Course “Algorithmic Foundations of Sensor Networks” Lecture 10: Mobility in WSNs

Sotiris Nikoletseas

Department of Computer Engineering and Informatics
University of Patras, Greece

Spring Semester 2020-2021

1. Sensor Mobility

Motivation

In several emerging applications, sensors are embedded in everyday objects

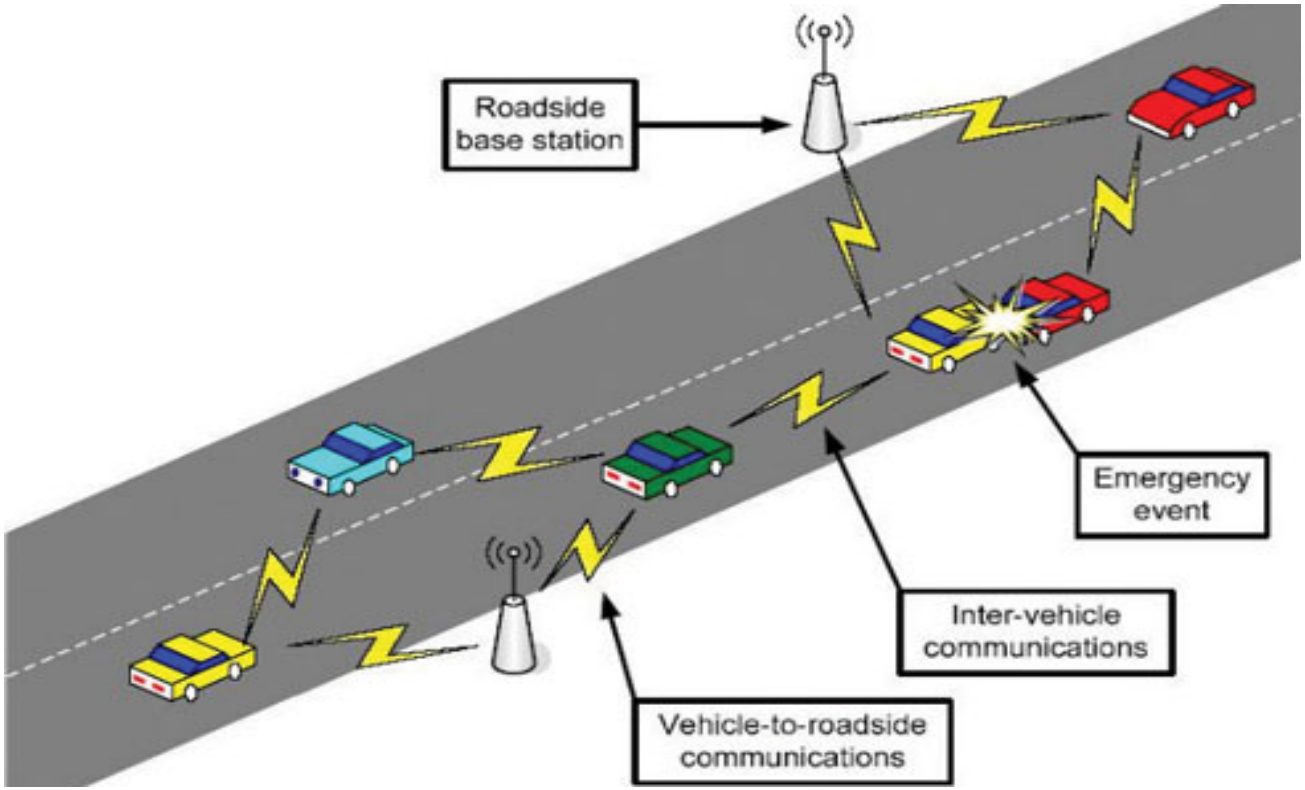
- smart phones, PDAs etc.
- vehicles
- smart clothes

Environmental mobility (water flow, wind etc.)

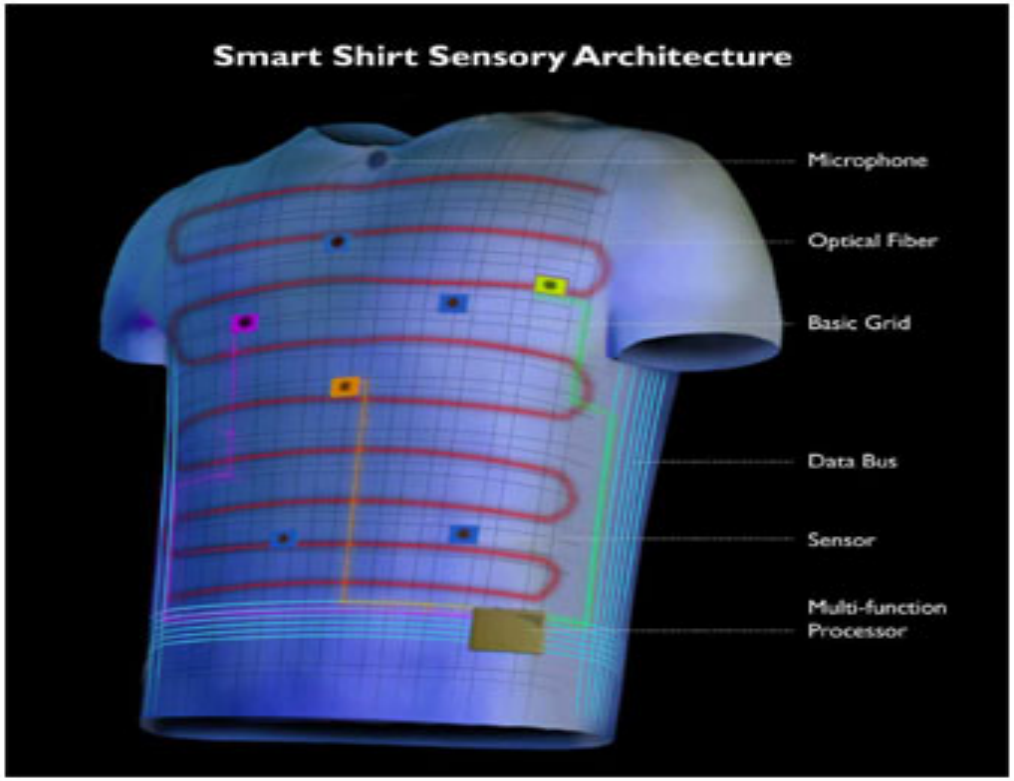
Mobility is a dominant characteristic of the system.

- Sensor nodes are attached to moving objects
- Movement is uncontrollable
- Topology, connectivity changes $\rightarrow$ Multihop routing is extremely expensive and may be even infeasible

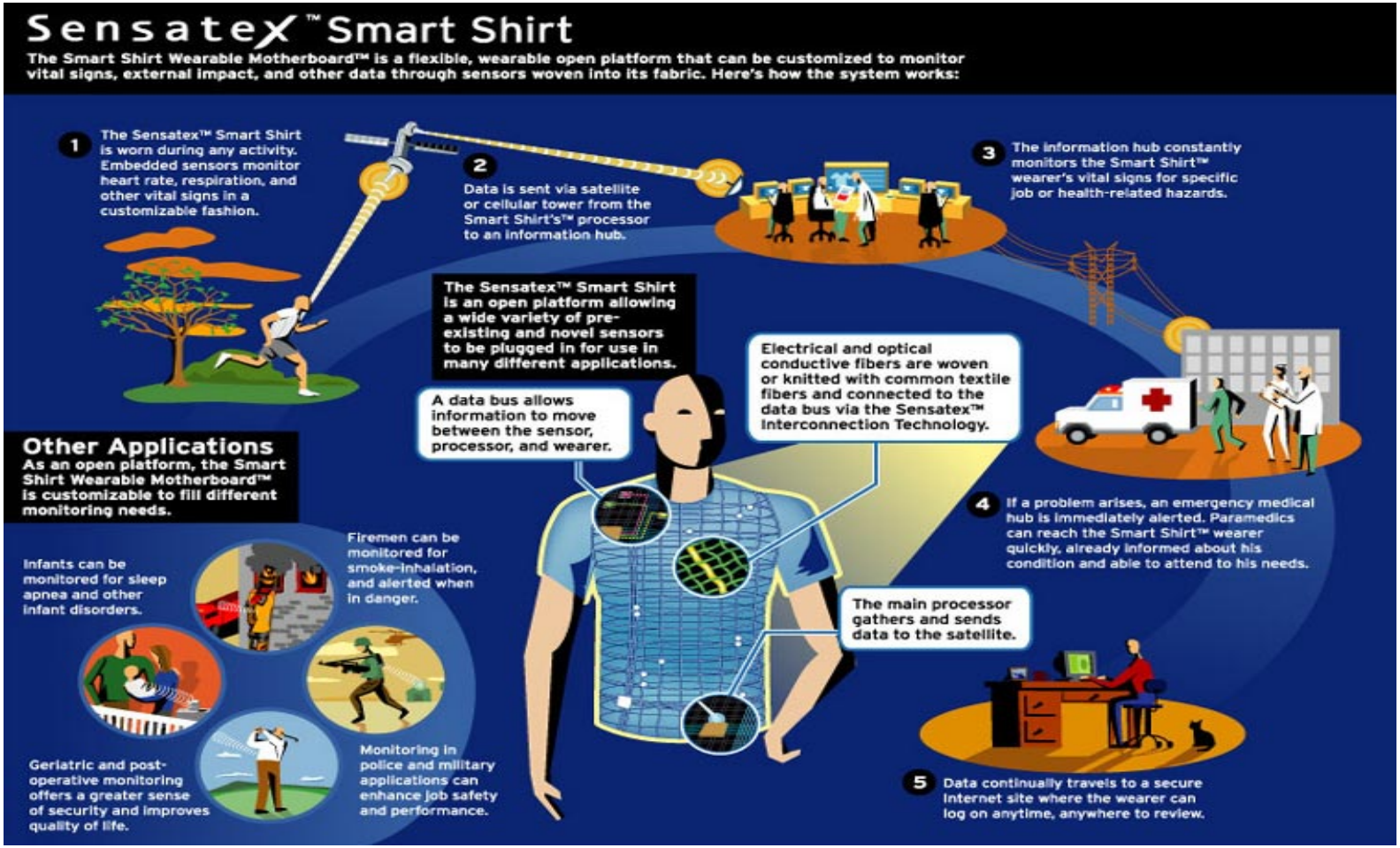
Vehicular Ad-Hoc Networks (VANETs)



Smart Clothes 1/2



Smart Clothes 2/2



Mobility in Wireless Sensor Networks 1/3

Mobile sinks/nodes exchange messages only when communicating directly or over few hops.

New Challenges

- Longer delivery delays
- Bad scalability when network area increases
- Routing and localization problems become more difficult

Mobility in Wireless Sensor Networks 2/3

Advantages of Sink Mobility

- Sparse, disconnected and irregular networks can be better handled
- Communication obstacles can be bypassed
- Better load distribution
- Scales well with respect to number of sensors
- Reduces communication distance
- Reduces energy consumption on the sensors $\rightarrow$ System lifetime increases

Mobility in Wireless Sensor Networks 3/3

Moving elements:

- Sink(s)
- Sensor nodes
- Relay nodes

Controlled Vs. Uncontrolled Mobility

Controlled Mobility

+ Topology Adaptivity
+ Using controlled mobility, resources can efficiently be moved to regions where they are required (Increase lifetime).
+ Delay can be bounded using controlled mobility
- Cost of mechanical movement (robots)
- Complex protocols

Uncontrolled Mobility

+ More “simple” protocols
- No guarantees for data delivery latency
- Non-coverage of certain network areas

Paper

P. Juang, H. Oki, Y. Wang, et al. **Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet**. In *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-X), Oct. 2002.*

What is ZebraNet? Why ZebraNet?

