FGDR: **F**ine-**G**rained Contact Pattern Characterisation and **D**ynamic Message **R**eplication Control for Message Routing in Delay-tolerant Networks(DTN)

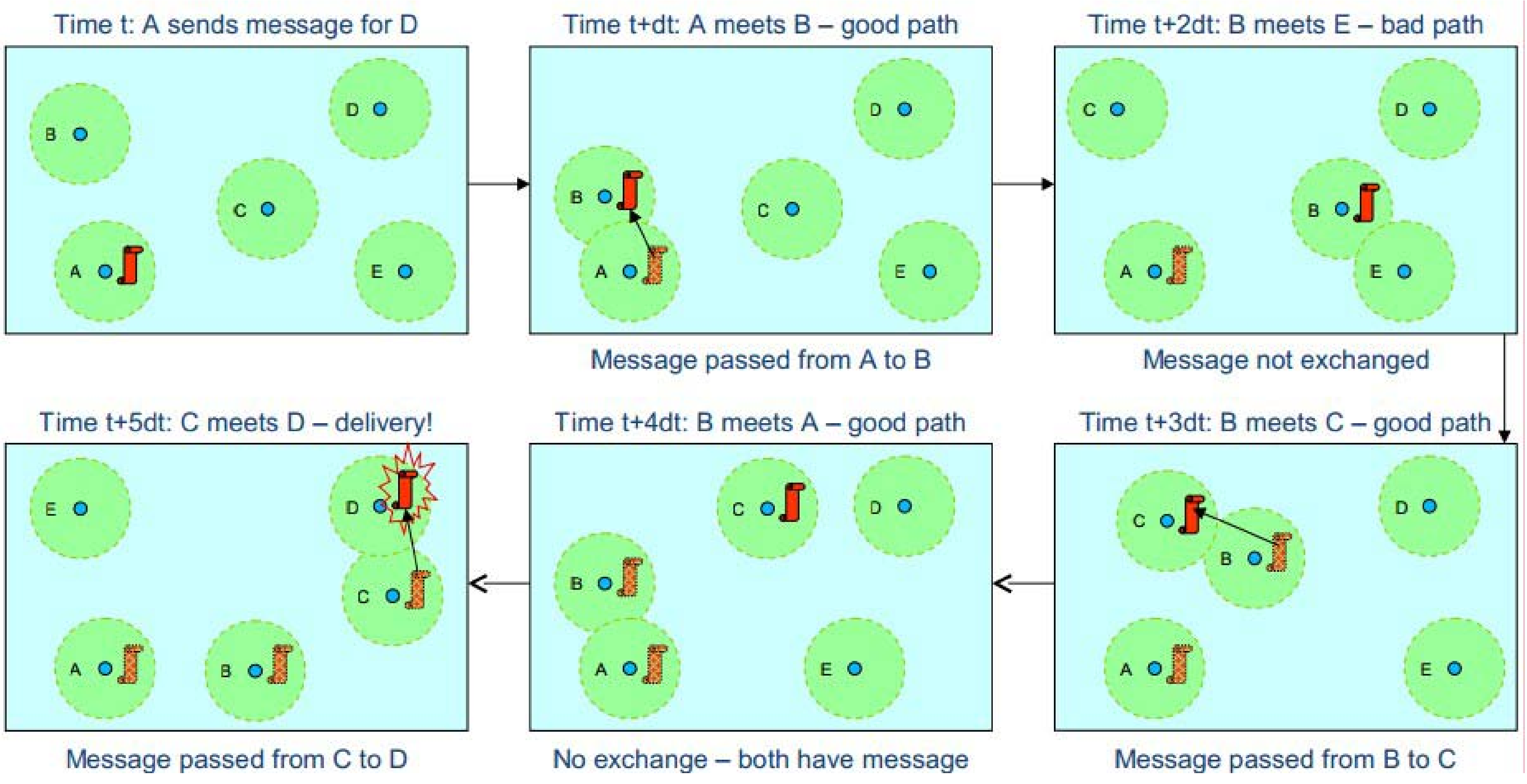
Luming Wan

Supervisor: Haibo Zhang

DTN

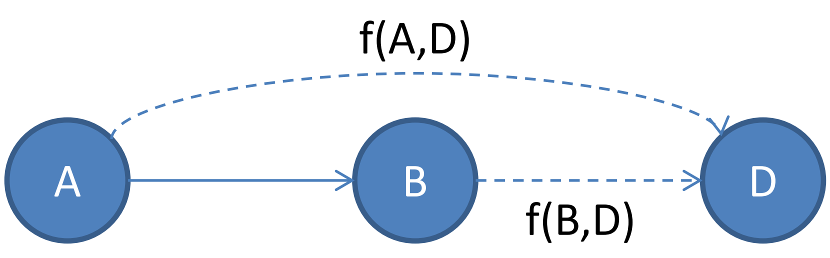
- Node:
  - high mobility, low power, fixed contact range
- Rare infrastructure
- No guaranteed end-to-end connectivity
- No fixed topology
- Hardware constraints: power, bandwidth, memory etc.
