Adaptive Power Management in Energy Harvesting Systems

Clemens Moser, Lothar Thiele
Swiss Federal Institute of Technology Zurich
{moser,thiele}@tik.ee.ethz.ch

Davide Brunelli, Luca Benini
University of Bologna
{dbrunelli,lbenini}@deis.unibo.it

ABSTRACT

Recently, there has been a substantial interest in the design of systems that receive their energy from regenerative sources such as solar cells. In contrast to approaches that attempt to minimize the power consumption we are concerned with adapting parameters of the application such that a maximal utility is obtained while respecting the limited and time-varying amount of available energy. Instead of solving the optimization problem on-line which may be prohibitively complex in terms of running time and energy consumption, we propose a parameterized specification and the computation of a corresponding optimal on-line controller. The efficiency of the new approach is demonstrated by experimental results and measurements on a sensor node.

1. INTRODUCTION

The idea of harvesting energy to power electronic devices is not new, but it has gained a lot of attention recently. Especially in the domain of wireless sensor networks, techniques to transform environmental energy into usable electrical power have been in the focus of the research community. As a matter of fact, the benefits of solar or vibrational energy harvesters for sensor nodes are striking: One of the main advantages of sensor networks connected via wireless technology is their ability to operate in the absence of a pre-established infrastructure. This independence, however, severely depends on the autonomy of the single nodes. It has been shown in several feasibility studies [5, 13], that sensor nodes scavenging ambient energy may operate perpetually, without the need of manually replacing or recharging their batteries.

In contrast to systems with a finite energy reservoir, systems powered by a regenerative energy source have to adapt to the stochastic nature of the energy available from the environment. The goal of such an adaptation is to maximize the utility of the application. For example, it has been shown that distributed applications such as clustering [14] and routing [9] in sensor networks can benefit if the underlying algorithms take advantage of spatial variations of the scavenged energy. By allocating different loads to the nodes of the network, the overall performance can be optimized.

Concerning the temporal variations of the environmental source at a single device, in [11] an optimal on-line algorithm to schedule a set of tasks within their deadlines is constructed. Taking into account available time as well as processable energy, an optimal task ordering is determined based on the prediction of the available energy in the future. In contrast to the work presented in this paper, the average power consumption is not affected by the approach in [11] and parameters of the application are not changed.

Early work addressing the long-term performance of a device has been presented in [7]. Here, the average power consumption of a sensor node is adjusted using different duty cycles, i.e. by switching the sensor node on and off. But the method described is not adaptive since a fixed duty cycle is applied after having learnt the characteristics of the energy source during a pre-operational phase.

In [12], a computing device may control its performance level by running different versions of a software application, each version having a different reward, time as well as energy demand. The energy source assumed is solar and comprises two simple states: day and night. The system considered assumes a minimum of harvested energy during daytime. In contrast to the work mentioned before, adaptive power management in [12] reduces to utilize surplus energy *after* having harvested it. To this end, several heuristics are presented.

The first work that can be classified as *adaptive* power management for energy harvesting systems has been published recently in [6, 4]. Here, the problem of tuning the duty cycle of a solar-driven sensor node has been formulated as a linear program (LP). This LP has to be solved periodically. Within each period, adaptations of the duty cycle become necessary if the observed energy values vary from the predicted ones. The system considered is (a) capable of using the solar energy directly and (b) storing energy in a storage device with low efficiency. The objective in [6, 4] is to maximize the utilization of solar energy, i.e. to maximize the *average* duty cycle. For this particular problem, the authors propose an algorithm which attempts to decrease the duty cycle in times when the scavenged energy is low (e.g. at night) and increase the duty cycle when scavenged energy is high (e.g. during the day).

In contrast to the latter work, the class of linear programs presented in this paper is capable of modeling a much larger variety of application scenarios, constraints and optimization objectives. For example, tradeoffs between the use of local memory and communication can be exploited. Finally, we are able to handle arbitrary, non-trivial objectives such as maximizing the *minimum* duty cycle. In addition, we propose to compute the optimal solution of a linear program by a method which exhibits low computational complexity and is thus well suited for resource constrained systems.

2. CONTRIBUTIONS

This paper contains the following new results:

- We present a specification model that is able to capture the performance, the parameters and the energy model of environmentally powered systems.
- As a main contribution, we suggest the use of simple, but optimal model predictive controllers to adapt parameters of an application. We are adapting results

978-3-9810801-2-4/DATE07 © 2007 EDAA

of the well-established framework of multiparametric programming to the emerging area of energy harvesting systems.

