Policy-Based Adaptive Routing in Autonomous WSNs

Carlos M.S. Figueiredo[1,2], Aldri L. dos Santos[3], Antonio A.F. Loureiro[1], and José M. Nogueira[1]

[1] Dept. of Computer Science, Federal University of Minas Gerais, Belo Horizonte-MG, Brazil
{mauricio, loureiro, jmarcos}@dcc.ufmg.br
[2] FUCAPI - Research and Tech. Innovation Center, Manaus-AM, Brazil
[3] Dept. of Computer Science, Federal University of Ceará, Fortaleza-CE, Brazil
aldri@lia.ufc.br

Abstract. Wireless sensor networks (WSNs) are employed in different domains and applications. The resource constraint on such networks, many times composed of hundreds to thousands of devices, and the requirement of autonomous operation become their management a challenging task. This work applies policies, a well-known approach in network management, in the core task of routing in autonomous WSNs. Policies are used to establish rules to take dynamic actions on the network according to its state. Our scheme offers a high-level and flexible way to realize management tasks related to routing in WSNs, which can be defined in a progressive way as knowledge from the environment is acquired or application requirements change. Case studies employing a policy-based adaptive hybrid solution allows the autonomous selection of the best routing strategy in view of network conditions and application requirements. Simulation results show the benefits and resource savings offered by the use of policies for adaptive routing in WSNs.

Keywords: Wireless Sensor Networks, Routing, Policy-based design.

1 Introduction

A wireless sensor network (WSN) consists of sensor nodes connected among themselves by a wireless medium to perform distributed sensing tasks [21,1]. These networks are employed in different applications such as environmental and health monitoring, surveillance, and security. An important aspect of WSNs comes from the fact that many sensors generate sensing data for the same set of events. The integration of sensing, signal processing, and data communication functions allows a WSN to provide a powerful platform for processing data collected from the environment.

WSNs diverge from traditional networks in many aspects due to a large number of nodes with strong energy restrictions and limited computational capacity. In general, WSNs demand self-organizing features, i.e., the ability of

J. Schönwälder and J. Serrat (Eds.): DSOM 2005, LNCS 3775, pp. 206–219, 2005.

autonomously adapt to the changes resulted from external interventions, as topological changes (due to failures, mobility or node inclusion), reaction to a detected event, or requests performed by an external entity.

The objective of such network is to collect and process data from the environment and send it to be further processed and evaluated by an external entity connected to a sink node (or gateway). Consequently, routing towards the sink node is a fundamental task and different algorithms have been proposed to this purpose [2]. However, different applications and scenarios demand algorithms with different features. Thus, given a specific scenario, the WSN can be designed to operate with the most appropriated routing algorithm, which can be defined *a priori*. However, in some cases the variations of these scenarios can be constant or even unpredictable. For example, an event-driven scenario may present a low incidence of events and, at a given moment several events can be detected generating a high traffic. Thus, a WSN should support autonomic solutions appropriated for different periods, since it might be infeasible or undesirable to an external entity to act dynamically on the network in order to adapt its behavior.

Adaptive and hybrid approaches for routing in WSNs consist of viable solutions to deal with variable scenarios (e.g. [23,7,22]); but, they generally provide rigid solutions for some specific cases. Policies are a well-known approach in network management that give a high-level and flexible way to realize management tasks [24]. This work shows how policies can effectively be used in autonomous WSNs for the task of routing. Further, it is illustrated an adaptive hybrid case for the autonomous selection of the better routing strategy taking into account both network conditions and application requirements. Simulation results show that the usage of policies for adaptive routing in WSNs can provide resource savings and benefits.

The rest of the paper is organized as follows. Section 2 extends the discussion on routing in WSNs and the usage of policies found in related proposals. Section 3 considers the conception and implementation of policies on these networks. Section 4 describes case studies of adaptive routing using policies, and the associated results are discussed in Section 5 showing the advantages of this approach. Section 6 presents our final considerations and future work.

2 Routing and Policies

In this section, we briefly discuss the main approaches to routing in WSNs and the usage of policies on networks.

2.1 Routing in WSNs

Routing in WSNs differs from traditional networks in many aspects. Essentially, energy efficiency is the main requirement in such constrained-resource networks and the routing activity should also consider it. Further, the basic goal of a WSN is to collect and process data from the environment and send it to be processed and evaluated by an external entity. Thus, routing towards the sink node is a

fundamental task and different algorithms have been proposed [2], each one of them being more suitable for a given case or scenario due to its specific features.

