Correlation-Resilient Path Selection in Multi-Path Routing*

Xin Zhang and Adrian Perrig
Carnegie Mellon University

Abstract

Multi-path routing is effective to enhance network availability, by selecting multiple *failure-independent* paths for reaching one destination in the hope to survive individual path failures. Researchers suggest to select IP-layer topologically disjoint paths, assuming that they are failure-independent and can hardly fail simultaneously. Unfortunately, failure correlations lurking behind the IP-layer topology can surreptitiously squash availability gained through multi-path routing because selected paths can fail simultaneously. Spurred by this observation, we propose a new path metric and selection scheme resilient to failure correlations between topologically disjoint paths, by utilizing path *availability history* to reveal failure correlations. This paper presents a first stride towards the new direction of availability-oriented multi-path selection, with formal and systematic problem definition, modeling, and algorithms.

1 Introduction

In recent years, *multi-path routing* has been popularly advocated amongst the research community [8, 11, 14, 15, 18–20]. In multi-path routing, each router can use multiple different paths for reaching one destination prefix, with the goal to enhance end-to-end availability. Specifically, when one of the paths fails, packets can still be delivered via other working paths and thereby maintaining end-to-end availability, *as long as not all paths between the source and destination fail concurrently*. We term such end-to-end availability provided by multiple paths between a source-destination pair as *multi-path availability*. One crucial component in multi-path routing is *multi-path selection*: the decision process of determining which paths to use. Obviously, the selection tactic and resulting path qualities will exercise direct influence on the effectiveness of multi-path routing. While respecting local policies [5] (if any) during the multi-path selection process, in this paper we focus on optimizing multi-path availability, which is the underlying requirement of multi-path routing.

As a new subject, multi-path selection has drawn little attention in the literature so far, despite its importance. Previous proposals on multi-path routing primarily focused on routing infrastructure design. Early investigations on path selection [3, 9, 13] were mostly in the context of conventional single-path routing (where each router is limited to using only a single "best path" for one destination prefix). Wendlandt et al. explicitly recognize multi-path selection as a vital component in multi-path routing in the context of defending against BGP attacks [18]. As a simple approach to handling availability-oriented multi-path selection, it was suggested to choose the most "IP-layer topologically disjoint" paths, in an attempt to minimize the probability that all the paths simultaneously fail [11,20]. While such an approach is simple and intuitive, it relies on the assumption that *IP-layer topologically disjoint paths are indeed failure-independent*. Regretably, this assumption has been suspected repeatedly in the literature [2, 7, 10, 16]. As we investigate in Section 2.2, IP-layer topologically disjoint paths can still be failure-correlated, due to the discrepancies between IP-layer and Physical-layer topologies, i.e., IP-layer disjoint paths can still share the same physical elements. Furthermore, routing-level congestion or common software vulnerabilities can also give rise to failure correlation between even physical-layer disjoint paths.

Motivated by the presence of failure correlation between topologically disjoint paths, we strive to tackle multi-path selection with resilience to such failure correlation. The key idea is to base multi-path selection on the knowledge of paths' *availability history*, in light of that failure correlation between paths can be automatically derived from their availability history. Specifically, if two paths tend to present concurrent failures in the history, we regard them as failure-correlated, otherwise failure-independent. Admitted that two failure-independent paths may also present certain concurrent failures by chance, such a *random correlation* has a rather negligible probability to occur, observing that in the real-world link failures are rare [7]. On the other hand, by using the historical failure correlation to predict future ones, our scheme is most resilient to the types of failures that can repeat in the future, while the effectiveness of our scheme will be hampered in circumstances where failures happen only once. Our underlying rationale for leveraging availability history to exploit failure-correlation lies in that: given the intricate causes to failure-correlation between different paths (Section 2.2), the best we can do is to derive such correlation *after* the failures take place, while it is an open challenge to detect failure-correlation *before* failures happen.

In this paper, we present a first step to study this important topic. Our contributions are four-fold:

Problem Definition. This paper raises the awareness of the multi-path selection problem, and formally presents a problem statement.

Metric. We derive a new metric from path availability history to reflect the failure correlation between disjoint paths.

*This research was supported in part by CyLab at Carnegie Mellon under grants DAAD19-02-1-0389 and MURI W 911 NF 0710287 from the Army Research Office, and grant CNS-0627357 from the National Science Foundation. The views and conclusions contained here are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of ARO, CMU, NSF, or the U.S. Government or any of its agencies.

