Nationaal Lucht- en Ruimtevaartlaboratorium

National Aerospace Laboratory NLR

Executive summary

On scheduling a single machine with resource dependent release times

Problem area
We consider a single machine scheduling problem with resource dependent release times that can be controlled by a non-increasing convex resource consumption function. The objective is to minimize the weighted total resource consumption and sum of job completion times with an initial release time greater than the total processing times. It is known that the problem is polynomially solvable in O(n^4) with n the number of jobs.

Description of work
We describe an improved algorithm of complexity O(n log n). We show further how to perform sensitiviy analysis on the processing time of a job without increasing the complexity.

Applicability
This method could be applied to the arrival scheduling problem where the time to win or lose for an aircraft in an arrival stream is a function of the fuel consumption.

Report no.
NLR-TP-2011-329

Author(s)
M. Janssen
R.J.D. Verbeek
A. Volgenant

Report classification
UNCLASSIFIED

Date
September 2011

Knowledge area(s)
Luchtverkeersmanagement(ATM)-en luchthavenoperaties

Descriptor(s)
Arrivals
Scheduling

This report is based on a paper published in Computers & Operations Research 38, March 2011, by Elsevier.

Nationaal Lucht- en Ruimtevaartlaboratorium
National Aerospace Laboratory NLR

NLR-TP-2011-329

On scheduling a single machine with resource dependent release times

M. Janssen[1], R.J.D. Verbeek and A. Volgenant[1]

[1] UvA

This report is based on a paper published in Computers & Operations Research 38, March 2011, by Elsevier.

Customer	National Aerospace Laboratory NLR
Contract number	----
Owner	NLR + partner(s)
Division NLR	Air Transport
Distribution	Unlimited
Classification of title	Unclassified
	September 2011

Approved by:

Author	Reviewer	Managing department
R.J.D. Verbeek [illegible] 25/7 2011	R. Seljee [illegible] 25/7	C.S. Beers [illegible] 25/7/2011

Summary

We consider a single machine scheduling problem with resource dependent release times that can be controlled by a non-increasing convex resource consumption function. The objective is to minimize the weighted total resource consumption and sum of job completion times with an initial release time greater than the total processing times. It is known that the problem is polynomially solvable in O(n^4) with n the number of jobs.
We describe an improved algorithm of complexity O(n log n). We show further how to perform sensitivity analysis on the processing time of a job without increasing the complexity.

Acknowledgement The authors are grateful to Ron Seljee who gave useful comments on an earlier version of the paper. We thank also two anonymous referees for their remarks.

Contents

Abbreviations

LAP	Linear Assignment Problem
MSP	Machine Scheduling Problem
NP hard	Non-deterministic Polynomial-time hard
CPU	Central Processing Unit
RAM	Random Access Memory

1 Introduction

Machine scheduling problems can have jobs with resource dependent release times that can be controlled by a resource consumption function. In reality the release times can be varied when jobs need to be *pre*processed before they are processed. The preprocessing times can be considered as job release times, and they can depend on the resource consumed for the preprocessing treatment. Choi et al. (2007) described an example in steel plants where ingots must be preheated by gas in soaking pits to the required temperature, before they can be hot-rolled by a blooming mill. The preheating time of an ingot is a non-increasing function of the amount of consumed gas; one can treat this time as the release time at which the ingot is available for the jobs of ingot rolling.

In Air Traffic management we mention arrival scheduling problems of aircraft as another application. In this type of problems the time to win or lose for an aircraft in an arrival stream is a function of the fuel consumption. So it can be efficient to take simultaneously fuel costs into consideration to schedule the release of aircraft to the Terminal Area..

In this paper, we consider a single machine scheduling problem (denoted as MSP) with release times that depend on a non-increasing convex function of consumed resources. The objective is to minimize the weighted total resource consumption and sum of job completion times with an initial release time greater than the total processing times. Choi et al. (2007) introduced a polynomial algorithm that solves the problem by a series of linear assignment problems, if there is a limitation for the initial release time of the jobs.

