An Analysis of Path-Vector Routing Protocol Convergence Algorithms

Dan Pei	Beichuan Zhang	Dan Massey	Lixia Zhang
UCLA CSD	USC/ISI	USC/ISI	UCLA CSD
peidan@cs.ucla.edu	bzhang@isi.edu	masseyd@isi.edu	lixia@cs.ucla.edu

Technical Report TR040009
UCLA Computer Science Department
March 22nd, 2004

Abstract

Today's Internet uses a path vector routing protocol, BGP, for global routing. After a connectivity change, a path vector protocol tends to explore a potentially large number of alternative paths before converging on new stable paths. Several techniques for improving path vector convergence have been proposed, however there has been no comparative analysis to judge the relative merit of each approach. In this paper we develop a novel analytical framework for analyzing the convergence delay bounds of path-vector routing protocols in general. Our framework can accommodate different message processing delay models. By incorporating the commonly used uniform processing delay model we are able to fill in all the cases where analytical results are missing previously. The results obtained by using our framework not only confirm the previous work but also provide new insights into the underlying network behavior. We then present a new delay model, the Q model, which takes into account the actual message queueing delay in actual BGP implementations *and* simulations. By incorporating the Q model in our framework, we are able to obtain tighter delay bounds and explain simulation results that cannot be explained using the previous uniform message delay model.

Keywords: Routing Protocol Convergence, Path Vector Protocol, BGP.

An Analysis of Path-Vector Routing Protocol Convergence Algorithms

Dan Pei
UCLA CSD
peidan@cs.ucla.edu

Beichuan Zhang
USC/ISI
bzhang@isi.edu

Dan Massey
USC/ISI
masseyd@isi.edu

Lixia Zhang
UCLA CSD
lixia@cs.ucla.edu

***Abstract*— Today's Internet uses a path vector routing protocol, BGP, for global routing. After a connectivity change, a path vector protocol tends to explore a potentially large number of alternative paths before converging on new stable paths. Several techniques for improving path vector convergence have been proposed, however there has been no comparative analysis to judge the relative merit of each approach. In this paper we develop a novel analytical framework for analyzing the convergence delay bounds of path-vector routing protocols in general. Our framework can accommodate different message processing delay models. By incorporating the commonly used uniform processing delay model we are able to fill in all the cases where analytical results are missing previously. The results obtained by using our framework not only confirm the previous work but also provide new insights into the underlying network behavior. We then present a new delay model, the Q model, which takes into account the actual message queueing delay in actual BGP implementations *and* simulations. By incorporating the Q model in our framework, we are able to obtain tighter delay bounds and explain simulation results that cannot be explained using the previous uniform message delay model.**

Keywords: Routing Protocol Convergence, Path Vector Protocol, BGP.

I. Introduction

The global Internet routing uses a *path vector* routing protocol, Border Gateway Protocol (BGP). In a path vector protocol, routing update messages list the entire path to destination. A well known performance issue with path vector protocols is that, after a topological failure, they tend to explore a large number of alternate paths, many of which may have been obsoleted by the same failure, before converging to new stable paths. Earlier measurement efforts [1][2] confirmed the existence of such slow convergence in the Internet, and research efforts over the last few years have produced a number of improvement proposals including SSLD [1][3], WRATE [1][3], Assertion [4], Route Cause Origin [5], Ghost Flushing [6], and Root Cause Notification (RCN) [7].

Unfortunately, each of these proposed schemes focuses on a different aspect of the path vector algorithm, some of them do not provide an analytical model, and none of them provides an analytical model that covers both path fail-down and path fail-over cases. When an analytical mode is provided for a scheme, it is often incompatible with that of another scheme. The lack of a general analytical framework makes it difficult to compare different path vector algorithms to judge the relative merit of each approach.

This paper presents a general framework for analyzing the convergence delay bounds of path vector routing protocols for both path fail-down and fail-over cases. We use our framework to analyze the convergence delay bound of the standard path vector algorithm and three representative convergence improvement schemes: Assertion, Ghost Flushing, and RCN. By using this analytical framework with a commonly used message delay model, we are able to not only confirm existing results but also fill in missing analytical results in previous work. Our results provide the first complete analytical comparison of these protocols, illustrating which factors play critical roles in each improvement. We believe our framework can also be used to analyze and compare future convergence improvement proposals.

All existing work [1][2][6][7] assume that the processing time of routing messages is either negligible or modeled as a random variable independent of the router load. In reality, however, bursty BGP updates can lead to long processing delay due to queueing. In popular BGP simulators such as SSFNET[8] and BGP++[9], BGP messages that arrive in burst are queued and processed in a FIFO order, as a real router would do, thus the total processing delay of a message depends on the router load and this delay can be long enough to affect the routing convergence behavior. To address this mismatch between the delay model and implementations, we develop a new message delay model, the Q model, which takes into

account the queueing delay of the messages at each node. Using the general framework and the Q model, we are able to obtain tighter convergence delay bounds, gain new insights into how different topological properties may affect the convergence time, and explain simulation results that cannot be explained by previous delay model.

II. The SPVP Model and Convergence Algorithms

In this section, we present a simplified model for the BGP routing protocol, *Simple Path Vector Protocol (SPVP)*, its convergence definitions, and the mechanisms to improve SPVP's convergence time.

A network is modeled as a directed connected graph $G = (V, E)$. $V = \{0, 1, \cdots, N-1\}$ represents the set of N nodes that run SPVP protocol, and they are connected by links in E. Without loss of generality, in this model we consider only a single destination node p which is connected to node 0 and $p \notin V$. A path to destination p is an ordered sequence of nodes $r = (v_k v_{k-1} \dots v_0)$ such that link $[v_i\ v_{i-1}] \in E$ and $v_i \in V$ for all $i, 0 \leq i \leq k$, and $v_0 = 0$. We say $v_i \in r, \forall i, 0 \leq i \leq k$; $[v_i\ v_{i-1}] \in r, \forall i, 1 \leq i \leq k$; $(v_i\ v_{i-1} \cdots v_0) \subset r, \forall i, 0 \leq i \leq k-1$. We define $length(r) = k$, and $length(\epsilon) = \infty$ for empty path. This model roughly matches Internet BGP routing: nodes in V correspond to Internet Autonomous Systems and p corresponds to an IP prefix. The following notations are used throughout the paper:

$degree(G, v)$	degree of node v in G
$distance(G, v, u)$	shortest distance between v and u
$Distance(G, v)$	$= \max_{u \in G}\{distance(G, v, u)\}$
$diameter(G)$	$= \max_{v \in G}\{Distance(G, v)\}$

SPVP is a path vector routing protocol in which each node advertises *only* its best path to its neighbor nodes. A node v stores the latest path received from all the neighbors, selects the best path, $r(v)$, according to its routing policies and ranking functions, and advertises $r(v)$ to its neighbors. In theory SPVP should be able to work with arbitrary routing policies. Previous studies showed that some path selection policies can lead to persistent path oscillation [10]. For clarity, this paper only considers shortest-path policy(when two paths have the same length, the path from the neighbor with lower node ID is preferred) which has been proven to converge [11]. SPVP is an event-driven protocol; after the initial path announcement, further updates are sent *only* if the best path changes.

During SPVP operations, links may fail and recover. Node v can detect the failure and recovery of link $[v\ u]$, node 0 can detect the failure and recovery of link $[0\ p]$. Upon detecting a link failure, each node recomputes the best path and sends updates if the best path changes. If link status changes or update messages result in no path to the destination, then $r(v) = \epsilon$ and a *withdrawal* message carrying ϵ as the path is sent to neighbors.

Like BGP, SPVP has a *Minimum Route Advertisement Interval (MRAI)* timer which guarantees that any two updates sent from v to u is separated by at least $\mathcal{M}$ seconds. Following the BGP specification [12], the MRAI timer is not applied to withdrawal messages.

A. SPVP Convergence Definitions

In [1][2][6][7], BGP routing events are categorized into four classes: $T_{up}, T_{short}, T_{down}$, and T_{long}. In T_{up}, a previously unavailable prefix is announced, in T_{short} an existing path is replaced by a shorter (more preferred) path. A failure of the link $[0\ p]$ causes a previously reachable destination p to become unreachable, resulting in a T_{down} event, and all nodes withdraw their paths to the destination p, potentially after exploring a number of obsolete alternate paths. In a T_{long} event, a link $[u\ v]$ fails, and the nodes relying on this link to reach p have to switch to another path.

***Definition** 1: Converged State*: a node v is in a converged state iff $r(v)$ will not change any more.

***Definition** 2: Network Convergence Delay:*, denoted $time(T)$, starts when a triggering event T occurs and ends when all the nodes in the network are converged.

Internet measurements [2] showed that in both T_{up} and T_{short} events, the convergence delay is roughly proportional to the network diameter. Slow convergence is commonly associated with T_{down} and T_{long} events [1], [2]. In this paper, we focus on T_{down} and T_{long} only. For clarity, our analysis and simulations focus on the impact of a *single* link failure event in a topology where each node has a degree of at least 2.

