PROSE: Providing Robustness in Systems of Embedded Sensors

Yanyong Zhang, Shengchao Yu, Jaewon Kang, Badri Nath, Wade Trappe

WINLAB, Rutgers University
Sensor Network Model

- A sensor net deployment has:
  - Many inexpensive sensor nodes
    - capable of sensing, computing, and communication
  - Resource constraints (energy)
  - Deployment redundancy
  - Stationary nodes
  - One or several data sinks
    - Connected with apps

- Sensor net coverage model:
  - Lifetime vs. coverage
    - a minimum set of active nodes
  - Grid-based coverage model
    - Network partitioned into grids
    - A node is able to monitor at least one grid point
    - Every grid point must be monitored

- Sensor net connectivity model
  - Multi-hop networking
  - More active nodes may be needed
Challenges in Sensor Networks

Many challenges in building sensor systems/applications (at WINLAB)
- Sensor deployment
- Lifetime maximization
- Programming model
- Routing and MAC protocols
- Localization algorithms
- Security and privacy

This project focuses on robust sensor services
- Unattended operations
- Hostile environment
  - Fire
  - Landfill
- Many unexpected events may happen
  - Node failure
  - Congestion
  - Erroneous data

How to extract a steady stream of valid data from all the interesting spots in the presence of unexpected interrupts?
PROSE Overview

PROSE has three components:

- **DADA**: a 2-Dimensional Adaptive Node Scheduling Framework
  - Provides data availability against random node failures
  - Repairs network coverage and connectivity by cleverly waking up redundant nodes upon node failures

- **TARA**: a Topology-Aware Resource Adaptation Framework
  - Provides data availability against congestion
  - Brings more sensor nodes “online” to accommodate higher traffic rate to eliminate congestion

- **MARA**: a Measurement Assurance and Robust Aggregation Framework
  - Provides data assurance
  - Classifies and cleanses sensed data

Each component has two units:

- Exception detection
- Exception handling
DADA: Overview

- DADA balances the tradeoff between network lifetime and network quality
- DADA achieves two goals:
  - Bounded recovery time upon node failures (recovery time \( \leq \delta \))
  - Minimize energy consumption
- Basic idea for DADA
  - A minimum set of active nodes
  - The redundant node wakes up every \( \delta \) time
  - When it wakes up, it finds out whether it needs to become active
  - If the active node dies at time \( t \), its redundant node will wake up at latest \( t + \delta \) to, and will become active
- Complications
  - How to decide an active node’s redundant nodes?
  - How to schedule multiple redundant nodes?
DADA: Gangs

- A node cannot always be replaced by one redundant node
  - On average, 3-5 nodes are needed to replace a node’s sensing area
  - One node may not be enough to repair the communication hole even if it is in the radio range

- The concept of “gangs”
  - A gang consists of a group of nodes that can completely replace an active node
  - A minimum gang is a gang itself, but none of its subsets is a gang
  - All the nodes belong to a minimum gang should wake up together
  - The minimum gangs are analogous to “sentries” in real life.
DADA: P-Sentry Algorithm

- Persistent Sentry (P-Sentry) provides interrupt-less network operations
  - One minimum gang stays awake all the time (the sentry)
  - The sentries can take over when the active node fails
  - The other redundant nodes can sleep for a much longer time

- Issues with P-Sentry algorithm
  - How do the sentries detect the failure of the active node?
    - Passive listening
    - Active probing
  - Which nodes should be chosen as sentries?
    - Energy
    - Functionality
    - Redundancy
  - How long should the non-sentry redundant nodes sleep?
    - Estimate the remaining lifetime
    - Considering the random failure rate
  - How to synchronize multiple schedules a redundant node may have?
    - Active node maintains states
Rotary sentries (R-Sentry) limits service loss
- All the minimum gangs take turns to wake up, with $\delta$ between two subsequent wakeups.

An example scenario
- A’s minimum gangs $\{H, I, C\}$, $\{H, I, G, E\}$, $\{B, D, C\}$, $\{B, E, G\}$, $\{E, F, G\}$, and $\{H, D, C\}$

Issues to be considered
- How to detect failures?
  - Probing for sensing ranges
  - Listening for HELP messages
- How to synchronize multiple schedules a redundant node may have?
- How to dynamically adapt the schedule when the redundant node fails?
TARA: Overview I

- A monitoring sensor network alternates between dormant periods and crisis periods
  - During dormant periods, minimum resources are kept online (DADA)
  - A hot spot will form during crisis periods

- Three types of hot spots:
  - Source hot spot
  - Sink hot spot
  - intersection hot spot
Traffic control vs. resource control

- Traffic control schemes throttle source traffic rates to eliminate hot spots
- Resource control is preferred in sensor networks
  - Data packets during crisis states can NOT be dropped
    - Fidelity requirement
  - Sensor nets are deployed for peak load
  - Bringing resources online is realistic
    - e.g. sensor nodes

Resource control schemes

- Forming alternative routing topology
- Multiple-path routing

Topology-aware resource control

- e.g. dose the new topology provide sufficient capacity?

