On Maintaining Sensor-Actor Connectivity in Wireless Sensor and Actor Networks

Yinying Yang
March 14, 2008

Content

- Introduction
- Model
- Proposed methods
- Extensions
- Simulations
- Conclusion
- Reference

Introduction

- Wireless sensor and actor network: *actors and sensors*

Introduction

- Idea: try to reduce the routing space by putting as many sensors to sleep as possible to limit the energy consumption subject to the following area coverage and fault-tolerance requirements:
  - Coverage: each sleeping sensor has at least one neighbor that is either an active sensor or an actor.
  - Connectivity: each active sensor is still connected to the same set of actors as it was before sensors were put to sleep (called *persistent actor-connectivity*) or to at least one actor (called *at-least-one actor-connectivity*).

Model

- An undirected graph *G* = (*V, E*). *V* = *S* U *A*, where *S* is the sensor set and *A* is the actor set. $E \subset (S * S) \cup (S * A)$ is the edge set for sensor-sensor and sensor-actor connections.
- There is no direct connection between any two actors.
- A graph *G* is actor-connected if each sensor is connected to an actor through nodes in *G*

Model

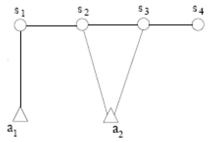
- Suppose a subset *S'* of *S* is put to sleep (for energy saving). We denote *G'* = *G*[*V*-*S'*], i.e., the network after removing *S'*

Definition 1: Given an actor-connected network G:

- G' *is persistent actor-connected if it maintains the same actor-connectivity as G, i.e., if a sensor, sleeping or active, is connected to an actor through nodes in G, then it is still connected to the actor through nodes in* G'.
- G' *is at-least-one actor-connected if each sensor, sleeping or active in G, is connected to at least one actor through nodes in* G'.

Model

- Example



- If *S'* = {s_4}, it is persistent actor connected
- If *S'* = {s_1, s_2, s_4}, it is no longer persistent but it is at-least-one actor-connected

Proposed methods

- Consider a WSAN that is initially actor connected:

> **Local rule for persistent actor-connectivity**: *The default status of a sensor is active. A sensor u is in sleep mode if, for any two of its neighbors w and v, w and v are connected by a path with all intermediate nodes (sensors or actors if any) having higher priorities than u.*

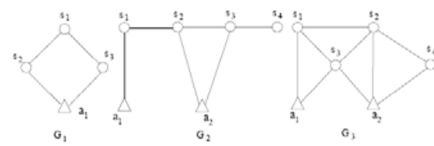
- The above path is called a *replacement path* for node *u*.
- The intuition behind this rule is that a sensor *u* can be put to sleep if any two neighbors can be re-connected through nodes on a replacement path.

Proposed methods

- To avoid inconsistencies and a possible iterative process of putting sensors to sleep, a global priority is defined on each node.

- Note that if a sensor does not have two neighbors, then the replacement path condition is satisfied and the status of the sensor is *sleeping*.

Proposed methods



- *p*(*a*1) = *p*(*a*2) > *p*(*s*1) > *p*(*s*2) >*p*(*s*3) > *p*(*s*4)
- Using 2-hop neighborhood information, *s*1 and *s*3 are put to sleep in *G*1 in persistent actor-connectivity; *s*4 is in sleeping in *G*2 and *s*3 and *s*4 are in sleeping in *G*3

Proposed methods

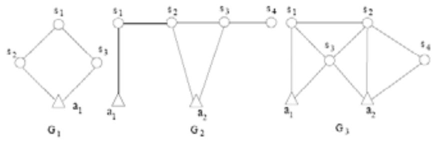
- To provide a local rule for at-least-one actor-connectivity, we define an *extended replacement path* as follows:
  - it is regular replacement path for *u* connecting two neighbors *w* and *v*, or
  - *w* and *v* are each connected to an actor. These two actors can be distinct and all intermediate nodes in these two connections have higher priorities than *u*.

Proposed methods

> **Local rule for at-least-one actor-connectivity**: *The default status of a sensor is active. A sensor u is in sleep mode if for any two of its neighbors w and v, an extended replacement path for u connecting w and v exists.*

- The intuition behind the above rule is that sensor *u* can be put to sleep as long as any two neighbors can be either connected through a regular replacement path or each of them is connected to an actor.

Proposed methods



- $p(a1) = p(a2) > p(s1) > p(s2) > p(s3) > p(s4)$
- Using 2-hop neighborhood information, *s*1 and *s*3 are asleep in *G*1 for at-least-one actor connectivity;
- *s*1, *s*2, and *s*4 are asleep in *G*2
- all sensors are asleep in *G*3.

Proposed methods

Theorem 1: Suppose S' is the set of sleeping sensors after applying the local rule for persistent actor-connectivity.

- *(Coverage) Each sensor in S' has a neighbor in V'.*
- *(Connectivity) G' has the same actor-connectivity as G.*

Theorem 2: Suppose S' is the set of sleeping sensors after applying the local rule for at-least-one actor-connectivity.

- *(Coverage) Each sensor in S' has a neighbor in V'.*
- *(Connectivity) Each node in G' is connected to at least one actor.*

Implementation issues

- Selection of priority
  - Node ID, energy level
- Controlling the path length
  - Set each replacement path to be bounded by *h* hops, then globally, the shortest path "stretch" of each sensor to an actor can be controlled.
- Static v.s. dynamic way
  - In static implementation, each node determines its status based on its *h*-hop information.
  - In dynamic implementation, each node acts on a message originated from an actor.

