Increasing Connectivity in Wireless Sensor-Actuator Networks Using Dynamic Actuator Cooperation

M. Farukh Munir and F. Filali
Institut Eurécom, Department of Mobile Communications, Sophia Antipolis, France
Email: {munir,filali}@eurecom.fr

Abstract—**Distributed systems based on networked sensors and actuators with embedded computation capabilities are commonly used to monitor and control the physical world. To provide a meaningful service such as disaster and emergency surveillance, meeting real-time-and-energy constraints and the stability of transmit queues are the basic requirements of communication protocols in such networks. In settings with sparse distribution of actuator nodes, multi-hop routing is traditionally used to relay information to a remote sink. A problem with this approach is that the loss of connectivity of actuator nodes may lead to partitioning of the network.**

In this paper, we address the problem of minimizing power consumption at each actuator node and minimizing assignment overhead at each sensor node while ensuring network connectivity. We propose that each actuator is enabled with two wireless interfaces; one to communicate with its assigned sensor network, and the other, to communicate with the network of neighboring actuators. At any instant of time, an actuator can adjust its transmit power level to ensure connectivity and proactively inform its attached sensor network in case of mobility. These strategies have an associated cost. We show in this paper that at any instant of time, the strategy chosen by an actuator to adjust power and control-overhead due to mobility are optimal subject to constraints. The proposal is validated by means of analysis and simulations.

I. INTRODUCTION AND RELATED WORK

Sensor-actuator networks (SANETs) enable an instrumentation of the physical world at an unprecedented scale and density, thus enabling a new generation of monitoring and control applications. Such networks consist of large number of distributed sensor and few actuator nodes that organize themselves into a multihop wireless network. Typically, these nodes coordinate to perform a common task. Whereas, the actuators gather this information and react accordingly.

In this paper, we discuss the problem of energy-efficient dissemination of data from sensors to actuators (sensor-actuator coordination) and minimum connectivity between actuators (actuator-actuator coordination) using dynamic actuator coordination. In SANETs, the two coordination frameworks (sensor-actuator and actuator-actuator) work *independently* from each other with some *implicit* assignment issues between sensors and actuators. In [2], the authors provide algorithms to augment an existing static network into a k-connected network , for any desided k by placing a minimum number of additional nodes. The potential problem of minimum actuator-connectivity in a SANET with mobile actuators has not been investigated yet. Further, this problem at actuator-actuator coordination level is an interesting and non-trivial problem in terms of penalties for minimum actuator-connectivity. We consider a SANET that is deployed on a remote location and is representative of collection of data generated in the network and some actuation tasks. Sensors are static while the actuators move to perform some collaborative actuation tasks. For sensor-actuator coordination, we propose that each sensor transmit its reading to only one of the actuators. The assigned actuator is optimal in *minimum-delay* sense since delay is a *hard* constraint for such networks [5]. For static SANETs, one can find an optimal actuator assignment for each sensor and compute an optimal routing [1] to disseminate information towards these actuators. With mobile actuators, such optimal routing can incur heavy protocol overhead penalties due to mobility. Therefore the sensors use a multihop and multipath (load-balanced approach) routing for data dissemination towards an actuator which is min-hop away at any instant of time. For actuator-actuator coordination layer, we propose a power control solution for minimum actuator-connectivity and a proactive approach to minimize assignment-overhead at each sensor node.

The organization of the paper is as follows. The network model is presented in Section II. In Section III, we discuss the sensor-actuator coordination issues. Section IV details the actuator-actuator coordination. The dynamic actuator coordination is presented in Section V. Section VI details the simulation results. We conclude the paper in Section VII with some future directions.

II. NETWORK MODEL

In this paper, we consider a SANET with N static sensors and M mobile actuators as shown in Fig. 1.

Antenna and Frequency: Each sensor node is equipped with an omni-directional antenna operating on the same frequency. Whereas, each actuator is provided with two

---

Institut Eurécom's research is partially supported by its industrial members: BMW Group Research & Technology - BMW Group Company, Bouygues Télécom, Cisco Systems, France Télécom, Hitachi Europe, SFR, Sharp, STMicroelectronics, Swisscom, Thales.

