versão impressa ISSN 0101-7438 / versão online ISSN 1678-5142

A NOTE ON SCHEDULING ON A SINGLE PROCESSOR WITH VARIABLE SPEED

Jorge M. S. Valente *
Rui A. F. S. Alves
Faculdade de Economia
Universidade do Porto
Portugal
jvalente@fep.up.pt

* *Corresponding author*/autor para quem as correspondências devem ser encaminhadas

Recebido em 02/2003; aceito em 09/2003
Received February 2003; accepted September 2003

Abstract

Alidaee and Ahmadian considered a single machine scheduling problem with varying processing times, and presented a polynomial algorithm that minimizes the sum of absolute deviations of jobs' completion times from a common due date. In this short note we remark that it is possible to eliminate one of the algorithm steps, therefore obtaining a more efficient procedure. We also show that the approach used can easily be generalized to the problem with different weights for earliness and tardiness.

Keywords: scheduling; variable speed; common due date.

Resumo

Alidaee e Ahmadian analisaram um problema de sequenciamento com um único processador e tempos de processamento variáveis, tendo apresentado um algoritmo que minimiza a soma dos desvios absolutos dos tempos de finalização das tarefas face a uma data de entrega comum. Neste artigo é estabelecido que um dos procedimentos desse algoritmo pode ser eliminado, sendo assim possível obter um procedimento mais eficiente. A abordagem utilizada é também generalizada ao problema com ponderações diferentes consoante o trabalho é concluído antes ou após a sua data de entrega.

Palavras-chave: sequenciamento; velocidade variável; data de entrega comum.

1. Introduction

Alidaee & Ahmadian (1996) consider a single machine scheduling problem with variable processing times that was introduced by Gawiejnowicz (1996). In this problem the speed of the machine is variable and depends on the number of jobs that have already been processed. The speed of the machine is described by a function v, which varies in the interval [0,1]. The speed starts from an initial value and changes after the completion of each job. After a certain number of jobs have been executed, the speed reaches one of its limit values (either 0 or 1). At that time, the machine stops for a certain amount of time before resuming the processing of the remaining jobs (with the speed v again assuming values in the interval [0,1]).

Formally, the model can be stated as follows. A set $N = \{1,\ldots,n\}$ of independent jobs, each consisting of a single operation, must be processed without preemption on a single machine. Let v: $\{1,\ldots,n\} \rightarrow [0,1]$ be a function that specifies the speed of the machine (at the beginning of the jth job) according to the number of already executed jobs. It is assumed that the speed of the machine does not change while a job is being processed. Let k be the number of jobs that can be executed without breaking the work of the machine, i.e., up to the time the speed reaches an end value. Therefore, the machine must stop after k jobs are executed. Let $p_j > 0$ denote the processing requirement of job j, j=1,…,n, and let t_b be the length of each processor break. Assuming that t_b is constant, we can also assume, without loss of generality, that $t_b = 0$. Let σ = ([1],…,[n]) be a sequence of the n jobs where [l] denotes the lth job in σ and let (i) represent the ith position in the sequence σ. The relative processing time of the jth job in the sequence σ is then $\tilde{p}_{[j]} = p_{[j]} / v(j)$, for j = 1,…,n.

The case where the speed of the machine depends not only on the number of jobs it has already processed, but also on the job currently being processed, was also considered in Alidaee & Ahmadian (1996). In that case, associated with each job $j \in N$ there is a speed function $v_j : \{1,\ldots,n\} \rightarrow [0,1]$, where $v_j(i)$ denotes the speed of the processor when processing job j in the ith position. One of the objective functions considered by Alidaee and Ahmadian was the sum of deviations of the jobs' completion times from a common due date, which can be stated as follows: find a sequence σ and a common due date d, i.e., a schedule (σ,d), that minimizes TET(σ,d) = $\sum_{j=1}^{n} \left| C_{[j]} - d \right|$, the total earliness and tardiness. Alidaee & Ahmadian (1996) presented an O($n^2 \log n$) algorithm for the problem where the speed depends only on the number of previously processed jobs, and a O(n^4) procedure for the case where the speed also depends on the job itself.

In this short note we first remark that one of the steps included in the procedures proposed by Alidaee & Ahmadian (1996) can be eliminated. This leads to improved algorithms with time complexities of O($n \log n$) and O(n^3). We then show that the procedures presented by Alidaee and Ahmadian can easily be extended to the problem with different weights for earliness and tardiness, which can be stated as follows: find a sequence σ and a common due date d, i.e., a schedule (σ,d), that minimizes the total weighted earliness and tardiness TWET(σ,d) = $\sum_{j=1}^{n} \left[h(d - C_j)^+ + w(C_j - d)^+ \right]$, where h and w are, respectively, the earliness and tardiness cost per unit time.