- Collaboration research work between wildlife biologists and mobile network computer scientists
- Tracking nodes (collars) with GPS, Flash Memory, wireless (radio) transceiver, small CPU
- There is no cellular infrastructure in the area of interest
- Peer-to-peer data communication
- Wireless sensor network for wildlife tacking

Design Considerations

- mobile base station
- moving nodes with unknown mobility models
- energy trade-offs

Use of Zebranet

It is important to learn:

- How human development into wilderness areas affects indigenous species there
- What are the migration patters of wild animals and how they may be affected by changes in weather patterns or plant life

In order to learn such details about animals requires both detailed long-term position logs as well as other biometric data such as heart rate, body temperature, and frequency of feeding.

Before Zebranet: VHF Approach

Many previous studies rely on collaring a sample subset of animals with simple VHF transmitters.

Researchers periodically drive through an area with a receiver antenna and listen for pings from previously-collared animals.

Limitations of the above approach:

- Data collection is infrequent and may miss many interesting events
- Data collection is often limited to daylight hours
- Data collection is impossible or severely limited for reclusive species that avoid human contact

Before Zebranet: GPS Approach

More sophisticated trackers use global positioning systems (GPS) to track position and use satellite uploads to transfer data to a base station.

Limitations of the above approach:

- Satellite uploads are slow and power-hungry
- Downloads of data from the satellite to the researchers are both slow and expensive

Design Goals

Zoologists' requirements

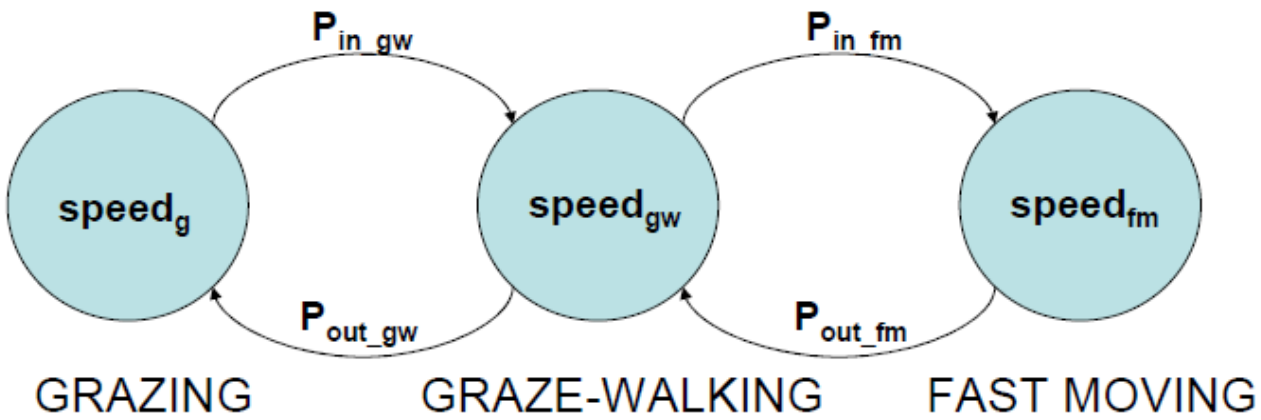
- GPS position samples, every 3 minutes
- Activity logs, taken 3 minutes every hour
- 1 year of operation with no human intervention
- Operate over thousands of square kilometers
- No fixed base station, antennas, cellular network
- High delivery rate of data logs (Latency is not critical)
- Limited collar weight (e.g. 3 -5 lbs for zebra collar)

Implications to design

- Weight limit, energy limitation
- Transmission range
- Storage capacity

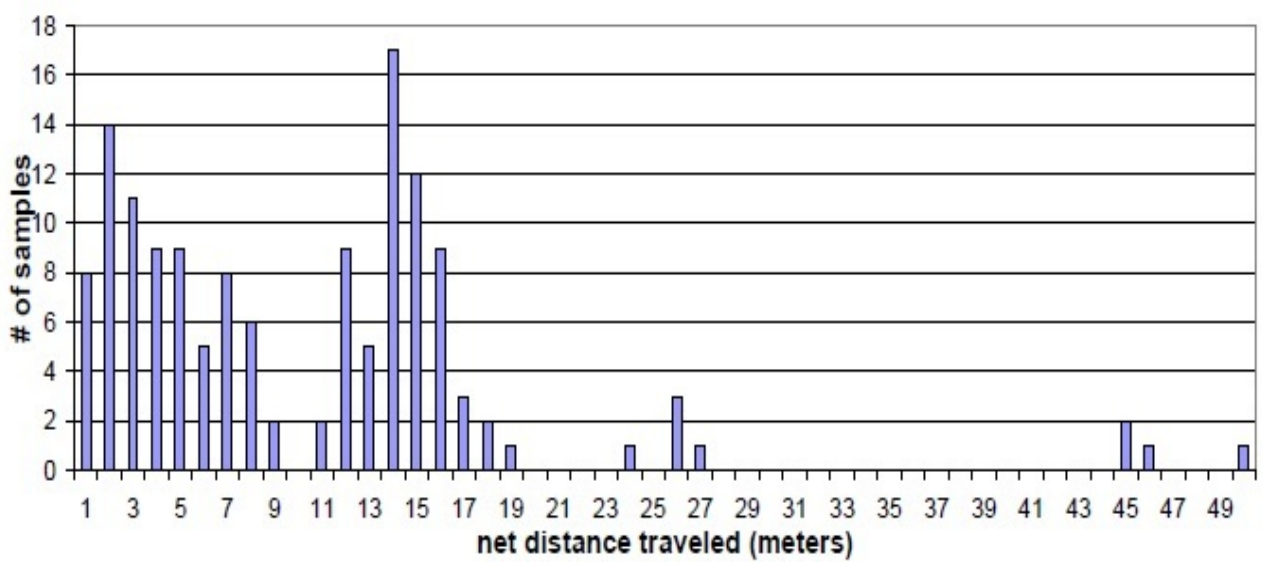
Effect of Mobility

- Nodes (collars) fitted on zebra
- To understand node mobility requires understanding of how fast, in what direction and with what forces of attraction/repulsion zebras move.
- Movement patterns: grazing, graze-walking, fast - moving



Distance Traveled

- Every sample indicates the net distance traveled in a 3 minute interval
- Zebras tend to move very slowly, as they spend most of their time simply grazing



Collar Design

Design goals

- Total weight 3-5 lbs
- Energy 5 days of no recharge
- Battery rechargeable using solar cell

Amount of data

- 30 coordinates per hour
- 240 bytes per hour
- 1 Collar-day 6KB

Collar Design

- GPS, Short/Long Radio, Flash Ram and CPU.

Protocol Design

ZebraNet Characteristic

- Not every collar is within range of the base station (hop by hop communication)
- The nodes (collar) move around almost constantly
- Base station is also mobile
- Base station is active from time to time
- High success rate is important (latency is not critical)

Protocol Strategies

- Flooding protocol
- History-based protocol

Flooding protocol

- Flood data to all neighbors whenever they are discovered
- Nodes that move extensively and meet a fair number of the nodes (highly - interactive), then given enough time, data will eventually migrate back to the base
- Base station does not necessarily have to come into contact with all the nodes in the system (by identifying a few highly - interactive nodes, we can collect a substantial amount of data readily)
- Large amount of data can lead to excessive demands for bandwidth, storage and energy

History-based protocol

- After peer discovery, choose at most one peer to send to per discovery period: the one with best past history of delivering data to base
- Can reduce amount of data in network
- ZebraNet is very dynamic (both collars and base station are mobile). Then, this protocol may mis-direct traffic and get a poor success rate

