Volume 2 No. 6, June 2012 ISSN 2223-4985

International Journal of Information and Communication Technology Research

http://www.esjournals.org

The Multi-Constrained Dynamic Programming Problem in View of Routing Strategies in Wireless Mesh Networks

Sabreen M. Shukr, Nuha A. S. Alwan, Ibraheem K. Ibraheem
Department of Computer Engineering, College of Engineering, University of Baghdad
Baghdad, Iraq

ABSTRACT

We address the problem of satisfying multiple constraints in a shortest-path Dijkstra-based dynamic programming (DP) framework. We solve this problem considering its applicability to routing in wireless mesh networks. Our work, therefore, relates to quality-of-service (QoS) metrics such as bandwidth, delay and the number of hops between network nodes. The present work is based on the optimization of one or more metrics while bounding the other. Specifically, we compute high-capacity paths while simultaneously bounding the end-to-end delay and the number of hops to an upper limit. We prospectively call the resulting algorithm the comprehensive mesh routing algorithm (CMRA). We show by computer simulations that the proposed algorithm features wide versatility when dealing with multiple constraints as compared to other approaches involving the optimization of a weighting function. This latter technique is not easily applicable to wireless networks due to different networking metrics having values with widely different orders of magnitude.

Keywords: *Dynamic programming, Dijkstra algorithm, routing, wireless networks, quality of service (QoS) measures, multi-constrained optimization.*

1. INTRODUCTION

Dynamic programming [1-3] (DP) is a very powerful algorithmic paradigm in which a problem is solved by identifying a collection of subproblems and tackling them one by one, smallest first, using the answers to small problems to help figure out larger ones, until the whole lot of them is solved [3]. Thus, dynamic programming is flexible and significantly reduces the complexity of the problem. Although based on profound theory the numerical computations are simple and full-fledged. DP is a technique of very broad applicability that can be adopted in applications where other specialized or application-specific methods fail. For example, evolutionary algorithms such as genetic algorithms (GA) are most appropriate for complex non-linear models where the location of a global minimum is a difficult task. Due to global search, GAs are computationally expensive and need more time to be computed as compared with dynamic programming [10].

At the heart of dynamic programming lies the problem of finding the shortest path in a directed graph. Let G = (V, E) be the directed graph under consideration, where V is the set of vertices and E is the set of edges between vertices. Edges may be bidirectional and each has a cost assigned to it. The shortest path between any two vertices is that consisting of consecutive edges, the overall cost of which is minimized. The individual costs in a path may be additive to compute the overall path cost, or they can be compared to find their minimum or maximum value and assign this value to the overall path. In some cases where the edge cost represents probability of error, for instance, the edge costs are multiplied [4] to find the probability of error of the corresponding path.

There are two fundamental processes to solve the shortest path problem in a multi-stage directed graph. These are termed forward and backward dynamic programming. These assume a unidirectional graph. For a graph with bidirectional edges, there are other applicable single-constraint DP algorithms such as the Dijkstra, Bellman-Ford and Floyd-Warshall techniques [5, 11]. These algorithms differ in certain aspects (the processing time and the amount of information that must be collected from other nodes [5]) making each of them advantageous or preferable for a specific application. Constrained shortest path problems and mutli-constrained optimization for path selection problems are highly challenging and have been proved to be NP-complete, which means that they are all solved in polynomial time [3, 6]. A problem is NP-complete if all decisions for the problem can be verified in polynomial time [3].

The routing problem is a multi-objective optimization problem with metrics such as path rate or capacity, end-to-end delay, and hop count. The path rate is the number of bits per second (bps) that can be sent along the path connecting the source node to the destination node. The end-to-end delay and path rate are typical QoS requirements that need to be bounded and optimized. Another commonly used metric is the minimum hop count. However, in a dense network, there may be many routes of the same minimum length, with widely varying qualities. Therefore, this metric alone is not likely to select the best path [7]. In general, the QoS metrics need to be optimized by making meaningful trade-offs due to their inter-conflicting nature.

The mesh routing algorithm (MRA) [8,9] is a dynamic programming approach to compute high-capacity end-to-end delay bounded paths. In contrast, weighted-sum techniques that simultaneously minimize the end-to-end delay and maximize the path capacity by optimizing an objective function are quite complex. The latter techniques optimize the objective function $f = \beta_1.(end-to-end-delay) + \beta_2.(path-capacity)$.