B. SPVP Convergence Algorithms

This section reviews existing convergence algorithms proposed to improve convergence time of the basic SPVP. Due to page limit, we focus on three representative algorithms: Assertion (SPVP-AS), Ghost Flushing (SPVP-GF), and Route Cause Notification (SPVP-RCN).

SPVP-AS This algorithm [4] reduces the chance of choosing or propagating obsolete paths by checking path consistency for incoming path advertisements. More specifically, assume that node v receives two paths, r and r', from two neighbors u and w respectively. SPVP-AS states that, if $u \in r'$, then it must be true that $r \subset r'$;

otherwise, r' is regarded as obsolete and removed. SPVP-AS does not eliminate the propagation of all obsolete paths, and its effectiveness is sensitive to the topology.

SPVP-Ghost Flushing(SPVP-GF) In SPVP-GF, if node u changes to a path less preferred and u cannot send the new path to neighbor v immediately due to MRAI delay, u will "flush out" a withdrawal message immediately to remove the path previously advertised to v. Therefore, even though the new path announcement may be delayed, the obsolete path is quickly removed from the network. SPVP-GF does not eliminate the propagation of all the obsolete paths; its effectiveness depends on topological details.

SPVP-RCN In SPVP-RCN, each node maintains a sequence number and increments it by 1 whenever its best path changes. When an event happens, the node that detects the event attaches a *root cause*, defined as the combination of the node's ID and its current sequence number, to the routing update message. If this update message causes other routers to change their paths to the destination, they will send out update messages containing the original root cause information. Suppose a routing event triggers node v to send an update with a root cause $(v, seqnum(v))$, any path containing v but with a sequence number smaller than $seqnum(v)$ is considered obsolete and removed. Since every update carries the root cause, once a node receives the first routing message, it can immediately discard all the obsolete paths.

Other Algorithms In *Sender Side Loop Detection (SSLD)*[1], the sender v checks the path r before sending it to the receiver u. If $u \in r$, r will be discarded by u due to loop detection, therefore v will send a withdrawal instead. *Withdrawal rate limiting (WRATE)* requires that the MRAI timer be applied to withdrawal messages as well. RCO (Route Change Origin) is similar to RCN, but not applicable to T_{long}, thus it has the same T_{down} delay bound as RCN, and the same T_{long} delay bound as SPVP. FESN (Forwarding Edge Sequence Number) [13] is the similar to RCN except that FESN uses link sequence number instead of node sequence number. Therefore FESN has the same T_{down} and T_{long} delay bound as RCN.

C. Algorithm Classification

SPVP and its improvement algorithms can be categorized into two classes. In SPVP and SPVP-GF, if v receives a path r from neighbor u, r is not invalidated unless u withdraws it or advertises a new path. In SPVP-AS, r can be invalidated if one of the nodes in r is a direct neighbor to v and it sends path information conflicting with r. In all theses algorithms, a path received from one neighbor has no impact or limited impact on the validity of paths received from other neighbors. We call this type of algorithms "Local Notification of Path Unavailability (**LNPU**)." SSLD and WRATE also belong to LNPU.

SPVP-RCN, RCO, and FESN fall into a different class, which we call "Remote Notification of Path Unavailability (**RNPU**)." In RNPU, every update carries its unique root cause. Once a node receives the ***first*** update during convergence period, it immediately knows the root cause of this routing event, and is able to invalidate all obsolete paths, regardless of from which neighbor the paths are received.

III. A Framework for Convergence Analysis

In this section, we develop a general framework for analyzing the bounds of convergence time for both LNPU and RNPU algorithms.

A. Notations and Definitions

For both the LNPU and RNPU classes, we define some common notation. In particular, we note that after a failure occurs, a path is either ***valid*** or ***invalid***. For T_{down} event, all non-empty paths are invalid. For a T_{long} event which is triggered by the failure of link $[c\ b]$, a path r is invalid iff $[c\ b] \in r$.

We use the following notations to denote various set of all the invalid paths. In particular $R_v(G, A)$ is all the invalid paths that a specific node v may have; R^l is all the invalid paths whose length is not longer than l.

$R(G, A)$: the set of all the *invalid* paths in G allowed by algorithm A
$R_v(G, A) = \{r \mid r \in R(G, A) \wedge r = (v \cdots 0)\}$
$R^l(G, A) = \{r \mid r \in R(G, A) \wedge length(r) \leq l\}$
$R_v^l(G, A) = R_v(G, A) \cap R^l(G, A)$

In order to analyze convergence time, we require some information about message passing delay. When node u changes its path, we are interested in the maximum time that may elapse before its neighbor v learns the path via u is *obsolete*. We let $\mathcal{D}(G, [u\ v])$ denote the upper bound on this time interval.

$\mathcal{D}(G, [u\ v])$ can include the MRAI delay, transmission delay, propagation delay, queueing delay, and processing delay. For example, suppose node u changes its path at time t_1. In SPVP, node u sends neighbor v an announcement listing the new path. The new path announcement may be delayed by the MRAI timer at u, then incurs some transmission, propagation and queueing delay before being accepted by the processor at v. Finally v takes some

time to process the update and update its routing table at time t_2. By definition, $\mathcal{D}(G,[u\ v]) \geq t_2 - t_1$.

In above example, the announcement implicitly obsoletes u's old path and, at the same time, provides a replacement path. However, in some algorithms there is a subtle but important distinction between the delay in learning a path is *obsolete* and the delay in learning a *replacement* path.

In any SPVP-based algorithm, node u's new announcement can be delayed by the MRAI timer and node u cannot send its replacement path until the MRAI timer expires. However, an SPVP-GF node u that is blocked by the MRAI timer can immediately send a "flushing withdraw" to announce its previous path is now *obsolete*. This is allowed since the MRAI timer does not apply to withdrawals. When the MRAI timer later expires, node u will send an announcement listing the replacement path. In other words, SPVP-GF provides a fast mechanism for *obsoleting* old information and only later sends the replacement path. Node v initially learns u's path is obsolete, replaces it with "no path", and only later learn u's actual replacement path. We use $\mathcal{D}_{replace}(G,[u\ v])$ to denote the upper bound on learning the *replacement* path. In algorithms such as SPVP, $\mathcal{D}_{replace}(G,[u\ v]) = \mathcal{D}(G,[u\ v])$. But in other algorithms such as SPVP-GF, we can have $\mathcal{D}_{replace}(G,[u\ v]) = \mathcal{D}(G,[u\ v]) + \mathcal{M}$.

$\mathcal{D}(G,[u\ v])$: maximum time that may elapse between u changes its path and its neighbor v learns the path via u is *obsolete*
$\mathcal{D}_{replace}(G,[u\ v])$: maximum time that may elapse between u changes its path and its neighbor v learns u' s replacement path
$f(r) = \sum_{i=1}^{l} \mathcal{D}(G,[v_{i-1}\ v_i])$: ***lifetime*** of invalid path $r = (v_i v_{i-1} \cdots v_0)$ during T_{down} convergence
$g(r) = \sum_{i=1}^{l} \mathcal{D}(G, v_{i-1}, v_i)$. The *lifetime* of an *invalid* path $r = (v_l\ v_{l-1} \cdots v_0 t_{J-1} \ldots t_0)$ during $T_{long}(c\ t_{J-1})$ convergence

B. T_{down} analysis

In a T_{down} event, the network converges when all the nodes learn that the destination is unreachable. We first consider LNPU algorithms, including SPVP, SPVP-GF, and SPVP-AS. In LNPU, all possible paths need to be explicitly withdrawn by direct neighbors before a node can conclude that the destination is unreachable. Therefore, we are interested in how long a path can exist in the network.

***Lemma** 1:* Given any path $r = (v_l\ \ v_{l-1} \cdots v_0)$ of length l, the path will be withdrawn by time $\sum_{i=1}^{l} \mathcal{D}(G,[v_{i-1}\ v_i])$ and will never be restored.

Proof: We prove this lemma by induction on l. Consider $l = 1$ and without loss of generality, let path $r = (v_1\ v_0)$. At time 0, the failure occurs, v_0 withdraws its path and will never restore it. This information propagates to v_1 and has been processed by v_1 by the time $\mathcal{D}(G,[v_1\ v_0])$. The path $(v_1\ v_0)$ will be withdrawn. Since a path of length 1 can only be learned from v_0, it will not be restored. Therefore the lemma is true for $l = 1$.

