Synchronous Distributed Greedy Weighted Graph Matching Algorithms For Wireless Sensor Networks

Ismail Ersin
International Computer Institute
Ege University, Izmir, Turkey
ismail.ersin@yahoo.com

Can Umut Ileri
International Computer Institute
Ege University, Izmir, Turkey
can.umut.ileri@ege.edu.tr

Orhan Dagdeviren
International Computer Institute
Ege University, Izmir, Turkey
orhandagdeviren@ege.edu.tr

***Abstract*—Wireless Sensor Networks (WSN) consist of devices that can communicate with each other without using any fixed infrastructure. These devices can gather necessary information from the environment via their sensors and share collected data with each others. WSNs can be modelled with graphs (G (V,E)) where V is the set of vertices (nodes) and E is the set of edges. Graph theoretical structures such as graph matching can be used to solve various problems such as backup assignment in WSNs. With this aim, we first design a synchronous distributed weighted graph matching algorithm based on Hoepman's algorithm. After this the synchronous algorithm is improved by using overhearing method to design ICO algorithm. Proposed ICO algorithm aims to decrease the transmitted message count for graph matching operation by applying in-network processing. These algorithms are tested on various settings having different node counts and degrees in TOSSIM simulator by comparing with each other. The results of these extensive tests reveal us that ICO is more effective in terms of energy consumption and transmitted bytes.**

***Keywords*—Matching Algorithms, Hoepman, Wireless Sensor Networks, Distributed Algorithms**

I. Introduction

Rapid advances in wireless communications and microelectromechanical systems (MEMS) technologies have enabled the deployment of wireless sensor networks (WSNs) [1]. Today, WSNs are increasingly used in difficult environments such as tunnels, underground mines, outer space and oceans [2]. A WSN can be modeled with a undirected graph $G(V, E)$. In this view, while the connections between the nodes may represent the wireless access channels in which the devices communicate, the devices in the WSN represent the nodes. In this way, it is possible to find effective solutions to many problems that exist in WSNs by operating the graph-theoretic algorithms on WSNss. From a different perspective; practical solutions of graph-theoretical algorithms can be achieved. Graph matching algorithms that is one of these algorithms are used for the various purposes in WSNs. The graph matching problem is one of the most important problems in combinatorial optimization [3]. A matching M can be expressed in a graph $G(V, E)$ as a subset of the edges of G such that no two edges in M are incident to the same vertex [4]. In maximum matching problem, the aim is to maximize the cardinality of matching set. In maximum weighted matching problem each edge has a weight and the total weight of matching set is aimed to maximize. The elements of this subset formed can be the partners used to back up each other's data in WSNs. Fig. 1 shows an example matching operation in a WSN. In this paper

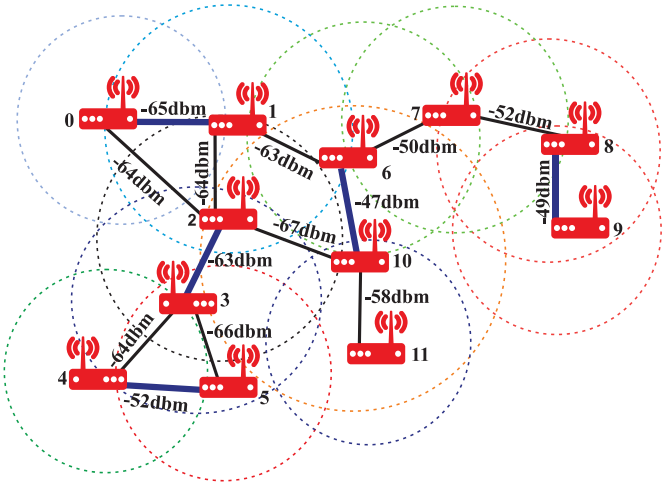


Fig. 1. A Sample Matching Operation in WSN. There are 12 sensor nodes in the network. The ids of nodes are written near of them. Dotted circles show transmission ranges. Each link has a link indicator value based on signal strength. Matched edges are shown with bold lines.

we proposed synchronous distributed greedy weighted graph matching algorithms for wireless sensor networks. Our first synchronous algorithm (Synchronous Hoepman) is based on asynchronous Hoepman algorithm [5]. Later, we improve this algorithm by utilizing overhearing technique and propose ICO algorithm. The algorithms are implemented in TOSSIM simulator by varying node counts and degrees.

