[RHS08] *Andreas Reinhardt, Matthias Hollick, Ralf Steinmetz;* **An Approach towards Adaptive Payload Compression in Wireless Sensor Networks;** In: Proceedings of the 7th GI/ITG KuVS Fachgespräch "Drahtlose Sensornetze", Berlin, Germany, September 2008. Seite

An Approach towards Adaptive Payload Compression in Wireless Sensor Networks

[Extended Abstract]

Andreas Reinhardt, Matthias Hollick, Ralf Steinmetz
Multimedia Communications Lab
Technische Universität Darmstadt, Merckstr. 25, 64283 Darmstadt
Andreas.Reinhardt@kom.tu-darmstadt.de

ABSTRACT

Most nodes in current wireless sensor networks are battery-powered and hence strongly constrained in their energy budget. While a variety of energy-efficient MAC protocols specifically tailored to sensor networks has been developed, the data rate limitation of the underlying hardware still represents a lower bound for the time required to transfer packets, and thus directly contributes to the energy requirement for transmissions. Further energy savings for given platforms can only be achieved by downsizing the packet, e.g. by means of in-network processing or data compression. In this paper, we present our approach towards an adaptive packet compression framework for sensor network applications that compresses sensor data with the locally optimal energy efficiency ratio.

1. INTRODUCTION

Wireless Sensor Network (*WSN*) deployments commonly comprise sensor nodes which distributedly take measurements, process the data, and subsequently forward the results to other nodes or external sinks [1]. Most existing platforms are powered by batteries, and hence inherently limited in their energy budgets. Once the entire battery charge has been consumed, the nodes stop operating and need to be replaced. Assuming a constant battery charge, there is a linear relation between power consumption and node lifetime, wherein the lifetime decreases with rising current consumption. Even solar powered nodes, such as the Trio platform [4], cannot completely tackle this challenge, as they rely on deployment in areas regularly exposed to direct sunlight.

It follows from the correlation between current consumption and node lifetime that maximizing a node's lifetime can only be achieved by minimizing its overall energy requirements. As available sensor node platforms commonly employ dedicated radio transceivers, sensors and peripherals, selectively disabling these components allows to measure the node's energy consumption in different configurations. Exemplarily, the current consumption figures for the widely used *tmote sky* platform have been taken from the datasheet [9] and were plotted against each other in Figure 1. It is clearly visible that the radio transceiver exhibits a power consumption that is about one order of magnitude higher than the corresponding values for the used microcontroller.

As the employed CC2420 radio transceiver does not provide low-power modes, a common solution to the problem of its comparably high energy consumption is limiting the time of its operation and putting it into sleep mode for the remaining period. Energy-efficient MAC protocols specifically tailored to sensor networks, such as [7, 11, 3, 6], make use of such schedules and thus allow for significant energy savings. We anticipate that combining such MAC protocols with supplementary extensions to minimize the number of packet transmissions and the corresponding payload sizes can lead to even higher savings and an extended node lifetime.

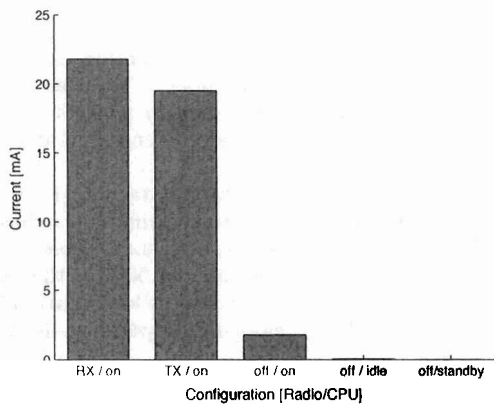

Figure 1: Power consumption of tmote sky nodes (numbers taken from [9])

This paper compares two methods to reduce traffic in sensor networks (see Section 2) and outlines their mode of operation. Special emphasis is put on their applicability in WSNs, as resource-constrained devices generally exhibit characteristics that differ from common desktop computers. Subsequently, we present our approach towards an adaptive packet compression framework for sensor network applications in Section 3. By compressing sensor data with the locally optimal energy efficiency ratio, energy can be preserved and thus the node lifetime extended. The description of our vision is followed by an analysis of related work. Finally, conclusions and an outlook will be presented in Section 4.

