Proactive Reconfiguration of Wireless Sensor Networks

Marcel Steine[1], Cuong Viet Ngo[2], Ramon Serna Oliver[2], Marc Geilen[1],
Twan Basten[1,4], Gerhard Fohler[2], Jean-Dominique Decotignie[3]
[1]Department of Electrical Engineering, Eindhoven University of Technology, The Netherlands
[2]Chair of Real-Time Systems, Technische Universtät Kaiserslautern, Germany
[3]Real-Time Software and Networking, CSEM, Neuchâtel, Switzerland
[4]Embedded Systems Institute, Eindhoven, The Netherlands
m.steine@tue.nl

ABSTRACT

Network dynamics, such as mobility and increase in network load, can influence the performance of a Wireless Sensor Network (WSN). In this paper, we introduce a method which exploits design-time knowledge of the application scenario dynamics to construct a *proactive* run-time reconfiguration approach. The approach anticipates for the impact that predefined dynamic events can have on the performance of the WSN by switching between various *modes* of operation defined at design-time. A mode defines the values for the controllable parameters of the network protocol stack. Our approach explicitly differentiates between parameters that can be adapted locally, per node, and those that should be considered globally for the whole WSN. Design-time definition of modes results in a very low run-time overhead as we only require detection of the mode to use and a low overhead synchronization to change global parameters. The approach is made robust by using a recovery approach for nodes unaware of their global mode after, for example, (re-)joining the network. Experiments with an office monitoring deployment and extensive simulations of a cow-health monitoring scenario show that our approach can easily be adopted by practical WSN deployments and results in a significant reduction in resource usage, e.g., power consumption in our examples, at a very low run-time overhead cost.

Categories and Subject Descriptors: C.2.1 [Network Architecture and Designs]: Wireless communication; C.2.4 [Computer-Communication Networks]: Distributed Systems; C.4 [Performance of Systems]: Performance attributes.
General Terms: Algorithms, Design, Performance.

1. INTRODUCTION

A Wireless Sensor Network (WSN) is a distributed network of small autonomous devices that are capable of sensing, processing and wireless communication. These devices, commonly called nodes, contain one or more sensors and are constrained by stringent resource limitations. They have to collaborate to perform a task specified by the end-user of the WSN. A typical example is a smart building scenario, where information from nodes is used to make smart decisions on, for example, energy management.

With the recent inclusion of WSNs in many different practical deployments, the impact of network dynamics, such as variation in network load and node mobility, on the behaviour of a WSN becomes more prominent. This dynamism should be taken into account when designing a WSN to ensure that the performance, i.e., Quality-of-Service (QoS), is sufficient to perform the required task, while optimizing the resource usage. The controllable parameters of the network protocols used by the nodes, for example the radio transmission power and MAC duty cycle, play an important role in how the network behaves, the resulting QoS it can provide, and the resources required. Currently, parameter values are often fixed or adapted at run-time in response to locally observed performance.

Fixing the value for the parameters at design-time, which is current practice for most WSN deployments, requires to anticipate for the worst-case dynamics of the network to ensure the required QoS at all times. This can result in a conservative selection of parameter values and QoS over-provisioning during the times the network is not experiencing its worst-case dynamics. Over-provisioning can result in a superfluous use of resources, such as the power consumed by nodes. To ensure a good performance for, for example, the monitoring of employees in an office building, we would have to configure the network for the maximum number of persons to monitor when using only single parameter values. This could mean that, for instance, the time the radio can spend in its power efficient idle state is limited to be able to deal with the worst-case amount of data. When there are only a few or no persons in the building, radio time is reserved but not needed, leading to an unnecessary use of battery power.

Adapting the parameter values at run-time provides more freedom to tune the behaviour of the WSN to the experienced dynamics and reduce this waste of resources, while QoS is still met. *Run-time reconfiguration* approaches define when and how parameters should be adapted. Various specialized protocols have been developed that include reconfiguration by adapting their own parameters in response to particular changes in the locally observed performance [3, 5, 8, 15, 18, 21]. With the use of run-time techniques, such

MSWiM'11, October 31–November 4, 2011, Miami, Florida, USA.

as feedback control [3, 8, 15, 21], or machine learning [5, 18], these approaches converge to parameter values suited for the current situation. These kind of approaches are *reactive*; they only adapt parameters after a local change of performance has been observed. This may result in a (long) phase, between the occurred dynamics and required change of parameters, in which the performance of the network might be unacceptable or resources might be wasted. On the other hand, too quick responses may lead to unnecessary adaptations to temporary glitches and changes and possibly to instability. Furthermore, current approaches typically only consider adaptation of parameters that can be changed locally. Additional parameters exist with an important impact on the network performance, which should change only in a coordinated manner for all nodes in the network. Moreover, the adaptation techniques used by these specialized approaches are not generally applicable to other protocols and induce a (possibly large) run-time overhead.

Existing reconfiguration approaches are defined independent of the application scenario. The design-time knowledge of the scenario can help in the design of a run-time reconfiguration approach, by anticipating for upcoming dynamics potentially impacting the QoS of the network. The closing of an office building will indicate the presence of a lot less employees to monitor. A person starting to walk will impact the mobility of attached nodes. By detecting these kinds of events we can *proactively* adapt parameters to deal with the (upcoming) effects of the dynamics on the QoS and avoid phases with insufficient QoS or waste of resources. With design-time knowledge of the frequency of events we can furthermore more accurately assess the option of changing global parameters, which can have a strong impact on QoS, but requires synchronization overhead in exchange.

