Towards Information-driven Networks: Research Challenges and Perspectives

Dimitri Papadimitriou
Bell Labs

IEEE HPSR 2016
June 14-17, 2016
Yokohama, Japan

Foreword

- Internet designed for two **main functions**
  - **Reachability**: destination-based packet routing
  - **Connectivity**: along logical communication channels identified by their (destination) address / network locator

  but mainly used for **information exchange/data distribution**
  - Data access remains invariably coupled to communication channel (location and identification)
  - Prevents seamless support of user/terminal and data mobility

- To better accommodate exchange/distribution function and account for information dynamics and uncertainty
  - Overlay (incl.peer-to-peer) model
  - Named-data routing model which adds
    - Information dimension to Internet functional model
    - Naming/resolution and placement/localization

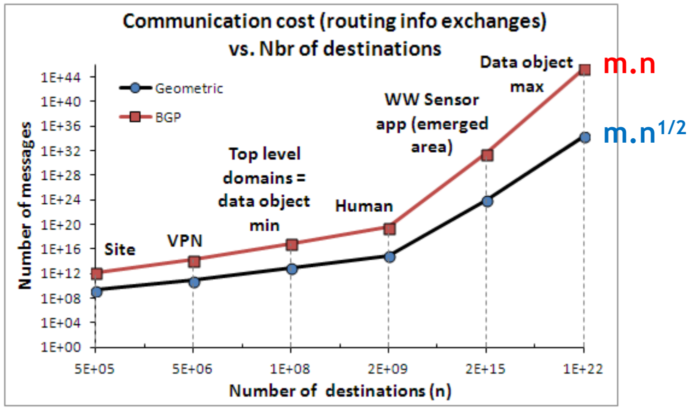
Alternatives vs. Growth

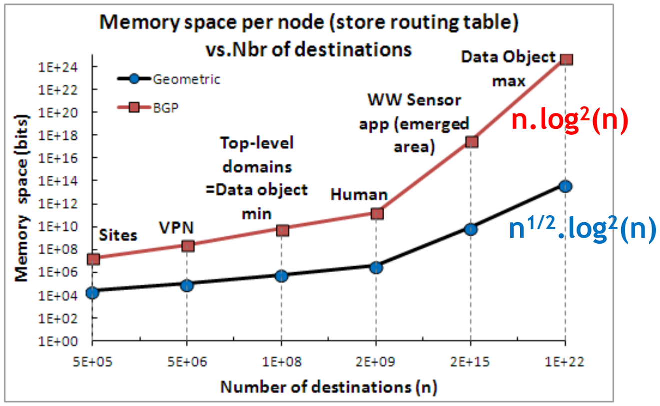
Alternatives

- Alternative 1: overlay (routing on network locators)
- Alternative 2: name-data routing (routing on names)
- *Alternative 3*: routing on data object locators

Orders of magnitude

- IPv4: BGP routing entries (Loc_RIB) ~ 500k, number of advertized AS: 50k
  - Ratio: 1 AS: 10 address prefixes
- Domain names (end 2012): 252 M domain name registrations across all TLD (Src: Verisign, Apr.2013)
- Number of content objects is very large (in between 10^{15} and 10^{22})





n = number of dest., m = number of edges

Definition: distance and metric

- Network modeled by finite undirected weighted graph G=(V,E) where
  - V, |V| = n, set of vertices/objects
  - E, |E| = m, set of edges representing relationships between objects
- Given edge set E, **distance function** = map $\mathbf{d_G}: V \times V \rightarrow \mathbb{R}+$ that is symmetric $d_G(u,v) = d_G(v,u), \forall\, u, v \in V$ and satisfies $d_G(u,u) = 0, \forall u \in V$
- Such distance function is said to be a **metric** if

  1. **triangle inequality** holds: $d_G(u,v) \leq d_G(u,w) + d_G(w,v), \forall\, u, v, w \in V$

  2. $d_G(u,v) = 0 \Leftrightarrow u = v, \forall\, u, v \in V$

Metricfull routing algorithms

- Rely on computation of distances from node where computation is performed until destination
- Most routing algorithms require distance function uniformity and consistent processing/policing on distances
- Example: distance-vector (RIP)

Metricless routing algorithms

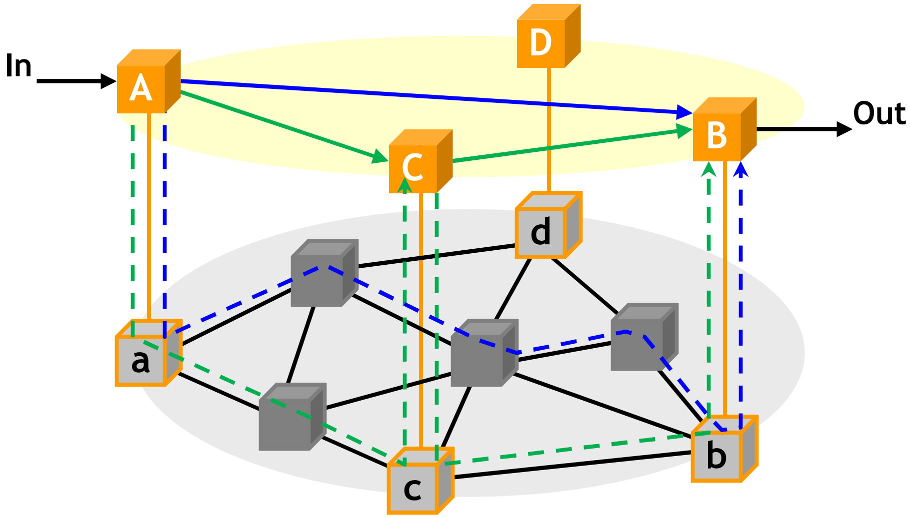
- Rely on filtering or ranking functions which defines how each node preferentially selects its routing paths
- Often operate in concert with processes/additional information preventing loop formation
- Example: path-vector (BGP)

Overlay model: main properties (1)

- Additional level of indirection between data object names and network attachment point identifiers/locators
  - Indirection realized by explicit resolution mechanism or implicit one (e.g., by extending semantic of existing locator spaces with host identification)
- Information distribution model: client-server, peer-to-peer
  - Covers wide spectrum of models ranging from Content Delivery Networks (CDN) but also multicast and mobile IP up to peer-to-peer (P2P) networks

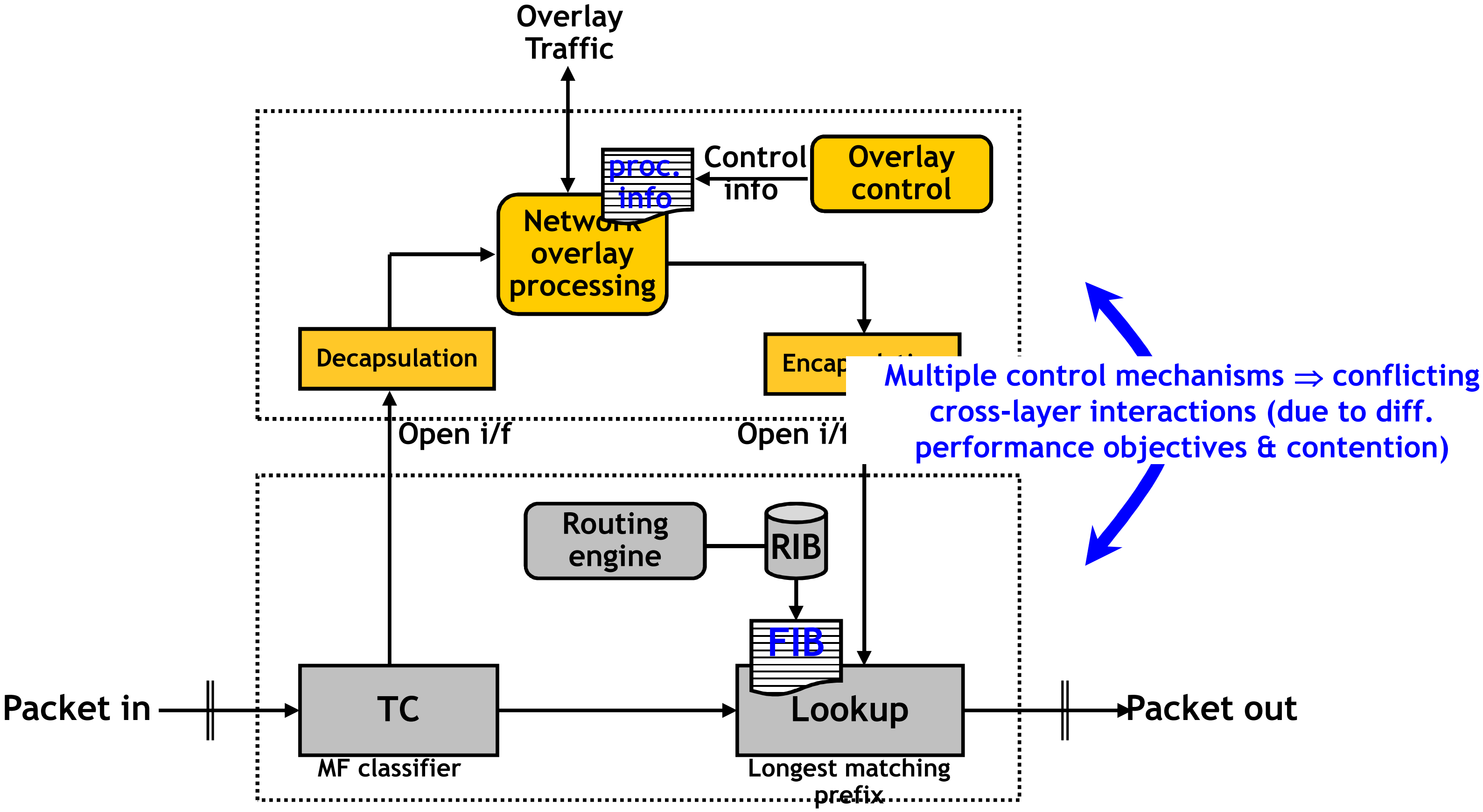
⇒ Distinction between shared infrastructure-based and host-based overlays

Infrastructure-based overlay



Overlay model: main properties (2)

- **Main disadvantage**: unnecessary inefficiency because operating-by definition- independently of any knowledge about structure, behaviour but also performance objectives of underlying network leading to **conflicting and contentious cross-layer interactions**



Multiple control mechanisms ⇒ conflicting cross-layer interactions (due to diff. performance objectives & contention)

