


Neural Network Based Algorithm for Multi-Constrained Shortest Path Problem

View metadata, citation and similar papers at core.ac.uk

brought to you by  CORE

provided by Xiamen University Institutional Repository

Jiyang Dong¹, Junying Zhang², and Zhong Chen

¹ Department of Physics, Fujian Engineering Research Center for Solid-State Lighting,
Xiamen University, Xiamen 361005, P.R. China

² National Key Laboratory for Radar Signal Processing,
Xidian University, Xi'an 710071, P.R. China
jydong@xmu.edu.cn

Abstract. Multi-Constrained Shortest Path (MCSP) selection is a fundamental problem in communication networks. Since the MCSP problem is NP-hard, there have been many efforts to develop efficient approximation algorithms and heuristics. In this paper, a new algorithm is proposed based on vectorial Autowave-Competed Neural Network which has the characteristics of parallelism and simplicity. A nonlinear cost function is defined to measure the autowaves (*i.e.*, paths). The M -paths limited scheme, which allows no more than M autowaves can survive each time in each neuron, is adopted to reduce the computational and space complexity. And the proportional selection scheme is also adopted so that the discarded autowaves can revive with certain probability with respect to their cost functions. Those treatments ensure in theory that the proposed algorithm can find an approximate optimal path subject to multiple constraints with arbitrary accuracy in polynomial-time. Comparing experiment results showed the efficiency of the proposed algorithm.

1 Introduction

Providing Quality-of-Service (QoS) guarantees in packet networks gives rise to several challenging issues. One of them is how to determine a feasible path that satisfies a set of constraints while maintaining high utilization of network resources. The latter objective implies the need to impose an additional optimality requirement on the feasibility problem. This can be done through a primary cost function according to which the selected feasible path is optimization [1,2]. In general, multi-constrained path selection, with or without optimization, is an NP-complete problem that cannot be exactly solved in polynomial time [3]. Heuristics and approximation algorithms with polynomial- and pseudo-polynomial-time complexities are often used to deal with this problem. One common heuristic approach is to find the k -shortest paths with respect to a cost function defined based on the link weights and the given constraints, hoping that one of these paths is feasible [4]. Increasing k improves the performance of this approach, but the computational cost becomes excessive that cannot be used for online network operation. Another approach is to exploit the dependencies among the constraints, and to solve the path selection problem assuming specific scheduling schemes at network routers [5]. Specifically in QoS

routing, if weighted fair queuing service discipline is being used and the constraints are bandwidth, queuing delay, delay-jitter, and loss, then the problem can be reduced to standard shortest path problem by expressing all the constraints in terms of bandwidth. As we can see, this approach only deals with special cases of the problem.

Jaffe proposed a new solution called multi-label routing for the MCSP problem [6]. It is simple, easy to implement. And most importantly, it can find the approximate optimal path with multiple constraints. Unfortunately, the computational complexity is also exponentially increased with the network scale for the reasons of too many labels to be reserved and too many vectors to be compared. Some modifications are suggested for the multi-label method to reduce the computational and space complexity [7], *e.g.*, limiting the total labels of each node, and the algorithmic loops are reduced to about 1/3 of the original multi-label algorithm. However, a large number of labels are needed for the improved multi-label routing algorithm to ensure the approximate optimal paths can be found. Therefore, the computational and space complexity are not reduced essentially.

In this paper, the Autowave-Competed Neural Network (ACNN) [8,9] is vectorized and applied successfully to the MCSP problem. The M -paths limited scheme is adopted to reduce the computational and space complexity. All the autowaves propagating to a neuron have to compete with the paths reserved on the neuron's threshold, and only M autowaves (*i.e.*, paths) can survive the competition. The winners will be reserved for the next competition by replacing the old paths on the neuron's threshold, while the losers will be discarded. At the same time, the winners will propagate forward to its adjacent neuron. A nonlinear cost function is defined to measure the paths. However, differing from the traditional MCSP algorithms which focus on finding the paths with minimum cost function, the proposed algorithm uses the cost function to build on a proportional path selection scheme. The discarded paths can revive with certain probability. So the optimal path can also be found in theory by the proposed algorithm even with small M . Furthermore, the vectorial ACNN based algorithm is parallel in architecture and in running mode.

