An International Journal
computers & mathematics with applications

PERGAMON Computers and Mathematics with Applications 40 (2000) 885–895
www.elsevier.nl/locate/camwa

Terminal-Pair Reliability in ATM Virtual Path Networks

STEEN J. HSU, YU G. CHEN AND MARIA C. YUANG
Department of Computer Science and Information Engineering
National Chiao Tung University, Taiwan, R.O.C.

(Received May 1998; revised and accepted March 1999)

Abstract—Terminal-pair reliability (TR) in an asynchronous transfer mode (ATM) virtual path (VP) network corresponds to probabilistic quantification of robustness between two VP terminators, given the VP layout and the failure probabilities of physical links. Existing TR algorithms are shown to be unviable for ATM VP networks owing to either high complexities or failure dependency among VPs. The goal of the paper is to propose efficient algorithms for the computation of TR between two VP terminators by means of variants of path-based and cut-based partition methods which have been effectively used for the computation of TR in traditional networks. The first variant, called the path-based virtual path reliability (PVPR) algorithm, partitions the search space based on a physical path embedding the shortest route of VPs from the source terminator to the destination terminator. The second variant, called the cut-based virtual path reliability (CVPR) algorithm, in lieu, performs the partition on the basis of a physical cutset separating the source from the remaining terminators. In both algorithms, each subproblem is recursively processed by means of partition until the source and destination terminators are contracted or disconnected. Experimental results demonstrate that, compared to one promising TR algorithm (called EBRM), both the PVPR and CVPR algorithms improve the running time by five orders of magnitude. In particular, the CVPR outperforms EBRM more than PVPR does in terms of computation time. The two algorithms and their promising results consequently facilitate the real-time computation of the reliability or robustness of ATM VP networks.

Keywords—Asynchronous transfer mode (ATM), Virtual path (VP), Terminal-pair reliability (TR), Path-based partition, Cut-based partition.

1. INTRODUCTION

Asynchronous transfer mode (ATM) [1–3] has been widely accepted to support the integration and economical transport of multirate traffic in broadband integrated services digital networks (B-ISDNs) [2]. In particular, the virtual path (VP) concept [4,5] in ATM networks has been proposed to significantly reduce control costs by grouping a number of virtual channel (VC) connections into a single unit, i.e., a VP connection. In such an ATM VP network, a VC is embedded in a concatenation of one or more VPs, and a VP is embedded in a concatenation of one or more physical links. Nodes switching and terminating VPs are called VP switches and VP terminators (or VC switches), respectively. Switches serving both functions are called VP/VC switches.

0898-1221/00/$ - see front matter Typeset by $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX
PII: S0898-1221(00)00204-2

The fundamental advantage of VPs is the allowance of a large group of VCs to be handled as a single unit, resulting in faster processing per connection, lower switching complexity, and superior utilization of network resources. An additional advantage is the assurance of network reliability despite link failures or network congestion by rerouting those impaired VCs in real time to alternative pre-established backup VPs [6,7]. Terminal-pair reliability (TR) in this case corresponds to probabilistic quantization of robustness of a given VP layout between any two VP terminators. As a result, the success of such real-time restoration of connections from network failures and congestion hinges on the efficiency of the TR computation. This paper aims for the design of efficient TR algorithms for ATM VP networks.

Existing TR algorithms [8–19], which have been mostly designed for traditional circuit-switched-based networks, can be categorized into two classes. The first class regards link failures as independent events, whereas the second class considers failures as dependent events. In the first class [8,9,11,16–19], existing algorithms efficiently computed TR by means of the path-based [9,11] and the cut-based [8,16] partition methods. These algorithms effectively partition the search space and recursively process the generated subproblems until the source and the destination are contracted or disconnected. These algorithms are viable but inappropriate for ATM VP networks owing to failure dependency among VPs.

In the second class, the ϵ model [12], requiring the specification of an exponential number of conditional probabilities of link failures, analyzed TR by chain rule expansion. The Page and Perry model [13] applied the pivotal decomposition theorem to factor out the dependent failures resulting in $O(2^n)$ of subproblems to be generated, where n is the number of links. Lam and Li [14] proposed an event-based reliability model (EBRM) to analyze TR by means of existing efficient TR algorithms of the first class with a procedure of transformation augmented. These algorithms, which incur high complexity rising exponentially with the number of links, render the TR computation for VP networks impracticable.

In this paper, we propose two efficient algorithms for the computation of TR between two VP terminators by means of variants of the path-based and cut-based partition methods. The first variant, called the path-based virtual path reliability (PVPR) algorithm, partitions the search space of the problem into a set of subproblems based on a physical path embedding the shortest route of VPs from the source terminator to the destination terminator. The second variant, called the cut-based virtual path reliability (CVPR) algorithm, instead performs the partition on the basis of a physical cutset separating the source from the remaining terminators. In both algorithms, each subproblem is recursively processed by means of partition until the source and destination terminators are contracted or disconnected.

