Routing in the Future Internet

Marcelo Yannuzzi

Graduate Course (Slideset 9)
Institute of Computer Science
University of the Republic (UdelaR)

September 5th 2012, Montevideo, Uruguay

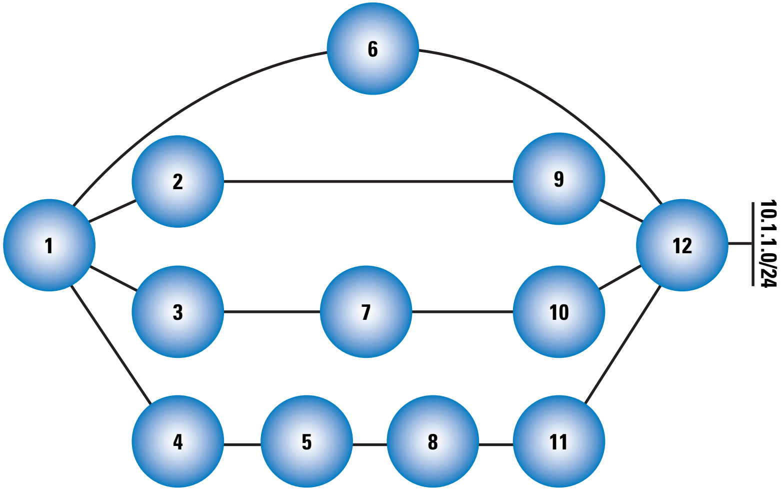
Department of Computer Architecture
Technical University of Catalonia (UPC), Spain

Institute of Computer Science
University of the Republic (UdelaR), Uruguay

Outline

Outline

Related Work



- Russ White, “Graph overlays on path-vectors: A Possible Next Step in BGP,” Internet Protocol Journal, Cisco vol. 8, no. 2, pp. 13–21, June 2005.

A two-fold challenge...

1. **The Theoretical Challenge:** from a computer science perspective, it is important to recall that the core strengths of link-state protocols are rooted in the computation and maintenance of a graph of the network. The theoretical challenge is that link state-like graphs cannot offer a consistent view of the forwarding paths of an internetwork subject to routing policies (... we will show this explicitly)
2. **The Practical Challenge:** from a practical viewpoint, the key challenge is to amalgamate the two different routing styles in a way that requires neither the wholesale replacement of BGP nor the development of upward-compatible extensions making BGP implementations more complex than they are today.

Assumptions

- The routes available in an internetwork are determined by the import and export policies of routing domains
- For each domain u, these policies can be split into 2 groups:
  - The first group consists of the application of filtering policies to ensure that the routing across domain u respects its policies (e.g., no transit; partial transit; transit through domain u is valley-free, etc.).
  - Once u enforces its transit policies, it can apply another group of policies to manage its traffic distribution, both inbound through its export policies and outbound through its import policies.

Assumptions (cont.)

- ...notice that the combination of filtering policies along with the traffic distribution policies independently applied by domains, is what ultimately determines the routing information available at each domain—thus conditioning the graphs that can be inferred by these latter.

Assumptions (cont.)

- We assume that the routes are composed of path vectors.
- Differently from BGP-based routing, where domains only export their best route for a destination, our interest is to analyze the graph representations that can be inferred by domains in a network that is subject to filtering policies and the traffic distribution policies locally applied by domains. Thus, we assume that, in principle, a domain can import and export **"all the routes"** compatible its filtering rules and its local traffic distribution policies.

Notice that, under these assumptions, the only possible pruning of the graphs inferred by domains will be due to the routing policies.

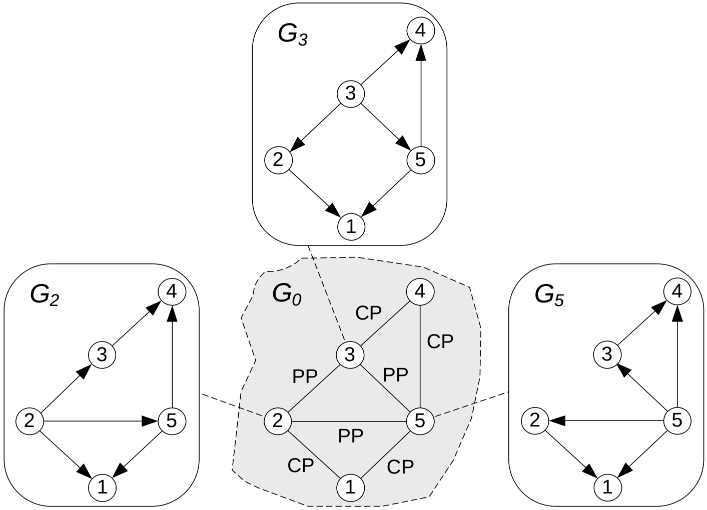
Notation

Let $G_0(V, E)$ denote the undirected AS graph representing the interconnectivity of the network.

Let $\mathcal{V}(G_0)$ represent the set of routing records resulting from the application of the Valley-Free Routing (VFR) policies on G_0. A set of routing records, $\mathcal{V}(G_0)$, organized in the form of the matrix shown in Slide 11, shall be referred to as the *Network Routing* (NR) table, and will be denoted as table $\mathcal{R} = \mathcal{V}(G_0)$.

The routing system shall be denoted as $\mathcal{S} = \langle G_0, \mathcal{R} \rangle$.

Different topological views of domains under VFR policies

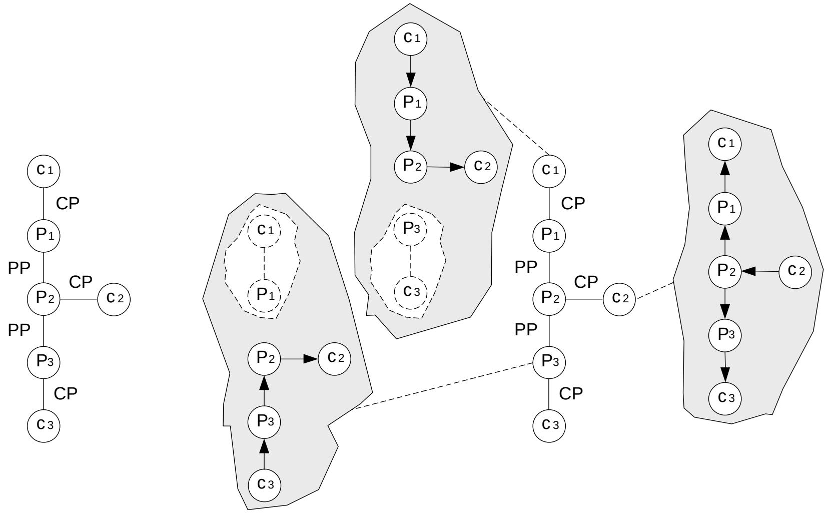


Network Routing table $\mathcal{R}$

Table: Network Routing (NR) table $\mathcal{R}$. The rows show the VF routes seen by each domain to the rest of domains in the network.

	1	2	3	4	5
1	-	[2] [5,2]	[2,3] [5,3]	[2,3,4] [5,4] [2,5,4] [5,3,4]	[5] [2,5]
2	[1] [5,1]	-	[3]	[3,4] [5,4]	[5]
3	[2,1] [5,1]	[2]	-	[4] [5,4]	[5]
4	[3,2,1] [5,1] [3,5,1] [5,2,1]	[3,2] [5,2]	[3] [5,3]	-	[5] [3,5]
5	[1] [2,1]	[2]	[3]	[4] [3,4]	-

Different topological views of domains under VFR policies



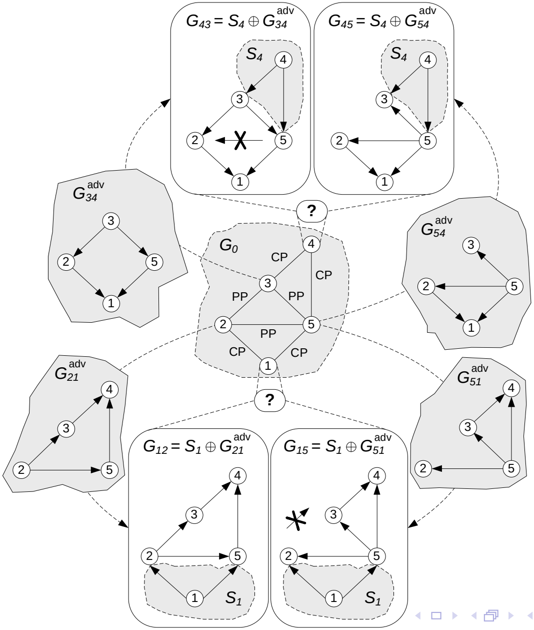
(a)

(b)

Paths on graphs under VFR policies

Let us now analyze to what extent the different topological views affect the availability of paths on the graphs inferred by domains ...

Conventional graphs under routing policies



$G_{43} = S_4 \oplus G_{34}^{adv}$
$G_{45} = S_4 \oplus G_{54}^{adv}$
S_4
S_4
G_{34}^{adv}
G_{54}^{adv}
G_0
CP
CP
PP
PP
PP
CP
CP
?
?
G_{21}^{adv}
G_{51}^{adv}
$G_{12} = S_1 \oplus G_{21}^{adv}$
$G_{15} = S_1 \oplus G_{51}^{adv}$
S_1
S_1
1
2
3
4
5

Paths on graphs under VFR policies

In summary ...

... when it comes to the state of the links, the routing information received by AS1 and AS4 is inconsistent.

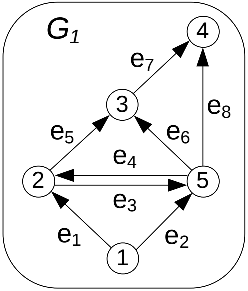
Therefore, the topology and paths admitted by the routing policies cannot be captured and maintained using a standard link-state database.[a]

Despite the inconsistencies, both AS1 and AS4 can still formally construct a graph of the network ...

[a] One of the main requirements in a link state-like paradigm is to warrant the consistency of the link state databases maintained by the nodes (domains) in the network. This is essential not only for constructing a consistent graph of the network, but also for finding paths on the graph.

The graph G_1 inferred by AS1

The graph inferred by AS1 from the valley-free routes contained in the first row of the NR table $\mathcal{R}_1$.



Paths on graphs under VFR policies

The interesting observation is that, although G_1 was inferred from $\mathcal{R}_1$, i.e., from the routes admitted by the routing policies, once constructed, the forwarding paths available on G_1 show inconsistencies with the paths allowed by the routing policies.

We now formalize this observation ...

Path inconsistencies – notice: “paths” not “edges”...

Theorem

Let v_i be a domain in a routing system $\mathcal{S}$ with $|V| > 2$ domains, and $\mathcal{R}_i$ be the corresponding row of v_i in table $\mathcal{R}$. Let G_i be the graph constructed by v_i from the path vectors contained in $\mathcal{R}_i$. For a path $\overrightarrow{p}$ to be in G_i, it is sufficient (but not necessary) that $\overrightarrow{p} \in \mathcal{R}_i$.

Path inconsistencies – notice: “paths” not “edges”...

Proof: The sufficiency is trivial, since if $\overrightarrow{p} \in \mathcal{R}_i$, then by Corollary 1 it holds that, $\forall \, (v_k, v_{k+1}) \subseteq \overrightarrow{p} \Rightarrow (v_k, v_{k+1}) \in G_i$. Therefore, $\overrightarrow{p} \subset G_i$.

To prove that it is not necessary, let us consider again G_1 in slide 41. Observe that G_1 admits the paths: [1,2,5,3], [1,5,2,3], [1,2,5,3,4], and [1,5,2,3,4], none of which is present in $\mathcal{R}_1$. Hence, there are paths in G_1 that are not present in $\mathcal{R}_1$. This completes the proof.

Main conclusion from the previous Theorem ...

$$\begin{tikzcd}
\text{From the routing information known by a domain } v_i \arrow[r] & v_i \text{ can construct a link state-like graph } G_i \text{ of the network} \arrow[d] \\
\boldsymbol{G_i} \textbf{ is not sufficient to reconstruct the paths known by } \boldsymbol{v_i} & \text{However, once } G_i \text{ is constructed} \arrow[l]
\end{tikzcd}$$

Possible alternatives ...

Distributing G_0

One option is that domains obtain the interconnectivity graph, G_0, directly from a source dedicated to its computation.

... however, the graph G_0 per se is inadequate for routing purposes, since it suffers from the same inconsistencies shown before. For instance, from the graph G_0 used as example in this slideset, it can be noted that many paths that are feasible in G_0, are actually not allowed by the routing policies; e.g., the path [1,2,5,3,4].

As a consequence, the conventional way of modelling an internetwork as an AS graph with the vertex-set representing domains and the edge-set the interconnection among these latter is inappropriate if the graph is to be used for routing purposes under routing policies.

Possible alternatives ... (cont.)

Obtaining G_0 and $AS_r(G_0)$

An alternative could be that domains obtain both G_0 and the AS relationships in G_0, $AS_r(G_0)$. Leaving aside issues such as which administrations would have the competence to compute $\langle G_0, AS_r(G_0)\rangle$, and the authority to advertise it in the Internet, there are at least two arguments against this approach.

First, Di Battista et al. showed that, the problem of computing the types of relationships between domains from the analysis of routing tables is NP-complete. Thus, inferring $AS_r(G_0)$ from the routing information requires approximation algorithms, which would entail a degree of inaccuracy in the tuple $\langle G_0, AS_r(G_0)\rangle$ advertised. In addition, Oliveira et al. (ToN 2010) as well Ager et al. (SIGCOMM 2012), have recently shown the infeasibility to obtain a complete AS-level topology from current data collection efforts....

Possible alternatives ... (cont.)

Obtaining G_0 and $AS_r(G_0)$ (cont.)

It seems that to have an accurate picture of $\langle G_0, AS_r(G_0)\rangle$, domains would be required to make public their commercial relationships, which is precisely the second and most important argument against this approach. Indeed, many providers will probably not support the proposal of making public and maintaining updated the tuple $\langle G_0, AS_r(G_0)\rangle$.

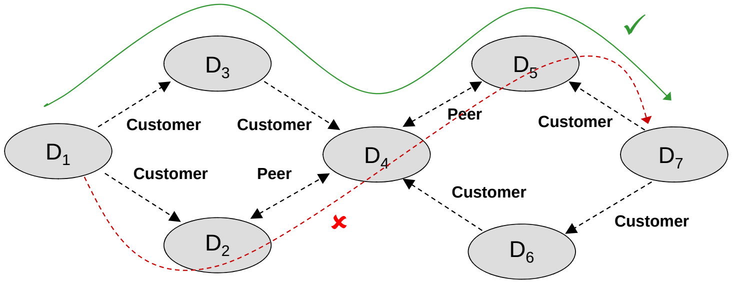
In summary

What is missing is ...

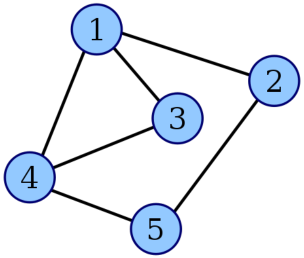
... a mechanism through which domains can independently construct a graph, without disclosing $AS_r(G_0)$. Such graph must accurately abstract the effects of routing policies, and capture the AS-level connectivity and paths admitted by the VFR policies in a consistent way.

Outline

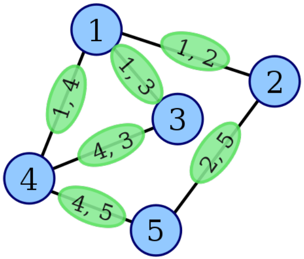
AS graphs under routing policies



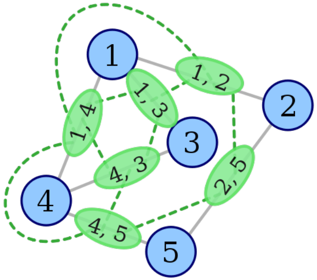
The line graph



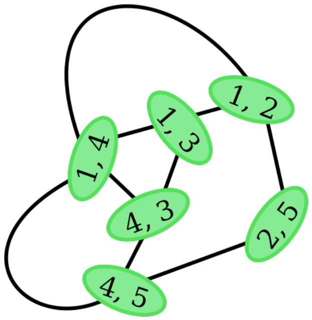
The line graph (cont.)



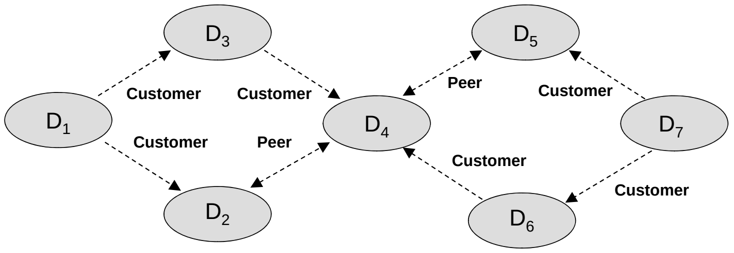
The line graph (cont.)

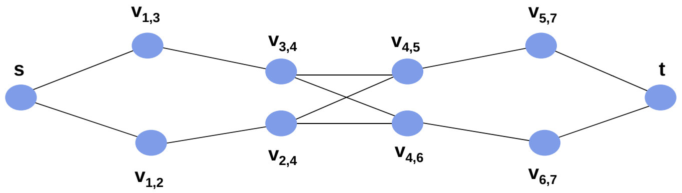


The line graph (cont.)

