BGP non-convergence

marcelo bagnulo

Introduction

- BGP has no guaranteed convergence
- Other routing protocols, they try to solve the shortest path problem
- What problem tries to solve BGP?
- The stabel path problem formulation

Modeling BGP route selection (I)

- Simplifying assumptions
  - Ignore IBGP related issues
  - Ignore MED attribute
  - Assume at most one link between two ASes
  - Ignore Route aggregation
- Information contained in UPDATE records
  - Nlri
  - next-hop
  - as_path
  - local_pref
  - c_set
- Ranking: for the same nlri

$$rank\_tuple(r) = \left\langle r.local\_pref, \frac{1}{r.as\_path}, \frac{1}{r.next\_hop} \right\rangle$$

Modeling BGP route selection (II)

- Route transformation *T*(r): operates by deleting, inserting or modifying atributes values
- If u and w autonomous systems, the a record moves from u to w suffers the folllowing transformations:
  - r_1=export(u<-w,r) export policies defined by w
  - r_2=PVT(u<-w, r_1) Path Vector Trans
    - add w to AS path, sets next hop, filters loops
  - r_3=import(u<-w, r_2) import policies defined by u
- Peering transformation
  - pt(u<-w,r)=import(u<-w,PVT(u<-w,export(u<-w,r)))

Modeling BGP route selection (III)

- AS u_0 is the origin of a destiantion d sending record r_0
- AS U_k and $P=u_k u_{k-1} \dots u_0$ a path, then r(P) is the route record received at u_k from u_0
  - $r(P)=pt(u_k \leftarrow u_{k-1}, pt(u_{k-1} \leftarrow u_{k-2}, \dots pt(u_1 \leftarrow u_0, r_0) \dots)$
  - P is permited at u_k if r(P) is non empty
- Ranking function

$$\lambda^{u_k}(P) = lexical\_rank(rank\_tuple(r(P))$$

Stable Path Problem (SPP) (I)

- G=(V,E), simple undirected graph
  - V={0,1,…,n] nodes
  - E, set of edges
- Node 0 (origin) special cause is the destination
- peers(u)
- Path: P= $(v_k, v_{k-1}, \dots, v_0)$ seq of nodes
- For each v of V, P^v is set of permited paths
- *P* is the union of all P^v
- For each v, ranking function $\lambda^v(P)$ where P is in P^v
  - $\lambda^v(P_1) > \lambda^v(P_2)$ => P_1 is preferred
  - $\Lambda = \{\lambda^v / v$ belongs to V-{0}}

Stable Path Problem (SPP) (II)

- Instance of the SPP *S*=(G,*P*,Λ) (graph, set of permited paths and ranking functions) and:
  - P^0={{0}} and for all v except 0
    - Empty path is permitted
    - Empty path is always ranked last
    - Strictness: If $P_1 \neq P_2$ and $\lambda^v(P_1) > \lambda^v(P_2)$=> they have the same next hop
    - Simplicity: all paths in *P* have no repeated nodes

Stable Path Problem (SPP) (III)

- Instance of the SPP *S*=(G,*P*,Λ)
- Path assigment function π maps a node u to a path π(u) from P^u
  - π(u) empty means u has no path to the origin
- Path choices(π,u)

$$choices(\pi,u) = \{ \begin{matrix} \{(uv)\pi(v) / \{u,v\} \in E\} \cap P^u, u \neq 0 \\ \{(0)\}, o.w. \end{matrix}$$

- W subset of P^u with different next hop

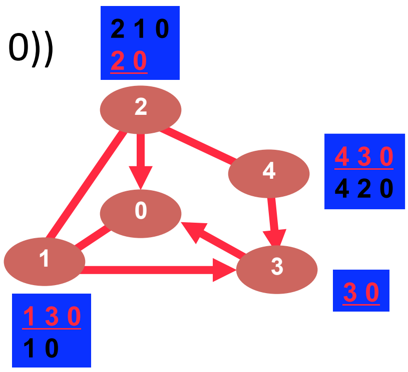
$$best(W,u) = P \in W, \max \lambda^u(P)$$

Stable Path Problem (SPP) (IV)

- A path assigment π is stable at a node u if
  π(u)=best(choices(π,u),u)
- A SPP *S*=(G,*P*,Λ) is solvable if if there is a stable path assigment for all u of S

