Disruption-Tolerant Networks (DTNs): Mobility and Forwarding Algorithm

Richard J. La
University of Maryland

Joint work with G. Lipsa and M. Kalantari

Outline

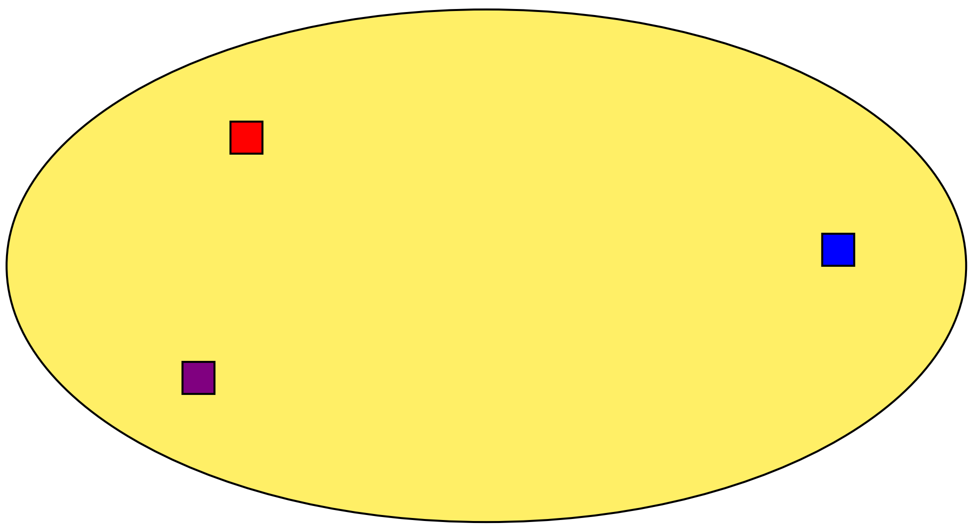
- Overview of DTNs
- Distribution of inter-contact times
  - Motivation
  - Existing work
  - Hybrid random walk
    - Distributional convergence
    - Simulation
- Ant-based packet forwarding algorithm
  - Overview of swarm intelligence & ant routing
  - Description of algorithm
  - Simulation

Overview of DTNs

- Evolution of Delay Tolerant Networks to Disruption Tolerant Networks
- Mobile nodes moving around an area
- Sparse node connectivity
  - Network disconnected most of the time
  - Nodes rely on other nodes to relay packets exploiting mobility
  - Mobility of nodes unknown in advance and may change over time
  - Multiple copies of packets may exist to increase the probability of successful delivery (e.g., controlled flooding)

Motivation

- Message delivery ratio and (average) delay affected by inter-contact times between nodes
  - Average value (intensity of contacts or meetings)
  - Correlation structure
  - **Distribution** of inter-contact times between two nodes



Background (inter-contact times)

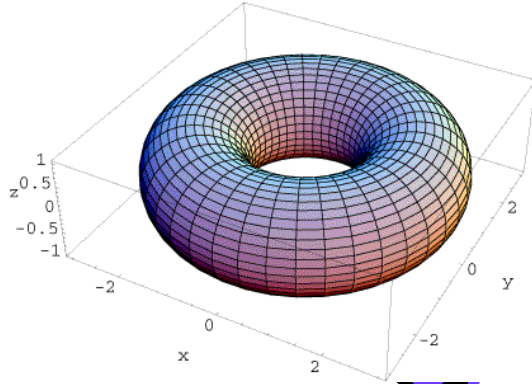
- What does the distribution of inter-contact times between two nodes look like in different environments?
  - Simulation and measurements
  - Can one say anything about it (even for simple models)?

Inter-contact times: simulations & measurements

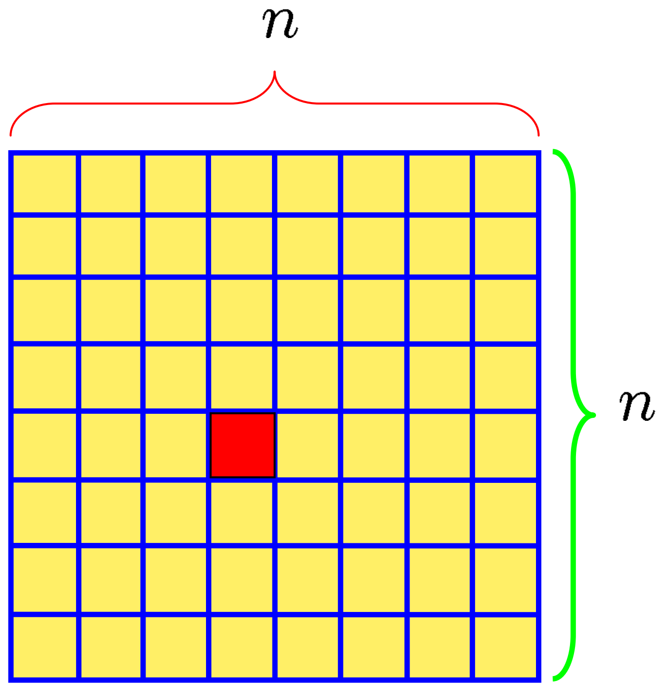
- Study by Groenevelt et al. ("The message delay in mobile ad hoc networks" – Performance, 2005)
  - suggests **exponential distribution** of inter-contact times (including random waypoint & random direction) using simulation

