Dynamic Selection of Network Protocols for Group Communications in Mobile Ad-hoc Networks

Aaron M. Rosenfeld
Advisor: Prof. William C. Regli

College of Computing & Informatics, Drexel University

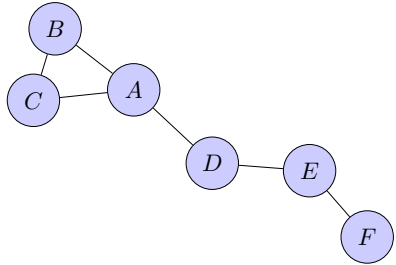
April 17th, 2014

Problem

It is difficult to select protocols for group-communication applications deployed in Mobile Ad-hoc Networks (MANETs).

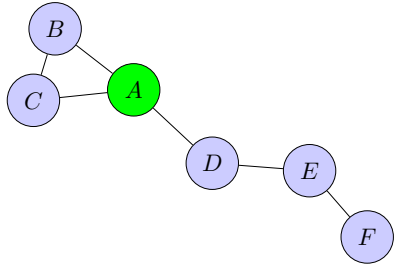
- No centralized control
- Unknown global state
- Frequent topology and traffic changes
- Disparate conditions across network

Motivating Example



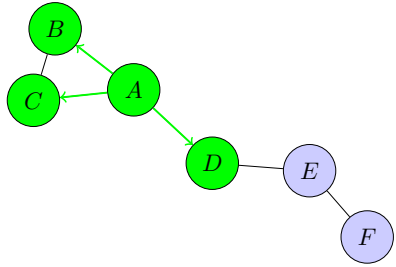
Only transmissions delivering a message shown for clarity.

Motivating Example



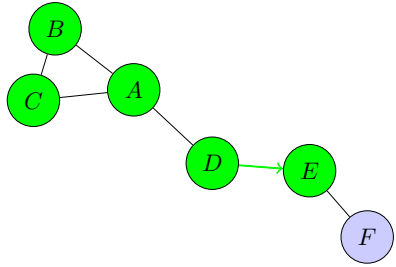
Only transmissions delivering a message shown for clarity.

Motivating Example



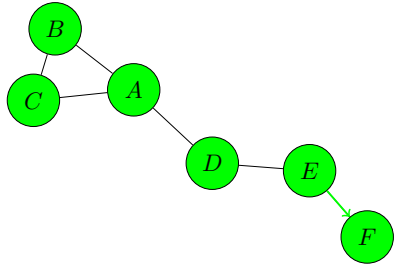
Only transmissions delivering a message shown for clarity.

Motivating Example



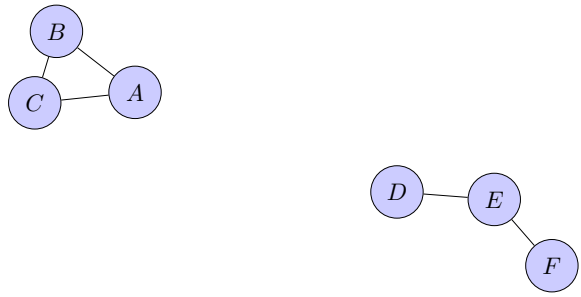
Only transmissions delivering a message shown for clarity.

Motivating Example



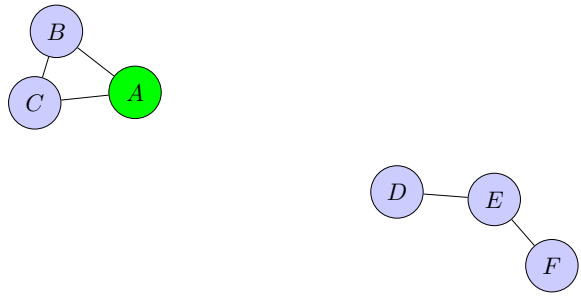
Only transmissions delivering a message shown for clarity.

Motivating Example



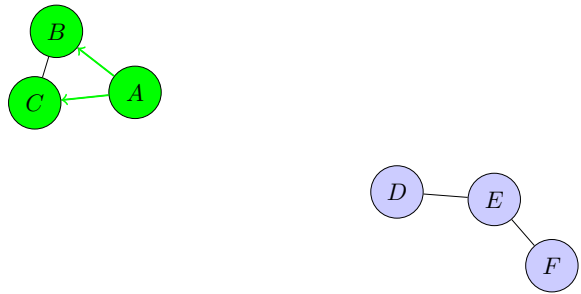
Only transmissions delivering a message shown for clarity.

Motivating Example



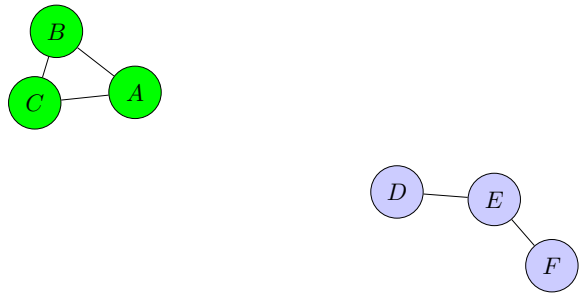
Only transmissions delivering a message shown for clarity.

Motivating Example



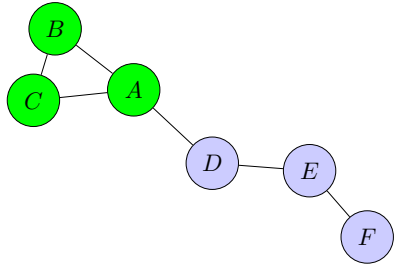
Only transmissions delivering a message shown for clarity.

Motivating Example



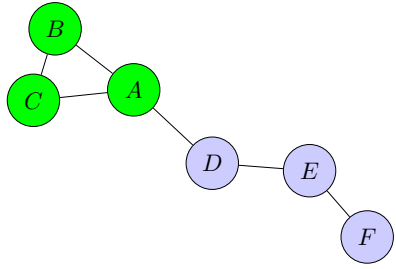
Only transmissions delivering a message shown for clarity.

Motivating Example



Only transmissions delivering a message shown for clarity.