This metric enables more accurate estimation of multi-path availability.

Problem Modeling. We mathematically model multi-path selection as an Integer Programming problem. This precise model facilitates us to better understand the problem, and also enables us to use various well-studied algorithms for Integer Programming.

Algorithm. For the ease of practical deployment, we also suggest a simple heuristic and prove its effectiveness via theoretical analysis.

The remainder of the paper is organized as follows. In Section 2, we introduce the problem statement of multi-path selection and the key observation which motivates our endeavors. Section 3 sketches the high-level picture of our scheme. Then, Sections 4 and 5 elaborate the mathematical modeling and algorithms of the problem, respectively. Finally, Section 7 concludes the paper and presents future work.

2 Motivation and Related Work

2.1 Problem Statement

For multi-path routing to be effective, it is desired to select paths that can provide the least probability that all the selected paths fail simultaneously, which in turn yields the highest multi-path availability. To this end, we focus on selecting a set of *failure-independent* paths. To sum up, the multi-path selection problem can be stated as follows:

A node Z has n candidate paths to reach destination D. The problem is to find the k most failure-independent paths such that the resulting multi-path availability is maximum, where the value of k is restricted by the communication overhead allowed in the routing infrastructure.

2.2 Why is Disjointness Inaccurate?

In light of the problem statement above, it is clear that an accurate metric for evaluating failure-correlation and multi-path availability forms the prerequisite of a successful multi-path selection scheme. Several proposals suggest to select "disjoint" paths in the IP-layer topology to minimize failure correlation [11, 20] between selected paths. The intuition is that paths overlapping at certain links must suffer from failure correlation introduced by the shared links. Though such an approach is intuitive and simple, its effectiveness in practice is nevertheless hampered. It has been observed that certain subsets of IP-layer links can be failure-correlated even if they are topologically disjoint, and the popularity of such a correlation is not negligible [2, 7, 10]. Therefore, paths can be failure-correlated even when their constitute links are all disjoint in the IP-layer topology.

An important source of causes to such failure correlation has been well recognized as the Shared Risk Link Groups (SRLG) [10, 16]: multiple IP-layer links that are associated with the same physical network element (e.g., fiber span, optical amplifier etc.) will experience failure at the same time when the physical object fails. Hence, the topological disjointness-based selection scheme intrinsically embeds the inaccuracy of using the *logical* network topology to conclude the *real-world* disjointness of links. More specifically, the IP-layer topology is basically a set of nodes representing routers, interconnected via point-to-point links. Such a logical abstraction is far insufficient to reveal the actual correlation of links at the physical layer. For example, a set of IP-layer disjoint routers may be supplied by the same power source and be "fate sharing", and a set of IP-layer disjoint links can share the same physical fiber paths, span, etc. Even if the links use disparate physical fibers, they are prone to be correlated if the physical fibers are placed in close geographical locations in which case they can be affected by the same exogenous accident. A notorious anecdote is the earthquake off the coast of Taiwan in Dec. 2006, which destroyed all the IP-layer "independent" fibers lying nearby.

In addition, we can also list multiple routing-level or software-related factors that may trigger failure correlation even between physical-layer topologically disjoint links/paths (*higher-level fate-sharing*). For example, some routers' software may exhibit vulnerabilities to the same attacks; system administrators may load the same faulty configuration onto different devices; and furthermore, different links can get congested together as well. Figure 1 provides an example of correlated congestion, where Z_1 directs its traffic via both A and B to D. When Z_1 sends large amounts of bursty traffic to D, the links $A \rightarrow D$ and $B \rightarrow D$ will get congested together and become prone to be failure-correlated, although they are topologically disjoint.

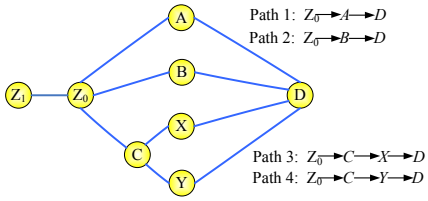

Figure 1: Example topology

In summary, the discrepancies between (a) IP-layer and physical-layer topologies as well as (b) physical-level and higher-level fate-sharing give rise to the considerable inaccuracy of using topological disjointness as a gauge of failure independence. Unfortunately, SRLG can only capture the failure correlations caused by physical sharing, which is only one source of link failure correlations. It cannot capture higher-layer correlations. This limitation motives us to seek better solutions as presented below.