- We propose a practical technique for the efficient implementation of the controller and demonstrate the practical relevance of our approach by measurements of the controller running on a real sensor node.

3. SYSTEM CONCEPT

The system model is depicted in Fig. 1. The whole hardware/software system is powered by an energy harvesting device that delivers in a unit time interval starting at t the energy $E_S(t)$ which loads the energy storage. In the same time interval, the system uses energy $E_D(t)$. At time t, there is the stored energy $E_C(t)$ available.

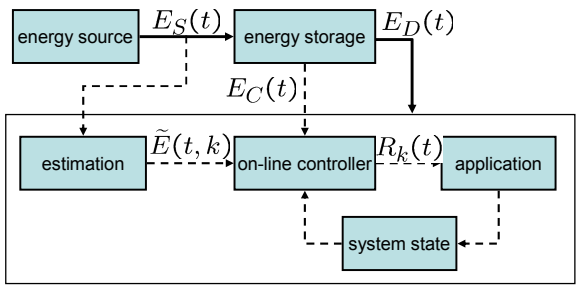

Figure 1: Illustration of the system concept.

Besides the application, there are two additional software tasks running on the target architecture. The estimator predicts future energy production of the harvesting device based on measurements of the past. The controller adapts properties of the application, e.g. task activation rates, based on the estimation of future available energy, the currently stored energy and additional information about the system state, e.g. the amount of available data memory. Parameters of the application are modified by the on-line controller. During execution, the system state (e.g. the amount of information stored in local memory and the stored energy) is changed.

It is the duty of the controller to adjust properties of the application such that overall objectives are optimized (for example maximizing the sampling rate of a sensor) while respecting system constraints (for example using not more than the available memory). The complexity of the controller design is caused by the fact that (a) the harvested energy changes in time, (b) the available future energy can only be estimated and (c) the overall objective is usually a long term goal, e.g. maximizing the minimal sampling rate of a sensor in the future. For example, a sensor node powered by an outdoor solar cell may save energy during the day in order to have enough energy available at night for transmitting sensor data. But it may also store the sensor data at night and send them during the day. As a result, complex controller strategies may be required.

Note that in contrast to other work in that area, energy $E_S(t)$ is always stored before used. For a direct usage of $E_S(t)$ with high efficieny bypassing the energy storage, our system concept (in particular the LP in 4.4) can be extended easily. In [6, 4], an example for such an extension is provided.

4. BASIC MODELS AND METHODS

4.1 Power Model

The modeling is based on the notion of discrete time $t \in \mathbb{Z}_{\geq 0}$ where the difference in physical time between two discrete time instances is denoted as T. Energy related sensing and control may happen only at times t. In a practical setting, one may have a basic time interval T of a few minutes or even an hour.

The energy harvesting device is modeled as a power source which delivers energy $E_S(t)$ in the time interval $[t, t+1)$ of length T. Therefore, in time interval $[t_1, t_2)$ with $t_1, t_2 \in \mathbb{Z}_{\geq 0}$ it delivers energy

$$E_S(t_1, t_2) = \sum_{t_1 \leq u < t_2} E_S(u)$$

The incoming power can be stored in an energy storage device, e.g. a rechargeable battery, a super-capacitor or a combination of both. The energy level at time t is denoted as $E_C(t)$.

The available energy can be used to execute tasks on various system components. Tasks may be as diverse as sensing, signal processing, A/D or D/A conversion, computing, or communicating. A task $\tau_i \in \mathbf{I}$ from the set of tasks $\mathbf{I}$ needs energy e_i for completing a single instance. The corresponding energy is drawn from the energy storage. We suppose that a task is activated with a time-variant rate $s_i(t)$, i.e. during the basic time interval T starting at t, the task is executed $s_i(t)$ times. Therefore, a task needs energy

$$E_i(t_1, t_2) = \sum_{t_1 \leq u < t_2} e_i \cdot s_i(u)$$

in time interval $[t_1, t_2)$ for successful execution. The detailed application and task model will be described in Section 4.3.

4.2 Estimation

According to the system model described in Section 3, an important component of the adaptive power management is the predictor. It receives tuples $(t, E_S(t))$ for all times $t \geq 1$ and delivers N predictions on the energy production of the energy source. We assume that the prediction intervals are of equal size denoted as the number L (in units of the basic time interval T). Therefore, at time t, the predictor produces estimations $\widetilde{E}_S(t + k \cdot L, t + (k+1) \cdot L)$ for all $0 \leq k < N$. We write $\widetilde{E}(t, k) = \widetilde{E}_S(t + k \cdot L, t + (k+1) \cdot L)$ as a shorthand notation, i.e. the estimation of the incoming energy in the $(k+1)$st prediction interval after t.

