OMNeTA: A Hybrid Simulator for a Realistic Evaluation of Heterogeneous Networks

Markus Jung
Institute of Telematics
Karlsruhe Institute of Technology, Germany
markus.jung@kit.edu

Anton Hergenröder*
anton.hergenroeder@kit.edu

ABSTRACT

High-quality simulation tools are crucial to evaluate the energy efficiency of applications and protocols for wireless sensor networks in the most realistic way. Sensor node emulators like Avrora are well-suited for homogeneous networks but can not cope with heterogeneous networking scenarios. Wireless sensor networks have been used mostly isolated for special purposes in the past. But use cases like the so called smart environments require wireless sensor nodes to communicate and interact with other devices too. In this paper, we present OMNeTA as a solution to this problem, offering the ability to simulate highly heterogeneous networks of different device classes. Our hybrid simulation approach combines the proven hardware emulation by Avrora, radio communication simulation by MiXiM and the flexibility of OMNeT++. Bridging the abstraction gap between hardware emulation and network simulation, OMNeTA may also reduce the effort required to experiment with new protocols, applications or other varying parameters, by enabling rapid-prototyping like development processes.

1. INTRODUCTION

Wireless sensor networks have been a topic of research for several decades. However, except for special purposes or research projects, they were not commonly deployed in the past. This expected to change, just as the way how those sensor networks are used in the future. Small interconnected microcomputers, thus sensor nodes, will be part of our everyday live, collecting and communicating data from their surroundings. This way, they provide the information which is required to enable our environment to adjust and act according to our needs, a vision which is known as *smart environment*.

In smart environments, or to take things one step further, in the *Internet of Everything*, sensor nodes are part of a network consisting of various devices and services, e.g. temperature sensors, smartphones and weather forecasts computed in the cloud. In terms of wireless sensor nodes, this heterogeneity is challenging. Typical sensor nodes are short of memory and energy, prohibiting protocols requiring to much state information, complexity or persistent connectivity. Most protocols commonly used in the Internet are therefore of limited suitability.

While protocols like 6LoWPAN aim to enable end-to-end communication even in constrained environments, most evaluations either consider homogeneous networks or partially abstract away from practical implementations for real sensor nodes. In sensor networks however, it is essential to involve implementations applied in real deployments to ensure most realistic evaluation results regarding energy efficiency. Therefore, high-quality simulation tools are fundamental for research and evaluation of communication protocols and applications in wireless sensor networks. But many simulation tools are either unable to simulate networks as heterogeneous as outlined before, or they can not provide evaluations results which are sufficiently precise and close to reality.

Our approach permits realistic evaluations of heterogeneous networks by combining the benefits of sensor node emulation and flexible but abstract network simulation. For sensor node emulation we use *Avrora* [10], which enables us to evaluate implementations without modifying them, as if they were running on real nodes. The wireless channel and less constrained network members are simulated by the simulator *OMNeT++* [11] including the simulation framework *MiXiM* [6]. By integrating the sensor node emulation of Avrora into OMNeT++ we developed a hybrid simulation tool called *OMNeTA*, which enables realistic simulation of complex application scenarios with sensor networks.

The reminder of this work is structured as follows. We discuss existing simulation tools for sensor and conventional networks. Then, we present our hybrid simulation tool OMNeTA and evaluation results regarding its correctness and performance. We conclude our work by summarizing the results and identifying future work.

1.1 Related Work

The tools typically used to simulate wireless sensor networks can be roughly divided into three categories: Network level simulators, platform specific simulators and hardware emulators. Simulators working at network level like OMNeT++ address the communication behavior of applications and protocols. Because of their scope, they abstract from hardware properties which simplifies modeling and im-

*Work performed while at Institute of Telematics, Karlsruhe Institute of Technology

Q2SWinet'15, November 2–6, 2015, Cancun, Mexico.

ACM 978-1-4503-3757-1/15/11 ...$15.00.
DOI: http://dx.doi.org/10.1145/2815317.2815331.

proves the simulation performance. Due to their high abstraction level, network level simulators are not well suited for realistic evaluations with regard to energy efficiency, even if there exist good simulation models targeting wireless sensor networks like Castalia or MiXiM.

Platform specific simulators like PowerTOSSIM z [9] are bound up with a sensor node operating system, TinyOS in this specific case. They integrate large parts of the operating system into the simulation while replacing low-level components by mock implementations. The platform specific approach is very close in code coverage to the real application binary. Moreover, it keeps the simulation overhead low by compiling application and operating system code into native machine code. However, the run-time behavior can not be simulated precisely since the application is not executed by the sensor node itself. Therefore the energy simulation quality suffers, too. Additionally, heterogeneous networks are beyond the scope of platform specific simulators.

In contrast, hardware emulators like Avrora or ATEMU [1] address only the sensor node hardware itself. They execute an application binary like a real sensor node, completely transparent to and oblivious of the application itself. Because of their low abstraction level, hardware emulators can offer a high simulation precision and quality at the cost of an increased computation time. But they are typically not designed to simulate heterogeneous networks including other devices than sensor nodes.