TARA has two main components:

- Capacity estimation tool
- Resource adaptation algorithm
TARA: Capacity Analysis

- **Goal:** to estimate the maximum end-to-end throughput of a given topology
  - The maximum rate at which the source can send packets towards the sink
- **Idea:** mapping this problem to a graph coloring problem
  - Due to link interference, the end-to-end throughput is a fraction of the 1-hop throughput (capacity fraction)
  - Suppose under an optimal schedule, the sink receives a packet every n time frames, then the capacity fraction is 1/n
  - The throughput is (1-hop throughput * capacity fraction)

Assumption: only adjacent nodes can hear/interfere each other
TARA: Capacity Analysis

- **Idea**: mapping this problem to a graph coloring problem
  - Suppose under an optimal schedule, the sink receives a packet every $n$ time frames, then the capacity fraction is $1/n$
  - To estimate $n$
    - Spatial interference graph
      - Vertexes are wireless links
      - Edges mean that two links are within each other's interference ranges
    - $N$ is equal to the number of colors assigned to the spatial interference graph

- **Graph coloring problem**
  - Theorems can provide an upper bound
  - Heuristic approaches provide tighter estimates
TARA: Resource Adaptation Scheme

- TARA has the following steps:
  - Congestion detection
    - e.g. channel loading, packet drop ratio, queue occupancy, etc.
  - Traffic distributor
    - Each node maintains the incoming traffic rate from each neighbor
TARA: Resource Adaptation Scheme

- TARA has the following steps:
  - Congestion detection
  - Traffic distributor
  - Traffic merger should
    - Have a low congestion level
    - Reside on the routing path to the intended sink
    - Result in sufficient capacity
      - Based on the observations gained using the capacity analytical tool
TARA: Resource Adaptation Scheme

- TARA has the following steps:
  - Congestion detection
  - Traffic distributor
  - Traffic merger
  - Build a detour path

- The merger initiates the process with flooding a REQ message including the TTL
- A node that receives the REQ message decrements the TTL, and forwards the message with the following optimizations:
  - A node drops REQ if its congestion level is high
  - A node drops REQ if it is on the original routing path
  - A node only forwards the REQ with a higher TTL
- Multiple REQ messages will reach the distributor, and it chooses the path with highest TTL
TARA: Resource Adaptation Scheme

- TARA has the following steps:
  - Congestion detection
  - Traffic distributor
  - Traffic merger
  - Build a detour path

- The merger initiates the process by flooding a REQ message including a TTL field
- A node that receives the REQ message decrements the TTL, and forwards the message with the following optimizations:
  - A node drops REQ if its congestion level is high
  - A node drops REQ if it is on the original routing path
  - A node only forwards the REQ with a higher TTL
- Multiple REQ messages will reach the distributor, and it chooses the path with highest TTL
TARA: Resource Adaptation Scheme

- TARA has the following steps:
  - Congestion detection
  - Traffic distributor
  - Traffic merger
  - Build a detour path
  - The distributor distributes the traffic between two paths
    - Traffic allocation is topology-dependent
TARA: A few issues ...

- TARA also reduces the online resource when traffic decreases
- A three-tiered resource controlling scheme
  - For very short-term congestion
    - Larger buffer size
    - Prioritizing packets
    - Storing data locally
  - For short-term congestion
    - TARA
  - For longer-term congestion
    - Traffic control
MARA

- MARA provides data assurance against sensing errors
- Data classification mechanisms
  - Constraint-based Consistency Checks
    - Predefined constraints such as “temperature between 0 and 100”
  - Redundancy Consistency Checks
    - Multiple sensors monitoring the same variable
  - Multi-modal Consistency Checks
    - Multiple physical properties may exhibit correlation
- Data cleansing mechanisms
  - Robust statistical tool or robust aggregation tool
  - Challenge is to make them suitable for sensor platform
Conclusion

- Making the sensor system robust should not be an afterthought
- If you need more details on
  - DADA ➔ Shengchao Yu’s poster
  - TARA ➔ Jaewon Kang’s talk in the afternoon
  - MARA ➔ Badri Nath’s talk in the afternoon

Questions ???