3 Overview of Our Approach

The success of multi-path selection necessitates two components, namely, (a) a *metric* that can accurately reflect failure correlation between different paths, and (b) a *selection algorithm* that can effectively leverage the metric to rule out failure-correlated paths from being selected together. In Section 3.1 we present a new mechanism which can not only evaluate individual path availability, but can also help deriving an accurate multi-path availability metric even in the presence of failure correlation between different links. Then in Section 3.2, we sketch how the new mechanism helps to derive a precise multi-path availability metric, and how we base our multi-path selection scheme on the derived metric.

In Section 6, we discuss real-world deployment issues of our scheme.

3.1 Availability History Window

Given intrinsic dynamics and interactions in Internet routing, as well as the overwhelming complexity of the hardware and software which an IP network relies on, it is difficult (if not impossible) to precisely predict or analyze the correlation between different paths (e.g., SLRG can only capture physical-layer failure correlations). To bypass such complexity while still exploring the failure correlation between different paths, we propose a new mechanism called *availability history window* (AHW), to record path availability histories, from which the failure correlation between different paths can be learned. In the following, we first define AHW on a per-link basis, from which path (multi-path) availability can be then easily derived.

A straightforward interpretation of an AHW is a 0-1 time history window, where '1' corresponds to the time instant when the link is *available* (working), while '0' corresponds to the time instant when the link is *unavailable* (failed). We define a *stable interval* in an AHW as a *continuous* time interval with duration *longer than* a threshold L, where the link is always available. Conversely, a *failure interval* in an AHW is defined as a *continuous* time interval, within which there is no stable interval.

The top two time series in Figure 2 present the example AHWs of links $A \rightarrow D$ and $Z_0 \rightarrow A$ in Figure 1. According to the definition, note that a short time interval during which a link is available is nevertheless attributed to failure interval, as long as its duration is shorter than L. This makes an unstable path less appealing in the selection algorithm (Section 3.2). We can also infer that an AHW is *exclusively* composed of a set of *disjoint* stable and failure intervals, i.e., any time epoch in an AHW must be *either* in a stable *or* a failure interval.

So far, AHW is used to characterize individual links. Now we present how to derive an AHW for an entire path consisting of concatenating links or sub-paths, using the following *series combination* operation.

Series Combination. The AHW of a complete path is computed as the logical AND operation of all 0-1 AHWs of the constitute links or sub-paths. For example, the AHW of path 1 in Figure 1 is computed as the AND operation of the AHWs of links $A \rightarrow D$ and $Z_0 \rightarrow A$, as depicted as the third AHW in Figure 2.

3.2 AHW-based Multi-Path Selection

From the availability history carried by AHWs, we can infer that two paths are highly correlated if they tend to fail at the same time in their AHWs, and vice versa. Several algorithms have been proposed to leverage such history records to cluster correlated links into groups [16, 17]. For our purpose of selecting failure-independent routing paths, we derive a multi-path availability metric from AHWs and employ a selection scheme that can *automatically* preclude failure-correlated paths from being selected together, without additional clustering efforts.

Recall that in multi-path routing, we aim at selecting multiple paths that provide the highest multi-path availability, thus

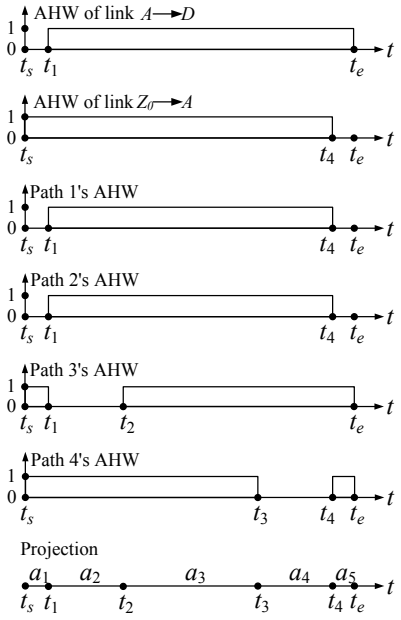

Figure 2: Example AHWs. t_s and t_e represent the start and end times, respectively

we first derive the AHW of a given set of k paths using the following *parallel combination* operation, from which we can eventually compute the *multi-path availability metric*.