The complexity is mainly due to the fact that both objectives take different orders of magnitude [8], and solutions

International Journal of Information and Communication Technology Research

http://www.esjournals.org

that best satisfy QoS requirements are not guaranteed although there exists research [12] on the use of genetic algorithms for multi-objective optimization for QoS routing employing objective functions despite the afore-mentioned shortcomings of this approach. In this work, we introduce a more comprehensive metric than that of the MRA in that it satisfies the additional constraint of bounding the hop count while maximizing path rate and limiting end-to-end delay. Moreover, our algorithm is different from the MRA in that it is Dijkstra-based. It therefore retains all the advantages of the Dijkstra algorithm [5], such as fast shortest path determination, and having an order of n^2 (n is the number of nodes) rendering it efficient enough to use for relatively large networks. The remainder of the paper is organized as follows: Section 2 presents first a two-constrained Dijkstra algorithm, and Section 3 presents the proposed three-constrained algorithm which we call the comprehensive mesh routing algorithm (CMRA). Section 4 summarizes the simulation results, and finally, Section 5 concludes the paper.

2. TWO-CONSTRAINED DIJKSTRA ALGORITHM

A Dijkstra-based dynamic programming technique to optimize the capacity of paths and at the same time bounding end-to-end delay is presented to determine shortest route in the wireless mesh network. Let G=(V,E) be a graph representing a wireless mesh network, where V is the set of nodes and E is the set of links. The end-to-end delay of a path P is defined as:

$$D(P) = \sum_{l \in P} t(l) \qquad (1)$$

where $t(l)$ is the expected delay through link *l* in Path *P*, and $l \in E$. The capacity or rate of a path *P* is defined as:

$$R(P) = \min_{l \in P}\{r(l)\} \qquad (2)$$

where *r(l)* is the rate of link *l*.

Figure 1 illustrates how the algorithm operates in a simple network where the rate in Mbps is shown over each link, and the delay $t(l)$ in ms is shown below each link. Denote the path from u to y with maximum capacity *R* and with delay exactly equal to 3ms, as $P_3^*(u,y)$. The way to reach y is either direct or through w.

$$R(P_3^*(u,y)) = \max\begin{cases}\min\{R(P_{3-1.5}(u,w)),6\} \\ \min\{R(P_{3-0}(u,y)),\infty\}\end{cases} \qquad (3)$$

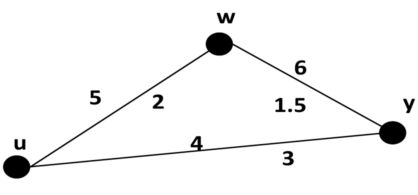

Figure 1: An example of a network with rate *r(l)* and delay *t(l)* shown on each link.

Since there is no path $P_{3-1.5}(u,w)$,(i.e. path from u to w with delay 1.5ms), then the only path remains to be considered as the maximum capacity path is the path that directly connects *u* and *y* ($P_3(u,y)$). If the end-to-end delay bound is less than 3ms, the two paths are disregarded. In general, finding a maximum-capacity path whose end-to-end delay is less than τ can be achieved by evaluating $P_\tau^*(vs,vd)$ where *vs* is the source node and *vd* is the destination node (see Figure 2). Beginning with the source node (*vs*), the algorithm finds node (*u*) whose *R(P(vs,u))* is the maximum capacity among all nodes. After that the algorithm finds the links (l_{uv}) that connect *u* and *v* for all *v* provided that the delay from *vs* to *v* does not exceed τ. So the path from *vs* to *v* is either $P(vs,v)$ or $P(vs,u) \oplus l_{uv}$ depending on which has the maximum capacity. The path from *vs* to *v* throught *u* has a rate equal to *min*{*R(P(vs,u)), r(l_{uv})*}.

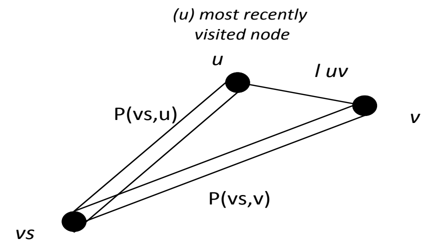

Figure 2: The path from *vs* to *v*.

Figure 2 explains how the algorithm works. The paths *P(vs,u)* and *P(vs,v)* are not necessarily disjoint. If $R(P(vs,u) \oplus l_{uv}) > R(P(vs,v))$ then $P(vs,v)$ is $P(vs,u) \oplus l_{uv}$.