The rest of paper is organized as follows. Section 2 gives the definition of MCSP problems. Section 3 introduces the scalar ACNN for traditional shortest path problem (no constraint). The vectorial ACNN, the nonlinear cost function and the new algorithm of MCSP is defined in section 4. Simulation results are presented in section 5 and some concluding remarks are drawn in section 6.

2 Problem Definition

Consider a network that is represented by a directed graph $G = (V, E)$, where V is the set of nodes which represent switches, routers, and hosts and E is the set of edges which represent communication links. Each edge $(i, j) \in E$ is associated with a primary cost parameter $c(i, j)$ and K QoS weights, $\omega_k(i, j)$, $k = 1, 2, \dots, K$, all parameters are non-negative. Given K constraints, C_k , $k = 1, 2, \dots, K$. The MCSP problem is to find a path P from a source node s to the destination node d such that [1]:

$$\begin{aligned}
 \text{(i)} \quad \omega_k(P) &= \sum_{(i,j) \in P} w_k(i,j) \leq C_k \quad \text{for } k = 1, 2, \dots, K \\
 \text{(ii)} \quad c(P) &= \sum_{(i,j) \in P} c(i,j) \text{ is minimized over all feasible paths} \\
 &\text{satisfying (i)}
 \end{aligned}
 \tag{1}$$

For the simplicity of expression and computation, the cost parameter can be regarded as one of the constraints on the link, *e.g.*, let $\omega_0(i,j) = c(i,j)$. Then the MCSP problem is to find a path satisfied all $K+1$ constraints and with the minimum cost from node s to node d . To solve the MCSP problem, one can find firstly all the paths satisfied the constraints, then chooses the one with minimum cost from those paths [6]. This paper also treats the cost parameter as a constraint of the link.

Each link in the network is associated with multiple parameters which can be roughly classified into additive and non-additive [10]. For the additive parameters (*e.g.*, delay, jitter, administrative weight), the cost of an end-to-end path is given by the sum of the individual link values along that path. In contrast, the cost of a path with respect to 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 deal with a preprocessing step by pruning all links that do not satisfy these constraints [5]. Hence, in this paper we will mainly focus on additive QoS parameters and assume that the optimality of a path is evaluated based on an additive parameter.

3 ACNN for Shortest Path Problem

The Shortest Path (SP) problem is well-documented. Many practical algorithms have been developed for shortest path problem. Recently, we proposed a neural network model called autowave-competed neural network (ACNN) for the SP problem [8,9]. The ACNN neuron is consisted of three parts, *i.e.*, the minimum selector, the autowave generator and the threshold updater, see Fig.1.

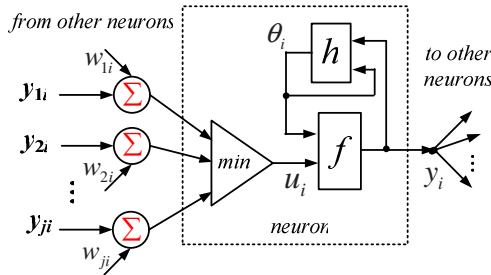


Fig. 1. Neuron model of ACNN

The ACNN neuron can be described with the following equations,

$$Z_i(t) = \{j \mid w_{ji} \neq \infty \text{ and } y_j(t-1) > 0\} \tag{2}$$

$$u_i(t) = \begin{cases} 0 & Z_i(t) = \emptyset \\ \min_{j \in Z_i(t)} (y_j(t-1) + w_{ji}) & \text{otherwise} \end{cases} \tag{3}$$