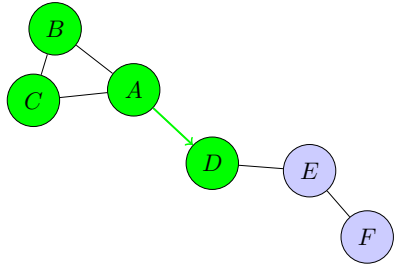
Motivating Example



1. Frequently retransmit all prior messages (fast, high bandwidth)

Only transmissions delivering a message shown for clarity.

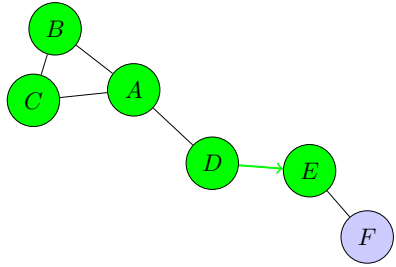
Motivating Example



1. Frequently retransmit all prior messages (fast, high bandwidth)

Only transmissions delivering a message shown for clarity.

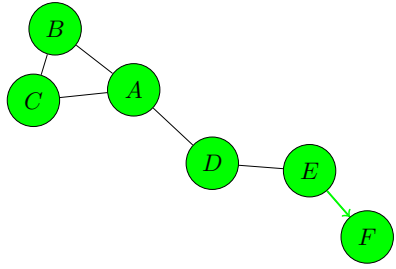
Motivating Example



1. Frequently retransmit all prior messages (fast, high bandwidth)

Only transmissions delivering a message shown for clarity.

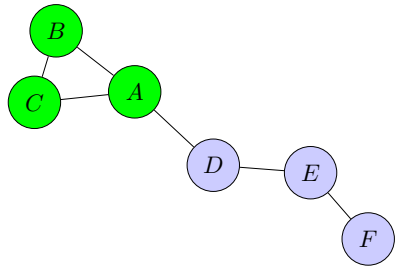
Motivating Example



1. Frequently retransmit all prior messages (fast, high bandwidth)

Only transmissions delivering a message shown for clarity.

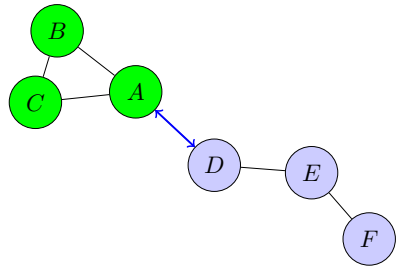
Motivating Example



1. Frequently retransmit all prior messages (fast, high bandwidth)
2. Reconcile missed messages with peers (slow, low bandwidth)

Only transmissions delivering a message shown for clarity.

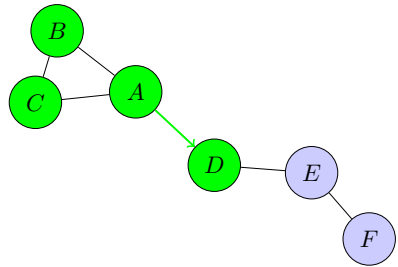
Motivating Example



1. Frequently retransmit all prior messages (fast, high bandwidth)
2. Reconcile missed messages with peers (slow, low bandwidth)

Only transmissions delivering a message shown for clarity.

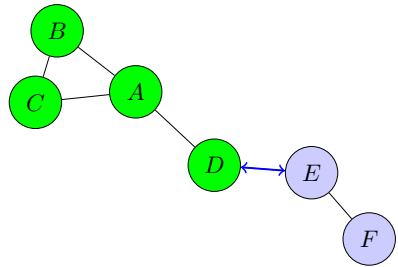
Motivating Example



1. Frequently retransmit all prior messages (fast, high bandwidth)
2. Reconcile missed messages with peers (slow, low bandwidth)

Only transmissions delivering a message shown for clarity.

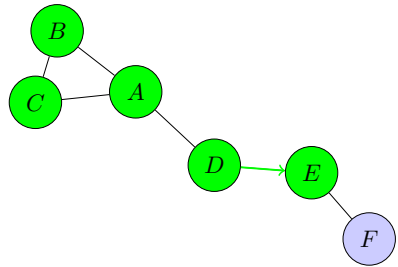
Motivating Example



1. Frequently retransmit all prior messages (fast, high bandwidth)
2. Reconcile missed messages with peers (slow, low bandwidth)

Only transmissions delivering a message shown for clarity.

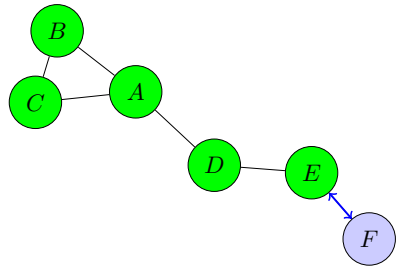
Motivating Example



1. Frequently retransmit all prior messages (fast, high bandwidth)
2. Reconcile missed messages with peers (slow, low bandwidth)

Only transmissions delivering a message shown for clarity.

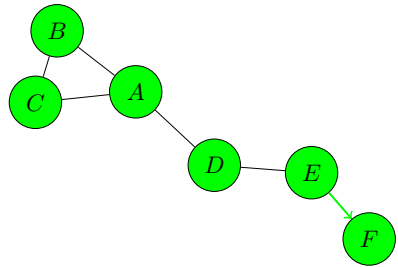
Motivating Example



1. Frequently retransmit all prior messages (fast, high bandwidth)
2. Reconcile missed messages with peers (slow, low bandwidth)

Only transmissions delivering a message shown for clarity.

Motivating Example



1. Frequently retransmit all prior messages (fast, high bandwidth)
2. Reconcile missed messages with peers (slow, low bandwidth)

Only transmissions delivering a message shown for clarity.

Motivating Example

So, which is better?

So, which is better? Neither is always best.

Motivating Example

So, which is better? Neither is always best.

- Depends on traffic load

Motivating Example

So, which is better? Neither is always best.

- Depends on traffic load
- Different parts of the network have varying properties

Motivating Example

So, which is better? Neither is always best.

- Depends on traffic load
- Different parts of the network have varying properties
- Future mobility is unknown

Motivating Example

So, which is better? Neither is always best.