Hybrid simulation tools work across different abstraction levels to overcome specific limitations. One example is COOJA [8], which integrates existing hardware emulators with a platform-level simulation of the Contiki operating system. This approach enables COOJA to simulate hybrid sensor networks but does not extend to more general scenarios involving different device types. OMNeTA in contrast is closer to a general purpose network simulator, flexible and extensible. It uses a hybrid approach as well but covers a broader range of abstraction. Therefore, to enable a cooperation between its components, different solutions are required to translate between these more distant levels of abstraction.

2. BASIC PRINCIPLES OF OPERATION

Our hybrid approach enhances Avrora to delegate the simulation of radio communication to OMNeT++/MiXiM. This enables an integration of emulated sensor nodes into OMNeT++ simulation models, permitting a transparent interaction.

In OMNeTA, the functionality of simulating a wireless sensor network is divided into three distinct parts: (a) sensor node emulation (b) radio transmission simulation (c) additional functionality and simulation infrastructure.

The first part is covered by Avrora. Avrora supports sensor node emulation and incorporates high quality energy models [2] e.g. for MICAz and IRIS nodes. MiXiM has been chosen to simulate the radio communication between sensor nodes. It offers many important wireless channel and radio models and supports node mobility. OMNeT++ provides the simulation kernel and further infrastructure as well as a programming interface for simulation models which is also used by MiXiM.

An example simulation scenario illustrating the basic architecture of OMNeTA is depicted in figure 1. In this scenario, two sensor nodes emulated by Avrora are running application images containing operating system specific hardware drivers and protocol stacks as well as application layers and logic. A third simulation participant, e.g. a 6LoWPAN-gateway or a smartphone app is implemented as an OMNeT++ model. All participants are using MiXiM to simulate the radio communication between each other. This also applies to emulated sensor nodes which are integrated into OMNeTA simulations by the means of a proxy radio medium whose functionality is transparently provided by the MiXiM framework.

Avrora and OMNeT++/MiXiM focus on different scopes of application. Consequently, they differ also with regard to their design philosophy and abstraction level, which leads to three main challenges:

Inter-simulator communication Both Avrora and OMNeT++ are self-contained simulation tools with different data types and representations. To be able to cooperate with each other, they need to exchange informations regarding the simulation state and activities. An inter-process communication facility provides this functionality.

Radio communication The radio devices are emulated in Avrora but the radio communication itself is simulated by MiXiM. A proxy implementation representing the sensor node in OMNeT++ models is therefore required to translate and forward radio control commands, results and transmission data between Avrora and OMNeT++/MiXiM.

Synchronization Dependencies between simulation events impose restrictions on their processing order. A synchronization mechanism controls the simulation kernels of Avrora and OMNeT++ to enforce those restrictions across the local simulator boundaries.

2.1 Inter-simulator communication

During simulation, Avrora and OMNeT++/MiXiM must forward all non-local changes among the simulation model to each other. These changes are always tied to a fixed point in time. They come into effect as soon as the simulation progress reaches this time of processing. The interaction between both simulators could therefore be described as some kind of *asynchronous remote method invocation*. The inter-simulator communication facilities in Avrora and OMNeT++ have been implemented non-blocking for performance reasons and to prevent deadlocks caused by circular control message dependencies. They use two distinct threads for incoming and outgoing communication and an additional message queue, parallelizing inter-simulator communication and local simulation kernel workload.

The simulation performance of OMNeTA depends strongly on the inter-process communication between Avrora and OMNeT++. Certain circumstances and tasks, especially the simulation progress synchronization, can enforce a serial control flow alternating quickly across the process boundary. This pattern can not be avoided, it is imposed by the control and data flow dependencies intrinsic to the simulation model. Each time one of the simulators is required to wait for the other one, the inter-process communication latency and context switching overhead add up to the computation time. Since this amplification effect can potentially increase the simulation time by a huge amount,