$$y_i(t) = f[u_i(t), \theta_i(t-1)] = \begin{cases} u_i(t) & u_i(t) < \theta_i(t-1) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$$\theta_i(t) = h[y_i(t), \theta_i(t-1)] = \begin{cases} \theta_i(t-1) & y_i(t) = 0 \\ y_i(t) & \text{otherwise} \end{cases} \tag{5}$$

where i is the index of neuron, t is the time (or says the iterations). $u_i(t)$, $\theta_i(t)$ and $y_i(t)$ are the internal activity, the threshold and the output of neuron i at time t respectively. w_{ij} is the connection weight from neuron i to neuron j . $z_i(t)$ is the set of neurons which fired at time t and is reachable to neuron i .

When applied to the SP problem, an ACNN isomorphic to the weighted graph G should be constructed, *i.e.*, each node of G is corresponding to a unique neuron of the network, and w_{ij} is associated with the weight of the edge (i, j) in G , see Fig. 2. All neurons are initialized with infinite threshold and zero-internal-activity. Fire the source neuron to run the network, and the firing would inspire some autowaves

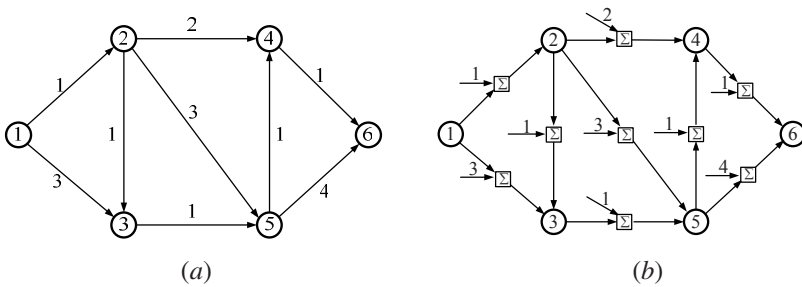


Fig. 2. ACNN topology for SP problem. (a) A weighted digraph. The circles with numbers inner are the vertexes, and the numbers on edges are the costs associated with the corresponding edges. (b) The ACNN model for the SP problem of the graph shown in (a). The circles with numbers inner are the neurons, and the squares with “ Σ ” inner are the summators on the corresponding links.

propagating through the whole network. When passed through the neuron i , the traveling distance of an autowave would be recorded on the threshold $\theta_i(t)$ if it is the shortest. All neurons decrease their thresholds progressively until the network stops.

When the network stops, the threshold θ_i is corresponding to the distance of the shortest path from the source neuron to the neuron i .

The ACNN based shortest path algorithm is parallel, non-parameter, and flexible, which can easily be modified to suit for the other problems concerned with shortest path [9]. Furthermore, it is suitable for large-scale network. Reader can refer to the reference [8] for more details about ACNN.

4 Vectorial ACNN for MCSP Problem

4.1 Vectorial ACNN

In order to solve the MCSP problem, a vectorial ACNN must be established, in which the connection weight, the threshold, the internal activity and the output of the neurons should be vectored.