By the same way we can expand the path to reach the destination node (*vd*) and find $P_\tau^*(vs,vd)$ and $R(P_\tau^*(vs,vd))$ denoting the shortest path (greatest capacity) and its rate respectively.

The algorithm in Table 1 is used to return a maximum-capacity path such that the end-to-end delay is bounded by τ and its rate.

Table 1: The Two-Constrained Dijkstra-based Algorithm

```
INPUT: no. of nodes n, source node vs, destination node vd, delay
bound τ, t(l) for all l, r(l) for all l.
OUTPUT: P*τ(vs,,vd) , R(P*τ(vs,,vd)).
/*Initialization*/
FOR ALL nodes
    Visited nodes = NIL
    R(P(vs,node))=0
    D(P(vs,node))=∞
    Parent(node)= NIL
END
R(P(vs,vs))=∞
D(P(vs,vs))=0
FOR i=1: (n-1)
    FOR ALL visited nodes
        Rate(node) = R(P(vs,node))
    END
```

International Journal of Information and Communication Technology Research

http://www.esjournals.org

```
    r_max = max[rate(node)]
    u=node corresponding to r_max .
    /* u is the visited node */
    FOR ALL nodes v
        IF (t(l_uv) + D(P(vs,u))) < τ THEN
            IF r(l_uv) < R(P(vs,u)) THEN
                cap=r(l_uv)
            ELSE cap= R(P(vs,u))
            END IF
            IF cap> R(P(vs,v)) OR D(P(vs,v)) > τ THEN
                R(P(vs,v)) = cap
                parent(v) = u
                /* P(vs,v) = P(vs,u) ⊕ l_uv */
                D(P(vs,v)) = D(P(vs,u)) + t(l_uv)
            ELSE IF D(P(vs,v))> τ THEN
                    R(P(vs,v)) = 0
                    D(P(vs,v)) = ∞
            END IF
        END IF
    END FOR
    IF parent(vd) ≠ NIL THEN
        P(vs,v) = [vd]
        t = vd
        WHILE t ≠ vs
            p = parent(t)
            P(vs,vd) = [p P(vs,vd)]
            t = p
        END
    END IF
END FOR
P*_τ(vs, , vd) = P(vs,vd)
RETURN P*_τ(vs, , vd) , R(P*_τ(vs, , vd))
```

3. THE PROPOSED CMRA ALGORITHM

The CMRA algorithm is an improvement over the two-constrained Dijkstra algorithm in that it satisfies the additional constraint of bounding the hop count while maximizing path capacity and maintaining a bounded end-to-end path delay.

The network in Fig 3 is used to explain how the CMRA algorithm operates. Let's assume that the source and destination nodes are c and i respectively, with the link rates in Mpbs over each link, and the link delays in ms below each link. The route from c to i is either through h or i. The maximum path capacity from c to i through a is 6 Mpbs with delay equal to 8ms and hop count equal to 4. If the allowed maximum hop count is less than 4 and the delay is bounded by 5ms or less, the path through a is disregarded. So the remaining path is through h with maximum capacity equal to 5Mpbs and delay 3ms and hop count equal to 2.

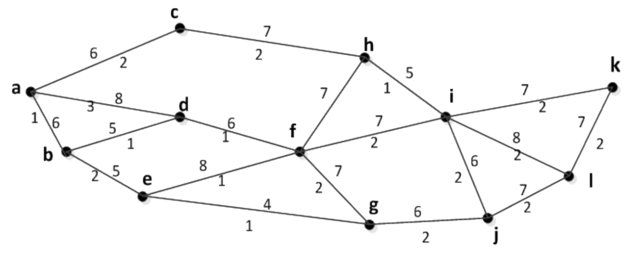

Figure 3: An example of a network with rate *r(l)* and delay *t(l)* shown on each link.

$P^*_{\tau,H}(vs,vd)$ is the path with maximum capacity, and end-to-end delay and maximum hop count bounded by τ and H respectively. This path can be determined by implementing the algorithm in Table 2.

