Energy efficient MAC protocols for Wireless Sensor Networks

Andri Toggenburger
tandri@student.ethz.ch

14.12.2004
Overview

- Introduction to Sensor Networks
- "A Distributed TDMA Slot Assignment Algorithm for Wireless Sensor Networks"
  - Goals
  - Model
  - Algorithm
- Conclusion
Wireless Sensor Networks: The Beginnings

- DSN (Distributes Sensor Network) project of DARPA, 1985
- Acoustic sensors
- Includes quiet diesel generator (power for days)
- 4 computers to process data (256 KB RAM)
- No dynamic topology
Wireless Sensor Networks: Today

- Intel Mote prototype
- Includes antenna for Bluetooth
- 12 Mhz processor
- Operation time: up to several months with AA Batteries (larger than sensor itself)
- Ad-hoc networking
Wireless Sensor Networks: Future?

- „Smart Dust“
- Size: like a grain of sand
- Solar powered
- Ad-hoc, P2P
- 1000's of nodes
- Price: below 1$
Wireless Sensor Networks: Technical Problems

- Limited hardware resources
  - Computing power
  - Memory
  - Communication power
  - Power supply
- Ad-hoc networking: node failure, dynamic topology, Media Access Control (MAC)
A Distributed TDMA Slot Assignment Algorithm for Wireless Sensor Networks

Ted Hermann and Sébastien Tixeuil

ALGOSENSORS 2004
**Goal**

MAC protocol which has the following properties:

- Distributed computation
- Self-stabilizing
- Expected local convergence in time $O(1)$
- Fairness among nodes
- Energy conservation
Why TDMA?

- Fairness
- No collisions
- Scheduled slots
  - Nodes can turn off their power

But recent work shows:
- TDMA may not improve bandwidth compared to other MAC protocols.
Graph Coloring and TDMA slot assignment

Distance-two coloring:
- No nodes within distance two have the same color
- Can be used to assign time slots
- Different solutions possible:

![Graph Coloring Diagram](image)
Graph Coloring and TDMA slot assignment

Distance-one coloring:
- does not work because of the „Hidden Terminal Problem“

![Graph and Range of node](image)
Wireless Network Model

- Synchronized clocks
- All nodes use the same frequency
- Node density is upper-bounded

- Infinite repetition of the algorithm at each node
- Shared variables among nodes, updated by messages.
- CSMA / CA slot for reservation of TDMA slots
Wireless Network Model

Neighborhood of a node $p$ at distance $i$:

$$N^i_p \quad |N_p| \leq \varrho$$
Illustration of a schedule

Final result schedule should look like this:
5 Algorithms to accomplish TDMA

1. Neighborhood identification
2. Neighborhood-unique naming
3. Leaders via maximal independent set
4. Leader assigned minimal coloring
5. Assignments of time slots from colors
Neighborhood identification

Goal: Learning of $\frac{N^2}{p}$ and $\frac{N^3}{p}$

- Shared List $L$ with pairs $(a:A)$, where $a$ is an id and $A$ is a list of id's. ($A = \text{list of nodes known by } a$)
- List $L_p$: $L$ augmented by an age value for each element
- $MaxAge$: maximum age of a list entry
Neighborhood identification

- N0: receive $m_N(a,A) \Rightarrow \text{update}(L_p, a: A \setminus \{p\})$
- N1: drop old entries in $L_p$
- N2: send $m_N(p, \text{neighbors}(L_p))$
Neighborhood identification

- A simple example:
Neighborhood identification

After round 1:

[6:X:0|x:X:1]  (6:X)  [5:X:0|7:X:0]

5  6  7

(5:X)  (7:X)  (6:X)
Neighborhood identification

- After round 2:

\[
\begin{align*}
\text{[6:[7]:0|x:X:2]} & \quad \text{[5:X:1][7:X:1]} \\
5 & \quad 6 & \quad 7 \\
\text{(6,[5,7])} & \quad \text{(6,[5,7])} \\
\text{[6:[5]:0|x:X:2]} & \\
\end{align*}
\]
Goal: Each node in $\frac{N^3}{p}$ has unique id