Experimental Results

The most important factors

- storage
- bandwidth

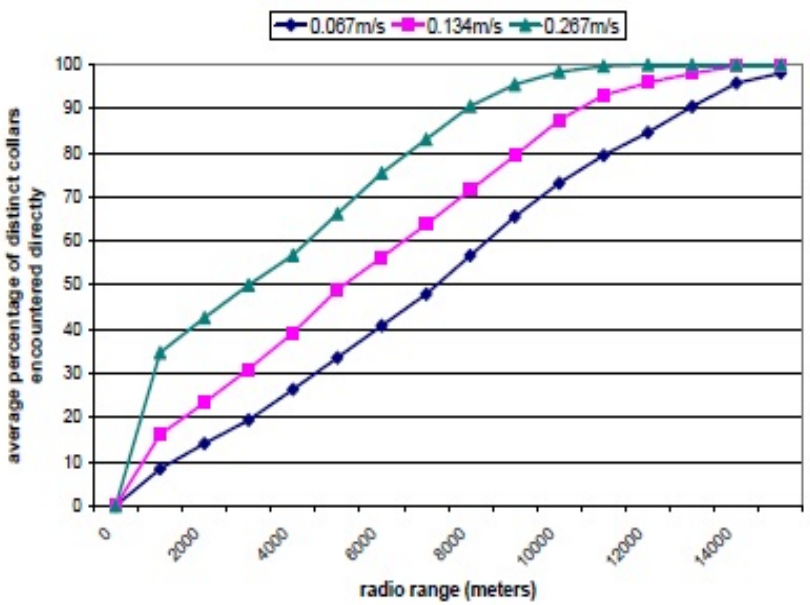
The most important metrics

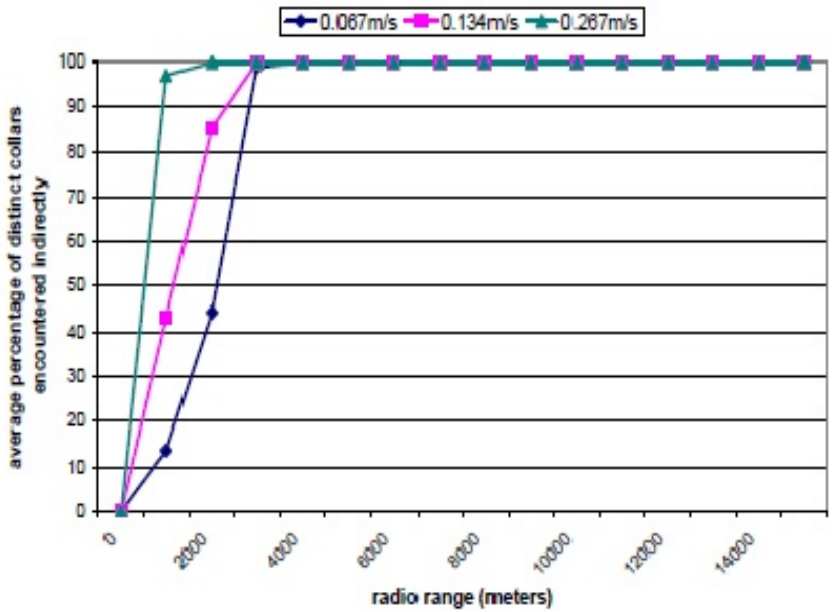
- success rate
- energy

Network Connectivity

Two metrics of connectivity

- Direct connectivity
- Indirect connectivity

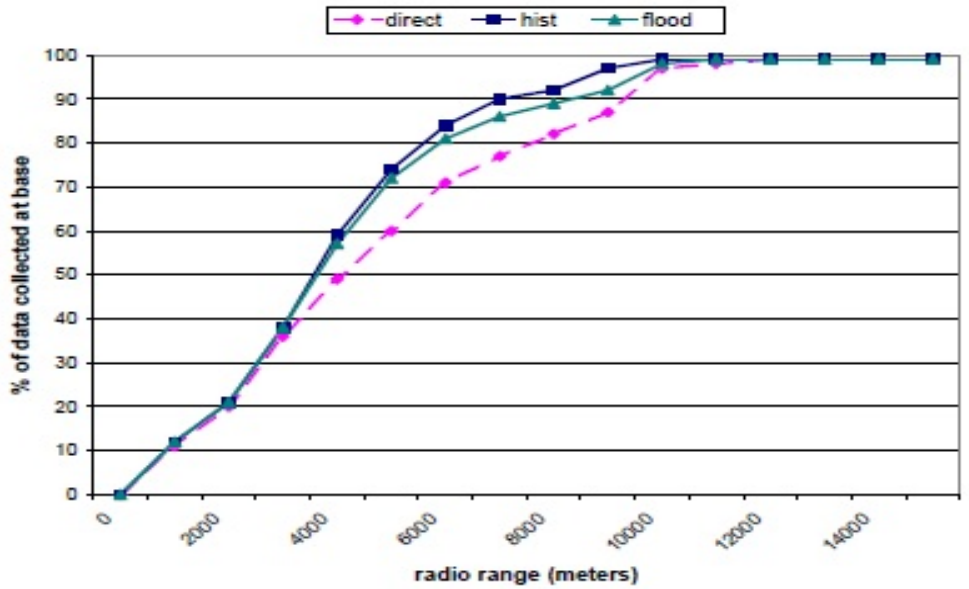




Storage Constraints

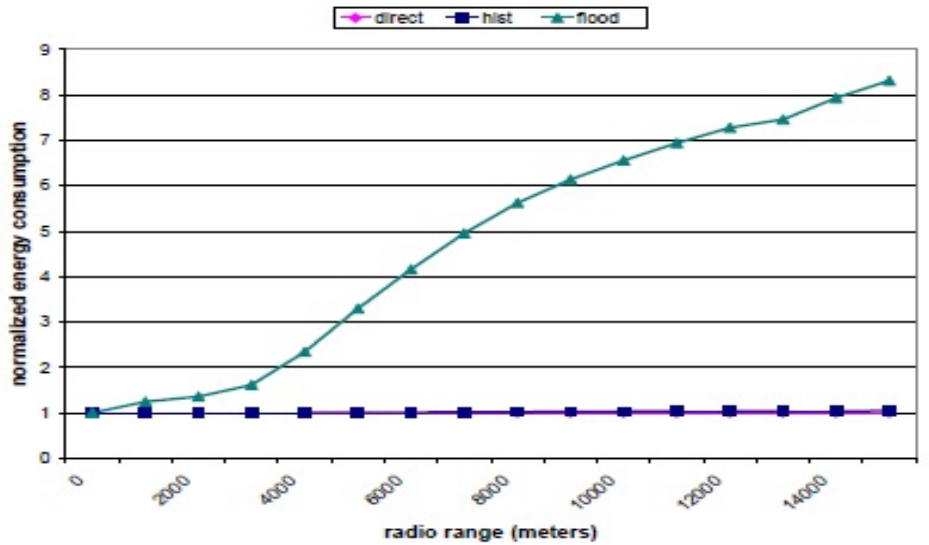
We assume constrained storage and infinite bandwidth.

- The peer-to-peer protocols perform better than the direct protocol. This is surprising since these protocols require that the storage handle both the collar's own data and of their peers.
- The deletion strategy followed prioritizes a node's own data over others. At worst, a protocol stores only its own data.



Energy Tradeoffs

- While flooding makes sense at low-radio-range and low-connectivity points, it is a poor choice for the high-connectivity regime
- The history-based and the direct protocol have similar performance



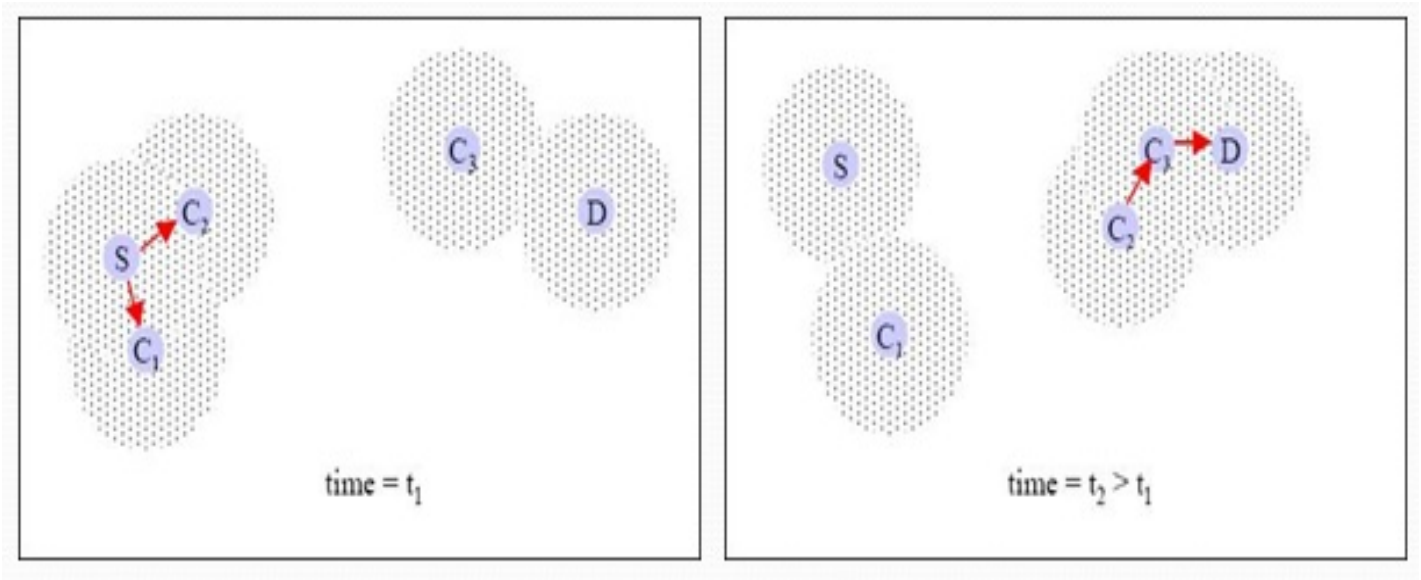
Delay tolerant networks

- Delay tolerant networks (DTNs) are occasionally-connected networks that may suffer from:
  - Network Partitioning
  - Network Interruptions, Failures and Heterogeneity
  - Asymmetric, Long and Variable Data-Rates
  - Energy, Bandwidth, Buffer and Cost Restrictions
- Representative DTNs include:
  - Remote Area Networks
  - Military Battlefield Networks
  - Mobile Sensor Networks

Routing in Delay tolerant networks

Keyword: Epidemic Routing (flooding in a disconnected context)

Goal is to deliver messages with high probability even when there is never a fully connected path.



Routing in Delay tolerant networks

The overall goal of Epidemic Routing is to

- maximize message delivery rate
- minimize message latency
- minimize the total resources consumed in message delivery

A mobility based approach

Main idea:

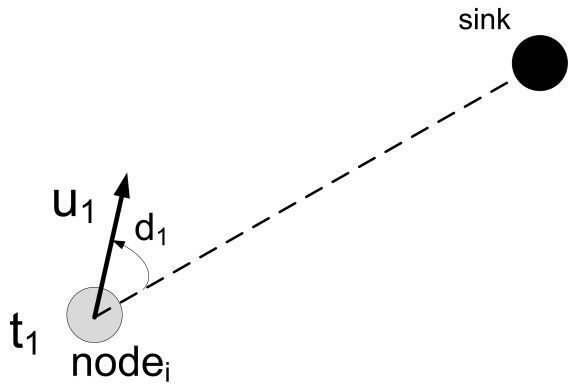
- mobility replaces connectivity
- "fast" nodes are more capable at ferrying data
- "slow" nodes have to transmit their data in order toaccelerate data propagation

Our approach:

- we propose a new (locally computable) network parameter,thedirection-aware mobility level
- we exploit sensory mobility as a low energy replacement forconnectivity and data propagation redundancy
- we propose a progress-sensitive message flooding inhibitionscheme

Estimating Mobility Level

- every t_l seconds node i records its speed and the angle $d_i(t)$ between its direction of movement and the line connecting the current position
- Let v_i(t) be the exponential weighted moving average (EWMA) speed of the last K samples
- Let d_i(t) be the EWMA direction of the last K samples

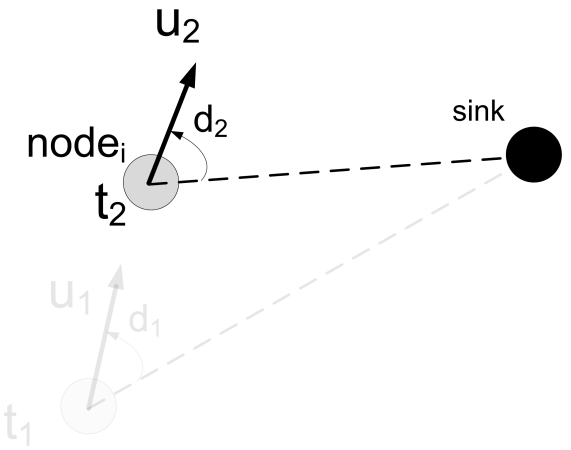


The mobility level of node i at time t is calculated as :

$$Ml_i(t) = v_i(t) * \left(1 - \frac{d_i(t)}{\pi}\right)$$

Estimating Mobility Level

- every t_l seconds node i records its speed and the angle $d_i(t)$ between its direction of movement and the line connecting the current position
- Let v_i(t) be the exponential weighted moving average (EWMA) speed of the last K samples
- Let d_i(t) be the EWMA direction of the last K samples

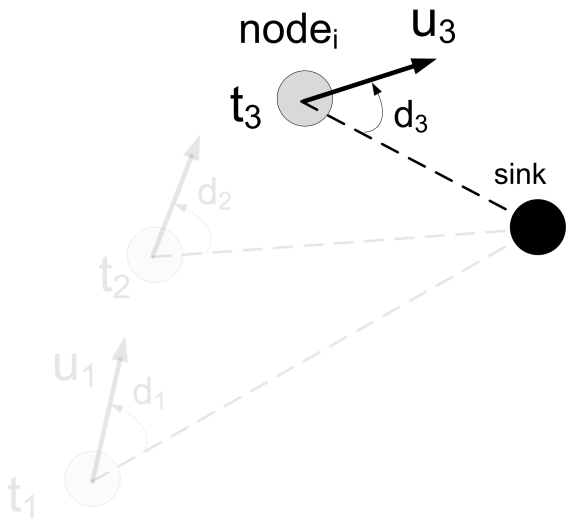


The mobility level of node i at time t is calculated as :

$$Ml_i(t) = v_i(t) * \left(1 - \frac{d_i(t)}{\pi}\right)$$

Estimating Mobility Level

- every t_l seconds node i records its speed and the angle $d_i(t)$ between its direction of movement and the line connecting the current position
- Let v_i(t) be the exponential weighted moving average (EWMA) speed of the last K samples
- Let d_i(t) be the EWMA direction of the last K samples



The mobility level of node i at time t is calculated as :

$$Ml_i(t) = v_i(t) * \left(1 - \frac{d_i(t)}{\pi}\right)$$

The Dissemination Scheme 1/3

Intuition:

- Fast nodes that move in "good" direction are more appropriate for message **ferrying**
- Slow nodes that move in "bad" direction should choose:
  - to transmit data **redundantly** to a number of direct neighbors
  - or to make a long **jump** by transmitting data to a long neighbor
- Redundant transmissions increase the likehood of message delivery
- With the "expensive" jump transmission we can overcome the "trap" consisting of nodes with low mobility level and make large progress towards the sink

The Dissemination Scheme 2/3

General dissemination protocol:

- Disconnected operation (no contact to sink):
  - New events or received messages enter a forward queue.
  - Decision Criterion: Node pops the next message from the front of the forward queue and decides to act suitably according to our decision criterion which is described in following slides.
  - Each message enters the forward phase only once.
- Connected operation (within range from sink):
  - Queued messages are forwarded to the sink.
- Queues are FIFO.