Table 2: The CMRA Algorithm

```
INPUT: no. of nodes n, source node vs, destination node vd, delay
bound τ, H, t(l) for all l, r(l) for all l, h(l) for all l.
OUTPUT: P*_τ,H(vs, , vd) , R(P*_τ,H(vs, , vd)).
/*Initialization*/
FOR ALL nodes
    Visited nodes = NIL
    R(P(vs,node))=0
    D(P(vs,node))=∞
    H(P(vs,node))=∞
    Parent(node)= NIL
END
R(P(vs,vs))=∞
 D(P(vs,vs))=0
 H(P(vs,vs))=0
FOR i=1: (n-1)
    FOR ALL visited nodes
        Rate(node) = R(P(vs,node))
    END
    r_max = max[rate(node)]
    u=node corresponding to r_max .
    /* u is the visited node */
  FOR ALL nodes v
      IF (t(l_uv) + D(P(vs,u))) < τ THEN
          IF (h(l_uv) + H(P(vs,u))) < H THEN
              IF r(l_uv) < R(P(vs,u)) THEN
                  cap=r(l_uv)
              ELSE cap= R(P(vs,u))
              END IF
              IF cap> R(P(vs,v)) OR D(P(vs,v)) > τ OR
                H(P(VS,V)) > H THEN
                R(P(vs,v)) = cap
                parent(v) = u
                /* P(vs,v) = P(vs,u) ⊕ l_uv */
                D(P(vs,v)) = D(P(vs,u)) + t(l_uv)
                H(P(vs,v)) = H(P(vs,u)) + h(l_uv)
              ELSE IF D(P(vs,v)) > τ OR H(P(vs,v)) > H
THEN
                  R(P(vs,v)) = 0
                  D(P(vs,v)) = ∞
                  H(P(vs,v)) = ∞
              END IF
          END IF
      END IF
  END FOR
  IF parent(vd) ≠ NIL THEN
      P(vs,v) = [vd]
      t = vd
      WHILE t ≠ vs
          p = parent(t)
          P(vs,vd) = [p P(vs,vd)]
          t = p
      END
END IF
P*_τ,H(vs, , vd) = P(vs,vd)
RETURN P*_τ,H(vs, , vd) , R(P*_τ,H(vs, , vd))
```

Volume 2 No. 6, June 2012 ISSN 2223-4985

International Journal of Information and Communication Technology Research

http://www.esjournals.org

4. RESULTS AND DISCUSSION

The simulation program to implement the two-constrained Dijkstra and the CMRA algorithms was coded in Matlab 8.0. The resultant shortest route depends on link rate, end-to-end delay and hop count. The simulation model parameters are chosen as follows: Number of nodes in the simulated network= 50. Topology area: Nodes are distributed randomly on 1000*1000 m^2. This network topology ensures that the node coverage area is 200 *m*. Thus, some nodes may be in the coverage area of others.

Figures 4, 5 and 6 show the topology of the network and the shortest route in terms of maximum capacity from node 23 to node 24 in Dijkstra with one constraint, two constraints and CMRA with three constraints respectively. The route in Figure 4 has the largest capacity (8.6788 Mbps) among the three routes in the three figures because there are no imposed constraints other than optimizing (maximizing) the capacity. The route in Figure 5 has a capacity of 8.3068 Mbps as it is bounded by a 50 *ms* delay. The route in Figure 6 is bounded with 50 *ms* delay and maximum hop count equal to 8 so it resulted in a capacity equal to 8.2614 Mbps.

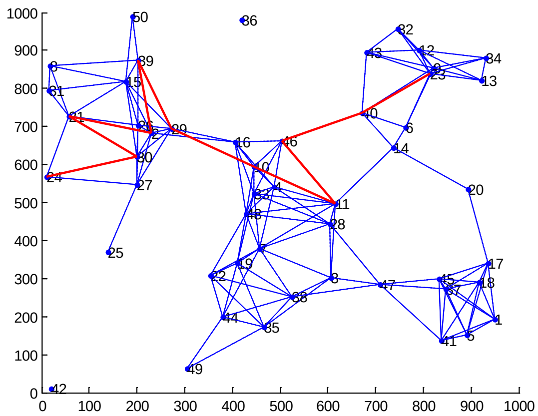

Figure 4: Network Topology showing Shortest Route under Single-Constraint Dijkstra Algorithm

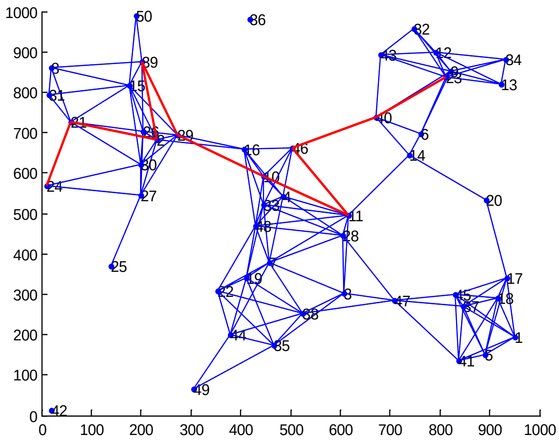

Figure 5: Network Topology Showing Shortest Route Under Two-Constrained Dijkstra Algorithm

