Efficient Load Balancing Algorithm for the Arrangement-Star Network
Ahmad M. Awwad¹, Jehad Al-Sadi¹

¹CS Dept., University of Petra, Amman, Jordan
²CS Dept., Arab Open University, Amman, Jordan- E-mail

Abstract — the Arrangement-Star is a known network in literature and it is one of the promising interconnection networks for future super computers, it is expected to be one of the attractive alternatives in the future for High Speed Parallel Computers. The Arrangement-Star network 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. In spite that some of the research work has been done on Arrangement-Star promising network, it still needs more time and efforts to be done on the issue of load balancing. 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.

Keywords: Interconnection Networks, Arrangement Network, Star Network, Arrangement-Star, Load balancing.

1. INTRODUCTION

The arrangement-star network as a case of study on vertex product networks [1, 7, 8], 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 [3, 2, 24]. 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 [2]. But still one of the important problems that the arrangement-star network still needs more efforts and researchers time 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 arrangement-star network. In this research efforts 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 [17]. A reasonable and efficient implementation of the ASCDEM algorithm on our network will make the arrangement-star network more attractive for the issue of load balancing problem.

This paper is organized as follows: In the next section we present the necessary basic notations and definitions, in section III introduces the related work on load balancing, section IV presents the implementation of the ASCDEM algorithm on the arrangement-star network, finally section V concludes this research work.

2. DEFINITIONS AND Basic Topological Properties

During the last two decades a big number of interconnection networks for High Speed Parallel Computers (HSPC) investigated and proposed in literature [3, 4, 5]. As an example one of these networks was the hypercube interconnection network [6, 17]. Also a well known example is the star graph [3]. Some properties of this network have been studied in the literature including its basic topological properties, parallel path classification, node connectivity and embedding [10, 11, 13, 14]. 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 [3].

The star graph, however, has few drawbacks [23]. One of the major problems of the star graph is related to its scalability. The size of the star graph increases according to a factorial function, and thus grows widely very rapidly; for example, the value of 7! is equal to 5040 while the value of 11! is about forty million. Despite its attractive topological properties, the star graph has not been used in practical systems yet.

In an attempt to address the scalability problem in the star graph, Day and Tripathi [24] 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 [24]. 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 [13], selection and sorting [14, 22], Fast Fourier Transform [12], and Matrix Multiplications [15] and load balancing. In an attempt to overcome the load balancing problem we present an efficient algorithm for load balancing 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 ≤ k ≤ m. For simplicity let 〈m〉 = {1, 2, ..., m} and 〈k〉 = {1, 2, ..., k}.

**Definition 1:** The (m,k)-arrangement graph Am,k = (V1, E1), 1 ≤ k ≤ m-1 is defined as follows [24]:

\[ V1 = \{p, p_2, ..., p_k \mid p_i \in \langle m \rangle \text{ and } p_i \neq p_j \text{ for } i \neq j \} = P_k^m, \]

\[ E1 = \{(p, q) \mid p \text{ and } q \text{ in } V1 \text{ and for some } i \in \langle k \rangle, p_i \neq q_i \text{ and } p_j = q_j \text{ for } j \neq i \}. \]

That is, the nodes of Am,k labelled with a unique arrangements of k elements out of m symbols (〈m〉), and the edges of Am,k connect arrangements which differ in exactly one of their k positions. An edge of Am,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)-neighbour of p. The (m,k)-arrangement graph Am,k is regular of degree k(m-k) and of size m!/(m-k)!, and diameter \(\lceil 3k/2 \rceil\). The (m, m-1)-arrangement graph Am,m-1 is isomorphic to n-star graph Sn [8, 24], and the (m, 1)-arrangement graph is isomorphic to the complete graph with m nodes [24].

**Definition 2:** The n-star graph, denoted by Sn, has n! nodes each labelled with a unique permutation on 〈n〉 = {1, 2, ..., 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 [3]:

\[ \delta, \text{ of } n \text{-star graph } = \lceil 1.5 (n-1) \rceil \]

\[ \alpha, \text{ of the } n \text{-star graph } = n-1, \text{ where } n>1. \]

**Definition 3:** The arrangement-graph is the cross product of the n-star graph and the (m, k)-arrangement graph, and is given by ASn,m,k = Am,k ⊗ Sn such that n>1 and 1 ≤ k ≤ m.

Note that if G1 and G2 are two undirected graphs then for any node X = ⟨x1, x2⟩ in the cross product graph, G = G1 ⊗ G2, has an address consisting of two parts, one coming from G1 and the other coming from G2. We will denote the earlier part by lp(X) = x1 and the later part by rp(X) = x2.

