An Integrated Failure and QoS Recoverable Transmission Scheme Based on Multi-path Network Coding

CHUNG-SHIH TANG AND WEN-TSUEN CHEN
Telecommunication Laboratories
Chunghwa Telecom Co. Ltds.
Taoyuan, 326 Taiwan

Department of Computer Science
National Tsing Hua University
Hsinchu, 300 Taiwan

Availability and QoS are two of the most important service requirements to end-to-end network transmission. Frequently, separate resource control schemes are adopted to meet network availability and QoS objectives, respectively. In this paper, we propose a novel multi-path transmission scheme that employs network coding technique to benefit both on QoS and availability. With the design of multiple disjoint paths from source to destination, where each path carries encoded data stream and the number of paths is greater than the number of original data streams, it is illustrated that not only the resilience to network failure can be achieved but also the end-to-end QoS is improved. The proposed scheme can be properly designed to meet the most stringent QoS and availability requirements while efficiently utilizing the network resource.

Keywords: network availability, quality of service, failure recovery, network coding, disjoint multi-paths

1. INTRODUCTION

Network availability and quality of service (QoS) are two of the most important requirements to end-to-end network transmission, particularly for incumbent telecommunication operators. Network availability has always been deemed as a key index of service level agreement (SLA) either between customers and the service provider or between two providers. Numerous works focus on designing redundant systems to refrain from node failure and constructing failure recovery mechanisms to diminish the system downtime and such that the service impact is minimized. Meanwhile, QoS attracted tremendous attentions even before the Internet Protocol (IP) dominated the next generation networks. A myriad of studies investigated the QoS issues on characterizing QoS metrics and objectives, developing QoS control schemes to achieve the target objectives, and evaluating the performance of QoS schemes, etc.

As a usual manner, failure recovery and QoS control are separately concerned. Some measures are taken to assure high availability and the others are used to promote or guarantee end-to-end QoS. However, at least the following drawbacks are noticed if these two categories of control schemes are not jointly considered. First of all, network resource may be over-utilized. Since both QoS control and failure recovery need to reserve...
network resource, if not properly coordinated it is liable to assign resource more than necessary to the same application at the same node. For example, to prevent data loss from link failure each node may reserve certain output buffer for retransmission. Apart from the reserved buffer space by failure recovery scheme, to guarantee smooth transmission, QoS control scheme may also reserve extra space that is in fact unnecessary. Second, the effect of QoS control may contradict or disagree with the failure recovery scheme. A typical example is that QoS control finds a shortest path for some application and that path ruins the possibility of finding disjoint alternative path for path recovery against network faults. As shown in Fig. 1, the shortest path from S to D is $S \rightarrow 1 \rightarrow 2 \rightarrow D$, which spoils the disjoint paths $S \rightarrow 3 \rightarrow 4 \rightarrow 2 \rightarrow D$ and $S \rightarrow 1 \rightarrow 5 \rightarrow 6 \rightarrow D$ required by failure recovery scheme of path protection. Third, double efforts and complexities are consumed. If QoS control and failure recovery can be integrated into one scheme, it is intuitively anticipated that the management efforts, as well as the overall time and space complexity, can be reduced.

![Fig. 1. The shortest path disables possible disjoint paths.](image)

Therefore, in this paper we propose an integrated transmission scheme attempting to simultaneously promote service availability and improve end-to-end QoS. The scheme, based on network-coding the ready-to-send data and transmitting over multiple disjoint paths, results in better tolerance to data loss and higher resilience to network faults by designing the number of data before encoding, $M$, less than the number of encoded data, $N = M + K$, where $N$ is also the number of disjoint paths embedded with $K$ redundancy. The sink, upon receiving any $M$ encoded data out of the $N$ data sent by source, can recover the $M$ original data simply by network decoding, thus enduring at least $K$ concurrent network faults and in the same way recovering packet error or packet loss bursts.

The rest of this paper is organized as follows. In the next section, some related works are reviewed. Then a packet-recoverable transmission scheme based on network coding is proposed and followed by the evaluation of QoS improvement. Next, the heuristic algorithms of finding disjoint multi-paths are presented and their efficiencies are explored. Finally, a conclusion is drawn and some future studies are introduced.

