Edge Disjoint Paths with Minimum Delay Subject to Reliability Constraint

Ruen Chze Loh, Sieteng Soh, and Mihai Lazarescu
Department of Computing
Curtin University of Technology, Perth, Australia
{ruen-chze.loh@postgrad.curtin, S.Soh@curtin, M.Lazarescu@curtin}.edu.au

***Abstract* – Recently, multipaths solutions have been proposed to improve the quality-of-service (QoS) in communication networks (CN). This paper describes a problem, λDP/RD, to obtain the λ-edge-disjoint-path-set such that its reliability is at least R and its delay is minimal, for λ≥1. λDP/RD is useful for applications that require non-compromised reliability while demanding minimum delay. In this paper we propose an approximate algorithm based on the Lagrange-relaxation to solve the problem. Our solution produces λDP that meets the reliability constraint R with delay≤(1+*k*)D_{min}, for *k*≥1, and D_{min} is the minimum path delay of any λDP in the CN. Simulations on forty randomly generated CNs show that our polynomial time algorithm produced λDP with delay and reliability comparable to those obtained using the exponential time brute-force approach.**

***Keywords* – approximate algorithm; Lagrange relaxation; multi-constrained edge disjoint paths; network reliability; network delay.**

I. INTRODUCTION

THE disjoint path set solutions [1-6] have been proposed to improve the end-to-end quality-of-service (QoS) of the communication networks (CN). Since the number of vertex disjoint paths in general is very limited, the edge disjoint path (DP) set that do not share edges is more commonly used [7]. References [1, 6, 8] propose algorithms to improve the reliability of CNs using DP. Reference [9] also shows that the lifetime of an end-to-end communication can be improved with a higher reliability DP.

Some CNs, such as those for time critical systems and multimedia applications, are subjected to multi-constrained QoS, *e.g.*, reliability, delay, cost and bandwidth. [8] considers cost and delay as the constraint parameters, [10, 11] consider cost and reliability and [4] uses reliability and delay. Note that the problem for generating a DP with two or more constraints has been shown NP-hard [12], and therefore heuristic and approximation algorithms [8, 13, 14] have been proposed to address the problem.

Orda and Sprintson [13] proposed four approximation algorithms to find two delay-constrained DPs with minimum total cost (2DP/DC). For a CN that contains two DPs with delay≤D and minimal cost OPT, their best algorithm, 2DP-4, always finds 2DP/DC with delay≤(1+1/*k*)D and cost≤*k*(1+γ)(1+ε)OPT, where *k* is a positive integer representing the approximate index, γ is a small value bounded by 2(log *k* + 1)/*k* and ε is an approximate factor. Applying Lagrange-relaxation, Peng and Shen proposed an algorithm (PSA for short) [8] that improves the performance of 2DP-4 to a delay≤(1+1/*k*)D with cost≤(1+*k*)OPT. They showed that PSA can be used to obtain λDP/DC, for λ>2. However, both algorithms in [13] and [8] have one significant limitation; they concentrate on finding only 2DP that satisfy the delay and cost constraints whereas other DP may also satisfy the user defined preconditions. In addition, no simulations were performed to benchmark the feasibility of the algorithms and find the optimal value of *k*. Loh, et al [2] have recently described a problem to obtain λDP/DR – the set of DPs with maximum reliability subject to delay constraint D, for λ≥1. The authors [2] used a similar Lagrange-relaxation method as in [8] to solve this problem.

Our contribution in this paper is twofold. First, we propose an important problem, which is to find λDP/RD – the set of DPs with minimum delay subject to reliability≥R, for λ≥1. The solution to this problem is obviously applicable to some important critical applications, *e.g.*, emergency response, rescue and military operations that demand certain levels of reliability assurances. Such applications require non-compromised reliability while demanding minimum system delay. Second, we present an approximation algorithm to solve this problem. Our solution generates DP with maximum delay no more than (1+*k*)D_{min}, where D_{min} is the minimum delay of a path set in the network.

This paper is organized as follows. Section II discusses the network model, notations and related works as the basis of our approach. Section III formulates the λDP/RD problem, while Section IV describes our approximate algorithm. Section V presents the simulation results and Section VI concludes our paper.

II. PRELIMINARIES

A. Network Model and Notations

A CN is modeled by an edge-weighted graph N=(V,E,d,p) where G=(V,E) is an undirected graph without multiple edges and self-loops. Each edge $e_j \in$ E is characterized by its delay $d_j \in$ d and its reliability $p_j \in$ p, where $d_j \geq 0$ is the time taken for traffic to be transferred from one end to the other of e_j and $0 \leq p_j \leq 1$ represents the probability that e_j is UP. An e_j is said to be UP (DOWN) if it is functioning (failed). All vertices in V are assumed to be always UP. The vertices and edges in N may represent computers and communication links, respectively.

