A Localized De-synchronization Algorithm for Periodic Data Reporting in IEEE 802.15.4 WSNs

Giuseppe Anastasi

Pervasive Computing & Networking Lab (PerLab)
Dept. of Information Engineering, University of Pisa
E-mail: giuseppe.anastasi@iet.unipi.it
Website: www.iet.unipi.it/~anastasi/

UNIVERSITÀ DI PISA

PerLab

Hong Kong Polytechnic University, IMC Lab, April 27, 2013

A Localized De-synchronization Algorithm for Periodic Data Reporting in IEEE 802.15.4 WSNs

Based on Joint work with

Domenico De Gugliemo, University of Pisa, Italy

Marco Conti, IIT-CNR, Italy

IN SUPREMÆ DIGNITATIS 1343

UNIVERSITÀ DI PISA

PerLab

Hong Kong Polytechnic University, IMC Lab, April 27, 2013

Overview

PerLab

- Introduction and Motivations
- Related Work
- ASAP Algorithm
- Simulation Results
- Current Activity

Introduction
& Motivations

Introduction & Motivations

PerLab

- Energy efficiency is typically the major concern in WSNs
- In certain scenarios, additional requirements need to be considered
  - Reliability
  - Scalability
  - Timeliness
  - …
- IEEE 802.15.4/ZigBee is unsuitable
  - TDMA is typically used in such scenarios

Introduction & Motivations

PerLab

- TDMA provides
  - guaranteed bandwidth
  - high energy efficiency
  - absence of collisions (i.e. reliability)
  - Predictable latency
- But
  - has limited flexibility
    - ⇨ A change in the operating conditions may require re-computing the transmission schedule
    - ⇨ Finding a collision-free schedule in multi-hop WSNs may be hard
  - requires synchronization among sensor nodes

De-synchronization

- De-synchronization is the opposite of synchronization
  - each sensor node performs its periodic data transmissions as far away as possible from all other nodes
- Goal
  - arrange periodic transmissions from different sensor nodes in an interleaved, round-robin style
    - ⇨ like in conventional TDMA
  - without requiring a strict synchronization among sensor nodes

Related Work

De-synchronization in single-hop WSNs

PerLab

- Bio-Inspired De-synchronization
  - Sensor nodes generates packets periodically
  - Initially, each node selects a random *firing time* within the period.
  - At each step, every node
    - ⇨ observes the firing time of all other nodes
    - ⇨ and derives its own firing time for the next period

      (e.g., as the midpoint between the firing times of the nodes that fired before and after it)

J. Degesys, I. Rose, A. Patel, R. Nagpal, **DESYNC: Self-Organizing Desynchronization and TDMA on Wireless Sensor Networks**, Proc. *Int'l Conference on Information Processing in Sensor Networks*, Cambridge, USA, April 25-27, 2007.

R. Pagliari, Y. Hong, A. Scaglione, **Bio-Inspired Algorithms for Decentralized Round-Robin and Proportional Fair Scheduling**, *IEEE Journal on Selected Areas in Communications (J-SAC)*, Vol. 28, N. 4, May 2010.

De-synchronization in multi-hop WSNs

PerLab

- Multi-hop DESYNC
  - The DESYNC algorithm has been extended to multi-hop sensor networks

    J. Degesys, R. Nagpal, **Towards De-synchronization of Multi-hop Topologies**, *IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2008)*, Venice, Italy, October 20-24, 2008.

- Localized Multi-Hop De-synchronization
  - Addresses the hidden node problem
  - Each node relies on external information
    - ⇨ number of nodes, priorities
  - Not fully localized

    H. Kang, J. Wong, A Localized Multi-Hop Desynchronization Algorithm for Wireless Sensor Networks, Proceedings of IEEE INFOCOM 2009, Rio de Janeiro, Brazil, April 19-25, 2009

Limits of previous schemes

PerLab

- They are not fully localized
  - Relies on external information to adjust their transmission (firing) times
    - ⇨ received from either the sink or other sensor nodes
- They are not robust against packet losses
  - a missing information may compromise the correct behavior
- They are not energy efficient
  - Nodes need to remain active more than necessary to receive external information

Our Proposal

Asynchronous Adaptive Periodic (AsAP) access

- Completely decentralized and local
  - Nodes decide transmission (firing) times *autonomously*
  - On the basis of *local information* only
  - Robust against losses and energy efficient
- Customized to IEEE 802.15.4 MAC
  - Non-Beacon Enabled Mode (CSMA/CA)
  - Can be extended to any contention-based MAC protocol
    - ⇨ With minor changes

AsAP Algorithm

Assumptions and Basic Ideas

PerLab

- Assumptions
  - Single-hop network
  - Packets are generated periodically
    - ⇨ All sensor nodes have the same period T_a
  - The algorithm operates on top of the IEEE 802.15.4 MAC

- Basic ideas
  - Initially all nodes select a random transmission time within the period T_a
  - Then, each node dynamically adjusts its transmission time depending on the outcome of the previous transmission
  - After some time, a transmission schedule (almost) free of collision is obtained.

