Analysis of On-off policies in Sensor Networks using Interacting Markovian Agents

M. Gribaudo[1], D. Cerotti [1], A. Bobbio[2]
[1] Dipartimento di Informatica, Università di Torino, Torino, Italy
[2] Dipartimento di Informatica, Università del Piemonte Orientale, Alessandria, Italy
[1] marcog,cerotti@di.unito.it [2] bobbio@mfn.unipmn.it

Abstract

*Power management in battery operated sensor networks, is a hot topic addressed by many ongoing researches. One of the most commonly employed technique consists in turning on and of the power of the radio unit to reduce the power consumption. In this paper we exploit the modelling power of Interacting Markovian Agent to evaluate the performance of four different on-off strategies. A Markovian Agent (*MA*) is an entity whose behaviour is described by a continuous time Markov chain (CTMC) and that is able to interact with other* MA's *by sending and receiving messages. A perceived message may induce a state transition in a* MA *according to a* perception function *than can depend on the geographical location of the* MA's*, on the message routing strategy and on the transmission property of the medium. We represent each sensor by a* MA *and we show how* MA's *can interact to produce the collective behaviour of the sensor network.*

Keywords*: Sensor network, performance and dependability measures, Markov Agents, on-off behaviour*

1. Introduction

In this paper, we consider different policies to reduce power consumption in battery operated wireless sensor networks. We focus on the of energy-saving mechanisms at the MAC layer, similar to the one proposed in [13, 12, 14]. These works propose wake-up scheduling schemes that activate sleeping nodes when they need to transmit/receive, thus avoiding a degradation in network connectivity or quality of service provisioning.

We consider a sensor network in which sensors are distributed in a continuous finite geographical area according to a known spatial Poisson density [7]. The sensors receive stimuli from the environment and transfer that stimulus to a central base station (called *the sink*), by means of intermediate hops through sensors of the same type. Each sensor can be in an active (radio unit on), or asleep (radio unit off) or failed condition and we model different energy saving policies. When in its active condition, a sensor can buffer the received messages and retransmit them to the most convenient sensor to minimize the number of hops to reach the central base station.

Performance and dependability measures of are obtained by modeling the system by means of a new entity called *Markovian Agent (MA)*. In particular, we show how the proposed entity is suited to model and analyze very large stochastic systems of interacting objects, whose dimensions exceed the capabilities of any state-space based model, while representing, at the same time, the local properties and the local dynamics of each agent.

A Markovian agent *(MA)* has a finite number of discrete states, and chooses its actions randomly according to the infinitesimal generator of a CTMC. The *MA* communicates with the environment and with the other *MA's* by sending messages that can be received (perceived) by the other agents. Upon acceptance, the perceived messages may induce a state transition, so that the final behaviour of a *MA* is determined by its local model and by the interaction with the other *MA's*. The *MA's* are distributed in a finite geographical area according to a given spatial density and their interaction is defined by a *perception function* that depends on the spatial distribution, the transmission property of the medium and the message routing strategy.

The modelling and analysis of large scale stochastic systems composed by interacting objects has been mainly faced in the literature by resorting to the superposition of interacting Markov chains or to fluid models. In the first case, the available techniques require the generation of the global state space, defined as the Cartesian product of the state spaces of the individual interacting objects. The explosion of the state space

can be mitigated by exploiting symmetry properties, often included in the system definition, and producing the global transition rate matrix by means of tensor algebra operators applied to the local matrices [6]. Representative attempts in this direction define the interacting objects directly as Markov chains [3, 1, 5, 2], or as finite state automata [10, 11] or as Petri nets [4, 9]. However, the compositional approaches, based on finite state objects, do not account for interactions related to the relative position of the local objects. On the other hand, fluid models [8, 7] are able to capture the global behaviour of the system but loosing the capability of detailing the local behaviour. In our approach, the local objects are finite state *MA's* but their interaction is represented by a fluid model.