- Packets: Store-carry-forward
- Long latency

Network Model Assumption



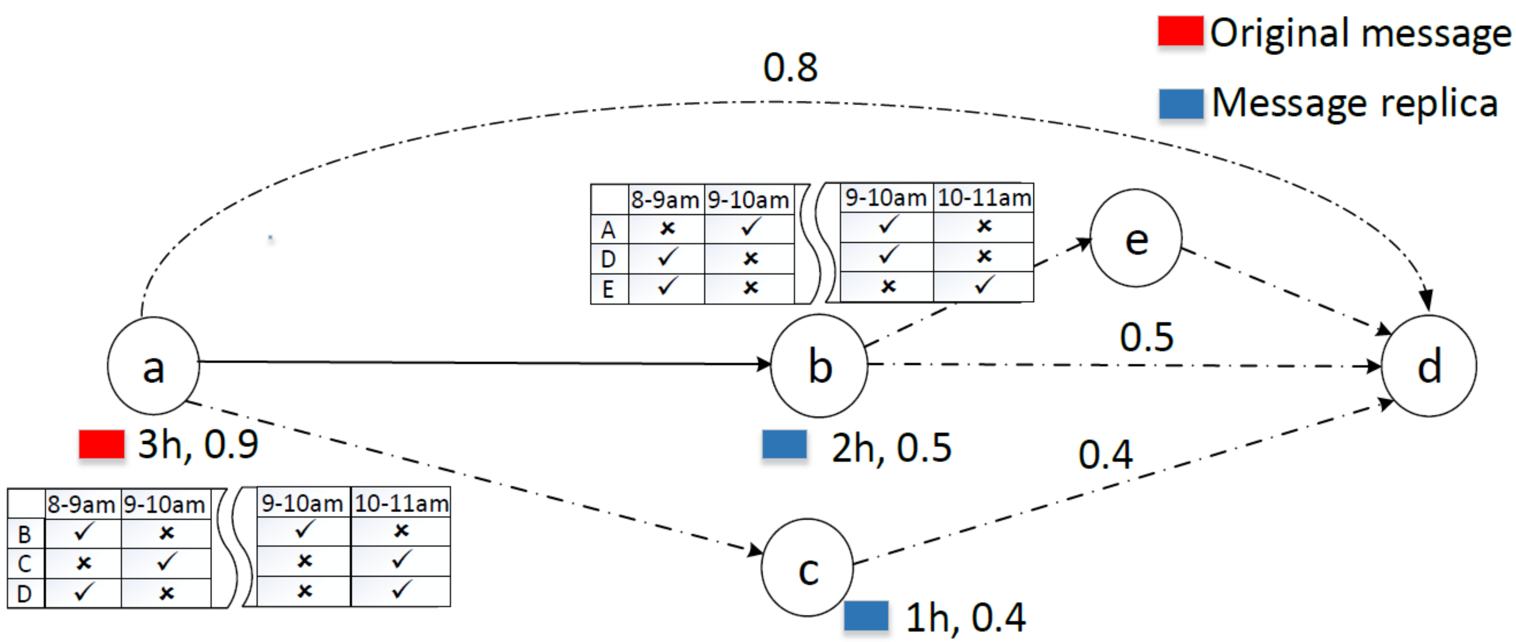
Motivation

- Most of the prediction-based routing protocols leverage the entire history contacts and make an average overall estimation for routing prediction, which leads to inaccuracy
- Cannot immediate detect contact pattern change
- Message replication control is not efficient by only using forwarding metric:
  – Node A forward message to B if: f(A,D) < f(B,D)