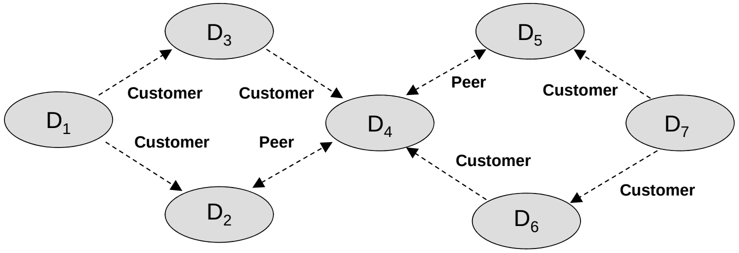


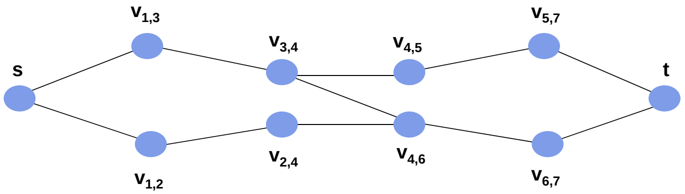
AS graphs under routing policies (cont.)



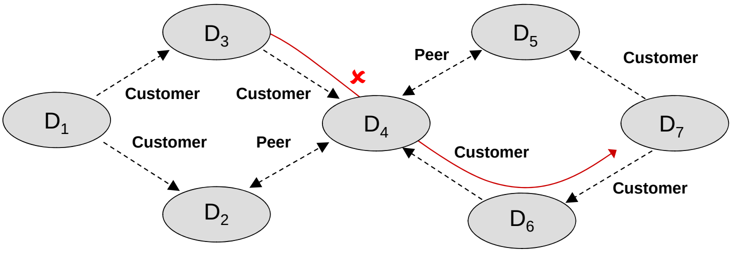


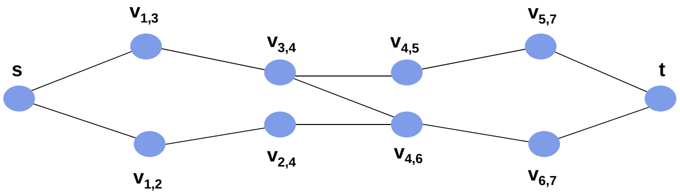
AS graphs under routing policies (cont.)



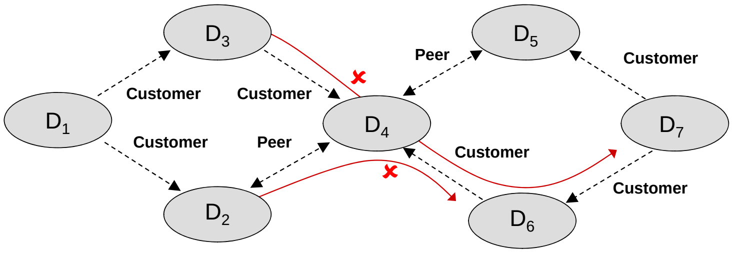


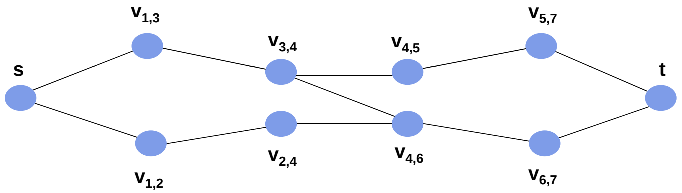
AS graphs under routing policies (cont.)



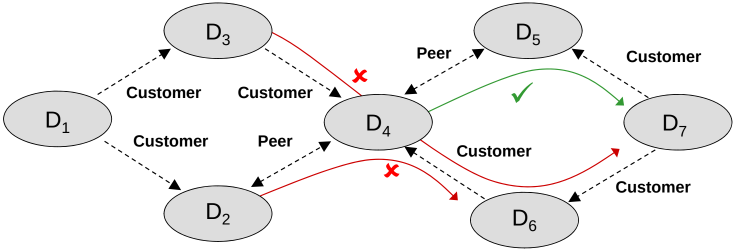


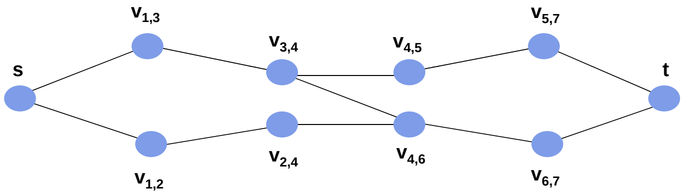
AS graphs under routing policies (cont.)



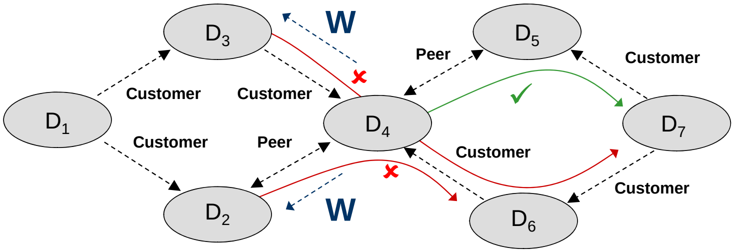


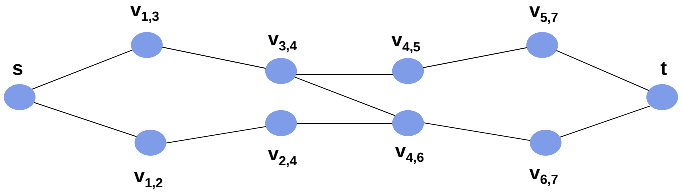
AS graphs under routing policies (cont.)



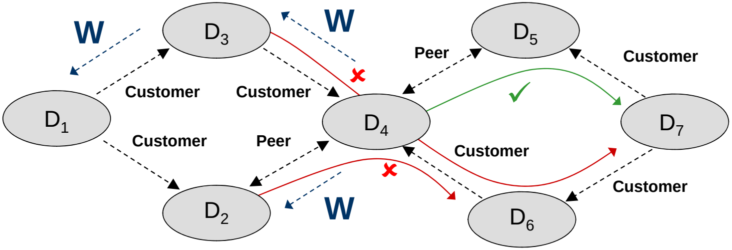


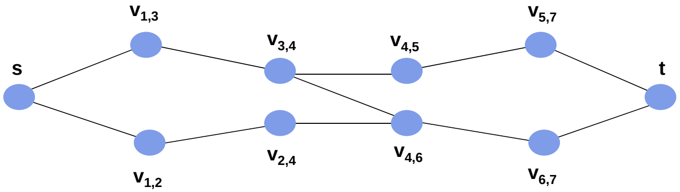
AS graphs under routing policies (cont.)



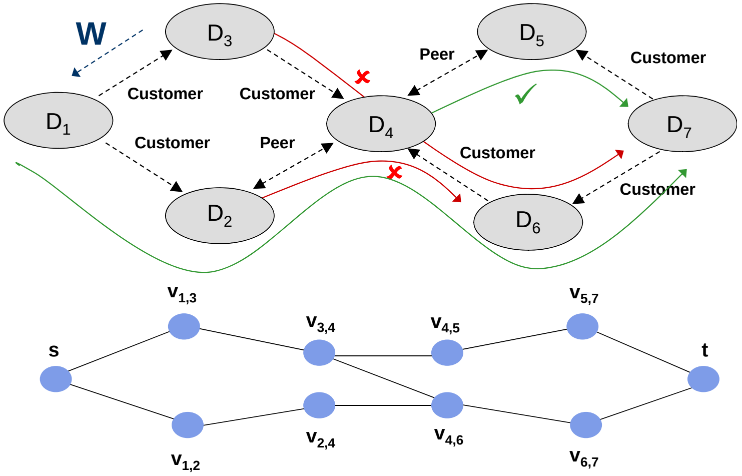


AS graphs under routing policies (cont.)

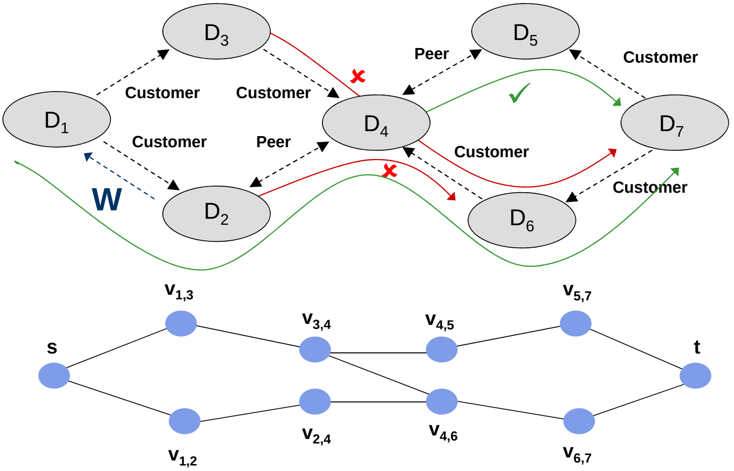




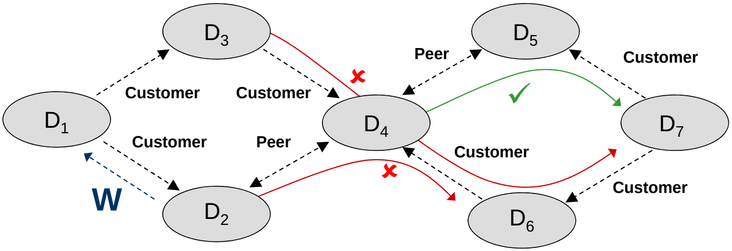
AS graphs under routing policies (cont.)

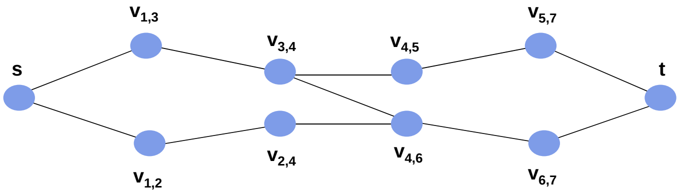


AS graphs under routing policies (cont.)

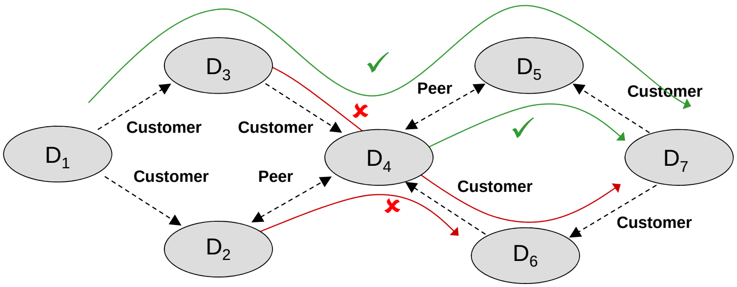


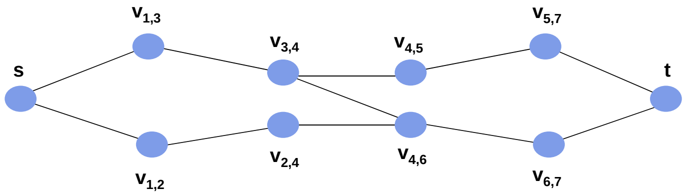
AS graphs under routing policies (cont.)



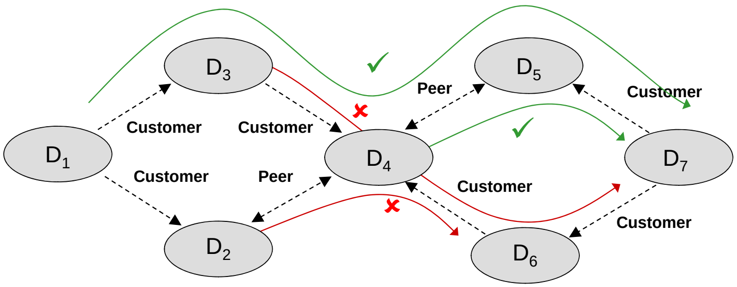


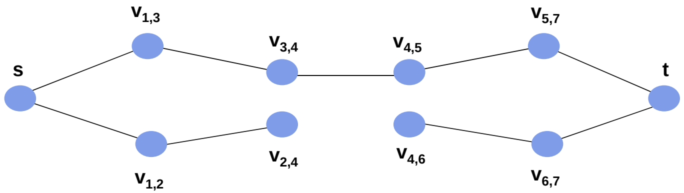
AS graphs under routing policies (cont.)





AS graphs under routing policies (cont.)





Outline

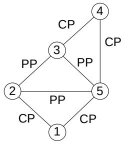
Path-State Graphs (PSGs)

Solving the inconsistencies

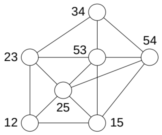
- Each domain can independently construct a PSG by collecting the different topological views advertised by its neighbors.
- PSGs can combine different pieces of topological information, which, once assembled, accurately capture the paths available in the network.
- The process for constructing the PSG-set is entirely distributed while preserving key properties, such as the assurance that the PSGs are both opaque[1] and loop-free.

[1] Opaqueness means that the AS relationships that can be inferred by a domain *u* are strictly those embedded in the routes received by *u*.

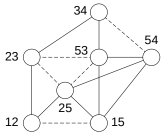
The intuition behind the construction of PSGs



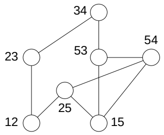
(a) AS graph G_0



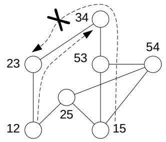
(b) Line graph $L(G_0)$



(c) Transit paths that are forbidden due to routing policies



(d) Pruned line graph $\hat{L}(G_0)$



(e) Forwarding loops

Path-State Graphs (PSGs) (cont.)

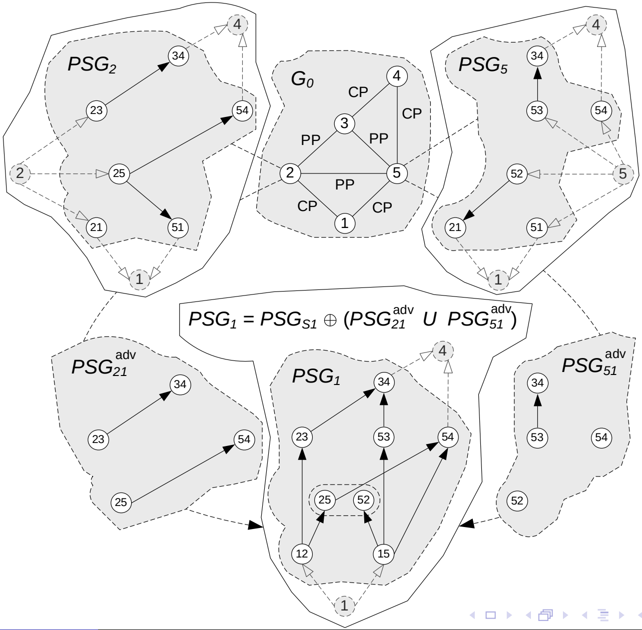
Solving the inconsistencies

On this basis, we seek to exploit the fact that a pruned line graph can effectively capture the link adjacencies permitted by the routing policies, and, at the same time, solve the two problems outlined before, i.e.:

- ensure the opaqueness of the graphs
- eliminate the possibility of forwarding loops

To achieve this, we propose the graph representations shown in the next slide.

Path-State Graphs (PSGs) (cont.)



PSG_2
G_0
PSG_5
CP
PP
$PSG_1 = PSG_{S1} \oplus (PSG_{21}^{adv} \cup PSG_{51}^{adv})$
PSG_{21}^{adv}
PSG_1
PSG_{51}^{adv}

Path-State Graphs (PSGs) (cont.)

Solving the inconsistencies

The key to obtain a compatible set of opaque graphs is that each domain independently computes a line graph, which can be locally shaped (e.g., pruned) before exporting it to a neighboring domain. Such graph shaping can be performed on a per neighbor basis, following the usual export policies of routing domains.

To fix ideas ...

- PSG_2, and PSG_5, are the directed line graphs of the digraphs G_2, and G_5; that is, $PSG_2 = L(G_2)$, and $PSG_5 = L(G_5)$.
- $PSG_{21}^{adv} = L(G_{21}^{adv})$, and $PSG_{51}^{adv} = L(G_{51}^{adv})$.
- PSG_{S_1} denotes the subscription PSG of AS1, which is the line graph of S_1 ($PSG_{S_1} = L(S_1)$). A subscription PSG, PSG_{S_i}, is a line graph composed solely by vertices, each of which represents a directed edge in graph S_i connecting domain v_i to one of its neighbors. For instance, $PSG_{S_1} = \{\bar{v}_{12}, \bar{v}_{15}\}$.
- AS1 constructs its PSG by composing PSG_{S_1} with the PSGs received from its providers as follows: $PSG_1 = PSG_{S_1} \oplus (PSG_{21}^{adv} \bigcup PSG_{51}^{adv})$.

To fix ideas ... (cont.)

- The composition ($\oplus$) corresponds to the addition of a set of directed edges in PSG_1 of the form $(\bar{v}_{1x}, \bar{v}_{x-})$, where x denotes a neighbor of AS1. These edges simply connect the vertices $\bar{v}_{1x}$ introduced by PSG_{S_1}, with the set of vertices $\bar{v}_{x-}$ contained in PSG_{x1}^{adv}.
- For example, the composition introduces in PSG_1 the edges $(\bar{v}_{12}, \bar{v}_{23})$, $(\bar{v}_{12}, \bar{v}_{25})$, $(\bar{v}_{15}, \bar{v}_{52})$, $(\bar{v}_{15}, \bar{v}_{53})$, and $(\bar{v}_{15}, \bar{v}_{54})$

The opaqueness goal...

- It is important to observe that the topological information exported to a neighboring domain will vary depending on the commercial relationship with that domain.
- For example, AS2 and AS5 will export the graphs PSG_{21}^{adv}, and PSG_{51}^{adv}, respectively, to AS1.
- However, the PSGs exported directly between them will differ from these latter, since they will **mutually hide** their interconnection to AS3. Indeed, the topological information exchanged between each other will be pruned according to the export policies, in the form of the pruned line graphs $PSG_{25}^{adv} = L(G_{25}^{adv}) = \hat{L}_{25}(G_2)$, and $PSG_{52}^{adv} = L(G_{52}^{adv}) = \hat{L}_{52}(G_5)$.
- This information, together with the PSGs received from the rest of their corresponding neighbors, is what enables the independent construction of PSG_2 and PSG_5 at AS2, and AS5, respectively.

