Energy Management for Battery-Powered Embedded Systems

DALER RAKHMATOV and SARMA VRUDHULA
University of Arizona, Tucson

Portable embedded computing systems require energy autonomy. This is achieved by batteries serving as a dedicated energy source. The requirement of portability places severe restrictions on size and weight, which in turn limits the amount of energy that is continuously available to maintain system operability. For these reasons, efficient energy utilization has become one of the key challenges to the designer of battery-powered embedded computing systems.

In this paper, we first present a novel analytical battery model, which can be used for the battery lifetime estimation. The high quality of the proposed model is demonstrated with measurements and simulations. Using this battery model, we introduce a new "battery-aware" cost function, which will be used for optimizing the lifetime of the battery. This cost function generalizes the traditional minimization metric, namely the energy consumption of the system. We formulate the problem of battery-aware task scheduling on a single processor with multiple voltages. Then, we prove several important mathematical properties of the cost function. Based on these properties, we propose several algorithms for task ordering and voltage assignment, including optimal idle period insertion to exercise charge recovery.

This paper presents the first effort toward a formal treatment of battery-aware task scheduling and voltage scaling, based on an accurate analytical model of the battery behavior.

Categories and Subject Descriptors: C.4.5 [**Performance of Systems**]: Performance Attributes; J.6.2 [**Computer-Aided Engineering**]: Computer-Aided Design (CAD)

General Terms: Algorithms, Performance

Additional Key Words and Phrases: Battery, modeling, low-power design, scheduling, voltage scaling

1. INTRODUCTION

Portable devices, such as mobile phones, personal digital assistants, communicators, palmtops, and so on, with powerful embedded computing capabilities, have become an indispensable part of our daily lives. Present-day handheld

This work was carried out at the National Science Foundation's State/Industry/University Cooperative Research Centers' (NSF-S/IUCRC) Center for Low Power Electronics (CLPE). CLPE is supported by the NSF (grant EEC-9523338), the State of Arizona, and a consortium of companies from the microelectronics industry (http://clpe.ece.arizona.edu).
Authors' address: Center for Low Power Electronics, Department of Electrical and Computer Engineering, University of Arizona, 1234 E. Speedway Blvd., Tucson, AZ 85721; email: daler@ece.arizona.edu, sarma@ece.arizona.edu.

computers are able to run computationally intensive applications (e.g., streaming multimedia) which, a few years ago, was possible only on a high-performance desktop machine. In addition to performance expectations, the requirement of portability imposes stringent constraints on size and weight of a portable system. Since mobility requires energy autonomy, portable devices commonly feature an attached finite-capacity energy source—a battery, which must be relatively small and light. Consequently, the system energy budget is severely limited, and efficient energy utilization becomes one of the key challenges faced by the system designer.

The battery lifetime is perhaps one of the most important characteristics of a portable computer. For many users, doubling the battery lifetime may be far more important than doubling the clock frequency. Unfortunately, improvements in battery capacity have not kept pace with the improvements in microelectronics technology. Consequently, methods to increase the battery lifetime must examine how the *energy consumer* (e.g., the processor and other units) can be made more efficient from the perspective of the *energy supplier*. To examine various alternatives to achieve this requires an understanding of the basic characteristics and principles of the battery operation. In other words, the system designer needs an adequate model relating the battery behavior to the discharge conditions. Once such a model is available, one can evaluate energy efficiency of various system design options and/or scenarios of application execution.

In this paper we address the issues of energy management for a generic battery-powered embedded system, composed of a processor, a voltage regulator, and a battery. We assume the availability of several supply voltages and clock frequencies at which the processor can operate.[1] A user runs a set of interdependent tasks, subject to the constraint on completion latency. During execution of user tasks, the processor draws a certain amount of current from the battery. This discharge current, varying over time, is referred to as a *load profile*.

The first problem is to relate a given load profile to the battery lifetime. This is difficult to accomplish as the battery behavior depends on how the battery is discharged (shortly, we will present a motivating example that demonstrates this dependency).

The second problem is to schedule tasks and select task voltages (and clock frequencies), so that the resulting load profile yields maximum improvement in the battery lifetime. An accurate relationship between the load profile and the battery lifetime is essential for this purpose.

1.1 Motivating Example

To motivate investigation of battery-related issues arising during energy management, we conducted several experiments on a 2.2 watt-hour lithium-ion battery (with the nominal discharge rate of 640 mA) used in a pocket computer

[1]Dynamic voltage and frequency scaling have proven to be one of the most effective ways to reduce energy consumption. Examples of commercial products featuring voltage/clock scaling capabilities include Intel microprocessors based on the *XScale*^{TM} technology [Intel 2002].

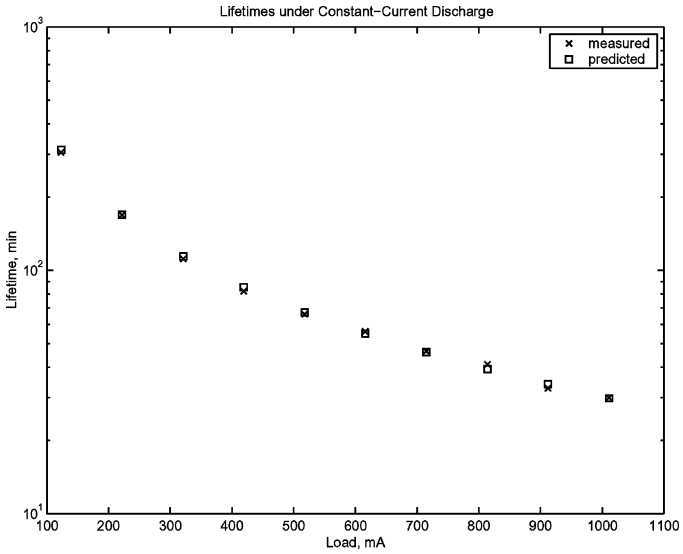


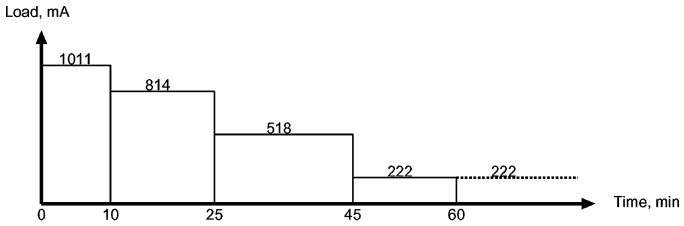
Fig. 1. Battery lifetimes for various constant-current loads.

[Hamburgen et al. 2001]. In addition to the battery, the experimental setup included the programmable electronic load Agilent 6060B, and also the host computer recording measurement data. The open-circuit voltage of the battery was 4.2 V, and the cutoff voltage was set to 3.0 V. The electronic load operated in the constant-current mode, and variable-current profiles were generated as a piecewise constant-current profile (a staircase). The battery voltage was sampled every second, and once the voltage dropped below the cutoff level the load was automatically disconnected from the battery. After each test the battery was recharged in the constant-current mode at 800 mA, until the battery voltage recovered to its open-circuit value. Next, we present the measurement results as well as the lifetime predictions obtained from our battery model in Section 3.

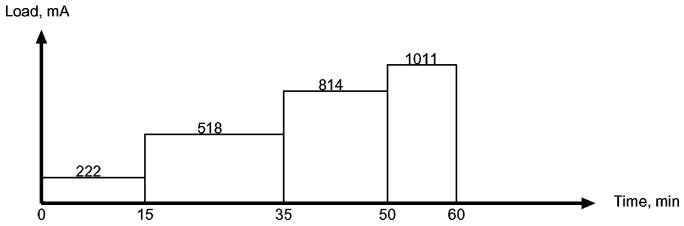
For the first ten experiments, the battery discharge current was constant in each test. The current values ranged from 1011 mA to 123 mA, and the measured battery lifetimes ranged from 30 min to over 300 min. Figure 1 shows the fit of our model. The maximum prediction error is 4%, with the average of 2%.

The next test set consisted of five variable-current load profiles P1–P5, and are shown in Figure 2. Table I shows the measured and predicted lifetimes (L_m and L_p, respectively) as well as the measured and predicted delivered charges (C_m and C_p, respectively). Note that the charge errors were within 2%, while the maximum lifetime error was 3%. One can see that our model has adequately captured the trend in battery behavior observed in the experiments, with very small prediction errors.

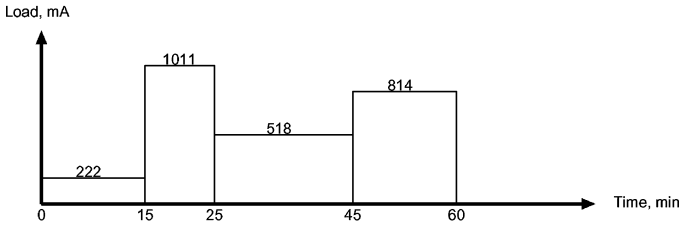
To obtain P1–P4 we selected four currents of certain durations (1011 mA for 10 min, 814 mA for 15 min, 518 mA for 20 min, and 222 mA for 15 min), which were arranged in different order. For each of these four profiles, the total



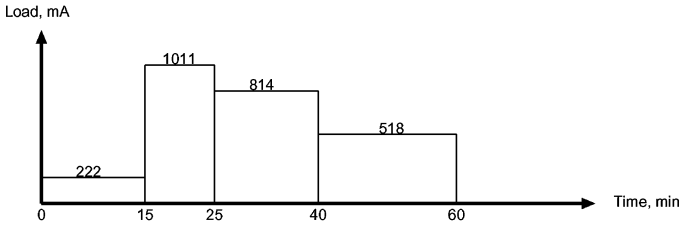
(a) Profile P1



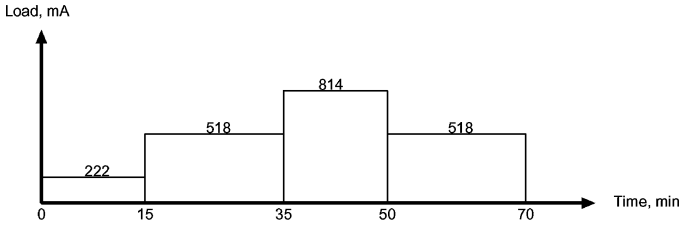
(b) Profile P2



(c) Profile P3



(d) Profile P4



(e) Profile P5

Fig. 2. Experimental load profiles.

Table I. Profile Lifetimes and Delivered Charges

Profile	Measured		Predicted		Lifetime Error (%)	Charge Error (%)
	L_m (min)	C_m (mA-min)	L_p (min)	C_p (mA-min)		
P1	64.9	37 098	66.9	37 542	3.1	1.2
P2	54.0	29 944	54.4	30 348	0.7	1.3
P3	55.8	32 591	55.0	31 940	1.4	2.0
P4	58.4	35 181	57.5	34 715	1.5	1.3
P5	67.5	34 965	67.0	34 706	0.7	0.7

length and delivered charge are 60 min and 36 010 mA-min, respectively. Note that in P1, after 60 min, the battery is discharged at 222 mA until a failure occurred.[2] In P1 the load is decreasing, and in P2 the load is increasing. The results show that P1 is the best sequence, and P2 is the worst sequence, from the battery perspective. *The battery behavior depends on the characteristics of the load profile*.

Indeed, in P1 after 60 min, the battery survives for another 4.9 min under 222 mA (residual 1088 mA-min charge). However, in P2 the battery fails to service the last 6.0 min under 1011 mA (undelivered 6066 mA-min charge). For P1 and P2, the difference in the total delivered charge is as much as 20% of 36 010 mA-min. As predicted by the battery model and demonstrated by the measurements, the other alternative sequences, P3 and P4, are neither better than P1 nor worse than P2.

The last profile, P5, shows the benefit of reducing battery load by decreasing energy consumption of a hypothetical processor through reducing its voltage. To obtain P5, we started from P2 and changed the failing 10-min load of 1011 mA to a 20-min load of 518 mA to reflect a change in the processor voltage. Note that charge demanded from the battery is approximately the same before and after voltage reduction.[3] The profile length has increased by 10 min, and the battery failure occurs at 67.5 min. The total delivered charge is 34 966 mA-min, which is a noticeable improvement over P2 with 29 944 mA-min.

1.2 Summary of Key Contributions

The main focus of the research described in this paper is the development of methods for scheduling tasks and selecting task voltages, so as to maximize a (new) charge-based cost function subject to the following constraints:

(1) *dependency constraint*—task dependencies are preserved;
(2) *delay constraint*—the profile length is within the delay budget; and
(3) *endurance constraint*—the battery survives all the tasks.

The first step toward addressing the above problem is the development of an accurate and efficient method for predicting the lifetime of the battery, given a time-varying load profile. Battery lifetime prediction is a difficult problem due

[2]222 mA is applied to determine how much residual charge is left.
[3]This is a pessimistic scenario, since the charge consumption is reduced after the supply voltage is scaled down.

to the fact that the amount of delivered charge, that is, the *actual capacity* of the battery, is a very complex function of the physical and chemical characteristics of the battery and the time-varying load that is applied.

Our investigation of batteries led to the development of a novel battery model that combines accuracy and generality of a simulation-based model and has the simplicity of an analytical model. The main objective here is to develop a model that is both physically justified and analytically simple, so that it can be used to construct a cost function for the optimization methods. A summary of the battery model appears in Section 3, including an example of how the model is applied.

The battery model is used to construct a unique *battery-aware* cost function that is used for optimizing task scheduling and voltage assignment (see Section 4). In contrast to previously reported research on battery-driven energy minimization [Benini et al. 2001; Liu et al. 2001; Luo and Jha 2001], the approach presented in this paper is the first effort to treat construction of battery-efficient load profiles formally, using a precise charge-based cost metric. For example, this makes it possible to formally demonstrate the ordering of a set of independent tasks so as to maximize the residual battery charge after all the tasks are completed, or to determine where idle periods should be inserted to maximize charge recovery, or to identify the best candidate task for voltage reduction (thereby utilizing available delay slack) from a set of scheduled identical tasks. These are all based on provable properties of the charge-based cost metric (see Section 5).

In Section 6, three different approaches toward solving the task scheduling and voltage assignment problem are described. Below is a summary of these methods.

1. The first approach is aimed at minimizing energy consumption that, in our case, corresponds to minimizing the total charge consumed during task execution. Task charges are controlled by scaling task voltages.[4] This approach starts with assigning voltages to tasks so that the total charge consumption is minimized subject to satisfying the delay budget. Energy minimization does not guarantee maximization of battery lifetime, since the battery lifetime is sensitive not only to task charges, but also to task ordering in time. The battery may fail before completing all tasks (i.e., the endurance constraint may be violated), even though the total charge consumption is minimized. In such situations, *task repair* is performed, which reduces the voltage for some tasks in order to reduce the stress on the battery. Once the profile has been repaired, its length may exceed the delay budget. To meet the delay constraint, a latency reduction procedure is applied. This scales up the task voltages, while ensuring that no failures are introduced.
2. The second method starts with the highest-power initial profile by assigning all tasks to the highest voltage. Since the clock frequency is also the highest (i.e. task durations are the shortest), the delay constraint is satisfied.

[4]It is assumed that voltage scaling is always accompanied by a corresponding change in the system clock frequency, that is, voltage and clock are scaled *simultaneously*.

However, high task currents may result in the failure of the battery. To satisfy the endurance constraint, task repair is performed while checking that the delay constraint is not violated. Once the profile no longer fails, there may be some delay slack available, that is (*delay budget—profile length*) may be a positive quantity. To further reduce the profile cost, a slack utilization procedure is applied that further scales down task voltages.

3. In contrast to the second approach, the third method starts with the lowest-power initial profile by assigning tasks to the lowest voltage. The endurance constraint is satisfied, but the delay constraint may be violated, since the clock frequency is the lowest (i.e., task durations are the longest). To meet the delay budget, latencies are reduced by scaling up the voltages; this time ensuring that the endurance constraint is not violated.

The techniques described in this paper were exercised on a number of different load profiles, and the results are reported in Section 7. These are compared with profile simulation results, using a microscopic-scale model of a lithium-ion cell. Differences between the simulation results and the results produced by the proposed methods are within 3%. These results demonstrate the accuracy of the battery model and the charge-based cost function.

2. PRIOR RELATED WORK

2.1 Battery Models

Perhaps the most accurate method of modeling a battery is to model the electrochemical processes that take place within the battery. This is the approach described in Doyle et al. [1993], Fuller et al. [1994], and Botte et al. [2000]. The result is the numerical solution to a system of partial differential equations. The main drawbacks of this approach are the long simulation times required and the large number of parameters that need to be specified. Other approaches aimed at reducing the time complexity of low-level simulation are generally based on constructing an abstract representation of the battery [Benini et al. 2000; Gold 1997; Panigrahi et al. 2001]. The main drawback to the above approaches is that they are difficult to justify based on the physics and chemistry of the battery. As with the simulation-based method, these approaches are also difficult to incorporate within the framework of battery lifetime optimization. Analytical models that capture some of the key factors determining the battery performance for special cases are described in Doyle and Newman [1995] and Pedram and Wu [1999].

2.2 Battery-Aware Task Scheduling

Several papers have considered the battery issues to improve system operation [Benini et al. 2001; Liu et al. 2001; Luo and Jha 2001]. In Benini et al. [2001], a VHDL-based simulation model [Benini et al. 2000] was used to expose the impact of different dynamic power management policies on the battery lifetime. The authors investigated both single-battery- and dual-battery-powered systems while studying both time-out open-loop (battery-voltage-independent)

and threshold-based closed-loop (battery-voltage-dependent) policies. Luo and Jha [2001] considered static scheduling of tasks with real-time constraints. Evaluation of the proposed method was based on the battery model combining Peukert's law [Linden 1995] and ideas from Pedram and Wu [1999]. The battery-sensitive schedule was achieved by reducing the variance and the peak power of a generated discharge current profile. Battery lifetime improvements reported in Benini et al. [2001] and Luo and Jha [2001] should be interpreted with care, since the results are heavily biased by the properties of an abstract model describing the battery behavior. Battery-aware scheduling under timing constraints was also addressed in Liu et al. [2001], where a NASA/JPL Mars Pathfinder rover was used as a motivating application. The rover featured two power sources: a battery and a solar panel. The objective was to utilize the solar panel (the "free" energy source) as much as possible and minimize the energy drawn from the battery. The scheduler accounted for the presence of an alternative energy source in addition to the battery, but not for the battery behavior.

2.3 Scheduling and Voltage Assignment to Minimize Energy Consumption

Minimizing the traditional metric—energy consumption—is *not sufficient* for maximizing battery lifetime. The cost function used here generalizes the energy consumption metric by incorporating a dependency on the task ordering and the profile duration. Moreover, the endurance constraint (i.e., the battery must survive until the last task is completed) imposes additional limitations on acceptability of a given task sequence with a given task voltage assignment. Much of the existing literature on task scheduling with voltage scaling focuses on energy minimization only. The following review is of papers that describe scheduling methods for a single processor.

