Reliability Problem on All Pairs Quickest Paths

Young-Cheol Bang¹, Hyunseung Choo^{2,*}, and Youngsong Mun³

¹ Department of Computer Engineering, Korea Polytechnic University,
Kyunggi-Do, KOREA
ybang@kpu.ac.kr

² School of Information and Communication Engineering
Sungkyunkwan University, Suwon 440-746, KOREA
choo@ece.skku.ac.kr

³ School of Computing, Soongsil University, Seoul, KOREA
mun@computing.ssu.ac.kr

Abstract. The reliability problem of the quickest path deals with the transmission of a message of size σ from a source to a destination with both the minimum end-to-end delay and the reliability of the path over a network with bandwidth, delay, and probability of fault free on the links. For any value of message size σ , we propose $O(n^2m)$ time algorithm for all-pairs quickest most reliable path problem, where n and m are the number of nodes and the number of arcs or links in the network, respectively.

1 Introduction

We consider point-to-point computer networks represented by a graph $G = (V, A)$ with n nodes and m arcs or links. Any node can be reached from any other node in this network, and two nodes are connected by at most single directed link in each direction. Each directed link $l = (i, j) \in A$ has a bandwidth $B(l) \geq 0$, delay $D(l) \geq 0$, and probability of fault free $0 \leq \pi(l) \leq 1$. A message of σ units can be sent along the link l in $T(l)\sigma / B(l) + D(l)$ time with reliability $\pi(l)$ as in [6].

Consider a simple path P from i_0 to i_k given by $(i_0, i_1), (i_1, i_2), \dots, (i_{k-1}, i_k)$, where $(i_j, i_{j+1}) \in A$, for $j = 0, 1, 2, \dots, (k-1)$, and all $i_0, i_1, \dots, i_k$ are distinct. Subsequently, a simple path is referred to simply as a path. The delay of this path P , denoted by $D[P]$,

is given by $\sum_{j=0}^{k-1} D(l_j)$, where $l_j = (i_j, i_{j+1})$. The bandwidth of this path is defined as

$B(P) = \min_{j=0}^{k-1} B(l_j)$. The reliability of P , denoted by $R(P)$, is $\prod_{j=0}^{k-1} \pi(i_j, i_{j+1})$. The end-

to-end delay of the path P in transmitting a message of size σ is $T(P) = \sigma / B(P) + D[P]$ with reliability $R(P)$.

* Corresponding Author

The path P from s to d is the most reliable (MR) if $R(P)$ is the maximum among all paths from s to d . The path P from s to d is the quickest for message size σ if $T(P)$ is the maximum among all paths from s to d . The path P is the quickest most reliable if it is the quickest path to transmit σ units among all MR paths from s to d .

Determining the end-to-end delay paths is very critical part of achieving QoS (Quality of Service) in wide area network. One of the well-known routing algorithms to support QoS is the quickest path algorithm in networks with bandwidth and delay constraints. The quickest path problem was first introduced by Moore [3] and formulated by Chen and Chin [1]. Due to its importance in the computer network, the problem received strong attention in the area of operations research and was extensively studied by Rosen et al. [5], Rao and Batsell [4], D.T. Lee [2], and Bang et al. [7]. In real-life computer networks, communication links may fail, and hence it is important to determine quickest paths that are reliable as well. Recently, such reliability aspects have been studied by Xue[6], and $O(rm + rn \log n)$ time algorithms were proposed for computing both QMR and the most reliable quickest (MRQ) path from a source to destination when a message size σ is given, where r is the number of distinct link bandwidths [6].

In this paper, we consider all-pairs QMR paths of which it is very difficult to derive an efficient algorithm in terms of time-complexity. Since MRQ is relatively easy, we do not consider the QMR algorithm. The all-pairs version of the classical quickest path problem was solved in [2] with time complexity of $O(mn^2)$. By applying the algorithm of Xue [6] to compute all-pairs QMR algorithm, the time complexity for the algorithm is $O(rm + rn^2 \log n)$ with n nodes, m links, and r distinct bandwidth in the network. In this paper, we present $O(mn^2)$ time algorithm to compute all-pairs QMR paths, which match the best-known complexity for the all-pairs classical quickest path problem.