The Dissemination Scheme 3/3

Decision Criterion:
The node pops the next message from the forward queue and decides to act suitably according to the following scenarios:

- **Data Ferrying** : If the node has high mobility level, it decides to ferry/carry the data.
- **Data Transmission** : If the node is not ideal to ferry/carry the data, it transmits data using one of the following choices:
  - **Redundancy** : If at least one direct neighbor of the node has a high mobility level, the node disseminates the data to β of its neighbors.
  - **Jump** : If all of the node's direct neighbors have low mobility level, the node transmits to a neighbor TRi-hop closer to sink in order to avoid the "trap" ("bad" neighborhood).

Calculation of data redundancy β

We propose two methods for selecting the number of neighbors β to disseminate a message:

a) **Completely local protocol :**

$$\beta_i = \left\lceil \left(1 - \frac{Ml_i(t)}{Ml_{max}}\right) \cdot \left(\frac{D_i}{D}\right) \cdot \delta_1 \right\rceil$$

Where D_i is the distance from sensor i to the sink, and δ_1 represents the maximum redundancy: $\delta_1 = \left\lfloor \frac{dist_{sink}(j)}{R} \right\rfloor$

$$ML_{max} = \upsilon_{max} \cdot 1 = \upsilon_{max}$$

- $0 \leq \beta_i \leq \delta_1$
- When Ml_i increases, β_i decreases, and vice versa.
- When D_i increases, β_i increases, and vice versa.

b) **Neighbor discovery protocol :**

$$\beta_i = \left\lceil \left(1 - \frac{Ml_i^{avg}(t)}{Ml_{max}}\right) \cdot \left(\frac{D_i}{D}\right) \cdot \delta_1 \right\rceil$$

$Ml_i^{avg}(t)$ captures the mobility at the neighborhood of node i at time t.

$$Ml_i^{avg}(t) = \frac{\sum_{j \in neigh_i(t)} Ml_j(t)}{|neigh_i(t)|}$$

- $0 \leq \beta_i \leq \delta_1$
- When Ml_i^{avg} increases, β_i decreases, and vice versa.
- When D_i increases, β_i increases, and vice versa.

The rationale is to calculate large values of β for "slow" moving in "bad" direction and distant from the sink nodes.
The opposite happens for "fast", moving in "good" direction and close to the sink nodes.

Calculation of length of jump $\mathbf{TR}_i$

$$TR_i = \left\lceil \left(1 - \frac{Ml_i(t) + Ml_i^{avg}(t)}{Ml_{max}}\right) \cdot \left(\frac{D_i}{D}\right) \cdot \frac{\delta_1}{2} \right\rceil$$

- $0 \leq TR_i \leq \frac{\delta_1}{2}$
- When Ml_i^{avg} increases, β_i decreases, and vice versa.
- When D_i increases, β_i increases, and vice versa.

The rationale is to calculate large values of TR_i for "slow", moving in "bad" direction, distant from the sink nodes which are in relatively "bad" neighborhood.

Neighbor Selection

Our protocol has to do neighbor selection in two cases, when selecting:

- direct neighbors in order to do redundancy
- when jumping to a long neighbor so as to avoid bad neighborhod

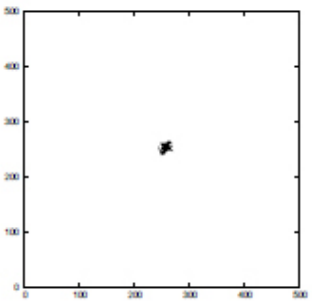
Direct Neighbor Selection

- **Completely Random Selection.** Select β_i random neighbors.
- **Fittest Candidate Selection.** Select β_i neighbors such that $Ml_i(\mathrm{t}) < Ml_j(\mathrm{t})$
- **Probabilistic Candidate Selection.** Select β_i neighbors with probability p_j

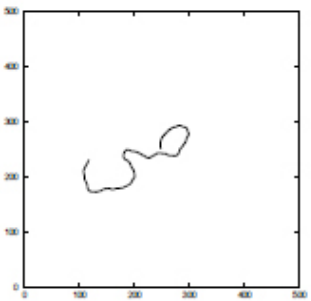
$$p_j = \begin{cases} \frac{Ml_j(t)}{Ml_i(t)} & Ml_j(t) \leq Ml_i(t) \\ 1 & Ml_j(t) > Ml_i(t) \end{cases}$$

Diversifying Mobility

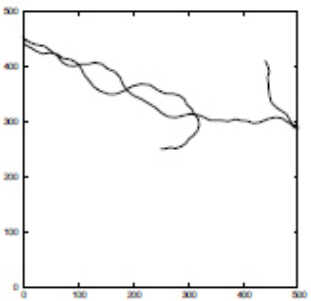
By choosing appropriate mobility models and parameters we define 4 mobility patterns each resembling a particular mobility role: M_{work}, M_{walk}, M_{bic}, M_{veh}



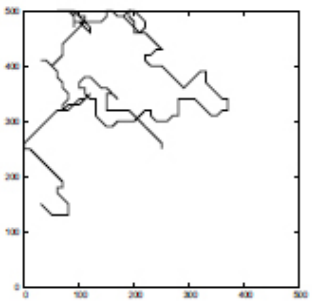
a) $\mathcal{M}_{work}$



b) $\mathcal{M}_{walk}$



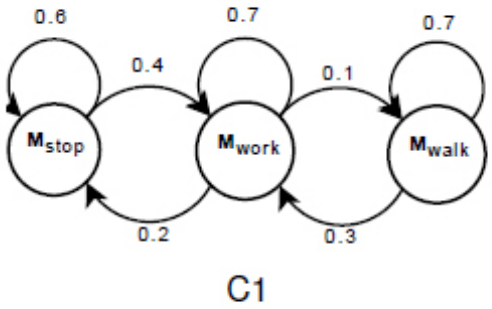
c) $\mathcal{M}_{bic}$



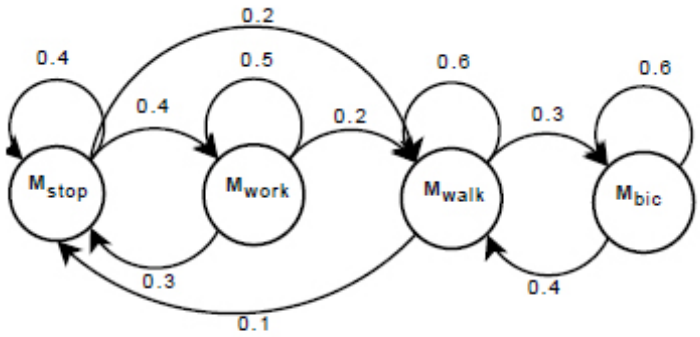
d) $\mathcal{M}_{veh}$

Mobility Transitions

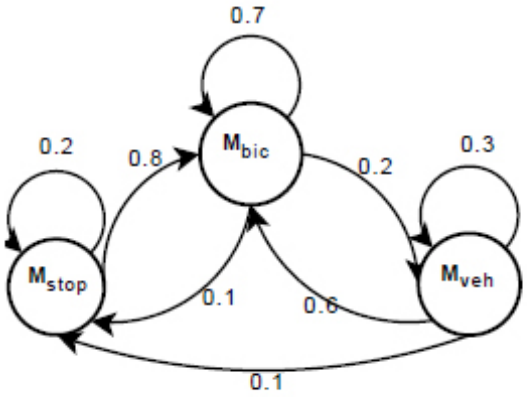
- Mobility role may vary with time
- Mobility transition diagram
  - Each vertex represents a mobility model
  - Edges are associated with a probability of transition



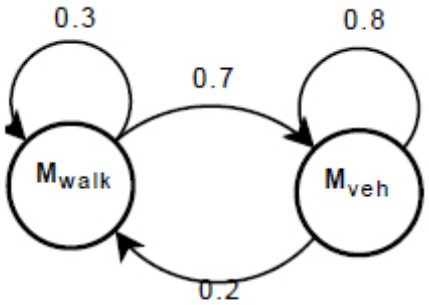
C1



C2



C3



C4

Experimental Setup 1/3

Network Simulator (ns-2 version 2.33) and TRAILS extensions

- We implement a flooding protocol $\beta_i = \infty$
- We implement a gossiping protocol
- We implement our adaptive Direction Sensitive Mobility Protocol (DSMP)
- We implement Adaptive Mobility Protocol (AMP) which is an other adaptive redundancy protocol
- 1000m $\times$ 1000m
- λ=0.025 events/sec, 20 events per sensor
- default transmission range R = 70m
- Sink is positioned at (500, 500)

Experimental Setup 2/3

- Node densities:
  - 300 for the first set of experiments
  - 50, 100, 150, 200, 250 and 300 for the second set of experiments
- Node movement:
  - 25% follow M_{work}, 25% follow M_{walk}, 25% follow M_{bic}, 25% follow M_{veh}
  - 25% follow **C1**, 25% follow **C2**, 25% follow **C3**, 25% follow **C4**

We measure

- Success Rate P_{rs}
- Energy Dissipated E_{tot}
- Delivery Delay D

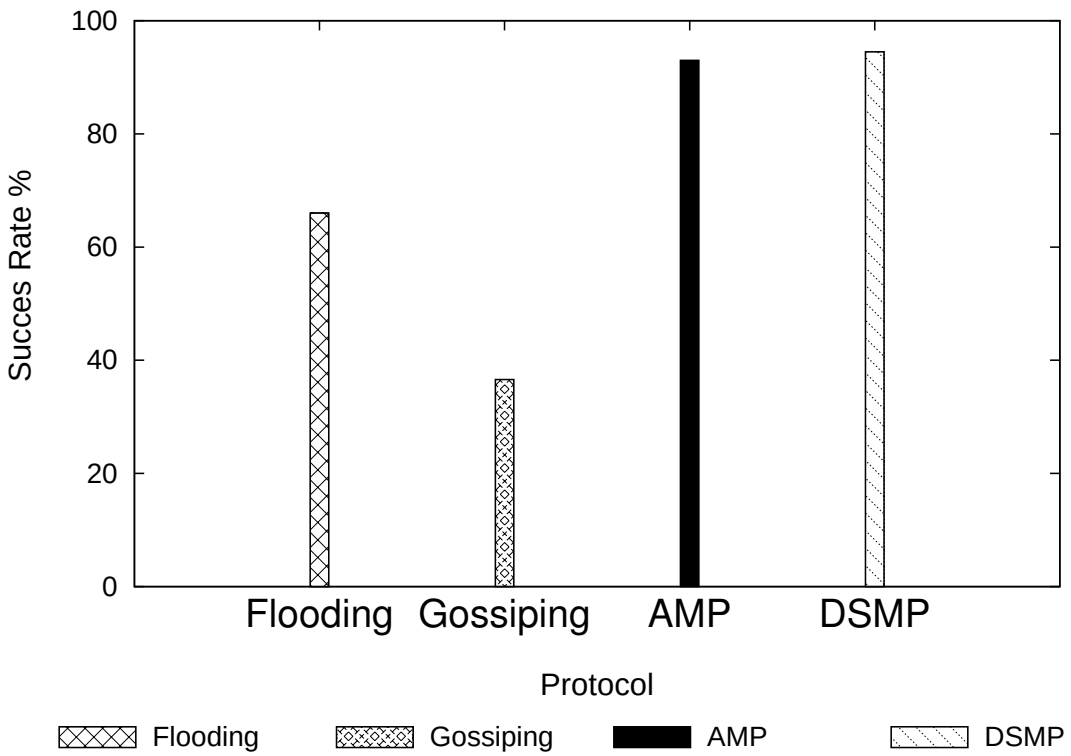
Experimental Setup 3/3

Two set of experiments:

1st set We compare our DSMP protocol with flooding, gossiping and AMP

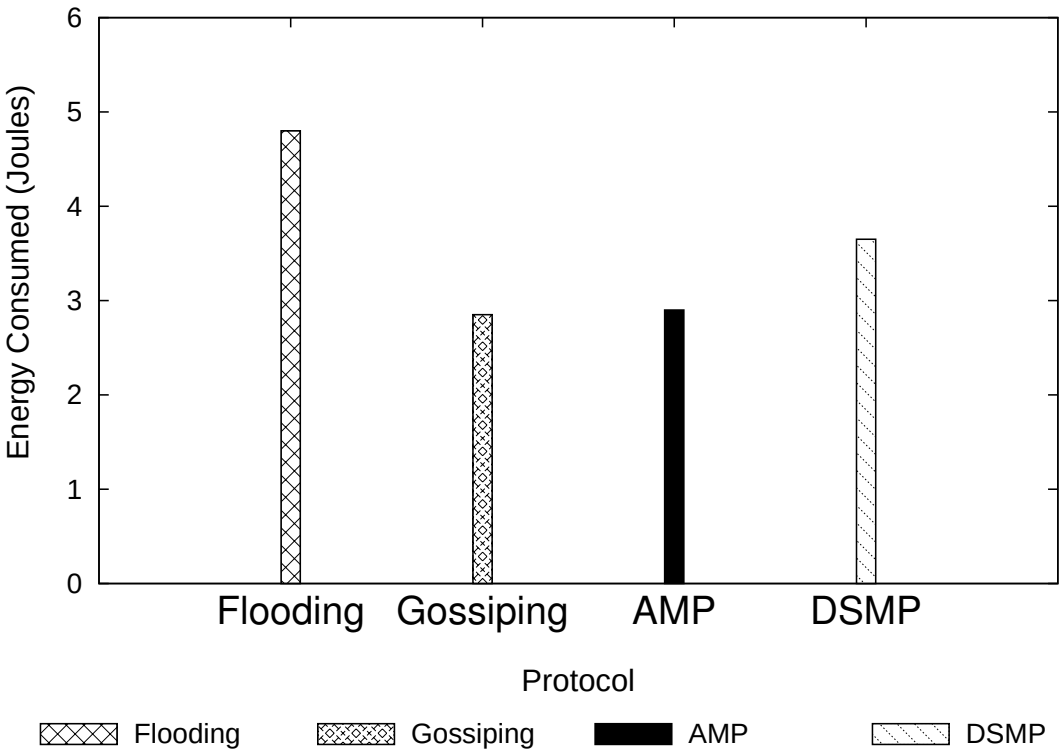
2nd set We investigate the impact of density to the two adaptive protocols (AMP, DSMP)

Mixed Roles: Success Rate



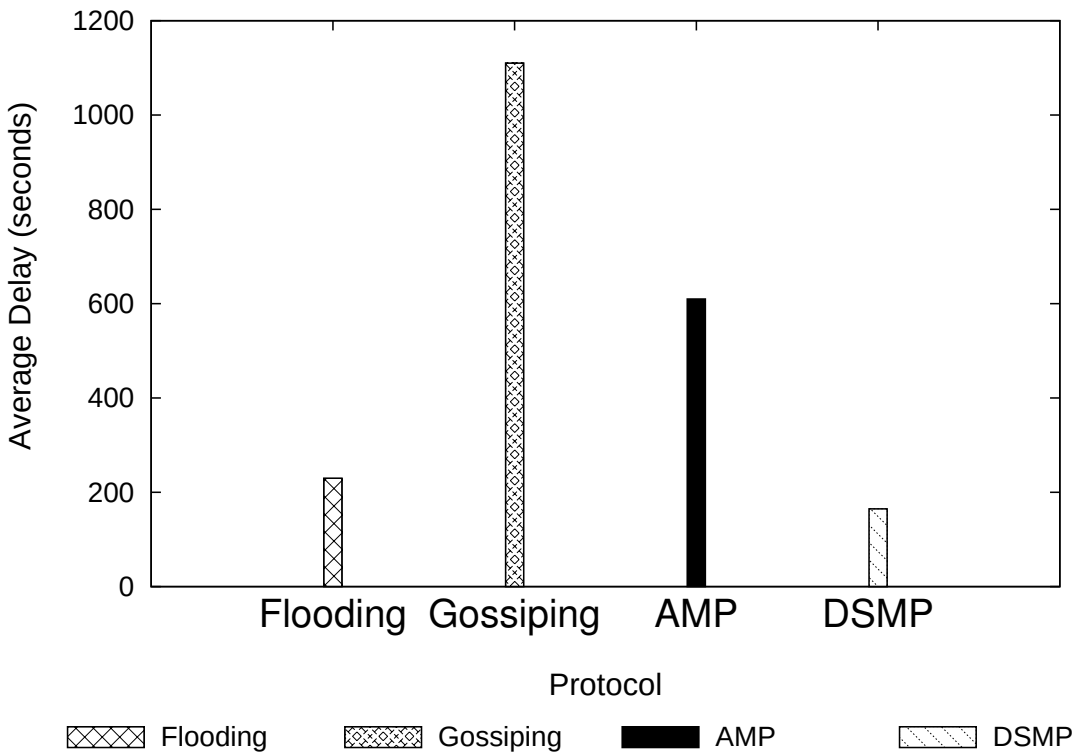
- Gossiping achieves the lowest success rate because of its unbound randomness
- Flooding achieves low success rate due to the limited buffers
- The adaptive protocols (AMP and DSMP) achieve highest success rate

Mixed Roles: Energy Dissipation



- Flooding consumes the highest amount of energy
- Our DSMP protocol consumes about 22% more energy than AMP protocol, due to long transmissions

Mixed Roles: Delay



- Our DSMP protocol achieves lower latency
- Flooding achieves low latency, but higher than our DSMP protocol
- Gossiping achieves the highest average delay among all protocols

Conclusions

- the mobility parameter captures the ability of a node to arrive close to the sink quite fast
- ferrying serves as a low cost replacement for data dissemination
- in the case of "low" mobility either:
  1. data propagation redundancy is increased
  2. long-distance data transmissions are used to accelerate data dissemination

2. Sink Mobility

Limitations of Static Sink(s)

- Computation and communication (energy) overhead *on the sensors*
- *Low Scalability*
- *Uneven load distribution* among the sensors
- Bypassing obstacles requires a lot of resources
- Dynamic networks have *great reconfiguration cost*

Mobility: Main Idea (1/3)

Advances in technology and new applications suggest that sensors may be mobile.

- Wildlife monitoring
- Monitoring in urban environments

Recently, a new approach has been developed *that shifts the burden from the sensor nodes to the sink*

Main Idea:

- Sink has significant and easily replenishable energy reserves
- The sink can move inside the sensor network area, in close proximity to the sensors
- By travelling in the whole network area, sink collects all the available data

Mobility: New Challenges (2/3)

Sink mobility in WSNs presents *many new challenges*

- Sink must *cover* the whole network
- Incurs *longer delivery delays*
- *Bad scalability* when network area increases
- Routing and localization problems become *more difficult*

Mobility: Advantages (3/3)

- *Sparse, disconnected and irregular networks* can be better handled
- The mobile sink can *bypass obstacles*
- *Better load distribution*
- *Scales well* with respect to number of sensors
- Reduces communication distance
- *Reduces energy consumption on the sensors* $\Rightarrow$ System lifetime increases
- Reduces adversarial overhearing $\Rightarrow$ *Enhances security*

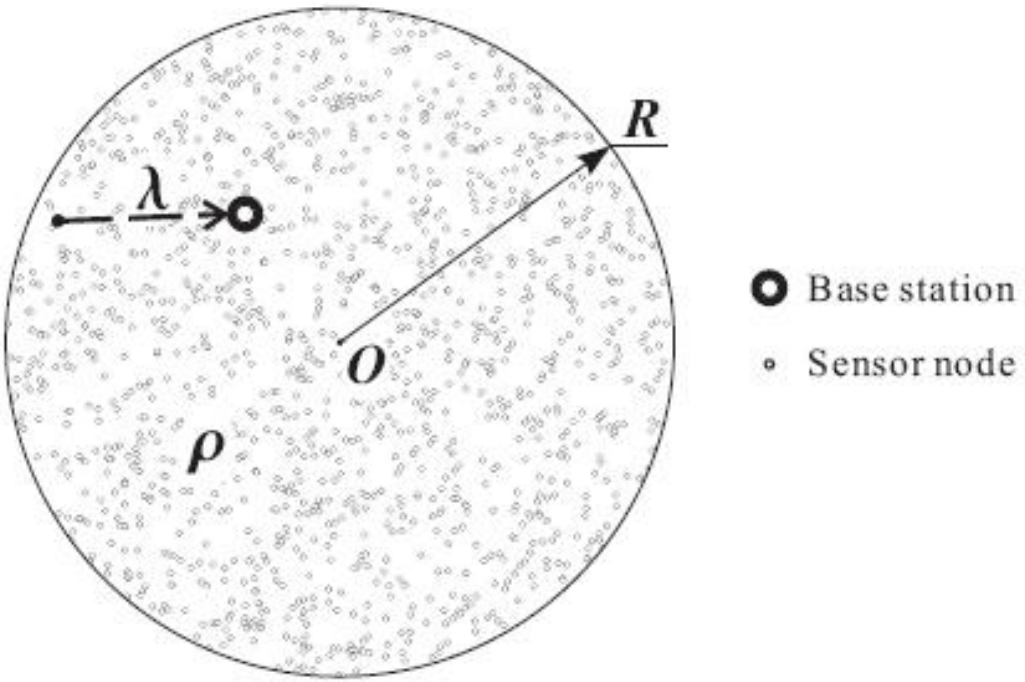
"**Joint Mobility and Routing for Lifetime Elongation in Wireless Sensor Networks**", Jun Luo and Jean-Pierre Hubaux, IEEE INFOCOM 2005

Problem

Let the Sink be mobile.
What is the optimal trajectory in order to traverse the network?

The Model

- A relatively dense and strongly connected network
- N *static* sensor nodes and one base station that collects data from all nodes
- nodes are distributed as a *Poisson process* with density ρ within a circle *COR* of center *O* and radius *R*.



Optimum Sink Position

Networks with a Static Base Station

- The centre of the circle

Networks with a Moving Base Station

- A periodic movement
- rotation symmetry of all degrees around the network center

Heuristically, The optimum symmetric strategy is the one whose trajectory is circle COR.

Via experimentation $\Longrightarrow$ Periphery of the circle

"**Coverage-Adaptive Random Walks for Fast Sensory Data Collection**", Constantinos - Marios Angelopoulos, Sotiris Nikoletseas, Dimitra Patroumpa, and Jose Rolim

Random Walks in WSNs

Random walks can serve as *fully local*, *very simple strategies* for sink motion, *reduce energy dissipation* a lot but increasing latency.

To achieve satisfactory energy-latency trade-offs the sink walks can be made adaptive, depending on local network parameters such as density and/or history of past visits in each network region.

The Network Model

Network

- Planar area
- Sensors deployment is random uniform over the network area
- All sensors have sensory data to deliver
- No data is generated during the network traversal
  (focus on data collection)
- Visited nodes are distinguished by lack of data.

Sink

- During the network initialization, a graph formation phase is executed by the sink
- The network area is partitioned in $j \times j$ equal square regions, called cells.
- A virtual lattice graph $G_o = G(V, E)$ is created which is overlayed over the network area.
- When the sink is located at the center of a cell, it can communicate with every sensor node within the cell area.