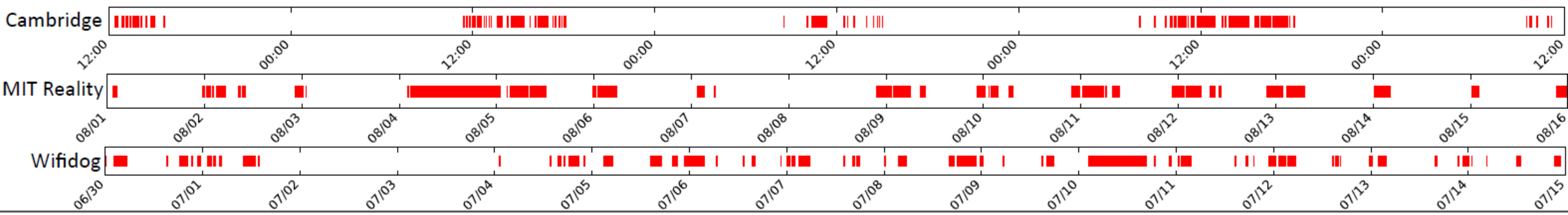


FGDR

- Fine-grained contact pattern characterisation
- Dynamic message replication control



Contact pattern

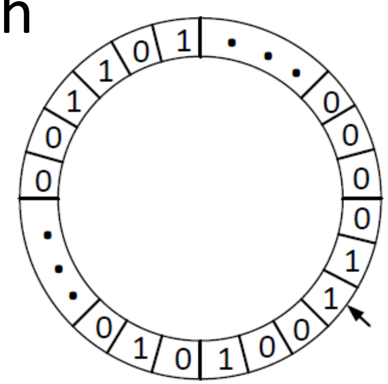


- Contacts between devices usually exhibit a high degree of repetition

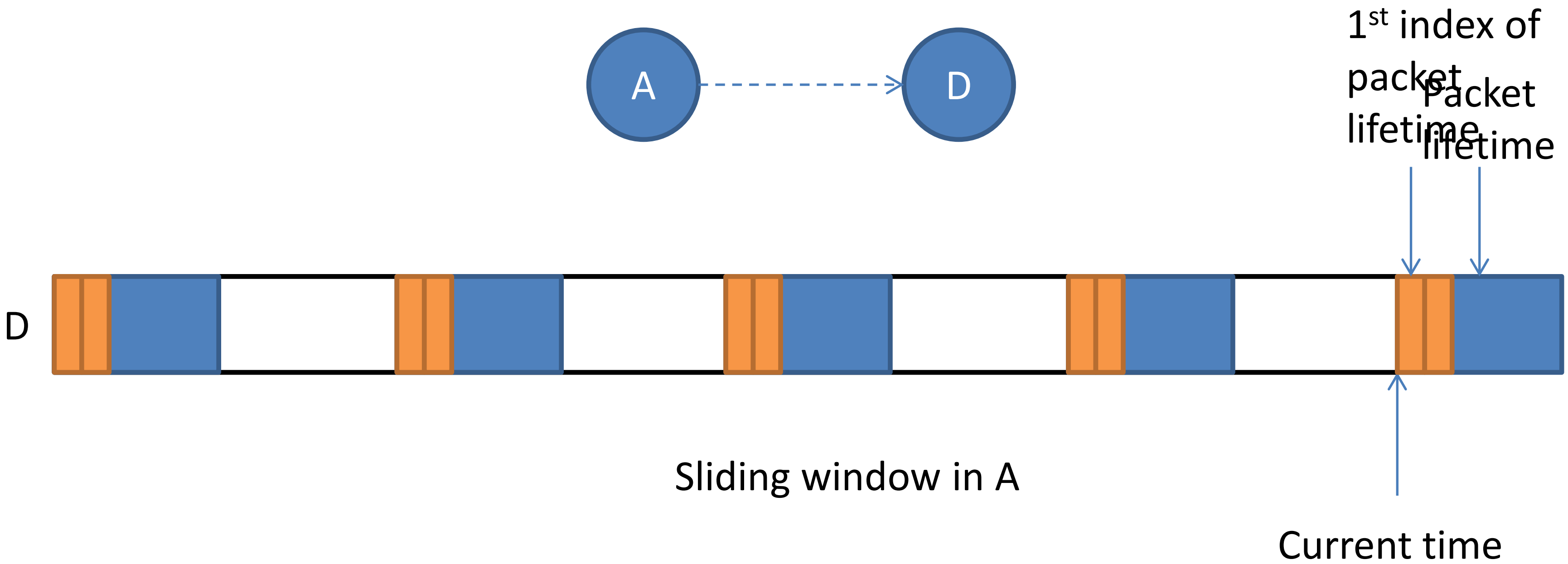
Fine-grained contact pattern characterisation