Parallel Combination. The AHW of multiple paths between a source-destination pair is computed as the logical OR operation of all AHWs of the paths. For instance in Figure 1, if Z_0 propagates all four paths whose AHWs are given in Figure 2, the resulting AHW is an entire stable interval.

Multi-Path Availability Metric (MAM). It is computed as the duration of all stable intervals in the AHW of multiple paths between a source-destination pair.

Accordingly, our algorithm selects the k AHWs that can produce the largest MAM, by which it can ensure that failure-correlated paths are bound to be less likely chosen together. To illustrate the intuition, consider the example topology in Figure 1 with the AHW of each path given in Figure 2. Suppose path 1 has already been selected, if we further select path 2 which is failure-correlated with path 1, the resulting combined AHW gains no increase in the duration of the stable interval. In contrast, if we parallel-combine path 3 which is failure-independent with path 1, the resulting combined AHW benefits from a significant increase in MAM, thus possessing precedence over path 2 in our selection mechanism.

4 Mathematical Modeling

Following the high-level description above, now we formally present our mathematical modeling of the multi-path selection problem, and ultimately characterize the problem using an Integer Programming formulation. We use the following standard notations: (a) "$\bigcup$" is the UNION operation of sets; (b) "$\vee$" stands for the logical OR operation; and (c) "$|\mathcal{X}|$" operation returns the cardinality of set $\mathcal{X}$. In our context, $|\mathcal{X}|$ refers to the duration time of interval $\mathcal{X}$.

4.1 Mathematical Problem Formulation

First, we formalize AHW, multi-path availability and multi-path selection problem using mathematical notation. Then we prove that the problem is NP-Complete.

Definition 1 An AHW of path i, denoted by $\mathcal{A}_i$, is a 0-1 time series defined as: $\mathcal{A}_i$: $t \rightarrow r_i(t) \in \{0, 1\}$, $t \in [t_s, t_e]$, where t is a historical time epoch in the window range $[t_s, t_e]$; and $r_i(t)$ is the availability record of path i at time t, i.e.:

$$r_i(t) = \begin{cases} 1 & \text{if } t \text{ is in a stable interval of } \mathcal{A}_i, \\ 0 & \text{if } t \text{ is in a failure interval of } \mathcal{A}_i. \end{cases}$$

Recall that $\mathcal{A}_i$ is exclusively composed of interleaving stable and failure intervals. Since the stable intervals are of particular interest for our subsequent modeling purpose, we further have:

Definition 2 Let d_i be the number of stable intervals in $\mathcal{A}_i$, and S_i be the union of all d_i stable intervals, i.e.: $S_i = \bigcup s_i^j = \sum s_i^j, \quad j = 1, 2, \ldots, d_i$, where s_i^j is a single stable interval in $\mathcal{A}_i$.

Definition 3 Let $U = \bigcup S_{i=1,2,\ldots,n}$ be the 1-dimensional universe. Path i's availability is defined as $\theta(i) = |S_i|$. Let M be a set of paths between the same source-destination pair. The multi-path availability of M, denoted by $\theta(M)$, is given by: $\theta(M) = |\bigcup S_i|$.

Definition 4 The multi-path selection problem is defined as a triple of $\langle input, output, objective \rangle$, as follows:

1. An $input$ consists of a family $N = \{S_1, S_2, \ldots, S_n\}$ and an integer k ($\leq n$).
2. An $output$ is a subfamily $M \subseteq N$, with $|M| \leq k$.
3. The $objective$ is to find a subfamily M^* such that:

$$\forall M \neq M^*, \ \theta(M^*) \geq \theta(M). \quad (|M| \leq k, |M^*| \leq k)$$

Theorem 1 *The multi-path selection problem in Definition 4 is NP-complete.*

Proof: We prove this theorem by reducing the well-known NP-complete problem "set-covering decision" [6] to our problem. In the set-covering decision problem, we are given a universe U and a family N of subsets of U. A *cover* is a subfamily $M \subseteq F$ whose union is U. The input to the problem is a pair $\langle U, N \rangle$ and an integer k; the question is whether there is a set-covering of size k or less.