An *(s,t)* simple path P_i between vertices *s* and *t* is formed by the set of UP edges such that no vertex is traversed more than once. Any proper subset of a simple path does not result in a path between the vertex pair. The pathset P_{st} is a set whose elements are *(s,t)* simple paths. Fig. 1 shows an

978-1-4244-4785-5/09/$25.00 ©2009 IEEE

example network for s=1 and t=11; the alphabets show the edge names and the values inside each bracket indicate the edge delay and edge reliability respectively. The P_{st} of Fig. 1 are: $P_1=\{a, b, c, d, e\}$, $P_2=\{a, b, c, r\}$, $P_3=\{a, b, c, h, k, m\}$, $P_4=\{f, g, k, m\}$, $P_5=\{f, g, h, d, e\}$, $P_6=\{f, g, h, r\}$, $P_7=\{n, p, q, m\}$, $P_8=\{n, p, q, k, h, d, e\}$, $P_9=\{n, p, q, k, h, r\}$.

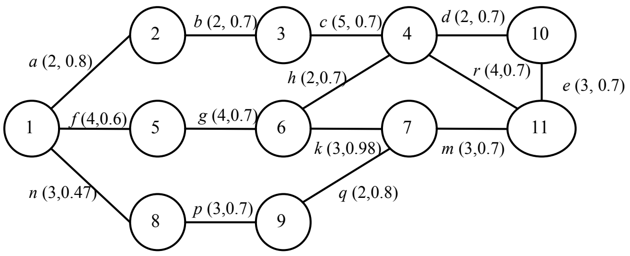

Fig. 1. A CN with 11 vertices and 14 edges

Paths P_i and P_j are edge disjoint paths (DP) if $e_\alpha \neq e_\beta$ for each $e_\alpha \in P_i$ and $e_\beta \in P_j$. In other words, there is no edge in P_i that is in P_j. Let $\lambda DP_\sigma \subseteq P_{st}$ be a DP, where $\lambda \geq 1$ is the total number of paths in the DP, and σ is any integer. For a given P_{st} there can be more than one λDP_σ, and none of them is a subset of any other. For example, the CN in Fig. 1 has six λDP_σ: $3DP_1=\{P_1,P_6,P_7\}$, $2DP_2=\{P_1,P_9\}$, $2DP_3=\{P_2,P_4\}$, $2DP_4=\{P_2,P_8\}$, $3DP_5=\{P_2,P_5,P_7\}$ and $1DP_6=\{P_3\}$.

The delay of path P_i, $\delta(P_i)$, is the sum of edge delays in P_i; *e.g.*, $\delta(P_1)$=2+2+5+2+3=14. The delay of λDP_σ, $\delta(\lambda DP_\sigma)$, is the maximum $\delta(P_i)$, for all $P_i \in \lambda DP_\sigma$, *i.e.*,

$$\delta(\lambda \mathrm{DP}_\sigma) = \max_{P_i \in \lambda \mathrm{DP}_\sigma} (\delta(P_i)) \tag{1}$$

For example, $\delta(3DP_1)=\max\{\delta(P_1), \delta(P_6), \delta(P_7)\} = \max\{14,14,11\}=14$.

The (s,t) reliability, $\rho(P_i)$, of a simple path P_i, is computed by multiplying p_j of each e_j that forms P_i, *i.e.*,

$$\rho(P_i) = \prod_{e_j \in P_i} p_j \tag{2}$$

For example, $\rho(P_1) = 0.8*0.7^4 = 0.19208$. The disjoint paths in an λDP_σ can be viewed as the components of a parallel system [15], and therefore its reliability, $\rho(\lambda DP_\sigma)$, can be computed as:

$$\rho(\lambda \mathrm{DP}_\sigma) = 1 - \prod_{P_i \in \lambda \mathrm{DP}_\sigma} (1-\rho(P_i)) \tag{3}$$

The multiplicative operations in (3) can be transformed into additive operations as:

$$\rho(\lambda \mathrm{DP}_\sigma) = 1 - \log^{-1}\Big(\sum_{P_i \in \lambda \mathrm{DP}_\sigma} \log(1-\rho(P_i))\Big), \tag{4}$$

where $\log^{-1}(X)$ is the antilog of X. Using Eq. (3), $\rho(3DP_1)$= $(1-(1-\rho(P_1)) * (1-\rho(P_6)) * (1-\rho(P_7))) = 1 - ((1-0.8*0.7^4) *(1-0.6*0.7^3)*(1-0.47*0.8*0.7^2)) = 0.4766$. Note, $\rho(3DP_1)$ can be equivalently computed using Eq. (4) as $1-\log^{-1}((-0.0926)+(-0.1001)+(-0.0884)) = 1-\log^{-1}(-0.2690) = 0.4766$.

B. Related Work

The Peng and Shen algorithm (PSA) [8] utilizes the Lagrange-relaxation to approximately generate 2DP/DC from a graph N(V,E,d,c), where c is the set of edge cost and d is the set of edge delay in G(V,E). For a given delay constraint D, PSA produces a 2DP/DC with delay less than $(1+1/k)$D and a total cost no more than $(1+k)$OPT, where OPT is the optimal cost among all 2DP that meet the delay constraint. The total cost of the 2DP/DC is defined as the sum of the cost of each path $P_i \in 2DP_\sigma$. Note that the algorithm can be extended to produce λDP/DC, for $\lambda>2$ [8]. Here, PSA aims to generate the minimum cost λDP/DC with the largest λ that meets the delay constraint. In contrast, a λDP/RD is not necessarily a DP with the largest λ. Therefore, PSA is not suitable for generating λDP/RD.