The rest of this paper is organized as follows. In Section II, related works are given. Communication model and basic notations are explained in Section III. Proposed approaches are explained in Section IV. Computational results are given in Section V. Conclusions are drawn in Section VII.

II. Related Work

Gabow [6], Preis [7], Pettie and Sanders [8] algorithms are some of the important maximum weighted central algorithms studied in the literature. In addition Uehara and Chen [9], Wattenhofer and Wattenhofer [10], Nieberg [11] algorithms can be given as distributed weighted graph matching algorithms. When it comes to central algorithms, the maximum

978-1-7281-3964-7/19/$31.00 ©2019 IEEE

Algorithm 1: ICO Algorithm

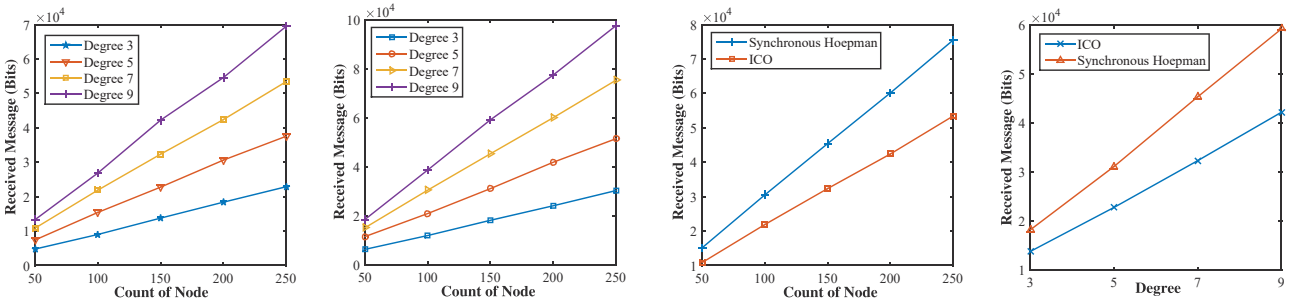
```
INITIALIZE
C ← R ← T ← I ← ∅
R ← {neighbors}
while N > 0 do
    receive message
    src ← message.source
    destination←message.destination
    if message.type=ROUND then
        updateRequestTable()
        if findCandidate()=NULL then
            break
        end if
        if C! = findCandidate() then
            C←findCandidate()
            if C∉T then
                if leafMessage=0 and isLeaf=1 then
                    if C∉T then
                        N← ∅
                    end if
                    send LEAF message to C
                    leafMessage←1
                else if C∈R and leafMessage=0 then
                    send REQUEST (1) to C
                else
                    I←I∪{C}
                    send REQUEST(0) to C
                end if
            else
                C←NULL
            end if
        end if
    else if message.type=REQUEST(m) then
        if destination=myID then
            if src∉R then
                R← R∪{src}
            end if
            if src∈T then
                T← T/{src}
            end if
            if m=1 and μ =1  then
                N← ∅
            end if
        else if m=0 then
            message send another node
            T←T∪{src}
        else if m=1 then
            N← N/{src,destination}
        end if
    else if message.type=DROP then
        N←N/{src}
        if C=src and src∉R then
            C← NULL
        end if
    else if message.type = LEAF then
        if (leafMessage=1 or src∈I) and isLeaf=1 then
            N← ∅
        else if src∉R then
            R←R∪{src}
            if src∈T then
                T ← T/{src}
            end if
        end if
    end if
end while
```