Basically, we found the following classes of protocols with respect to routing infrastructure building for WSNs: *Flooding or Gossiping* are classical mechanisms to forward data in sensor network that do not need to maintain any routing infrastructure or topology [8]. In the flooding mechanism, every node forwards data by broadcasting it to all its neighbors until it reaches the destination. Gossiping differs from flooding by choosing random nodes to forward the data. Although these mechanisms disable the cost of route creation and maintenance, they cause the data packet implosion problem which represents an excessive cost for WSNs. *Proactive protocols* are routing algorithms that create and maintain the routing infrastructure no matter the network behavior. In general, this process is realized by the destination nodes. Examples of proactive protocols are DSDV [20] for MANETs and various tree-based protocols for WSNs such as One-Phase Pull Diffusion [9], SAR [26], and some implementations in [28]. Although this approach can result in a better routing process, it has the disadvantage of a constant resource consumption. *Reactive protocols* are routing algorithms that build the routing infrastructure only when a node wants to transmit a packet, i.e., it is a source-based approach. AODV [19] is a well-known protocol for ad hoc networks and Push Diffusion [9] is an example for WSNs with such behavior. This approach turns resource savings in possible inactivity periods, but it may cause the overhead of path discovery for each source node.

Depending on the application, routing can be performed considering different models [27], such as continuous, event-driven, and observer-initiated. Continuous and observer-initiated models are favorable for proactive protocols, whereas event-driven are adequate for reactive ones. Thus, given a fixed scenario, the WSN can be designed to operate using the most adequate routing algorithm defined *a priori*. But, the application requirements may change as well as may exist scenarios where the behavior of the network varies in an unpredictable way along the time. Such situations aim different (pro-active, reactive) algorithms at different instants, becoming infeasible or undesirable to an external entity to act dynamically on a network to change its behavior. Given these scenarios, we should design routing protocols based on autonomic principles.

Adaptive and hybrid approaches for routing in WSNs are viable solutions to deal with variable scenarios. For example, in [7] we considered an event-driven scenario previously illustrated, where the data traffic is monitored in time intervals and a reactive or proactive behavior is taken if this traffic stays below or above a specified threshold, respectively. In [23], Shen et al. presented a hierarchical architecture with a query language such that different routing strategies can be taken autonomously according to the application requirements and the query nature. However, these solutions treat some specific cases previously considered by the designers. Our intention in this work is to discuss a flexible way to implement an adaptive hybrid routing solution using policies.

2.2 Policies on Networks

Policy-based systems have been studied mainly in management tasks for distributed systems and networks [24]. Policies can be specified as rules governing the choices on the behavior of a system, allowing changes in such system without the requirement of rebulding it. Thus, such systems must support dynamic update of policy rules interpreted by distributed entities to modify their behavior.

Different proposals have been presented in the literature on specification and deployment of policies, such as languages, frameworks and deployment models. Typically, policies are established by low-level specific rules which consider individual configuration parameters. However, high-level abstractions, through the specification of system goals, may turn easier the administrator task. For example, the framework described in [12], which is contextualized in autonomic computing, interrelates three types of policies: Action Policies, which dictate the actions that should be taken whenever the system is in a given current state, typically in the form of "IF(Condition) THEN(Action)" clauses; Goal Policies, which specify either goals for a desired system state or criteria that characterize them, rather than specifying exactly what to do in the current state; and Utility Function Policies, which define an objective function that expresses the value of each possible state, thus providing a more fine-grained and flexible specification of behavior than the Goal Policy.

Action Policies compose the low-level specific rules, and the other policy types are high-level abstractions which must be translated in a sequence of specific rules to be executed by the system components. Clearly, policies can be used in the design of WSNs. They establish a way to formalize the actions that will be taken from local interactions among network elements or from network perceptions. Flexibility is provided with the redefinition capacity of policies. Thus, new local rules can be established due to a new global goal or an unpredictable situation, for example.

Regarding network routing, some studies [4,5] propose the use of policies in order to obtain cost savings and quality of service (QoS) in the routing task. In this work, we deal with specific characteristics and solutions of WSNs.

3 Policy-Based Adaptive Routing in WSNs

We propose the use of policies in the task of adaptive routing in WSNs. The objective is to provide a flexible mechanism to define autonomous routing operation to achieve a better efficiency. Policies must establish rules to dynamically act on WSNs based on analysis of collected data. In the routing context, these actions can trigger changes in the routing strategy, such as an adaptive solution that chooses dynamically either a reactive or a proactive behavior based on the measured traffic [7], or still in the case of a proactive tree [29], change the parent selection strategy according to a specific routing metric.