2. RELATED WORKS

Essentially, network availability is determined by the frequency of network faults and the duration that each fault lasts. The former is recognized as mean time between failures (MTBF) and the latter as mean time to repair (MTTR). Usually redundant designs of equipment like backup module or cluster architecture are used to enhance sys-
Failure and QoS Recovery using Network Coding

System reliability, or equivalently, enlarge MTBF. Failure recovery, on the other hand, aims to reduce the duration of outages, no matter caused by node failure or link down. Failure recovery schemes can be categorized into protection model and restoration model and network faults are recovered in global-path, local-link, or failure-to-egress basis [1]. RFC 3469 [2] also defines a framework for MPLS-based recovery where re-routing and protection switching are distinguished and components of time to recover from a network fault are analyzed. Various failure recovery schemes are proposed and among them path protection is usually recommended because of its bandwidth efficiency and tractability [3, 4].

IETF has spent lots of effort on proposing QoS control frameworks for IP networks. The Integrated Services (IntServ) [5] and Differentiated Services (DiffServ) [6, 7] are two representatives. The former relies on the per-flow control that is not scalable in a large network. Thus, the latter emerges with greater granularity that bases on per hop behavior (PHB). Nevertheless, regardless of Controlled-Load services [8] and Guaranteed services [9] in IntServ, or Assured Forwarding (AF) PHB [10] and Expedited Forwarding (EF) PHB [11] in DiffServ, there needs some QoS control schemes like traffic classification, buffer management, packet scheduling, traffic shaping or policing, to realize corresponding requirements. Those schemes are traffic dependent and act as a reactive control manner. Much more studies [12, 13] develop new QoS control schemes or control protocols to fulfill or improve the QoS required in the IntServ or DiffServ service architectures.

The other category of QoS control schemes focus on a more preventive perspective. Multi-protocol label switching (MPLS) [14], integrating the label swapping forwarding paradigm with network layer routing, proposed another perspective of QoS control through traffic engineering [15]. Following MPLS traffic engineering, a series of works utilize multiple paths to avoid network congestion and to satisfy the end-to-end QoS needs [16-18]. Another branch of preventive QoS control is based on forward error correction (FEC). FEC enabled transmission schemes also provide resilient capability to network error or loss. [19-21] show only the iceberg of FEC-based control schemes. A digital-fountain based protocol [22] further applies the FEC concept on reliable transmission of bulk data. However, the collaboration of multi-path traffic engineering and FEC-like redundancy to improve QoS and failure resilience seems not yet considerably investigated. In this paper a QoS control scheme, which integrates the concept of multi-path load sharing and the capability of FEC-like recovery, is proposed. To acquire the robustness of recovery without interfering current network operation, the overlay linear network coding is adopted as the underlying transmission technique.

Network coding, first introduced to maximize the network information flow [23], extends diverse applications on many fields [24]. Typical ones include content distribution [25], wireless network throughput improvement [26], network monitoring [27, 28], network security [29], and network failure recovery [30, 31], etc. The features of linear network coding, including its robustness on data transmission and limited time complexity, are extracted to be the key ingredient of the proposed scheme. Linear network coding is based on the concept of solving systems of linear equations. A set of linear equations with n variables $x_1, x_2, \ldots, x_n$ can be represented by the matrix operation as follows.

\[
\begin{align*}
\begin{bmatrix}
\end{align*}
\end{align*}
\]
or equivalently, $Ax = b$, $A = (c_{ij})$, $x = (x_i)$, and $b = (b_i)$.

If matrix $A$ is nonsingular, $x = A^{-1}b$ is the solution vector to the system of linear equations. In terms of linear network coding, $x_1, x_2, \ldots, x_n$ are the original data before encoding, with coding coefficients $c_{11}, c_{22}, \ldots, c_{nn}$; encoded data $b_i$'s are generated using linear combination of $x_1, x_2, \ldots, x_n$. That means if the matrix of coefficients, $A$, is properly selected, those original data can be decoded after receiving $n$ encoded data $b_1$, $b_2$, ..., and $b_n$ by the decoding operation of $A^{-1}b$.

3. IFQRT: INTEGRATED FAILURE AND QOS-RECOVERABLE TRANSMISSION

Understanding that for most failure recovery schemes, the reserved extra network resource is never used on normal transmission and does not contribute to QoS improvement, this paper attempts to propose an integrated failure and QoS-recoverable transmission (iFQRT) scheme. iFQRT treats data error, loss and network faults indifferently and tries to reduce the impact of QoS degradation and network failure as far as possible.