The algorithm in [2] uses a similar technique to PSA to produce a λDP/DR for a given delay constraint D and a graph N(V,E,d,p) where p is the set of edge reliability and d is the set of edge delay in G(V,E). The algorithm [2] is guaranteed to produce a λDP with, respectively, delay and reliability bounded by Eq. (5) and Eq. (6):

$$\delta(\lambda \mathrm{DP}) \leq (1+1/k)\mathrm{D} \tag{5}$$

$$|\log(1-\rho(\lambda \mathrm{DP}))| \leq (1+k)|\log(1-\mathrm{OPT})|, \tag{6}$$

where OPT is the maximum reliability among all possible λDP that satisfy the delay requirement, D. Like λDP/RD, the generated λDP/DR is not necessarily a DP with the maximum λ. One may use the algorithm in [2] to generate λDP/RD if one substitutes D in Eq. (5), and OPT in Eq. (6) with the maximum path delay D_{max} of the network and the reliability requirement R, respectively. Notice that the generated λDP/RD would have a reliability value at most $(1+k)*|\log(1-R)|$, which may not satisfy the reliability requirement, R. Therefore, a more effective algorithm, presented in this paper, is needed to solve the λDP/RD problem.

III. λDP/RD PROBLEM FORMULATION

For a given reliability constraint R, let λDP_σ^r be λDP_σ that has reliability at least R, *i.e.*, $\rho(\lambda DP_\sigma^r) \geq R$. Let λDP_τ^{rd} be λDP_σ^r with minimum delay, *i.e.*,

$$\delta(\lambda \mathrm{DP}_\tau^{rd}) = \min(\{\delta(\lambda \mathrm{DP}_\sigma^r)\}) \tag{7}$$

Note that there can be more than one λDP_σ^{rd}. The edge-disjoint-path-set with minimum delay and reliability≥R problem (λDP/RD) is to find among all λDP_τ^{rd} a DP with the highest reliability (called λDP^{BM}), *i.e.*,

$$\rho(\lambda \mathrm{DP}^{\mathrm{BM}}) = \max(\{\rho(\lambda \mathrm{DP}_\tau^{rd})\}) \tag{8}$$

To illustrate the λDP/RD problem, consider R=0.4 and the λDP_σs of Fig. 1. Using Eq. (4), we obtained, $\rho(3DP_1)=0.4766$, $\rho(2DP_2)=0.2942$, $\rho(2DP_3)=0.4836$, $\rho(2DP_4)=0.3386$, $\rho(3DP_5)=0.4934$ and $\rho(1DP_6)=0.1882$. Among the λDP_σs there are three λDP_σ^r: $3DP_1^r$, $2DP_3^r$ and $3DP_5^r$. Using Eq. (1), we obtain, $\delta(3DP_1^r)=14$, $\delta(2DP_3^r)=14$ and $\delta(3DP_5^r)=15$; thus using Eq. (7), there are two λDP_σ^{rd}: $3DP_1^r$, $2DP_3^r$. Eq. (8) obtains $\max\{\rho(3DP_1^r)= 0.4766, \rho(2DP_3^r)=0.4836\}= 0.4836$, and thus $\lambda DP^{BM}=2DP_3$. Notice that the optimal $2DP_3$ is not a λDP with the largest λ.

One may obtain λDP^{BM} by exhaustively generating all possible λDP_σ^r path sets and using Eq. (7) to select the set of λDP_σ^{rd}, and using Eq. (8) select the most reliable one. Note that N(V,E,d,p), in general, contains an exponential number (in terms of |E|) of (s,t) paths ($|P_{st}|$), and therefore this brute force (BF) approach may generate an exponential number (in terms of $|P_{st}|$) of λDP_σ^r, and thus this solution has double exponential (in terms of |E|) time complexity.

IV. THE λDP/RD APPROXIMATE ALGORITHM

A. Lagrange-relaxation Approach for λDP/RD

We propose to use Lagrange-relaxation technique to solve the λDP/RD problem. As in [8] and [2], we combine the reliability p_i and delay d_i of each edge e_i into a weight $w_i = d_i + \eta * \log(p_i)$, rather than considering them separately. In other words, we transform the two constraint-weighted network N(V,E,p,d) into the one-constraint-weighted network N(V,E,w), for $w_i \in$ w. The value of η should be set properly to minimize the delay δ(λDP), and to maximize reliability ρ(λDP)≥R. Note that for λDP/DC, it had been shown that setting η (called α in [8]) to k*OPT/D produced good results. Since λDP/RD is similar to λDP/DC, we set $\eta_k = k * D_{min}/\log(R)$, *i.e.,* replacing OPT with D_{min} and the delay constraint D with the log of the reliability constraint R, where D_{min} is the minimum delay of all possible DPs in the network. One may obtain D_{min} by using DPSP [9] or iDPSP [6] from N(V,E,p,d) assuming perfect edges, *i.e.*, $p_i=1$ for all $p_i \in$ p. The following lemma states that when the value of |η| increases, ρ(λDP) decreases.