Generally, existing adaptive routing solutions for WSNs are rigid and do not allow changes in their adaptive rules. The great advantage of a policy-based solution is its possibility to be redefined by the management entity for matching

new application requirements, fixing or improving the network performance due to an unpredictable situation.

3.1 A General Model

A generic model for the use of policies in WSNs is shown in Fig. 1. The main idea is to separate basic routing mechanisms, difficult to implement and change, from the strategy of the adaptive process, usually defined by high-level rules, which are easy to build and can lead to better performance. In fact, basic routing mechanisms, or even the entire node, can be reprogrammable to allow the desired flexibility (e.g. dynamic code load [11]). But, such process can be highly complex for a management entity or demand more computacional resources from nodes to update and execute larger and more complex dynamic code.

Nodes that run policies must contain a module called *policy processor*. This module provides to the network nodes the ability to store, interpret, and execute specific policies. It can be a virtual machine that runs a script, the capacity of loading code dinamically or simply the execution of a parameter assignment, depending on how policies are defined and implemented in a given architecture. These policies can interact with the applications and other nodes in order to achieve their goals.

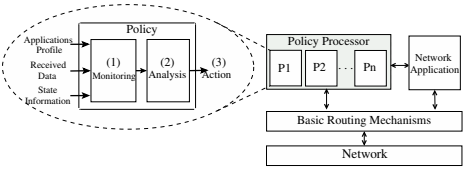

Fig. 1. A generic model for policies in WSNs

Policies must include functionalities to monitor, analyse, and act on a network. The monitoring phase (1) uses both data and state information collected from the network to analyze and use it in future actions. The profile (requirements) of each application running on the network can also be taken into account when making an action decision. The analysis phase (2) may use techniques from data fusion, intelligent agents, such as accounting, comparison with thresholds, prediction or inference techniques, and others to better detect situations where an action is needed (3).

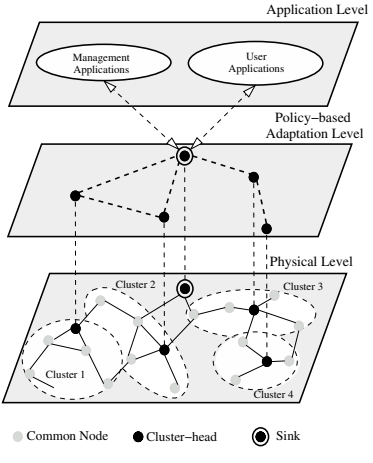

Fig. 2. Hierarchical structure

This policy model can be applied to an individual network node, determining its own routing strategy based on its network perceptions, but it may be very difficult to keep the consistency

of the entire network to perform the cooperative data collection task. Thus, this model is more viable only to strategic nodes, as sink nodes, which have more visibility of the network for monitoring and can take a proper autonomous monitoring decision for the whole network.

Hierarchical structures are very common in the network management and, in particular, in WSNs through clustering [10,14,13], allowing the grouping of related nodes for some collaborative task, minimizing the number of message transmissions, providing scalability, and dividing the complexity of the network management in sub-domains. Thus, such organization is also considered in the policy-based routing as depicted in Fig. 2. In the physical level, clusters are composed of common nodes coordinated by a *cluster-head* node. These cluster-heads usually compose an intermediate level where policies can be defined to provide the better routing strategy or parameter optimization in different moments for each cluster. Such policies follow the model previously described and use network perceptions collected from the cluster to choose its routing strategy. In the top level, applications, often external to the network, enable users and management entities to interact with the network, access data collected from clusters, and assign policies to different nodes in the network. The interaction between external entities and nodes happens through the sink node.

This hierarchical scheme for adaptive routing can provide resource savings and benefits. In particular, different parts of large scale networks are subjected to different kinds of queries (determined by the applications), different traffic characteristics and environment influence, which may cause topological changes on network. Thus, the creation of various adaptive subdomains allows the adoption of the best routing strategy for each one of them.

