The Internet Ecosystem and Evolution

Contents

- Network routing: basics
  - distributed/centralized, static/dynamic, link-state/path-vector
  - intra-domain/inter-domain routing
- Mapping the service model to AS-AS paths
  - valley-free routing: feasible/forbidden paths, computing valley-free paths
  - prefer-customer policy, shortest AS-path, policy routing

Routing: Basics

Routing versus forwarding

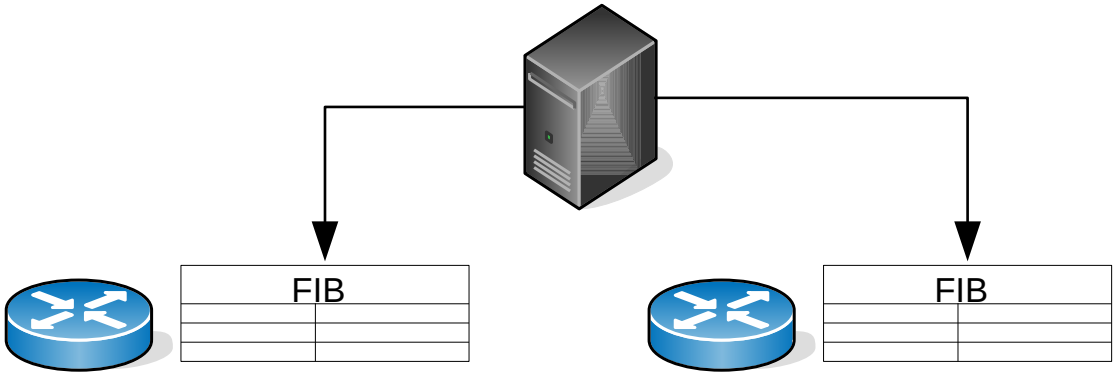
- The concepts of routing and forwarding are often confused
- **Forwarding:** passing each received packet to the next-hop IP address, by longest-prefix-matching the packet's destination address in the FIB (Forwarding Information Base)
- Forwarding needs a correct FIB to be in-place
- **Routing:** setting up/maintaining/updating the FIB based on topology information collected from the network
- Responsibility of dedicated routing protocols

Static versus dynamic

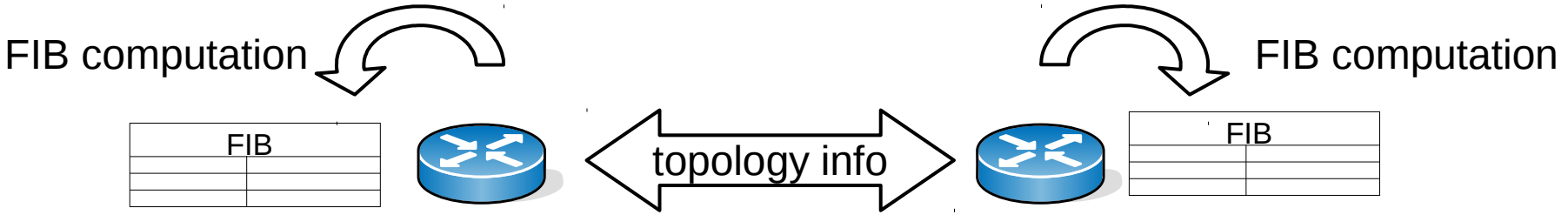
- **Static routing:** configured manually by the operator (via the CLI) – **rarely used**
  - simple, arbitrary routing can be configured
  - does not scale and does not adapt to topology changes (in case of a link/node failure)
- **Dynamic routing:** FIB maintained by a dedicated routing protocol – **widely used**
  - adaptive and scalable
  - but the paths are fixed by the routing protocol's policy (e.g., per-destination shortest path)

Centralized vs distributed

- **Centralized routing:** dedicated route server, PCE, SDN controller, sets all FIBs



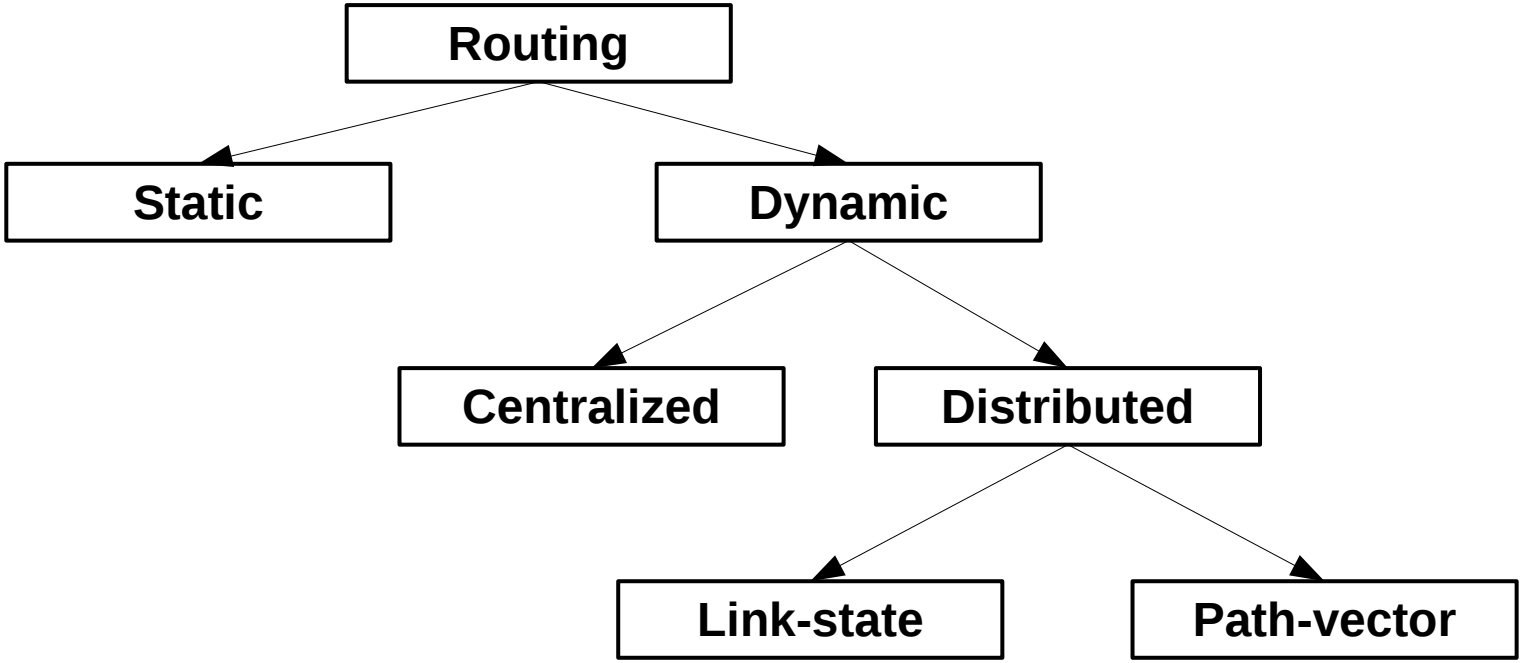
- **Distributed routing:** routers exchange topology descriptors and set up FIB individually



Link-state vs path-vector

- **Link-state:** routers maintain a distributed link-state database that describes the entire graph of the network
- They compute shortest paths and set FIB entries based on the first hop along the paths
- **Path-vector:** routers distribute available paths between each other
- Each router can pick a path it considers “best” according to its autonomous local policy
- Permits highly flexible policy routing
- **Distance-vector:** obsolete (RIP)

Taxonomy of routing schemes



Inside and between ASes

- There are different routing protocols deployed at different levels of the Internet
- **Intra-domain routing:** routing between hosts/ routers inside an AS
  - **Interior-Gateway Protocol (IGP)**
  - typically homogeneous routing policy (e.g., shortest path) across the AS: **link-state**
- **Inter-domain routing**: routing between ASes
  - **Exterior Gateway Protocol (EGP)**
  - heterogeneous routing policies: **path-vector**

