Delay Tolerant Networks
One step towards Information-Centric Networking

Chadi BARAKAT

INRIA Sophia Antipolis, France
DIANA group

Email: Chadi.Barakat@inria.fr
WEB: http://team.inria.fr/diana/chadi/

References - Sources

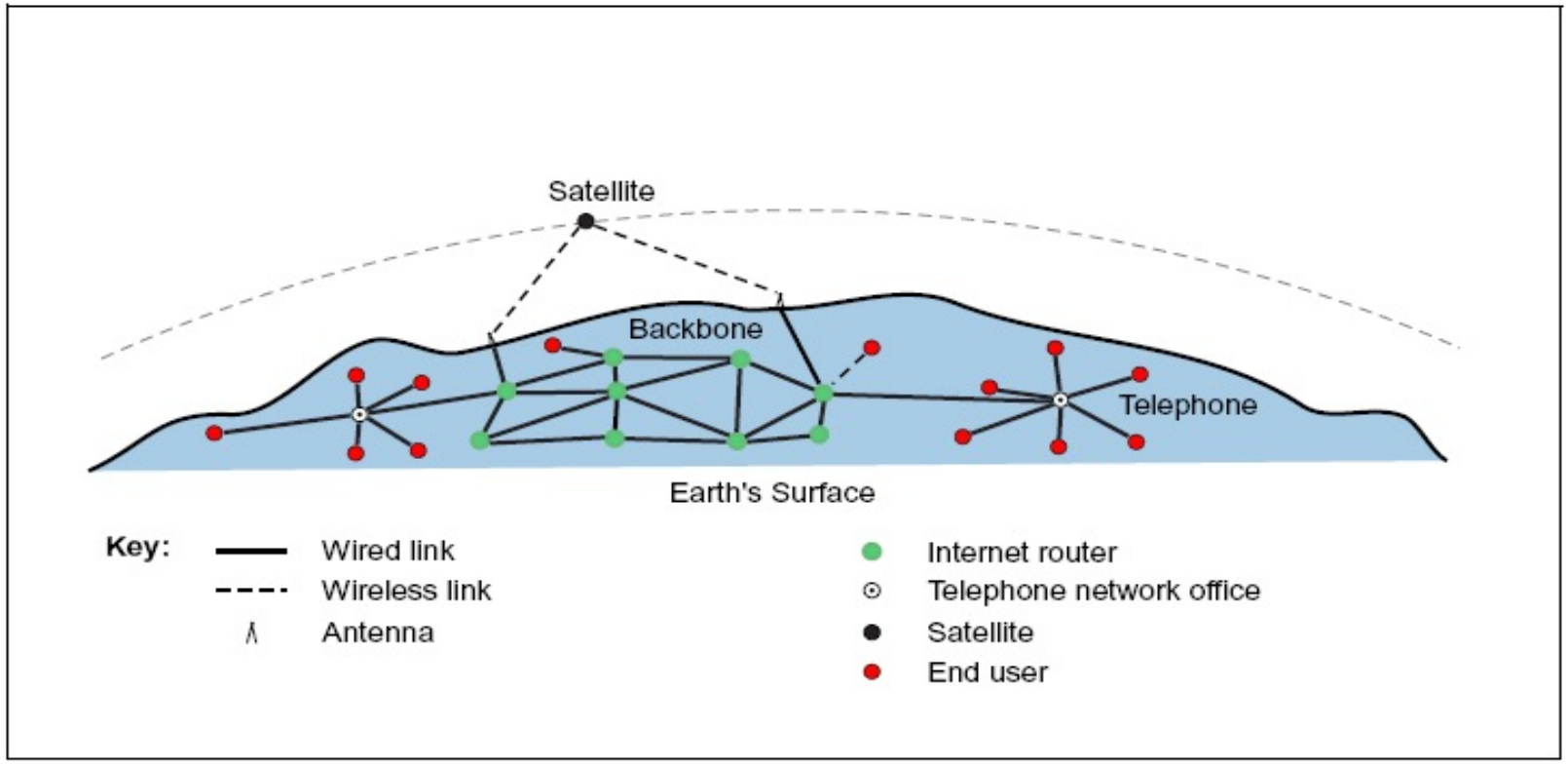
- Slides of Prof. Jennifer Rexford, Princeton Univ.
- Slides of Dr. Thrasyvoulos Spyropoulos, Eurecom

Context

- A new environment where disconnections are dominant
  - End-to-end principle behind the Internet and MANETs no longer apply
- Provide an overview of this new exciting area
- Motivate a new networking paradigm
- Describe the challenges for existing protocols and present proposed solutions (focus on routing)
- Present some modeling issues

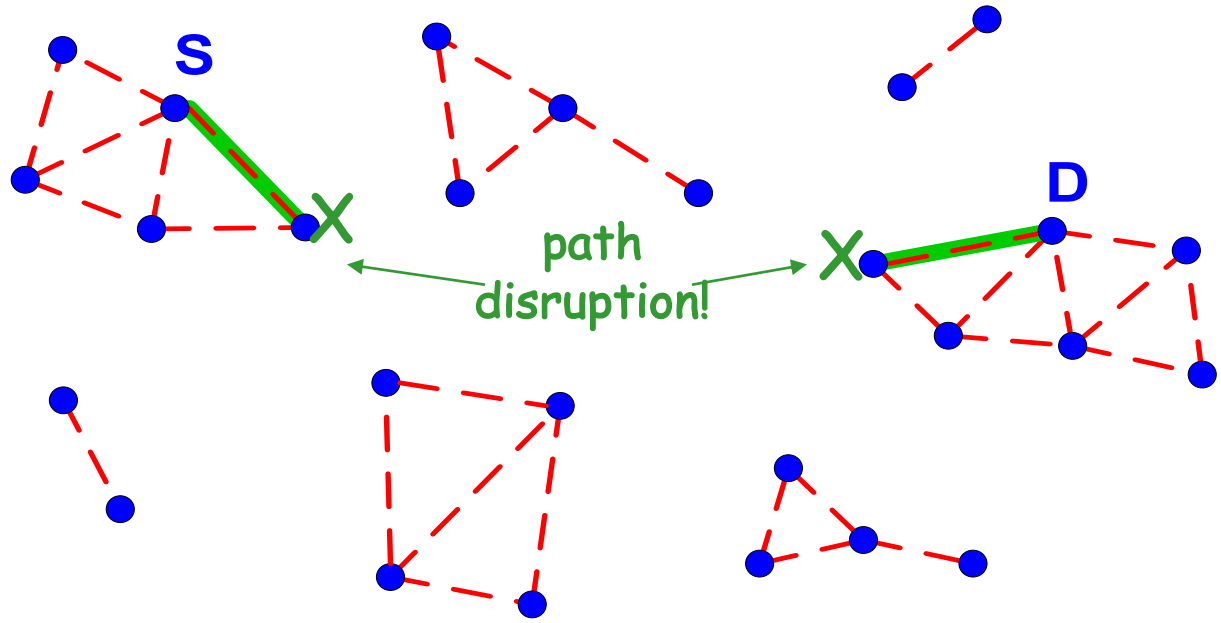
"Legacy" Networks

- Internet, Telephone network
- Wired or fixed links



- A SUCCESS STORY!

Wireless Connectivity: A Different View

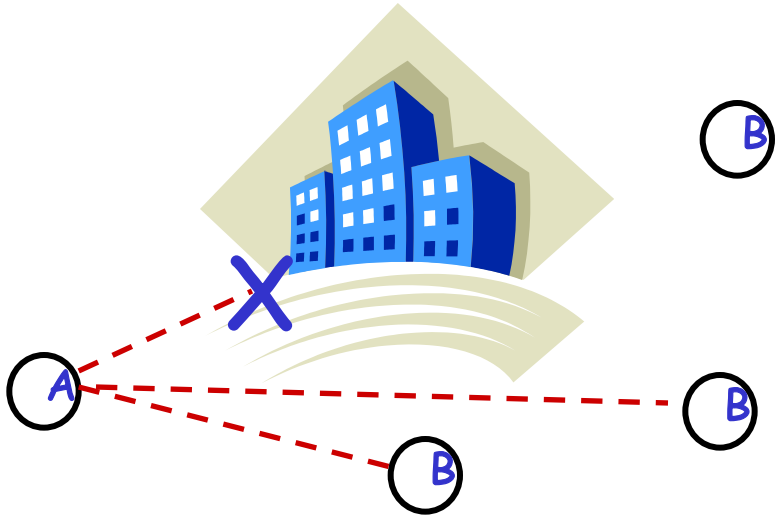


	Paths	E2E Delays
Internet	end-to-end	≈ ms
Wireless Ad Hoc	partial	sec, min(?), hours(?),…

Intermittent Connectivity: The Technical Argument

Intermittent Connectivity may appear because of:

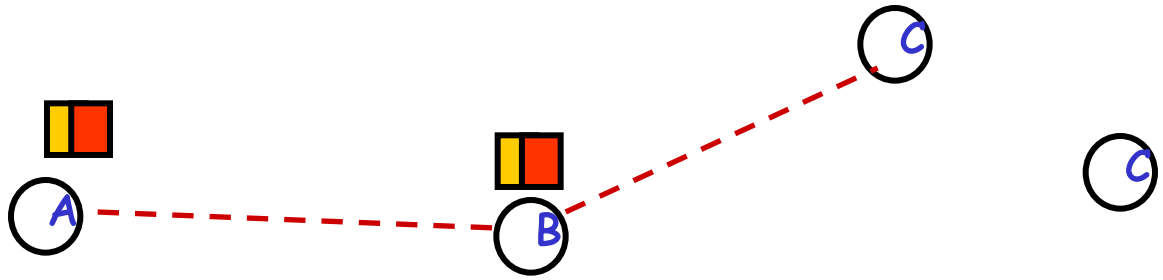
- propagation effects: shadowing, deep fades



Intermittent Connectivity: The Technical Argument(2)

Intermittent Connectivity may appear because of:

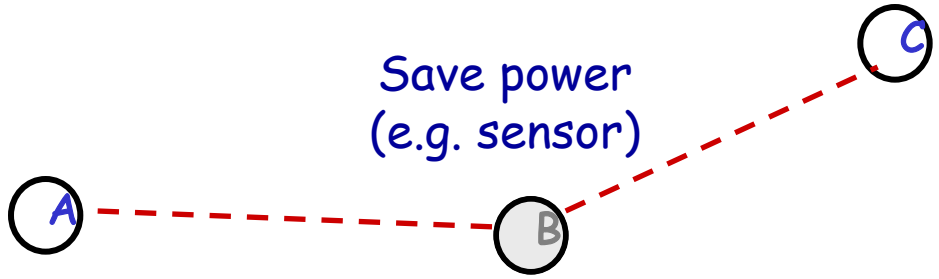
- **Propagation effects**, shadowing, deep fades
- **Mobility:** paths change too fast; devices become out of reach.



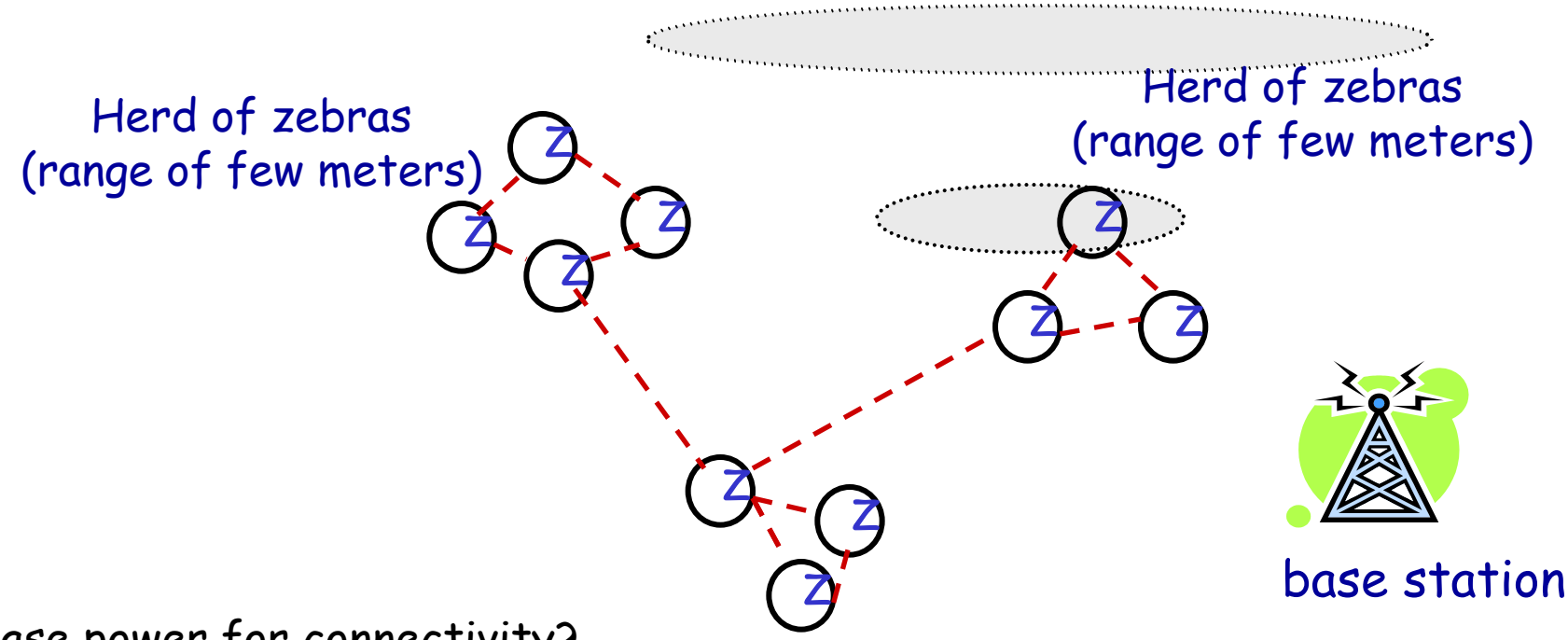
Intermittent Connectivity: The Technical Argument(3)

Intermittent Connectivity may appear because of:

- **Propagation effects**, shadowing, deep fades
- **Mobility:** paths change too fast; huge overhead for maintenance
- **Power:** nodes shut down to save power or "hide" (doesn't collaborate)

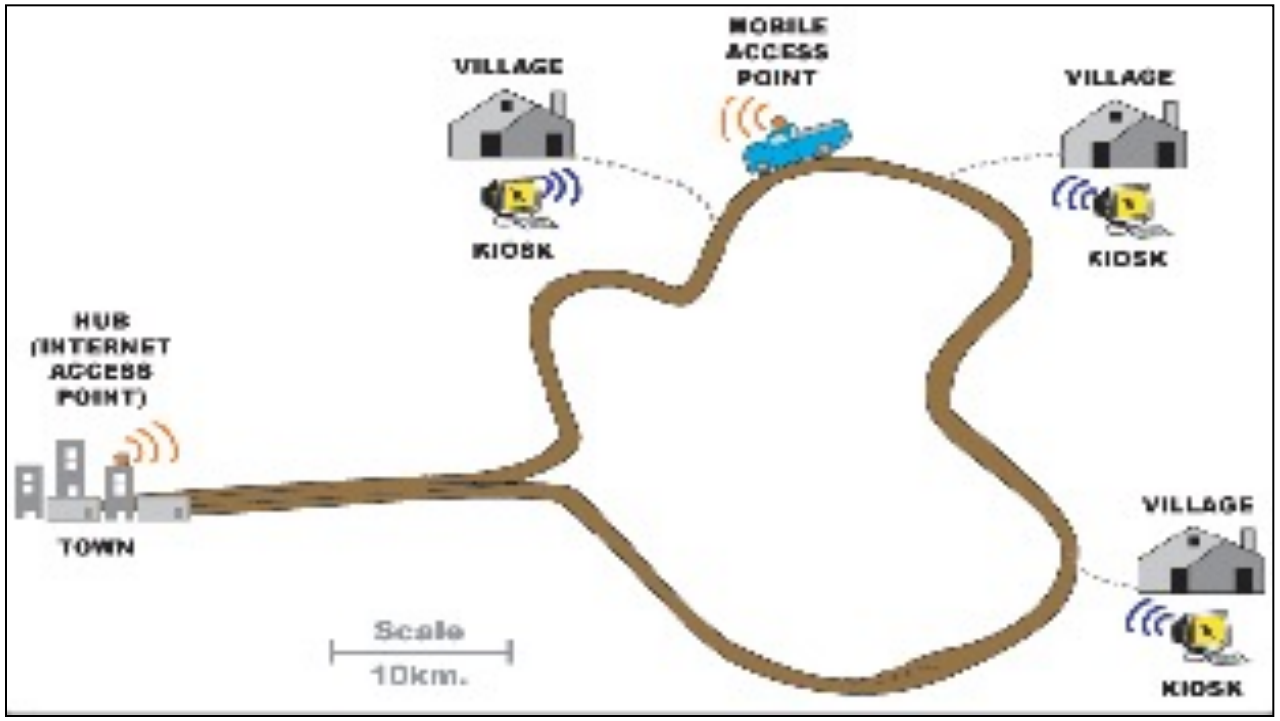


Applications: Sensor Networks for Habitat Monitoring



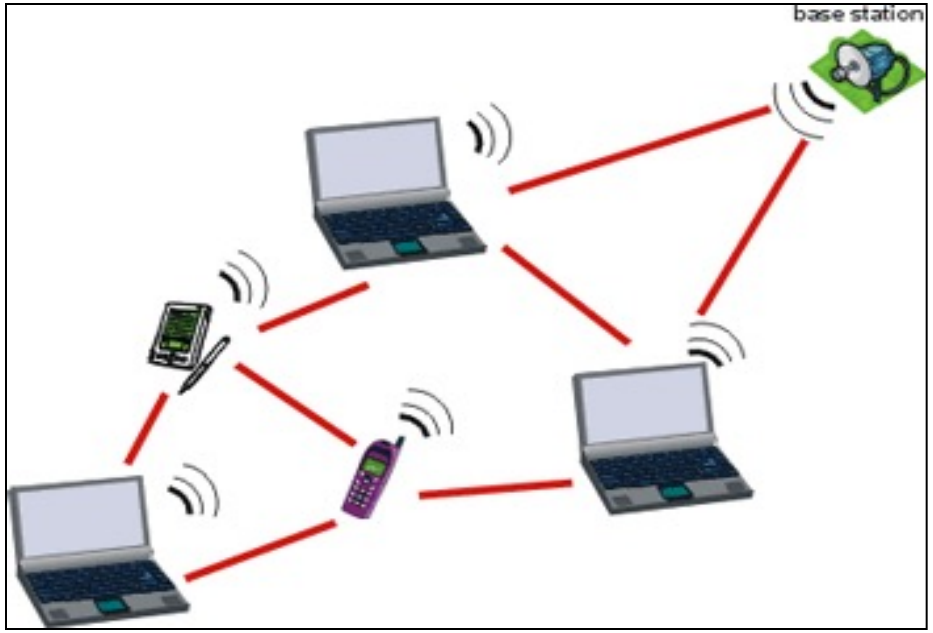
- Increase power for connectivity?
  - Considerably reduce lifetime of network! (power law)
  - What about obstacles?
- Live with a sparse network (connected clusters)
- Use DTN principles to carry traffic towards sink

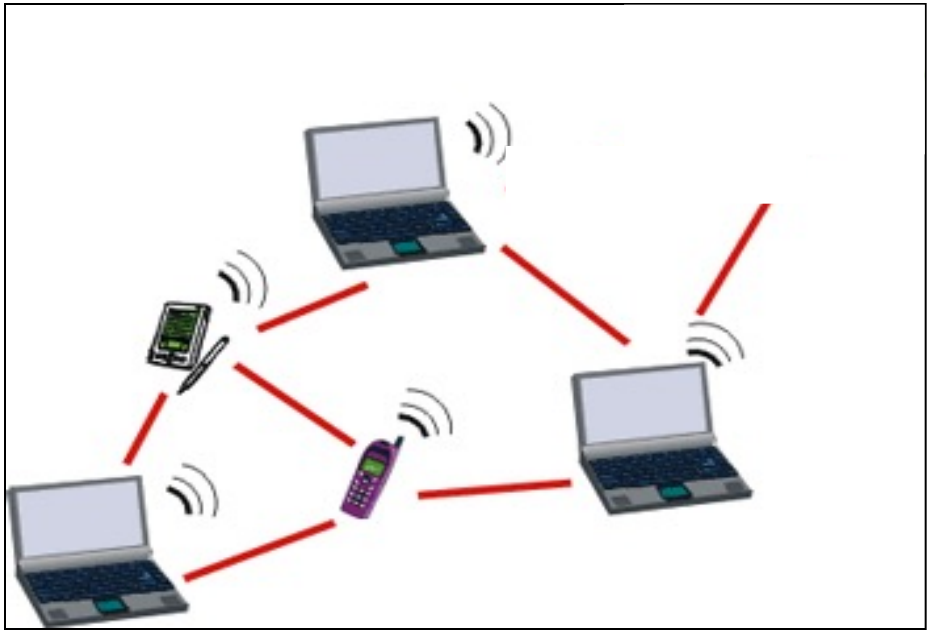
Applications: Internet to Remote Communities



- Service Kiosks: Banking and Governement, Email, WWW (cached)
- Leverage buses and cars etc (called ferries) to transfer delay tolerant information to close city and vice versa.