Weiser et al. [1994] introduce MIPJ (millions-of-instructions per Joule) as a quality metric for dynamic voltage scaling (DVS). The key idea is to eliminate idle time by reducing the processor voltage and clock for a given segment of computation. To predict processor utilization, either a fixed-size window of future events or a fixed-size window of past events is analyzed, and the corresponding DVS decisions are evaluated using trace-based simulations (further evaluations are reported in Pering et al. [1998]).

Yao et al. [1995] describe a minimum-energy preemptive scheduling algorithm, based on the notion of a critical interval. In such intervals, the corresponding subset of tasks must be assigned to the maximum constant voltage and clock in any optimal schedule. The algorithm works recursively: once the critical interval is identified and its tasks are scheduled, a new problem instance is created and solved for the remaining tasks. The authors assume that tasks are independent with arbitrary arrival times, and adopt the earliest-deadline-first scheduling policy. A similar approach is described in Quan and Hu [2001]; however, it is assumed that task priorities are fixed and task timing parameters (such as arrival times, deadlines, and the number of clock cycles) are known a priori. An efficient heuristic, based on handling critical intervals, computes a voltage schedule that is guaranteed to consume less energy than an alternative

of using the minimum constant voltage and shutting down the system during idle periods.

Shin et al. [2000] consider both fixed-priority and dynamic-priority scheduling of periodic tasks with the same arrival times. The proposed method consists of the two components: offline computation of the minimum constant voltage setting under the assumption of the worst-case task latencies, and online voltage adjustment or system power-down exploiting idle periods due to dynamic variations in the number of clock cycles required for completion of a given task. Note that execution time requirements can vary significantly, especially in multimedia applications such as streaming MPEG video. Simunic et al. [2001] develop and verify a stochastic model for prediction of execution times for multimedia tasks on a frame-by-frame basis. Finally, Sinha and Chandrakasan [2001] describe a modification of the preemptive earliest-deadline-first algorithm that minimizes energy in addition to minimization of maximum lateness for a set of independent arbitrary tasks.

In Ishihara and Yasuura [1998], equations for CMOS gate delay and dynamic power dissipation are used to show that (i) if continuously variable voltages are supported, assigning each task to a single voltage minimizes energy under a delay constraint; and (ii) if a small number of discrete voltages are supported, using at most two voltages for each task minimizes energy under a delay constraint. The authors also provide an ILP (integer linear programming) formulation of the voltage scheduling problem. An extension of this work can be found in Okuma et al. [2001], where offline and online voltage scheduling techniques are described. Manzak and Chakrabarti [2001] and Pering and Brodersen [1998] conclude that the minimum energy is obtained when all the tasks are assigned to the same voltage, provided that the deadlines are not violated. Also, Qu [2001] presents upper bounds on energy savings for various types of DVS systems.[5] To further increase energy savings due to intertask voltage scheduling (i.e., the supply voltage is adjusted on a task-by-task basis), Shin et al. [2001] advocates adjusting the supply voltage within individual task boundaries. Based on static timing analysis, the proposed scheduling algorithm selects locations in a program for inserting voltage-scaling code, so that all the slack time from dynamic variations of different execution paths is exploited.

3. BATTERY MODEL

An essential ingredient of any energy management strategy for a battery-powered system is a method for predicting the *lifetime* or *time-to-failure* of the battery given a load profile. In this section a new model of a battery is presented. The model, although highly simplified, is based on the electrochemical behavior of the battery. The result is a parametrically simple (contains two parameters that need to be estimated) analytical form that relates the battery lifetime to the time-varying load profile. This form also provides a means to

[5]DVS systems considered in Qu [2001] include (i) an ideal supply voltage that can be changed arbitrarily and instantaneously; (ii) a discrete set of supply voltages that can be switched to instantaneously; and (iii) a range within which the supply voltage can be varied at a limited maximum rate.

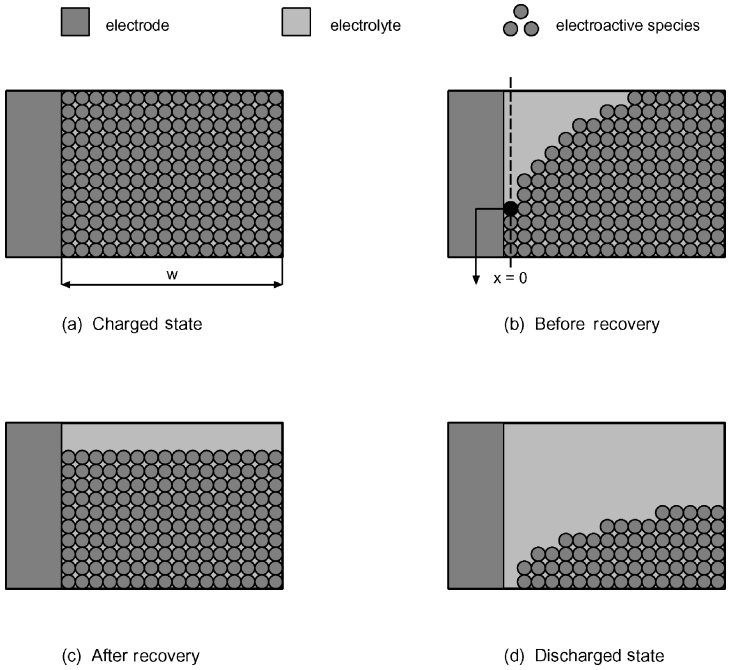


Fig. 3. Physical picture of our model.

define a charge-based cost function to be used in battery lifetime optimization procedures.

A battery consists of a positive (*cathode*) and negative (*anode*) electrode that are separated by an electrolyte. During the discharge phase, the anode releases electrons to the external circuit and the cathode accepts electrons from the circuit. The chemical processes are reversed during the charging phase. We assume that the battery is symmetric, and therefore the chemical processes at both electrodes are identical. Figure 3 illustrates a highly simplified, one-dimensional view of the battery operation. Initially, when the system is in equilibrium, the *electroactive species* are uniformly distributed across the linear diffusion region of width w (Figure 3(a)).

Once a load is attached to the battery, the external flow of electrons is established, and the electrochemical reaction results in reduction of the number of species near the electrode. Thus, a nonzero concentration gradient is created across the electrolyte (Figure 3(b)), and the laws of diffusion apply. If the load is switched off, then the concentration near the electrode surface will start to increase, or *recover* (Figure 3(c)), due to diffusion, and eventually, the concentration gradient will become zero again. The electroactive species will again become uniformly distributed in the electrolyte; however, the concentration level will be smaller than the initial value. Finally, once the concentration of the electroactive species at the electrode surface drops below a threshold, the reaction can no longer be sustained and the battery is considered to be discharged (Figure 3(d)).

3.1 Relationships Among Discharge Current, Battery Parameters, and Lifetime

We are interested in determining the time when the battery becomes *discharged*. The analysis is based on a one-dimensional model of diffusion in a finite region of length w. Let $C(x,t)$ denote the concentration of species at time $t \in [0,L]$ at distance $x \in [0,w]$ from the electrode. We are interested in the concentration values at the electrode surface ($x = 0$). Let the initial concentration be C^*, and let $\rho(t) = 1 - \frac{C(0,t)}{C^*}$. When $C(0,t)$ drops below the cutoff level C_{cutoff} at time $t = L$, the value of $\rho(L)$ crosses over the corresponding threshold $(1 - \frac{C_{\text{cutoff}}}{C^*})$. We need to find an analytical expression for $\rho(t)$ in order to compute the time-to-failure, L.

The following two *Fick's laws* describe concentration behavior due to one-dimensional diffusion [Bard and Faulkner 1980]:

$$-J(x,t) = D\frac{\partial C(x,t)}{\partial x}, \tag{1}$$

$$\frac{\partial C(x,t)}{\partial t} = D\frac{\partial^2 C(x,t)}{\partial x^2}. \tag{2}$$

$J(x,t)$ denotes the flux of species at time t at distance x, and D denotes the diffusion coefficient. In accordance with *Faraday's law*, the flux at the electrode surface ($x = 0$) is proportional to the current $i(t)$ (the external load applied) [Bard and Faulkner 1980]. The flux at the other boundary of the diffusion region ($x = w$) is zero. Therefore, the following two boundary conditions apply:

$$\frac{i(t)}{\nu FA} = D\frac{\partial C(x,t)}{\partial x}\bigg|_{x=0}, \tag{3}$$

$$0 = D\frac{\partial C(x,t)}{\partial x}\bigg|_{x=w}. \tag{4}$$

In (3), A is the area of the electrode, ν is the number of reacting electrons, and F denotes the Faraday's constant. It is possible to obtain an analytical solution for these pairs of partial differential equations and boundary conditions. Derivation of the solution is given in Appendix A. The final result is as follows:

$$\rho(t) = \frac{1}{\nu FAwC^*}\left[\int_0^t i(\tau)\,d\tau + 2\sum_{m=1}^{\infty}\int_0^t i(\tau)\,e^{-\frac{\pi^2 D(t-\tau)m^2}{w^2}}\,d\tau\right]. \tag{5}$$

Let $\beta = \frac{\pi\sqrt{D}}{w}$ and $\alpha = \nu FAwC^*\rho(L)$. Then, one obtains the following general expression relating the load, the time-to-failure, and the two battery parameters, α and β:

$$\alpha = \int_0^L i(\tau)\,d\tau + 2\sum_{m=1}^{\infty}\int_0^L i(\tau)\,e^{-\beta^2 m^2(L-\tau)}\,d\tau. \tag{6}$$

Equation (6) relates the lifetime L to the load profile $i(t)$. It involves two parameters, α and β, that need to be estimated. The unit of α is coulombs and that of β^2 is second^{-1}. The lifetime L is defined as the point in time when the concentration of the electroactive species at the electrode surface falls below a

given threshold. The right-hand side of Eq. (6) represents the *capacity* of the battery. The first term is simply the total charge consumed by the system. The second term is the amount of charge in the battery that could not be used by the system because it was not available at the electrode surface at the time of failure. As β increases, the second term goes to zero. Thus a large β means that the battery is practically an ideal source (total charge consumed by the system at the time of failure is the total capacity of the battery). Intuitively, this is because a larger value β implies a *faster* diffusion, which means that the electroactive species are able to reach the electrode surface faster, and able to generate electricity at a rate demanded by the system. On the other hand, a small value of β indicates a departure from an ideal source. In this case, at the time of failure, not all of the capacity has been used. Consequently, a rest period of sufficient duration will result in an equilibrium being reestablished (concentration gradient approaching zero), and some of the electroactive species are now available at the electrode surface to participate in electricity generation. This is the process of recovery.

To specify the model completely, the parameters α and β have to be estimated for a given battery. This can be accomplished by carrying out a set of *constant load* tests. Specifically, for a constant discharge current I, Eq. (6) reduces to

$$\alpha = IL\left[1 + 2\sum_{m=1}^{\infty}\frac{1 - e^{-\beta^2 m^2 L}}{\beta^2 m^2 L}\right]. \tag{7}$$

We apply a given a set of constant loads $I_{(1)}, \ldots, I_{(N)}$ until the battery is exhausted. This results in a set of lifetime measurements $L_{(1)}, \ldots L_{(N)}$. α and β are estimated by minimizing the sum of squares $\sum |I_{(k)} - \hat{I}_{(k)}|^2$, where $\hat{I}_{(k)}$ is given by[6]

$$\hat{I}_{(k)} = \frac{\alpha}{L_{(k)} + 2\sum_{m=1}^{\infty}\dfrac{1 - e^{-\beta^2 m^2 L_{(k)}}}{\beta^2 m^2}}. \tag{8}$$

Once α and β are estimated, the battery is characterized. Figure 4 shows a sequence of tasks, each of which imposes a constant load on the battery. The resulting *load profile*, which is a n-step staircase function, is also shown. I_k, Δ_k, and t_k denote the current, duration, and start time of task k, respectively. The load profile is specified by the three sets: the current set $S_I = \{I_k \mid k = 0, 1, \ldots, n-1\}$; the duration set $S_\Delta = \{\Delta_k \mid k = 0, 1, \ldots, n-1\}$; and the start time set $S_t = \{t_k \mid k = 0, 1, \ldots, n-1\}$. Assume that the battery fails during task u. Then, given a load profile, the relationship between the battery parameters, the discharge currents, and lifetime is obtained by applying Eq. (6). The result is

$$\alpha = \sum_{k=0}^{u-1} I_k F(L, t_k, t_k + \Delta_k, \beta) + I_u F(L, t_u, L, \beta), \tag{9}$$

[6]The terms of the infinite series diminish very rapidly, allowing truncation after a few values of m.

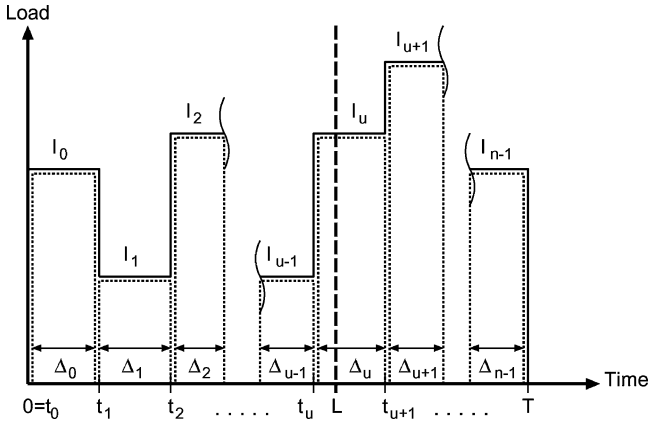


Fig. 4. Battery load profile.

where

$$F(x, y, z, \beta) = z - y + 2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (x-z)} - e^{-\beta^2 m^2 (x-y)}}{\beta^2 m^2}. \tag{10}$$

3.2 Example: Interrupted Load

To illustrate the utility of our model, we describe one of the experiments conducted with a lithium-ion battery. The open-circuit voltage of the battery was 4.2 V, and the cutoff voltage was set to 3.0 V. To estimate model coefficients, we performed ten constant-current discharge tests. From the corresponding load-lifetime samples, we obtained $\alpha = 39\,668$ and $\beta = 0.574$.

As a simple example of a variable-current discharge profile, we considered the following interrupted load. For the first 25 min the discharge current was 912 mA. Then, the load was turned off for 10 min and afterward, 912 mA was applied again for another 25 min. Under these conditions, the battery lasted for 43.8 min. Our model predicted 44.2 min, that is, the lifetime prediction error is 1%. The total charge was 30 826 mA-min, with our prediction of 31 190 mA-min, which yields 1% charge prediction error. Figure 5 shows the measured battery voltage and the residual charge predicted by our model. Note that the battery voltage and the residual charge exhibit the same behavioral trends.

In addition to the measurements presented here, we carried out an extensive evaluation of the model with respect to (i) a microscopic-scale simulation model of a lithium-ion cell, and (ii) measurements taken on a lithium-ion battery. Over twenty variable-current load profiles were tested, and the maximum error of lifetime predictions due to our model was less than 5%. Tests included interrupted, linear, periodic, and nonperiodic loads, which were inspired by typical applications run on a pocket computer [Rakhmatov et al. 2002].

While the model derivation is not specific to a particular chemistry, the validation is performed for lithium-ion batteries only. We focus on lithium-ion because it is the prevalent chemistry used in portable devices today, due to its high-energy density and low-maintenance requirements (e.g., no memory effect).

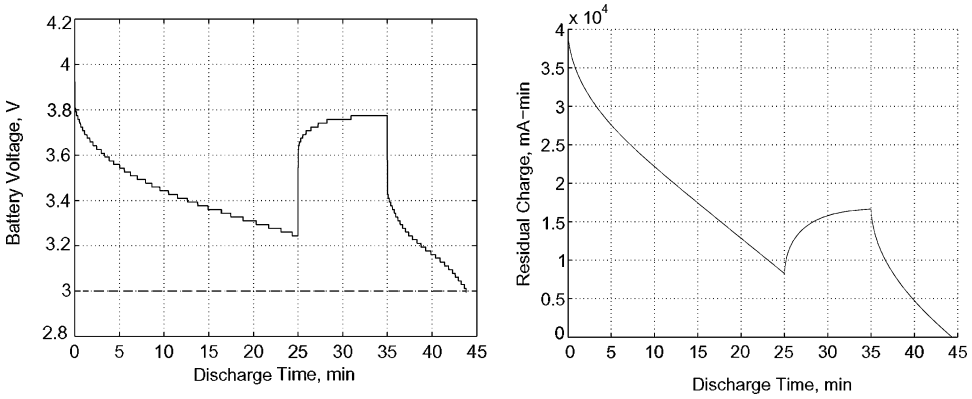


Fig. 5. Measured battery voltage and predicted residual charge for interrupted load.

4. BATTERY-AWARE TASK SCHEDULING PROBLEM

For a given profile of length T, let

$$\sigma = \sum_{k=0}^{n-1} I_k F(T, t_k, t_k + \Delta_k, \beta). \tag{11}$$

Comparing Eqs. (11) and (9), we see that σ is the charge that the battery has lost by time T. If $\sigma < \alpha$, then the battery is still operational at time T. We use σ as our *battery-aware cost function* to be *minimized*.

Let B denote the delay budget. For a valid profile, the latency T must not exceed B. Also, the battery must not fail anywhere within a profile, that is,

$$\alpha \geq \sum_{k=0}^{n-1} I_k F(t, \min\{t, t_k\}, \min\{t, t_k + \Delta_k\}, \beta)\ ,\ \forall t \leq T. \tag{12}$$

If, for some load k, its start time $t_k \geq t$, then it does not contribute to the value of the sum. Consequently, the right-hand side of Eq. (12) represents the total charge lost by the battery up to time t, and this must be less than the total capacity of the battery, for all $t \leq T$. This condition is necessary to account for the *relaxation effects* (charge recovery) that might mask a failure taking place before T. In order for the battery to be operational up to T, no subprofile of length $t \leq T$ may be too extreme.

4.1 Task Voltage/Clock Scaling

In Eq. (11), I_k denotes the current *drawn from the battery* during execution of task k. In other words, I_k is the input current of the DC–DC converter (voltage regulator) serving as an interface between the processor and the battery. Usually, a user can specify the power P_k demanded by task k from the output of the DC–DC converter. Let ϵ denote the power conversion efficiency, and assume that $\varepsilon =$ constant over the range of loads of interest. Also, let V_k and ϕ_k, respectively, denote the operating voltage and the corresponding maximum clock frequency for task k, and assume that the battery voltage is some

PROBLEM: Battery-Aware Task Sequencing with Voltage Scaling
INPUT: S_V, S_ϕ, $\vec{G}$, B, α, β
OUTPUT: S_I, S_Δ, S_t
OBJECTIVE: $min\{\sigma = \sum_{k=0}^{n-1} I_{ik} F(T, t_k, t_k + \Delta_{ik}, \beta)\}$
CONSTRAINTS:
(1) if q depends on p in $\vec{G}$, then $t_q \geq t_p + \Delta_p$
(2) $T \leq B$
(3) $\forall\, t \leq T,\ \sum_{k=0}^{n-1} I_{ik} F(t, min\{t, t_k\}, min\{t, t_k + \Delta_{ik}\}, \beta) \leq \alpha$

Fig. 6. Voltage/clock scaling problem.

averaged constant V_{ave}. Then, $I_k = \frac{P_k}{\epsilon V_{\text{ave}}}$. Since $P_k \propto V_k^2 \phi_k$ and ϕ_k is approximately proportional to V_k [Burd and Brodersen 2002], one obtains the following approximate relationship: $I_k \propto V_k^3$. On the other hand, task delay $\Delta_k \propto V_k^{-1}$, since $\Delta_k = \frac{N_k}{\phi_k}$, where N_k is the number of clock cycles necessary to complete task k.