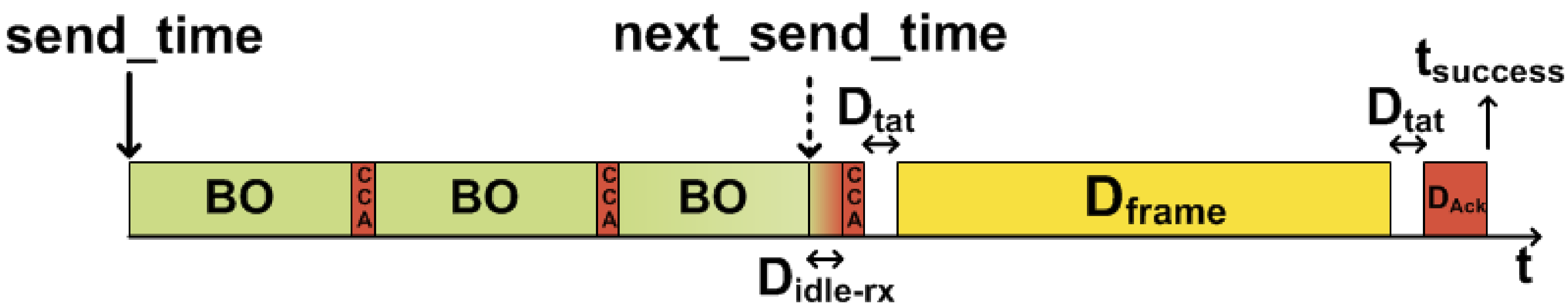
AsAP Algorithm

PerLab

- Upon sending a packet to the sink, four different outcomes can occur
  i. The packet is correctly received by the sink after the first attempt
  ii. The packet is correctly received by the sink after one or more re-transmissions
  iii. The packet is discarded due to exceeded number of channel access attempts
  iv. The packet is discarded due to exceeded number of re-transmissions

AsAP Algorithm (case i)

- If the packet is successfully transmitted at the first attempt (i.e., without retransmissions)
  - **no collisions** have occurred
  - the selected portion of the period is (apparently) free of competitors.
- The same time interval will be reused in the next period.



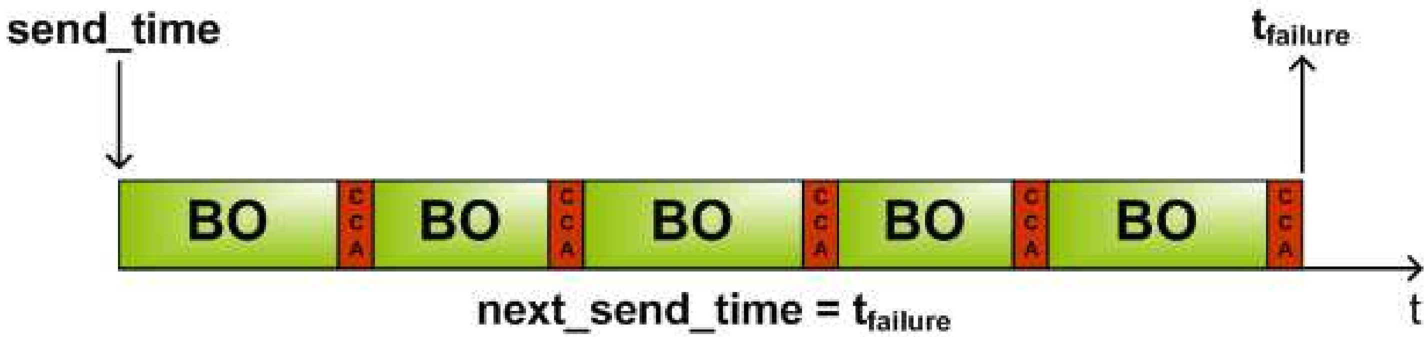
- To minimize latency and energy consumption, the preliminary phase due to the random backoff time will be avoided.

AsAP Algorithm (case ii)

- The packet is successfully transmitted with one or more re-transmissions
  - The algorithm assumes that re-transmissions are caused by channel errors
  - Channel errors are typically caused by a transient phenomenon
- The next send time is exactly the same as in the current period

AsAP Algorithm (case iii)

- The packet is discarded due to exceeded number of backoff stages.
  - The portion of time selected by the sensor node is congested



- It is convenient not to retry the same time in the next period
- The sensor node will start sending at a time corresponding to the end of the current transmission.