In our problem, the universe U is defined as $\bigcup S_i, \ i \in N$. A family N is composed of $S_{i=1,2,\ldots,n}$, each of which is a subset of U. The integer k corresponds to the constrained number of paths that we can select. Given the same input as the set-covering decision problem, by solving our problem we can derive the maximum multi-path availability $\theta(M^*)$. If $\theta(M^*) = |U|$, according to Definition 3, we have: $\bigcup S_i = U, \ i \in M^*$. Then there definitely exists a cover of U of size k, and the corresponding subfamily is given by M^*. If $\theta(M^*) < |U|$ on the other hand, we can ascertain that there is no set-covering of size k or less, since $\theta(M^*)$ by definition is the maximum value from all possible selections under constraint k. We can thus correctly answer the set-covering problem by solving the problem given in Definition 4. ■

4.2 Optimization Modeling

Based on preceding mathematical definitions, in this section we frame the multi-path selection problem given in Definition 4 as an optimization model and finally transform it into a '0-1' Integer Programming formulation [4] in Theorem 2.

4.2.1 General Optimization Model

Define $x_i \in \{0, 1\}$ as the *indicator variable* for path i, such that:

$$x_i = \begin{cases} 1 & \text{if } i \in M, \\ 0 & \text{if } i \notin M. \end{cases} \tag{1}$$

Then we can formulate the problem as the *optimization model* shown in Equation 2:

$$\begin{aligned} &\textit{Maximize}: \quad \theta(M) = |S_1 x_1 \cup S_2 x_2 ... \cup S_n x_n| \\ &\textit{Subject to}: \quad \sum_{i=1}^{n} x_i \leq k, \ \ x_i \in \{0, 1\} \end{aligned} \tag{2}$$

4.2.2 Optimization Model with Linear Objective

We first linearize the objective function shown in Equation 2, by the following two steps.

Step 1. Recall that all $S_{i=1,2,\ldots,n}$ constitute a 1-dimensional universe U (Definition 3), which can be interpreted as a time interval. And each $\mathcal{A}_i$ consists of d_i disjoint stable intervals $s_i^1, s_i^2, \ldots, s_i^{d_i}$ (Definition 2). Each stable interval can be identified by its two endpoints in the time interval of U. Now we project all end points of all stable intervals in $\mathcal{A}_{i=1,2,\ldots,n}$ into the time interval of U, and assume there are z end points in total after the projection. Let every two adjacent end points on the number line form a new *atomic interval*, then we obtain $z - 1$ atomic intervals denoted by $a_1, a_2, \ldots, a_{z-1}$. Figure 2 gives an example, where the end points t_s, t_1, t_2, t_3, t_4 and t_e produces five new atomic intervals after projection onto the bottom line in Figure 2.

Before we delve into the second step, we first introduce a definition and highlight an important property of atomic intervals. The property facilitates the transformation towards an Integer Programming model in our subsequent discussion.

Definition 5 An atomic interval a_j is *covered* (*not covered*) by $\mathcal{A}_i$ if a_j is *completely within* a stable (failure) interval of $\mathcal{A}_i$. a_j *intersects* with $\mathcal{A}_i$ when only *part* of a_j is within a stable or failure interval of $\mathcal{A}_i$.

Property 1 *The atomic intervals are all* disjoint. *An atomic interval a_j must be either covered or not covered by path i. It cannot intersect with $\mathcal{A}_i$.*

Step 2. Now we introduce a new indicator variable $y_j \in \{0, 1\}$ for atomic interval a_j, such that:

$$y_j = \begin{cases} 1 & \text{if } \exists\, i \in M, a_j \text{ is covered by } \mathcal{A}_i, \\ 0 & \text{if } \forall\, i \in M, a_j \text{ is not covered by } \mathcal{A}_i. \end{cases} \tag{3}$$

Then according to Property 1, we can transform the objective function of Equation 2 into the linear one below:

$$\textit{Maximize:} \quad \theta(M) = \sum_{j=1}^{z-1} |a_j| \cdot y_j \tag{4}$$

Be aware of that, the import of y_j is also accompanied by new constraints, i.e., the restricting relationship between $x_{i=1,2,\ldots,n}$ (Equation 1) and $y_{j=1,2,\ldots,z-1}$. We first introduce the concept of a *covering set* of an atomic interval as follows.

Definition 6 For an atomic interval a_j, its *covering set* $C(j)$ is the set of all $\mathcal{A}_i$ each of which covers a_j. Denote the indices in $C(j)$ as $p_1, p_2, \ldots, p_{|C(j)|}$. Then we have, $\forall p_i \in C(j)$, a_j is covered by $\mathcal{A}_{p_i}$.