The prediction algorithm should depend on the type of the energy source and the system environment. Standard techniques known from automatic control and signal processing can be applied here. In this paper, we provide a very simple and obvious technique only which is useful for solar cells that operate in an outdoor environment. It has been used for the experimental results presented in Section 5.

The estimation takes into account that the solar power can be considered to be periodic in $D = N \cdot L$, i.e. the length of a day (again in units of the basic time interval T), see also Fig. 2. Therefore, the prediction algorithm collects information about the received energy in the current unit time interval and combines it with previously received information whose age is a multiple of days. We are using a simple exponential decay of old data with factor $0 < \alpha < 1$. In

order to obtain predictions for the desired time intervals of length L we just add the predicted energy values for the corresponding intervals of length 1. Finally, this long term prediction could be enhanced using a short term predictor that directly uses the information of the received power $P_S(t)$. In the experiments, we did not make use of this possibility for simplicity reasons and used only the described algorithm with $\alpha = 0.5$.

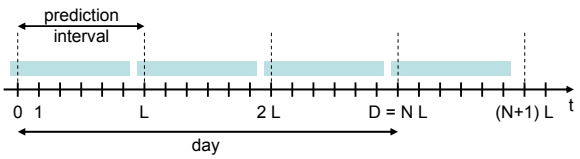

Figure 2: Illustration of prediction intervals.

4.3 Rate-Based Application Model

As described in Section 3, parameters of the application are changed during run-time in order to optimally use the available energy in the future. In this paper, we restrict ourselves to a rate-based application model.

The application consists of tasks τ_i, $i \in \mathbf{I}$. A task is instantiated $s_i(t) \geq 0$ times in the interval of length T starting at time t and the execution of each instance needs energy e_i. The activation of tasks can be modeled by a rate graph whose nodes and edges represent tasks and activation relations, respectively. In particular, an edge from i to j denotes that task τ_i activates task τ_j. We may also say, that an edge (i, j) is activated r_{ij} times; such an activation is caused by the activation of task τ_i and leads to an activation of task τ_j.

A scaling factor σ_{ij} is associated to an edge (i, j) that represents how often task τ_j is activated for each activation of it. The default scaling is $\sigma_{ij} = 1$. If there are several edges leaving a node, then the sum of their rates equals the activation rate s of the source node, e.g. $s_1 = r_{12} + r_{13}$. This way, we can model a decision in the application, i.e. the execution of a task may either lead to the activation of one or another subsequent task. If there are several edges leaving from the same (graphical) location at some node then their activation rates are equal. This models the case that two subsequent tasks are activated with the same rate, i.e. $r_{12} = r_{13}$ if $(1, 2)$ and $(1, 3)$ have their tails at the same location.

The rate relations that are covered by a rate graph as defined above can be formulated as a set of linear (in)-equalities. Free variables $R_k(t)$ in this system are determined by the controller shown in Fig. 1. In general, we can formulate the rate equations as

$$\mathbf{P} \cdot \mathbf{R}(t) + \mathbf{Q} = \mathbf{S}(t) \quad (1)$$

$$\mathbf{F} \cdot \mathbf{R}(t) + \mathbf{G} \geq \mathbf{0} \quad (2)$$

where $\mathbf{S}(t)$ is a vector containing all activation rates $s_i(t)$, $\mathbf{R}$ contains all controlled parameters $R_k(t)$ and $\mathbf{P}$, $\mathbf{Q}$, $\mathbf{F}$ and $\mathbf{G}$ are vectors and matrices of appropriate dimensions.

The next Fig. 3 exemplifies the rate graph and gives an example for the associated rate equations.

4.4 Linear Program Specification

The first step in constructing the on-line controller is the formulation of the optimization problem in form of a parameterized linear program (LP). The corresponding solution methods will be described in Section 4.5.

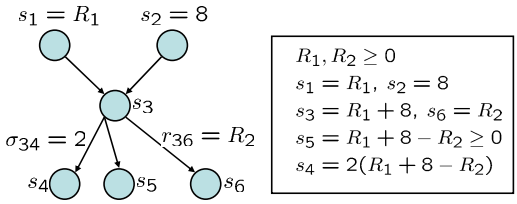

Figure 3: Illustration of rate graph and rate equations.

