Route or path: criteria of goodness

- hop count
- delay
- bandwidth
- loss rate
- policy
- manual configuration

Composition of performance metric:

$\longrightarrow$ quality of end-to-end path

- additive: hop count, delay
- min: bandwidth
- multiplicative: loss rate

Goodness of routing:

$\longrightarrow$ assume $N$ users or sessions

$\longrightarrow$ suppose path metric is delay

Two approaches:

- system optimal routing

  $\rightarrow$ choose paths to minimize $\frac{1}{N}\sum_{i=1}^{N} D_i$

  $\rightarrow$ good for the system as a whole

- user optimal routing

  $\rightarrow$ each user $i$ chooses path to minimize $D_i$

  $\rightarrow$ selfish route selections by each user

  $\rightarrow$ end result may not be good for system as a whole

Pros/cons:

- system optimal routing:
  - good: minimizes delay for the system as a whole
  - bad: complex and difficult to scale up
- user optimal routing:
  - good: simple
  - bad: may not make efficient use of resources
    → low utilization
    → tragedy of commons

Two pitfalls of user optimal routing:

- fluttering or ping pong effect

  $\rightarrow$ induced synchronization

- Braess paradox

  $\rightarrow$ adding more resources (extra link) can make things worse

Increasing resource should improve things but has the opposite effect

$\longrightarrow$ D. Braess (1969)

$\longrightarrow$ paradox possible due to user optimal routing

$\longrightarrow$ cannot arise in system optimal routing

Modus operandi of the Internet: user optimal routing

$\longrightarrow$ simplicity wins the day

. . . conceptually related problem in operating systems

$\longrightarrow$ page fault replacement algorithms

$\longrightarrow$ Belady's anomaly

Algorithms:

Find short, in particular, shortest paths from source to destination.

Key observation on shortest paths:

- Assume $p$ is a shortest path from $S$ to $D$

  $\rightarrow S \overset{p}{\rightsquigarrow} D$

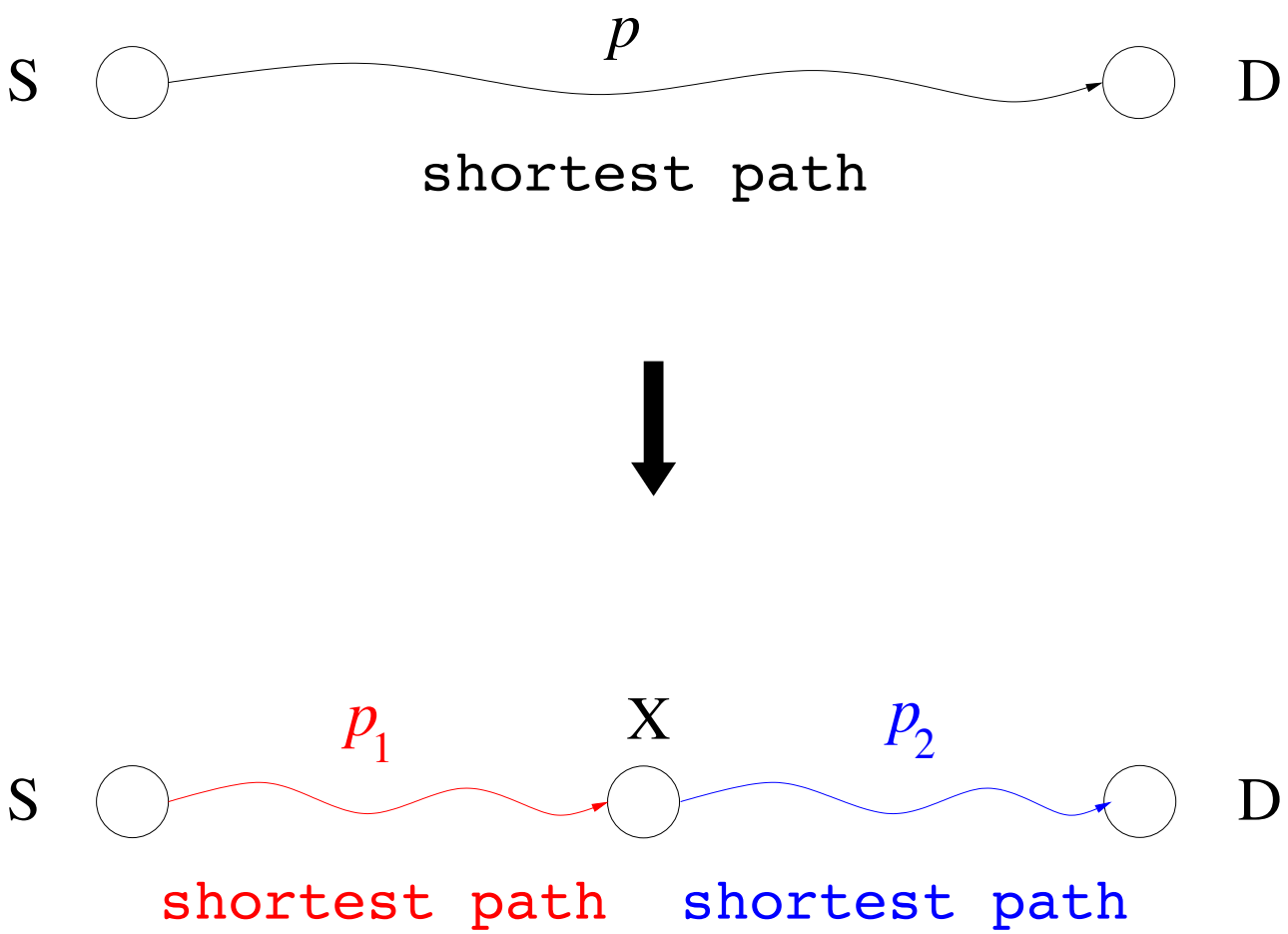
- Pick any intermediate node $X$ on the path
- Consider the two segments $p_1$ and $p_2$

