C-SDF: Practical Solar-aware Distributed Flow Control

Immanuel Schweizer, Tobias Petry, Max Mühlhäuser
Technische Universität Darmstadt, Telecooperation Lab
Email: schweizer@cs.tu-darmstadt.de

***Abstract*—Energy harvesting in wireless sensor networks leads to an interesting optimization problem: Given the spatial-temporal distribution of the energy harvested, how can nodes adapt their consumption to reach energy neutral operation, i.e., consume exactly the harvested amount of energy.**

Adapting the energy consumption of any node in the network requires control of the data flow routed through that node by arbitrary other nodes. This calls for flow control, which was mostly investigated for field monitoring networks in the past. Pertinent publications are usually based on simulations or artificial lab studies, results hardly carry over to the real world.

In this paper, we present C-SDF, the first Contiki implementation of any solar-aware distributed flow control approach. Different runs on two distinct testbed sites show an outstanding performance. Even under adverse network conditions, the sampling rate stabilizes quickly. Energy utilization is more than 80%, while ensuring no node is overloaded.

***Index Terms*—flow control, solar harvesting, wireless sensor networks, periodic reporting**

I. Introduction

Energy harvesting is often cited as a solution to the issue of battery-limited lifetime in wireless sensor networks [1], [2]. With batteries the challenge is to prolong the lifetime of the network. With harvested energy the challenge shifts to providing the best possible service, while no node will ever deplete. This is called the energy neutral operation.

This leads to the following optimization problem: Given the spatial-temporal distribution of the energy production, how can nodes adapt their energy consumption to reach energy neutral operation. Here, the definition of best possible service and the consumption of each node depend on the application. We focus on one of most prevalent applications in sensor networks: periodic reporting. In periodic reporting, all nodes periodically collect data and send it to a common base station. The quality of service is given by the time between collected data points, called sampling rate. Higher sampling rates lead to better sensor coverage, hence, a better service. Obviously, consumption of each node also depends mostly on the sampling rate and the resulting data packets forwarded to the base station. This is also the main challenge. Forwarding data packets implies that each nodes energy consumption depends to some extend on remote nodes.

This challenge was first introduced for homogenous sampling rates by Kansal et al. [1]. And it has lead to the introduction of different flow control mechanisms [3], [4], enabling nodes to control their sampling rate and the flow they need to forward for remote nodes. Fan et al. [4] provided the first centralized and distributed solution. Their theoretical results are very compelling. However, all flow recalculations are done on the complete path to the base station and back to the nodes. Additionally, they assume a static routing topology. Hence, the approach is not applicable to real networks. In 2011, we have introduced the solar-aware distributed flow (SDF) approach [3] as the first practical flow control heuristic. In SDF nodes distribute control messages only to their children in the routing tree. This enables almost instant adaptability to changes in the routing tree with neglectable overhead. However, SDF was only evaluated using simulations and not implemented in hardware. The interaction between energy utilization, flow control, and routing in real networks is still unclear.

In this paper, we evaluate flow control as part of the communication stack and validate the practicality of the SDF approach on real hardware. To this end, we present C-SDF, a implementation of SDF for the Contiki operating system[1]. C-SDF is integrated into the standard Contiki communication stack. It is implemented to work on top of *Routing Protocol for Low power and Lossy Networks* (RPL) [5] as routing protocol.

In this paper, we present results from four runs on two different testbeds. These runs are up to 7 days in duration. On average, nodes utilize around 80% of the energy available, even under poor network conditions. With delivery rates in the testbed dipping below 70%, SDF was still able to stabilize fast and increase the sampling rate from 5 to 30 messages per hour on average.

The remainder of this paper is organized as follows. Section II introduces the related work. The implementation of C-SDF is described in Section III. Section IV presents the results of the evaluation. Section V concludes the paper.

II. Related Work

Energy harvesting describes the concept of powering sensor nodes using renewable energy sources. The feasibility of energy-harvesting wireless sensor networks has been studied extensively. This section provides an overview by first introducing possible power sources and hardware for energy harvesting. Afterward, different solar-aware algorithms are introduced. Last, we introduce other flow or rate control algorithm related to SDF.

[1]C-SDF is open source and can be downloaded at: https://github.com/ischweizer/C-SDF

978-1-4673-6770-7/15/$31.00 ©2015 IEEE

Roundy et al. [2] review a great variety of potential power sources for wireless sensor networks. Besides the classical power storage in batteries, micro fuel cells, and micro heat engines, they also research the potential of different energy-harvesting techniques. These techniques include photovoltaic (i.e., solar panels), temperature gradients, human power, wind/air flow, and vibrations. Of these, solar panels offer the highest power density and are, therefore, the most likely candidate for reliably powering wireless sensor networks. Another advantage of solar energy is the predictability [1].

Predicting the availability of harvested energy is the basis of any harvesting sensor network. The prediction is used to adapt the duty cycle during times with no or insufficient energy [6] or to assign tasks to nodes with more energy [7].

Several working prototypes of wireless sensor nodes using solar power have been developed, showing that solar-powered wireless sensor networks are indeed feasible. Raghunathan et al. [8], [9] present the *Heliomote* (cf. Fig. 1), a Mica2 [10] mote enhanced with a circuit board equipped with a solar panel and NiMH batteries for solar energy harvesting.