In this paper, we introduce a method to exploit design-time knowledge of application scenario dynamics to construct a proactive run-time reconfiguration approach. At design-time, the WSN designer defines modes of operation, defining the values of the controllable parameters of the network stack, to be adapted in response to expected and detectable scenario dynamics. We explicitly distinguish parameters that can be adapted locally and those that should be considered globally. We therefore define a global mode of operation, equal for every node in the network, and a local mode, which can be different between nodes. These modes are selected by keeping in mind the benefit of and overhead involved in using them at run-time.

At run-time, nodes locally observe the design-time defined events. An event either triggers the change of a local mode or is communicated to a centralized location, for example as part of a distributed detection of a global event. A centralized location, such as a sink, initiates the change of global mode by communicating to all nodes in the network. Due to the design-time definition of the modes, a simple, low overhead flooding approach suffices. Our reconfiguration approach is made robust for practical deployments by a recovery approach that allows nodes to determine the currently used global mode after, for example, (re-)joining the network or missing a request to change a global parameter.

Experiments with an office monitoring deployment and extensive simulations of a cow-health monitoring scenario show the practical applicability of the method. It is shown that, compared to using a single set of parameter values or a reactive reconfiguration approach, a significant reduction in power consumption of the nodes can be obtained for these scenarios, while the QoS is maintained.

The remainder of this paper is organized as follows. In Section 2 related work is discussed. Section 3 discusses our proactive run-time reconfiguration approach in more detail. In Section 4 we show how the method can be applied to two practical scenarios. In Section 5 we discuss the results of the integration of the proactive reconfiguration approach with the use of extensive simulations and experiments with an actual deployment. Section 6 concludes.

2. RELATED WORK

The run-time reconfiguration approach as constructed with the method suggested in this paper, differentiates itself from existing related work by the combination of its (i) proactive approach as opposed to the common reactive approach, (ii) use of both local and global parameters and (iii) independence of protocol stack and focus on practical applicability.

In our method we explicitly exploit a priori knowledge and analysis of the application scenario. This allows us to use a proactive reconfiguration approach, which anticipates on changes in application dynamics which can be deduced from observable events. In the area of real-time embedded systems the benefits of using a proactive approach is already recognized. In [6], scenarios, related to what we call modes, are defined at design-time in order to reduce resource consumption by the system by predicting in which scenario the system operates.

Existing approaches in the area of WSN are independent of the application scenario and react to the effects caused by changes in dynamic behaviour instead. These reactive run-time reconfiguration approaches can be classified as either distributed [3, 15, 21] or centralized [8]. In a centralized approach a central location, such as the sink, collects information from all nodes in the network in order to make decisions on how to reconfigure the network.

Centralized approaches do not allow for adaptation of parameters in response to frequent changes, due to the overhead involved in taking the decision and communicating it. Purely centralized approaches are therefore not suitable for most practical deployments. In a distributed approach, all nodes decide locally on how to adapt their configuration, based only on locally observed quality aspects. For example, in [3] the transmission power of a node is adapted in response to fluctuations in the quality of links or workload observed by the routing protocol. In several MAC protocols [15, 21], the duty-cycle is adapted based on the amount of traffic.

In our method we consider adapting both local and global parameters as well as the issues involved in coordinating and maintaining consistent global parameter values. Nodes (frequently) adapt their local parameters in a distributed manner, while we take a centralized approach to coordinate changes of global parameters. With our method, the overhead of the coordination can be assessed and kept under control with design-time knowledge of the frequency of events.

While many reconfiguration approaches depend on specialized adaptive protocols, of which examples are given above, our approach considers the existing protocol stack as a black box with an interface of changeable protocol parameters. This is common for design-space exploration approaches which are used to find a single set of parameter values [4, 7]. It makes our method independent of the used

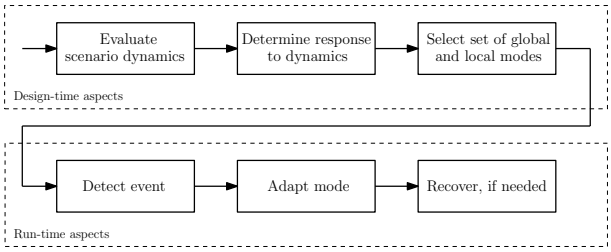


Figure 1: Method design-flow

protocol stack and allows us to change parameters for different protocol layers at the same time, increasing the applicability for new and existing WSN deployments.

The use of design-time defined modes results in a very low run-time overhead for reconfiguration, because we only need a lightweight detection of modes and switching mechanism for the global parameters. This makes it very well suitable for integration in diverse practical WSN deployments. Furthermore, the integration of a recovery approach is emphasized for the use in any practical deployment. The need of such a recovery approach is recognized [1, 14], but its current application in practice induces a large overhead. Compared to approaches as described in [1, 14] we require a smaller scale recovery of only a single (global) mode identifier instead of large code fragments, thereby limiting the overhead involved with the recovery.

3. PROACTIVE RUN-TIME RECONFIGURATION

In this section we introduce our method for designing a proactive run-time reconfiguration. We look in more detail at both the design-time and run-time aspects involved. An overview of the steps is shown in Figure 1. In general, at design-time we define different modes of operation, i.e., sets of parameter values, to be selected by the nodes at run-time, triggered by observable dynamic events, defined using knowledge of the actual environment in which the nodes are to be used. These modes are stored on the nodes locally when deployed. At run-time, we ensure that nodes are aware of the mode in which they should operate.

3.1 Design-time Mode Selection

Evaluate dynamics. The first step of our method is the classification of observable events indicating scenario dynamics that potentially lead to a change of QoS. This change may be due to the impact of the dynamics on important characteristics of the network, such as the network topology, node mobility or data being generated or handled by the nodes. From experience with actual deployments we see that these dynamics can often easily be derived using only high-level knowledge of a designer of the WSN and the target application. Detailed information on, for example, the internal functionality of the network protocols, is usually not required. For instance, when a person starts to walk after being seated, the mobility of the on-body node is expected to change. Or, at the start of a working day, the number of persons, and thereby the amount of data traffic, in an office environment is likely to increase.