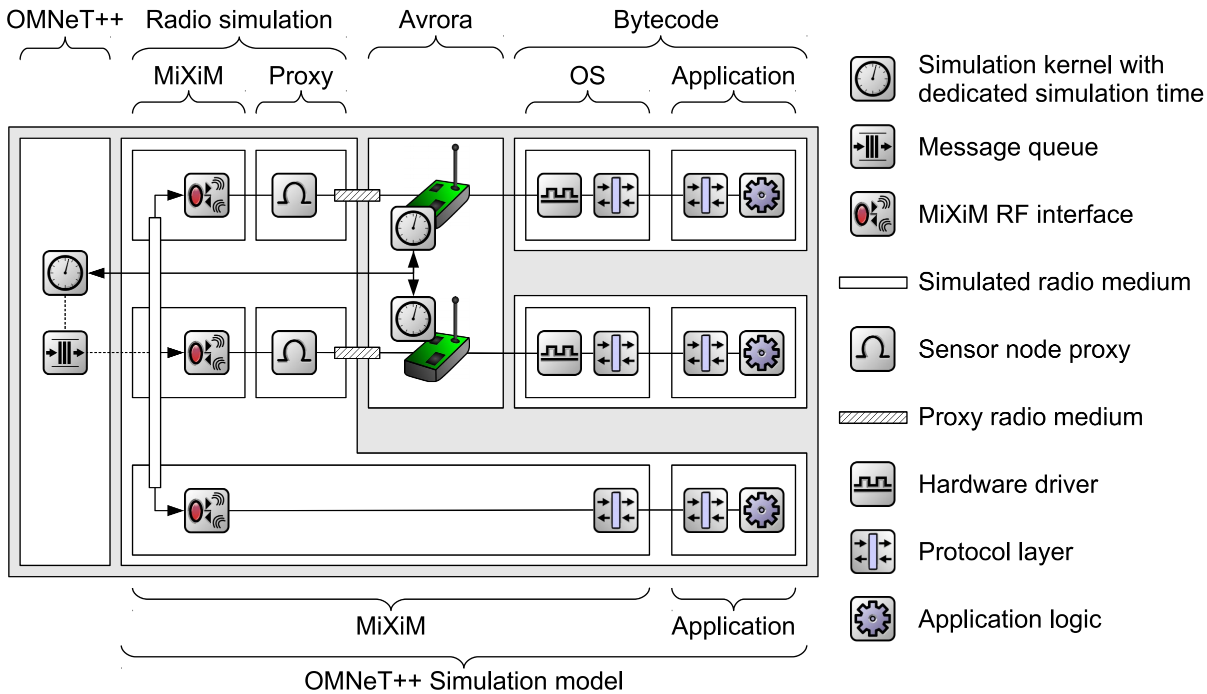


Figure 1: Architectural overview of OMNeTA, depicting an example simulation scenario with two sensor nodes and one additional participant. Nodes are emulated by Avrora while the additional participants like 6LoWPAN-gateways are realized as OMNeT++ simulation models. Icons used by courtesy of OpenSim Ltd.

the inter-process communication latency will be one of the most important parameters for performance optimizations.

2.2 Radio communication

In OMNeTA, MiXiM is used to simulate radio transmissions between sensor nodes. Therefore, the functionality of the so called *medium*, which connects the emulated radio devices in Avrora to each other, is not required any more. It has been replaced by an API compatible implementation which forwards commands and queries for further handling to OMNeT++ and feeds query results and transmission data back to the emulated radio devices. Each radio, and therefore each sensor node, in Avrora is assigned to a proxy instance in OMNeT++ which in turn translates the commands sent by Avrora to the MiXiM API and vice versa.

Avrora and OMNeT++/MiXiM model radio transmissions in an incompatible way. Since Avrora emulates the sensor node, it is forced to hand over received data as soon as possible to the radio device to match the timing of real hardware. For this reason, the basic unit of communication in Avrora is a single byte and transmissions are represented as a sequence of bytes. Working at a much higher abstraction level, OMNeT++/MiXiM addresses communication at packet level. MiXiM is consequently unable to forward a whole packet, for example to Avrora, until it has been received completely. Since OMNeTA is required to match the abstraction level of Avrora, all bytes are transmitted using distinct MiXiM packets. To be able to provide valid metadata to OMNeT++/MiXiM, each outgoing transmission needs to be finished first in Avrora before it can be forwarded to MiXiM. This approach bridges the abstraction gap between Avrora and OMNeT++/MiXiM but it also imposes two timing constraints which must be enforced by synchronization mechanisms:

1. The whole transmission data is required to be available to OMNeT++/MiXiM at the begin of the transmission. Therefore, the simulation time in OMNeT++ must lag behind the simulation time of sending sensor nodes in Avrora. Or more formally:

$$t_{node} \geq t_{tx,start} + t_{transmit} > t_{OMNeT++}$$

2. Each byte must be delivered to all receiving nodes at the end of its transmission time. Therefore, those nodes are only allowed to progress in simulation time if they lag behind OMNeT++.

2.3 Synchronization

Simulation models of event-based simulators represent all state changes and processes as a series of discrete events, each associated with a timestamp denoting the event time. Some of these events depend on each other, others do not, for example because they belong to different sensor nodes and only affect the node-local state. The event processing order imposed by those dependencies can be represented as a directed acyclic graph $G = (V = \{v : v \text{ is an event}\}, E = \{(u,v) : u,v \in V \text{ and } u \prec v\})$, with $u \prec v :=$ "v depends on u". The events along a path in this graph are ordered by their designated processing time. Figure 2 illustrates such a dependency graph.