Fig. 1: Heliomote [8], [9]

They show that, in principle, their device is capable of nearly perpetual operation. Jiang et al. [11] design and implement *Prometheus*, a wireless sensor node based on the Telos-mote [12] with a solar panel, supercapacitors and Li-ion batteries. They present and discuss results from a 10-day period. Minami et al. [13] develop and present *Solar Biscuit*, a wireless sensor node without a battery, solely relying on a solar panel and a supercapacitor. They do not present any results of an actual long-term deployment. Sikka et al. [14] present *Fleck1*, a solar-powered wireless sensor node using a solar panel and NiMH batteries. Their main contribution is the incorporation of a DC-DC converter enabling deeper battery discharge cycles between periods where solar power is available. The authors in [15] present results of a network operating 24/7 for over two years. Alippi et al. [16] develop and present a wireless sensor network framework based on their own solar-powered sensor nodes. The wireless sensor network is deployed in Moreton Bay, Brisbane, Australia, to deliver temperature and luminosity data of the marine ecosystem. They present and discuss results from a four-day period.

To make the operation of environmentally powered wireless sensor networks more efficient, several solar-aware protocols have been developed. Some focus on solar-powered wireless sensor networks, while others use a more general approach.

Lin et al. [17] develop E-WME (Energy-opportunistic Weighted Minimum Energy), an energy-aware routing algorithm for wireless sensor networks with any environmental energy sources. They show that their routing scheme is asymptotically optimal. Voigt et al. [18], [19], [20] present two variants of directed diffusion [21] modified to route in a solar-aware manner. Both protocols route packets preferably via nodes that are currently solar-powered and, thus, can relay the packet without using their battery. The authors present simulation results of both protocols and conclude that they offer significant energy savings. In another work, Voigt et al. [22] present solar-aware clustering protocols. Some more theoretical works focus on channel performance. Rajesh et al. [23] for example find the Shannon capacity when data is transmitted over an additive white gaussian noise channel. However, none of these approaches on optimizing environmentally powered wireless sensor networks takes the utilization of harvested power into account.

Some related work exists on the topic of assigning sampling rates in wireless sensor networks. Shu et al. [24] try to optimize the network sampling rates in terms of a scheduling problem without taking energy consumption into consideration. Bandyopadhyay et al. [25] analyze what trade-offs in sensor density, energy usage, throughput, and delay have to be made to achieve certain temporal and spatial sampling rates. But those sampling rates are always fixed and predefined. A more sophisticated idea is introduced by Lachenmann et al. [26]. They define different lifetime goals for a network and adjust the operation of any node to meet these goals by introducing different levels of operation. However, harvested energy is not taken into account. Thus, infinite lifetime goals and dynamic sampling rates are not considered.

The problem of maximizing sampling rates in solar harvesting sensor networks was first discussed by Kansal et al. [1]. They describe a field-monitoring application with the goal of maximizing homogeneous sampling rates while remaining energy neutral. They model the flow in the network based on linear equations. Solving the linear equations yields one sampling rate and a corresponding flow. This flow can be sustained by all nodes using no more than the energy neutral consumption rate. To calculate the solution, global knowledge over all nodes, flows, and energy consumptions in the network is required. This is infeasible in a real-world sensor network.

A more viable approach was presented by Fan et al. in [4]. They introduce a centralized and a distributed solution (DLEX) to the problem of fair rate allocation. The authors are able to show that both approaches achieve an optimal lexicographic rate assignment. They implemented the DLEX approach in the MoteLab testbed. However, they themselves state in the paper that it only works *"...when each source has a fixed route and the routing path is known"*. Unfortunately, this limitation meant, we were unable to integrate DLEX into the communication stack to compare it against our approach. Changes in the routing tree would degrade DLEX's performance too much and, thus, not allow for a fair comparison. To the best

of our knowledge, DLEX is the only related approach to SDF. Thus, we cannot compare C-SDF against any other approach in the evaluation.

Later, we introduced SDF [3]. To change the rate assignment nodes need only contact their children in the routing tree. Thus, SDF features much lower overhead and higher resilience against adverse routing performance. However, we showed results only from a stimulative study. No approach has yet been integrated into a standard communication stack. Thus, the interaction between routing or network performance and flow control in solar-aware networks are yet unknown.

III. Implementation

SDF [3] introduced the first approach for flow control relying on local knowledge only. However, real-world applicability and the interaction with routing protocols remained unclear. The contribution in this paper is a first implementation for the popular Contiki platform and evaluation results from different comprehensive testbed runs, illustrating the performance of the approach. Implementing SDF on hardware leads to significant changes to the protocol.

We will first give a short introduction of SDF (for more details, please refer to [3]), before introducing and discussing the adaptations required for SDF to run in hardware.

A. SDF Recap

The goal of solar-aware distributed flow (SDF) is to maximize the sampling rates achieved by each node while ensuring energy neutral consumption. SDF assumes a periodic reporting application. The nodes form a collection tree rooted at a base station. Each node has one successor or parent node and might have multiple predecessor P in the routing tree. We note that, in this scenario, energy is mainly consumed by either generating or relaying flow. Each node will sense the environment periodically, thus, generating and transmitting sampling messages. Each node will also forward sampling messages for all predecessors P in the collection tree.

SDF is based on the idea of sending control messages only to the direct predecessor nodes (D), granting them an *allowed flow*. Each node will periodically repeat three main steps:

1) Predict the *consumption rate* c.
2) Calculate the own *sampling rate* and the *allowed flow* of all predecessors in P.
3) Send control messages to all direct predecessors in P and set the own sampling rate.