AsAP Algorithm (case iv)

PerLab

- The packet is discarded due to exceeded number of retransmissions
  - Communication errors (bad channel)
    - ⇨ it makes no sense to change the transmission interval as channel unreliability is typically a transient phenomenon.
  - Collisions (e.g., due to a hidden node)
    - ⇨ a change in the transmission interval is convenient

- The real cause is unknown to the sensor node
  - The algorithm initially assumes that retransmissions are caused by communication errors → the transmission time remains unchanged
  - If the problem persists the algorithm selects a new transmission time, randomly in [0, T_a]

AsAP Algorithm

Algorithm 1: AsAP Algorithm

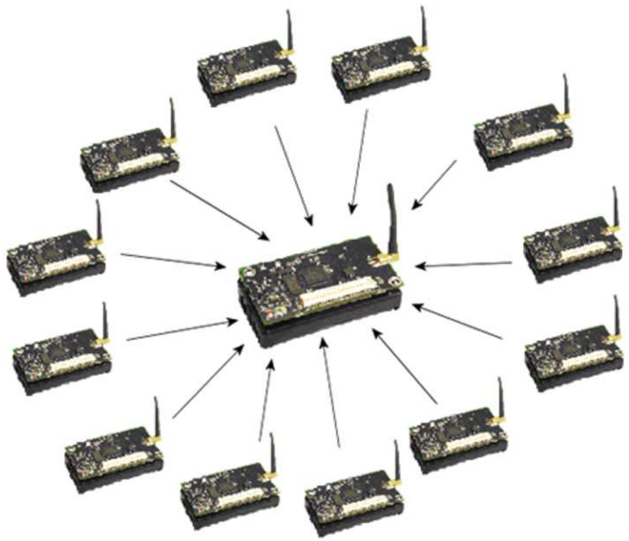
```
macMinBE = 3; P_c=0.5;
failure_count=0; failure_threshold=3;
choose τ in [0,T_a];
next_send_time= τ ;
Loop
sleep until time=next_send_time;
send packet;
wait(notification);
switch(notification) {
case(tx-success AND no-retransmissions)
  next_send_time =
      = (t_success-D_ack-D_tat-D_frame-D_tat-D_CCA-D_idle-rx) mod T_a;
  macMinBE = 0;
case (tx-success AND retransmissions)
  next_send_time = next_send_time;
case (tx-failure  AND exceeded-number-of-backoffs)
  next_send_time = t_failure mod T_a;  macMinBE=3;
case (tx-failure AND exceeded-number-of-rtx)
  failure_count++;
  If (failure_count < failure_threshold)
        next_send_time = next_send_time;
  else
    P_c:        choose τ in [0,T_a]; next_send_time= τ ;
                macMinBE=3;
    (1-P_c):    next_send_time = next_send_time;
  end if
}
end Loop
```

Simulation Analysis

Simulation Setup

- ns2 simulation tool

PHY layer	2.4 GHz
Bit Rate	250 Kbps
Sensor nodes	from 1 to 180
Distance from Coordinator Node	10m
CS range	30m
RX range	15m
Traffic Generation	Periodic (period ~1s)
Message Size	127 bytes
Messages per Period	1
Message Loss Rate	0%
Coordinator always ON	



Comparison

PerLab

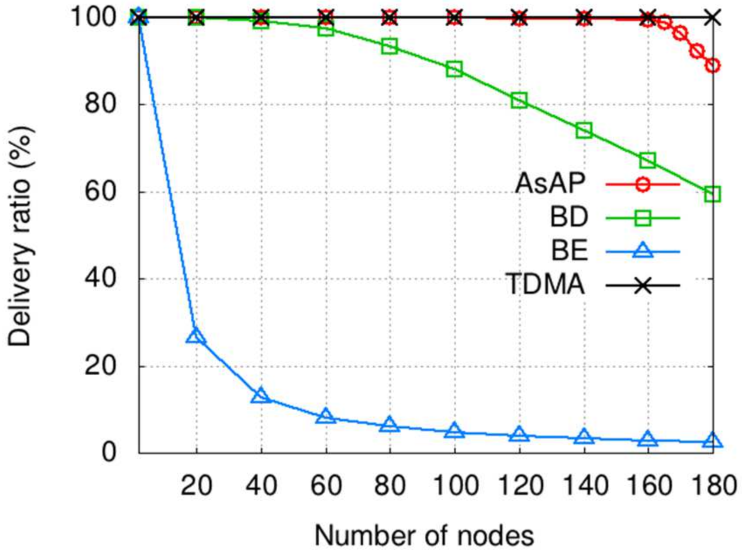
- 802.15.4 MAC in Beacon Enabled Mode (BE)
  - *All* sensor nodes compete for channel access at the beginning of *each* Beacon period.
  - This scheme maximizes competition
- 802.15.4 MAC in Beacon Disabled Mode (BD)
  - Sensor nodes are assumed to transmit data packets at random times within the period.
  - This scheme tries to minimize contention, however conflicts can still occurs.
- TDMA
  - Each node uses its own slot
  - Conflicts are avoided