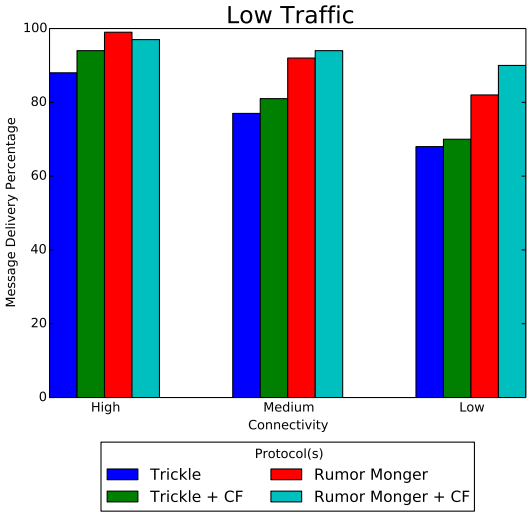
- Depends on traffic load
- Different parts of the network have varying properties
- Future mobility is unknown
- Is speed or overhead more of a concern?

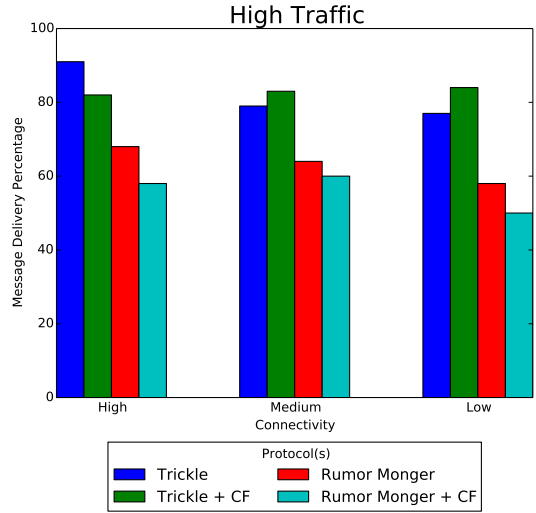
Empirical Motivation

Published paper in MILCOM 2012 demonstrating the difficulty in statically selecting protocols.

- Tested combinations of persistence protocols (Session layer) and transport protocols.
  - **Persistence**: Rumor Mongering (fast, high-bandwidth), Trickle (slower, low-bandwidth)
  - **Transport**: UDP link-local broadcast, UDP classical flooding multicast
- Showed that without full knowledge of both connectivity and traffic load, one cannot select a single protocol that delivers the most messages

Empirical Motivation

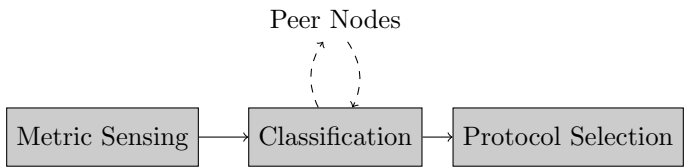




Proposed Solution

Dynamic Protocol Selection

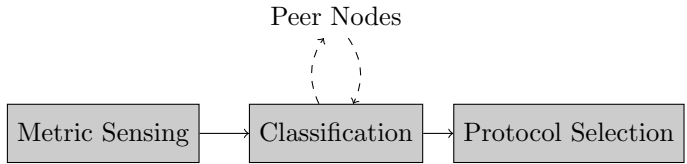
Using locally sensed network state and information from peers, dynamically select and utilize the best protocol(s) for that current moment.



Proposed Solution

Dynamic Protocol Selection

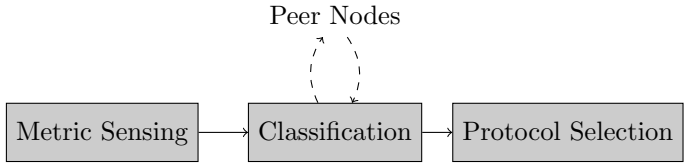
Using locally sensed network state and information from peers, dynamically select and utilize the best protocol(s) for that current moment.



Proposed Solution

Dynamic Protocol Selection

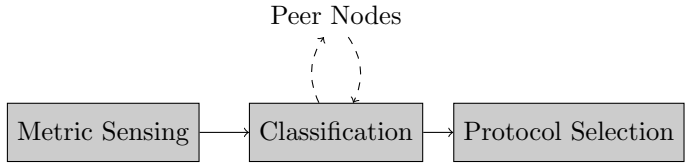
Using locally sensed network state and information from peers, dynamically select and utilize the best protocol(s) for that current moment.



Proposed Solution

Dynamic Protocol Selection

Using locally sensed network state and information from peers, dynamically select and utilize the best protocol(s) for that current moment.



Related Work — AMQP

AMQP is an standardized protocol for message-oriented applications.

- Allows applications to stipulate delivery requirements (e.g. order, guarantee)
- Alleviates the need for developers to specify a specific protocol
- Most implementations utilize TCP and UDP
- Do not generally utilize or exchange network state

Related Work — MASs

Protocols at the application-layer for Multi-agent Systems are necessary for coherent interaction patterns.

- Quenum, et al. introduced a novel way of negotiating protocols during runtime to optimize certain metrics (e.g. time to complete tasks)
- Requires some degree of agreement between agents
- Does not utilize any network state measurements, only agent metrics
- Protocol agreement is computationally very expensive

Related Work — Markov Random Fields

Doyle, et al.'s work in Markov Random Fields inspired portions of this thesis, including methods to merge local and remote information.

- Uses Markov Random Fields to assign the network labels based on sensed network state
- State is exchanged with peers and merged to find the label most probable
- Using this information, the routing protocol is changed
- Not actually implemented, only simulated at a high level

Related Work — Appia

Appia is a policy driven toolkit for dynamically changing and parameterizing protocols.

- Rosa, et al. added the ability for policies to react to network events:

```
WHEN BandwidthEvent: BandwidthEvent.value < BW_THRESHOLD DO
    setValue(timeoutvalue, UPDATED_TIMEOUT)
FOR TransportProtocol
```

- Does not implement data fusion with peer nodes
- Does not implement a formal method of using remote data to select protocols, only provides a framework

Related Work — Summary

- Most existing work is at only a single layer
- Do not fully address the challenges of exchanging local and remote state in real/emulated networks
- Provide a framework for dynamically selecting protocols, not a usable middleware implementation.