3.2 Implementation

A Basic Routing Mechanism Set. We propose the use of flooding, proactive and reactive routing mechanisms since they enable a WSN to operate in the three forms as described in Section 2.1. *Flooding* is fully infrastructureless and does not require any action from the receiver; thus, all data are propagated by broadcasts. The *proactive* mechanism, used is the classical tree-based approach that enables periodical constructions of a routing tree rooted at the destination node to create paths towards every network node. An implementation where the parent adoption strategy of a node is based on the firstly tree-build message received from its neighbors, called EF-Tree (Earliest-First Tree), is used in this work. The *reactive* mechanism adopted is based on the AODV behavior [19], where the routing process is triggered by source nodes only when a data is available. The routing process starts with a data flooding on the network. When a data message arrives at the destination node, that node will respond with periodical requisition messages to the neighbor node that sent the first data message. Thus, the response is recursively propagated in the same manner towards the data source node, establishing a route and inhibiting new floodings. The SID (Source Initiated Dissemination) protocol [7], which also provides similar behavior, is used in this work.

In the mechanisms described above, we note that the routing infrastructure starts at the destination node, generally the sink node in WSNs. To apply these mechanisms in a transparent way, we propose an integrated forwarding rule to change the routing algorithm based on the destination node decision, which can be defined by a policy. The rule is defined as follows: every data packet (generated or to be forwarded) is sent through a routing tree if it exists and it is valid (EF-Tree behavior); otherwise, the data is sent to the neighbor in which a specific and valid requisition was received (SID behavior); In latter case, data is sent by flooding. All nodes respond to control messages as their original protocols determine, but the validity is ensured by a timestamp and a predefined time period.

With this forwarding rule, anticipated tree-building messages (as EF-Tree) are used to achieve a proactive behavior. Such messages can also carry commands or queries to define data collection parameters (e.g., continuous collection and data rate). In the absence of proactive messages, nodes operate in an event-driven mode. Previously configured parameters characterize the events that need to be notified. When such events occur, the source nodes flood their data on the network. If the receiver desires to create a reactive routing infrastructure, it responds using the SID algorithm. If flooding is preferred to be maintained, the receiver only needs to receive data without any reaction.

Policy Implementation. There are several forms to implement policies in distributed systems and networks (Section 2.2). Scripts are preferable because they are codified in high-level languages, which give more versatility and power to the administrators. Further, there are solutions based on scripts to WSNs, such as the TinyScript of the Maté platform [15], found in the Mica2 architecture [6]. TinyScript is a high-level script language that is compiled to a bytecoded version to be distributed to the sensor nodes and executed in a specific virtual machine. This encoding allows resource savings and simpler interpretation on distribution and execution, which is needed in this kind of network.

```
privatecounter;
counter=getNumNodes();
if counter > THRESHOLD then
  takeAction(PROACTIVE);
else
  takeAction(REACTIVE);
endif
```

(a) Policy script

```
getnn           // read number of nodes
pushc THRESHOLD // puts THRESHOLD on operand stack
inv             // invert the THRESHOLD
add             // number of Nodes − THRESHOLD
blez 8          // if ((NN − THRESHOLD) > 0)
                // go to proactive
pushc REACTIVE  // puts REACTIVE on operand stack
act             // take a REACTIVE action
halt
pushc PROACTIVE // puts PROACTIVE on operand stack
act             // take a PROACTIVE action
halt
```

(b) Bytecode encoding

Fig. 3. Policy implementation

An implementation of the routing rule of Fig. 4, described in Section 4, using Maté's specification is exemplified in Fig. 3. It shows a simplified script,

Fig. 3(a), and its bytecoded version, Fig. 3(b), configured to check periodically the number of nodes sending data in a cluster, `getNumNodes()` function, in order to select the routing strategy empsloyed (`PROACTIVE` or `REACTIVE`). If such value is higher or lower than a threshold, a proactive or reactive strategy is assumed, respectively, using `takeAction` (⟨type⟩) function. Both functions must be built in the application to be executed on the node together with the Maté virtual machine.

A limitation of the Maté platform is the restrict set of instructions, reducing the policy codification possibilities. More powerfull script approaches exist for WSNs, such as the Sensorware platform [3], however it requires a more complex interpreter in the nodes and increases the number of data bytes transmitted on the network.

Policy Deployment. An advantage of the policy-based adaptive routing is the flexibility to redefine the rules when the network is in operation and based on the knowledge acquisition. Some strategies must be used to deploy them in the nodes. A flooding scheme is a simple solution to reach all nodes, although it may add a significant over-head for WSNs. We expect policy implementations to be small (simple rules) and their redefinitions not a frequent task. But, robust mechanisms are required to keep the network in a consistent state, since a new node or a sleeping one can be (re)joined to the network with an older policy version. Some solutions have proposed energy-efficient and robust code update mechanisms for WSNs, such as Trickle [16], which can be applied in our proposed approach.