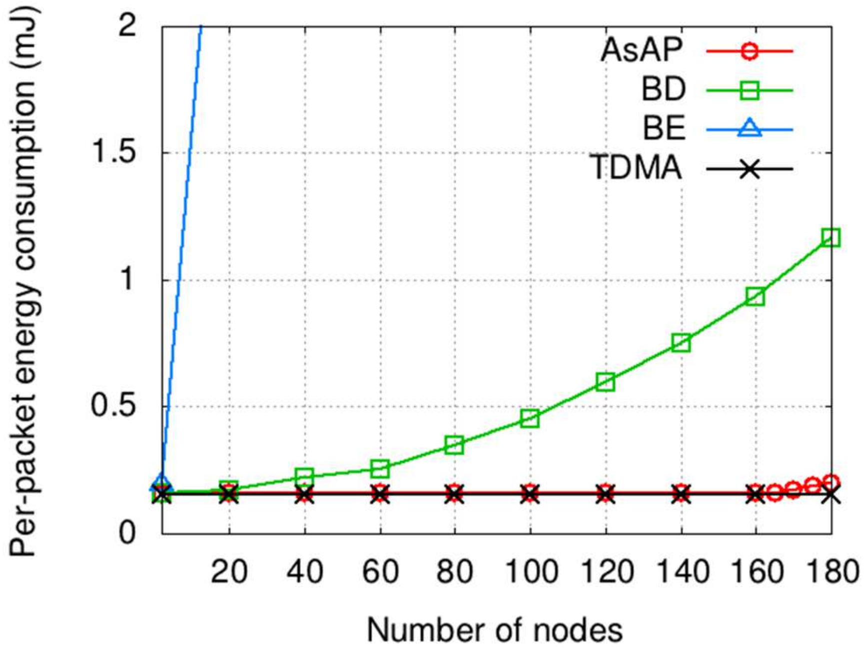
Performance Indices

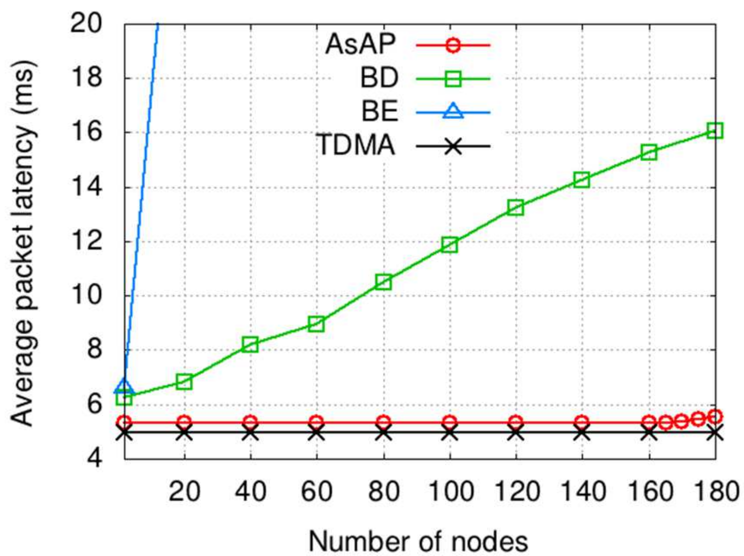
- Delivery Ratio
  - ratio between the number of data packets correctly received by the sink and the total number of data packets generated by all sensor nodes.
- Average Latency
  - average time from when the packet transmission is started at the source node to when the same packet is correctly received by the sink
- Avg. Energy Consumption per Packet
  - total energy consumed by all sensor nodes divided by the overall number of data packets correctly delivered to the sink

