An Efficient QoS Routing Algorithm for Solving MCP in Ad hoc Networks

Noureddine Kettaf, Hafid Abouaissa, Thang Vuduong, Pascal Lorenz

An Efficient QoS Routing Algorithm for Solving MCP in Ad hoc Networks

Noureddine Kettaf, Hafid Abouaissa, Thang Vu duong* and Pascal Lorenz

{nour-eddine.kettaf, a.abouaissa, p.lorenz}@uha.fr

Haute Alsace University
34, rue de Grillenbreit
68008 Colmar Cedex France

* thang.vuduong@orange-ft.com

France Telecom R&D
2, avenue Pierre Marzin
22307 Lannion Cedex France

Abstract

Providing guaranteed quality of service (QoS) in wireless networks is a key issue for deploying multimedia applications. To support such a QoS, an arduous problem concerning how to find a feasible end to end path to satisfy multiple QoS constraints should be studied. In general, multi-constrained path selection, with or without optimization, is an NP-complete problem that cannot be exactly solved in polynomial time. Approximation algorithms and heuristics with polynomial and pseudo-polynomial time complexities are often used to deal with this problem. However, existing solutions suffer either from excessive computational complexities that cannot be used for multimedia applications in ad hoc networks characterized by mobility and performance constraints (e.g., limited energy, wireless medium, etc.). Recently a promising heuristic algorithm H_MCOP using a non linear Lagrange relaxation path

functions has demonstrated an improvement in its success rate and in finding feasible paths. However, the H_MCOP is not suitable for ad hoc networks and has not exploited the full capability that a Lagrange relaxation could offer. In this paper, we propose an efficient multi-constrained path heuristic called E_MCP, which exploits efficiently the Lagrange relaxation and enhances the path search process to be adequate to mobile ad hoc networks. Using extensive simulations on random mobile network with correlated and uncorrelated link weights, we show that the same level of computational complexity, E_MCP can achieve a higher success ratio of finding feasible paths.

Keywords: Multiple constraints, path selection, QoS routing, ad hoc.

1- Introduction

Routing in ad hoc networks has been an active research area for many years. Much of the original work was motivated by mobile applications, such as on-line live movies and video conferencing. However, most routing protocols proposed to date purely function on best effort basis with no attempt to provide any quality of service (QoS). In essence, the purpose of QoS routing is to find a feasible path, which has sufficient unused resources to guarantee a certain service level agreement (SLA) between the service provider and the user applications. For example, the delay sensitive applications such as real time voice and video require the data flow to be received at the destination within a certain bounded time. In order to provide such QoS it is imperative that mobile ad hoc networks ensure QoS support in terms of bandwidth, delay, jitter, etc.

In general, routing consists of two basic tasks: distributing the state information of the network and searching this information for a feasible path [1]. In this paper, we focus on the second task and assume that the network information has been disseminated throughout the network using a QoS-based routing protocol (e.g., ACOR [2]). Specifically, each link in the network is associated with multiple parameters which can roughly classified into additive [3], [4]. For additive parameters (e.g., delay, jitter), the cost of an end to end path is given by the sum of the individual link values along that path [1]. However, the cost of the path with respect to (w.r.t) a non-additive parameter, such as bandwidth, is determined by the value of that constraint at the bottleneck link. It is known that constraints associated with non-additive parameters can be easily dealt with a pre-processing step by pruning all links that do not satisfy these constraints [5].

Recently, a diversity of QoS routing algorithms incorporating a variety of constraints have been proposed [1] [6] [8]. For unicast routing, the Multi-Constrained Optimal Path (MCOP) and Multi-Constrained Path (MCP) are the most notorious ones for their NP-complete property [9]. A MCOP problem needs to find a path with the minimal cost subject to one or more path constraints, while a MCP problem needs to find a path subject to two or more path constraints without necessarily finding an “optimal” solution. Regarding the two classes of problems in wired networks, much work has been done, in particular when the number of constraints is small [10], [11].

In ad hoc networks, where the topology changes dynamically, QoS routing is even more challenging. Hence, in this paper we propose an enhanced algorithm based on H_MCOP heuristic (Heuristic Multi-Constrained Optimal Path) [1], namely E_MCP (Efficient Multi-Constrained Path) to solve the MCP problem in mobile ad hoc networks. To the best of our knowledge, none of the previous published papers deal

with the MCP or the MCOP problem in ad hoc networks. Our heuristic aims to enhance the H_MCOP and explore the entire capability of the Lagrange relaxation technique to achieve a high success ratio of finding feasible paths in mobile ad hoc networks. The rest of the paper is organized as fallows. In section 2, we describe the H_MCOP heuristic. Then the proposed heuristic is presented in section 3. The formal specification and justification in section 4, and extensive simulation results are evaluated in section 5. In section 6, we briefly review some related works in wired networks, and finally our main conclusions are drawn in section 7.

2- The H_MCOP heuristic algorithm

In this section, we describe the heuristic algorithm H_MCOP proposed in [1] for the MCOP (Multi-Constrained Optimal Path) problem which only needs to run Dijkstra's algorithm (with slight modifications) twice to solve the MCOP problem with multiple constraints based on linear Lagrange relaxation.