4 Case Studies

Next we present two case studies showing the use of policy-based adaptive routing. The first case considers a scenario where different routing strategies are applied autonomously according to a policy definition. The second case deals with an adaptive parameter set instead of a constant default value. In both cases, the goal is to achieve a better performance regarding energy consumption, a constrained resource in WSNs.

4.1 Changing the Routing Strategy

We consider an unpredictable scenario where traffic variations may occur or the application requirements can change along the time. In such cases, different routing strategies may be more appropriated for different moments. Thus, an adaptive rule can be applied to achieve a hybrid behavior by switching among different routing strategies according to the monitored network condition. For example, in an event-driven scenario, the network may remain with a very low activity for days, favoring a reactive algorithm. But, in a given moment, a number of events may occur, generating a traffic large enough to use a proactive algorithm.

For this scenario, we have proposed an adaptive hybrid algorithm which determines the best routing strategy (reactive or proactive) based on the number of sources sending data [7]. This number can be counted observing the different source node identifiers (ids) and storing them in a bit-array to save memory, for example. A threshold used by the sink node evaluates the measured network traffic (number of source nodes sending data). If the traffic is lower than the given threshold, a reactive strategy is adopted; otherwise, a proactive strategy is taken. A rule that represents this routing strategy is shown in Fig. 4.

Although this simple implementation can lead to better performance than the specific algorithms, the traffic measurement does not show the advantage of a proactive action if a threshold is reached but no new detections happen. For example, occasional distribution of event detections can lead to a concentrated measurement, higher than the established threshold, leading to a proactive behavior, which may be not necessary.

As a better adaptive model, we consider the use of the simple signal processing method of Moving Average Filter (MAF) [25] on the number of nodes that are starting to detect events. Thus, a detection is counted observing when a source node starts to send data after its innactivity. The filter smoothes possible measurement noises and captures the seasonal component of event-occurrence on the last n monitoring periods (n is the filter window size). Hence, the filter can give a good estimation for the next time period, helping in the decision to take a proactive behavior or not. Assuming the basic mechanisms described before, if more than one new source is expected for the next period, the proactive behavior can be used to build the routing infrastructure for the entire network with the cost of a route discovery. Fig. 5 shows a routing strategy rule for this solution.

```
if Num_of_Sources > THRESHOLD then
  proactive_action();
else
  reactive_action();
endif
```

Fig. 4. Routing rule 1

```
Next_Est=MovingAverage(Num_of_Detections);
if Next_Est > 1 then
  proactive_action();
else
  reactive_action();
endif
```

Fig. 5. Routing rule 2

```
if Num_of_Sources < old_Num then
  proactive_action();
endif
old_Num = Num_of_Sources;
```

Fig. 6. Optimization rule

4.2 Adaptive Routing Optimization

Often, routing algorithms for WSNs define different parameters that must be configured according to a given application and network situation to achieve a better performance. But if the network condition changes, a policy can detect it and set dynamically the better routing parameter. For example, in routing solutions presented in Section 2.1, it is common the use of periodical updates to support topological changes caused, for instance, by failures. Pre-defined frequent updates can be set to a scenario with common failures, but this configuration is not appropriated to a stable scenario with a low failure rate due to the consequent communication overhead. If it is possible to determine the failure occurrence rate, a policy can dynamically change this routing parameter,

or even define when to take the corrective action, which optimizes the routing performance.

In this case study, let us consider that a given application starts requiring continuous environmental measurements. Thus, a proactive routing strategy such as EF-Tree is more adequate. Instead of setting a constant periodical rebuilding rate, in [17] we apply data fusion techniques in a solution that monitors the traffic and infers about failure occurrence to take this action only when necessary to save energy. This failure detection is possible by assuming a continuous traffic, thus, downsteps in the received traffic mean a failure possibility. In this work, we build a simplified similar rule, shown in Fig. 6, in which the number of nodes sending data (counted as described in Rule 1) is observed in the periods of the data generation rate. A reduction of this number means a failure, which triggers a tree building to recover it.