Assuming the lemma is true for any $r = (v_l v_{l-1} \cdots v_0)$, let's consider any path $r' = (v_{l+1} v_l v_{l-1} \cdots v_0)$. According to the induction hypothesis, v_l has withdrawn path r from its routing table by time $\sum_{i=1}^{l} \mathcal{D}(G,[v_{i-1}\ v_i])$ and sends a message x to its neighbors. Any earlier updates from v_l to v_{l+1} will have been overwritten by x, and it takes at most $\mathcal{D}(G,[v_l\ v_{l+1}])$ for message x to be processed by v_{l+1}. And v_l will never advertise r again according to the induction hypothesis. Therefore, the hypothesis is true for $l+1$. ■

Based on Lemma 1, for any invalid path $r = (v_l\ \ v_{l-1} \cdots v_0)$, we define its ***lifetime*** as $f(r) = \sum_{i=1}^{l} \mathcal{D}(G,[v_{i-1}\ v_i])$. After its lifetime, an invalid path is guaranteed to be withdrawn from the network and will not be restored later. For $r = (0)$, define $f(r) = 0$. Apparently, if $r \subset r'$, $f(r) < f(r')$.

***Theorem** 1:* For any network G and any LNPU algorithm A,

$$time(T_{down}) \leq \max_{r \in R(G,A)} \{f(r)\}$$

Proof: Based on Lemma 1, after the maximum lifetime of all paths in the network has passed, no path will exist in any node's routing table. Therefore all nodes must have concluded that the destination is unreachable, thus the network converges. ■

Algorithms of *Remote Notification of Path Unavailability (RNPU)* converges faster than LNPU, because every message carries a root cause notification, and once it is received, the receiving node will be able to discard all invalid paths. Therefore, the network converges when all nodes receive at least one message.

***Theorem** 2:* For any network G and any RNPU algorithm A,

$$time(T_{down}) \leq \max_{v \in V} \{ \min_{r \in R_v(G,A)} \{f(r)\}\}$$

Proof: Based on Lemma 1, by the time of $\min_{r \in R_v(G,A)}\{f(r)\}$, node v has withdrawn one of its invalid paths, which is a result of receiving a message from its neighbor. Therefore v knows the root cause and is converged. The maximum of this time over all nodes guarantees that all nodes are converged. ■

C. T_{long} Analysis

In T_{long} events, a link fails but the destination is still reachable via alternate paths. A node is *affected* if its

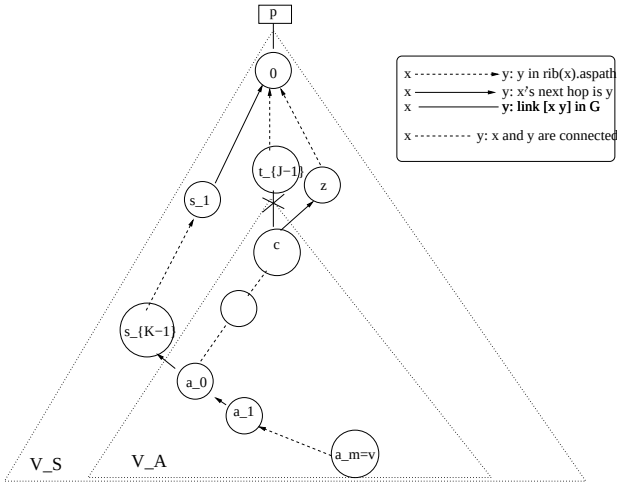


Fig. 1. Routing Tree after T_{long} Convergence

path becomes invalid after the failure. All affected nodes need not only discard invalid paths, but also converge to the new best paths. If the link $[c\ b]$ fails, and c is J hops away from node 0, all invalid paths have the form $(v_l \dots v_0 t_{J-1} \dots t_0)$, where $v_0 = c$, $v_i (0 \le i \le l)$ are affected nodes, and $t_i (0 \le i \le J-1)$ are not affected. Therefore, all affected nodes form a single connected subgraph $G_A(V_A, E_A)$. These notations are summarized as follows.

$G'(V, E')$	Topology after an event T occurs in $G(V, E)$
V_S	Nodes whose paths have not changed after T
V_A	Nodes whose paths changed after T
E_A	Links whose both ends belong to V_A
$G_A(V_A, E_A)$	Sub-graph of G' with V_A and E_A
$r^{old}(v)$	node v's best path in G
$r^{new}(v)$	node v's best path in G'
J	$= distance(G, c, 0)$, in $T_{long}([c\ b])$

***Lemma** 2:* After the network converges, for any $v \in V_A$, its new best path $r^{new}(v)$ must have the form $(a_m \dots a_0 s_{K-1} \dots s_0)$ where $v = a_m$, $a_i \in V_A (0 \le i \le m)$, $s_i \in V_S (0 \le i \le K-1)$, as illustrated in Figure 1. Note, this path is v's backup path, and its length is $K+m$.

Proof: Consider any link $[s\ w] \in r^{new}(v)$ where $s \in V_S$. It must be true that $w \in V_S$ too. Otherwise, $r^{old}(w) \ne r^{new}(w)$, which makes it impossible for $r^{new}(s) = (s\ r^{new}(w)) = r^{old}(s)$, which contradicts to $s \in V_S$. ■

Similar to the T_{down} analysis, we have the following lemma on how long an invalid path can exist in the network.

***Lemma** 3:* During $T_{long}([c\ b])$, for any invalid path $r = (v_l\ v_{l-1} \cdots v_0 t_{J-1} \cdots t_0)$, where $v_0 = c, t_{J-1} = b$, r will have been withdrawn by time $\sum_{i=1}^{l} \mathcal{D}(G, [v_{i-1}\ v_i])$ and never be restored later.

Proof: The proof is similar to Lemma 1, by induction on l.

Consider $l = 1$ and without loss of generality, let path $r = (v_1\ v_0\ t_{J-1} \cdots t_0)$. At time 0, the failure occurs, v_0 withdraws this path r and will never restore it. This information propagates to v_1 and has been processed by v_1 by the time $\mathcal{D}(G, [v_1\ v_0])$. The path $(v_1\ v_0\ t_{J-1} \cdots t_0)$ will be withdrawn. Since an invalid path of length J can only be learned from v_0, it will not be restored. Therefore the lemma is true for $l = 1$. Assuming the lemma is true for any $r = (v_l v_{l-1} \cdots v_0\ t_{J-1} \cdots t_0)$, let's consider any path $r' = (v_{l+1} v_l v_{l-1} \cdots v_0\ t_{J-1} \cdots t_0)$. According to the induction hypothesis, v_l has withdrawn path r from its routing table by time $\sum_{i=1}^{l} \mathcal{D}(G, [v_{i-1}\ v_i])$ and sends a message x to its neighbors. Any earlier updates from v_l to v_{l+1} will have been overwritten by x, and it takes at most $\mathcal{D}(G, [v_l\ v_{l+1}])$ for message x to be processed by v_{l+1}. And v_l will never advertise r again according to the induction hypothesis. Therefore, the hypothesis is true for $l+1$.

■

Based on Lemma 3, for T_{long} event, we define the *lifetime* of an *invalid* path $r = (v_l\ v_{l-1} \cdots v_0 t_{J-1} \dots t_0)$ as $g(r) = \sum_{i=1}^{l} \mathcal{D}(G, [v_{i-1}\ v_i])$. Define $g(r) = 0$ for $r = (c\ t_{J-1} \dots t_0)$. Apparently, $g(r) < g(r')$ if $r \subset r'$.

***Theorem** 3:* For any network G, and any LNPU algorithm A, $time(T_{long}) \le \max_{v \in V_A, r^{new}(v) = (a_m \cdots a_0\ s_{K-1} \cdots s_0)} \{larger\{ \max_{r \in R_{a_m}^{K+m}(G,A)} \{g(r)\}, \max_{r \in R_{a_0}^{K}} \{g(r)\} + \sum_{i=1}^{m} \mathcal{D}_{replace}(G, [a_{i-1}\ a_i]) \ \} \ \}$

Proof: In general, T_{long} convergence of node $v = a_m$ consists of two processes, the withdrawal of invalid paths and the propagation of new valid paths. In this theorem, the first term is the time necessary for withdrawing invalid paths, the second term is the time necessary for propagating new paths, and the overall convergence time is the larger of them as both will be guaranteed to be done by that time.

Note, for a_m, the length of its new best path is $K + m$. Therefore, according to Lemma 3, after time $\max_{r \in R_{a_m}^{K+m}(G,A)} \{g(r)\}$, all a_m's invalid paths that are shorter than its new best path have been withdrawn. Once the new path arrives, any remaining invalid paths will not affect the a_m's convergence. This is the first term in the theorem.