- History contacts are stored in a bit matrix
- The history contacts of each contacted node are stored in a sliding window
- Each index in the sliding window represents a fixed time duration in real world (1 hour by default)
- 0 represents no contact in current time slot, 1 otherwise
- The length of all sliding windows are limited to 672 for exactly four weeks of history record
- The outdated data are removed in FIFO approach

Index	0	1	2	669	670	671
n_2	1	1	1	1	0	0
n_3	1	0	1	0	0	0
n_4	1	0	0	0	1	1
n_5	0	1	0	1	1	1
n_7	1	1	0	0	1	1



Time-dependent Direct Probability Generation



0	**0**	**0**	**1**

Keep looping…

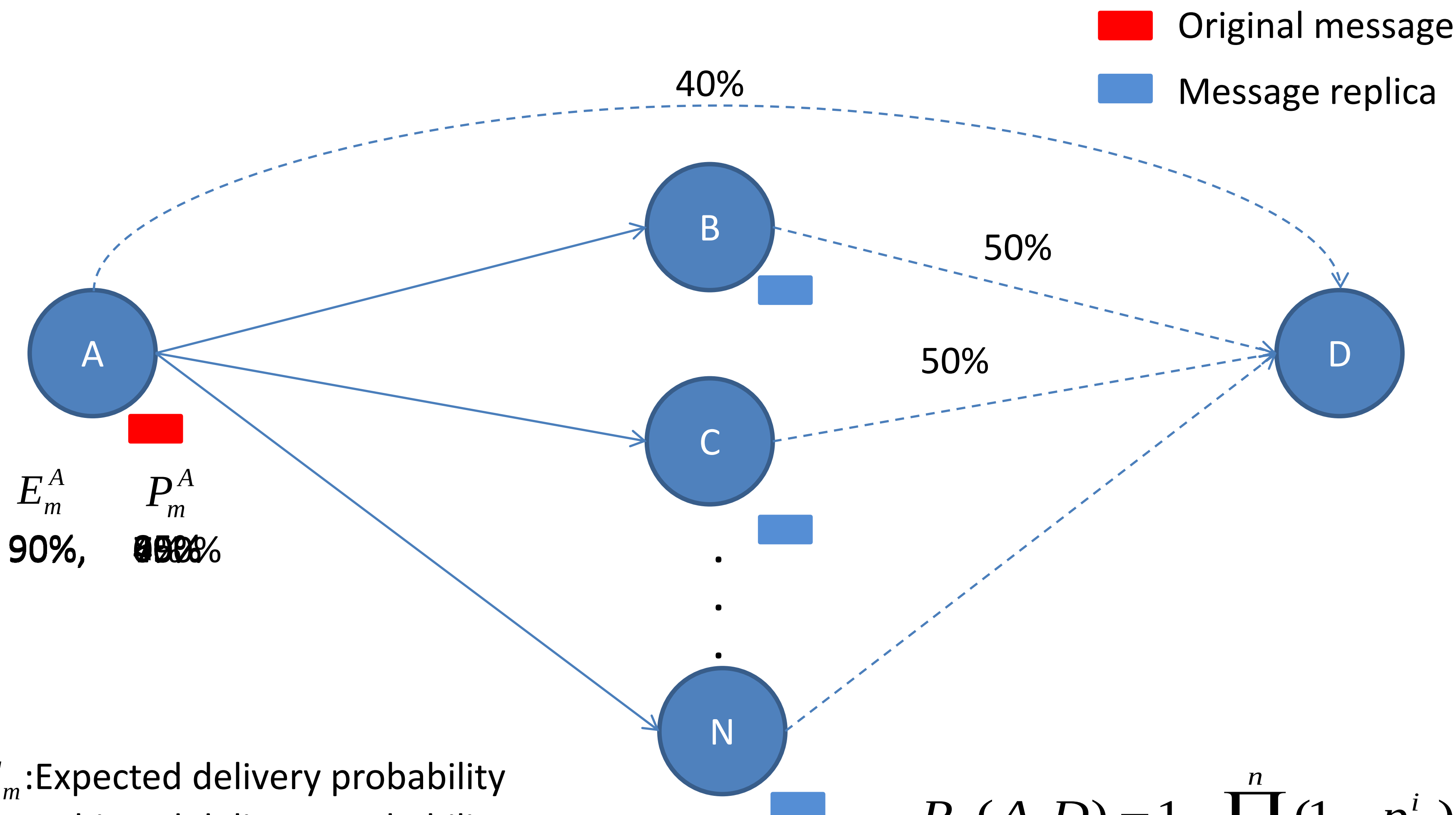
p2 = 3/4

$$p_m(A,D) = 1 - \prod_{i=1}^{n} (1 - p_m^i(A,D))$$

Dynamic Message Replication Control

- Eliminate unnecessary message replication
- A node stop forwarding replica for a message if there are enough message replicas in the network

