What Have We Learned from Reverse-Engineering the Internet's Inter-domain Routing Protocol?

Timothy G. Griffin

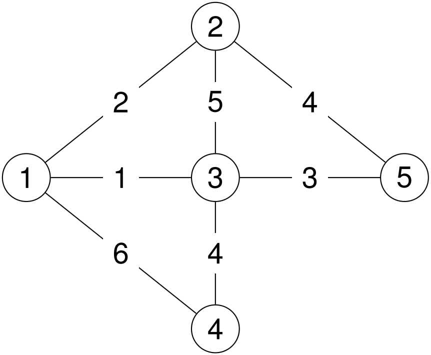
Computer Laboratory
University of Cambridge, UK
timothy.griffin@cl.cam.ac.uk

14èmes Rencontres Francophones sur
les Aspects Algorithmiques des Télécommunications
Hérault, France
29 May, 2012

Background

- Internet routing has evolved organically, by the expedient hack....
- ... basic principles need to be uncovered by reverse engineering.
- In the process, a new type of path problem is discovered!
- This may have widespread applicability beyond routing — perhaps in operations research, combinatorics, and other branches of Computer Science.

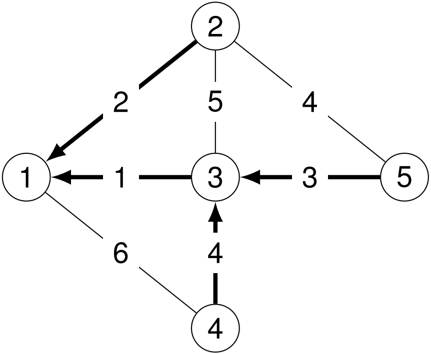
Shortest paths example, $\mathrm{sp} = (\mathbb{N}^{\infty},\ \text{min},\ +)$



The adjacency matrix

$$\mathbf{A} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \left[\begin{array}{ccccc} \infty & 2 & 1 & 6 & \infty \\ 2 & \infty & 5 & \infty & 4 \\ 1 & 5 & \infty & 4 & 3 \\ 6 & \infty & 4 & \infty & \infty \\ \infty & 4 & 3 & \infty & \infty \end{array}\right] \end{array}$$

Shortest paths example, continued



Bold arrows indicate the shortest-path tree rooted at 1.

The routing matrix

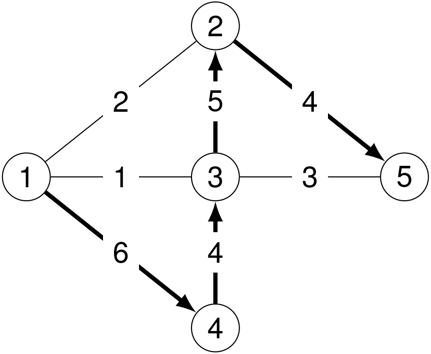
$$\mathbf{A}^* = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 2 & 1 & 5 & 4 \\ 2 & 0 & 3 & 7 & 4 \\ 1 & 3 & 0 & 4 & 3 \\ 5 & 7 & 4 & 0 & 7 \\ 4 & 4 & 3 & 7 & 0 \end{bmatrix} \end{array}$$

Matrix $\mathbf{A}^*$ solves this global optimality problem:

$$\mathbf{A}^*(i,\ j) = \min_{p \in P(i,\ j)} w(p),$$

where $P(i,\ j)$ is the set of all paths from i to j.

Widest paths example, $(\mathbb{N}^{\infty},\ \max,\ \min)$



Bold arrows indicate the widest-path tree rooted at 1.

