Sustainable Computing: Informatics and Systems 22 (2019) 134–138

Contents lists available at ScienceDirect

Sustainable Computing: Informatics and Systems

journal homepage: www.elsevier.com/locate/suscom

# Energy and time constrained scheduling for optimized quality of service

Keqin Li

*Department of Computer Science State University of New York New Paltz, New York 12561, USA*

ARTICLE INFO

*Article history:*
Received 1 October 2018
Received in revised form 4 March 2019
Accepted 2 April 2019
Available online 24 April 2019

*Keywords:*
Energy constrained scheduling
Quality of service
Time constrained scheduling
Total completion time
Total energy consumption

ABSTRACT

Minimizing the total completion time has practical application in providing the best quality of service. We consider energy and time constrained task scheduling for minimized total completion time and minimized total energy consumption. We show that there is a polynomial time algorithm to find a nonpreemptive schedule with the minimum total completion time for a given total energy consumption constraint. Similarly, there is a polynomial time algorithm to find a nonpreemptive schedule with the minimum total energy consumption for a given total completion time constraint.



## 1. Introduction

We consider scheduling $n$ independent sequential tasks on $m$ identical processors. Let $C_i$ be the completion time of task $i$. The maximum completion time (i.e., makespan) of a schedule is defined as $M = \max\{C_1, C_2, \ldots, C_n\}$. The total completion time (i.e., the average task response time when it is divided by $n$) of a schedule is defined as $T = C_1 + C_2 + \cdots + C_n$.

The makespan minimization problem is to find a nonpreemptive schedule such that the makespan is minimized. The total completion time minimization problem is to find a nonpreemptive schedule such that the total completion time is minimized. Both problems have practical implications and applications. Typically, from a system administrator's point of view, the makespan to complete a set of tasks should be minimized, since the $m$ processors should be occupied for the least amount of time. However, from a user's point of view, the response time (i.e., the duration of the period from task submission to task completion) is his main concern. This is particularly important in cloud computing, where the best quality of service (QoS) is a main objective in providing cloud services. While there may be many different perspectives of QoS, the average task response time (i.e., the total completion time divided by $n$) is a widely adopted QoS metric, which should be minimized.

For the makespan minimization problem, it is well known that there is a polynomial time approximation scheme (PTAS) [5]. However, the running time of the PTAS is an exponential function of the reciprocal value of the desired precision. For the total completion time minimization problem, there is a polynomial time algorithm to find an optimal solution [9].

Energy-efficient task scheduling (i.e., energy constrained task scheduling) has been studied extensively by many researchers in the last 20 years (see [1,10,12] for surveys of research on energy-efficient scheduling algorithms and scheduling techniques). For the makespan minimization problem, it is well known that there is a PTAS [2,8]. However, the running time of the PTAS is an exponential function of the reciprocal value of the desired precision.

The main contribution of the paper is to show that there is a polynomial time algorithm to find a nonpreemptive schedule with the minimum total completion time for a given total energy consumption constraint. Similarly, there is a polynomial time algorithm to find a nonpreemptive schedule with the minimum total energy consumption for a given total completion time constraint. To the best of the author's knowledge, such results are not available in the existing literature.

The rest of the paper is organized as follows. In Section 2, we define our problems and present our main result. In Section 3, we conduct some analysis. In Section 4, we develop our algorithm. In Section 5, we demonstrate numerical data. In Section 6, we conclude the paper and mention further research directions.

## 2. Preliminaries

Power dissipation and circuit delay in digital CMOS (complementary metal–oxide–semiconductor) circuits can be accurately

