Research Article

An Efficient Approximate Algorithm for Disjoint QoS Routing

Zhanke Yu, Feng Ma, Jingxia Liu, Bingxin Hu, and Zhaodong Zhang

College of Communication Engineering, PLA University of Science and Technology, Nanjing 210007, China

Correspondence should be addressed to Zhanke Yu; jackty_2004@163.com

Received 23 June 2013; Revised 2 November 2013; Accepted 4 November 2013

Academic Editor: John Gunnar Carlsson

Copyright © 2013 Zhanke Yu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Disjoint routing is used to find the disjoint paths between a source and a destination subject to QoS requirements. Disjoint QoS routing is an effective strategy to achieve robustness, load balancing, congestion reduction, and an increased throughput in computer networks. For multiple additive constraints, disjoint QoS routing is an NP-complete class that cannot be exactly solved in polynomial time. In the paper, the disjoint QoS routing problem was formulated as a 0-1 integer linear programming. The complicating constraints were included in the objective function using an adaptive penalty function. The special model with a totally unimodular constraint coefficient matrix was constructed and could be solved rapidly as a linear programming. An efficient algorithm using an adaptive penalty function and 0-1 integer linear programming for the disjoint QoS routing problems was designed. The proposed algorithm could obtain the optimal solution, considerably reducing the computational time consumption and improving the computational efficiency. Theoretical analysis and simulation experiments were performed to evaluate the proposed algorithm performance. Through the establishment of random network topologies using Matlab, the average running time, the optimal objective value, and the success rate were evaluated based on the optimal values obtained in Cplex. The simulation experiments validated the effectiveness of the proposed heuristic algorithm.

1. Introduction

In recent years, disjoint QoS routing has been given much attention because of its practical significance to various applications, such as reliable routing, load balancing, congestion reduction, and enhanced throughput [1–7]. This study determines the QoS-aware k disjoint paths from a source to a destination. These paths must be subjected to QoS constraints. The paths may be node-disjoint or edge-disjoint, and the network may be directed or undirected. Li et al. proved that all four disjoint QoS routing versions are strongly NP-complete, even for $k = 2$ [8]. In general, a link-disjoint path algorithm can be extended to a node-disjoint algorithm through the concept of node splitting [9]. Accordingly, we assume that the paths are edge-disjoint paths unless specified otherwise. The QoS metrics of a path can either be additive, multiplicative, or min./max. The path weight of the additive metrics (e.g., delay) is equal to the sum of the QoS weights of the links on the path. The multiplicative metrics (e.g., packet loss) can be transformed into additive measures using a logarithmic function. The path weight of the min./max. metrics (e.g., bandwidth) presents

the minimum/maximum of the QoS weights defining the path. Constraints on the min./max. QoS metrics can be easily settled by omitting all links (or disconnected nodes) that do not satisfy the requested QoS constraints. In practice, the constraints on additive QoS metrics are more difficult, and therefore, without loss of generality, the QoS metrics are assumed to be additive.

Numerous approaches for solving disjoint QoS routing problems have been proposed [10–19]. Almost all of the existing approaches for solving disjoint QoS routing are based on the classical methods for solving the shortest path problem, such as Dijkstra and Ford-Bellman algorithms [10]. Some approaches are polynomial ϵ -approximate solution methods [11–13], and others are based on integer linear programming [14–16, 18, 19]. In addition, several intelligent algorithms are used for these problems [17]. We aim to use the algorithm based on 0-1 integer linear programming (ILP).

Suurballe proposed an algorithm that finds the k disjoint paths with a minimal total length using the path augmentation method [20]. The main idea is to construct a solution set of two disjoint paths based on the shortest path and shortest augmenting path. The k disjoint paths can be

obtained by augmenting the $k-1$ optimal disjoint paths using this algorithm. Bhandari made an extension to Suurballe's algorithm to solve the span-disjoint path problem in more complex structured networks [10]. Carlyle considered the constrained shortest path problem as an integer program and presented a Lagrangian dual method to solve the ILP problems [14]. The main idea is Lagrangianizing those constraints, optimizing the resulting Lagrangian function, and then closing the duality gaps by enumerating the near-shortest paths, measured with respect to the Lagrangianized length. However, adjusting the Lagrangian multiplier is difficult. Son et al. formulated the QoS routing problems in the form of ILP problems and proposed a routing algorithm that solved the ILP problems based on the difference of convex function (DC) programming and DC algorithms (DCA) [15]. DC programming and DCA are efficient approaches for a nonconvex continuous optimization. The objective and constraint functions are written as the difference of two convex functions. Xiong et al. propose design principles of the exact algorithm for multiconstrained shortest link-disjoint paths based on analyzing the properties of optimal solutions and design an exact algorithm, called the link-disjoint optimal multiconstrained paths algorithm (LIDOMPA). The proposed algorithm can reduce the search space without loss of exactness by introducing three concepts, namely, the candidate optimal solution, the contractive constraint vector, and structure-aware nondominance [5].

