of the Star graph.

Proposition #2: The total number of sequential steps performed by ASCDEM algorithm in the second phase is equal to the diameter of the Arrangement network which is $\left\lfloor \frac{3}{2}(K) \right\rfloor$.

As mentioned earlier, the structure of the Arrangement-Star graph is based on representing an Arrangement graph as a node within the Star graph. By the Start of phase 2 of the ASCDEM algorithm, each sub-graph which is represented by an Arrangement graph will distribute the load sizes among its nodes distantly from other sub-graphs of the whole graph. To do this redistribution we need to perform a diameter sequentially steps to reach the farthest two nodes of the Arrangement graph.

Proposition #3: To perform the two phases of ASCDEM algorithm, the total number of sequential steps is equal to:

$\left\lfloor \frac{3}{2}(K) \right\rfloor + \left\lfloor \frac{3}{2}(n-1) \right\rfloor$ where $\left\lfloor \frac{3}{2}(K) \right\rfloor$ is the diameter of the Arrangement graph and $\left\lfloor \frac{3}{2}(K) \right\rfloor$ is the diameter of the Star graph.

Propositions number 1 and 2 justifies the outcome presented in proposition number 3.

Proposition #4: By the end of the first phase of ASCDEM algorithm, the total number of load sizes at each sub-graph of the whole network is almost equal.

After the $\left\lfloor \frac{3}{2}(n-1) \right\rfloor$ exchanges of load sizes which represent the diameter of the Star graph, every node of the Star structure will has almost an equal size of load. Furthermore since each node of the Star structure is represented by a sub-graph of Arrangement graph structure, then each sub-graph will have an almost the same size of load. These exchanges are performed in the first phase of the ASCEDM algorithm.

Proposition #5: By the end of phase 2 of ASCDEM algorithm, the total number of load sizes at each node of the whole network is almost equal.

Since phase 1 guarantees equal redistribution of load sizes among the sub-graphs of the network and by the end of second phase every sub-graph will redistribute the load size among its nodes, then each node will have an almost the same size of load. These exchanges are performed in the second phase of the ASCEDM algorithm.

Proposition #6: At each sequential step of phase 1 of ASCEDM algorithm, the total number of parallel exchanges is:

$\frac{n!}{2} * \frac{m!}{(m-k)!}$ where n denotes the n-Star network and m,k denotes the Arrangement $A_{m,k}$ network.

Since every two nodes of the Star graph exchanges their load sized at once and the number of nodes of the Star graph is $n!$, then there are $\frac{n!}{2}$ exchanges done in parallel. Also since every node in the Star graph is represented as an Arrangement graph, then these exchanges are actually performed within the nodes of every 2 Arrangement sub-graphs, where the size of each sub-graph is equal to $\frac{m!}{(m-k)!}$.

Proposition #7: The total number of exchanges performed by phase 1 of ASCEDM algorithm is equal to $\frac{n!}{2} * \frac{m!}{(m-k)!} * \left\lfloor \frac{3}{2}(n-1) \right\rfloor$ where n denotes the n-Star network and m,k denotes the Arrangement $A_{m,k}$ network.

By referring to proposition number 6 which explain the number of parallel exchanges at each sequential step and also by referring to proposition number 1 which explains the number of sequential steps of the first phase, it gives a clear justification of the outcome of this proposition.

Proposition #8: At each sequential step of phase 2 of ASCEDM algorithm, the total number of parallel exchanges is:

$n! * \frac{m!*(m-k)!}{2}$ where n denotes the n-Star network and m,k denotes the Arrangement $A_{m,k}$ network.

Since every two nodes of the arrangment graph exchanges their load sized at once and the number of nodes of the Arrangement graph is $\frac{m!}{(m-k)!}$, then there are $\frac{m!*(m-k)!}{2}$ exchanges done in parallel. Also since this occurs at every representation of a node in the Star graph, then these exchanges are actually performed $n!$ in parallel, where the Star graph has $n!$ number of nodes.

Proposition #9: The total number of exchanges performed by phase 2 of ASCEDM algorithm is equal to $\frac{m!*(m-k)!}{2} * n! * \left\lfloor \frac{3}{2}(k) \right\rfloor$ where n denotes the n-Star network and m,k denotes the Arrangement $A_{m,k}$ network

By referring to proposition number 8 which explain the number of parallel exchanges at each sequential step and also by referring to proposition number 2 which explains the number of sequential steps of the second phase, it gives a clear justification of the outcome of this proposition.

Proposition #10: The total number of exchanges performed by the whole algorithm of ASCEDM is equal to

$\frac{m*n!}{2}((m-k)! * \left\lfloor \frac{3}{2}(k) \right\rfloor + \frac{1}{(m-k)!} * \left\lfloor \frac{3}{2}(n-1) \right\rfloor)$ where n denotes the n-Star network and m,k denotes the Arrangement $A_{m,k}$ network

The above equation represents the total summation of proposition number 7 and proposition number 9.

By utilizing the above propositions to the well known general equations of latency time, communication cost, throughput, and processors speed we can extend our study to present real outcomes and results based on the specifications of the machines that could be built on the above network and utilizes the ASCEDM algorithm. We think this will be open ideas for any future work based on our algorithm.