Dynamic Message Replication Control



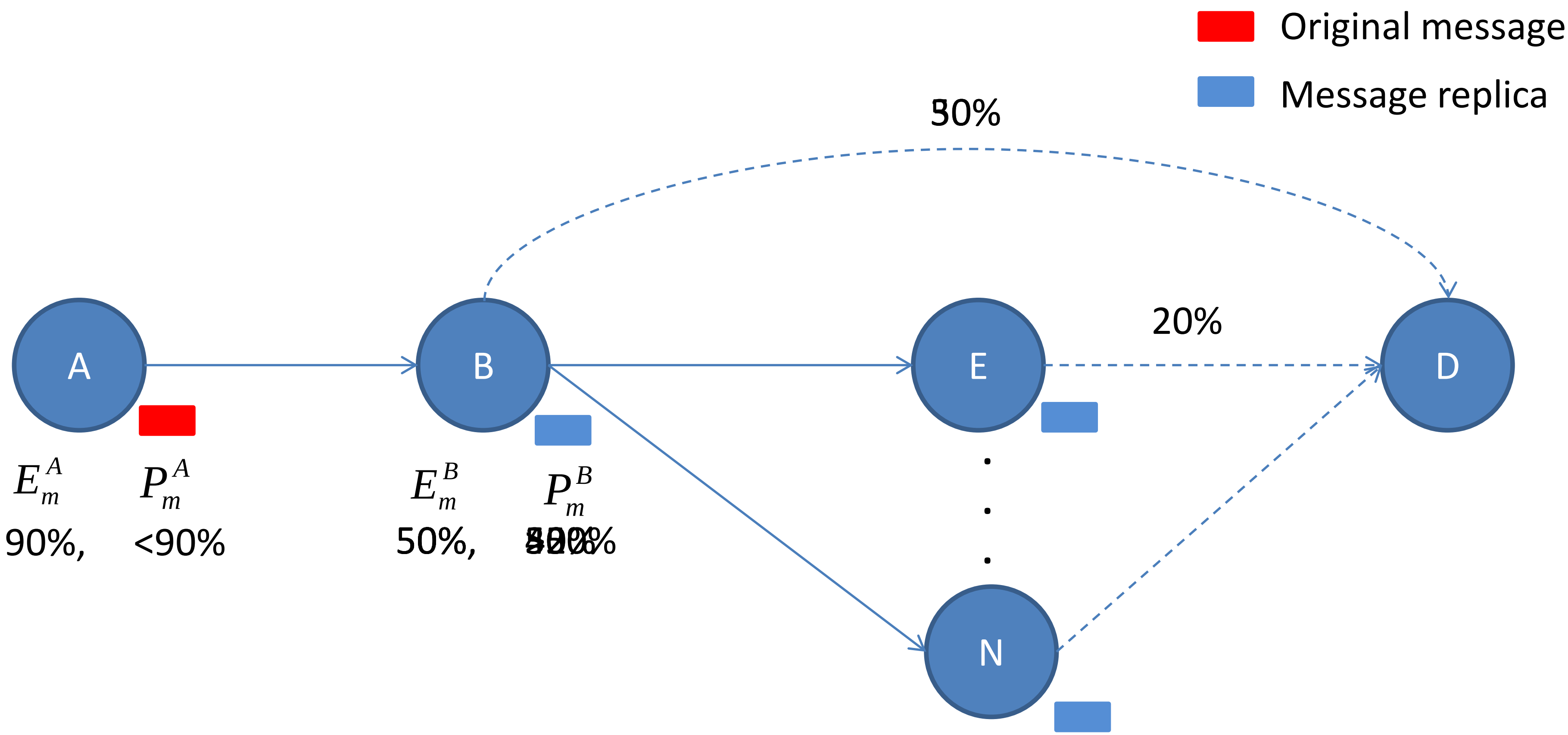
E_m :Expected delivery probability

P_m :Achieved delivery probability

p_m :Time-dependent direct delivery probability

$$P_m(A,D) = 1 - \prod_{i=1}^{n} (1 - p_m^i)$$

After a few time units



E_m :Expected delivery probability
P_m :Achieved delivery probability
p_m :Time-dependent direct delivery probability

$$P_m(B,D) = 1 - \prod_{i=1}^{n} (1 - p_m^i)$$

Dynamic Adjust Threshold γ_m

- γ_m is dynamic adjusted depends on the current status of the packet m:
  - If the remaining packet lifetime $\Delta t_m \downarrow, \gamma_m \downarrow$
  - If the remaining expected delivery probability $\Delta E_m \downarrow, \gamma_m \uparrow$

$$\gamma_m^{new} = \min\{\gamma_m^{old} \times \frac{\Delta t_m}{T_m} \times \frac{E_m}{\Delta E_m}, \gamma_0\}$$

Message Routing Algorithm

Algorithm 1: Replication and Forwarding at Sender

/* Upon encountering node b: */

for *each message* m *in* a'*s buffer* **do**
- **if** $Dest_m == b$ **then**
  - Forward message m to node b;
  - Delete message m;
- **else**
  - Add $< ID_m, Dest_m, T_m, \gamma_m >$ to SV_a;

Send SV_a to node b;