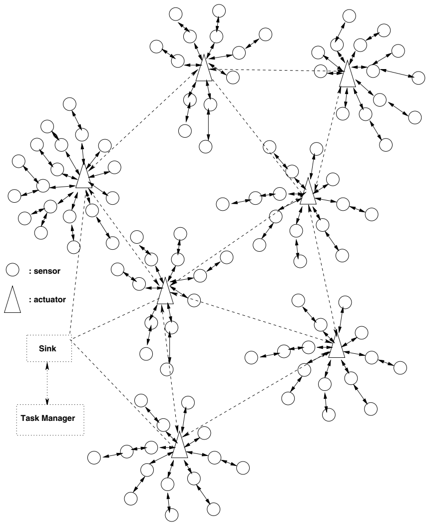

Fig. 1. SANET Architecture

omni-directional antennas: one to communicate with its assigned sensor-network, and the other to communicate with its neighboring actuator-network which operates on a different frequency. The antenna for actuator-actuator communication has defined finite power levels.

Neighborhood relation model: Given is an $N \times N$ neighborhood relation matrix $\mathcal{N}$ that indicates the node pairs for which direct communication is possible. We will assume that $\mathcal{N}$ is a symmetric matrix, i.e., if node i can transmit to node j, then j can also transmit to node i. For such node pairs, the $(i, j)^{th}$ entry of the matrix $\mathcal{N}$ is unity, i.e., $\mathcal{N}_{i,j} = 1$ if node i and j can communicate with each other; we will set $\mathcal{N}_{i,j} = 0$ if nodes i and j can not communicate. For any node i, we define $\mathcal{N}_i = \{j : \mathcal{N}_{i,j} = 1\}$, which is the set of neighboring nodes of node i.

Channel Access Mechanism: We assume that the network operates in discrete time, so that the time is divided into fixed length slots. We also assume that the packet length (or, transmission schedule length) is fixed throughout system operation. The system operates on CSMA/CA MAC. Assuming that there is no exponential back-off, the channel access rate of node i (if it has a packet waiting to be transmitted) is $0 \le \alpha_i \le 1$ (to avoid *pathological* cases). Thus, α_i is the probability that node i, if it has a packet to be transmitted, attempts a transmission in any slot. A node can receive a transmission from its neighbor if it is not transmitting and also no other neighboring node is transmitting, i.e., if the transmission is meant for some node j, $j \in \mathcal{N}_i$, then the transmission from node i to node j is successful iff none of the nodes in the set $j \cup \mathcal{N}_j \backslash i$ transmits.

Traffic Model: Each actuator node is assumed to be collecting data from its assigned sensor network at a defined average-rate; we let λ_i denote this collection rate for actuator i. The units of λ_i will be packets per second. Each sensor (actuator) node wants to use the sensor (actuator) network to forward its sampled (collected) data to a *common* actuator (sink) which is assumed to be a part of the network. Thus, each sensor (actuator) node acts as a forwarder of data from other sensor (actuator) nodes in the network. We let ϕ denote the $n \times n$ routing matrix. The $(i, j)^{th}$ element of this matrix, denoted $\phi_{i,j}$, takes value in the interval $[0, 1]$. This means a probabilistic flow splitting as in the model of [3], i.e., a fraction $\phi_{i,j}$ of the traffic *transmitted* from node i is forwarded by node j. Clearly, we need that ϕ is a stochastic matrix, i.e., its row elements sum to unity. Also note that $\phi_{i,j} > 0$ is possible only if $\mathcal{N}_{i,j} = 1$.

III. SENSOR-ACTUATOR COORDINATION

The coordination between sensors and actuators follows a hierarchical architecture and deals with the dissemination of information from sensors to actuators. For a sensor node, the energy consumption due to wireless communication (i.e receiving and transmitting) is considered the dominant source in power consumption. The power consumed by a sensor node i in receiving can be modeled as $P_r^i = P_{rx} \sum_{j \in N_i} f_{j,i}$, where $f_{j,i}$ is the rate $(packets/s)$ at which node j is transmitting packets toward node i. Also P_{rx} is the power consumed to receive a packet. The power consumed by a sensor node i in transmitting its data (both locally originated and forwarded packets) is given by $P_t(i, j) = c_{i,j}.f_{i,j}$, where $c_{i,j}$ is the power consumption coefficient for data transmission between sensor i and j. And $f_{i,j}$ is the total flow from sensor i to sensor j $pakcets/s$. Also $c_{i,j} = \alpha + \beta.d_{i,j}^m$, where α and β are constants, $d_{i,j}$ is the distance between the sensors i and j, and m is the path loss index. A sensor's destination actuator could be fixed as an outcome of this cost-function [5]. In this paper, as the actuators are mobile, the sensors always use a min-hop actuator-assignment to transmit their readings. In order to extend the network-lifetime, each sensor can dynamically update its route by retrieving the remaining energy of its *one-hop* uplink neighbors and selecting aroute with the maximum remaining energy. This calculation is simple as it requires only one-hop neighborhood communication and can result in extended network-lifetime.