Analysis in stationary conditions

PerLab

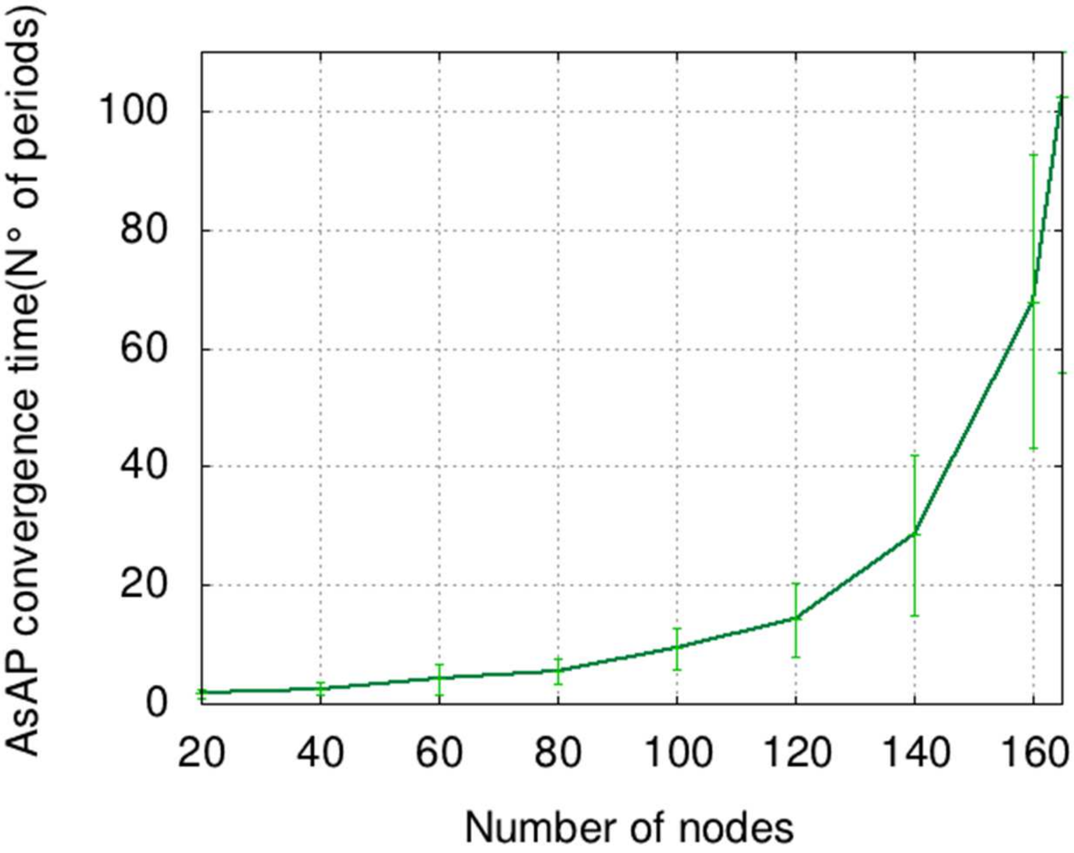






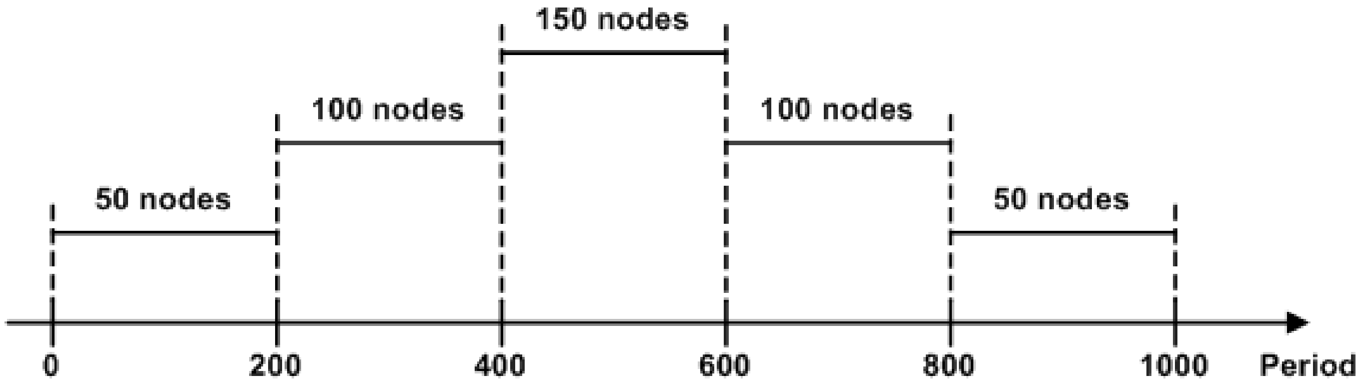
Converge Time

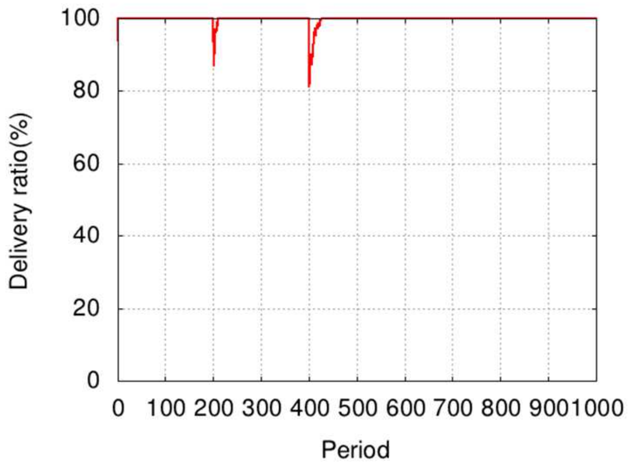
- Time taken to reach a steady state
  - when no sensor node changes its transmission interval