The routing matrix

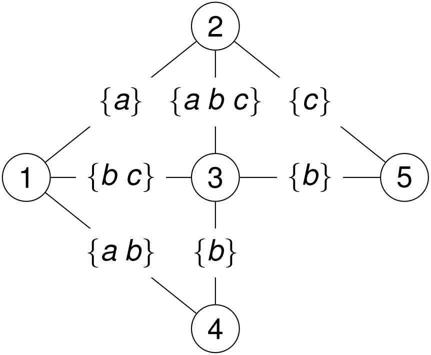
$$
\mathbf{A}^* = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
\infty & 4 & 4 & 6 & 4 \\
4 & \infty & 5 & 4 & 4 \\
4 & 5 & \infty & 4 & 4 \\
6 & 4 & 4 & \infty & 4 \\
4 & 4 & 4 & 4 & \infty
\end{bmatrix}
\end{array}
$$

Matrix $\mathbf{A}^*$ solves this global optimality problem:

$$\mathbf{A}^*(i,\ j) = \max_{p \in P(i,\ j)} w(p),$$

where $w(p)$ is now the minimal edge weight in p.

Fun example, $(2^{\{a,\ b,\ c\}},\ \cup,\ \cap)$



We want a Matrix $\mathbf{A}^*$ to solve this global optimality problem:

$$\mathbf{A}^*(i,\ j) = \bigcup_{p \in P(i,\ j)} w(p),$$

where $w(p)$ is now the intersection of all edge weights in p.

For $x \in \{a,\ b,\ c\}$, interpret $x \in \mathbf{A}^*(i,\ j)$ to mean that there is at least one path from i to j with x in every arc weight along the path.

Fun example, $(2^{\{a,\ b,\ c\}},\ \cup,\ \cap)$

The matrix $\mathbf{A}^*$

$$
\begin{array}{c c}
 & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} &
\left[\begin{array}{ccccc}
\{a\,b\,c\} & \{a\,b\,c\} & \{a\,b\,c\} & \{a\,b\} & \{b\,c\} \\
\{a\,b\,c\} & \{a\,b\,c\} & \{a\,b\,c\} & \{a\,b\} & \{b\,c\} \\
\{a\,b\,c\} & \{a\,b\,c\} & \{a\,b\,c\} & \{a\,b\} & \{b\,c\} \\
\{a\,b\} & \{a\,b\} & \{a\,b\} & \{a\,b\,c\} & \{b\} \\
\{b\,c\} & \{b\,c\} & \{b\,c\} & \{b\} & \{a\,b\,c\}
\end{array} \right]
\end{array}
$$

Semirings

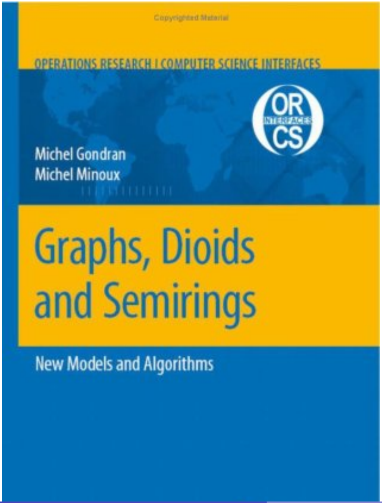
A few examples

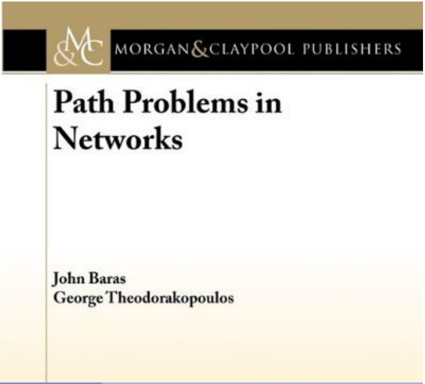
name	S	$\oplus,$	$\otimes$	$\overline{0}$	$\overline{1}$	possible routing use
sp	$\mathbb{N}^{\infty}$	min	$+$	∞	0	minimum-weight routing
bw	$\mathbb{N}^{\infty}$	max	min	0	∞	greatest-capacity routing
rel	$[0, 1]$	max	$\times$	0	1	most-reliable routing
use	$\{0, 1\}$	max	min	0	1	usable-path routing
	2^W	$\cup$	$\cap$	$\{\}$	W	shared link attributes?
	2^W	$\cap$	$\cup$	W	$\{\}$	shared path attributes?

