BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES
Vol. 57, No. 2, 2009
DOI: 10.2478/v10175-010-0115-2

Dedicated to Professor Wiesław Woliński
on the occasion of his eightieth birthday

On job models in power management problems

R. RÓŻYCKI and J. WĘGLARZ*

Institute of Computing Science, Poznan University of Technology, 2 Piotrowo St., 60-965 Poznań, Poland

Abstract. In the paper a model of job processing is presented where processing speed is related to the amount of power allotted to this job at a moment. Basic time-optimal results are showed assuming that energy is treated as a scarce doubly-constrained resource in a computer system. The proposed approach is applicable to the practical power management problems appearing in modern portable device systems.

Key words: power management, job model, scheduling problem, continuous resource, doubly-constrained resource.

1. Introduction

In the past few years the idea of green computing, seen as an efficient utilization of computer resources, achieves a growing interest. Since during the computing, energy is consumed as a main resource, appropriate power management is a basic technique for applying the green computing principles. Power management can also improve computing efficiency. As it is well known, the efficiency of computing a set of programs in a given computer system may be evaluated using various measures such as, for example, completion time or flow time. In general, there are two ways to improve computing efficiency by a proper power management. In the first one, a given performance measure is going to be improved assuming the given level of available energy, whereas in the second, the energy consumption is reduced supposing that the given value of a performance measure is maintained. The first problem, known as a laptop problem [1], is typical for portable electronic devices driven by electric energy accumulated in batteries with limited capacity. In the second problem (so-called server problem) the goal is to minimize the energy consumption in order, for example, to reduce the overall computing cost.

One of the research direction towards advanced power management is the use of variable speed processors. A variable speed processor (VSP) [2] is able to adjust its clock period at each cycle. For example, modern operating systems utilizing Intel's Speedstep or Foxton as well as AMD's PowerNow technologies are able to control the speed of processors to prolong the battery life or to improve computing efficiency. Therefore, an operating system of a computer can decide not only which job to perform at the moment but also at what processing rate this job has to be executed in order to minimize the given criterion.

The overall performance enhancement of a computer system and/or energy savings are possible because the relation between the processor speed and the electrical power used to process a job is nonlinear. This relation may be formalized by using different models of job processing. The power vs. processing speed model has been used up to now commonly. Various scheduling problems were formulated basing on the assumption that the resource (energy) consumption depends on the processing rate of a processor. Except for the total energy in the server versions of a scheduling problem, a number of optimization criterions are taken into account in the laptop form of a problem. These are, for example, schedule length, mean flow time or maximum lateness [3–5]. It is worth noting that power as a temporal usage of energy is usually assumed to be available with no limits. This means that energy is treated in fact as a nonrenewable resource only. Some exact and heuristic algorithms were proposed to solve both the deterministic and non-deterministic cases under this assumption. (see [4, 6] as surveys).

On the other hand, in the deterministic scheduling theory models of jobs in which job processing time (in general) depends on the allocated resources are quite popular. Among wide variability of such models one can distinguish two main classes. In the first class, amounts of resources allotted to a job determine its processing time directly. The second class contains models where processing rate of a job is related to the allocation of resources to this job. The models from the second class allow to consider the situations in which resource allocation may change during the execution of a job. In both classes of models various categories of resources may be considered. If the processing time (or processing rate) is defined on a finite set of possible number of resource units, then the model has a discrete nature. In the case of power management, natural are models with continuously-divisible resources (continuous resource in short) which may be allotted to jobs in amounts that are arbitrary within given intervals. For such resources, the model relating job processing speed to the resource amount allotted to this job at time t, is the most general one. In this paper we show that this model may be profitably applied in the case of scheduling problems in which power (in general – energy) consumed by a processor depends on the processing rate of a job.

Although we assume that job processing rate is related to the amount of the allotted power, it is still possible to involve other types or even categories of resources in the problem. These additional resources, required by jobs in predefined

*e-mail: Jan.Weglarz@cs.put.poznan.pl

amounts, do not influence on the processing time of a job. As a consequence the problem may be stated as a deterministic scheduling problem where jobs competing for power and other types of resources may be processed with various speeds.