Definition 1: Multi-Constrained Optimal Path (MCOP) problem:

Assume a network is modeled by a directed graph *G = (V,E)*, where *V* is the set of nodes and *E* is the set of links. Each link is associated with $(i, j) \in E$ is associated with a primary cost parameter $c(i, j)$ and *K* additive QoS parameters $w_k(i, j)$, $k = 1,2,\ldots K;$ all parameters are non-negative. Given *K* constraints c_k, $k = 1,2,\ldots K;$ the problem is to find a path *p* from a source node *s* to a destination node *d* such that:

(i) $w_k(p) = \sum_{(i,j)\in p} w_k(i, j) \le c_k$ for $k = 1,2,\ldots K;$ and

(ii) $c(p) = \sum_{(i,j)\in p} c(i, j)$ is minimized over all feasible paths satisfying (i).

The search of a feasible path with H_MCOP is done by approximating the non-linear cost function (1).

$$f_\lambda(p) = \left(\frac{w_1(p)}{C_1}\right)^\lambda + \left(\frac{w_2(p)}{C_2}\right)^\lambda + \ldots + \left(\frac{w_m(p)}{C_m}\right)^\lambda \qquad (1)$$

Where $\lambda \geq 1$.

As mentioned above, the H_MCOP runs two slightly modified versions of Dijkstra's algorithm. First, it computes in the backward direction the shortest paths from every node to *d* w.r.t. a linear combination of all weights. Second, in the forward direction, H_MCOP starts from *s* and discovers every node *u* based on a path *p*, where *p* is a heuristically determined complete *s*-*d* path that is obtained by concatenating the already traveled sub-path, from *s* to *u* and the estimated remaining sub-path from *u* to *d*. Since the algorithm considers complete paths before reaching *d*, it can foresee some feasible paths during the search. If paths seem feasible, the algorithm can switch to explore these feasible paths for the one with minimum length. A pseudo-code of the H_MCOP algorithm is shown in A1.

Algorithm A1: The heuristic algorithm H_MCOP for the MCOP problem:

```
H_MCOP (G = (V,E), s, d, C_m, m = 1, 2, …,K)
Reverse_Dijkstra (G = (V,E),d)
if r[u] > K then
        return failure /*no feasible paths*/
endif
Look_Ahead_Dijkstra (G = (V,E),s)
if G_m[d] ≤ C_m, m = 1, 2, …,K  then
        return the path /*a feasible path is found*/
endif
return failure
```

Among the existing work in the context of QoS routing algorithms, it is believed that the H_MCOP has the best performance in terms of finding feasible paths and low computational complexities for wired networks. However, in wireless mobile networks such as ad hoc networks characterized by node mobility and resource constraints have limited the application of the H_MCOP. Hence, in this paper, we investigate a new heuristic which adapts the H_MCOP algorithm and enhances its Lagrange relaxation procedure to provide a solution for the MCP problem in mobile ad hoc networks. By general, the proposed solution is applicable to any number of constraints, irrespective of their nature and interdependence [1].

3- The efficient multi-constrained path heuristic (E_MCP)

We now present our heuristic algorithm E_MCP, which attempts to the MCP problem in ad hoc networks. E_MCP adopts the basic idea of the H_MCOP but by using an enhanced Lagrange relaxation procedure to be adequate to the mobile environment of ad hoc networks. Specifically, H_MCOP determines the sub-path from *s* to *u* by Look_Ahead_Dijkstra w.r.t. a complete path *s*-*d*. At this stage, a predecessor of a node *v* is determined as follows. If there is a neighbouring node *u* which satisfies the path feasibility from *s* to *d*, the intermediate node *u* is the predecessor, otherwise, the predecessor will be a node *u* that minimizes $f_{\lambda}(.)$. The sub-path from *u* to *d* is determined using the linear cost function $f_1(.)$.

3.1- Nonlinear cost function for MCP

We consider the non linear cost function (1) [1] for any path *p* from the source to the destination. For a given $\lambda \geq 1$ suppose there is an algorithm χ that returns a path *p* by minimizing the cost function (1). Then, the following bounds on the performance of algorithm χ can be established.

Theorem 1: Consider the MCP problem and assume that there is at least one feasible path *p** in the network. Let p be a path that minimizes the cost function g_λ for a given $\lambda \geq 1$. Then,

(i) $w_k(p) \leq c_k$ for at least one *k*, and

(ii) $w_k(p) \leq \sqrt[\lambda]{Kc_k}$ for all other *k*'s.

Proof: if the returned path *p* is feasible, then from (1) the above bounds are correct. Assume that *p* is not feasible. Since the algorithm returns the path *p* (and not the feasible path *p**), it must be true that

$$g_\lambda(p) \leq g_\lambda(p^*)$$

In addition, since $w_k(p) \leq c_k$ for all *k's*, we have

$$w_k(p^*) \leq K$$

Thus,

$$w_k(p) \leq K \qquad (2)$$

If $w_k(p) > c_k$ for all *k's*, then $g_\lambda(p) > K$. Since this contradicts (2), we must have $w_k(p) \leq c_k$ for at least one *k,* and the bound in part (i) is correct. Note that if $g_\lambda(p) > K$, then it is guaranteed that there is no feasible path *p** in *G* because of at least one *k*, $w_k(q) > c_k$ for any path *q*.

To prove the part (ii), assume to the contrary that for at least one constraint c_j we have $w_j(p) \le \sqrt[\lambda]{Kc_j}$, so that $(\frac{w_j(p)}{c_j})^\lambda > K$. It readily follows that $g_\lambda(p) > (\frac{w_j(p)}{c_j})^\lambda > K$, which contradicts (2). Hence, part (ii) is proved.