The time of the new path takes to get to a_m is the sum of the time for a_0 to get its new path and the time spent on propagation from a_0 to a_m. For a_0, the

new path is from an unaffected neighbor, s_{K-1}, so it is already in a_0's routing table prior to the failure. Once a_0's invalid paths whose length are shorter than K has been withdrawn, a_0 will converge to its new path, and this time is $\max_{r \in R^K_{a_0}(G,A)}\{g(r)\}$. For the new path to propagate from a_0 to a_m, since it is to "replace" old paths along the way, each hop can have delay up to $\mathcal{D}_{replace}(G,[u\ v])$. Therefore the total propagation time is $\sum_{i=1}^{m}\mathcal{D}_{replace}(G,[a_{i-1}\ \ a_i])$. Combining these two together, we have the second term in the theorem as $\max_{r \in R^K_{a_0}}\{g(r)\} + \sum_{i=1}^{m}\mathcal{D}_{replace}(G,[a_{i-1}\ a_i])$ ■

***Theorem** 4:* For any network G and any RNPU algorithm A, $time(T_{long}) \leq$

$$\max_{v \in V, r^{new}(v)=(a_m \cdots a_0\ s_{K-1} \cdots s_0)} \{\min_{r \in R_{a_0}(G,A)}\{g(r)\} + \sum_{i=1}^{m}\mathcal{D}_{replace}(G,[a_{i-1}\ a_i])\}$$

Proof: For RNPU algorithms, once a_m receives the new path from a_0, it is guaranteed to converge, because this new path also contains root cause notification, which allows a_m to discard any invalid path regardless of its length. Therefore, we only need to calculate when the new path arrives at a_m. For a_0, it converges when it receives the first message, which happens at $\min_{r \in R_{a_0}(G,A)}\{g(r)\}$. After then, the new path takes time up to $\sum_{i=1}^{m}\mathcal{D}(G_A,[a_{i-1}\ a_i])$ to reach a_m. The network's convergence time can be obtained by taking the maximum over all nodes. ■

In this section we have derived the upper bound of convergence time for both T_{down} and T_{long} events in general cases. Given a particular algorithm A, topology $G(V,E)$, and delay model $\mathcal{D}(G,[u\ \ v])$, we can use the general theorems to obtain detail results for each individual case.

IV. U Model and Results

To obtain convergence time from the framework, we need to find $\mathcal{D}(G,[u\ v])$. Generally $\mathcal{D}(G,u,v)$ includes MRAI delay, transmission delay, link propagation delay, queueing delay and processing delay. The MRAI delay is bounded by the MRAI timer, $\mathcal{M}$, usually configured with the default value of 30 seconds with a random jitter. In this paper we assume that MRAI timer is *exactly* $\mathcal{M}$ seconds without jitter; our results can be easily extended to consider jittered MRAI timer. We define $h(G,[u\ v])$ as the sum of all the delays except MRAI delay.

The U model, commonly used in the literature, assigns a fixed upper bound, h, for each $h(G,[u\ v])$. Therefore, depending on the algorithm, either $\mathcal{D} = \mathcal{M}+h$ or $\mathcal{D} = h$, regardless of the topology and node. In this section, we provide T_{down} and T_{long} convergence time bound under U model, and the results are summarized in Fig. 2 and Fig. 3.

$h(G,[u\ v])$	sum of all delays except MRAI delay
h	a fixed upper bound of $h(G,[u\ v])$
$\mathcal{M}$	Minimum Route Advertisement Interval

A. T_{down} Results

A	$time(T_{down})$
SPVP	$(N-1)\cdot \quad (\mathcal{M}+h)$
*SPVP-AS	$(N-degree(G,0))\cdot \quad (\mathcal{M}+h)$
SPVP-GF	$(N-1)\cdot \quad h$
SPVP-RCN	$Distance(G,0)\cdot \quad h$

Fig. 2. T_{down} convergence results under U model. The SPVP-AS result was previously unavailable.

***Corollary** 1:* For any network G and any LNPU algorithm A, under U model,

$$time(T_{down}) \leq \mathcal{D} \cdot \max_{r \in R(G,A)}\{length(r)\}$$

Proof: Since $\mathcal{D}(G,[u\ v]) = \mathcal{D}$, a path r's lifetime becomes $f(r) = \mathcal{D} \cdot length(r)$. The corollary directly follows Theorem 1. ■

Considering all possible topologies, the longest path at most can include every node once, therefore $\max_{r \in R(G,A)}\{length(r)\} = N-1$. Different from SPVP and SPVP-GF, SPVP-AS has an additional constraint. Before the failure, node 0's direct neighbor v has a direct path $(v\ \ 0)$. During the convergence, the first message v receives is a withdrawal from node 0. As a result of assertion checking, v will never choose nor propagate any path containing node 0's other direct neighbors. Therefore, any invalid path during T_{down} convergence can have at most one of node 0's direct neighbors. Similarly, in T_{long} convergence, any invalid path can have at most one of node c's direct neighbors in V_A. Thus, for SPVP-AS, $\max_{r \in R(G,A)}\{length(r)\} = N - degree(G,0)$. For SPVP and SPVP-AS, $\mathcal{D} = \mathcal{M} + h$, while SPVP-GF has $\mathcal{D} = h$ because the "flushing" withdrawals are not delayed by the MRAI timer.

***Corollary** 2:* For any network G and any RNPU algorithm A, under U model,

$$time(T_{down}) \leq h \cdot Distance(G,0)$$

Proof: For RNPU algorithms, the first update is a withdrawal and all subsequent updates are also withdrawals. Therefore, the MRAI timer does not apply and $\mathcal{D} = h$. By definition,

$\max_{v \in V}\{\min_{r \in R_v(G,A)}\{length(r)\}\} = Distance(G, 0)$. ■

B. T_{long} Results

For T_{long} events, the lifetime of r is $g(r) = \mathcal{D} \cdot (length(r) - J)$ when the failure link $[c\ b]$ is J hops away from node 0.

First consider LNPU algorithms. $\mathcal{D} = \mathcal{M} + h$ for SPVP and SPVP-AS and $\mathcal{D} = h$ for SPVP-GF. The first term of Theorem 3, $\max_{r \in R^{K+m}_{a_m}(G,A)}\{g(r)\}$, becomes $\mathcal{D} \cdot smaller\{K+m-J,\ \max_{r \in R_{a_m}(G,A)}\{length(r)\} - J\}$. In SPVP and SPVP-GF, $\max_{r \in R_{a_m}(G,A)}\{length(r)\} - J = |V_A| - 1$, while in SPVP-AS, it is $|V_A| - degree(G_A, c) - 1$. Similarly, the first half of the second term in Theorem 3, $\max_{r \in R^{K}_{a_0}(G,A)}\{g(r)\}$, equals to $\mathcal{D} \cdot smaller\{K - J, |V_A| - 1\}$ for SPVP and SPVP-GF, and $\mathcal{D} \cdot smaller\{K - J, |V_A| - degree(G_A, c) - 1\}$ for SPVP-AS. And $\sum_{i=1}^{m} \mathcal{D}_{replace}(G, a_{i-1}, a_i) = (\mathcal{M} + h) \cdot m$ for all LNPU algorithms. Sum these terms and take the upper bound over all nodes. Note $Distance(G', 0)$ is the upper bound for $K + m$, and $diameter(G_A)$ for m.

For RNPU algorithms, the similar procedure can be repeated, with $\mathcal{D} = h$ and $\mathcal{D}_{replace} = \mathcal{M} + h$. Note $diameter(G_A)$ is the upper bound of both $distance(G, a_0, c)$ and m.

V. Q Model and Results

The limitation of U model is that it uses the same $h(G, [u\ v])$ for all nodes, but in fact different nodes may have different $h(G, [u\ v])$. The U model not only gives coarse estimate of the convergence time, but also fails to reveal important relationships between the convergence time and the network topology. For example, in section VI, we show that in clique topologies, U model based analysis of SPVP-GF [6] does not explain simulation results from SSFNET [7] and fails to match the behavior of Internet BGP implementations. This section introduces the Q model, which incorporates a queueing delay estimate into $h(G, [u\ v])$ and reflects BGP implementations better. With the Q model, we can obtain tighter bounds of convergence time and new insights on topology's impact.

A. Queueing Delay

ld	upper bound of the sum of transmission and propagation delay on one link
p_{max}	maximum message processing time
$h(G, [u\ v])$	sum of ld, queueing delay and processing delay

The Q model uses ld to denote the upper bound on the sum of link delay, transmission delay, and any delay due to retransmitting lost packets. In other words, an update sent by node u will be received by node v within time ld. The Q model assumes a node v processes update messages in FIFO order. If a message arrives while the processor is occupied, the message is placed in an FIFO queue. The queueing delay depends on the number of messages in the FIFO queue at the moment a message arrives. Once the message gets to the processor, it will be fully processed in $[p_{min}, p_{max}]$ seconds. Thus $h(G, [u\ v])$ equals to the sum of ld, queueing delay and processing delay.

If the message arrival rate is persistently higher than p_{max}, the queue will increase and result in very long delays [3]. However, in reality, this is rare because today's router hardware reduces p_{max}, and MRAI timer restricts the message sending rate. In particular, the MRAI timer (see Section II) ensures that two *announcements sent* by node u to v must be separated by at least $\mathcal{M}$ seconds. Since withdrawal messages are not restricted by the MRAI timer, and our algorithms do not send duplicate updates, during any period of $\mathcal{M}$ seconds, the most updates u can send to v is a sequence of $withdrawal$, $announcement$, $withdrawal$. This observation allows us to obtain a bound $h(G, [u\ v])$.