By partitioning based on the physical links and effectively reducing the number of generated subproblems, the PVPR and CVPR algorithms, as will be shown, dramatically reduce the computational complexity. Experimental results demonstrate that, compared to EBRM, both the PVPR and CVPR algorithms improve the running time by five orders of magnitude. In particular, the CVPR outperforms EBRM more than PVPR does in terms of computation time. The CVPR and PVPR algorithms and their promising results consequently facilitate the real-time computation of the reliability or robustness of ATM VP networks.

The rest of the paper is organized as follows. Section 2 presents a brief overview of the EBRM algorithm and discusses its application to ATM VP networks. The PVPR and CVPR algorithms are then formally proposed in Section 3. Experimental results are demonstrated in Section 4. Finally, Section 5 gives concluding remarks.

2. OVERVIEW OF THE EBRM ALGORITHM

The EBRM algorithm computes reliability subject to independent failure-causing events, each occurrence of which causes the simultaneous failures of several network links. Conceptually, the algorithm initially expresses the TR measurement as a function of the success/failure probabilities of network links. This can be achieved by means of performing existing TR algorithms assuming independent network failures. Each term of the reliability function is then further expanded into a function of the occurrence probabilities of the failure-causing events. The algorithm is described in more detail via the following example.

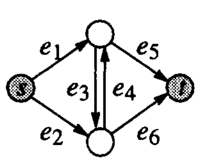
Consider a network (G) with a source (s), a destination (t), and the failure-causing events, as shown in Figure 1. Based on an efficient TR algorithm [10], the TR measurement from s to t, denoted as $\mathrm{Rel}(G)$, can be first expressed as

$$\begin{aligned}\mathrm{Rel}(G) = p_1p_5 + p_2p_6 + p_1p_3p_6 + p_2p_4p_5 - p_1p_2p_5p_6 - p_1p_3p_5p_6 - p_1p_2p_4p_5 - p_1p_2p_3p_6 \\ -p_2p_4p_5p_6 + p_1p_2p_3p_5p_6 + p_1p_2p_4p_5p_6,\end{aligned} \tag{1}$$

where p_i (i = 1 to 6) represents the probability that link e_i is operational. Notice that the complexity of deriving such $\mathrm{Rel}(G)$ in equation (1) increases exponentially with the number of network links. Since the failure-causing events are assumed to be independent, the probability that a group of links are all operational is just the product of the nonoccurrence probabilities of the events involved. Consequently, $\mathrm{Rel}(G)$ can be further expressed as

$$\mathrm{Rel}(G) = p_{\bar{a}}p_{\bar{b}}p_{\bar{e}} + p_{\bar{a}}p_{\bar{d}}p_{\bar{e}} - p_{\bar{a}}p_{\bar{b}}p_{\bar{d}}p_{\bar{e}}, \tag{2}$$

where $p_{\bar{x}}$, $x = a$, b, c, d, or e, corresponds to the probability of nonoccurrence of failure-causing event x. It is also worth noticing that, since the number of product terms in $\mathrm{Rel}(G)$ increases exponentially with the number of network links, the transformation from equation (1) to (2) requires exponential time.



(a) Network G.

Failure-Causing Events	Pertinent Links
a	e_1, e_2
b	e_3, e_4, e_5
c	e_3, e_4
d	e_6
e	e_1, e_3, e_4, e_6

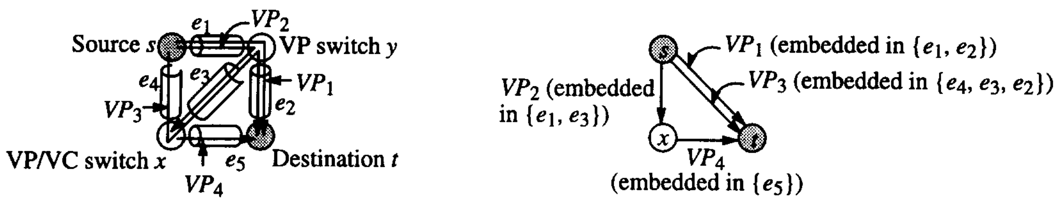
(b) Failure-causing events and pertinent links.

Figure 1. An example for illustrating the EBRM algorithm.

By regarding VPs and physical links as network links and failure-causing events, respectively, the EBRM algorithm can be applied to the computation of TR in ATM VP networks. Unfortunately, as depicted in the previous example, both deriving a TR expression and transforming a VP-based TR expression to a link-based TR expression require time complexities which increase exponentially with the number of VPs [20]. Consequently, the EBRM algorithm is impractical for ATM networks which often possess a large number of VPs. To alleviate this problem, we propose two efficient TR algorithms, namely the PVPR and CVPR algorithms, by means of the path-based and the cut-based partition methods described as follows.