One of the essential states of the system is the stored energy. Using the power model described in Section 4.1 we obtain the following state equations:

$$E_C(t + k \cdot L) = E_C(t) - k \cdot \sigma + \\ + \sum_{j=0}^{k-1} (\gamma \cdot \widetilde{E}(t, j) - \epsilon \cdot L \cdot \mathbf{E}^T \cdot \mathbf{S}(t + j \cdot L)) \quad (3)$$

They determine the expected contents of the energy storage at times $t + kL$ for $1 \leq k \leq N$. $\mathbf{S}(t)$ is a vector containing all activation rates $s_i(t)$ and $\mathbf{E}^T$ is a (row)-vector that contains all energy requirements e_i for all $i \in \mathbf{I}$. Therefore, $\mathbf{E}^T \cdot \mathbf{S}(t) = \sum_{i \in \mathbf{I}} e_i \cdot s_i(t)$.

The factor σ accounts for energy leakage and γ, ϵ take into account a reduced energy efficiency. The equation also supposes that we have available a new estimation of the received energy only every L time instances and that the rate of the different tasks s_i is constant during each of these time intervals.

In a similar way, we can also model other system states, for example memory. A task could produce a certain amount of data that is stored and another removes it, e.g. by means of communication to another unit. In this case we would have for $1 \leq k \leq N$ the state equations:

$$M(t + k \cdot L) = M(t) + L \sum_{j=0}^{k-1} \mathbf{M}^T \cdot \mathbf{S}(t + j \cdot L) \quad (4)$$

$M(t)$ denotes the amount of stored data at time t and m_i is the amount of data produced or consumed by a task τ_i with rate s_i in a time interval of length T. $\mathbf{M}^T$ is a (row)-vector that contains all data amounts m_i for all $i \in \mathbf{I}$.

Of course, (3) and (4) provide only examples of possible system states and their associated changes. Moreover, there may be additional constraints on the set of feasible states.

One can now easily combine (1)-(4) and obtain a system of linear equalities and inequalities that contain as free variables $M(t + k \cdot L)$ (the state of the memory), $E_C(t + k \cdot L)$ (the state of the energy) for $1 \leq k \leq N$ and $\mathbf{R}(t + k \cdot L)$ (the rate control) for $0 \leq k < N$.

So far, no optimization goal has been formulated and therefore, any feasible rate control $\mathbf{R}(t + k \cdot L)$ could be a solution. Any linear objective function that makes use of the free variables given above is possible in this case. One may also define additional variables in order to model specific objectives. One possible (very simple) example would be the objective

$$\text{maximize } \lambda \text{ subject to } s_1(t + k \cdot L) \geq \lambda \quad \forall 0 \leq k < N \quad (5)$$

which would attempt to maximize the minimal rate with

which the task τ_1 is operated in the finite horizon $0 \leq k < N$.

4.5 Controller Generation

Next, we will show how to design an on-line controller based on a state feedback law which avoids solving a linear program at each time step. Thereby, we are following the ideas in [1], where the regulation of discrete-time constrained linear systems is studied in the context of model predictive control (MPC).

As a first step, we define a state vector $\mathbf{X}$ consisting of the actual system state, the level of the energy storage as well as the estimation of the incoming energy over the finite prediction horizon (cp. Fig.1). Resuming the system dynamics formulated in (1)-(4), the state vector $\mathbf{X}$ can be written as

$$\mathbf{X}(t) = \Big(E_C(t)\,,\, M(t)\,,\, \widetilde{E}(t,0)\,,\, \ldots\,,\, \widetilde{E}(t,N-1) \Big)^T \qquad (6)$$

Furthermore, let us denote the vector of planned control inputs to the system, i.e., the vector of future rates $\mathbf{R}$ as

$$\mathbf{U}(t) = \Big(\mathbf{R}^T(t)\,,\, \mathbf{R}^T(t+L)\,,\, \ldots\,, \mathbf{R}^T\,(t+(N-1)\cdot L) \Big)^T . \qquad (7)$$

The state space of $\mathbf{X}$ (in our case $\mathbb{R}^{N+2}$ bounded by possible constraints on $E_C(t), M(t)$ and $\widetilde{E}(t,i)$) can now be subdivided into a number N_{CR} of polyhedrons. For each of these polyhedrons i (also called critical regions) the optimal solution $U_{opt,i}$ of the control problem can be made available explicitly as

$$\mathbf{U}_{opt,i}(t) = \mathbf{B}_i \mathbf{X}(t) + \mathbf{C}_i \quad \text{if } \mathbf{H}_i \mathbf{X}(t) \leq \mathbf{K}_i, i = 1, \ldots, N_{CR} \qquad (8)$$

where $\mathbf{B}_i \in \mathbb{R}^{(N+2)\times N}, \mathbf{C}_i \in \mathbb{R}^N$ and $\mathbf{H}_i\mathbf{X}(t) \leq \mathbf{K}_i, i = 1 \ldots N_{CR}$ is a polyhedral partition of the state space of $\mathbf{X}(t)$. The computation of the vectors and matrices of control law (8) is done off-line using, e.g., the algorithm presented in [3] or other efficient solvers cited in the latter work.