Determine response. We respond to an upcoming change in the QoS by the adaptation of controllable parameters of the used network protocols. We can differentiate between two kinds of controllable parameters. A *local parameter* can be set independently for every node in the network. An example is transmission power. A *global parameter* should be set to the same value for every node in the network. An example is the radio channel or frame length for certain MAC protocols. Changing a global parameter requires synchronization to maintain proper functioning of the WSN. Since a global parameter affects the behaviour of every node in the network it often has a larger impact on the QoS compared to changing a local parameter and therefore plays an important role in run-time reconfiguration.

Many possible parameter values for the different protocols and complex cross-layer dependencies might exist. Run-time exploration of the parameter-space is hard and time-consuming and it cannot be done in a distributed fashion without knowledge about the rest of the network, which makes it not suitable for any practical (proactive or reactive) reconfiguration. We therefore determine the parameters at design-time. For this we rely on existing techniques, such as analytical approaches [4, 7], simulations [9, 11], or experience with practical WSN deployments. While our approach is independent of the design-time analysis techniques and controllable protocol parameters considered, they are important for the overall improvement obtained when using the suggested reconfiguration approach.

Determine modes. The next step is to determine to which dynamics we actually want to respond at run-time, keeping in mind the trade-off between the benefits of adaptation and the overhead incurred. The local detection of an event on a node is cheap but a distributed detection of a global event can be expensive. Similarly, adapting a local parameter is cheap but the adaptation of a global parameter is expensive. The frequency of events plays an important role in the overall configuration overhead and depends on the run-time execution of the scenario. However, at design-time we can already analyse the (expected or maximum) frequency of events in the given scenario. In a smart building scenario we know it will become night only once a day, while a person starting to walk might occur much more frequent (but maybe still infrequent enough to change a global parameter depending on the scenario).

The set of parameter values for every combination of defined dynamics, is referred to as the *mode* of operation. Based on the difference between changing local and global parameters we divide the operational mode of a node into a *local mode* and a *global mode*. The local mode defines the local parameter values, while the global mode defines the global parameter values.

After selecting the modes and defining their parameter values, the modes can be stored locally on every node, so that they can quickly be accessed by the node at run-time. Note that the possible combinations of modes stored may be different for every node in the (heterogeneous) network. At any moment in time a node is required to operate according to a single combination of local and global mode.

3.2 Run-time Mode Detection

To make the nodes aware of the modes to use at a given moment in time, we have the three final steps of our method as depicted in Figure 1.

Detect event. Unless the detection of the required events is already part of the application, small procedures need to

be written for the nodes to detect the events for decision making on mode changes. Events are typically detected using sensor information and/or the current time. When an event is detected that is relevant for the local mode of other nodes in the network or the selection of global modes, it should be communicated to those other nodes or a central location. For the communication of detected events to other nodes we can communicate a unique event identifier to the required nodes using the existing routing infrastructure.

Respond to event. In response to events triggering a new local mode, a node can immediately adapt its local parameters based on the locally stored mode information. For example, a node may change its transmission power without any significant overhead, assuming the protocol stack provides an interface to adapt parameters externally. For the coordinated change of a global mode we rely on a centralized location. As for the local mode change, this centralized location can decide to change the global mode based on detection of local events and/or events of one or more nodes in the network. Once decided to change the global mode, all nodes in the network should be informed.

As the parameter values to be used for every mode can be accessed locally, the centralized location only needs to communicate a predefined unique identifier of the global mode to all nodes. Because of the small amount of information to communicate, we observed that we can resort to a simple and low overhead flooding approach initiated by the centralized location to inform all nodes about the mode change. Starting from the centralized location, the mode identifier is broadcast to all of its neighbours and every node receiving the information for the first time will broadcast it thereby disseminating the information to all nodes in an iterative fashion.

Recovery approach. In a practical WSN deployment, nodes might experience heavy external interference for a longer time and may need to re-join the network. This may result in nodes being in the wrong global mode, after missing a global mode change. Nodes may also freshly join an existing network. Inconsistent global modes lead to nodes not being able to communicate, e.g., when using different radio channels, or a reduced efficiency of the communication, e.g., when using different radio sampling intervals.

Nodes should become aware of using a wrong global mode in order to start looking for the right one. When inconsistent global modes lead to nodes not being able to communicate, nodes may start to acquire the global mode after being unable to communicate for a given amount of time. In general, nodes can then cycle though their different possible global modes and send packets to their neighbours requesting for the currently used global mode. In specific cases, nodes may be able to locally observe which global mode is used and thereby avoid any communication overhead, for example by overhearing communication while changing radio frequencies in order to find the appropriate globally used radio channel.

Discovering an erroneous global mode is less obvious when nodes are still able to communicate, but less efficiently, when using different global modes. A node may not be able to make a distinction between using a wrong global mode or experiencing increased interference. In this case nodes should either regularly request the active global mode from neighbouring nodes or observe this information based on overhearing communication if possible. In this work, we want to emphasize the need of such a recovery approach for practical applications. Exploring the trade-offs involved in designing a recovery approach and the impact of the used protocol stack in more detail is considered for future work.