/* Upon receiving ACK for SV_a: */

for *each* $< ID_m, \bar{p}_m >$ *in ACK* **do**
- **case** $P_m(k) < E_m \& \bar{p}_m \geq E_m$
  - Forward message m to b;
- **case** $P_m(k) < E_m \& \bar{p}_m < E_m$
  - Generate a replica of m and forward it to b;
  - Update $P_m(k)$;
- **case** $P_m(k) \geq E_m \& \bar{p}_m > p_m^0$
  - Forward message m to b;

Algorithm 2: Replication and Forwarding at Receiver

/* Upon receiving SV_a from node a: */

for *each* $< ID_m, Dest_m, T_m, \gamma_m >$ *in* SV_a **do**
- **if** *node* b *carries a replica with* ID_m **then**
  - Continue;
- **else**
  - Calculate $\bar{p}_{bDest_m}$ according to Eq. (7).
  - **if** $\bar{p}_{bDest_m} \geq \gamma_m$ **then**
    - Add $< ID_m, \bar{p}_{bDest_m} >$ to the ACK message;

Send ACK to node a;

When A encounter B:

for each message m in A's buffer:

- if B is the destination of m:
  - send message to B
- else:
  - Calculate p_m^B
  - if E_m^A is not fulfilled:
    - if $p_m^B \geq \gamma_m$:

Case1: $P_m^A < E_m^A$ and $p_m^B \geq E_m^A$

$$A \xrightarrow{\text{forward}} B$$

Case2: $P_m^A < E_m^A$ and $p_m^B < E_m^A$

$$A \xrightarrow{\text{replicate}} B$$

- else: (E_m^A is fulfilled)