Identifier vs. Locator

- **Locator**: identifies a location in an internetwork
  - Nodes and endpoints are assigned locators
    - A node is assigned only one locator
    - An endpoint can be assigned more than one locator
  - Locators identify "**where**" the node is positioned in/attached to network *Locators do not specify how to reach the node*
  - Value space: locator can take the form of a **topology dependent**
    - Label: flat and unstructured, structured
    - Address: structured
    - Coordinate: structure determined by the geometric space

- **Identifier**: identifies unambiguously nodes
  - Value space: identifier can take the form of **topology independent**
    - Names or simply identifiers
    - Address: structured

Routing scheme: Identifier vs. Locator

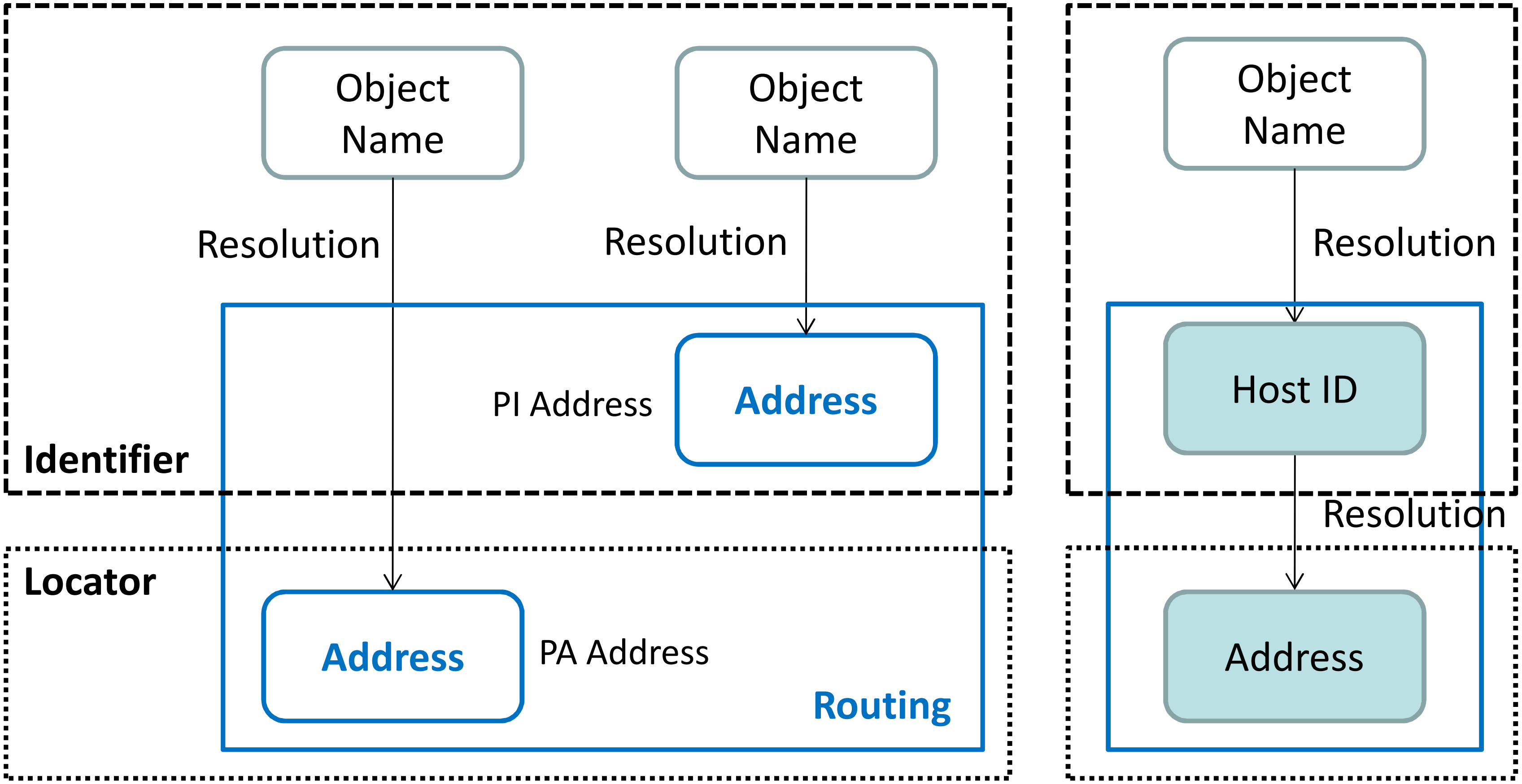
- **Label-based routing scheme**
  - Node identifiers (labels) assigned from value space which encode some topological information (thus cannot be arbitrarily selected)
  - Addressing scheme follows topology
    - Label encodes topological information useful for routing
    - Packet carries the chosen destination label in its header
    - Topology change $\Rightarrow$ Possible node label change (renaming)
- **Name-independent routing scheme**
  - Node identifiers assigned from topologically independent name space
  - Implications: addressing (naming) scheme does not follow topology and topology does not follow naming scheme
- name-independent routing (using topology-independent identifiers) ≡ identifier-to-locator resolution + label-based routing scheme performing on locators

Examples

- **IP routing** do not differentiate between IP addresses used as identifiers (Provider Independent addresses) or locators (Provider Allocated addresses)
- **Host Identity Protocol** (HIP) IP addresses function as locators, and applications use Host Identifiers to name peer hosts (instead of IP addresses)
- **Name-independent compact routing**: requires an identifier-to-locator resolution function (dictionary) distributed among nodes and performing on top of name-dependent compact routing using locators
- **Geometric routing**: coordinates are locators, and applications use Host Identifiers to name peer hosts

Overlay model: routing schemes

- Node address or locator refers to topologically informative identifier → Provider Allocated (PA) address
- Node identifier refer to topology agnostic address → Provider independent (PI) address



Overlay model: limits

- Exacerbate problems generated by **PI addresses**
  - Not topologically aggregatable: allocated independently of topology ⇒ CIDR becomes even more ineffective
  - Routing on PI addresses implications
    - Cost of additional routing entries (memory space and processing capacity) directly supported by underlying routing system rather than addresses owner
      - Example: if a single PI address prefix would be allocated to each content name domain ⇒ **number of active routing entries would increase from 5.10^5 to $2.5\ 10^8$**
    - Resulting size increase of routing tables and associated processing would worsen over time as number of domain increases also by 10-15% per year (Verisign report, April 2013)
- Consequence: increase in routers memory and processing cost ⇒ outweigh gain in capacity and transit cost

Named-data (routing) model

- Root: out of the seminal work initiated in the 70's
- Basic assumption: data objects can be named, duplicated, and reached/be accessible independently of their (spatial) location in the network, (logical) communication channel, and storage support
- Basic idea: decoupling data objects from their network location and duplicating them at multiple and heterogeneous storage entities / locations, would provide support for mobility of both information and hosts while matching message delivery delay requirements
- Example: Content-Centric Networking (CCN) where, destination network locators (IP address) specified by the source is replaced by the name of the data being requested

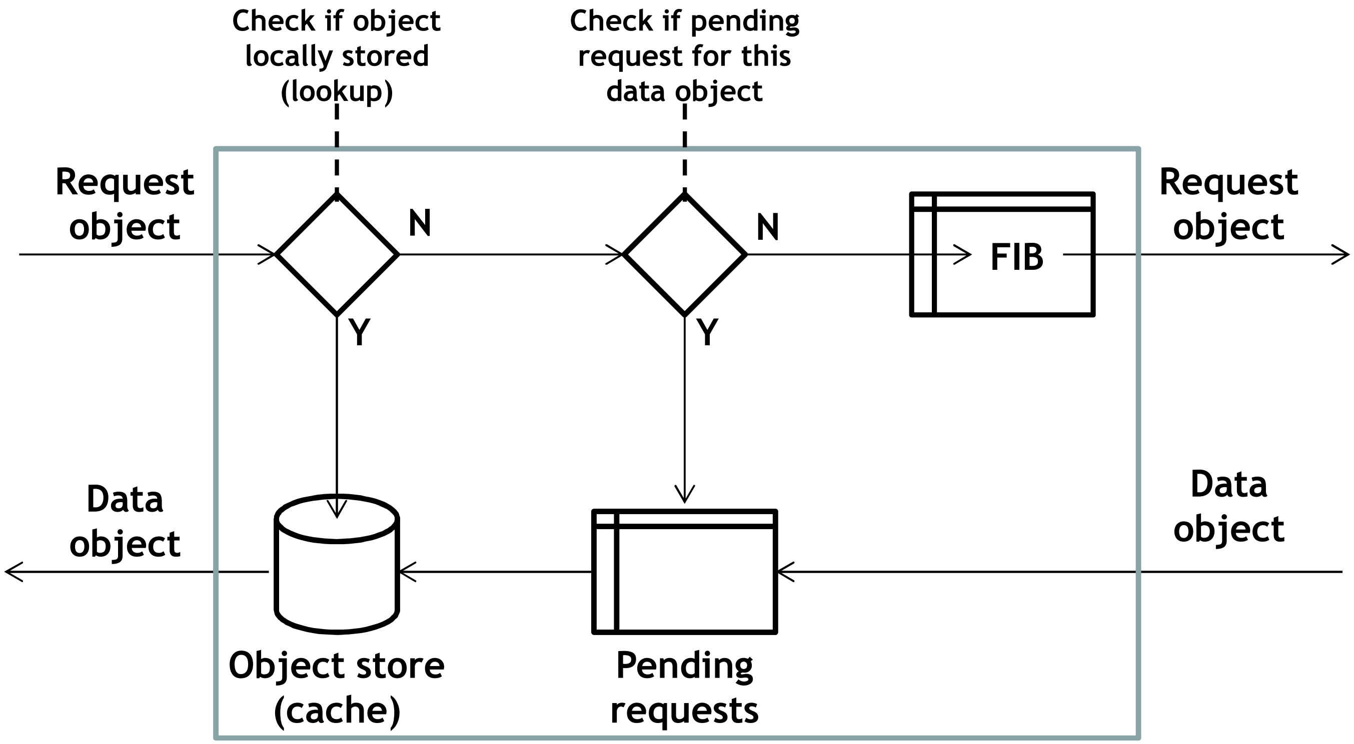
Named-data (routing) model: example

- **Content-centric networking (CCN)** and variants (altogether referred to as information-centric networking)
  - **Uniquely named-data and name-based data access** : data become independent from their network location, application, storage support but also their transport enabling to retrieve/request chunks of content by name
  - **Self-regulation of network traffic** (via flow balance which removes the need for additional congestion control techniques in the middle of a path)
  - **Replace Active Queue Management** (AQM) schemes by Least Recently Used (LRU) memory (cache) to decouple the hop-by-hop feedback control loops and to dampen oscillations
  - **Security primitives** (via signatures on all named data) are integrated into the protocol from the start
  - <u>Note</u>: lead to a completely different structure and behaviour of network stack

