Distributed and Piggybacked Monitoring in Low Power and Lossy Networks

Abdelkader Lahmadi, Alexandre Boeglin, Olivier Festor

81th IETF Meeting – NMRG WG – Quebec city, Canada, July 2011

Outline

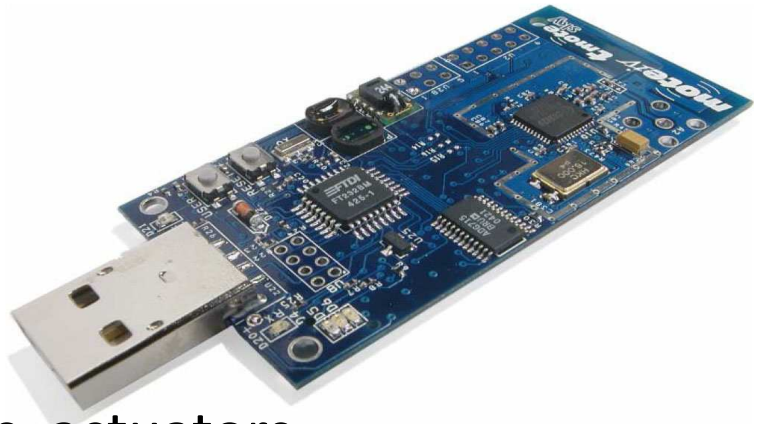
- Context and Introduction
- Basic Approach
- Mechanism Details
- Open Issues
- Future Directions

Introduction

- Problem: monitoring Low power and Lossy Networks (LLN)

6LoWPAN
IPv6-based Low-power Wireless Personal Area Networks

- Changing environment
  - Node failure
  - Poor radio conditions
  - Node movement
- Large number of deployed nodes
  - Urban networks
  - Smart grid
  - Advanced Metring Infrastructure



- High density neighbourhood
  - In-vehicle networks: piles of sensors, actuators
- Energy efficiency: years of operation
  - Structure Health Monitoring

Monitoring Solution: Design Goals

Adaptive Scalable **Cost sensitive**

Accurate Sound Timely **Resilient**

Reactive Proactive **Distributed**

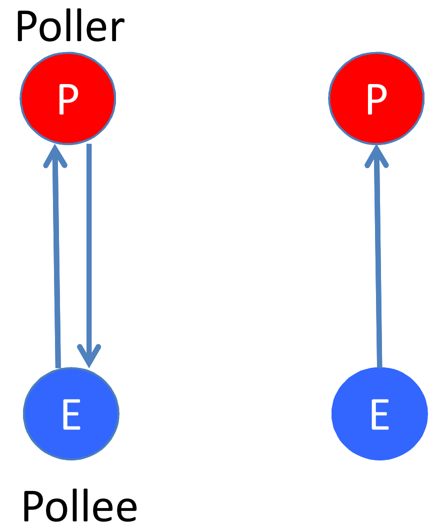
Robust Fault tolerant **Integrative**

Fetahi Wuhib, Mads Dam, Rolf Stadler, “A Gossiping Protocol for Detecting Global Threshold Crossing”, IEEE Transactions On Network And Service Management, Vol 7, March 2010

LLN Monitoring: Challenges

- Provide accurate information about the network in a timely manner
  - Link and signal quality, connectivity, neighbours, battery level, packet loss rate
- Limit the monitoring data overhead
  - Large number of nodes, high density, different types of data (sensors, actuators, multiple polling frequencies)
- Carry sensing traffic and the monitoring data
- Monitoring and sensing data will contend
  - Preserve channel time for the primary task

Poller-Pollee Monitoring Structure



- Polling
  - Poller pulls data through a request
  - Periodically
  - Up-to-data image of the network
  - Large overhead
- Pushing
  - Pollee pushes data towards the poller
  - Periodically or event driven: when a threshold is crossed
- Trap-directed polling
  - Mixture between both: when a report is received, start polling

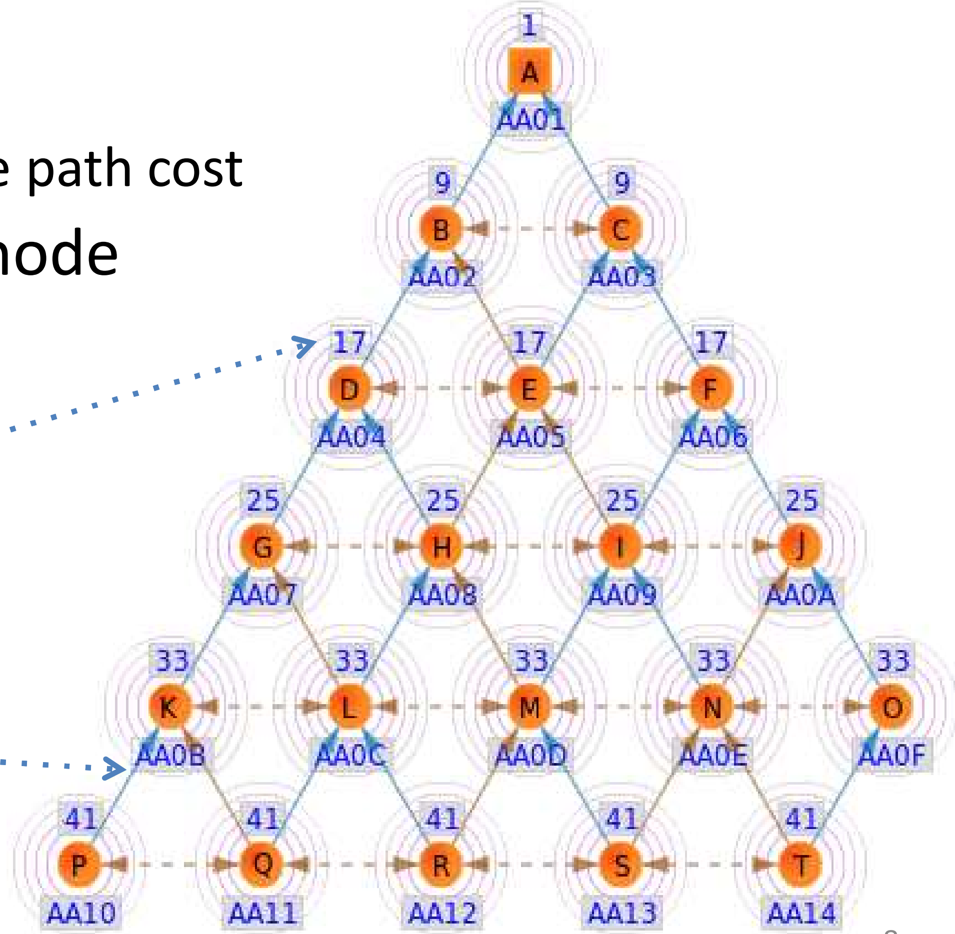
Approach Overview

- Assumption
  - Monitoring traffic is pollees-to-poller
  - A routing overlay is maintained over the LLN
    - RPL protocol is designed to this aim
- Approach: build an adaptive two-tier distributed monitoring overlay
  - Adaptive poller-pollee formation: no prior setup
  - Reduce competition with primary task traffic: raison d'être
  - Lossy monitoring data representation: saves battery