In the on-line case, the controller has to identify to which region i the current state vector $\mathbf{X}(t)$ belongs. After this simple membership test, the optimal control moves $\mathbf{U}_{opt}(t)$ for the next N prediction intervals are computed by evaluating a linear function of $\mathbf{X}$. However, only the first entry of $\mathbf{U}_{opt}(t)$ is actually used to drive the application and we obtain the optimal rates

$$\mathbf{R}_{opt}(t) = (\,\mathbf{I}_N,\, \mathbf{0},\, \ldots, \mathbf{0}\,) \cdot \mathbf{U}_{opt}(t) \qquad (9)$$

where $\mathbf{I}_N$ denotes the $N \times N$ identity matrix.

In summary, the main idea of the proposed technique is to treat the linear program (LP) formulated in the last section as a multiparametric linear programm (mp-LP) with parameters $\mathbf{X}$ and optimization variables $\mathbf{U}$. The controller computes the optimal control inputs $\mathbf{U}_{opt}(t)$ as an explicit function of the current state $\mathbf{X}(t)$. The resulting control profile can be seen as a piecewise affine function of the state vector $\mathbf{X}(t)$.

It should be mentioned that evaluating the solution of a mp-LP results in the same control vectors $\mathbf{R}_{opt}(t)$ as would be obtained by iteratively solving the LP. However, the computational demand is greatly reduced compared to solving a LP on-line. After having solved the mp-LP in advance, a set of N_{CR} polyhedra with associated control laws has to be stored and evaluated at each time step t. If the number of critical regions N_{CR} gets large, the computational effort still may be large as many tests of the form $\mathbf{H}_i\mathbf{X}(t) \leq \mathbf{K}_i$ must be performed in order to determine the correct law $\mathbf{U}_i(t) = \mathbf{B}_i\mathbf{X}(t) + \mathbf{C}_i$. Efficient methods to solve this potential bottleneck are described in Section 6.

5. EXPERIMENTAL RESULTS

In this section, both feasibility and practival relevance of the proposed approach is demonstrated by means of simulation and measurements. For this purpose, we implemented the computation of on-line controllers for two exemplary case studies using the MATLAB toolbox in [8]. We simulated the behavior of the controlled system and measured the resulting controller overhead on a sensor node.

Measurements of solar light intensity $[\frac{W}{m^2}]$ during 28 consecutive days recorded at [10] serve as energy input $E_S(t)$. The time interval between two samples is 10 minutes, so we set the basic time interval $T = 10\,\text{min}$. Of course, one would have to scale the measured power profile in [10] with the size, number and efficiency of the actually used solar panels. However, we expect the influence of this scaling on our qualitative results to be negligible.

5.1 Adaptation of Sensing Rate

In this example, a sensor node is expected to measure some physical quantity like e.g. ambient temperature or mechanical vibrations and has to transmit the sampled data to a base station. We can model these requirements as a single sensing task τ_1 with rate $R_1(t)$, i.e. a task which is instantiated R_1-times in the interval $[t, t+T)$. For the sake of simplicity, the sensing task τ_1 drains at every instantiation 1 energy unit from the battery. Assume further that we want the maximum interval between two consecutive reports to be as small as possible, as in (5). For this setup, we can formulate the linear program LP in (10). Note that the last inequality in (10) is used to stabilize the receding horizon controller.

maximize λ subject to:	(10)
$s_1(t+k\cdot L) \geq \lambda$	$\forall 0 \leq k < N$
$E_C(t+k\cdot L) = E_C(t) +$	
$+\sum_{j=0}^{k-1} \Big(\widetilde{E}(t,j) - L\cdot s_1(t+j\cdot L) \Big) \geq 0$	$\forall 1 \leq k \leq N$
$E_C(t+N\cdot L) \geq E_C(t) - 100$	

In general, the number of partitions N_{CR} of a multiparametric solution grows with the size of the state vector $\mathbf{X}$. Hence, it is of practical concern to keep the number of prediciton intervals $\widetilde{E}(t,i)$ and therewith the dimension of $\mathbf{X}$ as small as possible. We chose $L = 24$ and $N = 6$ and obtain the states $\mathbf{X}(t) = \Big(E_C(t), \widetilde{E}(t,0), \ldots, \widetilde{E}(t,5) \Big)^T$. The resulting on-line controller consists of $N_{CR} = 7$ partitions.