We model each sensor with a *MA*, and we assume that the spatial distribution of *MA's* is constant in time and we refer to this situation as *Static Markovian Agent Model (SMAM)*. The construction of the *MA* and the *SMAM* for the sensor network system, and the quantitative evaluation of some relevant performance and dependability indices of the collective system behaviour is the object of the present paper.

2. Markovian Agents

A *MA* is an extension of a continuous time Markov chain, that adds the possibility of receiving and generating *messages*. We define a Markovian Agent *MA* as a touple:

$$MA = \{\mathbf{Q}, \Lambda, P, R\} \tag{1}$$

Where:

$\mathbf{Q} = |q_{ij}|$ is the $n \times n$ infinitesimal generator matrix of a continuous time Markov chain.

$\Lambda = |\lambda_i|$, is a vector whose components represent the finite rate of *self-jump*, that is the rate at which the Markov chain reenters the same state.

$P = |p_{ij}|$ is a $n \times n$ matrix, that describes the message generation probability.

$R = |r_{ij}|$ is a $n \times n$ matrix, that describes the message acceptance probability.

$\mathbf{Q}$ is the infinitesimal generator of an ordinary CTMC, whose entry q_{ij} represents the transition rate from state i to state j, with $q_{ii} = -\sum_{j=1, j\neq i}^{n} q_{ij}$. We explicitly include the possibility of self-loops in the CTMC and we define λ_i as the rate at which the CTMC makes a jump reentering the same state. While resident in a state or during a transition a *MA* can generate a message. A message generated in a state can be viewed as a message generated during a self-loop.

Each element $p_{ij} \in P$, represents the probability that a message is generated when a transition from state i to j occurs. From the above definitions, we can compute the rate β_i at which messages are produced in state i:

$$\beta_i = \sum_{j \neq i} q_{ij} p_{ij} + \lambda_i p_{ii} \tag{2}$$

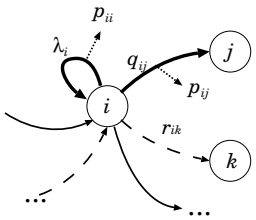


Figure 1. A representation of a Markovian Agent.

A *MA* can perceive the messages emitted by other *MA's*. A *MA* in state i has a probability r_{ii} of ignoring an arriving message and a probability $1 - r_{ii}$ of accepting an arriving message. An accepted message induces an immediate state change toward state j with probability r_{ij}. We have that:

$$\sum_{j=1}^{n} r_{ij} = 1, \qquad \forall 1 \leq i \leq n. \tag{3}$$

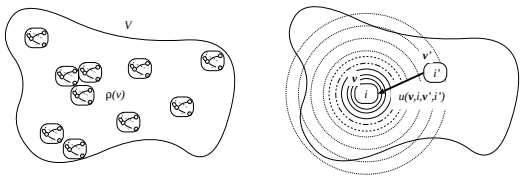
Since a *MA* can change state when it accepts a message, the transitions in the associated CTMC are not only determined by matrix $\mathbf{Q}$ but also by the rate at which received messages are accepted. The *perception function* models how emitted messages are perceived by other *MA's* and is primarily function of the spatial distribution of the agents, the routing strategy and the property of the medium. The *perception function* will be addressed in the following section. Figure 1, gives a visual representation of a *MA*. Continuous arrows represent state transitions governed by the infinitesimal generator $\mathbf{Q}$ of the Markov Chain. During such transitions, messages can be generated, as represented by the dotted arrows starting from a transition arc whose labels represent the generation probability. Dashed arrows that connect states represent transitions caused by the acceptance of an incoming message.

2.1. Static Markovian Agents Model

A *Static Markovian Agents Model (SMAM)*, is a collection of Markovian Agents over a space. In this paper we focus on a bi-dimensional topology. A *SMAM* is defined by the tuple:

$$SMAM = \{MA, \mathcal{V}, \rho, u\} \tag{4}$$

where:



a) b)

Figure 2. a) Markovian Agent over a continuous space, b) A representation of the perception function $u(\mathbf{v}, i, \mathbf{v}', i')$.