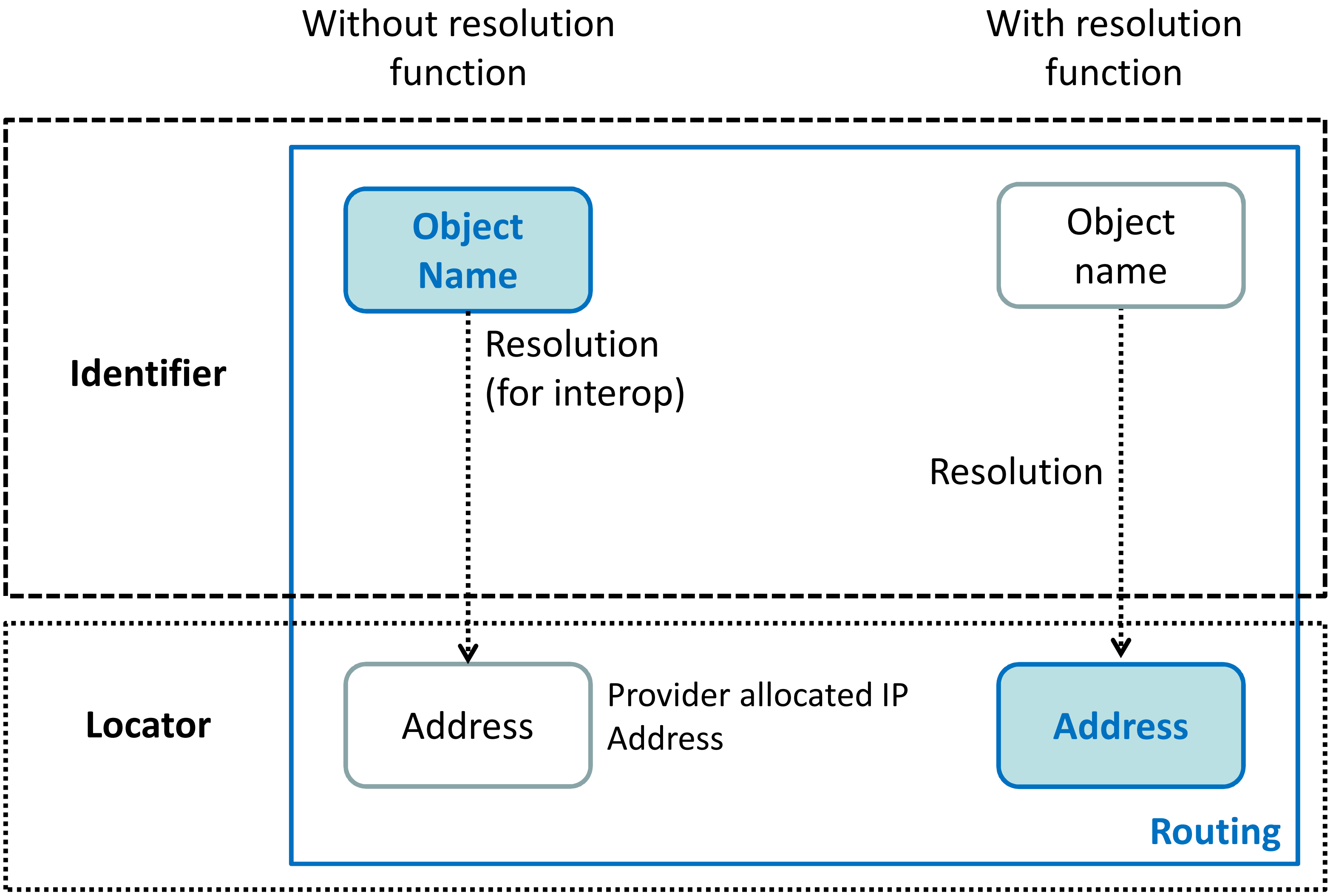
Name-data routing model: (sub-)functions

Three main sub-functions

- (optional) **name resolution**: translates name of requested data object into its network locator
- **Discovery**: routes requests based on their name
- **Delivery**: routes data object back to the requestor



Name-data routing model: resolution function



Name-data routing model: limits (1)

- Without resolution function
  - Name of data object directly used to route request towards hosting node of data object

    $\Rightarrow$ Routing information corresponding to each data object be maintained in routing table
  - Number of data objects very large (in between 10^{15} and 10^{22})

    $\Rightarrow$ Size of routing tables can be proportional to number of data objects (unless an aggregation mechanism is introduced)
    - Example: if routing tables would include one entry per top-level domain, name-data routing tables would include 2.10^{8} routes
  - Delivery function needs another identifier (ID) of either host or location to forward requested content object back to the requestor

Name-data routing model: limits (2)

- With resolution function
  - Translates name of requested data object into its locator
  - Discovery function is carried out based on the locator (that can take an IP address as value)

    ⇒ requested data object delivered to requestor based on locator
  - Delivery function similar to conventional IP routing
  - <u>Main challenge</u>: design of scalable resolution system which provides fast lookup (mapping name of data object to locators) and fast update (as location of data object expected to change frequently)
- Name-based routing approaches emphasize tradeoff
  - Alternative 1: exacerbates main drawback of push model, i.e., **storage**
  - Alternative 2: exacerbates main drawback of pull model, i.e., **latency**
  - ICNRG survey and analysis : demonstrate that all name-based routing approaches share common scaling problem

Name-data routing model: comparison

IntServ (rfc1633) - Resource Reservation Protocol (RSVP) (rfc2205)

- Local timer management (time-based soft-state)
- Memory scaling (state space) – O(n^2)
- Node-processing latency (along slow path)
- Dependent on routing algorithm (shortest-path): hyper-aggregation while exacerbating memory scaling limits of stretch-1 (local table) routing
- How applications could benefit from IntServ/RSVP

ICN aims at reconciling Web-content service networking with IntServ /RSVP

$\Rightarrow$ **Same type of problems**

- How long “pending requests” should be stored ?
- How many of them should be stored (state space) – O(2^n)
- Request – Sender (slow path) and Sender – Receiver (fast path)
- Still dependent on underlying routing algorithm: routing decisions remain decoupled from network topology and associated spatial metrics
- Which applications could take benefit of CCN (???)

Information routing-addressing challenge (1)

- **Packet networks**: routing decisions based on locators (WHERE)
  - Existing routing protocols perform on IP network locators having **no associated distance metric**

    ⇒ No associated distance computation: a router can never determine if its routing decision is distance decreasing (based on address only)
  - At end-points: with IP locators, **no selective localization** when same data object available at multiple locations
- **Content distribution networks (CDN)**: routing decisions based on Host ID (WHERE)
  - Routing table size increase from ~**5.10^5** (BGP) to **$\Omega(10^9)$** hosting domain names
- **Content-centric networks (CCN)**: routing decisions based on names (WHAT)
  - Exacerbates all the above
  - Number of addressable objects far beyond capacity of today's routing system : **6.10^5 - 10^9** $\rightarrow$ **10^{22}** inter-related data objects

Information routing-addressing challenge (2)

- **With both models**: identification (WHAT), loca(liza)tion (WHERE), and routing (HOW) refer to distinct functions associated to distinct units (names vs. address/locator vs. route) which can't be derived from each other using local knowledge
- **Locator space**: routing IS (in)directly associated to locators, otherwise flooding
  - As topology-independent as possible (necessary condition to dissociate communication channel/container identification from the content identification and renumbering)
  - Provide sufficient and timely information to compute distances where this information is processed;
- **Level of information units** at which routing decision is performed
  - Higher level (names)$\Rightarrow$ higher memory space (size and number of routing entries)
  - Lower level (locators) by providing dictionary + resolution processes $\Rightarrow$ increase communication cost and memory space (push) or latency (pull)

Network modes

	Sarnoff (Broadcast)	**Metcalfe (Ethernet) Baran (IP)**	**Information- oriented**
Spatio-temporal distribution of information	density u = cte (≡ centralized and static)	density u = u(x) (≡ distributed and *deterministic*)	Prob. density u = u(x,t) (≡ dynamic & stochastic)
Pattern	star, hub and spoke, concentric	Mesh (nodes = GTW between broadcast domains)	Complex network (nodes = GTW between information domains)
Scale (value)	n	n^2	2^n
Channel	**Physical (Optical)**	**Logical (TCP/IP)**	Data
Metric(s)	Spatio-temporal	Spatio-temporal	+Semantic-Structural
Deployment	Coordinated	Organic	not yet deployed
Example	CDN, cloud, etc.	Computer networks, web, mail	Communities

= cloud model = Internet model

Localization vs. Routing function

Localization function

- Selects locator obtained by means of *resolution system* mapping object names to their associated locator(s)
- Operates *at each node* (in particular at end-points)
- Often customized on per routing scheme basis

Distributed routing

- *Function* ***f*** $\overset{\text{def}}{=}$ $\forall$ u $\in$ V(G), $\forall$ for v $\in$ V(G) \ {u} determines locally and independently of other nodes v $\in$ V(G) \ {u} the adjacent node w ((u,w) $\in$ E(G)) along a loop-free path p(u,w,…,v) from u to v such that incoming messages directed to v can reach their destination
- **Performs on locators** : name-independent routing $\equiv$ identifier-to-locator resolution + name-dependent routing performing on locators

$\Rightarrow$ **Locator space** that can be processed at end-points by localization function and at intermediate nodes by routing function

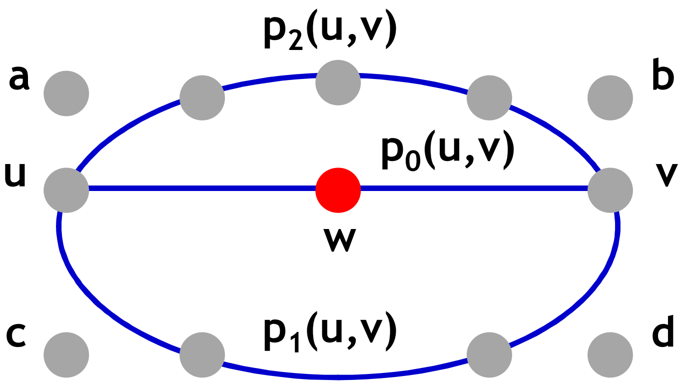
Shortest path routing $\Rightarrow$ Hyper-aggregation