Path problems focus on global optimality

$$\mathbf{A}^*(i, j) = \bigoplus_{p \in P(i, j)} w(p)$$

Recommended Reading





What algebraic properties are needed for efficient computation of global optimality?

Distributivity

$$\begin{array}{rcrcl} \text{L.D} & : & a \otimes (b \oplus c) & = & (a \otimes b) \oplus (a \otimes c), \\ \text{R.D} & : & (a \oplus b) \otimes c & = & (a \otimes c) \oplus (b \otimes c). \end{array}$$

What is this in $\text{sp} = (\mathbb{N}^{\infty},\ \min,\ +)$?

$$\begin{array}{rcrcl} \text{L.DIST} & : & a + (b \min c) & = & (a + b) \min (a + c), \\ \text{R.DIST} & : & (a \min b) + c & = & (a + c) \min (b + c). \end{array}$$

(I am ignoring all of the other semiring axioms here ...)

Lesson 1: Some realistic metrics are not distributive!

Two ways of forming “lexicographic” combination of shortest paths sp and bandwidth bw.

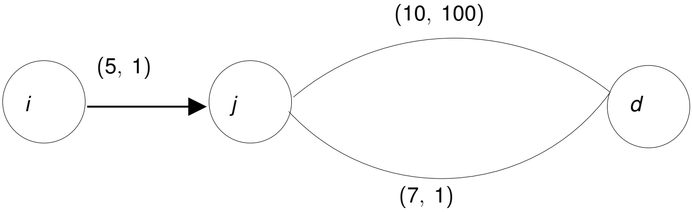
Widest shortest paths

- metric values of form (d, b)
- d in sp
- b in bw
- consider d first, break ties with b
- is distributive (some details ignored ...)

Shortest Widest paths

- metric values of form (b, d)
- d in sp
- b in bw
- consider b first, break ties with d
- not distributive

Example



- node j prefers $(10, 100)$ over $(7, 1)$.
- node i prefers $(5, 2)$ over $(5, 101)$.

$$(5, 1) \otimes ((10, 100) \oplus (7, 1)) = (5, 1) \otimes (10, 100) = (5, 101)$$

$$((5, 1) \otimes (10, 101)) \oplus ((5, 1) \otimes (7, 1)) = (5, 101) \oplus (5, 2) = (5, 2)$$

Lesson 2: Left-Local Optimality

Say that **L** is a left locally-optimal solution when

$$\mathbf{L} = (\mathbf{A} \otimes \mathbf{L}) \oplus \mathbf{I}.$$

That is, for $i \neq j$ we have

$$\mathbf{L}(i,\ j) = \bigoplus_{q \in V} \mathbf{A}(i,\ q) \otimes \mathbf{L}(q,\ j)$$

- $\mathbf{L}(i,\ j)$ is the best possible value given the values $\mathbf{L}(q,\ j)$, for all out-neighbors q of source i.
- Rows $\mathbf{L}(i,\ \_)$ represents **out-trees <u>from</u>** i (think Bellman-Ford).
- Columns $\mathbf{L}(\_,\ i)$ represents **in-trees <u>to</u>** i.
- Works well with hop-by-hop forwarding from i.

Right-Local Optimality

Say that **R** is a right locally-optimal solution when

$$\mathbf{R} = (\mathbf{R} \otimes \mathbf{A}) \oplus \mathbf{I}.$$

That is, for $i \neq j$ we have

$$\mathbf{R}(i,\ j) = \bigoplus_{q \in V} \mathbf{R}(i,\ q) \otimes \mathbf{A}(q,\ j)$$

- $\mathbf{R}(i,\ j)$ is the best possible value given the values $\mathbf{R}(q,\ j)$, for all in-neighbors q of destination j.
- Rows $\mathbf{L}(i,\ \_)$ represents **out-trees <u>from</u>** i (think Dijkstra).
- Columns $\mathbf{L}(\_,\ i)$ represents **in-trees <u>to</u>** i.
- Does not work well with hop-by-hop forwarding from i.

With and Without Distributivity

With

For semirings, the three optimality problems are essentially the same — locally optimal solutions are globally optimal solutions.