IV. ACTUATOR-ACTUATOR COORDINATION

The coordination between the actuators follows a QoS architecture which can be divided into a number of categories based on application requirements [5]. Since, we have only one sink in the network, the actuator network can form an aggregation tree towards the common sink and flow from an actuator can be splitted and send over multiple routes toward the sink for remote processing requirements. Since we also opt to perform optimization at this network level, the optimal flow problem to obtain minimum end-to-end delays at this coordination level can be done in a similar fashion as in [3].

A. Classification of Actuation Process

We classify the actuator coordination into two types which covers all the requirements for an effective actuation process.

1) Distributed Single-Actuator Actuation Process: A sensor transmits/forwards the readings to its optimal actuator. The actuator can process all incoming data and initiate appropriate actions without any involvement of neighboring actuators, e.g., a high alert security application. The actuators can later route this information back to the sink for some remote processing. This approach is referred to as AF (Action First) approach.

2) Distributed Multiple-Actuator Actuation Process: Upon receiving the event information, an actuator route it to the neighboring actuators in order to best decide the optimal actuation strategy, e.g., in case of fire, the actuators need to efficiently collaborate so that the fire can easily be extinguished before it becomes uncontrollable. In this fashion, an energy constrained sensor do not need to transmit its readings to multiple actuators. Instead, the first actuator to receive this event information will relay it to its neighboring actuators to come up with an optimal actuation plan. This approach is referred to as DF (Decision First) approach. This actuation expectation can be expressed as follows:

$$D_m^{a(x,y)} = \eta d\left(m,(x,y)\right) + \zeta p\left(a\left(x,y\right)\right)$$

where $D_m^{a(x,y)}$ is the expectation for actuator $m, (1 \leq m \leq M)$ to join the actuation process $a\left(x,y\right)$, where (x,y) determine the coordinates of the actuation area. $d\left(m,(x,y)\right)$ is the distance of actuator m from the actuation area (x,y). $p\left(a\left(x,y\right)\right)$ is the priority of of actuation process $a\left(x,y\right)$. η and ζ are application dependent adjustable parameters. Depending on the application, we can set a threshold ϵ . If $D_m^{a(x,y)} > \epsilon$, then the actuator m will participate in the actuation process. For this study, we do not take into account the energy consumption issues for actuation expectation $D_m^{a(x,y)}$ because the energy source is assumed to be infinity (rechargeable energy source). For cases, where the energy source is finite at the actuators, we can also model the actuation expectation with an additional energy constraint and its own adjustment parameters.

B. Data Collection Mechanism and Distributed Routing

At any instant of time, an actuator may have two types of packets to be transmitted:

1) Packets received by the assigned sensor network.
2) Packets from neighboring actuators that arrived at this actuator and need to be forwarded.

Clearly, an actuator needs to have some scheduling policy to decide on which type of packet it wants to transmit, if it decided to transmit. A first come first served scheduling is one simple option. Yet another option is to have two separate queues for these two types of packets and do a weighted fair queueing (WFQ) for these two queues. In this paper, we consider the second option. Under this mechanism, an actuator node i has two queues associated with it: one queue (denoted Q_i) contains the packets that i has received from its assigned sensor network and the other (denoted F_i) contains packets that i has received from one of its neighboring actuators and has to be relayed. The combined channel access/data sampling mechanism is as follows: Actuator i decides to attempt a channel access with probability α_i in any slot (else, it is sensing the channel for any possible transmissions). If decided to attempt a transmission, the actuator first checks the number of packets available in either of its transmit queues. We have following possibilities:

1) If only one of the transmit queue is non-empty, the actuator i selects packet from this non-empty queue to transmit.
2) If both Q_i and F_i are non-empty: In this case, actuator i will do the following:
    a) with probability $1 - f_i$ the head-of-line packet from from Q_i is attempted transmission.
    b) with probability f_i, forward the head-of-line packet from F_i waiting to be forwarded.