MA is a Markovian Agent.

$\mathcal{V}$ is the finite space over which Markovian Agents are spread.

$\rho : \mathcal{V} \rightarrow \mathbb{R}^+$ is the spatial density function of the Markovian Agents. We assume that the agents are distributed over the space $\mathcal{V}$ following a spatial Poisson process of finite parameter $\rho(\mathbf{v})$. That is, the number of agents in a finite area $A \subseteq \mathcal{V}$, is distributed according to a Poisson distribution $Pois(\int_A \rho(\mathbf{v})d\mathbf{v})$. In a *Static Markovian Agent Model*, the spatial density $\rho(\mathbf{v})$ remains constant. Figure 2a shows how the density $\rho(\mathbf{v})$ can be interpreted for the considered space.

$u : \mathcal{V} \times \mathbb{N} \times \mathcal{V} \times \mathbb{N} \rightarrow [0 \dots 1]$ is the *perception function*. $u(\mathbf{v}, i, \mathbf{v}', i')$ is the probability that an agent in position $\mathbf{v}$ in state i perceives a message generated by an agent in position $\mathbf{v}'$ in state i'. The definition of the *perception function* $u(\cdot)$ is quite general, and allows to model several message routing strategies and *MA* interdependencies. In particular, an agent in state i can distinguish the state i' in which the message was issued and take the corresponding action. A visual interpretation of function $u(\mathbf{v}, i, \mathbf{v}', i')$ is given in figure 2b.

3. Sensor network with base station

We model and analyze a wireless sensor network with a central base station topology using *SMAM*. Sensors are represented by *MA's* and are distributed in a continuous finite region $\mathcal{V}$ according to a spatial Poisson density of rate $\rho(\mathbf{v})$. Sensors receive stimuli from the environment that are transferred to the sink, by means of intermediate sensors of the same type.

Each sensor can be in three possible states, active (radio unit on), asleep (radio unit off) or failed. When active, the sensor can buffer the accepted messages by increasing the queue length by one, while the transmission of a buffered message decreases the queue length. A stimulus directly sampled from the environment generates an immediate message without increasing the buffer count. This action is represented by self loops. In order to show how the *MA* can reflect different local behaviours, we have modeled different policies for the active/asleep alternation.

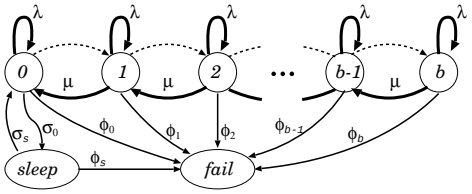


Figure 3. A sensor that can go to sleep only when its buffer is empty .

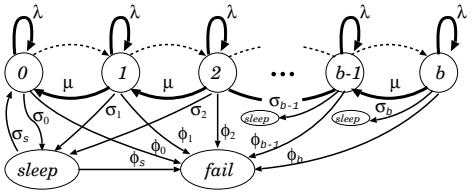


Figure 4. A sensor that discards all the content of its buffer when it goes to sleep (a preemptive repeat service policy).

i - The sensor can go to sleep only when the buffer is empty (Figure 3).

ii - The sensor discards the content of the buffer when it goes to sleep loosing all the buffered messages (Figure 4).

iii - The sensor first empties its buffer, and then goes to sleep (Figure 5).

iv - The sensor freezes the buffer when it goes to sleep (a preemptive resume service policy)(Figure 6).

