Finding Delay-constrained Cost-optimization Path (DCCOP)

Xiang DUAN, Nonmember, Zhisheng NIU, Member, and Junli ZHENG, Nonmember

Summary
In this paper, we propose a heuristic algorithm to the problem of finding delay-constrained cost-optimization path in communication networks, which is a NP-complete problem. In the algorithm we introduce an adjustable parameter $S$, which is the iteration times of the delay-shifting process, to control the tradeoff between the routing success ratio and the time-complexity of the algorithm. Simulation results show that, in a relatively low time-complexity, our algorithm can find the delay-constrained optimal-cost path with a high probability.

Key words:
DCCOP, NP-complete, heuristic

1. Introduction

Today’s communication network carries diverse multimedia sessions concurrently. Many of these sessions, such as voice and video, may have stringent end-to-end delay requirement. On the other hand, network service providers have great interests on how to lower the cost of providing service to users. Thus, finding the path with optimal operating cost and end-to-end delay constraint is desired, because with such a path, user’s quality of service (QoS) requirement on delay can be satisfied. Meanwhile, operating cost can be minimized, which is attractive to network service providers. Finding such a path is called a delay-constrained cost-optimization path (DCCOP) problem. DCCOP problem can be formalized as an extension of standard multi-constrained path (MCP) problem [1]. Unfortunately, the latter has been proved to be NP-complete [2].

Several heuristic algorithms have been proposed for the NP-complete MCP problem. Ref. [1] proposed a heuristic algorithm with polynomial-time complexity by simplifying one of the path constraints into discrete integers. Orda [3] summarized several solutions with $\varepsilon$’s scale of rate-quantizing (RQ-$\varepsilon$) in networks with WFQ-like scheduling schemes. In [4], the author proposed a heuristic algorithm for MCP problem used in hierarchical networks by mapping link metrics into a limited number of sets. These algorithms have one common property: for any one of these routing algorithms, it selects the pseudo-optimal path, i.e., the objective deviation between the selected path and the optimal path is bounded by a certain parameter of the algorithm. As the deviation bound becoming smaller, algorithm’s time complexity increases.

However, in many cases, we only care whether the selected path is the optimal one. We call the case routing success that the selected path is the optimal path (not the pseudo-optimal one), and routing success ratio, the proportion of success routing times to the total path-selection times. If most of the selected paths are optimal, i.e., the routing success ratio is close to 1, the paths that are non-optimal, though may vary greatly from the optimal one in the optimization objective, will not greatly impact the network’s performance. Based on this goal, we propose a simple heuristic algorithm to the DCCOP problem. This algorithm can control the tradeoff between the routing success ratio and complexity, using the parameter of delay-shifting times. Simulation results show that, with relatively low complexity, the routing success ratio is high.

This paper is organized as follow: Section 2 illustrates our heuristic algorithm in detail; Section 3 provides an experimental environment, and some numerical results, with their analysis also given in this section. Finally we summarize our work in the last section.

2. Heuristic Algorithm Description

Let $D(p)$ be the end-to-end delay of path $p$, $C(p)$ be the end-to-end cost, and $D_{\text{const}}$ be the user’s delay constraint. $p$ is defined as a feasible path if

$$D(p) \leq D_{\text{const}},$$

(1)

A path $p_o$ is called optimal if

$$C(p_o) = \min_{p \in F} \{C(p) \mid p \in F\},$$

(2)

where $F$ is the feasible path set. The number of paths in $F$ is always limited, so we know that if $F$ is not empty, the optimal path must exist. We call the optimal path the DCCOP problem’s solution.

In order to find the optimal path, we should first find feasible paths. Let $p_{\text{min-d}}$ be the minimal-delay path, and $p_{\text{min-c}}$ be the minimal-cost path. Here we have two lemmas:
Lemma 1: The DCCOP have a solution if and only if $p_{\text{min}-d}$ is feasible.