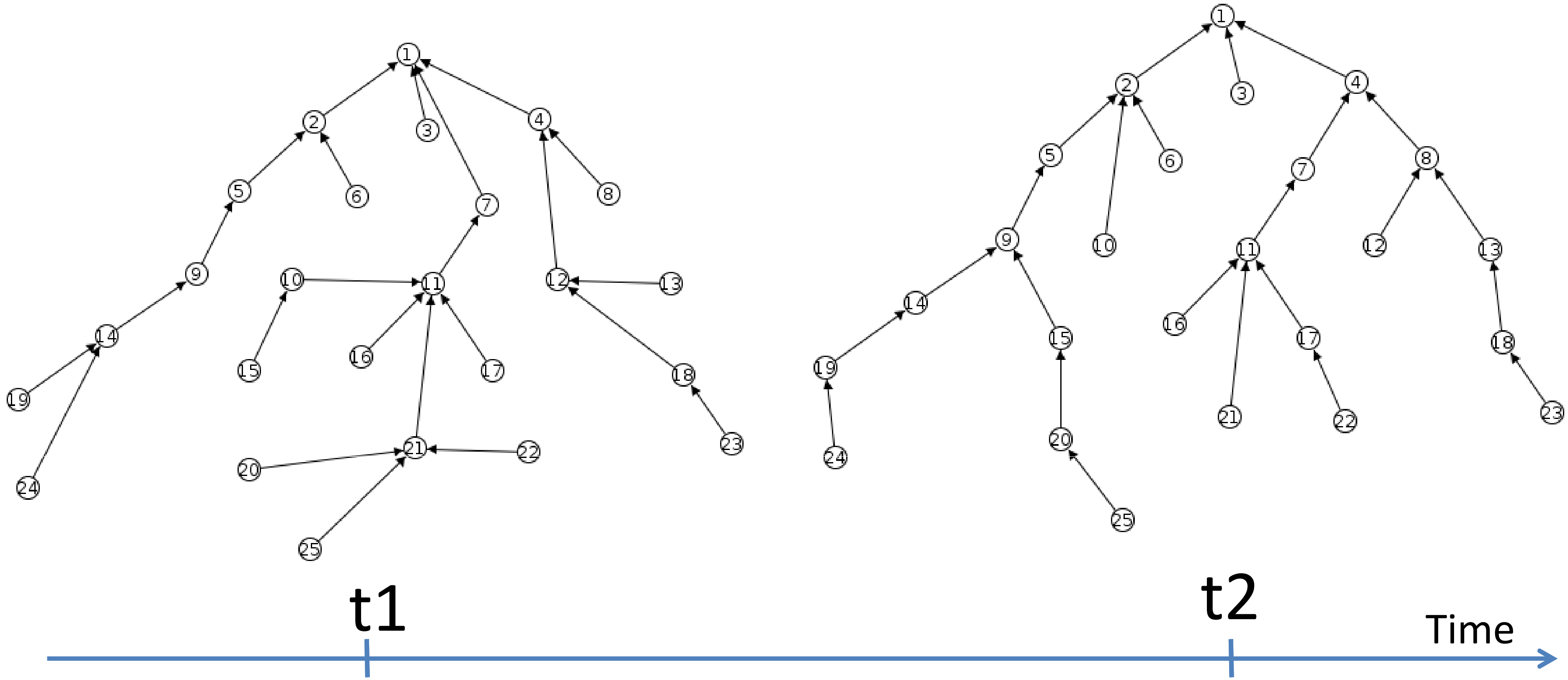
A. Gonzalez Priete, D. Gillblad, R. Steinert, A. Miron, « Toward Decentralized Probabilistic Management », IEEE Communications Magazine 2011

RPL DODAG Construction

- Defines a DAG that forms paths to a single logical root: Destination Oriented Directed Acyclic Graph
- Distance-Vector
  - Advertise cost of path to root
  - Choose parents that minimize path cost
- A rank is assigned to every node
  - Decreases towards root
- Node properties
  - ID
  - Rank
  - Candidate parents set
  - Preferred parent
  - Closed children set
  - Open Children set



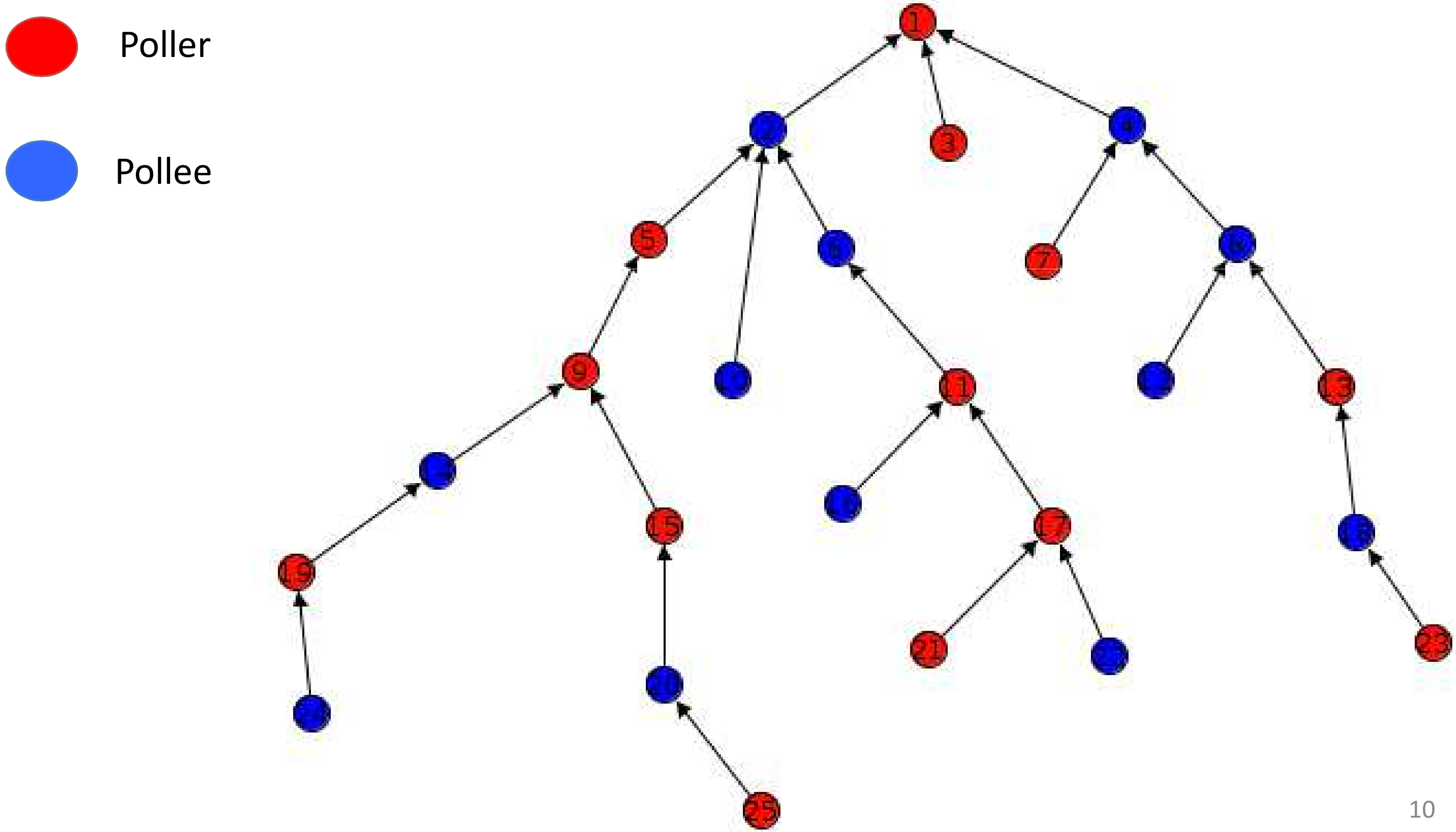
RPL DODAG: a Running Example



- A graph $G(V^t,E^t)$
  - V^t: set of nodes varying over time
  - E^t: set of edges varying over time