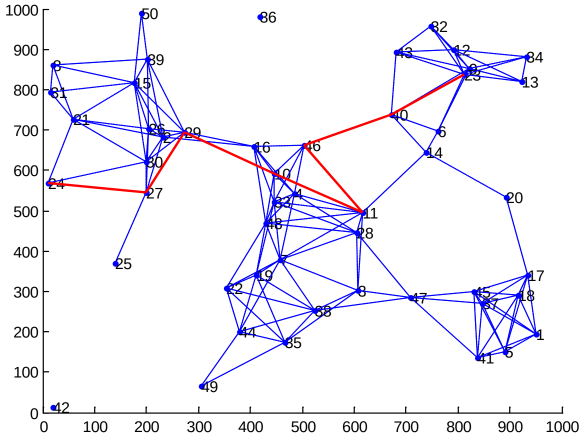

Figure 6: Network Topology Showing Shortest Route Under Three-Constrained Dijkstra Algorithm

From Figure 7, a comparison can be made between the capacities in the three algorithms on three different paths of the same network of Figure 4. The figure shows that the Dijkstra algorithm always has the highest capacity because there are no additional constraints on it. In paths 1 and 3, the capacities differ between the Dijkstra or two-constrained Dijkstra algorithms and the CMRA algorithm. This indicates that we can still have an almost optimized capacity despite the three imposed constraints of the CMRA algorithm. In path 2, the capacities in two-constrained Dijkstra and CMRA are the same, and are slightly less than that of the Dijkstra algorithm despite the additional constraints also.

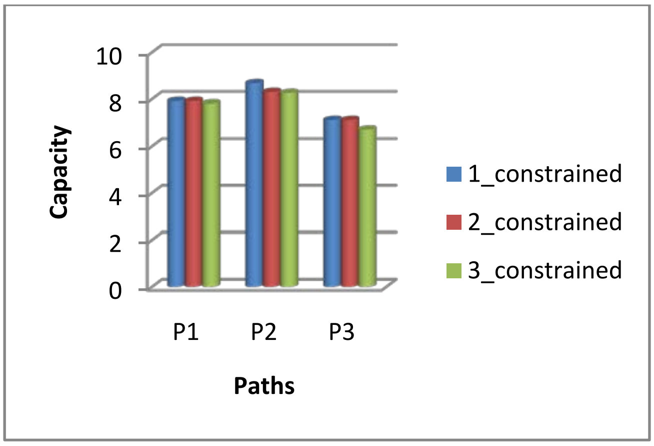

Figure 7: Capacity of Different Paths in Dijkstra, Two-Constrained Dijkstra and CMRA.

The relationship between delay and capacity in the two-constrained Dijkstra algorithm appears in Figure 8. The relationship shows that when the delay increases the capacity increases too and this is readily discernible because as the delay bound increases, more possible different routes with higher capacity can be found.

Volume 2 No. 6, June 2012 ISSN 2223-4985
International Journal of Information and Communication Technology Research

http://www.esjournals.org

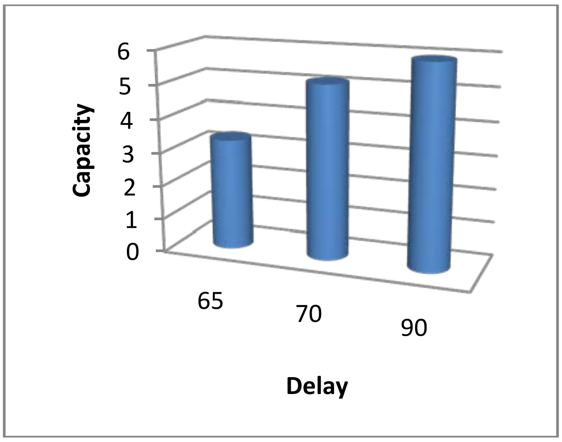

Figure 8: Capacity (in Mbps) with Different Delays in (ms) in the Two-Constrained Dijkstra Algorithm.