[Proof]: If DCCOP has a solution, it means an optimal path $p_o$ exists. Therefore, the following inequality holds:

$$D(p_{\text{min}-d}) \leq D(p_o).$$

Meanwhile, the optimal path is a feasible one, so

$$D(p_o) \leq D_{\text{cons}}.$$  

From Eqs. (3) and (4) we have

$$D(p_{\text{min}-d}) \leq D_{\text{cons}},$$

which means $p_{\text{min}-d}$ is a feasible path.

On the other hand, if the $p_{\text{min}-d}$ is a feasible path, the feasible path set $F$ is not empty. So, from Eq. (2) we know there must be an optimal path.

Lemma 2: If the $p_{\text{min}-c}$ is feasible, it is the solution of the DCCOP.

[Proof]: for any path $p$, the following inequality holds:

$$D(p_{\text{min}-c}) \leq D(p).$$

Because $p_{\text{min}-c}$ is a feasible path, the feasible path set $F$ is not empty. Combining (2) with (6) we get

$$C(p_{\text{min}-c}) = \min_p \{C(p) | p \in F\}.$$  

Hence, $p_{\text{min}-c}$ is the optimal path.

In DCCOP problem, with Lemma 1, we can exclude the ones without solutions. Using Lemma 2, we can easily find the solution for the problems whose solution is the minimal-cost path.

We adopt a method called delay-cost (D-C) plane, an expansion of delay-bandwidth plane used in [5], to describe our problem. Fig. 1 illustrates a D-C plane. In the Fig. 1(a), a point $p(d,c)$ denotes a path $p$ with $D(p) = d$ and $C(p) = c$. In DCCOP problem, it is clear we prefer less-delay and less-cost path. So we have the following definition of better point:

**Def.** A point $p(d,c)$ is better than a point $p'(d',c')$ if $D(p) \leq D(p')$ and $C(p) \leq C(p')$, and at least one of the two inequalities are stringent.

In the D-C plane, all the points denoting feasible paths are in the feasible domain, which is a half-infinite band area shown in Fig. 1. The optimal path is denoted by the optimal point ($p_o$ in Fig. 1(b)), and is the one with the lowest cost in the area. A point cannot be optimal if there is a point that is better than it. This is obvious because, if there is a point that is better than the optimal point, both its cost and delay are lower than the optimal point, so it is a point in the feasible area with a cost lower than the optimal
path, we increase the delay value of every link in \( p_{\text{min-d}} \) by a predetermined increment \( D_i \), so that the delay of the path changes to

\[
D(p_{\text{min-d}}) = D(p_{\text{min-d}}) + |p_{\text{min-d}}| \cdot D_i.
\]

Consequently, the point that denotes the minimal-delay path shifts right horizontally (Fig. 1(c), filled points move to shadowed points). Then we again look for the “minimal-delay” path with the changed delay metrics. In most cases, the delay of another feasible path \( q \) will increase if it has intersected links with the last-selected path \( p_{\text{min-d}} \); however, the increment of \( q \)'s delay depends on the number of links that \( q \) intersects with \( p_{\text{min-d}} \), and the following condition holds:

\[
t_{p_{\text{min-d}}} < |p_{\text{min-d}}| \cdot D_i \quad \text{for all } i \in p_{\text{min-d}},
\]

where \( t_{p_{\text{min-d}}} \) is the number of links that \( q \) intersects with \( p_{\text{min-d}} \). Thus, we have

\[
\Delta D_{p_{\text{min-d}}} = |p_{\text{min-d}}| \cdot D_i > t_{p_{\text{min-d}}} \cdot D_i = \Delta D_q,
\]

where \( \Delta D_{p_{\text{min-d}}} \) is the delay increment of path \( p_{\text{min-d}} \), and \( \Delta D_q \) the delay increment of path \( q \). This inequality illustrates that, during the delay-shifting process, other paths’ delay increments are all less than \( p_{\text{min-d}} \). This gives us a chance to select another path \( p_{\text{min-d}} \) that is different from the last-selected \( p_{\text{min-d}} \) due to the shifting of the points’ relative location. Hence, we select another minimal-delay path, record it, and then shift the point again. These steps will be repeated for a number of times. Finally we check all the recorded paths and choose the minimal-cost one in the feasible path set as the results. As discussed above, the success routing ratio becomes larger with more iteration times of delay-shifting process.