*E-mail address:* lik@newpaltz.edu

https://doi.org/10.1016/j.suscom.2019.04.001
2210-5379/

modeled by simple equations, even for complex microprocessor circuits. CMOS circuits have dynamic, static, and short-circuit power dissipation; however, the dominant component in a well designed circuit is dynamic power consumption $p$ (i.e., the switching component of power), which is approximately $p = aCV^2f$, where $a$ is an activity factor, $C$ is the loading capacitance, $V$ is the supply voltage, and $f$ is the clock frequency [3]. Since $s \propto f$, where $s$ is the processor speed, and $f \propto V^\gamma$ with $0 < \gamma \le 1$ [11], which implies that $V \propto f^{1/\gamma}$, we know that power consumption is $p \propto f^\alpha$ and $p \propto s^\alpha$, where $\alpha = 1 + 2/\gamma \ge 3$. We use $p_i$ to represent the power supplied to execute task $i$. For ease of discussion, we will assume that $p_i$ is simply $s_i^\alpha$, where $s_i = p_i^{1/\alpha}$ is the execution speed of task $i$. The execution time of task $i$ is $t_i = r_i/s_i = r_i/p_i^{1/\alpha}$. The energy consumed to execute task $i$ is $e_i = p_i t_i = r_i p_i^{1-1/\alpha} = r_i s_i^{\alpha-1}$.

Given $n$ tasks with execution requirements $r_1, r_2, \ldots, r_n$, $m$ identical processors, and energy constraint $E$, the problem of minimizing total completion time with total energy consumption constraint is to schedule the $n$ tasks on the $m$ processors nonpreemptively such that the total completion time is minimized and that the total energy consumption does not exceed the given constraint.

Given $n$ tasks with execution requirements $r_1, r_2, \ldots, r_n$, $m$ identical processors, and time constraint $T$, the problem of minimizing total energy consumption with total completion time constraint is to schedule the $n$ tasks on the $m$ processors nonpreemptively such that the total energy consumption is minimized and that the total completion time does not exceed the given constraint.

The main result of the paper is the following theorem.

**Theorem 1.** *Given $n$ tasks with execution requirements $r_1 \le r_2 \le \cdots \le r_n$, and $m$ identical processors, where $n = pm + q$, $p = \lfloor n/m \rfloor$, $q = n \bmod m$, the minimized total completion time $T$ and the total energy consumption $E$ satisfy $ET^{\alpha-1} = R^\alpha$, where*

$$R = (p+1)^{(\alpha-1)/\alpha}\sum_{i=1}^{q} r_i + \sum_{k=1}^{p}\left((p+1-k)^{(\alpha-1)/\alpha}\sum_{i=1}^{m} r_{q+(k-1)m+i}\right).$$

$$R = (p+1)^{(\alpha-1)/\alpha}\sum_{i=1}^{q} r_i + \sum_{k=1}^{p}\left((p+1-k)^{(\alpha-1)/\alpha}\sum_{i=1}^{m} r_{q+(k-1)m+i}\right).$$

*Given a total energy consumption constraint $E$, the minimized total completion time is $T = R^{\alpha/(\alpha-1)}/E^{1/(\alpha-1)}$. Given a total completion time constraint $T$, the minimized total energy consumption is $E = R^\alpha/T^{\alpha-1}$. A schedule that realizes the above optimization can be obtained in $O(n \log n)$ time.*

The rest of the paper is devoted to proving the above theorem.

## 3. Analysis

### 3.1. Uniprocessor

We first consider the case when $m = 1$, i.e., task scheduling on a uniprocessor computer. Assume that the $n$ tasks are executed in the order of $1, 2, \ldots, n$. Then, we have $C_i = t_1 + t_2 + \cdots + t_i$, for all $1 \le i \le n$. Hence, we get the total completion time

$$T = t_1 + (t_1 + t_2) + \cdots + (t_1 + t_2 + \cdots + t_n)$$
$$= nt_1 + (n-1)t_2 + \cdots + t_n.$$

In other words, we have

$$T(s_1, s_2, \ldots, s_n) = \sum_{i=1}^{n}(n-i+1)\frac{r_i}{s_i}.$$

The total energy consumption is

$$F(s_1, s_2, \ldots, s_n) = \sum_{i=1}^{n} r_i s_i^{\alpha-1}.$$

Notice that both the total completion time $T(s_1, s_2, \ldots, s_n)$ and total energy consumption $F(s_1, s_2, \ldots, s_n)$ are treated as functions of task execution speeds $s_1, s_2, \ldots, s_n$.