4. RECONFIGURATION SCENARIOS

In this section, we construct reconfiguration approaches for two practical scenarios. We start by looking at mobility monitoring of cows, based on actual end-user requirements as defined in the WASP project [20]. A second scenario is the monitoring of the location of employees as part of a smart-building scenario. We have chosen scenarios that differ in the deployment characteristics and protocol stack and we show that both allow an intuitive integration of a proactive reconfiguration approach using our method. In the next section, we discuss the experimental results of the integration for both scenarios, where we use extensive simulations for the cow-health monitoring scenario and experiments with an actual deployment for the office monitoring scenario.

4.1 Cow-health Monitoring Scenario

Health problems of cows often result in decreased mobility. Mobility monitoring leads to an early detection and treatment of health problems, improving animal well-being and lowering production losses. A WSN is proposed where a sensor node is attached to the leg of a cow [20]. For our scenario we consider the monitoring of thirty-eight cows with a single BSN [2] sensor node each in a barn of 40 by 60 meters. Twelve static BSN nodes are placed in a grid to support communication to a sink location (being a static node at one corner of the barn). When a cow is walking, acceleration is measured and sent, via the static network of nodes, to the sink for processing in order to detect mobility problems, such as limping. When a cow is stationary, the sink only needs to be informed about the posture of the cow, e.g., standing or laying, to allow activity monitoring. The goal of the deployment is to get a high delivery ratio of the information collected from the cows, while, as it is very inconvenient to regularly remove the nodes from the cows, the WSN operates as power efficient as possible.

When looking at the behaviour of the monitored dairy farm cows, we can observe that during most of the day they stay in the barn with limited mobility and twice a day, at regular times, the farmer leads them to walk to a milking parlor. This procedure can take up to two hours including the walking. With this knowledge of the application scenario we can identify several observable events of interest for reconfiguration. First of all, whether a cow starts walking or becomes stationary has an effect on both the amount of traffic as on the amount of mobility in the network. Furthermore, whether the cows are in the process of being milked or not heavily impacts the traffic load in the entire network as during this period all cows will (slowly) walk to the milking parlor. For the ease of explanation we only focus on these dynamics, but one could think of other interesting dynamics for a more fine-grained reconfiguration, such as going from day to night and vice versa or the time that cows are fed.

To find the appropriate response to the selected events we start by looking at the controllable parameters. For the protocol stack we focus on a basic TDMA MAC protocol with a fixed slot assignment. Every TDMA frame consists of an active period in which every node has a fixed slot for sending while it listens to all other nodes in the other slots. Furthermore, there is a sleep period in which all the nodes

Table 1: Mode selection for cow-health monitoring scenario

Global mode	-	*Not milking*	*Not milking*	*Milking*	*Milking*
Local mode (for mobile nodes)	-	*Stationary*	*Walking*	*Stationary*	*Walking*
Local mode (for static nodes)	-	*Static*	*Static*	*Static*	*Static*
TDMA slot-size	0.1 s	0.05 s	0.05 s	0.1 s	0.1 s
TDMA sleep-time	0 s	10 s	10 s	0 s	0s
GBR update interval	30 s	60 s	60 s	30 s	30 s
Radio TX power (mobile nodes)	-10dBm	-10dBm	-5dBm	-15dBm	-15dBm
Radio TX power (static nodes)	-15dBm	-15dBm	-15dBm	-15dBm	-15dBm

turn off their power consuming radio. To get the packets to the sink, we use Gradient Based Routing (GBR) [13], where nodes maintain their hop-count by periodic flooding initiated by the sink and only forward packets received from nodes with a higher hop-count. The controllable parameters we consider are the TDMA slot-size, used for sending and receiving packets, and sleep-time, which are both global parameters, and the radio transmission (TX) power, which is a local parameter. We furthermore consider the flooding interval of GBR, which is a parameter which can be changed by the sink node on its own, but affects the behaviour of all nodes in the network. It is therefore considered to be a global parameter.

Design-time analysis to decide on optimal values for these controllable parameters is done with the use of simulations. We implemented the scenario in OMNeT++ [11], a discrete-event simulation environment, with the use of MiXiM [9], a modeling framework created for wireless networks, to simulate with different parameter values to get an understanding of the impact of changing different parameters. Table 1 shows the values for the parameters resulting from the design-space exploration to get a delivery ratio higher than 90% at the lowest possible power consumption for the different possible combinations of modes. The table furthermore shows the values that we would use when no reconfiguration approach is applied, but just a single (worst-case) configuration is used over time, in the first column. This is the current state-of-the-art as has been implemented in the WASP project.

We can see in Table 1 that global modes are only changed for the milking period which happens two times a day. Furthermore, the change of local mode can be completely observed locally. The expected overhead of the reconfiguration is therefore negligible and we integrate all the considered modes in our reconfiguration approach.

For the (mobile) nodes to detect whether the monitored cow is changing its posture from walking to stationary and vice versa, needed to select their local mode, we use simple thresholding on the accelerometer values, obtained from the BSN node. This procedure is already part of the application and the observation of these events can therefore be reused. As milking periods start and end every day around the same time, initializing a global mode based on the wall-clock time monitored by the sink is the easiest solution. Another option would be, for example, to use a dedicated sensor informing the sink as soon as the milking parlor is used.

We observed during simulations that a single flood is enough to inform all nodes, due to the small size of the information packet, the used contention free MAC and large density of the network. Because of this and the fact that nodes remain connected to the network throughout the day, a recovery approach is not required for this scenario. For the office scenario discussed next, this plays a more prominent role.

4.2 Office Monitoring Scenario

From monitoring the location of persons in a building we could for instance automatically control the lights in every room of the building based on the presence and activities of persons, potentially reducing the power consumption compared to the manual (de-)activation of light by the employees. We can similarly imagine control of temperature, humidity or other environmental aspects, or provide feedback based on activity. This makes it an interesting scenario for a WSN deployment.