3.1 Design Idea

The basic idea of iFQRT is to use multiple paths to convey traffic from its source to destination where some extra paths and bandwidth are redundant and used for packet recovery. At the source node $s$, $M$ packets are encoded into $M + K$ packets using linear network coding, tagged with a common sequence number for correlation, and then sent to destination node over the pre-established $M + K$ parallel paths. At the destination node $d$, packets with the same sequence number are decoded together to get original packets. When receiving packets, the destination node will check packet content using parity or CRC to identify the erred packets, and determine a packet loss if the pre-set timer expires. A general model of $(M + K)$-iFQRT is demonstrated as in Fig. 2.

The design of $(M + K)$-iFQRT implicitly features the following capabilities:

(a) Sharing of traffic load over the $M + K$ parallel paths;
(b) Tolerance to at least $K$ concurrent network faults, including node failure and link down;
(c) Recovery from packet loss or packet error within simultaneous $K$ loss or error events;
(d) Latency reduction by parallel data transmission.

3.2 Detailed Algorithm

The network topology is represented by $G(V, E)$, where $V$ is the set of nodes and $E$ is the set of bidirectional links. For a connection request whose (source, destination) = ($s$, $d$),
FAILURE AND QoS RECOVERY USING NETWORK CODING

$t), s, t \in V$, and demanding bandwidth $= B$, the iFQRT is mainly composed of the following three steps:

**Step 1:** Find $M + K$ disjoint paths from $s$ to $t$.

Those paths can be determined based on Dijkstra’s shortest path algorithm or any other traffic engineering routing algorithms. Although parallel paths with overlapped portions may still work for iFQRT, disjoint paths work better by their capability to tolerate higher possibility of network faults and packet loss/error bursts. Therefore, heuristics to find disjoint multi-paths are investigated in the next section.

**Step 2:** On source node $s$,

1. Divide the demanding bandwidth $B$ into $M$ equal parts, each part equals to $B/M$.
2. Whenever transmission, encode $M$ packets into $M + K$ coded packets using random linear network coding as follows:

   (1) For each one of the $M + K$ paths, select $M$ random coefficients of $\text{GF}(2^z)$ forming a coding vector. That is, a set of random coefficients $c_{i1}, c_{i2}, \ldots, c_{iM}$, each with length of $z$ bits, are generated for the $i$th path. $\text{GF}(2^z)$ denotes the Galois Fields of $2^z$ finite elements, representing all possible strings of $z$ bits. Addition in the GF is the same as bit-wise exclusive-or. Multiplication, on the other hand, has to regard two elements in GF as polynomials with maximum degree of $z - 1$, divide the product of these two polynomials by an irreducible polynomial with degree of $z$, and find the remainder. More description of GF can refer to [32].

---

**Fig. 2. The generic model for $(M + K)$-iFQRT transmission.**

Examples:
- Generate packets $Q_i$'s at node $s$ by linearly combining $P_1, P_2, \ldots, P_M$.
- $M + K$ disjoint paths from node $s$ to node $t$ thus allowing $K$ concurrent packet error/loss or network faults.
- Receive at least $M$ packets $Q_i$'s at node $t$ by solving $M$ linear equations.
(2) Generate the coded packet $Q_i = c_{i1} \times P_1 + c_{i2} \times P_2 + \ldots + c_{iM} \times P_M$ for $i = 1, \ldots, M + K$.

2.3 Tag a common sequence number in front of each packet and parity check bytes at the end of each packet. The latter can be omitted if layer 2 protocol performs packet check.

2.4 Send the $M + K$ coded packets over the corresponding $M + K$ paths.

**Step 3:** On destination node $t$,

3.1 The destination node is aware of the set of coding coefficients used by the source node in advance, either designated manually or piggybacked in the protocol of request admission control. Since random linear network coding needs to generate pseudo random number (PRN) for the coding coefficients on the source node, it is applicable to configure common seeds for the same PRN generators on each source-destination node pair. Then the source node and destination node will not need to negotiate or carry the coding coefficients and reduce the extra overhead and bandwidth consumption.