  $\rightarrow S \overset{p_1}{\rightsquigarrow} X \overset{p_2}{\rightsquigarrow} D$

- The path $p_1$ from $S$ to $X$ is a shortest path, and so is the path $p_2$ from $X$ to $D$

  $\longrightarrow$ leads to Dijkstra's algorithm

Illustration:




→ suggests algorithm for finding shortest path

Dijkstra: single-source all-destination

Features:

- running time: $O(n^2)$ time complexity

  $\rightarrow$ $n$: number of nodes

- if heap is used: $O(|E| \log |V|)$

  $\rightarrow$ $O(n \log n)$ if $|E| = O(n)$

- can also be run "backwards"

  $\rightarrow$ start from destination $D$ and go to all sources

  $\rightarrow$ a variant used in inter-domain routing

  $\rightarrow$ forward version: used in intra-domain routing

- source $S$ requires global link distance knowledge

  $\rightarrow$ centralized algorithm (center: source $S$)

  $\rightarrow$ every router runs Dijkstra with itself as source

  $\rightarrow$ lots of broadcast management packets

- Internet protocol implementation
  - → OSPF (Open Shortest Path First)
  - → also called link state algorithm
  - → broadcast protocol
- builds minimum spanning tree rooted at $S$:
  - → to all destinations
  - → if select destination: called multicasting
  - → multicast group
  - → complexity including group membership management

Distributed/decentralized shortest path algorithm:

$\longrightarrow$ Bellman-Ford algorithm

Key procedure:

- Each node $X$ maintains current shortest distance to all other nodes

  $\rightarrow$ a distance vector

- Each node $X$ advertises to neighbors its current best distance estimates

  $\rightarrow$ i.e., neighbors exchange distance vectors

- Each node $X$ updates shortest paths based on neighbors' advertised information

  $$d(X,Z) \leftarrow \min\{\, d(X,Z),\ d(Y,Z) + \ell(X,Y) \,\}$$

  $\rightarrow$ same update criterion as Dijkstra's algorithm

Features:

- running time: $O(n^3)$, i.e., $O(n|E|)$

  $\rightarrow$ parallel/distributed: $O(|E|)$

- each source or router only talks to neighbors

  $\rightarrow$ local interaction

  $\rightarrow$ no need to send update if no change

  $\rightarrow$ if change, entire distance vector must be sent

- knows shortest distance but not path

  $\rightarrow$ just the next hop is known

- elegant but additional issues compared to Dijkstra's algorithm

  $\rightarrow$ e.g., stability

- Internet protocol implementation

  $\rightarrow$ RIP (Routing Information Protocol)

Data center networks: leaf-spine connectivity

→ each leaf switch connected to all spine switches

→ use equal-cost multipath routing (ECMP)

Suppose $n$ leaf switches, $m$ spine switches. At leaf switch $i \in [1, n]$:

- IP packet $p$ forwarded to spine switch $j \in [1, m]$ if $h(p) = j$ where hash function

  → input of $h$: IP source/destination addresses and port numbers

  → prevent packet reordering issue

- facilitates load balancing

  → TCP incast problem remains

QoS routing:

Given two or more performance metrics—e.g., delay and bandwidth—find path with delay less than target delay $D$ (e.g., 100 ms) and bandwidth greater than target bandwidth $B$ (e.g., 10 Mbps)

$\longrightarrow$ from shortest path to best QoS path

$\longrightarrow$ multi-dimensional QoS metric

$\longrightarrow$ other: jitter, hop count, etc.