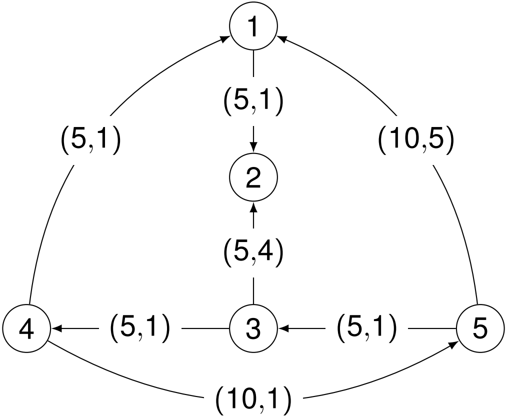
$$\mathbf{A}^* = \mathbf{L} = \mathbf{R}$$

Without

Suppose that we drop distributivity and $\mathbf{A}^*$, $\mathbf{L}$, $\mathbf{R}$ exist. It may be the case they they are all distinct.

Health warning : matrix multiplication over structures lacking distributivity is not associative!

Example



(bandwidth, distance) with lexicographic order (bandwidth first).

Global optima

$$\mathbf{A}^* = \begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} (\infty, 0) & (5, 1) & (0, \infty) & (0, \infty) & (0, \infty) \\ (0, \infty) & (\infty, 0) & (0, \infty) & (0, \infty) & (0, \infty) \\ (5, 2) & (5, 3) & (\infty,\ 0) & (5, 1) & (5, 2) \\ (10, 6) & (5, 2) & (5, 2) & (\infty, 0) & (10, 1) \\ (10, 5) & (5, 4) & (5, 1) & (5, 2) & (\infty, 0) \end{bmatrix} \end{array},$$

Left local optima

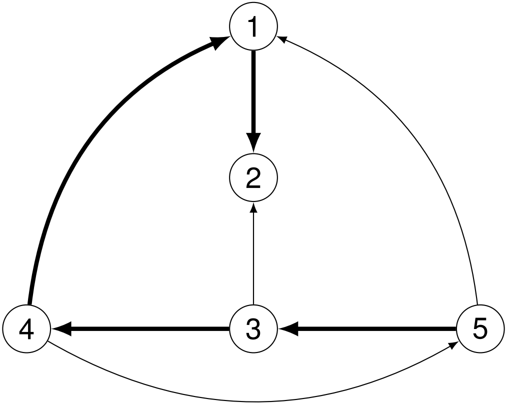
$$\mathbf{L} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \left[\begin{array}{ccccc} (\infty, 0) & (5, 1) & (0, \infty) & (0, \infty) & (0, \infty) \\ (0, \infty) & (\infty, 0) & (0, \infty) & (0, \infty) & (0, \infty) \\ (\mathbf{5}, \mathbf{7}) & (5, 3) & (\infty, 0) & (5, 1) & (5, 2) \\ (10, 6) & (5, 2) & (5, 2) & (\infty, 0) & (10, 1) \\ (10, 5) & (5, 4) & (5, 1) & (5, 2) & (\infty, 0) \end{array} \right] \end{array},$$

Entries marked in **bold** indicate those values which are not globally optimal.

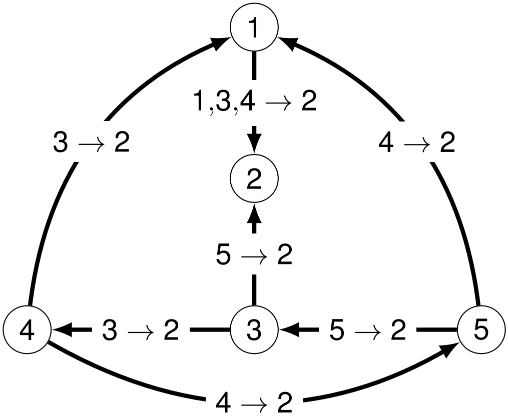
Right local optima

$$\mathbf{R} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \left[\begin{array}{ccccc} (\infty, 0) & (5, 1) & (0, \infty) & (0, \infty) & (0, \infty) \\ (0, \infty) & (\infty, 0) & (0, \infty) & (0, \infty) & (0, \infty) \\ (5, 2) & (5, 3) & (\infty,\ 0) & (5,\ 1) & (5, 2) \\ (10, 6) & (\mathbf{5}, \mathbf{6}) & (5, 2) & (\infty, 0) & (10, 1) \\ (10, 5) & (\mathbf{5}, \mathbf{5}) & (5, 1) & (5, 2) & (\infty, 0) \end{array} \right] \end{array},$$

