Reward-Based Voltage Scheduling for Fixed-Priority Hard Real-Time Systems

Han-Saem Yun and Jihong Kim
School of Computer Science and Engineering
Seoul National University
Seoul, Korea 151-742
{hsyun, jihong}@davinci.snu.ac.kr

Abstract

We address the combined scheduling problem of maximizing the total reward of fixed-priority *hard real-time systems with a given energy budget. We present a fully polynomial time approximation scheme (FPTAS) for the off-line job-level scheduling problem and an efficient heuristic for the task-level scheduling problem. We also describe an on-line algorithm which is effective in leveraging the execution time variation.*

1 Introduction

In the *reward-based scheduling* framework [2], the workload of each task is divided into a mandatory part and an optional part and a nondecreasing concave reward function is associated with each optional part; the more the optional part is executed, the higher the reward is. The reward-based framework can model various real-time *flexible* applications that allow approximate results such as image and speech processing, multimedia, robot control/navigation systems, real-time heuristic search [2].

As flexible applications are executed on variable voltage processors, the combined problem of reward-based scheduling and voltage scheduling, which we call the *reward-based voltage scheduling* problem, has been recently investigated [4, 5, 8]. The reward-based voltage scheduling involves two-dimensional objectives, maximizing the total reward scheduling) and minimizing the energy consumption and can be defined as duals. Without loss of generality, in this paper, we consider the problem of maximizing the total reward subject to energy constraints.

Reward-based voltage scheduling was first addressed by Rusu *et al.* [4, 5]. In [5], optimal off-line solutions for frame-based task sets (where all the jobs have *identical* release times and deadlines) and periodic EDF task sets with concave reward functions are considered. The reward-based voltage scheduling problem for frame-based task sets with 0/1 reward functions was proved to be NP-hard and a heuristic for the problem was presented in [4]. Recently, an optimal off-line algorithm and an on-line algorithm for the job-level EDF reward-based voltage scheduling problem have been proposed [8].

In this paper, we consider reward-based voltage scheduling for *fixed-priority* tasks. First, we describe an *FPTAS* for the off-line job-level scheduling problem. Second, we present an efficient heuristic for the off-line task-level scheduling problem. Finally, we present an *on-line* algorithm which effectively leverage the workload variation to increase the reward within energy budget.

2 Problem Formulation

We consider a set $\mathcal{J} = \{J_1, J_2, \cdots, J_{|\mathcal{J}|}\}$ of priority-ordered jobs with J_1 being the job with the highest priority. A job $J \in \mathcal{J}$ is associated with the following attributes, which are assumed to be known off-line:

- r_J and d_J: the release time and the deadline.
- m_J: the mandatory workload.
- u_J: the sum of m_J and the upper bound of the optional workload.
- ρ_J: the reward function.

We use p_J to denote the priority of the job J. In the rest of the paper, we use i instead of J_i as a subscript of timing parameters when no confusion arises.

For the on-line scheduling problem, m_i and u_i are the worst-case values and the actual mandatory workload and upper bound of the optional workload vary within $(0, m_i]$ and $(0, u_i - m_i]$ during runtime. The total workload of J_i (i.e., the sum of the mandatory and optional workloads of J_i) is denoted by o_i and is selected between $[m_i, u_i]$, i.e., $m_i \leq o_i \leq u_i$. Associated with each optional workload o_i is a reward function $\rho_i(o_i)$, which is assume to be non-decreasing, concave, and continuously differentiable over the interval $[m_i, u_i]$ as in [2, 5]. Given a workload tuple $\mathbf{o} = \{o_1, o_2, \cdots, o_{|\mathcal{J}|}\}$, the total reward F, our optimization goal, is given by $F(\mathbf{o}) = \sum_{i=1}^{|\mathcal{J}|} \rho_i(o_i)$.