How to find best QoS path that satisfies all requirements?

Brute-force

- enumerate all possible paths
- rank them

Is there a poly-time algorithm?

→ as of November 2025: unknown

→ specifically: QoS routing is NP-hard

→ strong belief no fast algorithm

In networking: several problems turn out to be NP-complete

→ e.g., scheduling, crypto, . . .

→ "P = NP" problem

→ one of the hardest problems in science

In practice: doesn't matter too much for QoS routing

→ no pressing demand (yet) for very good algorithm

→ intra-domain: short paths

→ inter-domain: policy routing

Policy routing:

$\longrightarrow$ meaning of "policy" is not precisely defined

$\longrightarrow$ almost anything goes

Criteria include:

- Performance

  $\rightarrow$ e.g., short paths

- Trust

  $\rightarrow$ what is "trust"?

- Economics

  $\rightarrow$ pricing

- Geo-politics, etc.

Implementation:

Internet routing protocols:

- RIP: intra-domain, Bellman-Ford
  - → also called distance vector routing
  - → metric: hop count
  - → UDP
  - → nearest neighbor advertisement
  - → popular in small intra-domain networks
  - → e.g., used at Purdue for many years
- OSPF: intra-domain, Dijkstra
  - → also called link state
  - → metric: average delay
  - → directly over IP: protocol number 89
  - → broadcasting via flooding
  - → popular in larger intra-domain networks

- IS-IS: intra-domain, Dijkstra

  $\rightarrow$ directly over link layer (e.g., Ethernet)

  $\rightarrow$ less complex than OSPF

  $\rightarrow$ popular in larger intra-domain networks

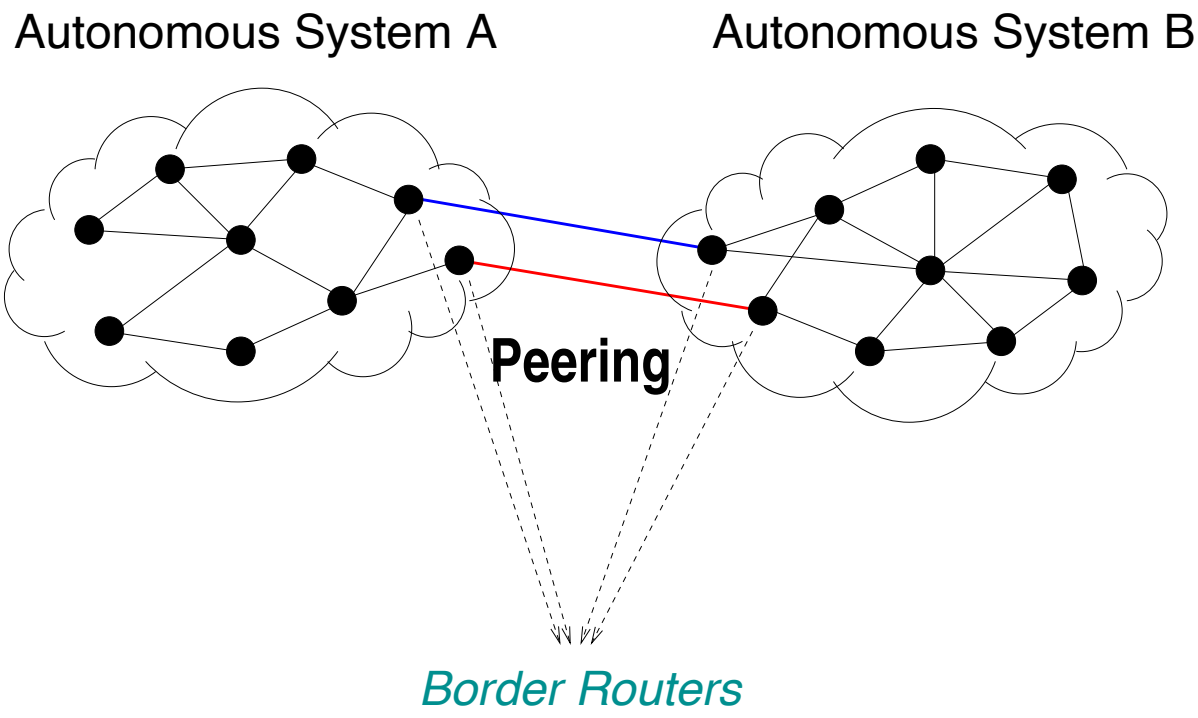
  $\longrightarrow$ intra-domain routing: performance based

iBGP (interior BGP): intra-domain protocol that bridges gap with BGP inter-domain routing