In this paper we consider a set of identical parallel machines as an additional discrete resource. It is justified by the fact that multicore processors become available even in the portable electronic devices. The microprocessor cores (seen as parallel machines) share the common source of the power to process the set of jobs. Thus, it is good example of the problem where optimal scheduling policy may improve energy savings and/or overall system performance.

In the next section we compare two main models of job processing. In Sec. 3 the considered problem is formulated. The Sec. 4 describes basic time optimal results for the case where power is the only limited resource. In Sec. 5 more general case with energy as a doubly constrained resource is considered. The paper is completed with some final remarks in Sec. 6.

2. Comparison of job processing models

Let us start with the model commonly used in the literature (e.g. [4, 6]) concerning the power management in microprocessor systems. The model is based on the assumption that the power consumed during job processing depends on the processing speed. If we represent a power usage of a processor as a non-decreasing function of its speed, then it is practically justified to assume that this function is strictly convex. It means that in order to minimize the total energy consumption it is advisable to execute jobs as slowly as possible. The following form of the function is used to express the relation between processing speed s of a processor, and a power p consumed during job processing:

$$p(s) = s^{\alpha} \quad (\alpha > 1). \tag{1}$$

We will call function $p(\cdot)$ a power usage function. In particular, for a microprocessor based on CMOS technology α is assumed to be equal to 3.

Moreover, it is assumed that job i is characterized by the parameter w_i ($w_i > 0$, $i = 1, 2, ..., n$). w_i represents the size of a job and may be measured as a particular number of CPU cycles needed by job i to be processed. Of course, processing time of job i depends on its size w_i and on the processing speed of a processor executing this job. A job i is accomplished at completion time C_i if the following equation is fulfilled:

$$\int_0^{C_i} s(t)dt = w_i.$$

During the execution of job i the energy amount E_i given by the formula

$$E_i = \int_0^{C_i} p(s(t))dt,$$

is consumed.

It is worth noticing that in model (1) the processing speed of a processor is treated as a decision variable and it determines the temporal usage of energy – power. An argument of the function in model (1) is not limited, i.e. $s \in [0, \infty)$. As a, consequence, the resulting power usage is theoretically unlimited too. It is worth stressing that an amount of power directly determines the temperature of a microprocessor (nearly all energy consumed by a processor is released as heat). Thus, no limits for power usage may lead to a processor overheating and, in consequence, to a serious damage of a computer system.

Let us pass to the model in which power allotted to a job at a time determines a temporal rate of the job execution. As a consequence, instead of power usage functions we will use in the model so-called processing speed functions (speed function in short). Formally the model is expressed as follows:

$$\begin{aligned} \dot{x}_i(t) = \frac{dx_i(t)}{dt} = s_i(p_i(t)), \\ x_i(0) = 0, \quad x_i(C_i) = w_i, \end{aligned} \tag{2}$$

where $x_i(t)$ – state of job i at time t, $s_i(\cdot)$ – increasing (positive), continuous speed function of job i, $s_i(0) = 0$, $p_i(t)$ – an amount of power allotted to job i at time t, w_i – size of job i, C_i – completion time (unknown a priori) of job i.

The value of $x_i(t)$ is an objective measure of work related to the processing of job i. This can be, for example, the number of CPU cycles already processed for performing a program.

Let us compare models (1) and (2) shortly. The model (1) suffers from the lack of generality. Particular type of power usage function is used in this model. Although, such simplification is justified for a class of contemporary microprocessor technologies (e.g. the cubic function is assumed for the aforementioned CMOS technology), future technologies and microcomputer architectures may require another type of functions. As a consequence, model (1) should be generalized in order to allow an analysis of properties of optimal solutions for the much broader class of power usage functions.

The assumptions imposed on a speed function in model (2) are less restrictive. The speed function $s_i(\cdot)$ must be increasing and continuous only. Thus, in model (2) it is possible to consider any type of practically justified processing speed functions.