Since OMNeTA combines Avrora and OMNeT++, inter-simulator synchronization is required to prevent event dependency violations. The OMNeT++ simulation kernel works single-threaded by default. It processes all events

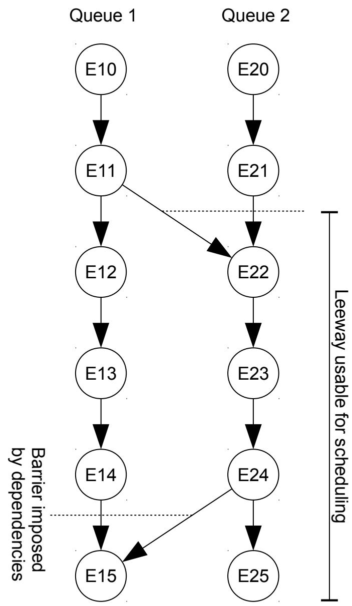


Figure 2: Event dependency graph for two simulation kernels running in parallel with synchronization barriers.

serially with monotonic increasing simulation time. This approach does not scale well, in the context of OMNeTA, it would require both Avrora and OMNeT++ to synchronize each single simulation step. Additionally, a permanent synchronization conflicts with the radio communication timing constraints explained in section 2.2. For this reason, OMNeTA applies a relaxed synchronization scheme based on dependencies between the simulation events in Avrora and OMNeT++. Sensor nodes in Avrora are emulated by distinct threads, each one running its own simulation kernel, progressing through simulation time independently from each other. A global ordering of simulation events, and therefore synchronization, is only required if sensor nodes interact with each other, as depicted by the barriers in figure 2. In case of OMNeTA there is no way how Avrora nodes could interact directly. They are only interconnected by their radio device and the radio communication is simulated by OMNeT++/MiXiM. For this reason, synchronization is only required if nodes exchange information with OMNeT++/MiXiM.

In fact, OMNeTA does not even synchronize all interactions between Avrora nodes and OMNeT++. Requests from Avrora get enqueued as event in OMNeT++ and are processed in order with regular simulation events. This requires OMNeT++ to progress slower in time than Avrora since even the last node in Avrora must be able to insert events into the event queue. Synchronization can not be avoided if sensor nodes in Avrora try to read the radio signal strength or to receive data. In more general terms, synchronization must take place if a node in Avrora tries to gather state information regarding the simulation model in OMNeT++.

One additional optimization is implemented in OMNeTA: Synchronization is required to allow OMNeT++ to process events which are due to a certain point in time. Those events are either added by Avrora or created in succession of other events. Avrora is therefore able to predict the earliest point in time at which a synchronization with OMNeT++ must be performed, by keeping track of the event queue of OMNeT++. Earlier synchronization requests require no interaction with OMNeT++, only the local inter-node timing constraints must still be enforced.

CPU	4× AMD Opteron 6134, 8 Cores at 2,3 GHz per CPU
RAM	128 GB
OS	Ubuntu Linux 14.04.2
Kernel	3.13.0-45 (generic, x86_64, Patchlevel 74)
C/C++-Compiler	gcc 4.8.2-19ubuntu1
Java VM	OpenJDK Server 7u75-2.5.4-1~trusty1
TinyOS	2.1.2
TinyOS-Compiler	nesC 1.3.4 and avr-gcc 4.5.3
Avrora	Avrora+ 1.7.117
OMNeT++	4.5
MiXiM	2.3

Table 1: Hardware specification and software versions of our evaluation environment.

To sum it up:

1. Avrora controls the simulation progress and defines the upper bound for OMNeT++. This bound is determined by the last node in Avrora.

2. In addition, OMNeT++ is not allowed to progress while the first timing constraint mentioned in section 2.2 could possibly be violated. Since $t_{transmit}$ is unknown, OMNeT++ and each receiving node in Avrora is required to wait until the last transmitting node is at least one maximum transmission unit ahead. This also holds true for idle sensor nodes, since they could start a transmission at any point in time.

3. The last receiving node, which is also globally the last node in Avrora, updates the progress state of OMNeT++. OMNeT++ in turn continues to process simulation events until it reaches the updated time barrier. Pending communication data as well as query results are returned to Avrora, fulfilling the second communication timing constraint.

4. The progress update step can be omitted if no events in OMNeT++ are due to be processed up to the time of synchronization.

3. EVALUATION

Our evaluation of OMNeTA has two different goals. At the one hand, we validate our hybrid simulation approach since OMNeTA can not be used reasonably as an evaluation tool without knowledge of its simulation quality. Our validation considers communication behavior and energy consumption of simulated sensor nodes but also covers timing constraints. Additionally, by comparing OMNeTA to Avrora, we also evaluate parts of Avrora itself, especially the radio communication model. We also take performance comparison between both simulators into account. Finally, we show first results of OMNeTA simulating a heterogeneous sensor network.

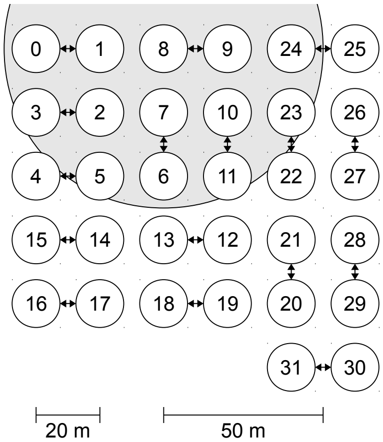


Figure 3: Grid topology used to simulate the application PingPong. The radio communication radius is highlighted for node 8.