The loop avoidance goal...

- A straightforward way to avoid forwarding loops in the inference of AS-paths is by composing directed line graphs in the PSG constructions
- For instance, PSG_1 solves the loop shown in the example before, since in PSG_1, $\bar{v}_{15} \rightarrow \bar{v}_{53} \rightarrow \bar{v}_{34}$ is feasible, but $\bar{v}_{15} \rightarrow \bar{v}_{53} \rightarrow \bar{v}_{34} \rightarrow \bar{v}_{23}$ is not, given that once the vertex $\bar{v}_{34}$ is reached, there is no egress edge from it.
- This means that $\bar{v}_{34}$ is a terminal vertex, which is consistent with the fact that there may be no further transit for traffic going from AS3 to AS4.

PSGs: the hooks ...

One of the most important aspects of the PSGs is that they can be locally constructed with information resulting from the enforcement of the usual policies between domains.

Another important aspect that is implicit is that the PSGs constructed by domains differ between each other. This is essential to avoid disclosing $AS_r(G_0)$. The key is to leverage the advertisement of these different topological views of domains, which, as we will **prove**, once assembled in a PSG, can consistently capture the valley-free paths while remaining opaque and loop-free.

More formally ...

Definition

(Union of Line Digraphs) The union of two directed line graphs (or line digraphs) $L' = (\bar{V}', \bar{E}')$ and $L'' = (\bar{V}'', \bar{E}'')$, is the line digraph $L = L' \bigcup L''$, whose vertex-set is $\bar{V} = \bar{V}' \bigcup \bar{V}''$ and the edge-set is $\bar{E}$, where a directed edge $(\bar{v}_{ij}, \bar{v}_{jk}) \in \bar{E}$ if and only if the single hop path $(\bar{v}_{ij}, \bar{v}_{jk}) \subset L' \vee (\bar{v}_{ij}, \bar{v}_{jk}) \subset L''$.

More formally ... (cont.)

Definition

(Composition of Line Digraphs) The composition of two line digraphs, $L = (\bar{V}, \varnothing)$, whose edge-set is empty, and $L' = (\bar{V}', \bar{E}')$, is the line digraph $L \oplus L'$, whose vertex-set is $\bar{V} \bigcup \bar{V}'$ and the edge-set is $\bar{E} = \bar{E}' \bigcup \{(\bar{v}_{-x}, \bar{v}'_{x-})\}$, where $\bar{v}_{-x} \in \bar{V} \wedge \bar{v}'_{x-} \in \bar{V}'$.

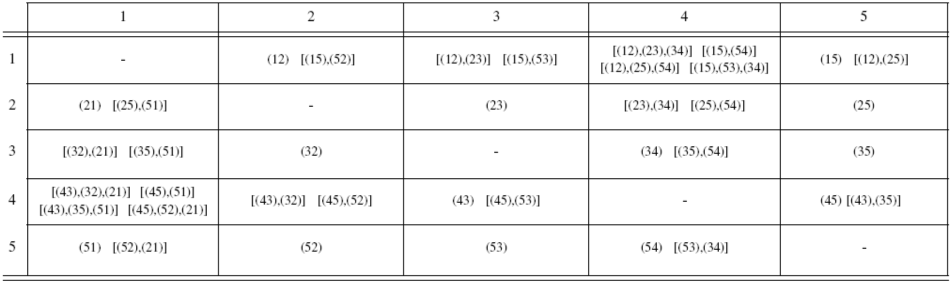
More formally ... (cont.)

Definition

(Path-State Graph) Let v_k and v_i be two neighboring domains in a routing system $\mathcal{S}$, and $L(S_i)$ be the line digraph of the subscription graph of v_i, S_i. Let G_k be the digraph that can be constructed by domain v_k from the path vectors contained in $\mathcal{R}_k$. Let $\hat{L}_{ki}(G_k)$ be the line digraph advertised from v_k to v_i, pruned by v_k according to the VFR policies in $\mathcal{S}$. The *Path-State Graph* PSG_i constructed by domain v_i is the graph resulting from the composition $PSG_i = L(S_i) \oplus \bigcup_k \hat{L}_{ki}(G_k)$.

Outline

Network Path-State table $\mathcal{P}$



	1	2	3	4	5
1	-	(12) [(15),(52)]	[(12),(23)] [(15),(53)]	[(12),(23),(34)] [(15),(54)] [(12),(25),(54)] [(15),(53),(34)]	(15) [(12),(25)]
2	(21) [(25),(51)]	-	(23)	[(23),(34)] [(25),(54)]	(25)
3	[(32),(21)] [(35),(51)]	(32)	-	(34) [(35),(54)]	(35)
4	[(43),(32),(21)] [(45),(51)] [(43),(35),(51)] [(45),(52),(21)]	[(43),(32)] [(45),(52)]	(43) [(45),(53)]	-	(45) [(43),(35)]
5	(51) [(52),(21)]	(52)	(53)	(54) [(53),(34)]	-

Paths consistency

Theorem

(Paths consistency) *Let v_i be a domain in a routing system $\mathcal{S} = \langle G_0, \mathcal{R} \rangle$, where the network routing table $\mathcal{R}$ contains the set of routing records resulting from the application of the VFR policies on G_0. Let $\mathcal{P} = \mathcal{T}(\mathcal{R})$ denote the transformation of the path vectors in table $\mathcal{R}$ into the routing records in table $\mathcal{P}$. Let $\mathcal{R}_i = \mathcal{T}^{-1}(\mathcal{P}_i)$ be the corresponding row of v_i in table $\mathcal{R}$, and PSG_i be the path-state graph constructed by v_i from the paths contained in row $\mathcal{P}_i$ of table $\mathcal{P}$. A path $\overrightarrow{p_l}$ is contained in PSG_i if and only if its corresponding path vector $\overrightarrow{p} = \mathcal{T}^{-1}(\overrightarrow{p_l}) \subset \mathcal{R}_i$.*

Paths consistency (cont.)

Proof: (Sufficiency) For each path $\overrightarrow{p} \subset \mathcal{R}_i$, it holds that there is a path $\overrightarrow{p_l}$ such that $\overrightarrow{p_l} = \mathcal{T}(\overrightarrow{p}) \subset \mathcal{P}_i$. Since PSG_i is constructed by v_i from the paths contained in $\mathcal{P}_i$, $\overrightarrow{p_l} \subset PSG_i$. Hence, if $\overrightarrow{p} \subset \mathcal{R}_i \Rightarrow \overrightarrow{p_l} \subset PSG_i$.

Paths consistency (cont.)

Proof (cont.): (**Necessity**) Suppose toward a contradiction that there exists a path $\overrightarrow{p_i} \subset PSG_i$, such that $\overrightarrow{p} = \mathcal{T}^{-1}(\overrightarrow{p_i}) \not\subseteq \mathcal{R}_i$. By construction, table $\mathcal{R} = \mathcal{V}(G_0)$, so $\mathcal{R}_i$ contains all the valley-free routes known by domain v_i. Since $\overrightarrow{p} \not\subseteq \mathcal{R}_i \Rightarrow \overrightarrow{p}$ is not valley-free, so $\overrightarrow{p_i}$ is also not valley-free. This would mean that there exists a non valley-free path $\overrightarrow{p_i}$ in PSG_i. Hence, there must be at least one edge $(\bar{v}_{xy}, \bar{v}_{yz})$ in $\overrightarrow{p_i} \subset PSG_i$, such that the transit ASx $\rightarrow$ ASy $\rightarrow$ ASz is not allowed in G_0 due to the VFR policies. From the definition of a PSG, there are two possibilities:

1. $(\bar{v}_{xy}, \bar{v}_{yz}) \subset \bigcup_k \hat{L}_{ki}(G_k)$
2. $(\bar{v}_{xy}, \bar{v}_{yz}) \not\subseteq \bigcup_k \hat{L}_{ki}(G_k)$ but arises from the composition operation ($\oplus$).

Paths consistency (cont.)

Proof (cont.): Consider the first possibility, i.e., $(\bar{v}_{xy}, \bar{v}_{yz}) \subset \bigcup_k \hat{L}_{ki}(G_k)$. From the definition of the "$\bigcup$" of line digraphs, it holds that the edge $(\bar{v}_{xy}, \bar{v}_{yz})$ can only be in the union if the single hop path $(\bar{v}_{xy}, \bar{v}_{yz})$ is contained in at least one line digraph $\hat{L}_{ki}(G_k)$. Thus, at least one of the line digraphs advertised to domain v_i contains a non valley-free edge. By induction to the origin, it is straightforward to show that this contradicts the PSG definition, since the line digraphs are pruned according to the VFR policies before being advertised to a neighboring domain.

Assume now that $(\bar{v}_{xy}, \bar{v}_{yz}) \not\subset \bigcup_k \hat{L}_{ki}(G_k)$, and consider the second possibility, i.e., $(\bar{v}_{xy}, \bar{v}_{yz})$ arises from the composition operation $(\oplus)$. This means that, the insertion of the vertices $\bar{v}_{iy}$ in the subscription PSG of domain v_i generates at least one edge $(\bar{v}_{iy}, \bar{v}_{yz})$ that is not valley-free. Hence, the transit ASi $\rightarrow$ ASy $\rightarrow$ ASz is not allowed in G_0 due to the filtering policies FP1–FP3 in ASy. Once again, this contradicts the definition of a PSG, since ASy will not advertise to ASi a path toward ASz. This completes the proof. ■

Main conclusion from the previous Theorem ...

$$\begin{tikzcd}
\text{From the routing information known by a domain } v_i \arrow[r] & v_i \text{ can construct a path state-graph } PSG_i \text{ of the network} \arrow[d] \\
\textbf{It is possible to reconstruct the paths known by } v_i \ \surd & \text{From } PSG_i \arrow[l]
\end{tikzcd}$$

Outline

Opaqueness

Definition

(Opaqueness of a graph) Let v_i be a domain in a routing system $\mathcal{S}$, and $\mathcal{R}_i$ be the set of valley-free paths known by v_i through $\mathcal{S}$. Let G_i be the graph representation of the internetwork constructed by v_i. The graph G_i is said to be *opaque* if every path available on G_i is contained in $\mathcal{R}_i$.

- ... the condition that a graph G_i must satisfy to be opaque is that the paths on G_i must be restricted to the valley-free routes distributed by $\mathcal{S}$ that reach v_i.
- This means that the AS relationships that can be inferred by v_i are strictly those embedded in the valley-free routes received at v_i.

Main Properties of the Path-State Graphs

Property

The PSGs are opaque.

Property

The PSGs are loop-free.

Opaqueness

Proof: Let PSG_i be the path-state graph constructed by v_i. By the previous Theorem, it holds that a path $\overrightarrow{p_i} \subset PSG_i \Leftrightarrow \overrightarrow{p} = \mathcal{T}^{-1}(\overrightarrow{p_i}) \subset \mathcal{R}_i$, so every path in PSG_i is in $\mathcal{R}_i$. Hence, PSG_i is opaque.

Loop Avoidance

Proof: Let PSG_i be the path-state graph constructed by v_i of an AS graph G_0. Suppose toward a contradiction that there exists a path $\overrightarrow{p_i} \subset PSG_i$, such that $\overrightarrow{p} = \mathcal{T}^{-1}(\overrightarrow{p_i})$ represents a loop in G_0. By the previous Theorem, it holds that the path $\overrightarrow{p}$ must be contained in $\mathcal{R}_i$. However, $\overrightarrow{p} \not\subseteq \mathcal{R}_i$, since by construction (see the import rule in equation (1)), the NR table $\mathcal{R}$ is free from routing loops—the domains in G_0 will never import routes containing loops into table $\mathcal{R}$.

Modeling definitively needs to be reviewed ...

Scientific papers......

“We model the connectivity between ASs in the Internet using an AS graph $G = (V, E)$, where the node set V consists of ASs and the edge set E consists of AS pairs that exchange traffic with each other.”

- L. Gao, “On Inferring Autonomous System Relationships in the Internet,” IEEE/ACM Trans. on Networking, Vol. 9, No. 6., pp. 733-745, December 2001.
- S. Jaiswal, A. L. Rosenberg, D. Towsley, “Comparing the structure of power-law graphs and the Internet AS graph,” 12th IEEE International Conference on Network Protocols (ICNP’04), Berlin, Germany, October 2004.
- D. Pei, M. Azuma, D. Massey, and L. Zhang, “BGP-RCN: Improving BGP Convergence through Root Cause Notification,” Computer Networks Journal, Elsevier, Vol. 48, No. 2, pp. 175-94, June 2005.
- A. Bremler-Barr, Y. Afek and S. Schwarz, “Improved BGP Convergence via Ghost Flushing,” in Proceedings of IEEE INFOCOM, 2003.
- J. Chandrashekar, Z. Duan, Z.-L. Zhang, and J. Krasky, “Limiting path exploration in BGP,” in Proceedings of INFOCOM, Miami, USA, 2005.
- ....

Outline

Some observations...

- We observe that for many scale-free graphs of interest in practice, the node degree exponent, β, is in the range $\beta \leq -2$.
- For example, the value of β for the Internet's AS graph has stayed at approximately $\beta \approx -2.2$ since 1998 [Faloutsos, Siganos, D. Papadimitriou,...].
- Hence, we shall focus on scale-free graphs with an exponent β in this range. Under these assumptions, the state complexity of the PSG can be upper bounded as follows...

State Complexity

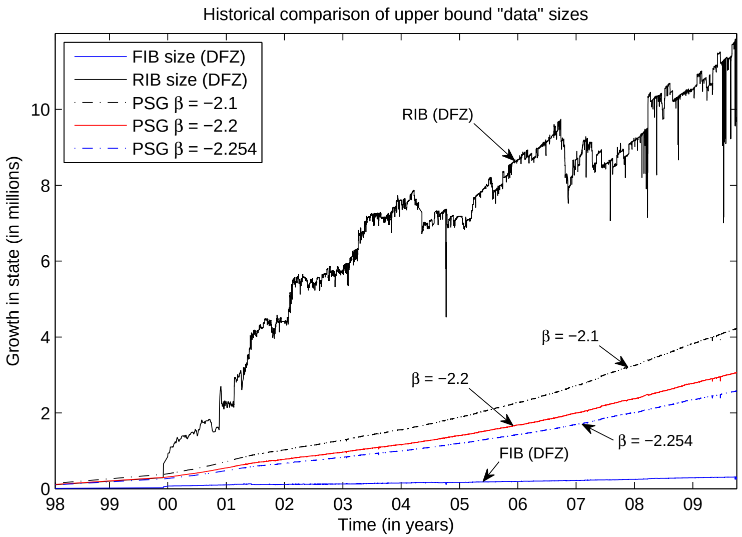
Theorem

Let $\mathcal{S} = \langle G_0, \mathcal{R} \rangle$ be a routing system, such that $G_0(V, E)$ is a scale-free graph with node degree exponent $\beta \leq -2$. Let k be the number of different frequencies of node degrees in G_0, and PSG_i be the path-state graph constructed by a node $v_i \in V$. The state complexity of the path-state graph PSG_i is $O\left(|V| k^{\beta+3}\right)$.

Remark

For many scale-free graphs, the node degree distribution is sparse, often with $k \ll |V|$. For example, while the current number of nodes in the Internet's AS graph is $|V| \approx 35.000$, there are only $k = 198$ different frequencies of node degrees in this graph.

State Complexity: $k \approx \sqrt{|V|} \Rightarrow O(n^{1.373})$



State Complexity (cont.)

Remark

- *The entries in BGP's RIB and FIB databases are highly dynamic ... so the size of the state is exacerbated by the dynamics!*
- *Conversely ... the state of the PSG is rather stable ... probably with very few changes per day.*

Proof

Let $PSG_i = (\bar{V}_i, \bar{E}_i)$ be the path-state graph of $G_0(V, E)$, constructed by a domain $v_i \in V$ from the valley-free routes known by the latter. The state complexity of PSG_i is $|\bar{V}_i| + |\bar{E}_i|$, hence we seek an upper bound for this sum.

Let d_j, $j = 1, \ldots, |V|$, denote the degree of node v_j in graph G_0. For each node $v_j \in V$, the maximum possible number of corresponding edges, $(\bar{v}_{-j}, \bar{v}_{j-})$, in PSG_i, is given by the permutations:

$$P_2^{d_j} = \frac{d_j!}{(d_j - 2)!}$$

This is the case when the transit between every pair of edges adjacent to v_j in G_0 must be present in PSG_i. Clearly, the minimum possible number of corresponding edges for v_j in PSG_i is zero, which occurs, e.g., when v_j is a non-transit domain, or when v_j is not reachable from v_i.

Proof (cont.)

This means that, within the set of transits $e \to e'$ that are valley-free, the function Y tests whether they must be contained in PSG_v. The test Y for the transit $(v_n, v_j) \to (v_j, v_m)$ in PSG_i is denoted as $Y(v_i, (v_n, v_j), (v_j, v_m)) = y_{nm}^{(j,i)}$.

To fix ideas, consider AS5 and AS1 in the example. For AS5 it holds that, e.g., $x_{23}^{(5)} = 1$, whilst $x_{43}^{(5)} = 0$. From the perspective of AS1, on the other hand, $y_{43}^{(5,1)} = 1$, since the transit AS4 $\to$ AS5 $\to$ AS3 is valley-free, but this transit path must not be contained in PSG_1 (notice that the path [4,5,3] is not contained in $\mathcal{R}_1$).

Indeed, from the $\frac{4!}{2!} = 12$ possible transits through AS5, only 4 are present in PSG_1 (see the PSG). The rest, either yield true in test X or in test Y, so they must not be present in PSG_1.

Proof (cont.)