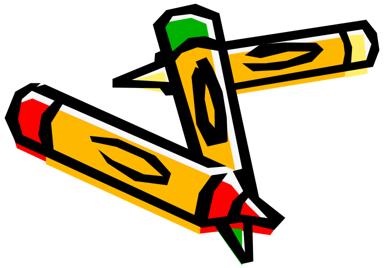
- Study by Chaintreau et al. ("Impact of human mobility on the design of opportunistic forwarding algorithms" – Infocom 2006)
  - Used real measurements obtained in several different settings
  - Reflects the underlying social network
  - Suggests **heavy tail distribution** (over a range of interest)

Random Walk



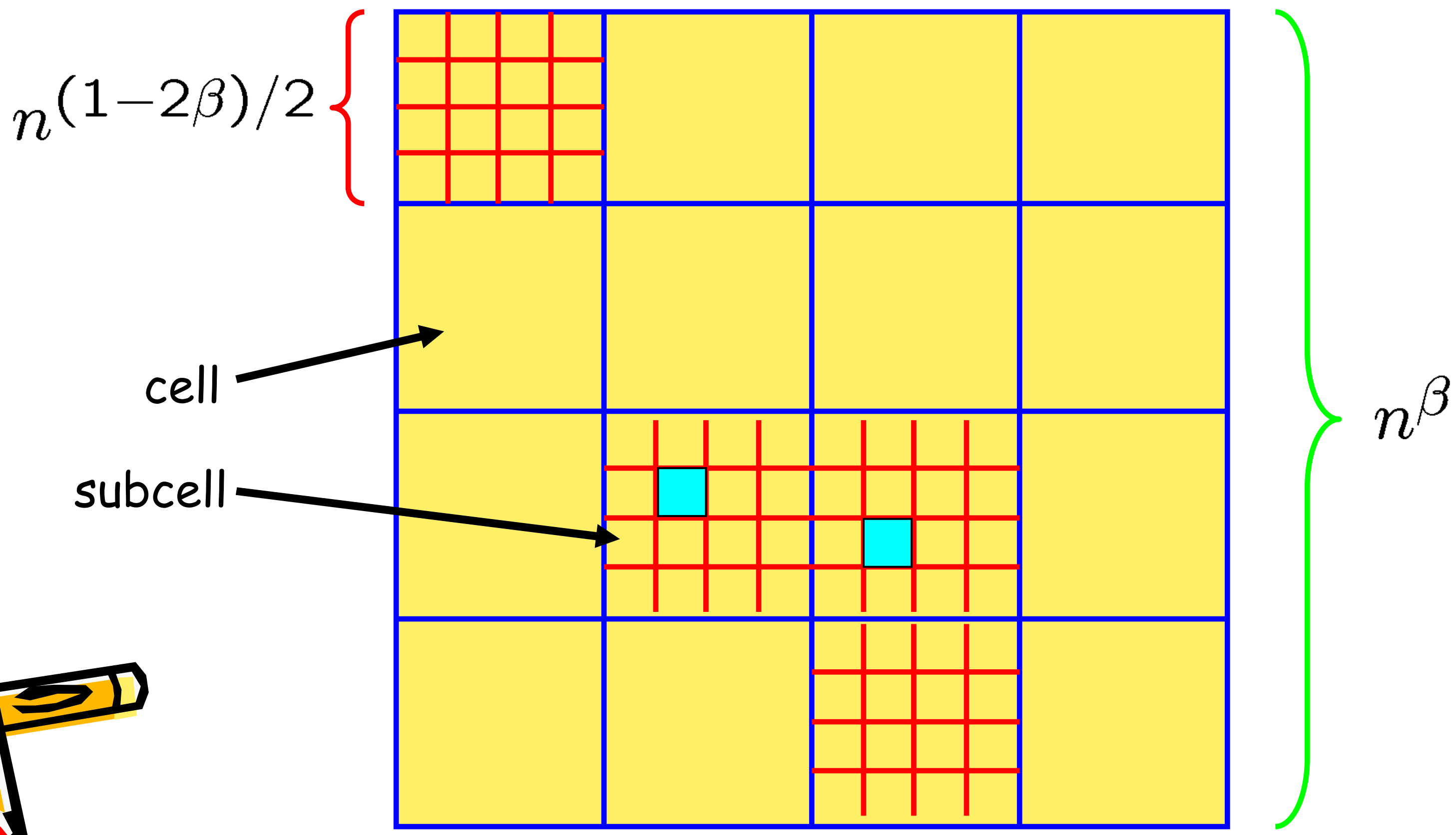
- El Gamal et al. – Infocom 2004
  - Unit square divided into n x n cells
  - wrapped around – a discrete torus of size n x n
  - Node moves to one of adjacent cells (up, down, left or right) with equal probability of $\frac{1}{4}$ (independent of the past)