3.1 Methodology

To evaluate OMNeTA, we used a low-traffic application called *PingPong* implemented with TinyOS. This application employs unicast communication between distinct pairs of sensor nodes. Similar to real-world applications, PingPong tries to conserve power by using *Low Power Listening*. Low Power Listening is the TinyOS implementation of BoX-MAC-2 [7], a MAC protocol designed for wireless sensor nodes and similar devices. The first node of each pair sends a message to its partner each two seconds. The second node polls each second for a received packet and sends it back, if necessary. One transmission has a total length of 27 bytes. In addition, we used a second application, *Flood*, to evaluate the simulation performance of OMNeTA especially at high-load conditions. Flood uses Low Power Listening and broadcast communication. A new message is flooded through the network each two seconds and forwarded just once by each node. Forwarding is done with a random delay of 32 ms to 160 ms to reduce the risk of collisions.

All applications have been simulated for 60 s using MICAz sensor nodes. They have been started sequentially with a delay of about 1,6 ms to prevent them from running in lockstep. Both Avrora and OMNeT++/MiXiM were configured to simulate radio communication using an unit-disk model with 50 m radius, which rules out all potential sources of data corruption except of collisions. The radio itself is modeled as `Decider802154Narrow` in MiXiM, using the default parameters for the radio chip of MICAz, CC2420. Table 1 lists the most important technical parameters of our evaluation environment.

Both Avrora and OMNeT++/MiXiM use and statistical models, for example to simulate radio transmissions. Even if randomness should have no significant impact on the unit-disk model, we decided to repeat each experiment sevenfold with different seed values of the random number generators.

3.2 Power & Communication

Power consumption and communication behavior are closely related to each other in sensor networks, especially if the nodes do not perform other computation-intensive tasks. To evaluate these aspects, we simulated the application PingPong with OMNeTA and Avrora, both configured to model a simulation environment as identical as possible. 2 up to 32 MICAz nodes have been virtually deployed in a grid with an inter-node spacing of 20 m. The topology and resulting node pairs are depicted by figure 3. We measured the current consumption of each node. Additionally, the number of sent, received and corrupted received bytes has been recorded.

3.2.1 Measurments

The energy measurements of all repetitions have been aggregated by choosing the median for each node independently. Figure 4a depicts the distribution of the consumed energy of all nodes in the simulated sensor network. It visualizes the minimum and maximum consumed amount of energy as well as all three quartiles. The communication statistics in figure 4b, 4c and 4d show the average amount of bytes per node which have been sent, received or received corrupted. The graph of OMNeTA in figure 4d uses the second y-axis since the amount of data corruption in Avrora is by magnitudes larger.

Comparing the energy consumption as shown in figure 4a, sensor nodes in Avrora consume up to about 20% more energy as in OMNeTA. The observed mean standard deviation within the simulation repetitions has been far below 1% of the total energy consumed by a node and is therefore negligible. Sensor nodes in Avrora use their radio much more intense as their counterparts in OMNeTA. As shown in figure 4b, they send up to 30% more data for network sizes of 8 or more nodes. The number of received bytes is up to 60% higher in comparison to OMNeTA (figure 4c). And finally, nodes in Avrora receive about 750 times more data which has been corrupted by collisions, as figure 4d visualizes.

3.2.2 Discussion

The simulation results show large differences between Avrora and OMNeTA despite the fact that both simulators have been configured to use the same radio model. But an in-depth analysis of the radio simulation code revealed discrepancies between the data corruption models implemented by Avrora and OMNeTA.

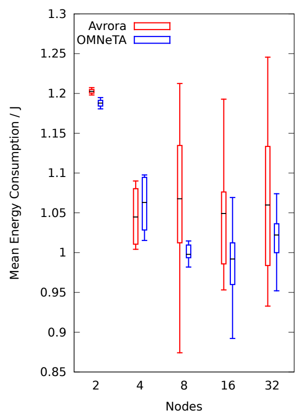
In OMNeTA, a model has been used which originates from the IEEE 802.15.4 standard [3] and is based on the *SINR* (*signal-to-interference-plus-noise ratio*). The unit-disk model defines the signal strength as constant within the radius of the disk. Therefore the SINR of two concurrent senders evaluates to:

$$\text{SNR} = P_{Signal}/P_{Noise} = 1/1 = 10\,\lg(1)\,\text{dB} = 0\,\text{dB}$$

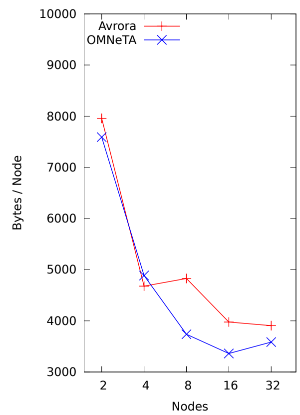
The expected bit error rate at 0 dB SINR is far below 10^{-3} according to the IEEE 802.15.4 model (see figure 5).

Avrora in comparison implements only a much simpler way to compute data corruption: If nodes send concurrently, their data is combined using an exclusive or. This implementation does not incorporate the distance between the sender and the receiver, the received signal strength or other influences. Even one concurrent transmission in reach will virtually always result in a packet loss. This design choice may seem legitimate in case of a unit-disk model but is absolutely inappropriate for more sophisticated radio models.