weighted matching problem can be solved in polynomial time. Edmonds' study on this problem [12], [13] is the first central algorithm to provide the exact solution for it and its time complexity is O (n^2m) where n and m denote the number of vertices and edges in the graph. In later studies, the worst case complexity of Edmonds' algorithm was gradually reduced. The time complexity of the problem has been reduced in the Lawler's [14] and Gabows [6] to O (n^3), and in the Galil at al. [15] to O($nmlog(n)$) and in the Gabows study [16] to O ($n(m+nlog(n))$). Although a precise distributed polynomial time solution of the problem has not yet been found, many approximation algorithms have been proposed. The algorithm, proposed by Israeli and Itai [17] to maximize the number of matching nodes in unweighted graphs, is still known as the best algorithm in the worst case. The algorithm is basically based on the idea of determining a random subgraph with the degree of each node being less than 2, and then finding the matching one by making random selections on this subgraph. The subgraph is found as follows: Each node randomly selects one of its neighbors and sends a proposal to it. Then each node accepts one of the coming offers randomly. In this case, a maximum of two edges of a node is selected. Selected edges create a subgraph. Then each node randomly selects one of its edges in the subgraph. If an edge is selected by both nodes that is connected to, it is assumed to be in the matching edge set. Each node performs the same steps until it matches another node or has no neighbors to match. Wattenhofer and Wattenhofer proposed a distributed algorithm [10] that applies the aforementioned unweighted matching algorithm on weighted graphs. This algorithm guarantees a 1/5 ratio approach to the highest appropriate weight. In contrast to this random algorithm, Hoepman [5] presented a greedy deterministic algorithm, which is the basis of our study. Accordingly, each node selects the maximum weighted edge between the edges attached to it and sends a REQUEST message to the neighbor to which it is connected. If any node receives a REQUEST message from the neighbor that is sent REQUEST, these two nodes are considered matched at any time of the algorithm. In this way, it is ensured that the edges with the maximum weight are removed from the graph at the time the first REQUEST messages are sent. Considering that each step will be run in this way, it is guaranteed that there is a solution in the worst case as many steps as the number of edges. The performance evaluation given in Ileri and Dagdeviren [18] has showed that the approximation ratio is over 95% in both "small-world" and geographical random networks. It does this with shorter runtime, using relatively few messages than other algorithms.

III. Communication Model and Basic Notations

For distributed weighted matching algorithms to run on the WSNs, a node must learn the weight values of its neighbors. In practice, the weight value between a node and its neighbor is taken as 1 / distance value. The following assumptions were made for the nodes in which the algorithms are running:

- Each node has a unique node number.
- The connections between the two nodes are symmetrical.
- Each node knows its neighbors that have one hop distance and weights with these neighbors.
- Each node can send multicast messages to its neighbors.



(a) ICO Algorithm (b) Synchronous Hoepman (c) Average Degree=7 (d) Count of Nodes=150

Fig. 2. Received message size with respect to the average degree and count of nodes

- In the algorithm, ROUND message is expressed as the time interval (TIN) in the time division multiple access (TDMA) protocol.

IV. Proposed Approaches

A. Synchronous Hoepman

The algorithm starts with a ROUND message for every node. In the beginning, every node sends REQUEST message to its candidate node. If nodes match, they send a DROP message to their neighbors. If an unmatched node receives a DROP message from its candidate then it changes its candidate. In the next round, it sends a REQUEST message to its candidate. Each node initially acquires knowledge of its weights with its neighbors. The C variable contains the candidate to which the node matches, the N variable stores the list of the neighbors of the node and their weights. The list of adjacent nodes that send a matching message to the node during the period until the end of the algorithm is stored in the R variable. The findCandidate (N) method finds the neighbor with the largest edge weight in the remaining neighbors of the node. There are three kind of messages in Synchronous Hoepman Algorithm;

- ROUND**:** If a node receives a new ROUND message, the candidate variable (C) is set to a new candidate if it is not equal to the candidate variable specified by the findCandidate (N) method, and the node sends a REQUEST message to that candidate.
- REQUEST**:** A node sends a REQUEST message to the node (candidate) it wants to match. The REQUEST message consists of 3 fields: the destination node ID (8 bits), source node ID (8 bits), and message type (1 bit).
- DROP**:** If a node receives a REQUEST message from the node which the REQUEST message is sent, they are matched. Then the matched nodes sends a DROP message to its neighbors to inform matching knowledge. We added a control mechanism that understands whether DROP's source is its partner. DROP message consists of 2 fields, source ID (8 bits) and message type (1 bit).

B. ICO Algorithm

ICO algorithm uses overhearing mechanism. In the ICO algorithm, each node in WSN listen messages in order to decide its next action. This decision-making procedure mechanism significantly reduces the number of sent messages. ICO algorithm is given in Algorithm 1. Messages are listed below.