We consider a deployment with a static backbone consisting of 15 TelosB [16] nodes, which monitor light intensity and forward packets to the sink, as shown in Figure 2. To monitor the location of employees in the building, a BSN node is carried by employees, in our case 5 persons, when they start working. They send the (static) parent node to which they communicate, as an indicator of their location, to the sink every 10 seconds.

The dynamics of employees show several interesting observable events for which we could reconfigure. Change in activity of the monitored employee, e.g., walking vs. stationary, impacts the mobility of the node carried by the person. Furthermore, there is an obvious daily pattern of day and night. At night no or just a limited number of employees need to be monitored, which can drastically reduce the amount of data that is communicated within the WSN.

The protocol stack we consider consist of the default Low-Power-Listening (LPL) MAC [10], available in TinyOS [17], a basic asynchronous MAC protocol which reduces idle listening by sampling the medium at a given interval. We use a tree based routing approach where the static nodes have a fixed parent. Due to the structure of the deployment, mobile nodes can determine their closest static node (to forward packets to and as an indication for their location) by iteratively broadcasting a request for hop-count information from the static nodes in range. The considered controllable parameters are the sampling interval of the MAC protocol, a global parameter, and the frequency at which the mobile nodes check their parent for the routing protocol and the transmission power of the radio, both local parameters.

We determined the parameter values to use based on experimenting with the deployment. With the actual deployment in place we determined the impact of changing the parameters on the delivery ratio and power consumption. This resulted in the parameter value selection as shown in Table 2. The first column of the table shows the configuration we would use when using only a single configuration (and thereby configure for the worst-case). Note that for the static nodes we use only a single local mode and retain the same local parameter values at all time.

With the switch between day and night we adapt the global LPL sampling interval parameter, such that nodes are able to handle the amount of traffic experienced during

Table 2: Mode selection for office monitoring scenario

Global mode	-	*Day*	*Day*	*Night*	*Night*
Local mode (for mobile nodes)	-	*Stationary*	*Walking*	*Stationary*	*Walking*
Local mode (for static nodes)	-	*Static*	*Static*	*Static*	*Static*
LPL sampling interval (for all nodes)	500 ms	500 ms	500 ms	2000 ms	2000 ms
Routing request interval (for mobile nodes)	3 s	60 s	3 s	60 s	3 s
Radio TX power (for mobile nodes)	-15 dBm	-15 dBm	-10 dBm	-15 dBm	-10 dBm
Radio TX power (for static nodes)	-15 dBm	-15 dBm	-15 dBm	-15 dBm	-15 dBm

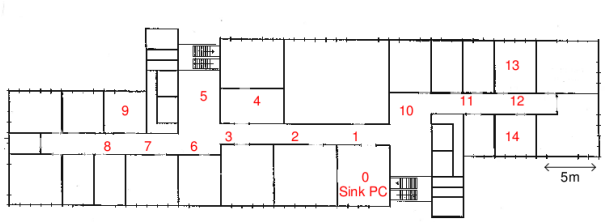


Figure 2: Deployment of 15 static TelosB nodes

that mode. While walking, the request interval is selected to be 3 seconds, which is enough to quickly respond to changing connectivity based on the distance between the nodes and walking speed. While stationary, nodes do not experience this change in connectivity and a high value of 60 seconds is selected. The transmission power is chosen such that a node can always connect to at least one static node in the monitored environment. When walking, we observed a higher interference on the sent packets, for which we compensate by a slightly higher transmission power. Note that we have chosen not to use this higher transmission power for the single configuration approach. This is because this higher transmission power is only found to be useful during the limited time a person is walking, which does not outweigh the disadvantage of having a high transmission power during the entire day. The use of modes allows for a more fine-grained exploitation of performance trade-offs.

For the (mobile) nodes to detect whether the monitored person is changing its posture, we can use the same approach as for the cow-health monitoring scenario. For the global mode to use, nodes start in a consistent global mode during the deployment and are informed about any change of the global mode afterwards. For this scenario we can initiate the change of global mode based on wall-clock time. Time is typically monitored by a computer controlling the WSN and a global mode change can be initiated by communicating to the connected sink node (via serial communication).

We observed that for our scenario, a single broadcast is often not enough to change a global mode due to the lack of redundancy in the deployment and protocols. With more retransmissions of the mode change information packet we quickly converge to successfully switching all (connected) nodes. Based on these results, we can flood a global mode request a conservative number of five times to ensure a sufficient minimal probability of connected nodes requiring to use a recovery approach. Since a global mode happens only two times a day this repeated flooding induces a negligible amount of the total traffic load during the entire day, i.e., 10 packets per node during the entire day.

As nodes (re-)joining the network are an integral part of the application, we do require a recovery approach for these nodes to determine the globally used parameter values. We implemented the recovery approach as proposed in this paper where the node cycles through the global modes while requesting their parent for the global mode if they observe they are entering the network after being disconnected. Experiments with this approach show that the time needed to find the global mode is typically less than a second when in range of the network. We did not implement a repeated check of the global mode for this scenario thereby accepting the small probability of a connected node experiencing limited connectivity if a mode change request is not received.