***Assumption** 1:* During any $\mathcal{M}$ second interval, node u can $send$ at most 3 updates to node v.

***Corollary** 3:* During any $(\mathcal{M} - ld)$ interval, node v can receive at most 3 updates from node u.

Proof: Consider any sequence of 4 updates from u to v, assume the first one is sent at time t_1, received at t'_1, and the last one is sent at t_4, received at t'_4. Assumption 1 ensures that $t_4 - t1 > \mathcal{M}$, and since the link delay is between $(0, ld]$, we have $t_4 < t'_4 \leq t_4 + ld$ and $t_1 < t'_1 \leq t_1 + ld$. Therefore, $t'_4 - t'_1 > \mathcal{M} - ld$. ■

***Lemma** 4:* In the Q model, if $\mathcal{M} - ld > 3 \cdot degree(G, v) \cdot p_{max}$, then at any moment t, there are at most $3 \cdot degree(G, v)$ messages in v's queue.

Proof: For a base case, at time $t = 0$, the queue starts with no messages. During the first $(\mathcal{M} - ld)$ seconds, at most 3 messages can be received from each neighbor according to Corollary 3.

Now suppose the Lemma is true for time period $[0, i \cdot (\mathcal{M} - ld))$, $i = 1, 2, 3...$, we examine the queue at any moment t between $[i \cdot (\mathcal{M} - ld), (i + 1) \cdot (\mathcal{M} - ld))$. At time $t' = t - (\mathcal{M} - ld)$, there are at most $3 \cdot degree(G, v)$ messages in the queue since t' falls in $[0, i \cdot (\mathcal{M} - ld))$. All these messages are processed within $3 \cdot degree(G, v) \cdot p_{max} < \mathcal{M} - ld$ seconds, therefore by time t, they have all left the queue. The number of messages that can arrive within $[t', t]$ is no more than $3 \cdot degree(G, v)$, thus the

A	T_{long} results under U model
*SPVP	$(\mathcal{M}+h)\cdot smaller\{Distance(G',0)-J, \lvert V_A\rvert + diameter(G_A)-1\}$
*SPVP-AS	$(\mathcal{M}+h)\cdot smaller\{Distance(G',0)-J, \lvert V_A\rvert + diameter(G_A)-degree(G_A,c)\}$
*SPVP-GF	$\mathcal{M}\cdot diameter(G_A)+h\cdot smaller\{Distance(G',0)-J, \lvert V_A\rvert + diameter(G_A)-1\}$
SPVP-RCN	$(\mathcal{M}+2h)\cdot diameter(G_A)$

Fig. 3. T_{long} results under U model

A	T_{down} results under Q model	
*SPVP	$(N-1)\cdot(\mathcal{M}+ld)+$	$3p_{max}\cdot(\lvert E\rvert - degree(G,0))$
*SPVP-AS	$(N-degree(G,0))\cdot(\mathcal{M}+ld)+$	$3p_{max}\cdot(\lvert E\rvert - \lvert E^0\rvert + Degree(G^0))$
*SPVP-GF	$(N-1)\cdot ld+$	$3p_{max}\cdot(\lvert E\rvert - degree(G,0))$
*SPVP-RCN	$Distance(G,0)\cdot ld+$	$p_{max}\cdot Distance(G,0)$

Fig. 4. Tighter bounds for T_{down} under Q model

hypothesis holds for $(i+1)$. ■

***Theorem** 5:* In the Q model, if $\mathcal{M} - ld > 3\cdot degree(G,v)\cdot p_{max}$, then $h(G,[u\ v]) \le 3degree(G,v)\cdot p_{max}+ld$.

Proof: The theorem follows directly from Lemma 4. ■

$\mathcal{M} > 3\cdot degree(G,v)\cdot p_{max}+ld$ is a sufficient condition to provide an upper bound for $h(G,[u\ v])$ and we assume this condition is true in the rest of the paper. Note that in practice, the default setting of $\mathcal{M}$ is 30 seconds, and ld in the Internet is at most several hundreds of milliseconds. For a loose upper bound of $p_{max}=0.01$, this assumption is true for topologies with $degree(G,v)\approx 1000$. In other words, our assumption allows each router to have up to 1000 direct neighbors. The assumption of $\mathcal{M} > 3\cdot degree(G,v)\cdot p_{max}+ld$ holds for nearly all practical networks.

B. Delay bounds under Q model

By using per node $h(G,[u\ v]) = ld + 3\cdot p_{max}\cdot degree(G,v)$ instead of a fixed number h, the Q model provides tighter bound for each convergence algorithm we have studied, and more insights of how topology affects convergence time.

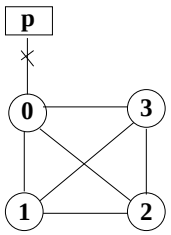
Theorem 1 shows that the T_{down} convergence time of LNPU algorithms is $time(T_{down}) \le \max_{r\in R(G,A)}\{f(r)\}$. Under Q model, the lifetime of path $r=(v_l\ v_{l-1}\cdots v_0)$ is $f(r)=\sum_{i=1}^{l}(\mathcal{M}+ld+3p_{max}\cdot degree(G,v_i))$ for SPVP and SPVP-AS, and $f(r)=\sum_{i=1}^{l}(ld+3p_{max}\cdot degree(G,v_i))$ for SPVP-GF.

Since SPVP and SPVP-GF do not restrict $R(G,A)$ (Section III), in the worst case an invalid path can include every node. Therefore, for SPVP, $time(T_{down}) \le \sum_{i=1}^{N-1}(\mathcal{M}+ld+3p_{max}\cdot degree(G,i)) = (N-1)\cdot(\mathcal{M}+ld)+3p_{max}\cdot\sum_{i=1}^{N-1} degree(G,i) = (N-1)\cdot(\mathcal{M}+ld)+3p_{max}\cdot(\lvert E\rvert - degree(G,0))$; for SPVP-GF, $time(Q,G,SPVP-GF,T_{down}) \le \sum_{i=1}^{N-1}(ld+3p_{max}\cdot degree(G,i)) = (N-1)\cdot ld+3p_{max}\cdot(\lvert E\rvert - degree(G,0))$. SPVP-AS restricts the invalid path to include only one of node 0's direct neighbors, therefore $time(T_{down}) \le \sum_{v,where\ [v\ 0]\notin G}(\mathcal{M}+ld+3p_{max}\cdot degree(G,v))+(\mathcal{M}+ld+3p_{max}\cdot Degree(G^0) = (N-degree(G,0))\cdot(\mathcal{M}+ld)+3p_{max}\cdot(\lvert E\rvert - \lvert E^0\rvert + Degree(G^0))$, where $G^0=(V^0,E^0)$ is the subgraph consisting of node 0 and its direct neighbors, and $Degree(G^0)$ is the maximum degree of any $u\in V^0$ by definition. These results are summarized in Figure 4.

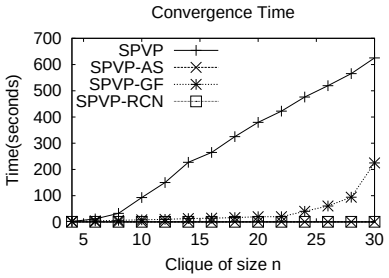
The results under U model (Fig. 2) imply the convergence time is proportional to the number of nodes in the network for SPVP, SPVP-GF, and SPVP-AS. However, the Q model reveals that each algorithm also has a term proportional to the number of links in the network, and this is an important hint in understanding the simulation results (Section VI).

For SPVP-RCN, since the first message received makes the receiver converged, queueing delay does not affect the convergence time. Thus $h(G,u,v) \le ld + p_{max}$ holds, and according to Theorem 2, $time(T_{down}) \le Distance(G,0)\cdot(ld+p_{max})$. Compared with the results of SPVP, SPVP-AS, and SPVP-GF, RCN's advantage is more pronounced than in Q model.

For T_{long} convergence, the improvements of convergence algorithms are mainly on helping a_0 converge faster. This process is similar to T_{down} thus we can obtain similarly tighter delay bounds for this process under Q model. For brevity, the detailed T_{long} results are not presented in this section, but they can be found in Appendix.