- (1) The vector-formed connection weight from neuron i to the reachable neuron j should be $\bar{\mathbf{w}}_{ij} = [\omega_1(i, j), \omega_2(i, j), \dots, \omega_K(i, j)]$, where $\omega_k(i, j), k = 1, \dots, K$ is the k^{th} QoS parameters on the link (i, j) .
- (2) A path $(s \rightarrow \dots \rightarrow i \rightarrow j \rightarrow \dots)$ can be written as a vector-formed $\mathbf{P} = (\bar{\mathbf{p}}, \bar{\mathbf{d}})$, where $\bar{\mathbf{d}} = (d_1, d_2, \dots, d_K)$ and $\bar{\mathbf{p}} = (s, \dots, i, j, \dots)$ are the total weights and the node sequence of the path respectively, and $d_k = \sum_{(i, j) \in \mathbf{P}} \omega_k(i, j), k = 1, \dots, K$.
- (3) If an autowave with traveling path $\mathbf{P} = (s \rightarrow \dots \rightarrow i)$ reaches neuron i , the internal activity of neuron i would be $\bar{\mathbf{u}}_i^m = \mathbf{P} = (\bar{\mathbf{p}}, \bar{\mathbf{d}})$ if $\bar{\mathbf{d}}$ satisfies the QoS constraints, where m is a temporal label for the autowave.
- (4) When passed through neuron i , the autowave whose traveling path is \mathbf{P} may be recorded on the threshold $\bar{\theta}_i^m = \mathbf{P} = (\bar{\mathbf{p}}, \bar{\mathbf{d}})$, where m is the index for path \mathbf{P} .
- (5) Neuron i outputs simultaneously all paths in its threshold to its neighborhood. *i.e.*, $\bar{\mathbf{y}}_i = \{\mathbf{P} = (\bar{\mathbf{p}}, \bar{\mathbf{d}}) | \mathbf{P} \in \bar{\theta}_i\}$.

Furthermore, a cost function must be defined for a path \mathbf{P} so that we can make a comparison between two different paths. It is documented that the performance of an MCSP algorithm is closely concerned with the cost function. A variety of cost functions have been proposed and discussed. For example, in [6] the author proposed for a two-constraint problem a cost function $f(\mathbf{P}) = \alpha\omega_1(\mathbf{P}) + \beta\omega_2(\mathbf{P})$, then one can search fast for a feasible path using Dijkstra's shortest path algorithm by minimizing the cost function $f(\mathbf{P})$. However, no performance guarantee for path \mathbf{P} is given with respect to individual constraints. In [11] the authors proposed an algorithm that dynamically adjusts the values of α and β within a logarithmic number of calls to Dijkstra's shortest path algorithm. However the problem is still unsolved. Some researchers [1] have recently proposed a nonlinear cost function whose minimization provides a continuous spectrum of solutions. In this paper, the cost function of a path \mathbf{P} is defined as:

$$f(\mathbf{P}) = \begin{cases} \sum_{k=1}^K \left(\frac{d_k}{C_k} \right)^{\lambda_k} & \mathbf{P} \text{ is feasible} \\ \infty & \text{otherwise} \end{cases} \quad (6)$$

Where C_1, C_2, \dots, C_K are the K QoS constraints, $\lambda_k \geq 1$ is the significant coefficient of the k^{th} QoS parameter.

4.2 Vectorial ACNN Based Algorithm

There are two improvements in vectorial ACNN based algorithm as to the original multi-label algorithm.

One is to limit the total labels of each node. In the original multi-label algorithm, a path can be discarded only when its constraints are all worse than others or don't satisfy the QoS constraints. So a large number of paths would be reserved on each node, which results in an exponential complexity of the algorithm. However, it is proved that the exponential complexity can be reduced to a polynomial one if the total labels of each node are limited to be no more than M (constant). In our algorithm, a maximal M thresholds on each neuron is limited.

Another is to select the paths for reserving. Because of the randomness of the QoS parameters on each link, no cost function can provide an exactly scalar evaluation for a multi-parameter path. A path $\mathbf{P}(s, i) = (s \rightarrow \dots \rightarrow i)$ with good fitness from the source node s to a middle node i may become feasible when outspread to the destination node d and *vice versa*. So the optimal path might not be found if we merely reserve paths according to their fitness. The proportional selection scheme is a simple and effective way to solve this kind of problem, which is well-documented in the heuristic algorithms, such as genetic algorithms [12]. In the proposed algorithm, a proportional selection scheme is used to prevent the optimal solution from being discarded.

The proposed algorithm can be described as following:

Step 1: Initialization. Let $\bar{y}_i(0) = \phi$, $\bar{\theta}_i(0) = \phi$, $\bar{u}_i(0) = \phi$, $\forall i \in V$.