This means that, within the set of transits $e \to e'$ that are valley-free, the function Y tests whether they must be contained in PSG_v. The test Y for the transit $(v_n, v_j) \to (v_j, v_m)$ in PSG_i is denoted as $Y(v_i, (v_n, v_j), (v_j, v_m)) = y_{nm}^{(j,i)}$.

To fix ideas, consider AS5 and AS1 in the example. For AS5 it holds that, e.g., $x_{23}^{(5)} = 1$, whilst $x_{43}^{(5)} = 0$. From the perspective of AS1, on the other hand, $y_{43}^{(5,1)} = 1$, since the transit AS4 $\to$ AS5 $\to$ AS3 is valley-free, but this transit path must not be contained in PSG_1 (notice that the path [4,5,3] is not contained in $\mathcal{R}_1$).

Indeed, from the $\frac{4!}{2!} = 12$ possible transits through AS5, only 4 are present in PSG_1 (see the PSG). The rest, either yield true in test X or in test Y, so they must not be present in PSG_1.

Proof (cont.)

Let $V_T \subset V$, denote the set of transit domains in graph G_0. Accordingly, the number of directed edges $|\bar{E}_i|$ in PSG_i is:

$$|\bar{E}_i| = \sum_{j=1}^{|V_T|} \left[\frac{d_j!}{(d_j-2)!} - \sum_{n=1}^{d_j} \sum_{\substack{m=1 \\ m \neq n}}^{d_j-1} x_{nm}^{(j)} + y_{nm}^{(j,i)} \right] < \sum_{j=1}^{|V_T|} d_j^2 \leq \sum_{j=1}^{|V|} d_j^2 \qquad (1)$$

Remark

The upper bound in (1) is very conservative. Finding a tighter bound is an open problem deserving further work, where the challenge is to quantify the negative terms in (1). For instance, the terms $y_{nm}^{(j,i)}$ involve a reachability problem in node v_i, which Griffin et al. showed to be NP-complete.

Proof (cont.)

The number of nodes with degree d is given by the frequency f_d, and since $f_d \propto d^\beta$, the probability that a node has degree d is thus:

$$p(d) = \frac{f_d}{|V|} = \frac{\mathcal{C}}{|V|} d^\beta \tag{2}$$

where $\mathcal{C}$ is the proportionality constant in the degree power-law in graph G_0. By sorting in the order of increasing degree, and grouping terms in the k different frequencies of degrees, the summation on the right-hand-side of inequality (1) can expressed as:

$$S_1 = \sum_{j=1}^{|V|} d_j^2 = f_{d_1} d_1^2 + \cdots + f_{d_k} d_k^2 \tag{3}$$

$$\Rightarrow \quad \frac{S_1}{|V|} = p(d_1)d_1^2 + \cdots + p(d_k)d_k^2 = \sum_{l=1}^{k} p(d_l)d_l^2 \tag{4}$$

Proof (cont.)

Hence, the bound in (1) can be expressed as follows:

$$|\bar{E}_i| < |V| \sum_{l=1}^{k} p(d_l) d_l^2 = \mathcal{C} \sum_{l=1}^{k} d_l^{\beta+2} \tag{5}$$

where the proportionality constant $\mathcal{C}$ can be derived by normalizing the probabilities in (2). Accordingly:

$$\mathcal{C} = \frac{|V|}{\sum_{l=1}^{k} d_l^{\beta}} \Rightarrow |\bar{E}_i| < |V| \frac{\sum_{l=1}^{k} d_l^{\beta+2}}{\sum_{l=1}^{k} d_l^{\beta}} \tag{6}$$

Proof (cont.)

Due to the sorting and grouping of node degrees in the different frequencies in (3), it holds that $d_l \geq l, \forall\, l \in \{1 \ldots k\}$, so when $\beta \leq -2$, the powers of the degrees in the numerator of the right-hand side of the inequality (6), can be bounded by: $d_l^{\beta+2} \leq l^{\beta+2}$. On the other hand, the series in the denominator in (6) converges for $\beta \leq -2$, and given that the sequence of the series is positive, it holds that $\sum_{l=1}^{k} d_l^{\beta} \geq d_1^{\beta}, \forall\, k > 1$.

In light of the above, (6) can be bounded as follows:

$$|\bar{E}_i| < d_1^{-\beta} |V| \sum_{l=1}^{k} l^{\beta+2} \tag{7}$$

Proof (cont.)

Since in (3) the degrees are sorted in the order of increasing degree, without loss of generality we assume that for the first frequency, the degree is $d_1 = 1$. Thus,

$$|\bar{E}_i| < |V| \sum_{l=1}^{k} l^{\beta+2} \tag{8}$$

It is easy to observe that the sequence in the series in (8) is positive and non-increasing for $\beta \leq -2$, so the integral test for convergence can be applied.

Proof (cont.)

Let $r(x)$ be an integrand function of the form $r(x) = x^{\beta+2}$, $r\,[1, +\infty) \to [0, +\infty)$. Then, for each $k \in \mathbb{N}$, it holds that:

$$\sum_{l=1}^{k} r(l) = \int_{1}^{k} r(x)dx + C_r + \varepsilon_r(k) \qquad (9)$$

with C_r constant, and $0 \le \varepsilon_r(k) \le r(k) - \lim_{x\to+\infty} r(x)$.

Since $\beta \le -2$, the limit of $r(x)$ when $x \to +\infty$ is zero when $\beta < -2$, and one if $\beta = -2$. Thus, when $\beta < -2$, $\varepsilon_r(k) \le r(k) \Rightarrow \varepsilon_r(k) = O\left(k^{\beta+2}\right)$, and $\varepsilon_r(k) = 0$ if $\beta = -2$.

Proof (cont.)

By integrating in (9) and using (8) we obtain:

$$|\bar{E}_i| < |V| \left(\frac{k^{\beta+3} - 1}{\beta + 3} + C_r + \varepsilon_r(k) \right) \tag{10}$$

Therefore, $|\bar{E}_i| = O(|V|k^{\beta+3})$.

Proof (cont.)

On the other hand, the number of vertices $|\bar{V}_i|$ in PSG_i is bounded by twice the number of links in G_0, which is given by:

$$|\bar{V}_i| = \sum_{j=1}^{|V|} d_j \tag{11}$$

Following the same line of argument as above, it holds that

$$|\bar{V}_i| = O(|V|k^{\beta+2})$$

Therefore, the state complexity of PSG_i is:

$$O(|\bar{V}_i| + |\bar{E}_i|) = O(|V|k^{\beta+3})$$

This completes the proof. ■

Outline

Some essential questions...

- Can path-state vectors coexist with BGP and be integrated seamlessly into the interdomain routing system?
- How does the data plane look like in a path-state vector protocol?
- Can path-state vectors remove the path exploration process from the routing system?
- Can path-state vectors improve the convergence time dramatically?
- Can path-state vectors reduce the churn of updates dramatically?
- Can path-state vectors help improving security? How?
- What about routing the control-plane messages for the routing we want to secure? ... beware of that!!
- How can path-state vectors be exploited for Traffic Engineering purposes?

Path-State Graphs (PSGs)

They are being prototyped...

2009 Cisco RFP, sponsors: Fred Baker, Russ White, and David Ward
2010/2011 Cisco SRA, sponsors: Russ White, Pere Monclus, and Fred Baker

- **Path-State Graphs (PSGs)** provide a graph representation through which domains can keep consistent views of the forwarding paths and domain-level connectivity of an internetwork subject to routing policies. PSGs respect the opaqueness of providers, since as we shall show, they are constructed with the routing information resulting from the enforcement of the usual export policies between routing domains.
- **Path-State Vectors (PSVs)**: a protocol suite $\rightarrow$ based on an overlay over the BGP protocol. PSVs exploit the PSGs, e.g., to dramatically reduce the time and churn rate of an interdomain routing convergence. In particular, we will show how PSGs can be applied for *removing* the path-exploration from the Internet's routing system; and more importantly, we will show how to achieve this without replacing nor extending BGP.

Outline

Degree-rank properties of the ARK's subgraph

Lemma

Let n_x denote the number of nodes in graph G_x, $x = \{S, A\}$. A subgraph $G_S \subset G_A$, with $n_S < n_A$, preserves the degree-rank properties of a scale-free graph $G_A \Leftrightarrow$ the approximation by linear regression for subgraph G_S in a log-log plot is the same as for G_A, with a negative vertical shift equal to $\gamma_A log(\frac{n_S}{n_A})$.

Proof

Necessity–The degree-rank properties of graph G_A are characterized by $d_i \propto r_i^{\gamma_A}$, $i \in G_A$. If $G_S \subset G_A$ preserves these properties, then, the degree for subgraph G_S is given by:

$$d_j = \left(\frac{r_j}{n_S}\right)^{\gamma_A}, \ \ j \in G_S \tag{12}$$

Therefore, in log-log form the degree-rank properties of graph G_S can be captured by linear regression as follows.

$$log\left(d^{(S)}\right) = \gamma_A log\left(r\right) - \gamma_A log\left(n_S\right) \tag{13}$$

Likewise, for the complete graph G_A we have:

$$log\left(d^{(A)}\right) = \gamma_A log\left(r\right) - \gamma_A log\left(n_A\right) \tag{14}$$

Evaluations for 1007 domains

Proof (cont.)

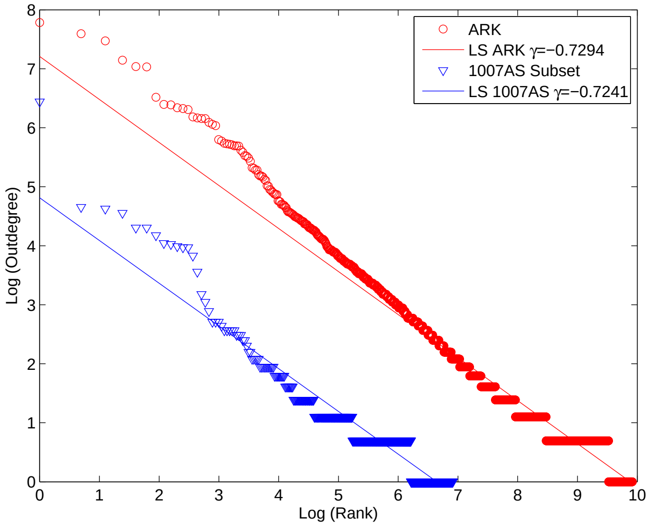
By subtracting (13) from (14) we obtain:

$$log\left(d^{(S)}\right) = log\left(d^{(A)}\right) - \gamma_A log\left(\frac{n_S}{n_A}\right) \qquad (15)$$

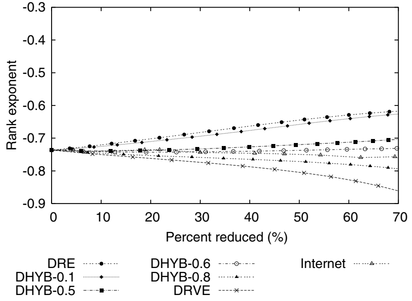
Given that the ranks are sorted in the order of decreasing degree, $\gamma_A < 0$, and since $n_S < n_A \Rightarrow -\gamma_A log(\frac{n_S}{n_A}) < 0$.

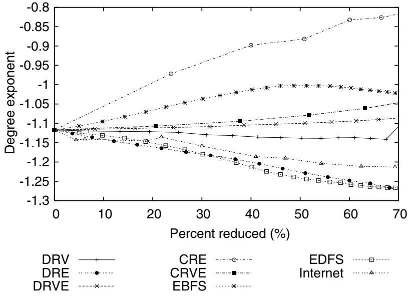
Sufficiency–Since the approximation by linear regression for subgraph G_S in a log-log plot is that of G_A minus $\gamma_A log(\frac{n_S}{n_A})$, then, by operating from (15) it is possible to obtain (13), which means that $d_j \propto r_j^{\gamma_A}$, $j \in G_S$. Therefore, graph G_S preserves the degree-rank properties of graph G_A. This completes the proof. ■

Rank power-law: 1007 domains (blue distribution)



Krishnamurthy 2007: power-laws





- V. Krishnamurthy, M. Faloutsos, M. Chrobak, J. H. Cui, L. Lao, and A. G. Percus, "Sampling large Internet topologies for simulation purposes," Computer Networks, Elsevier, Vol. 51, issue 15, pp. 4284-4302, October 2007.

The reduction algorithm

Algorithm 1 HBR0 algorithm: $G(V,E) \Rightarrow G_s(V_s,E_s)$

```
Input: original topology $G(V,E)$, sampling rate $0 < p \leq 1$
Output: smaller topology $G_s(V_s,E_s)$
 1: $TopASes \Leftarrow$ get top clique ASes from $G(V,E)$
 2: $V_s \Leftarrow TopASes$
 3: $CurrentLayerASes \Leftarrow TopASes$
 4: while $CurrentLayerASes$ not empty do
 5:    $NextLayerASes \Leftarrow$ empty
 6:    for all $AS$ in $CurentLayerASes$ do
 7:       for all $cust$ such that $cust$ is a customer of $AS$ do
 8:          if $0 \leq rand() < p$ and $cust$ not in $V_s$ then
 9:             add $cust$ into $NextLayerASes$
10:          end if
11:       end for
12:    end for
13:    $V_s \Leftarrow V_s \bigcup NextLayerASes$
14:    $CurrentLayerASes \Leftarrow NextLayerASes$
15: end while
16: $E_s \Leftarrow$ empty
17: for all $edge$ in $E$ do
18:    if both nodes at the two ends of the $edge$ is in $V_s$ then
19:       add $edge$ into $E_s$
20:    end if
21: end for
22: return $G_s(V_s,E_s)$
```

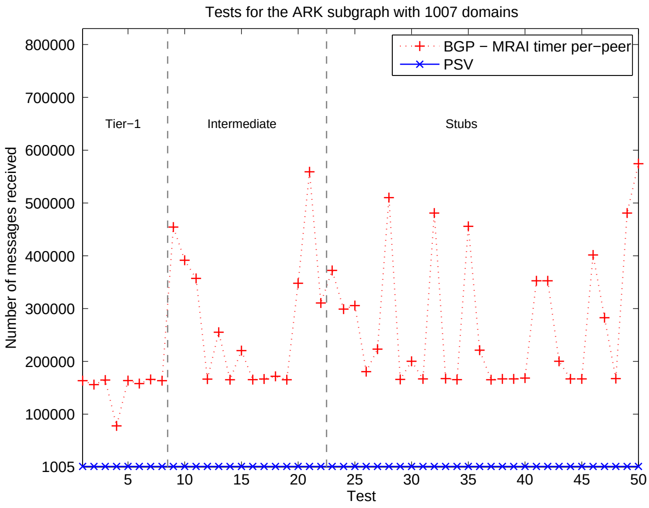
- Y. He, S. V. Krishnamurthy, M. Faloutsos, and M. Chrobak, “Policy-aware topologies for efficient inter-domain routing evaluations,” IEEE INFOCOM 2008 Mini-Conference, Phoenix, AZ, USA, April 2008.

Degree-rank properties of the ARK's subgraph

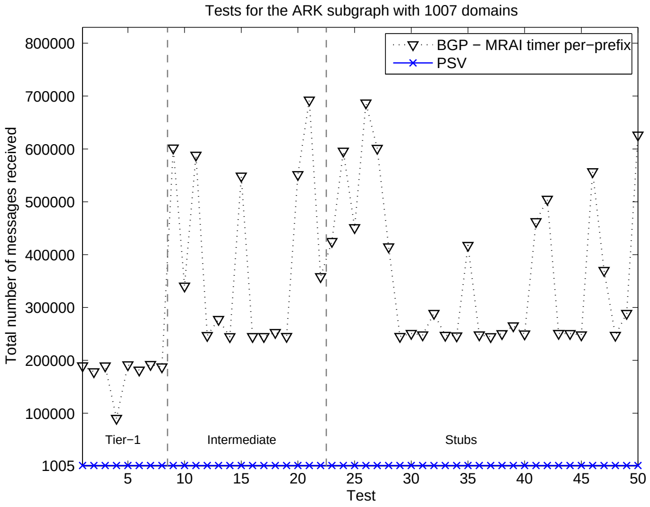
TABLE I
OUTDEGREE-RANK PROPERTIES OF THE ARK'S SUBGRAPH USED IN THE TESTS.