Moreover, model (1) assumes that the power usage function is the same for all jobs and jobs differ in sizes only, whereas speed function in model (2) depends on a job. However, it is well known (see e.g. [2]) that power consumption per-cycle may differ for various instructions executed by a processor, and thus different functions for different types of jobs should be considered. Notice that such functions may be estimated empirically for many popular algorithms. If we assume different speed functions for jobs, the size w_i of job i may be represented simply by an amount of data (e.g. size of a table or a file) to be processed. It is easy to observe that model (2) is equivalent to model (1) if the functions

$$s_i(\cdot) = s(\cdot) = p^{-1}(\cdot) = p_i^{1/\alpha}, \quad i = 1, 2, ..., n, \quad \alpha > 1.$$

3. Problem formulation

Consider a set of n independent, non-preemptive jobs and m parallel identical machines. All jobs are ready to be processed at the same moment. Each job requires for its processing a machine and an amount of an energy. Each job is performed by at most one machine at a time and a machine is able to process at most one job at a time. The processing rate of a job depends on the amount of the electrical power $p_i(t)$ allotted to job i at a time t and this relation is expressed by (2). As a consequence, we consider the problem, where the processing rate of a job may change during its execution. Job i is characterized by the processing speed function $s_i(\cdot)$ and the size w_i. Both power and energy are limited and available in amounts P and E, respectively.

This formulation allows to model the practical situation, where a set of independent programs has to be executed on a multiprocessor portable device with processors driven by the common energy source – a battery of limited capacity. To prevent the computer system from overheating a power usage limit is established. The size of a program can be measured as a required number of CPU cycles. We will assume that processing rate function of job i in model (2) is increasing and strictly concave. Such an assumption is justified to express the real relation between the temporal power usage and the processing rate in contemporary microprocessor systems.

The objective is to find a vector function $\boldsymbol{p(t)} = [p_1(t), p_2(t),...,p_n(t)]$, $p_i(t) \geq 0$, $i = 1, 2, ..., n$ which, under the constraints imposed, minimizes the schedule length T. Optimal values of $\boldsymbol{p(t)}$ and T will be denoted by $\boldsymbol{p^*(t)}$ and T^* respectively. Knowing $\boldsymbol{p^*(t)}$ we are able to calculate the energy consumption for job i by integrating $p_i^*(t)$ up to the completion time C_i.

4. Scheduling under a power constraint only

It is obvious, that the problem of overheating involve both the computer systems driven by battery with limited capacity, as well as those ones with permanent source of electricity (connected to the outlet). The simplest way to restrict the uncontrolled growth of temperature is to set up the power usage limit for a computer system. In such a case, model (2), with additional assumption on an available power, is a good starting point for analysis of time-optimal properties of schedules.

Let us treat now an energy as a renewable resource. It means that available total energy is unlimited ($E = \infty$) and the only scarce resource is power. We will denote by T^∞ the minimum schedule length for $E = \infty$.

Then, the following constraint has to be fulfilled in the feasible schedule at every moment:

$$\sum_{i=1}^{n} p_i(t) \leq P \tag{3}$$

For concave processing speed functions and $n \leq m$ the following result holds [7]:

Corollary 1

The optimal power allocation for strictly concave processing speed functions has the following form:

$$p_i^*(t) = p_i^* = s_i^{-1}(w_i/T^\infty), i = 1, ..., n, t \in < 0, T^\infty >, \tag{4}$$

where T^∞ is the positive root of the equation

$$\sum_{i=1}^{n} s_i^{-1}(w_i/T) = P. \tag{5}$$

From Corollary 1, one can see that in the optimal schedule jobs are performed in parallel using the constant power amount given in (4). Moreover, the optimal vector of power allocation $\boldsymbol{p}^*$ calculated using (4) guarantees that jobs are finished at the same moment.

In general, minimal T^∞ can be found from (5) numerically. Nevertheless, in some important practical cases, where the functions $s_i(\cdot) = c_i p_i^{1/\alpha_i}$, $c_i > 0$, $\alpha_i \in \{2, 3, 4\}$, $i = 1, 2, \ldots, n$, the solution of Eq. (5) can be found analytically.

The general methodology (e.g. for $n > m$) for solving the problems with discrete and continuous renewable resources is presented in [8]. The proposed methodology is general enough to cover the cases with concave and convex processing speed functions.