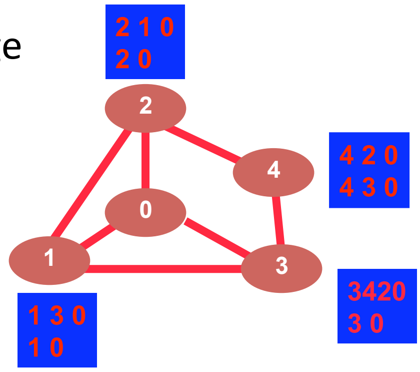
Example 1: good gadget

- Only one solution
- ((1 3 0),(2 0),(3 0),(4 3 0))
- Note that not only shortest paths are preferred

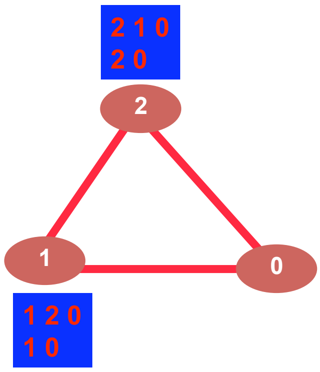


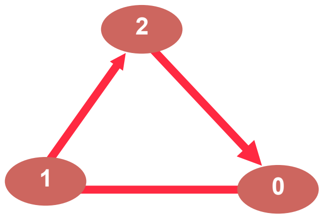
Example 2: bad gadget

- No solution
- Protocol always diverge

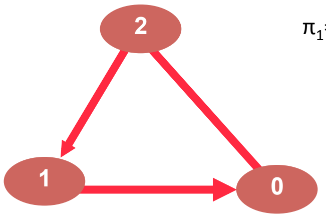


Example 3: Disagree





π_1=((1 2 0),(2 0))



π_2=((2 1 0),(1 0))

Simple Path Vector Protocol (SPVP)

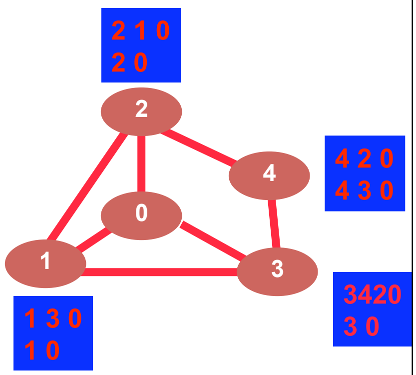
- Abstract version of BGP
- Always diverges when the SPP has no solution
- Assume reliable FIFO queue for messages
- Messages exhcnaged are simply paths
- When node u adopts one path P from P^u, it informs all its peers by sending them P
- Data strcutures in u
  - rib(u) contains current path to the origin
  - rib-in(u<=w) for each w, sotres the most recent path
- choices(u)={(u w)P of P^u / P=rib-in(u<=w)}
- Best possible path: best(u)=best(choices(u),u)

SPVP algorithm

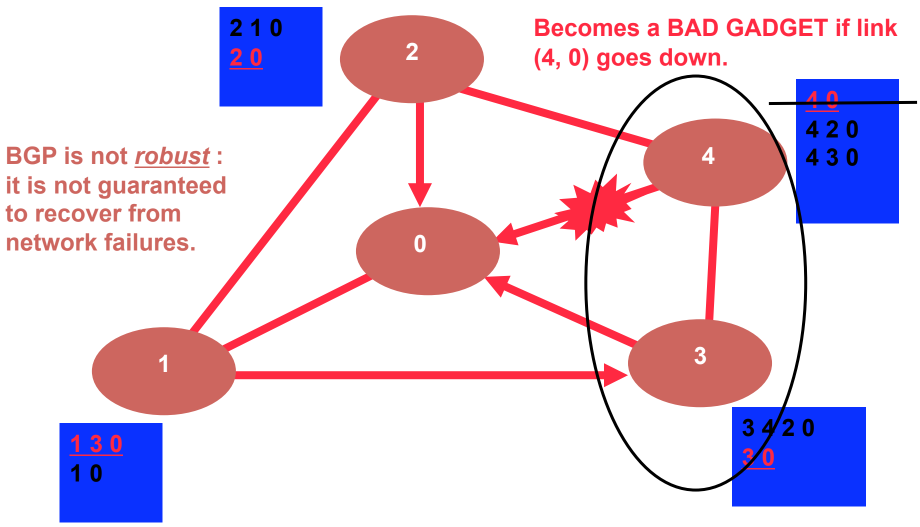
```
process svpv(u)
begin
    receive P from w
        begin
            rib-in(u<=w):=P
            if rib(u) ≠ best(u) then
            begin
                rib(u):=best(u)
                for each v of peers(u) do
                begin
                    send rib(u) to v
                end
            end
        end
end
```