5 Simulation and Evaluation

We evaluated the viability of our policy-based scheme through simulations. Experiments were performed using the ns-2 simulator [18]. The simulation parameters were based on the Mica2 node [6] using the 802.11 protocol in the MAC layer. Through its datasheet, bandwidth is 19200bps, radio range is 40m and Tx. and Rx. power consumption are 45 and 24mW, respectively. In all simulations, we assumed a network size of 50 nodes randomly distributed in an area of 100×100 m^2 with only one sink, and transmission of data and control messages (tree-building or requisition messages) of 20 bytes every 10 s and 100 s, respectively. The graphics show the summary result of 33 simulations and the error bars (vertical bars) represent the confidence interval of 95%. The evaluated metric was the total energy consumed by all sensor nodes. Simulation results consider scenarios and routing rules described in Section 4. For simplicity, we embedded the basic routing mechanisms and the policies at sensor nodes.

5.1 Changing the Routing Strategy

In a non-correlated event-driven scenario, we varied the number of sources generating data randomly, with a uniform distribution, along the simulation time of 4000 s. A policy equivalent to the routing rule of Fig. 4 was set to run in intervals of 10 s, the same period of data messages, and the traffic collected in an n-interval is compared with a static threshold in order to set the strategy activated in the $(n + 1)$-interval. Thus, it is expected that different thresholds lead to different results, as in Fig. 7, where it is shown the behavior of a network running with EF-Tree and SID algorithms alone, which compose the basic routing mechanism set described in Section 3.2, as well as the Policy-Based scheme with traffic thresholds (limit of the number of source nodes sending data to take an adaptive behavior) of 1, 2 and 3, called *PB-1*, *PB-2* and *PB-3*. This scheme autonomously adapts the routing mechanisms during the simulation, as described before. A threshold higher than 3 leads to a performance very close to

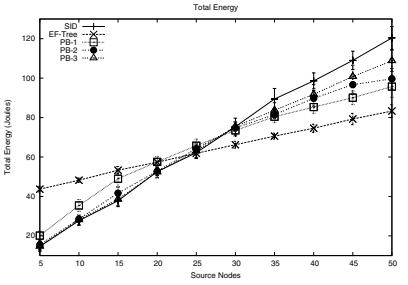

Fig. 7. Energy with routing rule 1

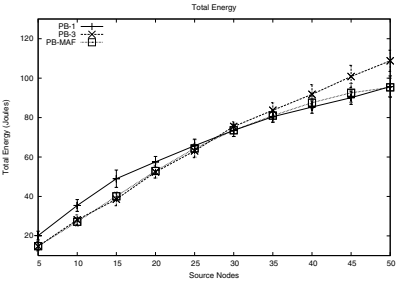

Fig. 8. Energy with routing rule 2

SID, because proactive behavior is rarely taken with the simulated parameters, thus their results were omitted.

The results for PB-2 and PB-3 are closer to the SID algorithm when the number of source nodes is less than 30. When a low traffic is measured, such policies do not assume the proactive behavior. Some differences are due to the random nature of the traffic, because source node data generations can be concentrated in a time interval without implying in a high occurrence ratio. In these cases, the performance of EF-Tree and PB-1 are not so good because they assume an unnecessary proactive behavior. But, when traffic increases, the EF-Tree and PB-1 have a better performance than PB-2 and PB-3 because their proactive behaviors avoid the cost of creating a routing infrastructure for new sources. PB-1 never reaches the EF-Tree performance for situations of high traffic since it always starts with a reactive mode, before changing to a proactive behavior.

In the previous results, the management entity perceives that the chosen policy is not so precise in the traffic evaluation. But, we can have advantages with the flexibility of the policy redefinition. For instance, that policy may be replaced by the one found in Fig. 5 that applies a Moving Average Filter (MAF). Fig. 8 shows the results of the new policy called PB-MAF. We can see that the PB-MAF results are closer to the best performance of PB-1 and PB-3, thus it is a more adaptable solution to respond to traffic changes by autonomously adjusting to its behavior.