Corollary 1: as λ increases, the likelihood of finding a feasible path also increases.

Proof: follows immediately from theorem 1. Therefore, to increase the possibility of finding a feasible path, it makes sense to set λ to its largest value, i.e., $\lambda \to \infty$. In order to provide a practical computational model for $\lambda \to \infty$, we can replace the cost function $g^*(p) = \lim_{\lambda \to \infty} g_\lambda(p)$ by another cost function that does not explicitly involve λ but that achieves the same ordering of candidate as g^*.

3.2- Proposed heuristic for MCP in ad hoc networks

E_MCP adopts the basic idea of H_MCOP, but by using an enhanced Lagrange relaxation procedure to adapt it to the mobile environment of ad hoc networks. Lagrange relaxation has the property of providing bounds on the value of the optimal objective function and, frequently, of quickly generate good, though not necessarily optimal, solutions with associated performance guarantees [12]. The major modifications are related to the use of the same cost function in both directions and the evaluation of sub-path instead of the complete path. Our modifications M1 and M2 aim to exploit the full capability that a Lagrange relaxation could offer and enhance the probability of finding feasible paths in mobile ad hoc networks characterized by frequent topology changes and link breaks. Hence, these modifications are:

M1: use of the same nonlinear cost function $f_\lambda(.)$ instead of $f_1(.)$ for Reverse_Dijkstra.

M2: evaluate the cost function w.r.t. the sub-path from *s* to *v* instead of the complete path from path from *s* to *d* in Look_Ahead_Dijkstra when determining the predecessor of node *v*.

Then, E_MCP can be easily implemented as simple as single objective algorithms. A pseudo-code for E_MCP is shown in A2. Its inputs are a directed graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$, in which each link (i, j) is associated with a primary cost $c(i, j)$ and *K* weights $w_k(i, j)$, $k = 1,2,\ldots,K$. A source node *s*, a destination node *d*, and *K* constraints c_k, $k = 1,2,\ldots,K$. For each node *u*, the algorithm maintains the following labels: $r[u]$, $R_k[u]$, $\pi_r[u]$, $g[u]$, $G_k[u]$, $\pi_g[u]$ and $c[u]$, $k = 1,2,\ldots,K$. Label $r[u]$ represents the cost of the shortest path from *u* to *d* w.r.t. the cost function g_λ. Labels $R_k[u]$, $k = 1,2,\ldots,K$ represents the individually accumulated link weights along the path. The predecessor of this path is stored in label $\pi_g[u]$. Label $g[u]$ represents the cost function of a foreseen sub-paths from *s* to *u* to *d* w.r.t. the nonlinear cost function g_λ. Labels $G_k[u]$, $k = 1,2,\ldots,K$ and $c[u]$ represent the individually accumulated cost of link weights and the primary cost along the already travelled segment of this path from *s* to *u*. The predecessor of *u* on this path is stored in the label $\pi_g[u]$.

Algorithm A2: The heuristic algorithm E_MCP for the MCP problem:

E_MCP *(G = (V,E), s, d, C_m, m = 1, 2, …,K)*

Reverse_Dijkstra (G = (V,E),d, w_m, m = 1, 2, …,K)

if $R_m[s] \le C_m$, $m = 1,2,\ldots,K$ **then**

    return failure /*there no feasible path*/

endif

Look_Ahead_Dijkstra (G = (V,E),s)

if $G_m[d] \le C_m$, *m = 1, 2, …,K* **then**

    return the path /*a feasible path is found*/

endif

return failure

As in H_MCOP, in E_MCP there are two directions in the algorithm: backward (from *d* to *u*) to estimate the cost of the remaining segment using the nonlinear cost function g_λ and forward (from *s* to *u*) to find the most promising feasible path. In the backward direction, the E_MCP algorithm finds the path from every node *u* to *d* w.r.t. the cost function $g_\lambda(.)$. For that it uses Reverse_Dijkstra [13] with some modifications to the relaxation procedure as shown in algorithm A3.

Algorithm A3: The relaxation procedure of subroutine Reverse_Dijkstra in E_MCP

Reverse_Dijkstra_Relax (u,v)

if $r[u] > \sum_{m=1}^{K} (\frac{R_m[v] + w_m(u,v)}{C_m})^{\lambda}$ **then**

$r[u] := \sum_{m=1}^{K} (\frac{R_m[v] + w_m(u,v)}{C_m})^{\lambda}$

$R_m[u] := R_m[v] + w_m(u,v)$ for $m = 1,2,\ldots,K$

$\pi_r[v] := u$

endif

Initially, Reverse_Dijkstra sets $r[u] = \infty$ and $\pi_r[u] = NIL$ for every node *u*. It then starts at node d by setting $r[d]$ and $R_k[d]$, $k = 1,2,\ldots,K,$ to zeros. It explores the graph and eventually returns a path *p* from *s* to *d*. The main advantage of this modification is the enhancement of the probability of finding feasible paths by increasing λ.

If there is a possibility that the network contains feasible paths, a heuristic search procedure called Look_Ahead_Dijkstra is executed in the forward direction. Then, the algorithm checks $r[s] > K$ to determine the possibility of finding feasible paths (please see Theorem 1).