$\longrightarrow$ eBPG (exterior BGP)

BGP (Border Gateway Protocol):

$\longrightarrow$ inter-domain routing




$\rightarrow$ peering between two domains

$\rightarrow$ typical: customer-provider relationship

$\rightarrow$ in some cases: $A$ and $B$ are equals (true peers)

- CIDR addressing
  - → i.e., $a.b.c.d/x$
  - → Purdue: 128.10.0.0/16, 128.210.0.0/16, 204.52.32.0/20
  - → check at `www.iana.org` (e.g., ARIN for US)
- Metric: policy
  - → e.g., shortest-path, trust, pricing
  - → meaning of "shortest": delay, router hop, AS hop
  - → mechanism: path vector routing
  - → BPG update message

BGP route update:

$\longrightarrow$ BGP update message propagation

BGP update message format:

$$\text{ASN}A_k \rightarrow \cdots \rightarrow \text{ASN}A_2 \rightarrow \text{ASN}A_1; \text{a.b.c.d/x}$$

Meaning: ASN $A_1$ (with CIDR address a.b.c.d/x) can be reached through indicated path

$\longrightarrow$ called path vector

$\longrightarrow$ also AS-PATH

Some AS numbers:

- BBN: 1
- UUNET: 701
- Level3: 3356
- Abilene (aka "Internet2"): 11537
- AT&T: 7018
- Purdue: 17

Policy:

- if multiple AS-PATHs to target AS are known, choose one based on policy

  → e.g., shortest AS path length, cheapest, least worrisome

- advertise to neighbors target AS's reachability

  → also subject to policy

  → no obligation to advertise!

  → specifics depend on bilateral contract (SLA)

SLA (service level agreement):

  ⟶ bandwidth (e.g., 40 Gbps)

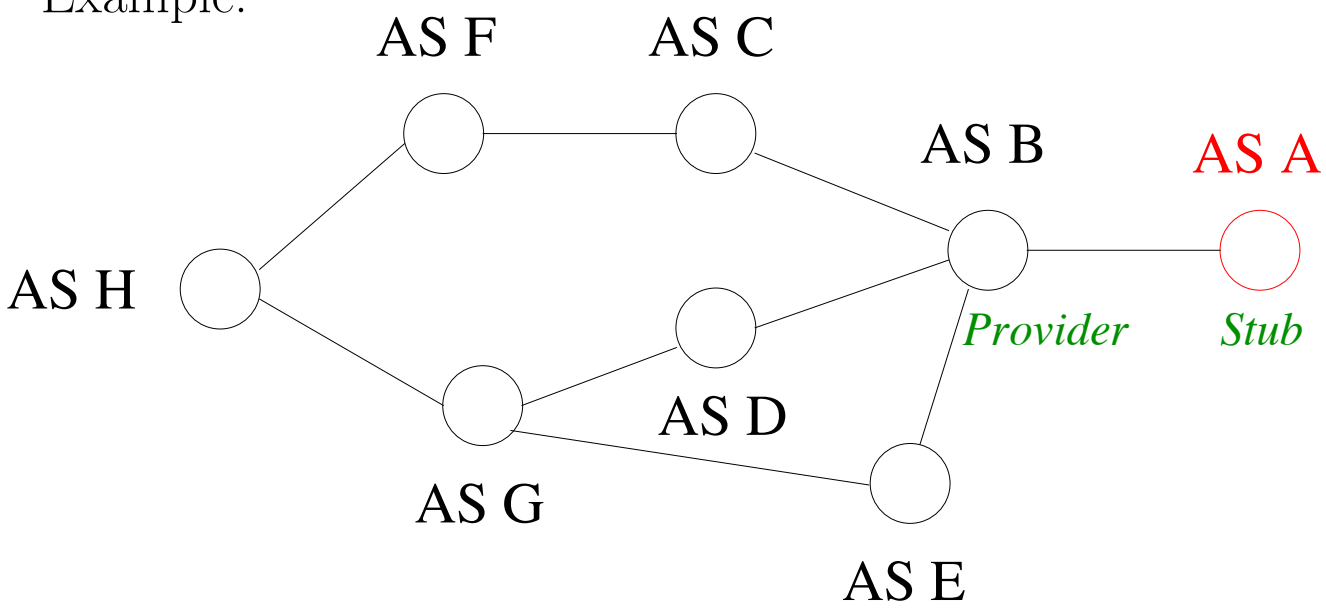
  ⟶ delay (e.g., avrg. 20ms US), loss (e.g., 0.01%)