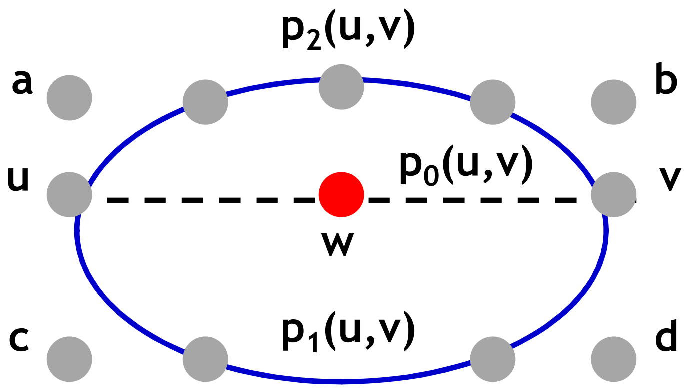
Traffic engineering

- Select among subset of paths that connect u,v proportionally to their **length** $\overset{\text{def}}{=}$ number of edges path p(u,v) traverses from node u to v
- Ranking at node u for path selection: $p_0(u,v) > p_1(u,v) > p_2(u,v)$

Exploiting geometric properties

- Length of given path p(u,v) $\overset{\text{def}}{=}$ sum of edge weights the path p(u,v) traverses from node u to v
  - $\forall$ edge $(i,j) \in E(G)$, weight $\overset{\text{def}}{=}$ length of segment [i,j]
- Ranking at node u for path selection: $p_2(u,v) \approx p_1(u,v) > p_0(u,v)$

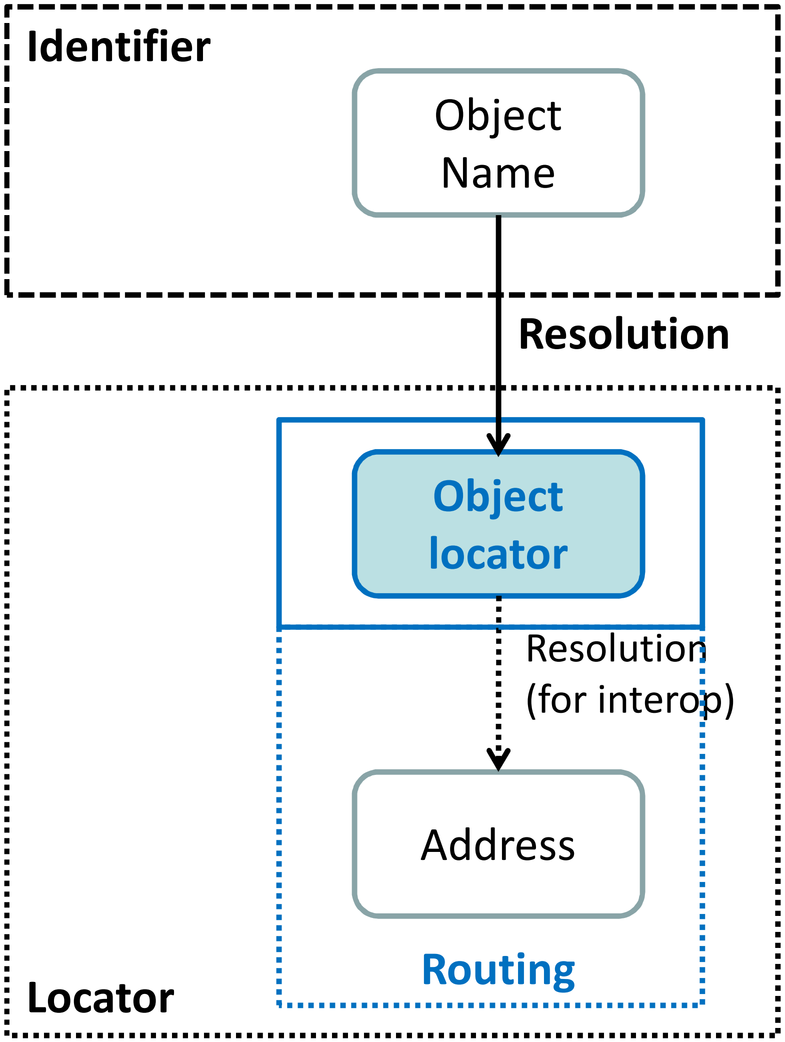


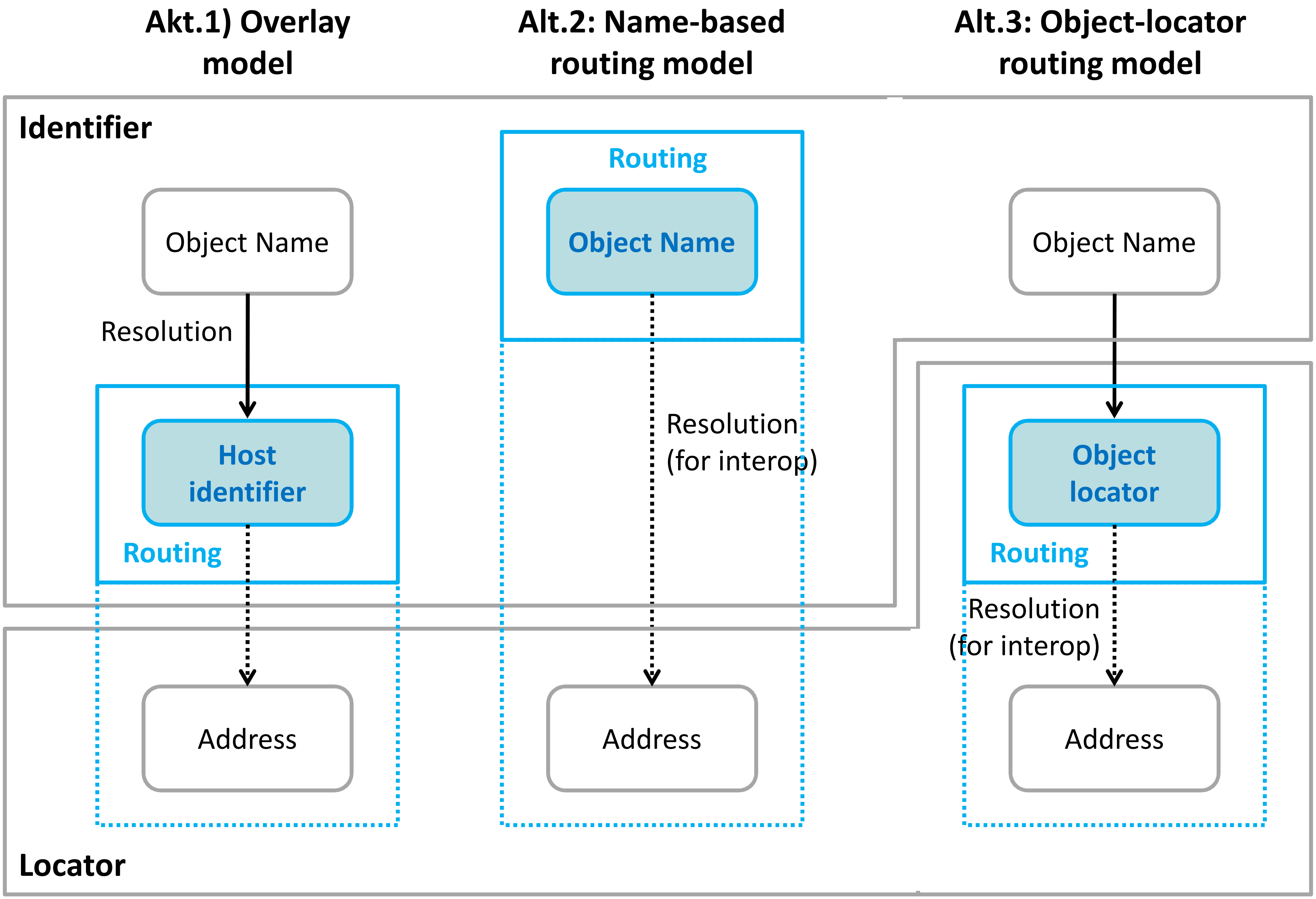


Routing on data object locators

Principles

- Assign locators to data objects (being an addressable information unit)
- Perform information routing decision on locators avoiding name-to-locator resolution by intermediate nodes
- Combine use of data object locators with dynamic storage on intermediate routers




Akt.1) Overlay model
Alt.2: Name-based routing model
Alt.3: Object-locator routing model
Identifier
Object Name
Resolution
Host identifier
Routing
Address
Routing
Object Name
Resolution (for interop)
Address
Object Name
Object locator
Routing
Resolution (for interop)
Address
Locator

Which locator space ?

- **Topology dependent labels** : renumbering even in case of non-local topological change ⇒ not good choice

- **IP addressing**
  - No associated distance metric
  - No distance computation and selective localization when same data object available at multiple locations

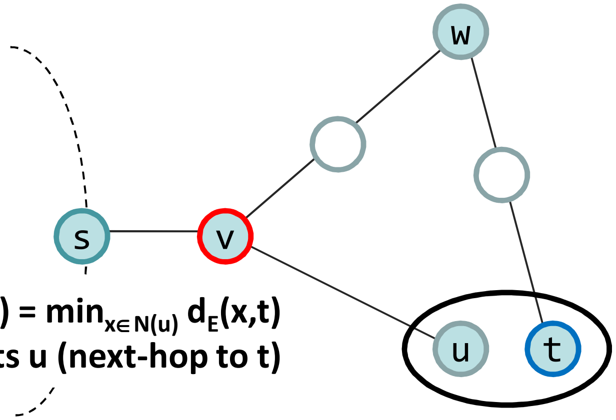
- **Geometric space**
  - **Coordinates** (≡ locator value space) assigned independently of nodes interconnection (to prevent renumbering)
  - Enabling path length computation (from source to dest and vice-versa)
  - Lead to routing capable to overcome memory space complexity (O(n.(log(n))) characterizing stretch-1 routing such BGP

Which geometric space ?

Euclidean space ($\mathbb{E}^n$)

- Two dimensions ($\mathbb{E}^2$) ⇒ Local minima
- dim(V) ~ *O*(log *n*): too high-dimensional for many applications

Distance decreasing routing function ?