3. PVPR AND CVPR ALGORITHMS

Generally, the PVPR and CVPR algorithms employ the factoring theorem [13] to partition the search space by means of different physical partition bases. Given an ATM network (G) with a source terminator (s) and a destination terminator (t), the PVPR algorithm uses the physical path (referred to as the shortest *s–t path* hereafter) embedding the shortest route of VPs from s to t as the partition basis in an attempt to locally minimize the number of generated subproblems. In contrast, the CVPR algorithm employs the physical cutset (referred to as the



(a) An ATM network with a VP layout. (b) Equivalent VP graph G.

Figure 2. An example of an ATM VP network.

source cut hereafter), separating s from the remaining terminators, as the partition basis to reduce the complexity of partitioning. In both algorithms, each subproblem is recursively processed by means of partition until s and t are contracted or disconnected.

Without loss of generality, we assume that VPs in network G are unidirectional. The failures of physical links are assumed to be mutually independent. Figure 2a depicts an example of an ATM VP network, consisting of a VP/VC switch, a VP switch, and two VP terminators. Moreover, Figure 2b shows the equivalent VP graph which will be used throughout the rest of the section. Notice that the equivalent VP graph is logically identical to the original ATM VP network. The transformation between two graphs requires no computation time and is only illustrated for easier illustration purpose.

3.1. PVPR Algorithm

Initially, let us define the length of a VP to be the number of physical links traversed by this VP. The *shortest s–t VP route* is defined as the minimum-length route of VPs from s to t. Moreover, the *shortest s–t path* is defined as the set of the physical links comprising the shortest s–t VP route. Now, by the factoring theorem, the TR measurement from s to t in G, i.e., $\mathrm{Rel}(G)$, can be represented given the shortest s–t path $\{e_1, e_2, \dots, e_l\}$, as

$$\begin{aligned}\mathrm{Rel}(G) &= q_1 \times \mathrm{Rel}(G - e_1) + p_1 q_2 \times \mathrm{Rel}(G * e_1 - e_2) + \cdots \\ &\quad + p_1 \dots p_{l-1} q_l \times \mathrm{Rel}(G * e_1 * \cdots * e_{l-1} - e_l) + p_1 \dots p_{l-1} p_l \times \mathrm{Rel}(G * e_1 * \cdots * e_{l-1} * e_l),\end{aligned}$$

where p_i (q_i) represents the success (failure) probability of link e_i, and "$*$" ("$-$") represents the contracting (deleting) operation of physical links. A VP emanating from s is contracted if all the physical links used by the VP are contracted, resulting in the contraction of the ending terminator of the VP with s. Notice that $\mathrm{Rel}(G * e_1 * \cdots * e_{l-1} * e_l)$ is equal to one due to the contraction of s and t. Subproblems $\mathrm{Rel}(G * e_1 * \cdots * e_{i-1} - e_i)$, $i = 1$ to l, are then recursively processed by means of partitioning based on the shortest s–t path until s and t are disconnected. The detailed PVPR algorithm is further outlined in Figure 3.

An example of illustrating how the PVPR algorithm performs for the network given in Figure 2 is shown in Figure 4. Initially, the shortest s–t VP route $\{\mathrm{VP}_1\}$ and the shortest s–t path $\{e_1, e_2\}$ can be simply derived according to the existing shortest path algorithms [21] and the definition given above. $\mathrm{Rel}(G)$ can thus be decomposed as $\mathrm{Rel}(G) = q_1 \times \mathrm{Rel}(G - e_1) + p_1 q_2 \times \mathrm{Rel}(G * e_1 - e_2) + p_1 p_2$. Both subproblems $\mathrm{Rel}(G - e_1)$ and $\mathrm{Rel}(G * e_1 - e_2)$ are continuously processed by means of partition until s and t are disconnected. Finally, $\mathrm{Rel}(G)$ is expressed as $\mathrm{Rel}(G) = q_1 p_4 p_3 p_2 + p_1 q_2 p_3 p_5 + p_1 p_2$.

The PVPR algorithm successfully reduces the computational complexity by performing the partition on the basis of physical links which usually yield smaller space than that of VPs. Notice that, based on the PVPR algorithm, the numbers of subproblems generated by partitioning are locally minimized at the expense of executing the path-searching algorithm for finding the partition basis in each subproblem.

Algorithm Rel_PVPR(G)

Input: ATM VP network G with source s, destination t, and the failure probabilities of the physical links;

Output: Terminal-pair reliability R between s and t;

```
Begin
    R := 0.0;
    factor := 1.0;
    if s = t return 1.0;
    Determine the shortest s–t path {e1, e2, ..., el} in G;
    if {e1, e2, ..., el} ≠ ∅
        for i := 1 to l do R := R + factor × qi × Rel_PVPR(G − ei);
                           G := G * ei;
                           factor := factor × pi;
        endfor
        R := R + factor;
    endif
    return R;
End
```

Figure 3. The PVPR algorithm.