We assume that the queue Q_i is always nonempty, i.e., sensor nodes make new measurements and continuously transmit packets to their assigned actuators. A detailed stability analysis of this scheme without power control is presented in [7]. Under the added freedom of traffic splitting, the routing algorithm is expected to use those routes for which the expected delays are smallest. Using the above model there will be a delay, say $\tau_{j,i}$ of the packet from actuator j to be served at actuator i; this packet could have originated at actuator j or may have been forwarded by actuator j. The Expected delay of a packet transmitted from actuator j is thus $\sum_{i \neq j} \phi_{j,i} \tau_{j,i}$. Since delays are additive over a path, packets from any actuator will have a delay over any possible route to the sink. Actuators iteratively keep updating the one-hop routing probabilities based on the delays incurred for every possible path [3].

C. Stability Analysis with Power Control

Let there be a finite set of power levels that an actuator node is allowed to use; denote this set by $\{l_1, ..., l_L\}$ $assume$ $(l_k < l_{k+1})$. A actuator has to decide on the next hop actuator (thus requiring appropriate power for transmission). Let $\mathcal{N}_i\left(k\right)$ be the set of actuators that can receive $i's$ transmission when actuator i is using power level l_k. Actuator i accesses channel with probability α_i and we are in the scenario where actuator i always have data to transmit (coming from its assigned sensor network).

The routing now gives the power level used for transmission; assume that $m_{i,j}$ is such that actuator i needs power $l_{m_{i,j}}$ to communicate with actuator j (this is assumed to be symmetric, i.e., $m_{i,j} = m_{j,i}$). Clearly, the routing will now change the neighbors of actuator, i.e., since routing determines the transmission power, the actuators which can use receive transmissions from i will also change. Since j_i denotes the next hop of actuator i, $l_{m_{i,j_i}}$ will denote the power used by actuator i for any transmission.

Lemma 1: The probability of success of a transmission from actuator i is then

$$s_i = \sum_{j_i \in \mathcal{N}_i} \phi_{i,j_i} \left(1 - \alpha_{j_i}\right) \prod_{k: j_i \in \mathcal{N}_k \left(m_{k,j_k}\right) \setminus i} \left(1 - \alpha_k\right) \quad (1)$$

Lemma 2: The throughput of data of actuator i is thus

$$\lambda_i = \alpha_i \left(1 - \pi_i + \pi_i \left(1 - f_i\right)\right) s_i \Rightarrow \alpha_i \left(1 - \pi_i f_i\right) s_i. \quad (2)$$

where π_i is the probability that the forwarding queue of actuator i is not empty.

Let H be a matrix with entries 0 or 1 so that $H_{i,j} = 1$ if $\sum_{n=1}^{\infty} \left(\phi^n\right)_{i,j} > 0$, i.e., data originated at actuator j is forwarded by actuator i. Then, the stability condition for the forwarding queues in the actuator network is

$$\alpha_i f_i s_i \geq \sum_{j \in \mathcal{N}_i} H_{i,j} \lambda_j \quad (3)$$

The idea in this case is that an actuator may be using large power for transmissions, thus reducing the end-to-end delay, however at the same time it interferes with more neighboring actuators (note that large transmission power of an actuator does not imply that it sees large amount of interference; it merely means that this actuators causes more interference). Hence, an actuator using large transmission power may be causing local inefficiency.

We are mainly interested in the throughput of the actuator nodes. Hence, we want to provide a fair throughput to all of the actuators. Recall that we are in a cooperative framework so that all the actuators in the network can be persuaded to compromise on their performance in order to have better overall performance. For this objective, we would like to be fair among the users, as well as, as efficient as possible. Further, when considering power control, we would like to have long term power constraint which will have the form

$$\alpha_i l_{m_{i,j_i}} \leq q_i \quad (4)$$

where q_i is an upper bound on the power consumption by an actuator i. The optimal transmit power for every actuator node can be calculated in a centralized fashion similar to one presented in [4], but this solution do not work well for mobile scenarios. Therefore, we present a distributed approach based on heuristics that adaptively adjusts each actuators transmit power in response to topological changes and attempt to maintain a connected topology using minimum power in Section V.