5. PERFORMANCE EVALUATION

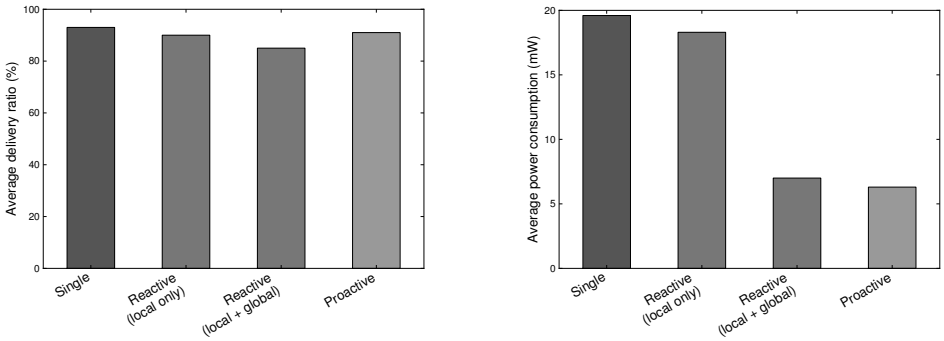
In this section, we discuss results of simulations for the cow-health monitoring scenario and experiments with the office building monitoring deployment. Results show that the integration of our reconfiguration approach has a significant positive effect on the power consumption, while there is no negative effect on the delivery ratio, compared to a basic approach using a static worst-case mode.

None of the existing reactive approaches, such as [3, 5, 8, 15, 18, 21], can directly be used for the entire protocol stack and controllable parameters we consider. To see the impact on the metrics and behaviour of the network when using a proactive compared to a reactive approach we have created our own. We create the best possible reactive approach for the two scenarios, also using the optimal parameter knowledge, expressed by the modes, obtained at design-time. It adapts the parameters in response to changes in the observed network performance instead of application events.

5.1 Cow-health Monitoring Scenario

We implemented the cow-health monitoring scenario and integrated the proactive reconfiguration approach, as constructed in the previous section, for OMNeT++ [11]. The results of the integration for a small-scale WASP demonstrator are discussed in [19]. We compare our proactive approach, in terms of the (conflicting) delivery ratio and power consumption of the nodes in the network, with the approach of using a single worst-case configuration (as is the current practice for this scenario) and a reactive approach.

The considered reactive approach determines the value for the local parameters, i.e., only the transmission power, based on the locally monitored quality of sending packets to the parent node. More specific, the transmission power is increased according to the values related to the modes as soon as the observed delivery ratio to the parent is lower than 85% and lowered as soon as the ratio is higher than 95%. We respond to the periods of high data traffic, as observed when milking, by monitoring the amount of data processed at the sink. We set the global parameters, TDMA slot-size, sleep-time and GBR update interval, to the values related to the milking mode (Table 1) when it is observed at the sink that at least 80% of the cows is sending at the rate related to walking and to the not milking values otherwise.



(a) (b)

Figure 3: Impact on average (a) delivery ratio (b) power consumption for cow-health monitoring scenario

The sketched scenario is simulated for a day, in which two milking periods of two hours occur, for the three approaches, i.e., a single (worst-case) configuration, reactive (adapting only the local parameters and adapting both the local and global parameters) and proactive using modes. To have statistically reliable results, every experiment was repeated 10 times; the results showed limited variation. The average delivery ratio and the average power consumption of the nodes are shown in Figure 3. Shown results are the averages over all runs. The delivery ratio was determined by the ratio of packets sent by the mobile nodes and the number of packets received by the sink. For the power consumption we focus on the main power consuming part of the node, i.e., the radio [12, 16], which, in this case, depends on the chosen TDMA schedule.

The first thing we can observe is that for this scenario a significant reduction of power consumption, and thereby increased lifetime of the WSN, can be obtained by the introduction of the proactive approach compared to the currently used single configuration, while the delivery ratio is still maintained at a high level. When we look at the results of the reactive approach we see that if we only consider adapting local parameters, as is typically done, we see only a limited improvement, but as soon as global parameters are adapted as well, the average power consumed by the nodes reduces and is only slightly higher than the proactive approach. This shows the importance of global parameters for this scenario.

When comparing the reactive and proactive approach the most profound difference can be seen when network dynamics occur. With the reactive approach there is a delay between observing the high network load and the adaptation of the global parameters, i.e., TDMA schedule. In this period packets are lost due to the limited capacity reserved by the TDMA schedule. This only slightly impacts the delivery ratio if it is averaged over a long period, due to the limited frequency of the events. This effect is discussed in more detail for the actual deployment used for the office monitoring scenario, discussed in the next subsection.

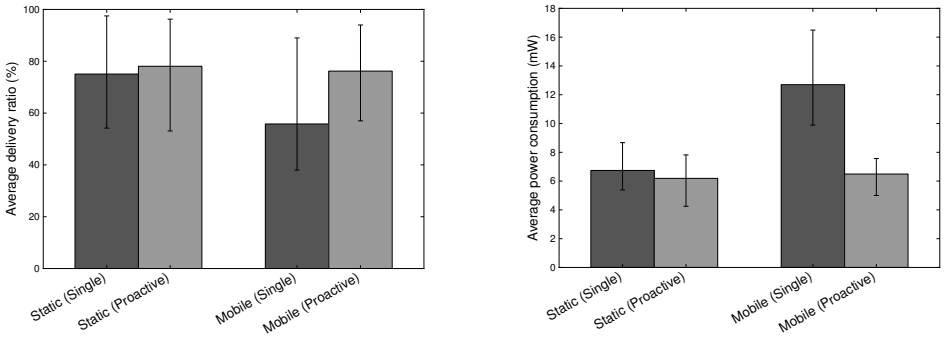
What we observe when looking in more detail at the simulations is that in the period between observing the high network load and adaptation of the global parameters, the reactive approach first tries to compensate for the low delivery ratio by increasing the transmission power, which increases power consumption, while the solution lies in the adaptation of the TDMA schedule. Such a cross-layer view is not present in reactive approaches for individual parameters, but it is an additional advantage present in our method.

