The Network Positioning Problem

CSC450/550

By Yueh-Hua Lee

Application

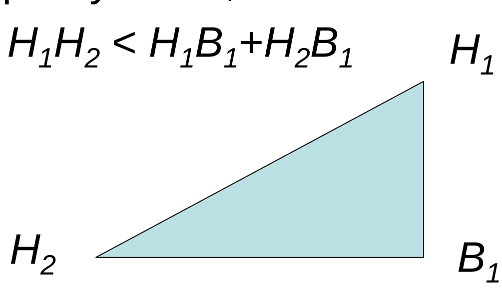
- When I use Google, the DNS server will direct me the nearest Google server (at Seattle) instead of a random chosen one (in Asia, for example).
- How do they know where exact I am?
  IP address?
  Domain Name?

Network Positioning Problem

- To predict the network distance between any two end hosts in terms of latency, bandwidth, or RTT.
- Avoid the geographical or physical relationship.

Triangulated Heuristic (Hotz)

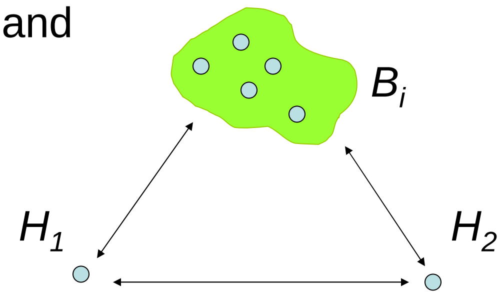
- Base node (B_1, …, B_n): well distributed infrastructure node.
- Given 2 hosts H_1 and H_2, assume the triangular inequality holds,
- $|H_1B_1-H_2B_1| < H_1H_2 < H_1B_1+H_2B_1$



Triangulated Heuristic (Hotz)

- *lower* < distance (H_1 , H_2) < *upper*.
  - *lower* = max(dist(H_1 , B_i) - dist (H_2 ,B_i))
  - upper = min(dist(H_1 , B_i) + dist (H_2 ,B_i))

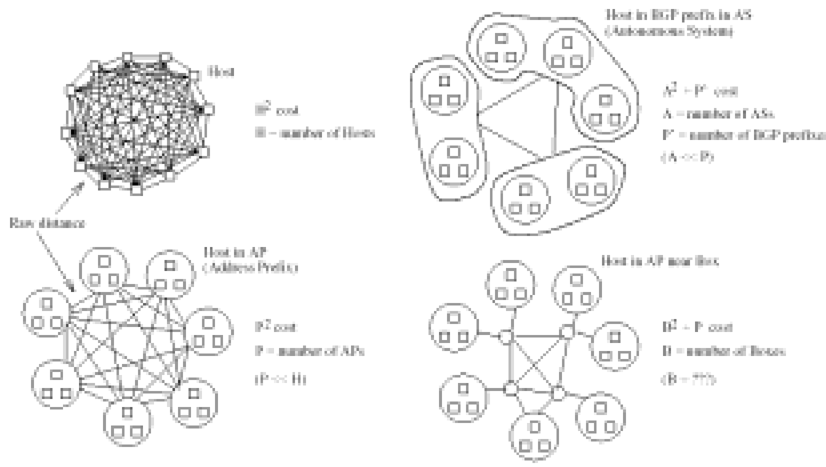
    i = 1, …, n
- Rely on efficient and correct routing



IDMaps (Francis et al)

- IDMaps: Internet Distance Map Service
- Goal: provide an estimate of the distance b.t.w. any 2 valid IP addresses on the Internet.
- Topology-based approach
- Tracker: well distributed infrastructure node.
- The distance between host A and B:
  dist (A, A's nearest Tracker T_1) +
  dist (B, B's nearest Tracker T_2) +
  dist (T_1, T_2)

IDMaps Architecture

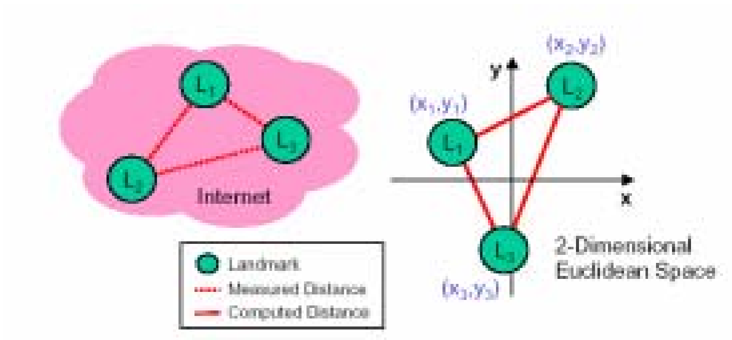


Global Network Positioning (by Eugene Ng)

- Coordinate-based approach
- Landmark: Infrastructure node (L_1, …, L_n) that is well distributed in the Internet
- GNP system measures the network distances between landmarks, d_{LiLj}, to compute the coordinates of landmarks in a geometric space, (C_{L1}, C_{L2}, …, C_{Ln}).