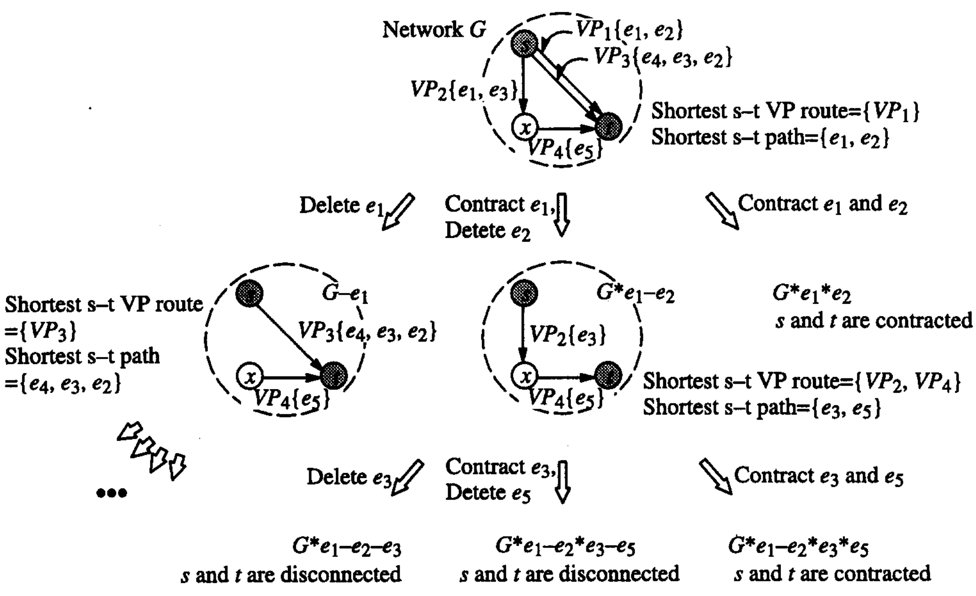


Figure 4. The PVPR algorithm—an example.

3.2. CVPR Algorithm

In the CVPR algorithm, the partition is performed based on the *source cut*, defined as the set of physical links first encountered by the VPs emanating from s. Given the source cut $\{e_1, e_2, \ldots, e_l\}$, by the factoring theorem, Rel(G) can be represented as

$$\mathrm{Rel}(G) = p_1 \times \mathrm{Rel}(G * e_1) + q_1 p_2 \times \mathrm{Rel}(G - e_1 * e_2) + \cdots$$
$$+ q_1 \ldots q_{l-1} p_l \times \mathrm{Rel}(G - e_1 - \cdots - e_{l-1} * e_l) + q_1 \ldots q_{l-1} q_l \times \mathrm{Rel}(G - e_1 - \cdots - e_{l-1} - e_l),$$

where p_i (q_i) represents the success (failure) probability of link e_i and "$*$" ("$-$") represents the contracting (deleting) operation of physical links. Similar to the PVPR algorithm, a VP emanating from s is contracted if all the physical links used by the VP are contracted. Notice that $\mathrm{Rel}(G - e_1 - \cdots - e_{l-1} - e_l)$ is equal to zero due to the disconnection between s and t.

```
Algorithm Rel_CVPR(G)
Input: ATM VP network G with source s, destination t, and the failure
       probabilities of the physical links;
Output: Terminal-pair reliability R between s and t;
Begin
    R := 0.0;
    factor := 1.0;
    if s = t return 1.0;
    Determine the source cut {e1, e2, ..., el} in G;
    if {e1, e2, ..., el} ≠ ∅
        for i := 1 to l do R := R + factor × pi × Rel_CVPR(G * ei);
                           G := G − ei;
                           factor := factor × qi;
        endfor
    endif
    return R;
End
```

Figure 5. The CVPR algorithm.