The rest of paper is organized as follows. In section 2, we present details of algorithm to compute the QMR paths for all pairs of nodes in a given network. In section 3, summary of our research is presented.

2 The Quickest Most Reliable Paths

As introduced, the QMR path P is quickest among all MR paths so that the simplest method is to compute all MR paths from source s to destination t and then select the quickest path among such paths. Since the number of all MR paths is possibly exponential, the most reliable path network (MRPN) was introduced in [6] to derive an efficient polynomial time algorithm to compute the QMR path. A MRPN, originally called shortest path network (SPN) in [6], is a directed and acyclic subnetwork of G that contains all MR paths from s to any other $t \in V$.

A MR path can be computed by invoking the shortest path algorithm with new weight π' such that $\pi'(l) = \log(1/\pi(l))$, where the weight of a path is the sum of the weights of its links. However there can be more than one MR paths from which QMR path may be chosen. To account for such paths, we should modify a classical shortest

path algorithm such as Dijkstra's shortest path algorithm that can compute at most one MR path.

Fig.1 illustrates one-to-all MRPN. Let a be the source in G . Then, the subnetwork illustrated in (b) is the MRPN rooted at a . As shown in (b), any path from a is the MR path. However, this property does not hold for all-pairs MR paths. Figure 2 represents the all-pairs MRPN (AMRPN) after merging all MR paths for all pairs of nodes in G . As shown in Figure 2, a path $P(a, b)$ is the MR path from a to b and a path $P(b, c)$ is the MR path from b to c , but $P(a, b, c)$ is not the MR path. We call the path like $P(a, b, c)$ as the redundant path (RP). To solve the problem caused by RPs, we introduce the labeling technique such that links on any MR path are labeled with source-destination pair. Since single link may be included in multiple MR paths, it may contain multiple labels. Thus, every link on a QMR path has at least one same label.

Fig. 1. (a) Network G (b) Subnetwork after merging all the most reliable paths from a to every other node in G ; Number in each link represents $\pi'(e)$.

Once AMRPN is constructed by merging all labeled links of all the MR paths, All-pairs Quickest Path algorithm (AQP) [2] is applied to AMRPN. As shown in Fig2, since AMRPN may consist of redundant paths, the classical AQP may compute the quickest paths that are not the MR paths. Thus, AQP should be modified to ensure that only edges on appropriate MR paths are considered in computing QMR paths

Our algorithm consists of following three summarized steps.

1. Compute all the MR paths for each pair of nodes $i, j \in V$, and merge such paths to construct AMRPN.
2. Label each link in AMRPN to distinguish redundant paths
3. Compute all pairs quickest paths of AMRPN with enhancements.

In order to compute all the MR paths, let us define the predecessor data structure as a two dimensional array $pred[i, j]$ for $i, j \in V$, where each element of the array points to a linked list. If $pred[i, j] \rightarrow p \rightarrow q$, then there are two paths from i to j with one of them passing through the link (p, j) and the other through the link (q, j) . Thus the data structure $pred$ can store all MR paths for every pair of nodes $i, j \in V$.

Let $\Phi[i, j] = \sum_{e=0}^{k-1} \pi'(l_e)$, where $l_e = (i_e, i_{e+1})$. Let $\tau[i, j]$ be minimum among all

$\Phi[i, j]$. Then we first compute $\tau[i, j]$ for all pairs of node (i, j) with $i, j \in V$. As shown in algorithm AQMR, whenever we find $\Phi[i, j]$ such that $\Phi[i, j] = \tau[i, j]$, new predecessor of node j is added to the list of $pred[i, j]$. Let it call elm . Since at least one of the MR paths from s to j passes through a link (elm, j) , the MRPN can be constructed by merging links (elm, j) for all $j \in V$.