Inter-contact times for Hybrid Random Walk (HRW)

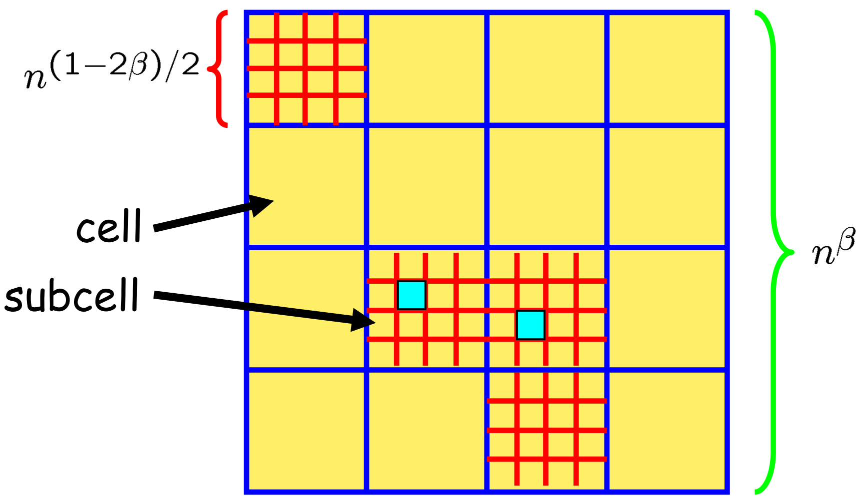
- Hybrid random walk mobility model: $0 \leq \beta \leq 1/2$



Inter-meeting times for Hybrid Random Walk (HRW)