Algorithm 1 provides a sketch of one execution of SDF. Here, again P is the set of all predecessors in the collection tree. D is the set of direct (1-hop) predecessors and P_p is the subtree rooted at predecessor p. Algorithm 1 is repeated periodically on each node to yield a maximized sampling rate, utilizing as much harvested energy as possible. Given the excellent simulation results [3], we were interested in integrating SDF into the communication stack to evaluate it in a real testbed.

The hardware implementation leads to a number of changes to SDF. The important changes are highlighted in the next section. Some changes are purely driven by shortcomings in the testbed, e.g., no solar cells and batteries are connected. These changes will be highlighted in the testbed description of Section IV.

Algorithm 1: Solar-aware distributed flow

Require: Allowed flow f_a from successor
Ensure: Sampling rate x and allowed flow f_a to direct predecessors
$c \leftarrow$ CONSUMABLE-POWER
$x \leftarrow$ CALCULATE-FLOW(c, f_a)
$D \leftarrow$ GET-DIRECT-PREDECESSOR-SET(P)
for all $p \in D$ **do**
  $P_p \leftarrow$ GET-FORWARDED(P, p)
  SEND-FLOW-UPDATE-MESSAGE$(pred, x \cdot$SIZE$(P_p))$
end for
SET-SAMPLING-RATE$(max(x_{min}, x))$

B. C-SDF

The previous section introduced the basic SDF approach. However, several changes had to be made to integrate SDF with the Contiki communication stack. We will now describe C-SDF, the implementation of SDF for Contiki.

1) Synchronization: SDF assumes execution rounds, which cannot be assumed in real networks. The sampling periods of the parent and child node may not be synchronized, leading to errors in the calculation of the required energy. To mitigate this, C-SDF assumes a fixed sampling period S. Here, S is the time interval between two SDF control messages and, thus, equivalent to one round of SDF. Whenever a control message is received a node will start a new sampling interval. At the end of each sampling period a small time buffer is introduced to mitigate small synchronization offsets. The remaining sampling period is dedicated to the application, i.e., sampling uniformly given the maximized sampling rate. This simple synchronization scheme is accurate enough for C-SDF to work.

2) Sampling Rate: The goal is to maximize the sampling rate of each node during the sampling period S given a prediction of the amount of harvested energy (consumption rate c). Given S, the sampling rate is measured as messages m per sampling period ($\frac{m}{S}$). However, a node can only send a discrete number of messages per sampling period. Hence, the number of messages is floored to the next integer. This will reduce energy utilization, but is done on purpose as a buffer against rapid changes in harvested energy.

3) Communication Stack: As routing layer, C-SDF builds upon ContikiRPL [27], the *Routing Protocol for Low power and Lossy Networks* (RPL) [5] implementation for Contiki. RPL builds a destination-oriented directed acyclic graph (DODAG), where each node picks one parent and knows about its children. RPL, thus, builds a collection tree topology perfectly suited for SDF. RPL is an IETF draft and builds upon IPv6. We use uIPv6 [28] and the IPv6 stateless address

auto-configuration standard [29] to assign IP addresses in a distributed fashion. All messages are sent by UDP, i.e., without reliability on the transport layer. We use ContikiMAC [30] as MAC protocol. This gives as some reliability on the MAC layer, but only per-hop and not per-path. However, since control messages are only send to child nodes, one-hop reliability is sufficient.

4) Allowed Flow: The calculation of the allowed flow in SDF is based upon knowledge of the required power for sending (P_{tx}) and receiving (P_{rx}). However, in reality P_{tx} and P_{rx} are not constant, but fluctuate over time. Each send and receive operation requires different amounts of energy, due to, e.g., the MAC preamble, retransmits for unacknowledged messages, packet size constraints (e.g., splitting a message into multiple frames for sending), etc.

We will approximate P_{rx} by calculating a running average over the past m messages received. In Contiki, the application layer has no knowledge of the exact number of messages received on the routing layer. Since each node controls the number of messages each child node is able to forward, the maximum number of allowed messages is used as an approximation. The power required to send, P_{tx}, can be approximated similarly. We divide the amount of energy necessary to send all messages by the approximated number of messages forwarded and sent by the node itself.

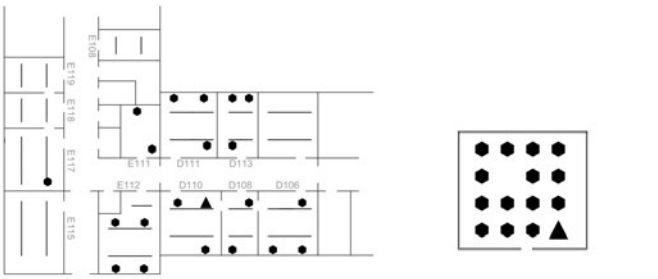
IV. Evaluation

To study the interaction of C-SDF with different network conditions and the routing protocol and to illustrate the feasibility, C-SDF has been evaluated on different testbeds and with different wireless link conditions. Unfortunately, we are unable to compare C-SDF to any related work. As discussed in the related work section, there is only one related approach, DLEX introduced by Fan et al. [4]. DLEX requires static routing trees, which we could not guarantee with RPL and given the testbed conditions. The performance of their approach would degrade to a point, where no fair comparison was possible. However, we compared both SDF and DLEX in simulation. While there is not enough space to present these results, they indicate a significant advantage for SDF.