Intermittent/Opportunistic Wireless Data Networks



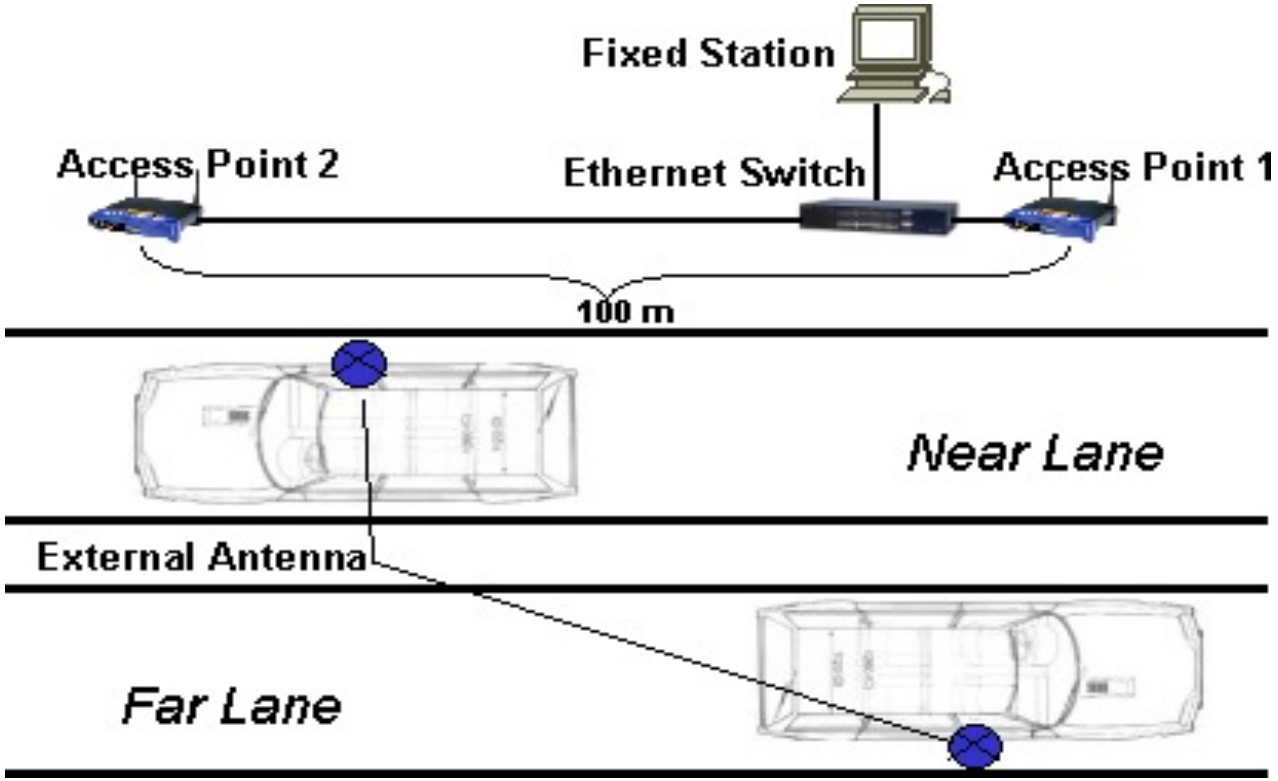


- Multi-hop access
- Extend infrastructure

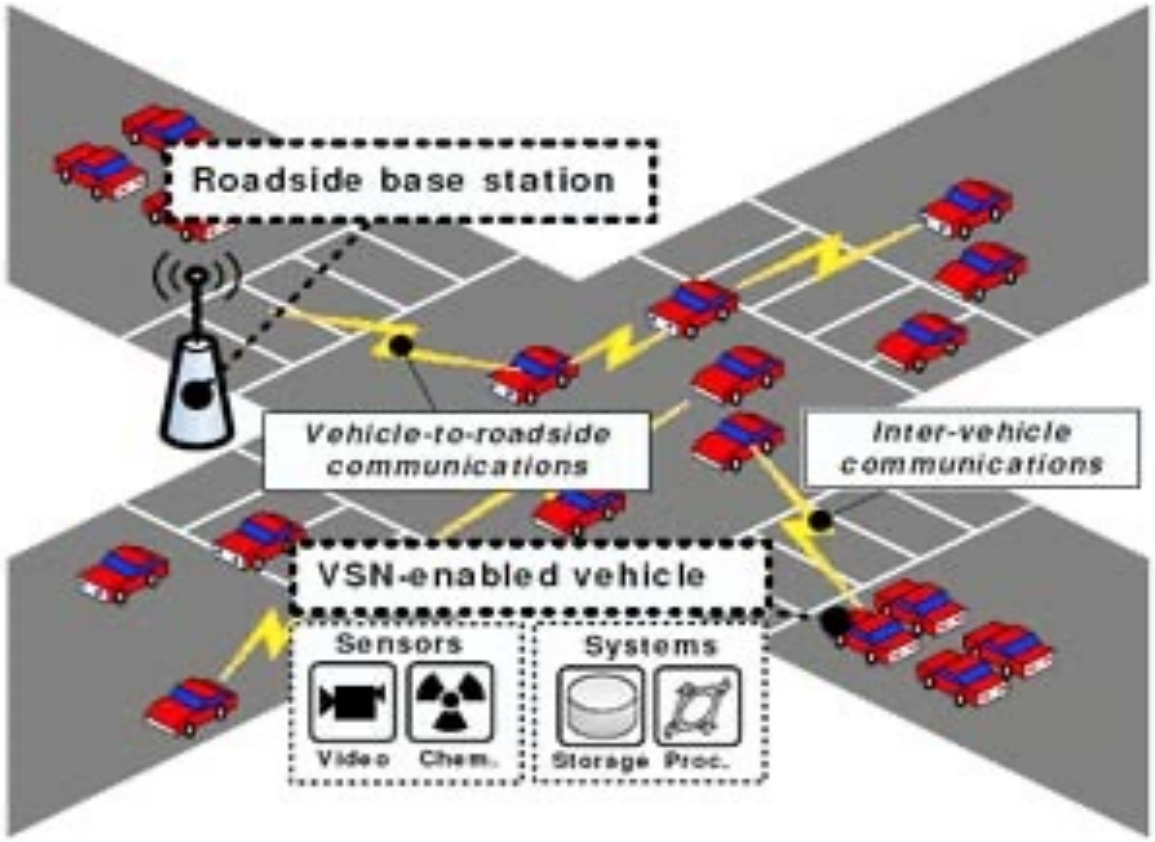
- Peer-to-peer access
- Local news/info may be available at a node nearby
- Bypass infrastructure($$)

Vehicular Networks: "Drive-Thru Internet"

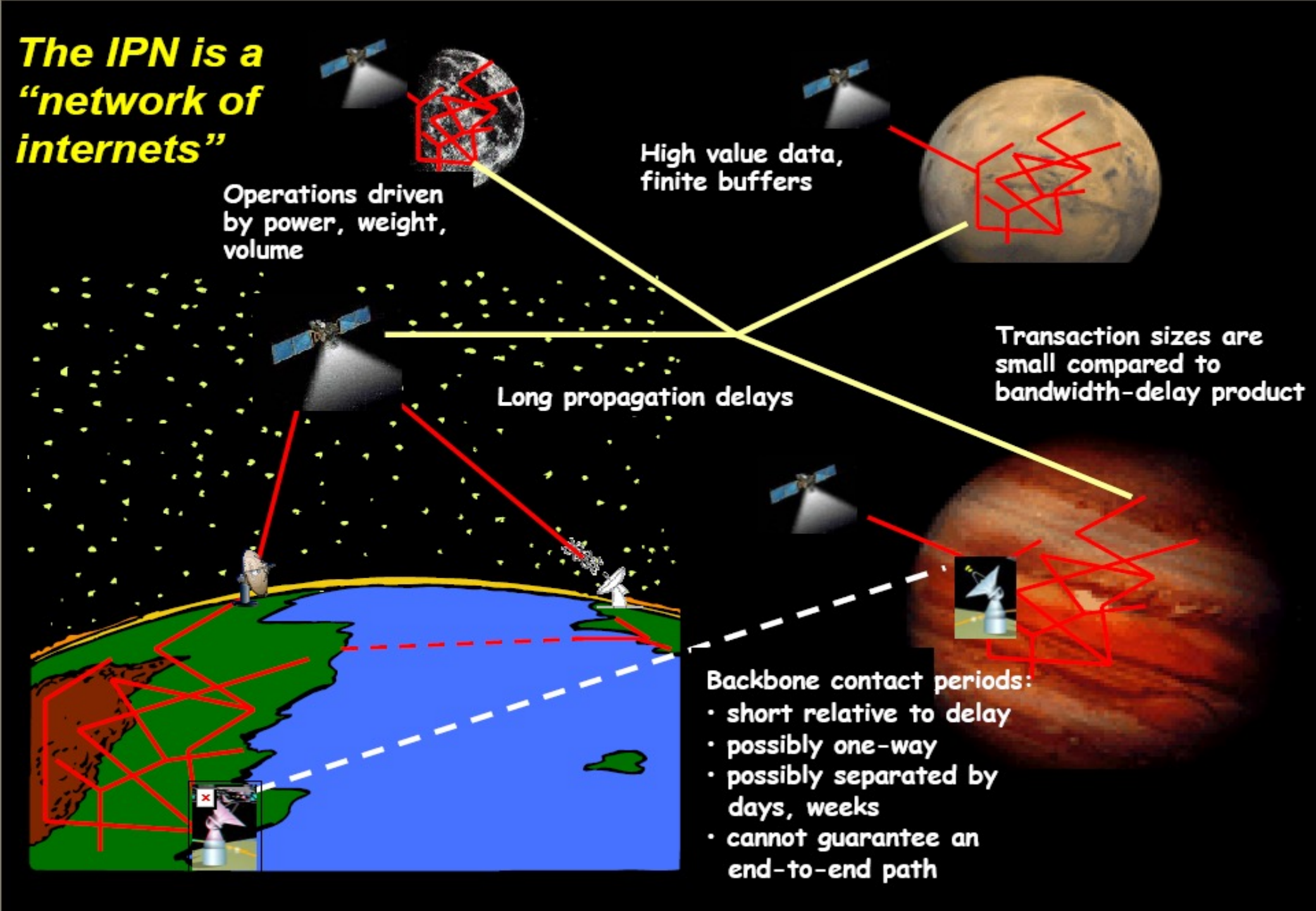
Vehicle-to-roadside (base station, sensors)



Inter-Vehicular Communications



- Safety, accident prevention (lightweight vehicles)
- Traffic reports, traffic routing
- Info-services (Internet, local adds, nearby gas stations, etc.)


The IPN is a "network of internets"
Operations driven by power, weight, volume
High value data, finite buffers
Long propagation delays
Transaction sizes are small compared to bandwidth-delay product
Backbone contact periods:
• short relative to delay
• possibly one-way
• possibly separated by days, weeks
• cannot guarantee an end-to-end path

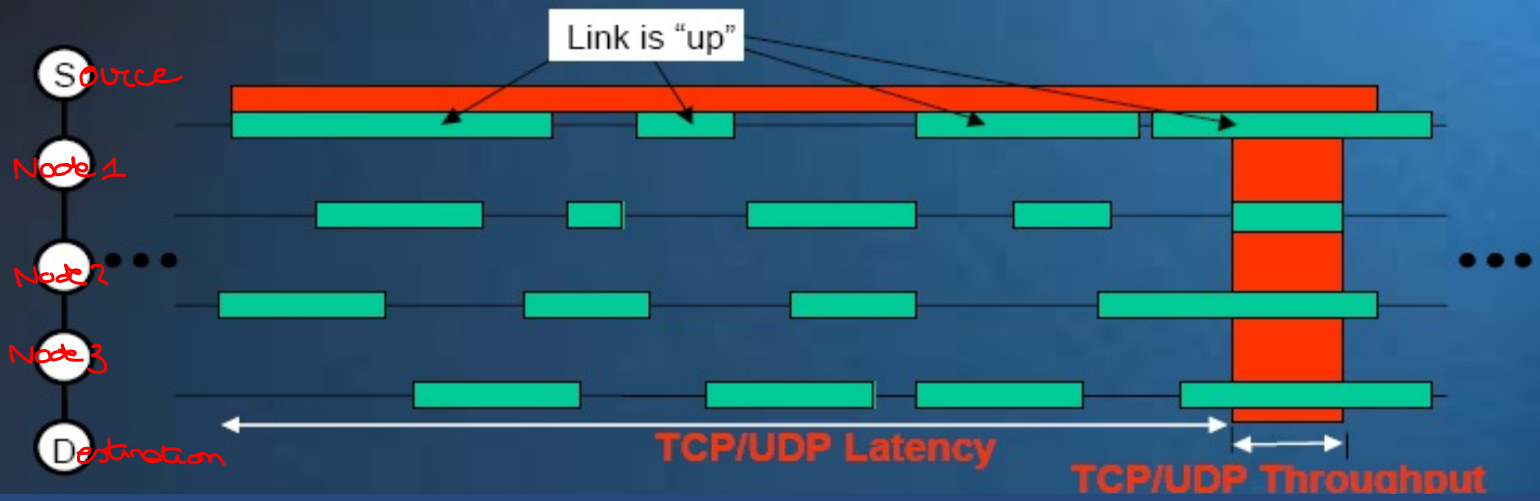
The "Internet" Assumptions (not holding)

1. Continuous, bidirectional end-to-end path
   - Intelligence at the edge, protocols are "conversational"
2. Short round-trips
   - TCP congestion control: receive ACK to send next packet
3. Low error rates

These assumptions are violated in DTNs. Traditional protocols suffer:

- ❑ Transport → how can I notify if the content is well received
- ❑ Naming → related to addressing
- ❑ Security → check the signature
- ❑ Routing

DTN vs End-to-End Internet Operation

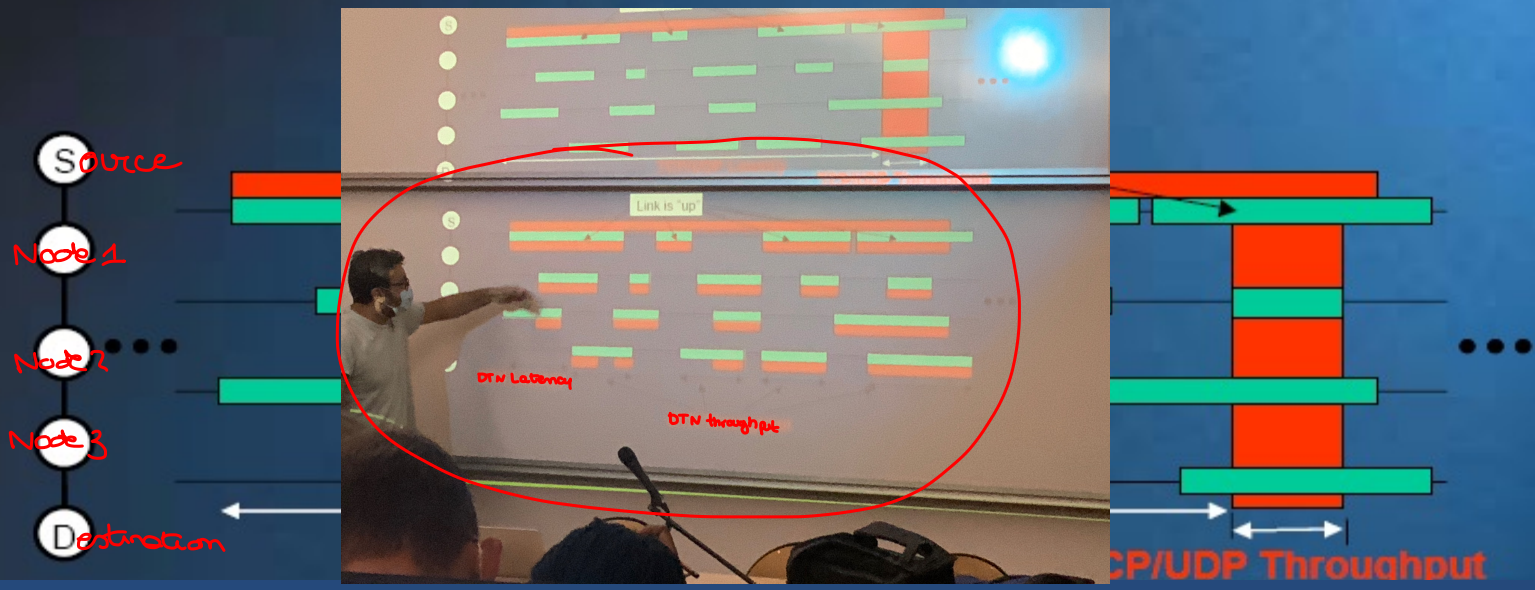


green bar = tells when the link is up

red bar = data available at the source

end-to-end → communication is possible only when the link is up
the bit rate depends on the speed

DTN vs End-to-End Internet Operation



with DTN we have more data compared

better in terms of throughput and delay

Problem - Episodic Connectivity: Hop-by-Hop vs. End-to-End: In End-to-End communication, all links in a path must be available at the same time for a message to be delivered over the path. In Hop-by-Hop communication even if one link is up, data can progress over that link, as long as there are some data in the queue.

Assume now a path of 5 hops in tandem: $A \to B \to C \to D \to E$. Assume further that time is slotted and that each link in the path is up in a given time slot (independently of other links) with probability $p_{ij} < 1$ (i.e. $p_{AB}; p_{BC}; p_{CD}; p_{DE}$). One packet can be transmitted in each slot, and nodes have infinite buffers to store messages in transit.

i node at the beginning

probability saying how often the link is up

- **Question 1:** What is the expected delay of one packet for end-to-end transmission? What is the expected delay of one packet for hop-by-hop transmission? Calculate it as a function of p_{ij} .
- **Question 2:** What are the above delays for the following values of p_{ij}: Case 1) All $p_{ij} = 0.9$. Case 2) All $p_{ij} = 0.5$. Case 3) All $p_{ij} = 0.1$.
- **Question 3:**. What is the maximum sustainable throughput in the cases of end-to-end and hop-by-hop communication?

1) END-TO-END

The delay is 1 : I need to wait until the path is up and when the path is up things are fine

↓

the time for the path to be up, knowing that the probability is $P(a)\,P(b)\,\dots\,P(e)$, you succeed in $\frac{1}{P_{ab} + P_{bc} + P_{cd} + P_{de}}$

HOP-BY-HOP

$$\frac{1}{P_{ab} \cdot P_{bc} \cdot P_{cd} \cdot P_{de}}$$

with Hop by hop we are doing better than end-to-end

2)

3) $\text{throughput}_{\text{end to end}} = \frac{\#\text{packets}}{\text{window}} = \frac{\text{volume of data}}{\text{time}}$ [from node A to E]

$= \text{Product of probability}$

$\text{throughput}_{\text{hop by hop}} = \min(\text{Probabilities})$

Naming Services (DNS)

- Normally:

1) Start with DNS name: www.inria.fr
2) Talk with DNS server to resolve it to an IP address
3) Use IP address to connect to the destination
4) Strong relation between content, location and IP @

What if no path to the DNS server currently available?
What if no clear locality of addresses because of mobility?

Maybe there is no need for IP addressing …
(way for ICN and name-based routing)

Security

Intermediate nodes can drop messages, change them or add new ones

- No guarantee on the delivery
  - ACK lasts long time. In the mean time the destination has the message and can profit from it
  - A network of kind regular post. No express mail.
- Incentives for collaboration
  - Money? Tit-for-Tat? Reputation? Check our MobiTrade@Inria
- Certification Authorities (for authentication):
  - Who signs certificates if CA is offline?
  - Who revokes certificates if CA is offline?

Traditional Routing (proactive)

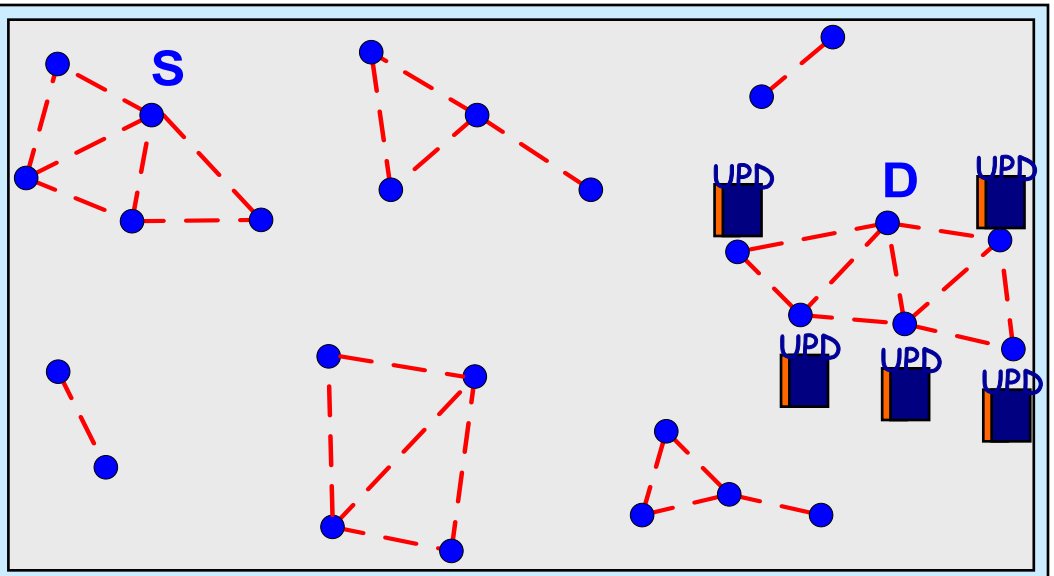
- how to set up routes between cluster of nodes
- not sensible to disconnections

Proactive Routing

(DSDV, OLSR, OSPF)

- Flood Periodic Topology Updates (UPD)
- S learns next hop to D

UPD reaches only same cluster as D!



any communication in a cluster cannot communicate with other clusters

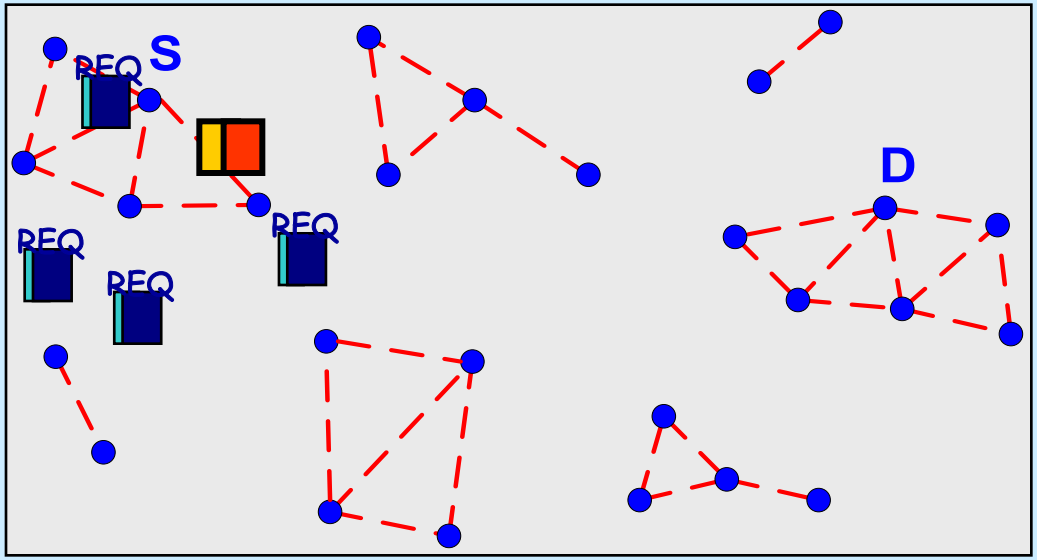
Traditional Routing (reactive)

Reactive Routing

(DSR, AODV)

- Flood Route Request (REQ)
- S waits for reply from D

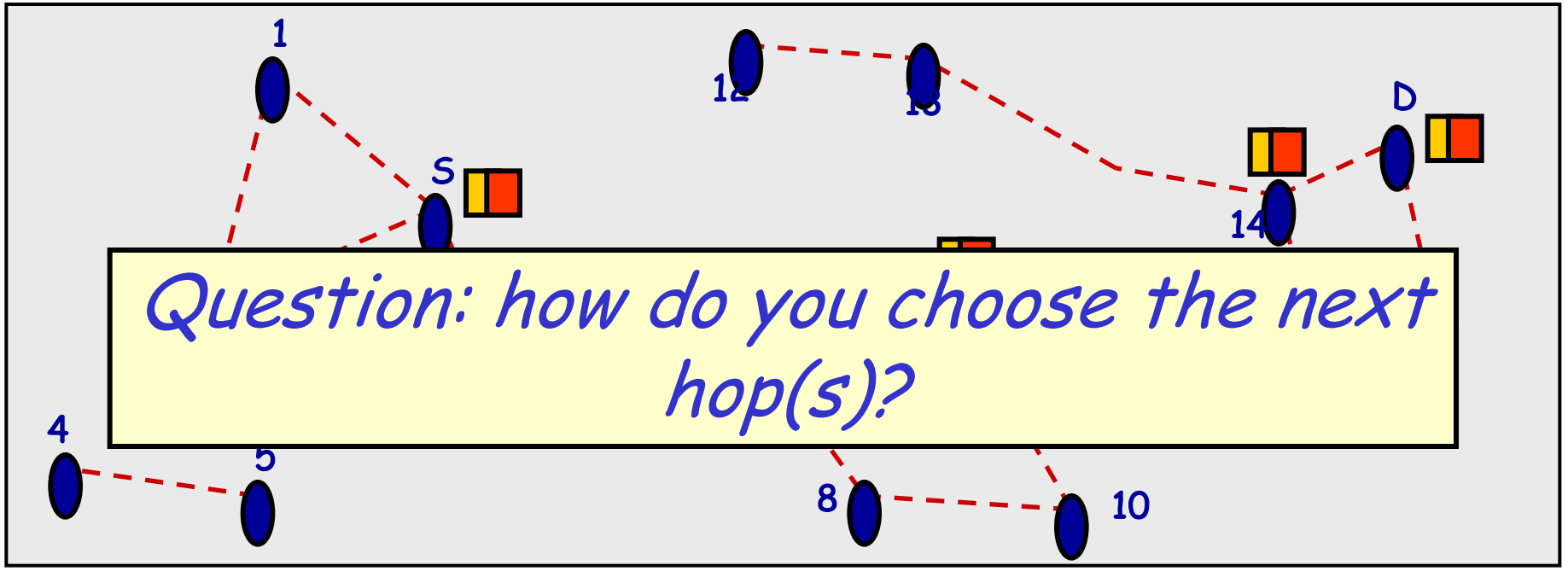
REQ reaches only same cluster as S!