V. DYNAMIC ACTUATOR COOPERATION

As detailed in Section II, we have a sensor-actuator network with N static sensors and M mobile actuators. In mobile scenarios, the topology is constantly changing. The solution must, therefore, continually re-adjust the transmit powers of actuators to maintain the desired topology. Further, the solution must use only local or already available information since updating global information such as positions of all the actuator nodes require prohibitive control overhead. Thus, the centralized solutions are not viable in this mobile context. Due to these constraints, the mechanism presented here is necessarily a heuristic algorithm and offer no guarantee on worst-case performance. In particular, power control is done using a cross-layer approach between MAC-PHY layers and is at best a poor approximation to an optimal solution.

PC: A Heuristic Algorithm

1) Every actuator is configured with three parameters, namely: the *desired* node degree A_d (for an application specific actuation process), a high threshold on node degree A_h, and a low threshold A_l . Periodically, an actuator checks its degree (the current node degree A_c) in its neighborhood set $\mathcal{N}_i$ (provided by routing). If $A_c \geq A_h$, then an actuator reduces its transmit power. If $A_c \leq A_l$, then an actuator increases its transmit power. If none of the above is true, no action is taken. The minimum and maximum transmit powers are l_1 and l_L, respectively (see Section IV-C). Further, the magnitude of power *change* is a function of A_d and A_c.
2) Let p_d and p_c be the desired and current transmit power levels, respectively. Then, the desired power level (A similar derivation of this desired power level calculation is provided in [4]. Therefore, we do not repeat it here to conserve space.) is given by

   $$p_d = p_c - 5.m.log_{10} \frac{A_d}{A_c}. \quad (5)$$

   A node knows its current transmit power level p_c and its current neighborhood node degree A_c (given by l_k and $\mathcal{N}_i(k)$, respectively) and A_d is a configured value. Also, m is the path loss index and it takes values $2 \leq m \leq 5$. In our work, we take the value of $m = 4$ as mentioned in Section III. Then, (5) can be used to calculate the required power periodically, iff

   $$s_i\left(A_d\right) \geq s_i\left(A_c\right). \quad (6)$$

   where the calculation of $s_i\left(A_d\right)$ and $s_i\left(A_c\right)$ can be easily performed at the MAC layer using (1) with associated parameters.

We are interested in power control if and only if it improves the success probability s_i, which is a function of $\mathcal{N}_i(k)$ (1). Further, it plays and important role in determining the throughput of an actuator (2). It is also seen in Section IV-C, that the routing with power control changes the neighborhood set $\mathcal{N}_i(k)$ of actuator i. Therefore, the desired power level in (5) is practically applied if and only if (6) is valid.

In addition to power control, the mobility of actuator nodes results in network disconnectivity with its assigned sensor-network. Therefore, if an actuator node is expected to move from its current location, it broadcasts a packet informing all the sensors in its cluster of a change in position. This change is typically broadcasted to neighboring actuators as well. Thanks to the distributed learning approach proposed in [5], after initial network learning each sensor has multiple paths available to possibly different destination actuators, which can be verified by sending a 'Hello' message. Hence, a new actuator attachment is obtained in a fairly delay-energy efficient manner for the constrained sensor nodes using *dynamic* actuator cooperation. This cooperation is dynamic in a sense that it is event based where the event is characterized by actuator mobility.

VI. SIMULATION RESULTS

The proposals presented in this paper are implemented in ns-2 [6]. Since, it is *hard* to simulate heterogeneous networks (like the one we are considering here), we modified the tcl-based ns-2 scripts in order to simulate the wireless sensor-actuator network. By hard, we mean that one can not simulate a network consisting of hybrid devices with different communication and networking capabilities. These scripts, in particular, modifies the communication capabilities of actuator nodes at run-time. The MAC uses is CSMA/CA and routing is performed as explained in Section III for sensor-actuator coordination level and in Section IV for actuator-actuator coordination. Owing to space limits, we do not detail all the simulation parameters. From Fig. 2, it can be seen that the throughput is maximum when an actuator only has one neighbor to route its data to the remote sink. This is only due to the presence of less *interference* in an actuators neighborhood. This can also be verified from (1), where an increase in the power level results in a increase in the neighborhood degree (minimizes the channel access due to more contending neighbors) and also changes the routing matrix. We could not present detailed results on the average-power used for transmissions and average-delay for the throughput results given in Fig. 2 due to space restrictions. We believe that there is still a need to do large amount of experimentation with different networking scenarios in order to provide a good insight into the working of PC heuristic algorithm. Fig. 3 shows the energy consumption due to routing control overhead both in the case of static and mobile topologies to perform power control. The results shown here are for 2-connectivity (at actuator-actuator coordination level) and 0.5 throughput. The updates are event based and require only one-hop message exchange among neighboring actuators. It also includes broadcast message transmissions to sensors in case of mobility.