2. REDUCING TRAFFIC IN THE WSN

Although several methods of reducing the size or number of packets in WSNs exist, we introduce two common solutions in this section. Data aggregation tries to optimize the number of transmissions throughout the sensor network, while data compression strives for a local optimization of the amount of transmissions and their payload size.

2.1 Data Aggregation

If incoming packets and locally generated data are addressed to the same destination, combining these data sets into a single packet can effectively reduce the number of transmissions required [5]. For example, consider the linear topology depicted in Figure 2. The rightmost node (S) acts as a sink to sensor readings from the remaining nodes (A, B, C).

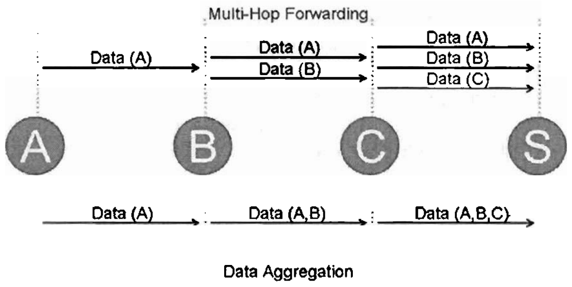

Figure 2: A linear sensor network topology

If only one-hop connectivity is given, a total of six transmissions are required to forward all measured data to the sink (depicted as the multi-hop forwarding case). Another downside of this issue is that node C, located next to the sink, must forward a higher count of packets than A or B and will thus most likely deplete its energy budget first.

Aggregating the incoming sensor data with local readings and forwarding these results to the sink can reduce the required number of transmissions from six to three in this basic scenario, hence achieving savings of 50% in the number of transmissions. This is indicated in the lower part of Figure 2, where only a single aggregated data packet is forwarded by each node.

Although node distributions in real deployments might considerably differ from this example, it should be remarked that the number of packets cannot increase by applying data aggregation.

2.2 Data Compression

Opposed to the previously described mechanisms of data aggregation, data compression targets to locally reduce the packet payload size by increasing the information density within a packet. In general, compression algorithms can be classified as either lossless or lossy. While some applications (such as software updates) require a bitwise correct transmission of data, intrinsic offsets in sensor hardware might reduce the achievable accuracy and thus provide an application area for lossy compression.

A variety of different approaches exist, and will be briefly introduced in Section 3.1. While generic compression algorithms often achieve good average compression ratios on various types of input data, algorithms adapted to specific applications often provide better performance, however at the cost of being limited to a narrow application area.

A common metric to evaluate the savings achieved by data compression is the energy cost per bit. Data compression is only feasible in terms of energy efficiency if the computational efforts to reduce the payload size by one bit are lower than the energy required for transmitting this additional bit. Depending on the used components, current sensor node platforms can perform 4.000 to 2.000.000 CPU cycles [8] instead of transmitting a single byte, while still maintaining a positive energy balance.

3. VISION

The vision of our research is to develop a *modular compression framework*, which can be deployed on nodes in a WSN. It comprises a set of different compression and decompression algorithms, which can be adaptively selected, and even combined to achieve additional savings. By compiling the algorithms into packages, different compression methods can be dynamically selected, exchanged and updated during runtime.

As many current sensor network applications follow predefined structure definitions in their output data formatting, different compression algorithms will typically produce differently sized output. The framework can either rely on application-defined preferences regarding the preferred compression engine, or compress an exemplary set of sensor data with its available compressors prior to selecting the algorithm with the highest energy efficiency. While this analysis might consume more energy before regular operation, it only needs to be performed once and will subsequently use the locally optimal compression algorithm.

Additionally, applications can also specify delay bounds for the compression of their packets. This mechanism ensures that lengthy algorithms are avoided when low latency transmissions are required. The framework is targeted to allow dynamic updates and modifications of the compressors and/or compressor parameters during runtime. This allows to exchange slow implementations of algorithms with optimized ones and tune parameters to fulfill the application's needs.

