Algorithms for All-Pairs Reliable Quickest Paths *

Young-Cheol Bang[1]†, Nageswara S.V. Rao[2], and S. Radhakrishnan[3]

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

[2] Computer Science and Mathematics Division, Oak Ridge National Laboratory
Oak Ridge, Tennessee 37831-6364
raons@ornl.gov

[3] School of Computer Science, University of Oklahoma
Norman, Oklahoma 73019
Sridhar@cs.ou.edu

Abstract. We consider the reliable transmission of messages via quickest paths in a network with bandwidth, delay and reliability parameters specified for each link. For a message of size σ , we present algorithms to compute all-pairs quickest most-reliable and most-reliable quickest paths each with time complexity $O(n^2m)$, where n and m are the number of nodes and links of the network, respectively.

1 Introduction

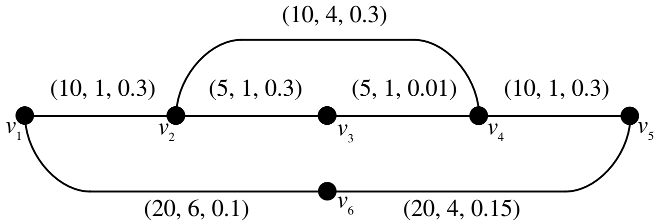
We consider a computer network represented by a graph $G = (V, E)$ with n nodes and m links. Each link $l = (i, j) \in E$ has a *bandwidth* $B(l) \geq 0$, *delay* $D(l) \geq 0$, and *reliability* $0 \leq \pi(l) \leq 1$, which is the probability of l being fault free. A message of σ units can be sent along the link l in $T(l) = \sigma / B(l) + D(l)$ time with reliability $\pi(l)$ as in [13].

Consider a *path* P from i_0 to i_k given by $(i_0, i_1), (i_1, i_2), \ldots, (i_{k-1}, i_k)$, where $(i_{j,} i_{j+1}) \in E$, for $j = 0, 1, \ldots(k - 1)$, and $i_0, i_1, \ldots, i_k$ are distinct. The *delay* of path P is $D(P) = \sum_{j=0}^{k-1} D(l_j)$, where $l_j = (i_j, i_{j+1})$. The *bandwidth* of P is $B(P) = \min_{j=0}^{k-1} B(l_j)$. The *reliability* of P is $R(P) = \prod_{j=0}^{k-1} \pi(i_j, i_{j+1})$. The end-to-end delay of P in transmitting a message of size σ is $T(P) = \sigma / B(P) + D(P)$ with reliability $R(P)$.

* Research of Rao is sponsored by Defense Advanced Research Projects Agency under MIPR No. K153, and by Engineering Research Program and High-Performance Networking Program of Office of Science, U. S. Department of Energy under Contract No. DE-AC05-00OR22725 with UT-Battelle, LLC. Research of Bang is sponsored by Electronics and Telecommunications of Research Institute under Contract No. 1010-2002-0057.

† Corresponding Author

P.M.A. Sloot et al. (Eds.): ICCS 2003, LNCS 2658, pp. 678–684, 2003.



	B(.)	D(.)	R(.)	T(.)
P_1 : {v_2, v_2, v_3, v_4, v_5}	$B(P_1) = 5$	$D(P_1) = 4$	$R(P_1)= 0.00027$	$T(P_1) = \sigma /5 + 4$
P_2 : {v_1, v_2, v_4, v_5}	$B(P_2) = 10$	$D(P_2) = 8$	$R(P_2) = 0.027$	$T(P_2) = \sigma /10 + 8$
P_3 : {v_1, v_6, v_5}	$B(P_3) = 20$	$D(P_3)= 10$	$R(P_3) = 0.015$	$T(P_3)= \sigma /20 + 10$

Fig. 1. Example network

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 is the quickest for message size σ if $T(P)$ is the minimum among all paths from s to d. The path P is the *quickest most-reliable* (QMR) if it is the quickest for σ among all MR paths from s to d. The P is the *most-reliable quickest* (MRQ) if it has highest reliability among all quickest paths from s to d for σ . For the network in Figure 1, for $s = v_1$ and $d = v_5$, P_2 is the MR path with $R(P_2) = 0.027$, and it is also the QMR path for any σ . For $\sigma < 40$, P_1 is the quickest path, and for $\sigma >$ 40, and P_3 is the quickest path. All P_1, P_2, and P_3 are quickest paths for $\sigma = 40$. Then P_1 is the MRQ path for $\sigma < 40$, and P_3 is the MRQ path for $\sigma > 40$, and P_2 is the MRQ path for $\sigma = 40$.