Store-Carry-and-Forward (Mobility-assisted) Routing

- Connectivity-over-time

1. *Store* messages (persistent) when no link
2. *Carry* until forwarding opportunity arises
3. *Forward* independently and opportunistically
   - no knowledge about destination



I can divide clusters based on the →

Types of Contacts

- **Scheduled contacts** → deterministic
  - e.g. satellite links, bus routes
  - enforced contacts: robots, message ferries (more later)
  - all info known
- **Probabilistic contacts** → they vary and they repeat themselves, so they can be transformed in something deterministic
  - statistics about contacts known
  - e.g. mobility model, or past observation+prediction
  - sensors with random wake-up schedule
- **Opportunistic contacts** leverage every opportunity arising
  - not known before it occurs
  - e.g. a tourist car that happens to drive by the village

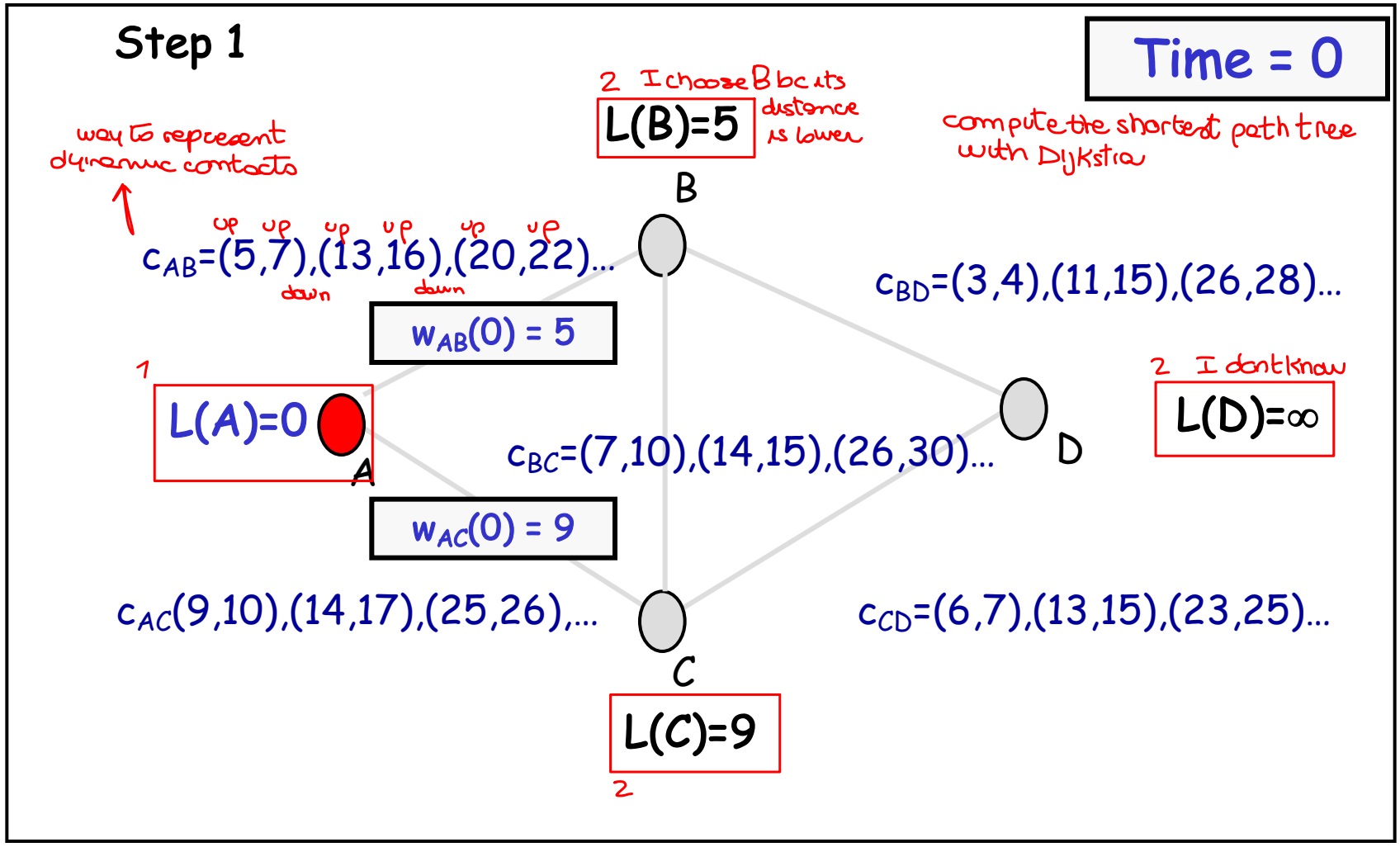
Routing with predicted contacts

(deterministic and probabilistic)

Routing with Complete Knowledge

- Computing minimum cost ("shortest") paths
- Delay:
  - Time till the contact (or link) is up again
  - Transmission → time to put the packet inside
  - Propagation → time to go to the other way
  - Queuing = Waiting for queue to drain → it depends on the traffic (which depends on the routing)
- Link weight **w(e,t)** = message arriving at edge e at time t, is predicted to arrive at end of e at time **t + w(e,t)**
  - edge e, time t → be it's not stationary
- Approach 1: MED (Minimum Expected Delay)
  - Dijkstra with edge delay equal to average inter-meeting time
- Approach 2: Modify Dijkstra's algorithm

Dijkstra's with Time-varying Costs



1) node A looks around $w_{AB}(0) = 5$ time for the link to be up

$w_{AC}(0) = 9$

2)

3)

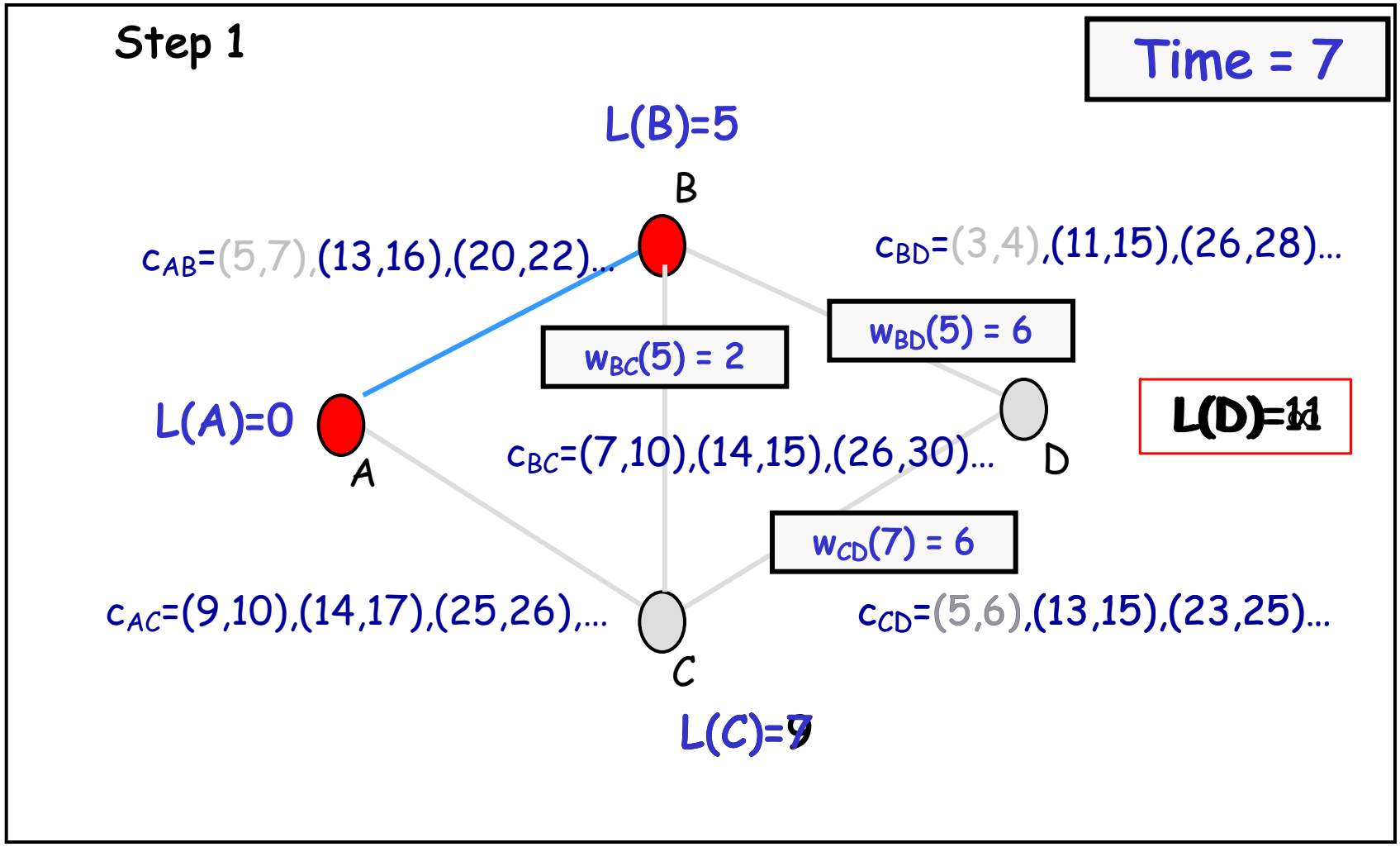
4)

5)

6)

7)

Dijkstra's with Time-varying Costs, cont'd



→ bc in reality you have to receive this information

MEED: Minimum Estimated Expected Delay (practical version of MED)

- Keep history of past contacts
- Maintain running average
  - Sliding window
  - Large window => slow reaction to changes
  - Small window => too many updates, oscillations
- Link-state epidemic dissemination
  - Whenever a contact changes significantly (x% form previous estimate) => flood topology update packet

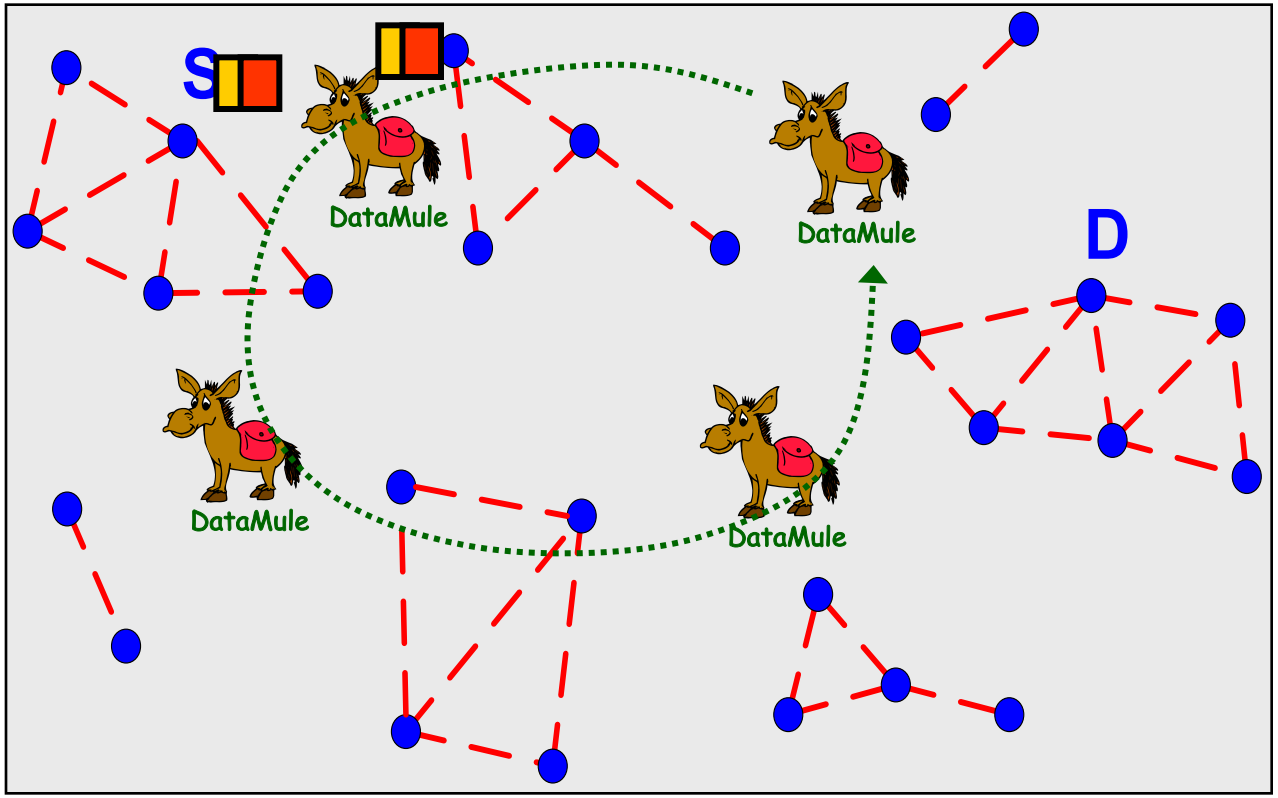
Enforcing the Contacts: Message Ferrying

- A sparse network of *"production"* nodes
- Nodes may be static (e.g. sensors) => how to bridge partitions?
- Nodes may be mobile, but slow => long delays
  - waiting for a contact to occur may take time

- Solution: Use specialized nodes (*DataMules* or *Message Ferries*) to carry traffic between production nodes
  - Ferries are always mobile
  - No energy considerations

Message Ferrying: Enforce Ferry Trajectory

- Robots, unmanned aerial vehicles (UAVs) *Li al '03, Zhao et al*



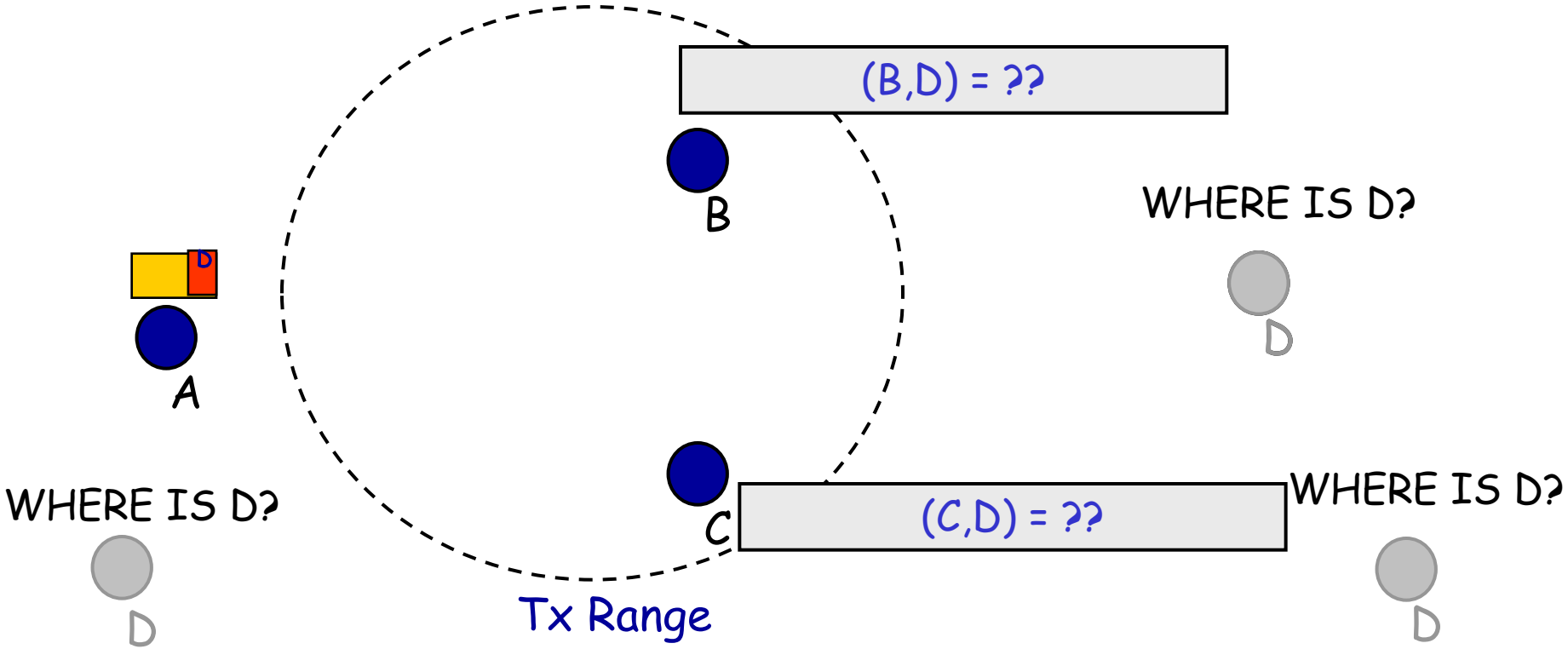
The problem: design optimal trajectories

Routing with unpredcited contacts

opportunistic

Opportunistic Routing with Unknown Contacts

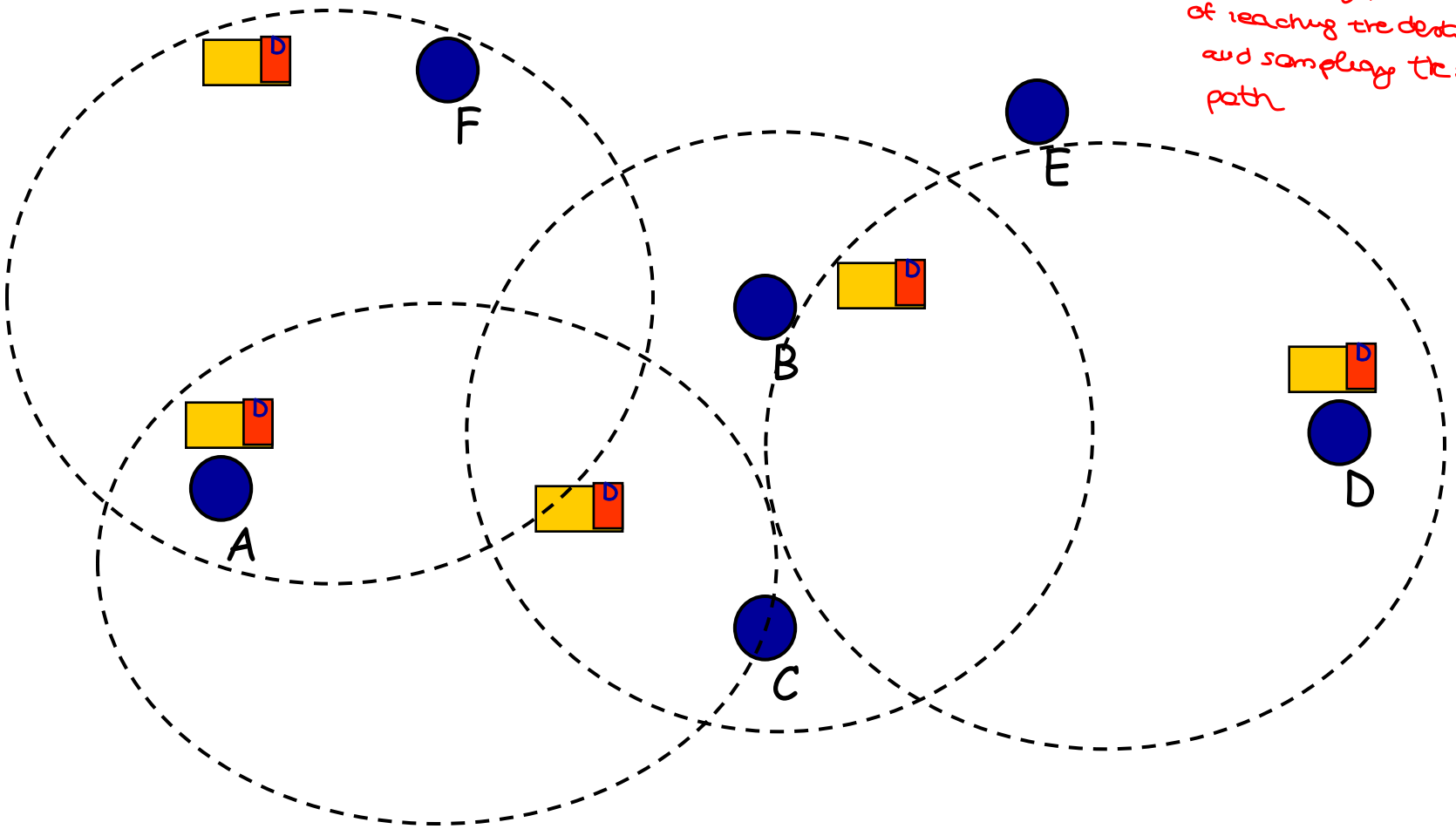
- Graph is disconnected and/or time-varying
- Set of contacts C: unknown!
- Set of nodes V: often unknown too!



Epidemic Routing

- Give a message copy to every node encountered
  - essentially: flooding in a disconnected context

flooding the network with your message, making sure of reaching the destination and sampling the shortest path



Epidemic Routing: Message Vectors

- Node A encounters node B

they infect each other if time is enough

Message Vector of A

Dest ID	Seq. Num.
D	0
G	1
F	0

(G,1)

(E,0),(F,1)

Message Vector of B

Dest ID	Seq. Num.
D	0
E	0
F	0
F	1

Epidemic Routing: Message Vectors (2)

- After message exchange

Message Vector of A

Dest ID	Seq. Num.
D	0
E	0
F	0
F	1
G	1

Message Vector of B

Dest ID	Seq. Num.
D	0
E	0
F	0
F	1
G	1

Epidemic Routing Performance

- How many transmissions?
  - At least N (number of nodes)
  - All nodes receive the message

- What is the delay?
  - Minimum among all possible routing schemes
  - If NO resource constraints (bandwidth, buffer space)
  - Otherwise it is bad … (too many replicas)

if there is no resource constraint

Modeling epidemic delay: Markov chain approach

- Assume a simple case where nodes meet each other according to a Poisson process of rate λ
  - Time between contacts of any nodes i and j is exponentially distributed with rate λ
  - One can make this rate different among nodes
- Consider a network of size N
- Suppose already n-1 replicas (n infected nodes including source)
- Mean time to n replicas is = $[n*(N-n)\,\lambda]^{-1}$

time to infect = $\frac{1}{n(N-n)\lambda}$

$n(N-n)\lambda$ infection rate → the peak is when $n = \frac{N}{2}$

- Mean time to infect the entire network:

$$\frac{1}{\lambda}\left[1/(N-1) + 1/(2.(N-2)) + 1/(3.(N-3)) + \ldots + 1/(N-1)\right] = \sum 1/(i.(N-i))$$

time to infect the third

the destination is somewhere in the middle

Modeling epidemic delay: Fluid approach

- Same assumptions, but network is very large and meeting are very frequent
- $x(t)$: Number of infected nodes up to time t

  $X(t)$: Mean of $x(t)$, $X^*(t)$: Second moment of $x(t)$
- $x(t+dt) - x(t) = 1\{\text{there is new infection}\}$ increase if there is a new infection (stochastic difference equation)
- By averaging: $dX(t) = \text{Prob}\{\text{new infetion}\} = \lambda.dt.(N.X(t) - X^*(t))$
- By assuming $X^*(t) \sim X^2(t)$, $X(t)$ becomes the solution of:

time rate number

$$dX(t)/dt = \lambda.X(t).(N - X(t)) \rightarrow \boxed{X(t) = N/(1 + (N-1).e^{-\lambda N t})}$$

hint: $u(t) = 1/X(t)$. Note the exponential increase with time.