For a given task execution order, we can find the optimal task execution speed setting $(s_1, s_2, \ldots, s_n)$ which minimizes $T(s_1, s_2, \ldots, s_n)$ subject to the constraint

$$F(s_1, s_2, \ldots, s_n) = E,$$

by using the Lagrange multiplier system:

$$\nabla T(s_1, s_2, \ldots, s_n) = \lambda \nabla F(s_1, s_2, \ldots, s_n),$$

where $\lambda$ is a Lagrange multiplier. Since

$$\frac{\partial T(s_1, s_2, \ldots, s_n)}{\partial s_i} = \lambda \frac{\partial F(s_1, s_2, \ldots, s_n)}{\partial s_i},$$

that is,

$$-(n-i+1)\frac{r_i}{s_i^2} = \lambda r_i(\alpha-1)s_i^{\alpha-2},$$

where $\lambda \le 0$, we have

$$s_i = \left(-\frac{n-i+1}{\lambda(\alpha-1)}\right)^{1/\alpha},$$

for all $1 \le i \le n$. Substituting the above $s_i$ into the constraint $F(s_1, s_2, \ldots, s_n) = E$, we get

$$E = \sum_{i=1}^{n} r_i\left(-\frac{n-i+1}{\lambda(\alpha-1)}\right)^{(\alpha-1)/\alpha}.$$

From the last equation, we obtain

$$\left(-\frac{1}{\lambda}\right)^{1/\alpha} = \frac{E^{1/(\alpha-1)}(\alpha-1)^{1/\alpha}}{\left(\sum_{i=1}^{n} r_i(n-i+1)^{(\alpha-1)/\alpha}\right)^{1/(\alpha-1)}},$$

which gives rise to

$$s_i = \frac{(n-i+1)^{1/\alpha}}{\left(\sum_{i=1}^{n} r_i(n-i+1)^{(\alpha-1)/\alpha}\right)^{1/(\alpha-1)}} \cdot E^{1/(\alpha-1)},$$

for all $1 \le i \le n$. Based on the $s_i$'s, we can calculate

$$T = \frac{1}{E^{1/(\alpha-1)}}\left(\sum_{i=1}^{n} r_i(n-i+1)^{(\alpha-1)/\alpha}\right)^{\alpha/(\alpha-1)}.$$

For notational convenience, we write

$$T = \frac{R^{\alpha/(\alpha-1)}}{E^{1/(\alpha-1)}},$$

where

$$R = \sum_{i=1}^{n} r_i(n-i+1)^{(\alpha-1)/\alpha}.$$

It is clear that $T$ is determined by the execution order of the tasks, and $T$ is minimized when $r_1 \le r_2 \le \cdots \le r_n$.

### 3.2. Multiprocessor

Now, we consider the case when $m > 1$, i.e., task scheduling on a multiprocessor system. A schedule of a set $S$ of $n$ tasks on $m$ processors is essentially a partition of $S$ into $m$ disjoint subsets $S_1, S_2, \ldots, S_m$, such that tasks in $S_j$ are executed on processor $j$, where $1 \le j \le m$. Assume that tasks in $S_j$ are numbered as $(j, 1), (j, 2), \ldots, (j, n_j)$, and $r_{j,1} \le r_{j,2} \le \cdots \le r_{j,n_j}$, where $n_j$ is the number of tasks in $S_j$, for all $1 \le j \le m$. Let $T_j$ and $E_j$ denote the total completion time and the total energy consumption of the task in $S_j$. Then, we have

$$T_j = \frac{1}{E_j^{1/(\alpha-1)}}\left(\sum_{i=1}^{n_j} r_{j,i}(n_j - i + 1)^{(\alpha-1)/\alpha}\right)^{\alpha/(\alpha-1)},$$

for all $1 \le j \le m$. Again, for notational convenience, we write

$$T_j = \frac{R_j^{\alpha/(\alpha-1)}}{E_j^{1/(\alpha-1)}},$$

where

$$R_j = \sum_{i=1}^{n_j} r_{j,i}(n_j - i + 1)^{(\alpha-1)/\alpha},$$

for all $1 \le j \le m$. The total completion time of the $n$ tasks is

$$T(E_1, E_2, \ldots, E_m) = \sum_{j=1}^{m} T_j = \sum_{j=1}^{m} \frac{R_j^{\alpha/(\alpha-1)}}{E_j^{1/(\alpha-1)}},$$

where $T(E_1, E_2, \ldots, E_m)$ is viewed as a function of energy consumptions $E_1, E_2, \ldots, E_m$.