The classical quickest path problem was extensively studied [7, 4, 12, 2], and has received increased attention recently due to its applicability to computer networks [9, 13]. 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 [13], and $O(rm + rn \log n)$ time algorithms were proposed for computing both QMR and MRQ paths from s to d for message size σ , where r is the number of distinct link bandwidths.

In this paper, we consider the all-pairs versions of computing QMR and MRQ paths considered in Xue [13]. The all-pairs version of the classical quickest path problem was solved in [3, 6] with time complexity of $O(n^2m)$. By applying the algorithms of Xue [13] for each $s \in V$, we can compute QMR and MRQ paths between all pairs s and d with time complexity $O(nrm + rn^2 \log n)$; since $r \leq m$, we have the complexity $O(nm^2 + n^2m \log n)$. In this paper, we present $O(n^2m)$ time algorithms to compute all-pairs QMR and MRQ paths, which match the best-known complexity for the all-pairs classical quickest path problem.

2 All-Pairs Quickest Most Reliable Paths

For any pair u, $v \in V$, a MR path is computed using the All-pairs Shortest Path (ASP) algorithm [1] with weight $\pi'(l) = \log(1 / \pi(l))$, for $l \in E$, where the weight of a path is the sum of the weights of its links. We compute $[\,\Phi[i, j]_{i,\, j \in V}\,] = [\,\Phi[i, j]\,]$ using ASP, where $\Phi[u,v]$ is the shortest weight of a path from i to j under the weight π'. There can be more than one MR paths from which the QMR path may be chosen. To account for such MR paths, we identify each link (i, j) that is on some most MR path from u to v, by checking the condition $\Phi[u,i] + \pi'(l) + \Phi[j,v] = \Phi[u,v]$ at an appropriate step in our algorithm.

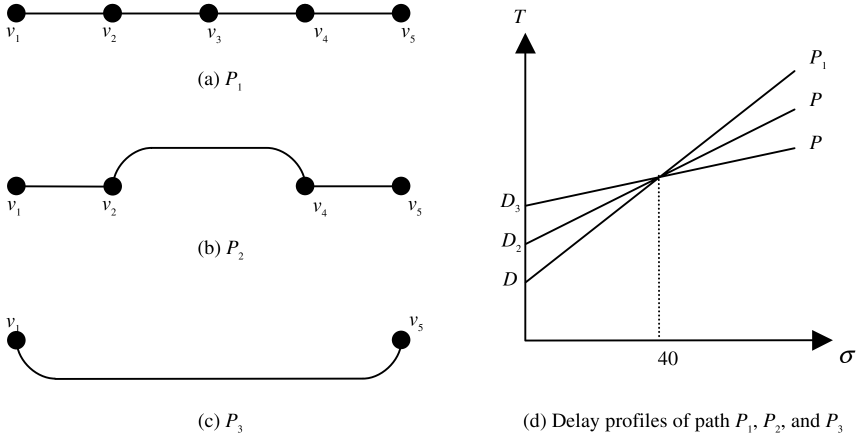
The rest of the algorithm is a modification of the All-pairs Quickest Path (AQP) algorithm of Chen and Hung [3] which ensures that only edges on appropriate MR paths are considered in computing QMR paths. We use three arrays, denoted by $d[u, v]$, $b[u, v]$ and $t[u, v]$ for u, $v \in V$, to represent the delay, bandwidth and end-to-end delay of the quickest path from u to v, respectively, at any iteration of the algorithm. The edges of G are considered in the non-increasing order in lines 5-13; top operation in line 6 returns the top element and removes it from the heap. In each iteration, the chosen edge (i, j) is checked if it is on some MR path from u to v in line 8. If yes, the algorithm identical to that in [3]: it checks if the path via (i, j) is quicker than the quickest path computed so far, and replaces that latter if so. If not, this edge (i, j) is not considered further.

```
Algorithm AQMR ($G, D, B, \pi, \sigma$)
1.  compute $[\,\Phi[i, j]\,]$ using ASP ($G$, $\pi'$)
2.  **for** pair $u$, $v \in V$ **do**
3.      $d[u, v] = \infty$; $b[u, v] = 0$; $t[u, v] = \infty$;
4.  arc_heap $\leftarrow$ top-heavy heap of all $l \in E$ according to bandwidth;
5.  **while** arc_heap $\neq \emptyset$ **do**
6.      $(i, j)$ = top(arc_heap); let $l = (i, j)$;
7.      **for** each pair $u$, $v \in V$ **do**
8.          **if** ( $\Phi[u,i] + \pi'(l) + \Phi[j,v] = \Phi[u,v]$ ) **then**
9.              $D_1 = d[u, i] + D(l) + d[j, v]$
10.             **if** $D_1 < d[u, v]$ **then**
11.                 $d[u, v] = D_1$
12.                 **if** $\sigma / B(l) + d[u, v] < t[u, v]$ **then**
13.                     $b[u, v] = B(l)$; $t[u, v] = \sigma / B(l) + D_1$;
```