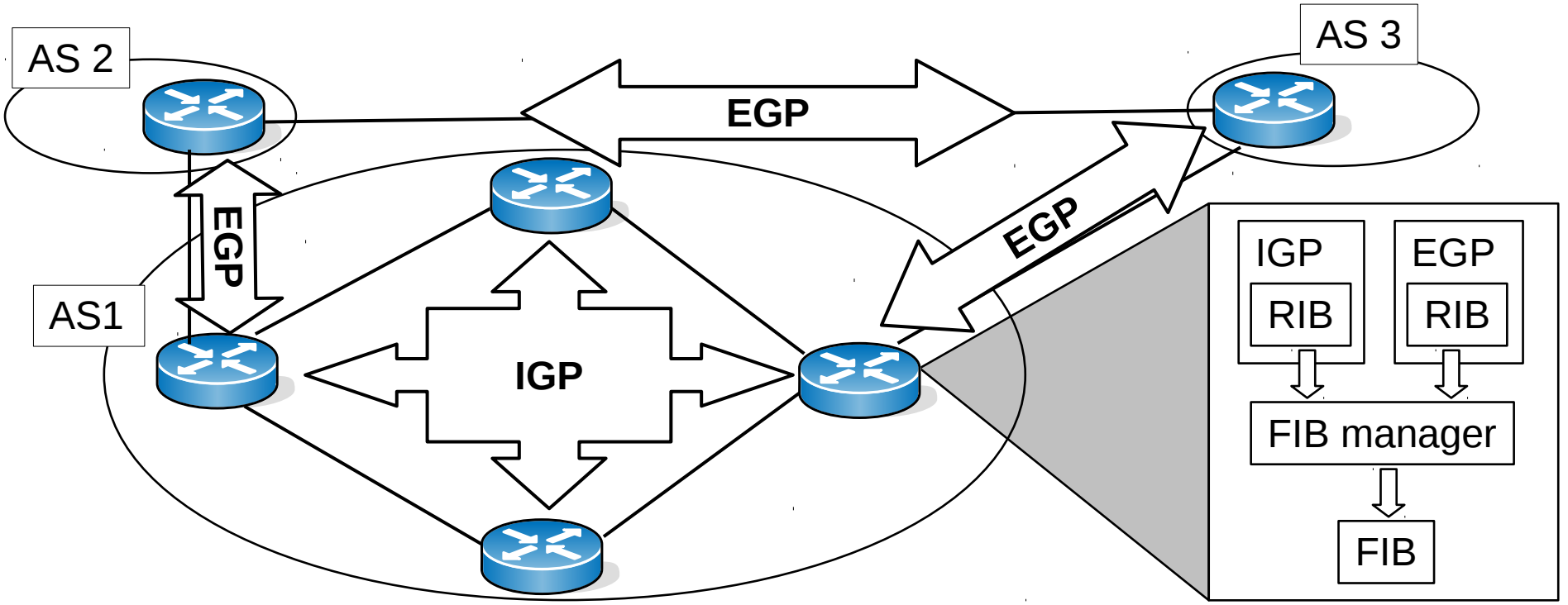
A „Big Four”

	Link-state	**Vector**
IGP	OSPF, IS-IS	RIP
EGP		BGP

- OSPF: Open-Shortest Path First
- IS-IS: Intermediate-System-to-Intermediate-System
- RIP: Routing Information Protocol
- BGP: Border Gateway Protocol

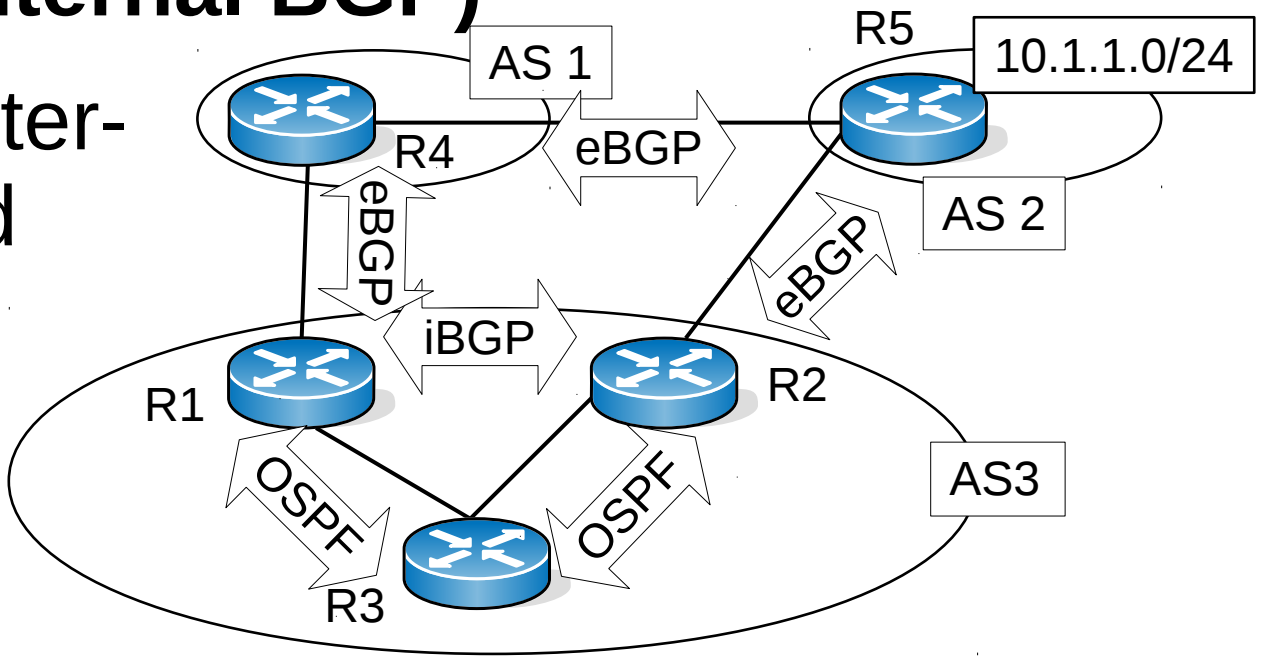
IGP/EGP cooperation

- Routers run multiple routing protocols side-by-side, e.g., an EGP and an IGP
  - each protocol maintains its own RIB
  - router distills a FIB from these RIBs



IGP/EGP cooperation

- `R1` can reach prefix `10.1.1.0/24` in two ways
  - via `AS1` (towards `R4`)
  - or via `AS2` (to `R2` and then through `R5`)
- Exterior routes must be distributed **inside** an AS as well: **iBGP (Internal BGP)**
- To distinguish, inter-AS BGP is called External BGP: **eBGP**



IGP/EGP collaboration

- Each routing protocol has its own RIB: all routing information known by the protocol (prefixes, available paths/next-hops to these prefixes, the best path/next-hop from these, etc.)
- Each router runs a **FIB manager**, responsible for assembling the FIB from the individual RIBs
- Operators associate **weights** to each protocol
- If a prefix occurs in more than one RIB, **the smallest weighted routing protocol's best path** is preferred

local < static < OSPF < eBGP < iBGP

- Local routes are always preferred
- eBGP < iBGP means that the path through a neighboring AS is preferred over a path received from a router within the same AS: **hot-potato routing** (later)
- Weights can be reconfigured arbitrarily

Mapping AS-AS business relationships to AS-level paths

Inter-domain routing

- **Routing policy:** a mechanism to represent an ISP's AS-level business strategies in the forwarding paths provisioned by the ISP
- Need a sophisticated routing protocol to be able to arbitrarily map business interests to paths (flexible policy routing) → BGP

How to encode these AS-level business interest into the forwarding paths?

Transit vs peer

- **Recall:** two ASes typically establish a transit or a peering relationship between one another
  - **transit:** global Internet access for a monthly fee
  - **peer:** „free” traffic exchange between two ASes and between any of their customers
- We focus on these two specific cases for now

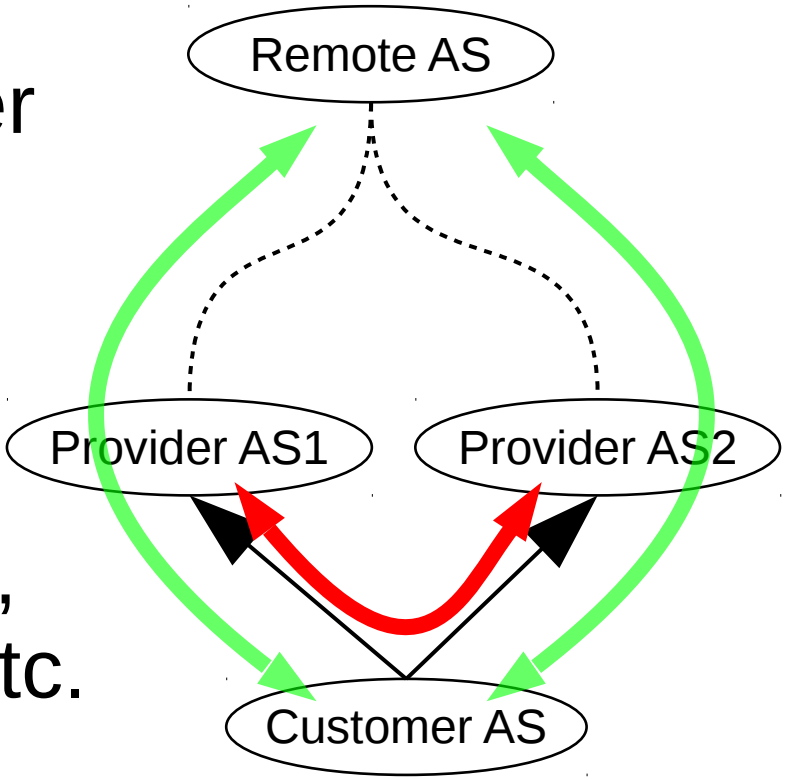
How does routes induced by the transit and, respectively, the peer relationships look like?