VII. CONCLUSIONS AND FUTURE WORK

The actuators can dynamically coordinate and perform power control to maintain a defined level of connectivity subject to throughput constraints. The control overhead for static and mobile actuator scenarios are analyzed using ns-2 simulations. The PC heuristic is applicable to multihop SANETs to increase throughput, battery life and connectivity.

In future, we will also present a detailed simulation based study of PC heuristic algorithm in different networking scenarios with some application specific actuation requirements and practical evaluation of distributed multiple-actuator actuation process. We will also work on the development of PC heuristic algorithm to improve some MAC layer performance metrics using a cross-layer approach.

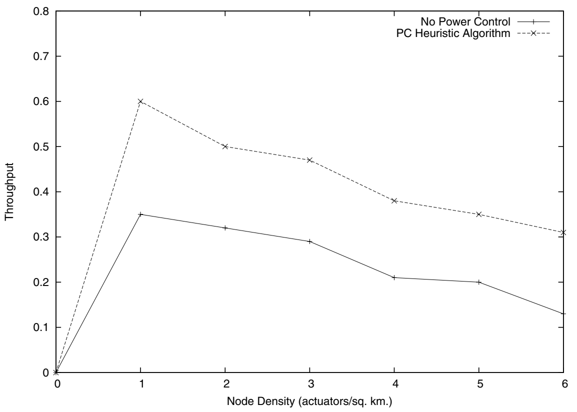

Fig. 2. Throughput vs. Actuator Density

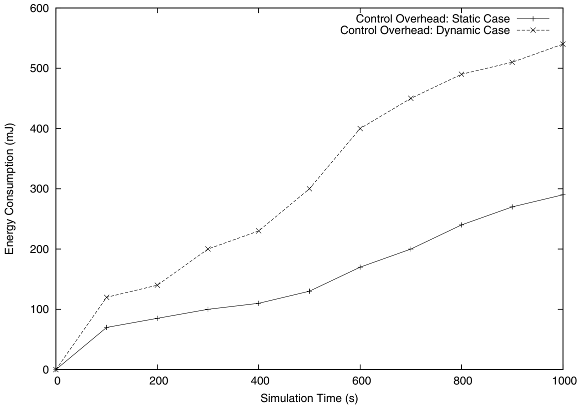

Fig. 3. Energy Consumption for Control Overhead

REFERENCES

[1] M. Farukh Munir and F. Filali. Maximizing Network-Lifetime in Large Scale Heterogeneous Wireless Sensor-Actuator Networks: A Near-Optimal Solution. In IEEE/ACM MSWIM PE-WASUN Workshop Proceedings, Chania, Cretes Island, Greece, Oct. 22-26, 2007, pp. 62-69.

[2] J. L. Bredin, E. D. Demaine, M. T. Hajiaghayi, and D. Rus. Deploying sensor networks with guaranteed capacity and fault tolerance. In ACM MobiHoc Conference Proceedings. Urbana-champaign, USA, May 25-27, 2005.

[3] M. Farukh Munir, Arzad A. Kherani, and F. Filali. On stability and sampling schemes in wireless sensor networks. In Proceedings of 5th IEEE/IFIP WiOpt Conference, Limassol, Cyprus, April, 2007.

[4] R. Ramanathan and R Rosales-Hain. Topology control of multihop wireless networks using transmit power adjustment. In Proceedings of IEEE Infocom Conference, Tel Aviv, Israel, Mar. 26-30, 2000.

[5] M. Farukh Munir and F. Filali. 2006. A novel self organizing framework for SANETs. In Proceedings of 12th EW Conference, Athens, Greece, Apr. 2-6, 2006.

[6] The Network Simulator - ns (version 2.27). http://www.isi.edu/nsnam/ns/.

[7] M. Farukh Munir, Arzad A. Kherani, and F. Filali. Stability and Delay Analysis for Multi-Hop Single-Sink Wireless Sensor Networks. In IEEE PerCom PerSeNS Workshop Proceedings, HongKong, Mar. 17-21, 2008.