In this paper, we propose an efficient approach using an adaptive penalty function and 0-1 ILP to solve the disjoint QoS routing problems. Compared with the exact algorithm designed in Xiong's paper, our algorithm is a heuristic algorithm and aims to obtain high-quality feasible solutions in short time. We formulate the disjoint QoS routing problem as a 0-1 ILP, labeled as DQSR. This formulation is computationally intractable because of integrality. We relax the problem by performing a partial Lagrangian relaxation, such that we exploit the special network structure of the relaxed problem for efficient algorithms. The algorithm includes the complicated 0-1 ILP constraints in the objective function as penalty terms and obtains the Lagrangian relaxation 0-1 integer linear problem. After constructing the special model with a totally unimodular constraint coefficient matrix, the relaxed 0-1 integer linear problem can be solved rapidly as linear programming. The numerical results show the effectiveness of the proposed algorithm.

The rest of the paper is organized as follows. In Section 2, we define the DQSR problem. In Section 3, we introduce the linear relaxation of DQSR. An algorithm based on an adaptive penalty function and 0-1 ILP for the problem is described in Section 4. The numerical simulation is reported in Section 5. Finally, conclusions and future works are presented in Section 6.

2. Problem Formulation

Let $G = (V, E)$ be the directed graph representing a network topology having the set of nodes V and the set of links E . The links are assumed to be bidirectional. The number of nodes,

the number of links, and the number of QoS measures are denoted, respectively, by n , m , and p . Let $(u, v) \in E$ be the link from node u to node v , where $u, v \in V$. Each link is characterized by a p -dimensional link weight vector. Each vector is composed of p nonnegative QoS weight components as $w_i(u, v)$, where $i = 1, \dots, p$, $(u, v) \in E$.

Definition 1 (disjoint QoS routing). Considering a network $G = (V, E)$, each link (u, v) is specified by p additive QoS weights $w_i(u, v) \geq 0$, where $i = 1, \dots, p$. Given that p constraint bounds L_i , the disjoint QoS routing problem is finding a set of k disjoint paths from a source to a destination $P_1, P_2, \dots, P_k$, such that the total cost of the k paths is the minimum, and each path satisfies the QoS constraints. These constraints are stated as follows [15].

The binary variable y_{uvl} is defined as

$$y_{uvl} = \begin{cases} 1, & (u, v) \in P_l, \quad l = 1, \dots, k, \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Every disjoint path P_l ($l = 1, \dots, k$) from a source node s to a destination node t should satisfy the following constraints:

$$\begin{aligned} \sum_{\{v|(u,v) \in E\}} y_{uvl} - \sum_{\{v|(v,u) \in E\}} y_{vul} &= 1, \quad \text{for } u = s, \\ \sum_{\{v|(u,v) \in E\}} y_{uvl} - \sum_{\{v|(v,u) \in E\}} y_{vul} &= 0, \quad \forall u \in V \setminus \{s, t\}, \\ \sum_{\{v|(u,v) \in E\}} y_{uvl} - \sum_{\{v|(v,u) \in E\}} y_{vul} &= -1, \quad \text{for } u = t. \end{aligned} \quad (2)$$

Moreover, the following constraint ensures that each link belongs to at most one path P_l ($l = 1, \dots, k$):

$$\sum_{l=1}^k y_{uvl} \leq 1, \quad \forall (u, v) \in E. \quad (3)$$

The QoS constraints for each path P_l ($l = 1, \dots, k$) are expressed as

$$\sum_{(u,v) \in E} w_i(u, v) y_{uvl} \leq L_i, \quad i = 1, 2, \dots, p. \quad (4)$$

Using constraint (4), every path P_l ($l = 1, \dots, k$) is ensured to satisfy the QoS constraints L_i ($i = 1, 2, \dots, p$).

If the value of $w_i(u, v)$ is fixed to each link $(u, v) \in E$ (e.g., the cost of sending one message or one unit data on the link (u, v) or the time delay on the link (u, v)), then the QoS constraints are linear constraints. When $w_i(u, v)$ is also a variable, the problem becomes more complex.