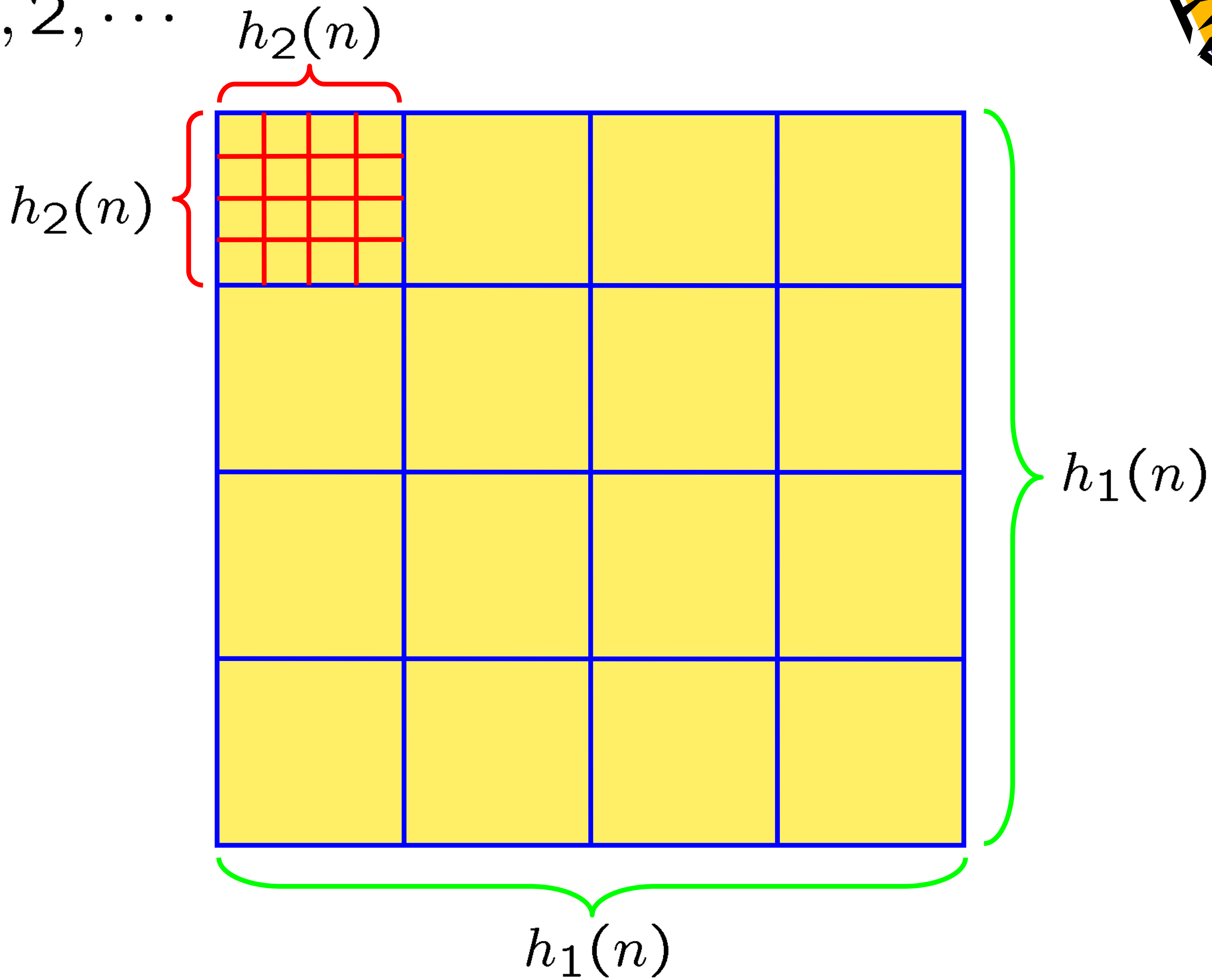
- $\beta = 0$: i.i.d. mobility
  - At each time t = 0, 1, … , a node is in one of n subcells with equal probability 1/n

- $\beta = \frac{1}{2}$: random walk



HRW mobility model

- $n = 1, 2, \cdots$



HRW mobility model

- For each fixed n,
  - Number of cells $= h_1(n) \times h_1(n)$
  - Number of subcells in each cell $= h_2(n) \times h_2(n)$

  Total number of subcells $= (h_1(n) \times h_2(n))^2$

Assumptions:

i. $h_1(n)$ is a positive odd integer and $h_2(n)$ is a positive integer

ii. $\lim_{n \to \infty} h_2(n) = \infty$

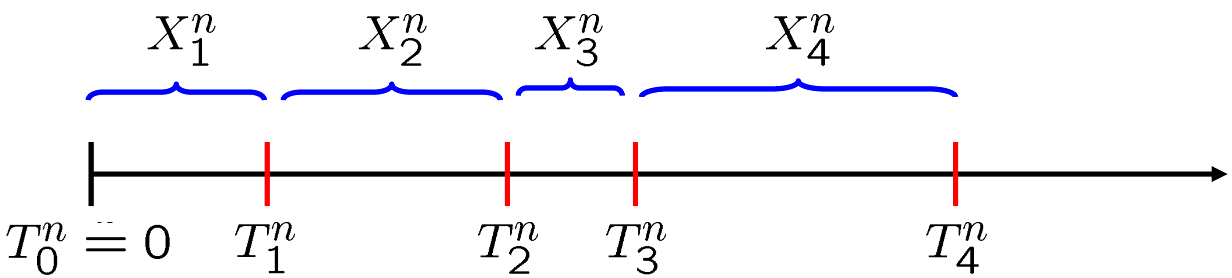
HRW mobility model

- We assume that two nodes meet (or have contact) when they are in the same subcell

- Let $\{T_k^n, k \geq 0\}$ (with $T_0^n = 0$) denote the sequence of times at which they meet each other, i.e., end up in the same subcell

  - T_k^n - time at which two nodes meet for the k-th time

  - $X_k^n := T_k^n - T_{k-1}^n$ - k-th inter-contact time (k=1,2,...)



HRW mobility model

- Note that the rvs $X_2^n, X_3^n, \cdots$ are i.i.d.