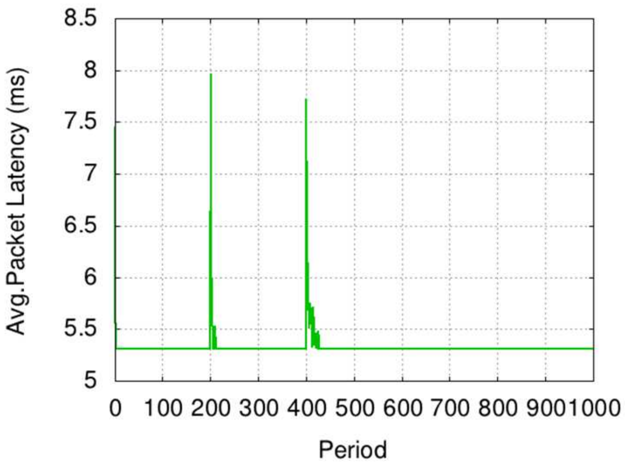


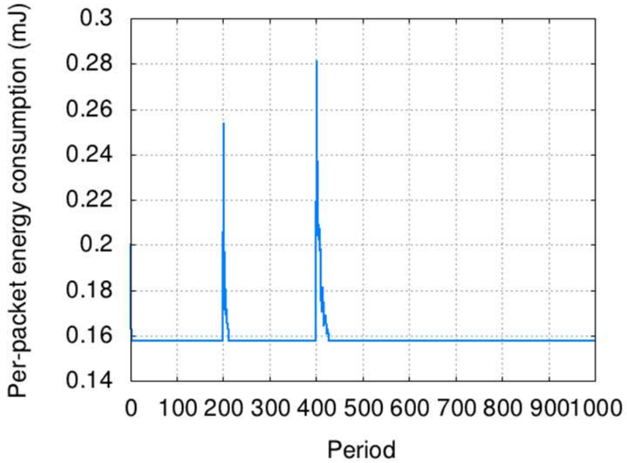
Analysis in Dynamic Conditions

PerLab









Summary

- De-synchronization in WSNs
  - Collision-free transmission schedule
  - Without requiring a strict synchronization
- Fully localized De-synchronization Algorithm
  - Only relies on local measurements
  - Robust against packet losses
  - Energy efficient
- Simulation analysis
  - AsAP is scalable and provides performance very similar to that of an ideal TDMA scheme.

Further Study

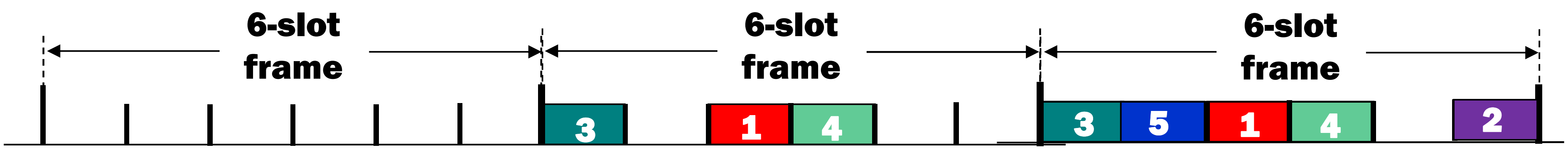
PerLab

- Convergence analysis
  - Derive analytically the time taken to get a complete de-synchronization
- Comparison with other de-synchronization schemes
  - in terms of convergence time, energy spent to converge, robustness to losses, …
- Extension to multi-hop topologies

Current Activity

Convergence Analysis

- Assumptions
  - Time is slotted
  - N slots available (N is also the # of sensor nodes)
- Distributed Slot allocation
  - Based on local information only



- Question
  - How much time it takes to achieve a complete slot allocation?