Disjoint QoS routing can be formulated as a 0-1 integer linear programming as follows.

DQSR:

$$\begin{aligned}
 \text{Min.} \quad & \sum_{l=1}^k \sum_{(u,v) \in E} c(u,v) y_{uvl} \\
 \text{subject to} \quad & \text{for } l = 1, 2, \dots, k, \forall u \in V \\
 & \sum_{\{v|(u,v) \in E\}} y_{uvl} - \sum_{\{v|(v,u) \in E\}} y_{vul} = 1, \\
 & \quad \text{for } u = s \\
 & \sum_{\{v|(u,v) \in E\}} y_{uvl} - \sum_{\{v|(v,u) \in E\}} y_{vul} = 0, \\
 & \quad \forall u \in V \setminus \{s, t\}, \\
 & \sum_{\{v|(u,v) \in E\}} y_{uvl} - \sum_{\{v|(v,u) \in E\}} y_{vul} = -1, \\
 & \quad \text{for } u = t \\
 & \sum_{l=1}^k y_{uvl} \leq 1, \quad \forall (u,v) \in E, \\
 & \sum_{(u,v) \in E} w_i(u,v) y_{uvl} \leq L_i, \quad i = 1, 2, \dots, p \\
 & y_{uvl} \in \{0, 1\}, \quad (u,v) \in E.
 \end{aligned} \tag{5}$$

In general, the solutions to the above problem may not be integral. However, every integer solution defines a set of k disjoint $s-t$ paths. Thus, an integer solution $Y_l = \{y_{uvl}\}_{(u,v) \in E}$ for $l = 1, 2, \dots, k$ is the flow vector corresponding to the l th path P_l that is, the link (u,v) is on the path P_l if and only if $y_{uvl} = 1$.

3. Linear Relaxation of DQSR

DQSR is a 0-1 ILP formulation. This formulation is computationally intractable because of integrality. For networks involving a small number of nodes and links, these problems can be solved using any general-purpose 0-1 ILP package. For larger networks, faster algorithms are desired. Thus, we are interested in solving these problems after relaxing the integrality requirement and exploiting the special network structure of these problems for efficient algorithms.

Theorem 2 (see [21]). *The following statements are equivalent.*

- (i) *A is a totally unimodular matrix.*
- (ii) *For every $Q \subseteq M = \{1, 2, \dots, m\}$, a partition to Q_1 and Q_2 of Q exists such that*

$$\left| \sum_{i \in Q_1} a_{ij} - \sum_{i \in Q_2} a_{ij} \right| \leq 1, \quad \text{for } j = 1, 2, \dots, n. \tag{6}$$

Theorem 3 (see [21]). *Consider the following integer linear programming (P1):*

$$\begin{aligned}
 \text{Min.} \quad & cx \\
 \text{subject to} \quad & Ax \leq b, \\
 & \text{where } x \geq 0, \text{ and } x \text{ an integer vector.}
 \end{aligned} \tag{P1}$$

Relaxing the integrality constraints one gets the following relaxed version (P2):

$$\begin{aligned}
 \text{Min.} \quad & cx \\
 \text{subject to} \quad & Ax \leq b, \\
 & \text{where } x \geq 0.
 \end{aligned} \tag{P2}$$

If A is totally unimodular and b is an integer vector, then one can neglect the integer constraints of (P1) and solve the remaining (P2) as an ordinary linear programming using the simplex method. The obtained optimum basic feasible solution of (P2) is actually an optimum solution of (P1).

Theorem 3 shows that without the integrality condition, we can obtain the integral values of the variables because matrix A is totally unimodular.

3.1. Adaptive Penalty Function. We use a penalty function to deal with the constraint matrix in (4). The QoS constraints are added as penalty terms to the objective of DQSR. The penalty function is defined as follows:

$$\sum_{l=1}^k \sum_{i=1}^p \lambda_i^t \left(\sum_{(u,v) \in E} w_i(u,v) y_{uvl} - L_i \right). \tag{7}$$

The elements of the nonnegative vector $\lambda_i^t = (\lambda_1, \lambda_2, \dots, \lambda_p)$ are called Lagrangian multipliers, where t is the number of iterations.

If no violation occurs, λ_i^t is zero; otherwise, it is positive. If the penalty is either too large or too small, the problem could be very hard. A large penalty causes the algorithm to converge to a feasible solution very quickly, even if the solution is far from the optimal solution, whereas a small penalty causes too much time spent in searching for an unfeasible region. We design an adaptive penalty function to solve this problem as follows.