Proposition: Under Assumptions i) and ii) we have the following distributional convergence:

$$\frac{X_2^n}{(h_1(n) \times h_2(n))^2} \Rightarrow_n \exp(1)$$

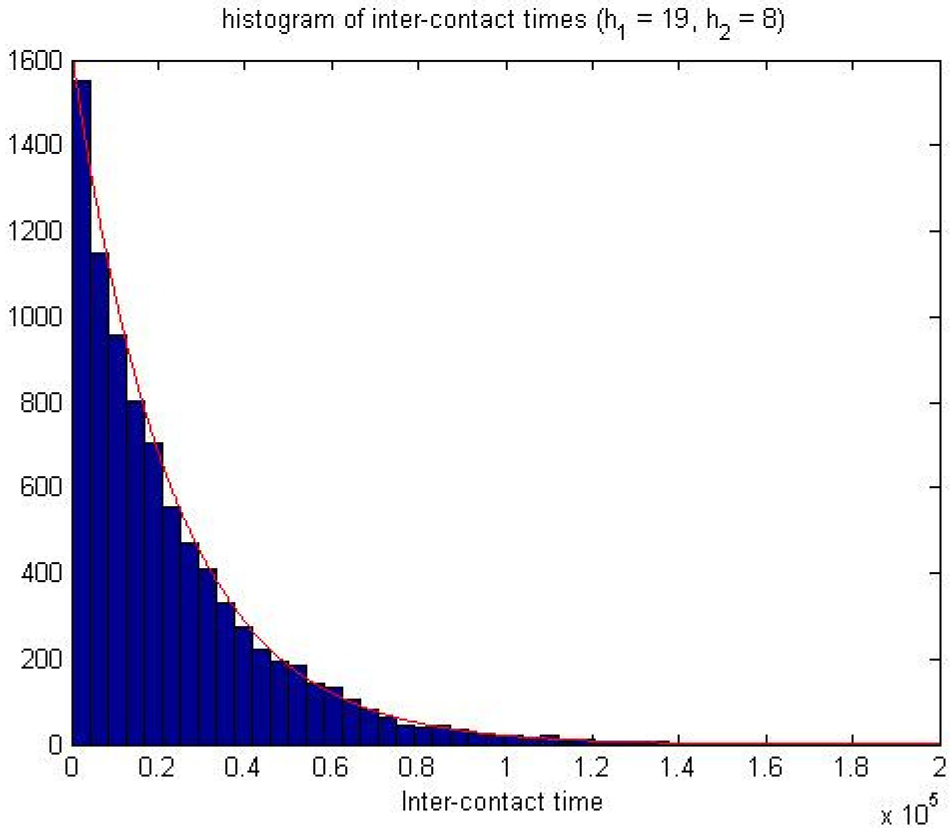
- Implications: for large h_2, inter-contact times $X_k, k \geq 2$, can be approximated using exp. rvs with mean