Blind Random Walk 1/2

- The most simple and straightforward of all.
- Each move is stochastically independent from all previous ones
- The sink selects its next position with the same probability for each one of the four coordinate directions.

Blind Random Walk 2/2

- Very robust solution
- Probabilistically guarantees that eventually all the cells of the network will be visited
- All data will be collected

However, in some network structures it may become inefficient, mostly with respect to latency.

Random Walks with Memory 1/2

Extend Blind Random Walk. Uses some (constant) memory of past visits.

- The sink maintains a First-in-First-out (FIFO) list M
- M contains the last K cells visited during the random walk.
- The next hop is chosen uniformly among the neighbors of the cell that are not in the memory list M

Random Walks with Memory 2/2

Trade-off: The use of memory eliminates loops in random walks, but it may also lead to a deadlock.

- If $K = 0$, the random walks become blind and can have loops but no deadlocks.
- For complete memory, the random walks can only have deadlocks and no loops.
- When the size of M is $0 \leq K \leq n - 1$, random walks can have both loops and deadlocks.

Random Walk with Inertia 1/2

- Four directions: North, South, East and West
- Each one is assigned a probability.
- the probability distribution on each step of the walk changes adaptively to the nodes' discovery

The following principle is followed:

Reinforce the direction where newly discovered sensors were found and weaken the direction where already visited sensors have been located.

Random Walk with Inertia 2/2

p_c: current direction probability at time t is:

$$p_c^{t+1} = \begin{cases} p_c^t + \delta & \text{, if new nodes discovered} \\ p_c^t - \delta & \text{, if no new nodes discovered} \end{cases}$$

while each one of the probabilities towards the rest three directions (p_r: rest direction probabilities) are:

$$p_r^{t+1} = \begin{cases} p_r^t - \frac{\delta}{3} & \text{, if new nodes discovered} \\ p_r^t + \frac{\delta}{3} & \text{, if no new nodes discovered} \end{cases}$$

Explore-and-Go Random Walk

The sink motion consists of two types of motion:

- moving on a straight line
- arbitrarily changing direction

The sink chooses between them via a *bias factor* β.

$$F_{motion} = \begin{cases} \textit{move straight, with probability } \beta \\ \\ \textit{change direction, with probability } 1-\beta \end{cases}$$

$$\text{where, } \beta = \begin{cases} 0.1 & \text{, when new nodes were discovered} \\ \\ 0.9 & \text{, when no new nodes were discovered} \end{cases}$$

Inertia vs Explore-n'-Go

Similarities

- Require zero knowledge for network area
- Require zero knowledge for sensor distribution.
- Have light-weight requirements in terms of memory and computational power

One main difference:

- Inertia walk performs network coverage by drawing big straight lines.
- Explore-n'-Go performs a more systematic network coverage, by sequencially visiting network subregions .

Visualization Inertia Walk

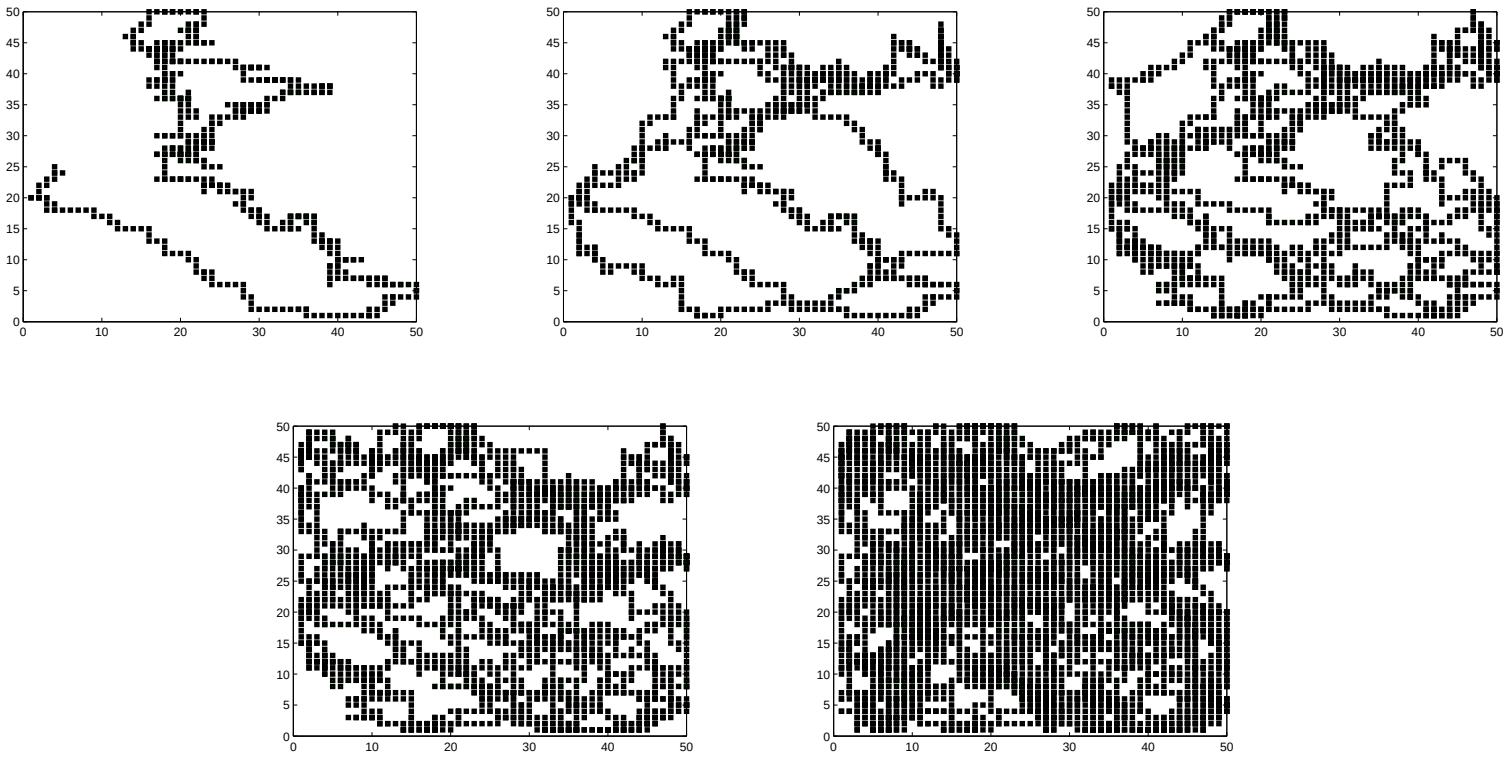


Figure: An example of network traversal following Random Walk with Inertia in a 50×50 network area (snapshots after: 500, 1000, 10^4, $2 * 10^4$ and $3 * 10^4$ hops).

Visualization Explore-n'-Go

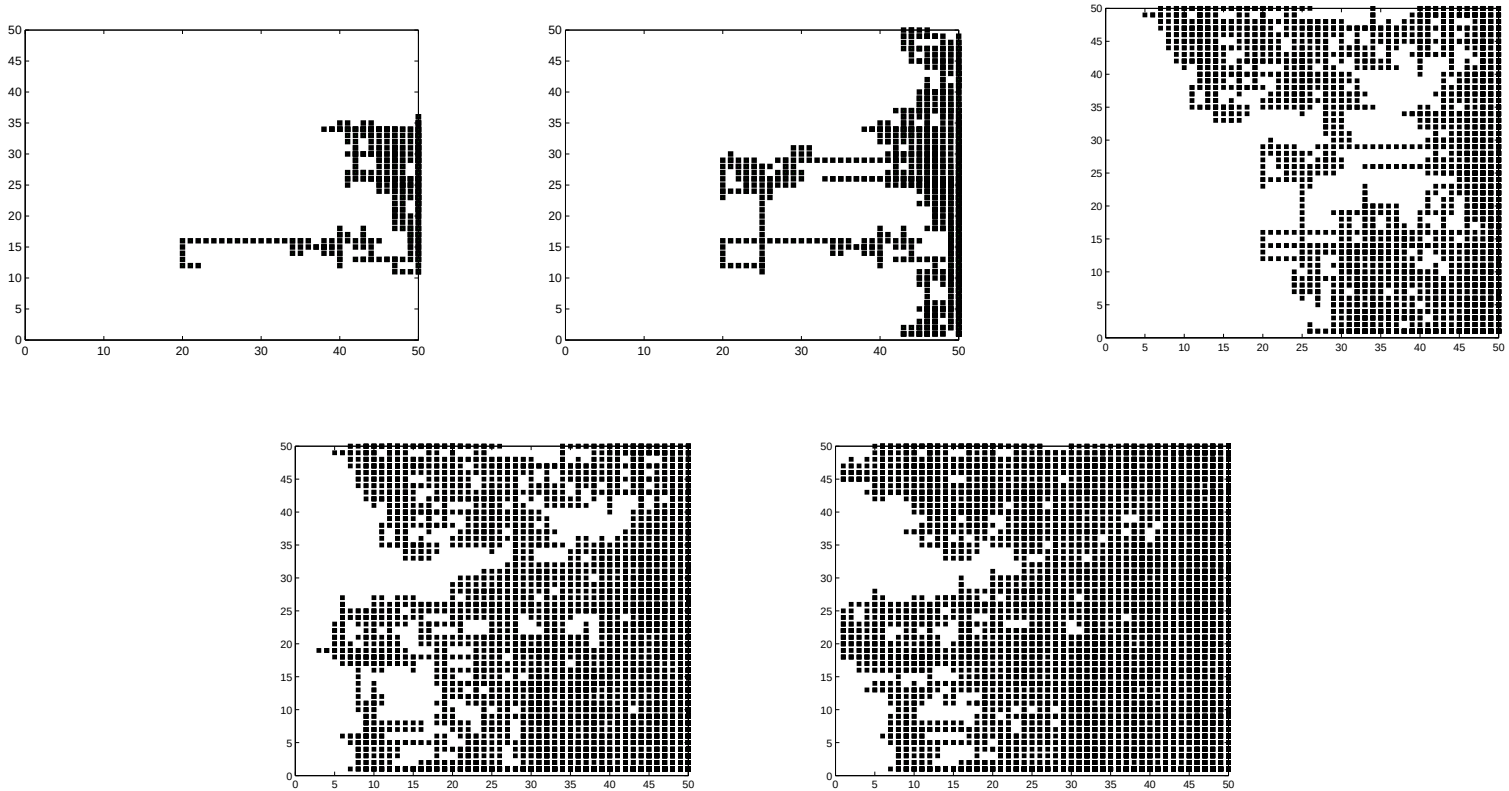


Figure: An example of network traversal following Explore-and-Go Random Walk in a 50×50 network area. (snapshots after 500, 1000, 10^4, $2 * 10^4$ and $3 * 10^4$ hops.)

Curly Random Walk 1/3

Intuition:

- start by visiting a confined subregion
- gradually allow the sink to perform a motion of higher degree of freedom
- eventually, the sink will cover the entire network area.

To achieve this, initially the sink performs frequent narrow left-turns, which gradually get wider.

Curly Random Walk 2/3

This type of motion can be modelled as a series of successive straight S and left- turn L moves.

$$SLSSLSSSLSSSSL...$$

Obviously, he probability distribution of left turns is the geometric distribution.

- probability mass function $P^i = (1 - p_L)^i p_L$
- p_L:probability of left turn
- i the number of successive straight moves before the next left turn.
- For a fixed i, $p_L^i = \frac{1}{i+1}$.