For a given schedule $(S_1, S_2, \ldots, S_m)$, we can find the optimal energy allocation $(E_1, E_2, \ldots, E_m)$ to the $m$ processors which minimizes $T(E_1, E_2, \ldots, E_m)$ subject to the constraint

$$F(E_1, E_2, \ldots, E_m) = E_1 + E_2 + \cdots + E_m = E,$$

by using the Lagrange multiplier system:

$$\nabla T(E_1, E_2, \ldots, E_m) = \lambda \nabla F(E_1, E_2, \ldots, E_m),$$

where $\lambda$ is a Lagrange multiplier. Since

$$\frac{\partial T(E_1, E_2, \ldots, E_m)}{\partial E_j} = \lambda \frac{\partial F(E_1, E_2, \ldots, E_m)}{\partial E_j},$$

that is,

$$-\frac{1}{\alpha-1} \cdot \frac{R_j^{\alpha/(\alpha-1)}}{E_j^{\alpha/(\alpha-1)}} = \lambda,$$

where $\lambda \le 0$, we have

$$E_j = \left(-\frac{1}{\lambda(\alpha-1)}\right)^{(\alpha-1)/\alpha} R_j,$$

for all $1 \le j \le m$. Substituting the above $E_j$ into the constraint $F(E_1, E_2, \ldots, E_m) = E$, we get

$$E = \left(-\frac{1}{\lambda(\alpha-1)}\right)^{(\alpha-1)/\alpha} \sum_{j=1}^{m} R_j,$$

From the last equation, we obtain

$$\left(-\frac{1}{\lambda}\right)^{(\alpha-1)/\alpha} = \frac{E(\alpha-1)^{(\alpha-1)/\alpha}}{\sum_{j=1}^{m} R_j},$$

which gives rise to

$$E_j = \frac{R_j}{\sum_{j=1}^{m} R_j} \cdot E,$$

for all $1 \le j \le m$. Based on the $E_j$'s, we can calculate

$$T = \frac{R^{\alpha/(\alpha-1)}}{E^{1/(\alpha-1)}},$$

where

$$R = \sum_{j=1}^{m} R_j.$$

It is clear that $T$ is determined by the schedule $(S_1, S_2, \ldots, S_m)$ of the $n$ tasks on $m$ processors. Once a schedule is available, we are able to find an optimal energy allocation $(E_1, E_2, \ldots, E_m)$ to the $m$ processors and an optimal execution speed setting $(s_1, s_2, \ldots, s_n)$ for all the $n$ tasks, such that $T$ is minimized subject to the constraint that the total energy consumption does not exceed the given limit $E$. Symmetrically, given a schedule, we are also able to find an optimal energy allocation $(E_1, E_2, \ldots, E_m)$ to the $m$ processors and an optimal execution speed setting $(s_1, s_2, \ldots, s_n)$ for all the $n$ tasks, with

$$E = \frac{R^{\alpha}}{T^{\alpha-1}},$$

such that the total completion time does not exceed the given limit $T$.

## 4. Algorithm

Based on the analysis in the last section, we conclude that both the problem of minimizing total completion time with total energy consumption constraint and the problem of minimizing total energy consumption with total completion time constraint can be solved by minimizing

$$R = \sum_{j=1}^{m} R_j = \sum_{j=1}^{m}\sum_{i=1}^{n_j} c_{j,i} r_{j,i},$$

where we call $c_{j,i} = (n_j - i + 1)^{(\alpha-1)/\alpha}$ the coefficient of $r_{j,i}$, for all $1 \le j \le m$, and $1 \le i \le n_j$. Notice that $c_{j,1} > c_{j,2} > \cdots > c_{j,n_j}$, for all $1 \le j \le m$.

In the following, we characterize the properties of any schedule (actually, an optimal schedule) which minimizes $R$.