Before presenting the results, we will introduce the setup and models used to evaluate C-SDF. Afterward, we present results from four long runs on two different testbeds. The emphasis was put on evaluating flow control with a wide variety of networks characteristics to get a better understanding of the interaction between flow control and the overall network and to validate the practicality of C-SDF.

A. Testbed sites and setup

We used two distinct sites of the TUDμNet testbed [31] in Darmstadt to evaluate SDF under completely different conditions. The first testbed is located inside the computer science building (*Piloty*) at the Technical University Darmstadt. 20 TMote Sky nodes have been deployed over eight rooms on a single floor. The location of each node is displayed in Figure 2(a). All sampling nodes (hexagonal) are transmitting their data to the base station (triangle). This testbed site has been chosen, as nodes suffer from high interference from other networks, e.g., the university Wi-Fi etc.



(a) Piloty testbed placement (b) TIZ testbed placement

Fig. 2: Placement in the two TUDμNet testbed sites

The second testbed site is located at the *TIZ* building in Darmstadt. At that time, the TIZ site had 15 TMote Sky nodes deployed in a single room. They have been arranged in a grid as displayed in Figure 2(b). Again all sampling nodes (hexagonal) are transmitting their data to the base station (triangle). At the TIZ site there is almost no interference. Thus, the TIZ site provides an estimate of the optimal network performance for C-SDF.

B. Models

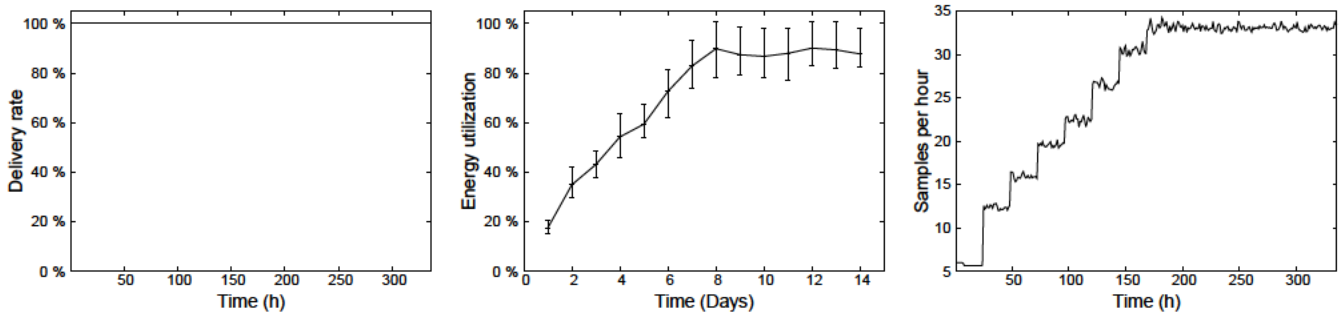
The nodes in the TUDμNet testbed feature a USB connection for charging and debugging. The nodes are neither connected to a solar panel nor a battery. Thus, we require a model for the battery, means to measure energy consumption and must simulate the solar cell. The battery in the nodes is modeled as perfect energy storage. The initial capacity is $1000mAh$ and the initial charge is 70%. In the following sections we will introduce how we measure energy consumption and model and calculate the harvested energy.

1) Energy consumption: In Contiki, the Energest framework is the standard included framework to estimate energy consumption [32]. The Energest framework delivers the number of ticks each hardware module was active. The most important consumers, CPU (active, sleep) and radio (transmit, receive), are already included by default.

In order to be more realistic, we have also added GPS, CO, and CO_2 sensors as hardware modules. For each sample, the sensors as well as the GPS device are activated for a given time. This time is picked uniformly at random and is between $[1, 5]$, $[0.5, 1]$, and $[10, 30]$ seconds for GPS, CO_2, and CO respectively.

The energy consumption of each module is summarized in Table I. The time ranges as well as the energy consumption, are extracted from the respective data sheets [33], [34], [35], [36].

2) Energy harvesting: The harvested energy is calculated similar to [3]. All values here have to be calculated on the nodes however. But floating point numbers are deactivated by Contiki. Hence, we implemented a fixed-point arithmetic



(a) Delivery rate (b) Energy utilization (c) Samples per hour

Fig. 3: Performance of C-SDF (TIZ)

Component	mAh	mAs
cpu (active)	1.8	0.0005
cpu (sleep)	0.0545	0.00001514
transceiver (send)	20.0	0.0056
transceiver (receive)	17.4	0.0048
CO sensor	3.0	0.0008
CO_2 sensor	50.0	0.0138
GPS modul	20.5	0.0057

TABLE I: Energy consumption of sensor modules

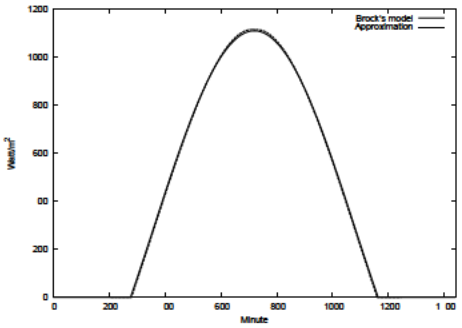


Fig. 4: Approximation of solar energy compared to Brock's model

for Contiki to circumvent this limitation. All floating point operations are calculated as Q15.16 numbers[2], which logically split a 32bit integer into 15 integer bits, 16 fractional bits and the two's complement sign bit.