Feasible and prohibited paths

- The forwarding paths set up by ASes reflect the business/policy interests of ISPs
- **Feasible path:** a path between two ASes that align with the business interests of intermediary ASes
  - e.g., any transit path from a customer through a provider: **economic incentive for the provider to route** that traffic through its network
- **Forbidden path:** a forwarding path through an AS that goes against the economic incentives of the AS
  - e.g., between two transit providers through a common customer's network
  - the customer was not contracted to route such traffic → **economic incentive to block** this traffic

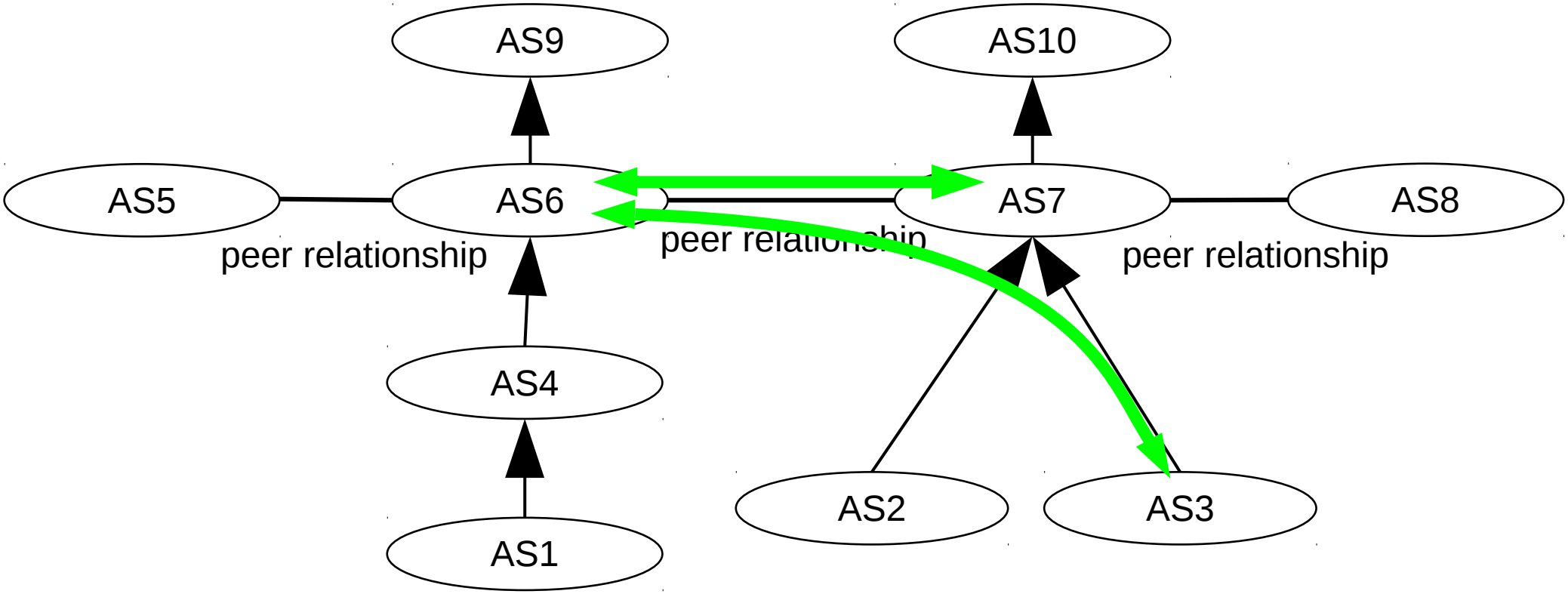
Transit: feasible/forbidden paths

- **Feasible path:** from the customer through any provider towards any remote AS (including the providers themselves and all their customers and peers): *Customer AS* ↔ *Provider AS1*, *Customer AS* ↔ *Remote AS*, etc.
- **Forbidden path:** between providers via the customer AS: *Provider AS1* → *Customer AS* → *Provider AS2*



Peer relationship: Feasible paths

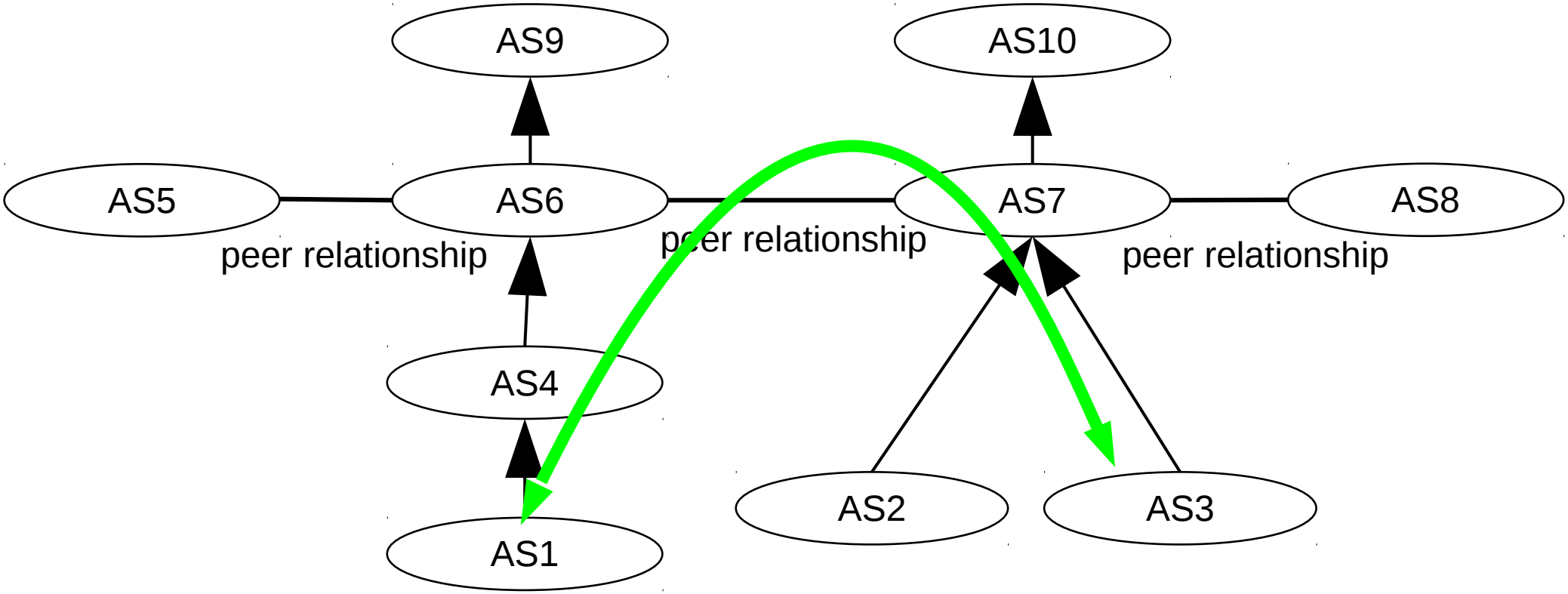
1) Between peer ASes: AS6-AS7, AS7-AS8, …
2) Between a customer and a peer: AS4-AS7, ...
   - customer's customer is also OK: AS1-AS7