2. Sum of absolute deviations of the jobs' completion times

In this section we remark that it is possible to improve the efficiency of the algorithms proposed by Alidaee & Ahmadian (1996) by eliminating one of their steps. Alidaee & Ahmadian (1996) first presented an algorithm with complexity O($n$log n) for the problem where the speed depends only on the number of previously processed jobs, and the number of early and on time jobs (and therefore the number of late jobs) is given. The problem where once again the number of early and on time jobs is given, but the speed depends also on the job itself, was formulated as a transportation problem, and can therefore be solved in O(n^3) time. Alidaee & Ahmadian (1996) then proposed that the general problems, where the number of early and on time jobs is not given, could be solved by simply running the above algorithms for all the n possible values for the number of early and on time jobs, and choosing the best of those n schedules. This resulted in procedures with complexity O(n^2log n) and O(n^4).

We now remark that there always exists an optimal schedule with a certain number of early and on time jobs, so that only that number needs to be considered in the search for an optimal schedule. It is well known that the due date must coincide with the completion time of a job, and that no unforced idle time should be inserted between the jobs (see, for instance, Baker & Scudder (1990)). Let $\boldsymbol{B}$ denote the set of jobs that complete before or at the common due date d and $\boldsymbol{A}$ denote the set of jobs that complete after d. The following result was proved by Baker & Scudder (1990) and states the number of jobs that will be early or on time in one (possibly not unique) optimal schedule.

Theorem 1 (Baker & Scudder) There exists an optimal schedule where $|\boldsymbol{B}| = \lceil n/2 \rceil$, i.e., $\lceil n/2 \rceil$ jobs are completed early or on time.

Theorem 1 shows that in the search for an optimal solution it suffices to consider schedules with $\lceil n/2 \rceil$ early or on time jobs, so the problems with speed dependent on the number of processed jobs only, and with speed also dependent on the particular job being processed, can therefore be solved in time O($n$log n) and O(n^3), respectively.

3. Weighted earliness and tardiness

3.1 Speed depends only on the number of previously processed jobs

We now show that the approach used in Alidaee & Ahmadian (1996) can be adapted to the problem with a total weighted earliness and tardiness objective function. Once more the due date must coincide with the completion time of a job, and that no unforced idle time should be inserted between the jobs (see, for instance, Baker & Scudder (1990)). We first consider the number of jobs that will be early or on time in an optimal schedule. Again let $\boldsymbol{B}$ denote the set of jobs that complete before or at the common due date d and $\boldsymbol{A}$ denote the set of jobs that complete after d. The following result was also proved by Baker & Scudder (1990), and states the number of jobs that will be early or on time in one (possibly not unique) optimal schedule.

Theorem 2 (Baker & Scudder) There exists an optimal schedule where $|\boldsymbol{B}| = \left\lceil n\frac{w}{h+w} \right\rceil$ and therefore $|\boldsymbol{A}| = \left\lfloor n\frac{h}{h+w} \right\rfloor$, i.e., $\left\lceil n\frac{w}{h+w} \right\rceil$ jobs are completed early or on time and $\left\lfloor n\frac{h}{h+w} \right\rfloor$ jobs are completed late.

Theorem 2 allows us to consider only schedules with $\left\lceil n\frac{w}{h+w} \right\rceil$ early or on time jobs. Please recall that $[l]$ denotes the lth job in the sequence σ, while $v(i)$ represents the speed of the machine when processing the job in the ith position in σ. The objective function can now be written as:

$$\text{TWET}(\sigma,d) = \sum_{j=1}^{n}\left[h(d-C_j)^+ + w(C_j-d)^+\right] =$$

$$= 0\,\tilde{p}_{[1]} + h\tilde{p}_{[2]} + 2h\tilde{p}_{[3]} + \ldots + \left(\left\lceil n\frac{w}{h+w}\right\rceil - 1\right)h\tilde{p}_{\left[\left\lfloor n\frac{w}{h+w}\right\rfloor\right]} +$$

$$+ \left(n - \left\lceil n\frac{w}{h+w}\right\rceil\right)w\tilde{p}_{\left[\left\lfloor n\frac{w}{h+w}\right\rfloor+1\right]} + \left(n - \left\lceil n\frac{w}{h+w}\right\rceil - 1\right)w\tilde{p}_{\left[\left\lfloor n\frac{w}{h+w}\right\rfloor+2\right]} + \ldots +$$

$$+ 3w\tilde{p}_{[n-2]} + 2w\tilde{p}_{[n-1]} + w\tilde{p}_{[n]} =$$

$$= p_{[1]}\frac{0}{v(1)} + p_{[2]}\frac{h}{v(2)} + p_{[3]}\frac{2h}{v(3)} + \ldots + p_{\left[\left\lfloor n\frac{w}{h+w}\right\rfloor\right]}\frac{\left(\left\lceil n\frac{w}{h+w}\right\rceil - 1\right)h}{v\left(\left\lceil n\frac{w}{h+w}\right\rceil\right)} +$$