5.2 Adaptive Routing Optimization

In this case study, we consider a continuous traffic scenario where the rule of Fig. 6 is applied to detect failures and rebuild the routing tree. We fixed the number of source nodes in 20, randomly chosen to send their data periodically towards the sink from the beginning to the end of the simulation. We varied the node failure ratio from 0 to 0.012 failures per second randomly distributed during the simulation. When a node fails, it stays inactive until the end of simulation. Fig. 9 shows the delivery rate improvement of the adaptive rule solution (EF-Tree adap) compared with the EF-Tree with fixed rebuildings (EF-Tree orig). It shows the advantage of failure detection to take an adaptive behavior. Fig. 10 shows the relative energy usage of the solutions. We note the better performance

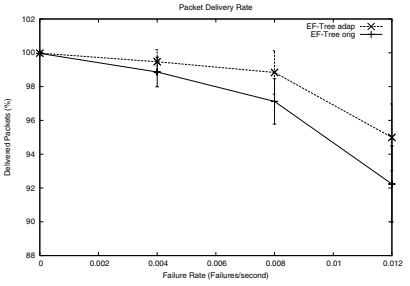

Fig. 9. Delivery rate with optimization rule

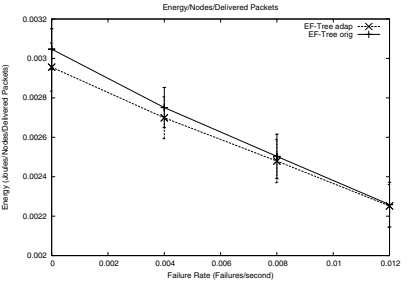

Fig. 10. Energy usage with optimization rule

of the adaptive solution when the failure rate is low due to rare tree rebuildings. However, with higher failure rates, the energy usage between the adaptive and original solutions approximates because more tree rebuidings are needed by the latter to maintain the delivery rate.

5.3 Policy Redefinition Cost

The advantage of redefining a policy depends on its gain versus the cost of its redefinition and redistribution. To take an idea of this cost, we assumed a byte-coded script of 20 bytes (large enough to store the script of Fig. 3(b)), totalizing an estimated packet size of 36 bytes with the control header. We used the classical flooding (broadcasts in the network) to redistribute policies to the entire network. The energy cost of a single distribution was 1.29 Joules, which is compensated by the difference of the PBs performance with different thresholds or with the dynamic one. In addition, this advantage may be higher if the network remains more time in the new situation.

5.4 Hierarchical Structure Evaluation

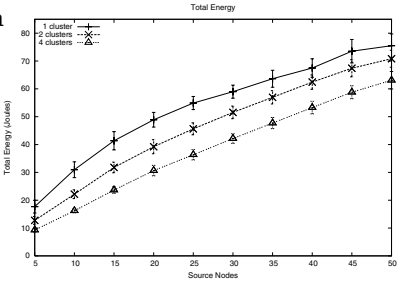

Fig. 11. Energy with 1, 2 and 4 clusters

To show the gain of a hierarchical scheme and the independent application of policies, we created a scenario like the event-driven where the sensing area is divided in quadrants defining separated clusters. We compare the performance of the PB scheme running with 2 and 4 clusters with a scenario with no clustering. Fig. 11 depicts the energy consumed by the algorithms with different number of source nodes. We observe a better performance as the number of clusters grows since independent actions are taken by their cluster-head's policies. An event may be restricted to a location and its detection in a partition does not imply in its detection in another area. Thus, a high

traffic in a partition activates the proactive behavior only inside itself, whereas other partitions may keep a reactive behavior saving their resources.

6 Final Considerations

In WSNs, routing algorithms are specific for the scenario of a given application. In scenarios of high variability, a given routing algorithm cannot achieve its best performance all the time. Thus, adaptive hybrid approaches, providing autonomic behavior, can be better than a single algorithm. This work discussed the usage of policies to establish adaptive routing rules for WSN elements to become more flexible and accessible development and maintenance tasks of a network. Although we focused on low-level policies, they are necessary for high-level policy building. Case study scenarios revealed that the usage of policies in autonomous WSNs can provide resource savings.

The next step of our work is to enable and evaluate the implementation of policy-based routing rules on real sensor nodes. Future work includes the implementation of policy-based routing rules to change the routing QoS metric, and the use of policies to dictate how the data is collected, aggregated and forwarded through the routing infrastructure with resource savings.