$d_E(u,t) = \min_{x \in N(u)} d_E(x,t)$

→ v selects u (next-hop to t)

Hyperbolic space ($\mathbb{H}^n$)

- Two dimensions ($\mathbb{H}^2$) sufficient for any connected graph [Kleinberg Theorem-2007]
- Vivaldi-modified algorithm to compute coordinates (hyperboloid model) or by means of (exact) greedy embeddings

Locator and Metric space

Locator space → **Space: $X = \mathbb{H}^2$**

- Associate locator $x \in X$ to data object
- Each locator $x \in X$ represented by its globally unique **coordinates c_x** ($\equiv$ label)

Associated **metric ($d=d_H$) → Metric space ($X=\mathbb{H}^2, d_H$)**

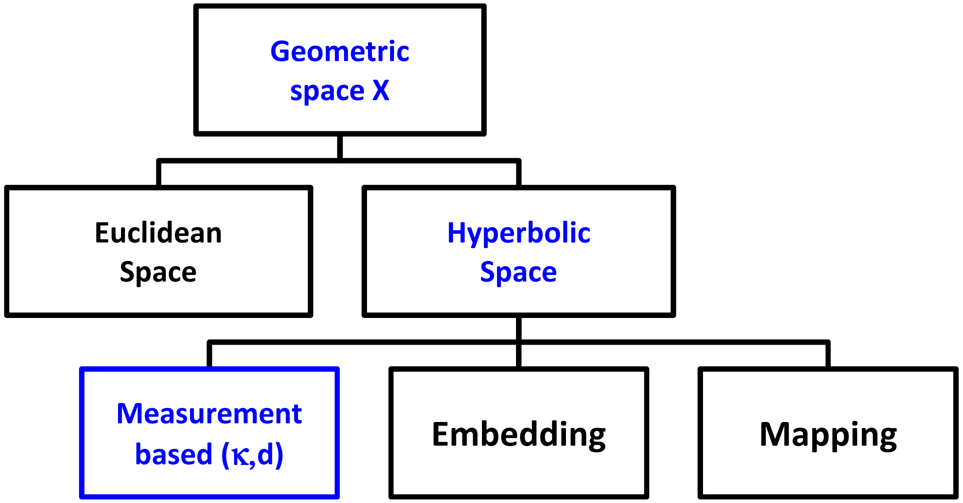
- Knowing locator (coordinates) of destination y, source x can determine distance $d_H(x,y)$ without additional input
- Reverse relationship holds since $d_H(y,x) = d_H(x,y)$
- **Locator space** that can be processed at end-points by localization function and at intermediate nodes by routing function

Identifier: locator relationship: **M:N**

- A given data object can be assigned to multiple locators (can be retrieved from multiple locations)
- A given locator (i.e., a given location) can host multiple names

Geometric routing: overview

- **Geometric routing**: assign to each node coordinates taken from metric space (X,d) that are used as locators to perform point-to-point (distance decreasing) routing



Geometric space X

Euclidean Space

Hyperbolic Space

Measurement based (κ,d)

Embedding

Mapping

Euclidean space: dim(V) ~ *O*(log *n*): too high-dimensional for many applications

Hyperbolic space: 2-dim are sufficient

Embedding (exact): requires construction of global structure

Measurement-based: organic (decentralized, peering basis)

- **Principle**: builds a set of local routing entries whose total memory space is proportional to the degree of each node/neighborhood
  - Note: excludes memory space mobilized for storing results of intermediate operations for coordinate assignment

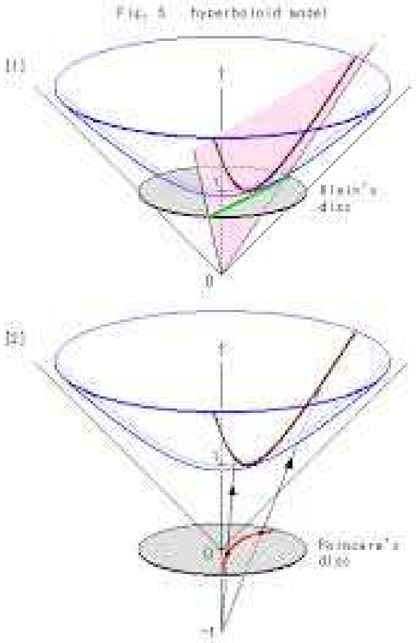
Geometric routing: main functions

- **Coordinate computation**: assign coordinates c_x to each node $x \in (X,d)$
  - Note: most critical part as determines stretch, computational complexity and communication cost
- **Localization** function: locator selection
- **Routing function**: coordinates are used as locators to perform point-to-point routing by selecting the neighbor that is closest to the destination
  - Assuming each node u of V(G) knows its own position (coordinate) and position of neighbors N(u)
  - Distance $\mathbf{d : X} \times \mathbf{X} \rightarrow \mathbb{R}^+$ only information necessary for local computation

    For each dest. $t \in V$, node u routes incoming messages (directed to destination t) to its neighbor $v \in N(u)$ if $\mathbf{d(c_v,c_t)) = min_{x \in N(u)}\{d(c_x,c_t)\}}$
  - When $d(c_u,c_t) > d(c_v,c_t))$ at each node along routing path from source s to dest. t, distance d decreases monotonically

Coordinates computation

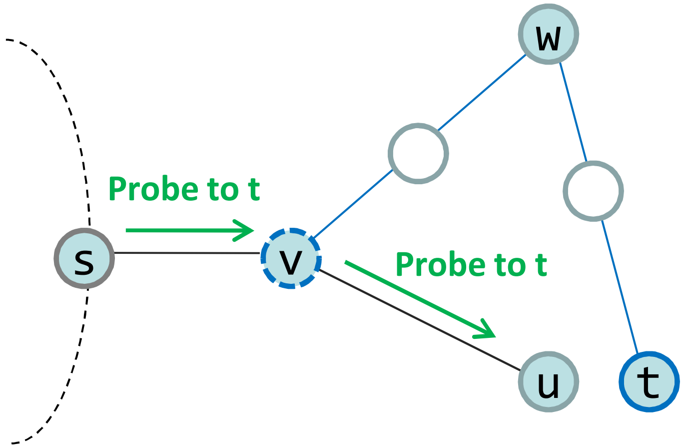
- Hyperboloid model: distance between two points computed along a line formed by the intersection of the hyperboloid with the plane determined by the two points and the origin of the space



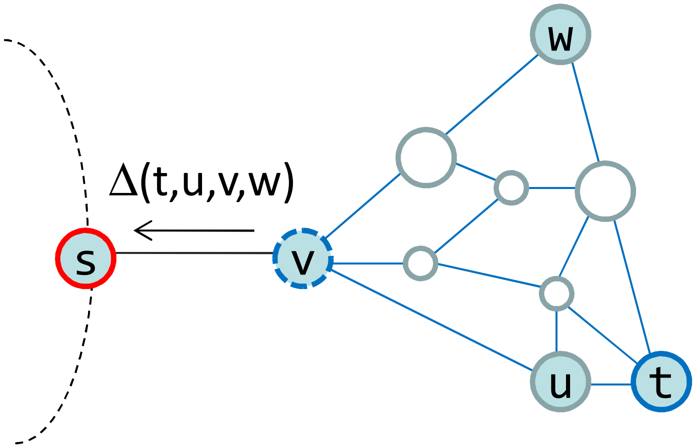
- Distance between points x=(x_1, x_2) and y = (y_1, y_2) in 2-dimensional unit hyperboloid of curvature κ :

$$d(x,y) = \operatorname{arccos}h\left(\sqrt{\left(1+\sum_{i=1}^{2} x_i^2\right)+\left(1+\sum_{i=1}^{2} y_i^2\right)} - \sum_{i=1}^{2} x_i y_i\right)\kappa$$

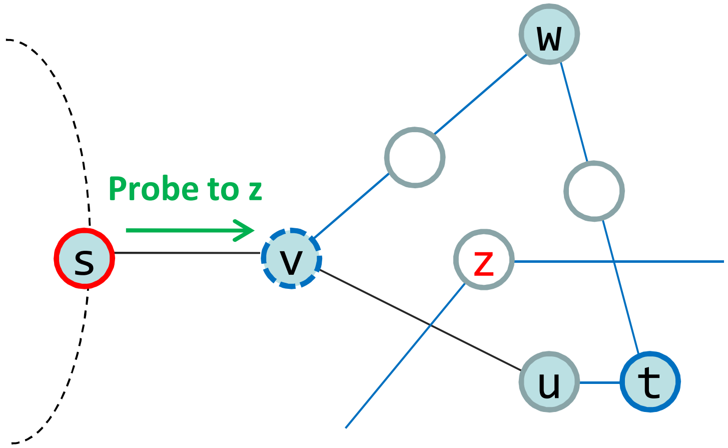
- Vivaldi-like algorithm of similar computational complexity but than Vivaldi algorithm for Euclidean coordinates



Neighbor's reachability discovery: vertex s,v proactive knowledge about vertex t reachability (probing)



Vertex v can describe partitions $\Delta=\Delta(t,u,v,w)$ of metric space (X,d)

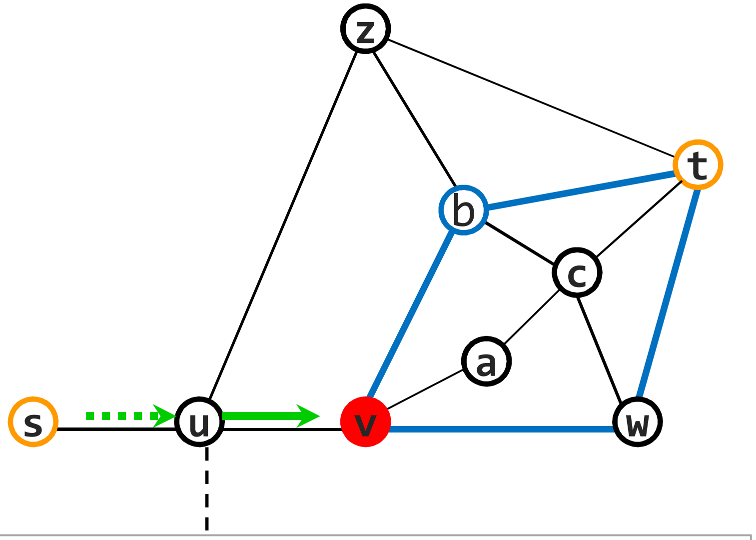


Probing enables detection of unreachable vertices (z) in partitions (pathological cases)

Vertex v can describe partitions Δ of the metric space (X,d) with exclusion set {z}