	Number of domains	Rank exponent of the power-law γ	Relative difference with the theoretical shift $\gamma_A log(\frac{n_S}{n_A})$	Correlation coefficient between the outdegree-rank distributions and their approximations
Complete ARK's graph G_A	20,305	– 0.7294	–	0.9685
ARK's subgraph G_{1007}	1007	– 0.7241	6.7%	0.9618

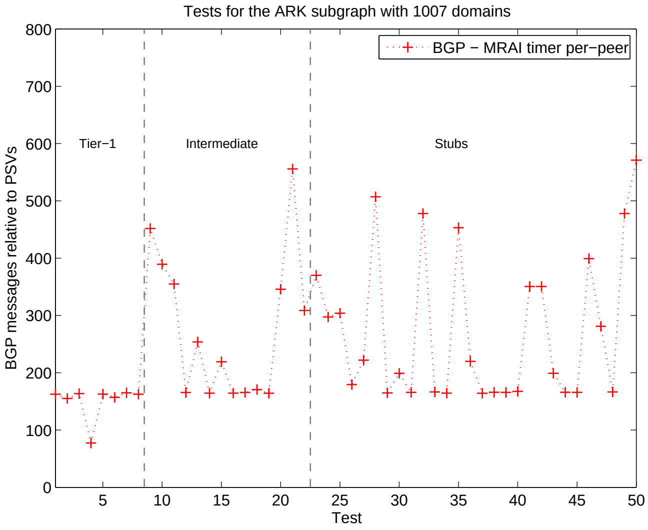
Churn: Unreachable case (MRAI per-peer)



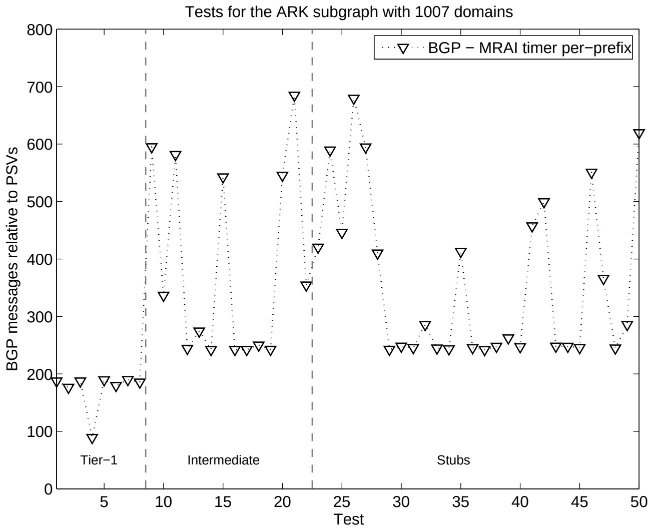
Churn: Unreachable case (MRAI per-prefix)



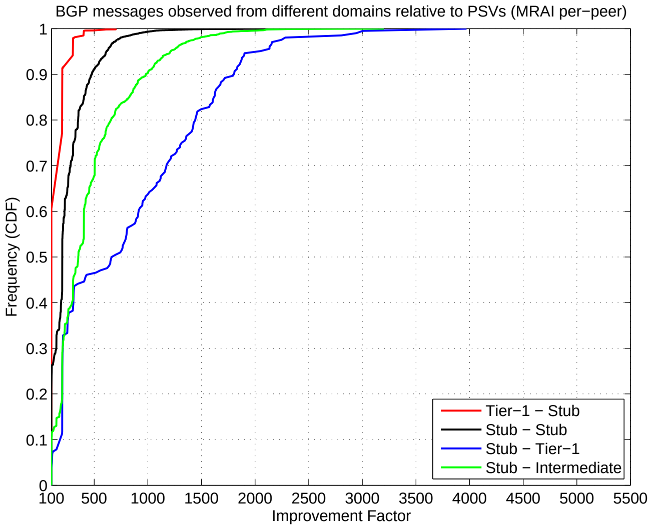
Churn: Unreachable case (MRAI per-peer)



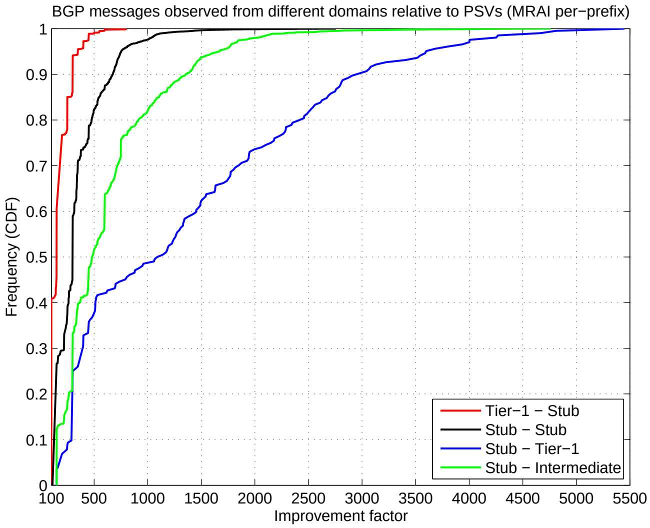
Churn: Unreachable case (MRAI per-prefix)



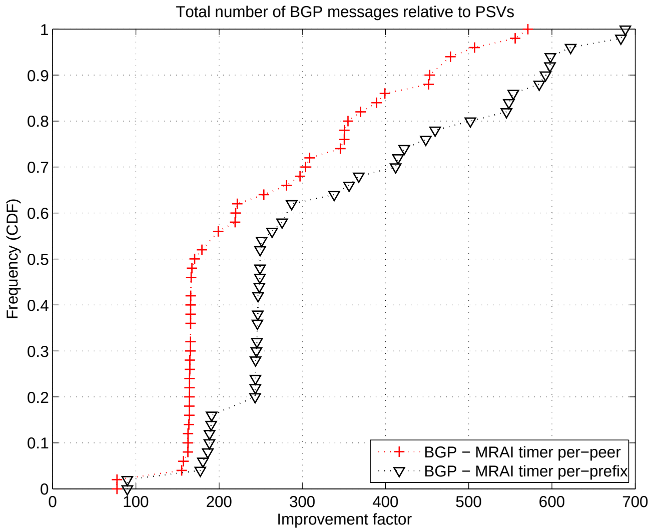
Churn: Unreachable case (MRAI per-peer)



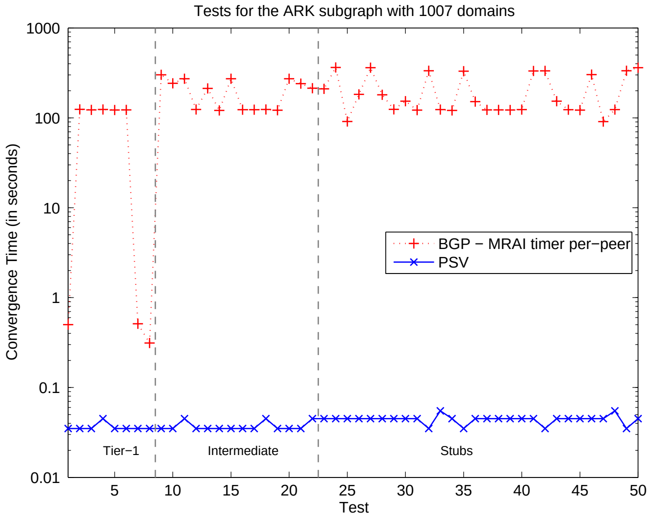
Churn: Unreachable case (MRAI per-prefix)



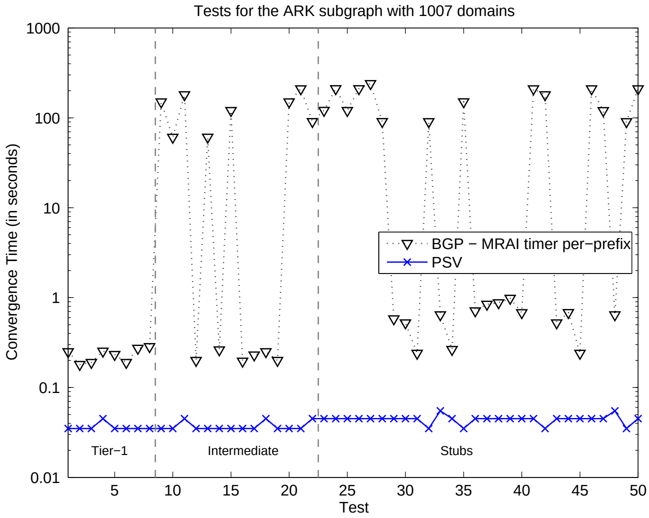
Churn: Unreachable case



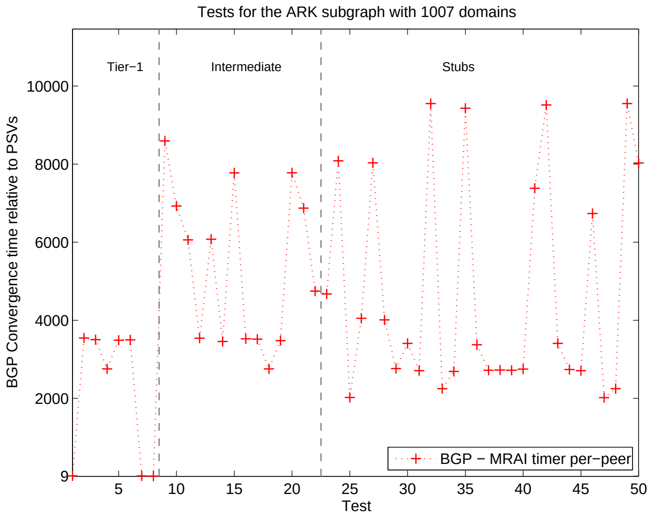
Conv. time: Unreachable case (MRAI per-peer)



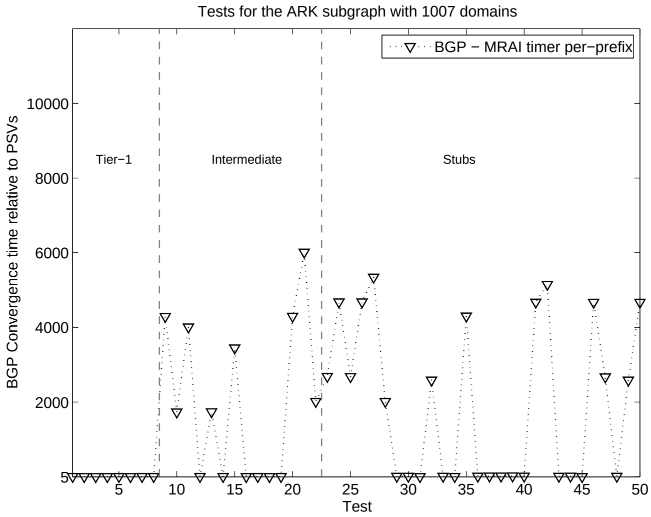
Conv. time: Unreachable case (MRAI per-prefix)



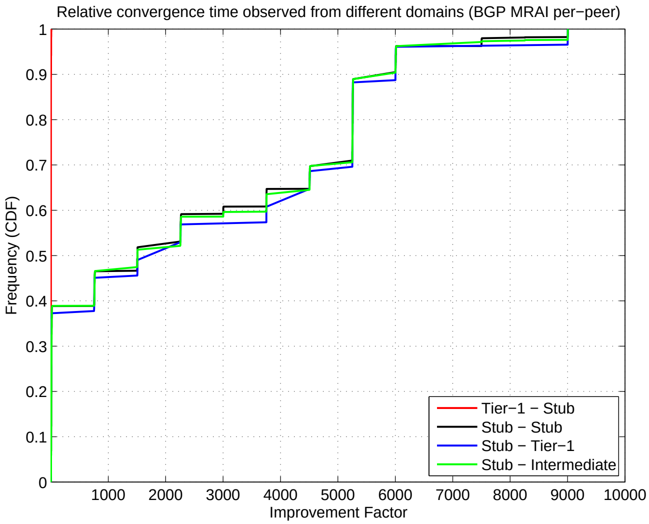
Conv. time: Unreachable case (MRAI per-peer)



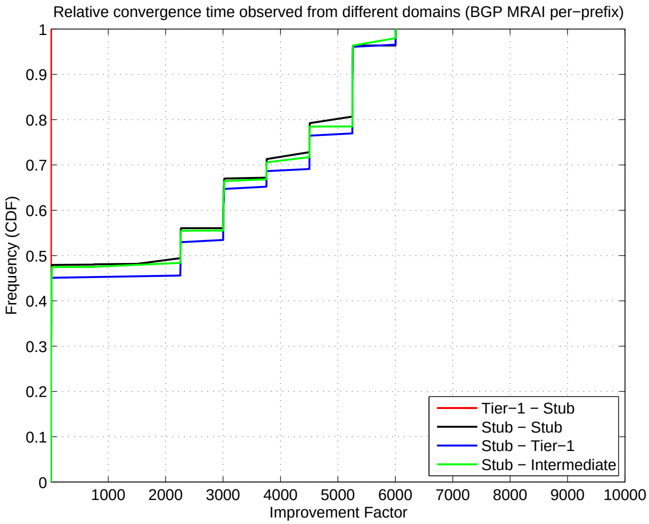
Conv. time: Unreachable case (MRAI per-prefix)



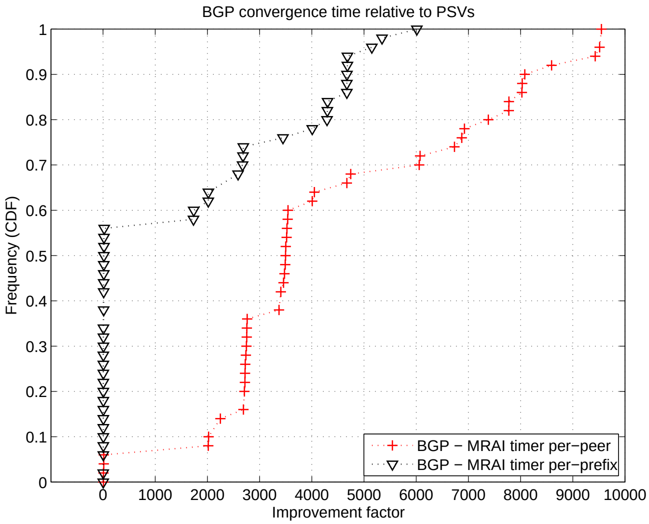
Conv. time: Unreachable case (MRAI per-peer)



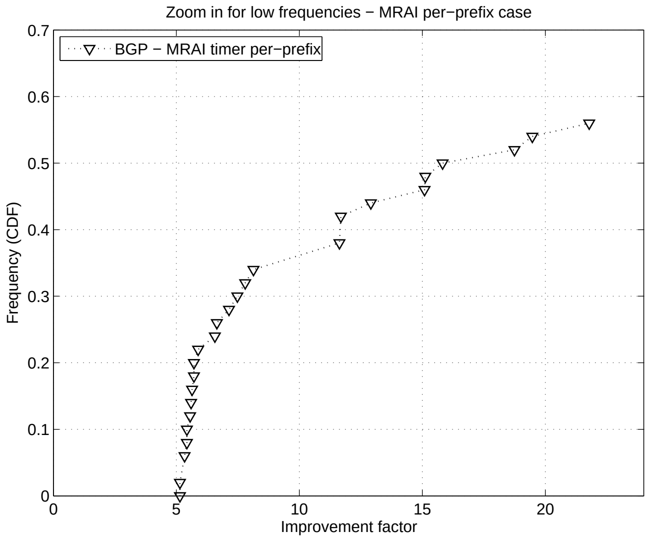
Conv. time: Unreachable case (MRAI per-prefix)



Conv. time: Unreachable case



Conv. time: Unreachable case



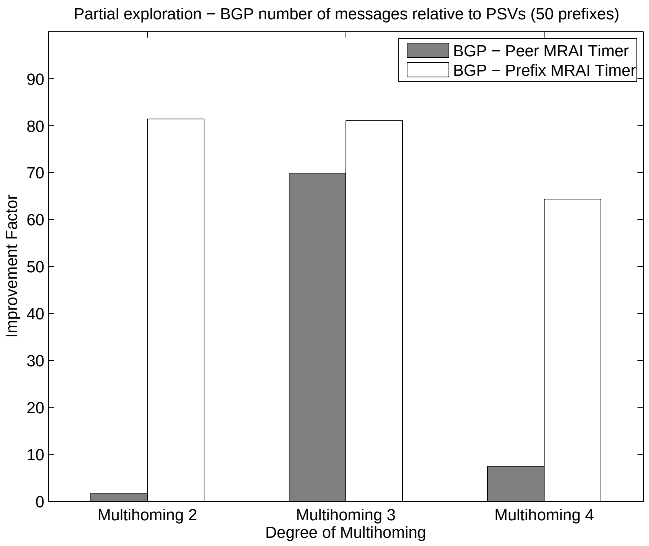
Convergence time

A PSV message contains control information indicating how the PSG has changed, and the set of prefixes affected by the change. In our example, the change to be informed is that network *D* is unreachable, and this change affects 50 IP prefixes. Considering that:

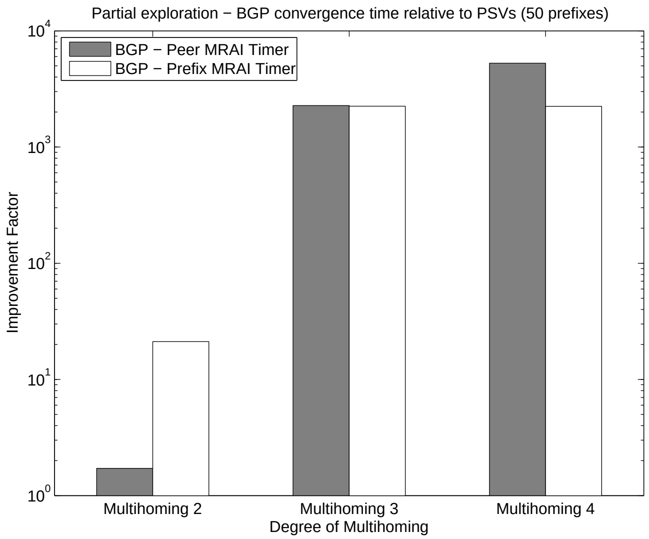
- An IPv4 prefix can be represented using 5 bytes.
- The protocol stack overhead is usually around 80 bytes.
- The control information detailing the disconnection of domain *D* in the PSG can be represented with 9 bytes (eight bytes — 32 bits x 2 — indicating each of the AS numbers of the domains adjacent to the change, and another byte for cause notification).

Then, the number of bytes, *b*, per inter-domain message is $b \approx 50 \text{ x } 5 + 80 + 9 = 339$ bytes. Thus, the total number of bytes needed considering all the inter-domain messages generated is $\sum b \approx 1005 \text{ x } 339 = 332.7$ Kbytes. This means that by sending only 332.7 Kbytes through the overlay, the provider of domain *D* can directly update the entire 1005-domain network, leading to a convergence process that is orders of magnitude faster than with BGP.