Now also according to Property 1, we can express the relationship between $x_{i=1,2,\ldots,n}$ and $y_{j=1,2,\ldots,z-1}$ as follows:

$$y_i = x_{p_1} \vee x_{p_2} \vee \ldots \vee x_{p_{|C(j)|}}, \;\; p_i \in C(j) \tag{5}$$

4.2.3 Integer Programming Model

Now we remove the logical OR operation in the constraints in Equation 5 to finally linearize the problem into an Integer Programming formulation.

Lemma 1 *The constraint $y_j = x_{p_1} \vee x_{p_2} ... \vee x_{p_{|C(j)|}}$ in Equation 5 is equivalent to $y_j \leq x_{p_1} + x_{p_2} .. + x_{p_{|C(j)|}}$.*

Proof: We prove this theorem by showing that in any case, the values of y_j yielded by both expressions are equal.
Case 1. When $\forall p_i \in C(j)$, $x_{p_i} = 0$, both expressions yield the same value, i.e., $y_j = x_{p_1} \vee x_{p_2} ... \vee x_{p_{|C(j)|}} = 0$, and $y_j \leq x_{p_1} + x_{p_2} .. + x_{p_{|C(j)|}} = 0 \Rightarrow y_j = 0$.
Case 2. When $\exists p_i \in C(j)$, $x_{p_i} = 1$, let w be the number of p_i that $x_{p_i} = 1$. Then $y_j = x_{p_1} \vee x_{p_2} ... \vee x_{p_{|C(j)|}} = 1$. On the other hand, $y_j \leq x_{p_1} + x_{p_2} .. + x_{p_{|C(j)|}} = w$. Note that $y_j \in \{0, 1\}$, now y_j can take either the value '0' or '1' under this inequality constraint ($y_j \leq w$). Yet, observe that the coefficient of y_j in the objective function (Equation 4) is positive, and we are maximizing the value of objective function. So given the constraint $y_j \leq l$, in the optimal solution there must be $y_j = w$.

Therefore in any case, both expressions render the same value to y_j. This proves the theorem. ■

Theorem 2 *The multi-path selection problem given in Definition 4 can be modeled as the following 0-1 integer programming formulation.*

$$\begin{aligned} &\textit{Maximize:} \quad \theta(M) = \sum_{j=1}^{z-1} |a_j| \cdot y_j \\ &\textit{Subject to:} \quad \sum_{i=1}^{n} x_i \leq k, \;\; x_i \in \{0,1\}, \;\; y_j \in \{0,1\}, \\ &\qquad\qquad y_j \leq \sum_{j=p_1}^{p_{|C(j)|}} x_i, \;\; j = 1, 2, \ldots, z-1 \end{aligned} \tag{6}$$

The proof of this theorem is straightforwardly given by Lemma 1. Till now, we eventually formulate the problem as a standard 0-1 Integer Programming.

5 Algorithm

In the current Internet, a router can have hundreds of neighbors, thus possibly hundreds of different paths for one destination. Therefore, a simple brute-force method for this NP-Complete problem (Theorem 1) with such an input size can be computationally prohibitive. The model we gave in Section 4 is a precise mathematical formulation which not only helps us thoroughly understand the problem, but also enables us to leverage various well-established algorithms to solve the Integer Programming problem, such as branch-and-bound and cutting plane methods [4], or some efficient approximation solutions [1, 12]. However, those algorithms may also be too slow to be implemented in core routers.

We present a simple and efficient approximation algorithm. Basically, in each iteration the algorithm greedily selects the path that can maximize the multi-path availability accumulated so far. Table 1 describes the algorithm. Suppose the number of stable intervals in each AHW is bounded by a small constant, then the complexity for selecting k out of n paths for one destination is $O(nk)$.