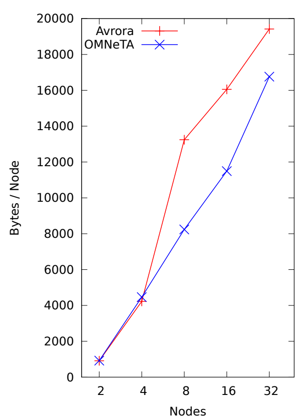
The implementation of Avrora leads to a much higher transmission error rate than in OMNeTA. If a packet is sent to a unicast address using Low Power Listening, it is



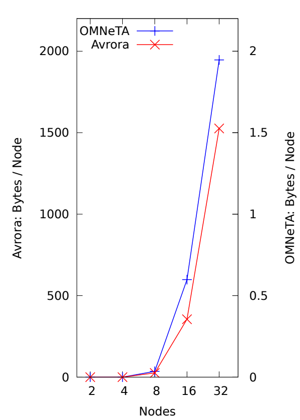
(a) Distribution: Mean Energy Consumption



(b) Bytes Sent



(c) Bytes Received



(d) Bytes Received Corrupted

Figure 4: Energy consumption and communication statistics for the application PingPong. The distribution visualizes minimum and maximum energy consumption plus all three quartiles. The graph for corrupted received bytes in OMNeTA is plotted against the Y2-axis.

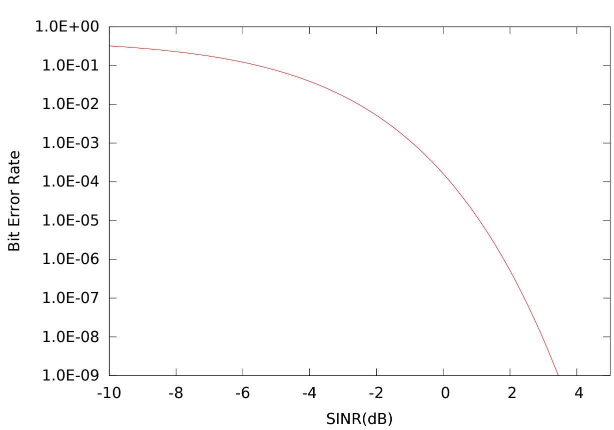


Figure 5: Bit error rate model for IEEE 802.15.4 receivers as a function of the SINR.

repeated for a certain amount of time until the destination node returns an acknowledgement. Both the packets themselves as well as the acknowledgements are affected by the increased error rate, resulting in more transmitted as well as received data. This also causes an increased energy consumption.

3.2.3 Summary

The simulation results of Avrora and OMNeTA differ by a significant amount. However, considering the laws of physics and the at least questionable modeling of colliding transmissions in Avrora, OMNeTA produced the more reasonable results.

3.3 Performance

OMNeTA introduces additional overhead in terms of inter-process communication and inter-simulator synchronization. Our performance evaluation of OMNeTA focuses on rather high-load conditions to determine the influence of those modifications. We simulated sensor networks consisting of 2 up to 32 nodes running the application Flood, arranged in a grid topology with an inter-node spacing of 5 m and 20 m. The 5 m-Topology represents a special case: All nodes are able to communicate directly with each other. We measured the wall time required to simulate 60 s using the Unix command `time`.

3.3.1 Measurments

The simulation time measurements of all repetitions have been aggregated by choosing the median to deal with changing load conditions at our simulation system. Figure 6 shows the resulting simulation time per node, OMNeTA plots refer to the second y-axis. Data points for OMNeTA with a network size of 32 nodes are missing, the simulations did not complete because of an error.

Comparing the simulation time per node in Avrora and OMNeTA, the latter one initially takes about 120-150 s longer than Avrora. Adding more nodes to the simulated network increases the time required for each node. This slowdown is less pronounced in case of Avrora, OMNeTA is about 2 to 3 times stronger affected. Simulations with the sparse topology appear to scale worse than the ones using the dense 5 m grid spacing.

3.3.2 Discussion

The graphs in figure 6 indicate a significant performance impact of the inter-process communication and synchronization in OMNeTA. We obtained similar graphs, but without the huge offset of 120 s and more for OMNeTA, by simulating the same application without Low Power Listening. Presumably this difference is caused by many radio state transitions initiated by the Low Power Listening mechanisms, leading to an excessive amount of synchronization requests.

Both Avrora and OMNeTA performed better in simulating the sensor network with a dense 5 m grid than with the sparse 20 m topology, even if the amount of sent and received data did not differ much. We assume this is an artifact of synchronization effects, too. All nodes within the dense topology are able to communicate directly with each other, which enables two possible behavioral patterns to interact with the synchronization mechanisms of OMNeTA:

1. The application flood periodically alternates between a phase of communication and an idle phase. As long

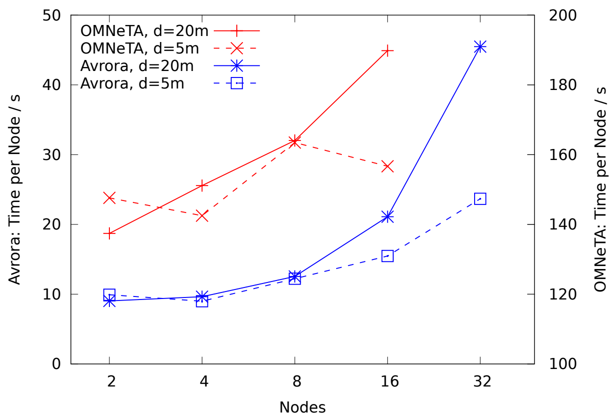


Figure 6: Mean simulation time per node running the application Flood in a grid topology of 5 m and 20 m inter-node spacing. The graphs for OMNeTA have been plotted against the Y2-axis.

as at least one node is still flooding the message, all nodes are active and either receiving or trying to send it by themselves. Therefore all nodes will enter the idle phase almost at the same time as soon as the last packet of a cycle has been sent. This self-clocking property reduces the time skew between the sensor nodes and therefore both the amount and costs of synchronizations required due to periodical Low Power Listening wakeups.

2. In opposite to the sparse topology, the nodes do not really flood the message across the network. They rather try to send it all at nearly the same time, which is closer to a short burst transmission because of the collision avoidance mechanisms of Low Power Listening. This shortens the communication phase and thereby the synchronization costs.

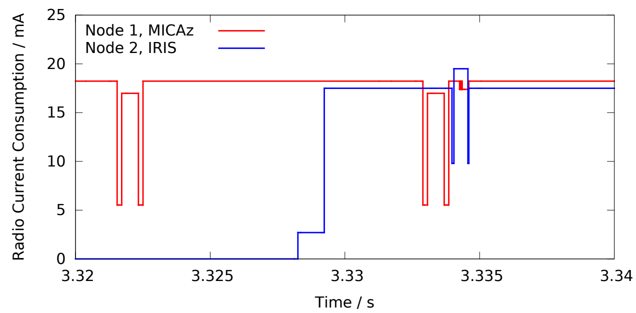
3.3.3 *Summary*

Simulating sensor networks in OMNeTA is significantly slower compared to Avrora. Our results suggest a strong influence of the inter-process communication delay on the simulation performance. The amount of data exchanged between Avrora and OMNeT++ appears to be far less important.

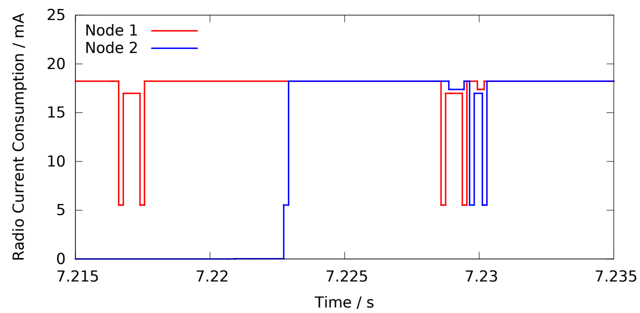
3.4 Heterogeneous network

The previous evaluation scenarios incorporated only homogeneous sensor networks using MICAz nodes. We conclude our evaluation of OMNeTA with a simulation of a heterogeneous network consisting of one MICAz and one IRIS node, running the PingPong application. In comparison, we show results of the same application emulated by Avrora on MICAz nodes only.

The sensor nodes have been chosen for two reasons: As first, they are compatible with each other but use different radio devices and as second, they use the same microcontroller clock frequency. This is required because of an implementation limitation of Avrora which has not been addressed by now.



(a) OMNeTA: Simulation with MICAz and IRIS sensor nodes



(b) Avrora: Simulation with two MICAz nodes

Figure 7: Current consumption of two sensor nodes running PingPong.

3.4.1 *Measurements*

Avrora records the current consumption of each device separately, which allows us to omit visual clutter caused by microcontroller activity. The graphs in figure 7a show a short interval of the current consumption of the radio devices in the heterogeneous network simulated by OMNeTA. Transmissions of the MICAz node (1, red) can be observed both at the beginning and at the last quarter of the graph. The IRIS node (2, blue) returns from sleep mode after the first transmission but early enough to receive the second one. The graph indicates an outgoing transmission afterwards, recognizable by the shortly increased current consumption. The MICAz node in turn receives this transmission, the current consumption is slightly decreased during this time.

The same application has been simulated in Avrora using a heterogeneous network. Figure 7b depicts the resulting graphs. Both nodes perform the same state transitions and actions. Hence, the explanations above apply to the heterogeneous case, too.

3.4.2 *Discussion*

Since Avrora itself is not able to simulate heterogeneous wireless sensor networks, a direct comparison between Avrora and OMNeTA as done in section 3.2 is not possible. However, since the energy consumption graphs reveal internal hardware state changes, it is possible to compare and analyze the state sequence and timing behavior of the emulated sensor nodes. The graphs in figure 7 clearly demonstrate a working Low Power Listening communication between two different node and radio types in OMNeTA, similar to the heterogeneous case in Avrora. The MICAz node succeeds in sending its message to the IRIS node at the second visible attempt. The latter one responds by sending a short acknowledgement back. Both nodes perform the same state changes and the temporal sequence of interactions between them is correct, too.