Overall, the harvested energy E_h is calculated as

$$E_h = g * d * E_b$$

where g and d are damping factors discussed later and E_b is an approximation of the solar energy per square meter using Brock's model [37], [3]. Brock's model calculates the solar energy per square meter for any given day and any given location. However, given the constraint resources of the motes we devised an approximation formula. The formula approximates the energy for May 7 in Darmstadt, Germany.

The approximation in fixed-points numbers for the solar energy in $\frac{\text{Watt}}{m^2}$ for this day is given by

$$E_b \approx \begin{cases} -9.55 * 10^{-6} * x^3 + 0.01018 * x - 567.95, x \leq 720 \\ -9.55 * 10^{-6} * y^3 + 0.01018 * y - 567.95, \text{else} \end{cases}$$

where $x \in [1, 1440]$ is the xth minute of the day and $y = (1441 - x)$. The maximal error between Brock's formula and our approximation is only 0.03%, as illustrated in Figure 4.

This inbound solar energy is damped by two factors per node. The first factor g is taken from an interval $[0.9, 1.1]$. It models persistent difference in sun exposure, e.g., through placement. The second factor d is from the same interval, but recalculated per day. It models weather effects, e.g., clouds.

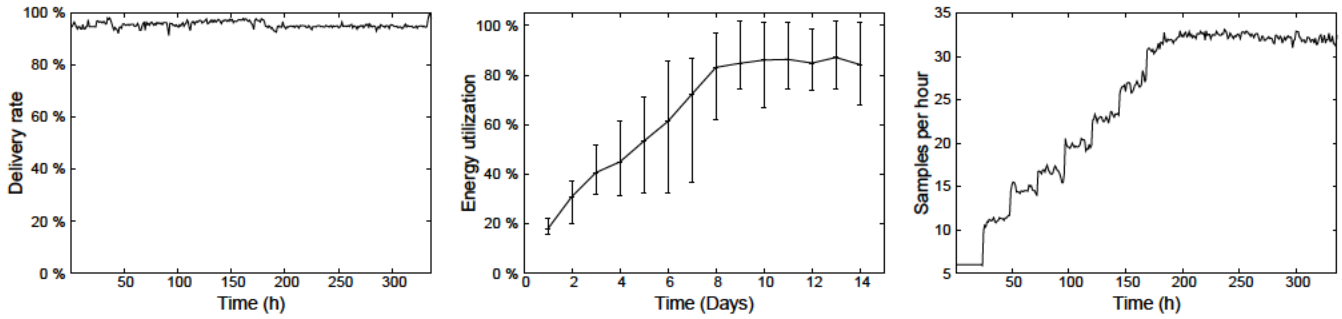
[2]The range of Q15.16 numbers is $[-32767.9999847412210937\overline{5}, 32767.9999847412210937\overline{5}]$ with a maximum fraction accuracy of 2^{-16}

To calculate the energy harvested, we also need to specify the area of the solar panel and the energy efficiency. For our following runs, the area is set to $100cm^2$ and the efficiency of the solar panel to 4%.

For a node to estimate the consumption rate c, it will measure the harvested energy E_h over the last 24 hours. The last 14 measurements are kept to calculate a very conservative 10% quantile. To take the initial uncertainty into account, this consumption is further multiplied by a *sample confidence sc* (similar to [3]), but only for the first seven days. Obviously, $sc = \frac{1}{7}$ after the first day and increased by $\frac{1}{7}$ per day, until it is equal to 1. Since it is updated once per day, we expect the energy utilization to jump accordingly over the first 7 days.

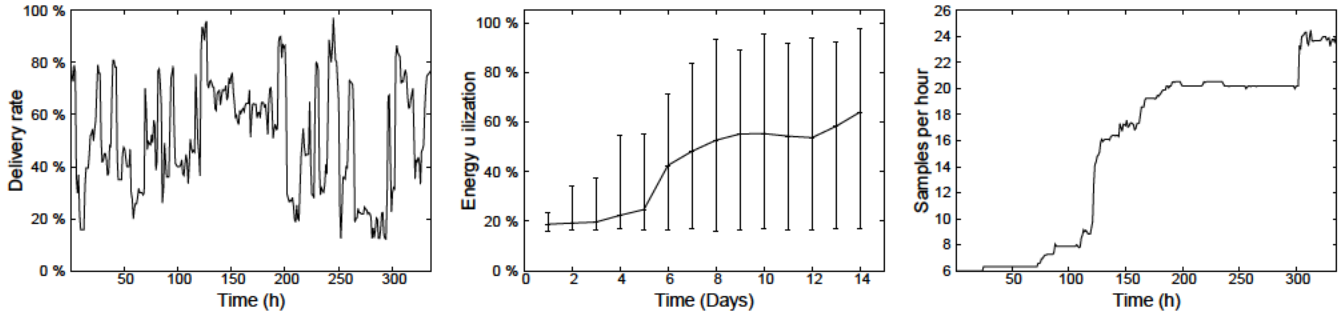
The longest evaluation run was exactly 7 days. We felt that this was not enough to evaluate long term effects of C-SDF. Fortunately, the models used for energy consumption, storage, and harvesting can be sped up using a time factor. We ran the energy simulation at eight times speed, which meant that during the 7 day run, 56 days were covered. Lastly, the initial sampling interval S was set to 10 minutes.