First, we must have $|n_{j_1} - n_{j_2}| \le 1$, for all $1 \le j_1 \ne j_2 \le m$. Assume that $n_{j_1} - n_{j_2} \ge 2$ for some $j_1$ and $j_2$. We can move $r_{j_1,1}$ from processor $j_1$ to $j_2$ as the first task. This reduces $R_{j_1}$ by $n_{j_1}^{(\alpha-1)/\alpha} r_{j_1,1}$ and increases $R_{j_2}$ by $(n_{j_2}+1)^{(\alpha-1)/\alpha} r_{j_1,1}$. Since $n_{j_1} > n_{j_2} + 1$, $R$ is reduced by $(n_{j_1}^{(\alpha-1)/\alpha} - (n_{j_2}+1)^{(\alpha-1)/\alpha}) r_{j_1,1}$.

Second, we require the smallest task first, i.e., $r_{j,1} \le r_{j,2} \le \cdots \le r_{j,n_j}$, for all $1 \le j \le m$. If $r_{j,i} > r_{j,i+1}$ for some $j$ and $1 \le i \le n_j$, we can exchange $r_{j,i}$ and $r_{j,i+1}$. This reduces $R_j$ (and $R$ as well) by $(c_{j,i} - c_{j,i+1})(r_{j,i} - r_{j,i+1})$.

Third, the above property should be extended across processors, i.e., a larger task cannot have a greater coefficient. For any two tasks $r_{j_1,i_1}$ and $r_{j_2,i_2}$, if $r_{j_1,i_1} < r_{j_2,i_2}$, we must have $c_{j_1,i_1} > c_{j_2,i_2}$. Otherwise, if $c_{j_1,i_1} < c_{j_2,i_2}$, we can exchange $r_{j_1,i_1}$ and $r_{j_2,i_2}$ and reduce $R$ by $(c_{j_2,i_2} - c_{j_1,i_1})(r_{j_2,i_2} - r_{j_1,i_1})$.

Based on the above observations, we can create a schedule with minimized $R$ as follows.