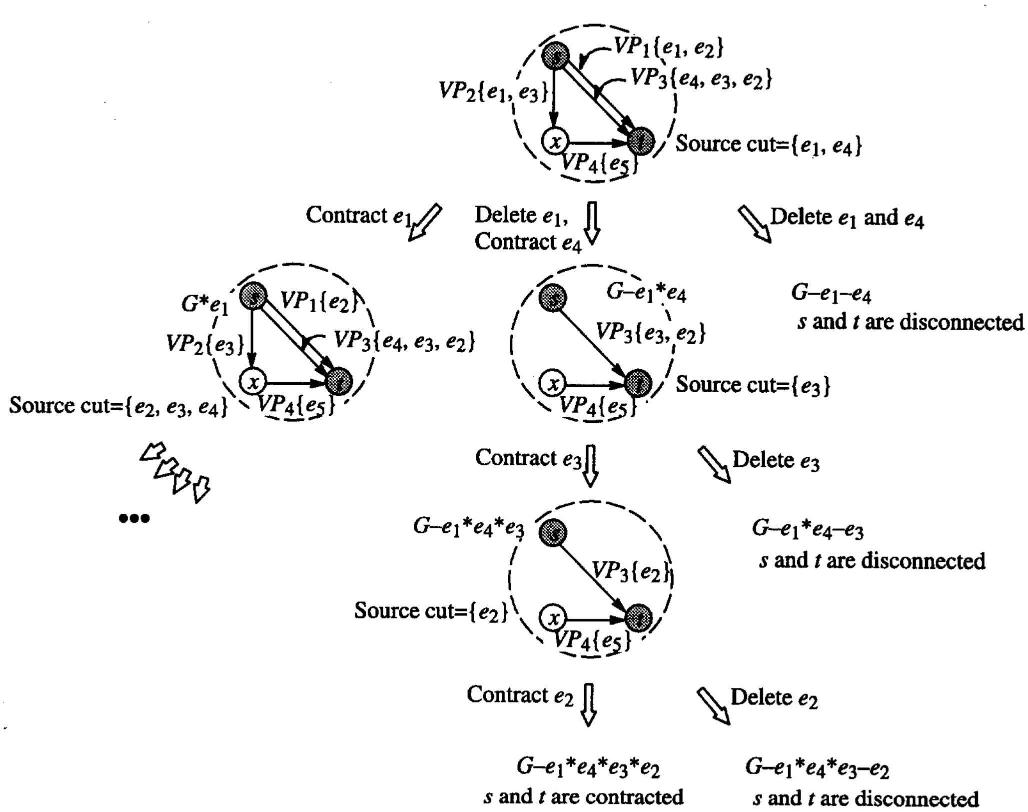


Figure 6. The CVPR algorithm—an example.

Subproblems $\mathrm{Rel}(G - e_1 - \cdots - e_{i-1} * e_i)$, $i = 1$ to l, are then recursively processed by means of partitioning based on the source cut until s and t are contracted or disconnected. The detailed CVPR algorithm is further outlined in Figure 5.

An example of illustrating how the CVPR algorithm performs for the network given in Figure 2 is shown in Figure 6. Initially, by definitions, the source cut is derived as $\{e_1, e_4\}$. $\mathrm{Rel}(G)$ is then decomposed as $\mathrm{Rel}(G) = p_1 \times \mathrm{Rel}(G * e_1) + q_1 p_4 \times \mathrm{Rel}(G - e_1 * e_4)$. Both subproblems

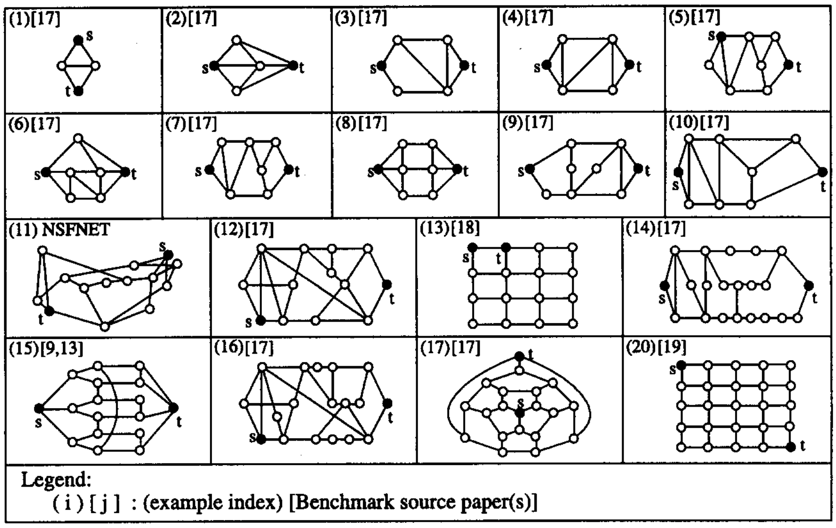


Figure 7. Benchmarks.