Curly Walk 3/3

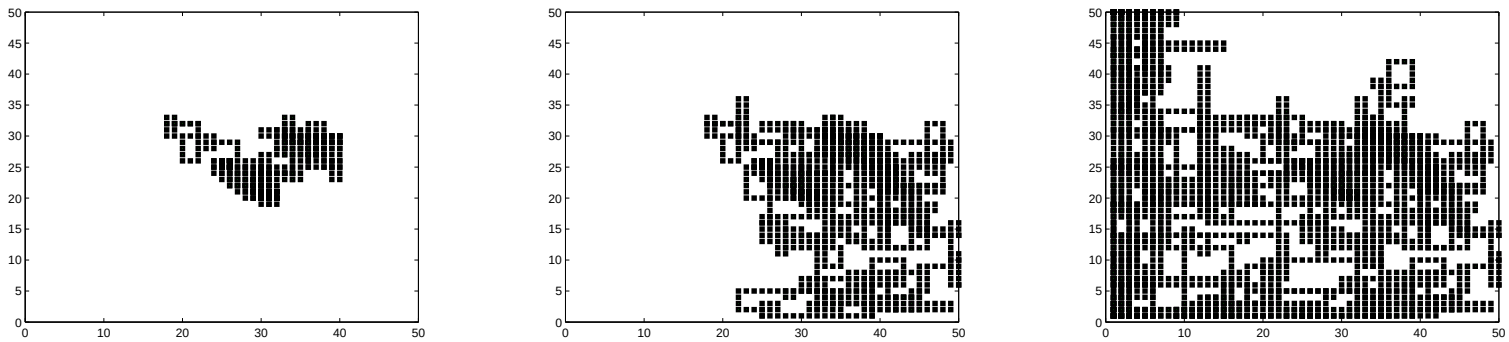
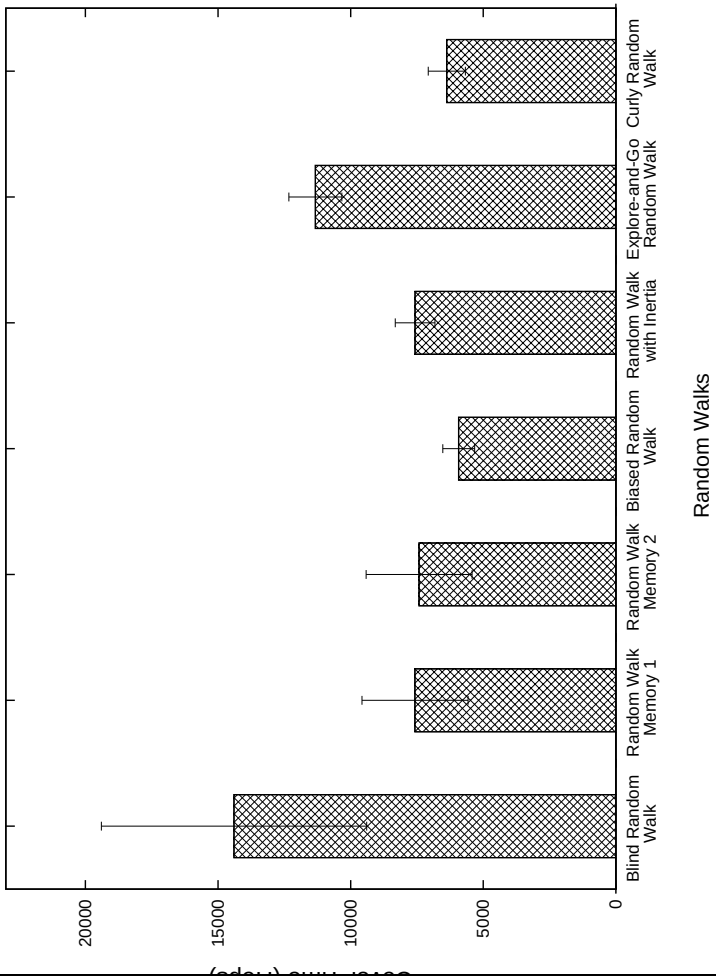
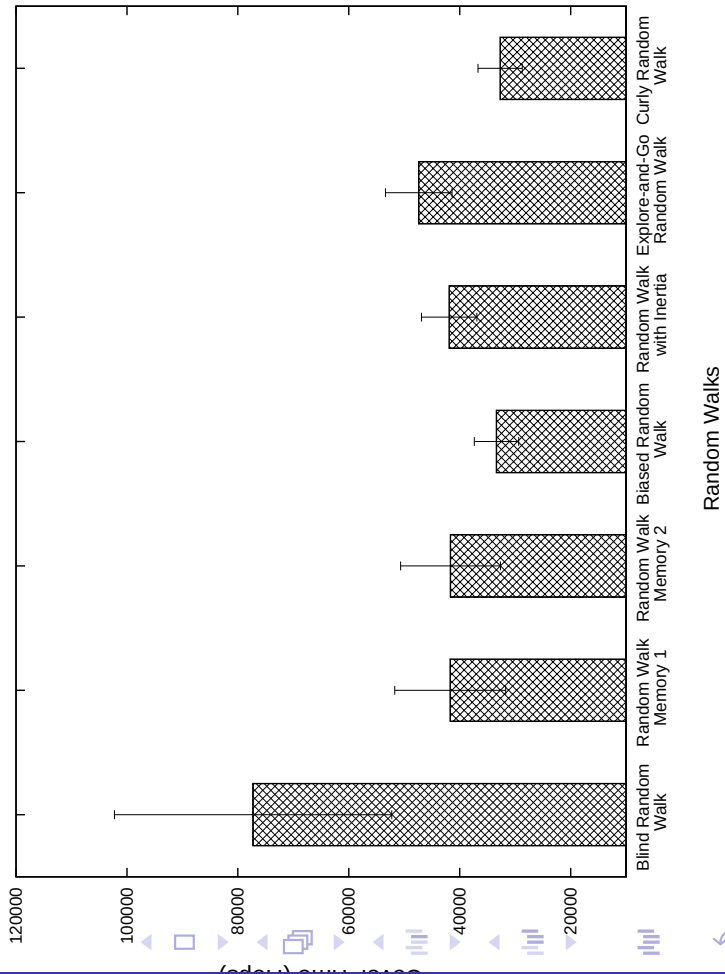


Figure: An example of network traversal following Curly Random Walk in a 50×50 network area. (snapshots after 500, $2 * 10^3$ and $2 * 10^4$ hops.)

Performance Evaluation





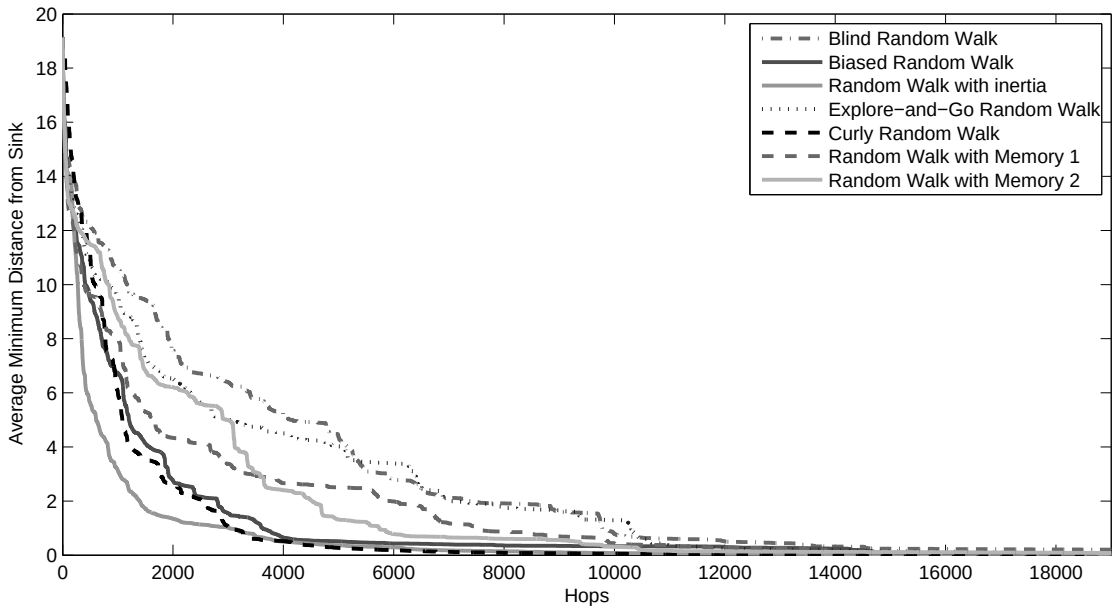


Figure: The Proximity Variation for a 50×50 network.

the mean value (over all cells) of the smallest distance from the sink for all the cells.

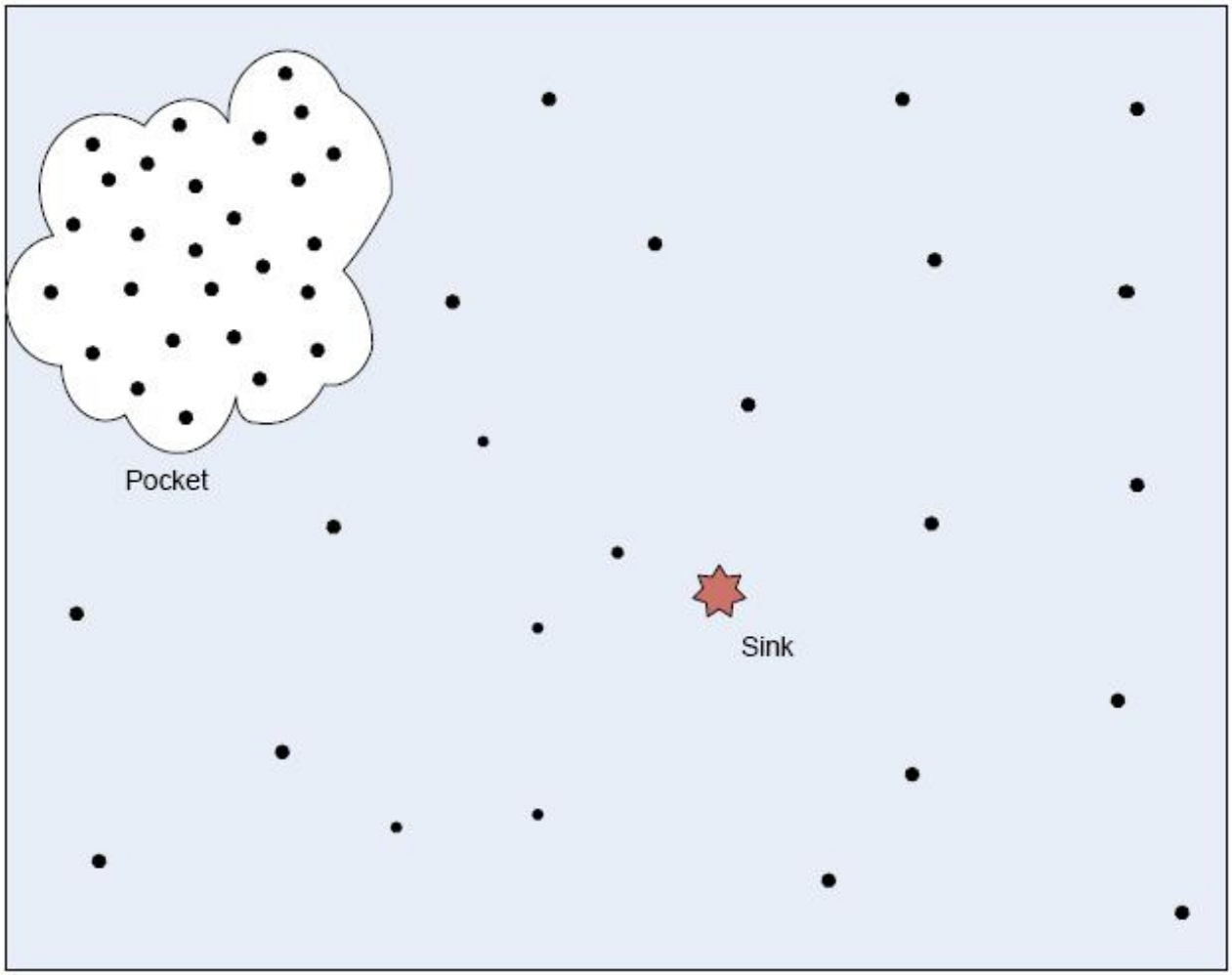
$$PV = \frac{\sum_{i=1}^{n} min(dist(i))}{n}$$

“**Aggregated Sensory Data Collection by Mobility-based Topology Ranks"**, Angelopoulos Constantinos Marios, Nikoletseas Sotiris, GLOBECOM 2009 - PE-WASUN 2009 (new version with aggregation)

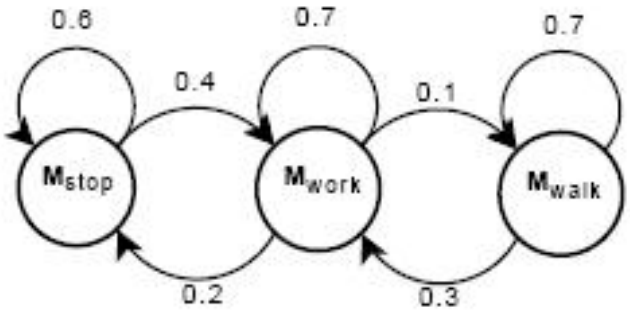
The Problem

In a WSN with *full mobility scheme*, where both sensors and sink move dynamically, how can the sink *efficiently* collect data from sensors?