Figure 4 shows the simulated sensing rate R_1 over a time period of 10 days for the generated optimizing controller. We started the simulation with an energy level $E_C(0) = 1300$ and found a nearly constant rate R_1 during the whole simulation period. On the other hand, the stored energy $E_C(t)$ is highly varying, since the controller successfully compensates the unstable power supply $E_S(t)$. As a consequence, the stored energy $E_C(t)$ is increasing during the day and decreasing at night. Even the 8th displayed day

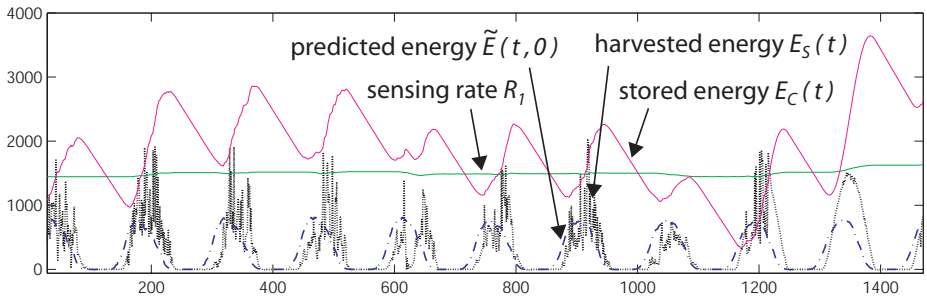

Figure 4: Adaptation of the sensing rate R_1.

with significant less sunshine is not jeopardizing the sensing rate R_1. Figure 4 clearly demonstrates that the simple on-line controller manages to meet the optimization goal.

5.2 Local Memory Optimization

Another simple scenario of practical importance may consist of two tasks running on a sensor node: A first task τ_1 is sampling some physical quantity, i.e., performing an A/D-conversion and storing the data in some local memory. A second task τ_2 is transmitting stored samples with a rate R_2 and thereby frees memory from the storage device. Clearly, the scaling factor $\sigma_{12} = R_1/R_2$ represents the ratio with which the amount of stored data M is increasing ($\sigma_{12} < 1$) or decreasing ($\sigma_{12} > 1$).

For this application, two reasonable optimization objectives would be (a) to minimize the unobserved intervals between any two consecutive samples and (b) to minimize the amount of stored data M. The purpose of the second objective is twofold: On one hand, sensor nodes are usually small, inexpensive low power devices with constrained hardware resources such as memory. On the other hand, the objective may to some extent enforce the freshness of data arriving at the base station.

maximize $(\lambda - \mu)$ subject to:	(11)
$s_1(t + k \cdot L) \geq \lambda$	$\forall 0 \leq k < N$
$M(t + N) \leq \mu$	
$E_C(t + k \cdot L) = E_C(t) + \sum_{j=0}^{k-1} \widetilde{E}(t, j) -$	
$- \sum_{j=0}^{k-1} (L \cdot [0.1 \;\; 0.9] \cdot \mathbf{S}(t + j \cdot L)) \geq 0$	$\forall 1 \leq k \leq N$
$M(t + k \cdot L) = M(t) +$	
$+L \sum_{j=0}^{k-1} [1 \;\; -1] \cdot \mathbf{S}(t + j \cdot L) \geq 0$	$\forall 1 \leq k \leq N$
$E_C(t + N \cdot L) \geq E_C(t) - 150$	

In general, radio communication is the main energy consumer on a sensor node. Hence we set the energies $e_1 = 0.1$ and $e_2 = 0.9$. The corresponding linear program LP is given by (11). To account for the additional system state $M(t)$ we reduced the number of prediction intervals and set $L = 36$ and $N = 4$. The computed control law divides the state space of $\mathbf{X}(t) = \left(E_C(t), M(t), \widetilde{E}(t,0), \ldots, \widetilde{E}(t,3)\right)^T$ in $N_{CR} = 39$ critical regions.

