Delay Tolerant Networking

Jeff Pang
Abhijit Deshmukh

(with slides borrowed from
Kevin Fall, Sushant Jain, Yogita Mehta, and Yong Wang)

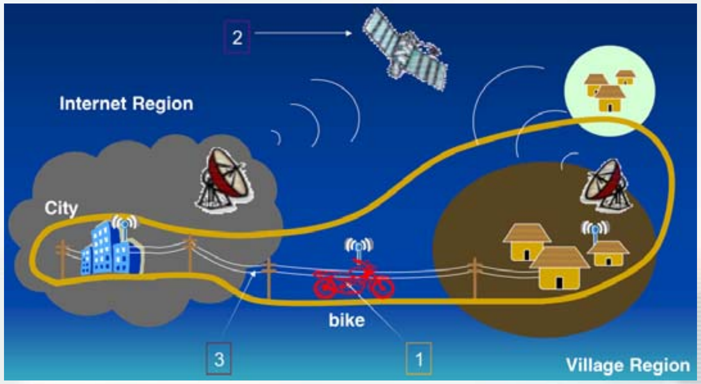
Unstated Internet Assumptions

- Some path exists between endpoints
  - Routing finds (single) "best" existing route
    - [some exceptions…e.g. ECMP]
- End-to-end RTT is not terribly large
  - A few seconds at the very most (usually much less)
  - →window-based flow/congestion control works
- E2E reliability using ARQ works well (enough)
  - True for low loss rates (under 2% or so)
- Packets are the right abstraction
  - Internet (IP) makes packet switching interoperable
  - Routers don't modify packets (much) when forwarding

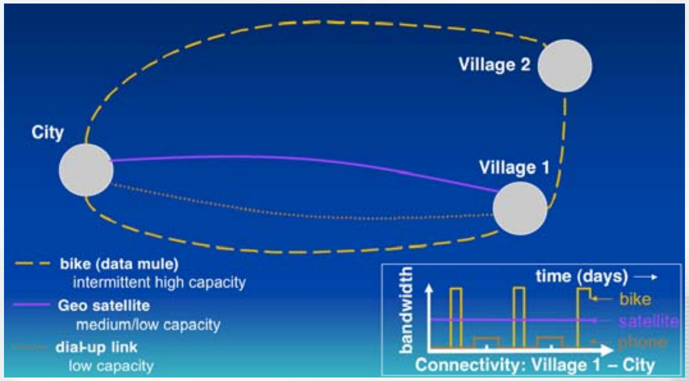
New challenges…

- Very Large E2E Delays
  - Natural prop delay could be seconds to minutes
  - If disconnected, queuing times may be much longer
- Intermittent and *Scheduled* Links
  - Disconnection may not be due to failure (e.g. LEO sats and scheduling links down for power management)
  - Retransmission may be very expensive
    - Unauthorized access could be a big problem
- 'Radically' Heterogeneous Network Architectures
  - Many specialized networks won't/can't ever run IP

DTN Example



DTN Example Abstraction



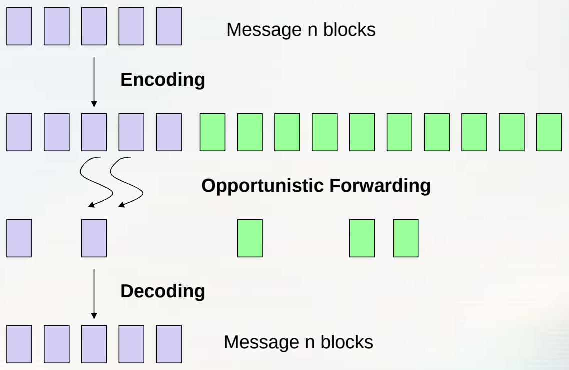
Using Redundancy to Cope with Failures in a Delay Tolerant Network

Sushant Jain, Michael Demmer,
Rabin Patra, Kevin Fall

Introduction

- Routing in Delay Tolerant Network (DTN) in presence of path failures is difficult
- Retransmissions cannot be used for reliable delivery
  - Timely feedback may not be possible
- How to achieve reliability in DTN?
  - Replication, Erasure coding

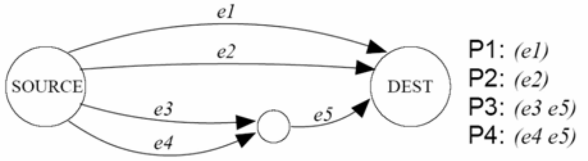
Erasure Codes



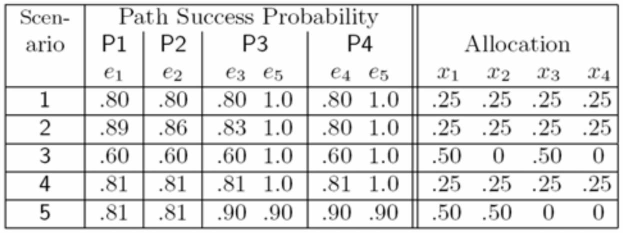
Erasure-coding based forwarding

- Message size *M*
- Replication factor *r*
- Code block size *b*
- Total number of blocks *n=(1+ε)M*r/b*

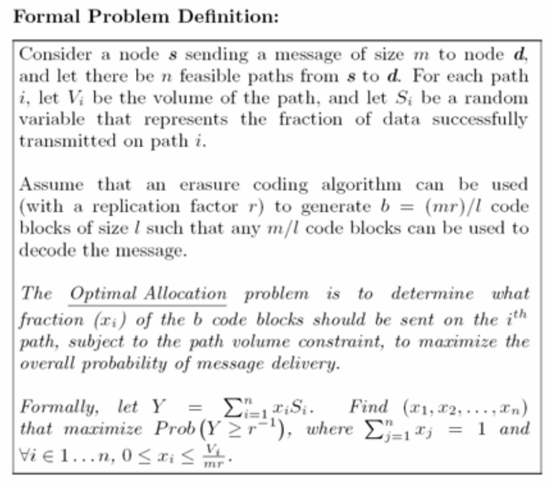
- Can decode with any n/r blocks



P1: *(e1)*
P2: *(e2)*
P3: *(e3 e5)*
P4: *(e4 e5)*



Scen- ario	Path Success Probability						Allocation			
	P1	P2	P3		P4					
	e_1	e_2	e_3	e_5	e_4	e_5	x_1	x_2	x_3	x_4
1	.80	.80	.80	1.0	.80	1.0	.25	.25	.25	.25
2	.89	.86	.83	1.0	.80	1.0	.25	.25	.25	.25
3	.60	.60	.60	1.0	.60	1.0	.50	0	.50	0
4	.81	.81	.81	1.0	.81	1.0	.25	.25	.25	.25
5	.81	.81	.90	.90	.90	.90	.50	.50	0	0



Formal Problem Definition:

Consider a node ***s*** sending a message of size m to node ***d***, and let there be n feasible paths from ***s*** to ***d***. For each path i, let V_i be the volume of the path, and let S_i be a random variable that represents the fraction of data successfully transmitted on path i.