In all the figures, b is the buffer dimension, σ_i is the sleeping rate from state i and ϕ_i is the failure rate from state i. λ_i is the self loop rate in state i coupled with a message generation probability $p_{ii} = 1$ (a

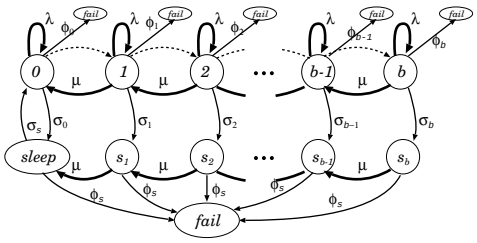


Figure 5. A sensor that first empties its buffer, and then goes to sleep .

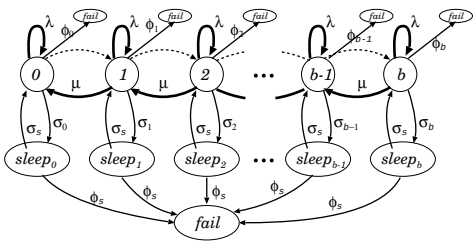


Figure 6. A sensor that freezes its buffer when it goes to sleep .

self loop always generates a message). μ is the message service rate coupled with a message generation probability $p_{i,i-1} = 1$ (a message service always generates a message). The dotted arc from state i to state $i+1$ signifies that the acceptance of a received message increases the buffer count by one.

We assume that the sensors are spread in a circular area of radius h and that the sink is in the center of the circle, and at the origin of the axis. The power of a message issued by a *MA* decays as the square root of the distance. Since we are looking at the perception capability we can assume that a maximum communication range d_{max} exists at which the signal has enough power to be received. Hence, a *MA* in position $\mathbf{v}$ is able to perceive a message from a *MA* in position $\mathbf{v}'$ if the distance $dist(\mathbf{v} - \mathbf{v}') < d_{max}$.

As presented in [7], a *minimum energy routing* strategy is obtained by transmitting the message to the sink through *MA's* located along the radius connecting the sensor to the sink, with hops of equal length. If $\mathbf{y}$ is the distance of the emitting *MA* from the sink, the minimum number of hops is $nh_{min} = \lceil \mathbf{y}/d_{max} \rceil$ and each hop has a length $d_h = \mathbf{y}/nh_{min} \leq d_{max}$.

The perception function $u(\mathbf{v}, i, \mathbf{v}', i')$ is constructed to route a message according to the *minimum energy routing* strategy. To this end, the destination sensors are located in a circular area of radius δ centered into the theoretically optimal next hop position for a sensor that is located along the radius connecting the sink at a distance d_h from the previous hop. Furthermore, we assume that the acceptance probability is $r_{i,i+1} = 1$ in the active states ($0 \leq i < b$) and zero elsewhere; i.e. a message received in an active state i, with at least one free buffer position, is accepted with probability 1 and produces a transition to state $i+1$.

3.1. Quantitative Performance Analysis

To address the transient analysis of the *SMAM* model, let us denote by $\pi_i(\tau, \mathbf{v})$ the probability that a *MA* at the spatial position $\mathbf{v}$ is in state i at time τ. We have that $\sum_{i=1}^{n} \pi(\tau, \mathbf{v}) = 1, \forall \tau, \mathbf{v}$. We collect all the $\pi(\tau, \mathbf{v})$ in a vector $\boldsymbol{\pi}(\tau, \mathbf{v})$ of size n. Due to the property of the Poisson distribution, the number of agents in a given state i, at a given position $\mathbf{v}$, at a given time τ, is distributed according to a Poisson distribution of parameter:

$$\rho_i(\tau, \mathbf{v}) = \rho(\mathbf{v})\pi_i(\tau, \mathbf{v}) \tag{5}$$

The total number of messages received by an agent in state i, depends on the tuple $(\tau, \mathbf{v})$, that specify its position $\mathbf{v}$, and the current time instant τ. We call $\gamma_i(\tau, \mathbf{v})$ the rate at which messages are observed by a *MA* in state i, at $(\tau, \mathbf{v})$. Due to the properties of the Poisson distribution, we can compute $\gamma_i(\tau, \mathbf{v})$ as:

$$\gamma_i(\tau, \mathbf{v}) = \int_{\mathcal{V}} \sum_{i'=1}^{n} \rho_{i'}(\tau, \mathbf{v}')u(\mathbf{v}, i, \mathbf{v}', i')\beta_{i'} d\mathbf{v}' \tag{6}$$