Peer relationship: Feasible paths

3) Between customers of peer ASes: AS4-AS3

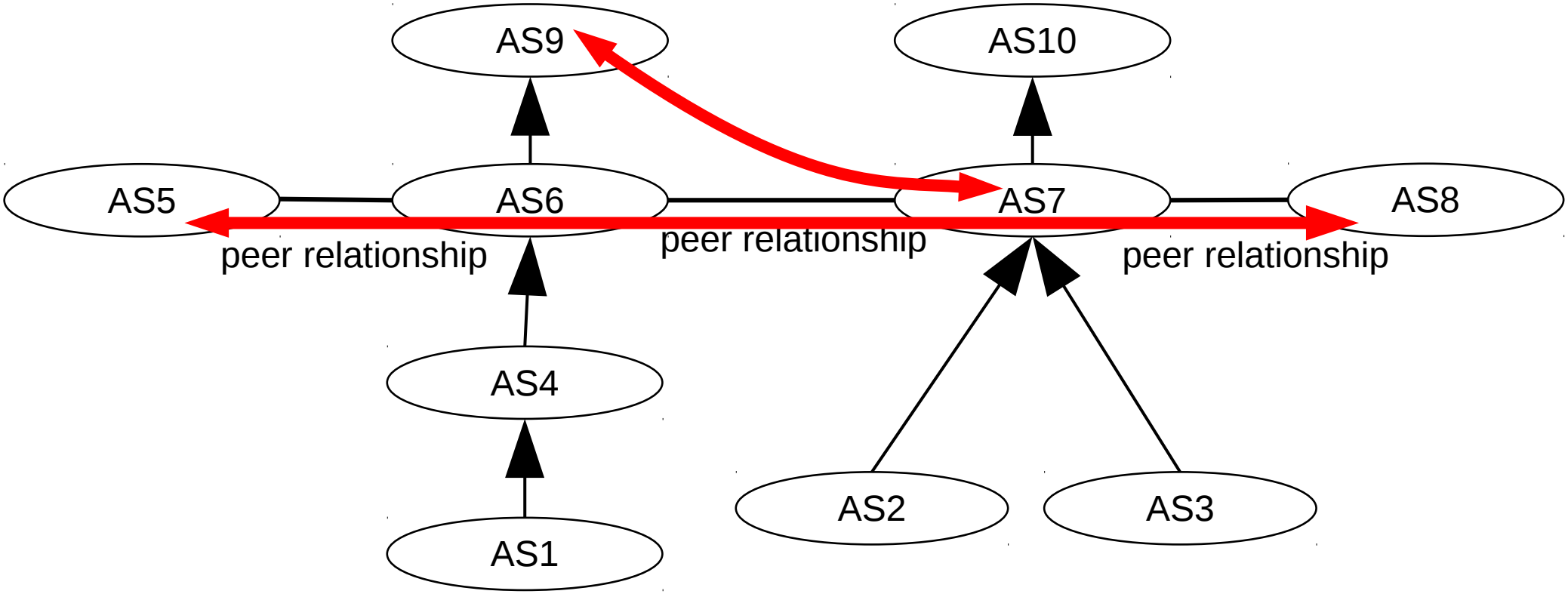
- “transit is transitive”: AS1-AS2, AS1-AS3 are also OK



Peer relationship: Forbidden paths

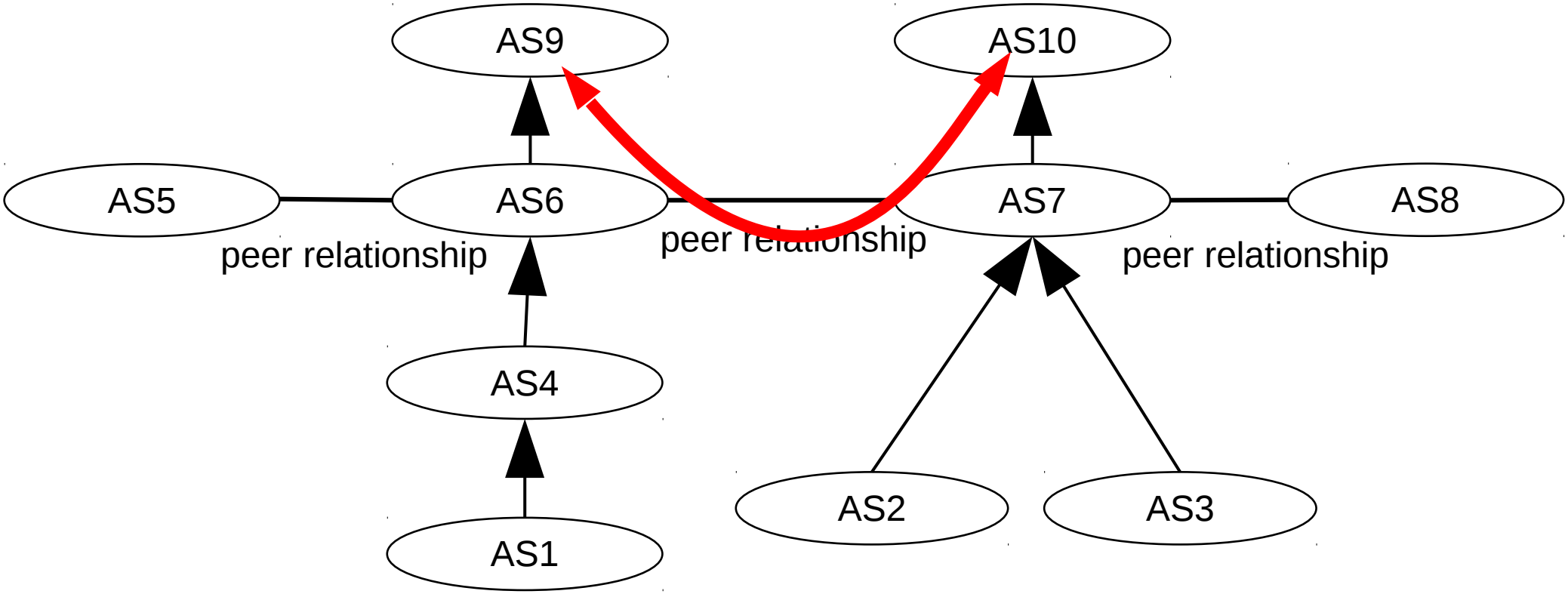
1) ASes connected via two or more peer-peer links: AS5-AS7, AS5-AS8 (“peering is not transitive”)

2) Between provider and a peer of an AS: AS9-AS7



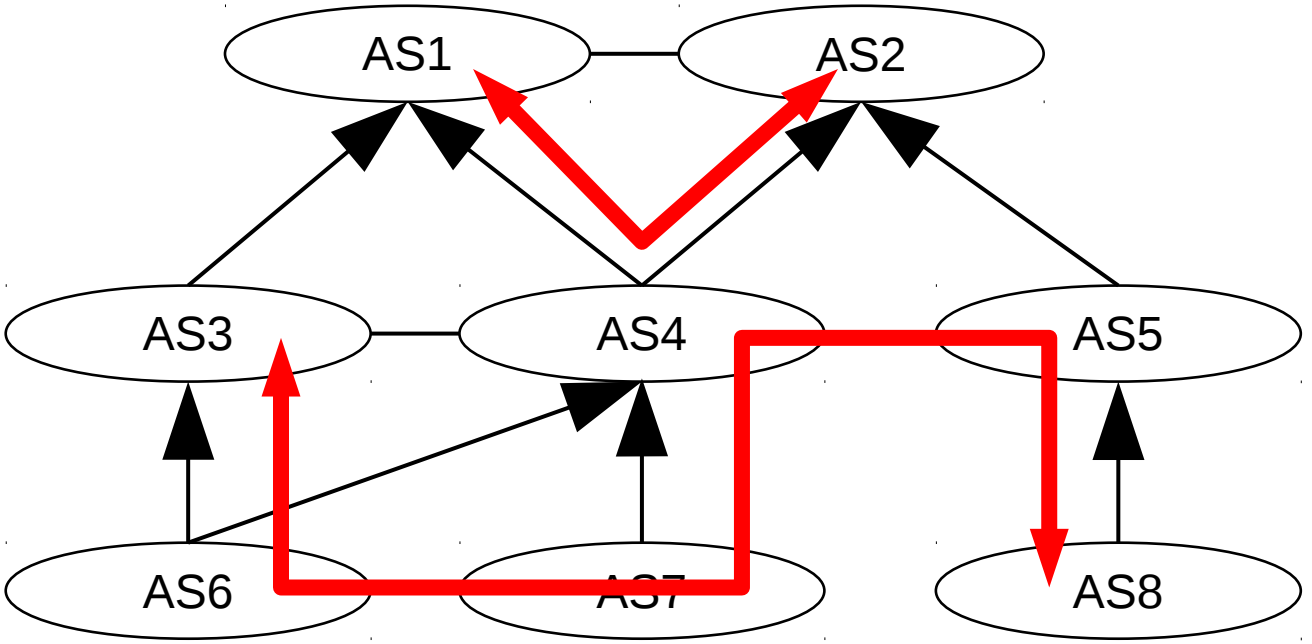
Peer relationship: Forbidden paths

3) Between an AS's provider and its peer's provider (AS9-AS10), as this would be a transit service for which transit fees could be charged