The labeling process can be easily done by backtracking each MR path from every $t \in V$ to s . If a link (i, j) is on the MR path from s to t , then $L_{ij}[s, t]$ is marked, where L_{ij} is the $n \times n$ matrix with $n = |V|$. By doing so, we may distinguish each link used by different MR paths. Once all links in each MR path are labeled, AMRPN can be constructed by merging all MR paths with labeled links. The following algorithm is to construct AMRPN with labeled links.

Fig. 2. Subnetwork after merging all the most reliable paths for all pairs of nodes in G ; Number in each link represents $\pi'(e)$.

Algorithm AMRPN

Comment : Suppose $\tau[u, v]$ are pre-computed for all pairs of nodes (u, v) with $u, v \in V$

1. $\Phi[u, v] = \infty$ for all pairs of nodes (u, v) with $u, v \in V$
2. **for** each node $s \in V$ **do**
3. $\Phi[s, s] = 0$
4. $pred[s, v] = \text{null}$ for all $v \in V$
5. $Q \leftarrow V[G]$
6. **while** $Q \neq \emptyset$ **do**
7. $u \leftarrow$ a node with minimum of $\Phi[s, v]$ from Q
8. **for** each vertex $v \in Adj[u]$ **do**
9. **if** $\Phi[s, v] \geq \Phi[s, u] + \pi'[u, v]$ **then**
10. $\Phi[s, v] = \Phi[s, u] + \pi'[u, v]$
11. **if** $\Phi[s, v] = \tau[s, v]$ **then**
12. $pred[s, v] \rightarrow elm = u$
13. **for** each node $u \in V$ **do**
14. **for** each node $v \in V$ **do**
15. backtrack every path from v to u .
16. **if** a link (i, j) is on a path from s to t **then**
17. mark $L_{ij}[u, v]$

18. **for** each node $u \in V$ **do**
19. **for** each node $v \in V$ **do**
20. **while** $\text{pred}[u, v] \neq \text{Null}$ **do**
21. add link $(\text{pred}[u, v] \rightarrow \text{elm}, v)$ to AMRPN

Lines 1–12 are modifications of Dijkstra's shortest path algorithm to compute all-pairs MR paths. In each iteration, line 11 checks if $\Phi[s, v] = \tau[s, v]$. If so, a predecessor of v is added to list of $\text{pred}[s, v]$. This process can be computed in $O(nm + n^2 \log n)$ by using Fibonacci heap [8]. Once all-pairs MR paths are determined, each MR path is labeled with (s, t) pair in lines 13–17 by backtracking each path from t to s . Since there are at most m links to be traveled for backtracking from each destination to single source, it takes at most $O(nm)$ to label. Thus, the time-complexity of this process for all pairs of nodes is $O(n^2m)$. Lines 18–21 are the process to compute AMRPN by merging links of all-pairs MR paths. Thus the algorithm AMRPN can be constructed in $O(n^2m)$.

To determine all-pairs QMR paths, we use $L_{ij}[s, t]$ which ensures that link (i, j) is on a MR path from s to t . If $L_{ij}[s, t]$ is marked then link (i, j) is on the MR path from s to t . Thus, (i, j) can be considered in computing a QMR path from s to t . Otherwise, (i, j) is ignored.

The following algorithm AQMR is to compute all-pairs QMR path. Let $d[u, v]$ and $D(u, v)$ be the delay of the quickest path from u to v and the delay of link (u, v) , respectively at any iteration of the algorithm. The links of AMRPN are considered in the non-increasing order in lines 4–11; top operation of line 5 returns the top element and removes it from the heap. In each iteration, the chosen link (i, j) is checked if it is on some MR path from u to v in line 8. If yes, the algorithm is identical to that in [3]: it checks if the path via (i, j) is quicker than the quickest path computed so far, and replaces the latter if so. If not, this link (i, j) is not considered further.

