Disruption-free routing convergence

Computing minimal link-state update sequences

François CLAD

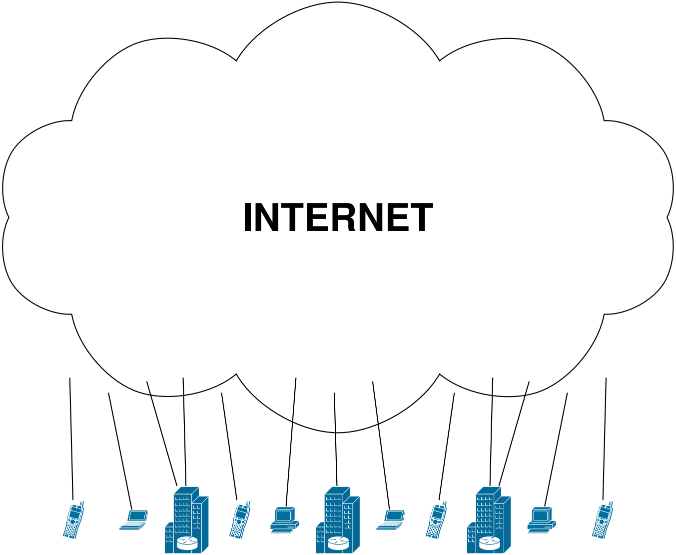
Jury de soutenance :

M. Jean-Jacques PANSIOT	Université de Strasbourg, Directeur de thèse
M. Pascal MERINDOL	Université de Strasbourg, Co-encadrant de thèse
Mme. Catherine ROSENBERG	University of Waterloo, Rapporteuse externe
M. Guy LEDUC	Université de Liège, Rapporteur externe
M. David COUDERT	INRIA Sophia Antipolis, Examinateur
M. Thomas NOËL	Université de Strasbourg, Examinateur

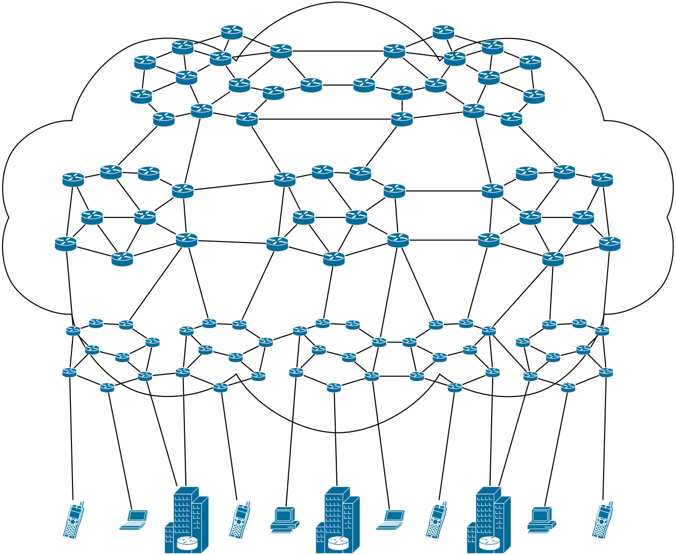
Outline

Outline

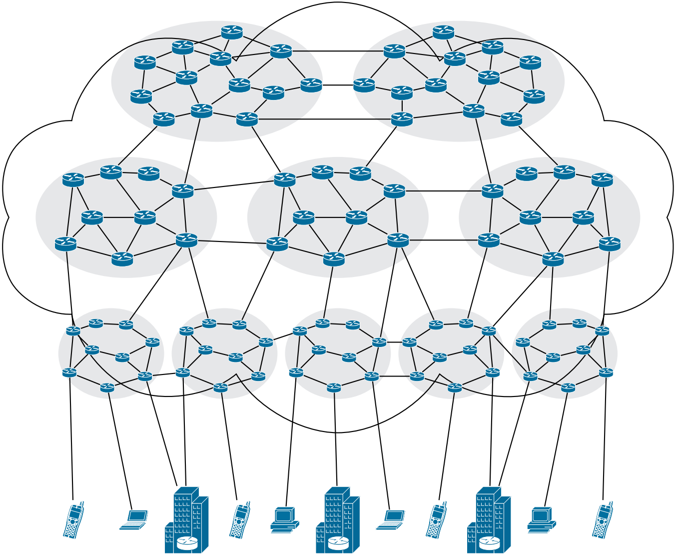
Architecture of the Internet



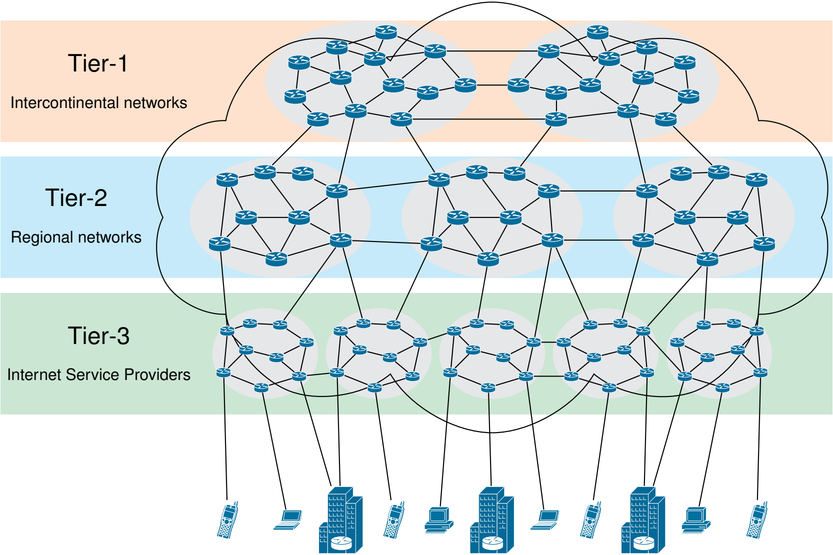
Architecture of the Internet



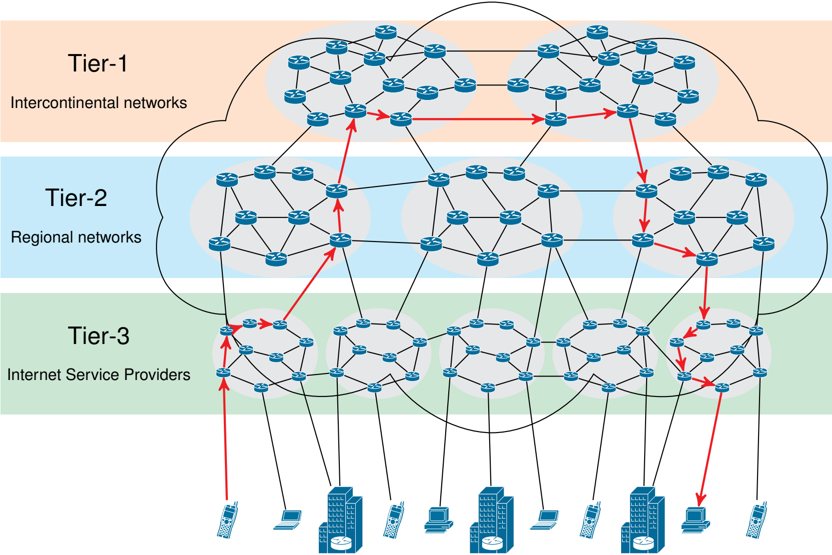
Architecture of the Internet



Architecture of the Internet



Architecture of the Internet



Routing and forwarding

- Forwarding: *data plane*
  - Reads the Forwarding Information Base (FIB)
    - Optimized for fast lookup
  - Steers data packets along best paths
    - Hop-by-hop forwarding (or through tunnels)

- Routing: *control plane*
  - Collects information about the network
    - Relies on *signalization messages*
  - Writes *best paths* in the Routing Information Base (RIB)
    - Various metrics: pricing, hop count, link capacity, delay,...

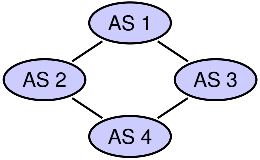
Routing and forwarding

- Forwarding: *data plane*
  - ▷ Reads the Forwarding Information Base (FIB)
    - ⋆ Optimized for fast lookup
  - ▷ Steers data packets along best paths
    - ⋆ Hop-by-hop forwarding (or through tunnels)

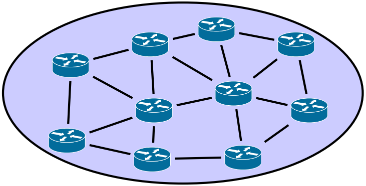
- Routing: *control plane*
  - ▷ Collects information about the network
    - ⋆ Relies on *signalization messages*
  - ▷ Writes *best paths* in the Routing Information Base (RIB)
    - ⋆ Various metrics: pricing, hop count, link capacity, delay,. . .