If there is a possibility that the network contains feasible paths, Look_Ahead_Dijkstra is executed in the forward direction to identify whether there is other paths *q*, which improve the performance over the returned path *p* using the information provided by the above Reverse_Dijkstra. To implement Look_Ahead_Dijkstra, we need a slight modification to the relaxation procedure of Dijkstra's algorithm [14].

Algorithm A4: The relaxation procedure of subroutine Look_Ahead_Dijkstra in E_MCP

Look_Ahead_Dijkstra_Relax *(u,v)*

Let tmp be a temporary node

$$f[tmp] := \sum_{m=1}^{K} \left(\frac{G_m[u] + w_j(u,v)}{C_m}\right)^{\lambda}$$

$$G_m[tmp] := G_m[u] + w_m(u,v) \text{ for } m = 1,2,\ldots,K$$

$$R_j[tmp] := R_j[v] \text{ for } m = 1,2,\ldots,K$$

if $(\Pr efer\_the\_best(tmp,v) = tmp)$ **then**

$$f[v] := f[tmp]$$

$$G_m[v] := G_m[tmp]$$

$$\pi_f[v] := u$$

endif

Initially Look_Ahead_Dijkstra $g[u] = \infty$ and $\pi_g[u] = NIL$ for every node u. Then, it starts from node *s*, setting $g[s], c[s]$ and $G_k[s],\ k = 1,2,\ldots,K,$ to zeros. It explores the graph by choosing the next node based on the preference rule shown in A5.

The preference rule takes as input two nodes and their labels. Then, it selects a path of these nodes such that minimizes the primary cost function if foreseen *s-d* paths passing through these nodes are feasible; otherwise, it selects the path that minimizes the objective function g_λ.

Algorithm A5: The preference rule for Look_Ahead_Dijkstra in E_MCP

Prefer_the_best (a,b)

if $(\forall m = 1,2,...,K),\ G_m[a] + R_m[a] \le C_m$ **then**

return (a)

if $(\forall m = 1,2,...,K),\ G_m[b] + R_m[b] \le C_m$ **then**

return (b)

if $f[a] < f[b]$ **then**

return (b)

The computational and space complexities of the resulting E_MCP algorithms are equal to that of Dijkstra's, since at most two modified versions of Dijkstra's algorithm are executed with the complexity of $O(n\log(n)+m)$. To improve the performance, the forward direction of the E_MCP can be used with the *k*-shortest path implementation of Dijkstra's algorithm presented in [15]. Note that *k*-shortest paths are considered w.r.t. the minimization of the nonlinear cost function. The complexity of the *k*-shortest path algorithm is $O(km\log(kn)+k^2m)$. Hence, the complexity of the E_MCP with *k*-shortest path is $O(n\log(n)+km\log(kn)+(k^2+1)m)$.

4- Justification of E_MCP heuristic algorithm

We aim with E_MCP to exploit the full capability that a Lagrange relaxation technique could offer to solve the MCP problem in a mobile environment such as ad hoc networks.

Respectively, it is necessary to justify the proposed Lagrange process and the method that the E_MCP determines a path according to the nonlinear cost function $f_\lambda(.)$.

Consider a special case where the following condition is satisfied:

$$\sum_{m=1}^{k} x_m(p) = \Delta \qquad (3)$$

Where Δ is a fixed value, $x_m(p) = \frac{w_m(p)}{C_m}$ and *p* is any path between two nodes, say from *u* to *v*.

Under this condition, we need to determine the path from *u* to *v* that minimizes $f_\lambda(.)$.

The problem can be described as the following constrained optimization problem:

$$Minimize \quad f_\lambda(p) = \sum_{m=1}^{k} (x_m(p))^\lambda$$

Subject to the condition in (3).

It can be further converted to the following unconstrained optimization problem:

$$Minimize \quad g_\lambda(p) = \sum_{m=1}^{K} (x_m(p))^\lambda + \sigma(\sum_{m=1}^{K} x_m(p) - \Delta)$$

Where σ is a Lagrange multiplier.

To solve the unconstrained optimization problem, we let

$$\frac{\partial h(p)}{\partial x_m(p)} = 0, \ m = 1,2,...,K, \ \frac{\partial h(p)}{\partial \sigma} = 0,$$

Which lead to

$$x_m(p) = \frac{\Delta}{K}, \ m = 1,2,...,K.$$

Remark: the derivation does not apply the case of $\lambda = 1$.

We demonstrate by the derivation above that under the condition in Eq.3, the optimal path would be the path that has the equal ratio of each weight to the corresponding upper bound. In real situation, Eq.3 does not hold normally, but we can infer that the optimal solution would be a path that has the following two characteristics.

C1: it has a relatively small summation of all ratios, i.e., $\sum_{m=1}^{K} x_m(p)$

C2: the ratios are very close to each other.

The E_MCP executes Look_Ahead_Dijkstra to find the sub-path from *s* to an intermediate node *v* by evaluating $f_\lambda(.)$ w.r.t. the path itself. Moreover, the sub-path *s*-*v* must meet C1 and C2. On the other hand, E_MCP executes the Reverse_Dijkstra to choose the sub-path from an intermediate node *v* to *d* by evaluating $f_\lambda(.)$ which also must meet C1 and C2. In terms of weight, the sub-path from *s* to *v* has nearly the smallest ratio of weights accumulated at any node *v* to the upper bound, which is very close to that for any other weight.

Similarly, for the sub-path *v*-*d* expect the weight is accumulated from *v* to *d*. Hence, using the sub-paths search in this fashion should achieve the highest probability of finding a feasible MCP solution.