Left-locally optimal paths to node 2



Right-locally optimal paths to node 2



Lesson 3: Bellman-Ford can compute left-local solutions

$$\begin{aligned} \mathbf{A}^{[0]} &= \mathbf{I} \\ \mathbf{A}^{[k+1]} &= (\mathbf{A} \otimes \mathbf{A}^{k}) \oplus \mathbf{I}, \end{aligned}$$

- Bellman-ford algorithm must be modified to ensure only loop-free paths are inspected.
- $(S, \oplus, \overline{0})$ is a commutative, idempotent, and selective monoid,
- $(S, \otimes, \overline{1})$ is a monoid,
- $\overline{0}$ is the annihilator for $\otimes$,
- $\overline{1}$ is the annihilator for $\oplus$,
- Left strictly inflationarity, L.S.INF : $\forall a, b : a \neq \overline{0} \implies a < a \otimes b$
- Here $a \leq b \equiv a = a \oplus b$.

Convergence to a unique left-local solution is guaranteed. Currently no polynomial bound is known on the number of iterations required.

Lesson 4 : Dijkstra's algorithm can work for right-local optima!

Input	:	adjacency matrix $\mathbf{A}$ and source vertex $i \in V$,
Output	:	the i-th row of $\mathbf{R}$, $\mathbf{R}(i, \_)$.

begin
  $S \leftarrow \{i\}$
  $\mathbf{R}(i, i) \leftarrow \overline{1}$
  for each $q \in V - \{i\}$: $\mathbf{R}(i, q) \leftarrow \mathbf{A}(i, q)$
  while $S \neq V$
    begin
      find $q \in V - S$ such that $\mathbf{R}(i, q)$ is $\leq^{L}_{\oplus}$ -minimal
      $S \leftarrow S \cup \{q\}$
      for each $j \in V - S$
        $\mathbf{R}(i, j) \leftarrow \mathbf{R}(i, j) \oplus (\mathbf{R}(i, q) \otimes \mathbf{A}(q, j))$
    end
end

The goal

Given adjacency matrix **A** and source vertex $i \in V$, Dijkstra's algorithm will compute $\mathbf{R}(i, \_)$ such that

$$\forall j \in V : \mathbf{R}(i, j) = \mathbf{I}(i, j) \oplus \bigoplus_{q \in V} \mathbf{R}(i, q) \otimes \mathbf{A}(q, j).$$

Main invariant

$$\forall k : 1 \leq k \leq \mid V \mid \implies \forall j \in S_k : \mathbf{R}_k(i, j) = \mathbf{I}(i, j) \oplus \bigoplus_{q \in S_k} \mathbf{R}_k(i, q) \otimes \mathbf{A}(q, j)$$

Routing in Equilibrium. João Luís Sobrinho and Timothy G. Griffin. The 19th International Symposium on Mathematical Theory of Networks and Systems (MTNS 2010).