There are four kinds of messages in the ICO Algorithm;

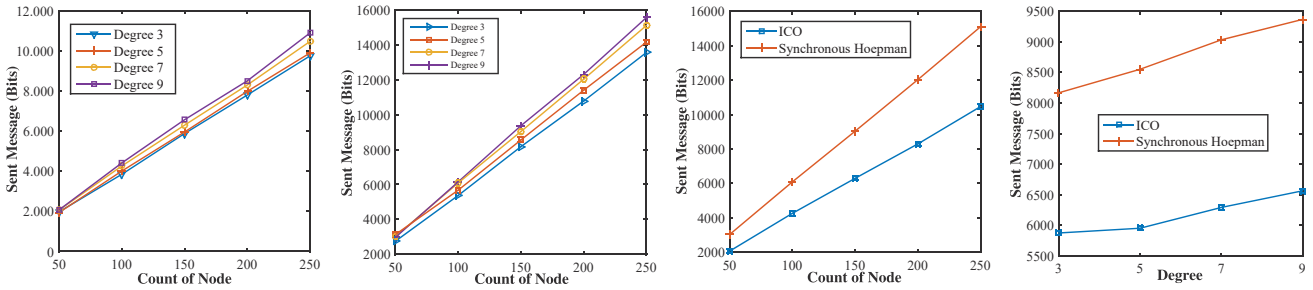
- REQUEST (0): A node sends a REQUEST message to the node (candidate) it wants to match. This message is similar to the REQUEST message in Synchronous Hoepman. It consists of two fields: destination node ID (8 bits), message type (2 bits).
- REQUEST **(1):** If a node has received a LEAF or a REQUEST(0) message from its candidate, it sends the REQUEST (1) message to its candidate. This message contains a drop message in the structure. Hence, the nodes that are sending this message do not need to send DROP messages. Nodes that receive this message must act as if they received a DROP message. This message consists of three fields: destination node ID (8 bits), source node ID (8 bits) and message type (2 bits).
- DROP**:** If the node has matched and has neighbors except the matching node, it informs other neighbors about match status via DROP message. This message consist of two fields: source node ID (8 bits) and message type (2 bits).

V. Computational Results

We implemented synchronous Hoepman algorithm and ICO algorithm on TOSSIM simulator. In two algorithms, the results obtained in the simulation environment were evaluated in terms of the size of transmitted messages, the running time of the algorithm and amount of the energy used.

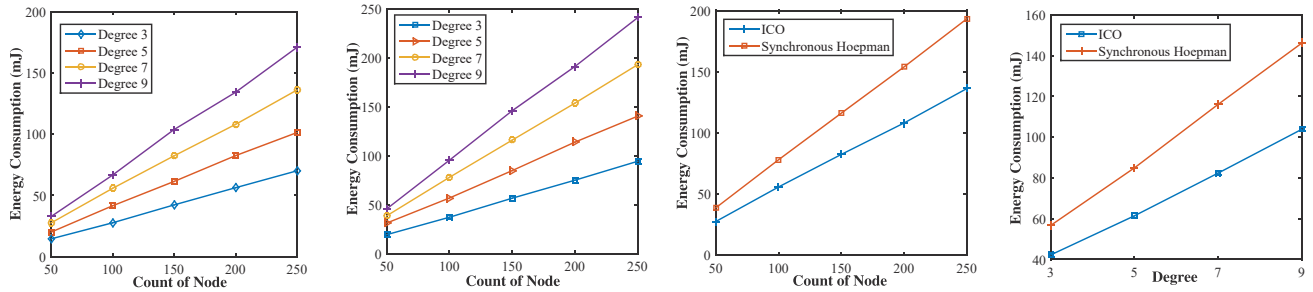
Received Message Size: One way to calculate the energy consumption of a node in WSNs is to calculate the message size received by the node because the energy consumption depends on the size of the total received and sent messages. In Fig. 2a and 2b the total received message size of the ICO and Synchronous Hoepman against the node degree and count is shown. When the node count is increased, the total received message size of ICO and Synchronous Hoepman increases linearly. As the degree of the graph increases, the number of nodes that receives a sent message increases. That's why the received message size increases. A comparison of the ICO and Synchronous Hoepman algorithms to the degree is given in Fig. 2d. The total size of the received message of the ICO is the smaller than Synchronous Hoepman. When we examine Fig. 2c, we see that the number of nodes and the size of the received messages increase linearly. When the results are analyzed, it is concluded that the ICO algorithm is 1.40 times better than the Synchronous Hoepman in terms of the received message size.