Lines 2-11 of algorithm AQMR are identical to that of [3] except for the condition in line 8 which ensures that only the edges on MR paths are considered in computing the quickest paths, and hence its correctness directly follows. All pairs shortest path algorithm has the complexity of $O(n^3)$ in line 1. The complexity of lines 2-13 is $O(n^2m)$. Thus, the all-pairs QMR paths for message size σ are computed by algorithm AQRM with time complexity $O(n^2m)$ and space complexity $O(n^2)$.

3 All-Pairs Most-Reliable Quickest Paths

To compute an MRQ path from s to d, we have to “account” for all quickest paths from s to d. Note that AQP returns a quickest path from s to d, which may not be a MRQ path, and hence a simple condition similar to line 8 of AQMR does not work. In particular, it is not sufficient to check if an edge l is on a quickest path P_1 with bandwidth $B(l)$; in fact, l can be on a quickest path with any $b = B(P_1) \leq B(l)$. In our algorithm, we compute the largest of such b and place l at an appropriate step in the computation, which is an iterative process similar to AQMR. Let $t[u, v]$ represent the end-to-end delay of quickest path from u to v for σ .



(a) P_1

(b) P_2

(c) P_3

(d) Delay profiles of path P_1, P_2, and P_3

Fig. 2. (a), (b), and (c) represent paths P_1, P_2, and P_3 for the network in Fig.1. (d) shows the plot representation of P_1, P_2, and P_3.

Let $b_1 < b_2 < \ldots < b_r$ be the distinct values of $B(l)$, $l \in E$. Let G_b be the subnetwork with all edges of G whose bandwidth is greater than or equal to b. To compute MRQ paths, we first compute all-pairs quickest paths in G using AQP with the following enhancement. For each bandwidth value b_k and pair u, $v \in V$, we store a matrix $[\,d_{b_k}[u,v]\,]$ where $d_{b_k}[u,v]$ is the delay of the shortest path from u to v in G_{b_k} . These matrices can be computed during the execution of AQP. Let $\Theta(u,v,\sigma)$ be the set of bandwidths of all quickest paths from u to v for given σ . As shown in Fig.2, we have $\Theta(v_1,v_5,20) = \{5\}, \Theta(v_1,v_5,40) = \{5,\ 10,\ 20\}$, and $\Theta(v_1,v_5,60) = \{20\}$ for the example of Fig.1. The following is an important property of $\Theta(u,v,\sigma)$, which is due to the fact that a quickest path from u to v is a shortest path in G_{b_i} for some b_i.

Lemma 3.1 $\Theta(u,v,\sigma) \neq \varnothing$ if and only if there is a shortest path from u to v in G_b for some b$\in$ $\Theta(u,v,\sigma)$.