Assume that an erasure coding algorithm can be used (with a replication factor r) to generate $b = (mr)/l$ code blocks of size l such that any m/l code blocks can be used to decode the message.

The Optimal Allocation problem is to determine what fraction (x_i) of the b code blocks should be sent on the i^{th} path, subject to the path volume constraint, to maximize the overall probability of message delivery.

Formally, let $Y = \sum_{i=1}^{n} x_i S_i$. *Find* $(x_1, x_2, \ldots, x_n)$ *that maximize* $Prob\left(Y \geq r^{-1}\right)$, *where* $\sum_{j=1}^{n} x_j = 1$ *and* $\forall i \in 1 \ldots n, 0 \leq x_i \leq \frac{V_i}{mr}$.

Bernoulli Path Failure, S_i are identical and independent

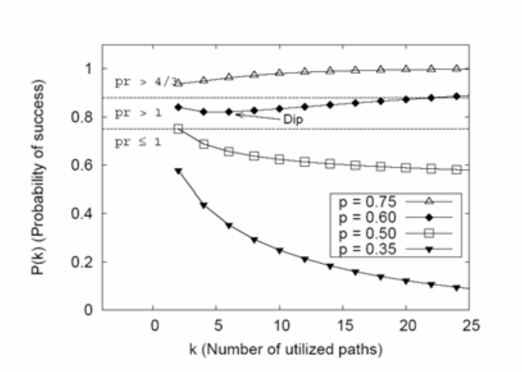
- Family of allocation strategies is used for kth strategy

$$x_i = \begin{cases} \frac{1}{k} & \text{if } 1 \leq i \leq k \\ 0 & otherwise \end{cases}$$

- Probability of success of kth strategy

$$\mathcal{P}(k) = \sum_{i=k/r}^{k} p^i (1-p)^{k-i} \binom{k}{i}$$

Bernoulli Path Failure Regimes



Bernoulli Path Failure, S_i are different

$$c_{ji} = \frac{j}{2^{i-1}} \; mod \; 2 \quad \text{for } i \in 1 \ldots n$$

Formulation of Mixed Integer Program (MIP)

$$\text{Maximize} \sum_{j=0}^{2^n-1} y_j w_j \qquad \text{subject to,} \qquad (1)$$

$$y_j = \{0,1\}, \quad \sum_{i=1}^{n} c_{ji} x_i \geq y_j / r \qquad \text{for } j \in 0 \ldots (2^n - 1) \quad (2)$$

$$0 \leq x_i \leq u_i \qquad \text{for } i \in 1 \ldots n \qquad (3)$$

$$\sum_{i=1}^{n} x_i = 1 \qquad (4)$$

Objective Function:

$$\sum_{j=0}^{2^n-1} Prob\left(Y > r^{-1} \middle| \, j^{th} \; outcome\right) Prob\left(j^{th} \; outcome\right)$$

Partial Path Failures

If Y is Gaussian with mean μ_Y and variance σ_Y^2, then:

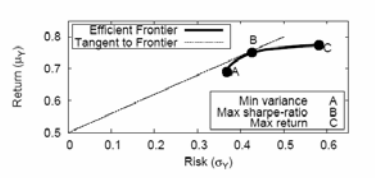
$$Prob(Y > r^{-1}) = \frac{1}{2}\left(1 + erf\left(\frac{\mu_Y - 1/r}{\sigma_Y \sqrt{2}}\right)\right), \text{where,}$$

$$erf(z) \equiv \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} dt$$

Objective: Maximize Sharpe Ratio $\left(\frac{\mu_Y - 1/r}{\sigma_Y}\right)$

$$\mu_Y = \sum_{i=1}^{n} x_i p_i, \qquad \sigma_Y^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j \sigma_{ij}$$

Use efficient frontier notion



Efficient frontier generated from an experiment with 6 paths with probabilities .85, .7, .65, .65, .6, .6

Markowitz algorithm

```
Normalization algorithm
Input: $n$, $\{u_i\}$, $\{x_i\}$ $i \in 1..n$
Output: $\{x_i\}$
$\forall i \in 1..n,\ x_i = x_i / \sum_1^n x_j$
$\forall i \in 1..n,$ if $x_i < 0$ then $x_i = 0$
$\forall i \in 1..n,$ if $x_i > u_i$ then $x_i = u_i$
if $\sum_1^n x_i > 1$ then
    $\forall i \in 1..n,\ x_i = x_i / \sum_1^n x_j$
else
    $l = 1 - \sum_1^n x_i$
    while $l > 0$ do
        $t = arg\max_{i \in 1..n}\{p_i \mid x_i < u_i\}$
        $\Delta = min(u_t - x_t, l)$
        $l = l - \Delta$ ; $x_t = x_t + \Delta$
    end while
end if
```

```
Markowitz algorithm
if $\max(\mu_Y) > 1/r$ then
    $\hat{x} = \frac{\hat{V}^{-1}(r\hat{p}-1)}{1\hat{V}^{-1}(r\hat{p}-1)}$
else
    Maximize $\mu_Y$
end if
Use normalization algorithm
```

Evaluation

- Three scenarios used for evaluation:
  - DTN routing over data MULEs
    - Path independent, data loss Bernoulli
  - DTN routing over set of city buses
    - Paths dependent, data loss Bernoulli
  - DTN routing large sensor network
    - Partial path failures

Data MULE Scenario

- Simulation Setup:

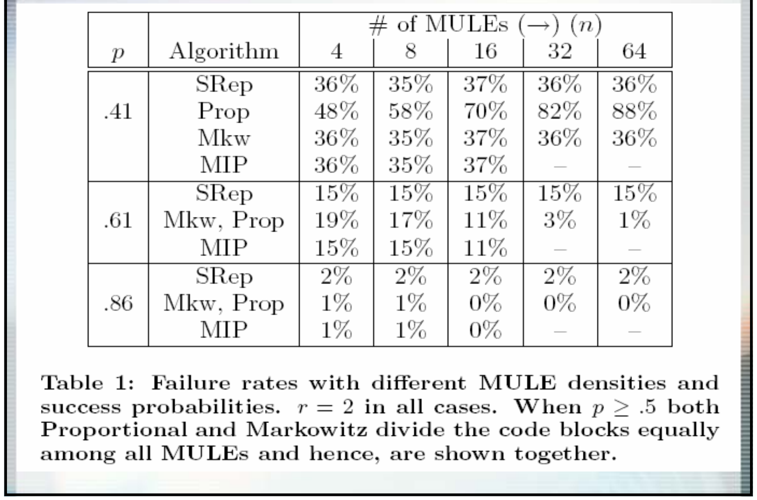
  1km x 1km planar area, source and destination at opposite corners.