Geometric routing: principle

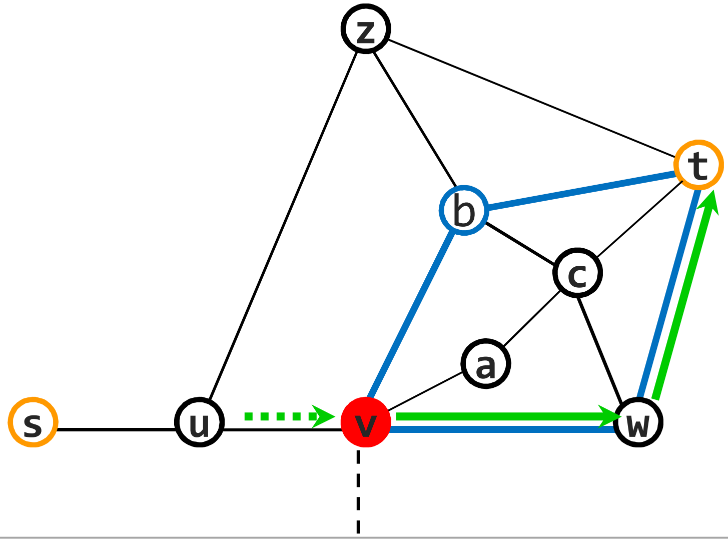
- Labeling nodes with discoverable coordinates from metric space ($X=\mathbb{H}^2, d=d_{HYP}$)
- Label space aggregation leads to routing tables with less memory consumption while keeping stretch deterioration limited: routing to dest. outside (local) partitions and table-routing to dest. in local partition



Routing decision:

Neighbor v | $d_{HYP}(v,t) = \min_{x \in N(u)}\{d_{HYP}(x,t)\}$

instead of z | $d_{GRAPH}(z,t) < d_{GRAPH}(v,t)$

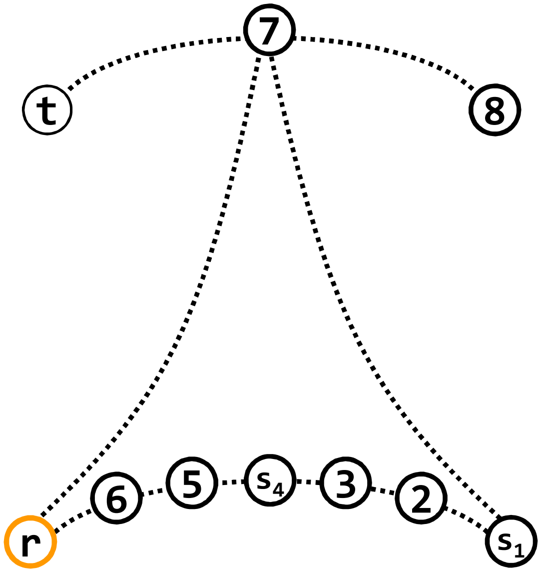


Routing decision:

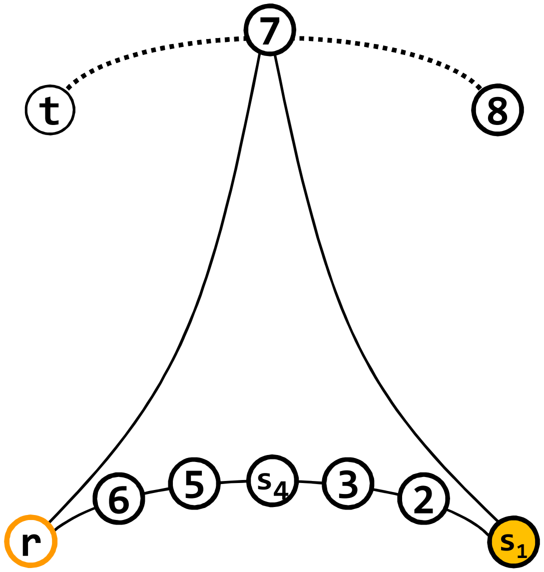
Neighbor w | $d_{GRAPH}(a,w) = \min_{x \in N(u)}\{d_{GRAPH}(x,t)\}$

Example (1)

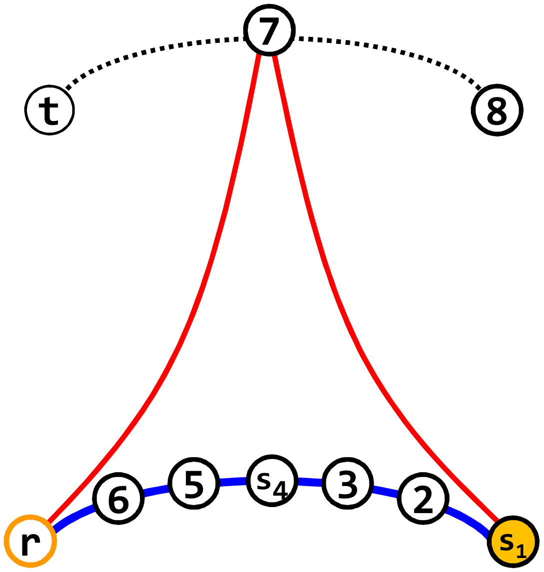
Vertex r (coord.x)
requests for data



Data localized at s_1



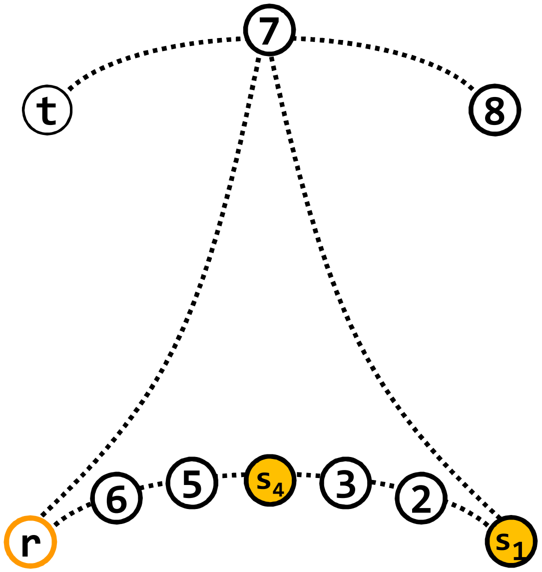
Selection of path to s_1
(coord. y_1) follows
hyperbolic distance
$d_{Hyp}(r,s_1) < d_{Hyp}(r,s_1)$



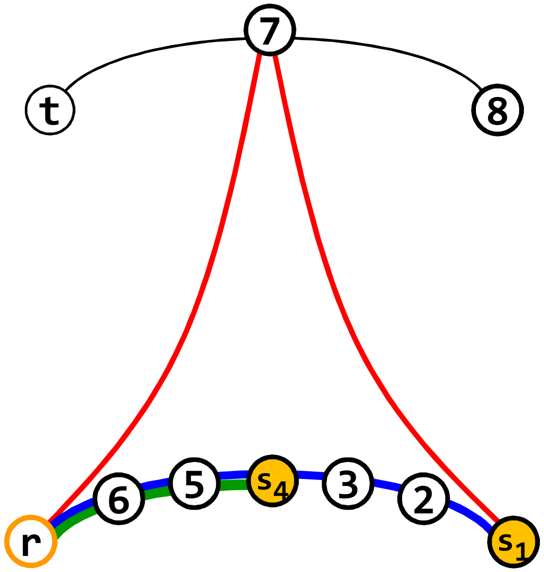
$d_{Graph}(r,s_1)$ (=6) > $d_{Graph}(r,s_1)$ (=2)
$d_{Hyp}(r,s_1) < d_{Hyp}(r,s_1)$

Example (2)

Requested data localized at s_1 and s_4



At r selection between location s_1 (coord. y_1) and s_4 (coord. y_4) conditioned by the hyperbolic distance **$d_H(r,s_4)$ < $d_H(r,s_1)$**

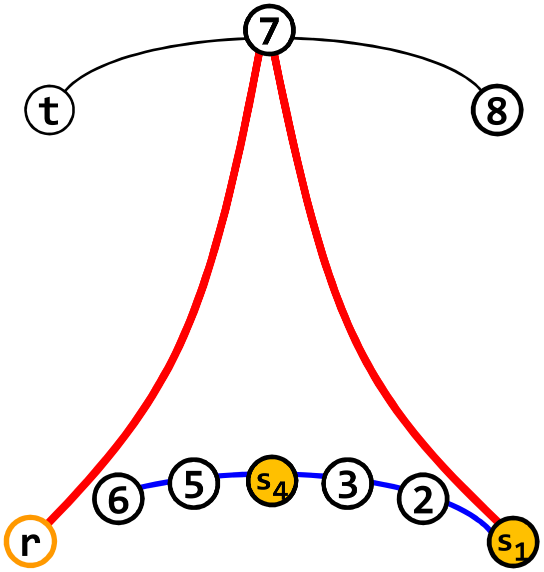


$d_G(r,s_4)$ (=3) > $d_G(r,s_1)$ (=2)

$d_H(r,s_4)$ < $d_H(r,s_1)$

if **geodesic path** (r,s_4) exists then r selects s_4

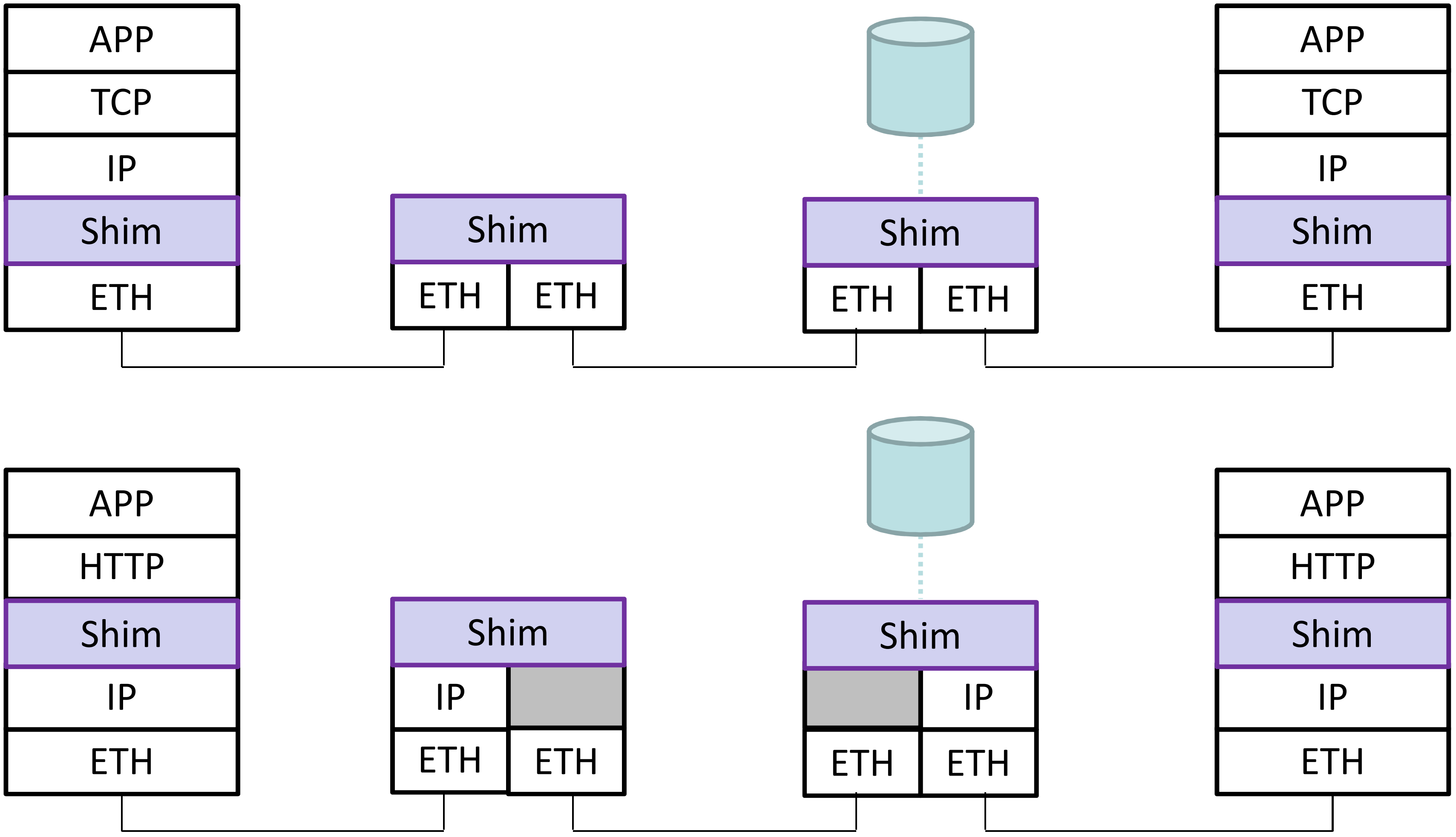
If geodesic path (r,s_4) doesn't exist then r selects s_1 along **quasi-geodesic path $d'_H(r,s_1)$** with $|$**$d'_H(r,s_1)$ – $d_H(r,s_1)$**$| < 2\delta$