Routing basics

- Interdomain routing
  - Graph of ASes
  - Based on *business relationships*
  - Limited cooperation

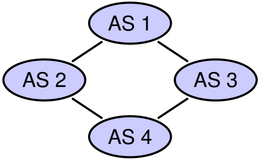


- Intradomain routing
  - Graph of routers, within an AS
  - Based on *shortest paths*
  - Operated by a single institution

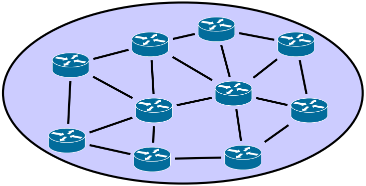


Routing basics

- Interdomain routing
  - Graph of ASes
  - Based on *business relationships*
  - Limited cooperation



- Intradomain routing
  - Graph of routers, within an AS
  - Based on *shortest paths*
  - Operated by a single institution



Intra-domain routing protocols

- Distance-vector protocols
  - ▷ Distance information from neighbors
  - ▷ Slow convergence, poor scalability
  - Routing Information Protocol (RIP) – IETF[1]
  - Interior Gateway Routing Protocol (IGRP) – Cisco
- Link-state protocols
  - ▷ State of each link flooded to all routers
  - ▷ Fast convergence, better scalability
  - Open Shortest Path First (OSPF) – IETF[1]
  - Intermediate System-to-Intermediate System (IS-IS) – ISO[2]

---

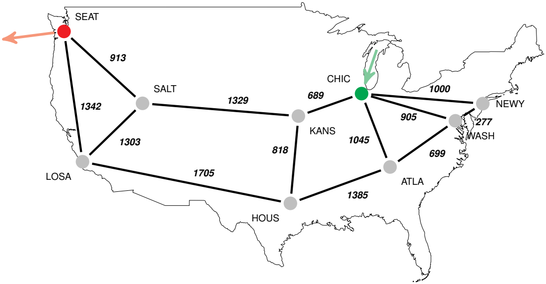
[1] IETF: Internet Engineering Task Force
[2] ISO: International Organization for Standardization

Intra-domain routing protocols

- Distance-vector protocols
  - ▷ Distance information from neighbors
  - ▷ Slow convergence, poor scalability
  - Routing Information Protocol (RIP) – IETF[1]
  - Interior Gateway Routing Protocol (IGRP) – Cisco
- Link-state protocols
  - ▷ State of each link flooded to all routers
  - ▷ Fast convergence, better scalability
  - Open Shortest Path First (OSPF) – IETF[1]
  - Intermediate System-to-Intermediate System (IS-IS) – ISO[2]

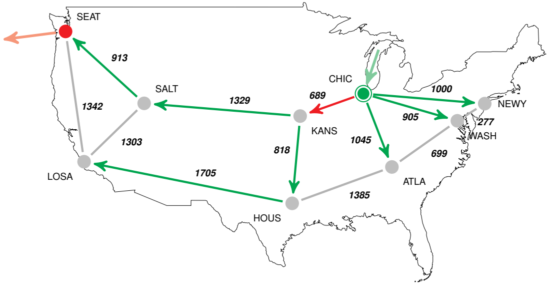
[1] IETF: Internet Engineering Task Force
[2] ISO: International Organization for Standardization

Illustration of link-state routing



Internet2 IP network

Illustration of link-state routing

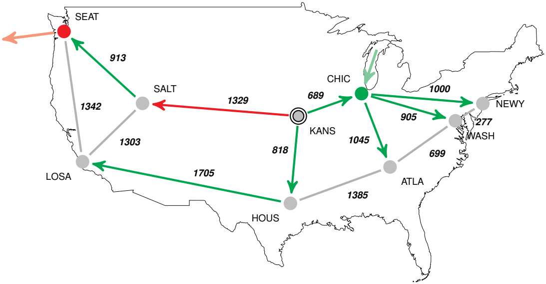


Internet2 IP network

Destination	Next-hop	Distance
SEAT	KANS	2931
LOSA	KANS	3212
SALT	KANS	2018
HOUS	KANS	1507
KANS	KANS	689
CHIC	–	0
ATLA	ATLA	1045
WASH	WASH	907
NEWY	NEWY	1000

Routing table of CHIC

Illustration of link-state routing

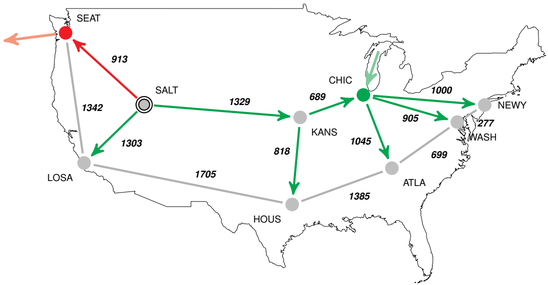


Internet2 IP network

Destination	Next-hop (CHIC)	Distance (CHIC)	Next-hop (KANS)	Distance (KANS)
SEAT	KANS	2931	SALT	2242
LOSA	KANS	3212	HOUS	2523
SALT	KANS	2018	SALT	1329
HOUS	KANS	1507	HOUS	818
KANS	KANS	689	–	0
CHIC	–	0	CHIC	689
ATLA	ATLA	1045	CHIC	1734
WASH	WASH	907	CHIC	1596
NEWY	NEWY	1000	CHIC	1689

Routing table of CHIC *Routing table of KANS*

Illustration of link-state routing

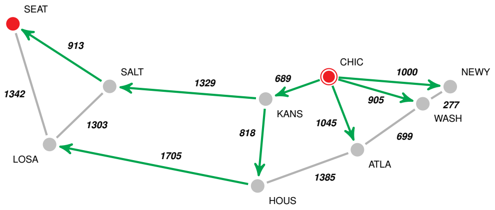


Internet2 IP network

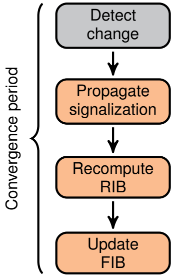
Destination	Next-hop	Distance	Next-hop	Distance	Next-hop	Distance
SEAT	KANS	2931	SALT	2242	SEAT	913
LOSA	KANS	3212	HOUS	2523	LOSA	1303
SALT	KANS	2018	SALT	1329	–	0
HOUS	KANS	1507	HOUS	818	KANS	2147
KANS	KANS	689	–	0	KANS	1329
CHIC	–	0	CHIC	689	KANS	2018
ATLA	ATLA	1045	CHIC	1734	KANS	3063
WASH	WASH	907	CHIC	1596	KANS	2923
NEWY	NEWY	1000	CHIC	1689	KANS	3018