$$+ p_{\left[\left\lfloor n\frac{w}{h+w}\right\rfloor+1\right]}\frac{\left(n - \left\lceil n\frac{w}{h+w}\right\rceil\right)w}{v\left(\left\lceil n\frac{w}{h+w}\right\rceil + 1\right)} + p_{\left[\left\lfloor n\frac{w}{h+w}\right\rfloor+2\right]}\frac{\left(n - \left\lceil n\frac{w}{h+w}\right\rceil - 1\right)w}{v\left(\left\lceil n\frac{w}{h+w}\right\rceil + 2\right)} + \ldots +$$

$$+ p_{[n-2]}\frac{3w}{v(n-2)} + p_{[n-1]}\frac{2w}{v(n-1)} + p_{[n]}\frac{w}{v(n)},$$

with $d = \sum_{j=1}^{\left\lceil n\frac{w}{h+w}\right\rceil}\tilde{p}_{[j]}$.

Note that only the value of $p_{[j]}$ depends on the job sequence. The remaining values in the previous equations, which we will call the positional weights, and denote by $\Delta(j)$, are independent of the job sequence. These positional weights $\Delta(j)$ are

$$\frac{0}{v(1)}, \frac{h}{v(2)}, \frac{2h}{v(3)}, \ldots, \frac{(\left\lceil n\frac{w}{h+w}\right\rceil - 1)h}{v(\left\lceil n\frac{w}{h+w}\right\rceil)}$$

for the early and on time jobs and

$$\frac{(n-\left\lceil n\frac{w}{h+w}\right\rceil)w}{v(\left\lceil n\frac{w}{h+w}\right\rceil+1)}, \frac{(n-\left\lceil n\frac{w}{h+w}\right\rceil-1)w}{v(\left\lceil n\frac{w}{h+w}\right\rceil+2)}, \ldots, \frac{3w}{v(n-2)}, \frac{2w}{v(n-1)}, \frac{w}{v(n)}$$

for the tardy jobs.

Theorem 3 An optimal solution can be obtained by successively assigning the unscheduled job with the smallest value of p to the free position with the largest value of Δ until all jobs have been scheduled.

Proof. The proof is identical to the proof of Proposition 1 in Alidaee & Ahmadian (1996). The objective function can be seen as a scalar product of two vectors, and it is known that such a product is minimised by matching the largest elements in one vector with the smallest elements in the other vector. ■

The TWET problem can therefore be solved in O($n$log n) time, since sorting the jobs and the positional weights requires O($n$log n) time, and the final assignment of jobs to positions can be done in O(n) time.

3.2 Speed also depends on the job being processed

The approach presented in Alidaee & Ahmadian (1996) for the TET problem when the speed also depends on the job being processed can once again be adapted to the TWET objective function. That approach involves a transformation to an assignment problem. The processing time of a job is variable, since it depends on the job's position in the sequence, while theorem 2 applies when the processing times are constant. However, since for any given sequence σ the processing times are indeed constant, it is clear that even in this case theorem 2 is still valid. Furthermore, it's still true that not only the due date must coincide with the completion time of a job, but also no unforced idle time should be inserted.

The contribution of job j to TWET is $p_j(i-1)\,h/v_j(i)$ if it is scheduled in the ith position and $i \le \left\lceil n\frac{w}{h+w}\right\rceil$, i.e., if j is early or on time. If job j is tardy and it is scheduled in the ith position ($i > \left\lceil n\frac{w}{h+w}\right\rceil$), its contribution will be $p_j(n-i+1)\,w/v_j(i)$. We can now define an

$n \times n$ matrix Q, composed of two matrices Q_e and Q_t as follows,

$$Q = \begin{bmatrix} Q_e \\ Q_t \end{bmatrix},$$

where Q_e is an $\left\lceil n\frac{w}{h+w} \right\rceil \times n$ matrix representing the contributions of the n jobs to the objective function when they are early or on time, and Q_t is an $(n - \left\lceil n\frac{w}{h+w} \right\rceil) \times n$ matrix representing the contributions of the n jobs to TWET when they are tardy. The ijth element of Q_e is equal to $p_j(i-1)\,{}^{h}\!/_{v_j(i)}$ and the ijth element of Q_t is equal to $p_j(n-i+1)\,{}^{w}\!/_{v_j(i)}$.

The TWET problem can then clearly be solved as an assignment problem that has Q as its matrix, and an optimal solution can therefore be obtained in $O(n^3)$ time.

References

(1) Alidaee, B. & Ahmadian, A. (1996). Scheduling on a single processor with variable speed. *Information Processing Letters*, **60**, 189-193.

(2) Baker, K. & Scudder, G. D. (1990). Sequencing with earliness and tardiness penalties: A review. *Operations Research*, **38**, 22-36.

(3) Gawiejnowicz, S. (1996). A note on scheduling on a single processor with speed dependent on a number of executed jobs. *Information Processing Letters*, **57**, 297-300.