We collect the $\gamma_i(\tau, \mathbf{v})$ in the matrix $\Gamma(\tau, \mathbf{v})$. Not all the messages observed by an agent in $(\tau, \mathbf{v})$ cause a change of state. In fact, messages are ignored with probability r_{ii}. We can use matrix $\Gamma(\tau, \mathbf{v})$ combined with matrix $\mathbf{R}$ to characterize the stochastic process that describes the transient behavior of an agent. We define $\mathbf{C}(\tau, \mathbf{v})$ as the actual transition rate at $(\tau, \mathbf{v})$:

$$\mathbf{C}(\tau, \mathbf{v}) = \mathbf{Q} + \Gamma(\tau, \mathbf{v})\,[\mathbf{R} - \mathbf{I}] \tag{7}$$

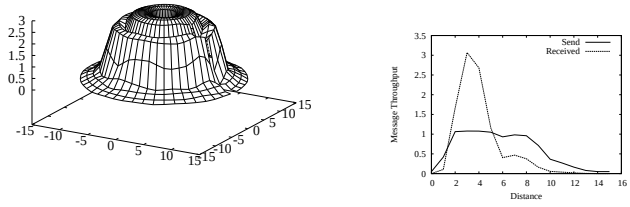
Matrix $\mathbf{C}(\tau, \mathbf{v})$ can be used to compute $\pi_i(\tau, \mathbf{v})$ using the standard Kolmogorov equations:

$$\frac{d\pi_i(\tau, \mathbf{v})}{d\tau} = \pi_i(\tau, \mathbf{v})\mathbf{C}(\tau, \mathbf{v}) \tag{8}$$

Solution of the Kolmogorov equations (8) can be obtained numerically, starting from a known initial condition $\pi(0, \mathbf{v})$ and solving iteratively.

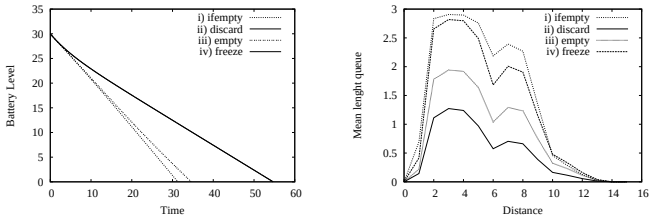
3.2. Numerical Results

We analyze the sensor model, on a circular space of radius $r = 16$, with a constant spatial sensor density of $\rho(\mathbf{v}) = 10$. We assume $\mu = 0.1$, and $\lambda = 0.01$. With these figures, each queue is very heavily loaded. In this example we assume a finite queue length of size $N = 3$. The maximum communication range is $d_{max} = 4.9$. We analyze the model by discretizing both the time and the space. All the results, presented here, were obtained on a standard Laptop PC, and took only a few seconds to be computed. We start by considering policy i) (i.e. the sensor can go to sleep only if its buffer is empty) without failure (i.e. $\phi_i = 0, \forall i$). Figure 7a shows the mean queue length for each sensor. Note that the model presents the expected axial symmetry. The rings with a higher queue length reflect the fact that *MA's* at certain distance from the sink are used more frequently as a relay by outer *MA's*. The mean queue



a) b)

Figure 7. a) spatial mean queue length, b) throughput of transmitted and received messages



a) b)

Figure 8. Comparison of policies: a) battery level vs time, b) mean queue length

length for all the four considered policies is presented in Figure 8b.