The Lagrangian multipliers are updated using the following formula:

$$\lambda_i^{t+1} = \lambda_i^t + \beta_i \Delta \lambda_i^t, \tag{8}$$

where $\Delta \lambda_i^t$ is the value of violation and β_i is adaptive penalty coefficient for constraint i . $\Delta \lambda_i^t$ is defined as follows:

$$\Delta \lambda_i^t = \max \left(0, \max \left(\sum_{(u,v) \in P_l} w_i(u,v) - L_i \right) \right) \tag{9}$$

($l = 1, 2, \dots, k; i = 1, 2, \dots, p$).

If the inequality holds, $\max(\sum_{(u,v) \in P_l} w_i(u,v) - L_i) \leq 0$, $\Delta \lambda_i^t$ is zero. Therefore, the constraint does not effect λ_i^t . If the constraint is violated, that is, $\Delta \lambda_i^t > 0$, a large term is added to the function, such that the solution is pushed back toward

the feasible region. The severity of the penalty depends on the adaptive penalty coefficient β_i . β_i is described as follows:

$$\beta_i = \begin{cases} 0, & \gamma_i \leq 0, \\ \frac{1}{\alpha}, & 0 < \gamma_i \leq 0.2, \\ 1, & 0.2 < \gamma_i \leq 0.5, \\ \alpha, & \gamma_i > 0.5, \end{cases} \quad (10)$$

where α is a constant greater than 1 and γ_i is the degree of violation. γ_i is defined as follows:

$$\gamma_i = \frac{\Delta \lambda_i^t}{\max \left(\sum_{(u,v) \in P_l} w_i(u,v) \right)}, \quad (11)$$

$$(l = 1, 2, \dots, k; i = 1, 2, \dots, p).$$

The penalty coefficient β_i is updated according to the information gathered from the solution, the degree of violation γ_i . The objective is to avoid a penalty that is too large or too small.

3.2. Linear Relaxation of DQSR. Based on Theorem 3, we relax the integrality constraints of DQSR to solve the DQSR problem using (4) in the penalty function, constructing the following Lagrangian relaxed form.

RELAX-DQSR:

$$\begin{aligned} \text{Min.} \quad & \sum_{l=1}^k \sum_{(u,v) \in E} c(u,v) y_{uvl} \\ & + \sum_{l=1}^k \sum_{i=1}^p \lambda_i^t \left(\sum_{(u,v) \in E} w_i(u,v) y_{uvl} - L_i \right) \\ \text{subject to} \quad & \text{for } l = 1, 2, \dots, k, \forall u \in V \\ Q_1 \quad & \begin{cases} \sum_{\{v|(u,v) \in E\}} y_{uvl} - \sum_{\{v|(v,u) \in E\}} y_{vul} = 1, & \text{for } u = s \\ \sum_{\{v|(u,v) \in E\}} y_{uvl} - \sum_{\{v|(v,u) \in E\}} y_{vul} = 0, & \forall u \in V \setminus \{s, t\}, \\ \sum_{\{v|(u,v) \in E\}} y_{uvl} - \sum_{\{v|(v,u) \in E\}} y_{vul} = -1, & \text{for } u = t \end{cases} \\ Q_2 \quad & \sum_{l=1}^k y_{uvl} \leq 1, \quad y_{uvl} \geq 0, \quad \forall (u,v) \in E. \end{aligned} \quad (12)$$

Theorem 4. The constraint matrix of RELAX-DQSR is totally unimodular.

Proof. Partition $Q \subseteq M = \{1, 2, \dots, m\}$ of the constraint matrix into Q_1 and Q_2 , as shown in the above RELAX-DQSR. For Q_1 and Q_2 , we have

$$\left| \sum_{i \in Q_1} a_{ij} - \sum_{i \in Q_2} a_{ij} \right| \leq 1, \quad \text{for } j = 1, 2, \dots, n. \quad (13)$$

Based on Theorem 2, the constraint matrix is totally unimodular. $\square$

Theorem 5. The optimum solution of RELAX-DQSR is a 0-1 integer vector.

Proof. Based on Theorem 4, the constraint matrix of RELAX-DQSR is totally unimodular. Based on Theorem 3, the optimum solution of RELAX-DQSR is an integer vector. Given that $\sum_{l=1}^k y_{uvl} \leq 1$ and $y_{uvl} \geq 0$, the optimum solution of RELAX-DQSR is a 0-1 integer vector. $\square$

Theorem 6. Let y_{uvl} be an optimum solution of RELAX-DQSR; one has the following.