The next section will present the results from different testbed runs. SDF [3] achieved around 90% average energy utilization per node in simulation. With the conservative implementation choices, e.g., flooring the number of messages per sampling interval, we assume C-SDF to be a little bit



(a) Delivery rate (b) Energy utilization (c) Samples per hour

Fig. 5: Performance of C-SDF (Piloty, short run, almost no message loss)



(a) Delivery rate (b) Energy utilization (c) Samples per hour

Fig. 6: Performance of C-SDF (Piloty, short run, high message loss)

lower.

C. Results

The main contribution of this paper is the study of the first integration of solar-aware flow control into a standard communication stack. To study the performance, we present results from four different runs on the two testbed sites described above. The results of each distinct testbed run will be described below. Average and variance are calculated over all nodes in the testbed.

The first three runs are 2 days each. Given the time scaling described above, this results in 14 days of C-SDF operation. These runs illustrate the performance of C-SDF under optimal, very good and very bad network conditions. The last run is 7 days or 49 days of C-SDF operation. It illustrates the performance of C-SDF and the network in the long run.

1) C-SDF performance under different network conditions: The first run was executed on the TIZ testbed site. Due to the setup of the TIZ site, where nodes are placed in a grid without any walls and almost no wireless interference from other networks, this run is the benchmark for C-SDF under optimal networking conditions (cf. Fig. 3(a)). The results of the run are given in Figure 3(b) and 3(c).

The delivery rate in the TIZ network was 100% for both consecutive days (cf. Fig. 3(a))[3]. This is the optimal case

[3]Please note: Delivery rate is not a result of C-SDF, but a property of the network

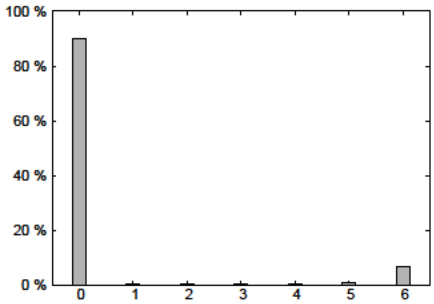
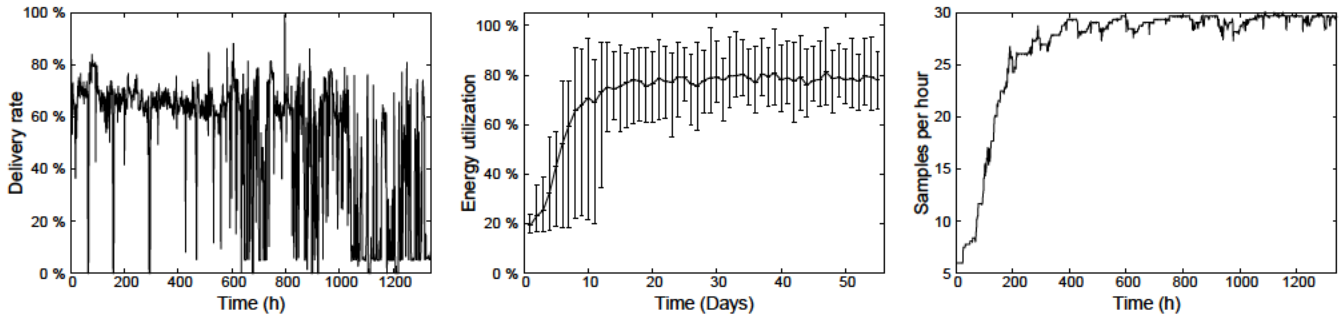


Fig. 7: Number of C-SDF messages received per hour (Piloty, short run, high message loss)

and the results indeed illustrate optimal C-SDF performance. To measure performance, we report *energy utilization* (the percentage of harvested energy that was consumed) and *samples per hour*. During the initialization phase – the first 7 days – the energy utilization increases. This is due to the increase in sampling confidence as described above. The average utilization stabilizes at over 85% (cf. Fig. 3(b)), after the initialization phase is over.

The increase of the energy utilization obviously implies an increase of the sampling rate. And indeed the number of samples increases from 6 to around 33 (cf. Fig. 3(c)). The sampling rate jumps up during the initialization phase as the



(a) Delivery rate (b) Energy utilization (c) Samples per hour

Fig. 8: Performance of C-SDF (Piloty, long run)

sampling confidence increases[4]. The jump illustrates the short stabilization time of C-SDF. The nodes adapt their sampling rate immediately as soon as more energy is available (cf. Fig. 3(c)). This run has given us the upper bound on possible C-SDF performance. The next three runs have been executed at the Piloty site. The performance of the network is more interesting, as the network is located in the computer science building and exposed to different sources of interference

The presented two shorter runs illustrate the possible spectrum in wireless performance. The first run is almost identical to the run at the TIZ site. The average delivery rate is very stable around 95%, which indicates a good network performance (cf. Fig. 5(a)). The sampling rate stabilizes very quickly during initialization and goes up to around 32 sample messages per hour (cf. Fig. 5(c)). The utilization plateaus at an average of approximately 80%. The variance is slightly higher compared to the TIZ run, but overall the results between both runs are comparable.