In the context of the considered problem we define variables $x_{ij} = 1$ if job i is scheduled in position j and 0 otherwise. The mathematical model of the *linear assignment problem* (LAP) is as follows: Given an $n \times n$ matrix $Q = ((q_{ij}))$, determine a solution $x = (x_1, \ldots, x_n)$ such that

$$\text{(LAP)} \qquad \min \sum\nolimits_{i=1}^{n} \sum\nolimits_{j=1}^{n} q_{ij} x_{ij} \text{ (1a)} \qquad \text{(1a)}$$

subject to

$$\sum\nolimits_{i=1}^{n} x_{ij} = 1 \qquad j = 1, \ldots, n \qquad \text{(1b)}$$

$$\sum\nolimits_{j=1}^{n} x_{ij} = 1 \qquad i = 1, \ldots, n \qquad \text{(1c)}$$

$$x_{ij} \in \{0, 1\} \qquad i, j = 1, \ldots, n \qquad \text{(1d)}$$

In section 2 we describe the known algorithm to solve MSP; it has complexity $O(n^4)$ with n the number of jobs. In section 3 we develop our improved algorithm of complexity $O(n \log n)$, followed by sensitivity analysis of a processing time within the same complexity in section 4. We illustrate the algorithm and the sensitivity analysis on an example in section 5. Finally we give some remarks and conclusions.

2 The known algorithm

We first introduce the notation and the problem definition following Choi et al. (2007).

Let J_j for $j = 1, \dots, n$ denote job j having a processing time p_j, a completion time C_j and r_j denote the actual release time. Let v be an initial release time of all the jobs and $f(r_j)$ a resource consumption function of r_j given as

$$f(r_j) = \max\{v - r_{\sigma(j)}, 0\},$$

i.e., it is non-increasing and convex; so it is assumed that each job starts as soon as it is released. Let α and β denote the weight of the total resource consumption function and the sum of the job completion times, respectively. Let $\boldsymbol{\sigma} = \{\sigma(1), ..., \sigma(n)\}$ be a job sequence, where $\sigma(j) = k$ means that job k is positioned jth in the sequence. The problem MSP is defined now as follows.

Given a single machine and n jobs with resource dependent release times, determine $(\boldsymbol{r}, \boldsymbol{\sigma})$ that minimizes $K(\boldsymbol{r}, \boldsymbol{\sigma})$ defined as

$$K(\boldsymbol{r},\boldsymbol{\sigma}) = \alpha\sum\nolimits_{j=1}^{n} \max\{v - r_{\sigma(j)}, 0\} + \beta\sum\nolimits_{j=1}^{n}(r_{\sigma(j)} + p_{\sigma(j)}), \qquad (2)$$

where $v \geq p^{+}$ with p^{+} defined as $p^{+} := \sum\nolimits_{j=1}^{n} p_j$. Clearly the term after β is the sum of the job completion times.

Choi et al. (2007) defined P(k) for $k = 1, \dots, n$, as a constrained version of MSP such that $r(k) = v$ and P(0) is the case where $r(1) = 0$. They proved that P(k) can be solved as a linear assignment problem for each value of k, with the related cost matrices $Q(k) = ((q_{ij}))$ where $p_i = p_{\sigma(j)}$, defined as follows:

problem P(0) has cost matrix $Q(0) = ((q_{ij}))$ with for $i = 1, \dots, n$,

$$q_{ij} = (\beta(n + 1 - j) - \alpha(n - j))p_i, \quad \text{for } j = 1, \dots, n;$$

problem P(k) for $k = 1, \dots, n$ has cost matrix $Q(k) = ((q_{ij}))$ with for $i = 1, \dots, n$,

$$\text{qij} = (\alpha\, \text{j} - \beta(\text{ j} - 1))\text{ pi} \qquad \text{for j} = 1, \dots, \text{k} - 1,$$

$$= \beta(\text{n} - \text{j} + 1)\text{ pi} \qquad \text{for j} = \text{k}, \dots, \text{n}.$$

As a consequence MSP can be solved by a series of $n + 1$ LAP instances under the condition $v \geq p^{+}$, i.e., MSP can be solved by minimizing P(k), for all $k = 0, 1, ..., n$ and taking a best schedule, i.e., a schedule with the lowest criterion value.

Choi et al. (2007) concluded that MSP can be solved in O(n^4), since each P(k) can be solved in O(n^3) by a LAP algorithm, see, e.g., Burkard, Dell'Amico and Martello (2009).