Step 2: Network starting. Let $\bar{y}_s(1) = \{(s, 0, \dots, 0)\}$, then fire the source neuron s and run the network.

Step 3: Internal activity calculating. Calculate the internal activity according to the following equation,

$$\bar{u}_i(t) = \{((\bar{p}, i), \bar{d} + \bar{w}_{ji}) \mid (\bar{p}, \bar{d}) \in \bar{y}_j(t-1), \bar{w}_{ji} \neq 0, \forall j \in V\}.$$

Noted that $\bar{u}_i(t)$ is a path set which collects the arriving autowaves at time t .

Step 4: Thresholds updating.

(1) Combine the two path set $\bar{u}_i(t)$ and $\bar{\theta}_i(t-1)$ into a new set $\bar{A}_i(t)$.

(2) Remove from $\bar{A}_i(t)$ the paths whose distance components d_1, d_2, \dots, d_K are all worse than others. And remove the unfeasible paths from $\bar{A}_i(t)$.

(3) Evaluate the paths in the new path set $\bar{A}_i(t)$ according to the Eq.(6).

(4) Update the threshold $\bar{\theta}_i(t)$ with M paths (if possible) chosen from $\bar{\mathbf{A}}_i(t)$ with the probabilities in inverse proportion to their fitness.

Step 5: Autowaves generating. The neurons generate M autowaves according to the paths in their threshold (or less than M if there are not enough paths in their threshold), and propagate those autowaves to their neighborhoods, *i.e.*,

$$\bar{\mathbf{y}}_i(t) = \{\mathbf{P} = (\bar{\mathbf{p}}, \bar{\mathbf{d}}) \mid \forall \mathbf{P} \in \bar{\theta}_i(t)\}.$$

Step 6: Network stop condition. Let $t = t + 1$. If the optimal path from the source neuron s to the destination neuron d is obtained or $t > Maxtime$, stop the network, otherwise, go to step 3.

The algorithm mentioned-above is called vectorial ACNN. One can have an intuitionistic interpretation of the vectorial ACNN as follows:

Once produced from the source neuron (see Step 2), the autowaves reproduce themselves and propagate from one neuron to another along with the links (see Step 3). When gathering in a neuron, the autowaves compete with each other (see Step 4). The M winners occupy the neuron's threshold waiting for the next competition (see Step 4), and their duplicates will propagate to the next neurons for another competition (see Step 5). This process is recurring again and again until the approximate optimal path is found (see Step 6).

The computation of this algorithm focuses on the threshold updating step (*i.e.*, Step 4). Assuming the average number of adjacent nodes to be m , and the node number of the network to be n , the algorithm needs to calculate and compare $n \cdot (m + 1) \cdot M$ paths at Step 4 in each algorithmic loop. So the computational complexity of the algorithm is $O(n^2(m + 1)M)$ to find a feasible path, which is a polynomial complexity.

5 Computer Simulation

Fig. 3 shows a network with 10 nodes. Let node 3 be the source, and node 10 be the destination. The problem is to find a shortest (cheapest) path from node 3 to node 10, whose cost does not exceed 80 and whose constraint does not exceed 60, *i.e.*, the QoS request is $R=(source=3,destination=10,cost \leq 80,constraint \leq 60)$.

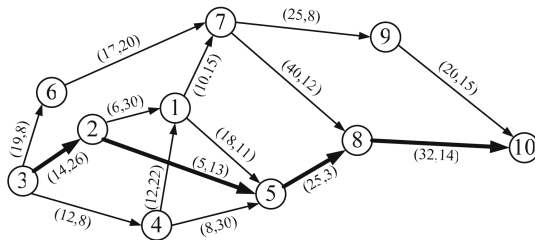


Fig. 3. A 10-node digraph. The two numbers on links are the cost and the constraint respectively, both of which are additive.

