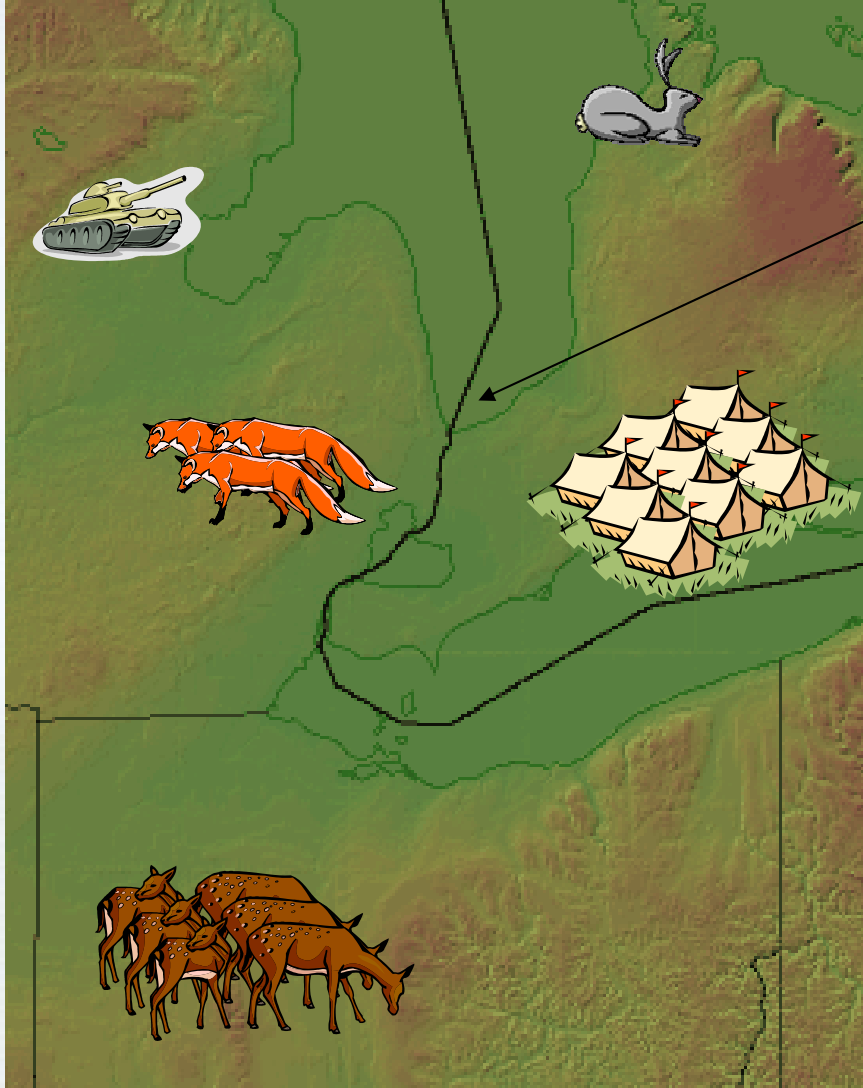


Mission-CRISIS

Mobile Computing Project Presentation

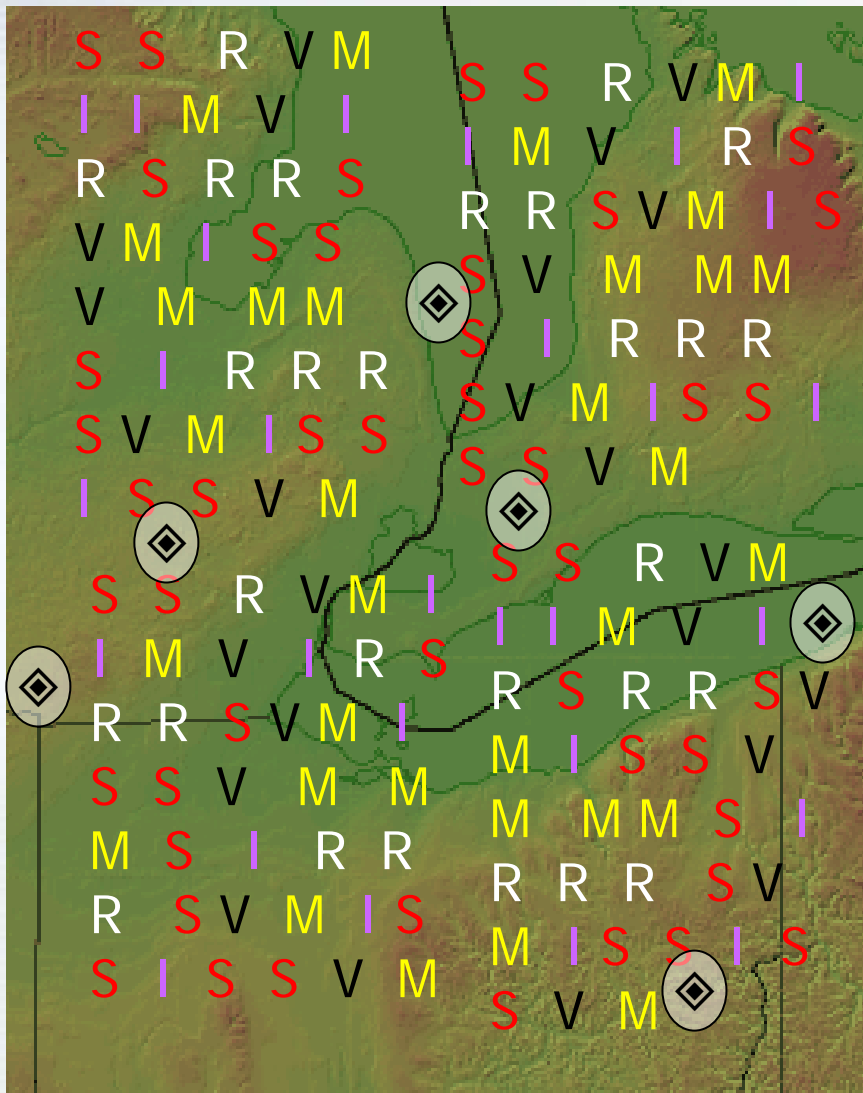
**Old Dominion University
CS895 Spring 2006
Joan A. Smith & Martin Klein
5 May 2006**

Sensor Information System (WSN)



- Imagine a remote area in hostile territory where we have dropped a collection of sensors of various types
- The dark line is an unpaved road thru the area
- The terrain is unpopulated except for the usual fauna (deer, rabbits, foxes, etc.)
- We are monitoring for secret enemy encampments
- But we are also interested in possible military activities in general

Sensor Deployment Map



- Sensors are dispersed at random in the area
- Various types are deployed
 - Sound
 - IR (infra-red)
 - Vibration
 - Magnetic
 - Chemical, etc.
- Sinks are also included among the sensors (◇)
 - Also random distribution
 - Fewer deployed
 - But with greater capabilities

Goal: Identify Friend from Foe (background)

- **IFF has always been central to deployment situations**
- **Current IFF technology is “active” type**
 - **Transponders affixed to planes, troops, etc**
 - **Special handshakes, keys, frequencies, etc**
 - **Deployed as part of military component**
 - **Not existing outside of active troops**
 - **Expensive, heavy**
 - **Has been successfully infiltrated in some situations (Afghanistan, e.g.)**

Goal: Identify Friend from Foe

- **IFF ID is difficult**
 - What about pack of foxes vs trained dogs with explosives?
 - What about hostile activities masquerading as normal activities (the local hoe-down)
 - But we often know what is expected in locale X
 - Therefore, we can create a rule set that says activity Y is outside that normal expectation

Goal: Identify Friend from Foe

- **What if the WSN “knew”??**
- **Passive solution with active option**
 - **Hidden from view**
 - avoid enemy notice
 - **Part of general WSN**
 - collect regular data, too
 - **Want to distinguish mission-critical (i.e., urgent, NOW) info from general info**
 - Have to still collect regular data (pull)
 - Have to be able to initiate emergency data push

Project Discussion – Problem Area

- **Problem:**
 - **How can sensor 'recognize' abnormal data?**
 - Must be more than a single anomaly
 - Must have 'normal' reference data
 - **How can data be corroborated?**
 - Nearest-neighbor confirm/deny
 - Count all and pass to each neighbor
 - **How can data be sent before collection time?**
 - Emergency sensor-sink could aggregate
 - Satellite transmission for emergency use
- **Solution: IFF for WSN**
 - Method to provide critical report to HQ from the sensor network

Project Proposal: Solution Discussion

CRISIS:

Critical Reporting In Sensor Information Systems

- Create baseline data set of “normal” sensor input
- Predefine the neighborhood:
 - Who to talk to
 - Priority order of conversation
- Define rules for responding to “abnormal” information
 - Who to ask for corroboration
 - Collecting votes
 - Forwarding to sink
- Enable “supervised learning”
 - Avoid the “cry wolf” problem
 - Handle “flaky” sensors without burning out whole network

What we do not address

- **Dense, stationary “mesh” network**
 - Passively queried at predetermined times
- **2 types of master sensors: parents and sinks**
 - Sinks capable of initiating regular and urgent transmissions
- **X types of worker sensors: radioactivity, vibration, sound, temperature, humidity, magnetic, chemical – i.e., appropriate types, not homogeneous**
- **Robust (like the Volcano sensors; see Bib)**
- **Self-organizing (UIDs – on-going research)**
- **multi-route capable (adaptive dissemination of data)**
- **controlled congestion**
- **Secure (via LEAP or OTMK, for example)**
- **Capable of sensing, listening, responding and transmitting**
 - Transmit (forward) any type data (multi-lingual, so to speak)
 - Update local lookup table (an aspect of learning & control)
 - Respond to like-type data queries (to confirm sighting)

Goals

- 1. Create method to ID urgent information**
- 2. Confirm the identification and the urgency**
- 3. Push the data to HQ**
- 4. Enable supervised learning by sensors**
 - Weed out flaky sensors**
 - Update urgency parameters if necessary**

1. Identify CRISIS data

- Defined as data needing **urgent** transmission to HQ
 - Cannot wait for regular data transfer time
 - Worth risking network depletion
- Specific to type of sensor deployed
 - Radioactivity sensor parameters differ from vibration sensor parameters, for example
 - Assumes urgent parameters are knowable for each sensor type
- Parent/Sink sensors aggregate, evaluate, forward, teach
- Sensors keep “quality” rating of self and neighbors
 - Higher quality = higher reliability
 - Determined by ratio of Quality to Threshold (these change by events and/or via an update from parent)

2. Confirm Urgency (Local Neighborhood Level)

- Compare ratio of quality to threshold
If above threshold then:
- Ask neighbors (*any* type) if urgent data is detected
- Respond to queries about data from neighbors
- Forward data to parent/sink

3. Push Data to HQ (Master Sink Level)

- Master sink triggered awake by nearest parent with urgent data**
- Aggregates information**
- Accepts and transmits (or rejects/waits and does not transmit)**
- Replies to parent(s) with quality and/or threshold updates as needed**

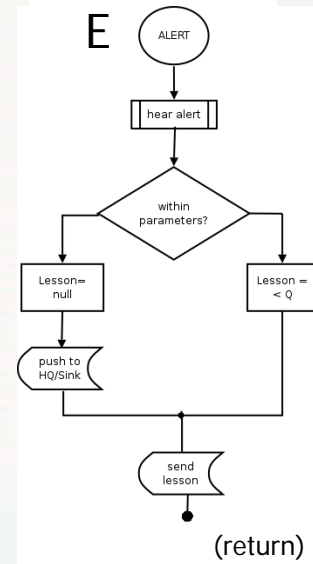
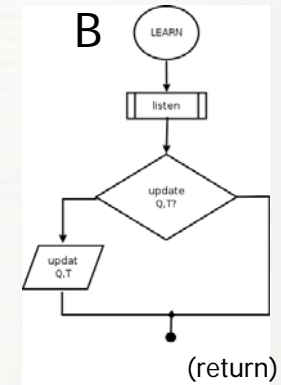
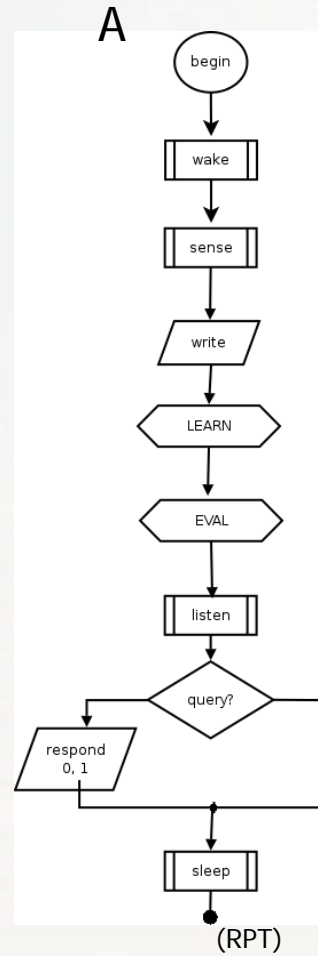
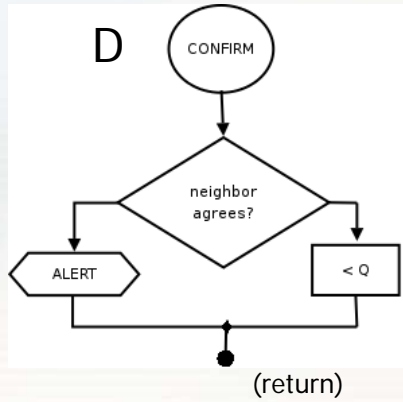
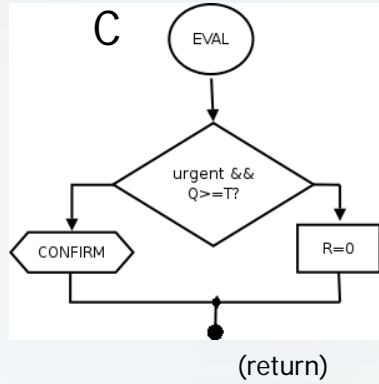
4. Enable Supervised Learning