Implementation issues

- actor-initiated dynamic implementation
  - At-least-one actor-connectivity: (1) Each actor sends out an invitation message. (2) Each sensor responds to the first invitation only and forwards the invitation to its neighbors. (3) Sensors receiving responses are active, and sensors not receiving responses are put to sleep.
  - Persistent actor-connectivity: (1) Each actor sends out an invitation message with its ID. (2) Each sensor responds to the first invitation for each ID and forwards the invitation to its neighbors. (3) Sensors receiving responses are active and sensors not receiving responses are put to sleep.

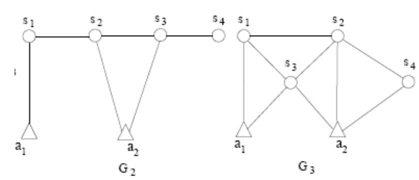
Extensions

- Two new notions of connectivity:

Definition 2: A WSAN, G, is called weak k-actor-connected if each sensor is connected to k actors. A WSAN, G, is called strong k-actor-connected if each sensor is connected to at least one actor after removing any $k-1$ nodes (sensors or actors) from G.

  - Weak *k*-actor-connectivity can tolerate *k*-1 actor failures, while strong *k*-actor-connectivity can tolerate *k*-1 failures of any nodes, sensors and actors.

Extensions



- *G*2 is weak 2-actor-connected but not strong 2-actor-connected. *G*3 is strong 2-actor-connected.
- To simplify the notation, we use *k*-actor-connected for strong *k*-actor-connected.

Extensions

Definition 3: *Given a k-actor-connected network G,*

- *G' is persistent k-actor-connected if it maintains the same actor-connectivity as G after removing any $k-1$ nodes (sensors or actors).*
- *G' is at-least-one k-actor-connected if each sensor, sleeping or active, in G is connected to at least one actor through nodes in G' after removing any $k-1$ nodes.*

- Here, "the same actor-connectivity" means that if a sensor in *G* is connected to an actor through nodes in *G*, then this sensor (which might be sleeping) is still connected to the actor through nodes in *G'*

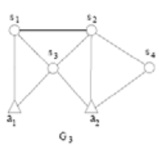
Extensions

Local rule for persistent k-actor-connectivity: *The default status of a sensor is active. A sensor u is in sleep mode if for any two of its neighbors w and v, k node-disjoint replacement paths for u connecting w and v exist.*

Local rule for at-least-one k-actor-connectivity: *The default status of a sensor is active. A sensor u is in sleep mode if for any two of its neighbors w and v, k node-disjoint extended replacement paths for u connecting w and v exist.*

- Two paths are called node-disjoint if they do not share any intermediate nodes.
- Note that both *w* and *v* can be actors and in this case, there is no intermediate node. Also, the actor cannot be shared in two extended replacement paths.

Extensions



- *p*(*a*1) = *p*(*a*2) > *p*(*s*1) > *p*(*s*2) >*p*(*s*3) > *p*(*s*4).
- In *G*3, *s*4 is put to sleep based on local rules for persistent 2-actor-connectivity.
- Sensors *s*3 and *s*4 are put to sleep from *G*3 for at-least-one 2-actor-connectivity.

Simulations

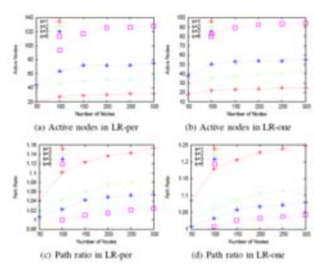
- Evaluate the two proposed algorithms, Local Rule for persistent *k*-actor-connectivity (LR-per) and Local Rule for at-least-one *k*-actor-connectivity (LR-one) with different system parameters.

Network Area	100×100
Transmission Range	25
Node Degree	12
Number of Sensors	n, 50 to 300
Number of Actors	m, 2 to 8
Number of Hops	h, 2 to 4
Connectivity Requirement	k, 1 to 6
Number of Trials	100

Simulations

- Metrics
  - *Active nodes*. The number of active sensors, which represents the energy consumption.
  - *Path ratio*. The ratio of the average length of the routing paths in *G'* to that in *G*, which represents the routing latency of the system.

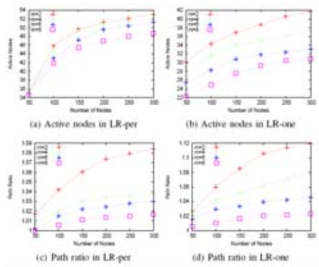
Simulations results



(a) Active nodes in LR-per (b) Active nodes in LR-one (c) Path ratio in LR-per (d) Path ratio in LR-one

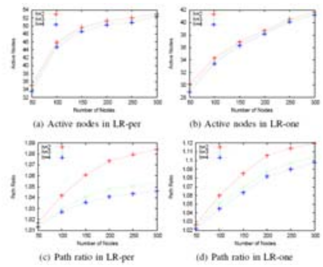
- Larger *k*=>more active nodes
- LR-one needs fewer active nodes than LR-per
- Larger k=>smaller path ratio

Simulation results



- Larger m=>smaller number of active nodes
- Larger m=>smaller path ratio

Simulation results



- Larger h=>smaller number of active nodes
- Larger h=>smaller path ratio

Conclusion

- Defined several sensor-actor connection requirements in wireless sensor and actor networks.
- Proposed several local solutions to ensure different connection requirements, where each node makes its decision (regarding its active and sleeping mode) purely based on local information, and there is no information propagation during the decision process. The methods are simple and easy to implement.
- Several fault-tolerance extensions, the persistent and at-least-one *k*-actor connectivity.

Reference

- Jie Wu, Shuhui Yang, and Mihaela Cardei, On Maintaining Sensor-Actor Connectivity in Wireless Sensor and Actor Networks, *INFOCOM*, 2008.