3.2 Thereafter the destination node, if receiving $M$ or more coded packets of the same sequence number, can recover the $M$ original packets by solving the set of linear equations formulated from the coded packets. For each received coded packet, its corresponding coding vector is put in last row of the so-called decoding matrix if the received packet is innovative in the sense that the inserted coding vector increases the rank of the decoding matrix. Then on receiving $M$ innovative packets $Q_1, Q_2, \ldots, Q_M$, the destination node can decode into original $P_i$’s by matrix multiplication of the inverse of the decoding matrix and the vector of $Q_i$’s.

It is noted that unsuccessful decoding will result in $M$ packet losses since all $M$ packets can not be recovered. However, if not for real time application, the destination node can notify the source node to retransmit the missing packets in original form without coding. As to the real time application, it is suggested to send $M$ original packets and $K$ encoded packets. In this way if the destination node receives not enough packets, the packet loss can be reduced as much as possible and service impact can be minimized. Through the negotiation of coding coefficients, the destination node can be aware of what received packets are original and what are encoded.

3.3 QoS Improvement by iFQRT

As mentioned, iFQRT contributes to network availability and QoS at the same time. The promotion on network availability is measured by the service downtime and has been comprehensively investigated in previous paper [31]. Where for various kinds of network topologies, it has been shown that reduction of service downtime ranged from 40% to 95% can be achieved if applying iFQRT scheme under MTTR of 0.05 hour. And if MTTR is controlled in 0.01 hour by integrating L3 re-routing to find alternative path for the failed one, two orders of improvement can even be possible. Hereafter, we simply focus on QoS improvement in terms of three most popular performance metrics in IP network, packet error, packet loss and packet transfer delay. First of all, let us consider the improvement of packet error performance that iFQRT may achieve.
3.3.1 Packet error ratio

Let $p_b$ denote the end-to-end bit error rate (BER) and $p_p(n)$ denotes the probability that a packet of length $n$ bits is an error packet. A packet is an error packet if it contains at least one bit error, this implies $p_p(n) = 1 - (1 - p_b)^n \approx n \cdot p_b$. Assume all packets traverse routes of the same hops and encounter similar end-to-end bit error condition. Then the end-to-end packet error ratio (PER) for a single path will be a function of BER, that is, $PER = (\text{Number of Erred Packets})/(\text{Number of Transmitted Packets}) = p_p(n) \approx n \cdot p_b$. However, for the proposed $(M + K)$-iFQRT transmission scheme, an erred packet will be received by the destination node only when more than $K$ packet errors happen over the $K$ disjoint routes simultaneously. That is,

$$PER(M, K) = \sum_{i=K+1}^{M+K} \binom{M+K}{i} \cdot [p_p(n)]^i \cdot [1 - p_p(n)]^{M+K-i}.$$

![Fig. 3. PER improvement for (a) and (b): PER vs. Link BER; (c) and (d): PER vs. packet length.](image)

Fig. 3 shows that $PER$ will increase in accordance with end-to-end BER. It is noticed that iFQRT approaches PER when the link condition turns worse. But under normal link conditions ($BER < 10^{-7}$) iFQRT significantly improves the PER performance at least of two orders, even for large packet length as shown in Fig. 3 (c). It is also found that $(2 + 1)$-iFQRT and $(3 + 1)$-iFQRT perform without discernible difference. That means applying iFQRT either with more disjoint paths $(3 + 1)$ or with higher bandwidth $(2 + 1)$ is pertinent depending on network status. The performance of iFQRT effectively depends
on the ratio of fault tolerance. 2 + 1 case has 1/3 of resource allocated for redundancy while 3 + 1 case only has 1/4 of resource for redundancy. Although 3 + 1 case requires more disjoint paths, it consumes less redundant bandwidth and thus performs less significantly than 2 + 1 case.

3.3.2 Packet loss ratio and packet transfer delay

The analysis of improvement by iFQRT on the packet loss ratio (PLR) is more complicated than on PER. Packet loss mainly results from buffer overflow or explicit node policy to drop packets under certain conditions. While packet loss due to failure of error checking has been addressed in PER analysis, the target of packet loss analysis here will be based solely on buffer overflow.