From the fact that each job runs at the constant speed under an energy-optimal voltage schedule [6, 7], the voltage schedule can be defined as a tuple of the allowed execution times $\mathbf{A} = (a_1, a_2, \cdots, a_{|\mathcal{J}|})$. Given a voltage schedule $\mathbf{A}$, the response time of each job is uniquely determined, and if every job finishes its execution by its deadline under $\mathbf{A}$, $\mathbf{A}$ is said to be *feasible*. The exact condition for a voltage schedule $\mathbf{A}$ to be feasible is given as follows (See [7] for a proof):

Condition I (Feasibility Condition).

There exists a $|\mathcal{J}|$-tuple $(f_{J_1}, f_{J_2}, \cdots, f_{J_{|\mathcal{J}|}}) \in \mathcal{T}^{\mathcal{J}}$ such that

$$\forall 1 \le i \le |\mathcal{J}| \quad \forall r \in \{t | t \in R_{\mathcal{J}} \wedge t < f_{J_i}\} \quad \sum_{J_k / p_{J_k} \le p_{J_i} \wedge r_{J_k} \in [r, f_{J_i})} a_k \;\le\; f_{J_i} - r \;.$$

A job set $\mathcal{J}$ is said to be an EDF job set if for any $J, J' \in \mathcal{J}$ (where $p_J < p_{J'}$), $d_J \le d_{J'}$ or $d_{J'} \le r_J$. When the priority assignment follows the EDF policy, that Condition I is simplified as follows (See [7] for a proof.):

Condition II (EDF Feasibility Condition).

For any $r_i < d_j$ $(1 \le i, j \le |\mathcal{J}|)$, $\sum_{k/[r_k, d_k] \subseteq [r_i, d_j]} a_k \;\le\; d_j - r_i$.

The energy consumption of the voltage schedule in terms of $\mathbf{A}$ is given by $E(\mathbf{A}) = \sum_{i=1}^{|\mathcal{J}|} a_i \cdot P(o_i/a_i)$. For a fixed workload tuple $\mathbf{o}$, the energy-optimal voltage scheduling problem is stated as maximizing $E(\mathbf{A})$ subject to Condition I (or Condition II for EDF job sets). Now, the reward-based voltage scheduling problem is formulated as follows:

Find a workload tuple $\mathbf{o} = \{o_1, o_2, \cdots, o_{|\mathcal{J}|}\}$ such that the total reward $F(\mathbf{o})$ is maximized while the corresponding energy-optimal voltage schedule does not consume more than E_{budget} .

3 Off-Line Job-Level Scheduling

The Job-level Reward-based Voltage Scheduling for Fixed-priority (J-RVSF) problem is a generalized version of the Job-level Energy-optimal Voltage Scheduling for Fixed-priority (J-EVSF) problem [7] of which the complexity was proved to be NP-hard (in the ordinary sense) [7], and can be easily shown to be NP-hard by reduction from the J-EVSF problem. (See [1] for a proof.) The main source of difficulty comes from the complicated solution space of job-level fixed-priority scheduling; it is not obvious how to directly explore the solution space given by Condition I. However, it is worthwhile to note that Condition I also represents the solution space of the J-EVSF problem and can be adequately handled by dynamic programming formulation as in [7].

We briefly review useful observations related to the feasibility condition which are shown in [7]. The feasibility condition for a fixed-priority job set (i.e., Condition I) is complicated and, consequently, inadequate for the efficient exploration of the solution space. On the contrary, the feasibility condition for an EDF job set (i.e., Condition II) is quite simple. The following key property establishes a link between Conditions I and II.

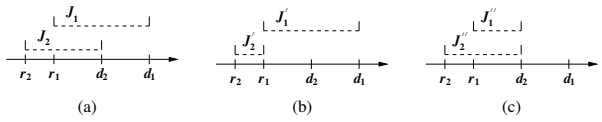

Figure 1. An example of EDF-equivalent job sets.