Monitoring Roles Placement

- Random placement of pollers and pollees



Monitoring Roles Placement

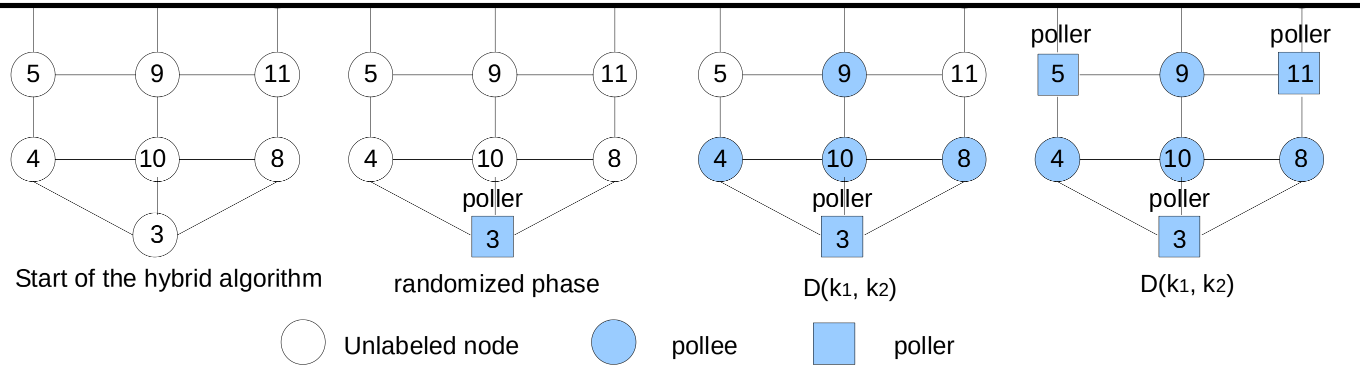
- Liu and Cao, "Distributed Monitoring and aggregation in wireless and sensor networks" , INFOCOM10

Advantages:

- **Minimizes number of pollers**
- **Guarantees a minimum distance between pollers, maximum distance between a pollee and a poller**

Drawbacks:

- **Requires specific setup and maintenance messages**
- **Independent from routing overlay, but some goal similarities**

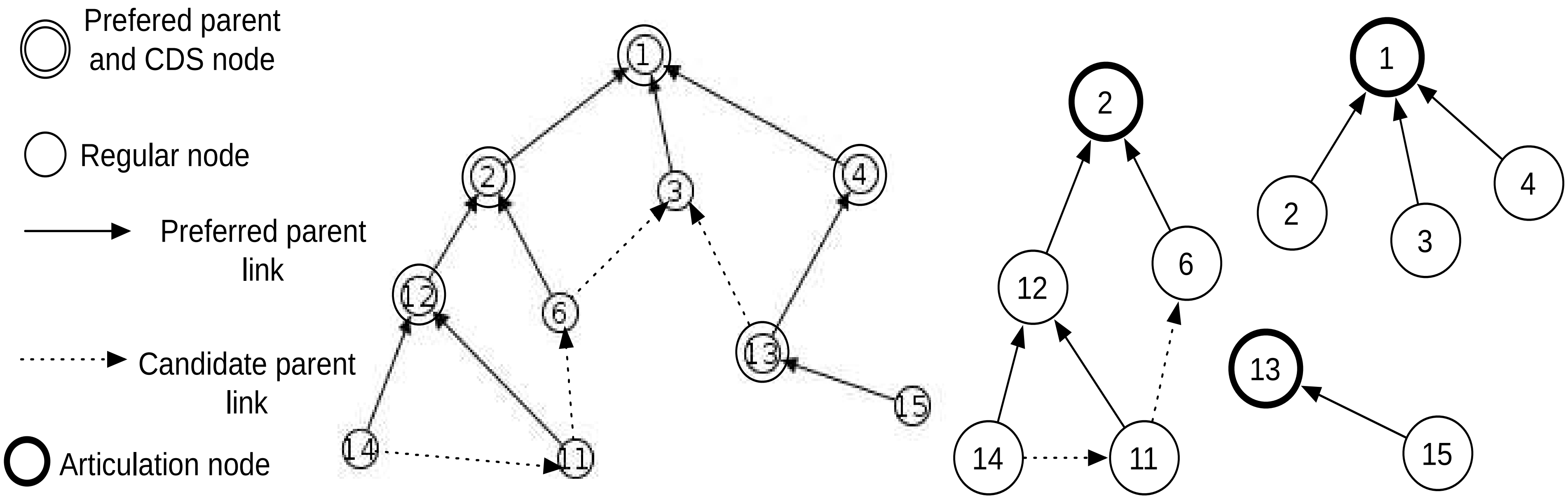


Definitions From Graph Theory

- A Graph G is biconnected if and only if either G is a single edge or for each tuple of vertices u, v there are at least two vertex-independent paths from u to v
- The intersection of two maximum size biconnected components consists of at most one vertex called an **articulation link**

RPL: DODAG

- A DODAG is a set of biconnected directed components
- An articulation parent v of the DODAG is a parent node that has at least a child who has no back parent other than v to reach the data sink.



Biconnected Components of a DODAG

DODAG-Based Monitoring Roles Placement

- Greedy algorithm: use routing information provided by RPL
- First strategy: looking for articulation parents within a DODAG
- Each node piggybacks the number of its candidate parents in the DAO message

---

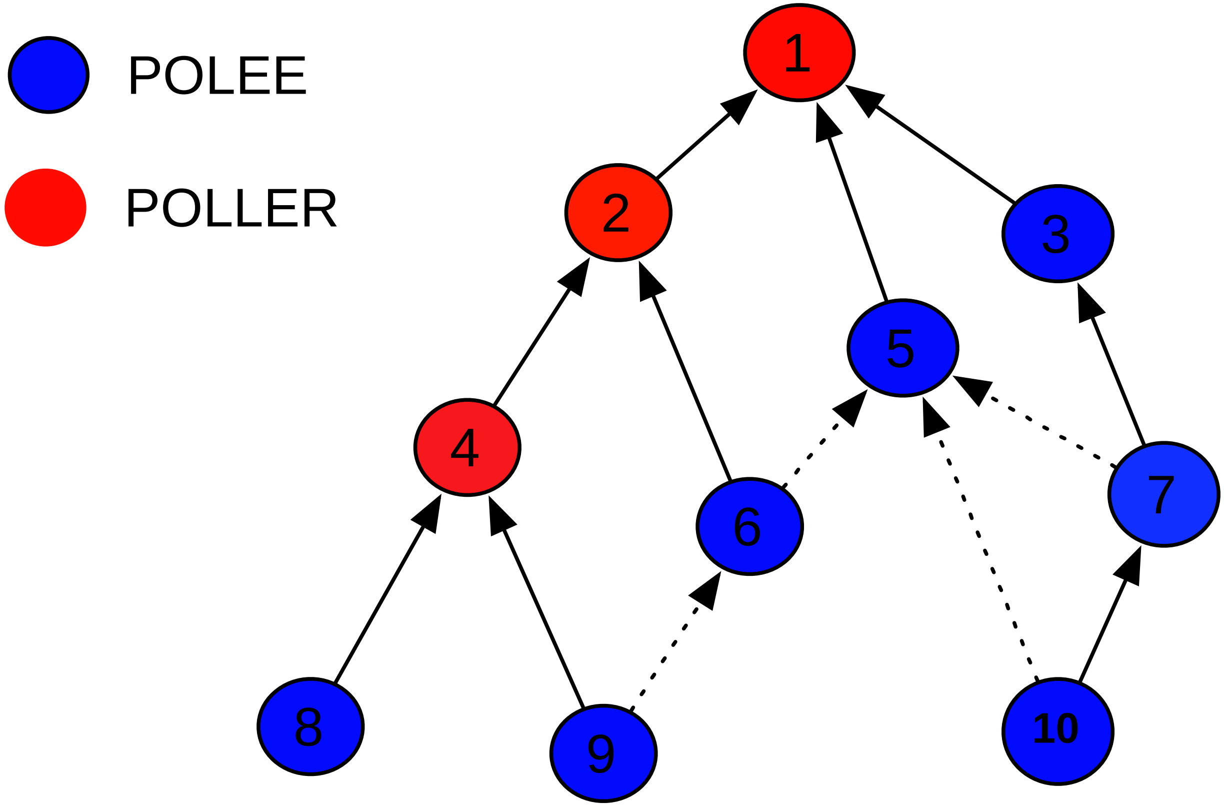
Algorithm 1 Roles placement algorithm.