In this section, the performance of general single-path (SP) transmission is compared with the proposed (2 + 1)-iFQRT transmission scheme. Since packet loss arisen from buffer overflow usually occurs when multiple inputs contend a common output link. For the case of SP transmission, to make link contention possible, at least two nodes connecting to a common node are required. But as previously stated, (2 + 1)-iFQRT needs three disjoint paths. Without loss of generality, let’s consider the network topology shown in Fig. 4. There are six sources and six sinks that communicate through three disjoint bottleneck links (7 → 7′), (8 → 8′) and (9 → 9′). Assume all links have the same capacity of 100 Mbps. Each node pair of (1 → 1′), (2 → 2′), …, (6 → 6′) sends packets in a burst-idle model, where within a burst in average $B$ packets of length $L$ bytes are transmitted, and then followed by an idle period, thus resulting in bandwidth consumption of $W$ Mbps. Packets incoming to a node are queued in the fixed-size output buffers of their corresponding next hop interfaces. For clarity, Figs. 4 (a) and (b) only depicts the traffic flows of two requests, (1 → 1′) and (2 → 2′), based on SP and iFQRT schemes, respectively.

Simulations are conducted based on the fixed parameters and variables as specified on Table 1. The offered load is controlled by the bandwidth $W$ of each node-pair request. The traffic model of each request is assumed to be an on-off bursty model. The burst period and idle period are exponentially distributed with mean of $P_{on}$ and $P_{off}$, respectively.
Table 1. Parameters and variables for simulation.

<table>
<thead>
<tr>
<th>Fixed parameters</th>
<th>Controlled variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link Propagation Delay (sec)</td>
<td>Request Bandwidth (Mbps)</td>
</tr>
<tr>
<td>Link Capacity (LC) 100 Mbps</td>
<td>Average Burst Size (B) Packets</td>
</tr>
<tr>
<td>Output Buffer Size 8K Bytes</td>
<td>Packet Length (L) Bytes</td>
</tr>
</tbody>
</table>

Assume $LC$ denotes link capacity and there are in average $B$ packets in a burst, each with fixed packet length of $L$ bytes, then $P_{on}$ is $B \times L \times 8/LC$ seconds. And for $W = LC \times P_{on} / (P_{on} + P_{off})$, $P_{off}$ can be calculated from $P_{on} \times (LC/W - 1)$, thus $P_{off} = B \times L \times 8 \times (1/W - 1/LC)$.

For each simulation run, packet loss ratio (PLR) and average packet transfer delay (PTD) are measured. Fig. 5 shows the PLR and PTD over increasing offered load. It is found that no matter the request bandwidth is, iFQRT can achieve 20 to 50 times of improvement in PLR. It is also found iFQRT gains 12% ~ 20% advantage in PTD over SP, and most importantly, iFQRT maintains PTD within a limited range regardless of the offered load. Fig. 6 explores the impact on PLR and PTD that enlarged mean burst size may bring about. The larger the packet burst size is, the higher probability of contention will be and packets are liable to accumulate at output buffer. However, noticed from Fig. 6 (a), PLR tends to stabilize after mean burst size of 12 packets. Even so, iFQRT still 5 times outperforms SP and reduces average PTD up to 30%, as depicted in Fig. 6 (b).
Figs. 7 and 8 investigate how packet length can influence the PLR and PTD. There are two scenarios considered here. First, during the burst period, the average number of bytes in a burst is fixed. Then larger packet length leads to fewer packets in a burst. Second, the average number of packets is fixed regardless of the packet length. Therefore, larger packet length implies larger volume of bytes being transmitted.

For scenario 1 as depicted in Fig. 7 (a), the packet length, though with negligible impact on PLR for SP, influences iFQRT performance quite a bit. The larger packet length leads to higher PLR for iFQRT because $(2 + 1)$-iFQRT contains at most three input streams to the same output buffer. When the packet length approaches 4K bytes, half size of the output buffer, concurrent packet arrivals of three streams tend to overflow the buffer. As a result, PLR for iFQRT at high packet length will approach that for SP.

For scenario 2, the larger packet length somewhat implies higher burstiness, that is, with longer burst period and idle period. It is even clearer when comparing the cases with mean burst size of 8 packets and 16 packets, as shown in Fig. 8. The PLR of 8-packet burst size with packet length $L$ approximately equals to that of 16-packet burst size with packet length $L/2$. As anticipated, burstiness worsens the performance, but no matter how large the burstiness is, iFQRT noticeably improves SP both in PLR and PTD.