VI. Conclusion

In this research we have investigated and proposed an algorithm named Arrangement-Star Clustered Dimension Exchange Method (ASCDEM), the proposed algorithm in based on the well-known efficient algorithm SCDEM which was proposed by Mahafza and et al named (CDEM).The main target of the ASCDEM algorithm is to redistribute the load balancing among all the processors of the Arrangement-Star network as evenly as possible. As shown above the algorithm was able to redistribute the load balance among all the nodes of the $AS_{n,m,k}$ in an efficient approach.

Furthermore, two detailed examples were conducted and discussed to explore and explain the two phases of the ASCDEM algorithm. Also an analytical study was performed on this algorithm which presents the quantities specifications of the algorithm. This analytical study could be utilized for any future work to propose a further performance study on the proposed algorithm such as: total execution time, efficient load balancing accuracy, latency, number of communication moves and complexity speed of the ASCDEM.

References

[1] K. Day and A. Al-Ayyoub, "The Cross Product of Interconnection Networks", *IEEE Trans. Parallel and Distributed Systems*, vol. 8, no. 2, Feb. 1997, pp. 109-118.

[2] S. B. Akers, and B. Krishnamurthy, "A Group Theoretic Model for Symmetric Interconnection Networks," *Proc. Intl. Conf. Parallel Proc.*, 1986, pp. 216-223.

[3] Ayyoub, "The Cross Product of Interconnection Networks", *IEEE Trans. Parallel and Distributed Systems*, vol. 8, no. 2, Feb. 1997, pp. 109-118.

[4] A. Al-Ayyoub and K. Day, "A Comparative Study of Cartesian Product Networks", *Proc. of the Intl. Conf. on Parallel and Distributed Processing: Techniques and Applications*, vol. I, August 9-11, 1996, Sunnyvale, CA, USA, pp. 387-390.

[5] Ahmad Awwad, "vertex Product networks", University of Glasgow, Computer Science Dept. thesis, 2001.

[6] S. B. Akers, D. Harel and B. Krishnamurthy, "The Star Graph: An Attractive Alternative to the n-Cube" *Proc. Intl. Conf. Parallel Processing*, 1987, pp. 393-400.

[7] K. Day and A. Tripathi, "Arrangement Graphs: A Class of Generalised Star Graphs," Information Processing Letters, vol. 42, 1992, pp. 235-241.

[8] Jehad Al-Sadi, "Implementing FEFOM Load Balancing Algorithm on the Enhanced OTIS-n-Cube Topology", Proc. of the Second Intl. Conf. on Advances in Electronic Devices and Circuits - EDC 2013, 47-5.

[9] G. Marsden, P. Marchand, P. Harvey, and S. Esener, "Optical Transpose Interconnection System Architecture," *Optics Letters*, 18(13), 1993, pp. 1083-1085.

[10] Qin Y, Xiao W, Zhao C (2007), "GDED-X schemes for load balancing on heterogeneous OTIS networks", In: ICA3PP, pp 482–492.

[11] Ranka, Y. Won, S. Sahni, "Programming a Hypercube Multicomputer", IEEE Software, 5 (5): 69 – 77, 1998.

[12] Zhao C, Xiao W, Qin Y (2007), "Hybrid diffusion schemes for load balancing on OTIS networks", In: ICA3PP, pp 421–432

[13] B.A. Mahafzah and B.A. Jaradat, "The Load Balancing problem in OTIS-Hypercube Interconnection Network", J. of Supercomputing (2008) 46, 276-297.

[14] N. Imani et al, "Perfect load balancing on Star interconnection network", J. of supercomputers, Volume 41 Issue 3, September 2007. pp. 269 – 286.

[15] K. Day and A. Tripathi, "A Comparative Study of Topological Properties of Hypercubes and Star Graphs", IEEE Trans. Parallel & Distributed Systems, vol. 5.

[16] Kaled Day and Abdel-Elah Al-Ayyoub, "Node-ranking schemes for the Star networks", Journal of parallel and Distributed Computing, Vol. 63 issue 3, March 2003, pp 239-250.

[17] I. Jung and J. Chang, "Embedding Complete Binary Trees in Star Graphs," *Journal of the Korea Information Science Society*, vol. 21, no. 2, 1994, pp. 407-415.

[18] Berthome, P., A. Ferreira, and S. Perennes, "Optimal Information Dissemination in Star and Panckae Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 7, no. 12, Aug. 1996, pp. 1292-1300.

[19] Mendia V. and D. Sarkar, "Optimal Broadcasting on the Star Graph," *IEEE Trans. Parallel and Distributed Systems*, Vo;. 3, No. 4, 1992, pp. 389-396.

[20] S. Rajasekaran and D. Wei, "Selection, Routing, and Sorting on the Star Graph," *J. Parallel & Distributed Computing*, vol. 41, 1997, pp. 225-33.

[21] A. Al-Ayyoub and K. Day, "The HyperStar Interconnection Network," J. Parallel & Distributed Computing, vol. 48, no. 2, 1998, pp. 175-199.

[22] A. Menn and A.K. Somani, "An Efficient Sorting Algorithm for the Star Graph Interconnection Network," Proc. Intl. Conf. on Parallel Processing, 1990, pp.1-8.

[23] P. Fragopoulou and S. Akl, "A Parallel Algorithm for Computing Fourier Transforms on the Star Graph," *IEEE Trans. Parallel & Distributed Systems*, vol. 5, no. 5, 1994, pp. 525-31.

[24] S. Lakshmivarahan, and S.K. Dhall, "Analysis and Design of Parallel Algorithms Arithmetic and Matrix Problems," McGraw-Hill Publishing Company, 1990.