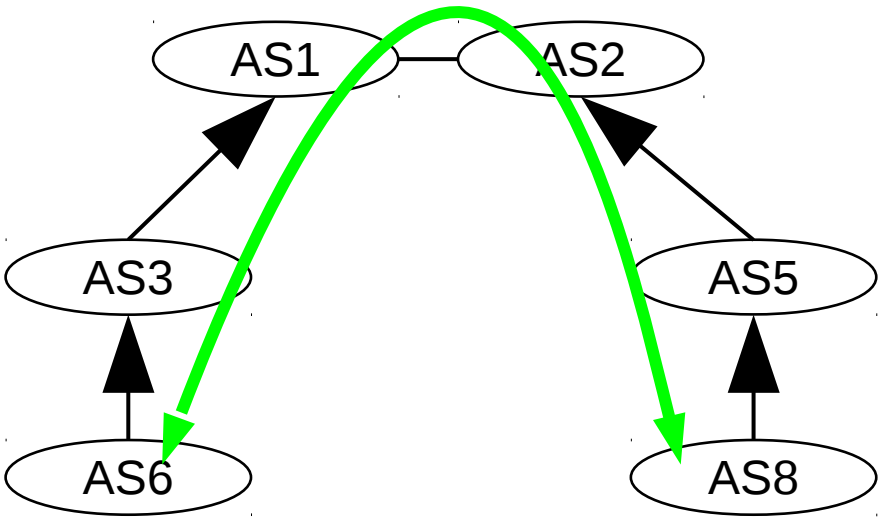
Valley-free routing

- Routing by these criteria: **valley-free routing**
- **Background:** paths conforming to the transit-peer relationships never contain “valleys” in the transit hierarchy



Valley-free routing: Rules

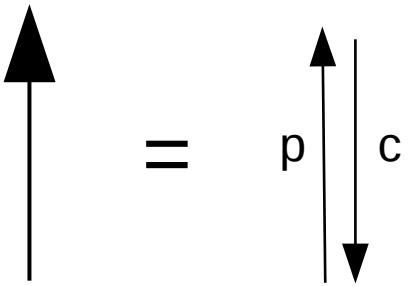
- **Definition:** a path is valley-free, if it contains
  - first, zero or more *customer → provider* links
  - then zero or one *peer ↔ peer* link
  - and finally zero or more *provider → customer* links

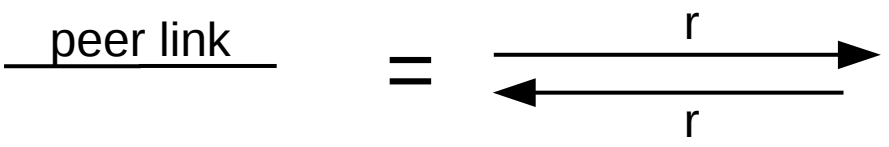


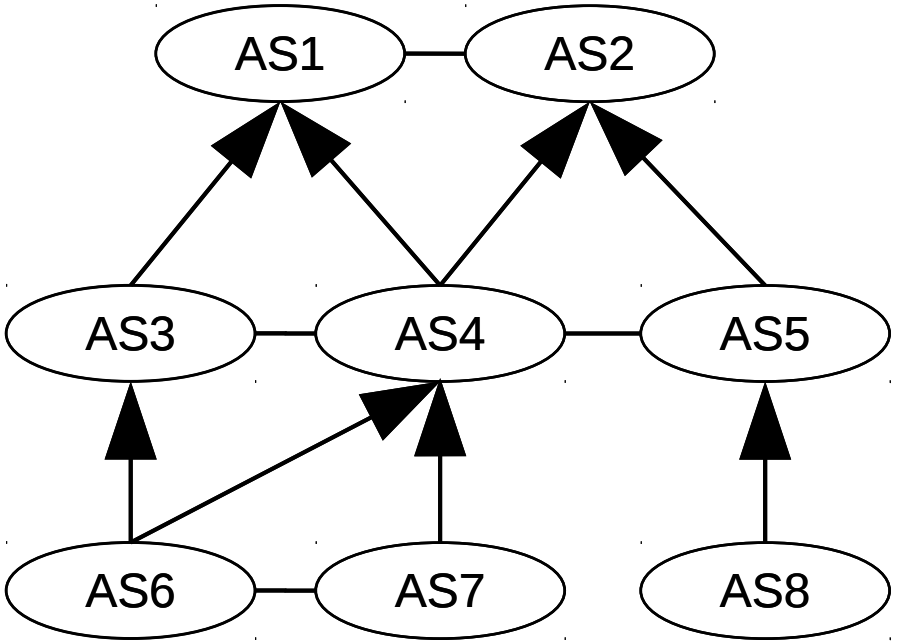
Valley-free routing: Representation

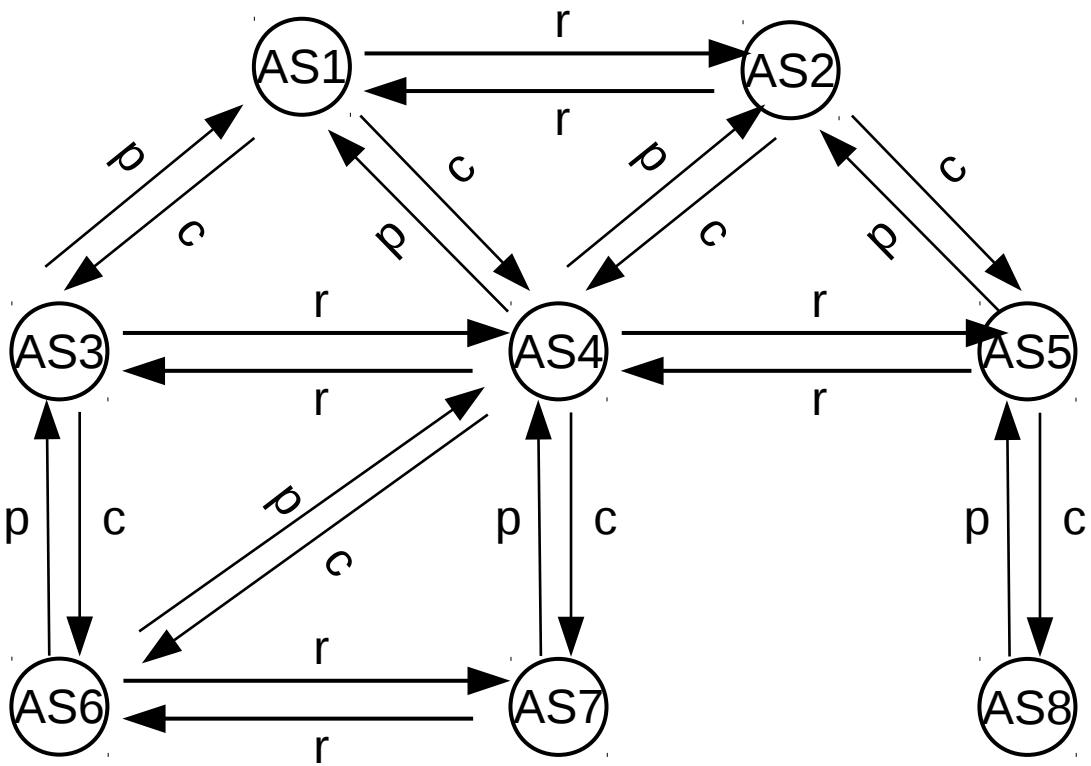
- How to check the valley-free property?
- We use a simple labeled **directed graph-model**
- The nodes of the graph are ASes and the arcs/ links are as follows:
  - transit link in the *customer→provider* direction: directed arc marked with label *p* (**p**rovider)
  - transit link in the *provider→customer* direction: directed arc marked by label *c* (**c**ustomer)
  - *peer* link in any direction: directed arc with label *r* (pee**r**)