4. FINDING DISJOINT MULTI-PATHS

Disjoint multi-paths are necessary under many circumstances. For example, equal
cost multiple path (ECMP) is used in Open Shortest Path First (OSPF) routing for load balancing. Path protection or restoration also needs disjoint multi-paths to recover from network failure. Furthermore, the well-known MPLS traffic engineering usually setup alternative routes to the shortest path for off-loading traffic from the congested “hot spot.” Consequently, some heuristics for finding disjoint multi-paths are suggested.

4.1 Heuristic Algorithm 1: Baseline

The most intuitive and basic heuristic algorithm to find disjoint multiple paths is based on shortest-path first (SPF) algorithm as follows. Here it is presented as a baseline to compare with other schemes.

**Step 1:** Find the shortest path from source to destination.
**Step 2:** Remove all links (or nodes) on the shortest path of step 1, find the shortest path again on the remaining graph.
**Step 3:** Repeat the above step until all link- (or node-) disjoint paths are found.

![Red paths based on SPF routing](a) ![Blue paths based on optimal routing](b)

Fig. 9. (a) Link-disjoint paths from node 3 to node 7; (b) Node-disjoint paths from node 7 to node 0.

Obviously, the baseline heuristic is not an optimal solution. Consider the network topology illustrated in Fig. 9. Based on SPF routing, two link-disjoint paths \( r_1 \) and \( r_2 \) can be found from node 3 to node 7 in Fig. 9 (a). However, three link-disjoint paths exist \( b_1, b_2 \) and \( b_3 \). Fig. 9 (b) shows a similar condition in finding node-disjoint paths from node 7 to node 0. Again, SPF routing finds \( r_1 \) and \( r_2 \) while three node-disjoint paths are available using optimal routing. Therefore, to improve the possibility of finding enough disjoint multi-paths for iFQRT, the algorithm in step 1 of iFQRT can be revised as follows:

**Step 1.1:** Use Baseline algorithm to find \( M + K \) link- (or node-) disjoint path, if not enough disjoint paths can be found, go to the next step.
**Step 1.2:** Use the heuristic algorithm presented in the next section to find \( M + K \) disjoint paths.
Step 1.3: If not enough disjoint paths can be found, normal (single path) data transmission instead of iFQRT is adopted.

4.2 Heuristic Algorithm 2: \textit{L} out of \textit{K}-Shortest Path (\textit{LooK})

This heuristics is composed of two basic algorithms. The first is the \textit{K}-shortest-path (KSP) algorithm which is intensively investigated over past literatures [33-35]. The second is an algorithm to find link-disjoint or node-disjoint paths from the \textit{K} shortest paths. The problem of finding link- or node-disjoint paths is transformed to the well-known coloring problem as follows.

\textbf{Step 1:} Given the \textit{K} shortest paths over a graph \(G(V, E)\), where \(V\) is the set of network nodes, \(i \in V, \ i = 1, 2, ..., |V|\), and \(E\) is the set of bi-directional links between nodes, \(l_{ij} \in E\) if \(i\) and \(j\) are neighbors.

\textbf{Step 2:} Define the Link-Traversing Path Set (L-TPS) or Node-Traversing Path Set (N-TPS).

\begin{enumerate}
\item The L-TPS\((l_{ij})\) consists of those paths that will traverse over link \(l_{ij} \in E\), or 
\item The N-TPS\((i)\) consists of those paths that will traverse through node \(i \in V\).
\end{enumerate}

\textbf{Step 3:} Construct a new graph where the nodes in the graph represent the \textit{K} shortest paths in step 1 and any two nodes are connected if their corresponding paths are not disjoint. That is, for the disjoint-path graph \(G'(V', E')\), \(V'\) is the set of \textit{K} shortest paths, \(e_{uv} \in E'\) for \(u, v \in V'\) if

\begin{enumerate}
\item \(\{u, v\} \subseteq \) any one of L-TPSs for finding link-disjoint paths;
\item \(\{u, v\} \subseteq \) any one of N-TPSs for finding node-disjoint paths.
\end{enumerate}

\textbf{Step 4:} Find the maximal independent set based on the new graph \(G'(V', E')\).