Modeling epidemic delivery: Fluid approach

- Same assumptions
- What is the mean time to infect one particular node?
- Y = Delivery delay (random variable)
- P(t) = Prob(Y > t)
- dP(t) = - P(t).Prob(infect destination in [t,t+dt])

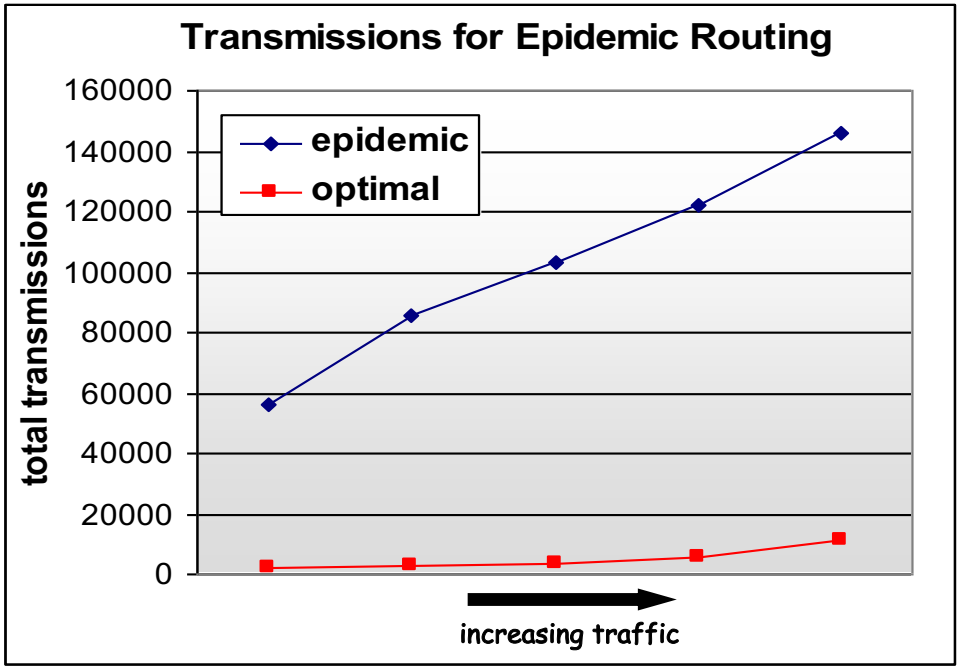
  = - P(t). λ.X(t).dt

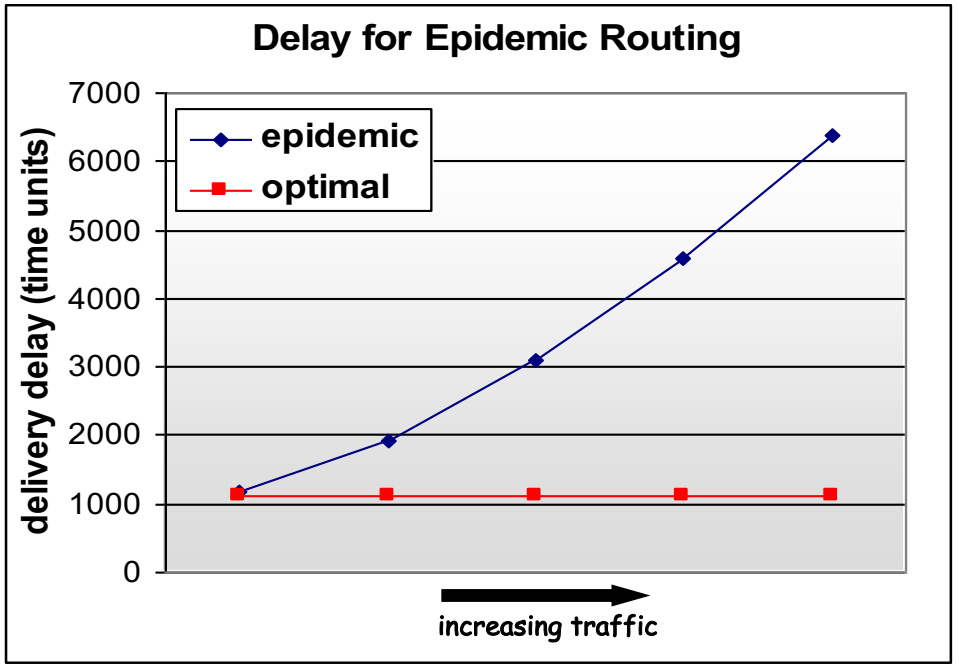
  P'(t)/P(t) = - λ.X(t) = - X'(t)/(N-X(t)) using ODE of X(t)

  thus, P(t) = 1 – X(t)/N
- E[Y] = $\int$ P(t).dt = ln(N)/(λ.(N-1))
- Moves to zero as N increases! Avoid the crowd to stay safe :)

Epidemic Routing Performance

- Redundant copies reduce delay
- But: too much redundancy is wasteful and often disastrous (due to contention)





Too many transmissions

Plagued by contention

Controlled Replication ("Spraying")

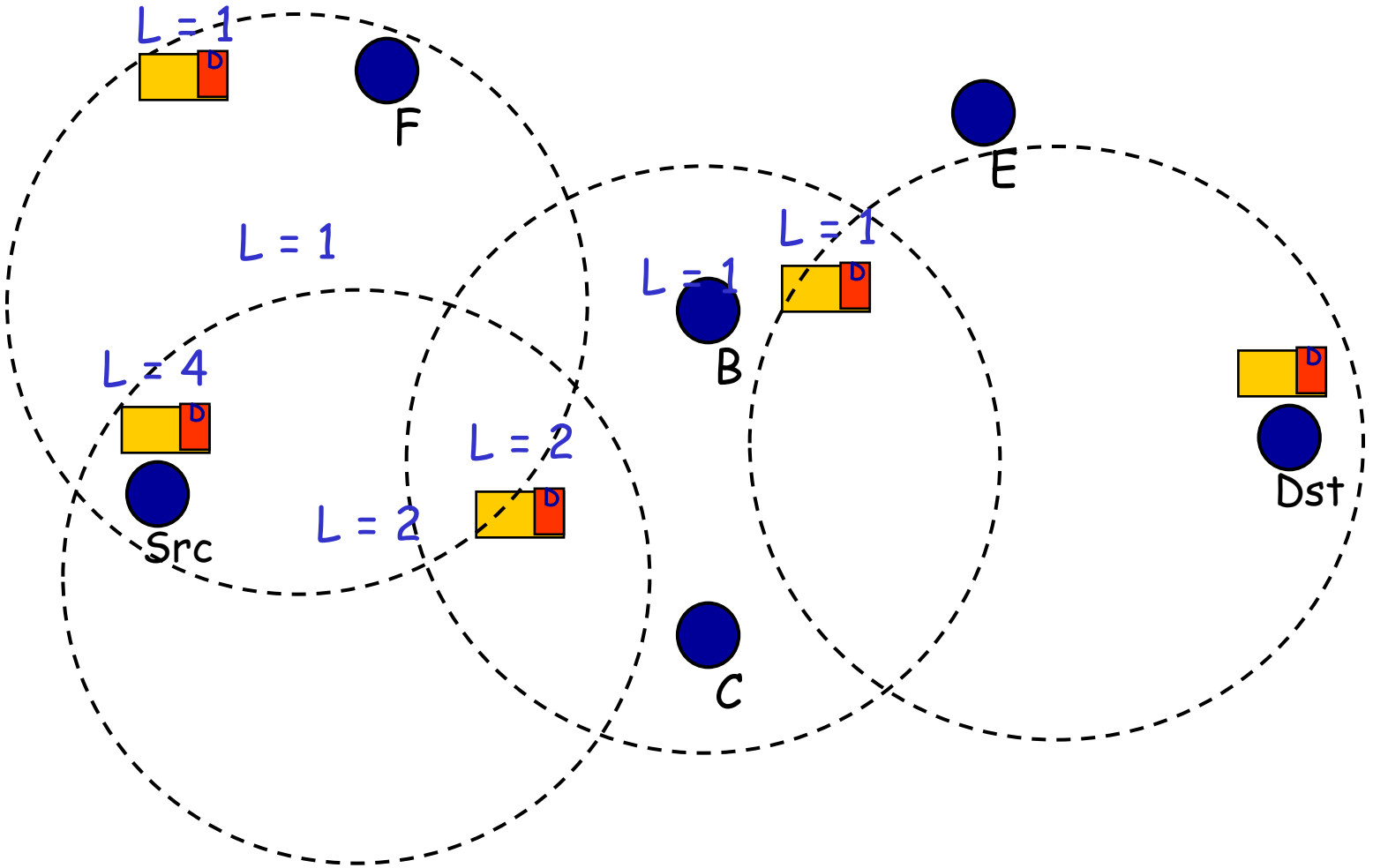
- **Limit number of copies to L** (small, fixed)
  - **Number of transmissions proportional to L**
- Give copies to the first encountered nodes
  - No sense for doing otherwise for random networks
- **L = 2)** Achieves O(1) per node capacity and deals with Kumar's and Gupta's conjecture (capacity $\rightarrow$0) (Grossglauser et al. '01)
- **L > 2 and L = O(1):** (constant L)
  - Retain capacity gain
  - Transmissions << N
  - Multi-path diversity to reduce delay (**Spray & Wait**)
  - X(t) as N - $e^{-\lambda L t}$ as P(t) as $e^{-\lambda L t}$. Not that dramatic finally.

limit the number of replication times

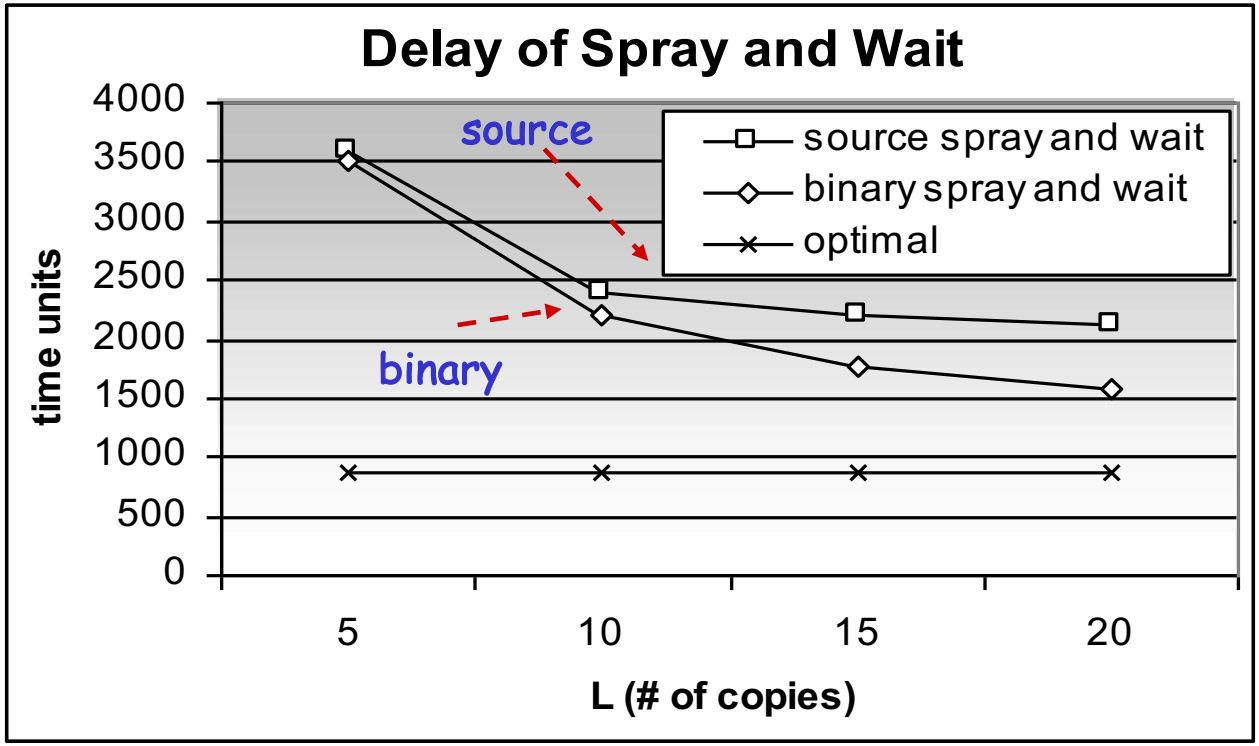
the capacity for individual is $\frac{1}{N}$, so if the network is big (N) it goes to zero

Spray and Wait (Binary Spraying)

- Use forwarding tokens; SRC starts with L tokens
- When L = 1, can only forward to destination



Benefits of Replication



100x100 grid
100 nodes
random walks

- exponential decrease of delays with the number of replicas
- the transmission capacity increases linearly

10x fewer transmissions => only 2x increase in delay

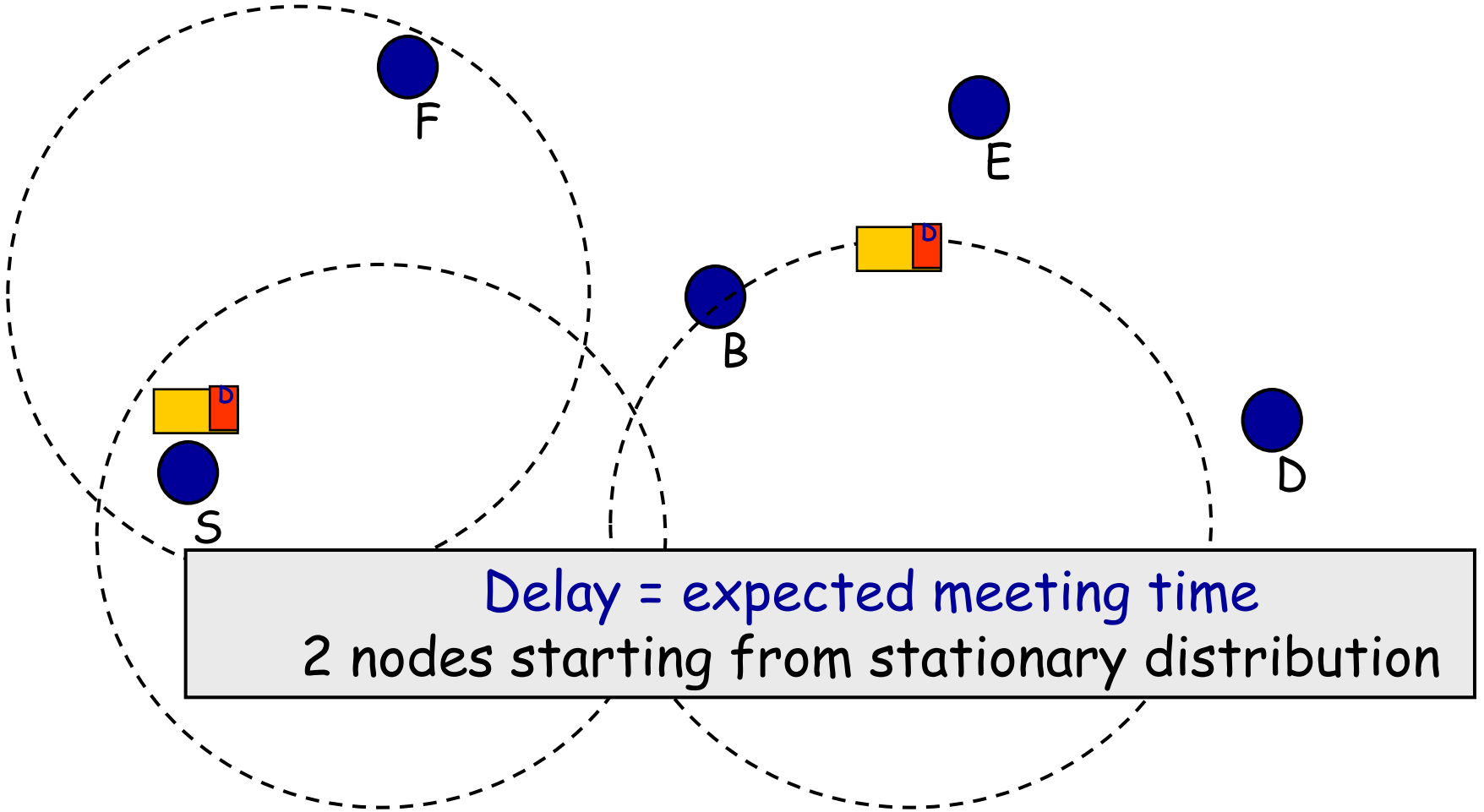
delegation needs to know that a relays is better than me

Spray & Wait: How Many Copies?

- L (e.g. L = 10) copies might be
  - Too little if number of nodes N is large (e.g. 10000) => almost no replication
  - Too many if number of nodes N small (e.g. 20) => almost epidemic
- Analytical equation for S&W delay as a function of L and N
  - Choose desired L (tradeoff delay vs. transmissions)
- What if number of nodes is unknown?
  - Estimate number of nodes online
  - Estimator of N based on sampled contacts: fish lake population estimation
  - Gossiping to converge fast

Direct transmission

- Forward message only to its destination
  - simplest strategy
  - minimizes transmissions



Opportunistic Routing $\stackrel{?}{=}$ "Random" Routing

- So far all schemes are random: no assumptions about relays
  - all relays equally fast, equally capable, similar mobility
  - No need for forwarding (giving the replica to another node)

- Is real life that random and homogeneous???
  - Nodes have different capabilities (PDA, sensor, laptop, BS)
  - Nodes move differently (vehicles vs. pedestrians, 1st year student vs. PhD)
  - Nodes have social relations
    - Same affiliation => same building, floor
    - Friends => meet more often than others
- **Learn and exploit the patterns => better routing**

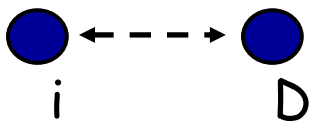
Prophet Routing: Route by estimated delivery probabilities

- Like Epidemic routing, but maintains a probability of delivery for each node pair p(i,D)
  - Probability exponentially decays with elapsed time
  - Increases when i and D meet
  - Updated when another node than D is met: p(i,D) = f(p(i,j),p(j,D)) Transitiveness of probability – kind of path probability
- Node i copies message to j only if p(j,D) > p(i,D)
- Algorithm:

Per Hop Probability

Path Probability

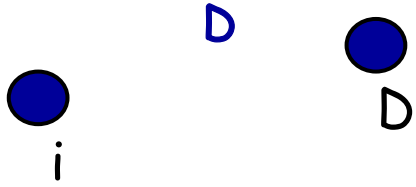
Contact with Dest D



p(i,D) = 1

when to nodes meet the probability is 1

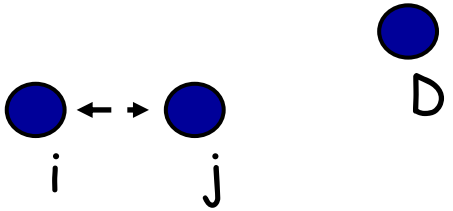
No Contact with Dest D



$p(i,D) * \gamma^t$, $(\gamma < 1)$

when nodes do not contact

Contact with j



p(i,D) = f(p(i,j),p(j,D))

Mobility Model: Important data for DTNs

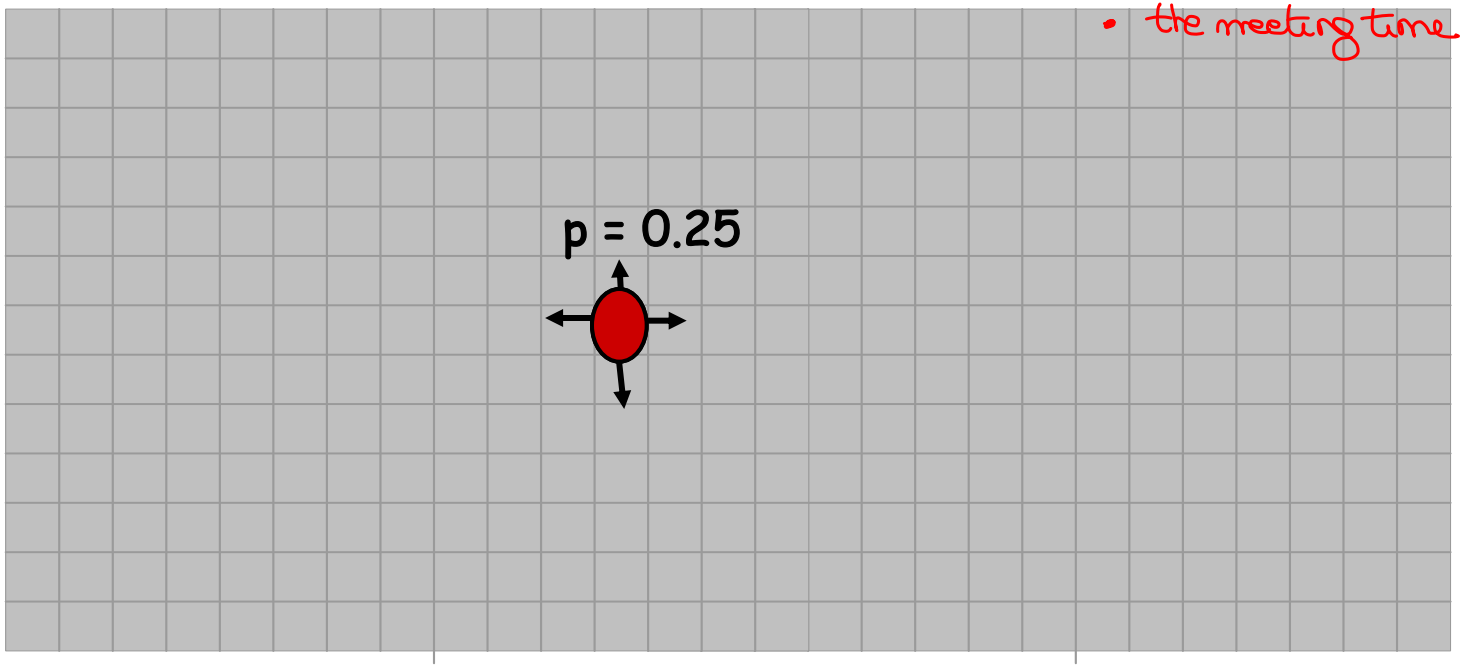
- Needed for the validation of any new proposal
  - Either for simulations
  - Or for anaylitical modeling
- Plenty of these models exist in the literature
  - No time to perform an exhaustive listing
- There is now more trend to use real traces
  - Humans, cars, animals, etc

Random Walk

at every step you move randomly in every direction

Result: after some time the node can be everywhere (equal prob. everywhere)

- All nodes perform independent random walks
- Move to any neighboring location with probability $\frac{1}{4}$

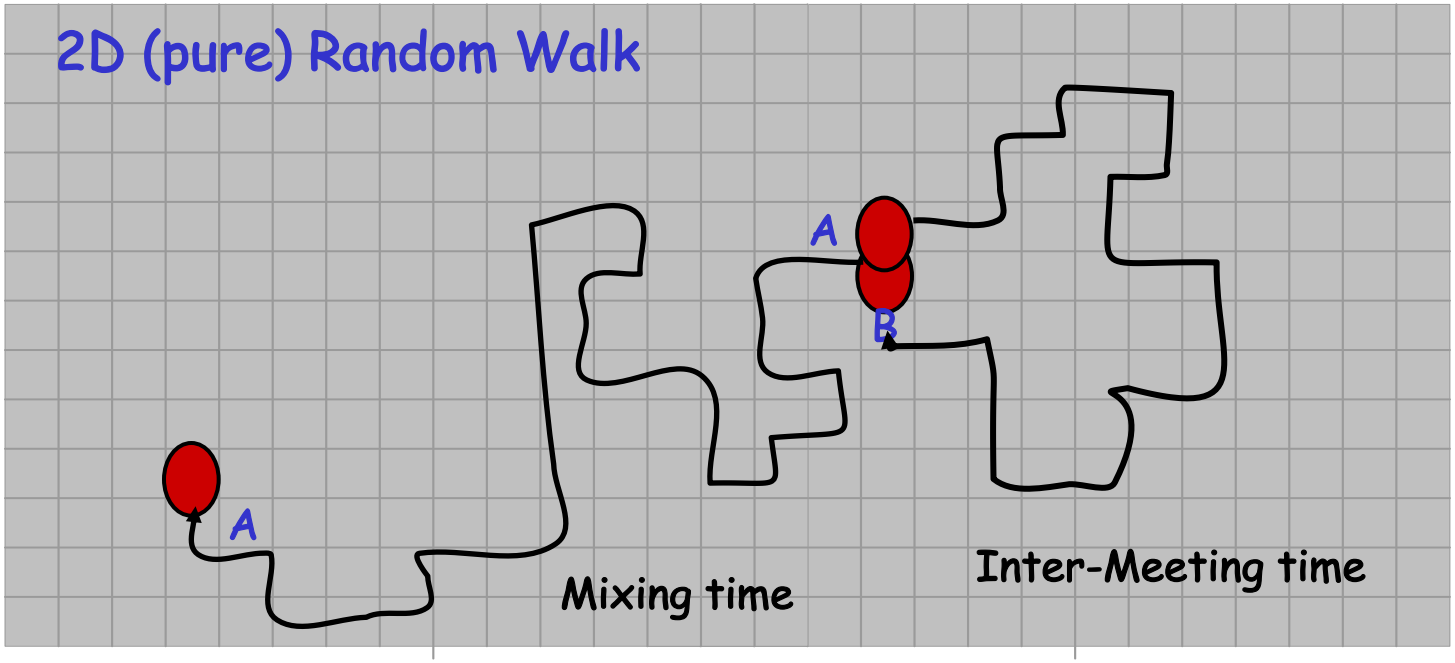


• the meeting time

the future is independent from the past

- Uniform stationary distribution
- Meeting time exponentially distributed once nodes are far enough
- There is a power-law part at the beginning when nodes split

Random Walk



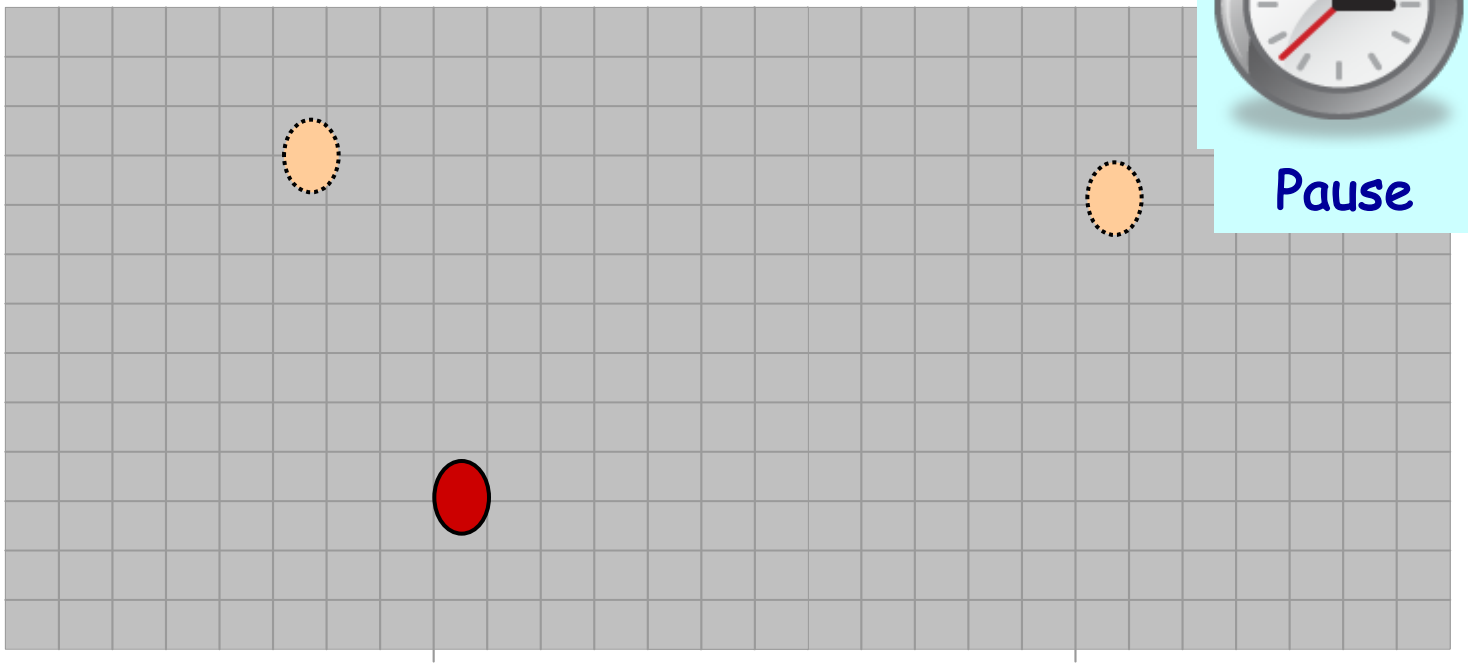
Inter-contact time < mixing time: power law

Inter-contact time > mixing time: exponential

Random Waypoint

- Choose a point in the network uniformly
  - Choose speed randomly
- Pause for a random amount of time
- Choose another point uniformly and repeat
- Tends to concentrate nodes in the middle

choose randomly a destination and stay there and then choose it again

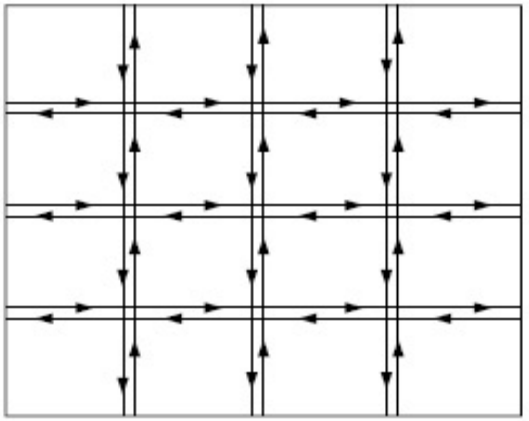


Pause

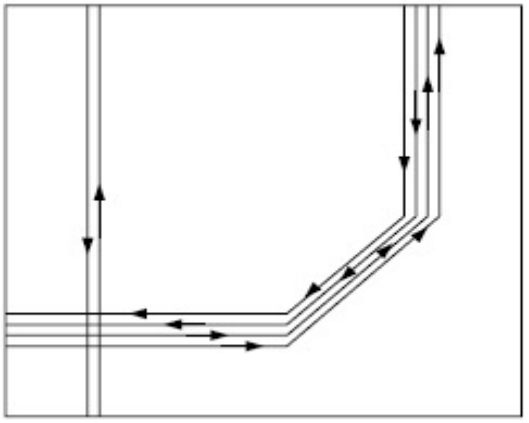
Other Models

having some determinism

- Manhattan Model
  - All nodes move within restricted street borders
  - Grid structure (vertical and horizontal streets, like Manhattan)
  - Stop lights?