Churn: Multihoming case



Convergence: Multihoming case



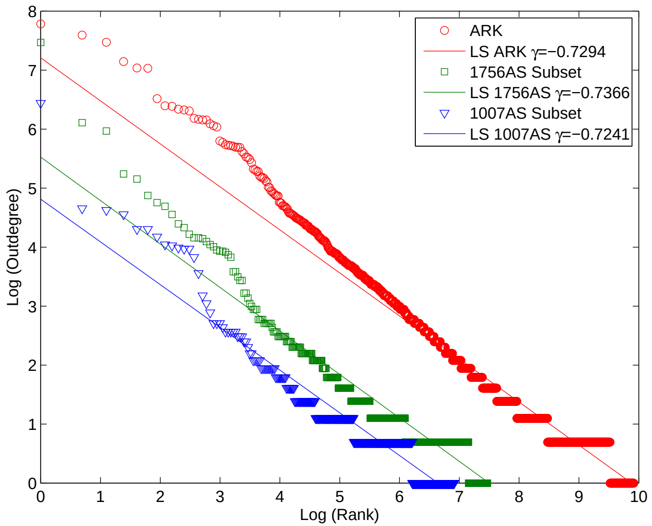
Summary of results

TABLE II
SUMMARY OF RESULTS FOR THE UNREACHABLE CASE

	Improvement factors of a PSV protocol over BGP in an internetwork with 1007 domains							
	MRAI timer per-peer				MRAI timer per-prefix			
Origin - Observation Point	Min. Churn	Max. Churn	Min. Conv. Time	Max. Conv. Time	Min. Churn	Max. Churn	Min. Conv. Time	Max. Conv. Time
Tier-1 - Stub	100	702	1	12.79	100	800	1	7.12
Stub - Stub	100	2313	1.50	9008.2	100	2756	1.50	6013.3
Stub - Intermediate	100	3200	1.25	9006	150	4760	1.50	6010.8
Stub - Tier-1	101	3972	1.88	9003.7	150	5449	1.50	6006.3

Evaluations for 1756 domains

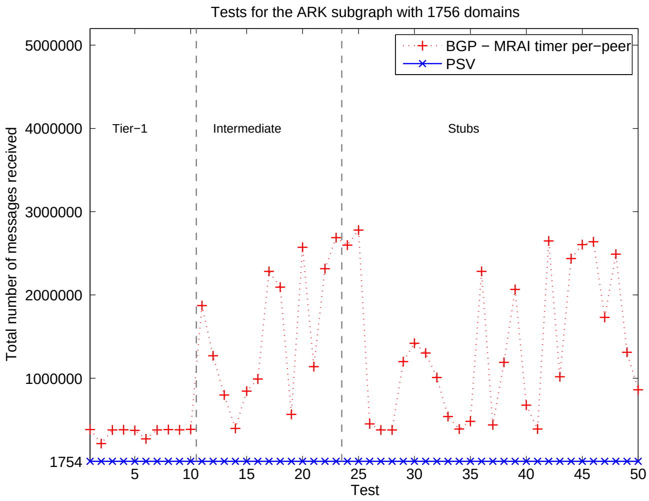
Rank power-law: 1756 domains (green distribution)



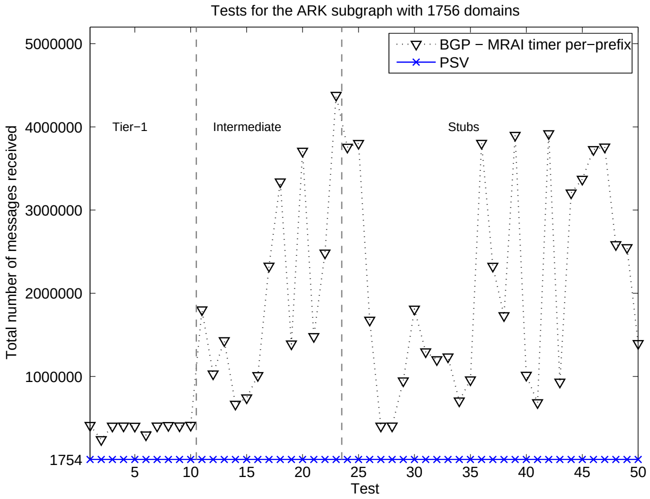
Outdegree-rank properties of the ARK's subgraph

	Number of domains	Exponent γ	Theoretical shift $\gamma_A log(\frac{n_S}{n_A})$	Correlation coefficients
Complete ARK's graph G_A	20,305	– 0.7294	–	0.9685
ARK's subgraph G_{1756}	1756	– 0.7366	2%	0.9555
ARK's subgraph G_{1007}	1007	– 0.7241	6.7%	0.9618

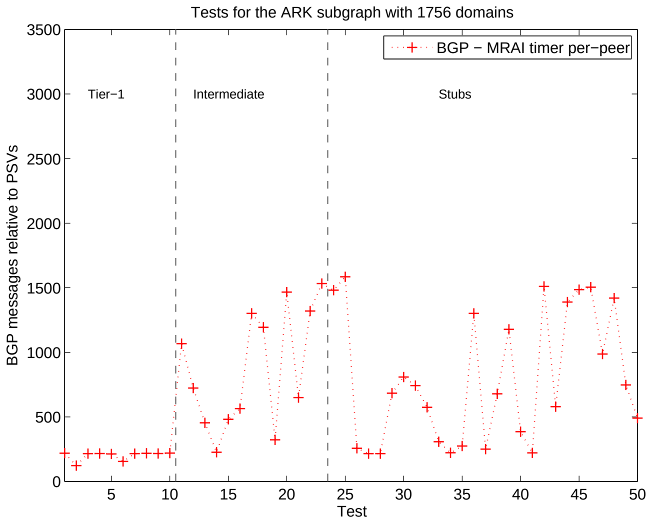
Churn: Unreachable case (MRAI per-peer)



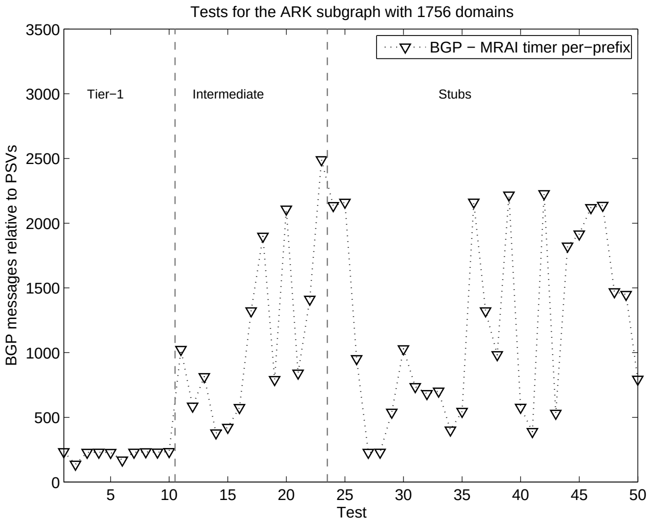
Churn: Unreachable case (MRAI per-prefix)



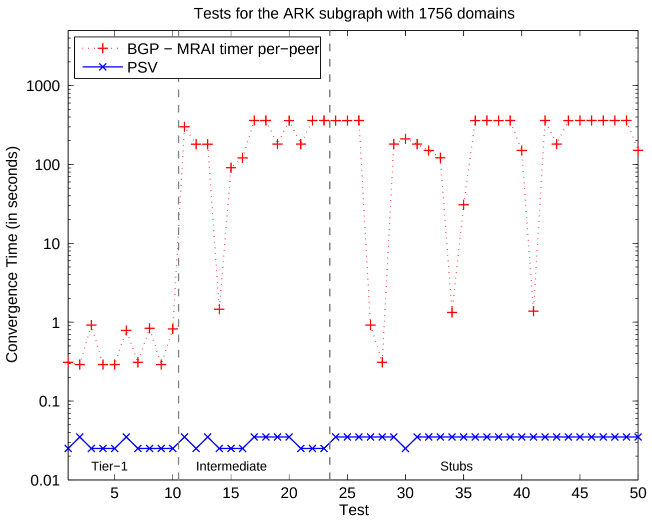
Churn: Unreachable case (MRAI per-peer)



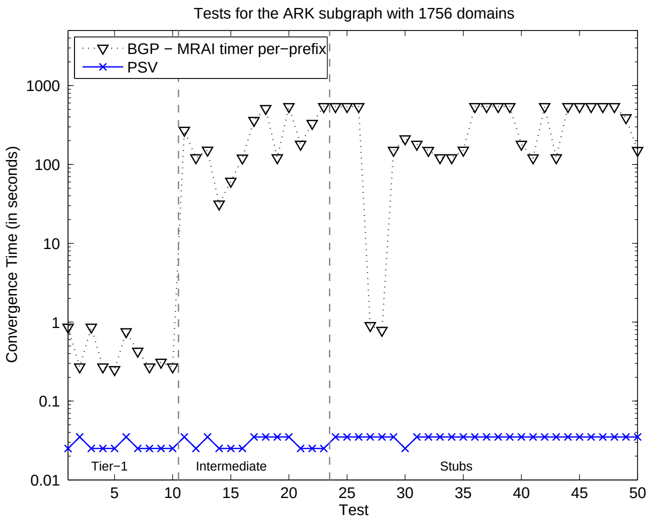
Churn: Unreachable case (MRAI per-prefix)



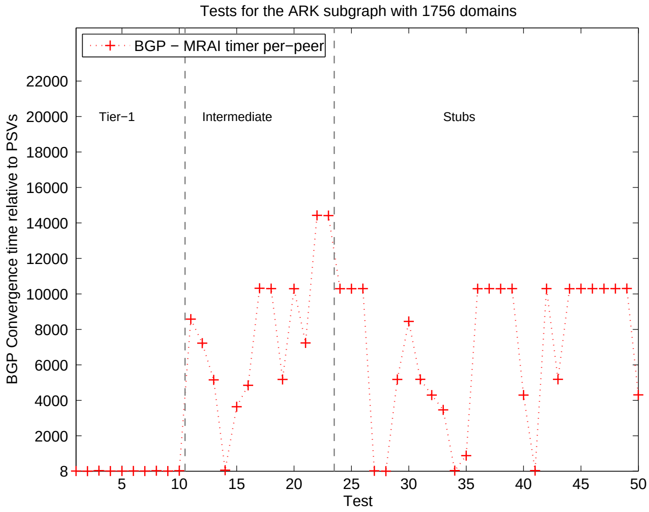
Conv. time: Unreachable case (MRAI per-peer)



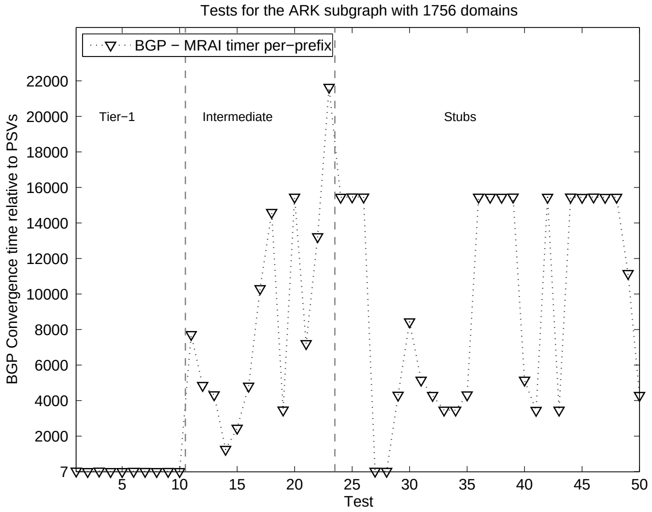
Conv. time: Unreachable case (MRAI per-prefix)



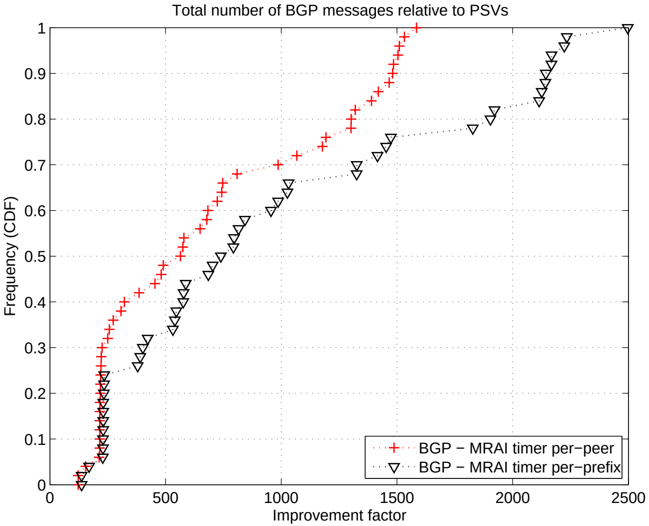
Conv. time: Unreachable case (MRAI per-peer)



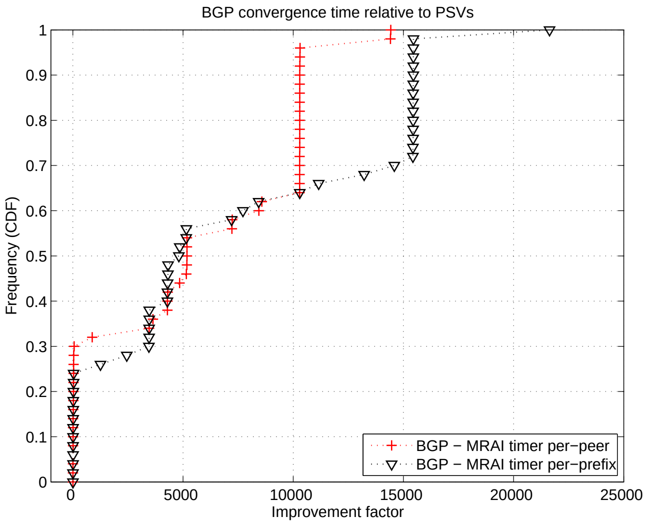
Conv. time: Unreachable case (MRAI per-prefix)



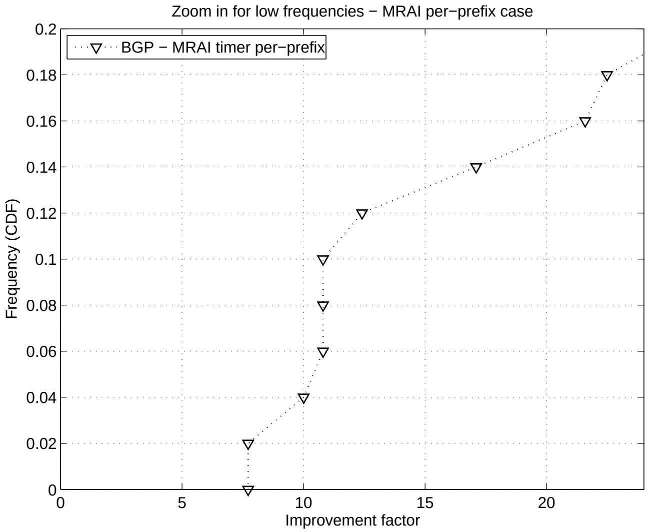
Churn: Unreachable case



Conv. time: Unreachable case

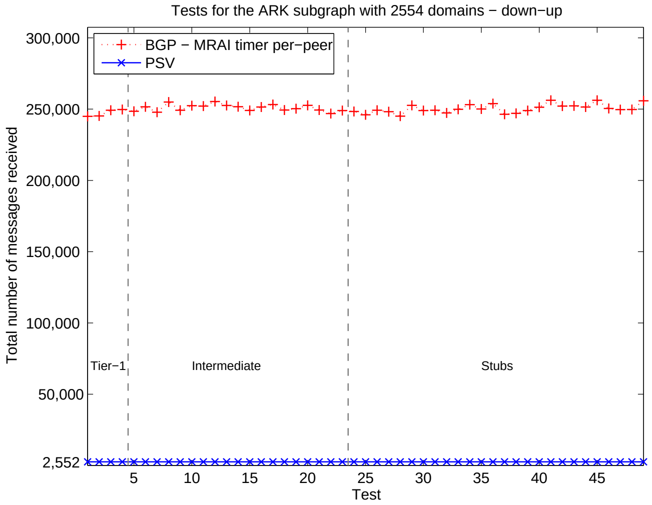


Conv. time: Unreachable case

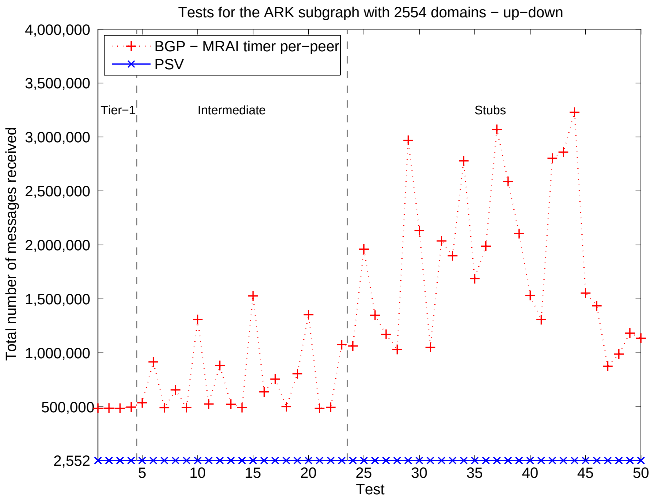


Evaluations for 2554 domains

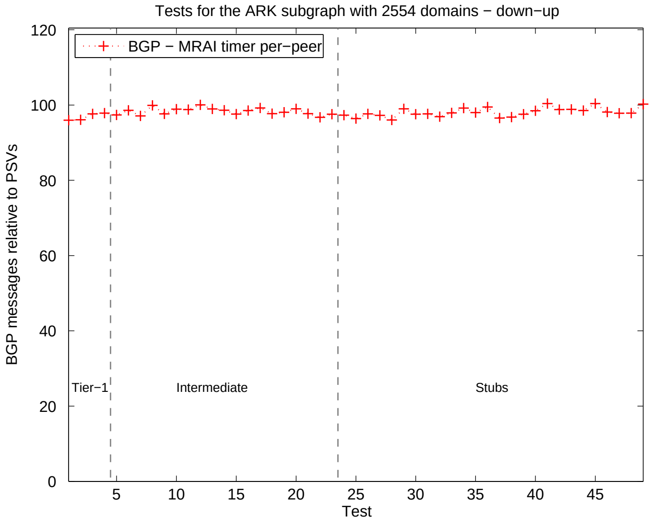
Churn: Unreachable case (MRAI per-peer)