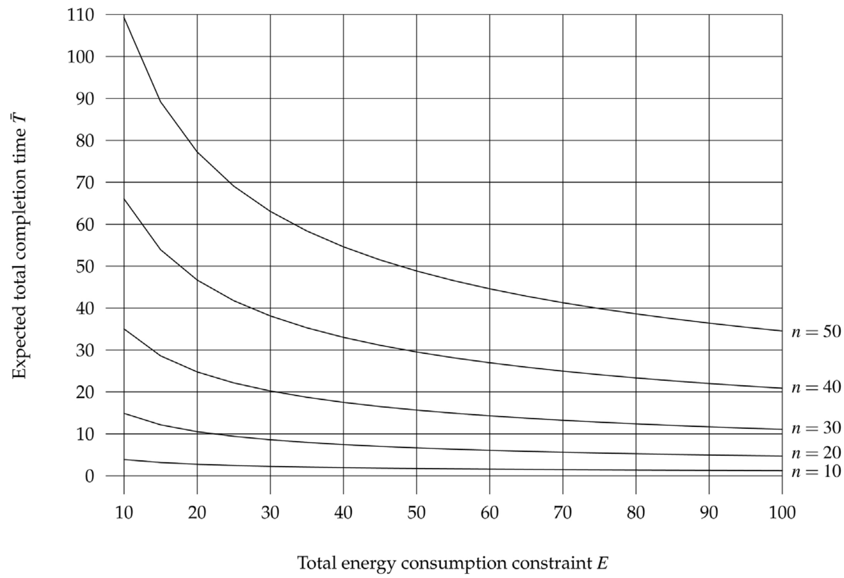


**Fig. 1.** Expected total completion time $\bar{T}$ vs. total energy consumption constraint $E$ (uniform distribution).

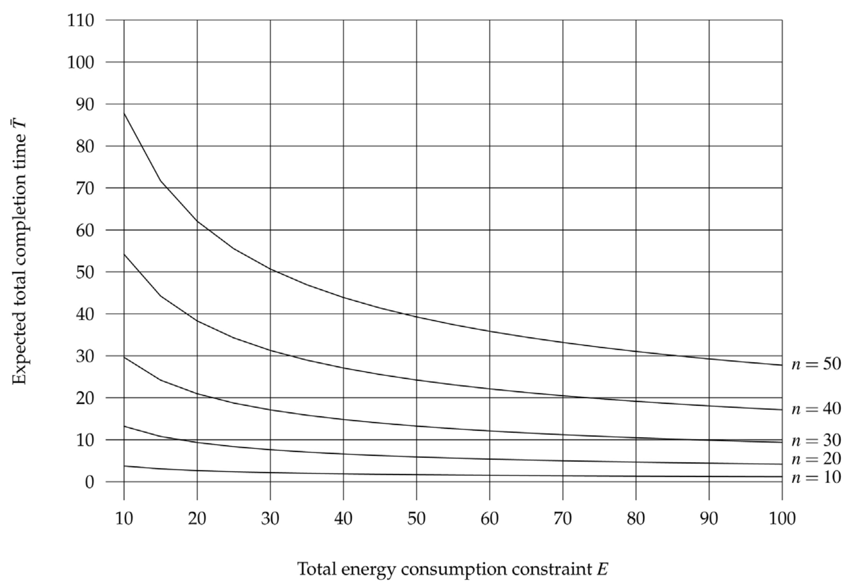


**Fig. 2.** Expected total completion time $\bar{T}$ vs. total energy consumption constraint $E$ (exponential distribution).

- Step (1): Let the $n$ tasks be sorted in such a way that $r_1 \le r_2 \le \cdots \le r_n$.
- Step (2): Assume that $n = pm + q$, where $p = \lfloor n/m \rfloor$, $q = n \bmod m$. The $n$ tasks are divided into $p+1$ batches $B_0, B_1, \ldots, B_p$. Batch $B_0$ contains the first $q$ tasks, i.e., $r_1, r_2, \ldots, r_q$. Batch $B_1$ contains the next $m$ tasks, i.e., $r_{q+1}, r_{q+2}, \ldots, r_{q+m}$. Batch $B_2$ contains the next $m$ tasks, i.e., $r_{q+m+1}, r_{q+m+2}, \ldots, r_{q+2m}$. In general, for $1 \le k \le p$, batch $B_k$ contains tasks $r_{q+(k-1)m+1}, r_{q+(k-1)m+2}, \ldots, r_{q+km}$.
- Step (3): Each processor is assigned $p$ or $p+1$ tasks. Let $c_i = (p+1-i)^{(\alpha-1)/\alpha}$, where $0 \le i \le p$. Tasks in $B_0$ are schedule on any $q$ of the $m$ processors, one task per processor, in any way, with coefficient $c_0$. Tasks in $B_k$ are schedule on the $m$ processors, one task per processor, in any way, with coefficient $c_k$, for all $1 \le k \le p$.

Such a schedule (which is clearly not unique) results in

$$R = (p+1)^{(\alpha-1)/\alpha} \sum_{i=1}^{q} r_i + \sum_{k=1}^{p} \left( (p+1-k)^{(\alpha-1)/\alpha} \sum_{i=1}^{m} r_{q+(k-1)m+i} \right),$$

which is minimized.

It is clear that Step (1) is the most time consuming, which is $O(n \log n)$.

## 5. Numerical data

When $r_1, r_2, \ldots, r_n$ are random variables, $R$ also becomes a random variable. Let $\bar{x}$ denote the expectation of a random variable $x$. Then, we have

$$\bar{R} = (p+1)^{(\alpha-1)/\alpha} \sum_{i=1}^{q} \bar{r}_i + \sum_{k=1}^{p} \left( (p+1-k)^{(\alpha-1)/\alpha} \sum_{i=1}^{m} \bar{r}_{q+(k-1)m+i} \right).$$

Let us assume that $r_1, r_2, \ldots, r_n$ are independent and identically distributed random variables. After $r_1, r_2, \ldots, r_n$ are arranged such that $r_1 \le r_2 \le \cdots \le r_n$, they become order statistics. It is well known that (1) for a uniform distribution in the range [0, 1],

$$\bar{r}_i = \frac{i}{n+1},$$

for all $1 \le i \le n$ ([4], p. 14); (2) for an exponential distribution with mean 0.5,

$$\bar{r}_i = \frac{1}{2}\sum_{j=1}^{i}\frac{1}{n-j+1} = \frac{1}{2}\left(\frac{1}{n}+\frac{1}{n-1}+\cdots+\frac{1}{n-i+1}\right),$$

for all $1 \le i \le n$ [4], p. 18).