The constant terms in the criteria of P(0) and P(k) ($k \geq 1$) are αnv and βnv, respectively. In the special case of $\beta = 0$ and given $Q(k)$, it is clear that P(1) produces the minimal criterion value $\beta nv = 0$.

In the next section we show how to solve MSP in O($n \log n$).

3 The improved algorithm

We will show that MSP can be solved in a complexity of $O(n \log n)$. We exploit the following known properties of the LAP:

Product LAP:

If a cost matrix C of a (minimization) LAP obeys the *Monge* property, see, e.g., Burkard, Klinz and Rudolf (1996), i.e.,

$$c_{ij} + c_{i+1,j+1} \le c_{i,j+1} + c_{i+1,j} \quad \text{for } i, j = 1, \ldots, n-1,$$

then an optimal schedule is given by $x_{ii} = 1$ for $i = 1, \ldots, n$, i.e., the *north-west corner rule* produces an optimal schedule.

The case that the costs c_{ij} are given as a product $c_{ij} = \omega_i \lambda_j$, $(i, j = 1, \ldots, n)$ obeys the *Monge* property, if the ω- and the λ-values have been sorted (ω_i from small to large and λ_j from large to small) and thus can be solved in $O(n \log n)$, the complexity of the sorting. It is easy to see that the P(k) problems have costs that obey the product form, so one can solve MSP in complexity $O(n^2 \log n)$. We now derive a solution method with a further reduced complexity of $O(n \log n)$.

As the case $\beta = 0$ was already treated in the previous section we assume β to be positive; for ease of presentation and without loss of generality we assume that $\beta = 1$.

We show how to find an optimal value of k – denoted as k^* – that denotes the assignment problem P(k^*) to be solved.

We consider first $k = 0$: The objective function of P(0) is:

$$K_0(\boldsymbol{r}, \boldsymbol{\sigma}) = \alpha n v + \sum_{j=1}^{n} ((1-\alpha)(n-j)+1) p_{\sigma(j)} = \alpha n v + p^+ + \sum_{j=1}^{n} (1-\alpha)(n-j) p_{\sigma(j)}.$$

Secondly $k \ge 1$: The objective function of P(k) is (by convention $\sum_{j=1}^{0} = 0$ and $\sum_{j=n+1}^{n} = 0$):

$$K_k(\boldsymbol{r}, \boldsymbol{\sigma}) = nv + \alpha \sum_{j=1}^{k-1} j p_{\sigma(j)} + \sum_{j=1}^{k-1} (1-j) p_{\sigma(j)} + \sum_{j=k}^{n} (n-j+1) p_{\sigma(j)}.$$

The second sum can be written as:

$$\sum_{j=1}^{k-1} (1-j) p_{\sigma(j)} = \sum_{j=1}^{k-1} (n-j+1) p_{\sigma(j)} - \sum_{j=1}^{k-1} n p_{\sigma(j)}$$

So we find

$$K_k(\boldsymbol{r}, \boldsymbol{\sigma}) = nv + \alpha \sum_{j=1}^{k-1} j p_{\sigma(j)} - n \sum_{j=1}^{k-1} p_{\sigma(j)} + \sum_{j=1}^{n} (n-j+1) p_{\sigma(j)}$$

$$= nv + p^+ + \sum_{j=1}^{k-1} (\alpha j - n) p_{\sigma(j)} + \sum_{j=1}^{n} (n-j) p_{\sigma(j)}.$$

Only the first summation of the right hand side depends on k. As long as $\alpha j \le n$ this summation adds non-positive values to the objective function, so the minimum of $K_k(\boldsymbol{r}, \boldsymbol{\sigma})$ is found for the value $k^* = n$ if $0 \le \alpha \le 1$ and

$$k^* = \arg\max_k \left(\sum_{j=1}^{k-1}(\alpha j - n)p_{\sigma(j)} \,\middle|\, k-1 \le n/\alpha\right) = \lfloor n/\alpha + 1 \rfloor \text{ if } \alpha > 1.$$

An optimal schedule of MSP is found by minimizing P(k) over $k = 0$ and k^*. By subtracting the two related objective function values one can identify the conditions under which $k = 0$ respectively k^* will result in an optimal schedule of MSP. The result of the subtraction $K_0(\boldsymbol{r}, \boldsymbol{\sigma}) - K_{k^*}(\boldsymbol{r}, \boldsymbol{\sigma})$ denoted as $\Delta(k^*)$ is:

$$\Delta(k^*) = (\alpha-1)nv + \sum_{j=1}^{n}\alpha(j-n)p_{\sigma(j)} + \sum_{j=1}^{k^*-1}(n-\alpha j)p_{\sigma(j)}.$$

We distinguish the cases $\alpha \le 1$ and $\alpha > 1$.