Routing table of CHIC *Routing table of KANS* *Routing table of SALT*

Topological changes and routing convergence



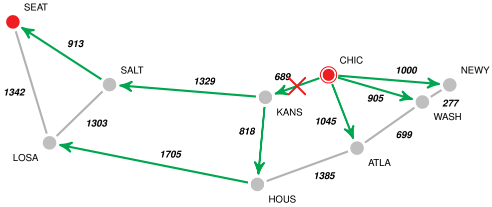
Shortest paths from Chicago



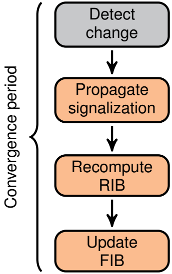
Initial route

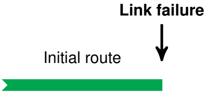
Convergence of the router at Chicago

Topological changes and routing convergence



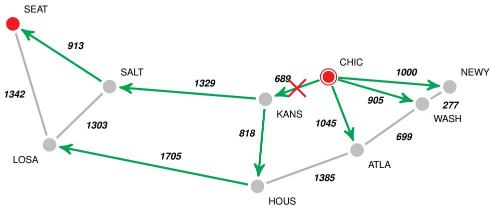
Shortest paths from Chicago



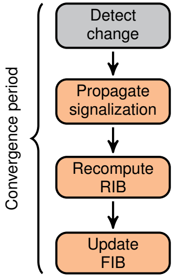


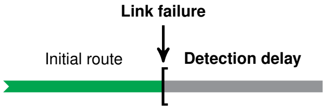
Convergence of the router at Chicago

Topological changes and routing convergence



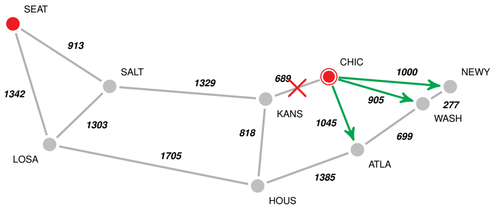
Shortest paths from Chicago



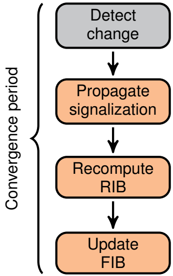


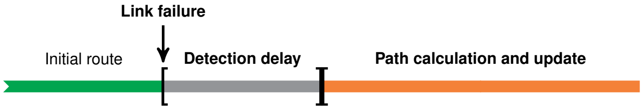
Convergence of the router at Chicago

Topological changes and routing convergence



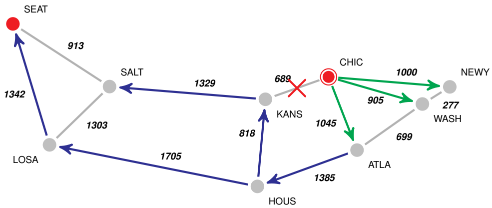
Shortest paths from Chicago



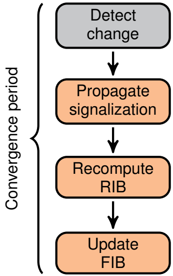


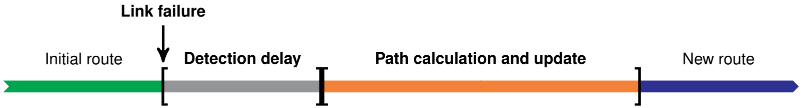
Convergence of the router at Chicago

Topological changes and routing convergence



Shortest paths from Chicago



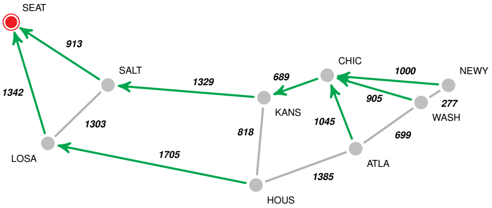


Convergence of the router at Chicago

Outline

Outline

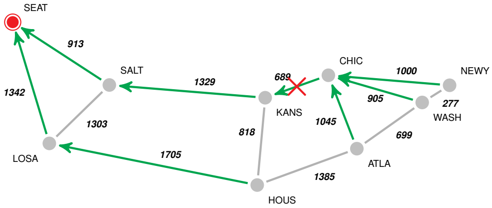
Transient routing inconsistencies



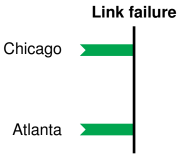
Shortest paths towards Seattle

Convergence of the routers at Chicago and Atlanta

Transient routing inconsistencies

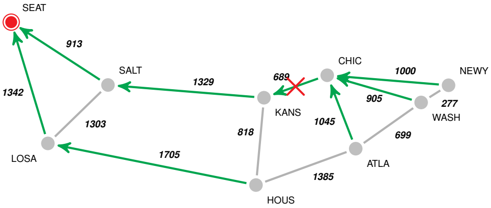


Shortest paths towards Seattle

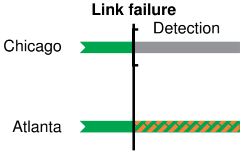


Convergence of the routers at Chicago and Atlanta

Transient routing inconsistencies

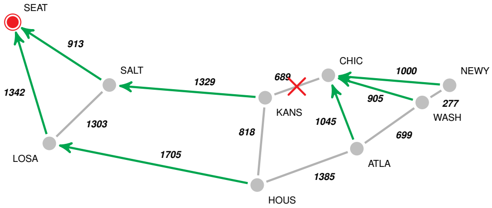


Shortest paths towards Seattle

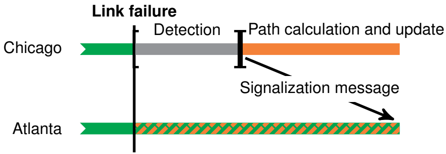


Convergence of the routers at Chicago and Atlanta

Transient routing inconsistencies

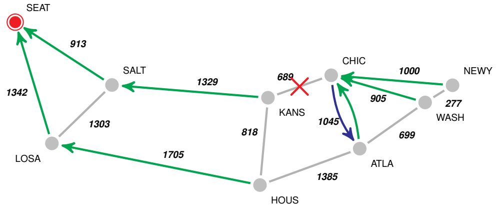


Shortest paths towards Seattle

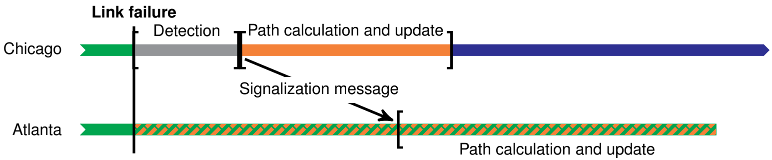


Convergence of the routers at Chicago and Atlanta

Transient routing inconsistencies

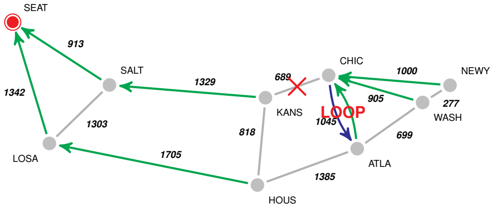


Shortest paths towards Seattle

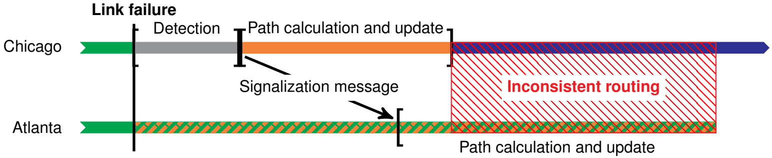


Convergence of the routers at Chicago and Atlanta

Transient routing inconsistencies

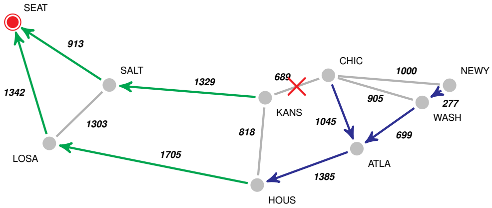


Shortest paths towards Seattle

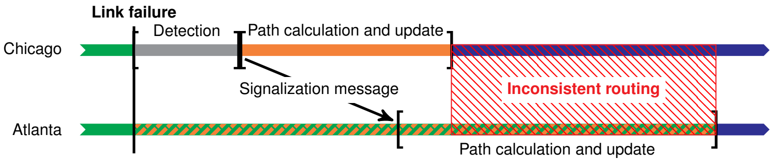


Convergence of the routers at Chicago and Atlanta

Transient routing inconsistencies



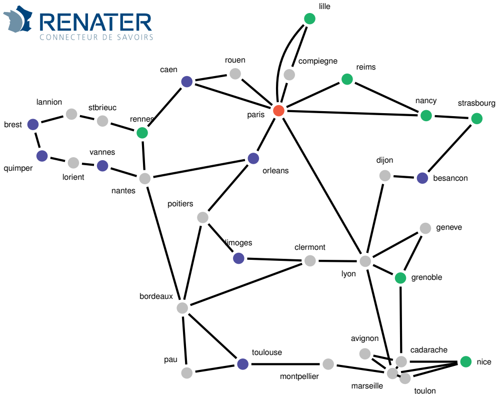
Shortest paths towards Seattle



Convergence of the routers at Chicago and Atlanta

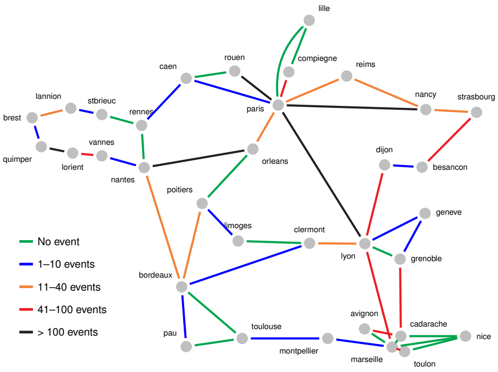
Outline

Experimental setup on a real ISP network



- 1 *IS-IS listener*: device recording every topological modification
- 10 *Raspberry Pi*: vantage points directly connected to RENATER routers
- 8 *PlanetLab nodes*: server-class machines used for network measurements

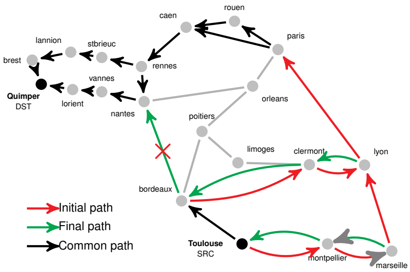
Routing events collected with the IS-IS listener



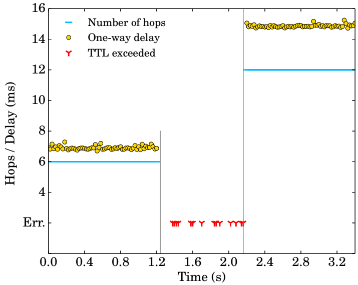
From June 6th to 27th and from July 24th to September 1st (61 days)

▷ 8956 signalization messages changing the topology (146 per day)

Transient loop illustration on a directed path



SPDAG from Toulouse to Quimper



Traffic disruptions on the route from Toulouse to Quimper

Failure on the link from *Bordeaux* to *Nantes*:

- Black-hole period of about 100ms
- Transient loop between *Montpellier* and *Marseille* for 900ms

Consequences of transient loops

Direct consequences

- Increased transmission delays
- Packet losses (exceeded TTL)
- Link saturation and congestions

Indirect consequences

- Link saturation and congestions
- Packet losses
  - ▷ **Worst case:** adjacency failure due to loss of signalization

Outline

Timer-based solutions

- Ordered FIB updates (oFIB)[3]
  - ▷ Removal / weight increment: farthest routers first
  - ▷ Addition / weight decrement: closest routers first
  - Prevents all transient loops
  - Non-incremental deployment

- Local delay[4]
  - ▷ 1-hop oFIB
  - Purely local solution
  - Prevents only local transient loops

[3] **pfr:2007:ofib**.
[4] **draft:uloop-delay**.

Other protocol extensions

- Tunneling[5]
  - ▷ Packet encapsulation during the convergence
  - Works for both link and node events
  - Non-incremental deployment

- Ships-in-the-Night (SITN)[6]
  - ▷ Two concurrent control planes on each router
  - ▷ Ordered migration from the old process to the new one
  - Supports network-wide migrations
  - Non-incremental deployment
  - Huge overhead for single link or node modifications

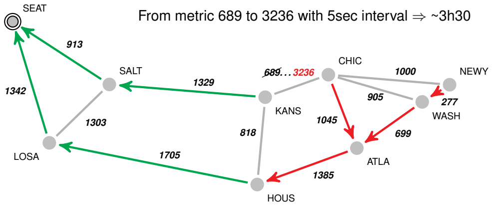
[5] **rfc5715**.
[6] **vanbever:2012**.

Progressive weight reconfigurations[7,8]

Theorem

In a stable network, incrementing or decrementing the weight of a link by 1 leads to a loop-free convergence.

- Loop-free weight update sequence for any single-link reconfiguration
- Complete network convergence required at each step



[7] **ito:2003**.
[8] **pfr:2007:mincr**.