This simple, first-order analysis clearly shows that voltage/clock scaling is a powerful tool for controlling the load profile. The main trade-off is between a decrease (increase) in the battery stress, that is, the discharge current, and an increase (decrease) in the duration of the stress. Note that supply voltage scaling is always accompanied by proper changes of the clock frequency. In the remainder of this paper, whenever voltage scaling is mentioned, the corresponding clock scaling is implied.

4.2 Problem Formulation

We assume that the system can operate at any voltage V_i from the ordered set $S_V = (V_0, V_1, \ldots, V_{K-1})$ at the corresponding maximum frequency ϕ_i from the ordered set $S_\phi = (\phi_0, \phi_1, \ldots, \phi_{K-1})$. The elements of S_V and S_ϕ are in ascending order. Let I_{ik} denote the current drawn from the battery, when the system is executing task k at V_i with the clock ϕ_i. Let Δ_{ik} denote the corresponding load duration. Figure 6 shows the formulation of the battery-aware voltage/clock scaling problem. The input consists of the sets S_V and S_ϕ, the task graph $\vec{G}$ representing task dependencies, the delay budget B, and the battery-specific parameters α and β. The objective is to assign each task k to a specific voltage V_i at the frequency ϕ_i (thus determining the task current I_{ik} and the task duration Δ_{ik}) and the start time t_k, so that the resulting profile cost σ is minimized and the following constraints are not violated:

(1) *dependency constraint*—task dependencies are preserved;
(2) *delay constraint*—the profile length is within the delay budget; and
(3) *endurance constraint*—the battery survives all the tasks.

Since both scheduling and voltage scaling are being considered, the output consists of not only the task start times (S_t), but also task currents (S_I) and task durations (S_Δ). This expands the search space considerably, offering much greater opportunity for improving battery discharge profiles.

The cost function σ and the endurance constraint (3) are *unique* features of the problem at hand. Traditionally, the objective of task scheduling with supply voltage scaling has been to minimize energy subject to task precedence and

deadline constraints. For a given set of n tasks, the traditional goal has been to minimize $\sum_{k=0}^{n-1} P_k \Delta_k$, or equivalently, $\varepsilon V_{\text{ave}} \sum_{k=0}^{n-1} I_k \Delta_k$ (see Section 4.1 for details). Thus, energy minimization translates into charge minimization. The existing work on task scheduling with voltage scaling reviewed in Section 2 focuses on energy minimization only. In the work described here, $\sum_{k=0}^{n-1} I_k \Delta_k$ is the *lower bound* on cost function σ. That is, for a given load profile, energy minimization does *not imply* maximization of battery lifetime. This is because the cost function σ is also sensitive to the task start times and the profile duration. Moreover, the endurance constraint (3) imposes additional limitations on the validity of a given task sequence with a given task voltage assignment.

We now consider two *special cases* of the problem and show how they can be formulated in terms of well-known optimization problems.

4.3 Special Case: Large α and β

Since α represents the battery *capacity*, a sufficiently large value of α means that the endurance constraint will be satisfied, and can therefore be ignored. A large value of β means that the battery behaves as an ideal source, or equivalently, for each task k,

$$F(T, t_k, t_k + \Delta_{ik}, \beta) = \Delta_{ik} + 2 \sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T - t_k - \Delta_{ik})} - e^{-\beta^2 m^2 (T - t_k)}}{\beta^2 m^2} \approx \Delta_{ik}. \quad (13)$$

If no idle periods are allowed, then the profile length T is equal to $\sum_{k=0}^{n-1} \Delta_{ik}$. Note also that task start times t_k no longer affect the cost function. Consequently, the dependency constraint (1) does not affect the quality of the solution. The delay constraint (2) is the only condition that needs to be considered. Let x_{ik} denote a 0-1 decision variable, $x_{ik} = 1$, if task k is assigned to the ith voltage level; otherwise, $x_{ik} = 0$. Then the objective function and the constraints are expressed as

$$\begin{aligned} &\min \left\{ \sum_{i=0}^{K-1} \sum_{k=0}^{n-1} x_{ik} I_{ik} \Delta_{ik} \right\}, \\ &\sum_{i=0}^{K-1} \sum_{k=0}^{n-1} x_{ik} \Delta_{ik} \le B, \\ &\sum_{i=0}^{K-1} x_{ik} = 1, \ \forall k = 0, 1, \ldots, n-1. \end{aligned} \quad (14)$$

The above formulation is an instance of the well-known *multiple-choice 0-1 knapsack problem*. This can be seen by making the following substitutions in symbols and terminology: $p_{ik} = -I_{ik} \Delta_{ik}$ is the *profit* of item k from class i; $w_{ik} = \Delta_{ik}$ is the *weight* of item k from class k; $c = B$ is the *capacity*. Then (14)

can be re-written as

$$\max\left\{\sum_{i=0}^{K-1}\sum_{k=0}^{n-1}x_{ik}p_{ik}\right\},$$
$$\sum_{i=0}^{K-1}\sum_{k=0}^{n-1}x_{ik}w_{ik} \le c, \tag{15}$$
$$\sum_{i=0}^{K-1}x_{ik} = 1,\ \forall k = 0, 1, \ldots, n-1.$$

The multiple-choice knapsack problem is known to be NP-hard; however, it can be solved optimally in pseudopolynomial time by dynamic programming techniques [Dudzinski and Walukiewicz 1987].

4.4 Special Case: Fixed Task Voltages

The battery-aware task sequencing problem has similarities to the problem of weighted completion time task sequencing [Hall et al. 1996; Lawler 1978; Sidney 1975]. In this classic problem, we are given tasks with dependencies as well as weights and durations associated with each task. If w_k and d_k denote the weight and the duration of task k, respectively, then the objective is to determine the start times s_k of each task (there is no idle time between consecutive tasks) such that $\sum_k w_k(s_k + d_k)$ is minimized, and dependencies are not violated. This problem is NP-complete [Lawler 1978]. However, for several special cases, an optimal solution can be determined in polynomial time. For example, if there are no dependencies, then the optimal solution is the sequence of tasks ordered in nonincreasing values of the ratio $\frac{w_k}{d_k}$ [Smith 1956]. For sequencing task "chains" rather than individual tasks, the ratios become $\sum w_k / \sum d_k$, where $\sum$ is taken over all tasks in a "chain." The optimal solution is obtained by ordering the "chains" in nonincreasing order of their ratios [Sidney 1975]. In Lawler [1978], it was shown that an optimal solution can be obtained for series—parallel graphs, and the subsets of tasks forming the optimal "chains" can be identified using network flows.

A (weak) link between the weighted completion time sequencing and battery-aware sequencing problems is established by replacing the cost function σ with one of its lower bounds, which will result in a cost function of the form $\sum_k w_k(s_k + d_k)$. Starting from Eq. (11), we have

$$\begin{aligned}
\sigma &= \sum_{k=0}^{n-1} I_k \left[\Delta_k + 2\sum_{m=1}^{\infty}\frac{e^{-\beta^2 m^2(T-t_k-\Delta_k)} - e^{-\beta^2 m^2(T-t_k)}}{\beta^2 m^2}\right] \\
&> \sum_{k=0}^{n-1} I_k \left[\Delta_k + 2\frac{e^{-\beta^2(T-t_k-\Delta_k)} - e^{-\beta^2(T-t_k)}}{\beta^2}\right] \\
&= \sum_{k=0}^{n-1} I_k \Delta_k + 2\sum_{k=0}^{n-1} I_k \frac{e^{-\beta^2 T} - e^{-\beta^2(T+\Delta_k)}}{\beta^2} e^{\beta^2(t_k+\Delta_k)}
\end{aligned}$$

$$= \sum_{k=0}^{n-1} I_k \Delta_k + 2\sum_{k=0}^{n-1} I_k e^{-\beta^2 T} \frac{1 - e^{-\beta^2 \Delta_k}}{\beta^2} \left[1 + \beta^2 (t_k + \Delta_k) + \frac{\beta^4 (t_k + \Delta_k)^2}{2} + \cdots\right]$$
$$> \sum_{k=0}^{n-1} I_k \Delta_k + 2\sum_{k=0}^{n-1} I_k e^{-\beta^2 T} \frac{1 - e^{-\beta^2 \Delta_k}}{\beta^2} + 2e^{-\beta^2 T} \sum_{k=0}^{n-1} I_k \left[1 - e^{-\beta^2 \Delta_k}\right] (t_k + \Delta_k). \quad (16)$$

Note that the terms $\sum_{k=0}^{n-1} I_k \Delta_k$, $\sum_{k=0}^{n-1} I_k e^{-\beta^2 T} \frac{1-e^{-\beta^2 \Delta_k}}{\beta^2}$, and $e^{-\beta^2 T}$ are the same for any sequence of tasks. Therefore, minimization of the last expression in (16) corresponds to minimization of $\sum_{k=0}^{n-1} I_k [1 - e^{-\beta^2 \Delta_k}](t_k + \Delta_k)$. This expression is of the form $\sum_k w_k (s_k + d_k)$, where $s_k = t_k$, $d_k = \Delta_k$, and $w_k = I_k [1 - e^{-\beta^2 \Delta_k}]$.

Finally, without alluding to the weighted completion time-sequencing problem, we show (Theorem B.2) that if there are no constraints and no idle periods, then the optimal solution is the sequence of tasks in nonincreasing order of currents, I_k.

5. COST FUNCTION PROPERTIES

In this section we present several important properties of the cost function σ given in Eq. (11). The relevant theorems and proofs are given in Appendix B.

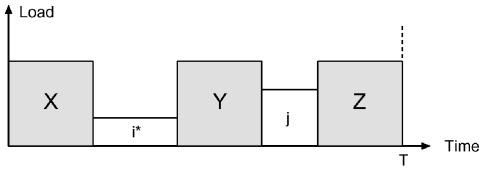
5.1 Properties with Respect to Sequencing

In the task scheduling problem at hand, there is only one processor available. There are $n!$ ways to sequence n tasks. If there are no dependencies, no endurance constraints, and no idle periods allowed, then the best (worst) solution is obtained by sequencing tasks in *nonincreasing* (*nondecreasing*) order of their currents (see Theorem B.2). This result is important not only for the case of no dependencies, but also in a general case when dependencies are present. It provides lower and upper bounds on the value of the cost function.

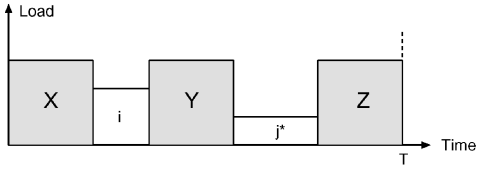
Another property of interest is related to exercising charge recovery effects to repair battery failures. Given a sequence of tasks, assume that a failure occurs during some task l, that is, $t_l \le L \le t_l + \Delta_l$. In other words, the subprofile of length $T_l = t_l + \Delta_l$ violates the endurance constraint. In order to repair l, we must insert an idle (offline) period somewhere within the subprofile in question. Let δ denote the duration of the inserted idle period. According to Theorem B.4, the subprofile cost is minimized if the idle period is inserted *immediately* before failing task l, that is, the load is turned off during the interval $[t_l, t_l + \delta]$. Placing the idle period immediately before the failing task also minimizes the delay penalty due to repair. There may exist a situation in which a profile *cannot* be recovered, regardless of the recovery period length. Theorem B.5 addresses this situation.

5.2 Properties with Respect to Scaling

Consider some task k in a profile. Assume that the voltage of task k is scaled down. Due to voltage down-scaling, the current of task k has *decreased* from I_k to $\hat{I}_k$, the duration of k has *increased* from Δ_k to $\hat{\Delta}_k$, and the profile length has *increased* from T to $\hat{T} = T - \Delta_k + \hat{\Delta}_k$. Note that the start time t_k of task k



(a) Scaling earlier task



(b) Scaling later task

Fig. 7. Voltage down-scaling for two identical tasks.

has not changed. Theorem B.8 states that scaling down the voltage of k *always* reduces the cost of the profile. In addition, if a profile is failure-free before the voltage is scaled for some tasks, then it will not fail after voltage down-scaling (see Theorem B.9).

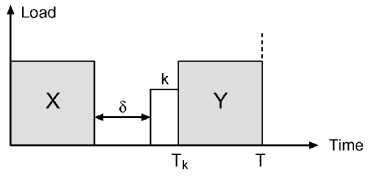
Next, consider two *identical* tasks i and j in a profile of length T. Assume that i precedes j (i.e., $t_i < t_j$), and there is a slack of length δ available, which can be utilized by down-scaling either the voltage of i or the voltage of j. These two possibilities are illustrated in Figure 7. Theorem B.10 states that voltage down-scaling of j is *better* than voltage down-scaling of i. This claim is trivially extended to the case of more than two identical tasks: one should always down-scale the voltage for the latest one to achieve the lowest cost.

Finally, we compare two ways of repairing a battery failure: idle period insertion and voltage down-scaling. Assume that some task k is failing, and a delay slack of length $\delta > 0$ is available. We have two options for repairing the failure: (i) insert an idle period of length δ immediately before k; or (ii) down-scale the voltage of k so that the slack is fully utilized. These options are illustrated in Figure 8. The second choice is *always* better than the first choice (Theorem B.11).

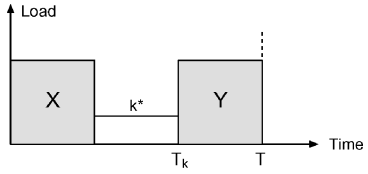
6. ALGORITHMS FOR TASK SCHEDULING WITH VOLTAGE SCALING

In this section we describe three approaches for performing task scheduling with voltage scaling, with the objective of maximizing the charge-based cost function given in Eq. (11). As an example, we use Table II, which shows dependencies and specifications for eight tasks $T1$–$T8$ with two possible supply voltages: V_0 and $V_1 > V_0$. The delay budget B is assumed to be 90 min, and we let $\alpha = 40\,000$ and $\beta = 0.2$.

The first method starts with a voltage assignment that consumes the minimum charge, and then it (i) sequences tasks; (ii) repairs failures, if any, by



(a) Idle period insertion



(b) Voltage down-scaling

Fig. 8. Idle period insertion versus voltage down-scaling.

Table II. Task Currents and Durations

Task	Voltage V_0		Voltage V_1		Parents
	I (mA)	Δ (min)	I (mA)	Δ (min)	
$T1$	125	10	1000	5	—
$T2$	93	10	750	5	—
$T3$	62	20	500	10	$T1$
$T4$	31	20	250	10	—
$T5$	100	10	800	5	$T2, T3$
$T6$	75	10	600	5	$T4, T5$
$T7$	50	20	400	10	$T1$
$T8$	25	20	200	10	$T2, T7$

scaling down the voltages; and (iii) reduces the profile duration, if necessary, through scaling up the voltages without introducing new failures. For our example, the minimum-charge initial profile P1 is shown in Figure 9(a). It is failure-free: steps (ii)–(iii) are not necessary. The task ordering is ($T4, T1, T7, T2, T8, T3, T5, T6$), and the task voltages are ($V_1, V_0, V_1, V_0, V_1, V_0, V_0, V_0$).

The second method scales down the voltages starting from the highest-power initial solution, as illustrated in Figures 9(b)–(d). Since the task currents are the highest possible, the endurance constraint may be violated. For our example, the highest-power initial profile P2, shown in Figure 9(b), fails. After failures are repaired by voltage down-scaling, we obtain profile P3—see Figure 9(c). Note that there is still some slack available, and the voltage is scaled down even further: Figure 9(d) shows the final solution, P4. The task ordering is ($T1, T2, T3, T5, T4, T6, T7, T8$), and the task voltages are ($V_0, V_0, V_1, V_0, V_1, V_0, V_0, V_1$).

Finally, the third method scales up the voltages, starting from the lowest-power initial solution, as illustrated in Figures 9(e)–(f). Since the task durations are the longest possible, the delay constraint may be violated. For our example,

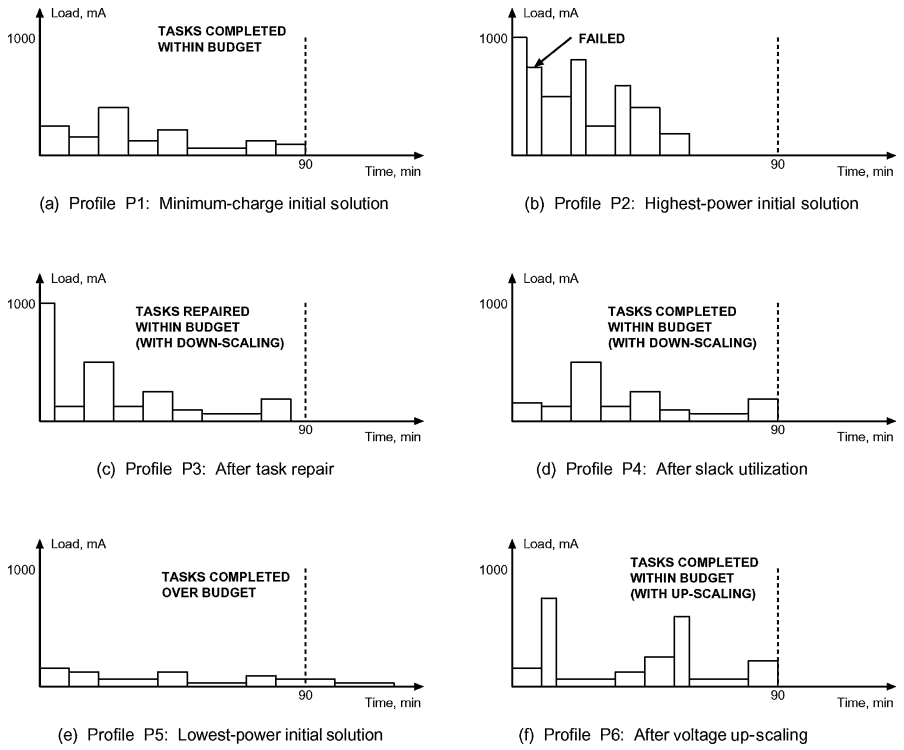


Fig. 9. Example: three approaches to task scheduling with voltage scaling.

the lowest-power initial profile P5 is shown in Figure 9(e). To meet the delay budget, we perform voltage upscaling without introducing failures, and obtain the final solution, P6. The task ordering is $(T1, T2, T3, T5, T4, T6, T7, T8)$, and the task voltages are $(V_0, V_1, V_0, V_0, V_1, V_1, V_0, V_1)$.

Next, we describe the proposed methods in detail.