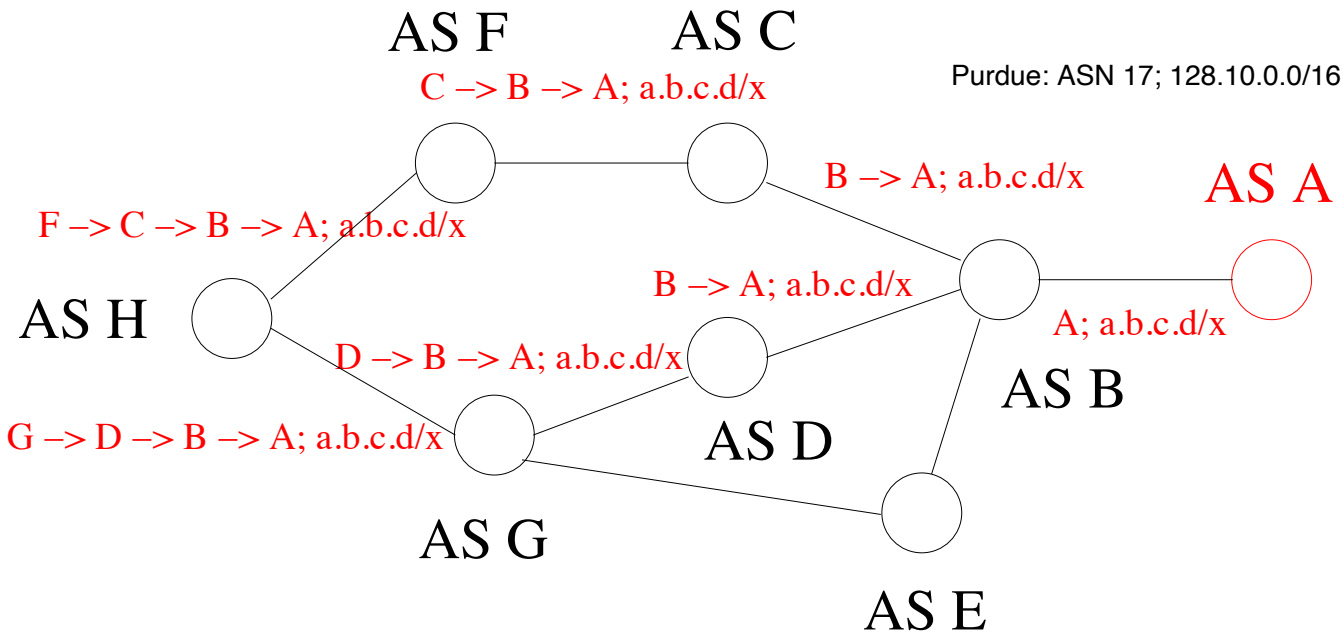
  ⟶ peak vs. average

  ⟶ pricing

  ⟶ availability, among others

Example:


AS F
AS C
AS B
AS A
AS H
Provider
Stub
AS D
AS G
AS E


AS F
AS C
C –> B –> A; a.b.c.d/x
Purdue: ASN 17; 128.10.0.0/16
B –> A; a.b.c.d/x
AS A
F –> C –> B –> A; a.b.c.d/x
B –> A; a.b.c.d/x
AS H
A; a.b.c.d/x
D –> B –> A; a.b.c.d/x
AS B
G –> D –> B –> A; a.b.c.d/x
AS D
AS G
AS E

Performance:

Route update frequency:

- ⟶ routing table stability vs. responsiveness
- ⟶ rule: not too frequently
- ⟶ 30 seconds
- ⟶ stability wins
- ⟶ hard lesson learned from the past (sub-second)
- ⟶ legacy: TTL

Other factors for route instability:

- ⟶ selfishness (e.g., fluttering)
- ⟶ BGP's vector path routing: can be unstable
- ⟶ more common: slow convergence
- ⟶ target of denial-of-service (DoS) attack

Route amplification:

- $\longrightarrow$ shortest AS path $\neq$ shortest router path
- $\longrightarrow$ e.g., may be several router hops longer
- $\longrightarrow$ AS graph vs. router graph
- $\longrightarrow$ policy: company in Denmark

Route asymmetry:

- $\longrightarrow$ routes are not symmetric
- $\longrightarrow$ mainly artifact of inter-domain policy routing
- $\longrightarrow$ various performance implications
- $\longrightarrow$ source traceback

Black holes:

$\longrightarrow$ persistent unreachable destination prefixes

$\longrightarrow$ BGP routing problems

$\longrightarrow$ further aggrevated by DNS