Acyclic Type-of-Relationship Problems on the Internet: An Experimental Analysis

Benjamin Hummel and Sven Kosub

Technische Universität München

IMC’07
Details

- Hummel, Kosub: *Acyclic Type-of-Relationship Problems on the Internet: An Experimental Analysis*
- The same as (more detailed) technical report
- Theoretical foundation:
  Kosub, Maaß, Täubig: *Acyclic Type-of-Relationship Problems on the Internet*

Here simplified version:
  only customer-provider relationships
The problem

Given *AS paths* from publicly available routing tables, determine the *business relationships* between the ASes.
The problem

Given AS paths from publicly available routing tables, determine the business relationships between the ASes.

BGP business relationships allow the inference of BGP policies and serve as input to

- BGP simulation
- analysis of BGP instabilities
- optimizing Internet operation
- business decisions
- ...
General Approach

Start from AS path set

- AS1, AS2, AS3, AS4
- AS5, AS2, AS3
General Approach

Start from AS path set

- AS1, AS2, AS3, AS4
- AS5, AS2, AS3

Construct AS graph
General Approach

Start from AS path set

- AS1, AS2, AS3, AS4
- AS5, AS2, AS3

Construct AS graph and find orientation

![AS Graph Example]
General Approach

Start from AS path set
- AS1, AS2, AS3, AS4
- AS5, AS2, AS3

Construct AS graph and find orientation

Orientation can be carried over to paths
- AS1 → AS2 ← AS3 ← AS4
- AS5 → AS2 ← AS3
How to find the orientation?

[Gao 2001]:
*Selective Export Rule* yields *valley-freeness* criterion

Intuitive meaning:
No two ASes may route their traffic over a common customer
How to find the orientation?

[Gao 2001]:
Selective Export Rule yields valley-freeness criterion

Intuitive meaning:
No two ASes may route their traffic over a common customer

Problem solved by “traditional” algorithms:
Find an orientation of the edges,
such that no path contains the pattern ← →
Is this a valid orientation?

If one of these ASes cancels its contract, it still has full connectivity!
Acyclic Inference

New problem:
Orient graph such that valley-freeness is respected and the graph is acyclic

- can be solved efficiently (similar to finding a topological sort)
- real instances are unsolvable \( \Rightarrow \) minimize violations heuristically
- Additional benefit: can deal with (verified) pre-knowledge
Experimental Comparison of Algorithms

- Gao
- Derandomised approximation algorithm (APX)
- Combinatorial/2-SAT (DPP*)
- Acyclic heuristic (AHeu)
Data used for Comparison

From publicly available routing tables:

- 2,002,680 AS paths (average length 3.43)
- containing 21,862 ASes (56,922 AS pairs)

From WHOIS (RPSL) and BGP communities attribute:

- 2,739 customer-to-provider edges
- graph only consisting of reference edges is acyclic!
# Experimental Results

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Invalid paths</th>
<th>Misclassified c-to-p for reliable edge set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gao</td>
<td>27.366% (249 not valley-free) (547811 with s-to-s edge)</td>
<td>1.387% (4 as p-to-c) (34 as s-to-s)</td>
</tr>
<tr>
<td>APX</td>
<td>4.483% (89775 not valley-free)</td>
<td>5.330% (146 as p-to-c)</td>
</tr>
<tr>
<td>DPP*</td>
<td>0.519% (10391 not valley-free)</td>
<td>0.913% (25 as p-to-c)</td>
</tr>
<tr>
<td>AHeu ($W = 10$)</td>
<td>0.483% (9666 not valley-free)</td>
<td>0.292% (8 as p-to-c)</td>
</tr>
</tbody>
</table>
Conclusion

- The Internet hierarchy seems to be acyclic . . .
- . . . so acyclicity should be respected when solving the ToR problem
- . . . and maybe also for other algorithms/applications
Thanks!

Questions?

Data & Code:
http://www14.in.tum.de/software/BGP/hummel-kosub-07.html
Email: {hummelb, kosub}@in.tum.de
Using Pre-Knowledge (1)

Number of misclassified C2P edges
Relative amount of pre-knowledge
Gao
APX
DPP*
AHeu (W=10)
Pre-Knowledge (2)

Number of invalid paths vs. Relative amount of pre-knowledge for different algorithms:
- Gao
- APX
- DPP*
- AHeu (W=10)

B. Hummel, S. Kosub
Acyclic Type-of-Relationship Problems
IMC'07
Example: Acyclic Inference Algorithm

Pfad 2: AS1 – AS2 – AS3 – AS4
Example: Acyclic Inference Algorithm

Pfad 1: AS2 – AS1 – AS3 – AS5 ← AS4
Pfad 2: AS1 – AS2 – AS3 ← AS4
Example: Acyclic Inference Algorithm

Pfad 1: AS2 – AS1 – AS3 – AS5 ← AS4
Pfad 2: AS1 – AS2 – AS3 ← AS4
Example: Acyclic Inference Algorithm

Pfad 1: AS2 – AS1 – AS3 ← AS5 ← AS4
Pfad 2: AS1 – AS2 – AS3 ← AS4
Example: Acyclic Inference Algorithm

Pfad 1: AS2 – AS1 – AS3 ← AS5 ← AS4
Pfad 2: AS1 – AS2 – AS3 ← AS4
Example: Acyclic Inference Algorithm

Pfad 1: AS2 – AS1 ← AS3 ← AS5 ← AS4
Pfad 2: AS1 – AS2 ← AS3 ← AS4
Example: Acyclic Inference Algorithm

Pfad 1: AS2 – AS1 ← AS3 ← AS5 ← AS4
Pfad 2: AS1 – AS2 ← AS3 ← AS4
Example: Acyclic Inference Algorithm

Pfad 1: AS2 ← AS1 ← AS3 ← AS5 ← AS4
Pfad 2: AS1 → AS2 ← AS3 ← AS4
Example: Unsolvable Instance

Pfad 1: AS1 – AS2 – AS3
Pfad 2: AS2 – AS3 – AS1
Pfad 3: AS3 – AS1 – AS2
Example: Unsolvable Instance

Pfad 1: AS1 → AS2 – AS3
Pfad 2: AS2 – AS3 – AS1
Pfad 3: AS3 – AS1 → AS2
Example: Unsolvable Instance

Pfad 1: AS1 → AS2 – AS3
Pfad 2: AS2 – AS3 → AS1
Pfad 3: AS3 → AS1 → AS2
Example: Unsolvable Instance

Pfad 1: AS1 → AS2 → AS3
Pfad 2: AS2 → AS3 → AS1
Pfad 3: AS3 → AS1 → AS2