In AMRQ, we organize the sets $\Theta(u,v,\sigma)$'s as stacks with bandwidths decreasing top to bottom. We use AQP [3] to compute [$t[u, v]$] and [$d_b[u, v]$] in line 1. In line 2, we compute all $\Theta(u,v,\sigma)$'s for each pair u, $v \in V$ with time complexity $O(m)$: for each b_k we simply check for the condition $t[u, v] = \sigma / b_k + d_{B_k} [u, v]$. There are $O(n^2m)$ iterations in the rest of the algorithm, where edges are considered in decreasing order of bandwidth with which they participate in quickest paths (if at all). In each iteration, we consider the current link bandwidth $B(l)$, and pair u, $v \in V$. Lines 9-10 compute the maximum bandwidth with which the edge l is used in a quickest path from u to v. The reliability of new path via l from u to v is then computed and the existing value is replaced appropriately in lines 11-12. Consider that as a result of while loop in lines 9-10, the retrieved bandwidth $b_{[u, v]}$ is strictly smaller than $B(l)$ if $b_{[u, v]}$ corresponds to link l_1, no more pop operations on $\Theta(u,v,\sigma)$ will performed until all edges with bandwidths in the range [$B(l_1)$, $B(l)$] have been retrieved from the heap and processed. For each pair u, $v \in V$, this algorithm can be viewed in terms of alternating subsequences of top operations on arc_heap and pop operations on stack $\Theta(u,v,\sigma)$ with no backtracking involved. In actual execution, however, all these subsequences corresponding to various $u - v$ pairs are intermingled among themselves as well as subsequences of top operations.

```
Algorithm AMRQ ($G, D, B, \pi, \sigma$)
1.  compute [$t[u, v]$] and [$d_b[u, v]$] using AQP ($G, B, D, \sigma$)
2.  compute stack $\Theta(u,v,\sigma)$ for each pair $u$, $v \in V$
3.  for each pair $u$, $v \in V$ do
4.      $b_{[u, v]}$ = top( $\Theta(u,v,\sigma)$ );
5.  arc_heap = top_heavy heap of all edges of $G$ according to the bandwidth
6.  while not arc_heap $\neq \emptyset$ do
7.      $(i, j)$ = top(arc_heap); let $l = (i, j)$;
8.      for each pair $u$, $v \in V$ do
9.          while ($B(l) < b_{[u, v]}$) and ( $\Theta(u,v,\sigma) \neq \emptyset$ ) do
10.             $b_{[u, v]}$ = pop( $\Theta(u,v,\sigma)$ );
11.         if ($B(l) \geq b_{[u, v]}$) and ( $d_{b_{[u,v]}}[u,v] = d_{b_{[u,v]}}[u,i] + D(i, j) + d_{b_{[u,v]}}[j,v]$ ) then
12.             $\Phi[u,v] \leftarrow min\{ \Phi[u,v] , \Phi[u,i] + \pi'(i, j) + \Phi[j,v] \}$;
```

Consider the correctness of the algorithm. If an edge l is on a quickest path from u to v, there is a corresponding $b \in \Theta(u,v,\sigma)$ by lemma 3.1. Consider a MRQ path P_1 and let l_{p_1} be the link with the lowest bandwidth, which implies $B(l_{p_1}) \in \Theta(u,v,\sigma)$. All other links l of P_1 are retrieved in line 7 before l_{p_1}: each $B(l)$ is checked with $b' \geq b$ or higher in line 11, and hence is accounted for in computing the shortest paths in G_b. Hence, by end of iteration in which l_{p_1} is examined, the reliability of P_1 is computed in line 12, since all its edges would have satisfied the condition in line 11 and hence accounted for in the reliability computation.

The complexity of lines 1 and 2 is $O(n^2m)$. For each pair u, v, each edge is considered at most one time in lines 7 – 12, and hence the time and space complexities of AMRQ are both $O(n^2m)$.

4 Conclusion

We presented algorithms to compute most-reliable quickest and quickest most-reliable paths between all pairs of nodes in a network. 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 [11], random queuing errors [8], dynamic bandwidth constraints [5] and various other routing mechanisms [10]. 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. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison Wesley Pub., Reading, MA, 1974.
2. 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, 2000.
3. G. H. Chen and Y. C. Hung, On the quickest path problem, *Information Processing Letters*, vol 46, pp. 125–128, 1993.
4. Y. L. Chen and Y. H. Chin, The quickest path problem, *Computers and Operations Research*, vol. 17, no. 2, pp. 53–161, 1990.
5. 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*, 2002, in press.
6. D. T. Lee and E. Papadopoulou, The all-pairs quickest path problem, *Information Processing Letters*, vol. 45, pp. 261–267, 1993.
7. J. F. Mollenauer, On the fastest routes for convoy-type traffic in flowrate-constrained networks, *Transportation Science*, vol. 10, pp. 113–124, 1976.
8. 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*, vol. 16, no. 3, 2002.
9. 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, 1997.
10. N. S. V. Rao, W. C. Grimmell, S. Radhakrishnan, and Y. C. Bang, Quickest paths for different network router mechanisms, Technical Report ORNL/TM-2000/208, Oak Ridge National Laboratory, Oak Ridge, TN, 2000.

11. N. S. V. Rao and N. Manickam, General quickest paths and path-tables, *Computer Systems: Science and Engineering*, vol. 17, no. 4/5, pp. 235–239, 2002.
12. 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, 1991.
13. G. Xue, End-to-end data paths: Quickest or most reliable?, *IEEE Communications Letters*, vol. 2, no. 6, pp. 156–158, 1998.