5. Scheduling under power and energy constraints

If we consider portable electronic devices, both a total energy consumption and power used for a job processing may not be neglected. It is obvious that the energy has to be treated as a doubly constrained resource, where both: usage at every moment (i.e. power) and total consumption (i.e. energy) are constrained.

Let us denote by $F(T)$ the total energy consumption as a function of schedule length T:

$$F(T) = \sum_{i=1}^{n} \int_0^T p_i(t)dt. \tag{6}$$

Of course, $F(T)$ is determined for $T \geq T^\infty$, and for $T^\infty < T < \infty$ we can always find $P_1 < P$ from (3).

If we assume that the energy is a doubly-constrained resource, then both: power usage limit P and energy consumption limit E are known in advance. A schedule is feasible if, together with (3), the following inequality holds at every time t:

$$\sum_{i=1}^{n} \int_0^T p_i(t)dt \leq E. \tag{7}$$

5.1. The case $n \leq m$. Let us start with the assumption that the number of jobs is not greater than the number of machines ($m \geq n$). Since the machines are identical and all jobs may be performed in parallel, the problem, as we will see, has pure continuous nature (the assignment of jobs to particular machines may be neglected).

It is proved [7] that for strictly concave $s_i(\cdot)$, $\boldsymbol{p^*(t)}$ exists if and only if

$$\lim_{T \to \infty} T \sum_{i=1}^{n} s_i^{-1}(w_i/T) < E.$$

It is worth stressing that this condition is fulfilled for a large class of functions(including power functions) even for $E = 0$.

Moreover, for strictly concave $s_i(\cdot)$, all the jobs are performed in parallel, the power allocation to jobs is constant, and the completion time C_i of all jobs is the same $C_i = T^*$ $(i = 1, 2, ..., n)$.

Corollary 2

The optimal power allocation for strictly concave processing rate functions has the form of

$$p_i^*(t) = p_i^* = s_i^{-1}(w_i/T^*), i = 1, ..., n, t \in < 0, T^* >, \quad (8)$$

where T^* is the positive root of the equation

$$T \sum_{i=1}^{n} s_i^{-1}(w_i/T) = E \quad (9)$$

if $\sum_{i=1}^{n} s_i^{-1}(w_i/T) \leq P$
or can be calculated using (4) otherwise.

In the first case the active constraint is from the side of E, not P. It means that (8) is the optimal solution to the problem since it does not exceed the power level P. It may happen however that that the amount of P is a critical constraint for the given instance of the problem. In this case, the minimum schedule length is equal to T^{∞}. The positive roots of (5) and (9) are unique because of the monotonicity of $s_i(\cdot)$.

Moreover, basing on [7] we may formulate the following results.

Observation 1

The minimum level of energy which ensures the minimum schedule length for a given level of power P is $E_{\min} = F(T^{\infty}) = P \cdot T^{\infty}$, where T^{∞} is the positive root of (5).

Observation 2

The minimum level of power $P_{\min}$ which ensures minimum schedule length for a given energy level E, may be found from:

$$P_{\min} = \sum_{i=1}^{n} s_i^{-1}(w_i/T^*),$$

where T^* is calculated as a positive root of (9).

Now let us comment the above results from the view point of finding optimal solutions. To find the schedule of the minimum length it is better to start with Eq. (9), since it is usually of simpler form than (5). For example for

$$s_i(\cdot) = c_i p_i^{1/\alpha_i}, \quad c_i > 0, \quad \alpha_i \in \{2, 3, 4, 5\},$$

it is analytically solvable because it is an algebraic equation of an order lower by 1 than in the case of (5). If we fail, i.e. if P has been exceeded, we have to solve (5), but the information obtained by solving (9) is valuable anyway.

The presented results are valid for the case of preemptive or non-preemptive jobs, since the ability of preemption of any job does not improve the schedule length.