Localized Slot Allocation

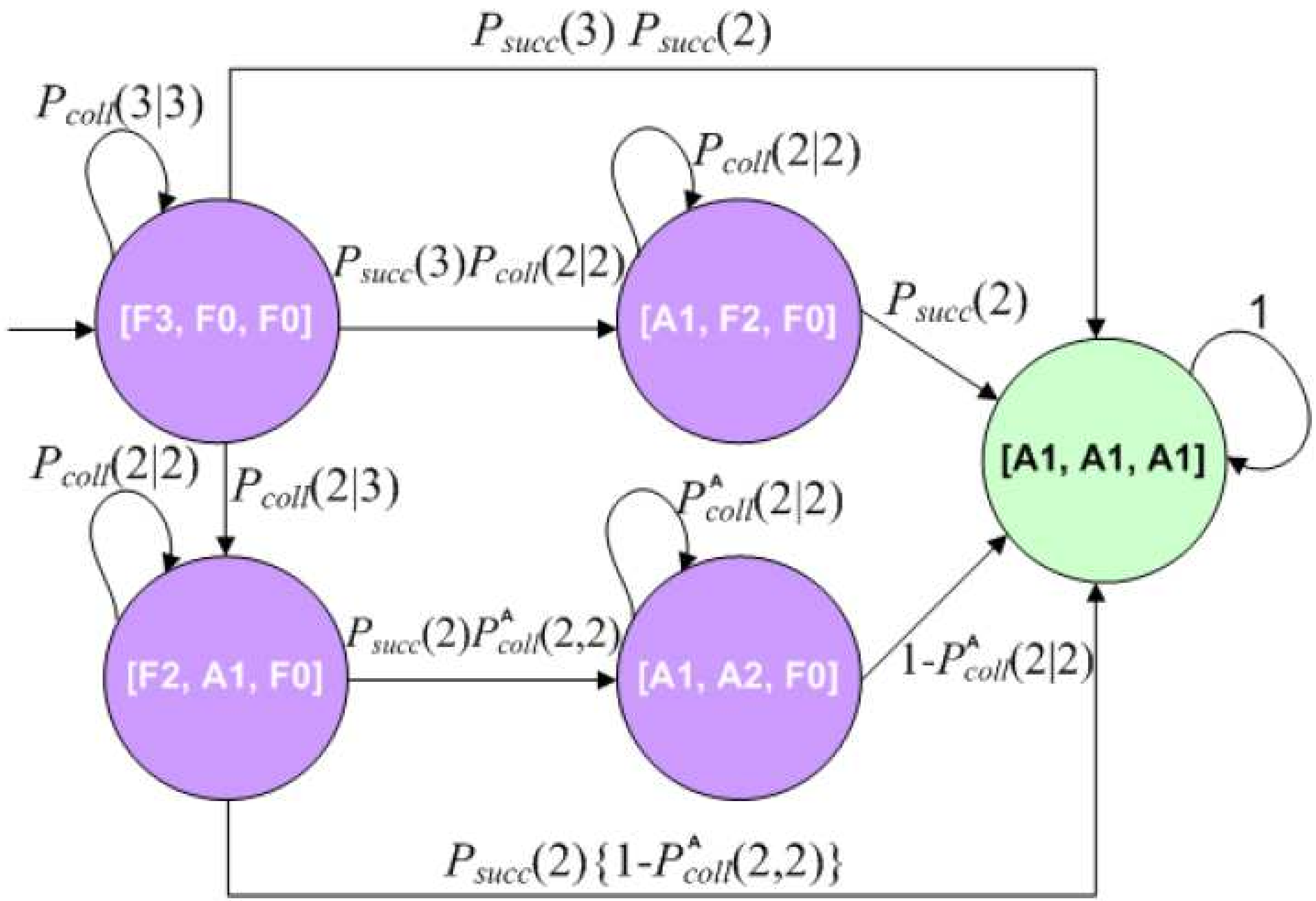
- Based on contention
  - Each node transmit a (fake) packet on a slot
  - And wait for the related success/failure notification
    - ⇨ Success: the slot is acquired
    - ⇨ Failure
      - – Collision
      - – Busy slot

Localized Slot Allocation Algorithm

Algorithm 1: LOCALL Algorithm

1 Choose a slot σ in $[1, N_s]$ randomly;
2 Try slot σ (using a random backoff B); // contention slot
3 **Wait** (Notification);
4 **Switch** (Notification);
5 **Case** SUCCESS:
6 Use slot σ in all subsequent periods with backoff B=0; //data slot
7 **Case** CHANNEL-BUSY:
8 Re-try slot $(\sigma + 1) \bmod N_s$ (with random backoff B);
9 **Case**: COLLISION
10 Re-try slot $\sigma + 1$ in the current period (with random backoff B) with probability p_r
11 Defer contention to slot σ in the next period (with random backoff B) with probability (1- p_r);