Progressive weight reconfigurations

- Relies on a core functionality of link-state routing
  - No protocol extension
  - Incrementally deployable
- Slow down the convergence
- Only single link reconfigurations

Outline

Objectives

- Improve the progressive reconfiguration solution
  - ▷ Minimize operational impact (sequence length)
  - ▷ Provide time-efficient algorithms
- Generalize the approach to router-wide reconfigurations
  - ▷ Minimize operational impact (sequence length)
  - ▷ Provide time-efficient algorithms
  - ▷ Prevent routing instabilities

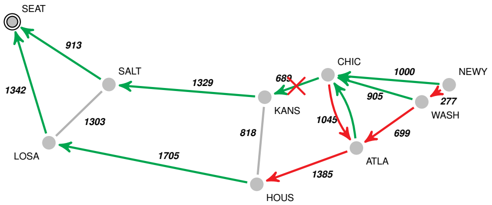
Outline

1. Introduction
2. Context
3. Contributions
   - Minimum link reconfiguration sequences
   - Generalization to router-wide operations

Pivot weight increment

For a given destination d, we define for each router a pivot increment, denoted $\Delta_d(x)$:

$$\forall\, x \in N, \quad \Delta_d(x) = C'(x,d) - C(x,d)$$



x	$C(x)$	$C'(x)$	$\Delta_{SEAT}(x)$
SEAT	0	0	0
LOSA	1342	1342	0
SALT	913	913	0
HOUS	3047	3047	0
KANS	2242	2242	0
CHIC	2931	**5477**	2546
ATLA	3976	**4432**	456
WASH	3836	**6176**	2340
NEWY	3931	**6453**	2522

General definitions:

$G(N,E,w)$	Directed weighted graph representing the network
$C(x,d)$, $C(x)$	Cost of a shortest path (*distance*) from x to d before the change
$C'(x,d)$, $C'(x)$	Cost of a shortest path (*distance*) from x to d after the change

Delta properties

Let i be an increment performed on the modified link

- If $i < \Delta_d(x)$, x only uses its initial paths towards d
- If $i = \Delta_d(x)$, x uses both its initial and final paths towards d (ECMP)
- If $i > \Delta_d(x)$, x only uses its final paths towards d

$i < 456$	$i = 456$	$i > 456$

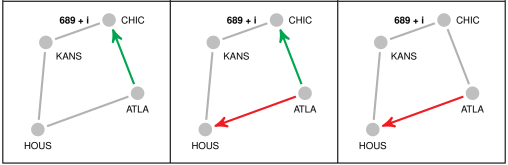
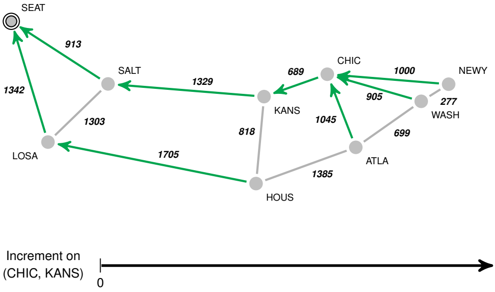


Illustration for the router at Atlanta: $\Delta_{SEAT}(ATLA) = 456$

Delta sequence

Lemma

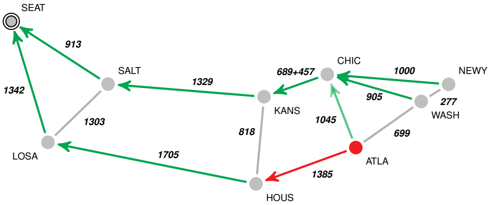
Given a destination d, the sequence of sorted $\Delta_d(x)+1$ increments for every router x in N provides a loop- free convergence for this destination.

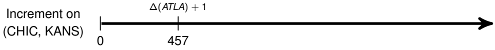


Delta sequence

Lemma

Given a destination d, the sequence of sorted $\Delta_d(x) + 1$ increments for every router x in N provides a loop- free convergence for this destination.



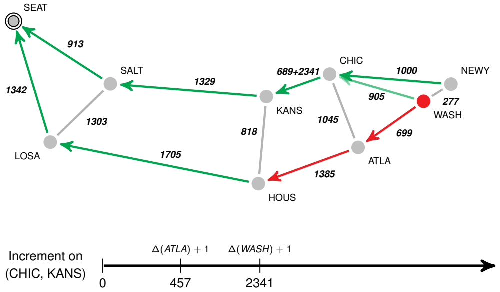


$+457$ Atlanta reroutes from ($ATLA$, $CHIC$) to ($ATLA$, $HOUS$)

Delta sequence

Lemma

Given a destination d, the sequence of sorted $\Delta_d(x) + 1$ increments for every router x in N provides a loop- free convergence for this destination.



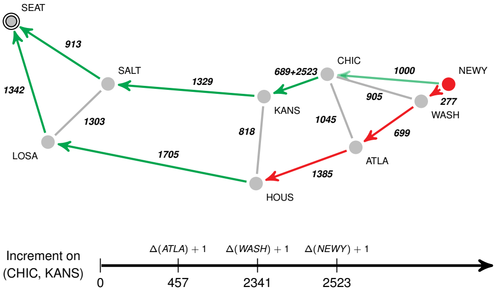
$+457$ Atlanta reroutes from ($ATLA$, $CHIC$) to ($ATLA$, $HOUS$)

$+2341$ Washington reroutes from ($WASH$, $CHIC$) to ($WASH$, $ATLA$)

Delta sequence

Lemma

Given a destination d, the sequence of sorted $\Delta_d(x) + 1$ increments for every router x in N provides a loop- free convergence for this destination.



$+457$ Atlanta reroutes from $(ATLA, CHIC)$ to $(ATLA, HOUS)$

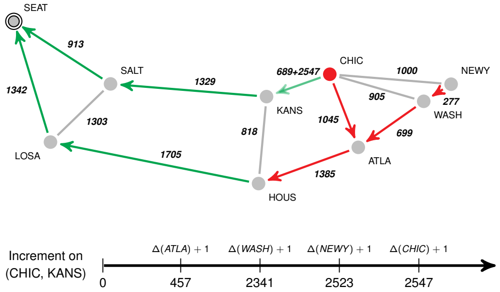
$+2341$ Washington reroutes from $(WASH, CHIC)$ to $(WASH, ATLA)$

$+2523$ New-York reroutes from $(NEWY, CHIC)$ to $(NEWY, WASH)$

Delta sequence

Lemma

Given a destination d, the sequence of sorted $\Delta_d(x) + 1$ increments for every router x in N provides a loop- free convergence for this destination.



+457 Atlanta reroutes from (*ATLA*, *CHIC*) to (*ATLA*, *HOUS*)

+2341 Washington reroutes from (*WASH*, *CHIC*) to (*WASH*, *ATLA*)

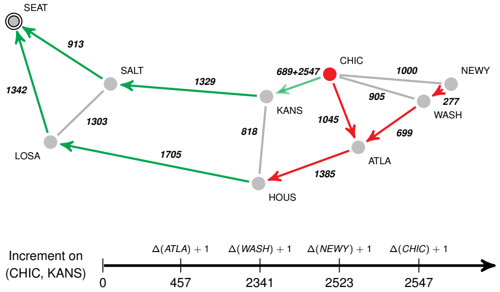
+2523 New-York reroutes from (*NEWY*, *CHIC*) to (*NEWY*, *WASH*)

+2547 Chicago reroutes from (*CHIC*, *KANS*) to (*CHIC*, *ATLA*)

Delta sequence

Lemma

Given a destination d, the sequence of sorted $\Delta_d(x) + 1$ increments for every router x in N provides a loop- free convergence for this destination.



$+457$ Atlanta reroutes from $(ATLA, CHIC)$ to $(ATLA, HOUS)$

$+2341$ Washington reroutes from $(WASH, CHIC)$ to $(WASH, ATLA)$

$+2523$ New-York reroutes from $(NEWY, CHIC)$ to $(NEWY, WASH)$

$+2547$ Chicago reroutes from $(CHIC, KANS)$ to $(CHIC, ATLA)$

Shorter sequences... but still too long.

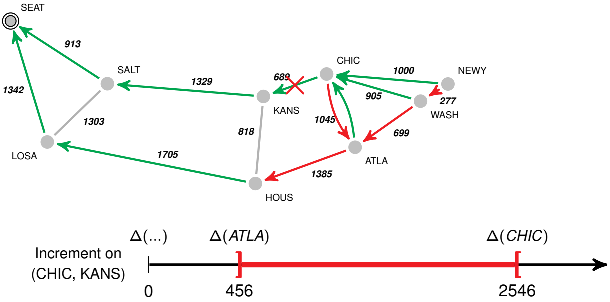
Intervals

Theorem

A monotonic weight update sequence S prevents a transient loop $L = \{x_1, x_2, \ldots, x_1\}$ for a destination d, if and only if there exists $e \in S$ such that:

$$MIN_{\forall x \in L}(\Delta_d(x)) < e < MAX_{\forall x \in L}(\Delta_d(x))$$

The sequence must contain a weight update that makes one router involved in the loop to completely reroute, while another is still in its initial routing state.



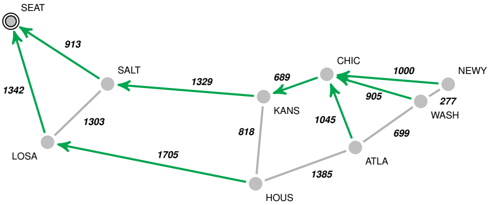
Intervals

Theorem

A monotonic weight update sequence S prevents a transient loop $L = \{x_1, x_2, \ldots, x_1\}$ for a destination d, if and only if there exists $e \in S$ such that:

$$MIN_{\forall x \in L}(\Delta_d(x)) < e < MAX_{\forall x \in L}(\Delta_d(x))$$

The sequence must contain a weight update that makes one router involved in the loop to completely reroute, while another is still in its initial routing state.



$e < 456$ Nothing happens...