- Smaller and constant name space (as compared to physical addresses)
- Provides (no good) solution for graph coloring
Neighborhood-unique naming

- Namespace: $\Delta = \varrho^t, t > 3$

- Every node stores set of latest known ids of all neighbors ($Q^3$ entries)

- Node keeps id if no other node has the same id

- Node changes id to random id if other node has same id
Neighborhood-unique naming

A simple example, \[ \Delta = 16, \varrho = 2, t = 4 \]
Neighborhood-unique naming

- After round 1:

1, [(6,2)]

2, [-]

5 -> 6

7

1, [(6,2)]
Neighborhood-unique naming

- After round 2:

1, [(6,2)]

2, [(5,1),(7,1)]

5 ➔ 6 ➔ 7

(5,1) ➔ (7,1) ➔ (5,1)
Neighborhood-unique naming

- After round 3:

9, [(6,2),(7,1)]

2, [(5,1),(7,1)]

5

6

(7,1)

1, [(6,2)]

7
Neighborhood-unique naming

- And so on...

- Once the name of a node is established (=unique) in all 3-neighborhoods a node is part of, it stays fixed!

- Algorithm self-stabilizes with probability 1 and has constant expected local convergence time
  - What about propagation of name changes through the whole network?
Leaders via Maximal Independent Set

- Simple distance-two coloring algorithms use too many colors, so leaders dictate color of nearby nodes.

- A Maximal Independent Set of the graph gives the leaders.

- An independent set $I$ of a graph $G$ is a set of nodes such that no two nodes in $I$ are neighbors.
Leaders via Maximal Independent Set

Algorithm for a node p:

(Flag $L_p$: leader flag of node p)

- Set $L_p$ to true if all neighbors have larger id.
- If there is a neighbor which is leader and has smaller id, set $L_p$ to false.
- If all neighbors with smaller id have their leader flag cleared, set $L_p$ to true.

Solution converges to a maximal independent set.
Leaders via Maximal Independent Set

An example graph with marked leaders
Leader assigned coloring

- Leaders assign colors to their neighbors and themselves.
- Each leader has list of preferred colors for each node in its neighborhood (shared variable).
- A node chooses a color in the cached color list of the leader with the smallest id in its neighborhood.
- Leader chooses color for a node from colors which haven’t already been assigned by leaders with smaller id’s somewhere in 2-neighborhood.
  - every non-leader stores colors of its neighbors and leader id which assigned them in a shared variable
Leader assigned coloring

Example

Arrows: assignment of color from minimum leader to a neighbor
Leader assigned coloring

Example Coloring which needs 5 colors from c0 to c4
Assigning Time Slots from Colors

- Actual number of colors used is not available in a global variable

- Each node should have about as much bandwidth as any other node in 2-neighborhood → fairness

- Allocate slots to nodes beginning with the most constrained to the least constrained in order not to waste bandwidth
Assigning Time Slots from Colors

2-neighborhood:

Max. Bandwidth:

Assignment in correct order:

Assignment in wrong order:
Assigning Time Slots from Colors

Algorithm (iterative)

- Count number of colors in 2-neighborhood and store it in "base" (gives upper bound on available time)
- Learn about time intervals chosen by nodes in 2-neighborhood which have larger "base"
- Choose as much time intervals which haven't already been assigned to reach maximum "1/base"
Conclusion

- 5 algorithms running in combination
- Probabilistically self-stabilizing solution to the problem
- \(O(1)\) local convergence time for every algorithm and consequently whole process in expectation
- Global convergence time? Suspected to be sublinear
Conclusion

- Relies on synchronized clocks

- How to decide on the length of the CSMA / CA slot?
  - And: percentual size of the CSMA / CA slot?

- Algorithm assumes bidirectional communication

- Simulation would be nice:
  - Length of CSMA / CA slot could be ascerted.
  - No statements about actual amount of energy which is saved
Thank you for your attention

Please feel free to ask question!