What we furthermore see is that the coordinated change of the global parameters becomes less efficient due to the high traffic and more than one retransmission is needed to successfully change. This increases the time until the required adaptation even more. Packets are lost during this transitional phase. This shows an advantage of our proactive approach; it can anticipate and adapt for the high traffic just before it actually occurs, avoiding adaptation in the period in which coordinating a mode change is more challenging and many packets can be lost.

5.2 Office Monitoring Scenario

We implemented the office monitoring scenario and integrated our reconfiguration approach, as described in the previous section, using TinyOS 2 [17]. We did a one day experiment based on the parameter values shown in Table 2. We decided to use the night mode before 9 AM, after which most employees start working, and after 7 PM, the time at which most employees have ended their working day. Because mobility and interference patterns are hard to control and differ between the different experiments, it is very hard to make a precise quantitative comparison between the different approaches. What the experiments do show is that we can successfully integrate our reconfiguration approach in a practical environment. We also observe significant positive trends for the delivery ratio and power consumption when using a proactive approach. In this subsection we furthermore look in more detail at the comparison with a reactive approach showing the benefit of proactivity compared to the delayed response of a reactive approach.

Figure 4 shows the impact of integrating our proactive approach on the average delivery ratio and average power consumption, where we differentiate between the static and mobile nodes. Unlike the simulations, practical experiments show a large difference between the performance of individual nodes, mainly due to the larger impact of external interference. The error bars show the maximum and minimum observed delivery ratio and power consumption of individual nodes. The results are based on experiments starting at 8 AM, before all monitored employees started working and 1 hour before the global mode switch, until 8 PM, after all employees stopped working and 1 hour after the global mode switch. For practical reasons we could not leave the



(a) (b)

Figure 4: Impact on average (a) delivery ratio (b) power consumption for office monitoring scenario

nodes over night. For the remaining 12 hours we therefore used a delivery ratio and power consumption based on experiments with only the static network in place and assumed it would be similar during the entire night. The results are based on the time they are on, which is the complete day for the static nodes and around 9 hours for the mobile nodes. The delivery ratio is based on the (application) packets sent, when the nodes were inside the building, i.e. connected to the network, in case of the mobile nodes. For the power consumption we focus on the power spent by the radio, which is derived by monitoring the time spent in every radio state, i.e., sending, receiving and idle [16].

The proactive reconfiguration resulted in a significant better power consumption for the mobile nodes, while the average delivery ratio is maintained and even improved. For the (less power constrained) static nodes no significant impact is observed due to the limited network dynamics and impact of changing parameters observed by these nodes.

Like in the cow-health monitoring scenario, when comparing with a reactive approach which is created to optimize the quality metrics averaged over a long time, it is expected to have a similar impact on the metrics due to limited network dynamism. The main difference between the reactive and proactive approach is found around periods of network dynamics, e.g., when people are moving. We did additional experiments with the deployment to zoom in on this behaviour and its effects on the performance metrics for local mode adaptation. We used a single mobile node which in 1 minute moves from one side of the corridor to the other, stays there for 2 minutes, moves to the other side in 1 minute and stays there for 2 minutes. During the experiment, the mobile node sends packets to the sink once every 5 seconds, while the static nodes only forward these packets. Results are collected after an initialization period of 1 minute.

We integrated a reactive approach in the routing protocol, which provides the parent to use for communication. The locally observable change of parent (for the mobile nodes) is a good indication of a changing topology and the routing protocol adapts local parameters in response, i.e., the transmission power (TX) is increased to -10dBm and the routing request interval reduced to 3 seconds. If no change of the parent is observed in the last 30 seconds, the activity of the protocols is relaxed by reducing TX to -15dBm and the interval to 60 seconds. With our experiment we focus on local

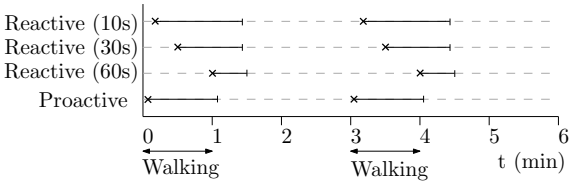
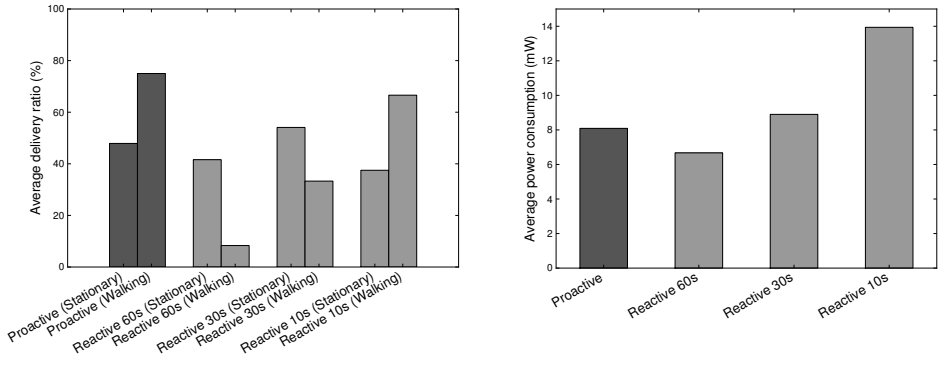


Figure 5: Adaptation responsiveness

events and retain the same global parameter values, i.e., an LPL sampling interval of 500 ms.

Figure 5 shows the reconfiguration moments in time of the proactive approach compared to the reactive approach as described above. Furthermore, it shows the results for the reactive approach when spending more time on the local analysis of the performance, i.e. reduce the request interval to 30 and 10 seconds, in order to increase the responsiveness. While the proactive approach can accurately adapt during the times it is required, using a reactive approach introduces a delay for the adaptation. After the node starts moving a delay arises until a parent change is observed. After the period of walking, there is still a small delay due to the introduced interval to be sure that the topology stopped changing. Reducing this interval can reduce the delay, but can result in a more frequent, unnecessary, adaptation for slow topology changes.