Figure 9 shows that when we take different delay bounds and different maximum hop count bounds in the CMRA algorithm for the same path, we get the following result. Case 1 is for a delay bound less than the delay bound in Case 2 but with larger number of hops. The result shows that the capacity in Case 1 is higher than capacity in Case 2 despite the smaller delay of Case 1. This, of course, is possible due to the larger number of hops. These advantageous results appear in many paths as shown in the Figure 9.

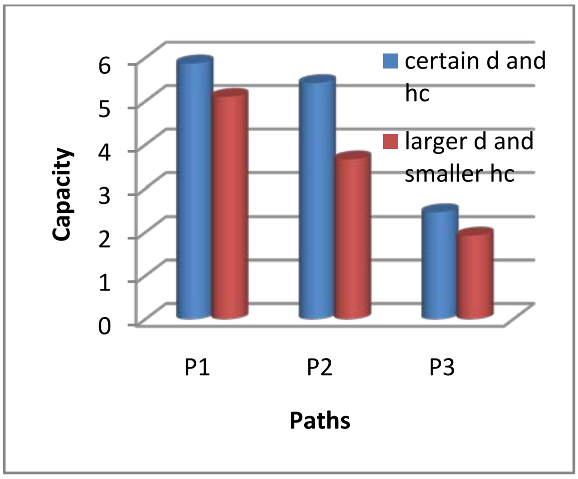

Figure 9: Capacity of Different Paths in CMRA Algorithm, d indicates delay and hc indicates hop count.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we present a solution to the problem of finding shortest paths with high capacity while bounding the end-to-end delay and hop count to desired values for wireless mesh networks. The proposed CMRA algorithm provides an alternative and more suitable approach to genetic algorithm with weighted sum technique [12] through considering multiple constraints in a shortest-path Dijkstra-based dynamic programming method. QoS measures comprising path capacity, end-to-end delay and hop count are considered simultaneously. We use Matlab simulation to realize our network and assign link capacities and delays randomly. Simulation results show that the CMRA algorithm features wide versatility when dealing with multiple constraints. As a future work, we plan to derive and simulate algorithms similar to the CMRA for energy-efficient routing in wireless sensor networks, where it is beneficial to route data through a multi-hop route to achieve energy conservation [13].

REFERENCES

[1] A. Lew, H. Mauch, “Dynamic Programming: A Computational Tool”, Springer-Verlag, Berlin Heidelberg, 2007.

[2] S. Sitarz, “Dynamic programming with ordered structure: Theory, examples, applications,” Fuzzy Sets and Systems, vol. 161, pp 2623-2641, March, 2010.

[3] S. Dasgupta, C. Papadimitriou, U. Vazirani "Algorithms", McGraw-Hill, 2008.

[4] T. K. Moon, W. C. Stirling, "Mathematical Methods and Algorithms for Signal Processing", Upper Saddle River, New Jersey: Prentice-Hall, 2000.

[5] W. Stallings, “Data and Computer Communications,” Prentice Hall, Ninth Edition, 2011.

[6] G. Allard, Ph. Jacquent, ”Heuristic for bandwidth reservation in multihop wireless networks,” INRIA Rocquencourt, No. 5075, January, 2004.

[7] D. S. J. De Couto, D. Aguayo, J. Bicket, R. Morris, "A high throughput path metric for multi-hop wireless routing", MobiCom'03, September, 14-19, 2003, San Diego, California, USA.

[8] J. Crichigno, J. Khoury, M. Y. Wu, W. Shu, “A dynamic programming approach for routing in wireless mesh networks,” IEEE Globecom 2008 Proceedings.

[9] J. B. Helonde, V. Wavhai, V. S. Deshpande, L. B. Bhagwat, "Performance analysis of mesh routing algorithm for routing metrics link capacity and delay against inter-arrival time dependencies', 2011.

[10] B. Doerr, A. Eremeev, C. Horoba, F. Neumann, M. Theili, "Evolutionary Algorithms and Dynamic Programming", GECCO'09: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, ACM, New York, NY, USA, pp. 771-778, 2009.

[11] S. Glisic, B. Lorenzo, "Advanced Wireless Networks", Second Edition, Wiley, 2009.

[12] A. Barolli, E. Spaho, F. Xhafa, L. Barolli, M. Takisawa, "Application of GA and Multi-objective Optimization for QoS Routing in Ad-hoc Networks", 2011 International Conference on Network-Based Information Systems.

[13] A. E. Khandani, J. Abounadi, E. Modiano, L. Zheng, "Cooperative Routing in Static Wireless Networks", IEEE Transactions on Communications, vol. 55, No. 11, November 2007.