Input: : N_i is the list of closed children of the current parent node

Input: : $\{n_j\}$ is the respective list of numbers of candidate parents of each child j of the current parent node.

```
Function setRole ()
   Degree = size (N(n_i))
   if Degree == 0 then
      Role = POLLEE
   else if min({n_j}) == 1 then
      Role = POLLER
   end if
```

DODAG-Based Monitoring Roles Placement



+ Adaptive, minor overhead (only the routing process cost), no rigid association between pollers and pollees (deliver to the nearest poller available on the route)

- Depending on the routing process, the distance between a poller and a pollee may be important

DODAG-Based Monitoring Roles Placement

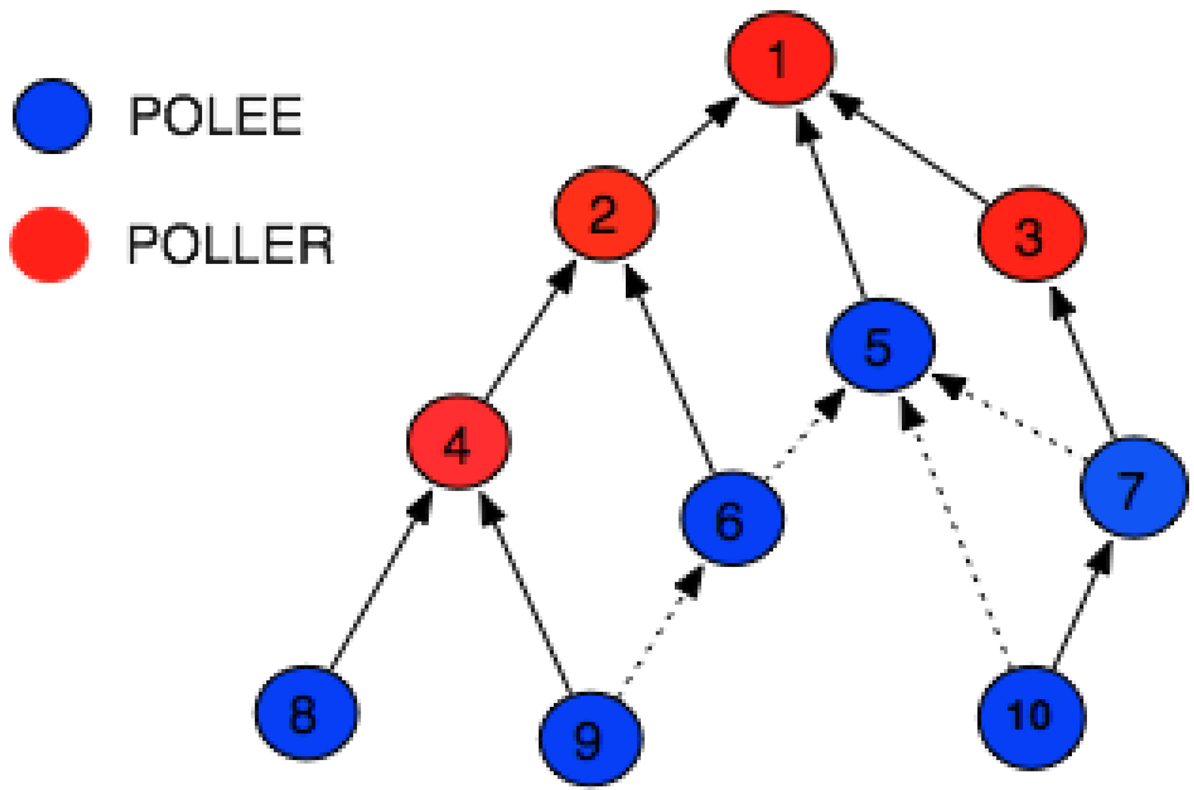
- Greedy algorithm: use routing information provided by RPL
- Second strategy: looking for articulation links within a DODAG

Algorithm 2 Roles placement strategy using articulation links.

```
Input: CH is the list of closed children provided by the
    routing layer.
Input: CP is the list of candidate parents.
Function setRole ()
    Degree = size (CH)
    if Degree == 0 then
        Role = POLLEE
    else if size(CP) == 1 then
        Role = POLLER
    else
        Role = POLEE
    end if
```

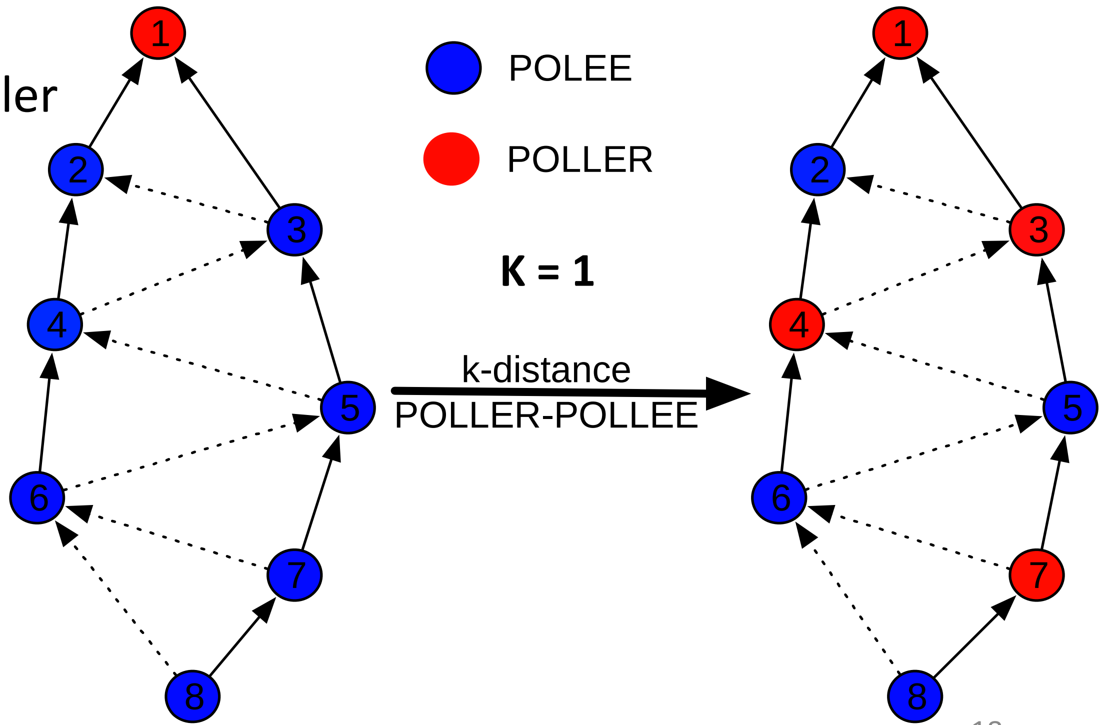
DODAG-Based Monitoring Roles Placement

- Second strategy: looking for articulation links within a DODAG



DODAG-Based Monitoring Roles Placement

- Third strategy: k-distance poller-pollee
- Piggyback the maximum distance *k* between a poller and a pollee in DAO messages
- When a DAO is received by a pollee
  - c = c - 1
  - If c == 0 then { Role = POLLER; c = k; c = c +1; }
  - Send DAO(c)
- When a DAO is received by a poller
  - c = k
  - c = c +1
  - Send DAO(c)