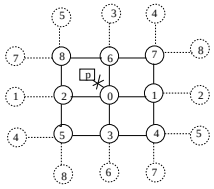


(a) $Clique(4)$

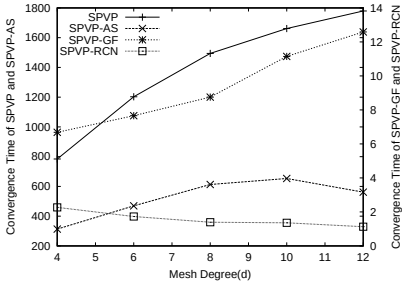


(b) Convergence Time

Fig. 5. T_{down} in $Clique(n)$



(a) $Mesh(3,4)$



(b) $Mesh(10,d)$

Fig. 6. T_{down} in $Mesh(n,d)$

A	T_{down} for $Clique(n)$
SPVP‘	$(n-1)(\mathcal{M}+ld)+ \quad 3p_{max}\cdot(n-1)^2$
AS	$(ld+\mathcal{M})+ \quad 3p_{max}\cdot(n-1)$
GF	$(n-1)\cdot ld+ \quad 3p_{max}\cdot(n-1)^2$
RCN	$ld+ \quad p_{max}$

(c) $time(T_{down})$ for $Clique(n)$

A	$Mesh(n,d)$
SPVP	$(n^2-1)(ld+\mathcal{M})+3d(n^2-1)p_{max})$
SPVP-AS	$(n^2-d)(ld+\mathcal{M}+3dp_{max}))$
SPVP-GF	$(n-1)ld+3d(n^2-1)p_{max}$
SPVP-RCN	$Distance(Mesh(n,d),0)(ld+p_{max})$

(c) $Time(T_{down})$ for $Mesh(n,d)$ under Q model

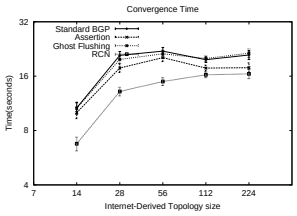
VI. Simulation Results

We conducted simulations using SSFNET [8]. The simulator uses FIFO queue for incoming messages, which makes it suitable to verify our analytical results under Q model. In simulations, we set $\mathcal{M}=30s$, $ld=0.002s$, $p_{min}=0.1s$ and $p_{max}=0.5s$. The results presented are averaged over multiple simulation runs. Although our analysis provides only the upper bound of convergence time, the insights from it help understand the simulation results that are otherwise not easy to comprehend. For brevity, some of the simulation results are presented in Appendix instead.

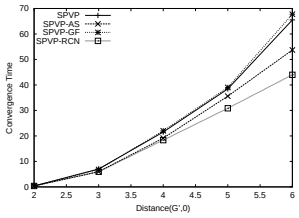
A. T_{down}

A $Clique(n)$ is a full-mesh of n nodes, which is commonly used in literature([1], [3], [14], [6], [7]) to evaluate different protocols' convergence properties. Figure 5 shows a sample topology, the Q model results, and simulation results of $Clique(n)$. The simulation results are consistent with our Q model results. In particular, for SPVP-GF, $time(T_{down})=(n-1)\cdot ld+3p_{max}\cdot(n-1)^2$ helps explain the abrupt increase when $n>20$, for which the U model result (Figure 2) would offer no explanation.

A $Mesh(n,d)$, is a 2-dimensional n by n grid, whose nodes have the same degree of d. Figure 6 shows a sample $Mesh(n,d)$ topology, Q model results, and simulation results for $n=10$ and d varies. Due to the orders of magnitude difference in the results, we use both the left and right Y axes. As d increases, the convergence time of $SPVP$ and $SPVP-GF$ increases, $SPVP-RCN$ decreases, and $SPVP-AS$ increases first but decreases later. These are consistent with the Q model results, while the U model (Figure 2) would have expected the convergence time fixed since the network size $N=100$ does not change.



(a) Net. Size(log-log)



(b) Distance(G',0) in 110-node topology

Fig. 7. T_{long} in Internet-like Topologies.

B. T_{long}

Prior to this work, there is a question about the T_{long} convergence time that have not been answered. Early Internet experiments [1] claimed that T_{long} and T_{down} belong to the same type of events, because they have similar convergence time due to path exploration. How-

ever, later algorithms such as SPVP-RCN and SPVP-GF, which improves T_{down} significantly by reducing path exploration, can only improve T_{long} modestly in simulations. Our analysis result explains why.

Figure 7(a) shows the averaged T_{long} convergence time versus the network size N in some Internet-like topologies. The results are average over various origin nodes and failure links, while $J = 1$ is kept. It is worth to note that SPVP performs well even in large network size, and none of SPVP-AS, SPVP-GF, or SPVP-RCN provides significant improvement. The analytical results of T_{long} delay under Q model is available presented in the Appendix. They are similar to the results under U model (Figure 3) in that the dominant factor in T_{long} convergence time is $Distance(G', 0)$ (Figure 7(b)). $Distance(G', 0)$ is usually a small value in a well connected network, e.g., ≤ 6 in our simulation topologies. Therefore, the room for improvement by any algorithm is far less than that in T_{down} event. In the real Internet, $Distance(G', 0)$ is likely to be a little bit more than 10, a relatively small value. Previous experiments [1] injected synthesized backup path with length around 30, which artificially increased the $Distance(G', 0)$ about 3 times, resulting in very long T_{long} convergence time.

VII. Related Work

There are several previous efforts in analyzing convergence delay in BGP (or SPVP). Labovitz *et al.* [1] analyzed the T_{down} convergence delay bound by using a synchronous model of BGP and observed that $Clique(n)$'s convergence time is bounded by $(n-1)\mathcal{M}$ seconds. Further analysis by Labovitz *et al.* in [2] showed that T_{down} convergence delay is upper bounded by $(p \cdot \mathcal{M})$, where p is the length of the longest possible backup path. The above results were obtained by ignoring the routing message queueing delay. Obradovic [15] developed a real-time BGP model which takes into account an edge delay similar to the definition of $\mathcal{D}$. Based on this real-time model, the author showed that the T_{down} convergence time bound for the shortest-path-first policy is ωp where p is defined above and ω the largest edge delay. The author did not specify how to calculate the edge delay, nor take into account the delay due to $MRAI$ timer. Our analytical framework is more general than these three works, and provides the T_{long} analysis results which is missing in the above works. Our Q model also provides more accurate and insightful results.

The analysis of both Ghost Flushing [6] and RCN [7] uses the U delay model. Analysis with Q model can provide tighter delay bounds than that provided by these two works. In addition, our general analytical framework allows us to provides T_{long} results for SPVP-GF, which were missing previously.

Simulation study using SSFNET by Griffin *et al.* [3] found that for each network topology there is an optimal $\mathcal{M}$ during which messages received from each neighbor can be "consumed". Our work provides a sufficient condition under which the messages can be consumed (Theorem 5).

VIII. Conclusion and Future Work

This paper presents the first general analytical framework for analyzing the convergence delay bounds of path vector routing protocols. For the first time we provide T_{long} results for SPVP, SPVP-AS, SPVP-GF, and T_{down} results for SPVP-AS. We also develop a new delay model, the Q model, which takes into account of message queueing delay, and provides not only tighter delay bounds, but also new insights into how topology affects the convergence delay.

In developing our analytical framework, we made several simplifying assumptions. In the future, we plan to examine the cases where convergence of *multiple* destination prefixes overlap with each other in time, especially when the MRAI timer is implemented on a per-peer instead of a per-destination basis. Furthermore, in our SPVP model, each AS is modeled as a single node, but the "processing delay" of a multi-router AS may be quite different from a single node. Obtaining a more accurate model for each AS represents a research challenge on its own.

References

[1] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian, "Delayed Internet Routing Convergence," in *Proceedings of ACM Sigcomm*, August 2000.

[2] C. Labovitz, R. Wattenhofer, S. Venkatachary, and A. Ahuja, "The Impact of Internet Policy and Topology on Delayed Routing Convergence," in *Proceedings of the IEEE INFOCOM*, April 2001.

[3] T. Griffin and B. Premore, "An Experimental Analysis of BGP Convergence Time," in *Proceedings of ICNP*, November 2001.

[4] D. Pei, X. Zhao, L. Wang, D. Massey, A. Mankin, F. S. Wu, and L. Zhang, "Improving BGP Convergence Through Assertions Approach," in *Proceedings of the IEEE INFOCOM*, June 2002.

[5] J. Luo, J. Xie, R. Hao, and X. Li, "An Approach to Accelerate Convergence for Path Vector Protocol," in *Proceedings of IEEE Globecom*, Nov. 2002.

[6] A. Bremler-Barr, Y. Afek, and S. Schwarz, "Improved BGP Convergence via Ghost Flushing," in *Proceedings of the IEEE INFOCOM*, April 2003.

[7] D. Pei, M. Azuma, N. Nguyen, J. Chen, D. Massey, and L. Zhang, "BGP-RCN: Improving BGP Convergence Through Root Cause Notification," Tech. Rep. TR-030047, UCLA CSD, October 2003, http://www.cs.ucla.edu/ peidan/bgp-rcn-tr.pdf.