Fig. 5 displays the simulated curves of the state and control variables during 6 days. The figure at the top represents a scenario with no use of local memory, i.e. the control problem in (10) whereas the bottom figure shows a comparable scenario using local memory according to (11). Until $t = 1700$, both tasks are adjusted to the same rate $R_1 = R_2$ and the memory $M(t)$ is empty. However, after two days with little harvested energy, the energy level $E_C(t)$ on the sensor node is falling and the controller starts to suspend the energy-costly communication task τ_2 by reducing rate R_2. Consequently, the number of buffered samples M is increasing starting before $t = 1800$. In the following, the controller achieves to autonomously regulate the tradeoff between $E_C(t)$ and $M(t)$. While a lack of $E_C(t)$ would cause the non-initiation of task τ_1, an increasing $M(t)$ directly affects the second optimization objective. After $t = 1950$, the controller reduces the amount of occupied local memory by increasing the rate R_2.

6. EFFICIENT IMPLEMENTATION OF THE CONTROL LAW

In general, the identification of the active region i dominates the linear function evaluation of a control law (8) in terms of time and energy consumption. In the worst case, for all N_{CR} regions a matrix multiplication has to be performed in order to identify the active region i at time t. However, the identification of the active region i can be simplified due to the following facts: First, the matrices $\mathbf{H_i}$ are sparse which reduces the number of necessary multiplications and additions significantly. Second, usually only a subset of all N_{CR} is activated in practice and some of the regions in this subset are activated more frequently than others.

The second observation can be exploited by starting the search always with the region with the highest statistical occurrence, continue with the second highest, ..., and so on. To this end, an algorithm should maintain a list of regions i ordered by their frequencies which is updated every time step t.

For the memory optimization problem in Section 5.2, we found that on average $\approx 40\%$ of the entries of a matrix $\mathbf{H}$ are different from zero. Moreover, only 7 of 39 regions were used during the whole simulation period. Using probabilistic region testing as described above, the critical term of the form $\mathbf{H} \cdot \mathbf{X} \leq \mathbf{K}$ is only evaluated ≈ 1.45 times at time t, taking the average over all 4032 time steps.

Figure 6 displays the voltage measured at a 10Ω shunt resistor in series with a BTnode [2]. At first, the current energy level $E_C(t)$ of the battery and the scavenged energy $E_S(t)$ are determined via two A/D-conversions, which mark the two major peaks in plot 6. Focussing on the overhead of the proposed controller, we omit predicting the future energies $\widetilde{E}(t,i)$. Instead, an average situation is displayed where the region with the second highest frequency is the active one. Subsequently, the optimal control output

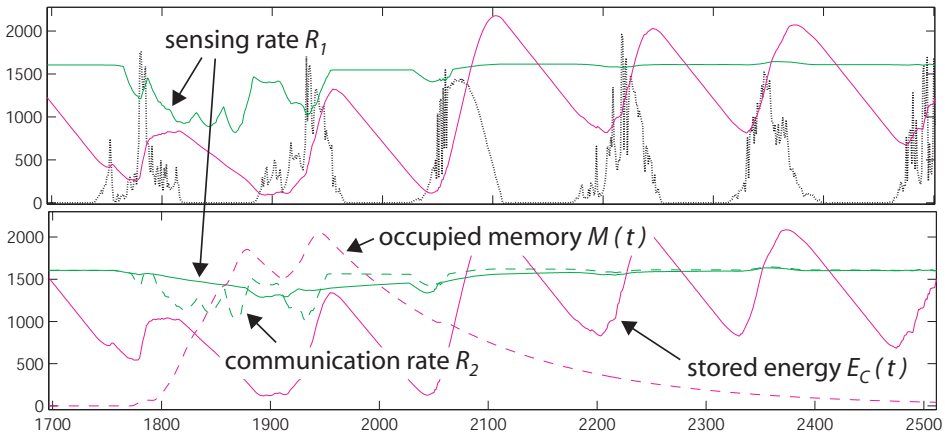

Figure 5: Top: scenario without local memory. Bottom: scenario with optimized local memory.

for this region is calculated. It becomes evident, that the computations leading to the actual control actions take as long as the two A/D-conversions ($\approx 2ms$), having at the same time a significant lower power consumption. Hence, these measurements demonstrate how the simple, but efficient implementation of the proposed controller is applicable to sensor nodes, involving only marginal computation overhead.