6.1 Charge Minimization Approach

This approach first ignores the component of the cost function that includes the task start times and determines a voltage assignment such that the sum of task charges is minimized, and the sum of task durations does not exceed the delay budget, B. Once the initial voltage assignment is found, the next step is to generate a task sequence. If the resulting profile is failing, voltage downscaling and/or idle period insertion is performed in order to repair failing tasks. If a failure-free profile exceeds the delay budget, one can use voltage upscaling in order to reduce the profile length, T, so that it is within B. Figure 10 shows the major steps in this approach.

6.1.1 *Step I: Initial Profile Construction.* We assume that the total profile duration does not exceed the delay budget, B, when every task is assigned the highest possible voltage, V_{K-1}. That is, $\sum_{k=0}^{n-1} \Delta_{(K-1)k} \leq B$. This guarantees that a solution to the corresponding knapsack problem exists.

```
ChargeMinimization (S_V, S_φ, G⃗, B, α, β)
  {S_I, S_Δ} = MultipleChoiceKnapsack(S_V, S_φ, B)
  S_t = TaskSequence(G⃗, S_I, S_Δ)
  FOR k = 0, 1, ..., n − 1
    {I'_k, Δ'_k} = LookUp(k, V_0, φ_0)
  B' = Σ_{k=0}^{n−1} Δ'_k
  {S_I, S_Δ, S_t, flag} = TaskRepair(G⃗, S_V, S_φ, S_I, S_Δ, S_t, max{B, B'}, α, β)
  IF flag = FAILURE
    RETURN {S_I, S_Δ, S_t, FAILURE}
  {S_I, S_Δ, S_t, flag} = LatencyReduction(S_V, S_φ, S_I, S_Δ, S_t, B, α, β)
  RETURN {S_I, S_Δ, S_t, flag}
```

Fig. 10. Voltage assignment minimizing total charge consumption.

Procedure *MultipleChoiceKnapsack*$(\cdot)$ returns an exact solution to the following problem:

$$\max\left\{\sum_{i=0}^{K-1}\sum_{k=0}^{n-1} x_{ik}(-I_{ik}\Delta_{ik})\right\}, \quad \sum_{i=0}^{K-1}\sum_{k=0}^{n-1} x_{ik}\Delta_{ik} \le B,$$
$$\sum_{i=0}^{K-1} x_{ik} = 1, \quad \forall k = 0, 1, \ldots, n-1. \tag{17}$$

Recall that $x_{ik} = 1$ if and only if task k is assigned the ith voltage level. Note that B and $\Delta_k,\ k = 0, 1, \ldots, n-1$ must be integers. If these quantities are not integers, then one needs to multiply them by an appropriate factor to achieve integrality. Problem (17) is solved by dynamic programming with the following recursion formula, adopted from Dudzinski and Walukiewicz [1987] with minor modifications:

$$f[k, d] = \max_{i\in[0,K-1]}\{-I_{ik}\Delta_{ik} + f[k-1, d-\Delta_{ik}]\}, \tag{18}$$

where $f[k, d]$ is the optimal value of the partial knapsack with $(k+1)$ tasks and the delay budget d. The permissible range of k is $\{0, 1, \ldots, n-1\}$, and the permissible range of d is $\{0, 1, \ldots, B\}$. Note that $f[k-1, d-\Delta_{ik}]$ equals $-\infty$ for $(k > 0, d \le \Delta_{ik})$ or $(k \le 0, d < \Delta_{ik})$, and $f[k-1, d-\Delta_{ik}]$ equals $-I_{ik}\Delta_{ik}$ for $(k = 0, d \ge \Delta_{ik})$. The final result is $f[n-1, B]$, that is, all n tasks with the total delay budget B have been considered.

Thus, *MultipleChoiceKnapsack*$(\cdot)$ generates the sets S_I and S_Δ, which contain the current and the duration for each task k: if $x_{ik} = 1$, then $\{I_k, \Delta_k\} =$ *LookUp*(k, V_i, ϕ_i). Subroutine *LookUp*(k, V_i, ϕ_i) is used to look up, in a user-specified table, I_k and Δ_k for task k operating at voltage V_i and clock rate ϕ_i.

To complete the load profile specification, we need to determine task start times S_t. For this purpose, we use *TaskSequence*$(\cdot)$ to sequence tasks with no idle periods allowed, so that the profile length T is equal to the sum of task durations $\sum_{k=0}^{n-1}\Delta_k$. Since the knapsack solver *MultipleChoiceKnapsack*$(\cdot)$ ensures that $\sum_{k=0}^{n-1}\Delta_k \le B$, the resulting profile does not violate the delay constraint. Task

sequencing is performed as follows [Rakhmatov et al. 2002]:

1. For each task p, compute its weight $w(p)$ as follows:
   (a) let $\vec{G}_p$ denote the subgraph of the task graph $\vec{G}$ induced by p;
   (b) set $w(p)$ equal to the greater of I_p and $(\sum_{k \in \vec{G}_p} I_k)/|\vec{G}_p|$.
2. Until all tasks are scheduled, repeat the following steps:
   (a) among tasks with no predecessors, select the heaviest-weight task;
   (b) schedule the selected task next;
   (c) remove the scheduled task from $\vec{G}$.

If there are no dependencies, then the task weights are equal to task currents, and the resulting schedule tasks are sequenced in nonincreasing order of their currents. According to Theorem B.2 this is an optimal sequence if the endurance constraint is ignored. If dependencies are present, then at any given scheduling step not all the tasks are ready to execute, but only those whose predecessors have been already scheduled. Selecting a task with the largest current among ready tasks (i.e., $w(p) = I_p$ for each task p) may be a poor strategy. For example, a task with very low current may have a successor with very large current, whose execution will be delayed until its predecessor is scheduled. To avoid such traps, we compute the average current for the entire subgraph induced by a given task in the task graph $\vec{G}$. Thus, if some low-current task p enables execution of high-current tasks, it may have the large enough weight $w(p)$ to be scheduled earlier than another ready task with the current larger than I_p.

In the worst case, the pseudopolynomial initial voltage assignment dominates the complexity of Step I. The dynamic programming algorithm for solving the multiple-choice knapsack problem takes $O(BnK)$ time.

It is clear that if α is sufficiently large (no endurance constraint), and tasks are independent (no dependency constraint), then the charge minimization approach will produce an optimal delay-constrained schedule during Step I, without the need for Steps II and III described next.

6.1.2 *Step II: Battery Failure Repair.* The initial profile is not guaranteed to be failure-free, that is, the battery may not survive execution of some tasks. Procedure *TaskRepair*$(\cdot)$ is called to first check if there is a failing task, and if so, repairs it by voltage down-scaling and/or insertion of idle periods. Note that if $T > B$ after repairing the profile, then procedure *LatencyReduction*$(\cdot)$ is called to perform voltage up-scaling (Step III) to reduce T.

One of the inputs to *TaskRepair*$(\cdot)$ is the deadline D. Note that *ChargeMinimization*$(\cdot)$ calls *TaskRepair*$(\cdot)$ with $\max\{B, B'\}$ as the value for D, where B' denotes the sum of all task durations when each task is assigned the lowest voltage V_0. This is necessary because *MultipleChoiceKnapsack*$(\cdot)$ does not normally leave any delay slack $\delta = B - T$ to be utilized during voltage down-scaling. In other words, we let task voltages be as low as possible in order to recover failures. The task repair procedure is outlined below.

1. To check the endurance constraint given by Eq. (12), compute lifetime L (if the endurance constraint is satisfied, then L will be **NULL**, i.e., the battery survives the profile).

2. If L is **NULL**, then terminate with **SUCCESS**.
3. Otherwise, find the earliest load step u during which the failure occurs and repeat the following steps:
   (a) let P be the subprofile ending with the uth step;
   (b) among all the tasks in P, select task s, for which reduction of its voltage to the next lower level results in the largest decrease of the cost of P without violating the deadline D;
   (c) if s is **NULL**, then exit this loop;
   (d) otherwise, reduce the voltage of s to the next lower level, and compute lifetime L;
   (e) if L is **NULL**, then terminate with **SUCCESS**;
   (f) otherwise, find the earliest load step u during which the failure occurs;
4. Insert idle periods.
5. Terminate with **SUCCESS** or **FAILURE**, depending on the success or failure of repairing by idle period insertion.

If there are no failures detected in Step 1, then the procedure terminates with **SUCCESS**. If a failure is present, we find the earliest failing profile step u and enter the loop of Step 3. Inside this loop, the procedure identifies task s, for which the voltage level decrement results in the lowest subprofile cost, while the deadline D is still met.[7] If s is not **NULL**, then its voltage level is decremented, and the new profile becomes the current solution $\{S_I, S_D, S_t\}$. If this solution is failure-free, then the procedure terminates with **SUCCESS**. Otherwise, the next earliest failure is detected, and Step 3 is repeated. Selection s may be **NULL**, because either (i) the voltage for all the tasks in P is already V_0; or (ii) decrementing the voltage level for any task in P results in the deadline violation.[8] In such cases, the only remaining choice is to perform idle period insertion, performed by *InsertIdlePeriods*$(\cdot)$. Below is an outline of this procedure [Rakhmatov et al. 2002].

1. Until all failing tasks have been considered, repeat the following steps:
   (a) find the earliest failing task q;
   (b) immediately before q, that is, at t_q, insert an idle period of minimum length $\delta \leq B$ such that q no longer fails, that is, the battery lifetime $L \notin [t_q + \delta, t_q + \Delta_q + \delta]$, if possible.
2. Let the new profile with idle periods be the current solution.
3. Until the current solution is not changed, repeat the following steps:
   (a) select the latest unvisited idle period $[t_{\text{start}}, t_{\text{finish}}]$;
   (b) among tasks scheduled after t_{finish}, find task q such that I_q is as low as possible provided that scheduling q at t_{start} will not violate dependencies;

[7] Recall that voltage down-scaling always reduces the cost (see Theorem B.8); therefore, eventually the cost of P will be small enough for the battery to survive the uth step. Another important point to note is as follows. In P no tasks, except the last the one at the uth position, is failing. By Theorem B.9, scaling down the voltage for these tasks will never introduce new failures.
[8] Step 3 is guaranteed to terminate, since eventually s will become **NULL**.

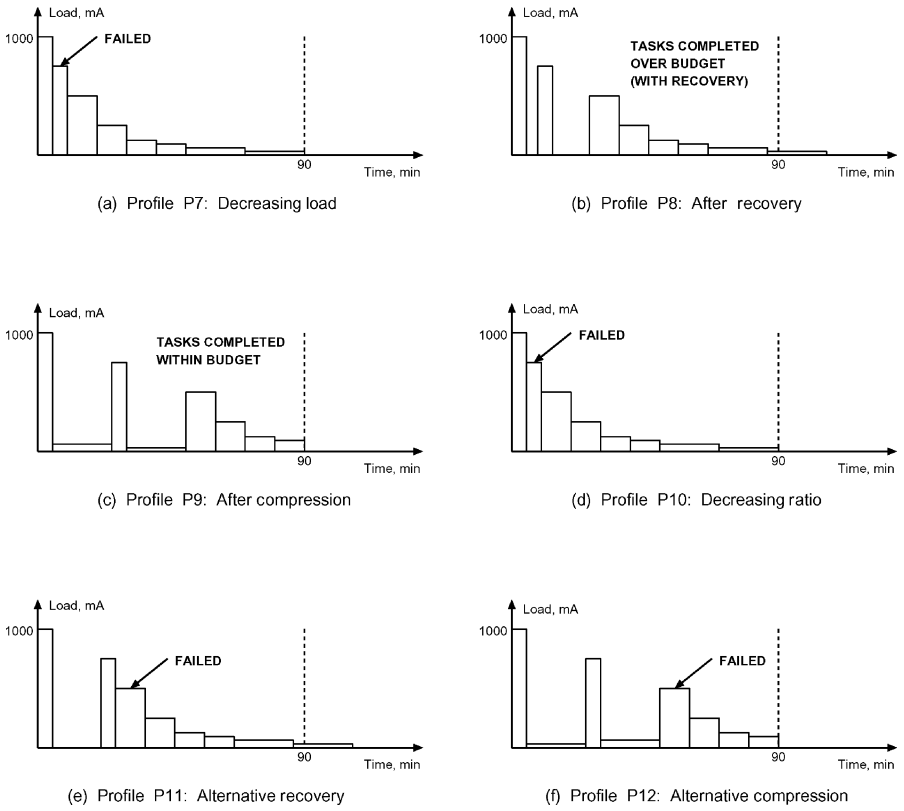


Fig. 11. Example: Scheduling tasks with fixed voltages.

(c) schedule q at t_{start} and eliminate previously inserted idle periods following q;
(d) insert new idle periods of minimum length to repair tasks following q, if necessary;
(e) if the length T of this new profile is reduced, then the new profile becomes the current solution;
(f) if T meets the delay budget, then the new profile is returned as the final solution with **SUCCESS**;
(g) if the length of the new profile is not less than that of the previous profile, then the current solution is not changed.

4. If the profile has not been repaired, return **FAILURE**.

The first step of *InsertIdlePeriods*$(\cdot)$ generates an optimal failure recovering solution for a given load profile, according to Theorems B.4. In the subsequent steps, the procedure attempts to reduce the total idle time by placing lighter tasks (i.e., tasks with lower current consumption) inside later idle periods, subject to precedence constraints. By filling later idle periods with lighter tasks, we aim at changing a minimal portion of the profile with a minimal cost penalty.

Figure 11 illustrates idle period insertion and other issues related to task scheduling with fixed voltages. In our example, let the voltage for tasks $T1$–$T4$

be fixed at V_1, and let the voltage for tasks $T5$–$T8$ be fixed at V_0. Figure 11(a) shows profile P7, generated by *TaskSequence*(·). Note that P7 forms a nonincreasing sequence of loads. The task ordering is ($T1, T2, T3, T4, T5, T6, T7, T8$). However, a battery failure occurs during execution of task $T2$. An idle period of 3 min is required to repair $T2$. The next failure occurs during execution of task $T3$. To repair $T3$ we need a 13-min idle period. Thus, Step 1 of *InsertIdlePeriods*(·) generates P8, shown in Figure 11(b), from P7. Note that the total delay penalty is 16 min, and the profile length must be reduced from 106 min to 90 min, without introducing any failures. This is successfully accomplished by Step 3 of *InsertIdlePeriods*(·). The final solution P9 is presented in Figure 11(c). The task ordering is ($T1, T7, T2, T8, T3, T4, T5, T6$).

Next, consider profile P10 in Figure 11(d), which was generated by scheduling loads in nonincreasing order of the ratios $I_k[1 - e^{-\beta^2\Delta_k}]/\Delta_k$ (recall a weighted completion time problem, discussed in Section 4). Note that P10 is identical to P7, that is, for this particular case, achieving minimum weighted completion time yielded the minimum value of the cost function σ.[9] Figure 11(e) shows profile P11, which has an idle period of 16 min. Thus, both P8 and P11 have the same profile length. However, the battery cannot survive P11 unless the length of the idle period is increased; whereas, P8 is already failure-free. Profile P12, shown in Figure 11(f), is an alternative to P9. The only difference between P9 and P12 is that tasks $T7$ and $T8$ are swapped (dependencies are ignored for P12). Note that P12 is failing, while P9 satisfies the endurance constraint.

Finally, note that idle period insertion is performed only after task voltages are scaled down as much as possible. Such an approach is suggested by Theorem B.11: voltage down-scaling is always more effective than idle period insertion with the same delay penalty.

During task repair the greatest amount of work is done during Steps 3 (voltage down-scaling) and 4 (idle period insertion). The complexities of these steps are $O(Kn^2X)$ and $O(n^3Y)$, respectively, where X is the complexity of lifetime computation, and Y is the complexity of computing the lengths of $O(n)$ idle periods. Thus, the worst-case complexity of Step II is $O(Kn^2X + n^3Y)$.

6.1.3 *Step III: Profile Length Reduction.* After successful completion of Step II, if the profile length T exceeds B, then the voltage and the clock rate for some tasks need to be increased. For this purpose, we use *LatencyReduction*(·). Note that *ChargeMinimization*(·) passes the delay budget B to *Latency Reduction*(·) as the deadline D to be met. The main steps of the latency reduction procedure are described below.

1. If $T \leq D$, then terminate with **SUCCESS**.
2. Otherwise, repeat the following steps:
   (a) among all the tasks, select task s, for which an increase of its voltage to the next higher level results in the smallest increase of the profile cost without violating the endurance constraint (i.e., L must be **NULL**);

[9]By Theorem B.2, profile P7 has the lowest cost compared to any other sequence of the same length.

(b) if s is **NULL**, then terminate with **FAILURE**;
(c) otherwise, increase the voltage of s to the next higher level;
(d) if $T \leq D$, then terminate with **SUCCESS**.

First, we check whether the profile length T exceeds the deadline. If $T \leq D$, then the procedure terminates with **SUCCESS**. Otherwise, the loop of Step 2 is entered. In this loop, the procedure selects task s such that incrementing its voltage level results in the lowest profile cost, provided that there are no failures introduced. The voltage of the selected task is scaled one level up, and the resulting profile becomes the current solution $\{S_I, S_\Delta, S_t\}$. If the profile length T meets D, then the procedure terminates with **SUCCESS**. Otherwise, Step 2 is repeated. Note that Step 2 is guaranteed to terminate, since eventually s will become **NULL**, that is, either the voltages of all the tasks are V_{K-1} (the highest level), or any further voltage up-scaling results in a profile failure. If s is **NULL**, the procedure terminates with **FAILURE** since the deadline has not been met.

Note that the complexity of Step III is $O(Kn^2X)$, which does not exceed the complexity of Step II. Therefore, the overall complexity of the charge minimization approach is $O(BnK + Kn^2X + n^3Y)$.

6.1.4 *Slack Utilization.* The approach based on charge minimization can be applied not only to initial voltage assignment, but also to delay slack distribution for a given set of scheduled tasks with a given voltage assignment. Let $\delta = B - T$ denote the available delay slack in some failure-free load profile. According to Theorem B.8, voltage down-scaling always reduces the profile cost. However, a decrease in task voltages results in an increase in task durations. Consequently, the profile length T increases and δ decreases. The objective of the slack utilization process is to distribute δ among tasks, so that the profile cost is reduced as much as possible. For example, let $\delta_k \geq 0$ be a portion of δ allocated to task k, that is, $\delta = \sum_{k=0}^{n-1} \delta_k$. Then, the voltage is scaled down for k so that $\delta_k \geq \Delta_k - \hat{\Delta}_k$, where Δ_k and $\hat{\Delta}_k$ are durations of k before and after voltage down-scaling, respectively.

Similar to the case of initial voltage assignment, slack utilization based on charge minimization is formulated as the multiple-choice 0-1 knapsack problem, with the following minor modification. For a given task k, let x denote its voltage level, that is, $\{I_k, \Delta_k\} = LookUp(k, V_x, \phi_x)$. Since the voltage may not be scaled up, we do not consider the currents and the delays corresponding to a voltage level higher than x for task k in question. In other words, for each given task k, we set $I_{ik} = I_{xk}$ and $\Delta_{ik} = \Delta_{xk}$, for all $i \in \{x + 1, \ldots, K - 1\}$. Thus, the voltage-current and voltage-duration task tables for slack utilization are slightly different from those used for initial voltage assignment. The corresponding slack utilization procedure is called *SlackUtilizationMinCharge*$(\cdot)$. It uses dynamic programming to solve the knapsack problem with the modified tables for task currents and durations. Therefore, slack utilization based on charge minimization takes $O(BnK)$ time.