Figure 1 shows the topology of AS2,3,2 that is obtained from the graph product of S2 and A3,2 networks. A node X = ⟨u, v⟩ in AS2,3,2 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 = ⟨u, v⟩ and Y = ⟨u’, v’⟩ are connected if, lp(X) = lp(Y) and rp(X) is connected rp(Y) in Am,k (in this case X and Y are said arrangement-connected) or rp(X) = rp(Y) and lp(X) is connected lp(Y) in Sn (in this case X and Y are said star-connected). For instance in Figure 1 the node ab13 is connected to the node ab12, and the node ab23 is connected to the node ba23.

3. 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 [2]. 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 infrastructure ranging from electronic networks [2] and also for Optoelectronic networks [20, 21].

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 [6, 16, 17]. 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 nth 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 [18].

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 [19].
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 [19]. In a different literature and research done by Zaho and Xiao they investigated a different algorithm named 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 [6, 18].

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) [6].

4. 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 [17].

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 subgraph, this will be the second level of the hierarchal $AS_{n,m,k}$ network, where each subgraph is an Arrangement graph representation, by the end of the phase 1, all subgraphs 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.

Note that n-1 is the number of neighbors of any processor in $S_n$:
1. for $p1 = 1; p1 <= n-1; p1++$ // Start of phase#1
2. for all neighbour nodes $p1$ and $p2$ which they differ in $1^{st}$ and $n+1$ position of $S_n$ do in parallel
3. Give-and-take $p1$ and $p2$ total load sizes of the two nodes
4. $The Average Load p_{ij} = Floor (Load p_i + Load p_j)/2$
5. if ( Totalload $p_i >= excess Average Load p_{ij}$ )
6. Send excess load $p_i$ to the neighbour node $p2$
7. Load $p2 = Load p2 - extra load$
8. Load $p2 = Load p2 + extra load$
9. $else$
10. Receive extra load from neighbour $p_j$
11. Load $p_j = Load p_j - extra load$
12. Load $p_j = Load p_j + extra load$
13.
14. Repeat steps (1 to 12) \( n!/2 \) times  // End of phase #1
15. for \( p2 = 1; p2 \leq k \times (m-k); p2++ \)  // Start of phase #2
16. for all neighbor nodes \( p_{ui} \) and \( p_{ij} \) which they differ in exactly one \( k \) position of \( A_{m,k} \) do in parallel
17. Give-and-take \( p_{ui} \) and \( p_{ij} \) total load sizes of the two nodes
18. TheAverageLoad \( p_{ui,ij} = \text{Floor} \left( \frac{\text{Load} \ p_{ui} + \text{Load} \ p_{ij}}{2} \right) \)
19. if ( Totalload \( p_{ui} >= \) excess AverageLoad \( p_{ui,ij} \) )
20. Send excess load \( p_{ui} \) to the neighbour node \( p_{ui} \)
21. Load \( p_{ui} = \text{Load} \ p_{ui} - \text{extra load} \)
22. Load \( p_{ij} = \text{Load} \ p_{ij} + \text{extra load} \)
23. else
24. Receive extra load from neighbour \( p_{ij} \)
25. Load \( p_{ui} = \text{Load} \ p_{ui} + \text{extra load} \)
26. Load \( p_{ij} = \text{Load} \ p_{ij} - \text{extra load} \)
27. Repeat steps (15 to 26) \( m!/2(m-k)! \) times  // End of phase #2

Fig. 2: The ASCDEM load balancing Algorithm

ASCDEM algorithm works on redistributing load balancing among all processors of the network, the two phases are done in parallel.

• Phase 1: The load balancing between the processors; subgraphs; of \( S_n \) based on ASCDEM algorithm is exchanged as in steps 2 to 12 in parallel, in first step the load exchange will be between all the processors in which they differ in 1\(^{st}\) position and 2\(^{nd}\) position for all the factor networks of \( S_n \) i.e. \( S_{n-1} \). Then the same process will be repeated continually until it reach the neighbours \( p_{i} \) that is \( n \) positions far away from \( p_{i} \). By the end of this phase all subgraphs will have almost the same total number of load sizes.

• Phase 2: The load balancing within the processors of each subgraph where each subgraph is an \( A_{m,k} \) network. The ASCDEM algorithm in steps 15 to 26 performed in parallel, in 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 \( m!/2(m-k)! \) neighbors. By the end of this phase all nodes of the network will have almost the same load size.

5. 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 we proposed by Mahafza and et al which was 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.
A further work will be done on the proposed algorithm includes: total execution time, efficient load balancing accuracy, latency, number of communication moves and complexity speed to show that the ASCDEM efficiency in terms of mathematically analysis.

References