  $(h_1 \times h_2)^2$

  - True if $\beta < \frac{1}{2}$ in the original HRW model

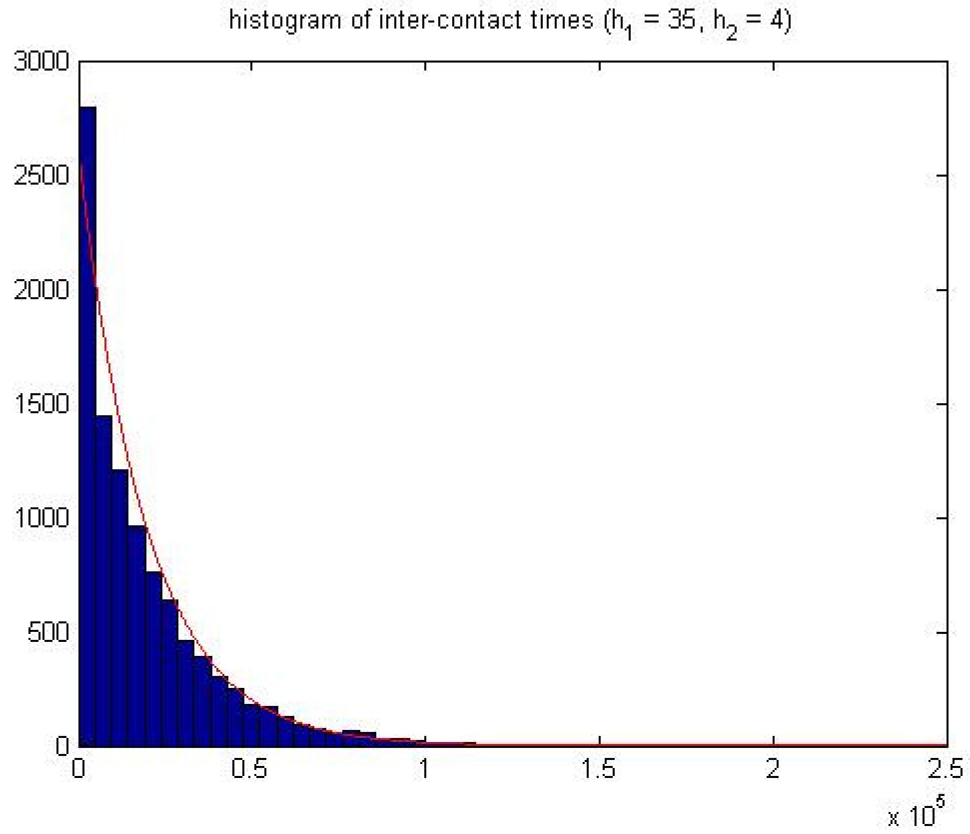
Simulation

- $h_1 = 19$ and $h_2 = 8$



Simulation

- $h_1 = 35$ and $h_2 = 4$



Role of Assumption ii)

- If $h_2(n) \leq B,\ n \geq 1,$ for some finite B, then

$$\mathbf{P}[X_2^n = 1] = \frac{1}{4 \cdot (h_2(n))^2} \geq \frac{1}{4 \cdot B^2}$$

  - Implies that, for all $\epsilon > 0$,

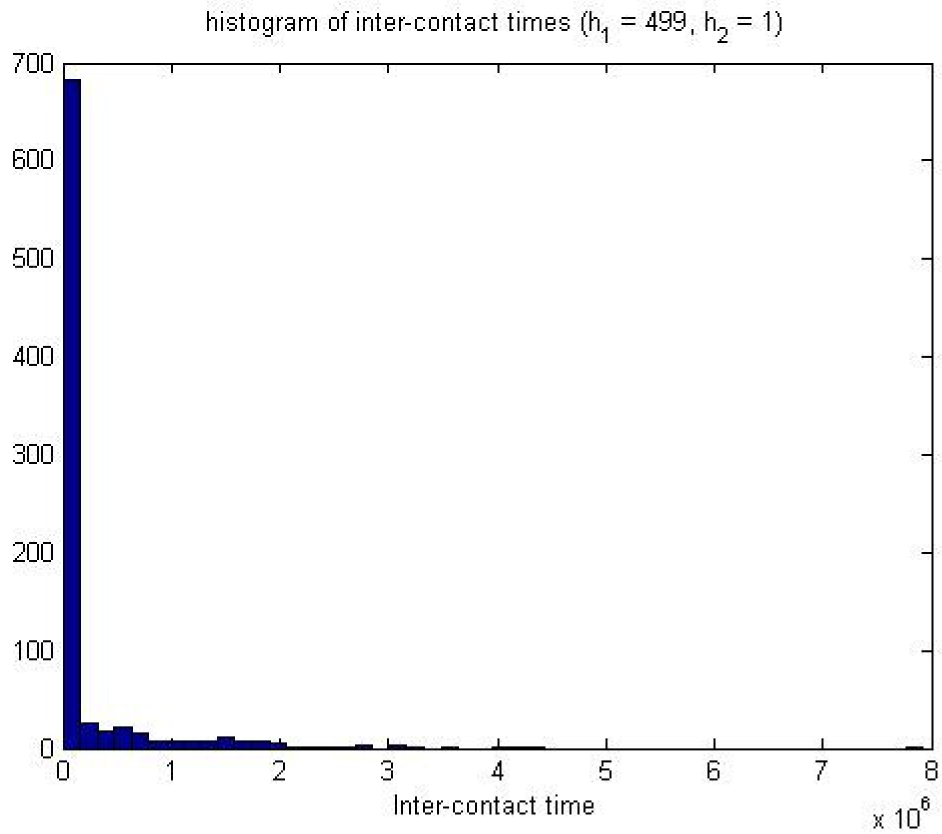
$$\limsup_{n\to\infty} \mathbf{P}\left[\frac{X_2^n}{(h_1(n) \cdot h_2(n))^2} \leq \epsilon\right] \geq \frac{1}{4 \cdot B^2}$$

⇒ $\frac{1}{4 \cdot B^2} > 1 - \exp(-\epsilon)$ for sufficiently small $\epsilon > 0$

⇒ $\frac{X_2^n}{(h_1(n) \times h_2(n))^2} \not\Rightarrow_n \exp(1)$

Simulation

- $h_1 = 499, h_2 = 1$



Ant-based packet forwarding algorithm

- No fixed routing
  - No end-to-end paths available from sources to destinations most of the time
  - Location of destinations and sequence of nodes to traverse unknown in advance
  - When two nodes come in contact with each other, they exchange information and figure out who will forward which packets (if any)
    - Must reflect who has a better chance of (eventually) successfully delivering packets, possibly through other relay nodes, to the intended destinations