If not, r selects s_1 along **quasi-geodesic path $d'_H(r,s_1)$**

$|$**$d'_H(r,s_1)$ – $d_H(r,s_1)$**$| < 2\delta$

Communication stack (end-to-end)



Geometric routing: properties and performance

Properties

- Coordinates can be used by distributed routing function to perform **geometric routing** decisions
- Operates by assigning to each node virtual coordinates in metric space (X,d) used as locators to perform point-to-point routing decisions
- Data object locators substitute to network locators
  - But can also be used in combination with other network locator spaces, e.g., IP addresses for interoperability

Performance

- **Tradeoff: memory space** (needed per node to store routing algorithm input + output (routing table entries))

  vs. **routing path stretch** (ratio between routing path length and topological path length)

  vs. **adaptation cost** (communication cost and computational complexity)
- **Convergence time**: upon occurrence of external/internal event, time elapsing before reaching new stable and consistent (no forwarding loops) routing state

Performance metrics

- **Multiplicative (additive) stretch**: max. over all source-dest. pairs (u,v) of ratio (difference) between navigation path cost (or distance) from node u to v and topological path cost (or distance) from same node u to v

- **Memory complexity**: memory bit-space required to store information used by the algorithm (input) and memory space required to locally store tables (output)

- **Communication complexity**
  - in space: total number of information messages exchanged between nodes (along graph edges) for local computation of navigation entries
  - in time: difference of time units between first emission of a message and last reception of a message during any execution of the algorithm (assuming slowest message uses one time unit per edge)

- **Computational complexity** in time: amount of time taken by the algorithm to run as a function of the input size

- Performance tradeoffs : **memory space** (per node to locally store entries) vs. **routing path stretch** vs. **communication cost** (distribution)

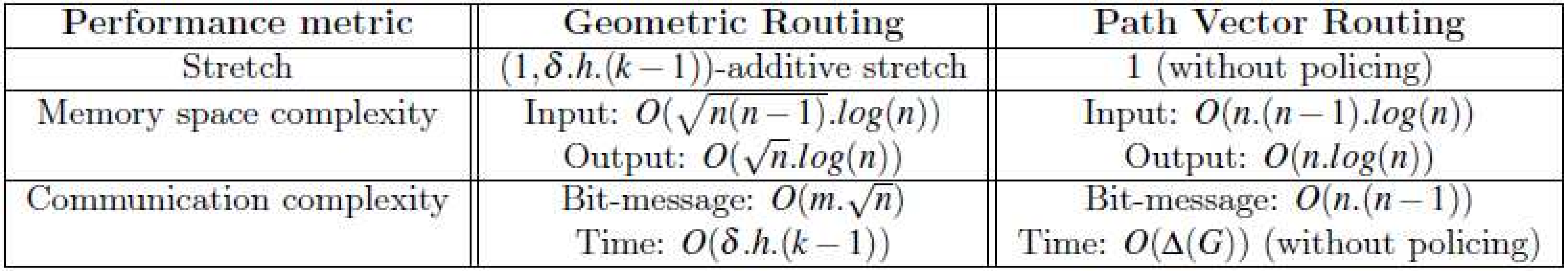
Performance comparison (1)

Geometric

- $(1, \delta.h.(k-1))$ additive stretch
- Memory space
  - Input: $O(\surd n. \surd(n-1).\log(n))$
  - Table: $O(\surd n.\log(n))$
- Com. complexity in bit-messages per vertex: $O(\surd n.m)$
- Com.complexity in time: $O(\delta.h.(k-1))$

Path-vector

- In absence of policing: stretch 1
- Memory space
  - Input: $O(\Delta(G) . (n-1) . n \log(n))$
  - Table: $O(\Delta(G) . n \log(n))$
- Com.complexity in bit-messages per vertex: $O(n . (n-1))$
- Com.complexity in time: $O(\Delta(G))$



Performance metric	Geometric Routing	Path Vector Routing
Stretch	$(1,\delta.h.(k-1))$-additive stretch	1 (without policing)
Memory space complexity	Input: $O(\sqrt{n(n-1)}.log(n))$ Output: $O(\sqrt{n}.log(n))$	Input: $O(n.(n-1).log(n))$ Output: $O(n.log(n))$
Communication complexity	Bit-message: $O(m.\sqrt{n})$ Time: $O(\delta.h.(k-1))$	Bit-message: $O(n.(n-1))$ Time: $O(\Delta(G))$ (without policing)

1. **Factor gain of n (#nodes) in memory space required to store routing information**
2. **Factor gain of ($n^{1/2}$) in memory space required to store routing tables**
3. **Limited routing path stretch increase to a small additive constant (characterizing the geometric property of the topology)**

Performance comparison (2)

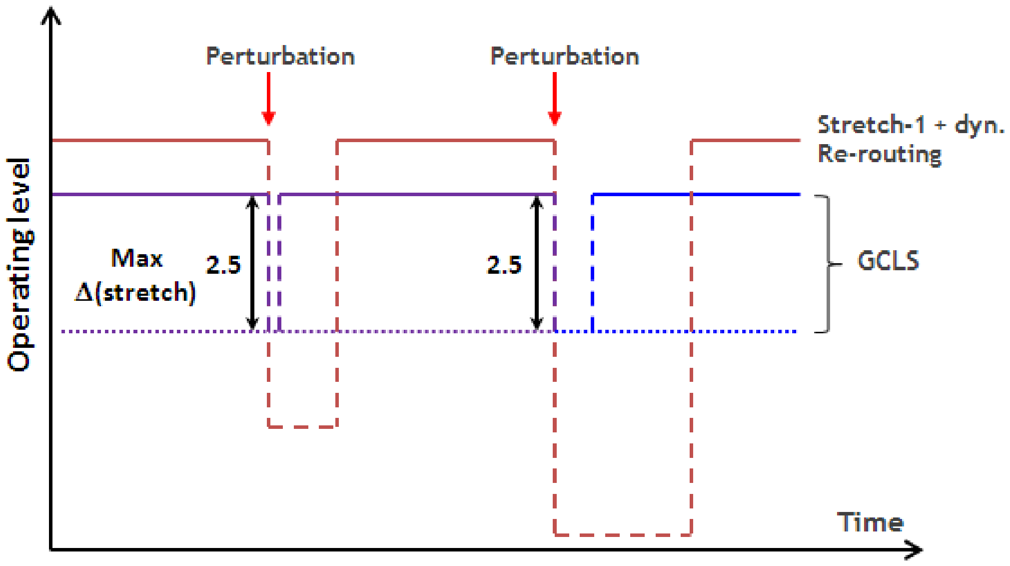
- If hyperbolicity of n-vertex graph G is $\delta \geq ½$

  Then G admits an additive $O(\delta.\log(n))$ spanner with at most $O(\delta.n)$ edges, and linear time construction of distance approximating trees with an additive error $O(\delta.\log(n))$
- Consequently, such graphs admit [Gavoille,2005][Chepoi,2008]
  - $\delta.\log(n)$-additive routing labeling scheme which uses $O(\delta.\log^2(n))$ bit labels and performs routing decision in $O(\log_2(4\delta))$ time
  - $\delta.\log(n)$-additive distance labeling scheme which uses $O(\log^2(n))$ bit labels and constant time distance decision
- In general, closer δ value to 0, lower stretch increase
  - Stretch gain trades against memory increase as each vertex maintains an association between distance derived from header and next-hop to the corresponding routing

Resilience

Resilience properties of Geometric-Coord. Labeling Scheme (GCLS)

- Principle: exploit structural and behavioral properties of the graph
- If the optimal (current) path is "too far" from any other alternate path the reconvergence time may be too slow or memory/processing consuming
- Solution: move from engineering model (failsafe = protection or safe-to-fail model = restoration) to ecological resilience model

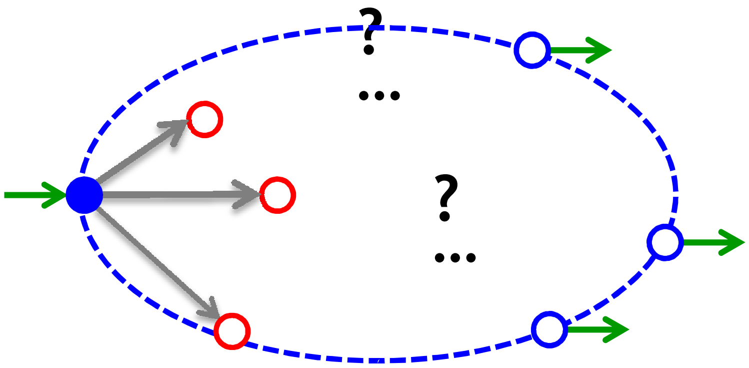


Network modes

	Sarnoff (Broadcast)	**Metcalfe (Ethernet) Baran (IP)**	**Information- oriented**
Spatio-temporal distribution of information	density u = cte (≡ centralized and static)	density u = u(x) (≡ distributed and *deterministic*)	Prob. density **u = u(x,t)** (≡ dynamic and stochastic)
Pattern	star, hub and spoke, concentric	Mesh (nodes = GTW between broadcast domains)	Complex network (nodes = GTW between information domains)
Scale (value)	n	n^2	2^n
Channel	Physical (Optical)	Logical (TCP/IP)	Data
Metric(s)	Spatio-temporal	Spatio-temporal	+Semantic-Structural
Deployment	Coordinated	Organic	not yet deployed
Example	CDN, cloud, etc.	Computer networks, web, mail	Communities