Let $W(\lambda DP_\sigma, \eta_k) = \delta(\lambda DP_\sigma) + \eta_k * \log(\rho(\lambda DP_\sigma))$ denote the weight of λDP_σ generated when $\eta=\eta_k$. In graph G=(V,E), let λDP^α be the DP with the minimum total weight when we set $w_i=d_i+ \eta_\alpha * \log(p_i)$ and λDP^β be the DP with the minimum total weight when we set $w_i=d_i+ \eta_\beta * \log(p_i)$, *i.e.*, for k=α and k=β, respectively.

Lemma 1. $\log(\rho(\lambda DP^\beta)) \le \log(\rho(\lambda DP^\alpha))$, for $0< |\eta_\alpha| \le |\eta_\beta|$.
Proof: Since λDP^α is the DP with the minimum total weight when we set $w_i=d_i+ \eta_\alpha * \log(p_i)$, we have,

$$\begin{aligned} W(\lambda DP^\alpha, \eta_\alpha) &= \delta(\lambda DP^\alpha) + \eta_\alpha * \log(\rho(\lambda DP^\alpha)) \\ &\le \delta(\lambda DP^\beta) + \eta_\alpha * \log(\rho(\lambda DP^\beta)) \end{aligned} \quad (9)$$

Since λDP^β is the DP with the minimum total weight when we set $w_i=d_i+ \eta_\beta * \log(p_i)$, we have,

$$\begin{aligned} W(\lambda DP^\beta, \eta_\beta) &= \delta(\lambda DP^\beta) + \eta_\beta * \log(\rho(\lambda DP^\beta)) \\ &\le \delta(\lambda DP^\alpha) + \eta_\beta * \log(\rho(\lambda DP^\alpha)) \end{aligned} \quad (10)$$

Adding Eq. (9) and Eq. (10), we obtain:
$\delta(\lambda DP^\alpha) + \eta_\alpha * \log(\rho(\lambda DP^\alpha)) + \delta(\lambda DP^\beta) + \eta_\beta * \log(\rho(\lambda DP^\beta)) \le \delta(\lambda DP^\beta) + \eta_\alpha * \log(\rho(\lambda DP^\beta)) + \delta(\lambda DP^\alpha) + \eta_\beta * \log(\rho(\lambda DP^\alpha))$, and thus $\log(\rho(\lambda DP^\beta)) \le \log(\rho(\lambda DP^\alpha))$.

Theorem 1. $\rho(\lambda DP^\beta) \le \rho(\lambda DP^\alpha)$, for $1 \le \alpha \le \beta$.
Proof: By definition, $\eta_\beta = \beta * D_{min}/\log(R)$ and $\eta_\alpha = \alpha * D_{min}/\log(R)$, and therefore, $|\eta_\alpha| \le |\eta_\beta|$. Since $0 \le \rho(\lambda DP^\beta) \le 1$ and $0 \le \rho(\lambda DP^\alpha) \le 1$, by Lemma 1 $\rho(\lambda DP^\beta) \le \rho(\lambda DP^\alpha)$ for $|\eta_\alpha| \le |\eta_\beta|$, and thus for $1 \le \alpha \le \beta$.

Theorem 2. Consider a λDP^k that is obtained when $\eta_k = k * D_{min}/\log(R)$, $D_{min} \le W(\lambda DP^k, \eta_k) \le (1+2k) D_{min}$ iff $D_{min} \le \delta(\lambda DP^k) \le (1+k) D_{min}$.
Proof: To prove that $D_{min} \le W(\lambda DP^k, \eta_k) \le (1+2k) D_{min}$ if $D_{min} \le \delta(\lambda DP^k) \le (1+k) D_{min}$, consider Fig. 2, which describes the feasible solution space for λDP/RD. The x-axis represents the logarithm of all possible reliability values ρ(λDP) when δ(λDP)=0, *i.e.*, when each edge has zero delay. On the other hand, the y-axis represents all possible delay values δ(λDP) assuming perfect edges, *i.e.*, ρ(λDP)=1 or log(ρ(λDP)=0. Since the minimum delay (lower bound) of all possible DPs in the network is D_{min}, the delay values of any point in line AB is D_{min}. A feasible solution must have a reliability of at least R, and therefore the value of each point in line BD is log(R). Since the algorithm aims to obtain a λDP with delay at most $(1+k)D_{min}$, line CD represents the upper bound of δ(λDP). Therefore, Fig. 2 shows $D_{min} \le \delta(\lambda DP^k) \le (1+k) D_{min}$ and $\log(\rho(\lambda DP^k)) \ge \log(R)$.

The upper and lower bounds of the weight of λDP^k, $W(\lambda DP^k, \eta_k)$, can be obtained as follows. Substituting $\log(\rho(\lambda DP^k)=0$ into $W(\lambda DP^k, \eta_k) = \delta(\lambda DP^k) + \eta_k * \log(\rho(\lambda DP^k))$, the weights of a λDP^k at points A and C are D_{min} and $(1+k)$ D_{min}, respectively. Note that $\delta(\lambda DP^k) = D_{min}$ at point A, and $\delta(\lambda DP^k) = (1+k) D_{min}$ at point C. Similarly, substituting $\rho(\lambda DP^k)=R$ and $\eta_k = k * D_{min}/\log(R)$ into $W(\lambda DP^k, \eta_k) = \delta(\lambda DP^k) + \eta_k * \log(\rho(\lambda DP^k))$, we obtain $W(\lambda DP^k, \eta_k) = \delta(\lambda DP^k) + k * D_{min}$, and since $\delta(\lambda DP^k) = D_{min}$ at B and $\delta(\lambda DP^k) = (1+k) D_{min}$ at D, the weights at B and D are $(1+k) D_{min}$ and $(1+2k) D_{min}$, respectively. Therefore, $D_{min} \le W(\lambda DP^k, \eta_k) \le (1+2k) D_{min}$ if $D_{min} \le \delta(\lambda DP^k) \le (1+k) D_{min}$.