Minimal subset of semiring axioms needed right-local Dijkstra

~~Semiring~~ Axioms

$$
\begin{array}{rclcl}
\text{ADD.ASSOCIATIVE} & : & a \oplus (b \oplus c) & = & (a \oplus b) \oplus c \\
\text{ADD.COMMUTATIVE} & : & a \oplus b & = & b \oplus a \\
\text{ADD.LEFT.ID} & : & \overline{0} \oplus a & = & a \\
\cancel{\text{MULT.ASSOCIATIVE}} & : & \cancel{a \otimes (b \otimes c)} & \cancel{=} & \cancel{(a \otimes b) \otimes c} \\
\text{MULT.LEFT.ID} & : & \overline{1} \otimes a & = & a \\
\cancel{\text{MULT.RIGHT.ID}} & : & \cancel{a \otimes \overline{1}} & \cancel{=} & \cancel{a} \\
\cancel{\text{MULT.LEFT.ANN}} & : & \cancel{\overline{0} \otimes a} & \cancel{=} & \cancel{\overline{0}} \\
\cancel{\text{MULT.RIGHT.ANN}} & : & \cancel{a \otimes \overline{0}} & \cancel{=} & \cancel{\overline{0}} \\
\cancel{\text{L.DISTRIBUTIVE}} & : & \cancel{a \otimes (b \oplus c)} & \cancel{=} & \cancel{(a \otimes b) \oplus (a \otimes c)} \\
\cancel{\text{R.DISTRIBUTIVE}} & : & \cancel{(a \oplus b) \otimes c} & \cancel{=} & \cancel{(a \otimes c) \oplus (b \otimes c)}
\end{array}
$$

Additional axioms needed right-local Dijkstra

$$
\begin{array}{rcrcl}
\text{ADD.SELECTIVE} & : & a \oplus b & \in & \{a,\ b\} \\
\text{ADD.LEFT.ANN} & : & \overline{1} \oplus a & = & \overline{1} \\
\text{ADD.RIGHT.ANN} & : & a \oplus \overline{1} & = & \overline{1} \\
\text{RIGHT.ABSORBTION} & : & a \oplus (a \otimes b) & = & a
\end{array}
$$

RIGHT.ABSORBTION gives inflationarity, $\forall a, b : a \leq a \otimes b$.

Expressed in Coq

```
Variable plus_associative     : ∀x y z, x ⊕ (y ⊕ z) = (x ⊕ y) ⊕ z.
Variable plus_commutative     : ∀x y, x ⊕ y = y ⊕ x.
Variable plus_selective       : ∀x y, (x ⊕ y == x) || (x ⊕ y == y).

(* identities                                                    *)
Variable zero_is_left_plus_id  : ∀x, zero ⊕ x = x.
Variable one_is_left_times_id  : ∀x, one ⊗ x = x.

(* one is additive annihilator                                    *)
Variable one_is_left_plus_ann  : ∀x, one ⊕ x = one.
Variable one_is_right_plus_ann : ∀x, x ⊕ one = one.

(* right absorbtion                                               *)
Variable right_absorption      : ∀ a b : T, a ⊕ (a ⊗ b) == a.

Definition lno (a b : T) := a ⊕ b == a.
Notation "A ≦ B" := (lno A B) (at level 60).

Lemma lno_right_increasing : ∀ a b : T,  a ≦ a ⊗ b.
```

Using a Link-State approach with hop-by-hop forwarding ...

Need left-local optima!

$$\mathbf{L} = (\mathbf{A} \otimes \mathbf{L}) \oplus \mathbf{I} \quad \Longleftrightarrow \quad \mathbf{L}^T = (\mathbf{L}^T \hat{\otimes}^T \mathbf{A}^T) \oplus \mathbf{I}$$

where $\otimes^T$ is matrix multiplication defined with as

$$a \otimes^T b = b \otimes a$$

and we assume left-inflationarity holds, $\textsc{L.INF} : \forall a, b : a \leq b \otimes a$.

Each node would have to solve the entire "all pairs" problem.

Inter-domain routing in the Internet

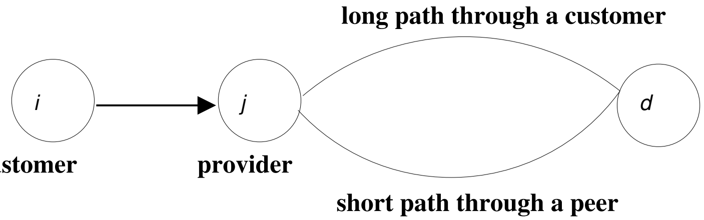
The Border Gateway Protocol (BGP)

- In the distributed Bellman-Ford family.
- Hard-state (not refresh based).
- Complex policy and metrics.
- Primary requirement: connectivity should not violate the economic relationships between autonomous networks.
- At a very high-level, the metric combines economics and traffic engineering.
- This is implemented using a lexicographic product, where economics is most significant.