  Message size 10KB, Contact bandwidth 100Kbps, Storage capacity of MULE 1MB

  Velocity of MULE 10m/s.
- Probability of success of ith path is

  $$p_i = Prob(D_i \leq T)$$
- D_i is the delay in distribution by ith MULE, T is the message expiration time

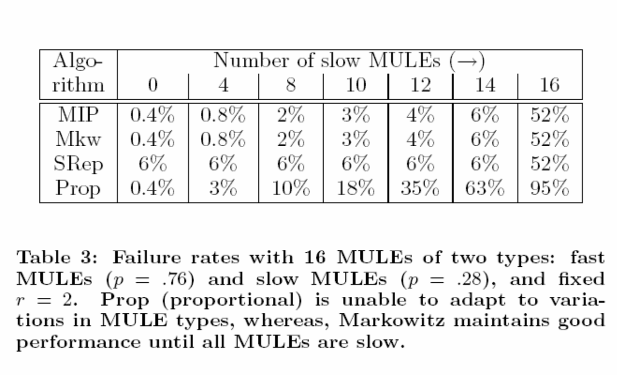
MULE Density



p	Algorithm	# of MULEs (→) (n)				
		4	8	16	32	64
.41	SRep	36%	35%	37%	36%	36%
	Prop	48%	58%	70%	82%	88%
	Mkw	36%	35%	37%	36%	36%
	MIP	36%	35%	37%	–	–
.61	SRep	15%	15%	15%	15%	15%
	Mkw, Prop	19%	17%	11%	3%	1%
	MIP	15%	15%	11%	–	–
.86	SRep	2%	2%	2%	2%	2%
	Mkw, Prop	1%	1%	0%	0%	0%
	MIP	1%	1%	0%	–	–

Table 1: Failure rates with different MULE densities and success probabilities. $r = 2$ in all cases. When $p \geq .5$ both Proportional and Markowitz divide the code blocks equally among all MULEs and hence, are shown together.

Different Success Probabilities



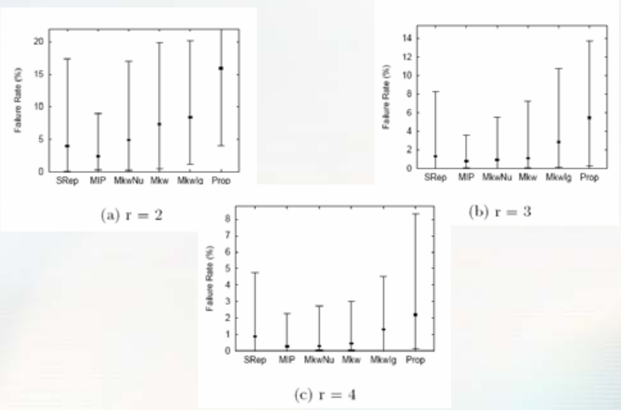
Algorithm	Number of slow MULEs (→)						
	0	4	8	10	12	14	16
MIP	0.4%	0.8%	2%	3%	4%	6%	52%
Mkw	0.4%	0.8%	2%	3%	4%	6%	52%
SRep	6%	6%	6%	6%	6%	6%	52%
Prop	0.4%	3%	10%	18%	35%	63%	95%

Table 3: Failure rates with 16 MULEs of two types: fast MULEs ($p = .76$) and slow MULEs ($p = .28$), and fixed $r = 2$. Prop (proportional) is unable to adapt to variations in MULE types, whereas, Markowitz maintains good performance until all MULEs are slow.

Bus Network Scenario

- Simulation Setup
  - Radio bandwidth 400kbps, radio range 100m
  - 20 messages of size 10kb, sent randomly every hour for 12 hours
  - bus storage 1Mb
  - Message expiration time 6 hours
  - Paths are multi-hop

Bus Network Scenario contd.



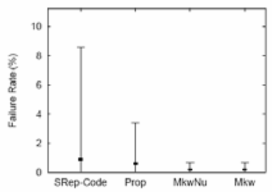
(a) r = 2

(b) r = 3

(c) r = 4

Sensor Network Scenario

- Simulation Setup
  - Nodes placed in 40x16 foot grid, grid size 8ft



Technique	Avg FR
SRep	95%
SRep-Code	2.88%
Prop	2.738%
MkwNu	1.71%
Mkw	1.70%

Figure 6: Failure rates in the sensor network scenario. The path success probabilities range from 0.4 to 0.6. $r = 2$. For each technique, we show the median, 5^{th} and 95^{th} percentiles failure rate across 50 messages. The average failure rates are listed in the table and show that SRep has a significantly high failure rate.

Benefits of Erasure Coding

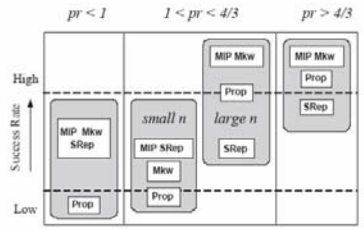


Figure 7: Qualitative performance of different techniques in three regimes. The first region corresponds to a low product pr, in which the use of more paths is detrimental, and simple replication performs well. In the second regime, the "gray zone," the benefits of erasure coding are only evident when many paths are used. In the third regime, all techniques have high success rate since all paths are good.

Summary

- Problem of reliable transmission in DTN
- Replication and erasure code for increasing reliability
- Formulate the optimal allocation problem
- Study of this problem for Bernoulli and partial path failures
- Evaluation of the analysis in three different scenarios

Discussion

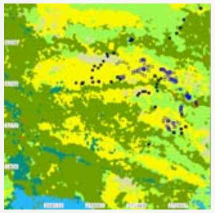
- What assumptions does this formulation make about the DTN graph?
  - Paths are known beforehand
  - Path success rates are not time varying
- What other problem formulations might be useful to DTN applications besides “max Pr(success), given replication factor r and max delay d”?
  - min r, given Pr(success) > k
  - min d, given r