Table 1: Greedy Algorithm for multi-path selection.

line	action
1	**for** each destination D in the network
2	*get* all the n candidate paths associated with AHWs
3	*initialize* $M = \emptyset, \theta(M) = \emptyset$)
4	//begin loops for greedy selection **while** $\|M\| \leq k$ and $\theta(M) \neq \|U\|$
5	*select* a path p that maximizes $\|\theta(M \cup p)\|$
6	*add* p to M, *update* $\theta(M) = \theta(M \cup p)$

Theorem 3 *Let $\theta(M^*)$ and $\theta(M)$ be the multi-path availabilities yielded by the optimal solution and the greedy heuristic given in Table 1, respectively. Then $\theta(M)$ has the lower-bound $\theta(M) \geq \theta(M^*) - \left(1 - \frac{1}{k}\right)^{k-1} (\alpha_1 - \beta_1)$ where k is the number of paths that we can select, α_1 is the union of time intervals that is not covered by the first greedily selected path but covered in $\theta(M^*)$, and β_1 is the union of time intervals that is covered by the first greedily selected path but not covered in $\theta(M^*)$.*

Proof: Let $\theta(M)_i$ be the multi-path availability produced by the first i selected paths in the greedy algorithm ($\theta(M)_k = \theta(M)$), and Δ_i be the difference between $\theta(M^*)$ and $\theta(M)_i$ (i.e., $\Delta_i = \theta(M^*) - \theta(M)_i$). In the following we prove the theorem by deriving the upper bound of Δ_k (i.e., $\theta(M^*) - \theta(M)$).

Let α_i be the union of time intervals in the universe U that is covered by $\theta(M^*)$ but not covered by $\theta(M)_i$; and conversely β_i be the union of time intervals in the universe U that is covered by $\theta(M)_i$ but not covered by $\theta(M^*)$. Accordingly, we have: $\Delta_i = \alpha_i - \beta_i \;\; (\geq 0)$.

Now we start from the beginning. After greedily selecting the first path (the one with the highest individual availability), we have $\Delta_1 = \alpha_1 - \beta_1$. Since by definition, α_1 is

covered in $\theta(M^*)$ (by the k paths in K^*). This means that there must exist a single path (say p) of the totally n paths that can cover $\frac{1}{k}\alpha_1$ (otherwise any k paths cannot cover α_1). Thus if p is selected as the second path by the greedy algorithm, $\theta(M)_2$ can be increased by at least $\frac{1}{k}\alpha_1$. And according to the greedy nature of the heuristic, by selecting the second path the increase in $\theta(M)_2$ must be at least $\frac{1}{k}\alpha_1$, i.e.: $\theta(M)_2 \geq \theta(M)_1 + \frac{1}{k}\alpha_1$. Then Δ_2 must be reduced by at least $\frac{1}{k}\alpha_1$, i.e.: $\Delta_2 \leq \Delta_1 - \frac{1}{k}\alpha_1$. Analogously, after greedily selecting the i^{th} path, we have: $\Delta_i \leq \Delta_{i-1} - \frac{1}{k}\alpha_{i-1}$. Note that we also have $\Delta_i \leq \alpha_i$, yielding:

$$\Delta_i \leq \Delta_{i-1} - \frac{1}{k}\alpha_{i-1} \leq \Delta_{i-1} - \frac{1}{k}\Delta_{i-1} = \left(1 - \frac{1}{k}\right)\Delta_{i-1}$$

By resolving this iteration finally we get:

$$\Delta_k \leq \left(1 - \frac{1}{k}\right)^{k-1}\Delta_1 = \left(1 - \frac{1}{k}\right)^{k-1}(\alpha_1 - \beta_1)$$

Since $\theta(M) = \theta(M^*) - \Delta_k$, this proves the theorem. ■

Corollary 2 *As k increases, the lower-bound of $\theta(M)$ is also increased, with the limit* $\lim_{k\to\infty} \theta(M) = \theta(M^*) - \frac{1}{e}(\alpha_1 - \beta_1)$.

6 Deployment Issues

In this section we discuss issues that arise in a deployment of our mechanism. Due to incentive and policy concerns in inter-domain environments, we envision the proposed algorithm to be applied in intra-domains.

AWH Propagation In a link-state routing protocol such as OSPF, each node can monitor its adjacent links and generate corresponding AHWs. An AHW of a link can be encapsulated in each Link State Announcement (LSA) and propagated along with LSAs. In this manner, every node has the knowledge of complete network topology with each link annotated with its AHW, from which the paths' AHWs can be locally computed by each node via the operations defined in Section 3. It may be expensive to straightforwardly propagate the naive time series of an AHW. Since an AHW is composed of a 0-1 binary sequence, and failure intervals are relatively rare and usually condensed into a short time range [7], the time series can be efficiently compressed (e.g., using run-length encoding).