The network we consider is modeled as a graph \( G(V,E) \), where \( V \) is the set of nodes and \( E \), the set of links, and let \( N = |V| \) and \( M = |E| \) denote the number of nodes and the number of links. We denote by \( H \) the maximal possible number of hops in a path, and \( S \), the iteration time of delay-shifting process during each path selection process. The routing success ratio increases with the increase of \( S \), while the complexity of our algorithm also increases. Therefore, we can find the tradeoff between the result’s precision and the time-complexity by adjusting \( S \).

Fig. 2 gives the flow chart of our algorithm. We can use either Bellman-Ford or Dijkstra’s algorithm to compute the minimal-delay (minimal-cost) path. When the Bellman-Ford algorithm is used, the time complexity of our algorithm is \( O((S+2)MH) = O(SMH) \); when the Dijkstra’s algorithm is used, the time complexity is \( O((S+2)N \log N + (S+2)M) = O(SN \log N + SM) \), and both are in polynomial-time complexity to \( N \).

3. Numerical Results

Our experimental network contains 21 nodes, with the average degree of each node 6. For each value of \( S \) and \( D_i \), 50 topologies are generated randomly. In each topology, every link is related with a link delay, which is randomly selected between 0 and 200 time units, and a link cost, between 0 and 200 cost units. For each topology, a connectivity check is performed. If the graph \( G(V,E) \) is not connected, topology will be regenerated. Otherwise, 420 connections are produced, with delay constraint of
each connection uniformly distributed in (0, 600 time units). Here we use the multi-label search algorithm proposed in [6] to search for the optimal path in order to provide a criterion for our algorithm. The multi-label algorithm has an exponential time-complexity to \( N^{2N_{\max} - 2^{N-1}} \), where \( N_{\max} \) is the degree value of the maximum-degree node in the network.

We evaluate the effectiveness of our algorithm using routing success ratio, i.e., the proportion of the number of success routing to the total path-selection times. Note that if there is no solution for DCCOP, no path can be selected by either our algorithm or the multi-label algorithm. We regard this case as the success routing.

![Graph](image)

Fig. 3 Correct Rate w.r.t. \( S \)

![Graph](image)

Fig. 4 Correct Rate w.r.t. \( D_i \)

Fig. 3 compares the trends of success routing ratio for the variation of \( S \) with different values of \( D_i \). From this figure we observe that, correct rate rises as the selected path number \( S \) become larger. Thus, We could find a balance between our algorithm’s reliability and time complexity. Larger value of \( S \) leads to higher reliability and also higher time complexity. However, for some value of \( D_i \) (curve \( D_i=16, D_i=32 \) and \( D_i=64 \), when \( S \) reaches a certain number, its variation has minor influence on the correct rate. Hence, by carefully choosing the value of \( S \) and \( D_i \), our proposed algorithm can solve the DCCOP problem with high reliability and relatively low time complexity.

Fig. 4 shows the correct rate as a function of increase value \( D_i \). Correct rate reaches a peak at the \( D_i \) between 16 and 64. With \( D_i \) continuing to increase, the correct rate drops. This is because a large \( D_i \) will lead to that the path computing process oscillates among a small number of paths, resulting lower probability to find the optimal path.

4. Conclusion

In this paper, we propose a heuristic algorithm for DCCOP problem in a polynomial time complexity. Our algorithm can find the tradeoff between effectiveness and complexity by adjusting the key parameter \( S \), the iteration times of the delay-shifting process. Simulation results show that, in a relatively low time-complexity, our algorithm can find the delay-constrained optimal-cost path with a high probability.

References