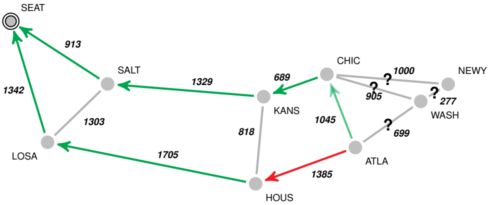
Intervals

Theorem

A monotonic weight update sequence S prevents a transient loop $L = \{x_1, x_2, \ldots, x_1\}$ for a destination d, if and only if there exists $e \in S$ such that:

$$MIN_{\forall x \in L}(\Delta_d(x)) < e < MAX_{\forall x \in L}(\Delta_d(x))$$

The sequence must contain a weight update that makes one router involved in the loop to completely reroute, while another is still in its initial routing state.



$e < 456$ Nothing happens...

$456 < e < 2546$ Atlanta reroutes.

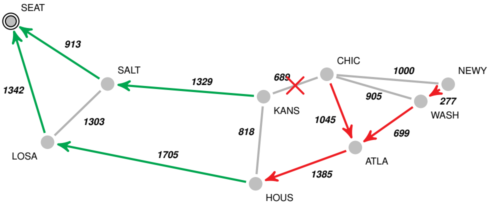
Intervals

Theorem

A monotonic weight update sequence S prevents a transient loop $L = \{x_1, x_2, \ldots, x_1\}$ for a destination d, if and only if there exists $e \in S$ such that:

$$MIN_{\forall x \in L}(\Delta_d(x)) < e < MAX_{\forall x \in L}(\Delta_d(x))$$

The sequence must contain a weight update that makes one router involved in the loop to completely reroute, while another is still in its initial routing state.

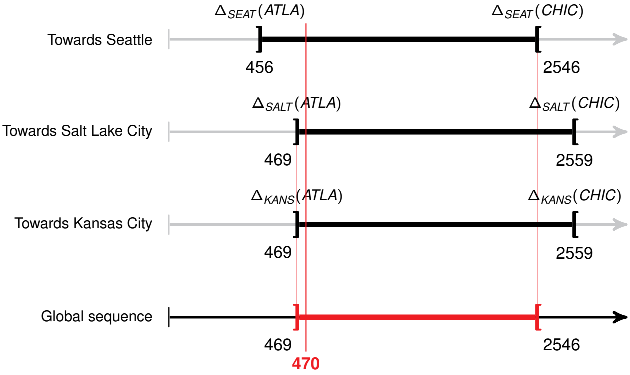


$e < 456$ Nothing happens...

$456 < e < 2546$ Atlanta reroutes.

$e > 2546$ Chicago reroutes.

Global sequence



Minimum loop-free sequence for the link $(CHIC, KANS)$: $S = \{470\}$

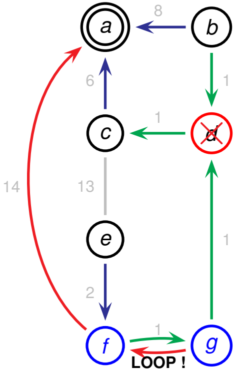
Outline

1. Introduction
2. Context
3. Contributions
   - Minimum link reconfiguration sequences
   - Generalization to router-wide operations

Possible approaches

- Link-by-link reconfiguration
  - Same problem as single-link
  - Routing instabilities
  - Sequence length proportional to the node degree
- Uniform multi-link reconfiguration
  - Same problem as single-link (virtual weight on the router)
  - Routing stability
  - Long sequences in some cases
- Non-uniform multi-link reconfiguration
  - Allow for minimal sequence length, but. . .
    - ▷ Multi-dimensional problem
    - ▷ Possible routing instabilities

Multi-dimensional pivot increments

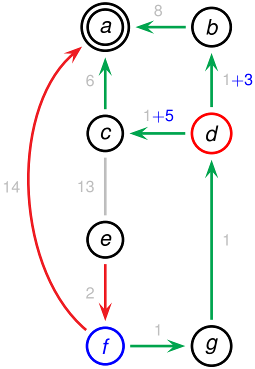


Minimal weight increment vector such that a node x uses a new path, not via d, to reach a.

$$\Delta_a(x)[i] = C'(x, a) - C(x, i, a)^9$$

[9] $C(x, i, a)$: the cost of a shortest path from x to d plus the cost of a shortest *simple path* from d to a via (d, i)

Multi-dimensional pivot increments



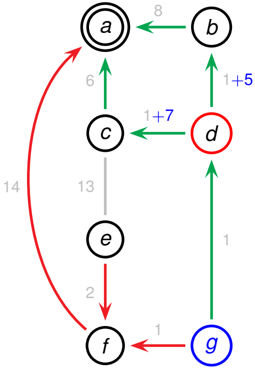
Minimal weight increment vector such that a node x uses a new path, not via d, to reach a.

$$\Delta_a(x)[i] = C'(x, a) - C(x, i, a)^9$$

- $\Delta_a(f) = \begin{pmatrix} 14 - (1 + 1 + 1 + 6) \\ 14 - (1 + 1 + 1 + 8) \end{pmatrix} = \begin{pmatrix} 5 \\ 3 \end{pmatrix}$

[9] $C(x, i, a)$: the cost of a shortest path from x to d plus the cost of a shortest *simple path* from d to a via (d, i)

Multi-dimensional pivot increments



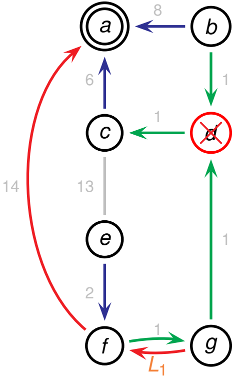
Minimal weight increment vector such that a node x uses a new path, not via d, to reach a.

$$\Delta_a(x)[i] = C'(x, a) - C(x, i, a)^9$$

- $\Delta_a(f) = \begin{pmatrix} 14 - (1 + 1 + 1 + 6) \\ 14 - (1 + 1 + 1 + 8) \end{pmatrix} = \begin{pmatrix} 5 \\ 3 \end{pmatrix}$
- $\Delta_a(g) = \begin{pmatrix} 15 - 8 \\ 15 - 10 \end{pmatrix} = \begin{pmatrix} 7 \\ 5 \end{pmatrix}$

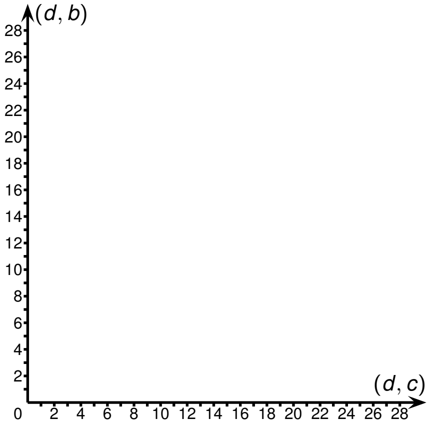
[9] $C(x, i, a)$: the cost of a shortest path from x to d plus the cost of a shortest *simple path* from d to a via (d, i)

Modeling transient loops as constraints



$$\Delta_a^d(f) = \binom{5}{3}$$

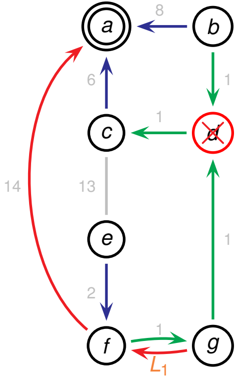
$$\Delta_a^d(g) = \binom{7}{5}$$



Constraint c associated with a loop L

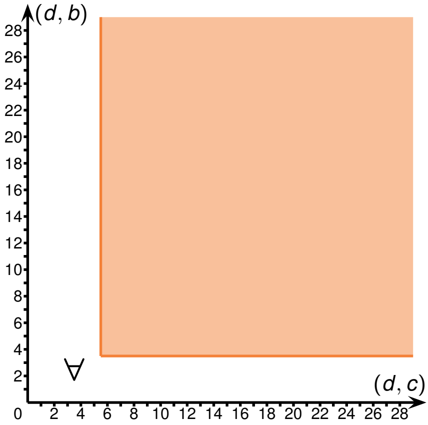
$$c := (\min_{\forall x \in L}(\Delta(x)), \max_{\forall x \in L}(\Delta(x)))$$

Modeling transient loops as constraints



$$\Delta_a^d(f) = \binom{5}{3}$$

$$\Delta_a^d(g) = \binom{7}{5}$$

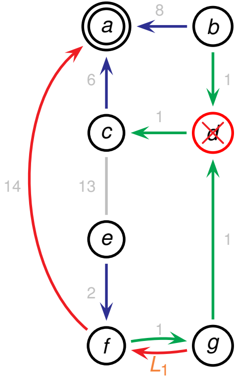


Constraint c associated with a loop L

$$c := \left(\min_{\forall x \in L}(\Delta(x)), \max_{\forall x \in L}(\Delta(x))\right)$$

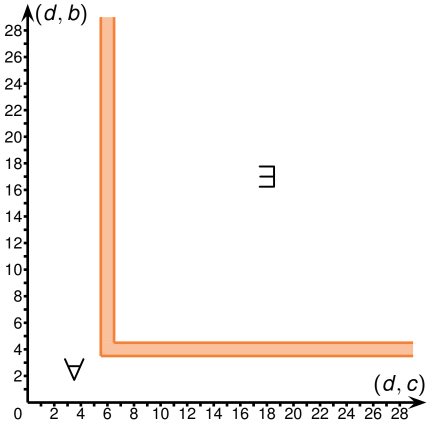
$$c_1 = \left(\binom{5}{3}\right.$$

Modeling transient loops as constraints



$$\Delta_a^d(f) = \binom{5}{3}$$

$$\Delta_a^d(g) = \binom{7}{5}$$



Constraint c associated with a loop L

$$c := \left(\min_{\forall x \in L}(\Delta(x)), \max_{\forall x \in L}(\Delta(x))\right)$$

$$c_1 = \left(\binom{5}{3}, \binom{7}{5}\right)$$

Modeling transient loops as constraints

Destination b:

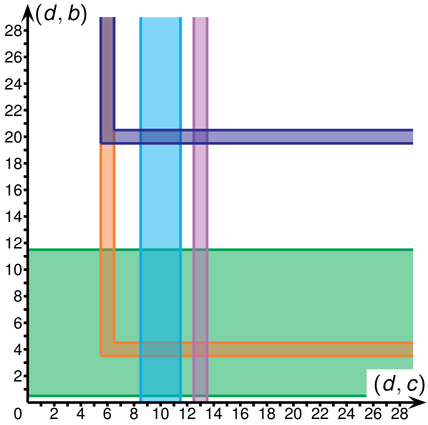
$$c_2 = \left(\begin{pmatrix} 5 \\ 19 \end{pmatrix}, \begin{pmatrix} 7 \\ 21 \end{pmatrix} \right)$$

$$c_3 = \left(\begin{pmatrix} -14 \\ 0 \end{pmatrix}, \begin{pmatrix} -2 \\ 12 \end{pmatrix} \right)$$

Destination c:

$$c_4 = \left(\begin{pmatrix} 8 \\ -6 \end{pmatrix}, \begin{pmatrix} 12 \\ -2 \end{pmatrix} \right)$$

$$c_5 = \left(\begin{pmatrix} 12 \\ -2 \end{pmatrix}, \begin{pmatrix} 14 \\ 0 \end{pmatrix} \right)$$



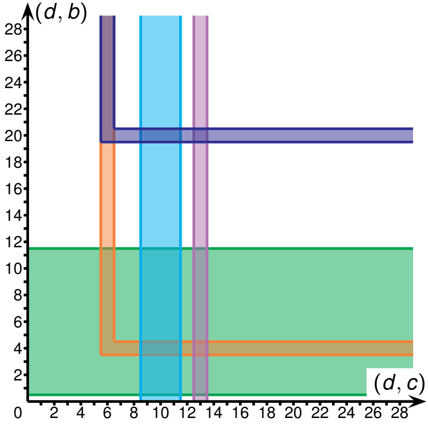
Theorem

A monotonic sequence S prevents a loop L if and only if S contains a vector that meets the associated constraint c.

Computing weight update sequences

Greedy Backward Algorithm (GBA)

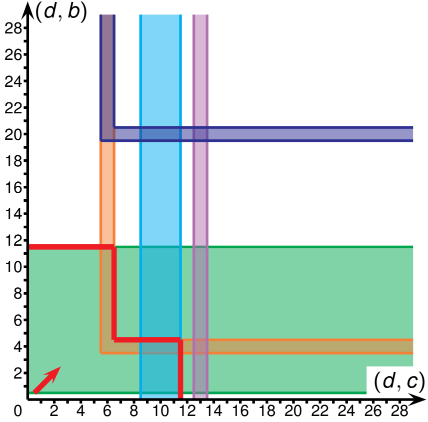
At each step, retrieve the maximum value on each index among the lower bounds of the remaining constraints.



Computing weight update sequences

Greedy Backward Algorithm (GBA)

At each step, retrieve the maximum value on each index among the lower bounds of the remaining constraints.

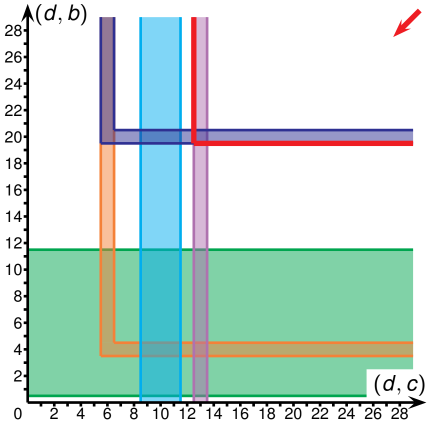


Computing weight update sequences

Greedy Backward Algorithm (GBA)

At each step, retrieve the maximum value on each index among the lower bounds of the remaining constraints.

$$S_{GBA} = \left\{ \qquad \qquad \right\}$$



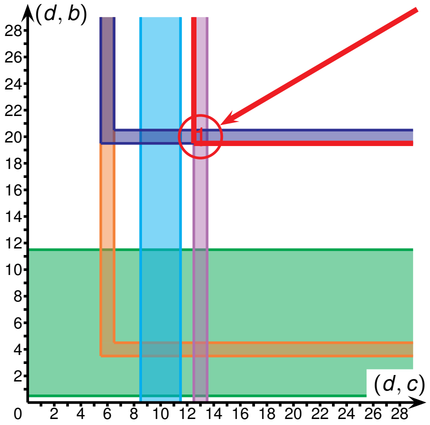
Computing weight update sequences

Greedy Backward Algorithm (GBA)

At each step, retrieve the maximum value on each index among the lower bounds of the remaining constraints.

$$S_{GBA} = \left\{ \quad \begin{pmatrix} 13 \\ 20 \end{pmatrix} \right\}$$

c_2
c_5



Computing weight update sequences

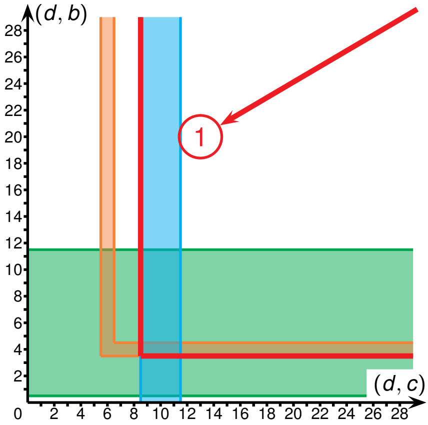
Greedy Backward Algorithm (GBA)

At each step, retrieve the maximum value on each index among the lower bounds of the remaining constraints.

$$S_{GBA} = \left\{ \quad \begin{pmatrix} 13 \\ 20 \end{pmatrix} \right\}$$

c_2

c_5



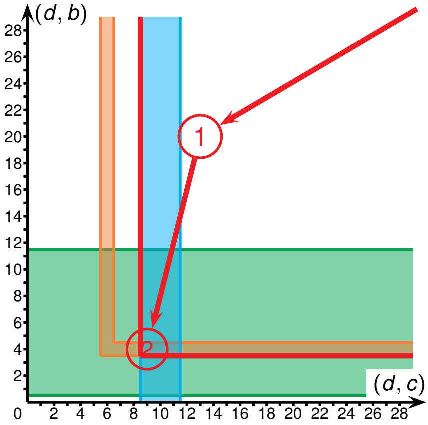
Computing weight update sequences

Greedy Backward Algorithm (GBA)

At each step, retrieve the maximum value on each index among the lower bounds of the remaining constraints.

$$S_{GBA} = \left\{ \begin{pmatrix} 9 \\ 4 \end{pmatrix}, \begin{pmatrix} 13 \\ 20 \end{pmatrix} \right\}$$

c_1 c_2
c_3 c_5
c_4



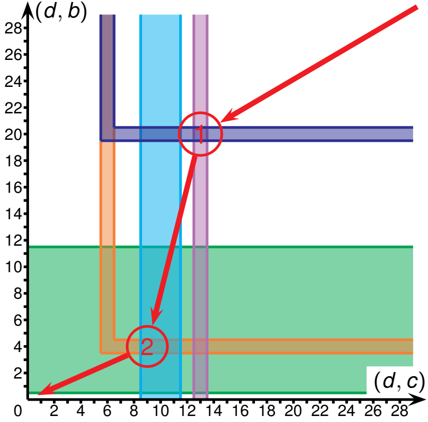
Computing weight update sequences

Greedy Backward Algorithm (GBA)

At each step, retrieve the maximum value on each index among the lower bounds of the remaining constraints.

$$S_{GBA} = \left\{ \begin{pmatrix} 9 \\ 4 \end{pmatrix}, \begin{pmatrix} 13 \\ 20 \end{pmatrix} \right\}$$

c_1 c_2
c_3 c_5
c_4

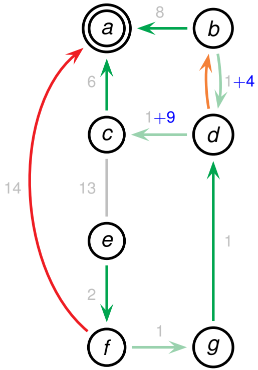


Theorem

Given a set of loop-constraints, *GBA* computes a minimal sequence of weight updates preventing all associated convergence loops.