Property 1 [7] *Given a $|\mathcal{J}|$-tuple $\mathbf{f} = (f_{J_1}, f_{J_2}, \cdots, f_{J_{|\mathcal{J}|}})$, let $\mathcal{J}^{\mathbf{f}}$ represent the job set $\{J_1', J_2', \cdots, J_{|\mathcal{J}|}'\}$ where $p_{J_i'} = p_{J_i}, c_{J_i'} = c_{J_i}, r_{J_i'} = r_{J_i}$ and $d_{J_i'} = f_{J_i}$ for all $1 \le i \le |\mathcal{J}|$. $\mathcal{J}^{\mathbf{f}}$ is said to be EDF-equivalent to $\mathcal{J}$ if $\mathcal{J}^{\mathbf{f}}$ is an EDF job set. Then, the set of all feasible schedule of $\mathcal{J}$ is equal to the set of all feasible schedules of $\mathcal{J}$'s EDF-equivalent job sets.*

From Property 1, there is a one-to-one correspondence between feasible schedules of a fixed-priority job set $\mathcal{J}$ and feasible schedules of $\mathcal{J}$'s EDF-equivalent job sets. Figure 1 shows an example of EDF-equivalent job sets. Figure 1.(a) shows the original job set $\mathcal{J} = \{J_1, J_2\}$. In this example, J_2 has a lower priority but earlier deadline than J_1, so $\mathcal{J}$ is not an EDF job set. In Figures 1.(b) and 1.(c), two job sets are shown, which are EDF-equivalent to $\mathcal{J}$. The job sets $\{J_1', J_2'\}$ and $\{J_1'', J_2''\}$ are obtained by choosing (r_{J_1}, d_{J_1}) and (d_{J_2}, d_{J_2}) as deadlines, respectively. Both job sets follow the EDF priority assignment and the maximum-reward schedule (resp. the energy-optimal voltage schedule) for each job set can be computed by the polynomial-time optimal algorithm for the J-RVSE problem [8] (resp. Yao's optimal algorithm [6] for the J-EVSE problem). The optimal schedule of $\mathcal{J}$ is equal to that of $\{J_1', J_2'\}$ or $\{J_1'', J_2''\}$ depending on the workload of J_1 and J_2. For a given fixed-priority job set $\mathcal{J}$, the problem of finding a maximum-reward (resp. minimum-energy) schedule of $\mathcal{J}$ is reduced to the problem of finding an EDF-equivalent job set of $\mathcal{J}$ that maximizes the total reward (resp. minimizes the energy consumption).

For a job set with N jobs, there are $O(N!)$ EDF-equivalent job sets in the worst case. However, using dynamic programming formulation, the EDF-equivalent job sets can be enumerated intelligently without actually enumerating all of them [7]. We first identify appropriate "overlapping" (or reusable) substructure to which dynamic programming can be applied iteratively. We note that the "optimal substructure" is naturally reflected by *blocking tuples*, which are just sequences of time points in $\mathcal{T}_{\mathcal{J}} = \{r_J, d_J | J \in \mathcal{J}\}$ in strictly increasing order. That is, any solution (satisfying Condition I) for the whole interval can be obtained by merging solutions of the sub-intervals defined by a blocking tuple.

Figure 2 shows an example job set and its corresponding EDF-equivalent job set whose time interval is partitioned

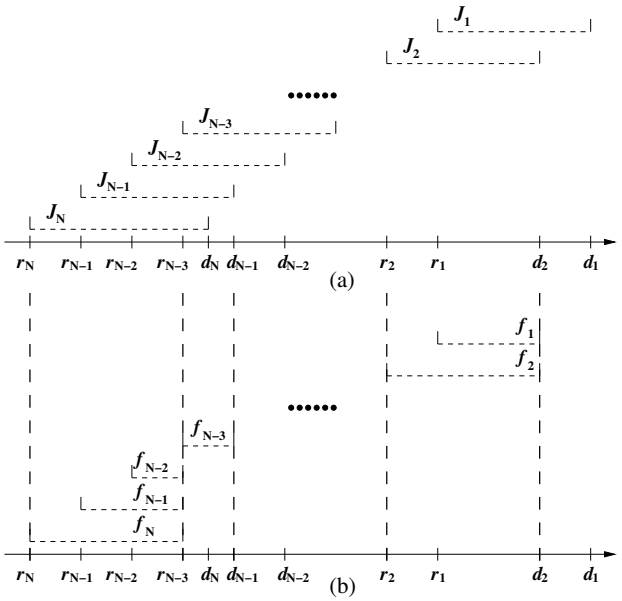

Figure 2. An example illustrating the dynamic programming formulation. (a) An original job set and (b) an EDF-equivalent job set defined by a blocking tuple $(r_N, r_{N-3}, d_{N-1}, \cdots, r_2, d_2, d_1)$.