SPVP and the bad gadget

step	π
0	(1 0) (2 0) (3 4 2 0) (4 2 0)
1	(1 0) (2 1 0) (3 4 2 0) (4 2 0)
2	(1 0) (2 1 0) (3 4 2 0) ε
3	(1 0) (2 1 0) (3 0) ε
4	(1 0) (2 1 0) (3 0) (4 3 0)
5	(1 3 0) (2 1 0) (3 0) (4 3 0)
6	(1 3 0) (2 0) (3 0) (4 3 0)
7	(1 3 0) (2 0) (3 0) (4 2 0)
8	(1 3 0) (2 0) (3 4 2 0) (4 2 0)
9	(1 0) (2 0) (3 4 2 0) (4 2 0)



System may become unstable after a failure



Stability and safety

- Network states are the collection of values of rib(u), rib-in(u<=v) and state of communication links
- A network state is stable if communications links are empty
- Path assigment of a stable network state is a stable path assignment
- A stable path problem is safe if the SPVP always converge

Dispute wheels (I)

- Deteming if a stable path assigment exsits is an NP hard problem
- Dispute wheels are an heuristic to find a stable paht assignment
- Suppose V' contained in V such that 0 is in V'
- Partial path assigment π for V' is a path assigment such as
  - For all u of V', every node in π(u) is in V'
- Heursitic procdure to construct seq $V_0 \subset V_1 \subset \ldots \subset V_n$ along with π_0, π_1,..., π_n partial assigments for V_i
- Then for each π_i we construct π'_i such as
  - π'_i (u) = π_i(u) for u of V_i
  - π'_i (u) is empty for other u

Dispute wheels (II)

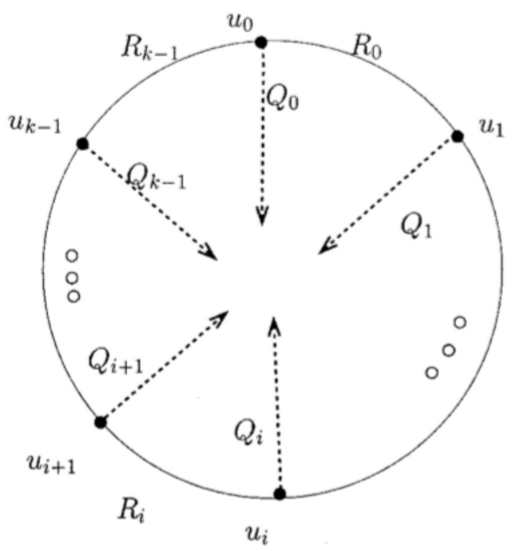
- If u belongs to $V-V_i$ and P belong to P^u, then P is consistent with π_i if
  - $P=P_1(u_i\ u_2)P_2$ where P_1 is a path in $V-V_i$ and u_2 belong to V_i and $P_2= \pi(u_2)$ and $\{u_1\ u_2]$ belongs to E
  - P is called direct path to V_i if P_2 is empty
- Let D_i be the set of nodes u of $V-V_i$ that have a direct path to V_i
- Let H_i the set of nodes of D_i that highest ranked path consistent with π_i is a direct path
  - This path is called B^u_i
- Let $V_{i+1} = V_i + H_i$
- Define partial assignment
- Continue till either V_k=V or V_k≠V and H_k=0

$$\pi_{i+1}(u) = \{ \begin{matrix} B_i^u, u \in H_i \\ \pi_i(u), u \in V_i \end{matrix}$$