The nodes in the graph are put on colors under the constraint that two adjacent nodes should not use the same color. Then the target is to find the maximum number of nodes in the same color, \textit{i.e.}, the maximum independent set. However, the problem of finding maximum independent set is NP-complete. It is reasonable to find the maximal independent set instead in the following steps.

\begin{enumerate}
\item Initially the maximal independent set \(S_{mI} = \emptyset\).
\item Nodes are sorted in the order of ascending degree, that is, the nodes \(p_1, p_2, ..., p_K\) in sequence, \(\text{Deg}(p_i) \leq \text{Deg}(p_j)\) for \(i < j\).
\item Move the first node into the maximal independent set and remove all its adjacent nodes from the sorted list, that is, \(S_{mI} = S_{mI} + \{p_1\}\) and remove \(p_i\) if \(e_{p_ip_j} \in E'\).
\item Repeat the previous step until all nodes in the list are removed.
\end{enumerate}

4.3 Simulation Results

To evaluate the effectiveness of the proposed heuristic algorithms, the number of disjoint paths that baseline algorithm can find and the ratio of \textit{LooK} outperforming baseline are simulated over massive random networks. Yet to assure network connectivity, a
basic ring topology is assumed and links are then randomly assigned between any two possible nodes. For fully-meshed requests between any two nodes in the random network, baseline heuristic is applied to find if there exists three disjoint paths that \((2 + 1)\)-iFQRT requires. The cases that the source node or destination node of a request only has two adjacent links are identified and excluded from further analysis, and such a request is named an invalid one since not enough disjoint paths can be established. The ratio of invalid requests over all possible ones among 5000 randomly generated networks is shown in Fig. 10 (a), which is concerned because it represents the possibility that iFQRT cannot be applied. It is obvious the networks with smaller average degree will encounter more invalid requests. However for average degree of four, the ratio of invalid requests will never exceed 25% and can be further reduced through considerate network planning.

![Fig. 10. (a) Ratio of invalid requests; (b) Unsuccessful routing ratio for baseline heuristic.](image)

It is noticed that most cases of unsuccessful routing, i.e., the number of disjoint paths less than three, come from the invalid network topology. With the invalid cases excluded, only very small percentage of requests fail to find enough disjoint paths using baseline heuristic, as shown in Fig. 10 (b). Only the ratio of baseline failure on finding node-disjoint (NDJ) paths for average degree of four goes beyond 2%, the others all below 0.2%. For the cases of baseline failure, the \textit{LooK} heuristic improves about 20% to 80%, more significantly on finding node-disjoint paths.

From the simulation results, it can be summarized that baseline heuristics accompanied with \textit{LooK} algorithm works on most situation. This implies the time complexity is efficiently restricted that makes iFQRT applicable and practical.

### 5. CONCLUSIONS

This paper proposed an integrated failure and QoS-recoverable transmission scheme by utilizing overlay network coding and multi-path load sharing. In addition to the failure recovery capability, the embedded network coding features enable iFQRT to recover from packet loss and packet error during transmission in the network. Not only significantly improving the \textit{PER} and \textit{PLR} performance, iFQRT also shortens the end-to-end packet transfer delay by its parallel transmission paradigm. Through the verification of
heuristic algorithms for finding multiple disjoint paths, iFQRT is proven a practical and successful failure and QoS recoverable transmission scheme, over which a managed IP network can be manipulated in a high-availability, QoS-controllable and securely-transmitted manner. The application of iFQRT on confidential communication will be one of our interested future studies. Besides, the effectiveness that allows intermediate nodes to involve the interaction to iFQRT will be further explored in the future.

REFERENCES

17. K. Long, Z. Zhang, and S. Cheng, “Load balancing algorithms in MPLS traffic en-

Chung-Shih Tang (唐崇實) received his B.S. and M.S. degrees in Industrial Engineering from National Tsing Hua University, Hsinchu, Taiwan, in 1988 and 1990, respectively. He has joined the researches in Telecommunication Laboratories (TL), MOTC since 1990. Working for TL through the privatization of Chunghwa Telecom, he is currently a Researcher of Telecommunication Laboratories, Chunghwa Telecom, Co. Ltd. His research interests include network failure recovery, network coding, VoIP networks, IP multimedia subsystem, and next generation networks.

Wen-Tsuen Chen (陳文村) received his B.S. degree in Nuclear Engineering from National Tsing Hua University, Taiwan in 1970, and M.S. and Ph.D. degrees in Electrical Engineering and Computer Sciences from University of California at Berkeley in 1973 and 1976 respectively. He has been a professor of the Department of Computer Science, National Tsing Hua University, Taiwan, since 1976. He is currently a Distinguished Chair Professor and the President of the University. His research interests include wireless Internet, broadband networks, parallel systems and algorithms.