

Distributed Data Aggregation Scheduling in Wireless Sensor Networks

Bo Yu, Jianzhong Li

**School of Computer Science and
Technology,**

Harbin Institute of Technology, China

{bo_yu, lijzh}@hit.edu.cn

Yingshu Li

**Department of Computer Science,
Georgia State University, US**

yli@cs.gsu.edu

April 23, 2009

Outline

- Introduction
 - Sensor networks
 - Data aggregation
 - Our contributions
- Related Work
- Problem Definition
- Our Algorithms
 - DAS
 - Theoretical Results
 - Adaptive DAS
- Experiments
- Conclusions

Introduction

- Sensor networks
 - environmental monitoring, medical care, battlefield surveillance, ...
- Data aggregation
 - essential operation
 - many approaches proposed
 - collision problem left to the MAC layer, not efficient
- We are focusing on:
 - **data aggregation scheduling**, aiming at minimizing **latency**

Our Contributions

- We propose a two-phase distributed algorithm DAS for aggregation scheduling for the first time
- We present the latency bound of our algorithm
 - $24D+6\Delta+16$
 - Nearly constant approximation
- We design an adaptive strategy for DAS
 - Suitable for dynamic topologies

Related Work

- Data aggregation scheduling (aiming at latency)
 - X. Chen, et.al. MSN 05
 - Minimum aggregation time problem is NP-hard.
 - An approximate algorithm with latency bound of $(\Delta-1)R$
 - S. Huang, et.al. Infocom 07 (mistakes in it)
 - An approximate algorithm with latency bound of $23R+\Delta-18$
- Existing problems:
 - Latencies are still very high: the best latency is $(\Delta-1)R$
 - Algorithms are **centralized** (inefficient when topology changes)
- Our Solution
 - DAS
 - The first distributed aggregation scheduling algorithm aiming at minimizing latency in sensor networks

Problem Definition

- Distributed Aggregation Scheduling
 - Preliminaries
 - Let A, B be subsets of V and $A \cap B = \Phi$.
 - We say data is aggregated from A to B in one time slot if all the nodes in A transmit data in this time slot simultaneously and all the data are received collision-free by some nodes in B .
 - Here A is called a sender set.

Problem Definition

■ Distributed Aggregation Scheduling

- Generate a data aggregation schedule distributedly, minimizing time latency ℓ :

- A data aggregation schedule is a sequence of sender sets S_1, S_2, \dots, S_l satisfying the following conditions:
 - 1) $S_i \cap S_j = \emptyset, \forall i \neq j$;
 - 2) $\bigcup_{i=1}^l S_i = V - \{r\}$;
 - 3) Data are aggregated from S_k to $V - \bigcup_{i=1}^k S_i$ at time slot k , for all $k = 1, 2, \dots, l$ and all the data are aggregated to the sink r in l time slots.

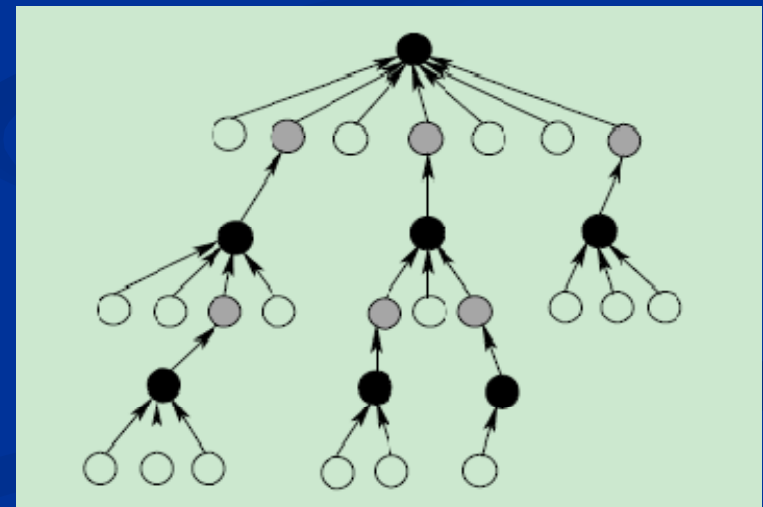
- Intuitively, the schedule for each node is a time slot in which it sends its data. All the nodes' schedules form the whole network's schedule.