Outline

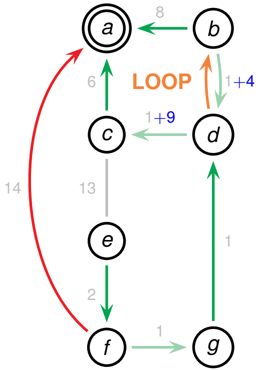
Intermediate forwarding changes



$$S_{GBA} = \left\{ \begin{pmatrix} 9 \\ 4 \end{pmatrix}, \begin{pmatrix} 13 \\ 20 \end{pmatrix} \right\}$$

- Triggered by non-uniform weight updates
- Routing instabilities
  - Increased probability of out-of-order delivery (disruptive for TCP)

Intermediate forwarding changes



$$S_{GBA} = \left\{ \begin{pmatrix} 9 \\ 4 \end{pmatrix}, \begin{pmatrix} 13 \\ 20 \end{pmatrix} \right\}$$

- Triggered by non-uniform weight updates
- Routing instabilities
  - Increased probability of out-of-order delivery (disruptive for TCP)
  - Additional transient loops (local to the modified router)

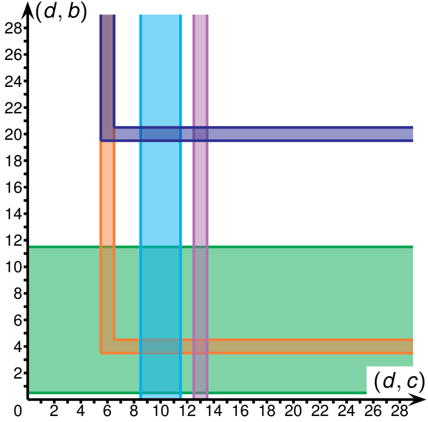
Local stability conditions

Local stability conditions for d

Let r be the modified router and $s \in Succ_d(r)$ an initial successor of r for d:

$$\begin{cases} v[x] = v[s^*] & \text{if } x \in Succ_d(r) \\ v[x] > v[s^*] - M(r, x, d)^{10} & \text{otherwise} \end{cases}$$

	$v[b]$	$v[c]$
Dest. a	$> v[c] - 2$	–
Dest. b	–	$> v[b] - 14$
Dest. c	$> v[c] - 14$	–



[10] $M(r, x, d) = C(r, x, d) - C(r, d)$

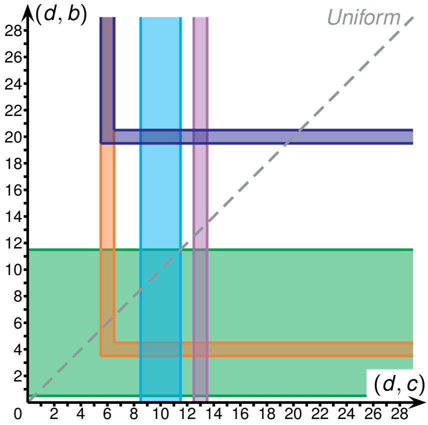
Local stability conditions

Local stability conditions for d

Let r be the modified router and $s \in Succ_d(r)$ an initial successor of r for d:

$$\begin{cases} v[x] = v[s^*] & \text{if } x \in Succ_d(r) \\ v[x] > v[s^*] - M(r,x,d)^{10} & \text{otherwise} \end{cases}$$

	$v[b]$	$v[c]$
Dest. a	$> v[c] - 2$	–
Dest. b	–	$> v[b] - 14$
Dest. c	$> v[c] - 14$	–



[10] $M(r,x,d) = C(r,x,d) - C(r,d)$

Local stability conditions

Local stability conditions for d

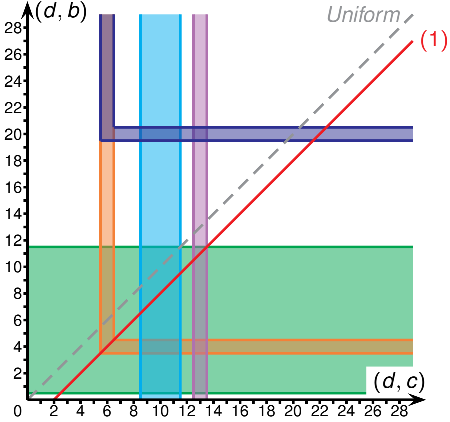
Let r be the modified router and $s \in Succ_d(r)$ an initial successor of r for d:

$$\begin{cases} v[x] = v[s^*] & \text{if } x \in Succ_d(r) \\ v[x] > v[s^*] - M(r,x,d)^{10} & \text{otherwise} \end{cases}$$

	$v[b]$	$v[c]$
Dest. a	$> v[c] - 2$	–
Dest. b	–	$> v[b] - 14$
Dest. c	$> v[c] - 14$	–

New constraints:

(1) $v[b] > v[c] - 2$



[10] $M(r,x,d) = C(r,x,d) - C(r,d)$

Local stability conditions

Local stability conditions for d

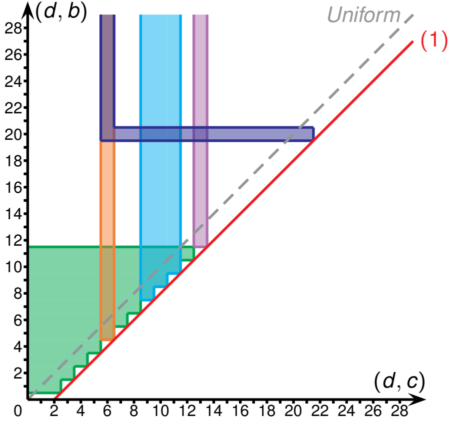
Let r be the modified router and $s \in Succ_d(r)$ an initial successor of r for d:

$$\begin{cases} v[x] = v[s^*] & \text{if } x \in Succ_d(r) \\ v[x] > v[s^*] - M(r,x,d)^{10} & \text{otherwise} \end{cases}$$

	$v[b]$	$v[c]$
Dest. a	$> v[c] - 2$	–
Dest. b	–	$> v[b] - 14$
Dest. c	$> v[c] - 14$	–

New constraints:

(1) $v[b] > v[c] - 2$



[10] $M(r,x,d) = C(r,x,d) - C(r,d)$

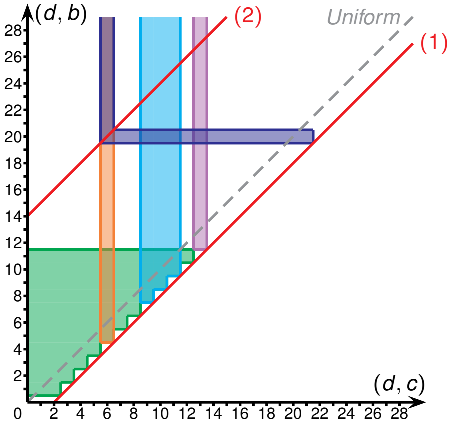
Local stability conditions

Local stability conditions for d

Let r be the modified router and $s \in Succ_d(r)$ an initial successor of r for d:

$$\begin{cases} v[x] = v[s^*] & \text{if } x \in Succ_d(r) \\ v[x] > v[s^*] - M(r,x,d)^{10} & \text{otherwise} \end{cases}$$

	$v[b]$	$v[c]$
Dest. a	$> v[c] - 2$	–
Dest. b	–	$> v[b] - 14$
Dest. c	$> v[c] - 14$	–



New constraints:

(1) $v[b] > v[c] - 2$

(2) $v[c] < v[b] - 14 \Rightarrow v[b] < v[c] + 14$

[10] $M(r,x,d) = C(r,x,d) - C(r,d)$

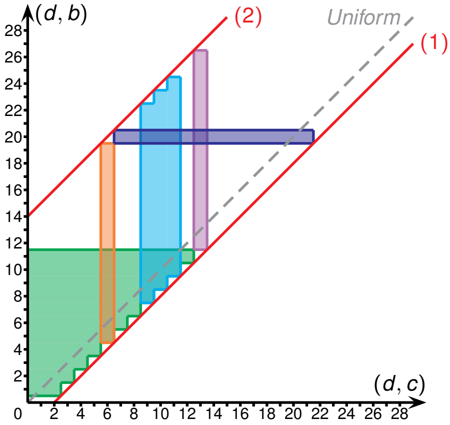
Local stability conditions

Local stability conditions for d

Let r be the modified router and $s \in Succ_d(r)$ an initial successor of r for d:

$$\begin{cases} v[x] = v[s^*] & \text{if } x \in Succ_d(r) \\ v[x] > v[s^*] - M(r,x,d)^{10} & \text{otherwise} \end{cases}$$

	$v[b]$	$v[c]$
Dest. a	$> v[c] - 2$	–
Dest. b	–	$> v[b] - 14$
Dest. c	$> v[c] - 14$	–



New constraints:

(1) $v[b] > v[c] - 2$

(2) $v[c] < v[b] - 14 \Rightarrow v[b] < v[c] + 14$

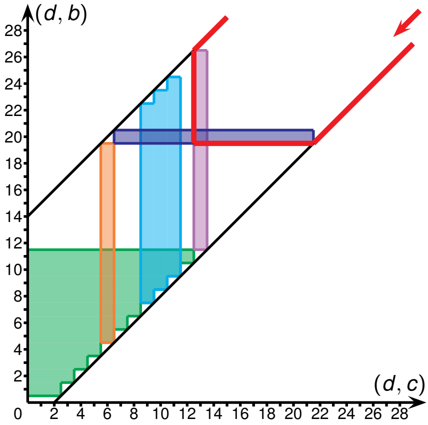
10: $M(r,x,d) = C(r,x,d) - C(r,d)$

Computing a sequence with CPCs

Adjusted GBA

Retrieve the maximum value on each index among the lower bounds of the remaining constraints, that meets all the CPCs.