- **Update quality setting and threshold**
 - Lower ratio if urgent is not confirmed
 - Change ratio on direction from parent
- **Local sensor evaluates its own data quality**
 - If below threshold, does not initiate urgent notification sequence
 - If no neighbors confirm urgency, lowers its own quality
- **Parent or sink can initiate change in one or both parameters to tweak performance**
- **Flaky nodes will die first due to extra energy expended waiting for other nodes' broadcasts**
- **Flaky nodes will disrupt less and less as their quality rating does down**

Algorithms

- **ID Abnormal activity**
- **Corroborate (Build consensus)**
- **Push notice to sink(s)**
- **Learn from bad behavior**
 - **Is this one a question of rating the node?**
 - **What if each node has own & neighbors' "quality" rating?**
 - **Gets quality from parent**
 - **Tracks confirms/denials from neighbors**
 - **Alternative: change the "abnormal" responses (not so good a solution)**

CRISIS Algorithm



Building the prototype

- **Wrote simulator from scratch**
 - Targeting newer sensors for which no simulator exists yet
 - Focus for project was restricted to IFF-related features
- **Features implemented:**
 - **3 types of sensors with threshold parameters**
 - Random deployment
 - Conflicting placement resolution (can't pile sensors on each other)
 - **Configurable network density**
 - #nodes, #sinks, size of matrix
 - **Configurable node characteristics**
 - Sensing and radio ranges
 - Threshold parameters
 - **Bootstrap process sets up route table & paths**
 - **Confirmation of critical data status**
 - **Supervised learning (threshold-based)**
 - **Status display**

Demo

1. Create matrix
2. Populate matrix with sensors
3. Bootstrap (automatic)
4. Initial training specific to sensor type
5. initiate events
 1. Normal
 1. Undetected
 2. Detected
 2. Urgent
 1. Detected by isolated node
 2. Detected by node 1-hop to sink
 3. Detected by node multi-hop to sink
 3. Confirmation process
 1. Confirmed
 2. Unconfirmed
 4. Learning process
 1. One of the above events causes threshold change for node
 2. Repeat of that event does not initiate Urgent signal

Future Work

- **Algorithm development**
 - Investigate impact of learning processes on energy depletion
 - Refine learning and notification processes
- **Prototype enhancements**
 - Add local learning (quality updates)
 - Implement parent-child relationship and data aggregation at “family” level
 - Multiple routing paths
- **Investigate alternative simulators for prototype experimentation**

Bibliography

1. S. Zhu, S. Setia, and S. Jajodia. Leap: Efficient security mechanisms for large-scale distributed sensor networks. In 10th ACM Conference on Computer and Communications Security, Washington D.C, USA, October 2003.
<http://www.cse.psu.edu/~szhu/papers/leap.pdf>
2. OTMK - http://www.cs.colorado.edu/~rhan/Papers/deng_key.pdf
3. SQL for motes - http://www.intel-research.net/Publications/Berkeley/021020031311_118.pdf
4. TAG - http://www.intel-research.net/Publications/Berkeley/120520021027_101.pdf