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

Advanced Networks Colloquium
ECE — University of Maryland
28 October, 2011
The Answer!

- Hitherto algebraic path problems have focused on **global optimality**: finding best paths over all possible paths.
- Another notion is **local optimality**: each node gets the best paths it can obtain given what is available from its neighbors (routing in equilibrium).
- The two notions coincide in the classical theory.

We have learned that in some cases ...

- Algebraic path problems admit unique local optima that are distinct from global optima.
- Local optima represent a more meaningful solution.
- We can find local optima in polynomial time.
Shortest paths example, $sp = (\mathbb{N}^\infty, \text{min}, +)$

The adjacency matrix

$$A = \begin{bmatrix}
\infty & 2 & 1 & 6 & \infty \\
2 & \infty & 5 & \infty & 4 \\
1 & 5 & \infty & 4 & 3 \\
6 & \infty & 4 & \infty & \infty \\
\infty & 4 & 3 & \infty & \infty
\end{bmatrix}$$
Shortest paths example, continued

The routing matrix

\[
A^* = \begin{bmatrix}
1 & 2 & 3 & 4 & 5 \\
1 & 0 & 2 & 1 & 5 & 4 \\
2 & 2 & 0 & 3 & 7 & 4 \\
3 & 1 & 3 & 0 & 4 & 3 \\
4 & 5 & 7 & 4 & 0 & 7 \\
5 & 4 & 4 & 3 & 7 & 0 \\
\end{bmatrix}
\]

Matrix \(A^*\) solves this global optimality problem:

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

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

\[
\begin{bmatrix}
1 & 2 & 3 & 4 & 5 \\
1 & \infty & 4 & 4 & 6 & 4 \\
2 & 4 & \infty & 5 & 4 & 4 \\
3 & 4 & 5 & \infty & 4 & 4 \\
4 & 6 & 4 & 4 & \infty & 4 \\
5 & 4 & 4 & 4 & 4 & \infty \\
\end{bmatrix}
\]

Matrix \( A^* \) solves this global optimality problem:

\[
A^*(i, j) = \max_{p \in P(i, j)} \min w(p),
\]

where \( w(p) \) is now the minimal edge weight in \( p \).
Fun example, \((2\{a, b, c\}, \cup, \cap)\)

We want a Matrix \(A^*\) to solve this global optimality problem:

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

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

For \(x \in \{a, b, c\}\), interpret \(x \in A^*(i, j)\) 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 \(A^*\)

\[
\begin{bmatrix}
  1 & 2 & 3 & 4 & 5 \\
  1 \{a, b, c\} & \{a, b, c\} & \{a, b, c\} & \{a, b\} & \{b, c\} \\
  2 \{a, b, c\} & \{a, b, c\} & \{a, b, c\} & \{a, b\} & \{b, c\} \\
  3 \{a, b, c\} & \{a, b, c\} & \{a, b, c\} & \{a, b\} & \{b, c\} \\
  4 \{a, b\} & \{a, b\} & \{a, b\} & \{a, b, c\} & \{b\} \\
  5 \{b, c\} & \{b, c\} & \{b, c\} & \{b\} & \{a, b, c\}
\end{bmatrix}
\]
Semirings

A few examples

<table>
<thead>
<tr>
<th>name</th>
<th>$S$</th>
<th>$\oplus$</th>
<th>$\otimes$</th>
<th>$\bar{0}$</th>
<th>$\bar{1}$</th>
<th>possible routing use</th>
</tr>
</thead>
<tbody>
<tr>
<td>sp</td>
<td>$\mathbb{N}^\infty$</td>
<td>min</td>
<td>$+$</td>
<td>$\infty$</td>
<td>0</td>
<td>minimum-weight routing</td>
</tr>
<tr>
<td>bw</td>
<td>$\mathbb{N}^\infty$</td>
<td>max</td>
<td>min</td>
<td>0</td>
<td>$\infty$</td>
<td>greatest-capacity routing</td>
</tr>
<tr>
<td>rel</td>
<td>$[0, 1]$</td>
<td>max</td>
<td>$\times$</td>
<td>0</td>
<td>1</td>
<td>most-reliable routing</td>
</tr>
<tr>
<td>use</td>
<td>${0, 1}$</td>
<td>max</td>
<td>min</td>
<td>0</td>
<td>1</td>
<td>usable-path routing</td>
</tr>
<tr>
<td>use</td>
<td>$\mathcal{2}^\mathcal{W}$</td>
<td>$\cup$</td>
<td>$\cap$</td>
<td>${}$</td>
<td>$\mathcal{W}$</td>
<td>shared link attributes?</td>
</tr>
<tr>
<td>use</td>
<td>$\mathcal{2}^\mathcal{W}$</td>
<td>$\cap$</td>
<td>$\cup$</td>
<td>${}$</td>
<td>$\mathcal{W}$</td>
<td>shared path attributes?</td>
</tr>
</tbody>
</table>