We believe that this method could achieve good performance when all the constraints are quite loose, but when C_0 becomes relatively tight, H_MCOP is likely to make mistakes. To enlighten this observation, consider the case where all constraints are quite loose except that C_0 is just slightly greater than $w_0(p_0)$, where p_0 is the shortest path between *s* and *d* w.r.t. weight w_0, which implies the availability of a very limited number of feasible paths.

5- Performance evaluation

In this section, we simulate our algorithm in ad hoc network environment. The results are for a network of 50 nodes randomly on a square 670m flat space. To preserve connectivity, every node has 12 neighbours on average. Nodes move following the Random Trip Model [7]. We associate two randomly generated weights with each link (i, j). As shown in Table 1, these weights are selected from uniform distributions under several types of correlation between them.

If there is positive correlation, we assume that both weights are selected from uniform distributions with either small mean or large mean. If there is negative correlation, we assume that one of weights is selected from a uniform distribution with small mean while the other is selected from another uniform distribution with large mean. If there is no specific correlation, we assume both weights are independently selected from uniform distributions. The primary cost of a link (i, j) is taken as $c(i, j) \sim uniform[1,50]$. The source and destination of a request are randomly generated such that the minimum hop-count between them is at least three.

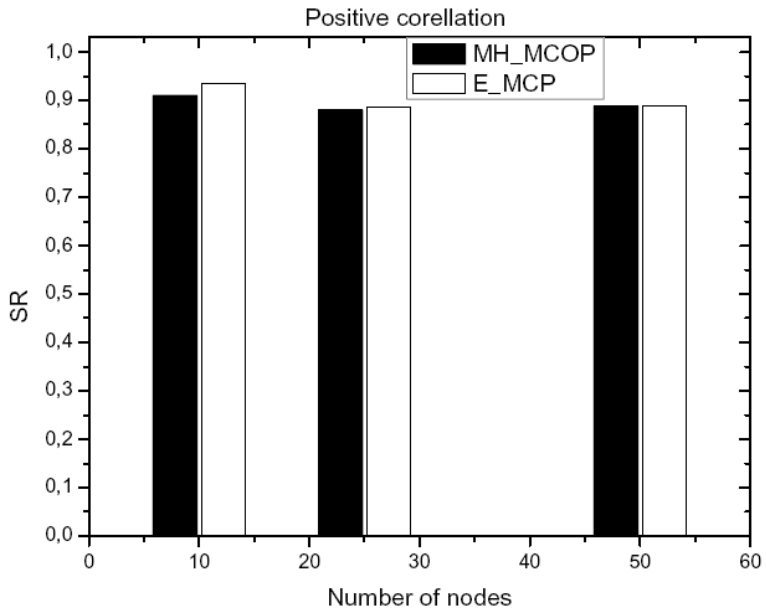
Positive correlation	No correlation	Negative correlation
$w_1(i, j) \sim uniform[1,10]$ $w_2(i, j) \sim uniform[1,25]$ OR $w_1(i, j) \sim uniform[10,25]$ $w_2(i, j) \sim uniform[25,50]$	$w_1(i, j) \sim uniform[1,25]$ $w_2(i, j) \sim uniform[1,50]$	$w_1(i, j) \sim uniform[1,10]$ $w_2(i, j) \sim uniform[25,50]$ OR $w_1(i, j) \sim uniform[10,25]$ $w_2(i, j) \sim uniform[1,25]$

Table 1. Ranges of link weights and the correlation between them

The constraints [1] are also randomly generated, but their ranges determined based on the best paths w.r.t. w_1 and w_2 as follows. Let p_1 and p_2 be two shortest paths from *s* to *d* w.r.t. w_1 and w_2, respectively. We take $c_1 \sim uniform[0.8 * w_1(p_2), 1.2 * w_1(p_2)]$ and $c_2 \sim uniform[0.8 * w_1(p_1), 1.2 * w_2(p_1)]$.

We contrast the performance of E_MCP and MH_MCOP, which is a modified H_MCOP to be MCP based, using the success ratio SR, which refers to the fraction of connection requests for which feasible paths are found by the given approximation algorithm or heuristic in several scenarios.

The results reported here are averaged over several runs. In each run, random graphs are generated according to the mobility model. For each instance of a random graph, ten independent realizations of link weights are generated using different random seeds. Finally for each instance of a random graph with given link weights, about 5~20 connection requests are generated for graphs with 10, 25, and 50 nodes, respectively.