by a blocking tuple $(r_N, r_{N-3}, d_{N-1}, \cdots, r_2, d_2)$, which is depicted by a set of the dashed thick lines in Figure 2.(b). Note that jobs in each sub-interval follow the EDF-priority assignment and that the maximum-reward schedule for each partitioned sub-interval within a given energy budget can be found in polynomial time by the algorithm for the J-RVSE problem [8]. The following property makes it possible to explore the solution space efficiently by using dynamic programming formulation.

Property 2 [7] *For an interval $[t,t']$ $(t,t' \in \mathcal{T}_\mathcal{J})$, let $\mathcal{J}_{[t,t']}$ represent the job set that consists of jobs in $\mathcal{J}$ whose release times are within the interval $[t,t')$ with their deadlines, if later than t', adjusted to t'. If $\mathcal{J}_{[t,t']}$ follows the EDF priority, it is said to be* atomic. *For any EDF-equivalent job set $\mathcal{J}'$ of a fixed-priority job set $\mathcal{J}$, there always exists a blocking tuple $(b_1, b_2, \cdots, b_l)$ such that $\mathcal{J}' \equiv \cup_{k=1}^{l-1} \mathcal{J}_{[b_k, b_{k+1}]}$ where $[b_k, b_{k+1}]$ is atomic for all $1 \le k < k$.*

Note that $\{\mathcal{J}_{[b_1,b_2]}, \mathcal{J}_{[b_2,b_3]}, \cdots, \mathcal{J}_{[b_{l-1},b_l]}\}$ is a partition of $\mathcal{J}'$ and the execution intervals of the partitioned job sets do not overlap one another. Property 2 establishes a one-to-one correspondence between EDF-equivalent job sets and blocking tuples. Since the each partitioned job set (i.e., $\mathcal{J}_{[b_k,b_{k+1}]}$) follows the EDF priority, it can be handled by the polynomial-time optimal algorithm for the J-RVSE problem [8], comprising the "optimal substructure" of the dynamic programming formulation.

By exploting the partitionable structure of the J-RVSF problem (i.e., Property 2), dynamic programming formulation for the J-RVSE problem can be constructed. For two time-instants $t,t' \in \mathcal{T}_\mathcal{J}$ $(t < t')$, let $F_{[t,t']}\langle e \rangle$ represent the maximum achievable total reward for the job set $\mathcal{J}_{[t,t']}$ with the energy budget of e. (It can be computed by the algorithm for the J-RVSE problem [8].) Then, from the Property 2, the maximum achievable total reward for the job set $\mathcal{J}$ with the energy budget of E_{budget} (i.e., $F_{[0,H]}\langle E_{\text{budget}} \rangle$) is given by

$$\max \Big\{ \sum_{k=1}^{l-1} F_{[b_k,b_{k+1}]}\langle e_k \rangle \;\Big|\; 0 = b_1 < b_2 < \cdots < b_l = H \;\wedge \quad (1)$$
$$1 \le k < l\,,\ [b_k, b_{k+1}] \text{ is atomic.} \wedge \sum_{k=1}^{l-1} e_k \;\le\; E_{\text{budget}} \Big\}\,.$$