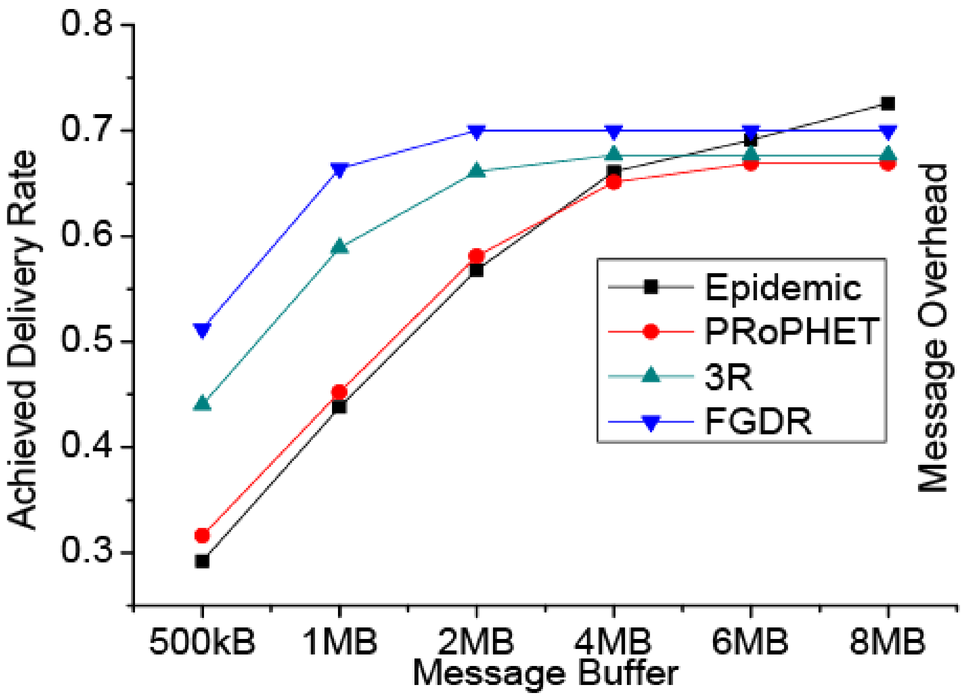
Case3: $P_m^A \geq E_m^A$ and $p_m^B > p_m^A$