- (i) If y_{uvl} satisfies (4), then the paths $P_1, P_2, \dots, P_k$ obtained by using (1) are the feasible solutions of DQSR.
- (ii) If y_{uvl} satisfies (4), and

$$\left(\sum_{(u,v) \in E} w_i(u,v) y_{uvl} - L_i \right) \lambda_i = 0, \quad (14)$$

$$i = 1, 2, \dots, p; \quad l = 1, 2, \dots, k,$$

then the paths $P_1, P_2, \dots, P_k$ obtained by using (1) are the optimum solution of DQSR.

Proof

- (i) Every 0-1 integer solution defines a set of k disjoint paths. Given that y_{uvl} is an optimum solution of RELAX-DQSR and satisfies (4), thus the paths $P_1, P_2, \dots, P_k$ obtained by using (1) are the feasible solutions of DQSR.
- (ii) Let $P = \{P_1, P_2, \dots, P_k\}$. Given that y_{uvl} satisfies (4), thus P is the feasible solution of DQSR. Given that y_{uvl} satisfies (14) and P is the optimum solution of RELAX-DQSR, letting $P' = \{P'_1, P'_2, \dots, P'_k\}$ be a feasible solution of DQSR, we have

$$\begin{aligned} & \sum_{l=1}^k \sum_{(u,v) \in P'} c(u,v) y_{uvl} \\ & \geq \sum_{l=1}^k \sum_{(u,v) \in P'} c(u,v) y_{uvl} \\ & \quad + \sum_{l=1}^k \sum_{i=1}^p \lambda_i^t \left(\sum_{(u,v) \in P'} w_i(u,v) y_{uvl} - L_i \right) \end{aligned} \quad (15)$$

$$\begin{aligned} & \geq \sum_{l=1}^k \sum_{(u,v) \in P} c(u,v) y_{uvl} \\ & \quad + \sum_{l=1}^k \sum_{i=1}^p \lambda_i^t \left(\sum_{(u,v) \in P} w_i(u,v) y_{uvl} - L_i \right) \\ & = \sum_{l=1}^k \sum_{(u,v) \in P} c(u,v) y_{uvl}. \end{aligned}$$

Hypothesizing that P is not the optimum solution of DQSR, hence, an optimum solution P^* exists, such that

$$\sum_{l=1}^k \sum_{(u,v) \in P^*} c(u,v) y_{uvl} < \sum_{l=1}^k \sum_{(u,v) \in P} c(u,v) y_{uvl}. \quad (16)$$

This statement contradicts (15). Hence, $P = \{P_1, P_2, \dots, P_k\}$ is the optimum solution of DQSR.

Applying Theorems 3, 5, and 6, RELAX-DQSR can be easily solved as a linear programming, and the optimum solution obtained is a 0-1 integer vector. Hence, DQSR can be solved by iterating the calculation of RELAX-DQSR. $\square$

4. Proposed Algorithm

4.1. Algorithm Solving k -Disjoint QoS Routing. The proposed heuristic algorithm based on an adaptive penalty function and ILP for disjoint QoS routing problem can be finally summarized as follows.

Algorithm 7.

Step 1 (initialize). The initial values are as follows: $t = 0$, the maximum iterate number T , the initial Lagrangian multipliers $\lambda_i^0 = 0$ ($i = 1, 2, \dots, p$) and the penalty parameter $\alpha = 2$.

Step 2 (formulate). The disjoint QoS routing problem is formulated as DQSR.

Step 3 (relax). The integrality constraints of DQSR, including the penalty term, are relaxed, and then RELAX-DQSR is obtained.

Step 4 (solve). RELAX-DQSR is solved, and k paths P_l ($l = 1, \dots, k$) are obtained using y_{uvl} and (1).

Step 5 (evaluate). For p constraints and k paths, $\Delta \lambda_i^t$ is computed according to (9). If $\Delta \lambda_i^t = 0$ ($i = 1, 2, \dots, p$), then the computation is stopped with the solution k paths P_l ($l = 1, \dots, k$). Otherwise, Step 6 is performed.

Step 6 (penalize and update). If $t < T$, multipliers λ_i^{t+1} are computed according to (8), $t := t + 1$. RELAX-DQSR is updated, and step 3 is repeated.

The proposed algorithm can make the computation more efficient because solving the relaxation linear programming of a 0-1 ILP is much easier than solving a 0-1 integer linear programming. Using a penalty function can guarantee the feasibility of a solution, thus obtaining k paths that satisfy the QoS requirements. Evidently, based on Theorem 6, we have the following.