An alternative slack distribution procedure, called *AlterSlackUtilization*$(\cdot)$, executes the following steps. First, among all the tasks it selects task s, for which decrementing its voltage level yields the lowest profile cost without violating the delay budget B. Second, after the voltage of the selected task is

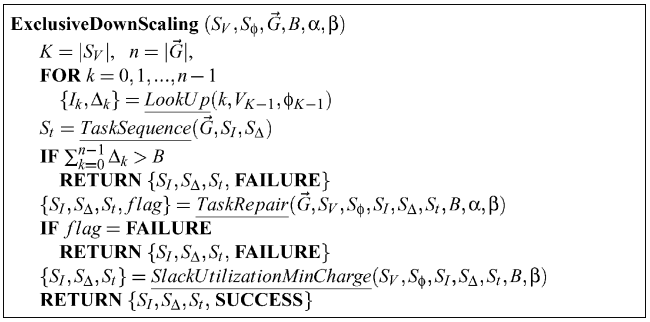


Fig. 12. Exclusive voltage down-scaling.

reduced, the resulting profile becomes the current solution $\{S_I, S_\Delta, S_t\}$. Then, the first and second steps are repeated. This process terminates once s is **NULL**, that is, when either (i) the voltages of all the tasks are V_0; or (ii) any further voltage down-scaling increases the profile length beyond B. The complexity of the alternative slack utilization procedure is $O(Kn^2)$.

Note that neither *SlackUtilizationMinCharge*$(\cdot)$ nor *AlterSlackUtilization*$(\cdot)$ introduces failures, in accordance with Theorem B.9.

6.2 Voltage Down-Scaling Based on Highest-Power Initial Solution

Figure 12 shows *ExclusiveDownScaling*$(\cdot)$, a method that uses voltage down-scaling exclusively to generate a low-cost load profile. Initially, all tasks are assigned to the maximum voltage V_{K-1}, so that the profile duration is minimized. For each task k, the current becomes $I_{(K-1)k}$ and the duration becomes $\Delta_{(K-1)k}$. Then, *TaskSequence*$(\cdot)$ is called to generate the initial set S_t. The length T of the initial profile is equal to $\sum_{k=0}^{n-1} \Delta_{(K-1)k}$. If $T > B$, then no solution will satisfy the delay budget (tasks are already of the shortest durations), and the procedure returns **FAILURE**. Otherwise, *TaskRepair*$(\cdot)$ is called to repair failing tasks, if any. If the profile is failure-free and within the delay budget B (i.e., $flag =$ **SUCCESS**), *SlackUtilizationMinCharge*$(\cdot)$ is called to improve the solution cost; otherwise, the procedure terminates with **FAILURE**. We may call *AlterSlackUtilization*$(\cdot)$ instead of *SlackUtilizationMinCharge*$(\cdot)$. To differentiate between these two possibilities, we name the procedure using *AlterSlackUtilization*$(\cdot)$ as *ExclusiveDownScaling*2$(\cdot)$.

The complexity of task repair and slack utilization determines the complexity of the voltage down-scaling approach. *ExclusiveDownScaling*$(\cdot)$ takes $O(BnK + Kn^2X + n^3Y)$ time, and *ExclusiveDownScaling*2$(\cdot)$ takes $O(Kn^2X + n^3Y)$ time.

Note that, in this approach, we start with a solution that satisfies the delay constraints, but may violate the endurance constraint. We can also start with the solution that satisfies the endurance constraint, but may violate the delay constraint. This alternative is explored next.

6.3 Voltage Up-Scaling Based on Lowest-Power Initial Solution

The last proposed method, *ExclusiveUpScaling*$(\cdot)$, for task sequencing with exclusive voltage up-scaling is shown in Figure 13. To obtain the initial sets S_I and

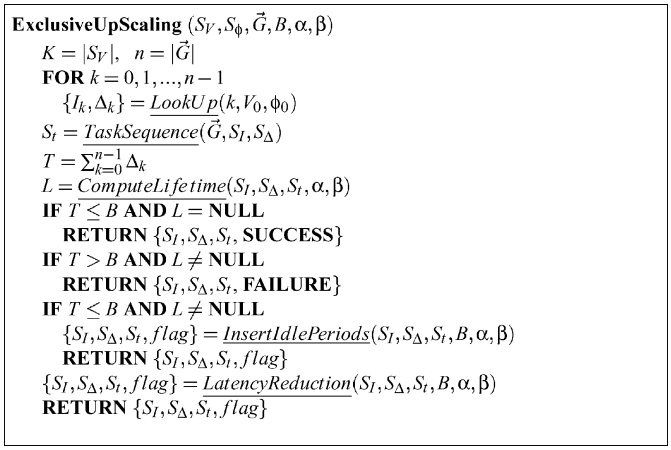


Fig. 13. Exclusive voltage up-scaling.

S_Δ, we assign all tasks to the lowest voltage V_0, so that the energy consumption is minimized as much as possible. For each task k, the current becomes I_{0k} and the duration becomes Δ_{0k}. The initial set S_t is generated by *TaskSequence*$(\cdot)$. The length T of the initial profile is equal to $\sum_{k=0}^{n-1} \Delta_{0k}$. Let L be the battery lifetime, computed by *ComputeLifetime*$(\cdot)$ for the load profile in question. If $T \leq B$ and $L =$ **NULL**, then the procedure terminates with **SUCCESS**. On the other hand, if $T > B$ and $L \neq$ **NULL**, then the procedure aborts any attempts to generate a valid profile. Thus, there are two cases of interest: (a) $T \leq B$ and $L \neq$ **NULL**, and (b) $T > B$ and $L =$ **NULL**.

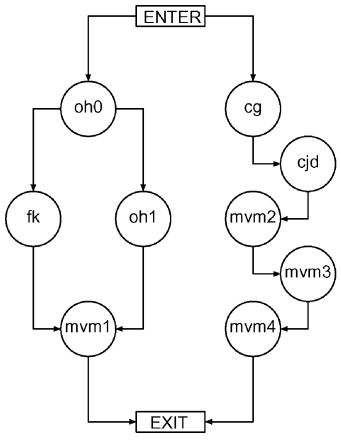
In case (a), the delay budget is met, but the battery does not survive the profile. Since the voltage level is the lowest, only idle period insertion is applicable, and *InsertIdlePeriods*$(\cdot)$ is called. In case (b), the battery survives the profile, but the delay budget is exceeded. Therefore, some tasks must be assigned to a higher voltage (resulting in greater currents but shorter durations) to satisfy the delay constraint. This is accomplished by calling *LatencyReduction*$(\cdot)$.

The running time of the voltage up-scaling approach is dominated by idle period insertion and latency reduction. Note that only one of these two procedures is called before voltage up-scaling terminates. The complexity of this approach is $O(\max\{Kn^2X, n^3Y\})$.

6.4 Simplified Task Repair, Latency Reduction, and Slack Utilization

According to Theorem B.10, given a set of identical tasks which are candidates for voltage down-scaling, the best result is achieved when the available delay slack is utilized by the latest task. This observation suggests certain heuristic simplifications for *TaskRepair*$(\cdot)$, *LatencyReduction*$(\cdot)$, and *AlterSlackUtilization*$(\cdot)$. Here, we provide a brief outline for the sake of completeness [Chowdhury and Chakrabarti 2002; Rakhmatov et al. 2002].

Given the earliest failing step u, the simplified task repair procedure considers tasks one by one in the reverse order, that is, $u, u-1, u-2, \ldots, 0$. For each task x under consideration, its voltage level is decremented until either (i) u no



Voltage / Task	V_0		V_1		V_2		V_3	
	I, mA	Δ, min	I, mA	Δ, min	I, mA	Δ, min	I, mA	Δ, min
cg	60	22.0	117	17.6	278	13.2	938	8.8
oh0	50	3.1	98	2.5	231	1.9	781	1.2
oh1	50	20.2	98	16.2	231	12.1	781	8.1
fk	42	9.0	82	7.2	194	5.4	656	3.6
cjd	50	16.3	98	13.0	231	9.8	781	6.5
mvm1	34	8.8	66	7.0	157	5.3	531	3.5
mvm2	34	8.8	66	7.0	157	5.3	531	3.5
mvm3	34	8.8	66	7.0	157	5.3	531	3.5
mvm4	34	8.8	66	7.0	157	5.3	531	3.5

Fig. 14. Task specifications.

longer fails; or (ii) the voltage level V_0 is reached; or (iii) the delay constraint is violated. In case (i), the new earliest failing task is detected, if any, and the voltage down-scaling process is repeated. In cases (ii) and (iii), one abandons x and starts reducing the voltage of the preceding task.

The simplified latency reduction procedure examines the earliest unvisited task, x, and scales its voltage up as much as possible, provided that the profile remains failure-free. Upon assigning new voltage to x, the next task following x is considered for voltage upscaling. Intuitively, it is an attempt to generate a nonincreasing load sequence (see Theorem B.2), without causing battery failures before the last task is completed.

The simplified slack utilization procedure considers tasks one by one, from the end to the beginning of the sequence. The voltage for each considered task x in the sequence, is scaled down as much as possible, provided that the profile length remains within the delay budget. This process is terminated once either (i) all tasks are at the lowest voltage level, or (ii) further voltage down-scaling for any task results in a violation of the delay constraint.

7. EVALUATION RESULTS

To illustrate the proposed methods for energy-aware task scheduling with voltage/clock scaling, we use an example of a robot arm controller from Mooney III and De Micheli [2000]. The task graph of interest is shown in Figure 14, which also specifies task currents and durations for four different voltages (V_0, V_1, V_2, V_3). Task specifications are somewhat artificial, but consistent with Mooney III and De Micheli [2000], reporting such data as task mapping (software or hardware); task execution delay (the number of clock cycles); silicon area of hardware-mapped tasks; code size of software-mapped tasks; and so on. In particular, for the voltage V_0, we let (i) task durations be proportional to the worst-case number of clock cycles; (ii) currents of software-mapped tasks be the same and equal to 50 mA; and (iii) currents of hardware-mapped tasks be proportional to the area. For the other voltages, task durations are

Table III. Task Ordering with Task Voltages

Profile	Task Sequence	Voltage Assignment
P1	$(cg, cjd, mvm2, mvm3, mvm4, oh0, fk, oh1, mvm1)$	$(V_2, V_2, V_3, V_3, V_3, V_2, V_3, V_2, V_3)$
P2	$(cg, cjd, mvm2, mvm3, oh0, fk, oh1, mvm1, mvm4)$	$(V_1, V_2, V_2, V_2, V_0, V_2, V_1, V_2, V_1)$
P3	$(oh0, cg, cjd, mvm2, mvm3, mvm4, oh1, fk, mvm1)$	$(V_1, V_0, V_1, V_1, V_1, V_1, V_0, V_0, V_1)$
P4	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_3, V_3, V_3, V_3, V_3, V_3, V_3, V_3, V_3)$
P5	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_3, V_2, V_3, V_2, V_3, V_3, V_3, V_2, V_2)$
P6	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_3, V_2, V_3, V_2, V_2, V_3, V_3, V_2, V_2)$
P7	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_3, V_2, V_3, V_2, V_3, V_3, V_3, V_2, V_1)$
P8	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_1, V_2, V_0, V_1, V_2, V_2, V_2, V_2, V_1)$
P9	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_1, V_2, V_3, V_1, V_2, V_2, V_2, V_2, V_0)$
P10	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_0, V_1, V_1, V_0, V_0, V_1, V_1, V_1, V_1)$
P11	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_0, V_1, V_1, V_0, V_1, V_1, V_1, V_1, V_0)$
P12	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_0, V_0, V_0, V_0, V_0, V_0, V_0, V_0, V_0)$
P13	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_2, V_3, V_3, V_2, V_3, V_3, V_3, V_2, V_2)$
P14	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_1, V_2, V_3, V_1, V_2, V_2, V_2, V_2, V_0)$
P15	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_0, V_1, V_2, V_0, V_1, V_1, V_1, V_1, V_0)$
P16	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_3, V_2, V_2, V_2, V_1, V_0, V_0, V_0, V_0)$
P17	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_3, V_0, V_0, V_0, V_0, V_0, V_0, V_0, V_0)$
P18	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_3, V_2, V_3, V_2, V_3, V_3, V_3, V_2, V_1)$
P19	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_3, V_2, V_3, V_2, V_1, V_0, V_0, V_0, V_0)$
P20	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_3, V_0, V_0, V_0, V_0, V_0, V_0, V_0, V_0)$

made inversely proportional to the scaling factor with respect to V_0, and task currents are made directly proportional to the cube of the scaling factor with respect to V_0. The scaling factors with respect to V_0 for voltages (V_0, V_1, V_2, V_3) are $(\frac{1}{1.0}, \frac{1}{0.8}, \frac{1}{0.6}, \frac{1}{0.4})$. Note that task durations are expressed in terms of fractions of a minute.[10] Such a coarse-grain timing scale is chosen for demonstration purposes only, for example, for exposing battery failures, lifetime sensitivity to task ordering, and so on. Note that the material presented in this paper is applicable to *any* timing scale of user's choice. Later in this section, we consider tasks with fine-grain timing characteristics.

7.1 Tasks with Coarse-Grain Timing Characteristics

Given task specifications and dependencies, as displayed by Figure 14, we generated twenty load profiles for three different delay budgets: 55.0, 75.0, and 95.0 min. Table III presents task ordering and task voltage assignment. Table IV presents the profile length T, the delay budget B, and the profile cost σ. As an alternative to σ, one can use a direct measure of the battery lifetime for a given profile. To cause a battery failure, one needs to apply some load starting at the end of the profile in question. Here, we use a constant-current load of 500 mA, applied starting at time T until the battery becomes discharged. Lifetime estimations based on our battery model are reported in the fourth column of Table IV. Also, in the third column of Table IV, we show the results produced by DUALFOIL—a microscopic-scale simulator of a lithium-ion

[10]In reality, task durations are on the order of fractions of a millisecond [Mooney III and De Micheli 2000].

Table IV. Profile Quality with Simulation Results

Profile	Length T (min)	Budget B (min)	Cost σ (mA-min)	Simulation: Lifetime $L_{500\ \mathrm{mA}}$ (min)	Prediction: Lifetime $L_{500\ \mathrm{mA}}$ (min)	Error (%)
P1	54.6	55.0	35739	62.3	60.7	2.6
P2	75.0	75.0	13885	100.8	97.9	2.9
P3	94.7	95.0	8517	127.5	124.3	2.5
P4	42.2	—	53841	14.7	15.2	3.4
P5	53.1	55.0/75.0/95.0	32062	57.5	57.7	0.3
P6	54.9	55.0	29885	62.4	61.5	1.4
P7	54.8	55.0	28984	60.8	60.8	0.0
P8	75.0	75.0	14251	100.7	97.8	2.9
P9	74.9	75.0	13862	99.7	96.9	2.8
P10	94.7	95.0	8766	127.3	124.3	2.4
P11	94.7	95.0	8004	127.4	124.3	2.4
P12	105.8	—	6312	140.4	137.6	2.0
P13	54.2	55.0	30434	60.8	60.6	0.3
P14	74.9	75.0	13862	99.7	96.9	2.8
P15	94.1	95.0	8205	126.5	123.4	2.4
P16	75.0	75.0	17259	93.6	90.7	3.1
P17	92.6	95.0	13268	116.7	113.4	2.8
P18	54.8	55.0	28984	60.8	60.8	0.0
P19	74.3	75.0	17781	92.1	89.4	2.9
P20	92.6	95.0	13268	116.7	113.4	2.8

cell [Arora et al. 2000]. For the DUALFOIL battery, the model parameters are $\alpha = 40375$ and $\beta = 0.273$. These parameters were used for generating all the profiles; that is, the results in this section are specific to the DUALFOIL battery. Note that our predictions closely match simulation data, with the maximum error of approximately 3%.

Profiles P1, P2, and P3 are constructed by *ChargeMinimization*(·) for delay budgets 55.0, 75.0, and 95.0 min, respectively. After *MultipleChoiceKnapsack*(·) assigned task voltages and *TaskSequence*(·) generated task sequences, no task repairs were necessary. As the delay budget grows, energy efficiency of the profiles increases. Note that P3 is four times less costly than P1. Consequently, the simulated residual lifetime ($L_{500\ \mathrm{mA}} - T$) of 32.8 min for P3 is much greater than that of 7.7 min for P1.

Profiles P4–P11 are due to *ExclusiveDownScaling*(·) and *ExclusiveDown Scaling*2(·).[11] Profile P4 is the highest-power initial solution (task voltages are at the highest level V_3), which is failing after the first 15 min. To repair P4, *TaskRepair*(·) constructs profile P5, where the voltages for tasks *cjd*, *oh*1, *mvm*1, and *mvm*4 are scaled down to V_2. The length of P5 is 53.1 min, which is within the delay budgets under consideration. To utilize the available delay slack ($B - T$), one can run either *SlackUtilizationMinCharge*(·) or *AlterSlackUtilization*(·). For the delay budget B of 55.0 min, the delay slack is 1.9 min. *SlackUtilizationMinCharge*(·) utilizes this slack by down-scaling the voltage for task *fk* from V_3 to V_2 (profile P6); whereas, *AlterSlackUtilization*(·)

[11]Recall that *ExclusiveDownScaling*(·) uses *SlackUtilizationMinCharge*(·) for delay slack utilization, while *ExclusiveDownScaling*2(·) uses *AlterSlackUtilization*(·).

down-scales the voltage of task $mvm1$ from V_2 to V_1 (profile P7). Note that the cost of P7 is smaller than that of P6. For B of 75.0 min, the available slack is 21.9 min, which allows for aggressive voltage scaling. *SlackUtilizationMinCharge*(·) and *AlterSlackUtilization*(·) generate profiles P8 and P9, respectively. Note that the cost of P9 is smaller than that of P8. On comparing P6 and P8 as well as P7 and P9, one can see that profile costs are reduced by approximately 2×, as the delay budget increases from 55.0 min to 75.0 min. Further energy utilization improvements are achieved for the delay slack of 41.9 min (B is 95.0 min): profiles P10 and P11 are generated by *SlackUtilizationMinCharge*(·) and *AlterSlackUtilization*(·), respectively. The cost of P11 is the lowest among all the profiles P1–P20. Comparing P1–P3 and P6–P11, one can see that *ExclusiveDownScaling*2(·) outperforms both *ExclusiveDownScaling*(·) and *ChargeMinimization*(·). However, note that differences are insignificant in terms of residual lifetimes for the profiles with the matching delay budgets.