[8] "The SSFNET Project," http://www.ssfnet.org.
[9] X. A. Dimitropoulos and G. F. Riley, "Creating realistic bgp models," in *11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems(MOSCOTS)*, 2003.
[10] T. Griffin, F. B. Shepherd, and G. Wilfong, "The stable path problem and interdomain routing," *IEEE/ACM Transactions on Networks*, vol. 10, no. 2, 2002.
[11] T. Griffin and G. Wilfong, "A Safe Path Vector Protocol," in *Proceedings of IEEE INFOCOMM*, March 2000.
[12] Y. Rekhter and T. Li, "Border Gateway Protocol 4," RFC 1771, SRI Network Information Center, July 1995.
[13] J. Chandrashekar, Z. Duan, Z.-L. Zhang, and J. Krasky, "Limiting path exploration in path vector protocols," Tech. Rep., University of Minnesota, 2003.
[14] Z. Mao, R. Govindan, G. Varghese, and R. Katz, "Route Flap Damping Exacerbates Internet Routing Convergence," in *Proceedings of ACM Sigcomm*, August 2002.
[15] D. Obradovic, "Real-time Model and Convergence Time of BGP," in *Proceedings of the IEEE INFOCOM*, June 2002.
[16] Y. Rekhter, T. Li, and S. Hares, "Border Gateway Protocol 4," http://www.ietf.org/internet-drafts/draft-ietf-idr-bgp4-22.txt, Oct 2003.

APPENDIX

A. T_{long} results under Q model

$\mathcal{D}_{replace}(G,[a_{i-1}\ \ a_i]) = \mathcal{M} + ld + 3 \cdot degree(G_A, a_i) \leq \mathcal{M} + ld + 3 \cdot Degree(G_A)$. Thus $\sum_{i=1}^{m} \mathcal{D}_{replace}(G_A,[a_{i-1}\ \ a_i]) \leq m \cdot (\mathcal{M} + ld + 3 \cdot Degree(V_A)) \leq diameter(G_A) \cdot (\mathcal{M} + ld + 3 \cdot Degree(V_A))$. This applies to both LNRU and RNRU algorithms.

Similar to T_{down}, under Q model the lifetime of path $r = (v_l\ v_{l-1} \cdots v_0\ t_{J-1} \cdots t_0)$ is $g(r) = \sum_{i=1}^{l}(\mathcal{M} + ld + 3p_{max} \cdot degree(G, v_i))$ for SPVP and SPVP-AS, and $g(r) = \sum_{i=1}^{l}(ld + 3p_{max} \cdot degree(G, v_i))$ for SPVP-GF.

Since SPVP and SPVP-GF do not restrict $R(G, A)$ (Section III), in the worst case an invalid path can include every node. Therefore, for SPVP, $max_{r \in R(G,A)}\{g(r)\} \leq \sum_{v \in G_A \wedge v \neq c}(\mathcal{M} + ld + 3p_{max} \cdot degree(G, v)) = (|V_A| - 1) \cdot (\mathcal{M} + ld) + 3p_{max} \cdot \sum_{v \in G_A \wedge v \neq c} degree(G, v) = (|V_A| - 1) \cdot (\mathcal{M} + ld) + 3p_{max} \cdot (|E_A| - degree(G_A, c))$; for SPVP-GF, $max_{r \in R(G,A)}\{g(r)\} \leq \sum_{v \in G_A \wedge v \neq c}(ld + 3p_{max} \cdot degree(G, v)) = (|V_A| - 1) \cdot ld + 3p_{max} \cdot (|E_A| - degree(G_A, c))$. SPVP-AS restricts the invalid path to include only one of node c's direct neighbors, therefore $max_{r \in R(G,A)}\{g(r)\} \leq \sum_{v \in G_A \wedge [v\ c] \notin G}(\mathcal{M} + ld + 3p_{max} \cdot degree(G_A, v)) + (\mathcal{M} + ld + 3p_{max} \cdot Degree(G_A^c)) = (|V_A| - degree(G_A, c)) \cdot (\mathcal{M} + ld) + 3p_{max} \cdot (|E_A| - |E_A^c| + Degree(G_A^c))$, where $G_A^c = (V_A^c, E_A^c)$ is the subgraph of G_A consisting of node c and its direct neighbors, and $Degree(G_A^c)$ is the maximum degree of any $u \in V_A^c$ by definition. These results are summarized in Figure 8.

For SPVP, and SPVP-AS, $\mathcal{D}(G,[u\ \ v]) \leq \mathcal{M} + ld + 3p_{max} degree(G_A, v) \leq \mathcal{M} + ld + 3p_{max} Degree(G_A)$. For SPVP-GF, $\mathcal{D}(G,[u\ \ v]) \leq ld + 3p_{max} degree(G_A, v) \leq ld + 3p_{max} Degree(G_A)$. For SPVP-RCN, the first message carries the root cause, thus it's enough to remove the invalid paths and message queueing has no effect on $\mathcal{D}$ for SPVP-RCN. Thus $\mathcal{D}(G,[u\ v]) \leq ld + p_{max}$. We use $\mathcal{D}^{max}$ to denote the largest $\mathcal{D}$ in each case.

To get LNPU algorithms' results, we need to first find the two terms in the formula of Theorem 3, then take the larger one of these two terms, and then take the maximum over all the affected nodes. The first term of Theorem 3, $\max_{r \in R_{a_m}^{K+m}(G,A)}\{g(r)\}$, is less than or equal to $smaller\{\mathcal{D}^{max} \cdot (K + m - J), \max_{r \in R(G,A))}\{g(r)\}\}$. Similarly, the first half of the second term in Theorem 3, $\max_{r \in R_{a_0}^{K}(G,A)}\{g(r)\}$, is less or equal to $smaller\{\mathcal{D}^{max} \cdot (K - J), \max_{r \in R(G,A)}\{g(r)\}\}$. Sum these terms and take the upper bound over all nodes, and we get the results shown in Fig. 9. Note $Distance(G', 0)$ is the upper bound

A	$max_{r \in R(G,A)}\{g(r)\}$
*SPVP	$(\|V_A\|-1)\cdot(\mathcal{M}+ld)+ \quad 3p_{max}\cdot(\|E_A\|-degree(G_A,c))$
*SPVP-AS	$(V_A-degree(G_A,c))\cdot(\mathcal{M}+ld)+ \quad 3p_{max}\cdot(\|E_A\|-\|E_A^c\|+degree(G_A^c))$
*SPVP-GF	$(V_A-1)\cdot ld+ \quad 3p_{max}\cdot(\|E_A\|-degree(G_A,c))$

Fig. 8. Upper Bound of Life time $g(r)$ of any invalid path $r \in R(G, A)$ for LNPU algorithms

A	$time(T_{long}([c\ b])$
*SPVP	$smaller\{(\mathcal{M}+ld+3p_{max}Degree(G_A))\cdot(Distance(G',0)-J),$ $(\|V_A\|-1)\cdot(\mathcal{M}+ld)+3p_{max}\cdot(\|E_A\|-degree(G_A,c))+$ $(\mathcal{M}+ld+3p_{max}Degree(G_A))\cdot diameter(G_A)\}$
*SPVP-AS	$smaller\{(\mathcal{M}+ld+3p_{max}Degree(G_A))\cdot(Distance(G',0)-J),$ $(\|V_A\|-degree(G_A,c))\cdot(\mathcal{M}+ld)+3p_{max}\cdot(\|E_A\|-\|E_A^c\|+degree(G_A^c))+$ $(\mathcal{M}+ld+3p_{max}Degree(G_A))\cdot diameter(G_A)\}$
*SPVP-GF	$smaller\{\mathcal{M}\cdot diamter(G_A)+(ld+3p_{max}Degree(G_A))\cdot(Distance(G',0)-J),$ $(\|V_A\|-1)\cdot ld+3p_{max}\cdot(\|E_A\|-degree(G_A,c))+$ $(\mathcal{M}+ld+3p_{max}Degree(G_A))\cdot diameter(G_A))\}$
*SPVP-RCN	$diameter(G_A)\cdot(\mathcal{M}+2\cdot ld+p_{max}(1+3Degree(G_A)))$

Fig. 9. T_{long} under Q model

for $K+m$, and $diameter(G_A)$ for m.

For RNPU algorithms, According to Theorem 4, we have $\min_{r\in R_{a_0}(G,A)}\{g(r)\} \le distance(G,a_0,c)\cdot(p_{max}+ld) \le diameter(G_A)\cdot(p_{max}+ld)$.

B. More Explanation for Assumption 1

BGP RFC [12] specifies that only one $announcement$ can be $sent$ out with $\mathcal{M}$ seconds, now we show that more than *one* withdrawals can be sent within $\mathcal{M}$ seconds. We then argue that they signal no new information and should cause negligible processing delay if they do occur, thus Assumption 1 is a reasonable.

Since he MRAI timer doesn't apply to *withdrawals*. a node v can send out a withdrawal immediately after $r(v)$ changes to ϵ. Suppose the MRAI timer is turned on at $t = 100$ seconds, and no announcement can be sent out before $t = 130$ seconds. Now suppose $r(v) = \epsilon$ at $t = 105$ seconds, $r(v) \neq \epsilon$ at time $t = 110$ seconds, and $r(v) = \epsilon$ at time $t = 115$ seconds. The non-empty path at $t = 110$ cannot be sent out, but the two withdrawals are sent out at $t = 105$, and $t = 115$, respectively.