Valley-free routing: Representation



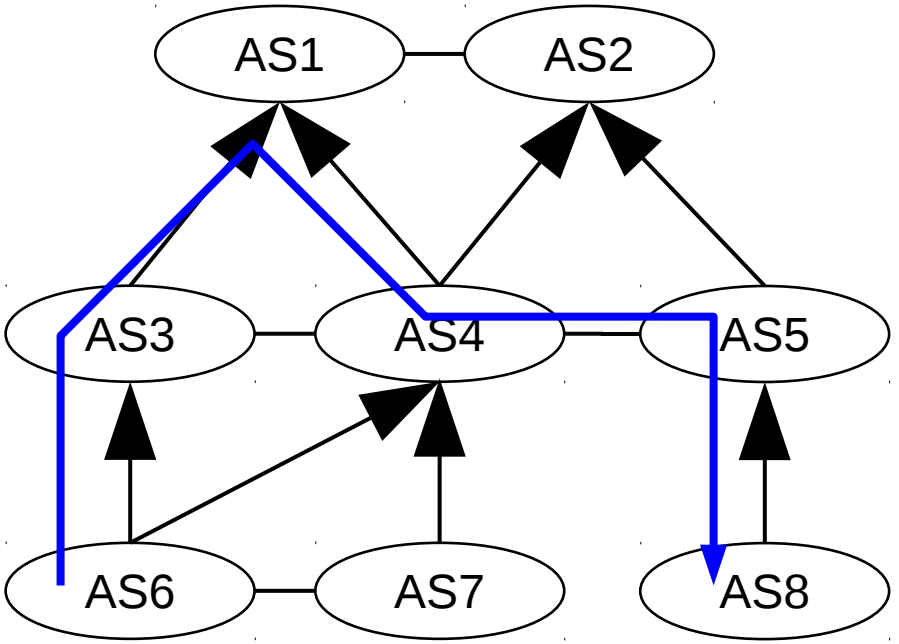






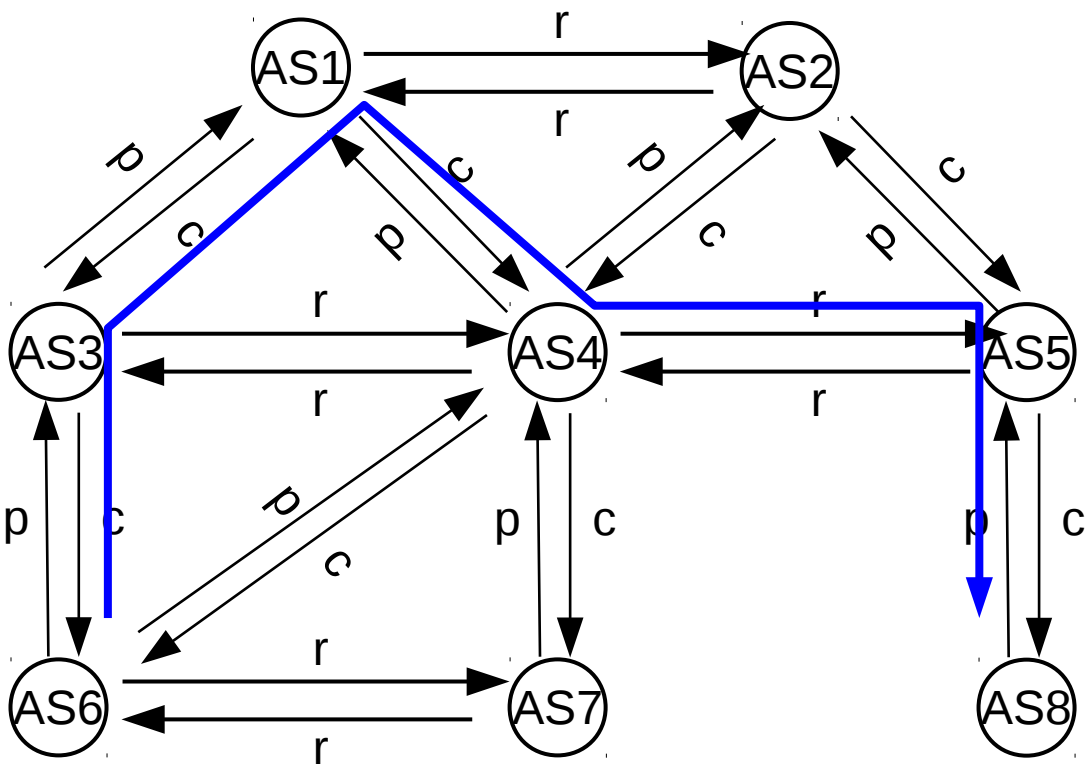
Valley-free routing: Calculation

- A graph representation can be readily used to decide whether an AS-level path is valley-free
- Is the below path feasible in valley-free routing?



Valley-free routing: Calculation

- **1) Obtain the directed graph representation**



- Note the consecutive labels of the arcs along the directed path: *p, p, c, r, c*

Valley-free routing: Calculation

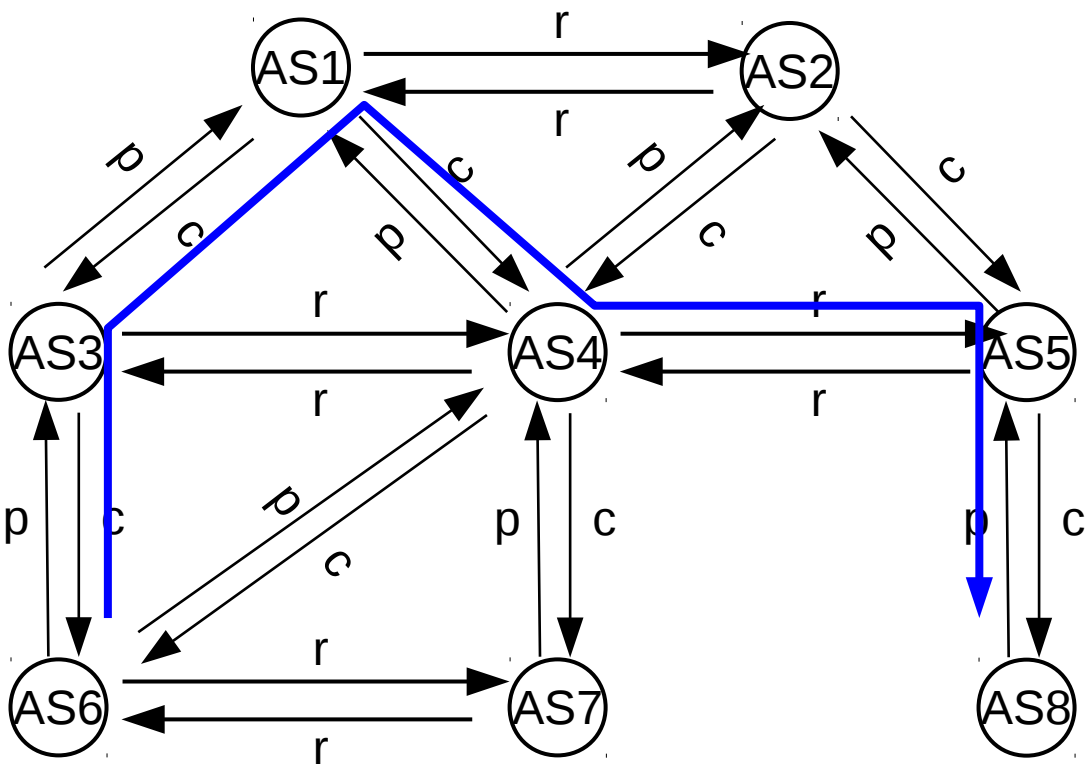
- **Theorem:** a path is forbidden if and only if it contains a two-hop *cp*, *cr*, *rp*, or *rr* subpath
- **Corollary:** a path is valley-free, if and only it the labels match the below regular expression:

*p*r?c**

- **Regular expression:** a context-free grammar to decide whether some input corresponds to a certain pattern
  - *: zero/more occurrences of the preceding label
  - *?*: zero or one occurrence of the preceding label

Valley-free routing: Example

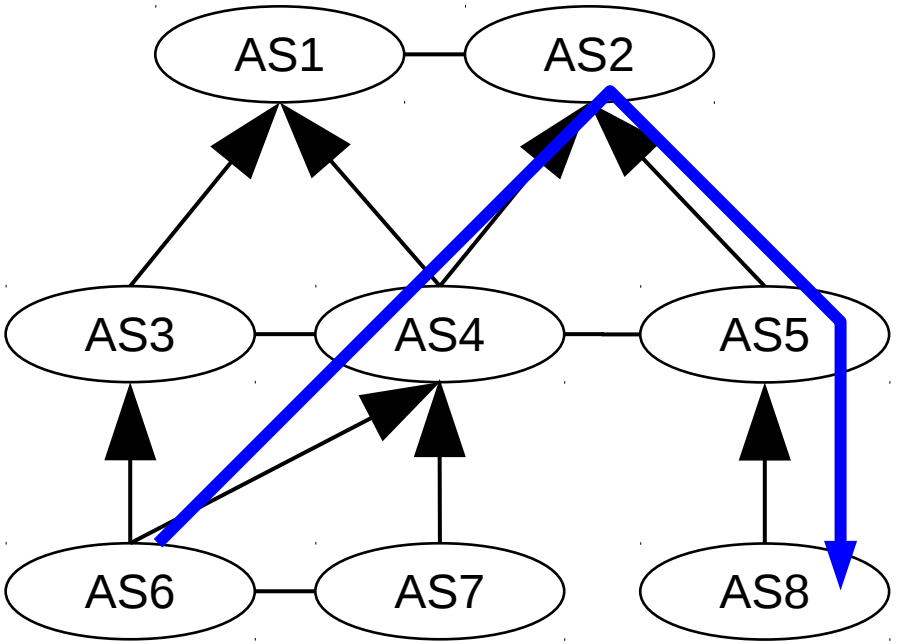
- **2) Check if labels match the regexp: *p*r?c****