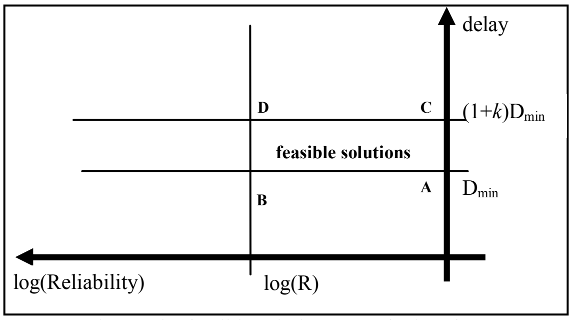

Fig. 2. The feasible solution space for λDP/RD

To prove that $D_{min} \le \delta(\lambda DP^k) \le (1+k) D_{min}$ if $D_{min} \le W(\lambda DP^k, \eta_k) \le (1+2k) D_{min}$, one may notice that any λDP that is generated when its weight is at a point (x,y) in the feasible solution, shown in Fig. 2, will give a delay of y. The maximum weight $W(\lambda DP^k, \eta_k) = (1+2k) D_{min}$ is obtained when the point is at D, *i.e.*, when $\log(\rho(\lambda DP^k)) = \log(R)$. Thus, at that point, $W(\lambda DP^k, \eta_k) = \delta(\lambda DP^k) + k * D_{min} * \log(R)/\log(R) \le (1+2k) D_{min}$ or $\delta(\lambda DP^k) \le (1+2k) D_{min} - k * D_{min} = (1+k) D_{min}$. Therefore, $\delta(\lambda DP^k) \le (1+k) D_{min}$. Similarly, the minimum weight $W(\lambda DP^k, \eta_k) = D_{min}$ is generated at point (0, D_{min}), which is at A, where $\log(\rho(\lambda DP^k))=0$. Thus, at this point, $W(\lambda DP^k, \eta_k) = \delta(\lambda DP^k) + k * D_{min} * 0/\log(R) \ge D_{min}$ or $\delta(\lambda DP^k) \ge D_{min}$.

B. Algorithm

Fig. 3 shows the λDP/RD algorithm. Let λDP^{Rmax} be the λDP with the maximum reliability, and λDP^{Dmin} be the λDP with the minimum total delay in N=(V,E,p,d). One may use the DPSP [9] or the iDPSP [6] algorithm to compute both λDP^{Rmax} and λDP^{Dmin} from N(V,E,d,p) by setting each $d_i \in$ d to 0 and $p_i \in$ p to 1, respectively. If the reliability of λDP^{Rmax} is less than the reliability constraint R, there is no feasible solution for the network, and therefore the algorithm terminates. Otherwise, the algorithm aims to obtain the optimal λDP^{BM} that has best metrics BM= ($delay_{BM}$, $reliability_{BM}$); λDP^{BM} and BM are initialized to Ø set and (∞, 0), respectively. Each iteration in the loop uses iDPSP() to obtain a new λDP, λDP^{NM}, that has the minimum weight from N(V,E,w), where the edge weights, $w_i \in$ w are calculated for each increasing k. Note that λDP^{NM} has new metrics NM= ($delay_{NM}$, $reliability_{NM}$). Following Theorem 1, if $\rho(\lambda DP^{NM})<R$ is generated when k=α, then all other $\rho(\lambda DP^{NM})$ generated when k=β will always be less than R

for $\beta>\alpha$. Thus, the algorithm stops the iteration and return λDP^{BM} as the approximately best λDP. Otherwise, from Theorem 2, if $D_{min} \leq W(\lambda DP^{k}, \eta_k) \leq (1+2k)D_{min}$, we use function optimal() to select the more optimal λDP between λDP^{BM} and λDP^{NM}, and continue the iteration for larger k. From Eqs. (7) and (8), λDP^{BM} is more optimal than λDP^{NM} if (i) $delay_{BM} < delay_{NM}$, or (ii) $delay_{BM} = delay_{NM}$ and $reliability_{BM} > reliability_{NM}$.

```
Input   : N : the directed graph N=(V,E,p,d);
          R : the reliability constraint;
Output : approximated λDP^BM;
1.  if ρ(λDP^Rmax)<R then return "No such solution"
        exit;
2.  D_min ← δ(λDP^Dmin);
3.  λDP^BM ← Ø; BM ← (∞, 0); k←0; done ←false;
4.  Repeat
5.    k++;
6.    η_k ← k*D_min/log(R);
7.    for each e_i∈E do w_i ← d_i + η_k*log(p_i) endfor;
8.    λDP^NM ← iDPSP(N(V,E,w));
9.    if ρ(λDP^NM)<R then done←true;
10.   else if (D_min≤W(λDP^k, η_k)≤(1+2k)D_min) then
        λDP^BM ← optimal(λDP^BM, λDP^NM);
11. until done;
12. output λDP^BM;
```

Fig. 3. The approximate algorithm for $\lambda DP/RD$

The time complexity of the algorithm can be calculated as follows. Either DPSP [9] or iDPSP [6] have the time complexity of $O(|V||E|^2)$, and line 7 and optimal() has O(E) and O(1) time complexity, respectively. Therefore, the time complexity of the algorithm depends on the total number the loop is repeated, *i.e.*, on the value of k. Thus, time complexity is $O(k*|V||E|^2)$.