Erasure-Coding Based Routing in Opportunistic Networks

Yong Wang, Sushant Jain
Margaret Martonosi, Kevin Fall

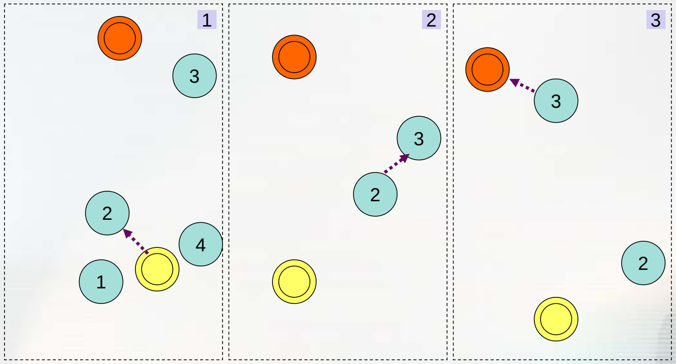
Motivation

- Data forwarding in opportunistic wireless networks
  - ZebraNet
  - Data Mule



- Challenges
  - End-to-end route is not always available
  - Contact connectivity is intermittent and hard to predict
  - Resource budget can limit transmissions
  - Sometimes messages have deadline

Illustration



Previous Solutions

- "Intelligently" distribute identical data copies to contacts to increase chances of delivery
  - Flooding (unlimited contacts)
  - Heuristics: random forwarding, history-based forwarding, predication-based forwarding, etc. (limited contacts)
- Given "replication budget", this is difficult
  - Using simple replication, only finite number of copies in the network [Juang02, Grossglauser02, Jain04, Chaintreau05]
  - Routing performance (delivery rate, latency, etc.) heavily dependent on *"deliverability" of these contacts* (or *predictability of heuristics)*
  - No single heuristic works for all scenarios!

Using Erasure Codes

- Rather than seeking particular "good" contacts, we "split" messages and distribute to more contacts to increase chance of delivery
  - Same number of bytes flowing in the network, now in the form of coded blocks
  - Partial data arrival can be used to reconstruct the original message
    - Given a replication factor of *r*, (in theory) any *1/r* code blocks received can be used to reconstruct original data
  - Potentially leverage more contacts opportunity that result in lowest worse-case latency
- Intuition:
  - Reduces "risk" due to outlier bad contacts

Background: Forwarding Algorithms

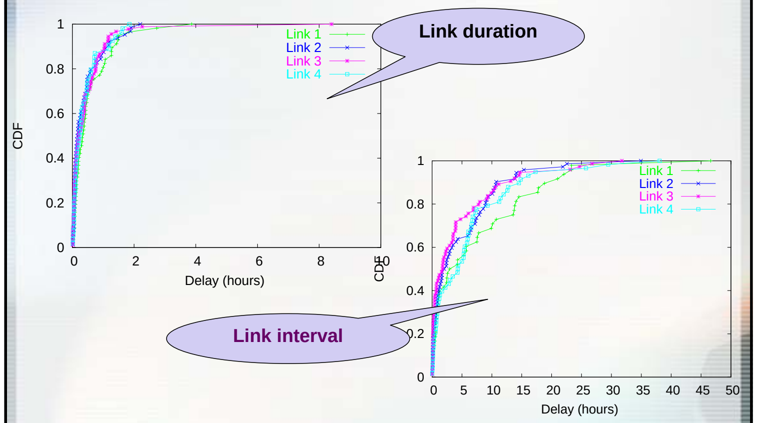
Algorithm	Who	When	To whom
Flood	All nodes	New contact	All new
Direct	Source only	Destination	Destination only
Simple Replication(r)	Source only	New contact	*r* first contacts
History (r)	All nodes	New contact	*r* highest ranked
Erasure Coding (ec-r)	Source only	New contact	*kr (k>=1)* first contacts (k is related to coding algorithm)

Evaluation Methodology

- We use a real-world mobility trace collected from the initial ZebraNet test deployment in Kenya, Africa, July, 2004

- Node 8 returned 32-hour uninterrupted movement data
  - Weather and waterproofing issues
- Semi-synthetic group model
  - Statistics of turning angles and walking distance

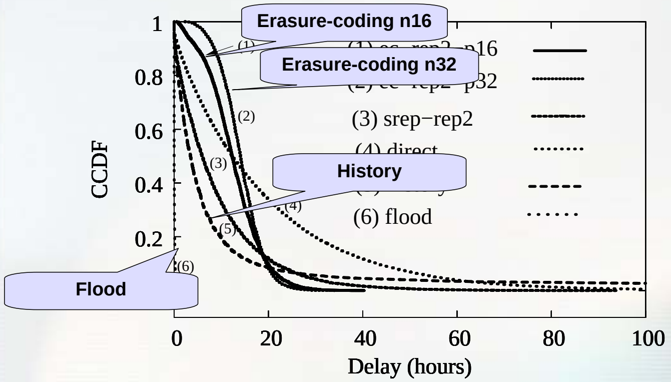
Trace Statistics



Link duration

Link interval

Performance Evaluation: Latency (64 nodes)



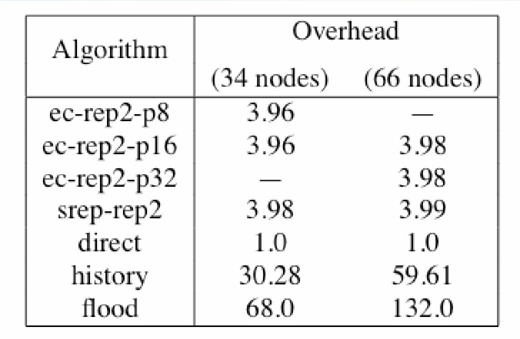
Erasure-coding n16

Erasure-coding n32

History

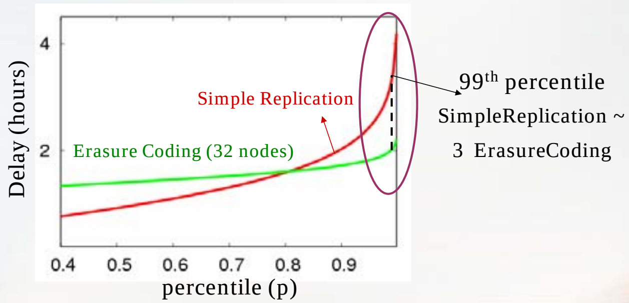
Flood

Routing Overhead



Algorithm	Overhead	
	(34 nodes)	(66 nodes)
ec-rep2-p8	3.96	—
ec-rep2-p16	3.96	3.98
ec-rep2-p32	—	3.98
srep-rep2	3.98	3.99
direct	1.0	1.0
history	30.28	59.61
flood	68.0	132.0

Theoretical Results on Delay Distribution



Erasure Coding:

– Get rids of the **‘bad’** cases

– Has few very low delay cases

Summary

- **A new application of an old idea**
  - **Use erasure codes to address contact delivery failures**
  - **More robust to mobility dynamics**
- **Primary goal is worst-case** latency
  - **Theorems show that erasure-coding based algorithm has a Gaussian delay distribution, independent of the underlying link characteristics**
  - **Simulation results on `dtnsim2` validated that ec-based algorithm has the** lowest **worst-case delay (almost** 1/3 **of SimpleReplication in the 64-node scenario), among all algorithms compared.**