To consider a tabular method for the problem, for the time being, let us assume that infinite columns that represent the continuous energy values are available. Each row represents an interval $[0,t_k]$ $(t_k \in \mathcal{T}_\mathcal{J})$ (i.e., the number of rows is $|\mathcal{T}_\mathcal{J}|$.). The procedure to fill in the table entries proceeds row by row. For the top row (i.e., the empty interval $[0,0]$), every entry is filled with 0. Each subsequent row is filled with the maximum achievable total reward for $\mathcal{J}_{[0,t]}$ with the energy budget e (i.e., $F_{[0,t_k]}\langle e \rangle$), which can be computed by using the entries in the previous row:

$$F_{[0,t_k]}\langle e \rangle \;=\; \max_h \Big\{ \max_{\Delta e} \big\{ F_{[0,t_h]}\langle e - \Delta e \rangle + F_{[t_h,t_k]}\langle \Delta e \rangle \big| 0 < \Delta e < e \big\}$$
$$\Big|\; [t_h, t_k] \text{ is atomic.} \Big\}\ .$$

Note that $F_{[0,t_h]}\langle e - \Delta e \rangle$ has been already stored in the previous row and that $F_{[t_h,t_k]}\langle \Delta e \rangle$ can be directly computed by the J-RVSE algorithm [8] since $\mathcal{J}_{[t_h,t_k]}$ is an EDF job set. Finally, once the entire table is filled in, the maximum achievable total reward (i.e., $F_{[0,H]}\langle E_{\text{budget}} \rangle$) is stored in the last row and the blocking tuple and the energy budget for each atomic interval (i.e., $(b_1, b_2, \cdots, b_l)$ and e_k in Eq.(1), respectively) can be found by tracking the filling procedure.

We can transform the tabular method into an FPTAS by considering discrete energy values that represent sufficiently close continuous energy values. The relative error of the FPTAS depends on how the closeness is defined; the smaller the threshhold for the closeness, the smaller the relative error at the cost of increasing computation time. For complete description of the algorithm and proofs, refer to [1].

4 Off-Line Task-Level Scheduling

In this section, we describe an off-line algorithm for the Task-level Reward-based Voltage Scheduling for Fixed-priority (T-RVSF) problem which is based on Gruian's algorithm for the Task-level Energy-optimal Voltage Scheduling for Fixed-priority (T-EVSF) problem [3]. We can easily reduce the T-EVSF problem to the T-RVSF problem in polynomial time. Furthermore, the T-EVSF problem can be proved to be NP-hard in the ordinary sense (refer to [1] for the proof.), implying that the T-RVSF problem is also NP-hard. Furthermore, the inherent complexity of fixed-priority schedulability analysis is not involed in the NP-hardness proof, i.e., the periods of transformed instances are determined such that the schedulability can always be checked in

polynomial-time. Therefore, we believe that the complexity of the T-EVSF problem (as well as the T-RVSF problem) is beyond the ordinary NP-hardness and, consequently, these problems are not likely to admit an FPTAS.

As with the algorithms for the J-EVSE and J-EVSF problems, the algorithm for the T-EVSF problem uses a voltage scheduling algorithm as a subroutine. In this paper, we consider Gruian's algorithms for the T-EVSF problem [3]. In devising the algorithm for T-RVSE problem, we adopt useful insights from the J-EVSE problem. The algorithm for the J-EVSE problem starts with the whole optional workloads and iteratively decreases the optional workloads until the energy consumption of the corresponding energy-optimal voltage schedule, which can be directly computed by Yao's algorithm [6], reaches the energy budget. The amount of optional workload of each job is determined such that the gradients of jobs are as uniform as possible. The *gradient* g_i of a job J_i is defined to be the decrease in the power dissipation per unit decrease in the reward, i.e., $g_i \stackrel{\text{def}}{=} P'(s_i)/\rho_i'(o_i)$ [8].

The procedure of our iterative algorithm for the task-level RVSF problem is very similar to that of the algorithm for the J-RVSE problem. The algorithm starts with the whole optional workloads and iteratively decreases the optional workload of each task such that the gradient of each task is as flat as possible. When computing the voltage schedule at each iteration, the algorithm uses Gruian's algorithm. (For complete description of our algorithm, refer to [1].) Although the heuristic is very simple, its performance is comparable to that of the FPTAS for the J-RVSF problem for real-world applications as shown in Section 6.