As an illustrating example, consider the network in Fig. 1 with R=0.4. Using iDPSP [6], we obtained $\rho(\lambda DP^{Rmax})=0.4934$ and $\delta(\lambda DP^{Dmin})=11$. Table I shows the delay, reliability and weight of the λDP obtained by the algorithm when k was incremented from 1 to 3.

TABLE I
RESULTS OBTAINED BY THE ALGORITHM FOR CN IN FIG.1

k	Delay	Reliability	$W(\lambda DP^k,\eta_k)$
1	15	0.493437	15.7
2	14	0.483619	27.5
3	18	0.338637	39.3

When $k=1$, $W(\lambda DP^k,\eta_k)=15.7$ which satisfies $11 \leq W(\lambda DP^1,\eta_1) \leq 33$ thus, BM=(15, 0.493437). When k was increased to 2, $W(\lambda DP^2,\eta_2)=27.5$ which still satisfies $11 \leq W(\lambda DP^2,\eta_2) \leq 55$ and the $delay_{NM}$ obtained was 14 which was lesser than $delay_{BM}$, therefore, line 10 replaces λDP^{BM} with λDP^{NM} with BM=(14, 0.483619). For $k=3$, the $reliability_{NM}<R$ so the algorithm output the $\lambda DP^{BM}=\{P_2, P_4\}$ obtained when $k=2$ with BM=(14, 0.483619).

V. SIMULATION AND DISCUSSION

We used BRITE [16] with the RTWaxman configuration to generate a random topology that contains 50 vertices, 72 edges and 1124 *s-t* paths. From the topology, we constructed 40 random networks, 10 each for the following four different network groups, CN1, CN2, CN3, and CN4. The edges of each network in CN1 and CN3 are randomly assigned with edge reliabilities ranging from 0.1 to 0.5 and 0.1 to 0.9 respectively with incremental value of 0.1. For CN2, we used edge reliability values ranging from 0.5 to 0.75 with incremental value of 0.05, while for CN4 the random reliability values are ranging from 0.9 to 0.99 with incremental values of 0.01. Further, we also assigned a random delay value ranging from 3 to 7 units to each edge of the 40 networks, and used reliability constraints R of 0.1, 0.5, 0.7 and 0.95 for CN1, CN2, CN3, and CN4, respectively. Note that we generated the 40 random networks such that each of them satisfies the constraint R. In all our simulations, the value of k starts from 1. We used the C implementation of our $\lambda DP/RD$ algorithm to obtain the λDP^{BM} for the 40 CNs. All simulations were run on a 2x Intel Pentium 2-2.6Ghz with 1.8GB of RAM, running Fedora Core 6.

A. The Effects of k on Reliability

We ran $\lambda DP/RD$ algorithm on all the 40 networks with increasing k=1, 2, …; this simulation is used to show the correctness of Theorems 1 and 2. The results for each network are consistent with the theorems; Table II shows the results when the algorithm was executed on one of the CN3s that had $D_{min}=23$.

TABLE II
RESULTS OBTAINED FROM $\lambda DP/RD$ ALGORITHM WITH ONE CN3

k	Delay	Reliability
1 to 12	40	0.825406
13 to 16	40	0.816747
17 to 22	40	0.804262
23 to 32	31	0.786859
33 to 34	31	0.75805
35 to 41	31	0.71994
42	31	0.667086

Consistent with Theorem 1, when $k=1$, the algorithm produced the λDP with the highest reliability. It remained as the λDP^{BM} until $k=23$ at which λDP^{NM} had a lower delay; this λDP^{NM} became the new λDP^{BM} since its reliability≥R=0.7. For $k=42$, λDP^{NM} has reliability 0.667086<R. Thus, the algorithm outputs as the solution with the smallest delay the λDP^{BM} generated when $k=23$. Note that for this CN, the delay $\delta(\lambda DP^{NM})$ decreased as k increased; however, this may not always be true. Therefore, we need to increase k sequentially to evaluate every λDP^{NM} using Eqs. (7) and (8) until a value of k produces a λDP^{NM} with $\rho(\lambda DP^{NM})<R$ before deciding on the λDP^{BM}. To validate Theorem 2, we checked that each feasible solution has $D_{min} \leq W(\lambda DP^{k}, \eta_k) \leq (1+2k)D_{min}$.