DODAG-Based Monitoring Roles Placement

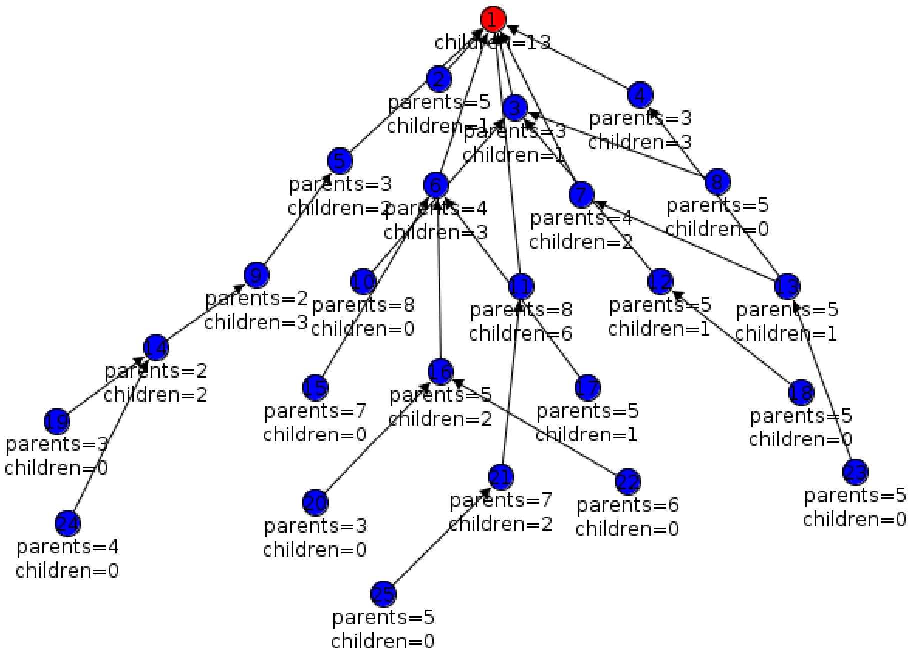
- Fourth strategy: k-distance poller-pollee
- Piggyback the maximum distance *k* between a poller and a pollee in DIO messages
  - Similar to previous algorithm, with top-down propagation
- Strict guarantee of maximum poller-pollee distance, and minimum poller-poller distance

Simulation Environment

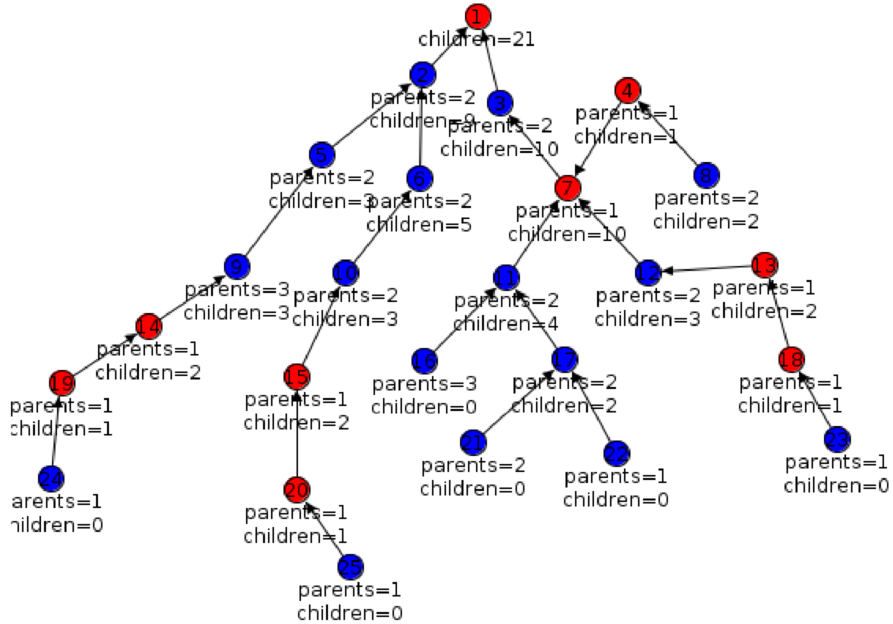
- Cooja simulation tool: contiki project
- RPL and 6loWPAN enabled: ETX Objective Function
- 25 nodes network
- 6 Scenarios

Scenario name	TX ratio	RX ratio	Transmission range	Inference range
Dense-No-Loss	100 %	100 %	100 m	120 m
Medium-No-Loss	100 %	100 %	70 m	90 m
Sparse-No-Loss	100 %	100 %	50 m	60 m
Dense-Loss	100 %	0 %	100 m	120 m
Medium-Loss	100 %	0 %	70 m	90 m
Sparse-Loss	100 %	0 %	50 m	60 m