Theorem 8. Let P_l ($l = 1, \dots, k$) be a solution obtained by using our algorithm; thus,

- (i) if our algorithm stops in Step 5 and P_l satisfies (4) and (14), then P_l is the optimum solution of DQSR;
- (ii) if our algorithm stops in Step 5 and P_l only satisfies (4), then P_l is the feasible solution of DQSR.

FIGURE 1: Topology and parameters of the networks.

FIGURE 2: The first solution.

4.2. Illustration with an Example. We consider the following example to provide a clear description of the proposed algorithm. Figure 1 shows a network topology consisting of 8 nodes and 13 links. Each link is denoted by the notation (c, d) , representing the cost and delay of the link, respectively.

Assuming that the source node is 1 and the destination node is 8, the QoS requirements consist of only the delay L_1 , where $L_1 = 15$. We find two disjoint paths from the source node 1 to the destination node 8, having a minimum total cost and satisfying the QoS requirement.

The iterative process is as follows.

- (1) *Initialize.* Set $t = 0$, $T = 20$, $\lambda_1^0 = 0$, and $\alpha = 2$.
- (2) *Formulate, Relax, and Solve.* The disjoint QoS routing is formulated as DQSR, and the relaxation form RELAX-DQSR is solved. The solution indicates that the two paths are $V_1 \rightarrow V_5 \rightarrow V_8$ and $V_1 \rightarrow V_2 \rightarrow V_6 \rightarrow V_7 \rightarrow V_8$, as shown in Figure 2.
- (3) The two paths are evaluated with constraint L_1 . The cost and delay of the first path are 7 and 9, respectively. The cost and delay of the second path are 15 and 20, respectively. The total cost of the two paths is the minimum. However, the delay of the second path destroys the delay constraint L_1 .
- (4) *Penalize and Update.* The multiplier is computed according to (8) and $\lambda_1^1 = 5$ is obtained. RELAX-DQSR is updated. Step 3 is repeated, and two new paths, $V_1 \rightarrow V_5 \rightarrow V_8$ and $V_1 \rightarrow V_3 \rightarrow V_4 \rightarrow V_8$, are obtained, as shown in Figure 3. The cost and delay of the first path are 7 and 9, respectively. The cost and delay of the second path are 17 and 12, respectively. Both paths are feasible paths, satisfying the two QoS requirements.

TABLE 1: Comparative results between our algorithm and Cplex.

Dataset	Metrics	Our algorithm					Cplex	
		CON	OBJ	OBJ-type	Time (in s)	ITE	OBJ	Time (in s)
Dataset 1	$n = 400$ $m = 3694$ $k = 2$	1	100	O	0.016	0	100	0.078
		3	100	O	0.016	0	100	0.125
		5	100	O	0.015	0	100	0.188
		7	113	F	0.016	1	109	0.281
		9	113	F	0.016	1	109	0.344
Dataset 2	$n = 800$ $m = 6666$ $k = 2$	1	117	F	0.047	1	117	0.171
		3	128	F	0.046	1	120	0.313
		5	120	F	0.048	1	120	0.484
		7	120	F	0.046	1	120	0.641
		9	120	F	0.046	1	120	0.843
Dataset 3	$n = 1200$ $m = 9838$ $k = 2$	1	117	O	0.031	0	117	0.266
		3	118	F	0.246	4	118	0.594
		5	143	F	0.405	8	123	1.015
		7	146	F	0.187	3	123	1.359
		9	136	F	0.157	3	136	3.516
Dataset 4	$n = 1600$ $m = 12902$ $k = 2$	1	165	F	0.109	0	151	0.485
		3	167	F	0.094	0	154	2.266
		5	162	F	0.11	0	155	2.422
		7	155	O	0.109	0	155	2.907
		9	155	O	0.078	0	155	3.891
Dataset 5	$n = 2000$ $m = 18352$ $k = 2$	1	132	O	0.094	0	132	0.688
		3	135	F	0.156	1	135	1.641
		5	135	F	0.172	1	135	2.469
		7	158	F	0.328	3	143	3.672
		9	158	F	0.266	2	143	4.562

FIGURE 3: The second solution.

5. Performance Evaluation

We demonstrate the effectiveness of our algorithm by comparing the results with the optimal values obtained by solving the 0-1 ILP formulation using the Cplex 12 solver. Three main performance metrics are considered, namely, average running time, optimal objective value, and the success rate. Our algorithm was coded in Matlab 2009 and implemented on an Intel Core 2, 2.53 GHz CPU with 2 GB RAM running on Windows XP.