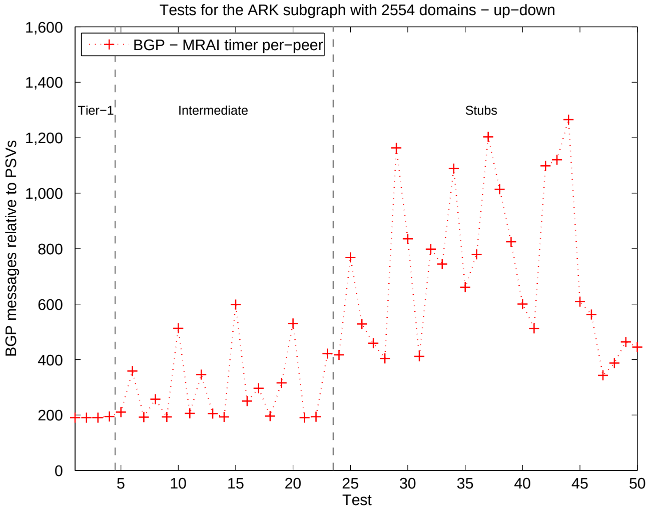
Churn: Unreachable case (MRAI per-peer)



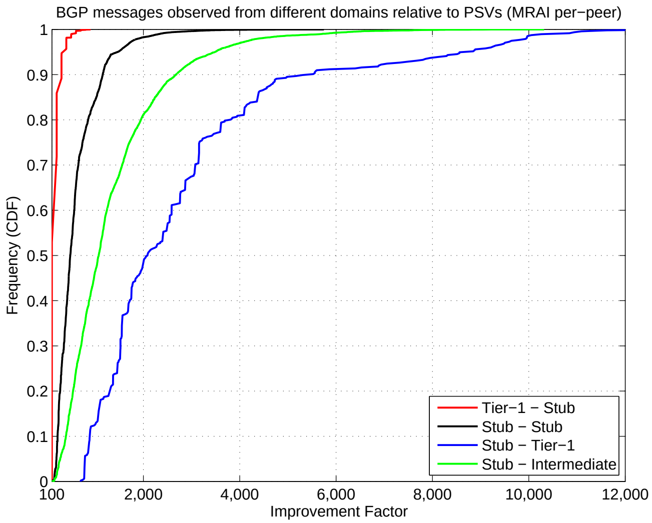
Churn: Unreachable case (MRAI per-peer)



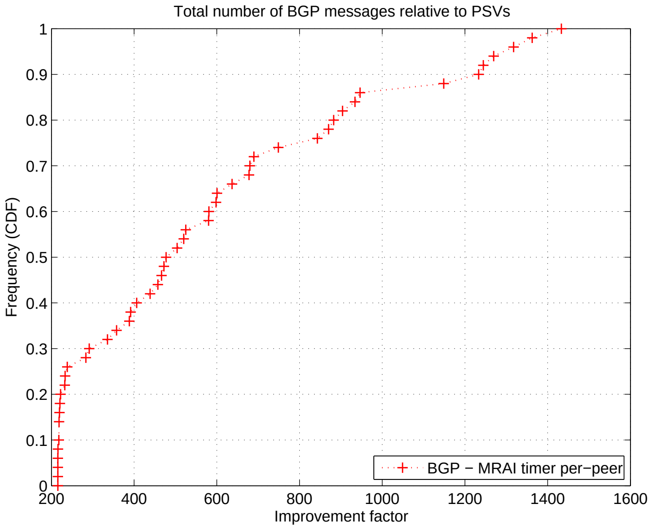
Churn: Unreachable case (MRAI per-peer)



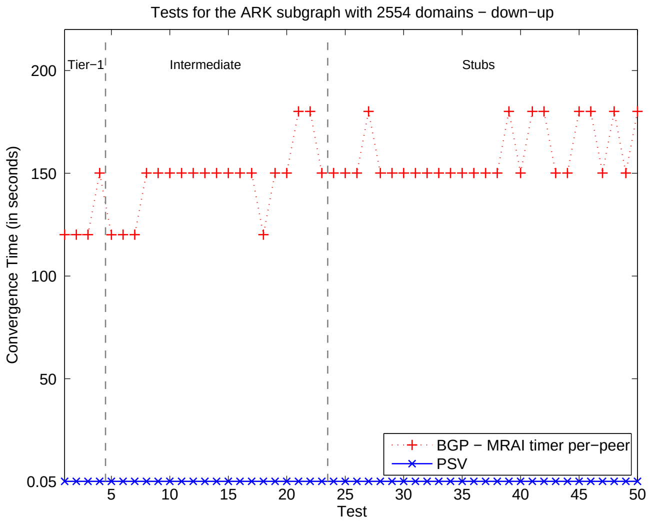
Churn: Unreachable case (MRAI per-peer)



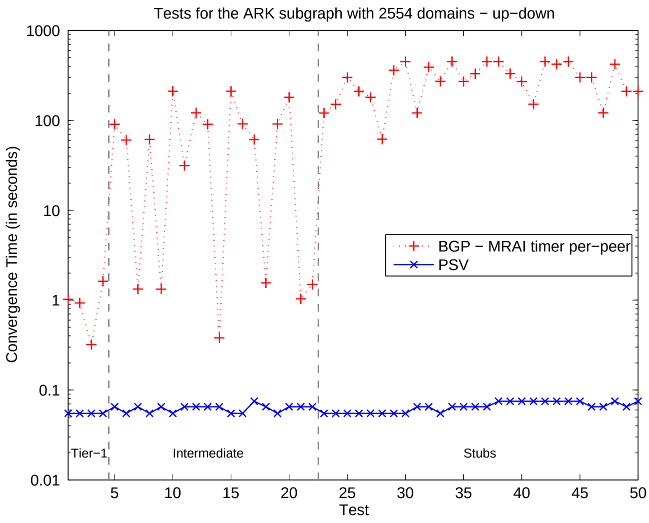
Churn: Unreachable case (MRAI per-peer)



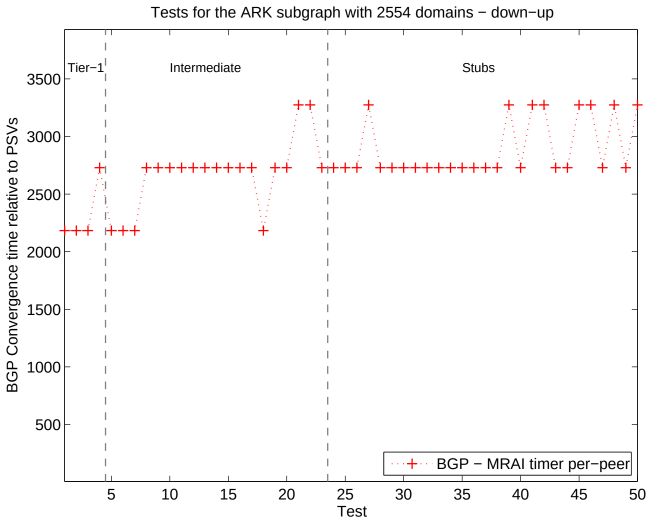
Conv. time: Unreachable case (MRAI per-peer)



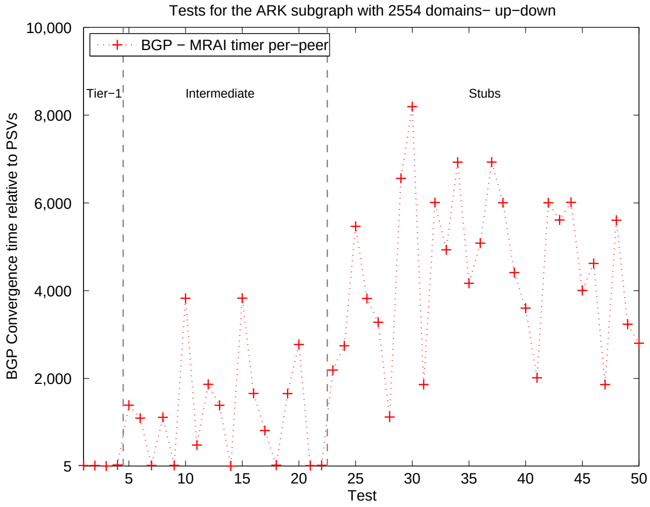
Conv. time: Unreachable case (MRAI per-peer)



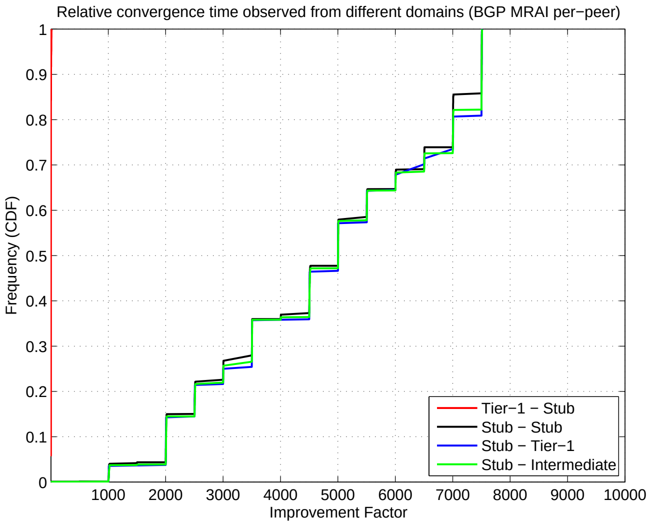
Conv. time: Unreachable case (MRAI per-prefix)



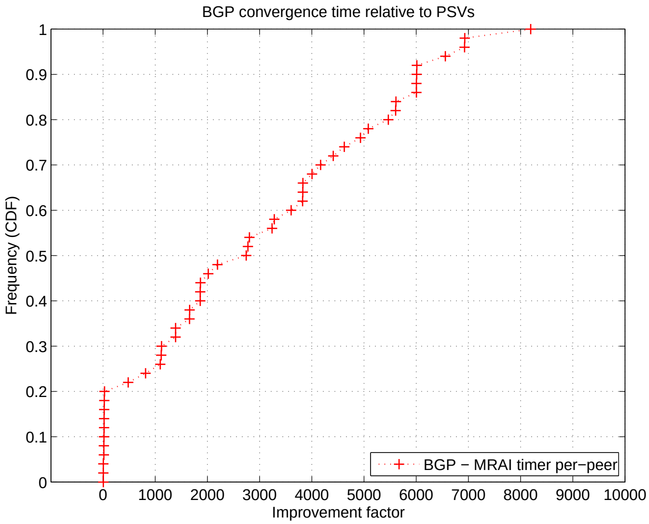
Conv. time: Unreachable case (MRAI per-prefix)



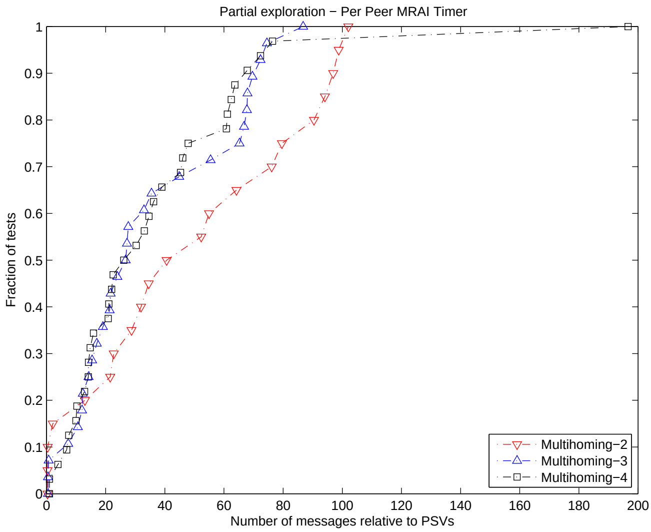
Conv. time: Unreachable case (MRAI per-peer)



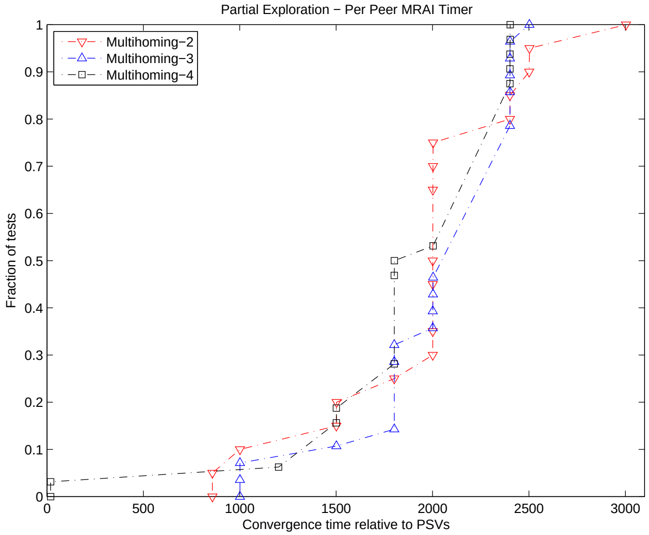
Conv. time: Unreachable case (MRAI per-peer)



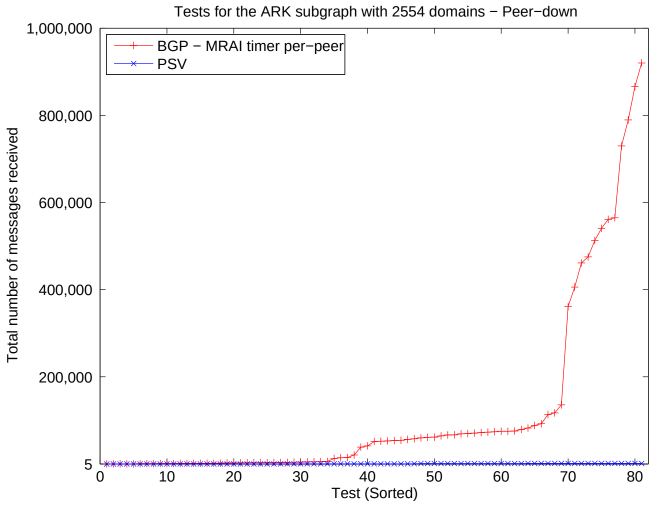
Churn: Multihoming case (MRAI per-peer)



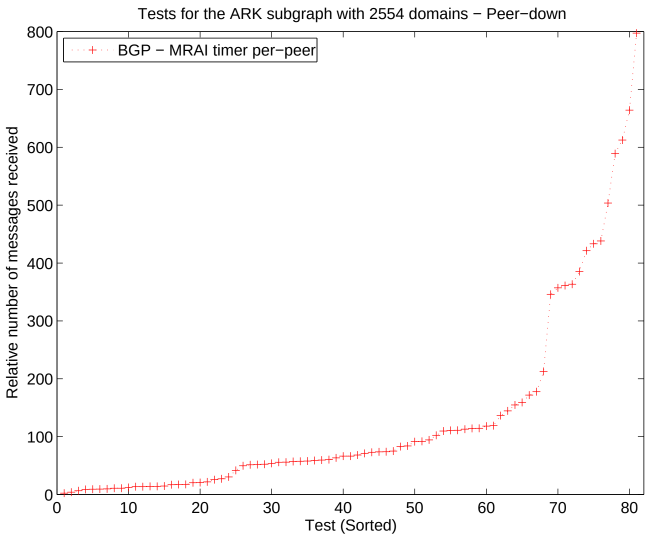
Convergence: Multihoming case (MRAI per-peer)



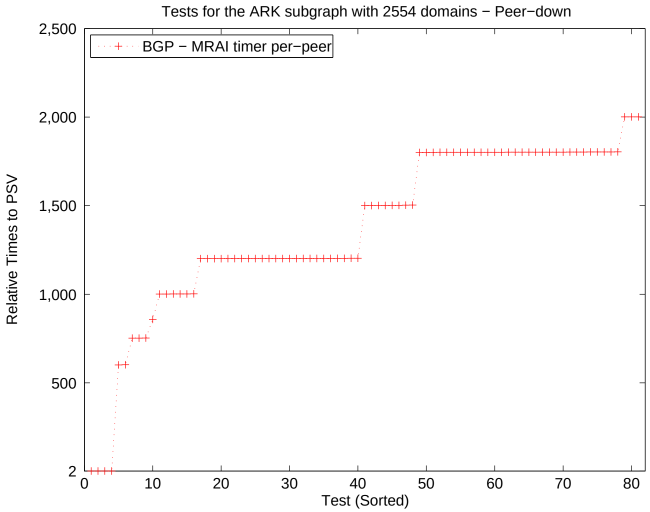
Convergence: Multihoming case (MRAI per-peer)



Convergence: Multihoming case (MRAI per-peer)



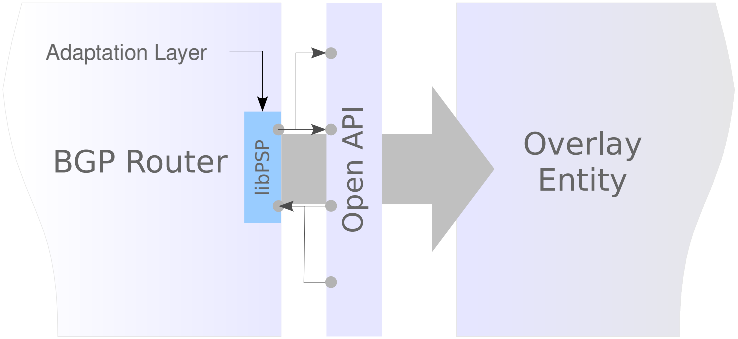
Convergence: Multihoming case (MRAI per-peer)



Outline

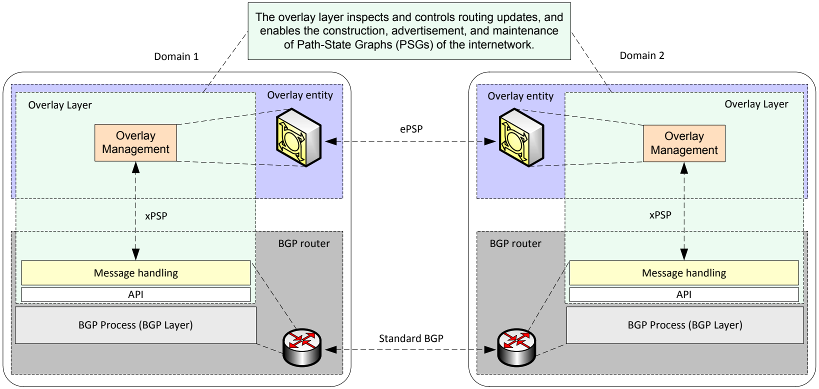
Design Goals

- Dramatically improve routing convergence
- Seamless inspection and control of basic BGP operations $\rightarrow$ enabler for enhanced security



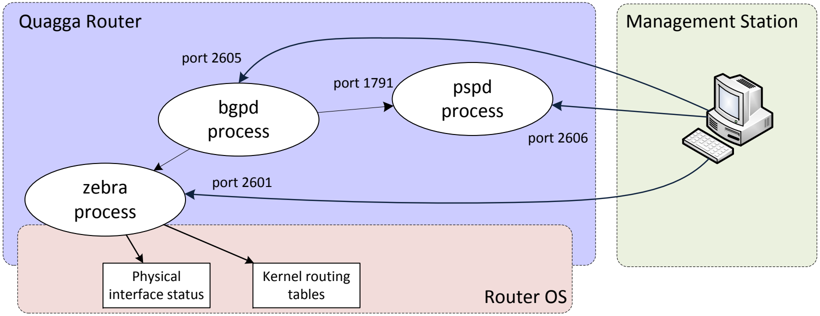
Interactions between the PSP and BGP protocols

- Approach inline with the current trend toward “open APIs” on the router operating system.