DAS

- DAS
 - two phase algorithm
 - Distributed Aggregation Tree Construction
 - Distributed Aggregation Scheduling

Distributed Aggregation Tree Construction

- Distributed Aggregation Tree Construction
 - existing approach in [Wan et al. INFOCOM 02]
 - maintaining *good property of aggregation tree*
 - All black nodes form an MIS
 - The root is black
 - White nodes are leaves
 - Gray nodes' children are black
 - Black nodes' children are gray or white



DAS

- DAS
 - two phase algorithm
 - Distributed Aggregation Tree Construction
 - Distributed Aggregation Scheduling (SCHDL)

SCHDL

■ Observation on the collisions

- A node is called a competitor of u if it cannot send data while u is sending data due to the collision.
- The set of u 's competitors is called u 's **competitor set**.
- Competitor set of each node u
 - u 's parent's neighbors
 - u 's neighbors' children

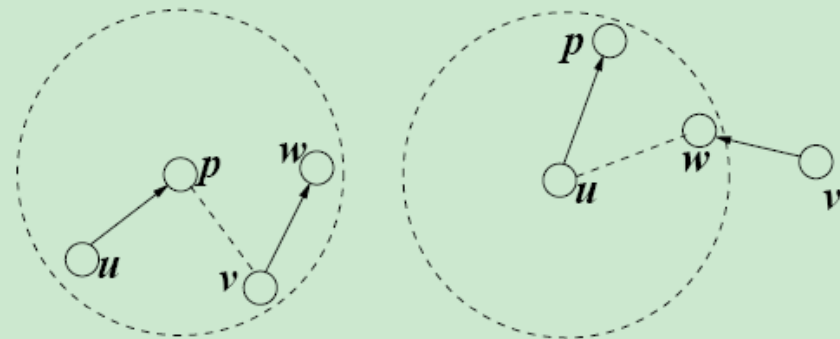


Fig. 2. Two cases of a collision

SCHDL

■ Basic Idea:

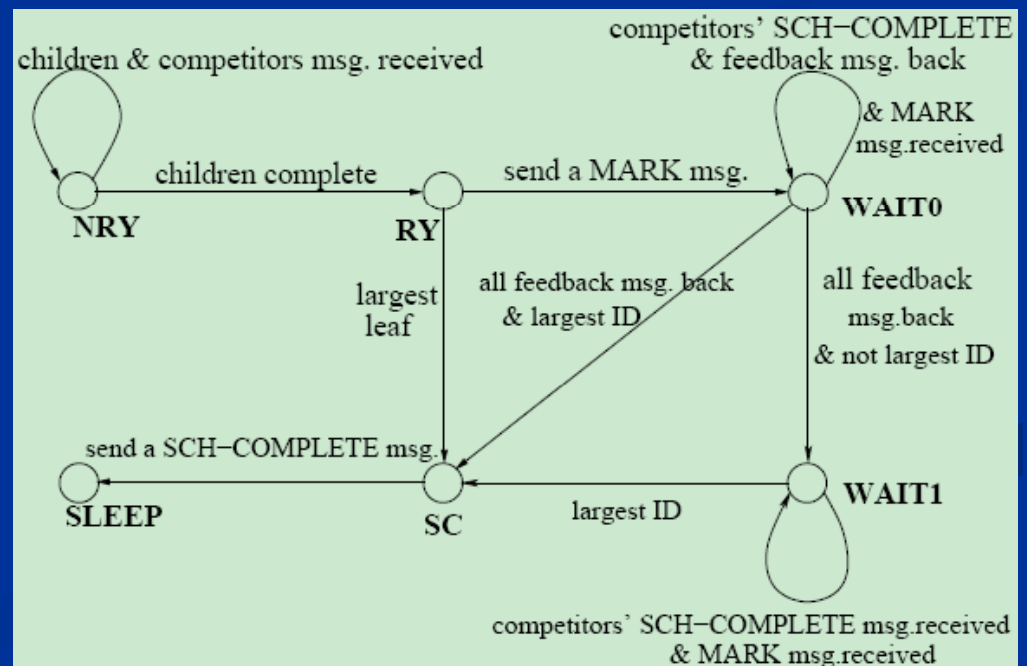
- All the sensor nodes make their schedules **in a certain order** according to their competitor sets

■ Order priority:

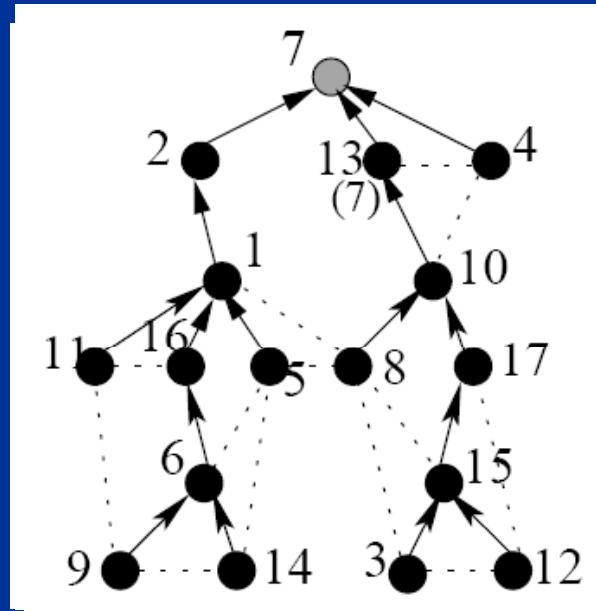
- **Node ID**
- Whether a node is ready
 - A node is ready if it is in RY state

■ Node State

- RY(ReadY)
- NRY(NotReadY)
- WAIT0
- WAIT1
- SLEEP
- SC(ScheduleComplete)



An Example



Theoretical Results

- Correctness

Theorem 2: (Correctness of DAS). DAS generates a collision-free aggregation schedule in finite time.

- Performance

Theorem 4: Our distributed scheduling algorithm has a time latency bound of $24D + 6\Delta + 16$, $O(n)$ time complexity and $O(n \cdot \max\{\Delta, \log n\})$ message complexity.

- This latency bound is loose

- we allow any network topology
- very complicated cases
- tight bound hard to be analyzed
- time latencies in the experiments are 7 to 8 times better than the theoretical bound

Adaptive DAS

- Adaptive DAS
 - Occasions when nodes join or fail
 - Two phase
 - Maintenance of Aggregation Tree
 - Two cases
 - Node joining
 - Node failure
 - Basic idea
 - Maintaining the property of current aggregation tree
 - Joining nodes find their parents
 - Failure nodes are deleted from the aggregation tree
 - Nodes that lose their parents find new ones
 - Adaptive Scheduling
 - All the nodes who change their parents form a set Upd and they are marked “renewed”
 - Any node who is an ancestor of a node in Upd is marked “renewed”
 - All the “renewed” nodes run SCHDL
 - New schedules generated

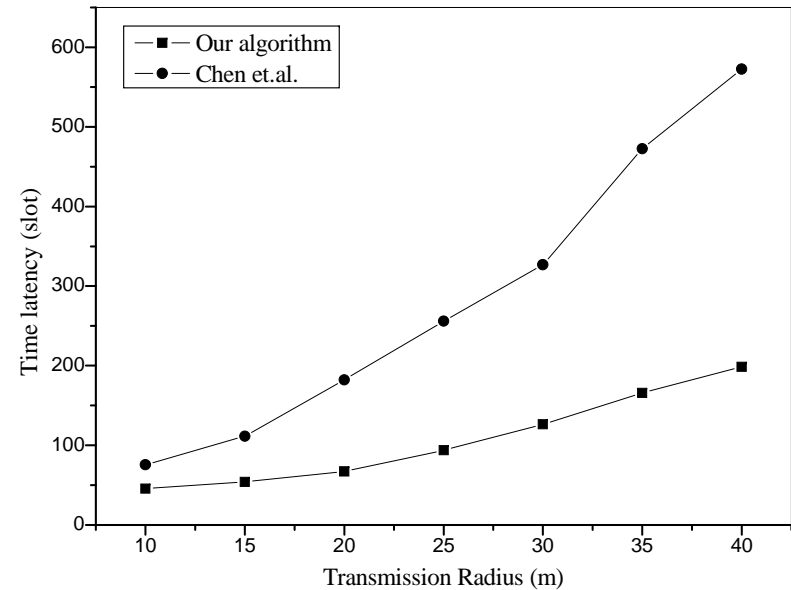
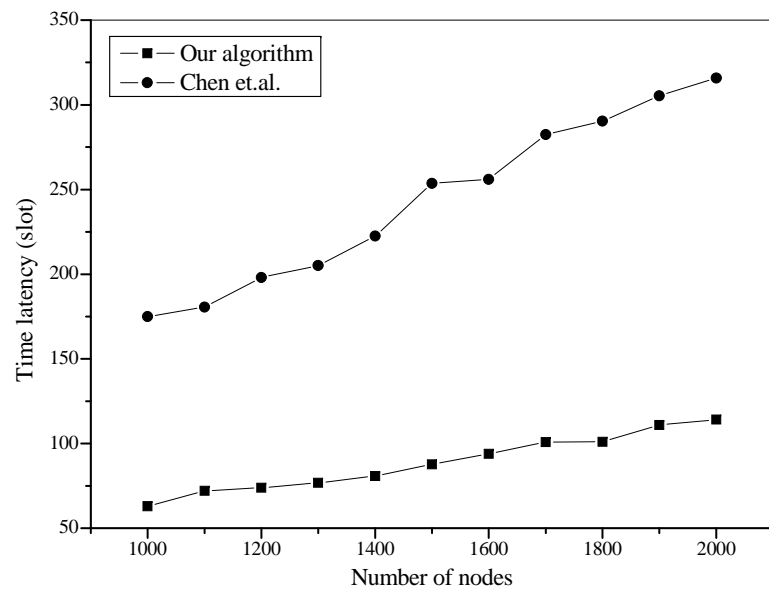
Experiments