Sent Message Size. Secondly, we measured the size of sent message of the algorithms. The size of the sent message is



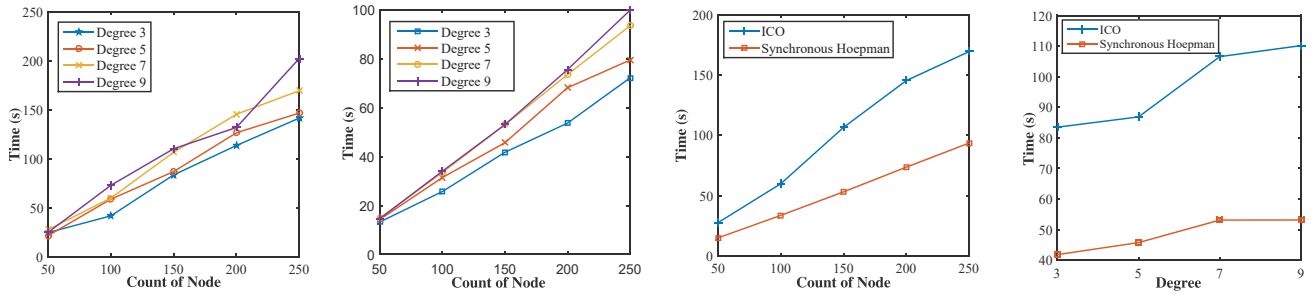
(a) ICO Algorithm (b) Synchronous Hoepman (c) Average Degree=7 (d) Count of Nodes=150

Fig. 3. Sent message size with respect to the average degree and count of nodes



(a) ICO Algorithm (b) Synchronous Hoepman (c) Average Degree=7 (d) Count of Nodes=150

Fig. 4. Energy consumption size with respect to the average degree and count of nodes



(a) ICO Algorithm (b) Synchronous Hoepman (c) Average Degree=7 (d) Count of Nodes=150

Fig. 5. Wallclock time with respect to the average degree and count of nodes

an important criterion for energy consumption. The total size of sent message of the ICO, Synchronous Hoepman against the node degree is shown in Fig. 3a, Fig. 3b, respectively. The size of send message of the ICO and Synchronous Hoepman increase linearly. Fig. 3d shows a comparison of the ICO and Synchronous Hoepman algorithms against the degree. Likewise Fig. 3c shows a comparison of implemented algorithms against the node count. The total size of the sent message of the ICO against the degree and count of node is the smallest among the other algorithms. ICO outperforms other algorithm, and is approximately 1.43 times better than Synchronous Hoepman.

Energy Consumption. We measured the energy consumption of algorithms. In practice, the amount of energy consumption by a node is closely related to the size of the message that the node has sent and received. In this respect, the analysis of the above figures gives us the idea that the ICO algorithm is more efficient in terms of energy consumption than other algorithms. The average size of energy consumption of the algorithms against the node degree and node count is shown respectively in Fig. 4a, Fig. 4b. The energy consumption of the algorithms increases linearly against node degree and node count. Fig. 4d and 4c show a comparison of the algorithms against the average degree and node count respectively. These figures show that ICO is more efficient than other algorithm. As a result, in terms of energy consumption, the ICO algorithm is 1.43 times better than the Synchronous Hoepman.

Wallclock Time. Finally, we measured the wallclock times of ICO and the other algorithm. Fig. 5a and Fig. 5b shows the wallclock times of the ICO, Synchronous Hoepman against the node degree and node count. The wallclock times of algorithms increases with the number of nodes and degrees. Due to the overhearing mechanism of the ICO algorithm, it has a larger wallclock time than Synchronous Hoepman. When Fig. 5c, 5d are analyzed, the ICO algorithm has 2 times worse wallclock time than synchronous Hoepman.

VI. Conclusion

In this study, weighted graph matching in WSNs is studied. Two algorithms are proposed in this manner where the first algorithm (Synchronous Hoepman) is based on Hoepman's weighted asynchronous graph matching algorithm. The second algorithm (ICO) uses overhearing method to reduce the number of transmitted messages during matching operation which will lead to consume less energy.

Proposed algorithms have been analyzed in terms of the running time, transmitted message size and the energy consumption. The results of these tests showed that ICO is more effective in terms of energy consumption and transmitted byte counts than Synchronous Hoepman. It is observed that ICO has up to 44% improvement in energy consumption on WSNs while it keeps wallclock time close to the Synchronous Hoepman algorithm.