- Label string "*ppcrc*" does not match the regexp: path is forbidden (there is a *cr* subpath)

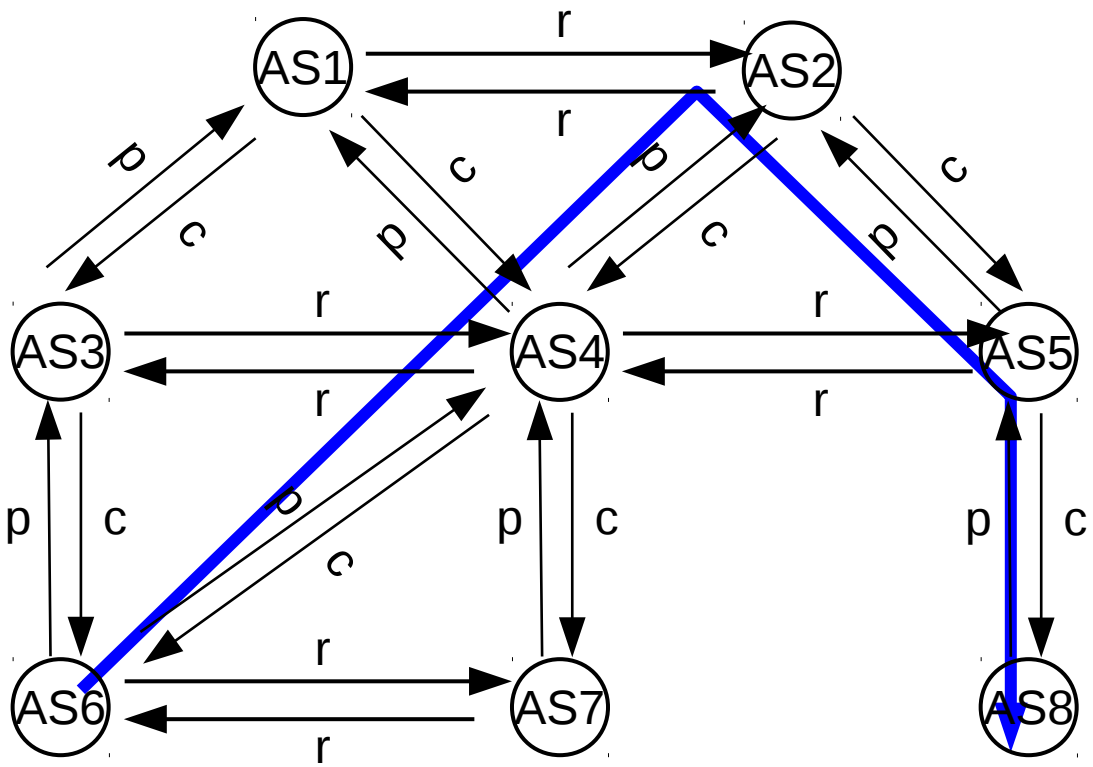
Valley-free routing: Example

- Is the below `AS6→AS8` path valley-free?



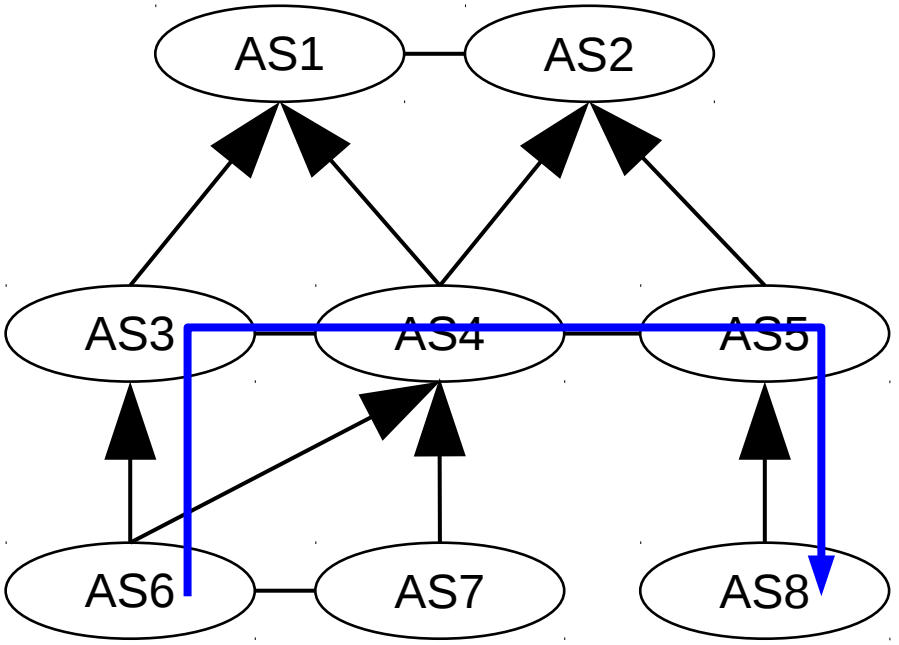
Valley-free routing: Example

- Reading out the path's labels from the graph representation: *p, p, c, c*
- Matches the regexp *p*r?c**: path is feasible



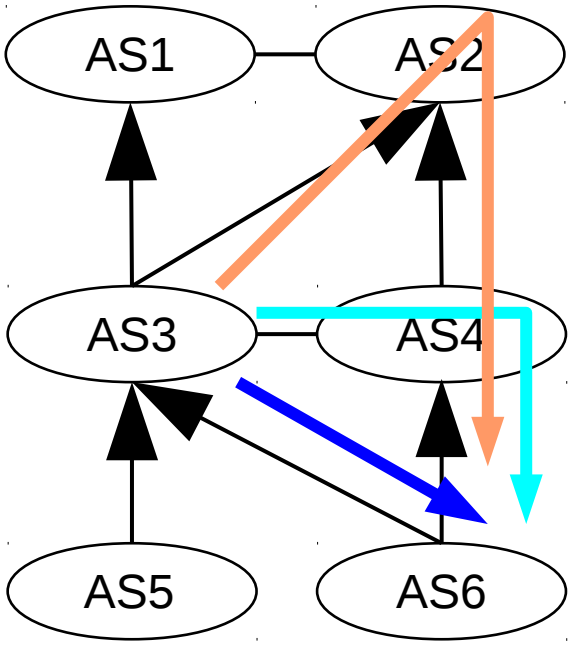
Valley-free routing: Example

- But the `AS6→AS8` path is forbidden: *p*, *r*, *r*, *c*
- Contains an *rr* subpath



Path preference

- Suppose that AS3 have three valley-free paths to a prefix in AS6:
  - via the transit provider AS2
  - another via the peer AS4
  - or directly through the direct customer link to AS6
- How to choose between these?

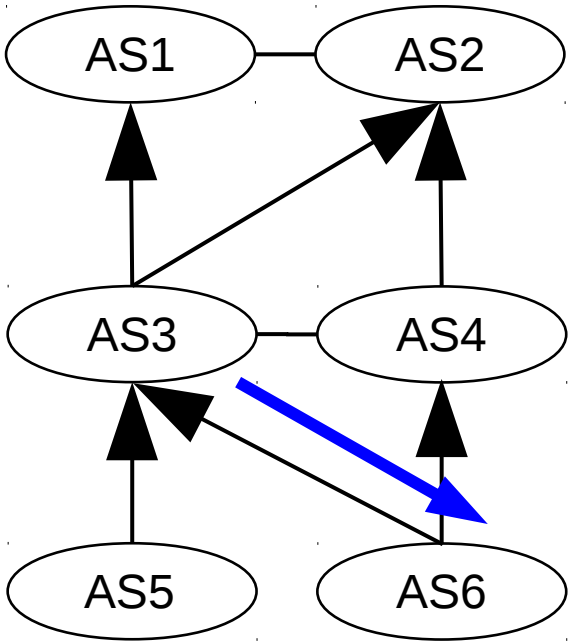


Path preference

- If traffic is **sent through the provider**: transit fee is charged for this traffic
- If the **path via the peer AS** is chosen: the traffic is most probably free of charge
  - but if we “overload” the peer link
  - the “symmetric” traffic demand requirement might be violated in the peering agreement
  - the other side might de-peer us
- If traffic is sent **through the customer**: it is guaranteed that no fees will be charged