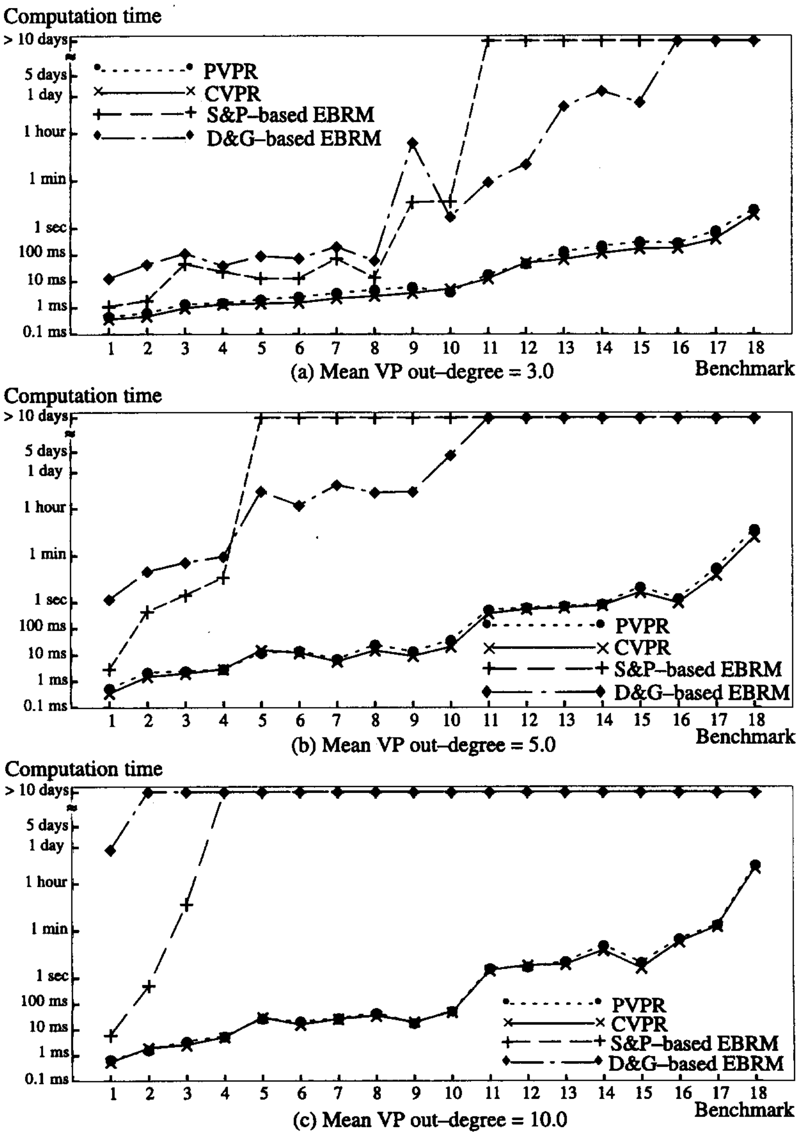


Figure 8. Computation time for the benchmarks.

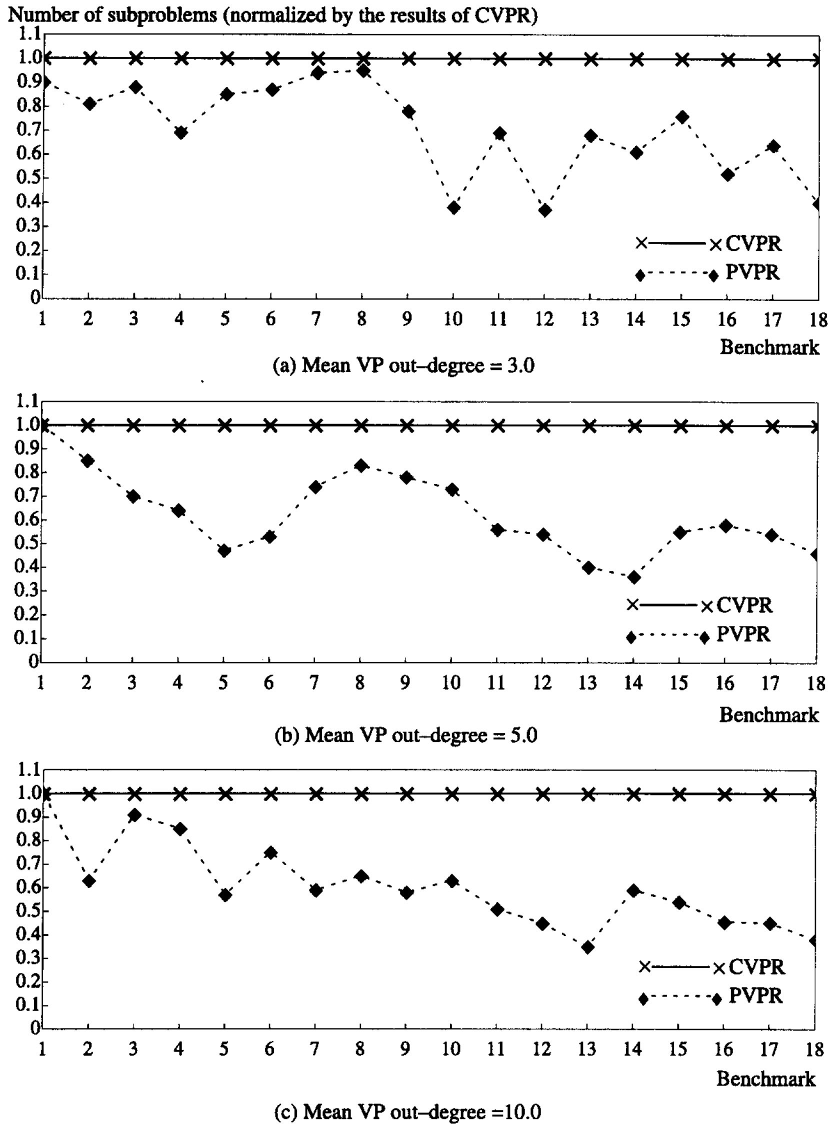


Figure 9. Comparison of the number of subproblems between PVPR and CVPR.