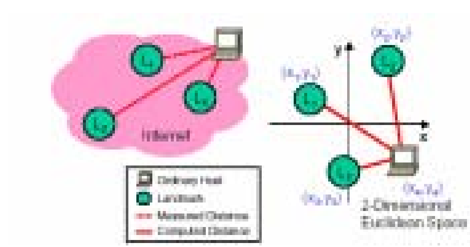
Global Network Positioning (by Eugene Ng)

- $F(C_{L1}, C_{L2}, \ldots, C_{Ln}) = \sum \varepsilon\ (d_{LiLj}, d'_{LiLj})$
- $\varepsilon = (d_{LiLj} - d'_{LiLj})^2$



Global Network Positioning (by Eugene Ng)

- Measure the distance btw host H and L_i.
- $F(C_H) = \sum \varepsilon\ (d_{HLj}, d'_{HLj})$
- The network distance is predicted by the geometric distance.



Comparison

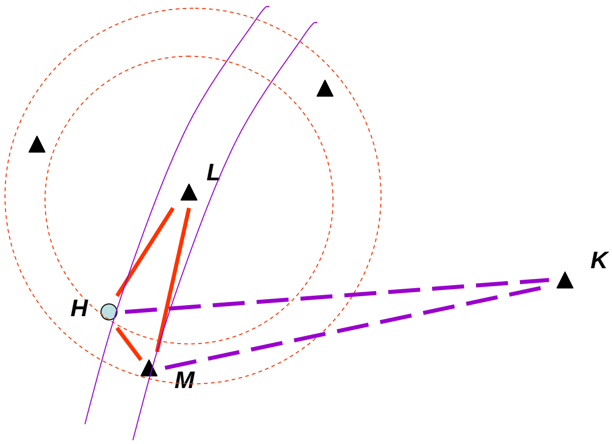
		IDMaps	Triangulated heuristic	GNP
# Paths measured		$O(N^2 + N*AP)$	$O(N*H)$	$O(N^2 + N*H)$
Communication cost	Off-line	$O(N^2 + N*AP)$ data sent to S HOPS servers	None	$O(N^2)$ data sent to one Landmark, $O(N*D)$ data sent to H hosts
	On-line (K hosts)	$O(K^2)$	$O(K*N)$	$O(K*D)$
	Server latency	Yes	No	No
Computation cost	Off-line	$O(AP*N*\log N) + O(N^3)$ at S HOPS servers	None	$O(N^2*D)$ per $t_{opt}()$ at one Landmark $O(N*D)$ per $t_{opt}()$ at H hosts
	On-line	$O(1)$ with $O(N^2 + AP)$ storage at S HOPS servers	$O(N)$	$O(D)$
Deployment	End hosts	Implement query/reply protocol	Perform measurements, exchange coordinates, and compute distances	Retrieve Landmarks' coordinates, perform measurements, compute own coordinates, exchange coordinates, and compute distances
	Infrastructure	Tracers measure all paths, send results to HOPS servers; HOPS servers implement query/reply protocol, compute distances	Base nodes reply to pings	Landmarks measure inter-Landmark paths, compute own coordinates and send them to end hosts; reply to pings
	Firewall compatibility	No	Yes	Yes

Find Nearest Landmark Server

- A new host pings all landmark server.
- For a D-dimensional space, at least D+1 distances must be measured.
- Suppose that all the landmark servers are well distributed. Given a host *H* and a landmark server *L*, the distance (*H*, *L*) and distance (*H*'s nearest server *K, L*) are roughly the same.

Find Nearest Landmark Server

|LM-LH| < HM < LM+LH



Find Nearest Landmark Server

- Measure the distances b.t.w. a host, *H*, and few randomly chosen landmark servers, *Li*.
- *Li* replies with a set of landmark servers, *Lij,* such that dist(*Li*, *Lij*) = dist (*Li*, *H*) ± r
- Find the nearest server among the intersection of all *Lij*

The End

Q & A