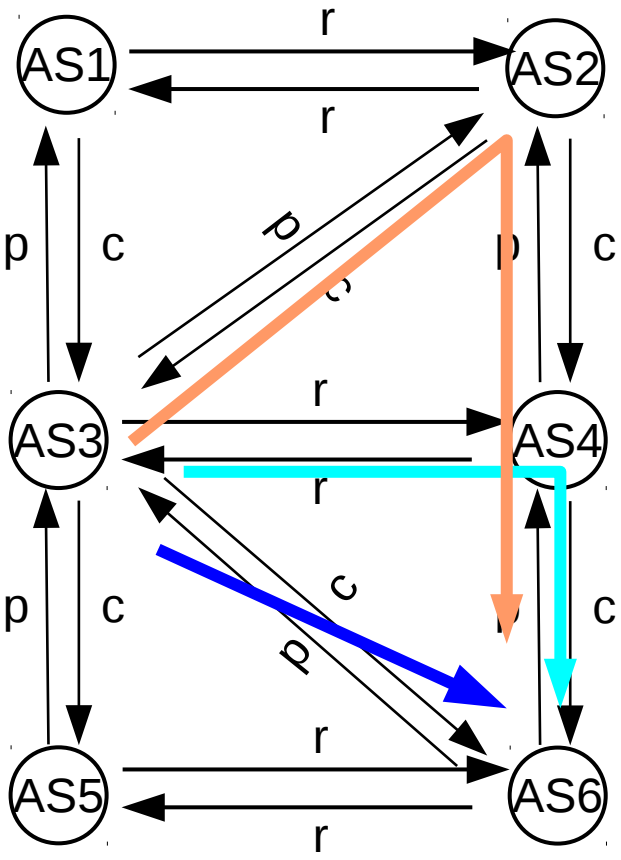
“Prefer customer” rule

- Typically, ISPs prefer paths via a customer (if available)
- Preference order:
  1. Path whose first link is a *provider→customer* link (free of charge)
  2. Path with a *peer→peer* link as the first link (probably free)
  3. Path starting with a *customer →provider* link (comes for a fee!)



“Prefer customer” rule

- In terms of the graph model:
  - path through the provider: “*pcc*”, first link is labeled *p*
  - the peer path: “*rc*”, first link: *r*
  - direct path: *c*
- **Observation:** the first link along a valley-free paths fixes its preference

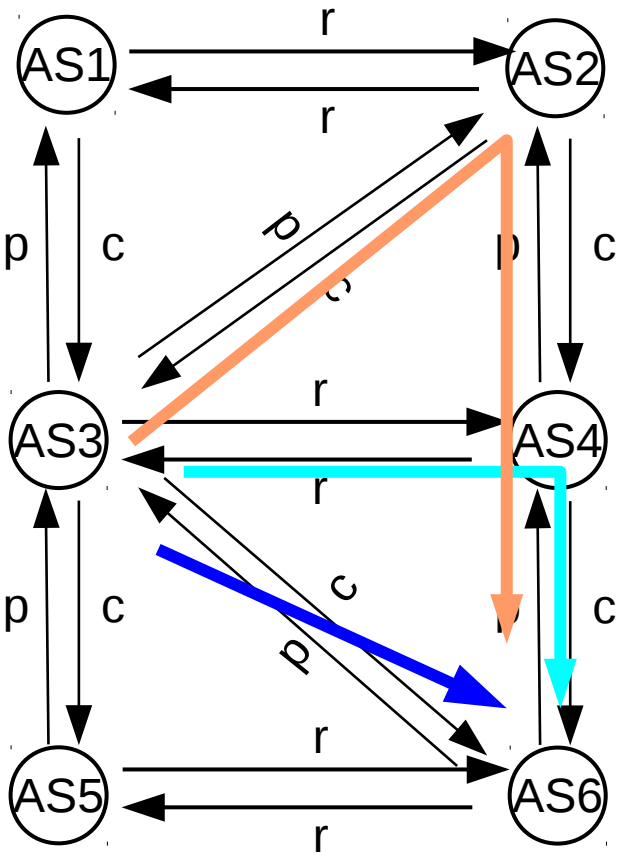


“Prefer customer” rule

- Denote with *l(P)* the label of the first link along a path *P*:
  - P_p (provider path): $l(P_p) = p$
  - P_c (customer path): $l(P_c) = c$
  - P_r (peer path): $l(P_r) = r$
- **Theorem:** the “prefer customer” rule equals the below relation:

$$P_c < P_r < P_p$$

- Smaller weighted path preferred



“Prefer customer” rule

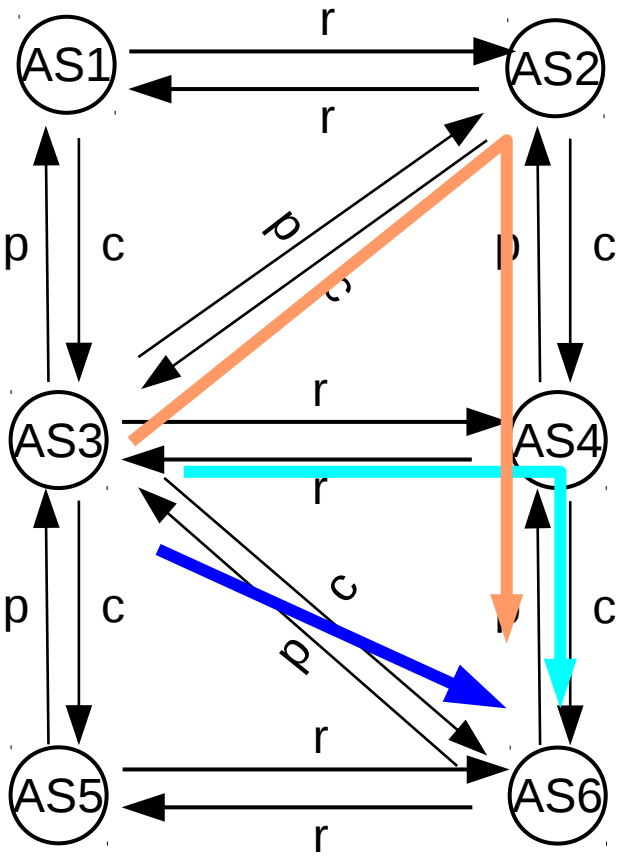
- In our specific example:

  $P_c <$ *(provider path)*

  $P_r <$ *(peer path)*

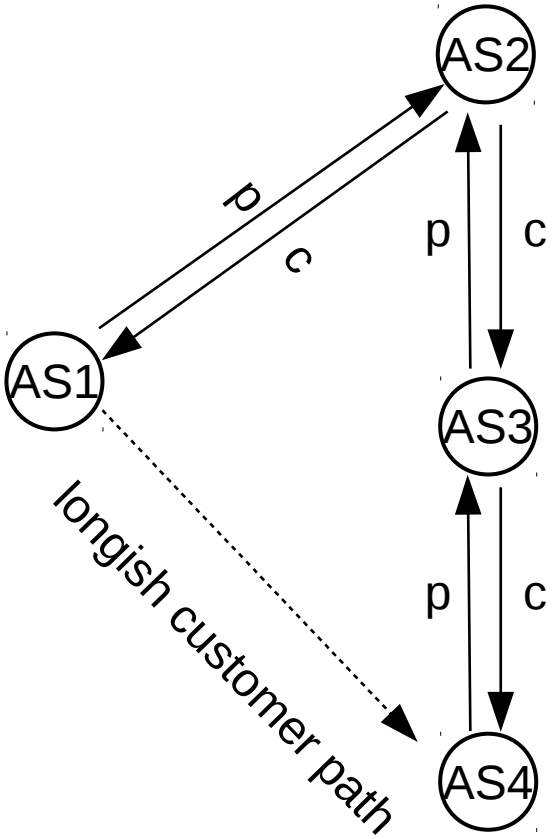
  P_p *(transit path)*

- Again, the “smaller the better”



“Prefer customer” rule

- The „prefer customer” rule does not always produce optimal paths
- If there is a longish sequence of customer links between `AS1` and `AS4`
- For smaller latency, it may be plausible to choose the transit via `AS2` instead and pay the transit fee
- Sovereign decision of an operator
- ISP policies can be really complex!



Shortest AS path

- If there are multiple, equally preferred valley-free paths available to a prefix
- Here, `AS1` can choose from a 2-AS-hop long and a “very long” customer path
- It is plausible to pick the shorter one: **shortest AS path policy**
- Order of evaluation:
  - take all **valley-free paths**
  - choose the most preferred ones in line with the **prefer-customer rule**
  - pick the **shortest** one from the resultant paths