Problem Formalization

Define the following:

- **Network Factors**: $\mathcal{F} = \{F_1, \ldots, F_{|\mathcal{F}|}\}$ where each F_i is a set of appropriate labels.

Problem Formalization

Define the following:

- **Network Factors**: $\mathcal{F} = \{F_1, \ldots, F_{|\mathcal{F}|}\}$ where each F_i is a set of appropriate labels.
- **Measured Network Conditions**: $\mathcal{M} = \{(d_1, v_1), \ldots, (d_{|\mathcal{M}|}, v_{|\mathcal{M}|})\}$ where all $v_i \in \mathbb{R}$.

Problem Formalization

Define the following:

- **Network Factors**: $\mathcal{F} = \{F_1, \ldots, F_{|\mathcal{F}|}\}$ where each F_i is a set of appropriate labels.
- **Measured Network Conditions**: $\mathcal{M} = \{(d_1, v_1), \ldots, (d_{|\mathcal{M}|}, v_{|\mathcal{M}|})\}$ where all $v_i \in \mathbb{R}$.
- **Dynamic Stack Size**: $c \in \mathbb{Z}^+$. Generally $c \leq 6$.

Problem Formalization

Define the following:

- **Network Factors**: $\mathcal{F} = \{F_1, \ldots, F_{|\mathcal{F}|}\}$ where each F_i is a set of appropriate labels.
- **Measured Network Conditions**: $\mathcal{M} = \{(d_1, v_1), \ldots, (d_{|\mathcal{M}|}, v_{|\mathcal{M}|})\}$ where all $v_i \in \mathbb{R}$.
- **Dynamic Stack Size**: $c \in \mathbb{Z}^+$. Generally $c \leq 6$.
- **Protocol Sets**: $\mathcal{P}_1, \ldots, \mathcal{P}_c$.

Example

- **Network Factors**: $\mathcal{F} = \{F_{conn}, F_{traff}\}$
  - $F_{conn} = \{\text{low}, \text{medium}, \text{high}\}$
  - $F_{traff} = \{\text{low}, \text{medium}, \text{high}\}$
- **Measured Network Conditions**: $\mathcal{M} = \{(\text{neighbors}, 3), (\text{bytes/sec}, 120)\}$
- **Dynamic Stack Size**: $c = 2$
- **Protocol Sets**:
  - **Session**: $\mathcal{P}_1 = \{\text{Trickle}, \text{Rumor}\}$
  - **Transport**: $\mathcal{P}_2 = \{\text{UDP}, \text{NORM}\}$

Problem Formalization

Given these values, there are the following tasks:

Problem Formalization

Given these values, there are the following tasks:

- A function for each factor that maps the *local* network state (given by $\mathcal{M}$) to an associated label

Problem Formalization

Given these values, there are the following tasks:

- A function for each factor that maps the *local* network state (given by $\mathcal{M}$) to an associated label
- A method of merging these local labels with remote ones, to create a final label to be applied locally

Problem Formalization

Given these values, there are the following tasks:

- A function for each factor that maps the *local* network state (given by $\mathcal{M}$) to an associated label
- A method of merging these local labels with remote ones, to create a final label to be applied locally
- A function mapping all combinations of possible labels, one per factor, to a protocol for each layer

Labelling Approach

For this thesis, chose:

- Network factors of **connectivity** and **traffic load** as $\mathcal{F}$.
- Labels for each of "high," "medium," and "low."
- Local Conditions of **number of neighbors** and **bytes/sec**.
  - Gathered from the network interface via the OS

Labelling Process

For each factor, we determine a label (e.g. “low connectivity”) by:

1. Assigning a local label to the network with a simple threshold function

Labelling Process

For each factor, we determine a label (e.g. “low connectivity”) by:

1. Assigning a local label to the network with a simple threshold function
2. Exchange local label belief with peers

Labelling Process

For each factor, we determine a label (e.g. “low connectivity”) by:

1. Assigning a local label to the network with a simple threshold function
2. Exchange local label belief with peers
3. Integrate local and remote beliefs to assign a final label for each network factor

Labelling Process — Local Labelling

Label each factor F as high, medium, or low by breaking ranges into thirds. Select a $(d, v) \in \mathcal{M}$ for F and assign a label by:

$$T(d,v) = \begin{cases} low, & 0 < \frac{v}{mv(d)} \leq \frac{1}{3} \\ medium, & \frac{1}{3} < \frac{v}{mv(d)} \leq \frac{2}{3} \\ high, & \frac{2}{3} < \frac{v}{mv(d)} \end{cases}$$

Where $mv(d)$ is the maximum value for the network condition d, currently measured as v.

Labelling Process — Belief Exchange

Each node maintains a timeout value t and for **each** network factor $F \in \mathcal{F}$:

Labelling Process — Belief Exchange

Each node maintains a timeout value t and for **each** network factor $F \in \mathcal{F}$:

- The probability of each label being correct, $P_F(l)$ for all $l \in F$.

Labelling Process — Belief Exchange

Each node maintains a timeout value t and for **each** network factor $F \in \mathcal{F}$:

- The probability of each label being correct, $P_F(l)$ for all $l \in F$.
- $C_F(l)$, the number of nodes in the past period t that believe l is the correct label for $\mathcal{F}$.

Labelling Process — Belief Exchange

Each node maintains a timeout value t and for **each** network factor $F \in \mathcal{F}$:

- The probability of each label being correct, $P_F(l)$ for all $l \in F$.
- $C_F(l)$, the number of nodes in the past period t that believe l is the correct label for $\mathcal{F}$.
- $\mathcal{B}_F$, the number of unique neighbors transmitting a belief for F.

Labelling Process — Belief Exchange

Each node maintains a timeout value t and for **each** network factor $F \in \mathcal{F}$:

- The probability of each label being correct, $P_F(l)$ for all $l \in F$.
- $C_F(l)$, the number of nodes in the past period t that believe l is the correct label for $\mathcal{F}$.
- $\mathcal{B}_F$, the number of unique neighbors transmitting a belief for F.
- M_F, the current value v of one network condition, $(d, v) \in \mathcal{M}$.