Software modules

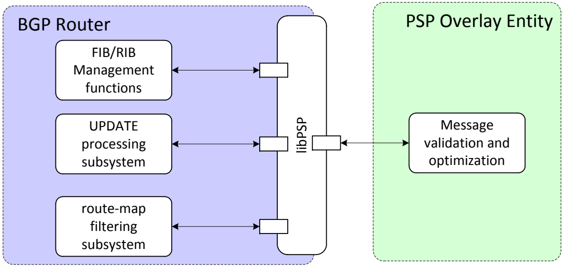
- Standalone process that can interact with BGP and other PSP entities
- It can be executed in the same or different equipment
- It presents active (**normal**) and passive (**bypass**) modes of operation
- Cisco IOS-like CLI and management



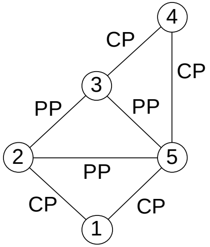
PSP implementation

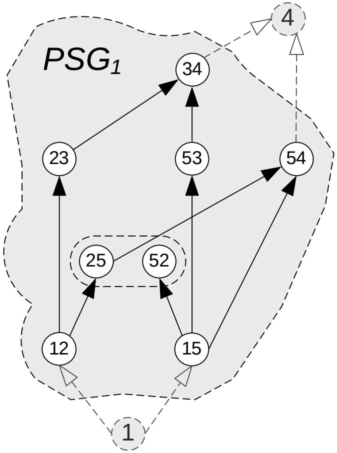
Modularized approach

- `libPSP` uses an Open API-like approach securely exposing internal BGP features to PSP
  - Reduced set of changes in Quagga's code
    - **Changed only 88 LOC to interact with `libPSP`**
  - Asynchronous message processing and management
  - Most of the processing is decoupled from BGP
  - Easily extensible to other BGP functionalities


BGP Router
FIB/RIB Management functions
UPDATE processing subsystem
route-map filtering subsystem
libPSP
PSP Overlay Entity
Message validation and optimization

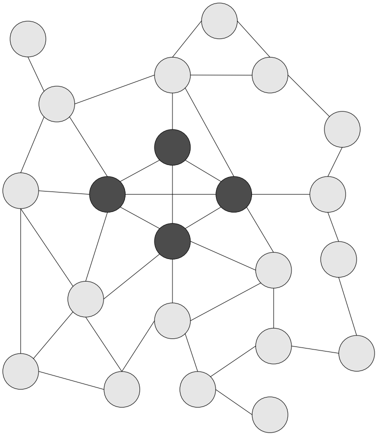
The Path-State Table

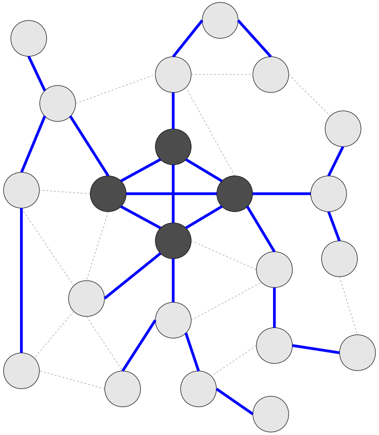




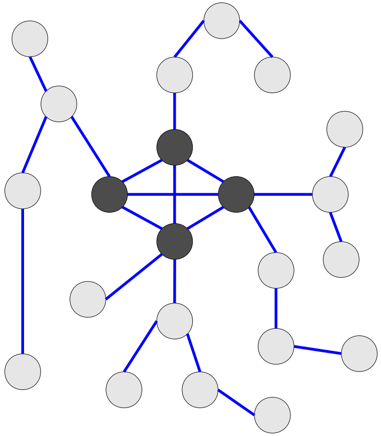
	1	2	3	4	5
1	-	(12) [(15),(52)]	[(12),(23)] [(15),(53)]	[(12),(23),(34)] [(15),(54)] [(12),(25),(54)] [(15),(53),(34)]	(15) [(12),(25)]

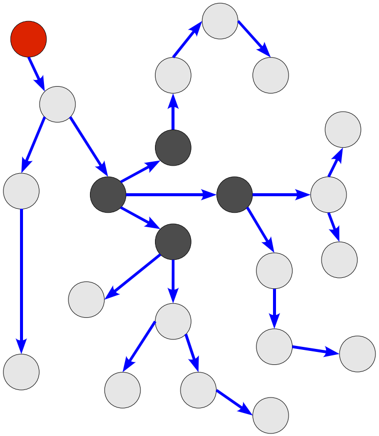
Constructing the Overlay



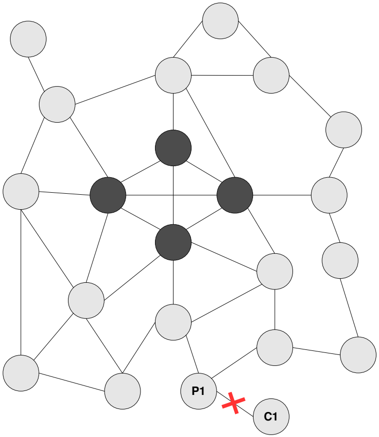


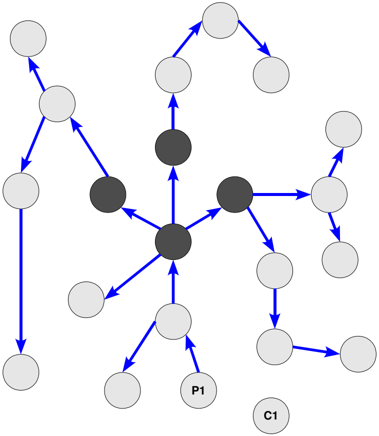
Constructing the Overlay (cont.)



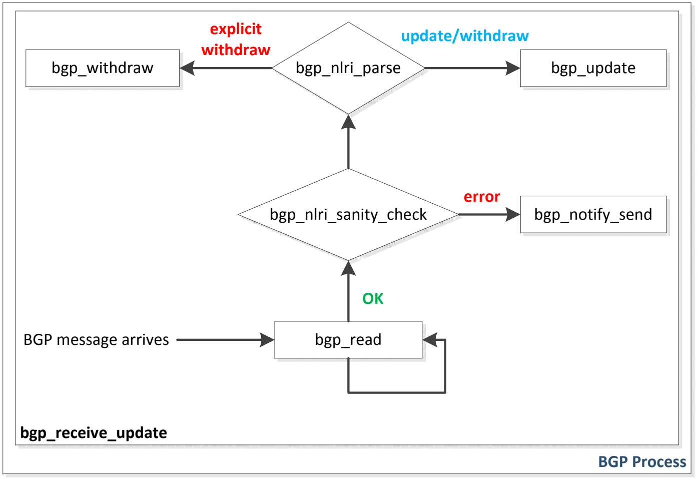


Convergence: Updating the PSGs through the Overlay

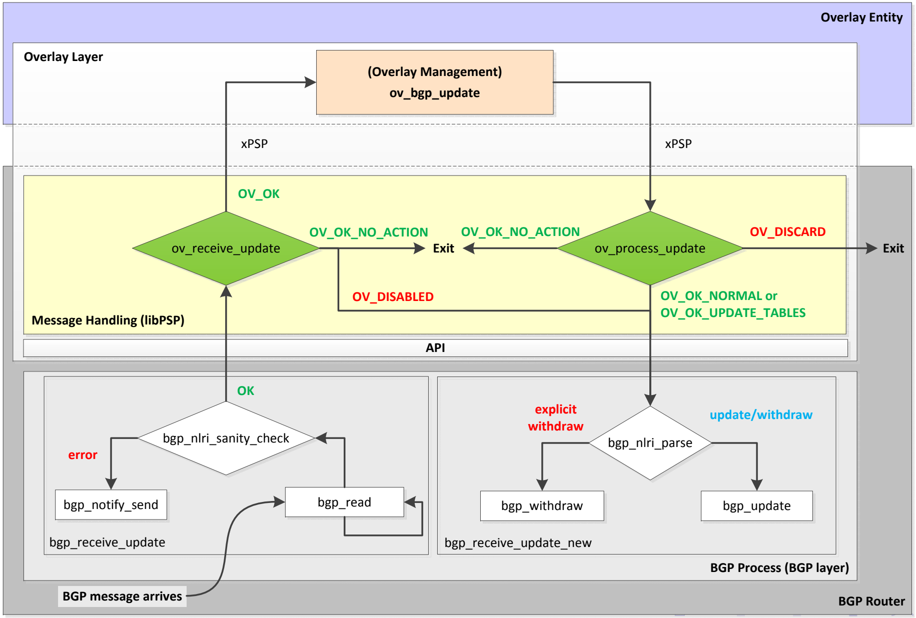




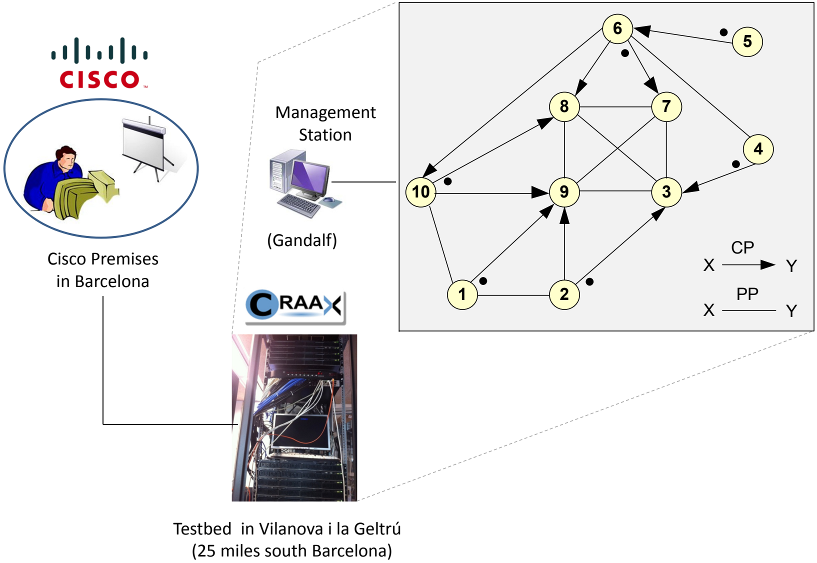
Inside Quagga (a simplified view)



Enabling the Overlay Inspection



Testbed and Topology for the Demo


CISCO
Cisco Premises in Barcelona
Management Station
(Gandalf)
CRAAX
Testbed in Vilanova i la Geltrú (25 miles south Barcelona)
6
5
8
7
4
10
9
3
1
2
X CP Y
X PP Y

Preliminaries: PSP Command Line

Objective

- To build a “friendly” CLI ... that is $\rightarrow$ IOS-like

Preliminaries: PSP Command Line

Configuring PSPs is very similar to configuring BGP

- Management console for PSP (ePSP)
  - **show psp ...**
- Extension of the management console for BGP (xPSP)
  - **show xpsp status**
- Overlay and BGP:
  - May work together
    - **xpsp router-id A.B.C.D**
  - Or they may work independently
    - **no xpsp**

Hooks...

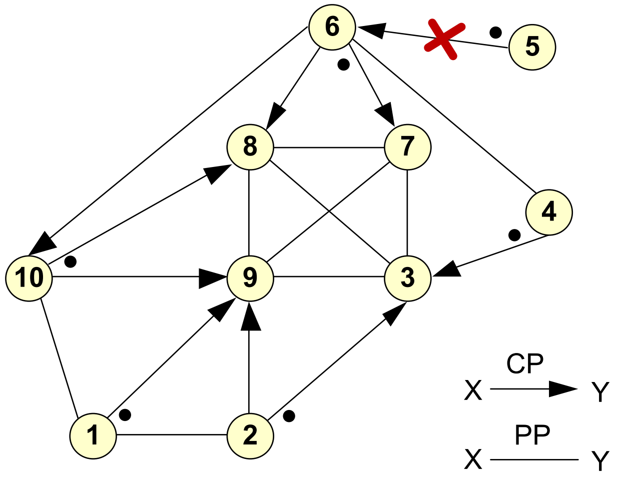
1. Command Line friendly

TEST 1: Convergence and Churn

Objectives and Questions addressed in this TEST

- Can our prototype make a BGP routing system converge without actually triggering a single BGP message?
- And more importantly .... can our prototype converge much faster than BGP?
- Even more .... can our prototype make a BGP routing system converge without actually using routing?

Test 1: inter-domain link failure (scenario)

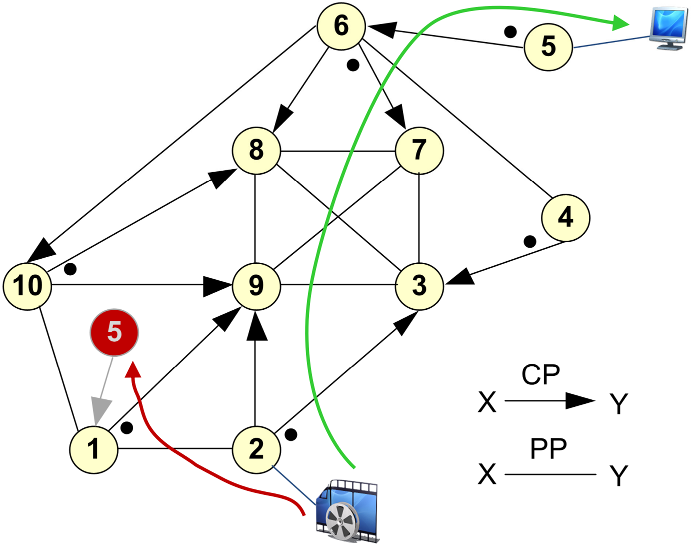


TEST 2: Route Hijacking

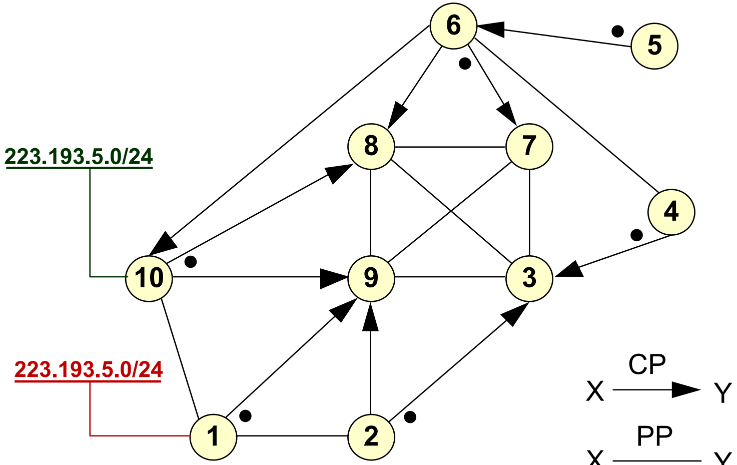
Objectives and Questions addressed in this TEST

- Can the PSP overaly be an enabler for validating and thus accepting/rejecting AS-paths?
- In particular, can our prototype be an enabler for avoiding route hijacking?

TEST 2: Route Hijacking (scenario)



TEST 3: Invalid Prefix Advertisement (scenario)



Hooks...

1. Command Line friendly
2. Enables inspection of BGP updates in a non-disruptive way (88 LOC)
3. Dramatically reduces both
   - The BGP convergence time
   - The churn rate of updates ... which provides credit for security overhead
4. Pure forwarding **(avoids relying on the routing to convey information about the routing system we want to secure)**
5. Events are transmitted optimally through the overlay on Internet-like graphs (compact routing)
6. **The information distributed through the overlay only contains data created, processed, and advertised by the overlay**...we do not trust BGP for constructing and maintaining the PSGs.
7. **Our prototype is an enabler for:**
   - **path validation**
   - **prefix ownership validation**

Developments made in this SRA

- Reach of our prototype at this stage:

  ✓ libPSP
  ✓ xPSP
  ✓ ePSP
  ✓ BGP message inspection (e.g., path and origin)

  ☒ Secure path and origin validation
  ☒ iPSP

Outline

Questions?