Correlation Predictability. Observe that, by utilizing availability *history*, in effect we only directly learn the *historical* failure correlation between different paths, and we essentially use such historical failure correlation to steer the *future* multi-path selections. Therefore, our scheme is most effective in environments where failures repeat in the future.

Time Synchronization. In our scheme, the failure correlation between paths is indicated by the *time correlation* between failure intervals in the AHWs; and our selection algorithm needs to take the end points (time epochs) of stable and failure intervals as input. Hence, to accurately exploit the failure correlation from AHWs, it is required that all AHWs involved in the selection must be time-synchronized.

7 Conclusion and Future Work

In this paper we launch an initial investigation in availability-oriented multi-path selection. We propose a new way to detect and avert failure-correlated paths by recording and utilizing path/link availability history.

We believe correlation-resilient multi-path selection is a promising research direction, and AHW mechanism can be potentially used for network planning as well. We hope this paper can motivate future endeavors in this direction. Though obtaining real-world link failure dataset is a challenging task, we plan as a future work to perform a dedicated measurement on real-world failure correlation and inspect the real-world effectiveness of the proposed scheme.

References

[1] N. Alon and A. Srinivasan. Improved parallel approximation of a class of integer programming programming problems. In *ICALP: Proceedings of International Colloquium on Automata, Languages and Programming*, 1996.

[2] D. Andersen, A. Snoeren, and H. Balakrishnan. Best-path vs. multi-path overlay routing. In *Internet Measurement Conference*, 2003.

[3] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. Overview and Principles of Internet Traffic Engineering. RFC 3272 (Informational), May 2002.

[4] Bradley. *Applied Mathematical Programming, Chapter 9*. Addison-Wesley, 1977.

[5] M. Caesar and J. Rexford. BGP routing policies in ISP networks. *IEEE Network Magazine*, Special issue on interdomain Routing, Dec 2005.

[6] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms (Second Edition), pp.1033*. 2001.

[7] G. Iannaccone, C. Chuah, R. Mortier, S. Bhattacharyya, and C. Diot. Analysis of link failures in an ip backbone. In *Proc. of ACM SIGCOMM Internet Measurement Workshop*, 2002.

[8] H. T. Kaur, S. Kalyanaraman, A. Weiss, S. Kanwar, and A. Gandhi. Bananas: An evolutionary framework for explicit and multipath routing in the internet. In *ACM SIGCOMM Future Directions in Network Architecture*, 2003.

[9] A. Khanna and J. Zinky. The revised ARPANET routing metric. *SIGCOMM Computer Communincation Review*, 1989.

[10] R. Kompella, J. Yates, A. Greenberg, and A. Snoeren. IP fault localization via risk modeling, May 2005.

[11] N. Kushman, S. Kandula, D. Katabi, and B. M. Maggs. R-BGP: Staying Connected In a Connected World. In *Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, April 2007.

[12] C.-J. Lu. A deterministic approximation algorithm for a minmax integer programming problem. In *SODA: Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms*, 1999.

[13] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. E. Anderson. The end-to-end effects of internet path selection. In *Proc. of ACM SIGCOMM*, 1999.

[14] A. Snoeren, K. Conley, and D. Gifford. Meshbased content routing using XML, 2001.

[15] I. Stocia and H. Zhang. Lira: An approach for service differentiation in the internet. In *Proceedings of Nossdav*, 1998.

[16] J. Strand, A. L. Chiu, and R. Tkach. Issues for routing in the optical layer. *IEEE Communications Magazine*, 39(2):81–87, Feb. 2001.

[17] A. Tachibana, S. Ano, T. Hasegawa, M. Tsuru, and Y. Oie. Locating congested segments on the Internet by multiple paths' delay performance clustering. *IEEE International Conference on Communications*, 2007.

[18] D. Wendlandt, I. Avramopoulos, D. Andersen, and J. Rexford. Don't secure routing protocols, secure data delivery. In *Proc. of ACM Workshop on Hot Topics in Networks (Hotnets-V)*, Nov. 2006.

[19] W. Xu and J. Rexford. MIRO: Multi-path Interdomain ROuting. In *ACM SIGCOMM*, 2006.

[20] X. Yang and D. Wetherall. Source Selectable Path Diversity via Routing Deflections. In *ACM SIGCOMM*, 2006.