Profile P12 is the lowest-power initial solution, due to *ExclusiveUpScaling*(·). Its length is 105.8 min, and task durations are to be decreased by *LatencyReduction*(·) through voltage up-scaling in order to satisfy the delay constraints. Profiles P13, P14, and P15 are obtained from P12 for the delay budget of 55.0, 75.0, and 95.0 min, respectively. Note that P14 is identical to P9, that is, *ExclusiveDownScaling*2(·) and *ExclusiveUpScaling*(·) arrived at the same solution. As the P13–P15 costs and residual lifetimes suggest, the performance of *ExclusiveUpScaling*(·) is as good as that of the knapsack-based and the voltage down-scaling approaches. However, among the proposed methods, the complexity of *ExclusiveUpScaling*(·) is the lowest since it does not involve *TaskRepair*(·). Our overall recommendation favors the use of the voltage up-scaling approach for solving the energy-aware task scheduling problem.

Finally, the last five profiles P16–P20 are constructed using simplified versions[12] of *TaskRepair*(·), *LatencyReduction*(·), and *AlterSlackUtilization*(·). Profiles P16, P17, P18, P19, and P20 are alternatives to P9, P11, P13, P14, and P15, respectively.[13] The only case when a simplified version produced a better result (i.e., it accidentally has managed to escape a local optimum) is P18 compared to P13; however, the cost improvement is only 5% (28 984 versus 30 434), which does not yield noticeable residual lifetime improvements. On the other hand, a simplified version may perform very poorly. Comparing P11 and P17, one can see the profile cost has increased by 66%, and more than 10 min of the residual lifetime has been lost.

Recall that original profiles P9 and P14 are identical, and note that their respective alternatives P17 and P20 are identical as well. Also, P18 is the same as P7, or in other words, *ExclusiveDownScaling*2(·) and *ExclusiveUpScaling*(·) with simplified latency reduction produce identical solutions.

[12]Recall that simplifications are based on the assumption that all tasks are identical.

[13]For the rest of the original profiles, no change is introduced due to using simplified task repair, latency reduction, and slack utilization.

Table V. Ordering and Voltages of Microtasks

1st Period	Task Sequence	Voltage Assignment
P21	$(cg, cjd, mvm2, mvm3, mvm4, oh0, fk, oh1, mvm1)$	$(V_2, V_2, V_3, V_3, V_3, V_2, V_3, V_2, V_3)$
P22	$(cg, cjd, mvm2, mvm3, oh0, fk, oh1, mvm1, mvm4)$	$(V_1, V_2, V_2, V_2, V_0, V_2, V_1, V_2, V_1)$
P23	$(oh0, cg, cjd, mvm2, mvm3, mvm4, oh1, fk, mvm1)$	$(V_1, V_0, V_1, V_1, V_1, V_1, V_0, V_0, V_1)$
P24	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_2, V_2, V_2, V_2, V_3, V_3, V_3, V_3, V_3)$
P25	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_2, V_2, V_2, V_2, V_3, V_3, V_3, V_3, V_3)$
P26	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_1, V_2, V_0, V_1, V_2, V_2, V_2, V_2, V_1)$
P27	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_1, V_1, V_3, V_1, V_2, V_2, V_2, V_2, V_2)$
P28	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_0, V_1, V_1, V_0, V_0, V_1, V_1, V_1, V_1)$
P29	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_0, V_0, V_1, V_0, V_1, V_2, V_1, V_1, V_1)$
P30	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_2, V_2, V_3, V_2, V_3, V_3, V_3, V_3, V_3)$
P31	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_1, V_1, V_3, V_1, V_2, V_2, V_2, V_2, V_2)$
P32	$(cg, cjd, oh0, oh1, fk, mvm2, mvm3, mvm4, mvm1)$	$(V_0, V_0, V_2, V_0, V_1, V_2, V_1, V_1, V_1)$

Table VI. Cost of the 1st Period and Predicted Lifetimes after 100 000 Periods

1st Period	Length T ($\times 10^{-5}$ min)	Budget B ($\times 10^{-5}$ min)	Cost σ ($\times 10^{-5}$ mA-min)	After 10^5 Periods: Lifetime $L_{500\ \mathrm{mA}}$, (min)
P21	54.6	55.0	39 794	61.3
P22	75.0	75.0	21 127	97.6
P23	94.7	95.0	12 740	124.3
P24	54.6	55.0	39 798	61.3
P25	54.6	55.0	39 798	61.3
P26	75.0	75.0	21 128	97.6
P27	74.6	75.0	21 471	96.9
P28	94.7	95.0	12 741	124.3
P29	94.5	95.0	13 001	123.9
P30	53.9	55.0	40 844	59.5
P31	74.6	75.0	21 471	96.9
P32	93.9	95.0	13 408	122.9

7.2 Tasks with Fine-Grain Timing Characteristics

To demonstrate the impact of our methods applied to scheduling tasks with durations on the order of a millisecond, we use the same task specifications as in Figure 14, but divide the timing scale by the factor of 100 000. We term these nine fine-grain tasks as *microtasks*. For example, the duration of microtask *fk* at V_0 becomes 9.0×10^{-5} min, and the delay budgets of interests are 55.0×10^{-5}, 75.0×10^{-5}, and 95.0×10^{-5} min. Note that microtask currents are not changed.

We apply the proposed algorithms to order microtasks and assign voltages. Then, a generated profile is repeated 100 000 times to form a periodic load. To determine how much residual charge can be delivered after 100 000 periods, we assume that the battery is discharged at the constant rate of 500 mA. For the three different constraints on a period duration (55.0×10^{-5}, 75.0×10^{-5}, and 95.0×10^{-5} min), we tackle the task scheduling problem with voltage scaling using three approaches described in Section 6. The corresponding profiles characteristics are described in Tables V and VI.

Profiles P21, P22, and P23 are due to *ChargeMinimization*$(\cdot)$ for the delay budgets of 55.0×10^{-5}, 75.0×10^{-5}, and 95.0×10^{-5} min, respectively. Note that

the cost of the first period of P21 is more than three times higher than that of P23. After 100000 periods, for P21 and P23, the battery is predicted to last (under 500 mA) for 6.7 and 29.6 min, respectively.

For the delay budget of 55.0×10^{-5} min, procedures *ExclusiveDownScaling*(·) and *ExclusiveDownScaling*2(·) have generated identical first periods P24 and P25. Note that microtasks in P21 and P24–P25 have the same voltage assignment but different ordering. One can observe practically no difference in the costs and the predicted lifetimes after 100 000 periods are applied. Since the period duration is very small and a single period is repeated many times, the impact of task ordering is negligible. The same observation holds for (i) P22 and P26 (the latter has been generated by *ExclusiveDownScaling*(·) for the delay budget of 75.0×10^{-5} min), as well as for (ii) P23 and P28 (the latter has been generated by *ExclusiveDownScaling*(·) for the delay budget of 95.0×10^{-5} min). For the delay budgets of 75.0×10^{-5} and 95.0×10^{-5} min, *ExclusiveDownScaling*2(·) constructed P27 and P29 of comparable quality.

Finally, P30, P31, and P32 for the budgets of 55.0×10^{-5}, 75.0×10^{-5}, and 95.0×10^{-5} min, respectively, have been constructed by *ExclusiveUpScaling*(·). When compared to *ChargeMinimization*(·) and *ExclusiveDownScaling*(·), its performance is the worst in terms of the profile quality. P31 is identical to P27, that is, the same final solution has been obtained (i) starting from the highest-power initial solution, and (ii) starting from the lowest-power initial solution.

Note that we locally apply our algorithms to the first period only. The results indicate that for a greater effect on a battery, periodic tasks should be treated globally (e.g., the voltage of the same task in different periods may not be the same).

8. CONCLUSION

Energy-autonomous embedded systems must have an attached finite-capacity energy source—a battery—that must be relatively small and light. Consequently, the system energy budget is severely limited, and efficient energy utilization becomes one of the key problems in the context of battery-powered embedded computing. In this paper, we addressed the battery-related issues arising in the process of energy management of such systems.

First, we introduced an analytical battery model, which can be used for the battery lifetime estimation. Measurements and simulation results have demonstrated high accuracy and robustness of the proposed model. Using this model, we defined a formal battery-aware cost function. This cost function generalizes the traditional minimization metric—the energy consumption of the system. We have proved several important mathematical properties of the cost function in the formulation of the problem of battery-aware task scheduling with voltage scaling in a single-processor environment. Based on these properties, we have designed several algorithms for task ordering and voltage assignment, including optimal idle period insertion to exercise charge recovery. We have demonstrated the utility of the proposed methods on the examples of tasks with coarse-grain and fine-grain timing characteristics.

We presented the first effort toward a formal treatment of battery-aware task scheduling and voltage scaling, based on an accurate analytical model of the battery behavior. This work needs to be extended in three ways: (i) modeling of the discharge–charge cycling effect and the temperature variation impact on battery capacity; (ii) static aperiodic and periodic task scheduling with voltage scaling for multiprocessor systems; and (iii) dynamic lifetime management through task scheduling with voltage scaling.

APPENDIX A

In this appendix we provide details on derivation of the battery model. This material is given for review purposes only. We are given the following system of two partial differential equations, two boundary conditions, and one initial condition:

$$-J(x,t) = D\frac{\partial C(x,t)}{\partial x}, \qquad \frac{\partial C(x,t)}{\partial t} = D\frac{\partial^2 C(x,t)}{\partial x^2}, \tag{19}$$

$$-J(0,t) = \frac{i(t)}{\nu FA}, \qquad J(w,t) = 0, \qquad C(x,0) = C^*, \ \forall x. \tag{20}$$

After applying the Laplace transformation $C(x,t) \to \bar{C}(x,s)$, we obtain

$$\bar{C}(x,s) = \frac{C^*}{s} + P\, e^{-x\sqrt{\frac{s}{D}}} + Q\, e^{x\sqrt{\frac{s}{D}}}, \tag{21}$$

$$\frac{d\bar{C}(x,s)}{dx} = -\sqrt{\frac{s}{D}}\left(P\, e^{-x\sqrt{\frac{s}{D}}} - Q\, e^{x\sqrt{\frac{s}{D}}}\right). \tag{22}$$

We are only interested in the concentration at the electrode surface ($x = 0$). The Laplace transformation $i(t) \to \bar{i}(s)$ and application of the boundary conditions for $x = 0$ and $x = w$ yield the following system of equations:

$$\bar{C}(0,s) = \frac{C^*}{s} + P + Q, \tag{23}$$

$$\frac{\bar{i}(s)}{\nu FAD} = -\sqrt{\frac{s}{D}}(P - Q), \tag{24}$$

$$0 = -\sqrt{\frac{s}{D}}\left(P\, e^{-w\sqrt{\frac{s}{D}}} - Q\, e^{w\sqrt{\frac{s}{D}}}\right). \tag{25}$$

The solution of this system is as follows:

$$\bar{C}(0,s) = \frac{C^*}{s} - \frac{\bar{i}(s)}{\nu FAD}\frac{\coth\left(w\sqrt{\frac{s}{D}}\right)}{\sqrt{\frac{s}{D}}}. \tag{26}$$

We utilize the property that multiplication in the s-domain corresponds to convolution in the time domain; after performing the inverse Laplace transformation of (26), we obtain [Roberts and Kaufman 1966]:

$$C(0,t) = C^* - \frac{i(t)}{\nu FAD} * \sqrt{\frac{D}{\pi t}}\sum_{m=-\infty}^{\infty} e^{-\frac{w^2m^2}{Dt}}, \tag{27}$$

$$1 - \frac{C(0,t)}{C^*} = \frac{1}{\nu FA\sqrt{\pi D}C^*} \int_0^t \frac{i(\tau)}{\sqrt{t-\tau}} \sum_{m=-\infty}^{\infty} e^{-\frac{w^2m^2}{D(t-\tau)}}\, d\tau. \tag{28}$$

Next, we use the following identity from the theory of theta functions [Bellman 1961]:

$$\sum_{m=-\infty}^{\infty} e^{-ym^2} = \sqrt{\frac{\pi}{y}} \sum_{m=-\infty}^{\infty} e^{-\frac{\pi^2m^2}{y}}, \ \mathrm{Re}(y) > 0. \tag{29}$$

In (29), let $y = \frac{w^2}{D(t-\tau)} > 0$. Then,

$$1 - \frac{C(0,t)}{C^*} = \frac{1}{\nu FAwC^*} \int_0^t i(\tau) \left[1 + 2\sum_{m=1}^{\infty} e^{-\frac{\pi^2D(t-\tau)m^2}{w^2}}\right] d\tau. \tag{30}$$

For $\tau \in [0,t]$, the infinite exponential series is uniformly convergent[14], and we can integrate the series term by term. Then,

$$1 - \frac{C(0,t)}{C^*} = \frac{1}{\nu FAwC^*} \left[\int_0^t i(\tau)\, d\tau + 2\sum_{m=1}^{\infty} \int_0^t i(\tau)\, e^{-\frac{\pi^2D(t-\tau)m^2}{w^2}}\, d\tau\right]. \tag{31}$$

APPENDIX B

B.1. Properties with Respect to Sequencing

LEMMA B.1. *For $0 \le \Delta \le t + \Delta \le T$, function $F(T, t, t+\Delta, \beta)$ is*

(a) *monotonically increasing in t;*
(b) *monotonically decreasing in T; and*
(c) *remains the same if t and T are changed by the same amount.*[15]

---

[14]Note that $\tau \in [0,t] \Rightarrow \frac{\pi^2D(t-\tau)}{w^2} > 0 \Rightarrow e^{-\frac{\pi^2D(t-\tau)m^2}{w^2}} < 1$ for all $m \ge 1$. Since $| e^{-\frac{\pi^2D(t-\tau)(n+m)^2}{w^2}} - e^{-\frac{\pi^2D(t-\tau)m^2}{w^2}} | < 1$ for all $n, m \ge 1$, Cauchy criterion for convergence holds; therefore, the series is uniformly convergent.

[15]To see an immediate implication of Lemma B.1, consider a case of resource-unconstrained scheduling (i.e., the number of processors is unlimited). Given a directed acyclic task graph, representing precedence relations among tasks, assume that there are no resource constraints, the endurance constraint can be ignored (i.e., the value of α is sufficiently large), and the delay budget is equal to the length of the critical path in the task graph. Then, an ASAP (as soon as possible) schedule is the best and an ALAP (as late as possible) schedule is the worst. The proof of this claim is as follows. Let T denote the critical path delay. Both ASAP and ALAP schedules yield profiles of length T. The cost of the ASAP schedule and the cost of the ALAP schedule are, respectively as follows: $\sigma_{\mathrm{ASAP}} = \sum_{k=0}^{n-1} I_k F(T, t_{k,\mathrm{ASAP}}, t_{k,\mathrm{ASAP}} + \Delta_k, \beta)$ and $\sigma_{\mathrm{ALAP}} = \sum_{k=0}^{n-1} I_k F(T, t_{k,\mathrm{ALAP}}, t_{k,\mathrm{ALAP}} + \Delta_k, \beta)$. For each task k, its start time in the ASAP schedule $t_{k,\mathrm{ASAP}}$ is the earliest possible, and its start time in the ALAP schedule $t_{k,\mathrm{ALAP}}$ is the latest possible. Thus, according to Lemma B.1(a), each term of the sum σ_{ASAP} is the smallest possible; whereas, each term of the sum σ_{ALAP} is the largest possible. Therefore, σ_{ASAP} is minimum (i.e., the ASAP schedule is the best), and σ_{ALAP} is maximum (i.e., the ALAP schedule is the worst).

PROOF. (a) The derivative of $F(T, t, t + \Delta, \beta)$ with respect to t is always nonnegative:

$$\frac{dF}{dt} = 2\sum_{m=1}^{\infty}\left[e^{-\beta^2 m^2(T-t-\Delta)} - e^{-\beta^2 m^2(T-t)}\right] \geq 0. \tag{32}$$

(b) The derivative of $F(T, t, t + \Delta, \beta)$ with respect to T is never positive:

$$\frac{dF}{dT} = 2\sum_{m=1}^{\infty}\left[-e^{-\beta^2 m^2(T-t-\Delta)} + e^{-\beta^2 m^2(T-t)}\right] \leq 0. \tag{33}$$

(c) Let $\hat{t} = t + \varepsilon$ and $\hat{T} = T + \varepsilon$. Then,

$$\begin{aligned} F(\hat{T}, \hat{t}, \hat{t} + \Delta, \beta) &= \Delta + 2\sum_{m=1}^{\infty}\frac{e^{-\beta^2 m^2(T+\varepsilon-t-\varepsilon-\Delta)} - e^{-\beta^2 m^2(T+\varepsilon-t-\varepsilon)}}{\beta^2 m^2} \\ &= \Delta + 2\sum_{m=1}^{\infty}\frac{e^{-\beta^2 m^2(T-t-\Delta)} - e^{-\beta^2 m^2(T-t)}}{\beta^2 m^2} = F(T, t, t + \Delta, \beta). \end{aligned} \tag{34}$$

□

THEOREM B.2. *Given n independent tasks, assume that the endurance constraint can be ignored (i.e., the value of α is sufficiently large), and the delay budget is $T = \sum_{k=0}^{n-1}\Delta_k$ (i.e., no idle periods allowed). Then,*

(a) $\sum_{k=0}^{n-1} I_k F(T, t_k, t_k + \Delta_k, \beta)$ *is minimized, if* $I_i \geq I_j \Rightarrow t_i \leq t_j$ *for all* $0 \leq i, j \leq n - 1$, *and*

(b) $\sum_{k=0}^{n-1} I_k F(T, t_k, t_k + \Delta_k, \beta)$ *is maximized, if* $I_i \geq I_j \Rightarrow t_i \geq t_j$ *for all* $0 \leq i, j \leq n - 1$.

PROOF. (a) Assume that for all pairs of tasks i and j adjacent in the sequence, the condition of the theorem holds ($I_i \geq I_j \Rightarrow t_i \leq t_j$), but the value of the sum is not optimal. Then, there must exist some pair of *adjacent* tasks p and q such that swapping them in the original sequence results in the new sequence with a smaller value of the sum. Loads other than p and q can be excluded from consideration because their contribution to the sum does not change due to swapping. Thus, the above suboptimality assumption can be restated as follows:

$$\begin{aligned} &I_p F(T, t_p, t_p + \Delta_p, \beta) + I_q F(T, t_q, t_q + \Delta_q, \beta) \\ &\quad \geq I_q F(T, t'_q, t'_q + \Delta_q, \beta) + I_p F(T, t'_p, t'_p + \Delta_p, \beta). \end{aligned} \tag{35}$$

Since p and q are adjacent in the sequence, $t_q = t_p + \Delta_p, t'_q = t_p, t'_p = t_p + \Delta_q$. Then, the above assumption becomes:

$$\begin{aligned} &I_p[F(T, t_p + \Delta_q, t_p + \Delta_p + \Delta_q, \beta) - F(T, t_p, t_p + \Delta_p, \beta)] \\ &\quad \leq I_q[F(T, t_p + \Delta_p, t_p + \Delta_p + \Delta_q, \beta) - F(T, t_p, t_p + \Delta_q, \beta)]. \end{aligned} \tag{36}$$

The following equality applies:

$$\begin{aligned} &F(T, t_p + \Delta_q, t_p + \Delta_p + \Delta_q, \beta) - F(T, t_p, t_p + \Delta_p, \beta) \\ &\quad = \Delta_p + 2\sum_{m=1}^{\infty}\frac{e^{-\beta^2 m^2(T-t_p-\Delta_p-\Delta_q)} - e^{-\beta^2 m^2(T-t_p-\Delta_q)}}{\beta^2 m^2} \end{aligned}$$

$$
\begin{aligned}
&- \Delta_p - 2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T - t_p - \Delta_p)} - e^{-\beta^2 m^2 (T - t_p)}}{\beta^2 m^2} \\
&= 2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T - t_p - \Delta_p - \Delta_q)}}{\beta^2 m^2} - 2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T - t_p - \Delta_p)}}{\beta^2 m^2} \\
&\quad - 2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T - t_p - \Delta_q)}}{\beta^2 m^2} + 2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T - t_p)}}{\beta^2 m^2} \\
&= \Delta_q + 2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T - t_p - \Delta_p - \Delta_q)} - e^{-\beta^2 m^2 (T - t_p - \Delta_p)}}{\beta^2 m^2} \\
&\quad - \Delta_q - 2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T - t_p - \Delta_q)} - e^{-\beta^2 m^2 (T - t_p)}}{\beta^2 m^2} \\
&= F(T, t_p + \Delta_p, t_p + \Delta_p + \Delta_q, \beta) - F(T, t_p, t_p + \Delta_q, \beta).
\end{aligned} \tag{37}
$$

Thus, the factor of I_p is equal to the factor of I_q in inequality (36): they cancel out. Since $I_p \geq I_q$, these factors must be nonpositive for the inequality to hold. However, this contradicts the statement (a) of Lemma B.1. Therefore, the nonincreasing load ordering does indeed result in the minimum value of the sum.