Discussion

- **What other overheads are there for ec vs. srep in a *wireless* MANET?**
  - **More small messages vs. less big messages**
    - **MAC overhead vs. collision cost**
- **Can we use the previous paper to model the same problem?**
  - **Path *i* = relay contact node *i***
  - **S_i = Pr(source contacts *i* and *i* contacts dest in time)**
  - **x_i = how many blocks to give to relay *i***

Routing in Delay Tolerant Network

Sushant Jain (University of Washington)
Kevin Fall (Intel Research, Berkeley)
Rabin Patra (University of California, Berkeley)

Abhijit Deshmukh

Instructor : Srinivasan Seshan

Outline

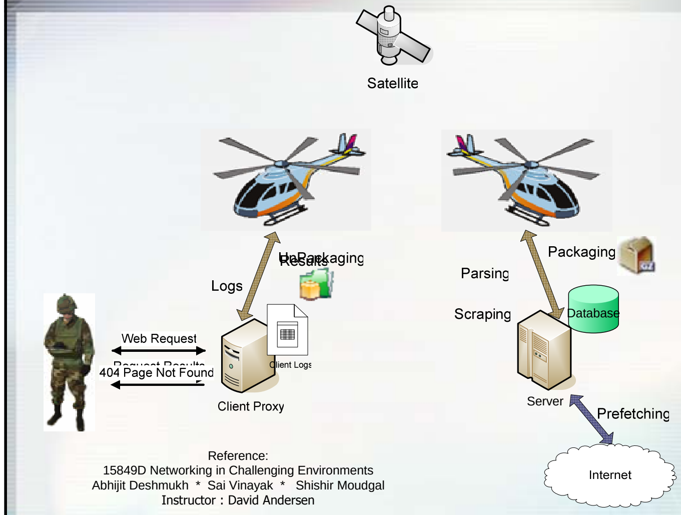
- Why do this? (a motivating example)
- What is routing in a DTN?
  - Why it is different (model assumptions)
  - Formulation
- Evaluation Framework
  - Oracle construction
  - Optimal solution
- Simulations
- Conclusions

The Problem: High Latency Networks

- Soldiers in Battle Field
  - Intermittent Internet connection
  - Packets physically moved on a helicopter
- Astronaut
- Village

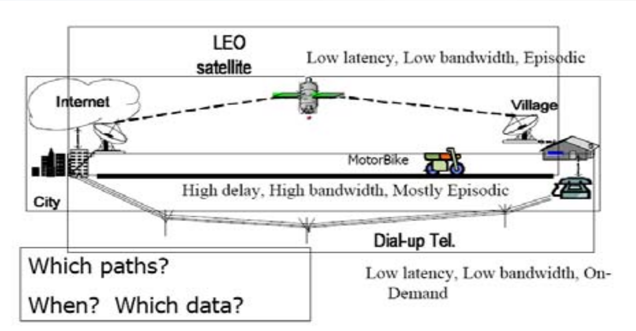
- Challenges
  - Providing Internet access
  - Use of Existing Infrastructure
  - Smart pre-fetching
  - Transparency
  - Cache Maintenance

WebEx: Architecture



Reference:
15849D Networking in Challenging Environments
Abhijit Deshmukh * Sai Vinayak * Shishir Moudgal
Instructor : David Andersen

Connecting a Remote Village

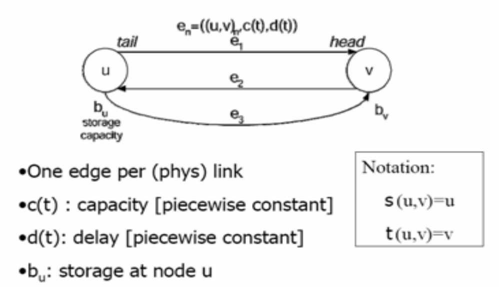

LEO satellite
Low latency, Low bandwidth, Episodic
Internet
Village
MotorBike
City
High delay, High bandwidth, Mostly Episodic
Dial-up Tel.
Which paths?
When? Which data?
Low latency, Low bandwidth, On-Demand

What is Routing in a DTN?

- Traditional routing
  - –Inputs: *G=(V,E), (s,d).* Find a shortest path from *s* to *d* in *G*.
  - –Dynamic: update as *G* changes
  - –but still assume some path *p(s,d)* exists. "Shortest" can vary.
- DTN Routing
  - –Inputs: Nodes with buffer limits, Contact List, Traffic Demand
  - –Contact list may contain periods of capacity zero
- Problem: given (some) metric of goodness, compute the path and schedule so as to optimize the metric. Multiple paths may be ok.
- Assumption: paths are not lossy (replication not used)

DTN Network Model

- Routing on Dynamic Graphs
  - –Contact : an opportunity to communicate
  - –Message : a tuple (u, v, t, m)
  - –Storage : nodes have finite long-term storage (buffers)
  - –Routing : store and forward fashion
- DTN r pology
  - –Link
- Sched
  - –May
  - –May



- One edge per (phys) link
- c(t) : capacity [piecewise constant]
- d(t): delay [piecewise constant]
- b_u: storage at node u

Notation:

s(u,v)=u

t(u,v)=v

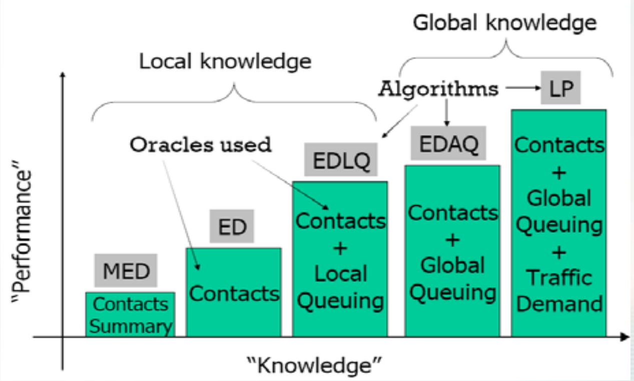
DTN Routing Objective

- A DTN Message k is an ordered tuple *(u,v,t,m)*
  - –*u*: source, *v*: destination, *t*: inject time, *m*: size [bytes]
- DTN Routing Objective
  - –Without violating these constraints:
    - Do not overrun buffer capacity
    - Do not overrun edge capacity
  - –Minimize average message delay
    - Optimal case will require multi-path
    - (other objectives are possible, but this helps most of them)
  - –Maximize probability of message delivery