■ Simulation

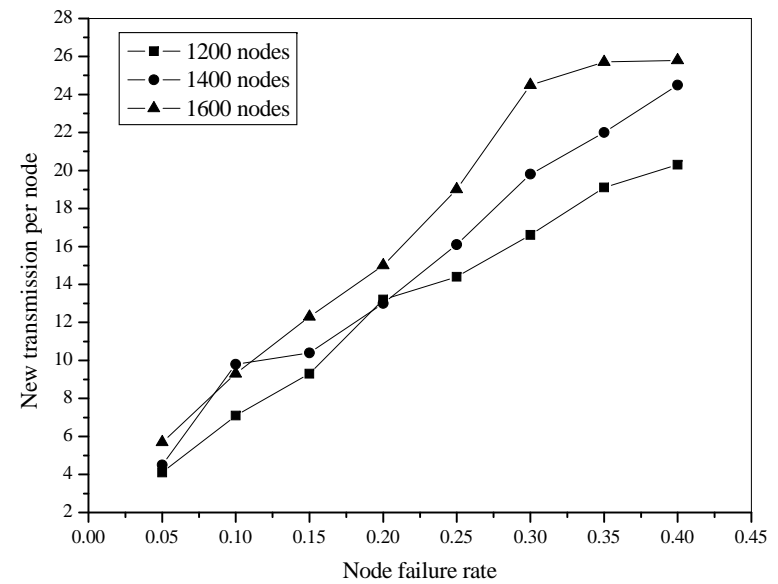
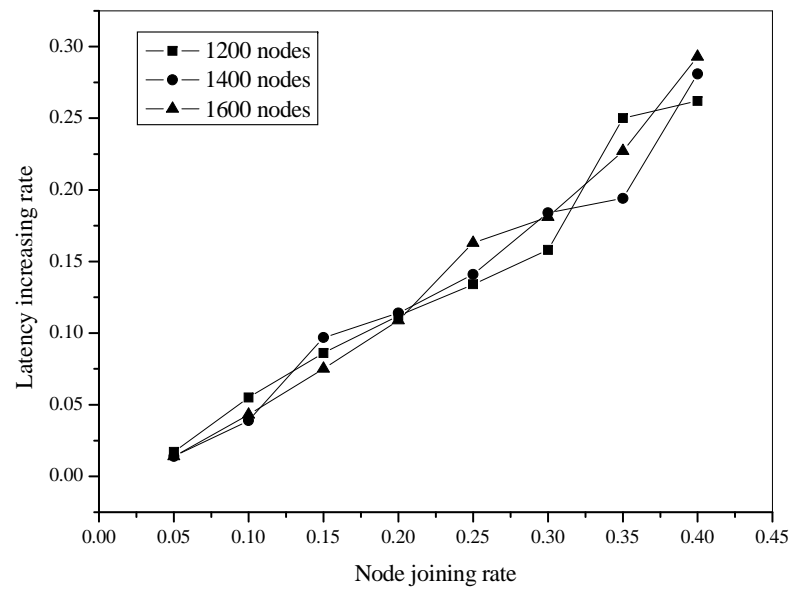
■ Methodology:

- A region of 200m*200m
- Same transmission radius for all the nodes
- Transmission radius varies from 20m to 40m
- # of nodes varies from 1000 to 2000

Experiments (cont.)



Experiments (cont.)



Conclusions

- Two problems of previous work on aggregation scheduling:
 - High latency bound: $(\Delta-1)R$
 - Centralized algorithm
- We present a **distributed** algorithm DAS with latency bound **$24D+6\Delta+16$**
- We present an adaptive strategy suitable for sensor networks with **dynamic topologies**

Thank you!

Questions?