Labelling Process — Belief Exchange

Each node maintains a timeout value t and for **each** network factor $F \in \mathcal{F}$:

- The probability of each label being correct, $P_F(l)$ for all $l \in F$.
- $C_F(l)$, the number of nodes in the past period t that believe l is the correct label for $\mathcal{F}$.
- $\mathcal{B}_F$, the number of unique neighbors transmitting a belief for F.
- M_F, the current value v of one network condition, $(d, v) \in \mathcal{M}$.
- A normal distribution function $\mathcal{N}_F^l : \mathbb{R} \to (0, 1)$ based on the mean and variance of previous M_F values when $T(d, v) = l$.

Labelling Process — Belief Exchange

Each node maintains a timeout value t and for **each** network factor $F \in \mathcal{F}$:

- The probability of each label being correct, $P_F(l)$ for all $l \in F$.
- $C_F(l)$, the number of nodes in the past period t that believe l is the correct label for $\mathcal{F}$.
- $\mathcal{B}_F$, the number of unique neighbors transmitting a belief for F.
- M_F, the current value v of one network condition, $(d, v) \in \mathcal{M}$.
- A normal distribution function $\mathcal{N}_F^l : \mathbb{R} \to (0, 1)$ based on the mean and variance of previous M_F values when $T(d, v) = l$.

Occasionally[1] broadcast a set:

$$b = \langle (F_1, l_1), \ldots, (F_{|\mathcal{F}|}, l_{|\mathcal{F}|}) \rangle$$

where each l_i is most probable for F_i.

[1] 2 seconds for this implementation

Labelling Process — Belief Fusion

For each label, l_i received in b for F_i, update the associated $P_{F_i}(l_i)$ to:

$$P_{F_i}(l_i) = \tfrac{1}{2}\left(\underbrace{(1-\lambda)\mathcal{N}_{F_i}^{l_i}(M_{F_i})}_{\text{Local measurement}} + \underbrace{\lambda\frac{C_{F_i}(l_i)}{\mathcal{B}_{F_i}}}_{\text{New broadcasts}}\right)$$

The label assigned for factor F_i is then given by

$$L_{F_i} = \arg\max_{l \in F_i} P_{F_i}(l)$$

Labelling Effectiveness — Experimental Setup

- Ran experiments to determine how well approach works

Labelling Effectiveness — Experimental Setup

- Ran experiments to determine how well approach works
- Used the Common Open Research Emulator (CORE)

Labelling Effectiveness — Experimental Setup

- Ran experiments to determine how well approach works
- Used the Common Open Research Emulator (CORE)
- Used the MGEN traffic generator to send periodic traffic at various rates

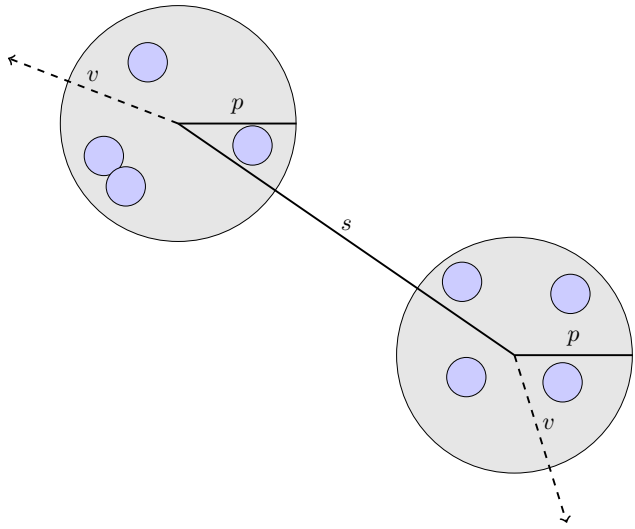
Labelling Effectiveness — Experimental Setup

- Ran experiments to determine how well approach works
- Used the Common Open Research Emulator (CORE)
- Used the MGEN traffic generator to send periodic traffic at various rates

CORE Configuration	
Parameter	**Value**
Scenario Dimensions	1000 × 1000 meters
Range	20 meters
Bandwidth	5 Mbps
Delay	20 ± 5 milliseconds
Jitter	0 milliseconds

Traffic Configuration		
Load	**Frequency**	**Size**
High	1 seconds	1024 ± 50 bytes
Medium	5 seconds	512 ± 20 bytes
Low	10 seconds	256 ± 10 bytes

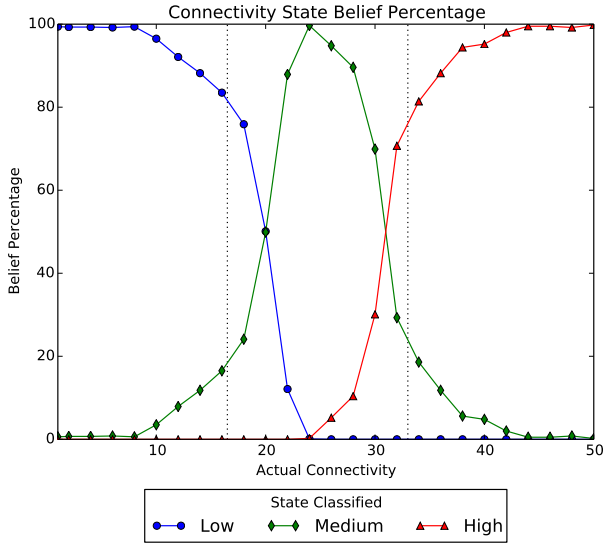
Labelling Effectiveness — Experimental Setup