References

1. I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cyirci, *Wireless sensor networks: A survey*, Computer Networks, 38 (2002), pp. 393–422.
2. J. N. Al-Karaki and A. E. Kamal, *Routing techniques in wireless sensor networks: a survey*, IEEE Wireless Communications, 11 (2004), pp. 6–28.
3. A. Boulis, C.-C. Han, and M. B. Srivastava, *Design and implementation of a framework for efficient and programmable sensor networks*, in MobiSys, May 2003.
4. I. CISCO Systems, *White paper: Policy-based routing*, Access: May 2005. [Online] Available: http://www.cisco.com/warp/public/cc/techno/protocol/tech/.
5. D. Clark, *RFC 1102: Policy Routing in Internet Protocols.* MIT Lab for Computer Science, 1989.
6. Crossbow, *Mica2 platform*, Access: February 2004. [Online] Available: http://www.xbow.com/.
7. C. M. Figueiredo, E. F. Nakamura, and A. A. Loureiro, *Multi: A hybrid adaptive dissemination protocol for wireless sensor networks*, in Algosensors, vol. 3121 of Lecture Notes in Computer Science, Turku, Finland, July 2004, Springer, pp. 171–186.
8. S. Hedetniemi and A. Liestman, *A survey of gossiping and broadcasting in communication networks*, Networks, 18 (1988), pp. 319–349.
9. J. Heidemann, F. Silva, and D. Estrin, *Matching data dissemination algorithms to application requirements*, in 1st SenSys, Los Angeles, CA, USA, 2003, ACM Press, pp. 218–229.
10. W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, *Energy-efficient communication protocols for wireless microsensor networks*, in 33rd HICSS, Maui, Hawaii, USA, January 2000.

11. C. T. Inc., *Mote in-network programming user reference*, Access: August 2004. [Online] Available: http://webs.cs.berkeley.edu/tos/tinyos-1.x/doc/xnp.pdf.
12. J. O. Kephart and W. E. Walsh, *An artificial intelligence perspective on autonomic computing policies*, in 5th Int'l Workshop on Policies for Dist'd Systems and Networks, 2004.
13. M. Kochhal, L. Schwiebert, and S. Gupta, *Role-based hierarchical self organization for wireless ad hoc sensor networks*, in Proc. of the 2nd ACM Int'l Conf. on Wireless Sensor Networks and Applications, ACM Press, 2003, pp. 98–107.
14. R. Krishnan and D. Starobinski, *Message-efficient self-organization of wireless sensor networks*, in IEEE WCNC 2003, March 2003, pp. 1603–1608.
15. P. Levis and D. Culler, *Maté: A tiny virtual machine for sensor networks*, in 10th Int'l Conf. on Architectural Support for Prog. Lang. and Operating Sys., ACM Press, 2002, pp. 85–95.
16. P. Levis, N. Patel, D. E. Culler, and S. Shenker, *Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks.*, in 1st NSDI, 2004, pp. 15–28.
17. E. F. Nakamura, C. M. Figueiredo, and A. A. Loureiro, *Information fusion for data dissemination in self-organizing wireless sensor networks*, in 4th ICN, April 2005.
18. NS-2, *The network simulator - ns-2*, Access: February 2004. [Online] Available: http://www.isi.edu/nsnam/ns/.
19. C. Perkins, E. Belding-Royer, and S. Das, *Ad-hoc on-demand distance vector routing*. RFC 3561, 2003.
20. C. Perkins and P. Bhagwat, *Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers*, in ACM SIGCOMM'94, 1994, pp. 234–244.
21. G. J. Pottie and W. J. Kaiser, *Wireless integrated network sensors*, Communications of the ACM, 43 (2000), pp. 51–58.
22. V. Ramasubramanian, Z. Haas, and E. Sirer, *SHARP: A hybrid adaptive routing protocol for mobile ad hoc networks*, in 4th MobiHoc, 2003, pp. 303–314.
23. C. Shen, C. Srisathapornphat, and C. Jaikaeo, *Sensor information networking architecture and applications*, IEEE Personal Communication, 8 (2001), pp. 52–59.
24. M. Sloman, *Policy driven management for distributed systems*, Journal of Network and Systems Management, 2 (1994), pp. 333–360.
25. S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*, California Technical Publishing, San Diego, CA, USA, 2nd ed., 1999.
26. K. Sohrabi, J. Gao, V. Ailawadhi, and G. Pottie, *Protocols for self-organization of a wireless sensor network*, IEEE Personal Communications, 7 (2000), pp. 16–27.
27. S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman, *A taxonomy of wireless micro-sensor network models*, ACM Mobile Computing and Communications Review (MC2R), 6 (2002), pp. 28–36.
28. A. Woo, T. Tong, and D. Culler, *Taming the underlying challenges of reliable multihop routing in sensor networks*, in 1st SenSys, ACM Press, 2003, pp. 14–27.
29. C. Zhou and B. Krishnamachari, *Localized topology generation mechanisms for self-configuring sensor networks*, in IEEE Globecom, San Francisco, USA, December 2003.