Ant-based packet forwarding algorithm

- Set-up
  - Set of mobile nodes move in a compact region in $\mathbb{R}^2$
    - Mobility of the users given by a joint process

      $\mathbf{X} = \{(X_j(t), j \in \mathcal{N}), t \geq 0\}$
      - E.g., Random Waypoint, Random Direction, group mobility models, etc.
  - Connectivity given by a disk model
    - Two nodes i and j connected if $\left\| X_i(t) - X_j(t) \right\| \leq \gamma$
  - Packets delivered to a set of gateways
    - Single commodity (can be trivially generalized to multiple commodities case)

Ant-based packet forwarding algorithm

- Goals:
  - Maximize the packet delivery ratio (fraction of packets delivered to gateways)
    - Finite buffer sizes at nodes
  - Minimize end-to-end packet delays to gateways
    - Capture mobility patterns of the nodes
  - Simplicity of the algorithm
    - Minimal exchange of information when two nodes meet

30 second overview of swarm intelligence and ant routing

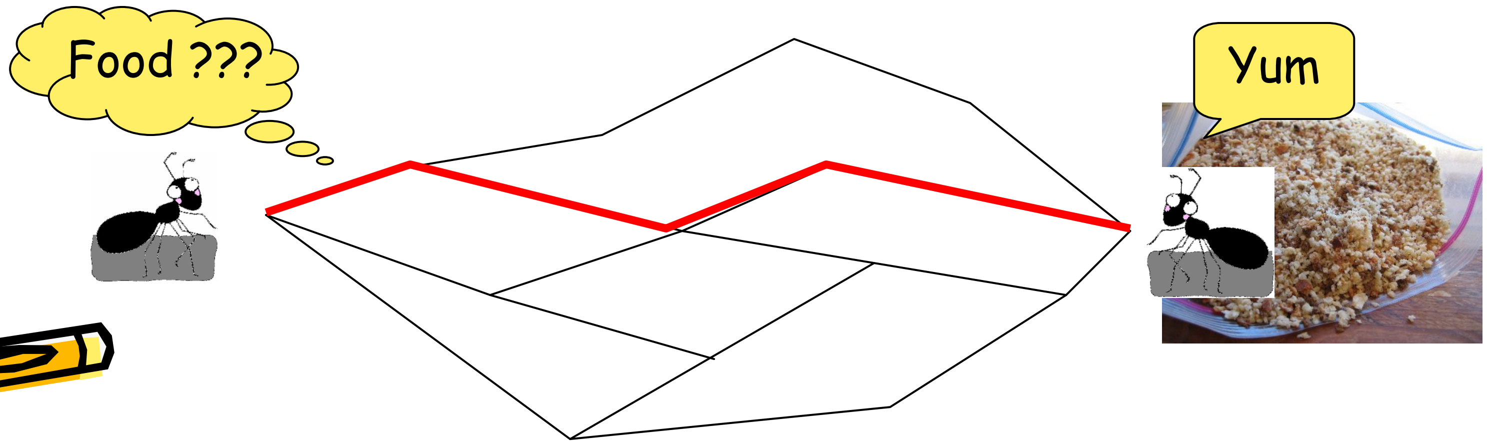
- Premise: Individual insects not so intelligent

- A swarm of insects can solve fairly complex problems
  - Finding shortest path, minimum spanning tree, sorting, task (re-)assignment, graph partitioning, etc.

- Question is …. How do they solve these problems with (supposedly) such low intelligence?
  - More importantly, how do we mimic their behavior to solve engineering problems in a distributed, robust, scalable manner?

Ant Foraging
(or my understanding of it)

- Foraging behavior of ants
  - Exploration – randomly venture out to food sources
  - Backtracking – when returning to the nest, deposit a little bit of chemical called "pheromone" along trail
    - Signals to other ants that there is food down the road
      - A form of indirect communication (called "stigmergy")



Ant Foraging

- Pheromone deposit along a path attracts more ants to follow the path
  - Reinforcement
  - Decay (through evaporation)
    - Allows switches to other paths when preferred paths change or become unavailable

- Ants follow better (shorter) paths more quickly
  - Pheromone strength increases faster for shorter paths