(b) In this case, the proof is similar to (a). Again, consider swapping two adjacent tasks p and q in the nondecreasing sequence ($I_p \leq I_q$). For the sake of contradiction, assume that, after swapping, the cost increased:

$$
\begin{aligned}
&I_p F(T, t_p, t_p + \Delta_p, \beta) + I_q F(T, t_p + \Delta_p, t_p + \Delta_p + \Delta_q, \beta) \\
&\quad \leq I_q F(T, t_p, t_p + \Delta_q, \beta) + I_p F(T, t_p + \Delta_q, t_p + \Delta_p + \Delta_q, \beta).
\end{aligned} \tag{38}
$$

$$
\begin{aligned}
&I_p [F(T, t_p + \Delta_q, t_p + \Delta_p + \Delta_q, \beta) - F(T, t_p, t_p + \Delta_p, \beta)] \\
&\quad \geq I_q [F(T, t_p + \Delta_p, t_p + \Delta_p + \Delta_q, \beta) - F(T, t_p, t_p + \Delta_q, \beta)].
\end{aligned} \tag{39}
$$

The factors of I_p and I_q cancel out. Since $I_p \leq I_q$, these factors must be nonpositive for the inequality to hold. However, this contradicts the statement (a) of Lemma B.1. Therefore, the nondecreasing load ordering does indeed result in the maximum value of the sum.

If p and q are not adjacent in the original sequence, then swapping them can be viewed as a series of swaps of adjacent tasks between p and q. Each swapped pair complies with the conditions of the theorem. In case (a), no swap improves the cost of a sequence, and in case (b), no swap worsens the cost of a sequence. □

COROLLARY B.3. *Given n tasks, assume that the endurance constraint can be ignored (i.e., the value of α is sufficiently large), and the delay budget is $T = \sum_{k=0}^{n-1} \Delta_k$ (i.e., no idle periods allowed). Then, the cost of any task sequence complying with the precedence constraints is bounded by the interval $[\sigma_\downarrow, \sigma_\uparrow]$, where $\sigma_\downarrow$ is the cost of a sequence with nonincreasing load (ignoring dependencies), and $\sigma_\uparrow$ is the cost of a sequence with nondecreasing load (ignoring dependencies).*

PROOF. According to Theorem B.2, $\sigma_{\downarrow}$ is the smallest possible value of the cost function, and $\sigma_{\uparrow}$ is the largest possible value of the cost function. □

Next, let δ denote the duration of an idle period inserted into a load sequence ending with load l that fails. Let the length of this subprofile be denoted by T_l. Assume that δ is placed between adjacent loads i and j such that $t_i < t_j \leq t_l$. As a result, the load profile duration T_l and the start times of loads following and including j are increased by δ. Their contribution to the cost of the subprofile is not changed. The *difference* Ω between the cost of the original subprofile (without recovery) and the cost of the new subprofile (with recovery) is as follows:

$$\Omega = \sum_{k|t_k<t_j} I_k F(T_l, t_k, t_k + \Delta_k, \beta) - \sum_{k|t_k<t_j} I_k F(T_l + \delta, t_k, t_k + \Delta_k, \beta). \tag{40}$$

THEOREM B.4. (a) *The value of Ω is maximized, if $t_j = t_l$.* (b) *To achieve maximum Ω by placing an idle period earlier than t_l, the duration of that inserted idle period must be greater than δ.*

PROOF. (a) Due to Lemma B.1(b),

$$F(T_l, t_k, t_k + \Delta_k, \beta) - F(T_l + \delta, t_k, t_k + \Delta_k, \beta) \geq 0. \tag{41}$$

Therefore, the greater the value of t_j, the greater the number of positive terms summed up. Thus, the maximum value of $t_j = t_l$ yields the maximum value of Ω.

(b) Let P1 denote the subprofile, ending with task l, where an idle period of length δ is inserted at t_l. Let σ_1 denote the cost of P1. Let P2 denote the subprofile, ending with task l, where an idle period of some length $\hat{\delta}$ is inserted at $t_j \leq t_l$ (i.e., before some task j preceding l). Let σ_2 denote the cost of $P2$. It is given that $\sigma_1 = \sigma_2$ (i.e., Ω is maximum for both profiles). We need to show that $\delta \leq \hat{\delta}$.

Note that

$$\sigma_1 = \sum_{k|t_k<t_l} I_k F(T_l + \delta, t_k, t_k + \Delta_k, \beta) + I_l F(T_l + \delta, t_l + \delta, t_l + \Delta_l + \delta, \beta), \tag{42}$$

and

$$\begin{aligned} \sigma_2 = & \sum_{k|t_k<t_j} I_k F(T_l + \hat{\delta}, t_k, t_k + \Delta_k, \beta) \\ & + \sum_{k|t_j \leq t_k \leq t_l} I_k F(T_l + \hat{\delta}, t_k + \hat{\delta}, t_k + \Delta_k + \hat{\delta}, \beta). \end{aligned} \tag{43}$$

According to Lemma B.1(c), $F(T + \varepsilon, t + \varepsilon, t + \Delta + \varepsilon, \beta) = F(T, t, t + \Delta, \beta)$. Thus,

$$\begin{aligned} & \sum_{k|t_k<t_j} I_k F(T_l + \hat{\delta}, t_k, t_k + \Delta_k, \beta) + \sum_{k|t_j \leq t_k \leq t_l} I_k F(T_l, t_k, t_k + \Delta_k, \beta) \\ & = \sum_{k|t_k<t_l} I_k F(T_l + \delta, t_k, t_k + \Delta_k, \beta) + I_l F(T_l, t_l, t_l + \Delta_l, \beta). \end{aligned} \tag{44}$$

The term corresponding to the task l can be dropped from both sides of the equation:

$$\sum_{k|t_k<t_j} I_k F(T_l+\hat{\delta}, t_k, t_k+\Delta_k, \beta) + \sum_{k|t_j\le t_k<t_l} I_k F(T_l, t_k, t_k+\Delta_k, \beta)$$
$$= \sum_{k|t_k<t_j} I_k F(T_l+\delta, t_k, t_k+\Delta_k, \beta) + \sum_{k|t_j\le t_k<t_l} I_k F(T_l+\delta, t_k, t_k+\Delta_k, \beta). \tag{45}$$

According to Lemma B.1(b),

$$\sum_{k|t_j\le t_k<t_l} I_k F(T_l, t_k, t_k+\Delta_k, \beta) \ge \sum_{k|t_j\le t_k<t_l} I_k F(T_l+\delta, t_k, t_k+\Delta_k, \beta). \tag{46}$$

For the equality to hold, the following must be true:

$$\sum_{k|t_k<t_j} I_k F(T_l+\hat{\delta}, t_k, t_k+\Delta_k, \beta) \le \sum_{k|t_k<t_j} I_k F(T_l+\delta, t_k, t_k+\Delta_k, \beta). \tag{47}$$

Therefore, according to Lemma B.1(b), $\delta \le \hat{\delta}$. □

THEOREM B.5. *Failing task l is unrecoverable if*

$$\alpha < \sum_{k|t_k<t_l} I_k\Delta_k + I_l\left[\Delta_l + 2\sum_{m=1}^{\infty}\frac{1-e^{-\beta^2m^2\Delta_l}}{\beta^2m^2}\right]. \tag{48}$$

PROOF. Note that $F(x, y, z, \beta) \to z - y$ as $x \to \infty$. Therefore, as δ grows, $F(t_l+\Delta_l+\delta, t_k, t_k+\Delta_k, \beta)$ tends to Δ_k. Also,

$$F(t_l+\Delta_l+\delta, t_l+\delta, t_l+\Delta_l+\delta, \beta) = \Delta_l + 2\sum_{m=1}^{\infty}\frac{1-e^{-\beta^2m^2\Delta_l}}{\beta^2m^2}. \tag{49}$$

Therefore, if $\sum_{k|t_k<t_l} I_k\Delta_k + I_l[\Delta_l + 2\sum_{m=1}^{\infty}\frac{1-e^{-\beta^2m^2\Delta_l}}{\beta^2m^2}]$ exceeds the value of α, then even an infinite-length recovery period cannot prevent load l from failing.

This condition can be relaxed to the following intuitive form: $\alpha < \sum_{k|t_k\le t_l} I_k\Delta_k$. □

B.2. Properties with Respect to Scaling

When the voltage is scaled down, it is implied that the clock frequency is reduced as well. All the results presented here are valid under a certain assumption about task durations and charges before and after voltage down-scaling. Let I and Δ denote the task current and duration, respectively, *before* its voltage is scaled down. Let $\hat{I}$ and $\hat{\Delta}$ denote the task current and duration, respectively, *after* its voltage is scaled down. We assume that, for any task,

$$\hat{\Delta} \ge \Delta \text{ and } \hat{I}\hat{\Delta} \le I\Delta. \tag{50}$$

In other words, voltage down-scaling increases task durations and decreases task charges. These conditions are easily satisfied considering the fact that task

currents are approximately proportional V^3, and task clock rates are approximately proportional to V, where V is the task voltage.

LEMMA B.6. *If* $\hat{\Delta} \geq \Delta$, *then*

$$\frac{1 - e^{-\beta^2 m^2 \Delta}}{\beta^2 m^2 \Delta} \geq \frac{1 - e^{-\beta^2 m^2 \hat{\Delta}}}{\beta^2 m^2 \hat{\Delta}}. \tag{51}$$

PROOF. Let $f(\Delta) = \frac{1-e^{-\beta^2 m^2 \Delta}}{\beta^2 m^2 \Delta}$. To demonstrate (51), we need to show that $f(\Delta)$ is monotonically *decreasing* as Δ grows ($\hat{\Delta} \geq \Delta$). In other words, the derivative $\frac{df}{d\Delta}$ must be negative—we prove this by contradiction. Assume the opposite:

$$\frac{df}{d\Delta} = \frac{e^{-\beta^2 m^2 \Delta} + \beta^2 m^2 \Delta e^{-\beta^2 m^2 \Delta} - 1}{\beta^2 m^2 \Delta^2} \geq 0. \tag{52}$$

Then,

$$\begin{aligned} e^{-\beta^2 m^2 \Delta}(1 + \beta^2 m^2 \Delta) &\geq 1 \\ 1 + \beta^2 m^2 \Delta &\geq e^{\beta^2 m^2 \Delta} = \sum_{i=0}^{\infty} \frac{(\beta^2 m^2 \Delta)^i}{i!} \\ 1 + \beta^2 m^2 \Delta &\geq 1 + \beta^2 m^2 \Delta + \frac{\beta^4 m^4 \Delta^2}{2} + \cdots \end{aligned} \tag{53}$$

Clearly, the last inequality in (53) is a contradiction; thus, (51) is true. □

LEMMA B.7. *Under assumption* (50)*, for a given task* k *before and after voltage down-scaling*

$$I_k F(T, t_k, t_k + \Delta_k, \beta) \geq \hat{I}_k F(\hat{T}, t_k, t_k + \hat{\Delta}_k, \beta), \tag{54}$$

where $\hat{T} = T - \Delta_k + \hat{\Delta}_k$.

PROOF. The inequality (54) can be expressed as

$$\begin{aligned} & I_k \left[\Delta_k + 2 \sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T - t_k - \Delta_k)} - e^{-\beta^2 m^2 (T - t_k)}}{\beta^2 m^2} \right] \\ & \geq \hat{I}_k \left[\hat{\Delta}_k + 2 \sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (\hat{T} - t_k - \hat{\Delta}_k)} - e^{-\beta^2 m^2 (\hat{T} - t_k)}}{\beta^2 m^2} \right]. \end{aligned} \tag{55}$$

Since $\hat{T} = T - \Delta_k + \hat{\Delta}_k$, we obtain another form of (54):

$$\begin{aligned} & I_k \Delta_k + 2 \sum_{m=1}^{\infty} e^{-\beta^2 m^2 (T - t_k - \Delta_k)} \frac{1 - e^{-\beta^2 m^2 \Delta_k}}{\beta^2 m^2 \Delta_k} I_k \Delta_k \\ & \geq \hat{I}_k \hat{\Delta}_k + 2 \sum_{m=1}^{\infty} e^{-\beta^2 m^2 (T - t_k - \Delta_k)} \frac{1 - e^{-\beta^2 m^2 \hat{\Delta}_k}}{\beta^2 m^2 \hat{\Delta}_k} \hat{I}_k \hat{\Delta}_k. \end{aligned} \tag{56}$$

Given that $I_k \Delta_k \geq \hat{I}_k \hat{\Delta}_k$, one can see that (54) always holds due to Lemma B.6. □

THEOREM B.8. *Let σ_1 be the cost of some given profile P1. Assume that the voltage for some task l is scaled down, thus forming a new profile, P2. Let σ_2 be the cost of P2. Under the assumption* (50)*, $\sigma_1 \geq \sigma_2$.*

PROOF. Let $\delta = \hat{\Delta}_l - \Delta_l$, where Δ_l and $\hat{\Delta}_l$ are the durations of task k before and after voltage down-scaling, respectively. Let X represent the set of tasks preceding l, and let Y denote the set of tasks following l in the sequence. Note that the length of P2 is greater than the length of P1 by δ. The costs of P1 and P2 can be expressed as follows:

$$\begin{aligned}\sigma_1 &= \sum_{k\in X} I_k F(T, t_k, t_k+\Delta_k, \beta) + I_l F(T, t_l, t_l+\Delta_l, \beta) \\ &\quad + \sum_{k\in Y} I_k F(T, t_k, t_k+\Delta_k, \beta). \qquad (57)\\ \sigma_2 &= \sum_{k\in X} I_k F(T+\delta, t_k, t_k+\Delta_k, \beta) + \hat{I}_l F(T+\delta, t_l, t_l+\hat{\Delta}_l, \beta) \\ &\quad + \sum_{k\in Y} I_k F(T+\delta, t_k+\delta, t_k+\Delta_k+\delta, \beta). \qquad (58)\end{aligned}$$

According to Lemma B.1(c), $F(T, t_k, t_k+\Delta_k, \beta) \geq F(T+\delta, t_k, t_k+\Delta_k, \beta)$. Therefore, each task in X contributes more to σ_1 than to σ_2. According to Lemma B.1(b), $F(T+\delta, t_k+\delta, t_k+\Delta_k+\delta, \beta) = F(T, t_k, t_k+\Delta_k, \beta)$. Therefore, the contribution of tasks in Y to the profile cost does not change due to scaling down the voltage of l. Finally, by Lemma B.7, $I_l F(T, t_l, t_l+\Delta_l, \beta) \geq \hat{I}_l F(T+\delta, t_l, t_l+\hat{\Delta}_l, \beta)$. Thus, $\sigma_1 \geq \sigma_2$. □

THEOREM B.9. *Assume that a given task sequence is failure-free. If voltage is scaled down for some tasks, then the resulting profile is still failure-free, provided that* (50) *holds.*

PROOF. A given profile of length T is failure-free if

$$\alpha \geq \sum_{k=0}^{n-1} I_k F(t, \min\{t, t_k\}, \min\{t, t_k+\Delta_k\}, \beta)\ ,\ \forall t \leq T. \qquad (59)$$

Consider an arbitrary time instance $T_0 \leq T$. Let q denote a task during which T_0 occurs, that is, $T_0 \in [t_q, t_q+\Delta_q]$. Since there are no failures, the cost of the subprofile of length T_0 does not exceed α:

$$\sum_{k|t_k<t_q} I_k F(T_0, t_k, t_k+\Delta_k, \beta) + I_q F(T_0, t_q, T_0, \beta) \leq \alpha. \qquad (60)$$

If voltage down-scaling is applied to any task following q, then the subprofile in question does not change, and (60) still holds. If voltage down-scaling is applied to any task preceding q, then the subprofile length is increased to $\hat{T}_0 = T_0 + \delta$, where δ is the increase in the duration of a scaled task. The cost of the subprofile of interest is reduced, according to Theorem B.8. Thus, (60) still holds. Finally, assume that the voltage of q itself is scaled down, which increases Δ_q by δ and decreases the current to $\hat{I}_q$. We want to show that for any $\hat{T}_0 \in [T_0, T_0+\delta]$,

the following inequality holds:

$$\sum_{k|t_k<t_q} I_k F(\hat{T}_0, t_k, t_k + \Delta_k, \beta) + \hat{I}_q F(\hat{T}_0, t_q, \hat{T}_0, \beta) \leq \alpha. \tag{61}$$

As $\hat{T}_0$ grows from T_0 to $T_0 + \delta$, the sum $\sum_{k|t_k<t_q} I_k F(\hat{T}_0, t_k, t_k + \Delta_k, \beta)$ decreases (see Lemma B.1). For a given $\hat{T}_0$, task q can be treated as a task q' with the duration $T_0 - t_q$ and $\hat{T}_0 - t_q$ before and after scaling, respectively. In other words, the duration of q' increases by $\hat{T}_0 - T_0$ after scaling. Note that the statement of Lemma B.7 is applicable to q', that is, $\hat{I}'_q F(\hat{T}_0, t_q, \hat{T}_0, \beta) \leq I_q F(T_0, t_q, T_0, \beta)$, where $\hat{I}'_q$ is the corresponding current of q' after scaling. Since $\hat{T}_0 - T_0 \leq \delta$, it follows that $\hat{I}'_q \geq \hat{I}_q$. Therefore, $\hat{I}_q F(\hat{T}_0, t_q, \hat{T}_0, \beta) \leq I_q F(T_0, t_q, T_0, \beta)$, and (61) is true.