5.2. The case $n > m$. Let us pass now to the general case of $n > m$. Basing on the results for the strictly concave processing speed functions and $n \leq m$ presented in Subsec. 5.1, it is easy to see that as much as possible jobs should be performed in parallel in an optimal schedule. This means that last m jobs have to be completed at the same moment in the optimal schedule. Thus any potential solution of the problem (any potentially optimal schedule) can be divided into $r \leq n - m + 1$ intervals of length T_k, $k = 1, 2, ..., r$ defined by the completion times of consecutive jobs. Of course, no machine is idle until the completion of the last m jobs. Let Z_k, $k = 1, 2, \ldots, r$ denote the m-combination of jobs corresponding to the k-th interval. Thus a feasible sequence S of m-combinations Z_k, $k = 1, 2, \ldots, r$ is associated with each such potentially optimal schedule (Fig. 1). By K_i we will denote the set of indices of Z_k's such that job $i \in Z_k$. Moreover, denote by E_k the energy consumption related to Z_k.

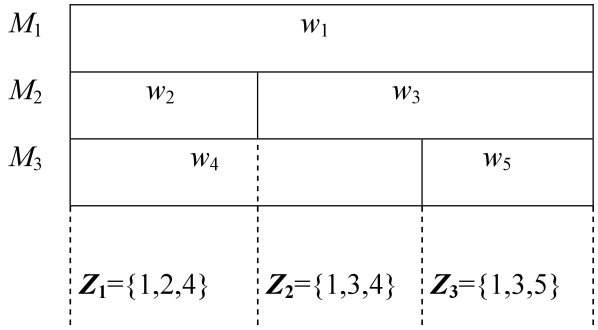

Fig. 1. An example of a potentially optimal schedule and related feasible sequence S={1,2,4},{1,3,4},{1,3,5}

It is worth noticing that for each m-combination Z_k, $k = 1, 2, \ldots, r$, an optimal allocation of the continuous resource among machines depends on the amounts w_{ik} of each w_i assigned to this m-combination (so-called *demand division* – Fig. 2.) and is constant in time [8].

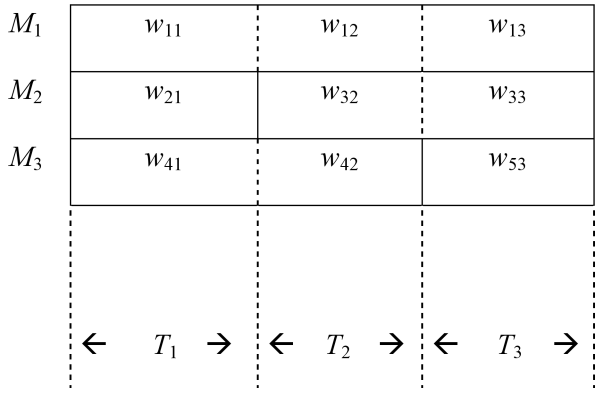

Fig. 2. An example of demand division

As a consequence, for a given feasible sequence S, an optimal continuous resource allocation can be found by solving the following non-linear mathematical programming problem: Minimize:

$$T = \sum_{k=1}^{r} T_k \left(\{w_{ik}\}_{i \in Z_k}, E_k \right). \quad (10)$$

Subject to:

$$\sum_{k=1}^{r} E_k \le E, \tag{11}$$

$$\sum_{k \in K_i} w_{ik} = w_i \quad i = 1, 2, ..., n, \tag{12}$$

$$w_{ik} \ge 0, \quad T_k \ge 0, \quad i = 1, 2, ..., n; \quad k \in K_i, \tag{13}$$

where T_k are calculated as functions of $\{w_{ik}\}$ for $i \in Z_k$ and E_k using Corollary 2, and (12) guarantee that all jobs will be processed.

Corollary 3

The minimum energy level which ensures the minimum schedule length for a given feasible sequence S and power level P is equal to:

$$E_{\min} = \sum_{k=1}^{r} T_k^* \sum_{i \in Z_k} s_i^{-1}(w_{ik}^*/T_k^*)$$

where T_k^*, w_{ik}^*, $k = 1, 2, ..., p$, $i \in Z_k$ are the optimal values for a given S obtained by solving problem (10), (12), (13) in which values of T_k are calculated for the level of power P from the equations:

$$\sum_{i \in Z_k} s_i^{-1}(w_{ik}/T_k) = P, \quad k = 1, 2, ..., r.$$

Corollary 4

The minimum level of power which ensures the minimum schedule length for a given feasible sequence S and energy level E has the form:

$$P_{\min} = \max_k \left\{ \sum_{i \in Z_k} s_i^{-1}(w_{ik}^*/T_k^*) \right\},$$

where T_k^*, w_{ik}^*, $k = 1, 2, ..., r$, $i \in Z_k$ are the optimal values for a given S obtained by solving problem (10), (11), (12) and (13) in which "=" should be put in (11), and T_k^* are calculated from the equations:

$$T_k \cdot \sum_{i \in Z_k} s_i^{-1}(w_{ik}/T_k) = E_k, \quad k = 1, 2, ..., r.$$