DTN Routing Objective

- Oracle (definition)
  - –Abstract machine used to study decision problems
  - –Mechanism to produce predicted outcome, to be compared with actual outcome
- Contacts Oracle
  - –Complete link availability schedule (c(t), d(t))
  - –Time dependent information
- Contacts summary Oracle
  - –Average link availability
  - –Time independent information
- Queuing Oracle:
  - –Link queues, available storage
  - –Two versions: Local vs. Global
- Traffic Demand Oracle

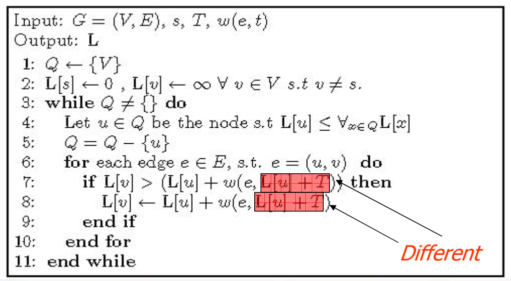
Conceptual Performance



Routing Algorithms

- First Contact (FC)
  - No use of Oracle
  - Random choice of edge
  - Advantages
    - Easy to implement
    - Performs fine for trivial cases
  - Disadvantages/Drawbacks
    - Message may oscillate (truly random choice of next hop)
    - Cannot route around congested networks
  - Improvements?
    - Directionality

Modified Dijkstra's Algorithm



```
Input: $G = (V, E)$, $s$, $T$, $w(e, t)$
Output: L
$Q \leftarrow \{V\}$
$L[s] \leftarrow 0$ , $L[v] \leftarrow \infty \; \forall \; v \in V \; s.t \; v \neq s.$
while $Q \neq \{\}$ do
    Let $u \in Q$ be the node s.t $L[u] \leq \forall_{x \in Q} L[x]$
    $Q = Q - \{u\}$
    for each edge $e \in E$, s.t. $e = (u, v)$ do
        if $L[v] > (L[u] + w(e, \boxed{L[u] + T}))$ then
            $L[v] \leftarrow L[u] + w(e, \boxed{L[u] + T})$
        end if
    end for
end while
```

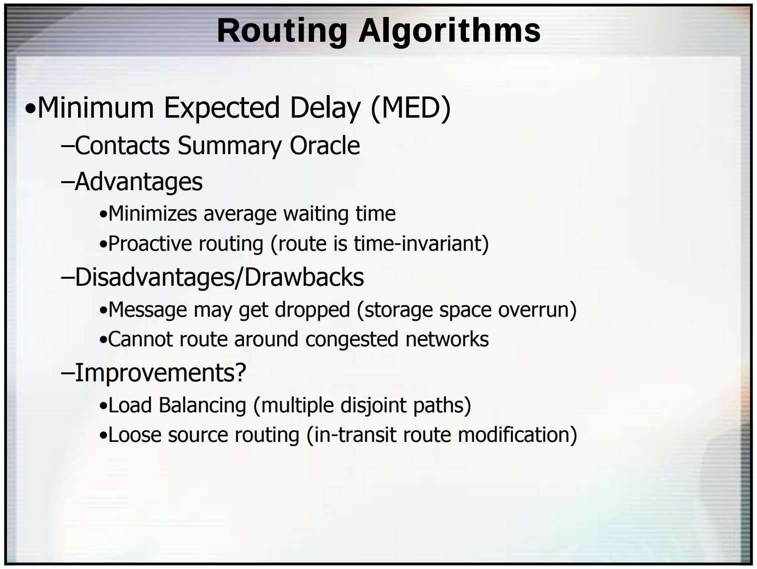
Different

Takes into account the time the message arrives at a node

- *T: start time*
- *L[]: path cost from s to all nodes*
- *w(e,t): cost (time) on e at time t*

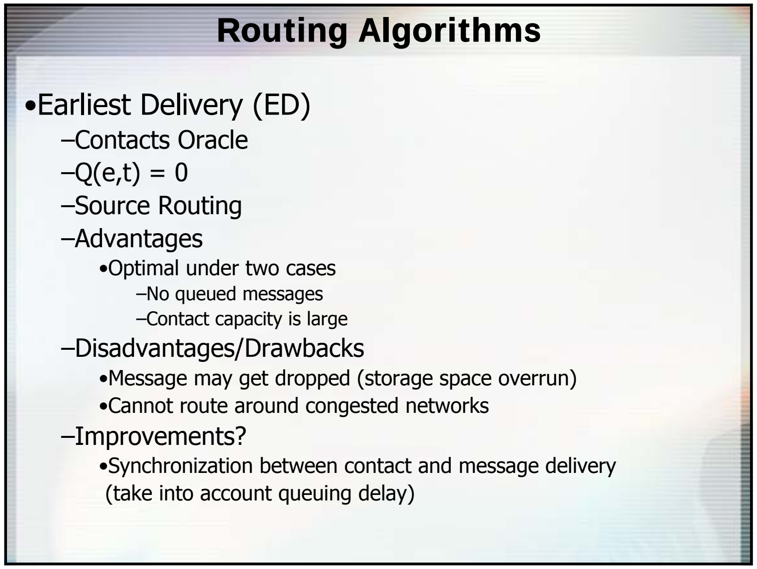
Adapting Dijkstra

- Using this framework we can assign w(e,t):
  - w(e,t) = msgsize/c(e,t) + Q(e,t)/c(e,t) + d(e,t)
  - **cost = transmission + queuing / waiting + propagation**
- Time-Varying cost
  - w(e,t) = w'(e, t, m, s)
- Q(e,t): amount of data queued for edge e at time t
  - Q(e,t) = 0 (for ED: earliest delivery)
- Q(e,t) = amount of data queued locally on e at time t
  - (for EDLQ: ED with local queuing information)
- Q(e,t) = amount of data queued anywhere for e at time t
  - (for EDAQ: ED with all queuing information)



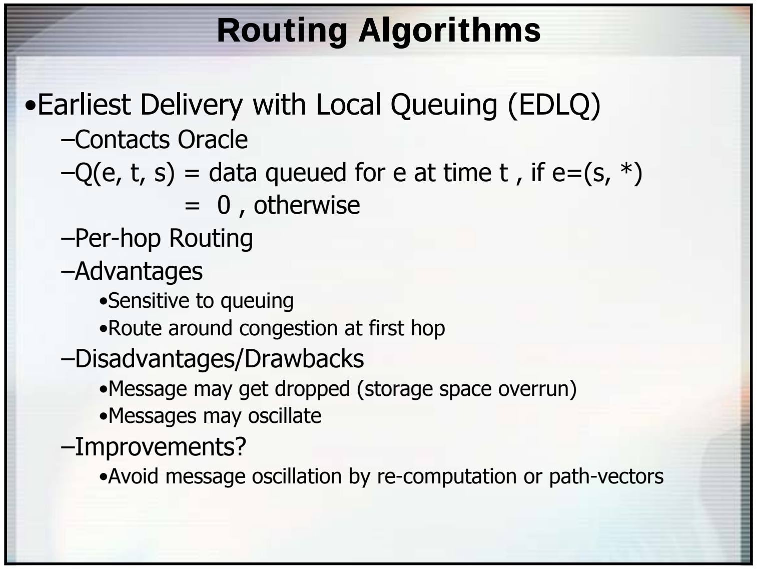
Routing Algorithms

- Minimum Expected Delay (MED)
  - Contacts Summary Oracle
  - Advantages
    - Minimizes average waiting time
    - Proactive routing (route is time-invariant)
  - Disadvantages/Drawbacks
    - Message may get dropped (storage space overrun)
    - Cannot route around congested networks
  - Improvements?
    - Load Balancing (multiple disjoint paths)
    - Loose source routing (in-transit route modification)