$\text{Rel}(G * e_1)$ and $\text{Rel}(G - e_1 * e_4)$ are continuously processed by means of partition until s and t are contracted (as in $G - e_1 * e_4 * e_3 * e_2$) or disconnected. Finally, $\text{Rel}(G)$ is expressed as $\text{Rel}(G) = p_1p_2 + p_1q_2p_3p_5 + q_1p_4p_3p_2$. Compared to the PVPR algorithm, the CVPR algorithm makes no attempt to locally minimize the numbers of subproblems, though as will be shown, greatly reduces the computation time for the partitioning of each subproblem.

4. EXPERIMENTAL RESULTS

To demonstrate the viability of our algorithms, we implemented the EBRM, PVPR, and CVPR algorithms in the C language and executed these algorithms in Sun ServexStation 5 using a collection of physical network benchmarks [9,13,17–19], as shown in Figure 7. Furthermore, for deriving symbolic TR expressions in the EBRM algorithm, we carried out two versions of implementations: S&P-based [10] and D&G-based [11]. Moreover, VP layouts in the benchmarks were randomly constructed from sparse to dense (by varying the mean VP out-degree) with mean VP lengths near 2.0 in benchmarks 1–9, 3.0 in benchmarks 10–15, and 4.5 in benchmarks 16–18, and a standard deviation of VP length near 1.0.

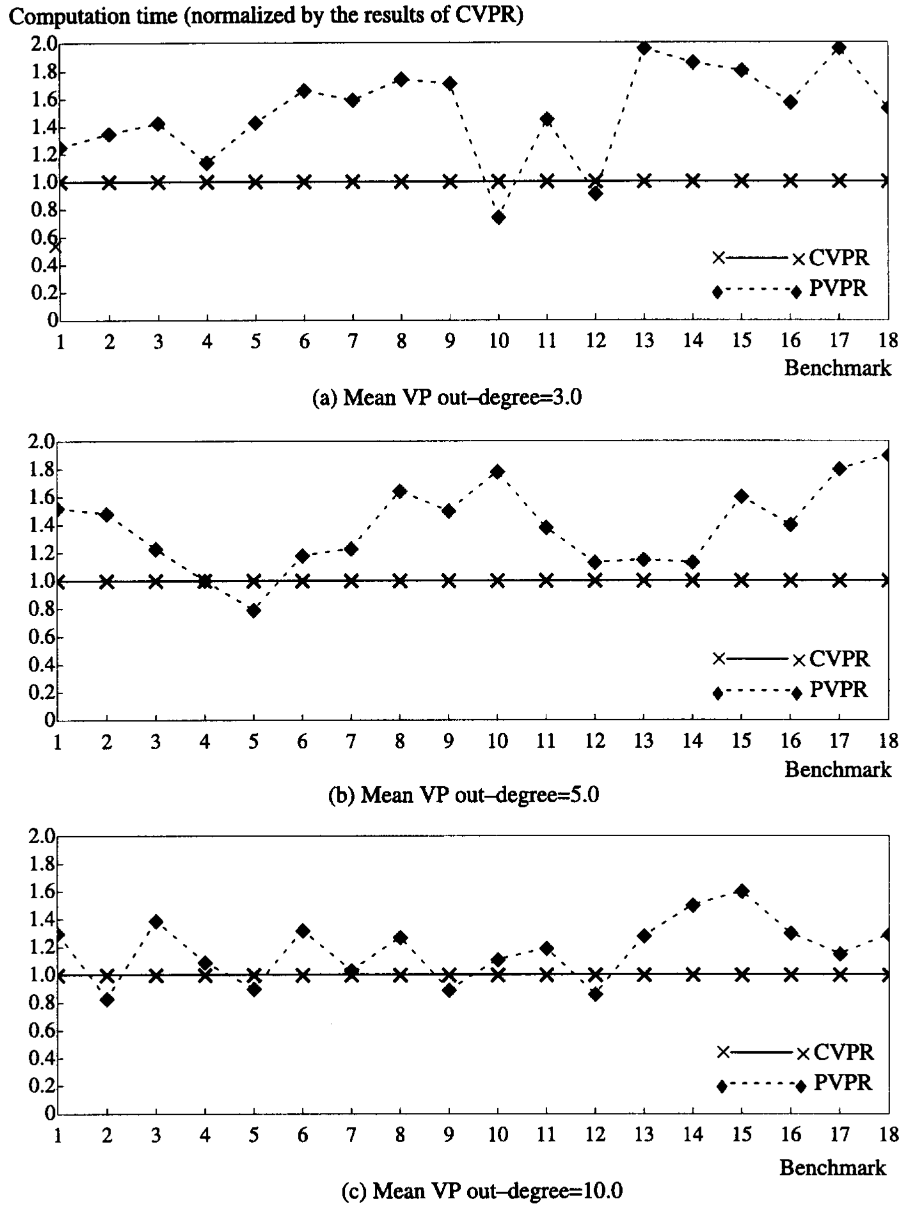


Figure 10. Comparison of the computation time between PVPR and CVPR.