When $R$ is a random variable, the total completion time $T$ is also a random variable for a given total energy consumption constraint $E$. In Figs. 1 and 2, we demonstrate the expected total completion time $\bar{T} = \bar{R}^{\alpha/(\alpha-1)}/E^{1/(\alpha-1)}$ vs. the total energy consumption constraint $E$ for the above uniform and exponential distributions respectively, where $n$ = 10, 20, 30, 40, 50. We observe that when $n$ increases, $\bar{T}$ increases noticeably, i.e., more tasks result in much longer average response time.

## 6. Concluding remarks

We have shown that both the problem of minimizing total completion time with total energy consumption constraint and the problem of minimizing total energy consumption with total completion time constraint can be solved in $O(n \log n)$ time.

There are several possible extensions to our work. The first direction is to include static power consumption [7]. The second direction is to study weighted total completion time. The third direction is to consider processors with bounded and discrete and irregular clock frequency and supply voltage and execution speed and power consumption levels [6]. The fourth direction is to investigate online scheduling. All these further research directions have significant theoretical and practical importance.

## Acknowledgments

The author is grateful to the two anonymous reviewers for their constructive comments and suggestions.

## References

[1] S. Albers, Energy-efficient algorithms, Commun. ACM 53 (5) (2010) 86–96.
[2] N. Alon, Y. Azar, G.J. Woeginger, T. Yadid, Approximation schemes for scheduling on parallel machines, J. Schedul. 1 (1) (1998) 55–66.
[3] A.P. Chandrakasan, S. Sheng, R.W. Brodersen, Low-power CMOS digital design, IEEE J. Solid-State Circuit. 27 (4) (1992) 473–484.
[4] H.A. David, H.N. Nagaraja, Order Statistics, 3rd ed., John Wiley & Sons, Inc., Hoboken, New Jersey, 2003.
[5] D.S. Hochbaum, D.B. Shmoys, Using dual approximation algorithms for scheduling problems: theoretical and practical results, J. ACM 34 (1) (1987) 144–162.
[6] K. Li, Experimental study of energy and time constrained task scheduling with irregular speed and power levels, Sustain. Comput.: Inf. Syst. 19 (2018) 61–71.
[7] K. Li, Optimal task execution speed setting for delay and energy minimization, J. Parallel Distrib. Comput. 123 (2019) 13–25.
[8] K. Pruhs, R. van Stee, P. Uthaisombut, Speed scaling of tasks with precedence constraints, Theory Comput. Syst. 43 (2008) 67–80.
[9] W.E. Smith, Various optimizers for single-stage production, Naval Res. Logist. 3 (1-2) (1956) 59–66.
[10] G.L. Valentini, W. Lassonde, S.U. Khan, N. Min-Allah, S.A. Madani, J. Li, L. Zhang, L. Wang, N. Ghani, J. Kolodziej, H. Li, A.Y. Zomaya, C.-Z. Xu, P. Balaji, A. Vishnu, F. Pinel, J.E. Pecero, D. Kliazovich, P. Bouvry, An overview of energy efficiency techniques in cluster computing systems, Cluster Comput. 16 (1) (2013) 3–15.
[11] B. Zhai, D. Blaauw, D. Sylvester, K. Flautner, Theoretical and practical limits of dynamic voltage scaling, Proceedings of the 41st Design Automation Conference (2004) 868–873.
[12] S. Zhuravlev, J.C. Saez, S. Blagodurov, A. Fedorova, M. Prieto, Survey of energy-cognizant scheduling techniques, IEEE Trans. Parallel Distrib. Syst. 24 (7) (2013) 1447–1464.