Algorithm AQMR(AMRPN)

1. $\text{arc_heap} \leftarrow$ top-heavy heap of all $l \in E$ according to $B(l)$
2. **for** pair $u, v \in V$ **do**
3. $d[u, v] = \infty$;
4. **while** $\text{arc_heap} \neq \emptyset$ **do**
5. $(i, j) = \text{top}(\text{arc_heap})$; let $l = (i, j)$;
6. **if** $d[i, j] > D(l)$ **then**
7. **for** each pair $u, v \in V$ **do**
8. **if** $L_l[u, v]$ is marked **then**
9. **if** $d[u, i] + D(l) + d[j, v] < d[u, v]$ **then**
10. $d[u, v] = d[u, i] + D(l) + d[j, v]$;
11. update QMR path from u to v
12. return all-pairs QMR paths

Lines 1–12 of algorithm AQMR are identical to that of [3] except for the condition in line 8 which ensures that only the links on MR paths are considered in computing the quickest paths, and hence its correctness directly follows. Since each link in

AMRPN has $n \times n$ matrix to keep the labeling information, $O(n^2m)$ space is required. Therefore, the following theorem is as follows.

Theorem 2.1 *The all-pairs quickest most reliable paths for any size of message can be computed by algorithm AQMR with time complexity $O(n^2m)$ and space complexity $O(n^2m)$.*

3 Conclusion

In this paper, we introduced the quickest the most reliability paths for all pairs of nodes in a given network G for any message size σ . These algorithms match the best known computational complexity for the classical all-pairs quickest path problem, namely without the reliability considerations. It would be interesting to obtain all-pairs algorithms for other variations of the quickest path problem such as general bandwidth constraints [9], random queuing errors [10], and dynamic bandwidth constraints [11]. Another future direction is the on-line computation of quickest paths so that path's quality can be traded-off for computational speed.

References

1. Y. L. Chen, Y. H. Chin, "The quickest path problem", *Computers and Operations Research* vol. 17, no. 2, pp. 153–161.
2. D. T. Lee and E. Papadopoulou. "The all-pairs quickest path problem", *Information Processing Letters*, 45(5): 261–267, 1993.
3. J. F. Mollenauer, "On the fastest routes for convoy-type traffic in flow-rate-constrained networks", *Transportation Science*, vol. 10, pp. 113–124, 1976.
4. N. S. V. Rao and S. G. Batsell, "Algorithm for minimum end-to-end delay paths", *IEEE Communications Letters*, vol. 1, no. 5, pp. 152–154.
5. J. B. Rosen, S. Z. Sun, and G. L. Xue, "Algorithms for the quickest path problem and the enumeration of quickest paths", *Computers and Operations Research*, vol. 18, no. 6, pp. 579–584.
6. Guoliang Xue, "End-to-End data paths: Quickest or most reliable?", *IEEE Communications Letters*, vol. 2, no. 6, pp. 156–158.
7. Y. C. Bang, S. Radhakrishnan, N. S. V. Rao, and S. G. Batsell, "On update algorithms for quickest paths", *Computer Communications*, vol. 23, pp. 1064–1068.
8. T. H. Cormen, C. E. Leiserson, and R. L. Rivest, "Introduction to Algorithms", New York: McGraw-Hill, 1990.
9. N. S. V. Rao and N. Manickam. General quickest paths and path-tables. *Computer Systems: Science and Engineering*, 17(4/5):235–239, 2002.
10. N. S. V. Rao. LetLets for end-to-end delay minimization in distributed computing over Internet Using two-paths. *International Journal of High Performance Computing Applications*, 16(3), 2002.
11. W. C. Grimmell and N. S. V. Rao. On source-based route computation for quickest paths under dynamic bandwidth constraints. *Journal of Foundations of Computer Science*, 1002, in press.