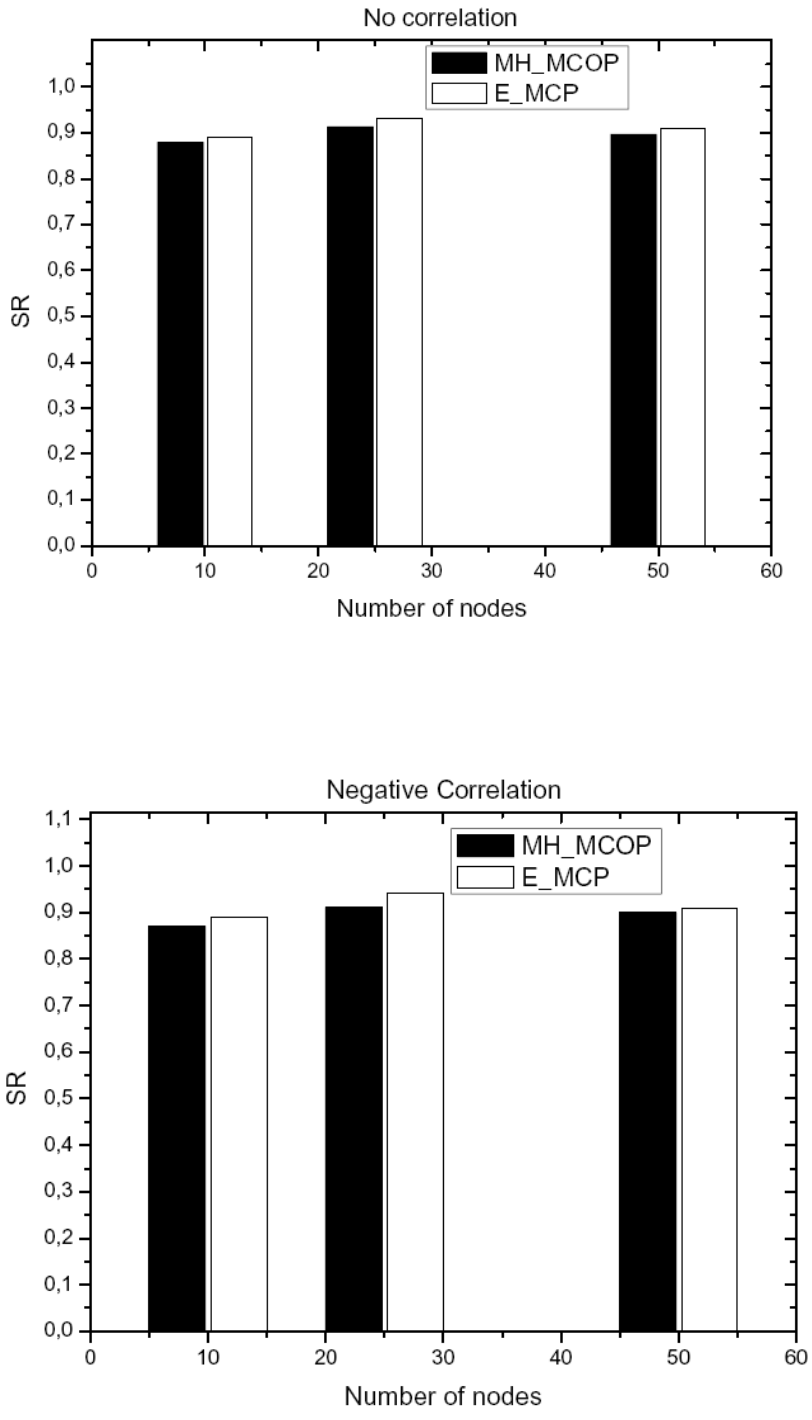
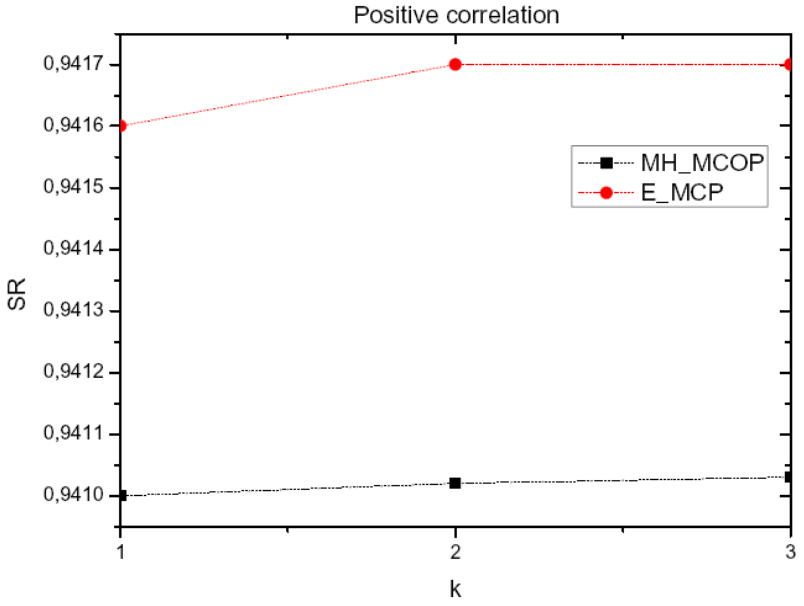
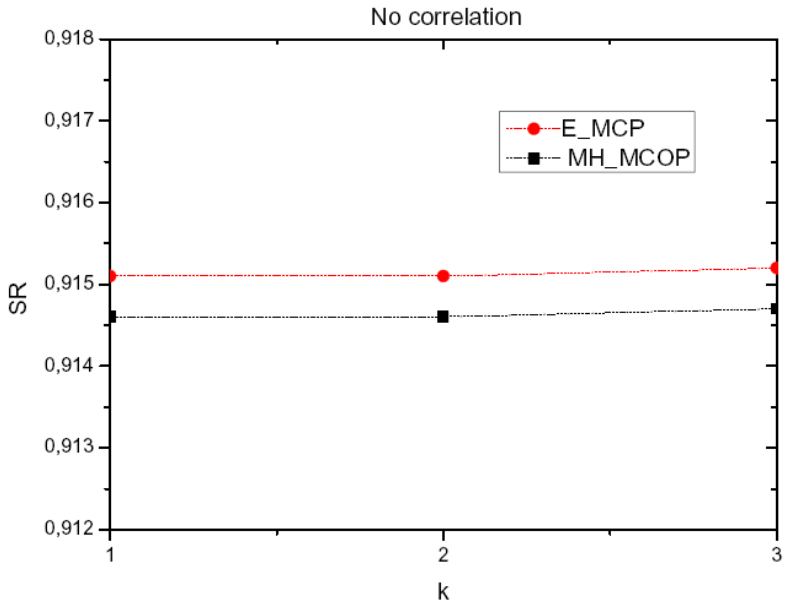