Articulation-Link Algorithm



Dense-Loss

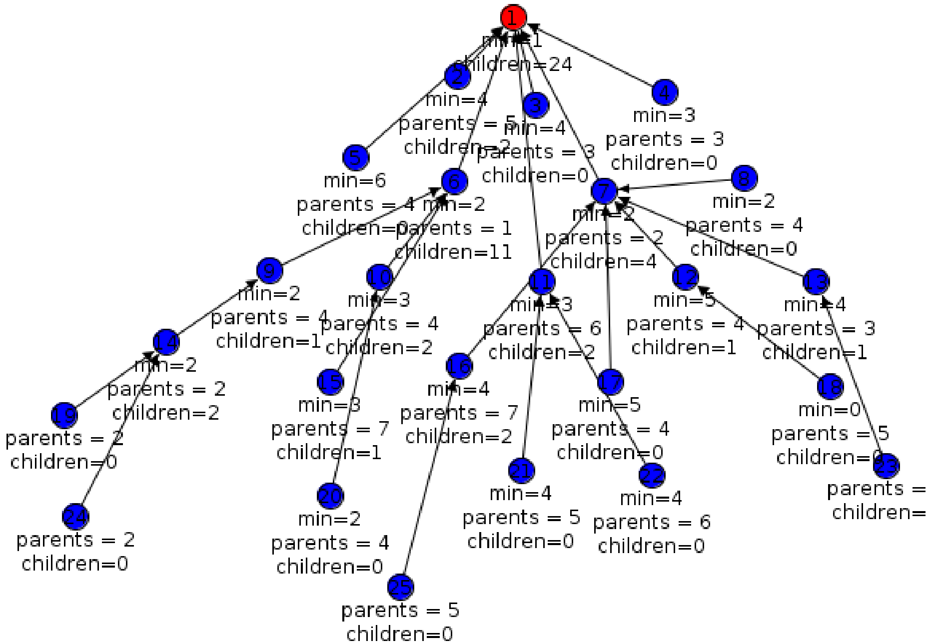


Sparse-Loss

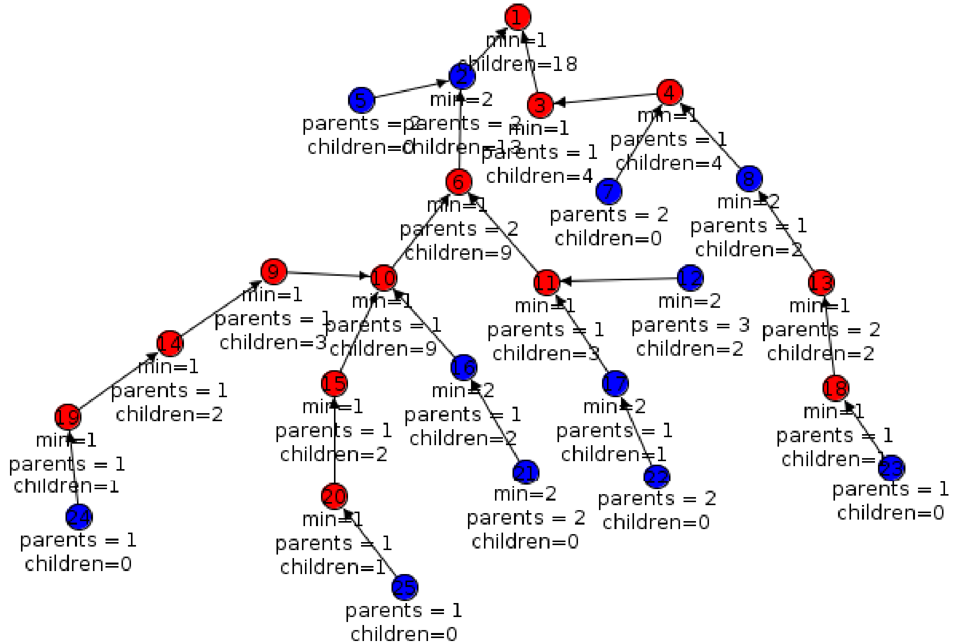
- Only one poller is selected: the root
- Each node has a number of parents > 1
- No articulation links

- 9 pollers are selected
- Average distance between a poller and a pollee is 1.75 hops
- 50 % of pollees are 1 hop from their poller

Articulation-Parent Algorithm



Dense-Loss



- Only one poller is selected: the root
- Each node has a number of parents > 1
- No articulation links

- 13 pollers are selected
- Average distance between a poller and a pollee is 1.25 hops
- 75 % of pollees are 1 hop from their poller

Simulation Results Summary

- Link-articulation algorithm

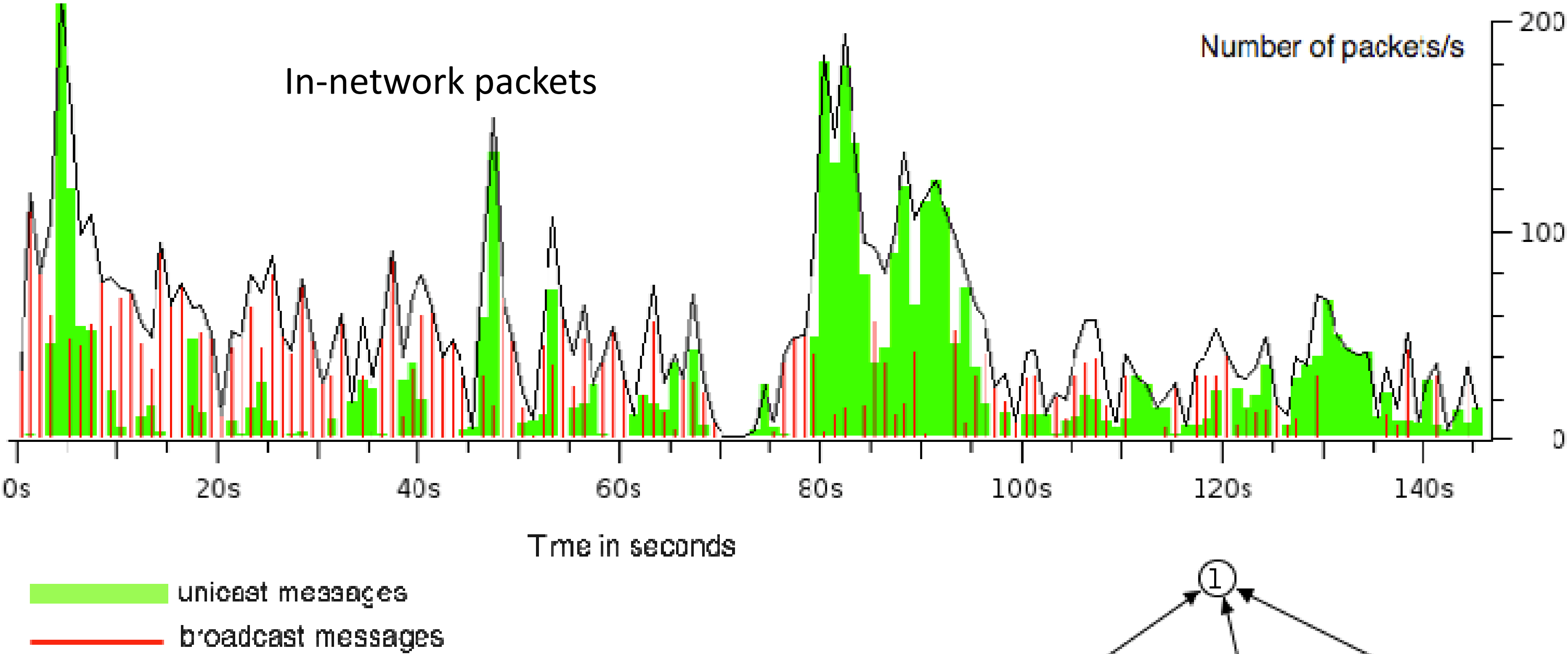
Scenario	Nb Pollers	Nb Pollers w/o Pollees	Average distance	Distance distribution
Dense-No-Loss	**2**	**0**	**1.59**	**50% : 1-hop** **40% : 2-hop** **10%: 3-hop**
Dense-Loss	**1**	**0**	**2.16**	**25%: 1-hop** **41%: 2-hop** **25%: 3-hop** **9% : 4-hop**
Medium-No-Loss	**5**	**0**	**1.45**	**60%: 1-hop** **35%: 2-hop** **5% : 3-hop**
Medium-Loss	**11**	**4**	**1.28**	**78%: 1-hop** **14%: 2-hop** **8%: 3-hop**
Sparse-No-Loss	**8**	**2**	**1.88**	**47%: 1-hop** **23%: 2-hop** **23%: 3-hop** **6%: 4-hop**
Sparse-Loss	**9**	**3**	**1.75**	**50%: 1-hop** **25%: 2-hop** **25%: 3-hop**