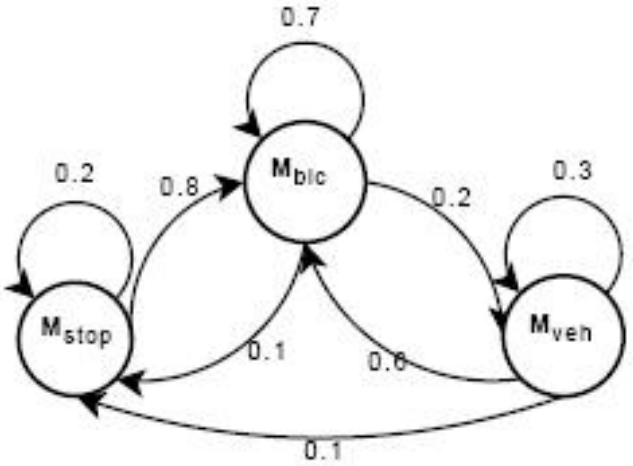
The Problem



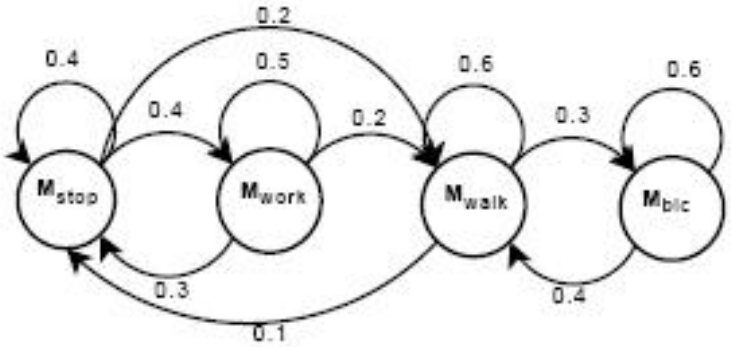
Modelling Dynamic Sensory Mobility



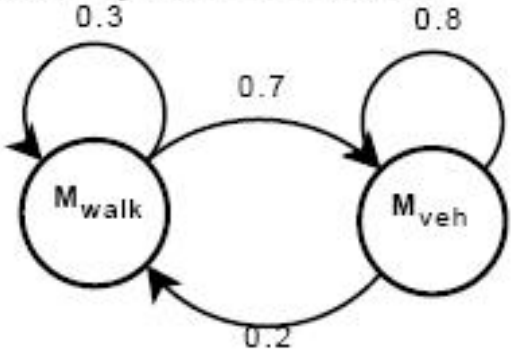
C1: Transitions between slow mobility roles



C3: Transitions for medium mobility with fast bursts



C2: Transitions for medium mobility level



C4: Transitions for fast mobility

Our Approach

- Sensors are moving inside the network area, independently of each other.
- Each sensor periodically measures the number of its neighbors and stores a triplet.
- Each triplet consists of: the number of neighbors, a timestamp, the current position

Based on triplets, the sink is going to exploit general topological information.

Our Approach II

Each triplet is asigned a value via the ranking function

$$R = \frac{d^2_{local}}{\Delta P \Delta T}$$

where:

- d is the number of neighbors.
- ΔP the distance between position where d was measured and current position.
- ΔT the time interval when d was measured and current time.

Our Approach II

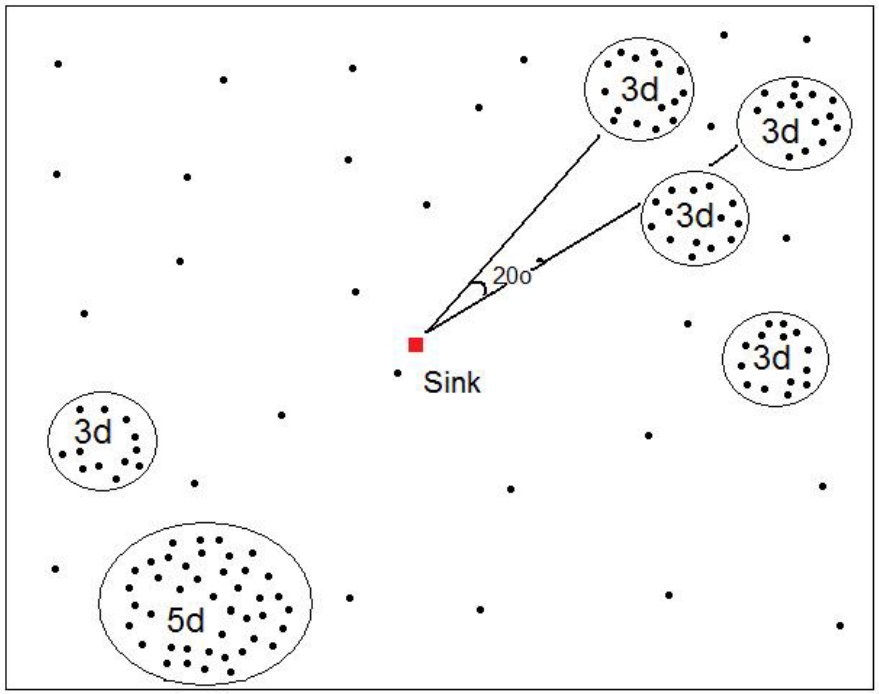
Each triplet is asigned a value via the ranking function

$$R = \frac{d_{local}^2}{\Delta P \Delta T}$$

- Only one triplet is stored each time.
- A new triplet replaces stored one in sensor memory, **iff** it is asigned a higher value.
- When a sensor reaches the radio range of the sink, along with the sensory data, the stored triplet is also sent.
- The sink chooses to move towards the direction corresponding to the highest ranking collected triplet.

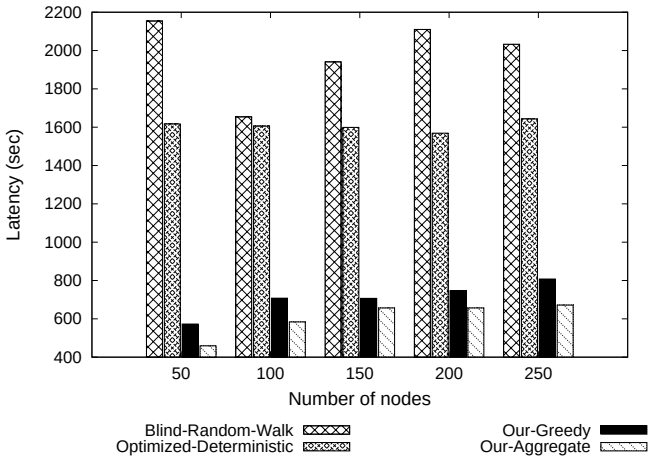
The Aggregation Process

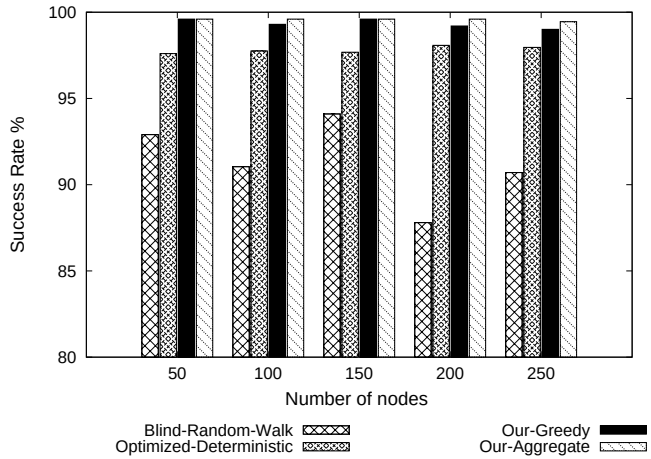
The sink initially traverses the network area at random direction. For a short period of time it collects triplets. Triplets A,B corresponding to positions relatively close are aggregated into C, based on an $angle_{thresh}$

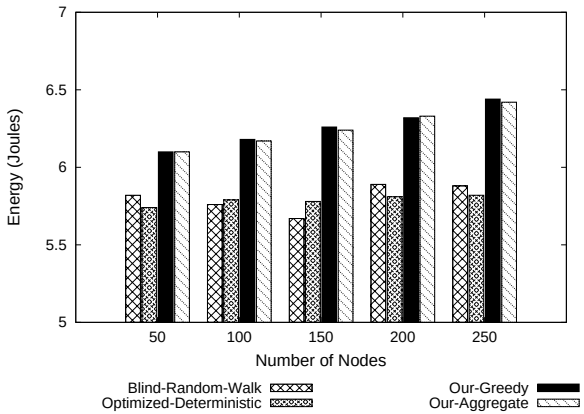


- $C_{d_{local}} = A_{d_{local}} + B_{d_{local}}$
- $C_P = \frac{A_P * A_{d_{local}} + B_P * B_{d_{local}}}{A_{d_{local}} + B_{d_{local}}}$
- $C_T = \frac{A_T * A_{d_{local}} + B_T * B_{d_{local}}}{A_{d_{local}} + B_{d_{local}}}$

Performance Findings







- Latency improves up to 8 times
- Better success rate, from 93% to 98%
- Slightly (10%) more energy dissemination
- Improvements **also in homogeneous** placement

Papers

- P. Juang, H. Oki and Y. Wang, **Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet**.
- C. Liu and J. Wu. **Scalable routing in delay tolerant networks**. In *Mobihoc*, 2007.
- W. Wang, V. Srinivasan and K-C. Chua. **Trade-offs between mobility and density for coverage in wireless sensor networks**. In *Mobicom*, 2007.
- A. Kinalis and S. Nikoletseas, **Adaptive Redundancy for Data Propagation Exploiting Dynamic Sensory Mobility**. MSWiM 2008.
- A. Boukerche, D. Efstathiou and S. Nikoletseas, **Adaptive, Direction-Aware Data Dissemination for Diverse Sensor Mobility**. MOBIWAC 2009.
- A. Clementi, F. Pasquale, and R. Silvestri, **MANETS: High mobility can make up for low transmission power**. In *36th ICALP*, 2009.