Figure 6 shows the delivery ratio and power consumption of the mobile node for the different activities, i.e., walking and stationary, when using the different adaptation approaches. The delivery ratio when walking can in this case potentially be higher than stationary due to the lower average number of hops the packets need to travel to the sink. With the reactive approach, delay in adaptation results in nodes using an unreachable parent. For the reactive approach we can therefore see a clear trade-off between the responsiveness to topology changes, which is reflected in the delivery ratio when walking, and power spent by the nodes. Putting more effort in observing the local performance, during the entire run-time, results in a better responsiveness to dynamic events, but also requires more power during the period in which no dynamics occur. With the proactive approach we actively observe the dynamics and this trade-off does not need to be made, resulting in both a better delivery ratio (when walking) and overall power consumption.



(a) (b)

Figure 6: Impact on average (a) delivery ratio per activity (b) power consumption of the mobile node for small scale experiment

6. CONCLUSIONS

In this paper, we have introduced a proactive run-time reconfiguration approach for Wireless Sensor Networks (WSNs). This approach explicitly exploits a priori knowledge of the application scenario to guide a reconfiguration process.

At design-time, observable events signalling a change in dynamics experienced in the network and thereby potentially the Quality-of-Service (QoS) provided by the network are identified. Using existing design-time analysis techniques, the optimal response to these dynamics in terms of adapting controllable parameters is determined. Based on this analysis, different modes of operation are defined. We distinguish parameters that can be adapted locally and those that should be considered globally. We therefore define a global mode of operation, equal for every node in the network, and a local mode, which can be different between nodes. These modes are selected by keeping in mind the benefit of using them at run-time and the overhead involved.

At run-time, nodes locally observe the design-time defined events, which either trigger the change of a local mode or is communicated to a centralized location. A centralized location, such as a sink, initiates the change of global mode by communicating to all nodes in the network, for which a simple, low overhead flooding approach suffices. Our reconfiguration approach is made robust for practical deployments by a recovery approach that allows nodes to determine the currently used value for global parameters after, for example, (re-)joining the network or missing a global mode request.

We showed a proactive reconfiguration approach for practical cow-health monitoring and office monitoring scenarios. The use of a proactive approach results in a significant positive effect on the used resources, in this case power consumption, while there is no negative effect on other QoS metrics, i.e. the packet delivery ratio. Due to the controlled number of global parameter changes and design-time definition of the modes, there is only a very low run-time overhead incurred by our approach. By comparing with the most applicable reactive approach, we have seen that being proactive allows an early reconfiguration for network dynamics thereby avoiding phases in which the performance of the network is limited.

Acknowledgment

This work was in part supported by EC FP6 project IST-034963 WASP.

7. REFERENCES

[1] S. Brown and C. J. Sreenan. Software Update Recovery for Wireless Sensor Networks. *SAEL*, 29:107–125, 2010.
[2] BSN website, ICL London. http://vip.doc.ic.ac.uk/bsn/m621.html.
[3] O. Chipara et al. Real-time Power-Aware Routing in Sensor Networks. *IWQoS*, pages 83–92, 2006.
[4] K. P. Ferentinos and T. A. Tsiligiridis. Adaptive design optimization of wireless sensor networks using genetic algorithms. *Computer Networks*, 51(4):1031 – 1051, 2007.
[5] A. Forster. Machine learning techniques applied to wireless ad hoc networks: Guide and survey. *ISSNIP*, 2007.
[6] S. V. Gheorghita et al. System-scenario-based design of dynamic embedded systems. *ACM Trans. Des. Autom. Electron. Syst.*, 14(1), article 3, 45 pages, 2009.
[7] R. Hoes et al. Analysing QoS trade-offs in wireless sensor networks. *MSWiM*, pages 60–69, 2007.
[8] S. Kogekar et al. Constraint-guided dynamic reconfiguration in sensor networks. *IPSN*, pages 379–387, 2004.
[9] MiXiM website. mixim.sourceforge.net.
[10] D. Moss and P. Levis. BoX-MACs: Exploiting Physical and Link Layer Boundaries in Low-Power Networking. Technical report, Stanford Information Networks Group, 2008.
[11] OMNeT++ website. www.omnetpp.org.
[12] A. Prayati et al. A modeling approach on the TelosB WSN platform power consumption. *J. Syst. Softw.*, 83:1355–1363, 2010.
[13] C. Schurgers and M. Srivastava. Energy efficient routing in wireless sensor networks. *MILCOM*, 2001.
[14] M. Strasser and H. Vogt. Autonomous and distributed node recovery in wireless sensor networks. SASN, pages 113–122, 2006.
[15] Y. Sun et al. RI-MAC: a receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks. SenSys, pages 1–14, 2008.
[16] TelosB Datasheet, Crossbow Inc. www.xbow.com.
[17] TinyOS website. www.tinyos.net.
[18] S. Tomforde et al. Adaptive Control of Sensor Networks. *ATC*, pages 77 – 91, 2010.
[19] C. Viet Ngo, R. Serna Oliver, and G. Fohler. SIMOSP: A Simple Mode Switch Protocol for Wireless Sensor Networks. Technical report, Technische Universtät Kaiserslautern, October 2010.
[20] WASP (Wirelessly Accessible Sensor Populations) project website. http://www.wasp-project.org/.
[21] W. Ye, J. Heidemann, and D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *IEEE/ACM Trans. Netw.*, 13(3):493–506, 2004.