B. The Accuracy of the Algorithm

We used our algorithm to generate λDP^{BM} and computed its delay (D_{ours}) and reliability (R_{ours}) as shown in Table III. To evaluate the optimality of our algorithm, we compared D_{ours} and R_{ours} with D_{BF} and R_{BF}, respectively, which were generated using an exponential time brute force (BF) algorithm (described in Section III). Note that D_{ours} (R_{ours}) always satisfies the delay bound $D_{bound}=(1+k)D_{min}$ (reliability constraint R) for each CN.

We consider four possible comparison results: (i) $D_{ours}=D_{BF}$ and $R_{ours}=R_{BF}$, (ii) $D_{ours}=D_{BF}$ and $R_{ours}< R_{BF}$, (iii) $D_{ours}>D_{BF}$ and $R_{ours}\geq R_{BF}$, and (iv) $D_{ours}>D_{BF}$ and $R_{ours}< R_{BF}$. The columns "%D_{ours}" and "%R_{ours}" in Table III show the percentage differences between D_{ours} and R_{ours} against D_{BF}

and R_{BF} respectively. As shown in column 8 of Table III our algorithm is optimal 25% of the time (marked with (i)) both in terms of delay and reliability, and 47.5% of the time it generates the same delay as BF but with, on average, 2.6% less reliable (a negative value marked with (ii)). Note that 5% of the time our algorithm produced results in category (iii) (marked with (iii)), where there is a tradeoff of a higher delay (average 9.59%) for higher reliability (average 1.95%). Even though 22.5% of the time our approach produced results in category (iv) (marked with (iv)), D_{ours} is at most 9.76% higher than D_{BF} and R_{ours} is at most 2.38% lower than R_{BF}. Our λDP/RD algorithm and the BF approach took on average 1.2 seconds and 191 seconds respectively to generate the λDP^{BM} of each CN.

TABLE III
COMPARISON BETWEEN λDP/RD AND BF RESULTS

			Delay			Reliability		
CN	k_{opt}	k_{max}	D_{BF}	$\%D_{ours}$	D_{bound}	R_{BF}	$\%R_{ours}$	$\%R'_{ours}$
CN1, R=0.1								
1	49	66	40	0.00	1550	0.1038	-3.62 (ii)	-3.62
2	1	59	39	0.00	62	0.1025	0.00 (i)	0.00
3	23	56	39	0.00	792	0.1015	0.00 (i)	0.00
4	1	8	40	0.00	60	0.1035	-1.28 (ii)	-1.28
5	9	15	41	0.00	155	0.1002	0.00 (i)	0.00
6	3	47	45	0.00	124	0.1001	0.00 (i)	0.00
7	17	37	40	2.50	512	0.1083	-0.51 (iv)	-0.51
8	12	59	40	2.50	403	0.1096	-1.95 (iv)	-1.95
9	10	77	39	2.56	341	0.1130	-0.22 (iv)	-0.22
10	25	53	41	9.76	806	0.1052	-0.20 (iv)	-0.21
CN2, R=0.5								
1	12	35	32	12.50	286	0.5082	2.67 (iii)	-1.22
2	2	36	31	0.00	66	0.5111	0.00 (i)	0.00
3	1	25	32	0.00	42	0.5217	0.00 (i)	0.00
4	35	37	32	3.13	805	0.5330	-4.10 (iv)	-4.10
5	27	47	32	0.00	532	0.5328	-1.80 (ii)	-1.80
6	1	42	31	0.00	46	0.5629	0.00 (i)	0.00
7	37	49	30	6.67	816	0.5253	1.23 (iii)	-0.66
8	51	63	32	0.00	1040	0.5423	-5.77 (ii)	-5.77
9	1	36	32	0.00	44	0.5270	0.00 (i)	0.00
10	3	60	38	0.00	80	0.5251	0.00 (i)	0.00
CN3, R=0.7								
1	23	42	33	0.00	552	0.8078	-2.60 (ii)	-2.60
2	47	52	31	0.00	912	0.7398	-1.84 (ii)	-1.84
3	17	27	31	3.23	374	0.7568	-2.28 (iv)	-2.28
4	1	17	31	0.00	40	0.8277	-3.97 (ii)	-3.97
5	36	55	34	2.94	740	0.7710	-2.38 (iv)	-2.38
6	1	17	32	0.00	34	0.7440	-1.17 (ii)	-1.17
7	19	65	30	0.00	420	0.8277	-5.38 (ii)	-5.38
8	22	43	29	0.00	460	0.7477	0.00 (i)	0.00
9	5	37	33	0.00	54	0.7507	-0.10 (ii)	0.10
10	1	19	34	0.00	50	0.7472	-3.67 (ii)	-3.67
CN4, R=0.95								
1	42	43	32	0.00	1032	0.9999	-3.89 (ii)	-3.89
2	44	68	29	0.00	1035	0.9999	-1.87 (ii)	-1.87
3	75	84	28	3.57	1520	0.9999	-1.12 (iv)	-1.12
4	55	59	29	0.00	1092	0.9999	-1.20 (ii)	-1.20
5	31	57	28	0.00	640	0.9997	-0.57 (ii)	-0.57
6	32	48	32	0.00	759	0.9999	-0.85 (ii)	-0.85
7	21	46	29	3.45	462	0.9999	-0.01 (iv)	-0.01
8	51	65	32	0.00	987	0.9999	-0.78 (ii)	-0.78
9	25	36	29	0.00	520	0.9999	-0.30 (ii)	-0.30
10	19	57	29	0.00	420	0.9999	-0.03 (ii)	-0.03