Path problems focus on global optimality

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

Graphs, Dioids and Semirings

New Models and Algorithms

Path Problems in Networks

John Baras
George Theodorakopoulos
What algebraic properties are needed for efficient computation of global optimality?

Distributivity

L.D : \( a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c), \)

R.D : \( (a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c). \)

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

L.DIST : \( a + (b \min c) = (a + b) \min (a + c), \)

R.DIST : \( (a \min b) + c = (a + c) \min (b + c). \)

But some realistic metrics are not distributive! What can we do?
Left-Local Optimality

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

$$L = (A \otimes L) \oplus I.$$ 

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

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

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

\[ \text{tgg (cl.cam.ac.uk)} \]
Right-Local Optimality

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

$$R = (R \otimes A) \oplus I.$$  

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

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

- $R(i, j)$ is the best possible value given the values $R(q, j)$, for all in-neighbors $q$ of destination $j$.
- Rows $L(i, \_)$ represents **out-trees from** $i$ (think Dijkstra).
- Columns $L(\_, i)$ represents **in-trees to** $i$.
- 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.

\[ A^* = L = R \]

### Without

Suppose that we drop distributivity and \( A^*, L, R \) 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

\[
\mathbf{A}^* = \begin{bmatrix}
(\infty, 0) & (5, 1) & (0, \infty) & (0, \infty) & (0, \infty) \\
(0, \infty) & (\infty, 0) & (0, \infty) & (0, \infty) & (0, \infty) \\
(5, 2) & (5, 3) & (\infty, 0) & (5, 1) & (5, 2) \\
(10, 6) & (5, 2) & (5, 2) & (\infty, 0) & (10, 1) \\
(10, 5) & (5, 4) & (5, 1) & (5, 2) & (\infty, 0)
\end{bmatrix},
\]
Left local optima

\[ \mathbf{L} = \begin{bmatrix}
  1 & 2 & 3 & 4 & 5 \\
  1 & (\infty, 0) & (5, 1) & (0, \infty) & (0, \infty) \\
  2 & (0, \infty) & (\infty, 0) & (0, \infty) & (0, \infty) \\
  3 & (5, 7) & (5, 3) & (\infty, 0) & (5, 1) & (5, 2) \\
  4 & (10, 6) & (5, 2) & (5, 2) & (\infty, 0) & (10, 1) \\
  5 & (10, 5) & (5, 4) & (5, 1) & (5, 2) & (\infty, 0)
\end{bmatrix}, \]

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

\[
\begin{bmatrix}
1 & 2 & 3 & 4 & 5 \\
1 & (\infty, 0) & (5, 1) & (0, \infty) & (0, \infty) \\
2 & (0, \infty) & (\infty, 0) & (0, \infty) & (0, \infty) \\
3 & (5, 2) & (5, 3) & (\infty, 0) & (5, 1) & (5, 2) \\
4 & (10, 6) & (5, 6) & (5, 2) & (\infty, 0) & (10, 1) \\
5 & (10, 5) & (5, 5) & (5, 1) & (5, 2) & (\infty, 0)
\end{bmatrix}
\]
Left-locally optimal paths to node 2
Right-locally optimal paths to node 2

1 → 2, 3, 4 → 2
3 → 2
4 → 2
5 → 2
3 → 2
5 → 2
4 → 2
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
The primary source for violations of distributivity.
Bellman-Ford can compute left-local solutions

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

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

Convergence to a unique left-local solution is guaranteed. Currently no bound is known on the number of iterations required.
Of course BGP does not satisfy these conditions!