Figure 8 depicts the computation time (on a logarithmic scale) of the S&P-based EBRM, D&G-based EBRM, PVPR, and CVPR algorithms for the benchmarks with different mean VP out-degrees. In the figure, we observe that the PVPR and CVPR algorithms outperform both of the two versions of the EBRM algorithm for all benchmarks. Essentially, the superiority of the PVPR and CVPR algorithms has a pronounced increase with the complexity of the embedded VP layout. It is worth noting that computation by PVPR and CVPR for realistic benchmarks can be achieved in an order of a few minutes rather than days required by both EBRM algorithms. This justifies the practicability of the PVPR and CVPR algorithms for the efficient determination of the robustness of any given VP layout.

We further draw comparisons between the PVPR and CVPR algorithms in terms of the number of subproblems and the computation time in Figures 9 and 10, respectively. As shown in Figure 9, the number of subproblems generated by PVPR is unsurprisingly lower than that generated by CVPR for all the benchmarks. As for the computation time, CVPR outperforms PVPR in most of the benchmarks, as shown in Figure 10.

5. CONCLUSIONS

This paper proposed the PVPR and CVPR algorithms for the efficient computation of TR in ATM VP networks by means of the variants of the path-based and cut-based partition methods, respectively. By partitioning based on the physical links and effectively reducing the number of generated subproblems, the two algorithms yield significantly low computational complexity compared to existing TR algorithms. Experimental results revealed that, compared to S&P-based and D&G-based EBRM algorithms, both the CVPR and PVPR algorithms exhibited superior performance for all the benchmarks. Moreover, the CVPR algorithm was shown to outperform the PVPR algorithm for most of the benchmarks in terms of computation time.

REFERENCES

1. CCITT Recommendation I.150, B-ISDN asynchronous transfer mode functional characteristics, Geneva, (1992).
2. CCITT Recommendation I.311, B-ISDN general network aspects, Geneva, (1992).
3. CCITT Recommendation I.363, B-ISDN ATM Adaptation Layer (AAL) specification, Geneva, (1993).
4. CCITT Recommendation I.361, B-ISDN ATM layer specification, Geneva, (1993).
5. J. Burgin and D. Dorman, Broadband ISDN resource management: The role of virtual paths, *IEEE Commun. Mag.*, 44–48, (September 1991).
6. H. Fujii and N. Yoshikai, Restoration message transfer mechanism and restoration characteristics of double-search self-healing ATM network, *IEEE J. Select. Areas Commun.* **12**, 149–158, (January 1994).
7. R. Kawamura, K. Sato and I. Tokizawa, Self-healing ATM networks based on virtual path concept, *IEEE J. Select. Areas Commun.* **12**, 120–127, (January 1994).
8. Y.G. Chen and M.C. Yuang, A cut-based method for terminal-pair reliability, *IEEE Trans. on Reliability* **45** (3), (September 1996).
9. N. Deo and M. Medidi, Parallel algorithm for terminal-pair reliability, *IEEE Trans. Reliability* **41**, 201–209, (June 1992).
10. A. Satyanarayana and A. Prabhakar, New topological formula and rapid algorithm for reliability analysis of complex networks, *IEEE Trans. Reliability* **27**, 82–100, (June 1978).
11. W.P. Dotson and J.O. Gobien, A new analysis technique for probabilistic graphs, *IEEE Trans. Circuits & Systems* **26**, 855–865, (October 1979).
12. S.N. Pan and J.D. Spragins, Dependent failure reliability models for tactical communications networks, In *Proc. Int. Conf. Commun.*, 1983, pp. 765–771.
13. L.B. Page and J.E. Perry, A model for system reliability with common-cause failures, *IEEE Trans. Reliability* **38**, 406–410, (December 1989).
14. Y.F. Lam and V.O.K. Li, Reliability modeling and analysis of communication networks with dependent failures, *IEEE Trans. Commun.* **34**, 82–84, (January 1986).
15. L.B. Page and J.E. Perry, Reliability of directed networks using the factoring theorem, *IEEE Trans. Reliability* **38**, 556–562, (December 1989).
16. S. Rai, A. Kumar and E.V. Prasad, Computer terminal reliability of computer network, *Reliability Engineering* **16**, 109–119, (1986).
17. S. Soh and S. Rai, CAREL: Computer aided reliability evaluator for distributed computing networks, *IEEE Trans. Parallel & Distributed Systems* **2**, 199–213, (April 1991).
18. D. Torrieri, An efficient algorithm for the calculation of node-pair reliability, In *Proc. IEEE MILCOM '91*, November 1991, pp. 187–192.
19. D. Torrieri, Calculation of node-pair reliability in large networks with unreliable nodes, *IEEE Trans. Reliability* **43**, 375–377,382, (September 1994).
20. M.O. Ball, Computational complexity of network reliability analysis: An overview, *IEEE Trans. Reliability* **R35**, 230–239, (August 1986).
21. E.W. Dijkstra, A note on two problems in connexion with graphs, *Numerische Mathematik*, 269–271, (1959).