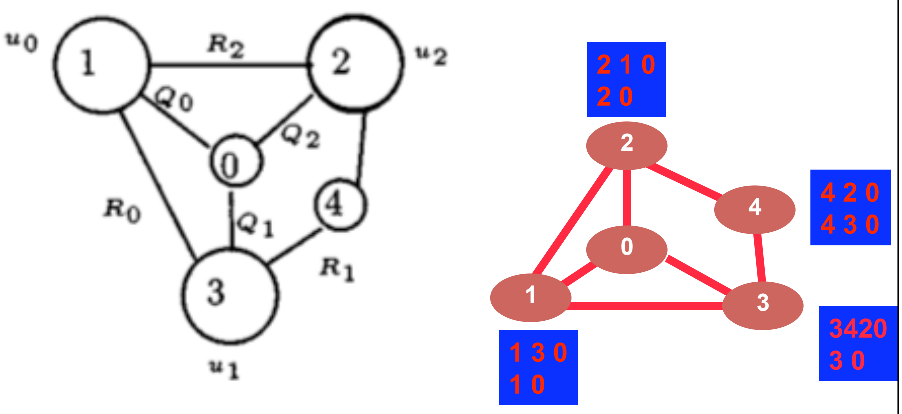
Dispute wheels (III)

- If we are in the second case, we have a circular set of conflicting rankings between nodes, called a dispute wheel
- Dispute wheel $\Pi = (\vec{U}, \vec{Q}, \vec{R})$ of size k
  - Seq of node $\vec{U} = u_0, u_1, ..., u_{k-1}$
  - Seq of non empty paths $\vec{Q} = Q_1, Q_2, ..., Q_{k-1}$ $\vec{R} = R_1, R_2, ..., R_{k-1}$
  - Such that for for each 0≤i≤k-1
    1. R_i is a path from u_i to u_{i+1}
    2. Q_i belongs to P^{u_i}
    3. R_iQ_{i+1} belongs to P^{u_i}
    4. $\lambda^{u_i}(Q_i) \le \lambda^{u_i}(R_i Q_{i+1})$

Dispute wheel (IV)



Dispute wheel for bad gadget



Properties of Dispute wheels

- No dispute wheel implies solvability
- No dispute wheel implies a unique solution
- No dispute wheel implies safety

Can we guarantee that BGO will not diverge?

- Operational practices
  - See next section
- Static analysis
  - Routing policy registry
  - Check for convergence
    - NP hard problem
    - ASes don't want to show policy information
- Dynamic solution?

Reference

- IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 10, NO. 2, APRIL 2002 The Stable Paths Problem and Interdomain Routing, Timothy G. Griffin, F. Bruce Shepherd, and Gordon Wilfong

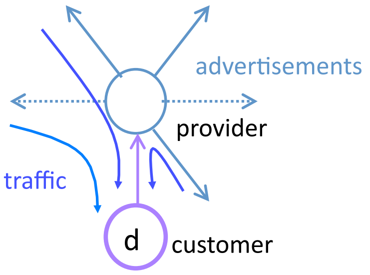
Relationships between ASes

- Peering
- Transit

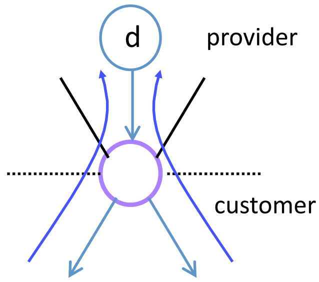
Transit relationship

- Customer pays provider for access to the Internet
  - Provider exports its customer's routes to everybody
  - Customer exports provider's routes only to downstream customers

Traffic **to** the customer



Traffic **from** the customer

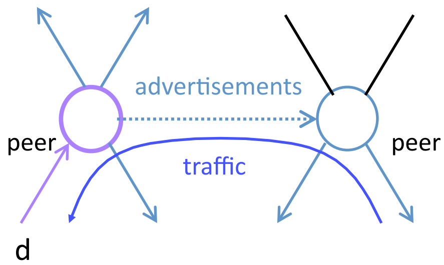


slide form Rexford

Peer relationship

- Peers exchange traffic between their customers
  - AS exports *only* customer routes to a peer
  - AS exports a peer's routes *only* to its customers