5.1. Network Topology. We consider random network topologies generated using Waxman's model [22]. For a topology $G = (V, E)$, the number of nodes, links, disjoint paths, and QoS metrics is, respectively, denoted as n , m , k , and p . Nodes 1 and n are chosen as the source and target nodes, respectively. The link costs, delays, and other QoS metrics are uniform randomly generated integers in the range from 1 to 20. For each QoS metric $i \in p$, the metric bounds are $L_i = \alpha L_{\max,i} + (1-\alpha)L_{\min,i}$, where $L_{\min,i}$ denotes the total value of the minimum-metric path with respect to the metric i , and $L_{\max,i}$ denotes the total values, with respect to the metric i , of the shortest path (with respect to cost). We examine α , set to the low (L), medium (M), and high (H) values of 0.05, 0.50, and 0.95, respectively; "L-instances" are tightly constrained, "H-instances" are loosely constrained, and "M-instances" are in between.

5.2. Results and Discussion. We generate ten random datasets using Waxman's model with 400, 800, 1200, 1600, and 2000 nodes. Table 1 lists the number of nodes, links, disjoint paths, and QoS constraints, denoted by n , m , k , and CON, respectively. The QoS metrics are loosely constrained. The numerical results obtained from our algorithm for these

FIGURE 4: Comparison of the average running times and the variation in the average running time with the number of nodes.

FIGURE 5: Comparison of the average running times and the variation in the average running time with the number of constraints.

datasets are also listed in Table 1. These results include the objective value, solution type, average running time, and number of iterations, denoted by OBJ, OBJ-type, time, and ITE, respectively. The solution can be classified into three types, namely, optimal solution (O), feasible solution (F), and null (N). The average running times are the average values over twenty iterations. The numerical results obtained from Cplex are also given in Table 1.

FIGURE 6: Comparison of the objective values.

FIGURE 7: Comparison of the success rates.

From the numerical results, our algorithm is much faster than the commercial quality Cplex package for the randomly generated topologies and converges after only a few iterations. In some cases, the objective values given by our algorithm and the optimal values obtained by Cplex are the same, that is, 15 out of 25 problems for the randomly generated datasets. For the remaining cases, the difference is small.

Figure 4 shows a comparison of the average running times between our algorithm and Cplex for the datasets in

Table 1. Our algorithm is much faster than Cplex. With an increase in the number of nodes, the average running time of both algorithms also increases. However, the increase rate of our algorithm is far less than Cplex.

Figure 5 plots the average running time against the number of constraints. The average running time of our algorithm was not significantly affected by the number of constraints p .

Figure 6 shows the comparison of the objective values obtained by the two algorithms. Some objective values are the same, whereas others are just close.

Finally, we analyse the success rate of the proposed algorithm under three different constraint conditions, namely, tightly constrained, generally constrained, and loosely constrained. In the simulations, the number of network nodes N is assumed to be 400, 800, 1200, 1600, and 2000. For each N , 20000 random topologies are generated under three different constraint conditions. Figure 7 plots the success rate of the proposed algorithm against the three different constraint conditions and shows a comparison of the success rate between our algorithm and Cplex. The results show that the success rates of our algorithm is significantly higher than Cplex under three different constraint conditions. The success rate is relatively low under tight constraint condition because tight constraints resulting feasible solution may not exist. With the relaxation of constraint conditions, the success rate gradually increased. The success rate is close to 100% under loose constraint condition.

6. Conclusion

In this paper, we study the NP-complete disjoint QoS routing. An efficient algorithm based on an adaptive penalty function and 0-1 ILP is proposed for solving the disjoint QoS routing problem. The computational results obtained indicated that the proposed algorithm is efficient and much faster than the commercial quality Cplex package for the randomly generated topologies.

Acknowledgments

This work was supported by the National Nature Science Foundation of China (no. 70971136) and the Youth Foundation of Institute of Sciences, PLA University of Science and Technology (no. KYLYZL001235).