Simulation Results Summary

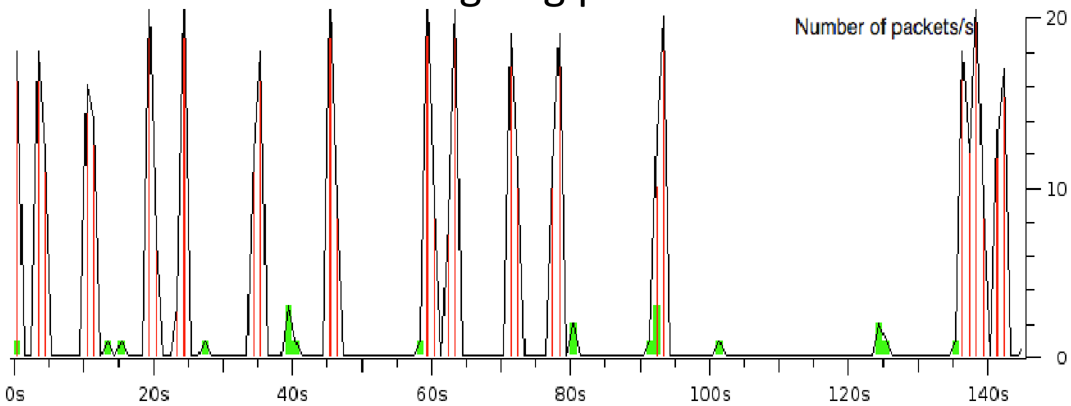
- Parent-articulation algorithm

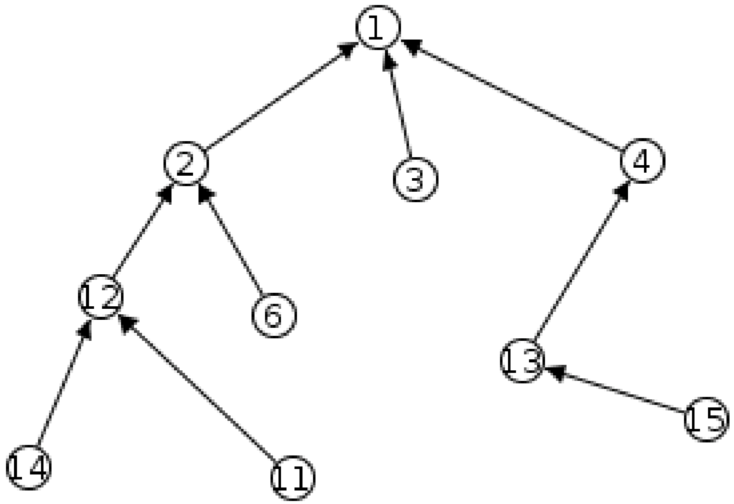
Scenario	Nb Pollers	Nb Pollers w/o Pollees	Average distance	Distance distribution
Dense-No-Loss	**1**	**0**	**2.16**	**25%: 1-hop 41%: 2-hop 25%: 3-hop 9% : 4-hop**
Dense-Loss	**1**	**0**	**2.16**	**25%: 1-hop 41%: 2-hop 25%: 3-hop 9% : 4-hop**
Medium-No-Loss	**6**	**1**	**1.36**	**68%: 1-hop 26%: 2-hop 6%: 3-hop**
Medium-Loss	**8**	**2**	**1.47**	**59%: 1-hop 35%: 2-hop 6%: 3-hop**
Sparse-No-Loss	**12**	**4**	**1.38**	**70%: 1-hop 23%: 2-hop 7%: 3-hop**
Sparse-Loss	**13**	**6**	**1.25**	**75%: 1-hop 25%: 2-hop**

Monitoring Reports Piggybacking



Node 3 outgoing packets



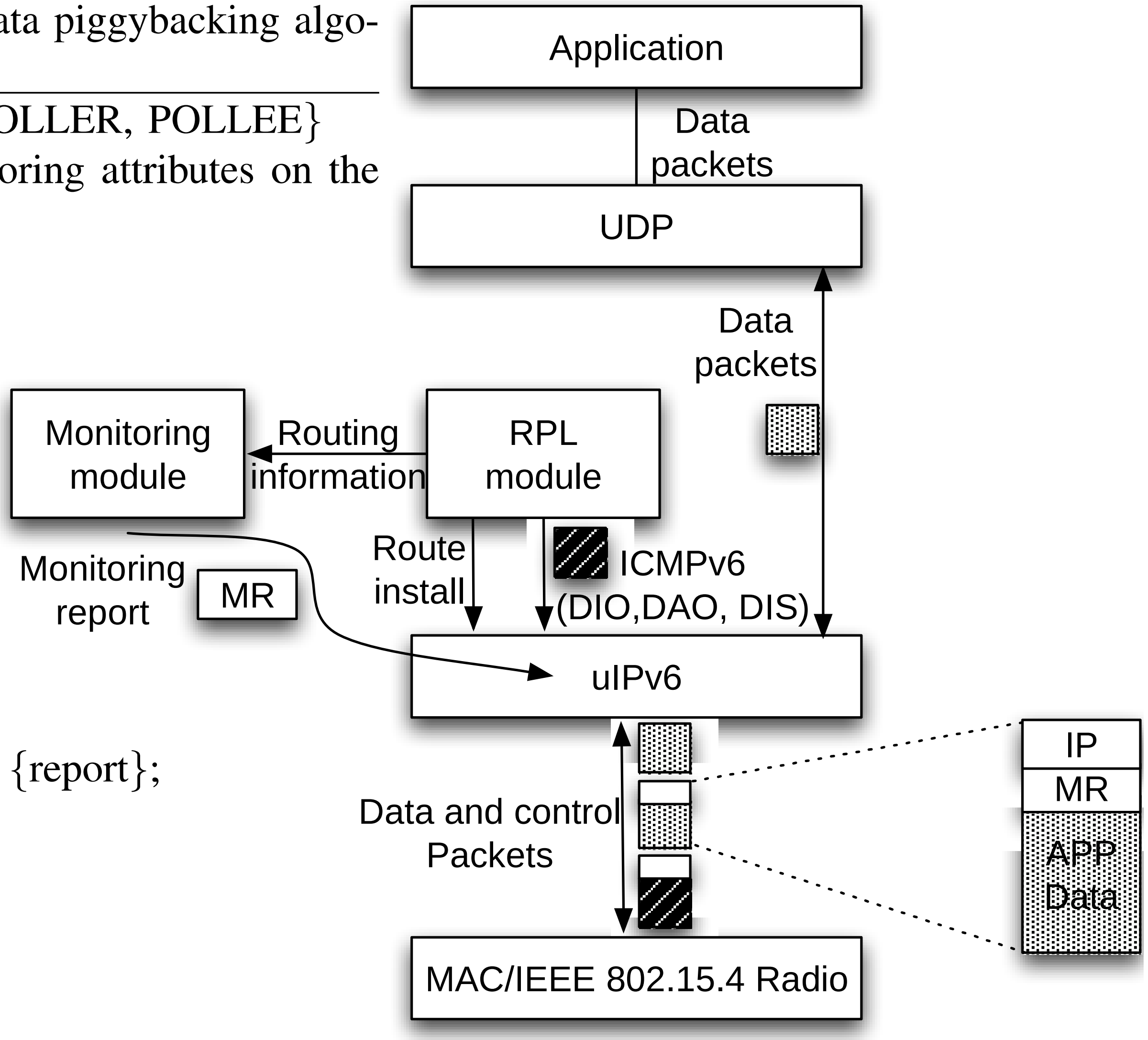


Piggybacking Basic Operation

Algorithm 2 The greedy monitoring data piggybacking algorithm.