5 On-Line Algorithm

The on-line algorithm for fixed-priority tasks slightly differs from the previously proposed on-line algorithm for EDF tasks [8]. The existing on-line algorithm for EDF tasks consists of the following parts: slack estimation and slack distribution. The goal of the slack estimation part is to identify as much slack time as possible and records the residual energy reserved by an unexpected lower speed or idle time. The goal of the slack distribution part is to distribute the slack time and the energy slack so that the gradient of the resultant schedule is as uniform as possible. Among these, only slack-time estimation part is changed because other parts are not dependent on the priority assingment policy. For the slack-time estimation, we adopt the existing method developed by Gruian [3], which is based on the priority-based slack stealing method.

6 Experimental Results

In order to evaluate the performance of the proposed algorithms, we performed experiments with test job sets constructed from periodic task sets of three real-world applications: MPEG4 Videophone, CNC and Avionics. We used logarithmic reward functions of the type $\alpha_i \cdot \log(\beta_i \cdot o_i + 1)$. First, we compared the FPTAS for the J-RVSF problem in Section 3 and the heuristic for the T-RVSF problem in Section 4. the quality of solution (the total reward) computed by the heuristic was very close to that of the provably close to optimal solution obtained by the FPTAS with $\varepsilon = 1.0\%$; within 0.7% for MPEG4 Videophone application, and only about $2 \sim 3\%$ worse than the FPTAS. Next, we evaluated the performance of the on-line algorithm in Section 5. For a comparison, the FPTAS computes the near-optimal solution (within 1.0%) with the complete execution trace information. The result obtained by on-line algorithm was only $5 \sim 12\%$ worse than the base schedule.

7 Conclusion

We investigated the problem of reward-based voltage scheduling for fixed-priority hard real-time systems. First, we present an FPTAS for the off-line job-level scheduling problem. Second, we propose an heuristic for the off-line task-level scheduling problem whose performance is comparable to the FPTAS. Finally, an efficient low-overhead on-line algorithm was presented. The proposed algorithms can be further extended in several directions. As our immediate future work, we are interested in a more realistic processor model with a limited number of voltage levels and transition overheads in time and energy. In addition, we plan to develop off-line and on-line algorithms for 0/1 reward functions.

References

[1] –. Reward-Based Voltage Scheduling for Fixed-Priority Hard Real-Time Systems. Technical report. Available at *http://davinci.snu.ac.kr/Download/parc04_techrep.pdf*.

[2] H. Aydin, R. Melhem, D. Mossé, and P. M. Alvarez. Optimal Reward-Based Scheduling for Periodic Real-Time Tasks. *IEEE Transactions on Computers*, 50(2):111–130, 2001.

[3] F. Gruian. Hard Real-Time Scheduling for Low-Energy Using Stochastic Data and DVS Processors. In *Proc. of International Symposium on Low Power Electronics and Design*, pages 46–51, 2001.

[4] C. Rusu, R. Melhem, and D. Mossé. Maximizing the System Value While Satisfying Time and Energy Constraints. In *Proc. of Real-Time Systems Symposium*, pages 246–255, 2002.

[5] C. Rusu, R. Melhem, and D. Mossé. Maximizing Rewards for Real-Time Applications with Energy Constraints. *ACM Transactions on Embedded Computing Systems*, 2(4):537–559, 2003.

[6] F. Yao, A. Demers, and S. Shenker. A Scheduling Model for Reduced CPU Energy. In *Proc. of IEEE Annual Foundations of Computer Science*, pages 374–382, 1995.

[7] H.-S. Yun and J. Kim. On Energy-Optimal Voltage Scheduling for Fixed-Priority Hard Real-Time Systems. *ACM Transactions on Embedded Computing Systems*, 2(3):393–430, 2003.

[8] H.-S. Yun and J. Kim. Reward-Based Voltage Scheduling for Hard Real-Time Systems with Energy Constraints. In *Proc. of RTCSA'04*, to appear. Available at *http://davinci.snu.ac.kr/Download/rtcsa04.pdf*.