$S_{AGBA} = \{ \qquad \}$



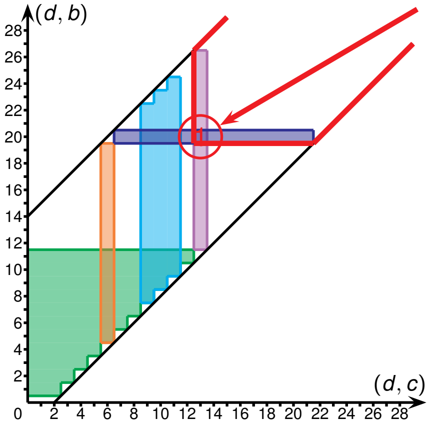
Computing a sequence with CPCs

Adjusted GBA

Retrieve the maximum value on each index among the lower bounds of the remaining constraints, that meets all the CPCs.

$$S_{AGBA} = \left\{ \begin{pmatrix} 13 \\ 20 \end{pmatrix} \right\}$$

c_2
c_5



Computing a sequence with CPCs

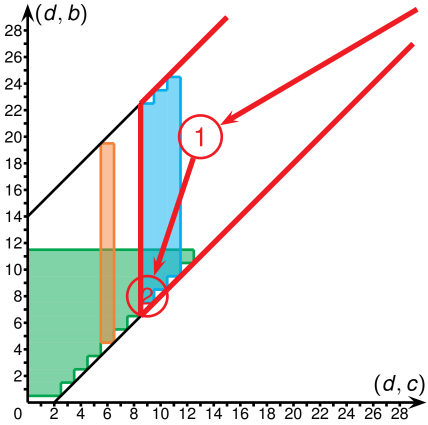
Adjusted GBA

Retrieve the maximum value on each index among the lower bounds of the remaining constraints, that meets all the CPCs.

$$S_{AGBA} = \left\{ \binom{9}{8}, \binom{13}{20} \right\}$$

c_3 c_2

c_4 c_5



Computing a sequence with CPCs

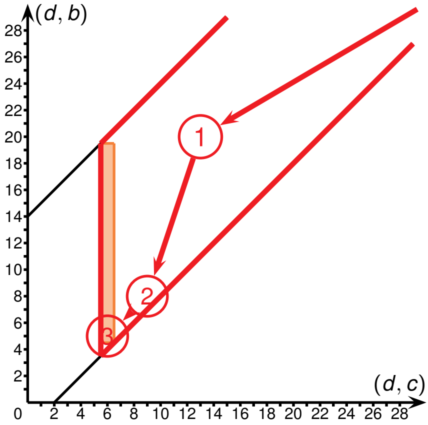
Adjusted GBA

Retrieve the maximum value on each index among the lower bounds of the remaining constraints, that meets all the CPCs.

$$S_{AGBA} = \left\{ \begin{pmatrix} 6 \\ 5 \end{pmatrix}, \begin{pmatrix} 9 \\ 8 \end{pmatrix}, \begin{pmatrix} 13 \\ 20 \end{pmatrix} \right\}$$

c_1 c_3 c_2

c_4 c_5



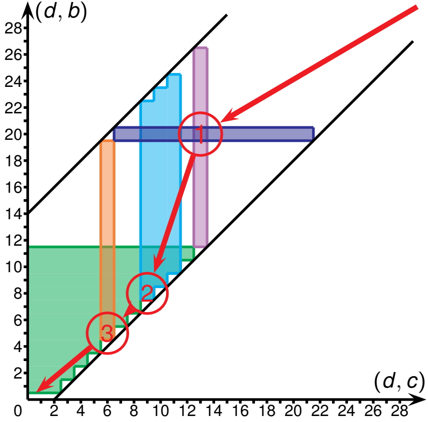
Computing a sequence with CPCs

Adjusted GBA

Retrieve the maximum value on each index among the lower bounds of the remaining constraints, that meets all the CPCs.

$$S_{AGBA} = \left\{ \begin{pmatrix} 6 \\ 5 \end{pmatrix}, \begin{pmatrix} 9 \\ 8 \end{pmatrix}, \begin{pmatrix} 13 \\ 20 \end{pmatrix} \right\}$$

c_1 c_3 c_4 c_2 c_5



$$S_{GBA} = \left\{ \begin{pmatrix} 9 \\ 4 \end{pmatrix}, \begin{pmatrix} 13 \\ 20 \end{pmatrix} \right\}$$

Transient loop prevention alternatives

- Dynamic Greedy Backward Heuristic (DGBH)
  - Sequence length not minimal
  - Very short sequences in practice
- Combination of GBA and *local-delay*
  - Minimal sequence length (GBA)
  - Requires support of local-delay on the modified router

	Intermediate disruptions avoidance		Sequence minimality
	Transient loops	Forwarding changes	
Uniform	✓	✓	✗
AGBA	✓	✓	✓
DGBH	✓	✗	✗
GBA w/ local delay	✓	✗	✓

Outline

1. Introduction
2. Context
3. Contributions

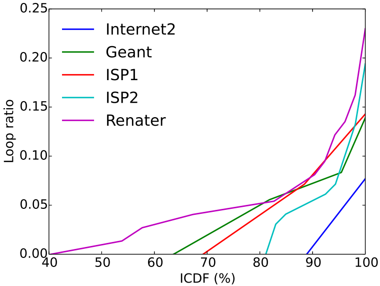
Evaluation setup and criteria

Network	$\|N\|$	$\|E\|$	$\varnothing$	Max. degree	Weight space
Internet2	9	26	4	4	[277, 1705] (13)
GEANT	22	72	4	6	[1, 20050] (18)
RENATER	70	230	11	13	[1, 1000] (14)
ISP 1	25	55	6	6	[1, 11] (4)
ISP 2	55	200	5	20	[10, 50000] (8)
ISP 3	110	350	11	8	[1, 9999] (32)
ISP 4	150	400	13	9	[1, 9999] (32)
ISP 5	200	800	13	14	[1, 66666] (55)
ISP 6	1200	4000	12	56	[1, 100010] (105)

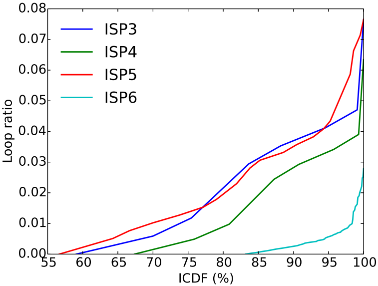
Evaluation criteria:

- Update sequence lengths
- Computing time efficiency

Impact of router removal operations



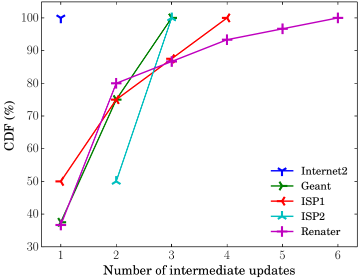
Affected links ratio on small ISPs



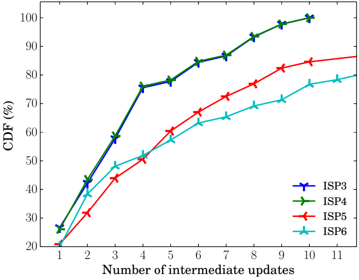
Affected links ratio on large ISPs

- Loop potentialities depend on the *shape* of the network
- Removing a single router may affect more than 20% of the links

Sequence lengths produced by GBA



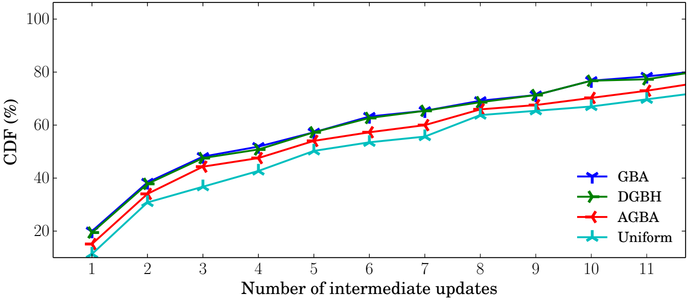
Node shutdown operations on small ISPs



Node shutdown operations on large ISPs

- ▷ Very short sequences for *small networks*
- ▷ Reasonable length for most sequences even in *large networks*

Sequence length distribution of GBA alternatives



Sequence length distribution on ISP6

- Sequences of same length for GBA and DGBH in most cases
- AGBA sequences significantly shorter than uniform ones, for the same routing stability

Computing times

Network	Min	Max	Mean	3^{rd} quartile	9^{th} decile
Internet2	0.06 ms	0.06 ms	0.06 ms	0.06 ms	0.06 ms
Geant	0.21 ms	0.35 ms	0.28 ms	0.30 ms	0.33 ms
ISP1	0.34 ms	0.51 ms	0.41 ms	0.47 ms	0.51 ms
ISP2	1.43 ms	2.68 ms	1.96 ms	2.08 ms	2.67 ms
Renater	0.35 ms	2.68 ms	1.28 ms	1.48 ms	1.78 ms
ISP3	0.49 ms	10.91 ms	6.08 ms	7.25 ms	7.75 ms
ISP4	0.99 ms	18.04 ms	10.18 ms	12.07 ms	12.95 ms
ISP5	0.64 ms	49.63 ms	23.80 ms	30.01 ms	34.64 ms
ISP6	3.63 ms	**2.15 s**	**1.40 s**	**1.70 s**	**1.77 s**

- ▷ Negligible computing times ($<$ 50 ms) even for large ISPs[11]
- ▷ Still reasonable for very large networks[12]

[11] ISP5: 200 nodes / 800 edges
[12] ISP6: 1200 nodes / 8000 edges

Outline

Conclusion

- ✓ Transient loops impact evaluation
  - ▷ Loops do occur and impact the traffic in ISP networks
- ✓ Improvement of the existing approach
  - ▷ Sequence minimality with polynomial time algorithms
  - ▷ Efficient implementation
- ✓ Generalization to node-wide operations
  - ▷ Practical solutions to deal with routing instabilities

Perspectives

- Evaluate the approach on a production network
  (delay between updates, impact on inter-domain routing, . . .)
- Devise methods to deal with longest sequences
  (skip some elements, ignore destinations, . . .)
- Assess the impact of intermediate changes on the traffic

- Reduce sequence lengths by allowing negative weight updates
- Investigate complexity of the intermediate transient loop problem

- Extend the approach to different contexts
  (multicast, wireless communications,. . .)

Thank you for your attention.