The vectorial ACNN based algorithm is used to solve the problem. Where the maximal number of paths reserved on each neuron is $K=2$, and the significant coefficients of the two weights are $r_1 = 2$ and $r_2 = 1$ respectively (see Eq.(6)). Tables 1 and 2 show the result path sets of the neuron's thresholds at first five iterations. Where, a path is denoted as $[(s, node 1, node 2, \dots), (cost, constraint)]$, i.e., the number in first bracket are the nodes sequence of the path, and the two numbers in second bracket are the cost and the constraint of the path respectively.

Table 1. Path sets on the threshold of different neurons (1~5) at first five iterations (t)

t	Node 1	Node 2	Node 3	Node 4	Node 5
1	ϕ	ϕ	[(3),(0,0)]	ϕ	ϕ
2	ϕ	[(3,2),(14,26)]	[(3),(0,0)]	[(3,4),(12,8)]	ϕ
3	[(3,4,1),(24,30)] [(3,2,1),(20,56)]	[(3,2),(14,26)]	[(3),(0,0)]	[(3,4),(12,8)]	[(3,2,5),(19,39)] [(3,4,5),(20,38)]
4	[(3,4,1),(24,30)] [(3,2,1),(20,56)]	[(3,2),(14,26)]	[(3),(0,0)]	[(3,4),(12,8)]	[(3,2,5),(19,39)] [(3,4,5),(20,38)]
5	[(3,4,1),(24,30)] [(3,2,1),(20,56)]	[(3,2),(14,26)]	[(3),(0,0)]	[(3,4),(12,8)]	[(3,2,5),(19,39)] [(3,4,5),(20,38)]

Table 2. Path sets on the threshold of different neurons (6~10) at first five iterations (t)

t	Node 6	Node 7	Node 8	Node 9	Node 10
1	ϕ	ϕ	ϕ	ϕ	ϕ
2	[(3,6),(19,8)]	ϕ	ϕ	ϕ	ϕ
3	[(3,6),(19,8)]	[(3,6,7),(36,28)]	ϕ	ϕ	ϕ
4	[(3,6),(19,8)]	[(3,6,7),(36,28)] [(3,4,1,7),(34,45)]	[(3,4,5,8),(45,41)] [(3,2,5,8),(44,42)]	[(3,6,7,9),(51,36)]	ϕ
5	[(3,6),(19,8)]	[(3,6,7),(36,28)] [(3,4,1,7),(34,45)]	[(3,4,5,8),(45,41)] [(3,2,5,8),(44,42)]	[(3,6,7,9),(51,36)]	<u>[(3,2,5,8,10),(76,56)]</u> [(3,4,5,8,10),(77,55)]

Tables 1 and 2 show that the optimal solution (the underline path on the threshold of neuron 10 at fifth iteration) was found after five iterations. This result path is [(3,2,5,8,10),(76,56)], see the thick path on Fig. 3, which is better than the result path found by Liu's algorithm [7], *i.e.*, the path [(3,4,5,8,10),(77,55)].

Furthermore, comparative simulation experiments of Jaffe's algorithm [6], Liu's algorithm [7] and our algorithm (the vectorial ACNN based algorithm) are done in different scale networks with different constraint number. Table 3 shows the iteration number of one constraint problem in the networks with 10, 100 and 200 nodes. While Table 4 shows the results of two constraint problem in different scale networks. The cost and additive constraints on the links of those networks are all produced randomly by computer, which subject to Gaussian distribution ranging from 1 to 100 with mean 50 and stand deviation 20. The adjacent nodes number is also random ranging from 1 to 5 with mean 3. M is set to be 2, 10 and 20 for 10-node networks, 100-node networks and 200-node networks respectively. For the sake of simplicity, the significant coefficients of the constraints λ_k are all set to be 1. The parameters of Liu's algorithm are set to be the same to document [7].

Tables 3 and 4 show that the iteration number is dramatically reduced using the vectorial ACNN based algorithm. With the increasing of networks scale, the iteration number increases slower than that of the Liu's algorithm. The results are all better than that of Liu's algorithm.