Simplified model (Gao and Rexford)

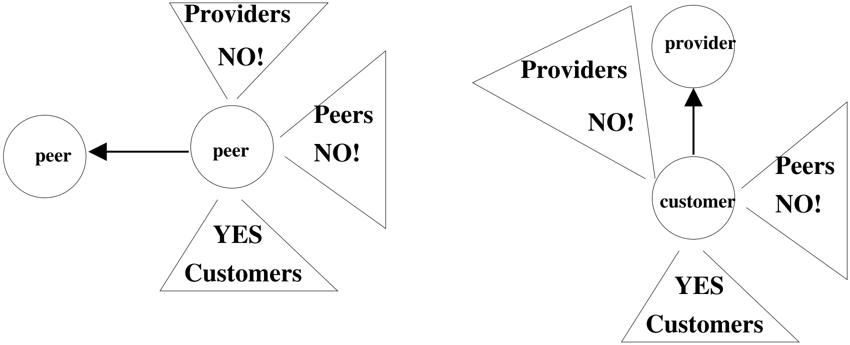
- **customer route** : from somebody paying you for transit services.
- **provider route** : from somebody you are paying for transit services.
- **peer route** : from a competitor.
  - If you are at top of food chain you are forced to do this.
  - Smaller networks do this to reduce their provider charges.
- **customer** $<$ **peer** $<$ **provider**

Example



- node *j* prefers long path though one of its customers
- node *i* prefers the shorter path through its provider

Route visibility restriction



These restrictions are another source for violations of distributivity.

BGP policies are not constrained ...

As a result ...

- Protocol will diverge when no solution exists.
- Protocol may diverge even when a solution exists.
- BGP Wedgies, RFC 4264.
  - Multiple stable states may exist.
  - No guarantee that each state implements intended policy.
  - Manual intervention required when system gets stuck in unintended local optima.
  - Debugging nearly impossible when policy is not shared between networks.

How to fix?

First, allow functions on arcs.

$$(S,\ \oplus,\ F \subseteq S \to S,\ \overline{0})$$

General conditions

- $(S,\ \oplus,\ \overline{0})$ is a commutative, idempotent, and selective monoid,
- $\forall f \in F : f(\overline{0}) = \overline{0}$
- For local-optima need $\textsc{inf} : \forall a, f : a \leq f(a)$

Simplest model for "fixed" interdomain routing

- metrics of the form $(c,\ d)$ or ∞, where $c \in \{0, 1, 2\}$ and d is a path length,
- metrics compared lexicographically.
- 0 is for *downstream* routes (towards paying customers),
- 1 is for *peer* routes (towards competitor's customers),
- 2 is for *upstream* routes (towards charging providers),

The inflationary policy functions

- Gao/Rexford rules in red.

	0	1	2		0	1	2
a	0	1	2	**m**	2	1	2
b	0	1	∞	**n**	2	1	∞
c	0	2	2	**o**	2	2	2
d	0	2	∞	**p**	2	2	∞
e	0	∞	2	**q**	2	∞	2
f	0	∞	∞	**r**	2	∞	∞
g	1	1	2	**s**	∞	1	2
h	1	1	∞	**t**	∞	1	∞
i	1	2	2	**u**	∞	2	2
j	1	2	∞	**v**	∞	2	∞
k	1	∞	2	**w**	∞	∞	2
l	1	∞	∞	**x**	∞	∞	∞

What Have We Learned from Reverse-Engineering the Internet's Inter-domain Routing Protocol?

Lessons

- Some non-distributive algebras make are useful.
- Local optimality is a useful notion for non-distributive algebras.
- Bellman-Ford (path vectoring) can compute left-local optima ...
- ... and so can Dijkstra's algorithm!