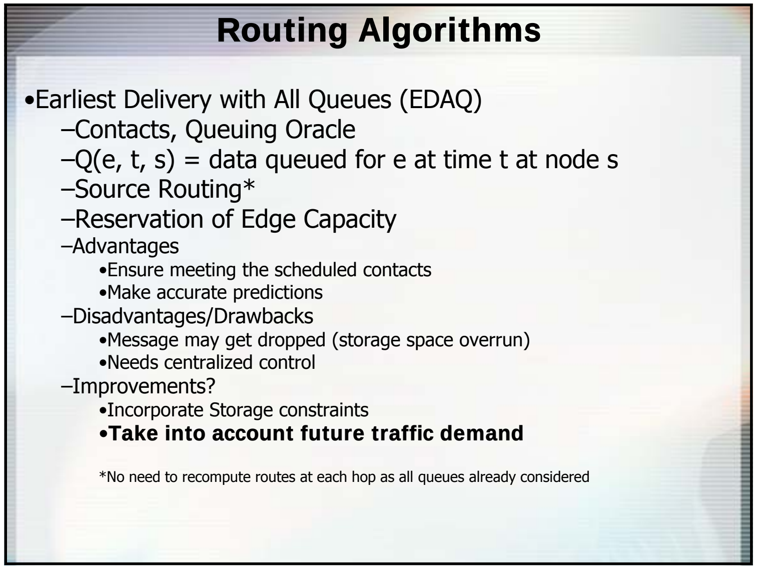
Routing Algorithms

- Earliest Delivery (ED)
  - Contacts Oracle
  - Q(e,t) = 0
  - Source Routing
  - Advantages
    - Optimal under two cases
      - No queued messages
      - Contact capacity is large
  - Disadvantages/Drawbacks
    - Message may get dropped (storage space overrun)
    - Cannot route around congested networks
  - Improvements?
    - Synchronization between contact and message delivery (take into account queuing delay)



Routing Algorithms

- Earliest Delivery with Local Queuing (EDLQ)
  - Contacts Oracle
  - Q(e, t, s) = data queued for e at time t , if e=(s, *)
    = 0 , otherwise
  - Per-hop Routing
  - Advantages
    - Sensitive to queuing
    - Route around congestion at first hop
  - Disadvantages/Drawbacks
    - Message may get dropped (storage space overrun)
    - Messages may oscillate
  - Improvements?
    - Avoid message oscillation by re-computation or path-vectors



Routing Algorithms

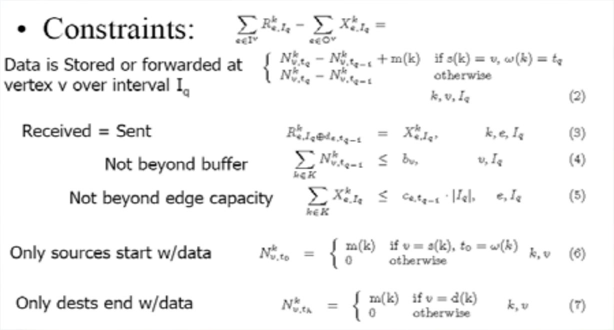
- Earliest Delivery with All Queues (EDAQ)
  - Contacts, Queuing Oracle
  - Q(e, t, s) = data queued for e at time t at node s
  - Source Routing*
  - Reservation of Edge Capacity
  - Advantages
    - Ensure meeting the scheduled contacts
    - Make accurate predictions
  - Disadvantages/Drawbacks
    - Message may get dropped (storage space overrun)
    - Needs centralized control
  - Improvements?
    - Incorporate Storage constraints
    - **Take into account future traffic demand**

*No need to recompute routes at each hop as all queues already considered

Linear Programming

- Flow Balance Equation for Time Interval
  - –Flows entering/leaving nodes and local buffers
  - –Contact start/end times and message arrival times
- Two steps
  - –Determine the time intervals
  - –Construct other LP constraints for DTN routing
- LP Formulation uses time intervals:
  - –Ie = {I1, …, Ih}, Iq = [tq-1,tq) (tq-1 < tq)
- Traffic Demand Definitions
  - –K [set of all messages (commodities)]
  - –K^v [set of messages destined for v]
  - –N^k_v,t [amount of k residing in v at time t]
  - –X^k_e,I [amount of k placed into e during I]
  - –R^k_e,I [amount of k received from e during I]

Linear Programming



- Constraints: $\sum_{e \in I^v} R^k_{e,I_q} - \sum_{e \in O^v} X^k_{e,I_q} =$

Data is Stored or forwarded at vertex v over interval I_q

$$\begin{cases} N^k_{v,t_q} - N^k_{v,t_{q-1}} + \mathrm{m}(\mathrm{k}) & \text{if } s(\mathrm{k}) = v,\ \omega(k) = t_q \\ N^k_{v,t_q} - N^k_{v,t_{q-1}} & \text{otherwise} \end{cases} \quad k, v, I_q \qquad (2)$$

Received = Sent

$$R^k_{e,I_q \oplus d_{e,t_{q-1}}} = X^k_{e,I_q}, \quad k, e, I_q \qquad (3)$$

Not beyond buffer

$$\sum_{k \in K} N^k_{v,t_{q-1}} \leq b_v, \quad v, I_q \qquad (4)$$

Not beyond edge capacity

$$\sum_{k \in K} X^k_{e,I_q} \leq c_{e,t_{q-1}} \cdot |I_q|, \quad e, I_q \qquad (5)$$

Only sources start w/data

$$N^k_{v,t_0} = \begin{cases} \mathrm{m}(\mathrm{k}) & \text{if } v = s(\mathrm{k}),\ t_0 = \omega(k) \\ 0 & \text{otherwise} \end{cases} \quad k, v \qquad (6)$$