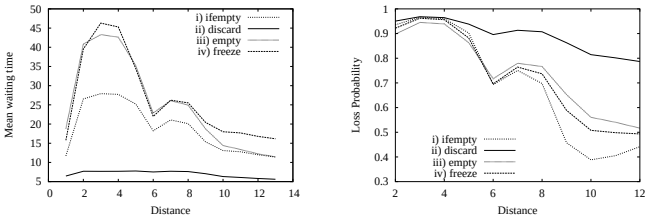
Two interesting performance indices can be computed for each sensor: the number of transmitted messages $T(\tau, \mathbf{v})$ and the number of received messages $R(\tau, \mathbf{v})$, per time unit. The two indices can be computed as:

$$T(\tau, \mathbf{v}) = \sum_{i=1}^{n} \pi_i(\tau, \mathbf{v})\beta_i$$

$$R(\tau, \mathbf{v}) = \sum_{i=1}^{n} \rho_i(\tau, \mathbf{v})\gamma_i$$

These indices, for the considered model, are shown in Figure 7b. Due to the axial symmetry, only a radial slice is presented.

We then study the effect of the different sleeping policies on the power consumption in the sensors. We assume that the battery is initially charged with 30 unit of energy and we assume a power consumption rate of 1 unit of energy per unit time when the sensor is active and a power consumption rate of 0.01 when the sensor is asleep. Further, we set for the sleeping rates $\sigma_i = \sigma_s = 0.1$. Figure 8a shows the power consumption as a function of the time, for the described sleeping policies. The policy *i)* is the most power consuming because there is a little probability of finding an empty queue that allows the transition to the sleeping mode. The policy *iii)* leads also to a significative power consumption since the queue must be emptied before going to sleep. Policies *ii)* (preemptive repeat) and *iv)* (preemptive resume) lead to the same low power consumption since the transition to the sleeping mode is independent of the queue length.



a) b)

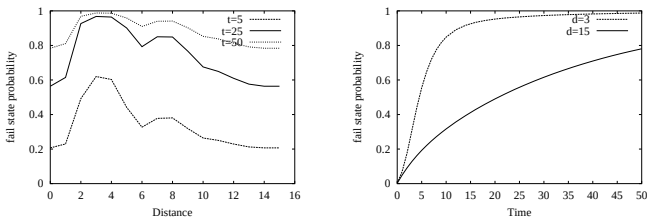
Figure 9. Comparison between policies: a) mean waiting time, b) probability of loss

However, the on-off policies have also large effects on the mean waiting time required by a packet to traverse a sensor queue (and thus on the total end-to-end delay), and on the loss probability as shown in Figure 9a and 9b. The discard policy (*ii*) is the one that experience the lowest waiting time: this is due to the fact that discarding the packets, increases the probability of finding the queue empty. Both preemptive policies have more or less the same waiting time.

The packet loss probability is computed by taking into account two factors: packets might be lost when they arrive with the buffer full, or when the sensor is sleeping. A further loss factor arises for the discard policy (*ii*) only, since packet might be lost when emptying the queue. As expected, the discard policy, has the highest packet loss probability. In all the other cases, it is interesting to see how the curves intersect when moving away from the center. This is due to the fact that in most heavily loaded areas, the highest contribution to the loss probability is given by packets arriving at a full queue. In the less loaded area, the highest contribution to the loss probability is given by the packets arriving when the sensor is sleeping.

3.3. Design issues

The proposed model seems to suggest some design issues for the choice of the best sleeping strategy in different situations. If we have a very redundant network, where packets losses may be tolerated, and a fast response time is required, then the discard policy (*ii*) is



a) b)

Figure 10. Probability of being in the fail state versus: a) space, b) time

the best choice since it has both the lowest latency and the lowest power consumption. If reducing the packet loss is vital, then the if-empty policy (*i*) should be chosen, since it is the one where sensors expect the minor loss (at the expense of an higher power consumption). Preemptive policies should be used in intermediate cases, preferring the resume policy (*iv*) when a longer system lifetime is required, and the one that first empties the buffer (*iii*), when a shorter response time is required.