The network performance degraded heavily for the second run. The delivery rates exhibited over the 2 days were 50% on average, but the network degraded to below 20% at times (cf. Fig. 6(a)). This is a very demanding performance for any algorithm running on top of the network. Thus, this run is the perfect condition to study the resilience of C-SDF against detrimental performance of the network. To illustrate the impact of network performance on C-SDF, we measured the number of control messages received per hour. Figure 7 shows the distribution of C-SDF control messages received per hour for all nodes. 90% of the time, nodes have not received any updates per hour. The remaining 10% the maximum of six control messages has been received. This indicates that links were either working or down and there was no sudden degradation or recovery in performance.

The utilization of C-SDF degrades from over 80% in the last two runs to around 64% for this run (cf. Fig. 6(b). However, the utilization is still increasing at the end of the run, indicating that the utilization could plateau on the 80% level, but only after a much longer convergence time.

In summary, even though we lose 90% of the control messages, it is important to note that the average number of samples per hour is still increasing steadily and jumps to around 24 at the end (cf. Fig. 6(c)). This is really uplifting, because it indicates the resilience of C-SDF against adverse conditions in the network. Due to the low overhead of C-SDF, we could afford even shorter update cycles, further boosting resilience and adaption times.

2) Long term performance of C-SDF: The final run is supposed to study long term performance and was executed on the Piloty testbed. The run was 7 days, which translates to 56 days of C-SDF operation, due to the scaling factor described last section. Again, C-SDF was exposed to a high variance in network performance. Figure 8(a) shows the delivery rate over time. In the beginning the delivery rate is stable, but low at around 70%. It drops off rapidly later with delivery rates of below 10%.

However, the short period of acceptable delivery rates is enough to stabilize at around 80% average energy utilization after 7 days (cf. Fig. 8(b)). Consequently, the sampling rate increases to around 30 samples per hour (cf. Fig. 8(c)). The performance is almost comparable to the first two runs, even though the network is performing worse.

This run, again, illustrates that C-SDF is resilient to harsh network conditions. We fully achieve our goal of utilizing the harvested energy to increase the quality of service. 80% average energy utilization is excellent, and, according to other runs, we lose approximately 10% in utilization due to flooring the calculated samples per sampling period to the next integer. Overall, the results are in line with the simulation results presented in [3].

In summary, C-SDF works even under adverse network conditions. It has neglectable overhead although we send control messages every 10 minutes. The sampling rate stabilizes quickly. Energy utilization is more than 80% while ensuring that no node is overloaded.

V. Conclusion

In conclusion, C-SDF is the first implementation of a practical flow control approach for field monitoring applications. It stabilizes even under adverse conditions, where more than 90% of all control messages are lost. The overhead is neglectable even though control messages are sent every 10 minutes.

[4]Please note that the utilization does also jump, but is only measured once per day. Hence, the line drawn for readability makes it appear linear.

The sampling rate improves significantly. Energy utilization is more than 80% while ensuring that no node is overloaded.

We plan to improve upon C-SDF by implementing a more aggressive approach to calculating the messages per sampling period. This should push utilization we above 90%. The most severe limitation of SDF is the dependency on the routing topology. Hence, we plan to combine SDF with topology control approaches and adapt RPL to allow for multi-path routing. This should lead to more balanced paths.

ACKNOWLEDGMENT

This work has been co-funded by the DFG as part of the CRC 1053 MAKI.

REFERENCES

[1] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava, "Power management in energy harvesting sensor networks," *ACM Transactions on Embedded Computing Systems*, vol. 6, no. 4, Sep. 2007.

[2] S. Roundy, D. Steingart, L. Frechette, P. Wright, and J. Rabaey, "Power Sources for Wireless Sensor Networks," in *Wireless Sensor Networks*, ser. Lecture Notes in Computer Science, H. Karl, A. Wolisz, and A. Willig, Eds. Springer Berlin / Heidelberg, 2004, vol. 2920, pp. 1–17.

[3] I. Schweizer, N. Fleischhacker, M. Mühlhäuser, and T. Strufe, "SDF - Solar-Aware Distributed Flow in Wireless Sensor Networks," in *2011 IEEE 36th Conference on Local Computer Networks*, 2011.

[4] K.-W. Fan, Z. Zheng, and P. Sinha, "Steady and fair rate allocation for rechargeable sensors in perpetual sensor networks," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, 2008, pp. 239–253.

[5] T. Winter, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks (RFC 6550)," 2012.

[6] C. M. Vigorito, D. Ganesan, and A. G. Barto, "Adaptive Control of Duty Cycling in Energy-Harvesting Wireless Sensor Networks," in *4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, 2007, pp. 21–30.

[7] A. Kansal and M. Srivastava, "An environmental energy harvesting framework for sensor networks," in *Proceedings of the 2003 International Symposium on Low Power Electronics and Design*, 2003, pp. 481–486.

[8] V. Raghunathan, A. Kansal, J. Hsu, J. Friedman, and M. Srivastava, "Design considerations for solar energy harvesting wireless embedded systems," in *Proceedings of the 4th international symposium on Information processing in sensor networks*, 2005, pp. 457–462.

[9] K. Lin, J. Yu, J. Hsu, S. Zahedi, D. Lee, J. Friedman, A. Kansal, V. Raghunathan, and M. Srivastava, "Heliomote: enabling long-lived sensor networks through solar energy harvesting," in *Proceedings of the 3rd international conference on Embedded networked sensor systems*, 2005.