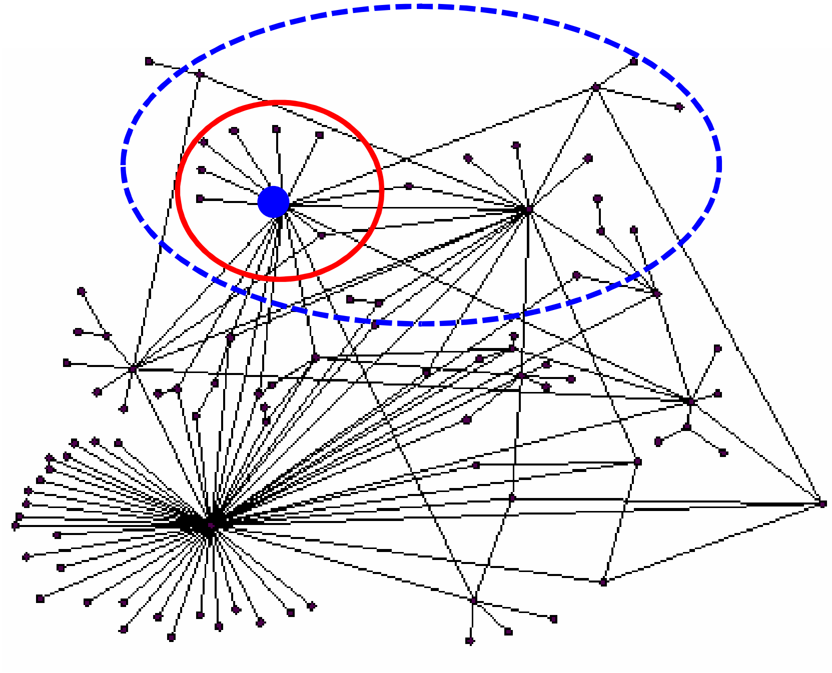
= cloud model = Internet model

Information Propagation in Complex Networks



Modeling behavior of information propagation

- **Microscale-level**: nodes interact locally with their neighbors f(n) prop. to node degree d(n) or joint degree ($d(n_1)$,$d(n_2)$) $\Rightarrow$ **Probabilistic model** (de-facto model in complex system modeling)
- **Mesoscale-level**: nodes interact with e.g. n^{1-a} nodes (α = scaling parameter) $\Rightarrow$ **Stochastic model**



Stochastic propagation model: propagation rate (b) and dispersion/fluctuation ($\sigma > 0$)

- Continuous time (uncontrolled) **Ito process** X_t driven by Wiener process W_t

$$dX_t = b(X_t, t)dt + \sigma(X_t, t)dW_t$$

- Temporal evolution of probability density function u(x,t) of X_t satisfies the **Fokker-Planck equation** a.k.a. Kolmogorov forward equation

$$\frac{\partial u}{\partial t} = -\frac{\partial}{\partial x}[b(x,t)u(x,t)] + \frac{1}{2}\frac{\partial^2}{\partial x^2}[\sigma^2(x,t)u(x,t)]$$

Forward propagation of uncertainty (intrusive PCE method)

- u → u(x,t,ξ) ≡ linear combination of orthogonal polynomial $\psi_k = \psi_k(\xi)$, with Gaussian random variable ξ

$$u(x,t,\xi)=\sum_{i=0}^{P}\underbrace{u_i(x,t)}_{\text{deterministic}}\underbrace{\psi_i(\xi)}_{\textbf{stochastic}}$$

$$\text{Hermite}:\psi_i(\xi)=(-1)^k e^{(\xi^2/2)}\frac{d^i}{d\xi^i}e^{(-\xi^2/2)},\ i=0,1,2,\ldots$$

Orthogonality property (inner product in ξ space):

$$<\Psi_m\Psi_n>=\int_\Omega \Psi_m(\xi)\Psi_n(\xi)w(\xi)d\xi=c_m\delta_{nm}$$

- $b \rightarrow b(\xi)=\sum_{j=0}^{Q} b_j(x,t)\psi_j(\xi)$

Substitution in governing Fokker-Planck equation:

$$\sum_{i=0}^{P}\frac{\partial u_i}{\partial t}\psi_i(\xi)+\sum_{j=0}^{Q} b_j\psi_j(\xi)\left(\sum_{i=0}^{P}\frac{\partial u_i}{\partial x}\psi_i(\xi)\right)=\frac{\sigma^2}{2}\sum_{i=0}^{P}\frac{\partial^2 u_k}{\partial x^2}\psi_i(\xi)$$

Multiplication by $\psi_k(\xi)$ (k=0,1,..,P) and integration over prob.space Ω for each k (stoch. Galerkin projection on polynomial basis) yields:

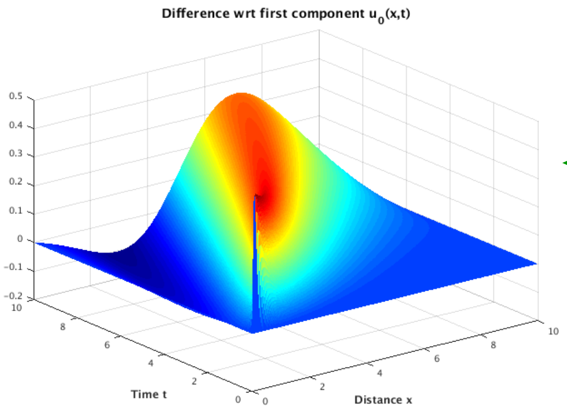
$$\sum_{i=0}^{P}\frac{\partial u_i}{\partial t}<\psi_i\psi_k>+\sum_{i=0}^{P}\frac{\partial u_i}{\partial x}\left(\sum_{j=0}^{Q} b_j<\psi_i\psi_j\psi_k>\right)=\frac{\sigma^2}{2}\sum_{i=0}^{P}\frac{\partial^2 u_k}{\partial x^2}<\psi_i\psi_k>$$

Set P=2, Q=2 and exploit orthogonality property of Hermite polynomials

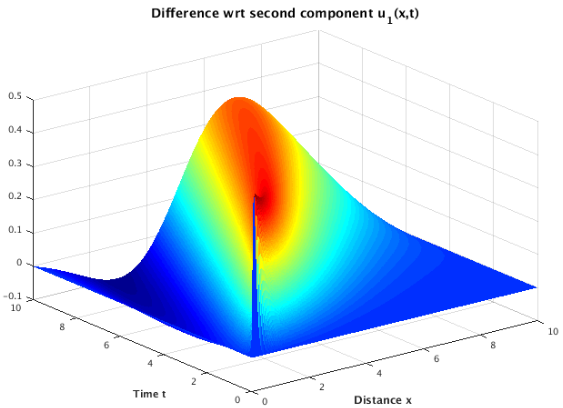
$\Rightarrow$ System of P(=2)+1 = 3 coupled differential eq. (independent of ξ):

$$\begin{cases} \dfrac{\partial u_0}{\partial t} + b_0 \dfrac{\partial u_0}{\partial x} + b_1 \dfrac{\partial u_1}{\partial x} + 2b_2 \dfrac{\partial u_2}{\partial x} = \dfrac{\sigma^2}{2} \dfrac{\partial^2 u_0}{\partial x^2} \\ \dfrac{\partial u_1}{\partial t} + b_1 \dfrac{\partial u_0}{\partial x} + (b_0 + 2b_2) \dfrac{\partial u_1}{\partial x} + 2b_1 \dfrac{\partial u_2}{\partial x} = \dfrac{\sigma^2}{2} \dfrac{\partial^2 u_1}{\partial x^2} \\ \dfrac{\partial u_2}{\partial t} + b_2 \dfrac{\partial u_0}{\partial x} + b1 \dfrac{\partial u_1}{\partial x} + (b_0 + 4b_2) \dfrac{\partial u_2}{\partial x} = \dfrac{\sigma^2}{2} \dfrac{\partial^2 u_2}{\partial x^2} \end{cases}$$

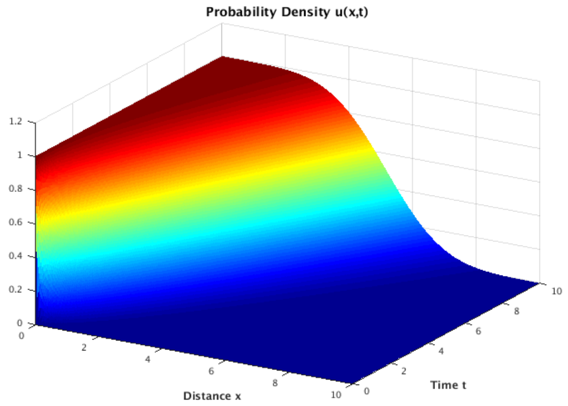
Numerical results



Difference between u(x,t) vs. components u_0(x,t) and u_1(x,t) as obtained by solving system of 3 coupled differential equations



Diff.eq. system solved using `Matlab R2015` with initial cond. $u_k(x,t=0)=d(x-x_0)$, k=0,1,2



Model without uncertainty in propagation rate (still) overestimates micro-scale effects and underestimates meso-scale effects

Analysis

- Assignment of locators to data objects, where locators identify "position" in data object space
  - Locators drawn from hyperbolic metric space ($\mathbb{H}^2$,d_{HYP}) enables geometric information routing on hyperbolic coordinates
  - Variant of geometric routing (GCLS): measurement-based labeling scheme
  - Deep implications on routing stretch, succinctness but also robustness
- Invariants in all BGP/path-vector routing alternatives
  - Combine two types of routes: routes for destination in close neighborhood and routes outside their neighborhood
  - Main difference in discovery process results from exchanges of routing information: pull (search, route servers, etc.) vs. push (dissemination)
  - Use of distance metric
- Routing schemes such as BGP
  - Independent of global or link metrics (AS path length being a route selection parameter among others)
  - Driven by local policy decisions
  - Difficult to replace as long as Internet operated organically

Thanks for your attention

Acknowledgment: **EULER FP7 project** (Grant: 258307) is funded by the EU Commission EC Future Internet Research and Experimentation (FIRE) Initiative (DG-INFSO Unit F4)