3.4. Sensor failures

We consider now the failure state included in the figures 3, 6, to perform also a dependability analysis. We assume a state-dependent failure rate $\phi_i = 0.05 + 0.1 \cdot i$ when the sensor is active, $\phi_s = 0.01$ when the sensor is sleeping, and study the failure probability as a function of the space and as a function of the time. Figure 10a, shows that the sensor with a higher load presents a higher probability of being in the failed state at a given time instant. Figure 10b shows the probability of being in the failed state versus time for sensors located at different distances ($d = 3$ and $d = 15$) from the sink. The curve for $d = 3$ refers to sensors with a heavy load (see Figure 7a), and the curve for $d = 15$ addresses the case of sensors at the boundary of the considered region with a lighter load.

4. Conclusions

The paper has shown that it is possible to model a large sensor network by means of interacting *MA's*. By defining the spatial density of the *MA's*, very large systems of interacting objects can be analyzed and their relevant collective properties quantitatively computed, while maintaining the local identity and the local behaviour of each agent. Research work is in progress to increase the modeling as well as the decision power of the *MA* technique.

Acknowledgments

This work has been partially supported by the EU-Project Crutial IST-2004-27513.

References

[1] F. Ball, R.K. Milne, I.D. Tame, and G.F. Yeo. Superposition of interacting aggregated continuous-time Markov chains. *Advances in Applied Probability*, 29:56–91, 1997.

[2] H. Boudali, P. Crouzen, and M. Stoelinga. Dynamic fault tree analysis using input/output interactive markov chains. In *Proceedings of the 37th International Conference on Dependable Systems and Networks (DSN 07)*, 2007.

[3] P. Buchholz. Hierarchical Markovian models - symmetries and aggregation. *Performance Evaluation*, 22:93–110, 1995.

[4] P. Buchholz. Hierarchical structuring of superposed GSPNs. *IEEE Transactions Software Engineering*, 25:166–181, 1999.

[5] P. Buchholz and T. Dayar. Comparison of multilevel methods for Kronecker based Markovian representations. *Computing*, 73:349–371, 2004.

[6] P. Buchholz and P. Kemper. Kronecker based matrix representations for large Markov chains. In M. Siegle B. Haverkort, H. Hermanns, editor, *Validation of Stochastic Systems*, pages 256–295. Springer Verlag - LNCS, Vol 2925, 2004.

[7] M. Gribaudo, C.-F. Chiasserini, R. Gaeta, M. Garetto, D. Manini, and M. Sereno. A spatial fluid-based framework to analyze large-scale wireless sensor networks. In *IEEE International Conference on Dependable Systems and Networks, DSN2002*, 2005.

[8] J.M. Kelif and E. Altman. Downlink fluid model of CDMA networks. In *IEEE 61th Vehicular Technology Conference (VTC 2005)*, 2005.

[9] P. Kemper. Transient analysis of superposed GSPNs. *IEEE Transactions on Software Engineering*, 25:182–193, 1999.

[10] B.D. Plateau and K. Atif. Stochastic automata network for modeling parallel systems. *IEEE Transactions on Software Engineering*, 17:1093–1108, 1991.

[11] B.D. Plateau and J.M. Fourneau. A methodology for solving Markov models of parallel systems. *Journalof Parallel and Distributed Computing*, 12:370–387, 1991.

[12] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava. Topology management for sensor networks: Exploiting latency and density. In *3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2002.

[13] W. Ye, J. Heidemann, and D. Estrin. An energy efficient mac protocol for wireless sensor networks. In *IEEE Infocom, New York, NY*, 2002.

[14] R. Zheng, J. Hou, and L. Sha. Asynchronous wakeup for power management in ad hoc networks. In *MobiHoc 2003, Annapolis, MD*, 2003.