Acknowledgment

Authors would like to thank TUBITAK for the project grant 215E115.

References

[1] J. Zheng and A. Jamalipour, *Wireless Sensor Networks: A Networking Perspective*. Wiley-IEEE Press, 01 2009.

[2] O. Yilmaz, O. Dagdeviren, and K. Erciyes, "Localization-free and energy-efficient hole bypassing techniques for fault-tolerant sensor networks," *J. Netw. Comput. Appl.*, vol. 40, pp. 164–178, Apr. 2014.

[3] R. Duan and S. Pettie, "Linear-time approximation for maximum weight matching," *J. ACM*, vol. 61, pp. 1:1–1:23, Jan. 2014.

[4] D. E. Drake and S. Hougardy, "Improved linear time approximation algorithms for weighted matchings," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques* (S. Arora, K. Jansen, J. D. P. Rolim, and A. Sahai, eds.), (Berlin, Heidelberg), pp. 14–23, Springer Berlin Heidelberg, 2003.

[5] J. Hoepman, "Simple distributed weighted matchings," *CoRR*, vol. cs.DC/0410047, 2004.

[6] H. N. Gabow, "An efficient implementation of edmonds' algorithm for maximum matching on graphs," *Journal of the ACM (JACM)*, vol. 23, no. 2, pp. 221–234, 1976.

[7] R. Preis, "Linear time 1/2 -approximation algorithm for maximum weighted matching in general graphs," in *Proceedings of the 16th Annual Conference on Theoretical Aspects of Computer Science*, STACS'99, (Berlin, Heidelberg), pp. 259–269, Springer-Verlag, 1999.

[8] S. Pettie and P. Sanders, "A simpler linear time $2/3 - \epsilon$ approximation to maximum weight matching," *Information Processing Letters*, vol. 91, no. 6, pp. 271–276, 2004.

[9] R. Uehara and Z. Z. Chen, "Parallel approximation algorithms for maximum weighted matching in general graphs," in *Theoretical Computer Science: Exploring New Frontiers of Theoretical Informatics* (J. van Leeuwen, O. Watanabe, M. Hagiya, P. D. Mosses, and T. Ito, eds.), (Berlin, Heidelberg), pp. 84–98, Springer Berlin Heidelberg, 2000.

[10] M. Wattenhofer and R. Wattenhofer, *Distributed Computing: 18th International Conference, DISC 2004, Amsterdam, The Netherlands, October 4-7, 2004. Proceedings*, ch. Distributed Weighted Matching, pp. 335–348. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004.

[11] T. Nieberg, "Local, distributed weighted matching on general and wireless topologies," in *Proceedings of the Fifth International Workshop on Foundations of Mobile Computing*, DIALM-POMC '08, (New York, NY, USA), pp. 87–92, ACM, 2008.

[12] J. Edmonds, "Maximum matching and a polyhedron with 0, 1-vertices," *Journal of Research of the National Bureau of Standards B*, vol. 69, no. 125-130, pp. 55–56, 1965.

[13] J. Edmonds, "Paths, trees, and flowers," *Canadian Journal of mathematics*, vol. 17, no. 3, pp. 449–467, 1965.

[14] E. L. Lawler, *Combinatorial optimization: networks and matroids*. Courier Corporation, 1976.

[15] Z. Galil, S. Micali, and H. Gabow, "An o(ev\logv) algorithm for finding a maximal weighted matching in general graphs," *SIAM Journal on Computing*, vol. 15, no. 1, pp. 120–130, 1986.

[16] H. N. Gabow, "Data structures for weighted matching and nearest common ancestors with linking," in *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '90, (Philadelphia, PA, USA), pp. 434–443, Society for Industrial and Applied Mathematics, 1990.

[17] A. Israeli and A. Itai, "A fast and simple randomized parallel algorithm for maximal matching," *Information Processing Letters*, vol. 22, pp. 77–80, 01 1986.

[18] C. U. Ileri and O. Dagdeviren, "Performance evaluation of distributed maximum weighted matching algorithms," in *Digital Information and Communication Technology and its Applications (DICTAP), 2016 Sixth International Conference on*, pp. 103–108, IEEE, 2016.