Markov Chain

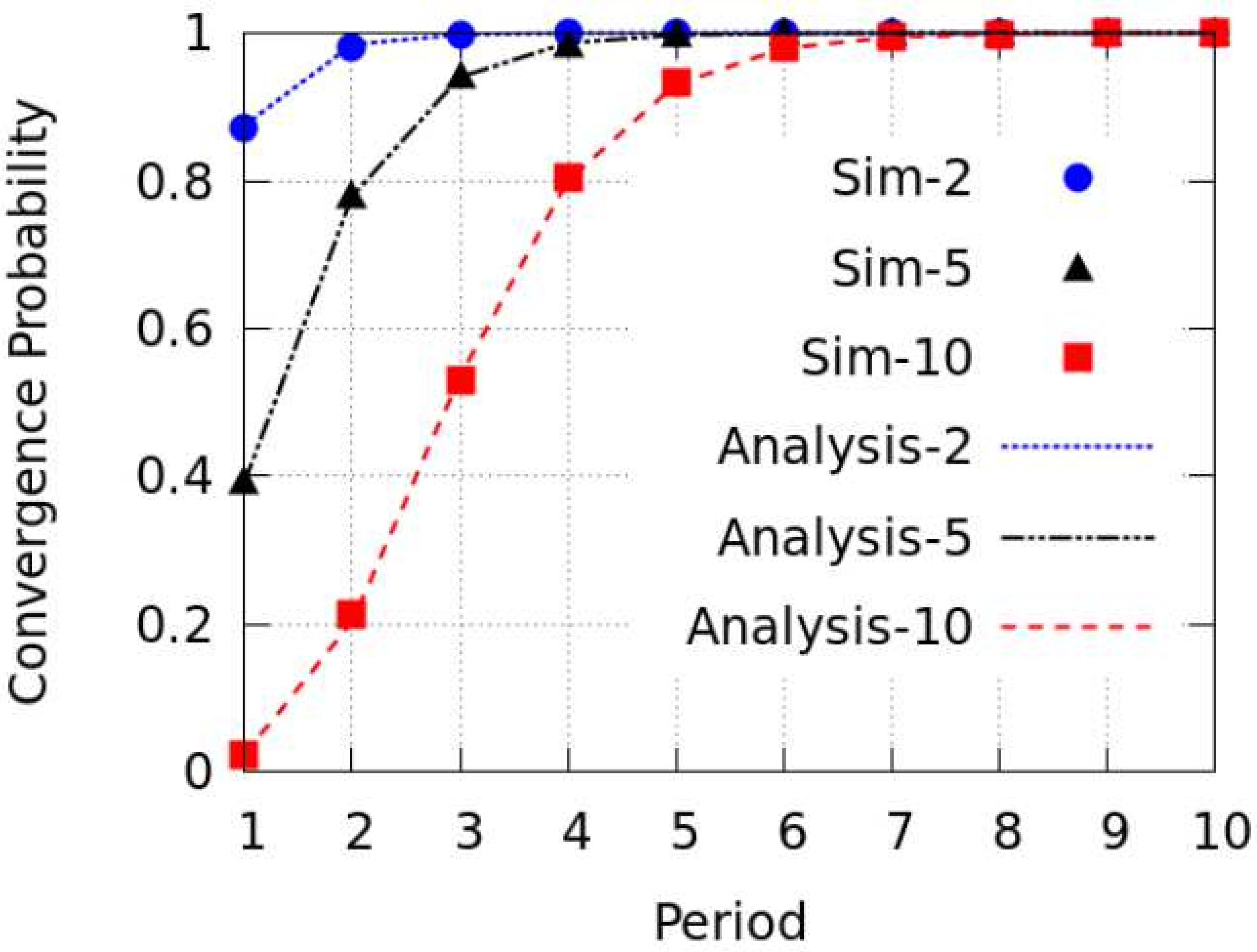


Parameters

PerLab

Parameter	Value
N, N_s	10
Bit Rate	250 Kbps
Data Frame (Payload) Size	127(118) bytes
ACK Frame Size	11 bytes
N_B (slot acquistion phase)	8
Power Consumption in RX mode (P_{rx})	35.46 mW
Power Consumption in TX mode (P_{tx})	31.32 mW
Power Consumption in Idle mode (P_{idle})	0 mW

Results



Convergence Probability
1
0.8
0.6
0.4
0.2
0
1
2
3
4
5
6
7
8
9
10
Period
Sim-2
Sim-5
Sim-10
Analysis-2
Analysis-5
Analysis-10

Comparison with CDM

PerLab

- Collision Detection with Memory
  - based on a lightweight vertex-coloring approach
- Initially, all sensor nodes are in *search* mode.
- At each round (period) *p*, a generic node *v*
  - (i) picks up randomly a color from the set of colors
  - (ii) checks whether it has a conflict with any other node
    - ⇨ If there is no conflict, node *v* enters *permanent* mode, selects *c* as its permanent color, and stops.
    - ⇨ Otherwise, it waits for a new round and performs the same actions again.
- The algorithm ends when *all* sensor nodes are in permanent mode.

Results

PerLab

TABLE III. 95% CONVERGENCE TIME (# OF PERIODS).

# of nodes	LocAll Simulation	CDM Simulation
2	2.00 (±0.00)	4.8 (±0.34)
5	3.80 (±0.43)	16.3 (±0.77)
10	5.10 (±0.32)	34.3 (±1.59)
20	8.00 (±0.41)	71.1 (±2.53)
30	10.50 (±0.54)	113.1 (±5.92)
40	12.70 (±0.50)	150.4 (±7.51)
50	14.80 (±0.43)	178.1(±9.63)

Results

PerLab

TABLE IV. AVG. ENERGY SPENT FOR SLOT SCHEDULING (mJ)

# of nodes	Model	Simulation without Rand.	Simulation with Rand.
2	0.38	0.38 (±0.01)	0.38 (±0.00)
5	1.21	1.21 (±0.01)	1.02 (±0.01)
10	3.32	3.32 (±0.03)	2.28 (±0.02)

Next Activity

- Extension to multi-hop topologies
- Comparison between
  - Ideal localized slot allocation
  - Asynchronous de-synchronization

PerLab

Thank you!

Questions