Input: Role is the monitoring role {POLLER, POLLEE}
Input: A is the set of available monitoring attributes on the node

```
Function sendReport()
  if Role is POLLEE then
    p = nextPacket();
    report = selectAttributes(A,p);
    piggyback(report,p);
  end if
  if Report received then
    if Role is POLLEE then
      p = nextPacket();
      report = selectAttributes(A,p);
      reports = {Received reports} U {report};
      piggyback(reports,p);
    else
      apply monitoring algorithm;
    end if
  end if
```

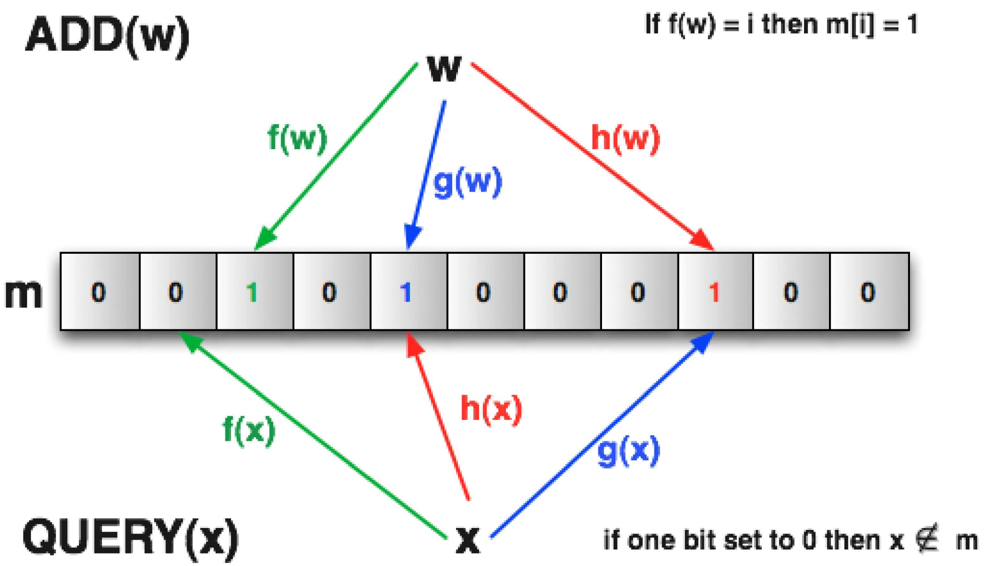


Monitoring Report Representation

- Bloom filter: lossy data structure

A space efficient probabilistic data structure used to test whether an element is a member of a set

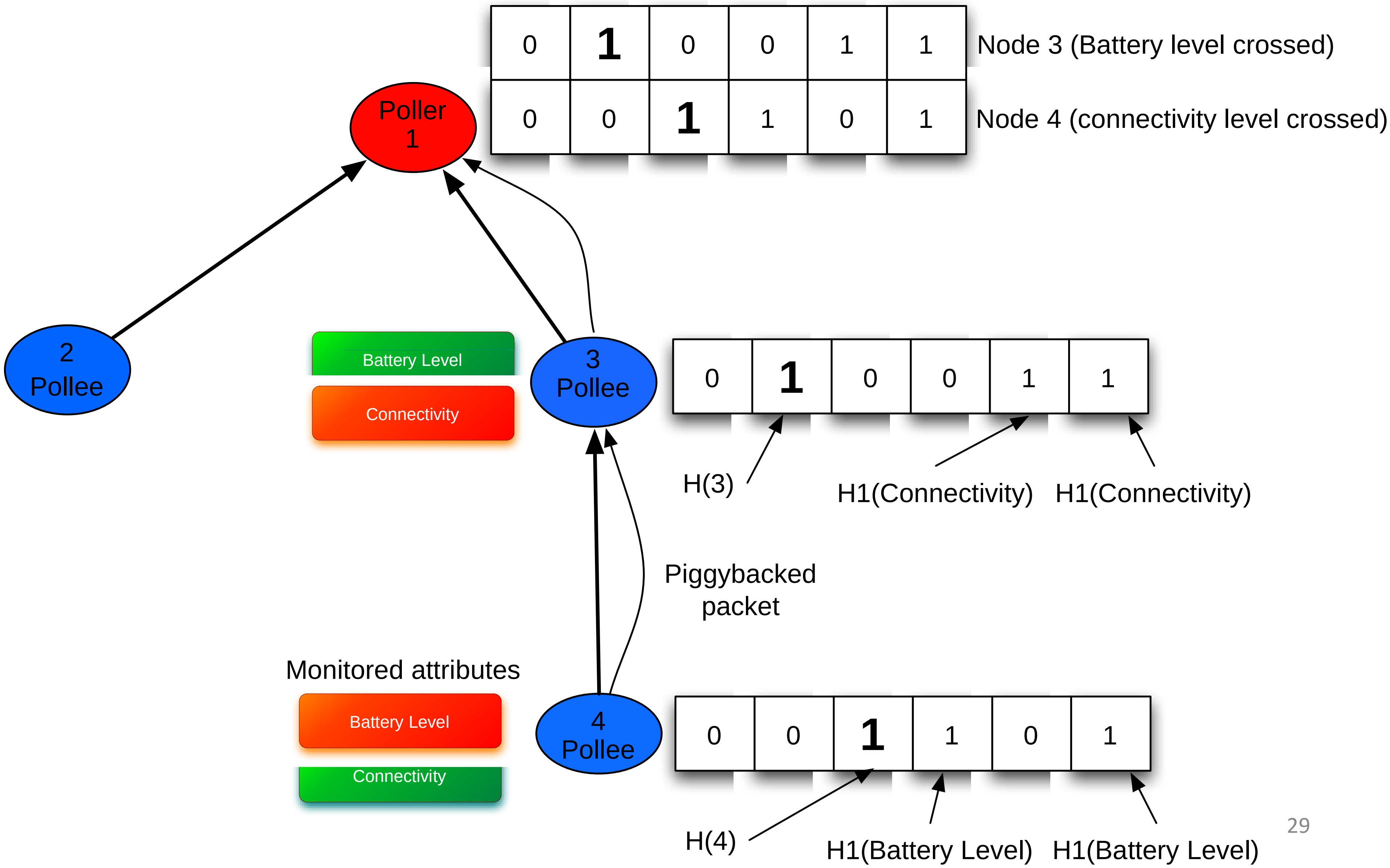
- **m** bits (initially set to 0)
- **k** hash functions: different or the same function with different salts



Reactive Monitoring Bloom Filters

- Each node maintains *N* thresold-based monitoring attributes
- Put *n* crossed threshold-attributes identifiers and the node ID into a Bloom filter and deliver it to the nearest poller
- Bloom filter size $m = -\frac{n \times ln(p)}{ln(2)^2}$
  - P: probability of a false positive, a monitoring variable is matched although it has not been inserted in the filter.
  - The value of P allows to adjust the monitoring report communication cost.

Monitoring Bloom Filter: Basic Operation



Summary and Work-in-progress

- Monitoring Low power and Lossy Networks (LLN) is challenging
- Our approach
  - DODAG based monitoring: monitoring roles placement
  - Piggybacking monitoring reports in traveling packets: reduce cost
  - Reactive monitoring Bloom filters: reduce and adjust cost

- Monitoring Bloom filters analysis
- 6LowPAN piggybacking extension
- Intensive simulation

Future Directions

- Identify LLN requirements in terms of monitoring, for different applications
- Define a set of metrics to evaluate protocols
  - Cost: communication cost (link properties), node cost (node properties)
  - Coverage: number of monitoring nodes, distance between poller and pollees
  - Quality: false alarms when links are unstable
- Determine if one or more existing monitoring protocols meet these requirements

ROLL WG: Survey of Existing Routing Protocol

	p		p		
AODV	pass	fail	pass	fail	fail
DYMO	pass	?	pass	?	?
DSR	fail	pass	pass	fail	fail

- Routing state: limited memory resources of low-power nodes.
- Loss Response: what happens in response to link failures
- Control cost: constraints on control traffic
- Link and Node cost: link and node properties are considered when choosing routes

Source: Slides ROLL-IETF 72

Questions, Comments?