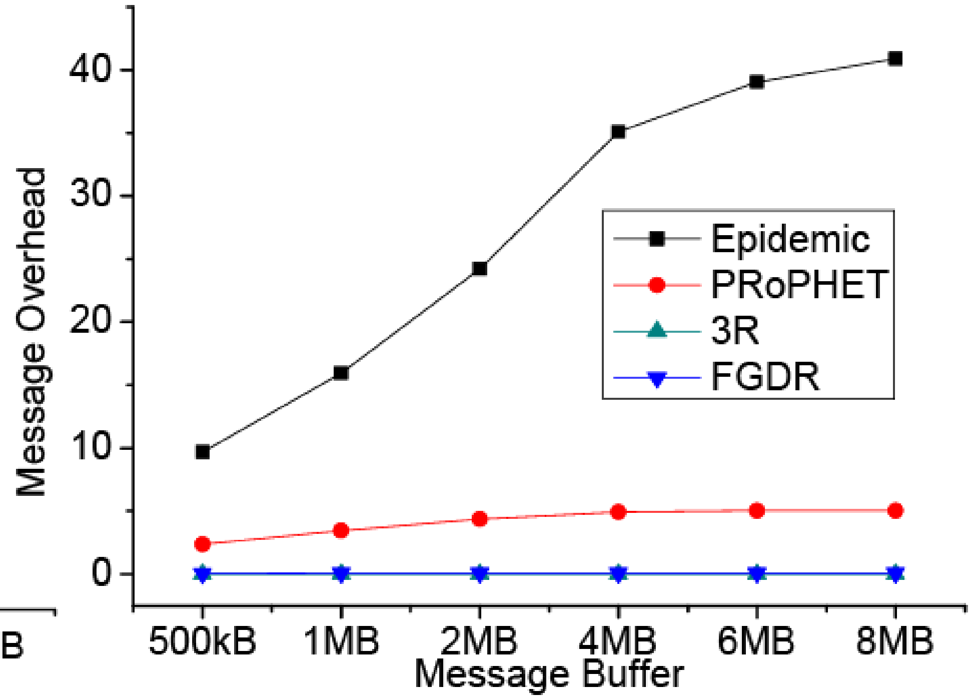
$$A \xrightarrow{\text{forward}} B$$

Performance Evaluation

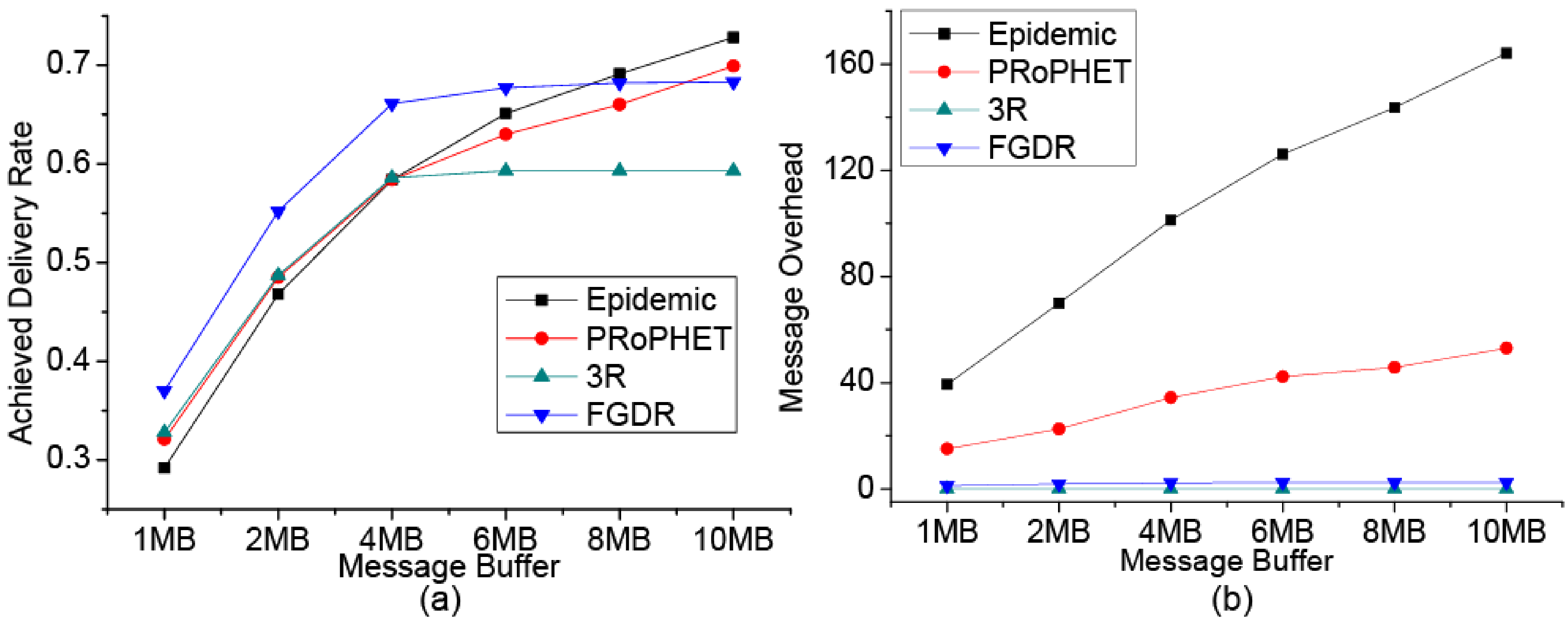
- Nodes are generated based on data traces
- Packets' lifetime: randomly selected from 2 hours to 1 week
- Real world data trace:
  - Cambridge:
    - 223 nodes in total
    - 2 months
    - Almost perfect regular contact pattern
  - MIT Reality:
    - 20883 nodes
    - 11 months
    - Lots of noise and opportunistic contacts

Performance Evaluation(Cambridge)





Performance Evaluation(MIT Reality)



Thanks!