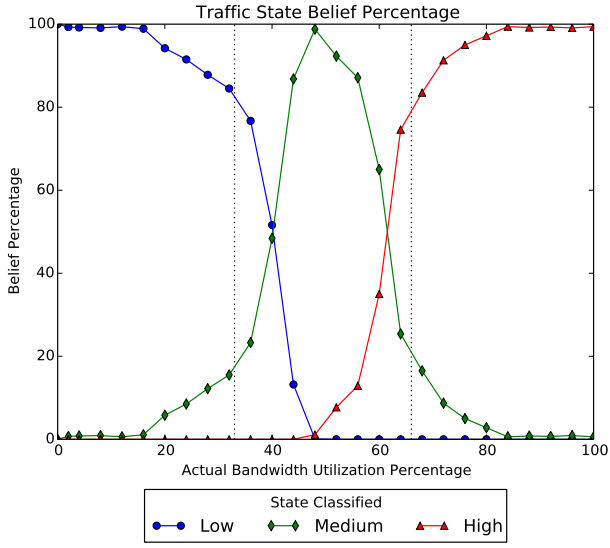
Labelling Effectiveness — Experimental Setup

Parameter	**Value**
Number of Nodes	50
Number of Groups	2, 4, 5, 10, 25
Reference Point Separation	150, 250, 800 meters
Node Separation	25 meters
Speed	2 ± 5 meters per second
Pause Time	2 ± 2 seconds

Labelling Effectiveness — Connectivity



Labelling Effectiveness — Traffic



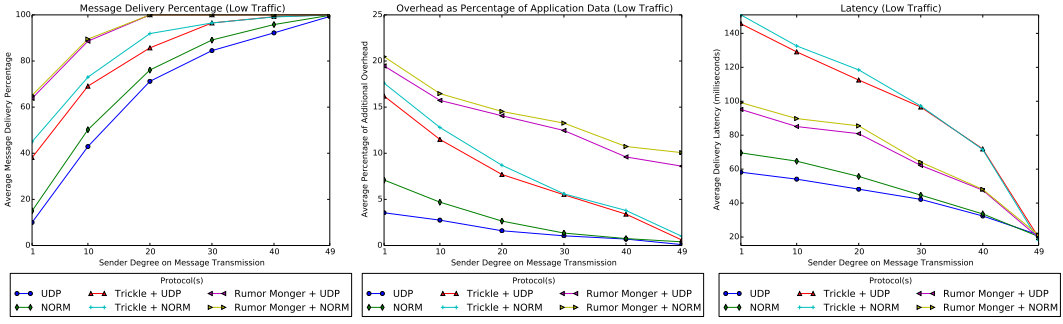
Protocol Performance

- The network can now be labelled
- Based on the labels, determined which protocol performs best
- Individually tested protocols at the Session and Transport layers with various connectivity levels and traffic loads

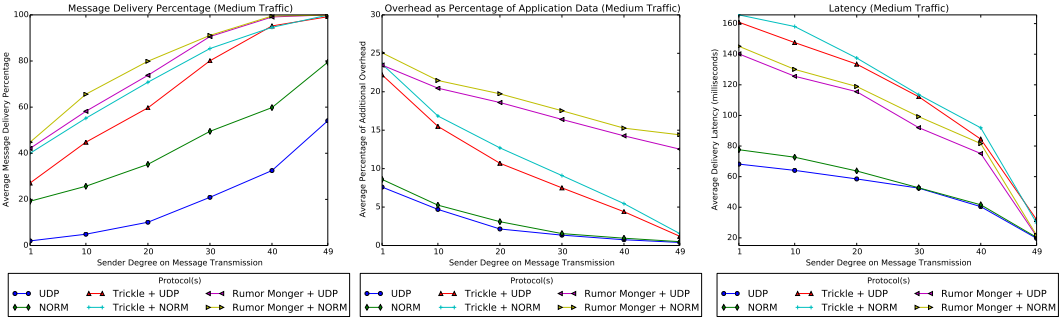
Protocol Performance — Experimental Setup

- Used RPGM model as in prior experiments
- Trickle and Rumor Mongering at the Session layer for data persistence
  - Included no Session protocol for comparison purposes
- UDP and NORM at the Transport layer
  - NORM is a reliable multicast protocol utilizing Negative ACKs
  - Both were set to broadcast only link-locally

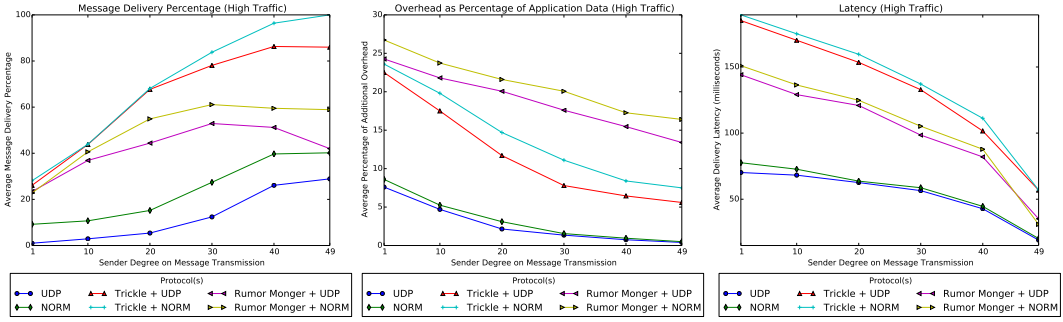
Protocol Performance — Low Traffic Results



Protocol Performance — Medium Traffic Results



Protocol Performance — High Traffic Results



Protocol Performance — Summary

- In low- and medium-traffic, Trickle and Rumor Mongering increase in MDR as connectivity increases at approximately the same rate.
- In high-traffic, Rumor Mongering performs poorly when highly connected
- Using no persistence protocol is not advantageous if MDR is important

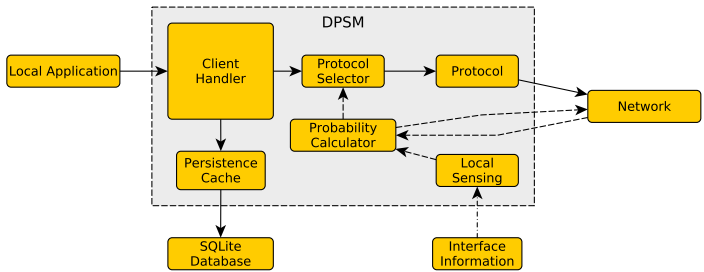
Protocol Performance — Best Protocols

Based on these results, the following appear to offer the best performance tradeoffs:

		Connectivity		
		Low	*Medium*	*High*
Traffic	*Low*	RM + UDP	TR + NORM	TR + UDP
	Medium	RM + NORM	TR + NORM	TR + UDP
	High	TR + UDP	TR + NORM	TR + NORM

Dynamic Protocol Selection Middleware (DPSM)

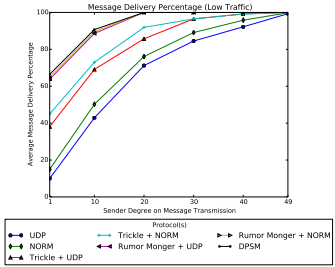
- Created the Dynamic Protocol Selection Middleware (DPSM) to dynamically switch protocols based on these results
- Implemented in Java and acts as an abstraction between applications and lower layer protocols
- Applications subscribe and publish to specific destinations
- For each message, assert the delivery requirements (reliable and/or persistent), destination, and payload

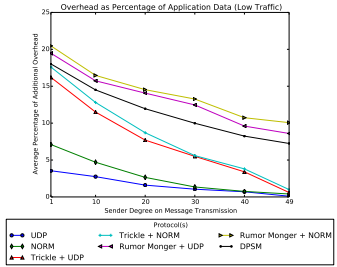


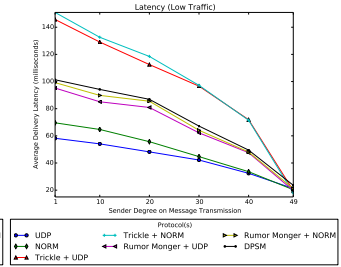
Baseline Results

- Compared the dynamic approach in DPSM to that of the static protocols
- CORE and MGEN were again used with the same parameters
- All messages were marked as both persistent and reliable

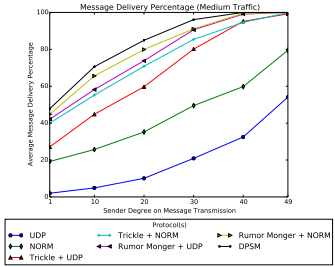
Baseline Results — Low Traffic

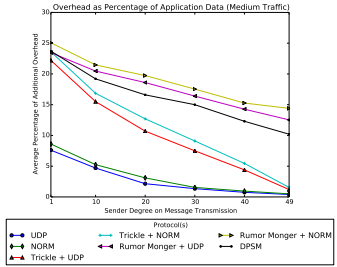


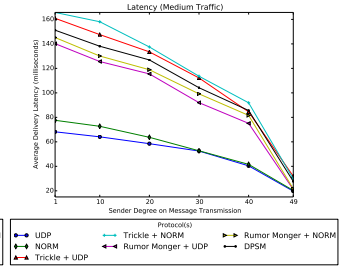




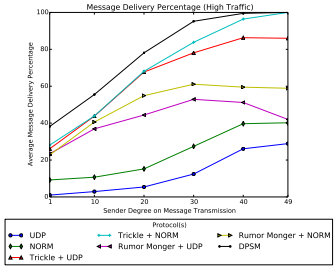
Baseline Results — Medium Traffic

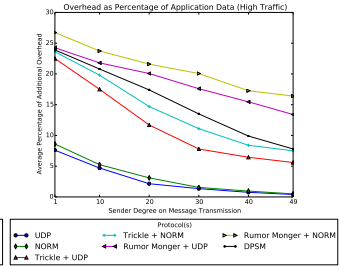


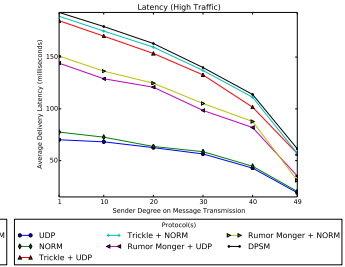




Baseline Results — High Traffic







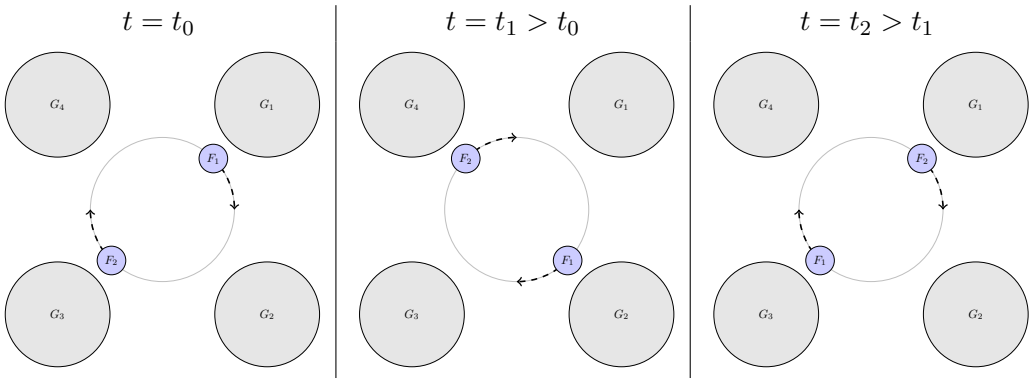
Real-World Scenarios

- Most real-world scenarios do not involve groups moving at random
- Some structure based on a task or objective to accomplish
- Compared DPSM to static approaches in two real-world scenarios
  - Message ferry
  - Group following

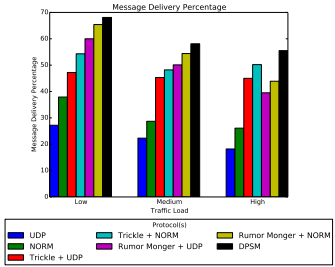
Message Ferrying

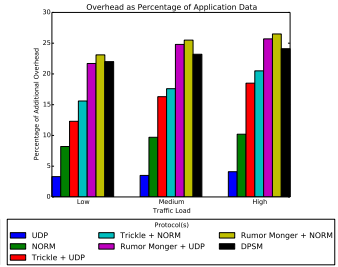
- 48 nodes split into 4 equal groups in a square
- Each group nearly remains connected
- No two groups ever directly come into contact
- Two nodes circle the four groups delivering messages between them

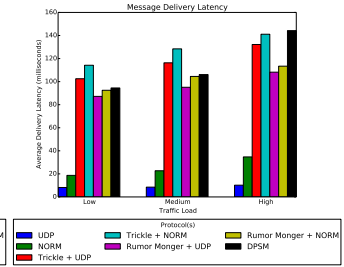
Message Ferrying



Message Ferrying — Results



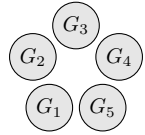




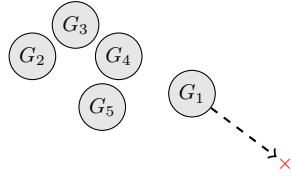
Group Following

- 50 nodes split into five equally sized groups
- First group randomly selects a waypoint and travels there
- Upon reaching it, the second group meets it temporarily
- The first group then moves on
- Repeats for following groups.