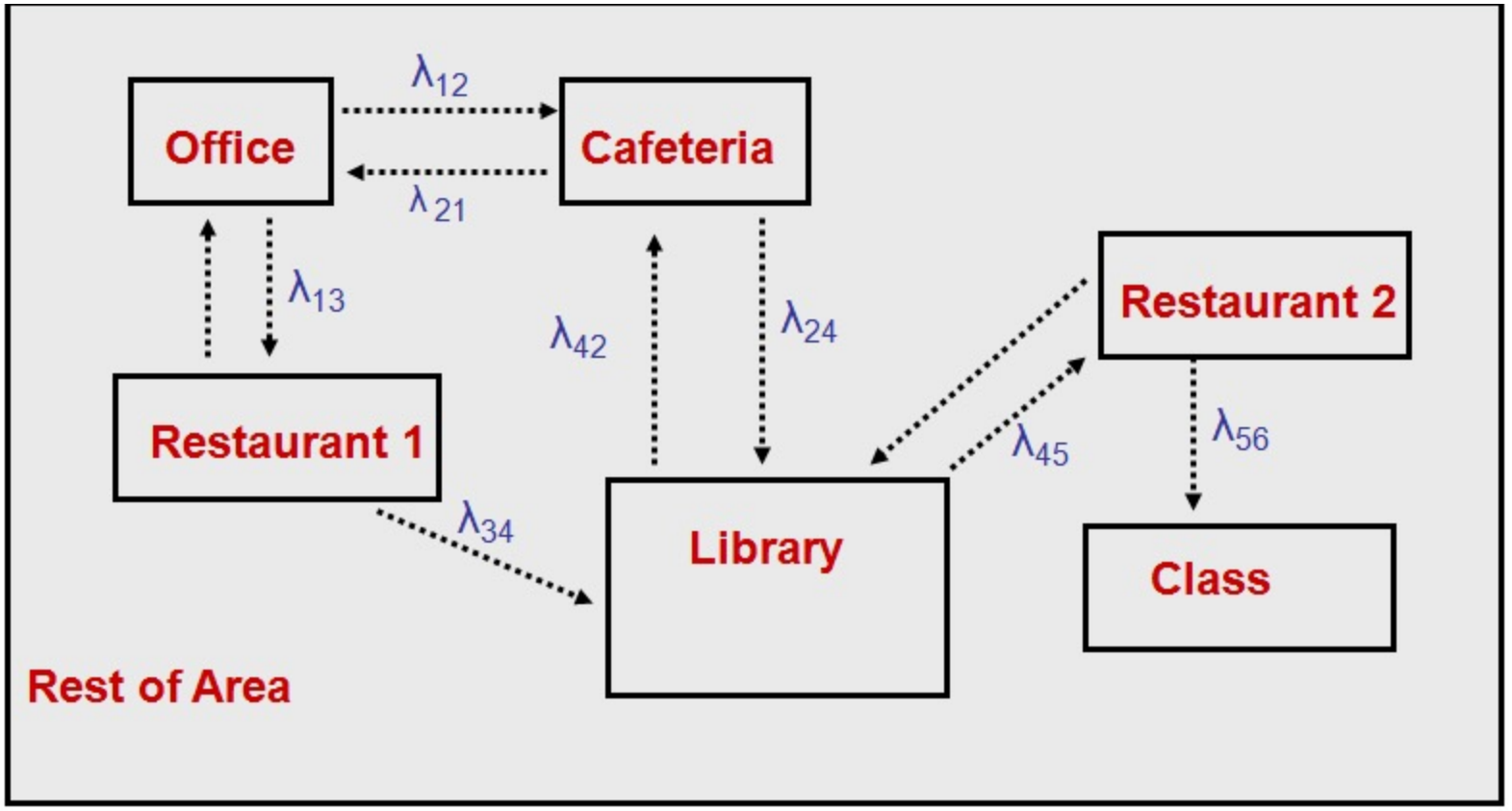


- Freeway Model
  - Nodes move on lanes of one line; lanes in both directions
  - Potentially other crossing freeways
  - Speed considerations between nodes in same lane



- Group Mobility
  - Subset of nodes associated with a leader
  - Followers make move based on leader's move

Profile-based models

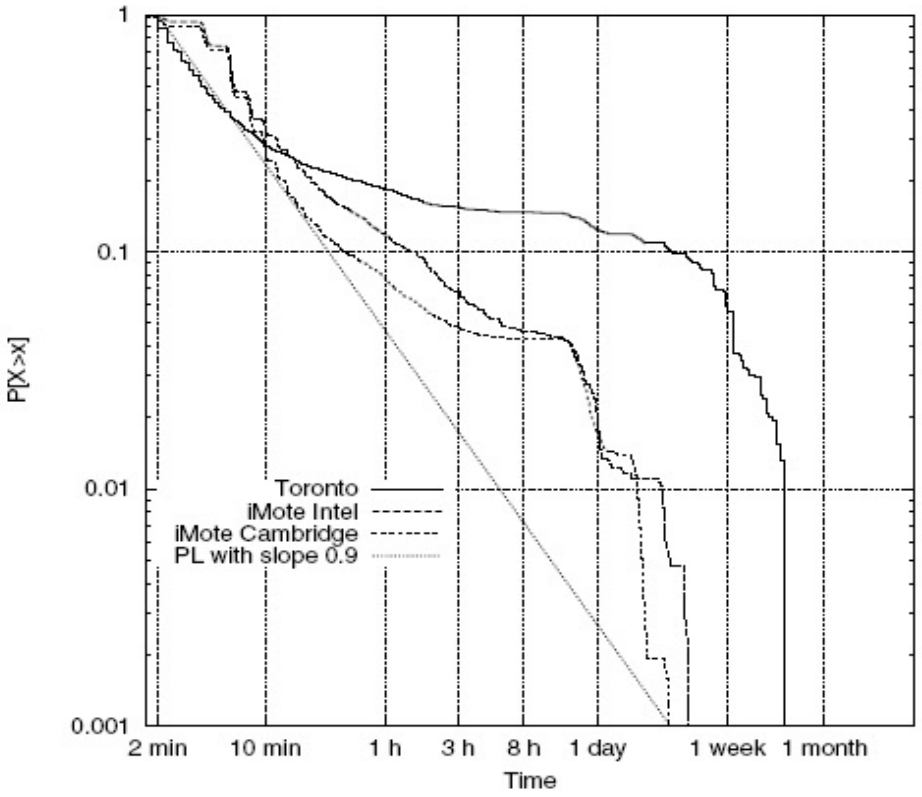


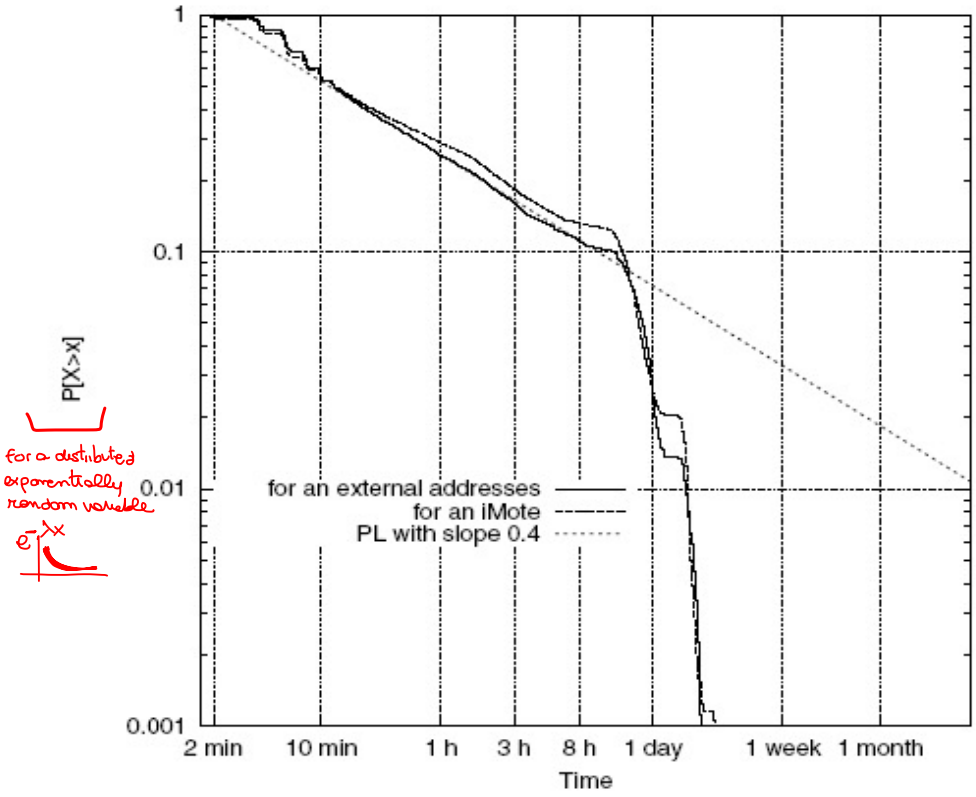
- Markov Chain model: Next location depends on current location
- The profile of someone can be inferred from social networks

Real Mobility Traces

- There is more and more trend to use real traces
- WLAN Traces
  - Associations between AP and terminal
- Encounter-based traces
  - Give wireless devices to people
  - Log encounters (associations) between them
- Vehicular Traces
  - GPS traces
- **Trace Analysis: reality is different from simple random models**
  - **Heterogeneity, time-dependency, location-preference**

CCDF for Inter-contact Times





for a distributed exponentially random variable

$e^{-\lambda x}$

Small experiments

Large experiments

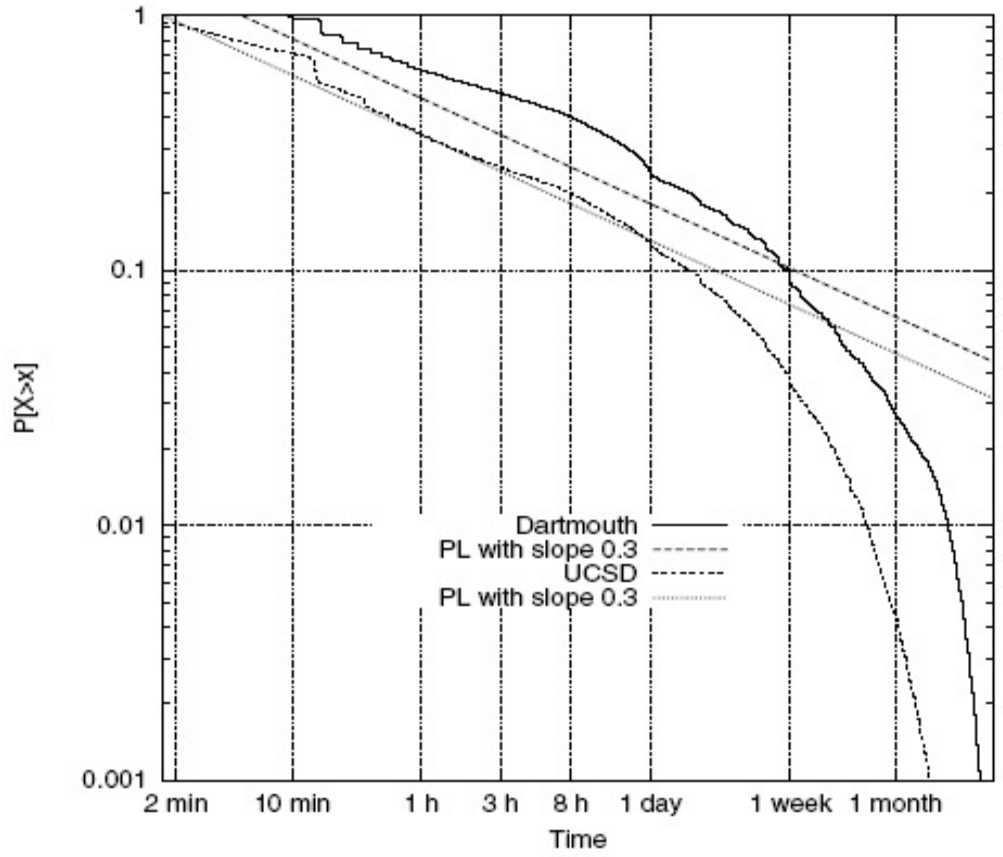
- LOG-LOG plots (log log scale)

Chaintreau et al. Infocom 2006

- Straight line in log-log plot => power law/heavy-tailed (slope = exponent)

Pareto → random variable: the tail of the distribution decreases exp fast

CCDF for Inter-contact Times (2)



- WLAN traces: similar behavior

Power Law Distributions

- $P[X > x] = x^{-a}$
- $a < 2$: Infinite variance
- $a < 1$: infinite mean
- *There is a high probability that some very large values will be drawn if X is sampled sequentially*
- Contrast: exponential decay variables
  - Very large values: almost improbable
- *The more you haven't seen someone, the less likely you will see him i.e. $P(X > x + y \mid X > y)$ increases with y*

Most of the mobility models (synthetic) so far had exponential tails (i.e. are markovians)

Power Law Distributions: Complications

- Theory: most analysis (Markov, ODEs, combinatories) assumes exponential tail
  - Essentially for $X_1, X_2, \ldots, X_n$ IID and exponential
  - $E[\min\{X_1, X_2, \ldots, X_n\}] = EX / n$

- Protocol Performance
  - Opportunistic routing: give a copy *randomly*
  - *Past to be considered (i.e. Prophet protocol)*
  - *Depending on the exponent (a) any opportunistic protocol (e.g. direct transmission, spray&wait) may have infinite delay!*

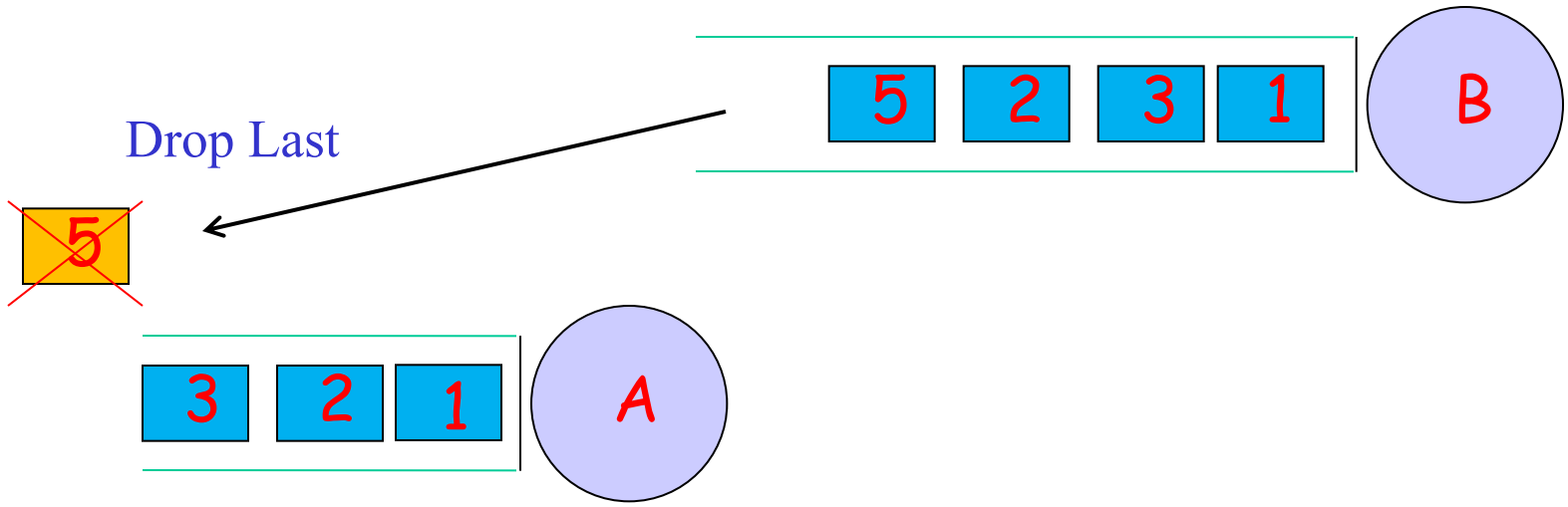
Scheduling and Drop in DTNs
Another important component

Scheduling and Drop in DTNs Another important component

- Routing limits the number of copies, but is not the final solution …
- Nodes' buffers can still overflow due to many messages.
  - Which message to drop in case of congestion ?
    - Last In ? First In ? Oldest ? Youngest ? Other ?
- Contact times can be shorter than what is needed to exchange messages between nodes.
  - In there is a way to forward the most useful messages first ?
- Two important and still open problems

Message Drop in DTNs

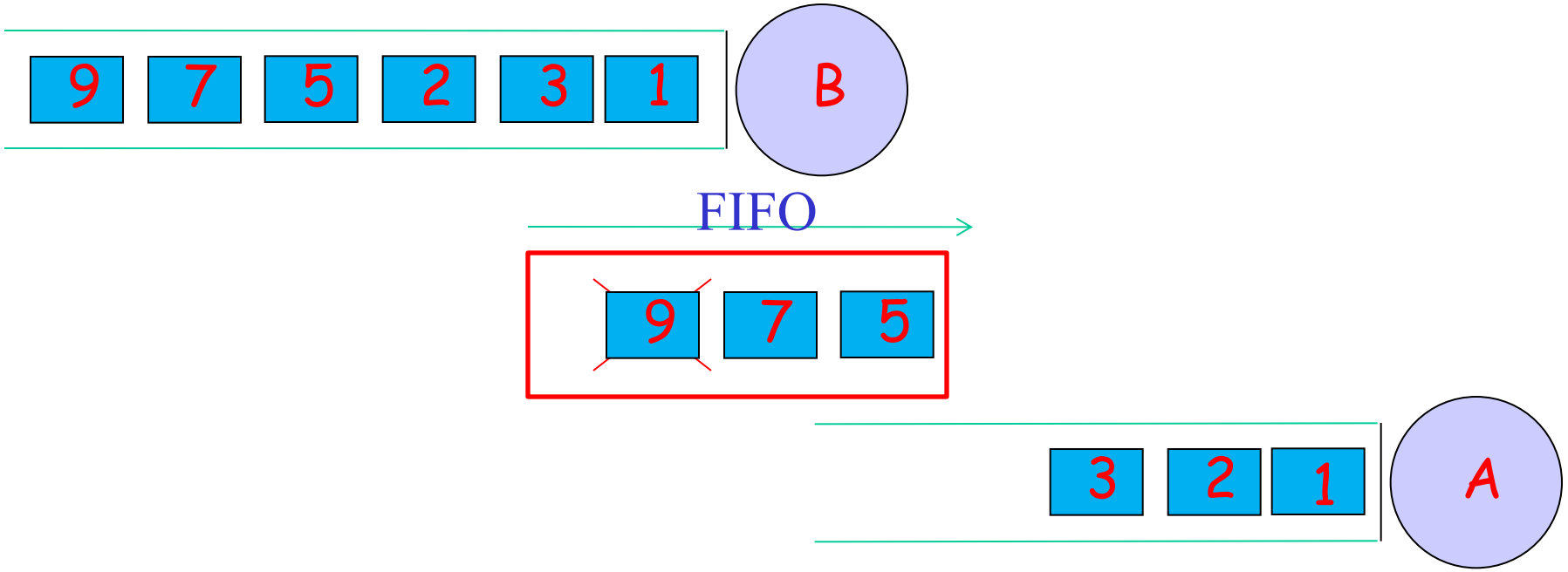
B gives A a copy of message 5 but A's buffer is full



Could be a bad decision if message 5 was a young message.

Message scheduling in DTNs

B gives A the three missing messages in FIFO order, but A leaves before 9 is sent



Could have been better to give message 9 first

State of art

- Mostly used policies: Drop Tail for queue management and FIFO for scheduling.
- Better delivery rate and delivery delay with Drop Front
  - There is a high chance that messages at the head have more copies
- No consensus on drop youngest and drop oldest.
- RAPID [Levine et al. sigcomm 2007]
  - Relevant to our work but only focuses on scheduling
  - Requires global knowledge that is supposed to be obtained by flooding (flooding may take long time in DTNs ☹)

Methodology / Assumptions

- Consider point-to-point communications.
- Suppose first global knowledge.
- Take a global routing metric as the delay or delivery rate
- Find what will be the best policy to drop and schedule.
  - Ignore collaboration among nodes.
  - Drop locally (or schedule first) the message that leads to the best marginal gain in the considered global metric.
  - Our optimization is then local (or greedy).
- Then try to estimate the global knowledge using global information BUT on old messages …

Some notations

$K(t)$	Number of distinct messages in the network at time t,
TTL_i	Initial Time To Live for message i,
L	Number of nodes in the network,
R_i	Remaining Time To Live for message i,
$T_i = TTL_i - R_i$	Elapsed Time for message i. It measures the time since this message was generated by its source,
$n_i(T_i)$	Number of copies of message i in the network after elapsed time Ti,
$m_i(T_i)$	Number of nodes (excluding source) that have *seen* message i since its creation until elapsed time Ti.
λ	Meeting *rate between two nodes* *= 1 /* average meeting time

Case of delivery rate

- Suppose each message has limited lifetime (TTL)
  - Our framework is general enough to allow different TTLs
- Suppose global knowledge on the number of copies per message (estimators to be proposed later).
- Global delivery rate at the time of congestion is then:

$$DR = \sum_{i=1}^{K(t)} P_i = \sum_{i=1}^{K(t)} \left(1 - \frac{m_i(T_i)}{L-1}\right) * \left(1 - \exp\left(-\lambda n_i(T_i) R_i\right)\right) + \frac{m_i(T)}{L-1}$$

One assumption: meeting times have an exponential tail

Case of delivery rate

We differentiate:

$$\Delta(DR)=\sum_{i=1}^{k(t)}\frac{\partial P_i}{\partial n_i(T_i)}*\Delta(n_i(T_i))=\sum_{i=1}^{K(t)}\left(1-\frac{m_i(T_i)}{L-1}\right)\lambda R_i\exp\left(-\lambda n_i(T_i)R_i\right)*\Delta(n_i(T_i))$$

The best message to drop is the one minimizing this quantity:

$$\left(1-\frac{m_i(T_i)}{L-1}\right)\lambda R_i\exp\left(-\lambda n_i(T_i)R_i\right)$$

And the message to schedule first is the one maximizing it.