[10] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," *ACM SIGPLAN Notices*, vol. 35, no. 11, pp. 93–104, Nov. 2000.

[11] X. Jiang, J. Polastre, and D. Culler, "Perpetual environmentally powered sensor networks," in *Fourth International Symposium on Information Processing in Sensor Networks*, 2005, pp. 463–468.

[12] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," in *Fourth International Symposium on Information Processing in Sensor Networks*, 2005, pp. 364–369.

[13] M. Masateru, T. Morito, H. Morikawa, and T. Aoyama, "Solar Biscuit: A Battery-less Wireless Sensor Network System," in *The 2nd International Workshop on Networked Sensing Systems*, 2005.

[14] P. Sikka, P. Corke, and L. Overs, "Wireless sensor devices for animal tracking and control," in *29th Annual IEEE International Conference on Local Computer Networks*, 2004, pp. 446–454.

[15] P. Corke, P. Valencia, P. Sikka, T. Wark, and L. Overs, "Long-duration solar-powered wireless sensor networks," in *Proceedings of the 4th workshop on Embedded networked sensors*. ACM Press, 2007, pp. 33–37.

[16] C. Alippi, R. Camplani, C. Galperti, and M. Roveri, "A Robust, Adaptive, Solar-Powered WSN Framework for Aquatic Environmental Monitoring," *IEEE Sensors Journal*, vol. 11, no. 1, pp. 45–55, Jan. 2011.

[17] L. Lin, N. B. Shroff, and R. Srikant, "Asymptotically Optimal Energy-Aware Routing for Multihop Wireless Networks With Renewable Energy Sources," *IEEE/ACM Transactions on Networking*, vol. 15, no. 5, pp. 1021–1034, Oct. 2007.

[18] T. Voigt, H. Ritter, and J. Schiller, "Utilizing solar power in wireless sensor networks," in *28th Annual IEEE International Conference on Local Computer Networks*. IEEE, 2003, pp. 416–422.

[19] T. Voigt and H. Ritter, "Solar-aware routing in wireless sensor networks," *Personal Wireless Communications*, vol. 2775, pp. 847–852, 2003.

[20] J. Schiller, A. Liers, H. Ritter, R. Winter, and T. Voigt, "ScatterWeb - Low Power Sensor Nodes and Energy Aware Routing," in *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, 2005.

[21] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 2–16, Feb. 2003.

[22] T. Voigt, A. Dunkels, J. Alonso, H. Ritter, and J. Schiller, "Solar-aware clustering in wireless sensor networks," in *Proceedings of the Ninth International Symposium on Computers And Communications*, 2004, pp. 238–243.

[23] R. Rajesh, V. Sharma, and P. Viswanath, "Capacity of fading gaussian channel with an energy harvesting sensor node," in *IEEE GLOBECOM*, 2011.

[24] W. Shu, X. Liu, Z. Gu, and S. Gopalakrishnan, "Optimal Sampling Rate Assignment with Dynamic Route Selection for Real-Time Wireless Sensor Networks," in *Real-Time Systems Symposium*, Nov. 2008, pp. 431–441.

[25] S. Bandyopadhyay and E. Coyle, "Spatio-temporal sampling rates and energy efficiency in wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 13, no. 6, pp. 1339–1352, Dec. 2005.

[26] A. Lachenmann, P. J. Marrón, D. Minder, and K. Rothermel, "Meeting lifetime goals with energy levels," in *Proceedings of the 5th international conference on Embedded networked sensor systems*, 2007, pp. 131–144.

[27] N. Tsiftes, J. Eriksson, and A. Dunkels, "Low-power wireless IPv6 routing with ContikiRPL," in *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2010.

[28] M. Durvy, N. Finne, A. Dunkels, J. Abeillé, P. Wetterwald, C. O'Flynn, B. Leverett, E. Gnoske, M. Vidales, G. Mulligan, and N. Tsiftes, "Making sensor networks IPv6 ready," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, 2008, pp. 421–422.

[29] S. Thomson and T. Narten, "IPv6 Stateless Address Autoconfiguration (RFC 4862)," 2007.

[30] A. Dunkels, "The ContikiMAC Radio Duty Cycling Protocol," SICS, Tech. Rep., Dec. 2011.

[31] P. E. Guerrero, A. P. Buchmann, A. Khelil, and K. Van Laerhoven, "TUDμNet, a Metropolitan-Scale Federation of Wireless Sensor Network Testbeds," in *9th European Conference on Wireless Sensor Networks*, Feb. 2012.

[32] A. Dunkels, F. Osterlind, N. Tsiftes, and Z. He, "Software-based on-line energy estimation for sensor nodes," in *Proceedings of the 4th workshop on Embedded networked sensors*, 2007, pp. 28–32.

[33] u-blox AG, "ANN-MS active GPS antenna," 2011.

[34] ——, "MAX-6 u-blox 6 GPS Modules," 2011.

[35] Figaro USA Inc., "TGS 2442 - for the detection of Carbon Monoxide," 2005.

[36] ——, "TGS 4161 - for the detection of Carbon Dioxide," 2005.

[37] T. D. Brock, "Calculating solar radiation for ecological studies," *Ecological Modelling*, vol. 14, no. 1-2, pp. 1–19, Nov. 1981.