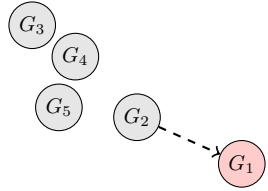
Group Following — Example



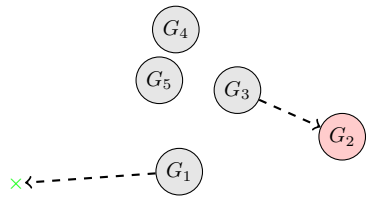
Group Following — Example



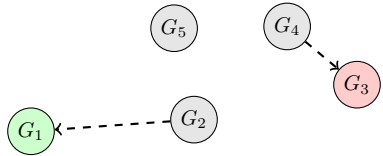
Group Following — Example



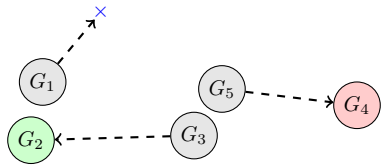
Group Following — Example



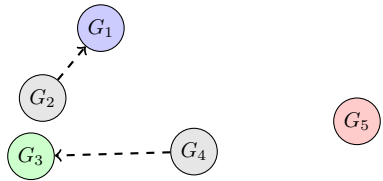
Group Following — Example



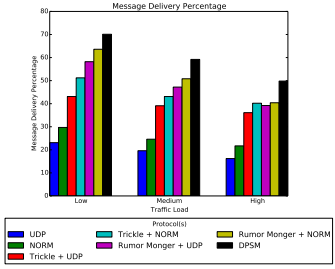
Group Following — Example

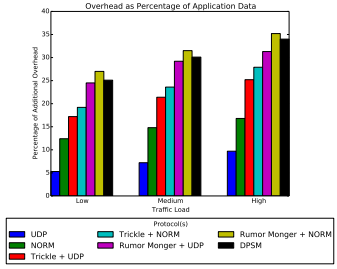


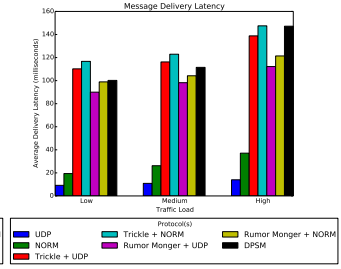
Group Following — Example



Group Following — Results







Problem

Motivation

Proposed Solution

Related Work

Problem Formalization

Approach

Implementation & Results

Conclusions

Summary

- Motivated the need for dynamic protocol selection with an example and empirical results

Summary

- Motivated the need for dynamic protocol selection with an example and empirical results
- Formalized the problem of protocol selection and provided an approach to solve it

Summary

- Motivated the need for dynamic protocol selection with an example and empirical results
- Formalized the problem of protocol selection and provided an approach to solve it
- Proposed a generic solution to problem using local and remote network state information

Summary

- Motivated the need for dynamic protocol selection with an example and empirical results
- Formalized the problem of protocol selection and provided an approach to solve it
- Proposed a generic solution to problem using local and remote network state information
- Implemented the approach as a generic middleware, DPSM

Summary

- Motivated the need for dynamic protocol selection with an example and empirical results
- Formalized the problem of protocol selection and provided an approach to solve it
- Proposed a generic solution to problem using local and remote network state information
- Implemented the approach as a generic middleware, DPSM
- Evaluated the performance of static protocol to determine which performed the best with each label combination

Summary

- Motivated the need for dynamic protocol selection with an example and empirical results
- Formalized the problem of protocol selection and provided an approach to solve it
- Proposed a generic solution to problem using local and remote network state information
- Implemented the approach as a generic middleware, DPSM
- Evaluated the performance of static protocol to determine which performed the best with each label combination
- Provided empirical evidence in both generic and real-world scenarios that DPSM can improve message delivery with slight latency or overhead tradeoffs

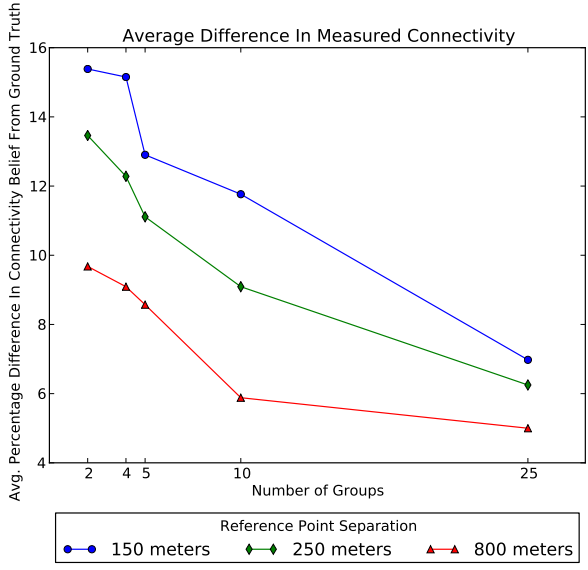
Future Work

- Labelling without knowing bounds on the network
- Alternative labelling approaches using different fusion methods
- Smoothing for fringe nodes
- Dynamic learning of optimal protocol selection rather than pre-programming

Questions?

Contact
Aaron M. Rosenfeld — ar374@drexel.edu
Advisor: Prof. William C. Regli — regli@drexel.edu

Local Labelling Effectiveness



Local Labelling Effectiveness