This is a function of global information on message i. We call it per-message utility relative to delivery rate.

We call the resulting policy GBSD (Global knowledge based scheduling and drop)

Case of delivery delay

In the same way, one can find the per-message utility relative to the global delivery delay:

$$\frac{1}{n_i^2(T_i)\lambda} * \left(1 - \frac{m_i(T_i)}{L-1}\right)$$

To minimize the global delivery delay:

- Drop the message having the smallest utility.
- Schedule first the message having the largest utility.

For more details:

Amir Krifa, Chadi Barakat, Thrasyvoulos Spyropoulos, "Optimal Buffer Management Policies for Delay Tolerant Networks", to appear in proceedings of the SECON conference, San Francisco, June 2008.

Some observations

- The optimal decision is function of the number of copies (and elapsed time for the delivery)
  - One CANNOT easily use this result to compare the basic policies: Drop Tail, Drop Front, Drop Youngest and Drop Oldest.
- A node only needs to know the global information on the messages present in its buffer.
- More importantly, our policy does not make any assumption on the underlying routing protocol.

Distributed version: How to calculate n and m ?

- n = number of copies of a message
  m = number of nodes that have seen the message.
- Flooding does not work because it takes long time.
- Our solution:
  - Flood global information BUT on old messages.
  - All nodes have the same information after some time.
  - Estimate n and m from what has happened to old messages at the same elapsed time.
  - Assuming of course stationarity and ergodicity.

Distributed version (ctd)

- Requirements:
  - m and n are correlated, they need to be estimated jointly.
  - Estimators are to be plugged in the utility expressions.
  - We want the global network performance to be preserved
    - Same average delivery delay and same average delivery rate
- Suppose m and n follow two random variables M and N.
  Take for example the delivery rate.
  Estimators should satisfy this equation:

real delivery rate = estimated delivery rate

$$E\left[\left(1-\frac{M(T)}{L-1}\right)*\left(1-\exp\left(-\lambda N(T)R_i\right)\right)+\frac{M(T)}{L-1}\right]=\left(1-\frac{\hat{m}(T)}{L-1}\right)*\left(1-\exp\left(-\lambda \hat{n}(T)R_i\right)\right)+\frac{\hat{m}(T)}{L-1}$$

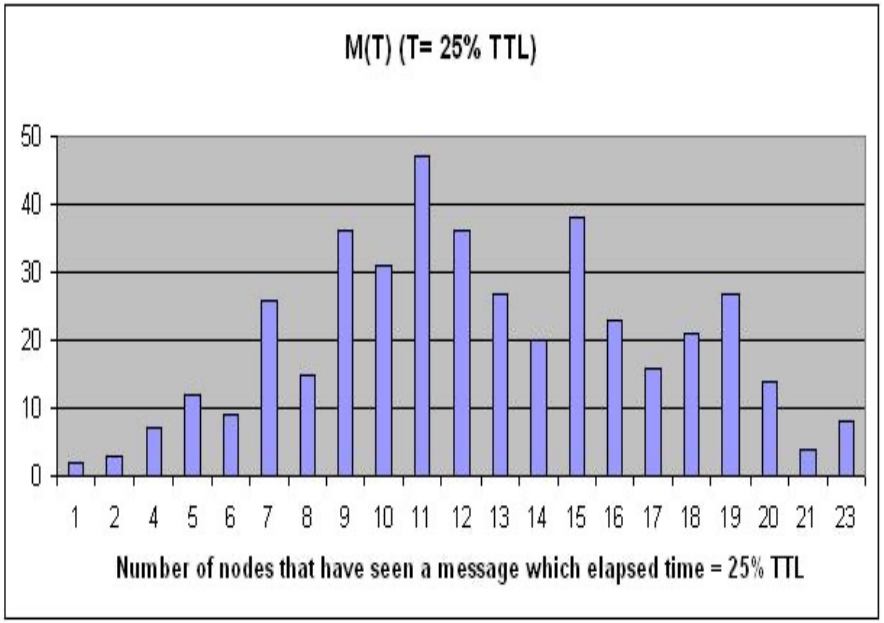
Distributed version (ctd)

- We set the estimator of m to its expectation (justified by a Gaussion distribution)
  - Another value is possible.

$$\hat{m}(T) = \overline{m}(T) = E\left[M(T)\right]$$

- We solve the previous equation to get the other estimator.

$$\hat{n}(T) = -\frac{1}{\lambda R_i} * Ln\left(\frac{E\left[\left(1-\frac{M(T)}{N-1}\right) * \exp\left(-\lambda R_i N(T)\right)\right]}{\left(1-\frac{\overline{m}(T)}{L-1}\right)}\right)$$



- Then we plug in the message utility expression

Distributed version: Message utility expressions

For the delivery rate:

$$\lambda R_i E\left[\left(1-\frac{M(T)}{L-1}\right)*\exp\left(-\lambda R_i N(T)\right)\right]$$

For the delivery delay:

$$\frac{E\left[\frac{L-1-M(T)}{N(T)}\right]^2}{\lambda*\left(L-1\right)*\left(L-1-\overline{m}(T)\right)}$$

History Based SD (HBSD)

Expectation calculated by summing over old messages.

Experimental results: Setup

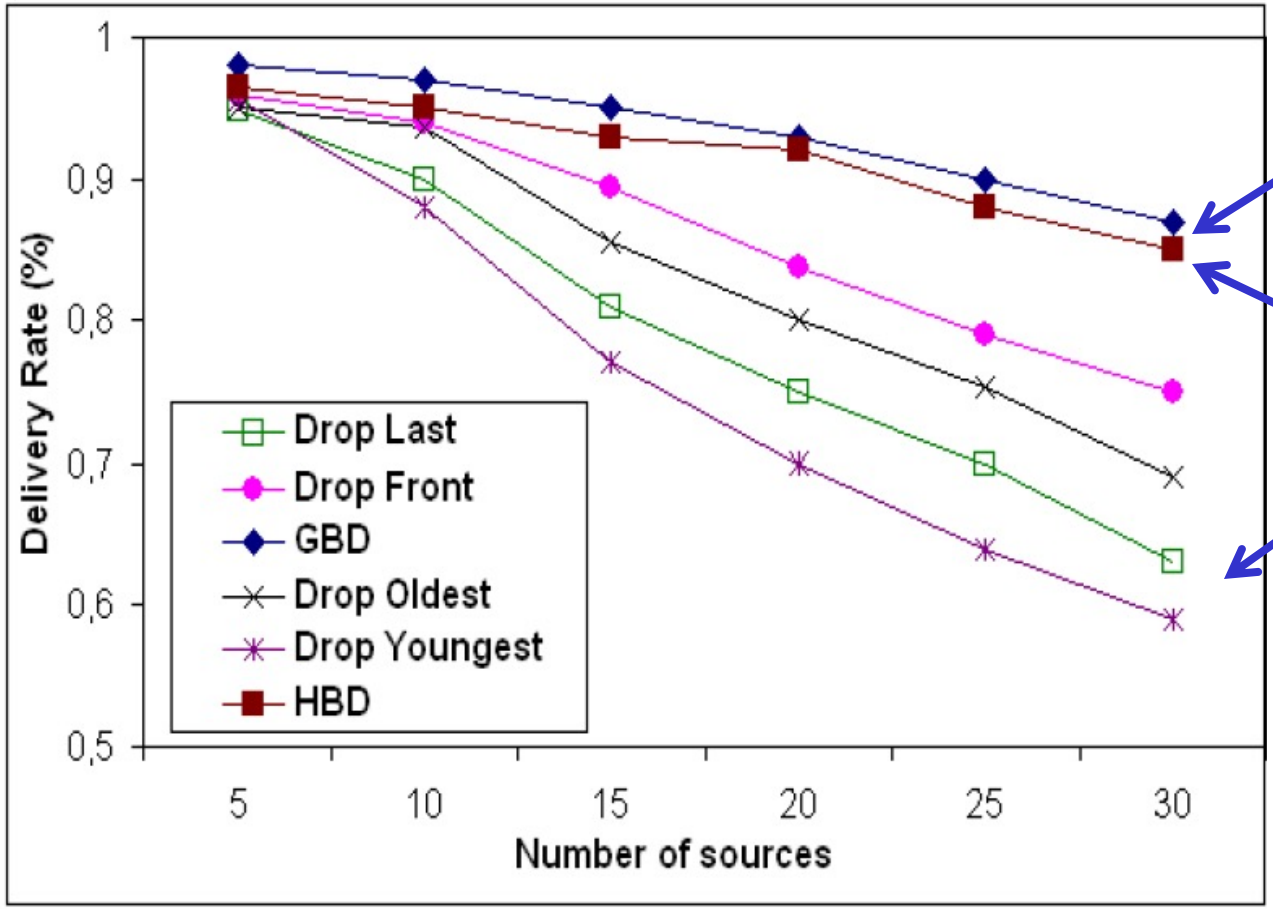
Mobility model	Random Waypoint	Traces du projet ZebraNet	Traces du projet Cabspotting
Simulation duration (s):	5000	5000	36000
Simulated Surface (m^2):	1000*1000	1500*1500	-
Number of nodes:	30	40	40
Average speed (Km/h) :	6	-	-
TTL (s) :	650	650	7200
Intervalle CBR (s) :	200	200	2100

DTN architecture added to the ns-2 simulator

MAC = 802.11b, range=100m, CBR sources, random sources and destinations

No bandwith limitation: Delivery

Small messages of 1Kbytes each (Random Way Point)

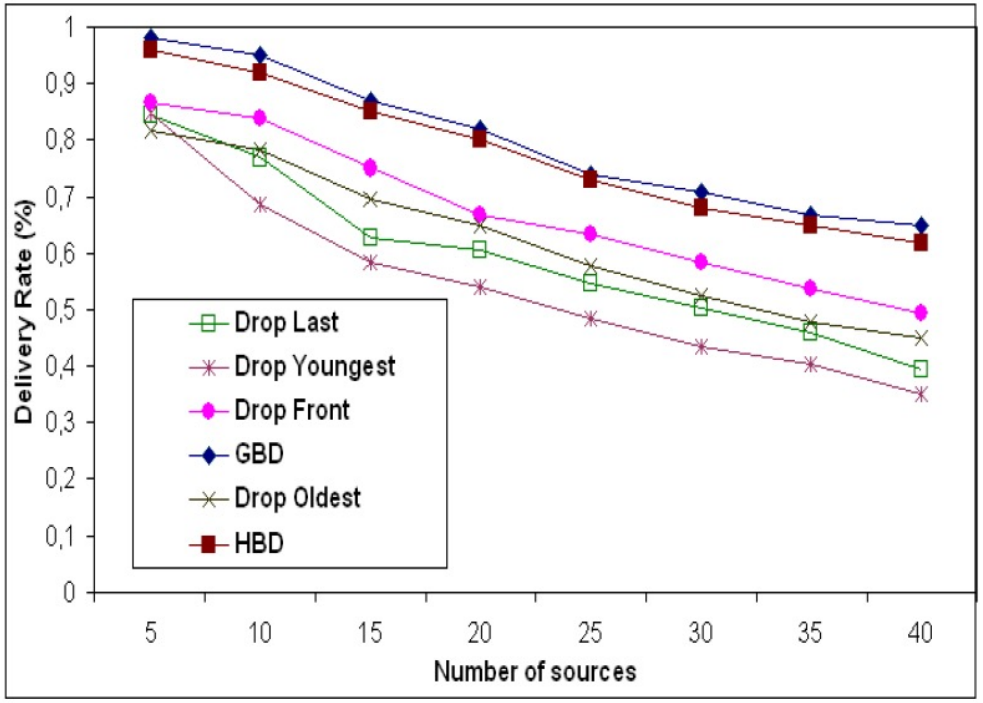


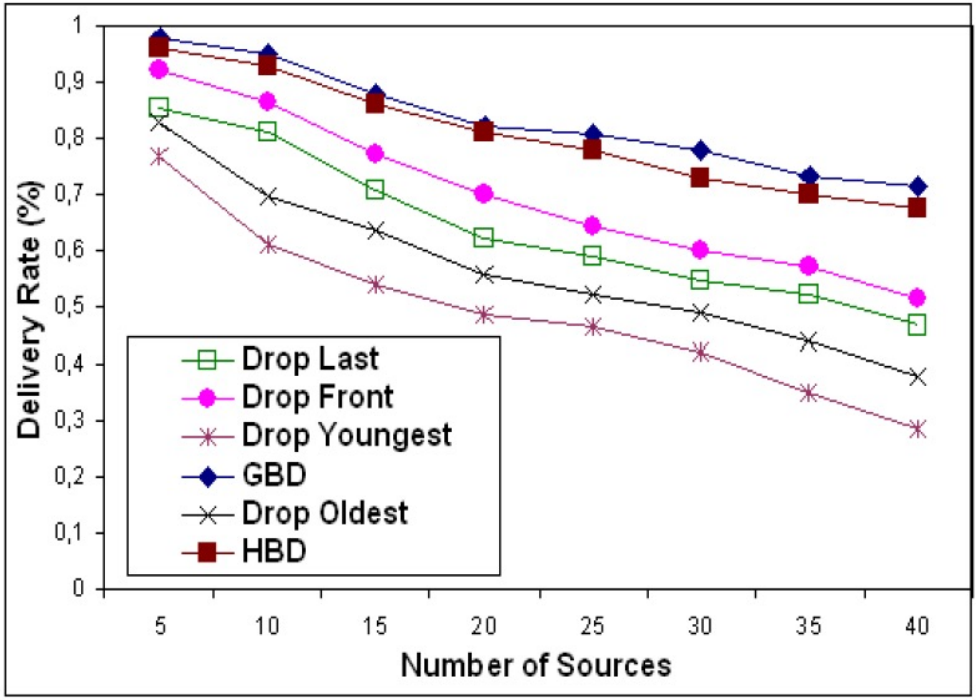
Very close

Almost 50% gain over DropTail

No bandwith limitation: Delivery

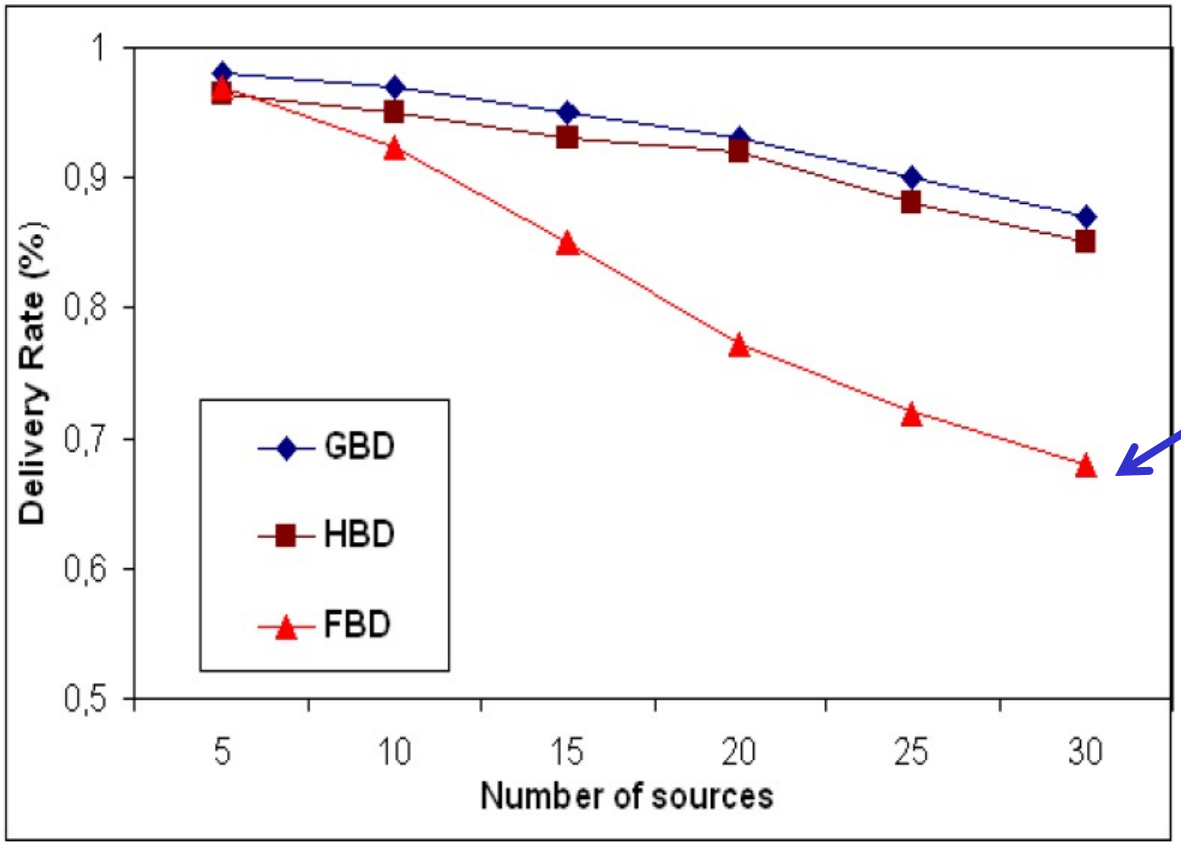
Small messages of 1Kbytes each (Real Traces)





No bandwith limitation: Flooding vs. History

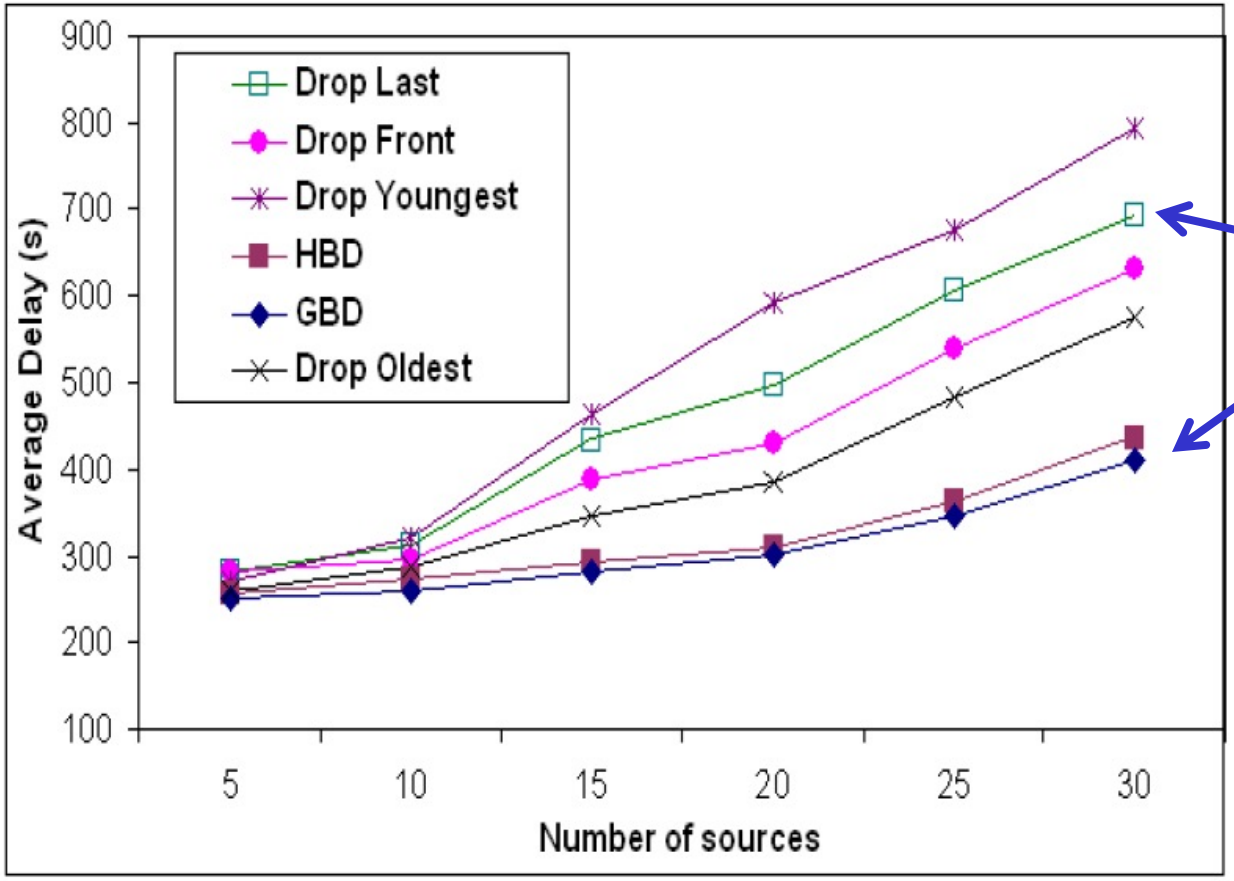
Small messages of 1Kbytes each (Random Way Point)



RAPID

No bandwith limitation: Delay

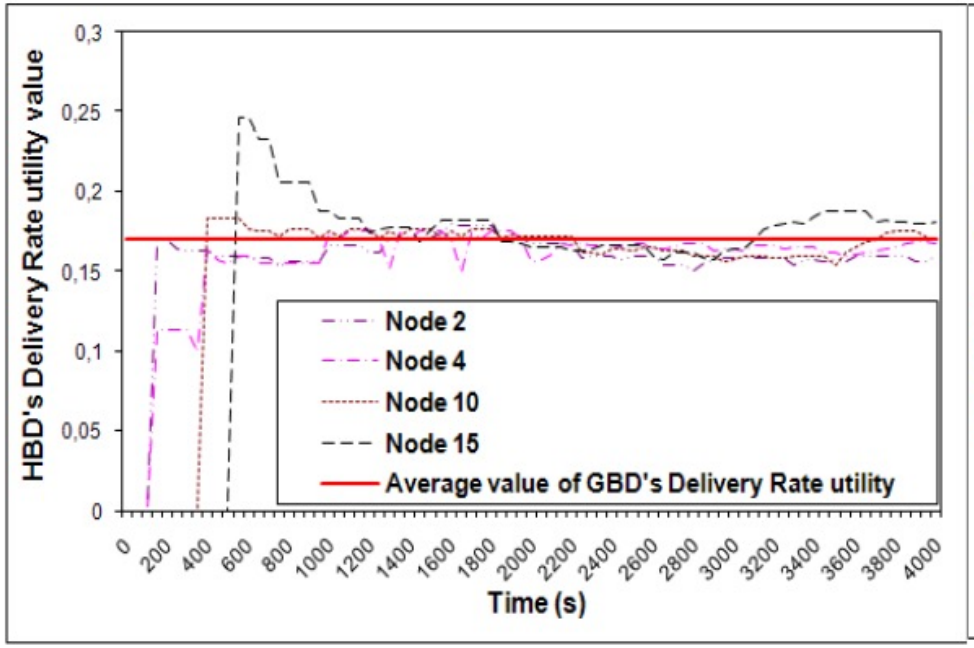
Small messages of 1Kbytes each (Random Way Point)



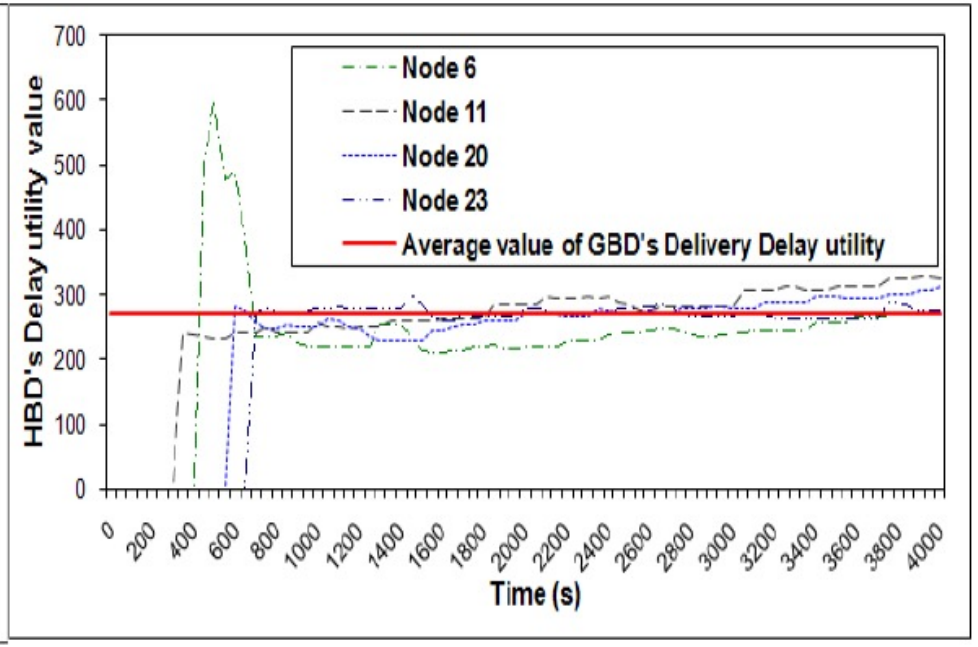
Again, almost 50% gain

Utility values convergence

Small messages of 1Kbytes each (Random Way Point) (for elapsed time T = 100s)