3.4.3 Summary

At a time scale of single simulation events and radio packets, sensor nodes exhibit the same communication behavior and radio usage patterns independent of the used simulator. This proves the synchronization facilities of OMNeTA to work correctly as designed, even in case of heterogeneous sensor networks.

4. CONCLUSIONS

We presented OMNeTA, a hybrid simulation tool for a realistic evaluation of wireless sensor networks. OMNeTA uses Avrora to emulate sensor nodes, MiXiM to simulate their radio communication and OMNeT++ to provide a flexible and extensible simulation infrastructure. Leveraging their individual strengths enables OMNeTA to offer high-quality energy and radio models as well as the possibility to simulate heterogeneous sensor networks. By combining both hardware emulation and network simulation, OMNeTA facilitates a development process similar to rapid prototyping. Before a protocol or application is costly implemented for real sensor nodes, a first evaluation of design choices may take place using a much simpler OMNeT++ models.

OMNeTA bridges the gap between the low abstraction level of Avrora and the much higher abstractions used in OMNeT++/MiXiM. A tailored synchronization mechanism ensures simulation correctness while keeping the overhead low.

Evaluating OMNeTA by comparing it to Avrora did not reveal any issues in OMNeTA, but an at least questionable design decision in Avrora. There is absolutely no relation to reality in the way how collisions of transmissions are modeled in Avrora. The simulation results of OMNeTA were plausible and consistent.

The current implementation of OMNeTA provides a foundation for highly heterogeneous network simulations incorporating hardware emulation by Avrora. Node to node communication has been proven to work, but OMNeT++ simulation models will need to use an additional translation layer to be able to interact with emulated nodes. This layer is not implemented yet. Additionally, real protocol implementations are required to allow OMNeT++ peers to interact with nodes emulated by Avrora. Many OMNeT++ protocol models are sufficiently precise simulations but do not really implement the protocol itself. Suitable IEEE 802.15.4 [5] and 6LoWPAN [4] models exist but are not integrated into OMNeTA yet.

OMNeTA already archives a high simulation quality. Hence, future work should focus on enhancements of both the inter-process communication facilities in OMNeTA and the synchronization mechanisms.

Avrora offers potential for improvements, too. The whole radio communication subsystem would benefit from a redesign. The maybe most important aspects are the way how colliding transmissions are handled and the transmission timing computations.

The source code of OMNeTA as well as further documentation is available at:
http://telematics.tm.kit.edu/english/projects_omneta.php

5. REFERENCES

[1] D. Blazakis, J. McGee, D. Rusk, and J. Baras. ATEMU: a fine-grained sensor network simulator. In *Prooceedings of SECON '04, First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, IEEE SECON 2004, pages 145–152, Oct 2004.

[2] C. Haas, V. Stöhr, and J. Wilke. Realistic Simulation of Energy Consumption in Wireless Sensor Networks. In *Proceedings of the 9th European Conference on Wireless Sensor Networks (EWSN)*, pages 82–97. Springer, Feb. 2012.

[3] IEEE. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs) (IEEE Std 802.15.4-2006)*. Institute of Electrical and Electronics Engineers, Inc., Sept. 2006.

[4] M. Kirsche and J. Hartwig. A 6LoWPAN Model for OMNeT++: Poster Abstract. In *Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques*, SimuTools '13, pages 330–333, ICST, Brussels, Belgium, 2013. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[5] M. Kirsche and M. Schnurbusch. A New IEEE 802.15.4 Simulation Model for OMNeT++ / INET. In *Proceedings of the 1st International OMNeT++ Community Summit (OMNeT 2014)*, 09 2014.

[6] A. Köpke, M. Swigulski, K. Wessel, D. Willkomm, P. T. K. Haneveld, T. E. V. Parker, O. W. Visser, H. S. Lichte, and S. Valentin. Simulating Wireless and Mobile Networks in OMNeT++ the MiXiM Vision. In *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, Simutools '08, pages 71:1–71:8, ICST, Brussels, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[7] D. Moss and P. Levis. BoX-MACs: Exploiting Physical and Link Layer Boundaries in LowPower Networking. Technical report, Stanford Information Networks Group, 2008.

[8] F. Österlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt. Cross-level sensor network simulation with cooja. In *LCN*, pages 641–648. IEEE Computer Society, 2006.

[9] E. Perla, A. O. Catháin, R. S. Carbajo, M. Huggard, and C. Mc Goldrick. PowerTOSSIM z: realistic energy modelling for wireless sensor network environments. In *Proceedings of the 3nd ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*, PM2HW2N '08, pages 35–42, Vancouver, British Columbia, Canada, 2008. ACM.

[10] B. L. Titzer, D. K. Lee, and J. Palsberg. Avrora: scalable sensor network simulation with precise timing. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, IPSN '05, Piscataway, NJ, USA, 2005. IEEE Press.

[11] A. Varga et al. The OMNeT++ discrete event simulation system. In *Proceedings of the European simulation multiconference (ESM'2001)*, volume 9. sn, 2001.