Figure 1. SR of E_MCP and MH_MCOP used on random graphs with 10, 25 and 50 nodes

Figure 1 shows the comparison of SR's of our proposed E_MCP and the MH_MCOP. When link weights are positively correlated, the path weights also become positively correlated [1], and thus a nonlinear approximation heuristic has a better success ratio of finding feasible path. However, if the link weights are negatively correlated, there will be more paths for which $w_1(p) >> w_2(p)$, or vice versa. This degrades the performance of a nonlinear approximation algorithm, which works better than a linear approximation, when the two link weights are comparable in value. Such a problem is partially absent in [1] since H_MCOP introduces a nonlinear cost function. However, this problem in E_MCP is totally absent, due to the use of the nonlinear cost function for Reverse_Dijkstra instead of the linear function.

We now contrast our study on the performance comparison using the *k*-shortest paths for E_MCP and MH_MCOP. Figure 2 bellow shows the SR's of both algorithms versus the number of considered shortest paths. E_MCP gives better SR using smaller values of *k* than MH_MCOP regardless of the number of nodes or correlation between the link weights. In other words, the SR of MH_MCOP increases with *k* and finally converges the SR that initially provided by E_MCP with smaller *k*. In essence, the SR provided by E_MCP with *k* = 1 can only achieved by MH_MCOP when *k* = 2. Comparing the equal complexities of E_MCP and HH_MCOP, we claim that E_MCP outperforms MH_MCOP in performance for the same amount of computational complexity.


Positive correlation
0,9417
0,9416
0,9415
0,9414
0,9413
0,9412
0,9411
0,9410
SR
1
2
3
k
MH_MCOP
E_MCP


No correlation
0,918
0,917
0,916
0,915
0,914
0,913
0,912
SR
1
2
3
k
E_MCP
MH_MCOP

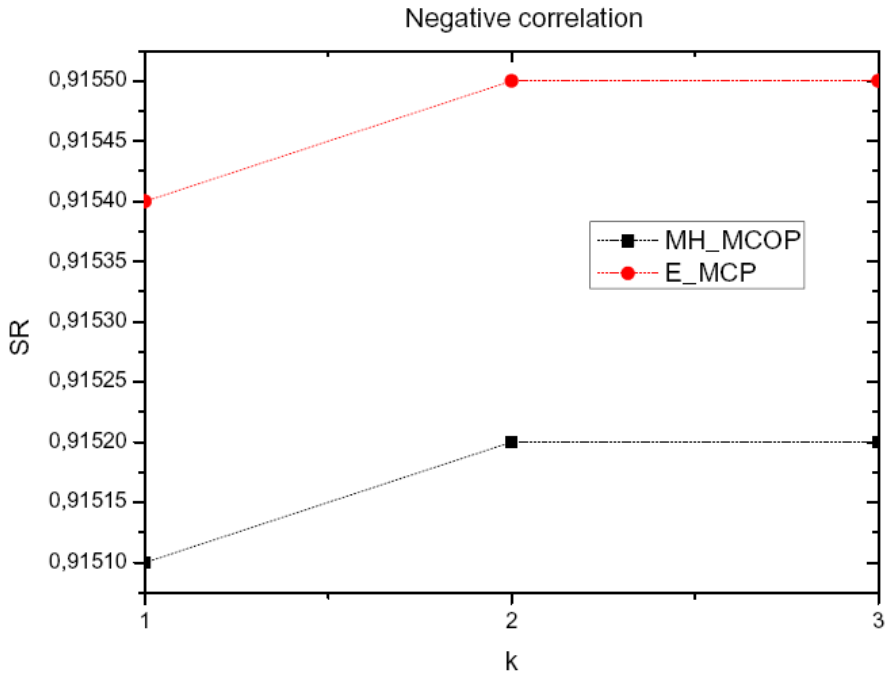


Figure2. SR's of E_MCP and MH_MCOP with k-shortest paths of 50 nodes.

6- Related works

In wired networks MCP has been extensively studied. Jaffe [16] presented two heuristics. One takes pseudo-polynomial-time. The other takes polynomial-time. The latter is to minimize (w_1 (p) + d*w_2 (p)), where $w_1(p)$ is path length, $w_2(p)$ is path cost and d is a weight factor. Chen and Nahrstedt's approach [17] is to map the cost (or delay) of every link from an unbounded number of integers, which reduces the original NP-hard problem to a simpler problem that can be solved in polynomial time. When Dijkstra's algorithm [18] is used, the running time of their algorithm is $O(x^2|V|^2)$; when Bellman-Ford algorithm is used, the running time is $O(x|V||E|)$, where x is an adjustable positive integer and its value can be as high as $10|V|$, resulting in an overall running time of $O(|V|^4)$. In [19], Korkmaz *et al.*'s idea is firstly to prune all the links that cannot be on any feasible path connecting the source-destination pair. It

then uses a randomized search to find a feasible path with a running time of $O(|V|^3)$. In [21], considering a combined weight $\alpha w_1(p) + \beta w_2(p)$, Korkmaz *et al.* provided an approximation algorithm that searches for appropriate values of α and β to determine a path subject to two additive constraints with a running time of a logarithmic number of Dijkstra's algorithm.