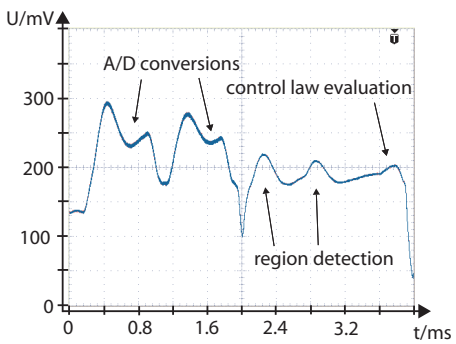

Figure 6: Computation of the optimal rates R_1, R_2 for the memory optimization problem on a BTnode.

7. CONCLUDING REMARKS

We presented an approach to optimize the performance of energy harvesting systems subject to temporal variations of the energy source. For a rate-based application model, we propose to solve a linear program in a multiparametric fashion and shift most of the associated overhead to an offline computation. The solution of the mp-LP is optimal with respect to the energy prediction but involves significantly lower overhead than solving the respective LP on-line. Measurements of the controller running on a real sensor node show how the method can be implemented efficiently.

Acknowledgements

The work presented in this paper was partially supported by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322. In addition, this research has been founded by the European Network of Excellence ARTIST2.

8. REFERENCES

[1] A. Bemporad, F. Borrelli, and M. Morari. Model Predictive Control Based on Linear Programming - The Explicit Solution. *IEEE Transactions on Automatic Control*, 47(12):1974–1985, Dec. 2002.

[2] J. Beutel, M. Dyer, M. Hinz, L. Meier, and M. Ringwald. Next-generation prototyping of sensor networks. In *Proc. 2nd ACM Conf. Embedded Networked Sensor Systems (SenSys 2004)*, pages 291–292. ACM Press, New York, Nov. 2004.

[3] F. Borrelli, A. Bemporad, and M. Morari. A Geometric Algorithm for Multi-Parametric Linear Programming. *Journal of Optimization Theory and Applications*, 118(3):515–540, Sept. 2003.

[4] J. Hsu, S. Zahedi, A. Kansal, M. Srivastava, and V. Raghunathan. Adaptive duty cycling for energy harvesting systems. In *ISLPED '06: Proceedings of the 2006 international symposium on Low power electronics and design*, pages 180–185, New York, NY, USA, 2006. ACM Press.

[5] X. Jiang, J. Polastre, and D. E. Culler. Perpetual environmentally powered sensor networks. In *Proceedings of the Fourth International Symposium on Information Processing in Sensor Networks, IPSN 2005*, pages 463–468, UCLA, Los Angeles, California, USA, April 25-27 2005.

[6] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava. Power management in energy harvesting sensor networks. In *ACM Transactions on Embedded Computing Systems (in revision)*, May 2006, also available from: NESL Technical Report Number: TR-UCLA-NESL-200605-01.

[7] A. Kansal, D. Potter, and M. B. Srivastava. Performance aware tasking for environmentally powered sensor networks. In *Proc. of the Int. Conf. on Measurements and Modeling of Computer Systems, SIGMETRICS 2004*, pages 223–234, New York, NY, USA, June 10-14 2004. ACM Press.

[8] M. Kvasnica, P. Grieder, and M. Baotić. Multi-Parametric Toolbox (MPT), 2004.

[9] L. Lin, N. B. Shroff, and R.Srikant. Asymptotically optimal power-aware routing for multihop wireless networks with renewable energy sources. In *Proceedings of IEEE INFOCOM 2005*, pages 1262 – 1272, Miami, USA, March 13-17 2005.

[10] Sunergy. Recordings of solar light intensity from 08/08/2005 to 04/09/2005, datasheet. http://sunergy.dmclf.org/cms/, June, 2006.

[11] C. Moser, D. Brunelli, L. Thiele, and L. Benini. Real-time scheduling with regenerative energy. In *Proc. of the 18th Euromicro Conference on Real-Time Systems (ECRTS 06)*, pages 261–270, Dresden, Germany, July 2006.

[12] C. Rusu, R. G. Melhem, and D. Mosse. Multi-version scheduling in rechargeable energy-aware real-time systems. In *15th Euromicro Conference on Real-Time Systems, ECRTS 2003*, pages 95–104, Porto, Portugal, July 2-4 2003.

[13] F. Simjee and P. H. Chou. Everlast: Long-life, supercapacitor-operated wireless sensor node. In *Proc. Int. Symposium on Low Power Electronics and Design (ISLPED)*, Oct. 2006.

[14] T. Voigt, H. Ritter, J. Schiller, A. Dunkels, and J. Alonso. Solar-aware clustering in wireless sensor networks. In *Proc. of the 9th IEEE Symp. on Computers and Comm.*, June 2004.