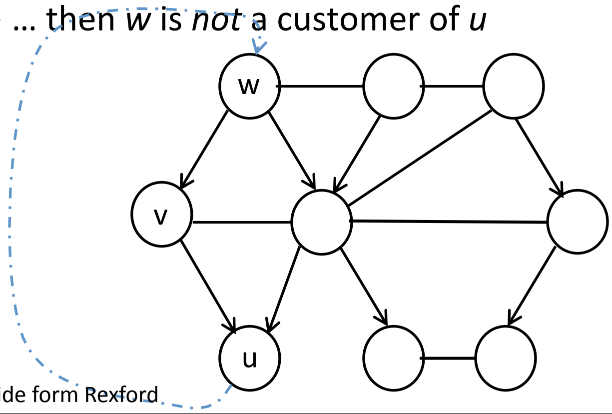
Traffic to/from the peer and its customers



slide form Rexford

Resulting hierarchy

- Provider-customer graph is a directed, acyclic graph
  - If *u* is a customer of *v* and *v* is a customer of *w*
  - … then *w* is *not* a customer of *u*



slide form Rexford

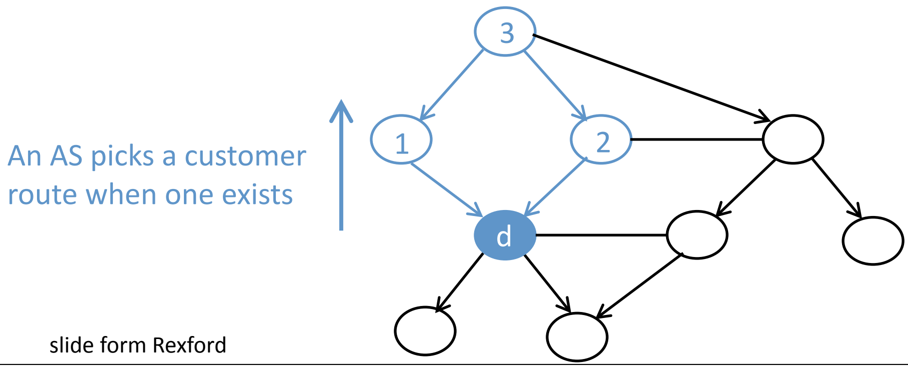
Proposed route selection

- Classify routes based on next-hop AS
  - Customer routes, peer routes, and provider routes
- Rank routes based on classification
  - Prefer *customer* routes over peer and provider routes
- Allow *any* ranking of routes within a class
  - E.g., can rank one customer route higher than another
  - Gives network operators the flexibility they need
- Consistent with traffic engineering practices
  - Customers pay for service, and providers are paid
  - Peer relationship contingent on balanced traffic load

slide form Rexford

Proof, Phase 1: Selecting Customer Routes

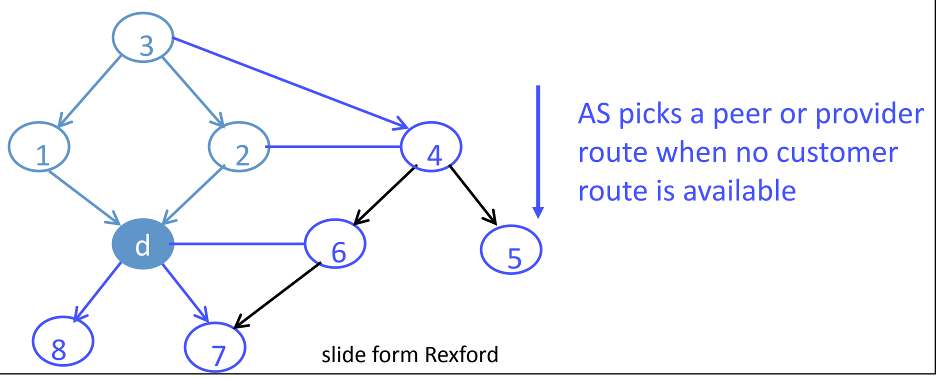
- Activate ASes in customer-provider order
  - AS picks a customer route if one exists
  - Decision of one AS cannot cause an earlier AS to change its mind



slide form Rexford

Proof, Phase 2: Selecting Peer and Provider Routes

- Activate rest of ASes in provider-customer order
  - Decision of one phase-2 AS cannot cause an earlier phase-2 AS to change its mind
  - Decision of phase-2 AS cannot affect a phase 1 AS



slide form Rexford

Reference

- L. Gao, J. Rexford, Stable Internet routing wihtout global coordination
- http://www.cs.princeton.edu/~jrex/teaching/spring2005/reading/gao01.pdf

Assignment

- Theorem 5.1 & proof
- Theorem 5.2 & proof
- Theorem 5.3 & proof