Only dests end w/data

$$N^k_{v,t_h} = \begin{cases} \mathrm{m}(\mathrm{k}) & \text{if } v = \mathrm{d}(\mathrm{k}) \\ 0 & \text{otherwise} \end{cases} \quad k, v \qquad (7)$$

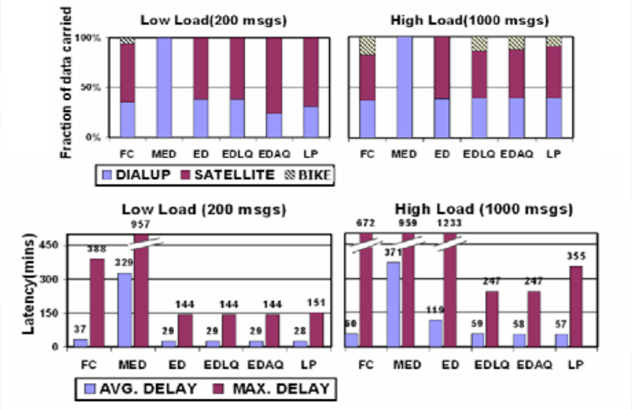
DTN Simulation

- Developed own DTN simulator (Java)
  - –Dynamic nature of nodes and links
  - –Nodes have finite storage capacity
- Special focus on link disconnection:
  - –Complete failure (all transiting msgs dropped)
  - –Close at source (all transiting msgs are delivered)
  - –Reactive fragmentation
- Simulated two scenarios
  - –Village network
  - –Bus network in San Francisco

Village Simulation

- Locations
  - –Kwazulu-Natal (Village) [see http://wizzy.org.za]
  - –Capetown, S. Africa (City)
- Network (based on a true story…)
  - –Dialup (4kbps at night 23:00-06:00 local time, 20msec)
  - –3 PACSATs (bent pipes, 4-5 passes/day, 10 min/pass,10kbps, 25msec)
  - –3 Motorbikes (2hr journey, 1Mbps to bike, 128MB storage, 5 min contacts)
- Traffic Pattern
  - –V → C traffic is small (1KB avg, ~web requests)
  - –C →V traffic is larger (10KB avg, ~web pages)
  - –Two loadings: 200 msgs/day (low), 1000 msgs/day (high)
  - –Traffic injected uniformly over 1st 24-hours of 48-hour simulation run

Observations



Observations

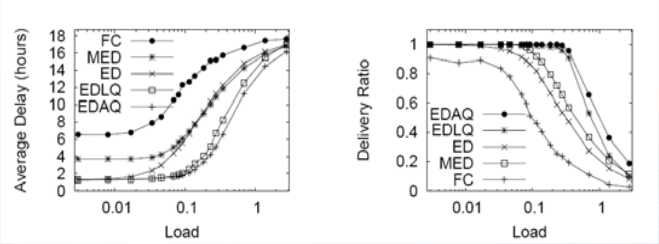
- A simplistic yet rich "routing" scenario
- MED: dialup always used during high or low load
  - Best average delay
- ED: most traffic over sat (60%), the rest uses dialup (low or high load)
  - Three satellites, 4 times a day
- FC: sometimes chooses bike (10%),
  - which explains its high maximum delay
  - avg delay is nominal
- EDAQ/EDLQ identical for low-load
- At high load, some differences appear:
  - MED, ED same as low load (not queuing aware)
  - ED deteriorates rapidly as it tries to route all messages over a satellite
    - High load, only few requests satisfied
    - Rest have to wait (at times even for 10 hours)
  - EDLQ/EDAQ now start using motorbike (~25%), leading to a significant reduction in delay
  - FC winds up routing more traffic over the bike which, interestingly, helps it out too
- LP took 7.5 min, for 16k iterations in CPLEX (8-proc PIII@700Mhz each with 3GB memory), producing about the same results as EDLQ/EDAQ (500k constraints)
  - Trades off higher max delay for the best minimum avg delay

Bus Network in San Francisco

- Locations
  - San Francisco City (4400m X 5600m)
  - 20 bus route network
- Graph Generation
  - Ordered sequence of stops (actual bus routes)
  - Contact time intervals (Disc model)
- Network
  - Uniform bus base speed between 10 and 20 m/s.
  - Radio Range : 100 meters
  - Default Storage Capacity : 100 Mbytes
  - Default Link Bandwidth : 100 Kb/s
- Traffic Pattern
  - 12 hours , 12 intervals of 1 hour each
  - 20 random source destination pairs
  - Source Bus → Destination Bus : 200 messages in 1 hour interval

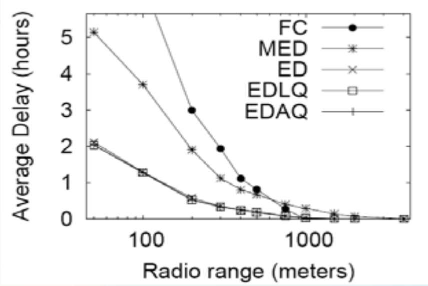
Results of Varying Bandwidth

- Low Load
  - No improvement in delay due to increased bandwidth
  - Insufficient volume of contacts
- Increased Load
  - Multiple contacts required
  - ED performance deteriorates (messages queued, contacts missed)
- High Load
  - Data undelivered, similar results across algorithms



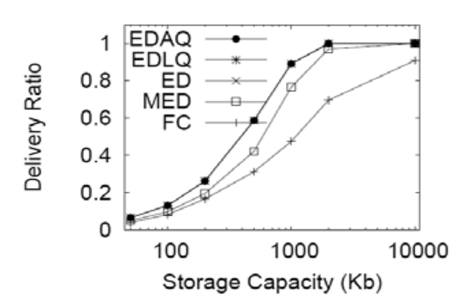
Results of Varying Radio Range

- Radio Range ⬆→Contact Time ⬆→Waiting Time ⬇→Avg Delay ⬇
- Low Radio Range
  - Smart Algorithms are a lot smart
- High Radio Range
  - Not so smart



Results of Varying Buffer Capacity

- Bandwidth : 400 Kb/s , Radio Range : 100m
- EDAQ, EDLQ, ED overlap !!
- Smarter algorithms are beneficial (limited storage capacity) ??



Conclusions

- DTN routing : challenging issue
- Limited Resources : Smarter algorithms of some use
- Light load: moderate scheme (ED) optimal
- Higher load: congestion aware scheme (EDLQ) ok
- Not a profound benefit for going to EDAQ or LP (!)

For More Information

- Delay Tolerant Networking Research Group
  - http://www.dtnrg.org
- Internet Research Task Force
  - http://www.irtf.org
- DTN Mailing list
  - dtn-interest@mailman.dtnrg.org
- Interplanetary Internet SIG (ISOC group)
  - http://www.ipnsig.org