Note that the optimization problem described in Corollary 4 is usually easier to solve than the problem described in Corollary 3 and also easier than the problem (10)–(13). Thus, it is reasonable to start by solving the problem described in Corollary 4. If, in the optimal solution of this problem, we obtain a power amount for each Z_k not greater than P, it means that it is the optimal energy allocation for a given feasible sequence S. In the opposite case, when power for each Z_k is not less than P, we have to solve the problem described in Corollary 3 to find the optimal solution for a given S. Lastly, when the power is greater than P for some Z_k and less than P for the others, we are forced to solve for S the problem (10)–(13).

As a consequence, in order to solve the considered problem it is sufficient to find a feasible sequence which corresponds to the minimal length schedule. Unfortunately, no effective method of constructing such an optimal sequence is yet known. Instead of this, one can utilize some properties of optimal solutions to construct a set of potentially optimal sequences (POS set) and then to solve the problem of energy allocation optimally for all the sequences from this set. Of course, it is necessary to ensure that at least one feasible sequence in POS corresponds to the optimal schedule. Notice, that the size of POS is crucial in the above approach. For the considered scheduling problem the size of POS grows exponentially with the number of jobs. Thus it is justified to apply metaheuristics as a tool for searching for an optimal solution. Unfortunately, the described procedure of optimal energy allocation for given S makes the overall process very time consuming. As an opposite approach a constructive heuristics may be taken into account as an efficient tool for generating reasonably good feasible sequences. In this case a single schedule is built basing on a particular scheduling policy and values of parameters for the problem instance.

6. Final remarks

In this paper we demonstrated usefulness of the model job processing speed vs. power for the power management in systems driven by a common limited power source (e.g. portable electronic devices). It is worth stressing that treating energy as a doubly constrained resource is an important direction for further research in power management, since it is more relevant to real-world situations. The approach presented here describes some properties of time-optimal schedules and outlines ways of finding optimal or suboptimal solutions for the considered problem. For some important cases analytical solutions can be obtained.

Acknowledgements. The research was partially supported by a grant from the State Committee for Scientific Research, Poland.

REFERENCES

[1] D.P. Bunde, "Power-aware scheduling for makespan and flow", *Proc. Eighteenth Annual ACM symposium on Parallelism in Algorithms and Architectures*, 190–196 (2006).

[2] F.R. Boyer, H.G. Epassa, and Y. Savaria, "Embedded power-aware cycle by cycle variable speed processor, computers and digital techniques", *IEE Proc.* 153 (4), 283–290 (2006).

[3] N. Bansal, K. Pruhs, and C. Stein, "Speed scaling for weighted flow time", *ACM/SIAM Symposium on Discrete Algorithms* (SODA), 805–813 (2007).

[4] S. Irani and K. Pruhs, "Algorithmic problems in power management", *ACM SIGACT News*, 36 (2), 63–76 (2005).

[5] K. Pruhs, R. van Stee, and P. Uthaisombut, "Speed scaling of tasks with precedence constraints", *Proc. WAOA*, 307–319 (2005).

[6] N. Bansal, T. Kimbrel, and K. Pruhs, "Dynamic speed scaling to manage energy and temperature", *Lecture Notes in Computer Science* 3404, 460–471 (2005).

[7] J. Węglarz, "Project scheduling with continuously-divisible doubly constrained resources", *Management Science* 27 (9), 1040–1053 (1981).

[8] J. Józefowska and J. Węglarz, "On a methodology for discrete-continuous scheduling problems", *Eur. J. Operational Research* 107 (2), 338–353 (1998).