To see the optimality of our algorithm in terms of the reliability of the λDP^{BM}, we compared its results with those generated by the BF approach while setting D_{BF}=D_{ours}. The column $\%R'_{ours}$ in Table III shows the percentage differences between the reliability obtained by our algorithm against that obtained by BF. As indicated earlier (shown in column 8 in Table III), 25% of the time, our algorithm generates optimal reliability (0% in the column). Even though 75% of the time our approach does not produce the λDP^{BM} with optimal reliability, R_{ours} is at most only -5.77% off the optimal result, while using only 0.79% of the CPU time required by the optimal BF approach.

Table III shows k_{max} and k_{opt} that denote the value of the smallest k for which the algorithm results in $\rho(\lambda DP^{NM})<R$ (thus it outputs λDP^{BM}) and the value of k when $\rho(\lambda DP^{BM}) = \max(\{\rho(\lambda DP_{\sigma}^{rd})\})$, respectively. As shown in column 3, $k_{max}<85$, *i.e.*, our algorithm in Fig. 3 iterates the repeat loop at most 84 times for generating each λDP^{BM}.

VI. CONCLUSION

We have addressed an important λDP/RD problem to generate a λDP with minimum delay while meeting a reliability constraint, R. An approximate Lagrange-relaxation algorithm has been presented to solve the problem. Our simulations on forty randomly generated CNs with random edge reliabilities and delays show that our polynomial time method is able to generate λDP/RD with delay and reliability values comparable to those generated using the optimal but time-expensive brute force approach.

We are investigating a method to bound the value of k to further reduce the complexity of our approach. We also plan to use some alternative heuristic algorithms for the problem.

REFERENCES

[1] R. C. Loh, S. Soh, M. Lazarescu, and S. Rai, "A Greedy Technique for Finding the Most Reliable Edge-disjoint-path-set in a Network," *Proc. IEEE PRDC, Taiwan,* pp. 216-223, 2008.

[2] R. C. Loh, S. Soh, and M. Lazarescu, "An Approach to Find Maximal Disjoint Paths with Reliability and Delay Constraints," *Proc. IEEE AINA, Bradford, UK,* pp. 959-964, 2009.

[3] Y. Guo, F. Kuipers, and P. Van Mieghem, "Link-disjoint paths for reliable QoS routing," *Int'l J. Comm. Sys.,* vol. 16, pp. 779-798, 2003.

[4] X. Huang and Y. Fang, "Multiconstrained QoS multipath routing in wireless sensor networks," *Wireless Networks,* vol. 14, pp. 465-478, 2008.

[5] M. Razzaque, A. Alam, M. M. Mamun-Or-Rashid, M. Hong, and C. Seon, "Multi-Constrained QoS Geographic Routing for Heterogeneous Traffic in Sensor Networks," *5th IEEE CCNC,* pp. 157-162, 2008.

[6] R. C. Loh, S. Soh, and M. Lazarescu, "Finding the most reliable edge-disjoint-path-set in a communication network," *Proc. PEECS, Perth, Australia,* pp. 121-126, 2007.

[7] A. Agarwal and B. Jain, "Routing reliability analysis of segmented backup paths in mobile ad hoc networks," *ICPWC,* pp. 52-56, 2005.

[8] C. Peng and H. Shen, "A New Approximation Algorithm for Computing 2-Restricted Disjoint Paths," *IEICE Trans. Info. and Sys.,* vol. 90, pp. 465-472, 2007.

[9] P. Papadimitratos, Z. J. Haas, and E. G. Sirer, "Path set selection in mobile ad hoc networks," *Proc. ACM MobiHoc, Lausanne, Switzerland,* pp. 1-11, 2002.

[10] A. K. Andreas, J. C. Smith, and S. Kucukyavuz, "Branch-and-Price-and-Cut Algorithms for Solving the Reliable h-Paths Problem," October 9, 2007.

[11] A. K. Andreas and J. C. Smith, "Exact Algorithms for Robust k-Path Routing Problems," *Procs. of GO,* pp. 1-6, 2005.

[12] V. Guruswami, S. Khanna, R. Rajaraman, B. Shepherd, and M. Yannakakis, "Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems," *J. Comp. and Sys. Sci.,* vol. 67, pp. 473-496, 2003.

[13] A. Orda and A. Sprintson, "Efficient algorithms for computing disjoint QoS paths," *Proc. IEEE INFOCOM, Hong Kong,* vol. 1, pp. 727–738, 2004.

[14] G. Xue, W. Zhang, J. Tang, and K. Thulasiraman, "Polynomial time approximation algorithms for multi-constrained QoS routing," *IEEE Trans. Networking,* vol. 16, pp. 656-669, 2008.

[15] S. Soh, S.Rai, "Telecommunication Network Reliability," *Encyclopedia of Life Support Systems, UNESCO/Eolss Publishers, Oxford, U.K.,* 2007.

[16] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: Universal Topology Generation from a User's Perspective," *Proc. IEEE MASCOTS, Ohio, USA,* pp. 346-353, 2001.