Table 3. Iteration number of one constraints problem in 3 different scale networks

method	10 nodes	100 nodes	200 nodes
Jaffe's algorithm	9	167	532
Liu's algorithm	5	89	204
Our algorithm	5	76	159

Table 4. Iteration number of two constraints problem in 3-different scale networks

method	10 nodes	100 nodes	200 nodes
Jaffe's algorithm	39	716	6113
Liu's algorithm	25	328	1527
Our algorithm	21	262	967

6 Conclusions

Multi-constraint quality-of-service (QoS) routing, which is in fact an MCSP problem, will become increasingly important as the Internet evolves to support real-time services. In this paper, a vectorial ACNN based algorithm is proposed for MCSP problem, which can find the approximate optimal path in polynomial time. Performance of the proposed algorithm is improved greatly comparing to the original multi-label routing algorithm (*i.e.*, Jaffe's algorithm). Firstly, the proposed algorithm

updates all neurons' threshold synchronously, *i.e.*, it is a parallel algorithm. Secondly, only the maximal M paths are reserved on the threshold of each neuron, which reduces greatly the memory space and the computation of the algorithm needed. Thirdly, although the M -paths limited scheme may discard the optimal paths sometimes because of the randomness of the constraints and the rigidity of the cost function, the proportional selection scheme would afford the optimal autowaves (*i.e.*, paths) enough opportunity to revive or to survive the search process. Comparing experiments are given in the paper, and the results show the efficiency of the proposed algorithm. In conclusion, the proposed algorithm has the characteristics of parallelism, efficiency and lower computational complexity.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Grant No. 60574039) and the "863" Project of National Ministry of Science and Technology (Grant No. 2006AA03A175).

References

1. Korkmaz, T., Krunz, M.: Multi-constrained optimal path selection. The 20th Annual Joint Conference of the IEEE Computer and Communications Societies **2** (2001) 834-843.
2. Xu, D., Chen, Y., Xiong, Y., Qiao, C.: On the Complexity of and Algorithms for Finding the Shortest Path With a Disjoint Counterpart, IEEE/ACM Trans. Networking, **14** (2006) 147-158
3. Wang, Z., Crowcroft, J.: Quality-of-service routing for supporting multimedia applications. IEEE J. Select. Area. Commun. **14** (1996) 1219-1234
4. Jia, Z., Varaiya, P.: Heuristic Methods for Delay Constrained Least Cost Routing Using k-Shortest-Path, IEEE Trans. AC. **17** (2006) 707-712
5. Dumitrescu, I., Boland, N.: Improved Preprocessing, Labeling and Scaling Algorithms for the Weight-Constrained Shortest Path Problem, Networks **42** (2003) 135-153
6. Jaffe, J.M.: Algorithm for finding paths with multiple constraints, Networks **14** (1984) 95-116
7. Liu, J., Niu, Z., Zheng, J.: An improved routing algorithm subject to multiple constraints for ATM networks. ACTA ELECTRONICA SINICA **27** (1999) 4-8 (In Chinese)
8. Dong, J., Wang, W., Zhang, J.: Accumulative competition neural network for shortest path tree computation. International Conference on Machine Learning and Cybernetics, Vol.III, Xi'an China (2003) 1157-1161
9. Dong, J., Zhang, J.: Accumulating Competition Neural Networks based Multiple Constrained Routing Algorithm. Control and Decision **19** (2004) 751-755
10. Wang, Z.: On the complexity of quality of service routing. Information Processing Letters **69** (1999) 111-114
11. Korkmaz, T., Krunz, M., Tragoudas, S.: An Efficient Algorithm for Finding a Path Subject to Two Additive Constraints. Proceedings of the ACM SIGMENTRICS **1** (2000) 318-327
12. Gelenbe, E., Liu, P., Laine, J.: Genetic algorithms for route discovery. IEEE Trans. on SMC--Part B **99** (2006) 1247 - 1254