Thus, voltage down-scaling cannot introduce failures to within a given subprofile. Since the choice of T_0 is arbitrary, the inequality (60) holds at any point of the profile before and after the supply voltage is scaled down. □

Consider two identical tasks i and j in a profile of length T. Assume that i precedes j (i.e., $t_i < t_j$), and there is a slack of length δ available, which can be utilized by down-scaling either the voltage of i or the voltage of j. These two possibilities are illustrated in Figure 7. For task i, let the current (the duration) before and after voltage down-scaling be denoted by I_i (Δ_i) and $\hat{I}_i$ ($\hat{\Delta}_i$), respectively. For task j, let the current (the duration) before and after voltage down-scaling be denoted by I_j (Δ_j) and $\hat{I}_j$ ($\hat{\Delta}_j$), respectively. Let X be the set of tasks scheduled before i in the profile, Y—the set of tasks scheduled between i and j, and Z—the set of tasks scheduled after j (see Figure 7). In case (a)—the slack δ is utilized by task i—the start times of j and tasks in Y and Z increase by δ. In case (b)—the slack δ is utilized by j—the start times of tasks in Z increase by δ. Note that in both cases, the profile length T also increases by δ and becomes equal to $\hat{T} = T + \delta$. Let Ω denote the *difference* between the profile cost in case (a) and the profile cost in case (b):

$$\begin{aligned}
\Omega = & \left[\sum_{k \in X} I_k F(T + \delta, t_k, t_k + \Delta_k, \beta) + \hat{I}_i F(T + \delta, t_i, t_i + \hat{\Delta}_i, \beta) \right. \\
& + \sum_{k \in Y} I_k F(T + \delta, t_k + \delta, t_k + \delta + \Delta_k, \beta) + I_j F(T + \delta, t_j + \delta, t_j + \delta + \Delta_j, \beta) \\
& \left. + \sum_{k \in Z} I_k F(T + \delta, t_k + \delta, t_k + \delta + \Delta_k, \beta) \right] - \left[\sum_{k \in X} I_k F(T + \delta, t_k, t_k + \Delta_k, \beta) \right. \\
& + I_i F(T + \delta, t_i, t_i + \Delta_i, \beta) + \sum_{k \in Y} I_k F(T + \delta, t_k, t_k + \Delta_k, \beta) \\
& \left. + \hat{I}_j F(T + \delta, t_j, t_j + \hat{\Delta}_j, \beta) + \sum_{k \in Z} I_k F(T + \delta, t_k + \delta, t_k + \delta + \Delta_k, \beta) \right].
\end{aligned} \tag{62}$$

We want to demonstrate that $\Omega \geq 0$, in other words, voltage down-scaling of j is *better* than voltage down-scaling of i. According to Lemma B.1 the cost of a task is (1) decreasing as the profile length grows; (2) increasing as its start

time grows; and (3) remains the same if the profile length and the task start time increase by the same amount. Therefore,

$$\begin{aligned}\Omega \;=\; & \left[\hat{I}_i F(T+\delta, t_i, t_i+\hat{\Delta}_i, \beta) + \sum_{k\in Y} I_k F(T, t_k, t_k+\Delta_k, \beta)\right. \\ & \left. + I_j F(T, t_j, t_j+\Delta_j, \beta)\right] - \left[I_i F(T+\delta, t_i, t_i+\Delta_i, \beta)\right. \\ & \left. + \sum_{k\in Y} I_k F(T+\delta, t_k, t_k+\Delta_k, \beta) + \hat{I}_j F(T+\delta, t_j, t_j+\hat{\Delta}_j, \beta)\right]. \end{aligned} \tag{63}$$

$$\begin{aligned}\Omega \geq \Omega' \;=\; & [\hat{I}_i F(T+\delta, t_i, t_i+\hat{\Delta}_i, \beta) + I_j F(T, t_j, t_j+\Delta_j, \beta)] \\ & - [I_i F(T, t_i, t_i+\Delta_i, \beta) + \hat{I}_j F(T+\delta, t_j, t_j+\hat{\Delta}_j, \beta)]. \end{aligned} \tag{64}$$

THEOREM B.10. *If tasks i and j are identical, then $\Omega \geq 0$ under the assumption* (50).

PROOF. Since $\Omega \geq \Omega'$, it is sufficient to prove that $\Omega' \geq 0$:

$$\begin{aligned}\Omega' \;=\; & [\hat{I}_i F(T+\delta, t_i, t_i+\hat{\Delta}_i, \beta) + I_j F(T, t_j, t_j+\Delta_j, \beta)] \\ & - [I_i F(T, t_i, t_i+\Delta_i, \beta) + \hat{I}_j F(T+\delta, t_j, t_j+\hat{\Delta}_j, \beta)] \geq 0. \end{aligned} \tag{65}$$

Tasks i and j are identical: $I_i = I_j = I$, $\Delta_i = \Delta_j = \Delta$, $\hat{I}_i = \hat{I}_j = \hat{I}$, $\hat{\Delta}_i = \hat{\Delta}_j = \hat{\Delta}$, and $T + \delta = T - \Delta + \hat{\Delta}$. We want to show that

$$\begin{aligned}\Omega' \;=\; & [\hat{I}F(T-\Delta+\hat{\Delta}, t_i, t_i+\hat{\Delta}, \beta) + IF(T, t_j, t_j+\Delta, \beta)] \\ & - [IF(T, t_i, t_i+\Delta, \beta) + \hat{I}F(T-\Delta+\hat{\Delta}, t_j, t_j+\hat{\Delta}, \beta)] \geq 0. \end{aligned} \tag{66}$$

The inequality (66) can be rewritten as follows:

$$\begin{aligned} & [IF(T, t_j, t_j+\Delta, \beta) - IF(T, t_i, t_i+\Delta, \beta)] \\ & \quad \geq [\hat{I}F(T-\Delta+\hat{\Delta}, t_j, t_j+\hat{\Delta}, \beta) - \hat{I}F(T-\Delta+\hat{\Delta}, t_i, t_i+\hat{\Delta}, \beta)]. \end{aligned} \tag{67}$$

$$\begin{aligned} & I\left[2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T-t_j-\Delta)} - e^{-\beta^2 m^2 (T-t_j)}}{\beta^2 m^2} - 2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T-t_i-\Delta)} - e^{-\beta^2 m^2 (T-t_i)}}{\beta^2 m^2}\right] \\ & \geq \hat{I}\left[2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T-t_j-\Delta)} - e^{-\beta^2 m^2 (T-\Delta+\hat{\Delta}-t_j)}}{\beta^2 m^2}\right. \\ & \left. - 2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T-t_i-\Delta)} - e^{-\beta^2 m^2 (T-\Delta+\hat{\Delta}-t_i)}}{\beta^2 m^2}\right]. \end{aligned} \tag{68}$$

$$\begin{aligned} & I\Delta \sum_{m=1}^{\infty} \frac{1-e^{-\beta^2 m^2 \Delta}}{\beta^2 m^2 \Delta}\left[e^{-\beta^2 m^2 (T-t_j-\Delta)} - e^{\beta^2 m^2 (T-t_i-\Delta)}\right] \\ & \quad \geq \hat{I}\hat{\Delta} \sum_{m=1}^{\infty} \frac{1-e^{-\beta^2 m^2 \hat{\Delta}}}{\beta^2 m^2 \hat{\Delta}}\left[e^{-\beta^2 m^2 (T-t_j-\Delta)} - e^{\beta^2 m^2 (T-t_i-\Delta)}\right]. \end{aligned} \tag{69}$$

It is given that $I\Delta \geq \hat{I}\hat{\Delta}$. Due to the inequality (51) and the fact that task i precedes task j,[16] it is clear that (69) is true. Thus, we conclude that $\Omega' \geq 0 \Rightarrow \Omega \geq 0$. □

Next, assume that some task k is failing, and there is a time slack of length $\delta > 0$ available. The failure may be repaired either (i) by inserting an idle period of length δ immediately before k; or (ii) by down-scaling the voltage of k so that the slack is fully utilized. These options are illustrated in Figure 8. Let I_k and $\hat{I}_k$ denote the current of task k before and after scaling, respectively, and let Δ_k and $\hat{\Delta}_k$ denote the duration of task k before and after scaling, respectively. Note that $\hat{\Delta}_k = \Delta + \delta$, and the time interval available for k is $[t_k, T_k]$, where $T_k = t_k + \Delta_k + \delta = t_k + \hat{\Delta}_k$. Let

$$\begin{aligned} \sigma_{k,r} &= I_k F(T_k, t_k + \delta, t_k + \Delta_k + \delta, \beta), \\ \sigma_{k,s} &= \hat{I}_k F(T_k, t_k, t_k + \hat{\Delta}_k, \beta). \end{aligned} \tag{70}$$

If task k is repaired by recovery insertion, then its cost is $\sigma_{k,r}$ (the start time is $t_k + \delta$, the duration is Δ_k, the current is I_k, and the finish time is T_k). Alternatively, if voltage scaling is used, then the cost of k is $\sigma_{k,s}$ (the start time is t_k, the duration is $\hat{\Delta}_k$, the current is $\hat{I}_k$, and the finish time is T_k). The following theorem compares $\sigma_{k,r}$ and $\sigma_{k,s}$.

THEOREM B.11. *Under the assumption* (50)*,* $\sigma_{k,r} \geq \sigma_{k,s}$.

PROOF. Recovery cost $\sigma_{k,r}$ and scaling cost $\sigma_{k,s}$ can be rewritten as follows:

$$\begin{aligned} \sigma_{k,r} &= I_k \left[\Delta_k + 2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T_k - t_k - \Delta_k - \delta)} - e^{-\beta^2 m^2 (T_k - t_k - \delta)}}{\beta^2 m^2} \right], \\ \sigma_{k,s} &= \hat{I}_k \left[\hat{\Delta}_k + 2\sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 (T_k - t_k - \hat{\Delta}_k)} - e^{-\beta^2 m^2 (T_k - t_k)}}{\beta^2 m^2} \right]. \end{aligned} \tag{71}$$

Since $T_k = t_k + \Delta_k + \delta = t_k + \hat{\Delta}_k$,

$$\begin{aligned} \sigma_{k,r} &= I_k \left[\Delta_k + 2\sum_{m=1}^{\infty} \frac{1 - e^{-\beta^2 m^2 \Delta_k}}{\beta^2 m^2} \right] = I_k \Delta_k \left[1 + 2\sum_{m=1}^{\infty} \frac{1 - e^{-\beta^2 m^2 \Delta_k}}{\beta^2 m^2 \Delta_k} \right], \\ \sigma_{k,s} &= \hat{I}_k \left[\hat{\Delta}_k + 2\sum_{m=1}^{\infty} \frac{1 - e^{-\beta^2 m^2 \hat{\Delta}_k}}{\beta^2 m^2} \right] = \hat{I}_k \hat{\Delta}_k \left[1 + 2\sum_{m=1}^{\infty} \frac{1 - e^{-\beta^2 m^2 \hat{\Delta}_k}}{\beta^2 m^2 \hat{\Delta}_k} \right]. \end{aligned} \tag{72}$$

Next, we want to show that

$$1 + 2\sum_{m=1}^{\infty} \frac{1 - e^{-\beta^2 m^2 \Delta_k}}{\beta^2 m^2 \Delta_k} \geq 1 + 2\sum_{m=1}^{\infty} \frac{1 - e^{-\beta^2 m^2 \hat{\Delta}_k}}{\beta^2 m^2 \hat{\Delta}_k}. \tag{73}$$

Due to Lemma B.6, $\frac{1-e^{-\beta^2 m^2 \Delta}}{\beta^2 m^2 \Delta}$ is monotonically decreasing as Δ grows. Since $\hat{\Delta}_k = \Delta_k + \delta \geq \Delta_k$, the claim (73) is true.

[16] For $t_j > t_i$, the term $e^{-\beta^2 m^2 (T - t_j - \Delta)} - e^{\beta^2 m^2 (T - t_i - \Delta)}$ is positive.

It is given that $I_k \Delta_k \geq \hat{I}_k \hat{\Delta}_k$, and due to (73), the multiplicative factor of $I_k \Delta_k$ is proven to be greater than that of $\hat{I}_k \hat{\Delta}_k$. Therefore, the inequality $\sigma_{k,r} \geq \sigma_{k,s}$ always holds. □

ACKNOWLEDGMENTS

We are also grateful to William Hamburgen and Deborah Wallach of the Hewlett-Packard Research Laboratory as well as Chaitali Chakrabarty of Arizona State University for their invaluable help.

REFERENCES

ARORA, P., DOYLE, M., GOZDZ, A., WHITE, R., AND NEWMAN, J. 2000. Comparison between computer simulations and experimental data for high-rate discharges of plastic lithium-ion batteries. *J. Power Sources 88*.

BARD, A. AND FAULKNER, L. 1980. *Electrochemical Methods*. Wiley, New York.

BELLMAN, R. 1961. *A Brief Introduction to Theta Functions*. Holt, Rinehart and Winston, New York.

BENINI, L., CASTELLI, G., MACII, A., MACII, E., PONCINO, M., AND SCARSI, R. 2000. A discrete-time battery model for high-level power estimation. In *Proceedings of Design, Automation, and Test in Europe*.

BENINI, L., CASTELLI, G., MACII, A., AND SCARSI, R. 2001. Battery-driven dynamic power management. *IEEE Design and Test 18*, 2.

BOTTE, G., SUBRAMANIAN, V., AND WHITE, R. 2000. Mathematical modeling of secondary lithium batteries. *Electrochimica Acta 45*.

BURD, T. AND BRODERSEN, R. 2002. *Energy Efficient Microprocessor Design*. Kluwer, Boston.

CHOWDHURY, P. AND CHAKRABARTI, C. 2002. Battery-aware task scheduling for a system-on-a-chip using voltage/clock scaling. In *Proceedings of Work. Signal Processing Systems*.

DOYLE, M., FULLER, T., AND NEWMAN, J. 1993. Modeling of galvanostatic charge and discharge of the lithium/polymer/insertion cell. *J. Electrochem. Soc. 140*, 6.

DOYLE, M. AND NEWMAN, J. 1995. Modeling the performance of rechargeable lithium-based cells: Design correlations for limiting cases. *J. Power Sources 54*.

DUDZINSKI, K. AND WALUKIEWICZ, S. 1987. Exact methods for the knapsack problem and its generalizations. *European J. Oper. Research 28*.

FULLER, T., DOYLE, M., AND NEWMAN, J. 1994. Simulation and optimization of the dual lithium ion insertion cell. *J. Electrochem. Soc. 141*, 1.

GOLD, S. 1997. A pspice macromodel for lithium-ion batteries. In *Proc. Battery Conference*.

HALL, L., SCHULZ, A., SHMOYS, D., AND WEIN, J. 1996. Scheduling to minimize average completion time: Off-line and on-line approximation algorithms. In *Proceedings Symposium on Discrete Algorithms*.

HAMBURGEN, W., WALLACH, D., VIREDAZ, M., BRAKMO, L., WALDSPURGER, C., BARLETT, J., MANN, T., AND FARKAS, K. 2001. Itsy: Stretching the bounds of mobile computing. *IEEE Computer 34*, 4.

INTEL. 2002. http://developer.intel.com/communications/app_processors.htm.

ISHIHARA, T. AND YASUURA, H. 1998. Voltage scheduling problem for dynamically variable voltage processors. In *Proceedings of International Symposium on Low Power Electronics and Design*.

LAWLER, E. 1978. Sequencing jobs to minimize total weighted completion time subject to precedence constraints. *Ann. Discrete Math. 2*.

LINDEN, D. 1995. *Handbook of Batteries*. McGraw-Hill, New York.

LIU, J., CHOU, P., BAGHERZADEH, N., AND KURDAHI, F. 2001. Power-aware scheduling under timing constraints for mission-critical embedded systems. In *Proceedings of Design Automation Conference*.

LUO, J. AND JHA, N. 2001. Battery-aware static scheduling for distributed real-time embedded systems. In *Proceedings Design Automation Conference*.

MANZAK, A. AND CHAKRABARTI, C. 2001. Variable voltage task scheduling algorithms for minimizing energy. In *Proceedings of International Symposium on Low Power Electronics and Design*.

MOONEY III, V. AND DE MICHELI, G. 2000. Hardware/software co-design of run-time schedulers for real-time systems. *J. Design Automation Embed. Systems*.

OKUMA, T., YASUURA, H., AND ISHIHARA, T. 2001. Software energy reduction techniques for variable-voltage processors. *IEEE Design and Test 18*, 2.

PANIGRAHI, D., CHIASSERINI, C., DEY, S., RAO, R., RAGHUNATHAN, A., AND LAHIRI, K. 2001. Battery life estimation of mobile embedded systems. In *Proceedings of VLSI Design*.

PEDRAM, M. AND WU, Q. 1999. Design considerations for battery-powered electronics. In *Proceedings Design Automation Conference*.

PERING, T. AND BRODERSEN, R. 1998. Energy efficient voltage scheduling for real-time operating systems. In *Proceedings of Real-Time Technology and Applications*.

PERING, T., BURD, T., AND BRODERSEN, R. 1998. The simulation and evaluation of dynamic voltage scaling algorithms. In *Proceedings of International Symposium on Low Power Electronics and Design*.

QU, G. 2001. What is the limit of energy savings by dynamic voltage scaling? In *Proceedings of International Conference on Computer-Aided Design*.

QUAN, G. AND HU, X. 2001. Energy efficient fixed priority scheduling for real-time systems on variable voltage processors. In *Proceedings of Design Automation Conference*.

RAKHMATOV, D., VRUDHULA, S., AND CHAKRABARTI, C. 2002. Battery-conscious task sequencing for portable devices including voltage/clock scaling. In *Proceedings of Design Automation Conference*.

RAKHMATOV, D., VRUDHULA, S., AND WALLACH, D. 2002. Battery lifetime prediction for energy-aware computing. In *Proceedings of International Symposium on Low Power Electronics and Design*.

ROBERTS, G. AND KAUFMAN, H. 1966. *Table of Laplace Transforms*. Saunders, Philadelphia.

SHIN, D., KIM, J., AND LEE, S. 2001. Intra-task voltage scheduling for low-energy hard real-time applications. *IEEE Design and Test 18*, 2.

SHIN, Y., CHOI, K., AND SAKURAI, T. 2000. Power optimization of real-time embedded systems on variable speed processors. In *Proceedings of International Conference on Computer-Aided Design*.

SIDNEY, J. 1975. Decomposition algorithms for single-machine sequencing with precedence relations and deferral costs. *Oper. Research 23*.

SIMUNIC, T., BENINI, L., ACQUAVIVA, A., GLYNN, P., AND DE MICHELI, G. 2001. Dynamic voltage scaling and power management for portable systems. In *Proceedings of Design Automation Conference*.

SINHA, A. AND CHANDRAKASAN, A. 2001. Energy efficient real-time scheduling. In *Proceedings of International Conference on Computer-Aided Design*.

SMITH, W. 1956. Various optimizers for single-stage production. *Naval Research Log. Quart. 3*.

WEISER, M., WELCH, B., DEMERS, A., AND SHENKER, S. 1994. Scheduling for reduced CPU energy. In *Proceedings of OS Design and Implementation*.

YAO, F., DEMERS, A., AND SHANKAR, S. 1995. A scheduling model for reduced CPU energy. *IEEE Found. Comp. Science*.

Received March 2002; accepted July 2002