In [20] the authors proposed a similar dynamic algorithm for the MCP problem. However, the computational complexity of this algorithm grows exponentially with the size of the network. In [17] the authors proposed a heuristic algorithm that modifies the MCP problem by scaling down the values of one of the two link weights to bounded integers. It was shown that the modified problem can be solved by using Dijkstra's shortest path algorithm and that the solution to the modified algorithm is also a solution to the original one.

7- Conclusions

Solving the MCP problem in ad hoc networks is a key issue for QoS application. In this article, we introduced a new heuristic algorithm E_MCP based on Lagrange relaxation technique for solving MCP problem in mobile ad hoc networks by introducing some efficient modifications to the H_MCOP algorithm, which proves a high success ratio in finding feasible paths. Hence, our modifications aimed to explore the full capability of the Lagrange relaxation and enhance the probability of finding feasible paths. We first, modified the relaxation procedure in Look_Ahead_Dijkstra's algorithm. Second, instead of using a linear cost function $f_1(.)$, we employed the same nonlinear cost function $f_\lambda(.)$ in Reverse_Dijkstra.

Extensive simulations demonstrate that E_MCP can achieve a higher probability of finding feasible paths than the original H_MCOP (or MH_MCOP) with better performance.

As future work, we will investigate how E_MCP performs in the presence of inaccurate state of information and what modifications need to be done.

References

[1] Turgay Korkmaz, Marwan Krunz, ”Multi-constrained optimal path selection”. pages 834–843. INFOCOM 2001, 2001.

[2] N. Kettaf et al. “Admission Control enables Routing Protocol -ACOR-” Internet draft, IETF, (draft-kettaf-manet-acor-00.txt), July 2006, (Work in progress).

[3] A. Alles, “ATM internetworking,” White Paper, Cisco Systems, Inc., May 1995.

[4] Z. Wang, “On the complexity of quality of service routing,” Information Processing Letters, vol. 69, no. 3, pp. 111–114, 1999.

[5] Z. Wang and J. Crowcroft, “Bandwidth-delay based routing algorithms,” in Proceedings of the GLOBECOM ’95 Conference. IEEE, Nov. 1995, vol. 3, pp. 2129–2133.

[6] D. H. Lorenz, A. Orda, D. Raz, and Y. Shavitt, “Efficient QoS partition and routing of unicast and multicast,” in IWQoS 2000, June 2000, pp. 75–83.

[7] M. V. Jean-Yves Le Boudec. “Perfect simulation and stationarity of a class of mobility models”. Pages 834–843. INFOCOM 2005, 2005.

[8] Balazs Szviatovszki, et al. “Lagrange relaxation based method for the qos routing problem”. pages 782–791. INFOCOM 2001, 2001.

[9] Douglas S. Reeves, et al., “A distributed algorithm for delay-constrained unicast routing”. IEEE/ACM Transactions on Networking, 8(2):230–250, April 2000.

[10] Feng Gang, Makki Kia. “An efficient approximate algorithm for delay-cost-constrained qos routing”. page 395. International Conference on Computer Communications and Networks (ICCCN’01), 2001.

[11] Gang Cheng, Ye Tian. “A new qos routing framework for solving MCP”. IEICE Transaction on Communications, E86-B(2):534–541, February 2003.

[12] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, “Network Flows: Theory, Algorithms, and Applications”, Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[13] S. D. Patek, R. Venkateswaran, and J. Liebeherr, "Simple alternate routing for differentiated services networks," Computer Networks 37 (2001), pp: 447-466.

[14] W. C. Lee, M. G. Hluchyi, and P. A. Humblet, “Routing subject to quality of service constraints in integrated communication networks,” IEEE Network, pp. 46–55, July/August 1995.

[15] E. I. Chong, S. R. Sanjeev Rao Maddila, and S. T. Morley, “On finding single-source single-destination k- shortest paths,” in the Seventh International Conference on Computing and Information (ICCI ’95), July 5-8, 1995, pp. 40–47.

[16] J. Jaffe, “Algorithms for finding paths with multiple constraints,” Networks, Vol. 14, No. 1, pp. 95-116, 1984.

[17] S. Chen and K. Nahrstedt, “On finding multi-constrained paths,” In Proc. IEEE ICC ’98, pp. 874-879, June 1998.

[18] D. Bertsekas and R. Gallager, Data Networks. Prentice-Hall, second edition,1992.

[19] T. Korkmaz and M. Krunz, “A randomized algorithm for finding a path subject to multiple QoS constraints,” In Proc. IEEE GLOBECOM '99, pp. 1694-1698, Dec. 1999.

[20] D. Blokh and G. Gutin, “An approximation algorithm for combinatorial optimization problems with two parameters,” IMADA preprint PP-1995-14, May 1995, http://www.imada.ou.dk/Research/Preprints/Abstracts/1995/14.html

[21] T. Korkmaz and M. Krunz, and S. Tragoudas, “An efficient algorithm for finding a path subject to two additive constraints,” in Proc. ACM SIGMETRICS’00, pp. 318-327, Jun. 2000.