References

- [1] K. Xiong, Z.-D. Qiu, Y. Zhang, and H.-K. Zhang, "Link-disjoint routing algorithm under multiple additive QoS constraints," *Journal on Communications*, vol. 31, no. 6, pp. 127–135, 2010.
- [2] S. Ramasubramanian, H. Krishnamoorthy, and M. Krunz, "Disjoint multipath routing using colored trees," *Computer Networks*, vol. 51, no. 8, pp. 2163–2180, 2007.
- [3] C. Liu, M. Yarvis, W. S. Conner, and X. Guo, "Guaranteed on-demand discovery of node-disjoint paths in ad hoc networks," *Computer Communications*, vol. 30, no. 14-15, pp. 2917–2930, 2007.
- [4] Y. Guo, F. Kuipers, and P. van Mieghem, "Link-disjoint paths for reliable QoS routing," *International Journal of Communication Systems*, vol. 16, no. 9, pp. 779–798, 2003.
- [5] K. Xiong, Z.-D. Qiu, Y. Guo, and H. Zhang, "Multi-constrained shortest disjoint paths for reliable QoS Routing," *ETRI Journal*, vol. 31, no. 5, pp. 534–544, 2009.
- [6] P. P. Pham and S. Perreau, "Performance analysis of reactive shortest path and multi-path routing mechanism with load balance," in *Proceedings of the 22nd Annual Joint Conference on the IEEE Computer and Communications Societies (INFOCOM '03)*, pp. 251–259, IEEE Press, San Francisco, Calif, USA, April 2003.
- [7] S. Murthy and J. J. Garcia-Luna-Aceves, "Congestion-oriented shortest multipath routing," in *Proceedings of the 15th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '96)*, pp. 1028–1036, IEEE Press, San Francisco, Calif, USA, March 1996.
- [8] C.-L. Li, S. T. McCormick, and D. Simchi-Levi, "Finding disjoint paths with different path-costs: complexity and algorithms," *Networks*, vol. 22, no. 7, pp. 653–667, 1992.
- [9] J. W. Suurballe and R. E. Tarjan, "A quick method for finding shortest pairs of disjoint paths," *Networks*, vol. 14, no. 2, pp. 325–336, 1984.
- [10] R. Bhandari, "Optimal diverse routing in telecommunication fiber networks," in *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM '94)*, pp. 1498–1508, IEEE Press, June 1994.
- [11] D. H. Lorenz, A. Orda, D. Raz, and Y. Shavitt, "Efficient QoS partition and routing of unicast and multicast," in *Proceedings of IEEE/IFIP International Workshop on Quality of Service (IWQOS '00)*, pp. 1336–1447, IEEE Press, Pittsburgh, Pa, USA, 2000.
- [12] G. Liu and K. G. Ramakrishnan, "A*Prune: an algorithm for finding K shortest paths subject to multiple constraints," in *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '01)*, pp. 743–749, IEEE Press, Anchorage, Alaska, USA, April 2001.
- [13] A. Orda and A. Sprintson, "Efficient algorithms for computing disjoint QoS paths," in *Proceedings of the Annual Joint Conference on the IEEE Computer and Communications Societies (INFOCOM '04)*, pp. 727–738, IEEE Press, Hong Kong, China, March 2004.
- [14] W. M. Carlyle, J. O. Royset, and R. K. Wood, "Lagrangian relaxation and enumeration for solving constrained shortest-path problems," *Networks*, vol. 52, no. 4, pp. 256–270, 2008.
- [15] T. Son, L. An, D. Khadraoui, and P. Tao, "Solving QoS routing problems by DCA," in *Intelligent Information and Database Systems: 2nd International Conference (ACIIDS '10)*, pp. 460–470, Springer, Berlin, Germany, 2010.
- [16] S. G. Kolliopoulos and C. Stein, "Approximating disjoint-path problems using packing integer programs," *Mathematical Programming*, vol. 99, no. 1, pp. 63–87, 2004.
- [17] M. J. Blesa and C. Blum, "Finding edge-disjoint paths in networks: an ant colony optimization algorithm," *Journal of Mathematical Modelling and Algorithms*, vol. 6, no. 3, pp. 361–391, 2007.
- [18] Z. Yu, M. Ni, Z. Wang, and H. Huang, "Heuristic algorithm for K-disjoint QoS routing problem," in *Proceedings of the 4th International Joint Conference on Computational Sciences and Optimization (CSO '11)*, pp. 353–356, IEEE Computer Society Press, Kunming, China, April 2011.

- [19] M. F. Ni, S. Y. Gao, X. R. Wu, and W. Tong, "Integer linear programming method for finding link-disjoint paths for QoS routing," *Control and Decision*, vol. 27, no. 10, pp. 1597–1600, 2012.
- [20] J. W. Suurballe, "Disjoint paths in a network," *Networks*, vol. 4, pp. 125–145, 1974.
- [21] L. A. Wolsey, *Integer Programming*, John Wiley & Sons, New York, NY, USA, 1998.
- [22] B. M. Waxman, "Routing of multipoint connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617–1622, 1988.