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.
Recent observation: Dijkstra’s algorithm can work for right-local optima.

**Input**: adjacency matrix $A$ and source vertex $i \in V$,

**Output**: the $i$-th row of $R$, $R(i, \_)$.

```
begin
  S ← \{i\}
  R(i, i) ← 1
  for each $q \in V - \{i\}$ : $R(i, q) \leftarrow A(i, q)$
  while $S \neq V$
    begin
      find $q \in V - S$ such that $R(i, q)$ is $\leq_L$ -minimal
      $S ← S \cup \{q\}$
      for each $j \in V - S$
        $R(i, j) \leftarrow R(i, j) \oplus (R(i, q) \otimes A(q, j))$
    end
end
```
Assumptions on \((S, \oplus, \otimes, \bar{0}, \bar{1})\) that guarantee existence of right-local optima

- \((S, \oplus, \bar{0})\) is a commutative, idempotent, and selective monoid,
- \((S, \otimes, \bar{1})\) is a monoid,
- \(\bar{0}\) is the annihilator for \(\otimes\),
- \(\bar{1}\) is the annihilator for \(\oplus\),
- Right inflationarity, \(R.INF : \forall a, b : a \leq a \otimes b\)

Here \(a \leq b \equiv a = a \oplus b\).
Using a Link-State approach with hop-by-hop forwarding ...

Need left-local optima!

\[ L = (A \otimes L) \oplus I \iff L^T = (L^T \hat{\otimes} A^T) \oplus I \]

where \( \otimes^T \) is matrix multiplication defined with as

\[ a \otimes^T b = b \otimes a \]

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

Each node would have to solve the entire “all pairs” problem.
Functions on arcs

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

- \((S, \oplus, \overline{0})\) is a commutative, idempotent, and selective monoid,
- \(\forall f \in F : f(\overline{0}) = \overline{0}\)
- For local-optima need \(\text{INF} : \forall a, f : a \leq f(a)\)
Simplest model for interdomain routing

- 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)

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>m</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>b</td>
<td>0</td>
<td>1</td>
<td>∞</td>
<td>n</td>
<td>2</td>
<td>1</td>
<td>∞</td>
</tr>
<tr>
<td>c</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>o</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>d</td>
<td>0</td>
<td>2</td>
<td>∞</td>
<td>p</td>
<td>2</td>
<td>2</td>
<td>∞</td>
</tr>
<tr>
<td>e</td>
<td>0</td>
<td>∞</td>
<td>2</td>
<td>q</td>
<td>2</td>
<td>∞</td>
<td>2</td>
</tr>
<tr>
<td>f</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
<td>r</td>
<td>2</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>g</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>s</td>
<td>∞</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>h</td>
<td>1</td>
<td>1</td>
<td>∞</td>
<td>t</td>
<td>∞</td>
<td>1</td>
<td>∞</td>
</tr>
<tr>
<td>i</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>u</td>
<td>∞</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>j</td>
<td>1</td>
<td>2</td>
<td>∞</td>
<td>v</td>
<td>∞</td>
<td>2</td>
<td>∞</td>
</tr>
<tr>
<td>k</td>
<td>1</td>
<td>∞</td>
<td>2</td>
<td>w</td>
<td>∞</td>
<td>∞</td>
<td>2</td>
</tr>
<tr>
<td>l</td>
<td>1</td>
<td>∞</td>
<td>∞</td>
<td>x</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
</tbody>
</table>
Conclusion

Take away message
If your algebraic model is not distributive, then ask yourself if a left- or right-local solution is reasonable. If so, use Dijkstra’s algorithm (with care).

A few open problems
- How many Bellman iterations are needed to find \( L \)?
- Is there an equational axiomatization of local optimality? (For classical theory we have Kleene Algebras).