- Ants concentrate on shortest paths after a while

Max. Packet delivery ratio

- Infinite buffer sizes
  - Backpressure algorithm (Tassiulas&Ephremides)
    - Uses queue size differences – equalize queue differentials as much as possible
    - Stable queue sizes, i.e., queues do not blow up
- Finite buffer size
  - Want to mimic the stabilizing behavior of backpressure algorithm
  - Somehow capture and exploit mobility pattern of nodes
    - Fast moving nodes and nodes with smaller (node) delays may be preferred
      - More likely to unload packets faster

Min. E2E delays

- Backpressure algorithm does not explicitly take delay into account
  - End-to-end delays difficult to estimate
    - Reverse path ACK may not be available or even desirable
    - Adds to the load
  - Use local holding times at each node
    - Find a sequence of nodes with smaller holding times for routing
    - Again, nodes that can unload packets quickly likely to experience fewer packet losses as well

Ant-based packet forwarding algorithm

- Each node maintains a pheromone value $\phi_i(t),\ t = 0, 1, \cdots$
  - Pheromone value decays with time (discounting)

$$\phi_i(t) \leftarrow \phi_i(0) \cdot \exp(-\beta t), \qquad \beta > 0$$

  - After a successful transmission of a packet at time t

$$\phi_i(t^+) \leftarrow \begin{cases} \phi_i(t) + \alpha \cdot \Delta(D) & \text{if the packet originated at } i \\ \phi_i(t) + \Delta(D) & \text{if the packet originated elsewhere} \end{cases}$$

where $0 \le \alpha \le 1$, D is holding time of the packet, and

$\Delta : \mathbb{R}_+ \to \mathbb{R}_+$ (where $\mathbb{R}_+ = [0, \infty)$) (e.g., $\exp(-\xi \cdot D)$)

Ant-based packet forwarding algorithm

- Let $\mathcal{L}(t)$ denote the set of groups of unidirectional links that can be activated simultaneously

- For each unidirectional link $\ell := (i, j)$, let

$$\Delta\Lambda_\ell(t) := \frac{Q_i(t)}{\phi_i(t)} - \frac{Q_j(t)}{\phi_j(t)}$$

- A set of links in

$$L^\star(t) := \arg \max_{L \in \mathcal{L}(t)} \sum_{\ell \in L} \Delta\Lambda_\ell(t)$$

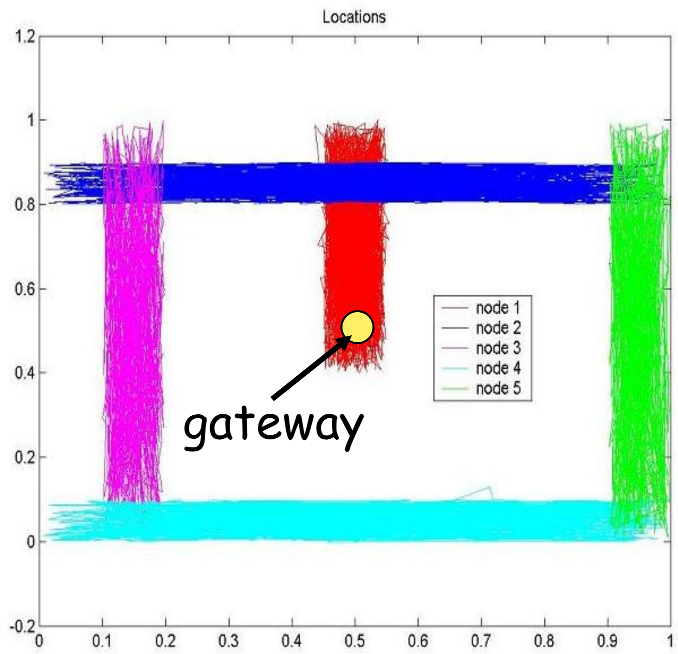
is selected at time t = 0, 1, 2, …

Ant-based packet forwarding algorithm

- Intuition
  - Want to estimate the "extra" capacity for carrying other nodes' packets
    - Credit smaller for forwarding its own packets (through pheromone increment)
  - Prefer nodes with smaller holding times and quicker unloading time
    - Reflects underlying mobility of nodes (through $\Delta(D)$)

Simulation

- Five nodes and a gateway
  - Time slotted into timeslots
    - One packet transmission per timeslot
  - Nodes move according to RWP mobility model
    - Heterogeneous mobility
  - Finite buffer size of 200 packets
  - Packets arrive at each node according to i.i.d. Bernoulli rvs
    - λ_0 = [0.025 0.03 0.04 0.023 0.02]
  - Tx radius of 0.2



Simulation

- Arrival rate given by $\beta \times \lambda_0 = \beta \times [0.0250.030.040.0230.02]$