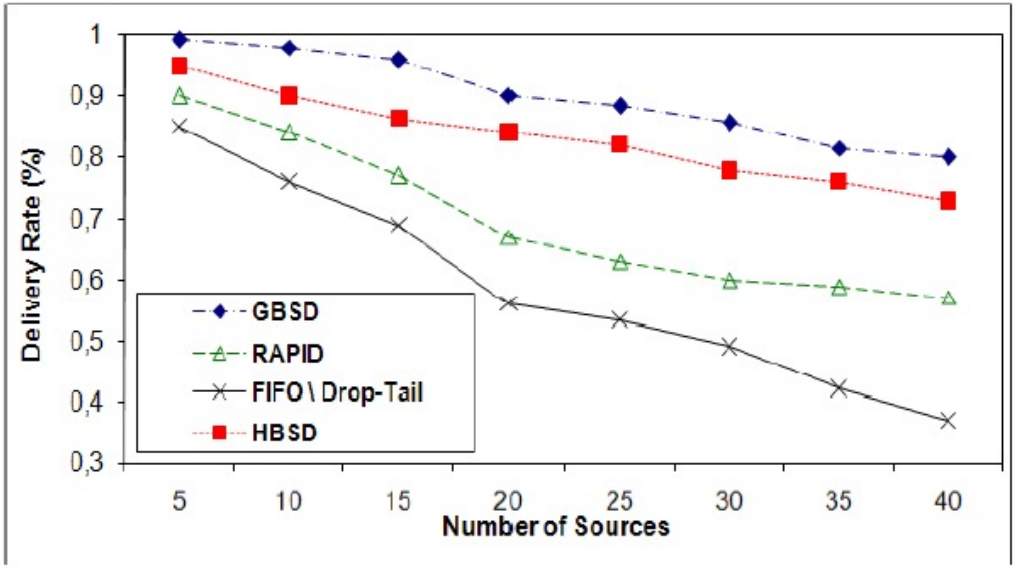
Message Utility for delivery rate



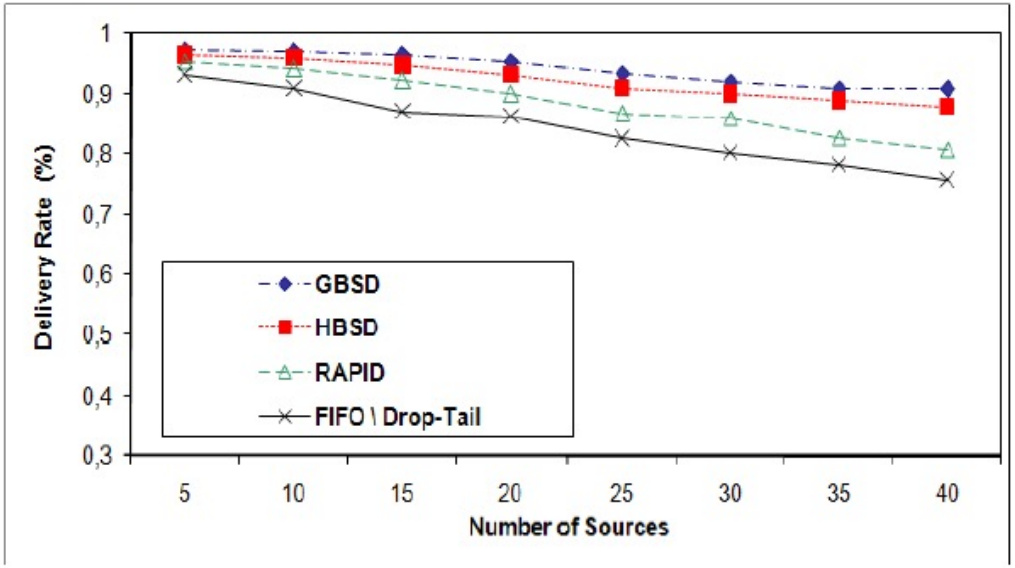
Message Utility for delivery delay

Bandwith limitation: Delivery

Large messages of 85 Kbytes each (Taxi trace)



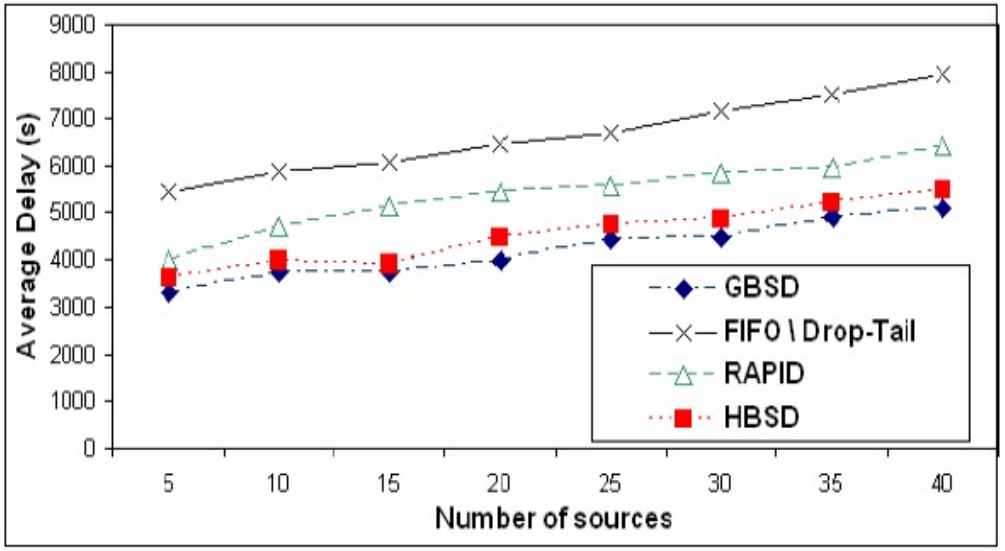
Limited Buffer



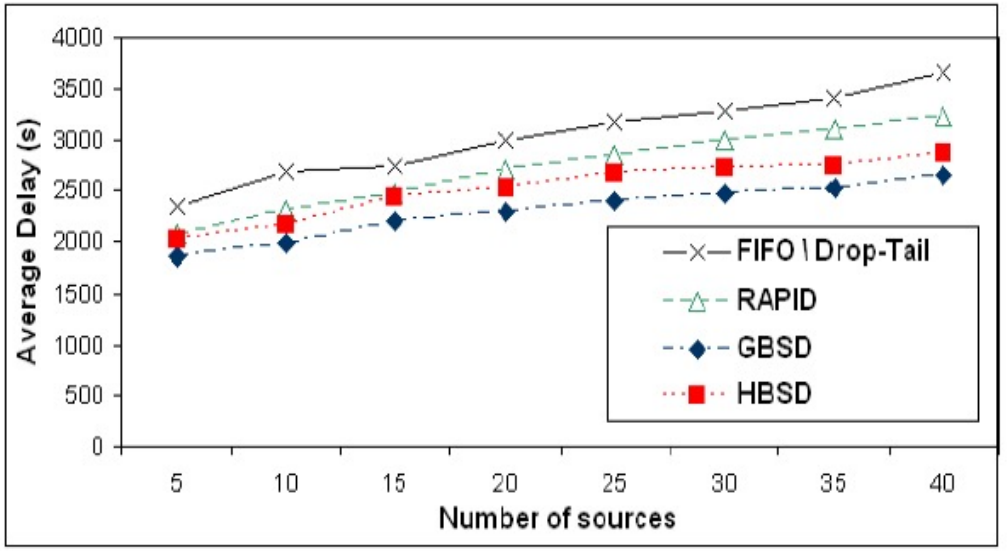
Unlimited Buffer

Bandwith limitation: Delay

Large messages of 85 Kbytes each (Taxi trace)



Limited Buffer



Unlimited Buffer

For more details on the bandwith limited case:

Amir Krifa, Chadi Barakat, Thrasyvoulos Spyropoulos, “An Optimal Joint Scheduling and Drop Policy for Delay Tolerant Networks”, to appear in proceedings of the WoWMoM Workshop on Autonomic and Opportunistic Communications, Newport Beach (CA), June 2008.

Conclusions

- An interesting architecture
- Open the door for new applications
- And networking in challenging environments
- Still many open issues:
  - Collabration? Encentives?
  - Security? Everything is distributed …
  - Fairness? How to split the load
  - Congestion control? at some time we should stop injecting new messages into the network
  - Connectivity with the current Internet
  - Large scale experimentation

Additional slides

- For further information and details
- Not included in the course

DTN Routing

- Graph is disconnected and/or time-varying
- $G(t) = \{V, E(t)\}$
- $G = \{V, C\}$, C = set of contacts c_i
- $c_i = \{v_i, v_j, t_{start}, t_{finish}, \text{bandwidth}, \text{prop. delay}, \ldots\}$

Static Nodes + Single Ferry

- b_{ij} = traffic (rate) requirement from node i to j
- Ferry route L of length |L|
- Ferry speed f: ferry cycle T = |L|/f
- d_{ij}^{L} = average delay for traffic from i to j
  - Wait for ferry: T/2 on average
  - Upload data (queuing at node): f (ferry in range, upload rate)
  - Wait for destination (on ferry): f (destination between i and j)
  - Download data to recipient: f(ferry in range, download rate)
- $d^{L} = \frac{\sum_{i,j} b_{ij} d_{ij}^{L}}{\sum_{i,j} b_{ij}}$ average delay for all traffic

Static Nodes + Single Ferry (2)

Problem: find trajectory L, such that:

- $\min_L d^L$ (Delay Problem)

- while satisfying traffic matrix $B = \{b_{ij}\}$ (Bandwidth Problem)

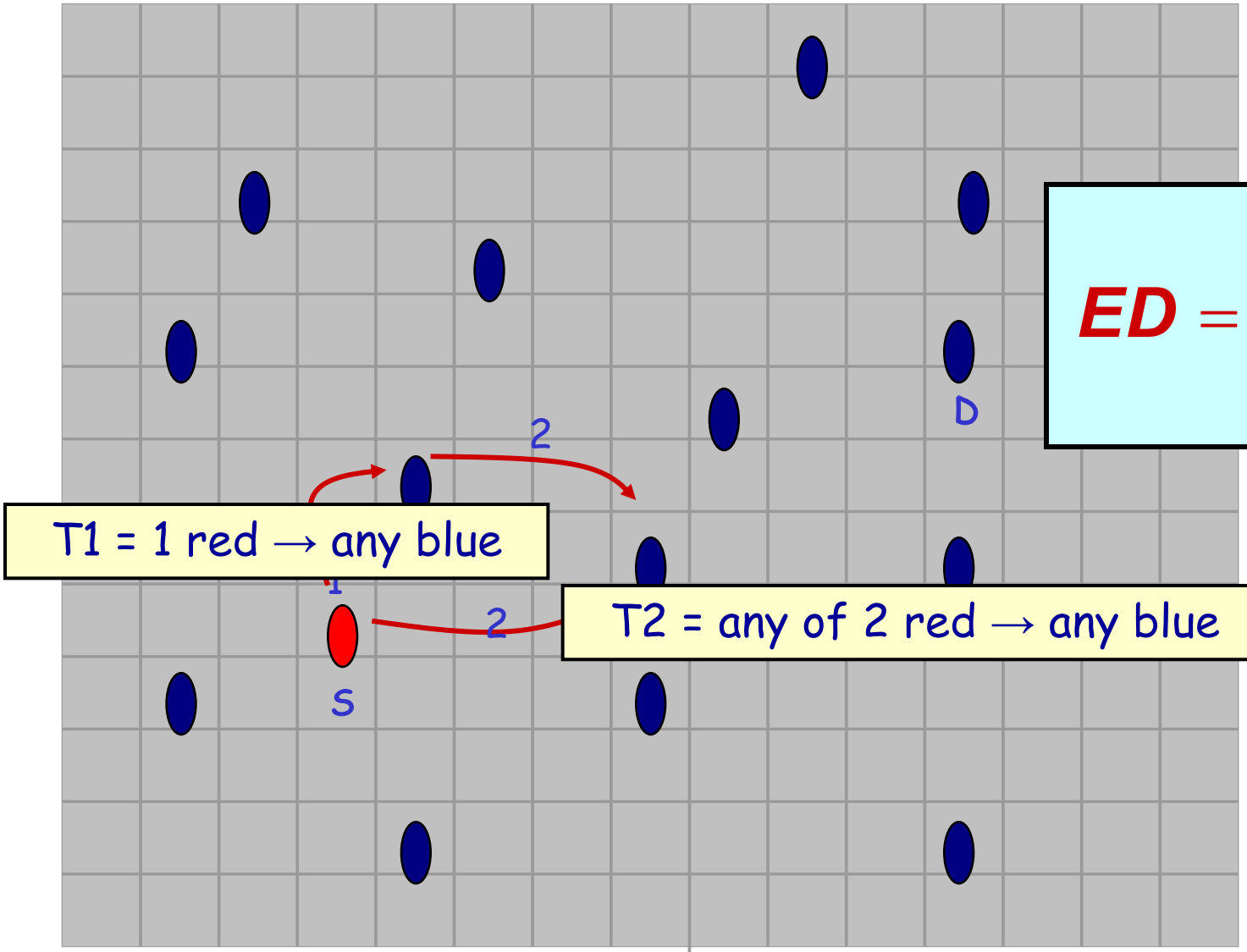
Delay problem:

Assume infinite/enough bandwidth for b_{ij}

- All data uploaded when encountered
- $\min_L d^L$ such that L passes by all nodes

Delay Problem = Traveling Salesman Problem (NP-complete)

Delay of Epidemic Routing



$$ED = \frac{1}{N-1}\sum_{K=1}^{N-1}\sum_{i=1}^{K} T_i$$

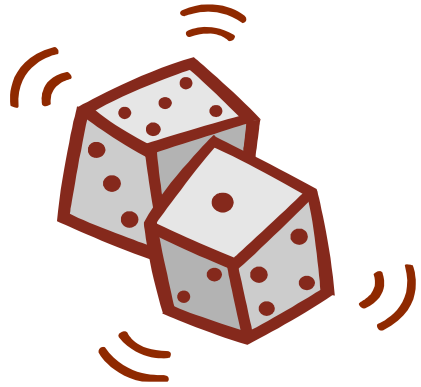
N nodes

I.I.D. mobility

Reducing the Overhead of Epidemic

Randomized Flooding ("Gossiping")

- "Spread" the message with a probability $p \leq 1$
- p = 1) epidemic
- p = 0) direct transmission



+ fewer transmissions

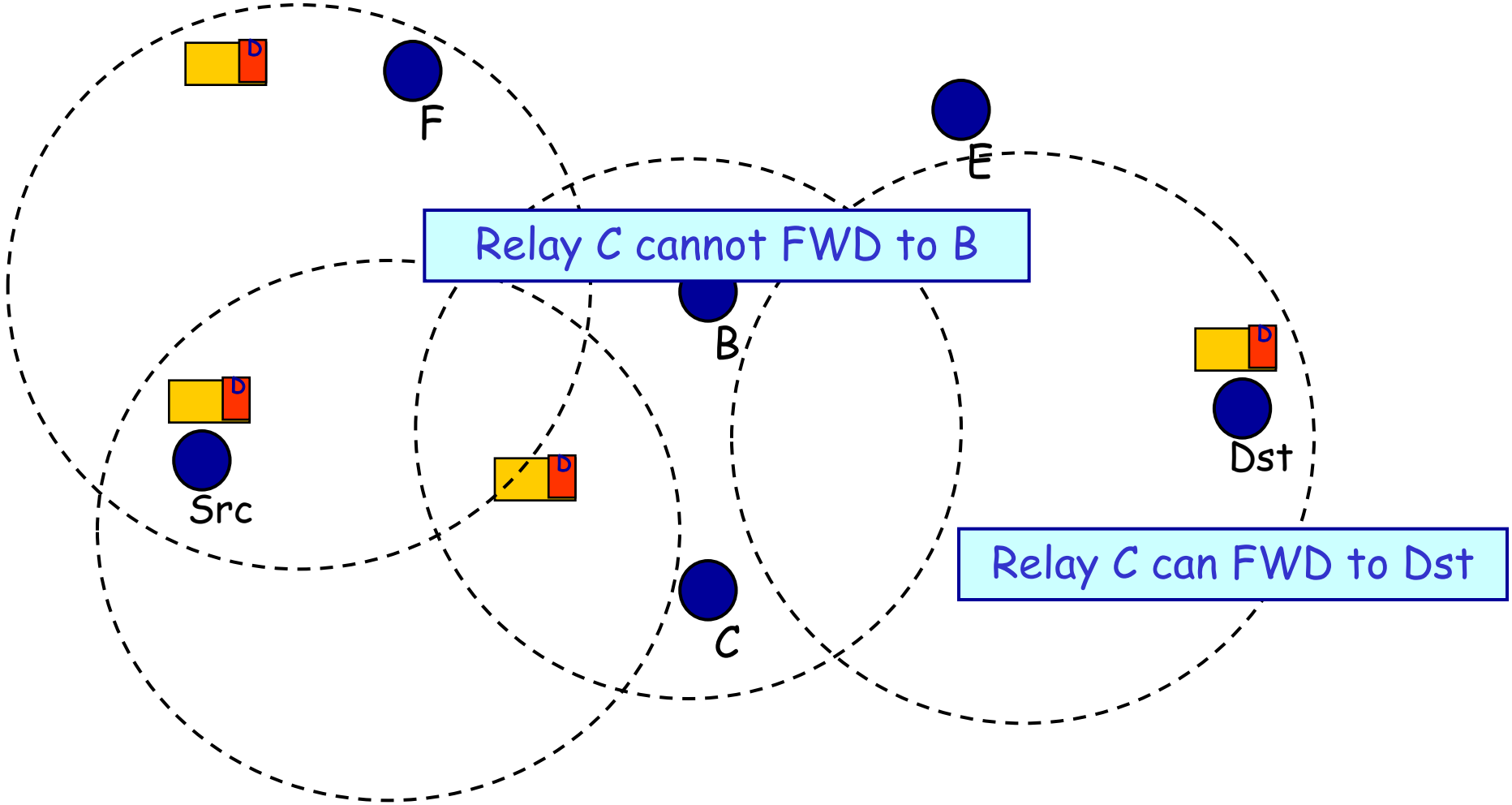
- Long delay and number of transmissions O(M) if we are not lucky

Other flooding-based variants:

- each node forward up to K_{max} times
- self-limiting epidemic (SLEF)

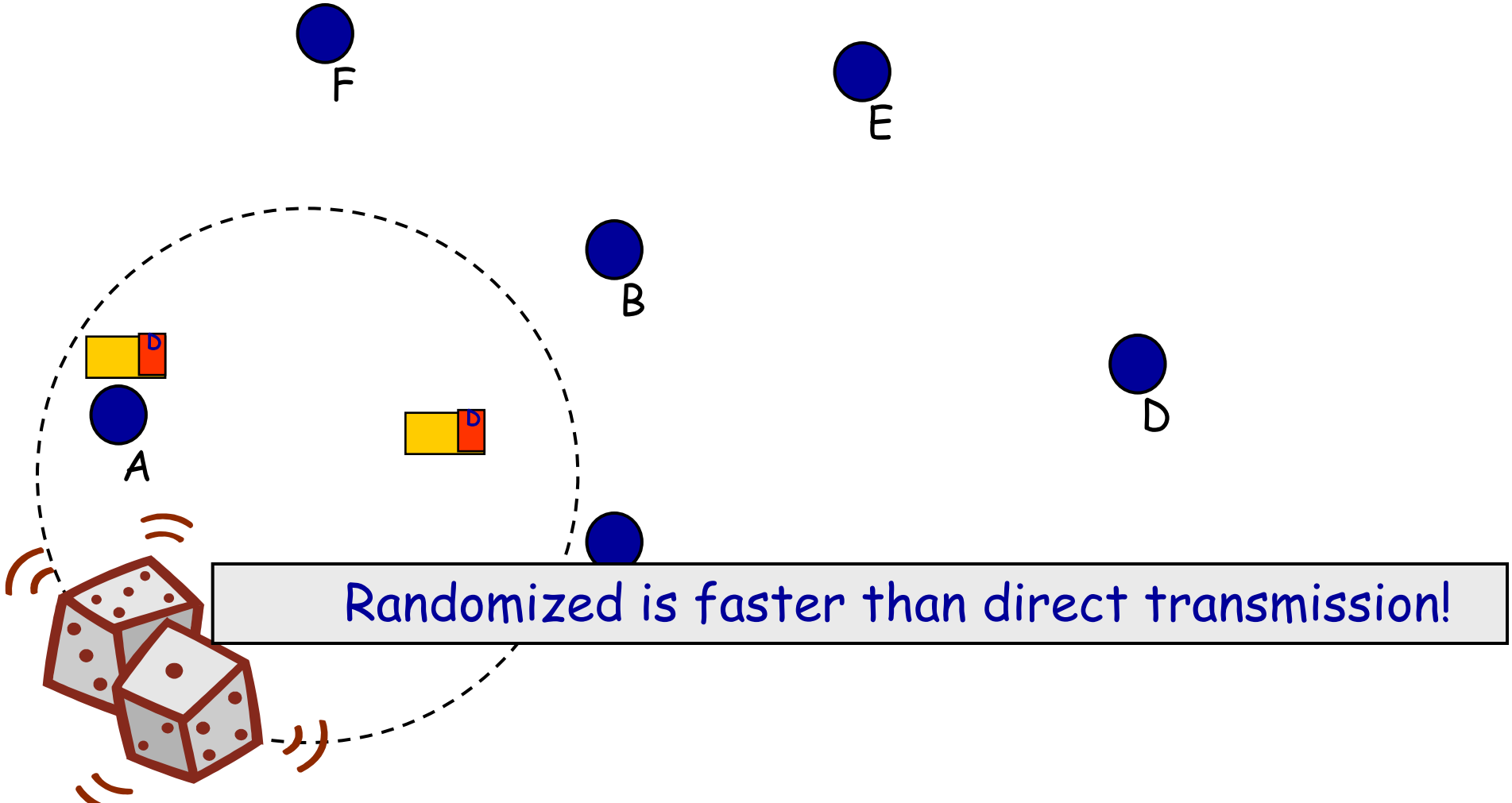
2-hop Scheme

- Source gives a copy to any relay encountered
- Relays can only give copy to destination