3.1 Related Work

A variety of data compression (and decompression, respectively) algorithms are well-known, although only few of them were specifically tailored to fit the needs of sensor networks. The emerging field of sensor networks however poses different constraints on resulting algorithms than common desktop computers do. Both processing power and available memory are tightly limited in sensor nodes, and their tight energy budget must additionally be considered. For these reasons, heavyweight compressors, such as partial predictive matching (PPM), or similar algorithms that require big dictionaries to be stored in RAM, cannot be run on the nodes. The even more sophisticated PAQ compressor quotes a minimum RAM consumption of 35 megabytes, although values range up to 1712MB[1]. Barr and Asanović performed an analysis of several compression algorithms on a Compaq Personal Server handheld device in [2]. The platform featured a 233MHz CPU with 32 megabytes of RAM, rendering a direct mapping of the results to sensor node platforms equipped with low-power CPUs and a few kilobytes of RAM impossible.

Other authors have noticed this discrepancy between highly demanding but efficient compression algorithms and sensor node capabilites, and have thus designed algorithms specifically adapted to sensor networks, such as S-LZW [8] and SBZIP [10]. However, Tsiftes et al. have mainly regarded software updates that were compressed before deployment in [10]. Software updates are deployed less frequently than data packets, and even when a higher number of packets are required to transmit an application image, the necessity to compress packet payloads still persists.

4. CONCLUSION

Limited energy budgets of sensor network nodes make data compression a necessity when energy savings are required. However, it is very likely that there is no universally optimal compression algorithm for sensor network data. Instead, the need to adapt to the payload contents and dynamically choose the appropriate compression algorithm that yields best compression rates whilst preserving energy is arising to keep the energy consumption low and thus enhance node lifetime.

Compression algorithms implemented for desktop computers are generally not suited for sensor nodes, as they have high requirements towards CPU and memory size. However, a lot of existing approaches and concepts can be applied in sensors networks in modified forms. We propose the use of a modular compression framework that can be dynamically updated during runtime, while automatically detecting the locally optimal compression scheme for application-specific data.

When data from different sources is present in sensor nodes, generic compression algorithms will not always yield optimum results. In such application scenarios, we expect our compression framework to outperform solutions that rely on a single static compressor and thus achieve high energy savings.

5. REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A Survey on Sensor Networks. *IEEE Communications Magazine*, 40:102–114, 2002.

[2] K. Barr and K. Asanović. Energy Aware Lossless Data Compression. In *Proceedings of the First International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2003.

[3] M. Buettner, G. Yee, E. Anderson, and R. Han. X-MAC: A Short Preamble MAC Protocol For Duty-Cycled Wireless Sensor Networks. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (SenSys)*, 2006.

[4] P. Dutta, J. Hui, J. Jeong, S. Kim, C. Sharp, J. Taneja, G. Tolle, K. Whitehouse, and D. Culler. Trio: Enabling Sustainable and Scalable Outdoor Wireless Sensor Network Deployments. In *Proceedings of the Fifth International Conference on Information Processing in Sensor Networks (IPSN)*, 2006.

[5] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2000.

[6] B. H. Liu, N. Bulusu, H. Pham, and S. Jha. CSMAC: A Novel DS-CDMA Based MAC Protocol for Wireless Sensor Networks. In *Proceedings of the 2004 IEEE Global Telecommunications Conference Workshops (GLOBECOM)*, 2004.

[7] J. Polastre, J. Hill, and D. Culler. Versatile Low Power Media Access for Wireless Sensor Networks. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys)*, 2004.

[8] C. M. Sadler and M. Martonosi. Data Compression Algorithms for Energy-Constrained Devices in Delay Tolerant Networks. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (SenSys)*, 2006.

[9] Sentilla Corp. Tmote Sky Datasheet. Online: `http://www.sentilla.com/moteiv-endoflife.html`, 2007.

[10] N. Tsiftes, A. Dunkels, and T. Voigt. Efficient Sensor Network Reprogramming through Compression of Executable Modules. In *Proceedings of the 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, 2008.

[11] W. Ye, J. Heidemann, and D. Estrin. An Energy-Efficient MAC protocol for Wireless Sensor Networks. Technical Report ISI-TR-543, Information Science Institute (ISI) / University of Southern California (USC), 2001.

[1] According to the developer homepage at `http://www.cs.fit.edu/~mmahoney/compression/`