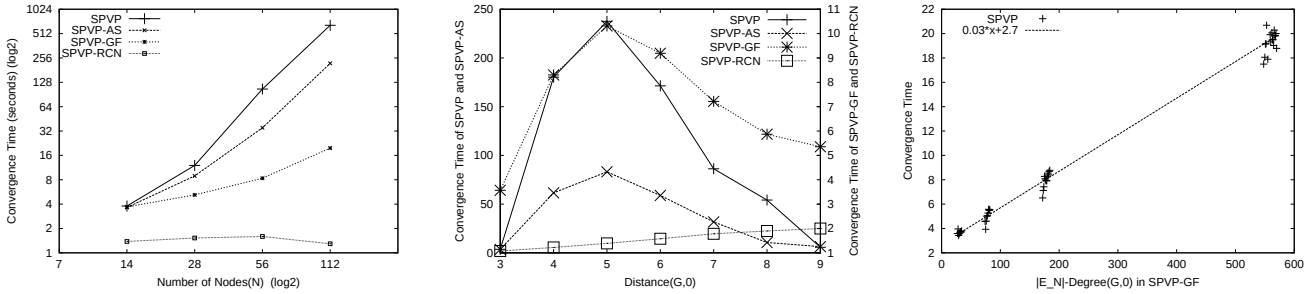
In the worst case, there can be $y = \lfloor \frac{\mathcal{M}}{p_{min}} \rfloor$ path changes generated within $\mathcal{M}$ seconds. On the other hand, according to the protocol definition, $r(v)$ cannot change from ϵ to ϵ, the maximal number of $r(v)$ changes to ϵ is $\lceil \frac{y}{2} \rceil$ withdrawals while the $r(v)$ change series is in a pattern of "$W, A, W, A, \cdots, W, A$", or "$A, W, A, W \cdots, A, W$", or "$W, A, W, A \cdots W, A, W$". In the worst case, each of the $\lceil \frac{y}{2} \rceil$ withdrawals can be sent out within $\mathcal{M}$ seconds.

However, it is not clear whether the above worst case will ever happen in reality. Some forms of duplicate update elimination can help remove part of consecutive withdrawals, and processing time for a $duplicate$ withdrawal might be neglectable compared to a normal update which needs policy checking and best-path recomputation, and a duplicate withdrawal's contribution to the queueing time is neglectable. In addition, a duplicate withdrawal cannot change the receiver's path. In another word, duplicate withdrawals are *ineffective*. Therefore, we have ignored duplicate withdrawals to simplify our analysis.

In addition, the latest BGP standard revision [16] has proposed to apply the MRAI time to withdrawal messages as well(called Withdrawal Rate Limiting, or WRATE). The duplicate withdrawal problem discussed above does not exist if WRATE is used. Furthermore, Assumption 1 should be revised to "During any $\mathcal{M}$ second interval, node u can $send$ at most 1 update to node v". Our Q model results in this paper can be easily revised with a different constant factor(replacing "3" with "1") to get the results with WRATE used.

C. Additional T_{down} Simulation Results

To further understand the T_{down} convergence, we simulate Internet-like topologies similar to those used in [7]. One node x was chosen as the only origin AS which advertised a destination prefix, and we simulated the T_{down} event by marking x down. We repeated simulations



(a) time vs N in different topologies

(b) time vs $Distance(G,0)$

(c) time vs $|E| - degree(G,0)$ in SPVP-GF

Fig. 10. Simulation Results for T_{down} Convergence Time in Internet-like topologies.

for each node in each topology. Q model analytical results in the second column of Figure 4 show that N, $|E| - degree(G,0)$ and $Distance(G,0)$ are important factors to different algorithms in the worst case, so we are interested in their impact as well as the convergence time comparison between these algorithms. From the network size (N) point of view, Figure 10(a) shows that the convergence time of SPVP-RCN and SPVP-GF are 2 to 3 order of magnitudes better than that of SPVP and SPVP-AS, given that fact that SPVP-RCN and SPVP-GF have removed the factor of $\mathcal{M}$ from T_{down} convergence. This performance difference is also confirmed by the results from $Distance(G,0)$ point of view in Figure 10(b). In addition, the linear increasing trend of SPVP-RCN can be easily explained by its worst case $Distance(G,0)(ld + p_{max})$.

We have shown in the worst case SPVP-GF's convergence time is $(N-1)ld + 3(|E| - Degree(G,0))p_{max}$, and Figure 10(c) shows that SPVP-GF's convergence time is indeed approximately proportional to $(|E| - Degree(G,0))$.

D. Additional T_{long} Simulation Results

To further understand T_{long} convergence of $SPVP$, we construct the $B - Clique(K,J,V)$ topology shown in Figure 12(a) to de-couple the factors in T_{long} convergence. Node 0 prepends its ID by J and K(as opposed to 1) times when announcing prefix p to neighbor 1 and V(which are in a $Clique(V)$ topology), respectively. Such "ID padding" in $B - Clique(K,J,V)$ allows us to change the different factors of $Distance(G',0) = K + 1$(backup path length), J(failure location) and $|V_A|$(the number of affected nodes) in the formula of T_{long} convergence independently. We got $time(T_{long}) \leq smaller\{K+1-J, V\} \cdot (3(V-1)p_{max} + \mathcal{M} + ld)$ by plugging K, J, V into the Q model results in Figure 9.

Figure 12 shows the simulation results when one parameter is varied, while the other two are fixed. The trend in these figures is very similar to those worst case analytical results shown in the titles of the corresponding figures. The trend in these figures is very similar to those worst case analytical results shown in the titles of the corresponding figures. We can see that SPVP's convergence time is approximately proportional to K when J and V are fixed, but if K is larger than V, increasing K have little impact, since there is no invalid paths to explore any more. On the other hand, the location of the failure, J, can reduce SPVP T_{long} convergence time linearly when K and V are fixed, by increasing the the smallest invalid path length. When K and J are fixed, V affected the longest invalid path length, and increasing of J will cause SPVP convergence time to increase.

To further confirm our analysis of the impact of J and $Distance(G',0)$, and we simulate SPVP T_{long} in $Mesh(n,4)$ topologies, while varying both n and J. The results are shown in Figure 11, which show the the same trend as in the second part of Figure 12(b) in all the $Mesh(n,4)$ topologies we simulated.

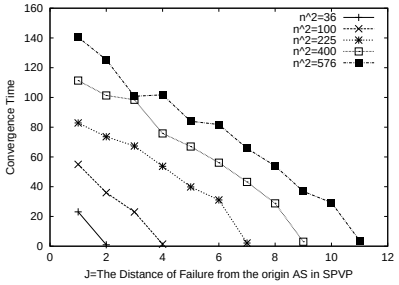
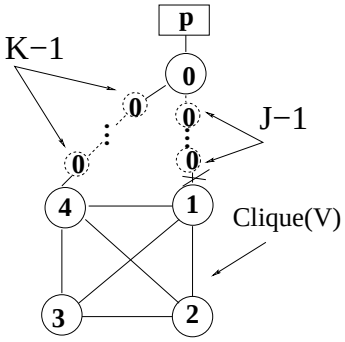
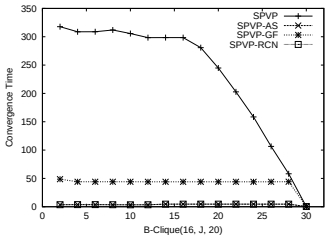


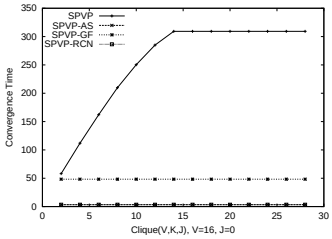
Fig. 11. SPVP T_{long} Convergence Time in Mesh(n,4) topologies, with different J



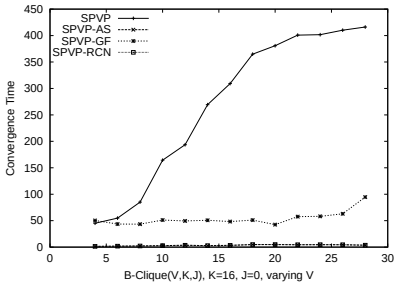
(a) B-Clique(K,J,V), whose T_{long} convergence time $smaller\{K+1-J, V\}(3(V-1)p_{max} + \mathcal{M} + ld)$



(b) K=30, V=16, $time(SPVP) \leq smaller\{31 - J, 16\}(45p_{max} + \mathcal{M} + ld)$



(c) J=1, V=16, $time(SPVP) \leq smaller\{K, 16\}(45p_{max} + \mathcal{M} + ld)$



(d) J=1, K=16,$time(SPVP) \leq smaller\{16, V\}(3(V-1)p_{max} + \mathcal{M} + ld)$

Fig. 12. T_{long} in B-Clique(K,J,V).