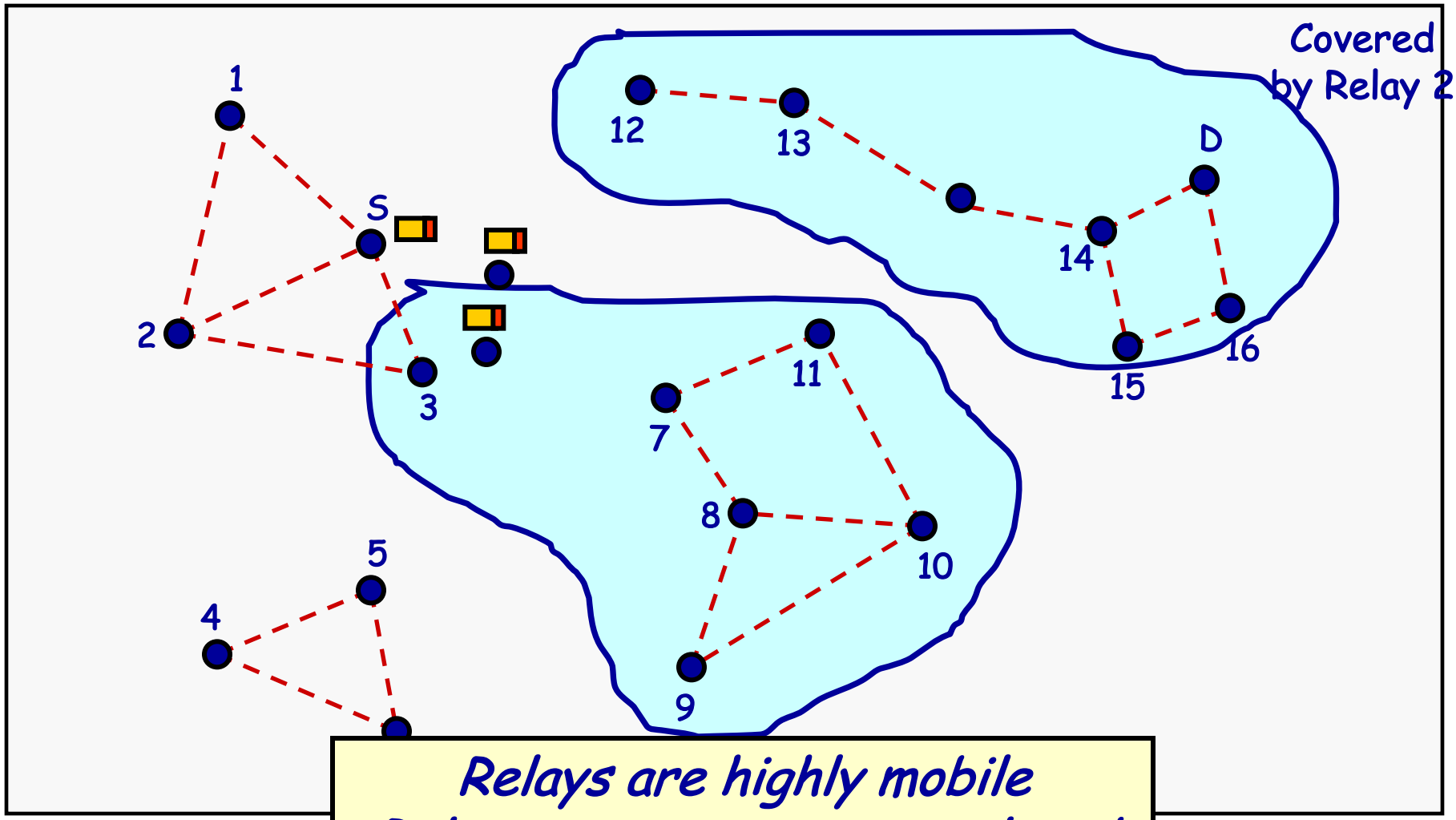


Randomized routing

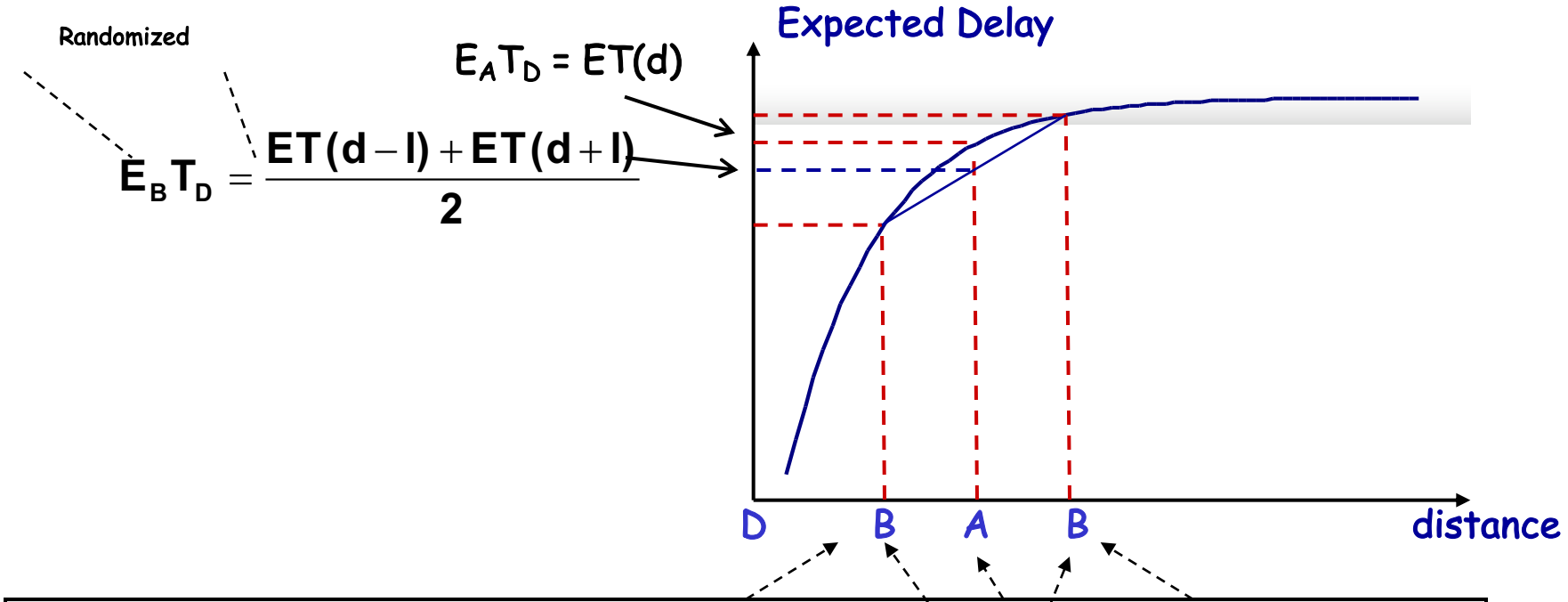
- A node forwards message to a new node with probability p; NO Duplication! It's Hand-over!



Spray and Wait: A "good" scenario

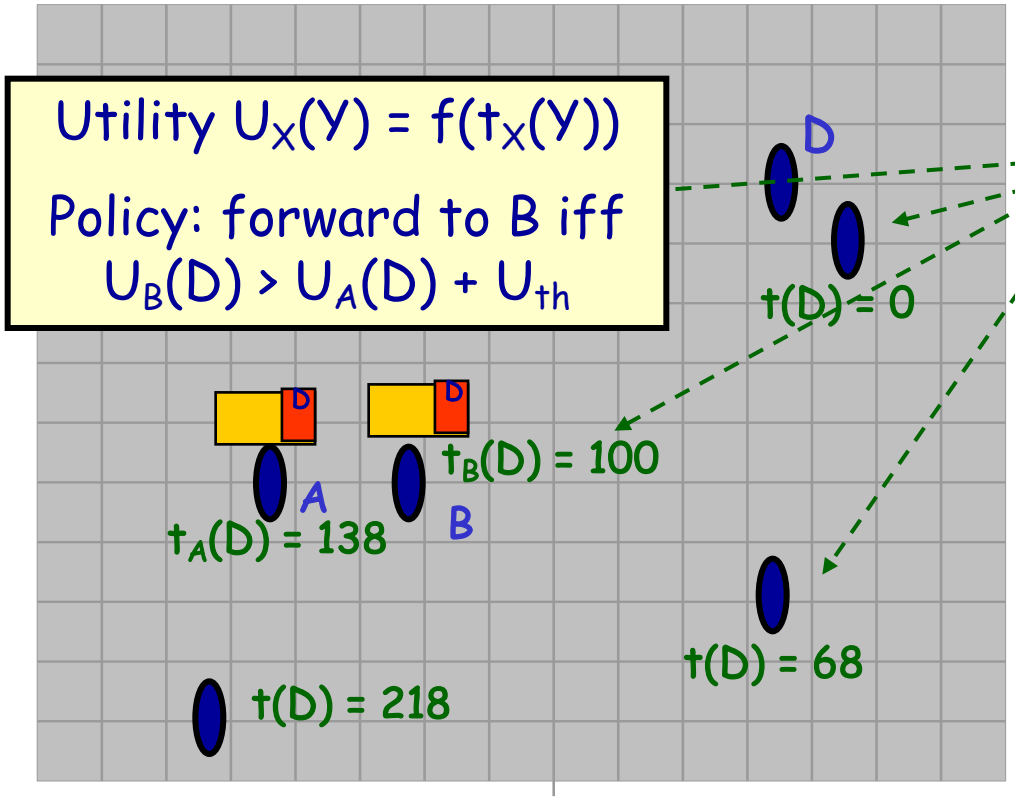


Why Transmitting is Faster Than Not!



Transmitting is better than not
if Tx range is greater than mean hop step,
if network is not infinitely sparse (partial paths)
mobility is IID

A Simple Example: Age of Last Encounter



Last encounter timers

$t_X(Y)$: time since X last saw Y

Indirect location information

- diffused with node mobility

smaller timer ⇒ closer distance

- For most mobility models

Age of Last Encounter Routing

- Flooding) duplicate only to nodes who have a smaller timer for the destination
- Single-copy) handover to node with smaller timer
  - Better than randomized (per step)
- Spraying) better than simple spray & wait

MaxProp: Schedule by estimated delivery probability

- Flood-based forwarding like Epidemic
- Which messages to schedule first?
- No time notion
- Each node estimates the likelihood it meets each other node
  - P(i,j) = Probability that next node met by node i is node j
- Broadcast the P(i,j), and build the meshed graph having P(i,j) as directed link cost
- Algorithm at node i:
  - Calculate shortest paths from i to all other nodes
  - Schedule messages by increasing path lengths
- See it as minimizing the delivery delay

Alternative Ways to Reduce Overhead

Anti-packets

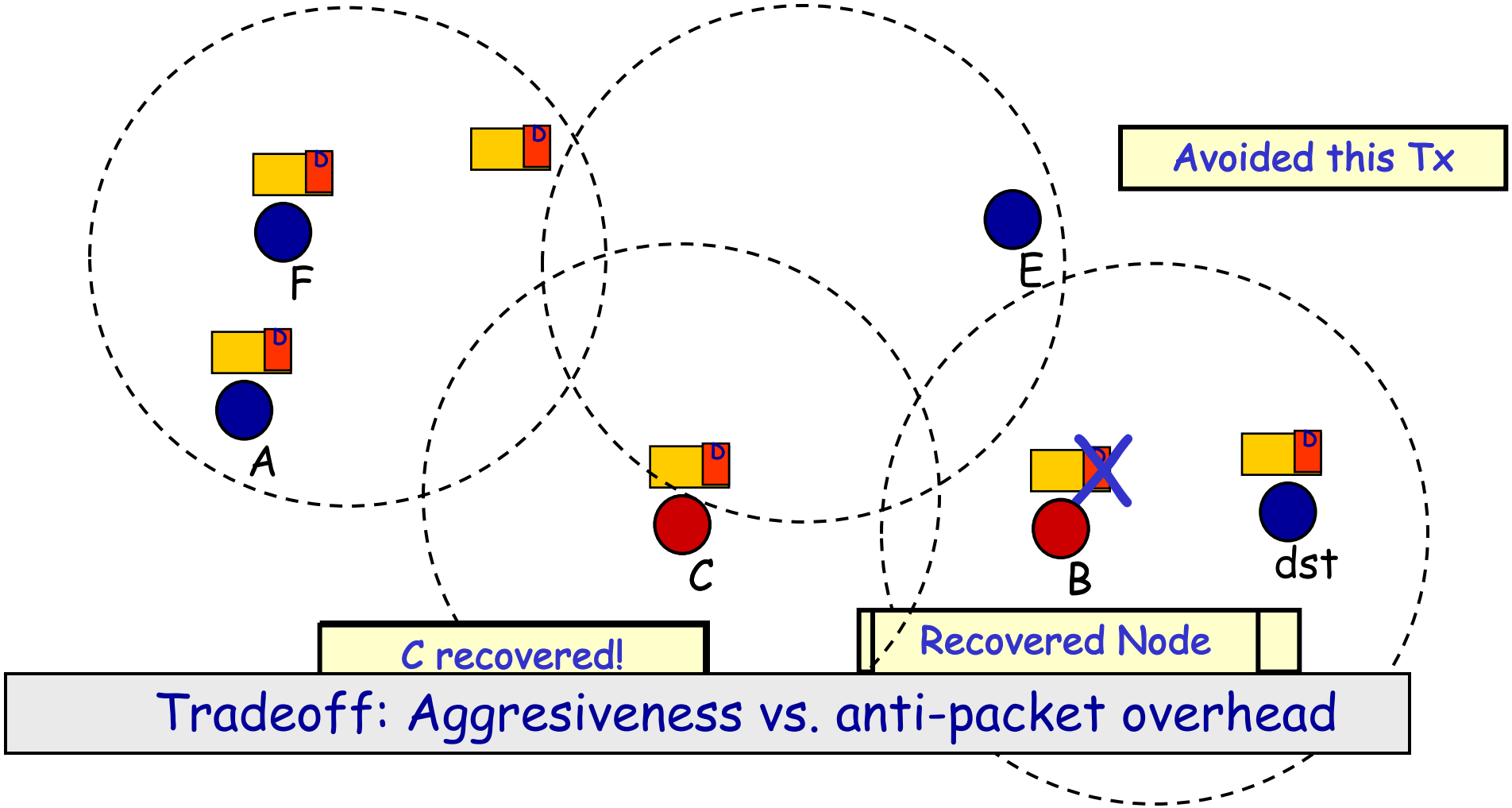
- Epidemic routing hands over a copy to every node encountered…

Even after the message has been delivered!

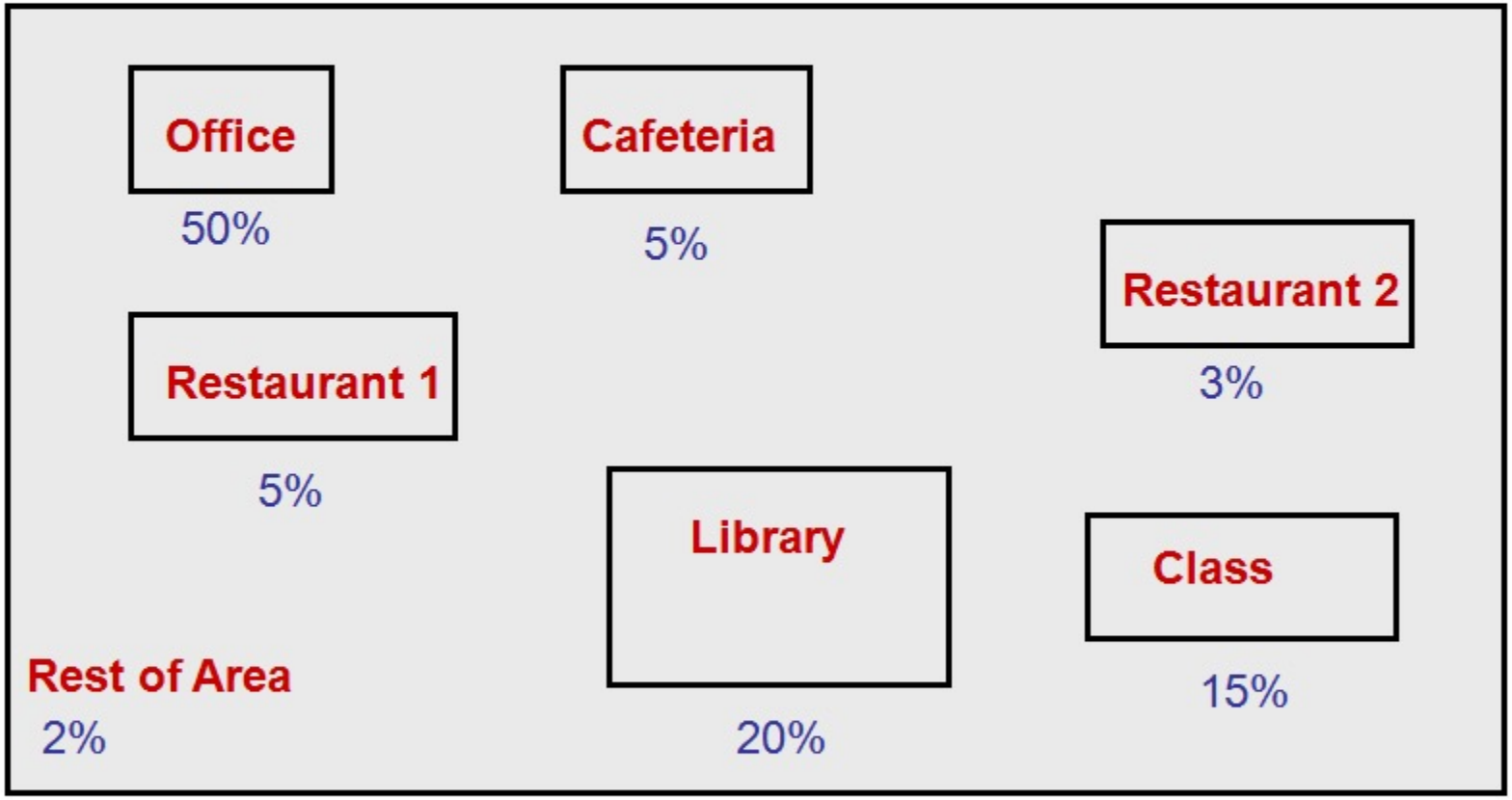
- After one relay finds destination
  - Remove message from buffer
  - Inform other nodes to stop forwarding
- Can do this with different aggressive-ness
  - Immune
  - Vaccine

An Example: IMMUNE-TX

- Propagate anti-packet to already infected nodes



Mobility profile-based Prediction



- Mobility Profile: Each node visits specific locations more often

Mobility Profile based Prediction

- Separate network into K locations
- Represent each user as a K-dimensional vector M_n
  - Binary (e.g. 1 = visits location i often): [1 0 1…0 1 0 1]
  - Scalar (e.g. visits location i with probability p_i): [p_1 p_2…p_K]
- "Distance" metric between nodes n and m: $|M_n - M_m|$
  - E.g. Euclidean distance $\sqrt{\sum_{i=1\ldots K}\left(M_n(i) - M_m(i)\right)^2}$
- Encounter probability p(i,j) = f($|M_n - M_m|$)

Mobility Profile based Prediction

STEP 1: Decide on model

- e.g. how many locations K
- K too large: little overlap (overfit)
- K too small: too crude prediction

STEP 2: "Learn" model parameters

- Node n must estimate its vector M_n
- $M_n(i)$ = Time at location i / $Time_{window}$
- e.g. track associated AP, GPS, etc.

STEP 3: Use estimated model to predict future encounters

- $p(i,j) = f(|M_n - M_m|)$: which function f()?
- Per contact vs. path probability

A Complete DTN Routing Scheme

- Prediction-based schemes are more efficient than random

MAXIM 1: Use prediction based algorithm to forward a copy or to decide whether to create one more

- Using a single copy is not enough
  - Get stuck in local maxima
  - Delay/delivery variance too high (depending on model)

MAXIM 2: Use multiple but few copies (e.g. spray) and do prediction-based routing for each in parallel

Mobility Modeling

- Simulations are not enough
  - Long time to run for each new parameter set (min, hours, days!)
  - Do not scale to large number of nodes (memory, processing issues)
  - Can get lost in the details of elaborate implementation
  - Design and Optimization is heuristic
- Traces and experiments are not enough
  - Usually even smaller than simulations; proof of concept
  - Too difficult to check/try various design alternatives
  - Design and Optimization is heuristic
- Theory
  - Quick rough estimate of performance
  - Interpretability of design choices and their effect on performance
  - Modeling and Optimization

Random Direction (Random Waypoint): Hitting Time

- Movement is a set of "epochs"

Method:

1. Probability that any given epoch hits the destination

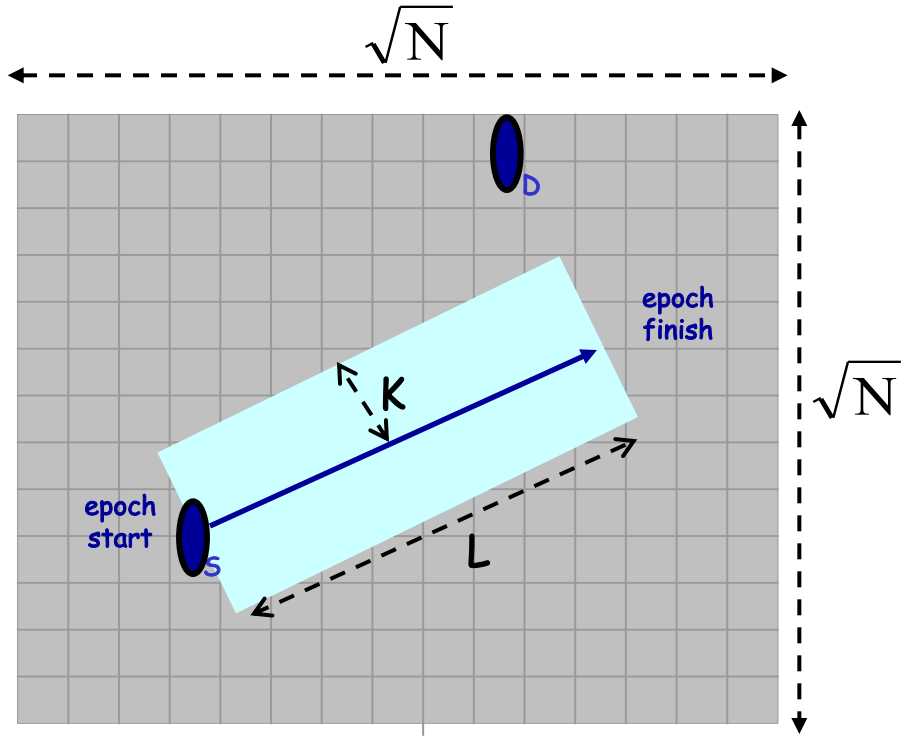
$$P_{hit} = \frac{2KL}{N}$$

2. *Expected* number of such epochs (geometric)

$$\overline{N}_e = \frac{1}{P_{hit}} = \frac{N}{2KL}$$

3. Multiply by the *expected* duration of each epoch T_e

$$ET = \frac{N}{2KL} T_e$$



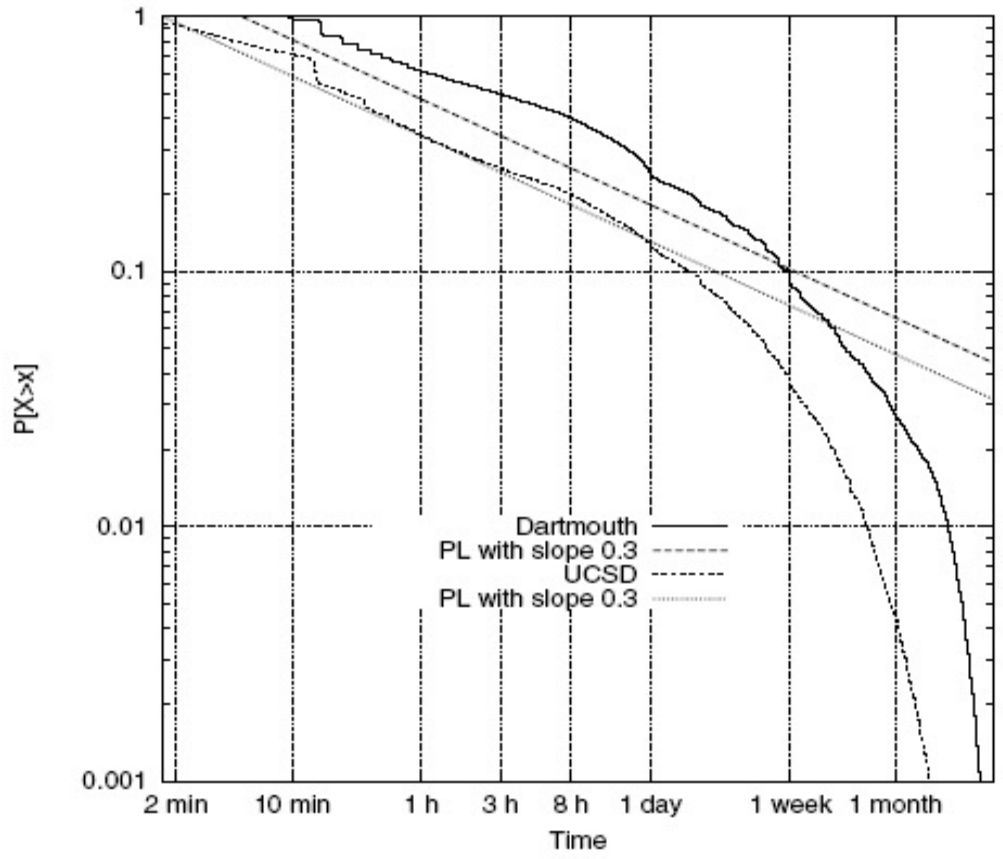
4. EM: divide by (normalized) relative speed between S and D, $v_r = \frac{E[|\vec{v}_S - \vec{v}_D|]}{E[|v_S|]}$

$$EM = \frac{ET}{\overline{v}_r}$$

Inter-contact Times Measurements

- Consecutive transmission opportunities to a given node
- Contact-based trace measurements: what is the distribution of inter-contact times?
  - WLAN traces (Dartmouth, UCSD)
  - Inter-node (ad hoc mode) traces (Cambridge, Toronto)

But is it REALLY Heavy-tailed?



- Power-law only within a range of CCDF
- What about the rest of the tail (artifact of experiments, or not power-law really)?