Case $\alpha \le 1$: write $\Delta(n)$ as:

$$\Delta(n) = (\alpha-1)nv + \sum_{j=1}^{n}(\alpha(j-n)p_{\sigma(j)} + \sum_{j=1}^{n-1}(n-\alpha j)p_{\sigma(j)}$$

$$= (\alpha-1)nv - n\sum_{j=1}^{n-1}(\alpha-1)p_{\sigma(j)} = (\alpha-1)n\left(v - \sum_{j=1}^{n-1}p_{\sigma(j)}\right).$$

By assumption $v \ge p^+$ while $\alpha \le 1$ and $n \ge 1$ we find that $\Delta(n) \le 0$. Thus it is sufficient to solve the assignment problem P(0).

Case $\alpha > 1$: write $\Delta(k^*)$ as:

$$\Delta(k^*) = (\alpha-1)nv + \sum_{j=1}^{n}\alpha(j-n)p_{\sigma(j)} + \sum_{j=1}^{k^*-1}(n-\alpha j)p_{\sigma(j)}$$

$$= (\alpha-1)nv - (\alpha-1)np^+ + \sum_{j=k^*}^{n}(\alpha j - n)p_{\sigma(j)}$$

$$= (\alpha-1)n(v-p^+) + \sum_{j=k^*}^{n}(\alpha j - n)p_{\sigma(j)} > 0.$$

As $v \ge p^+$ the first term is non-negative and further, for $j \ge k^* = \lfloor n/\alpha + 1 \rfloor$ we have $\alpha j > n$. Thus the last sum is positive and it is sufficient to solve the assignment problem P(k^*).

Theorem MSP can be solved in a complexity of $O(n \log n)$.

Proof

Because P(0) and P(k^*) are LAPs with product form costs, the complexity follows immediately.

The above discussion of the improved MSP algorithm can be summarized in a stepwise description:

MSP algorithm

```
Step 1: if $\beta = 0$ then solve LAP P(1) and stop;
Step 2: $\alpha := \alpha / \beta$;
Step 3: if $\alpha \le 1$ then solve LAP P(0) by the north-west corner rule
Step 4: else solve LAP P($k^*$) with $k^* = \lfloor n/\alpha + 1 \rfloor$ by the north-west corner rule.
```

4 Sensitivity analysis

We show that the improved algorithm enables us to perform sensitivity analysis on a processing time; it can be performed in $O(n \log n)$ for all the values of a processing time.

Without loss of generality we consider the sensitivity on the processing time p_1. Note that the value of k^* is independent of the processing times. We introduce π_1 as the variable value of p_1 in the analysis and $z(\pi_1)$ as the optimal criterion value. Because of the product form costs of the solved LAP we can write

$$z(p_1) = nv + p^+ + \sum_{j=1}^{n} p_j \lambda_j.$$

After sorting the λ-values from large to small we denote these values as $\lambda_{\sigma(j)}$ $(1 \le j \le n)$; then the job sequence $\{\sigma(1), \ldots, \sigma(n)\}$ is an optimal schedule.

We show that $z(\pi_1)$ is a piecewise linear function on the following intervals:

$$0 \le \pi_1 \le p_2,\ p_\ell \le \pi_1 \le p_{\ell+1} \text{ for } \ell = 2, ..., n-1 \text{ and } p_n \le \pi_1 \le v + p_1 - p^+.$$

Note that the last bound of π_1 is a consequence of $v \ge p^+ - p_1 + \pi_1$. As long as $\pi_1 \le p_2$ an optimal schedule remains optimal. The function $z(\pi_1)$ on the interval $0 \le \pi_1 \le p_2$ is found as a linear function in π_1:

$$z(\pi_1) = nv + p^+ + \pi_1 \lambda_{\sigma(1)} + \sum_{j=2}^{n} p_j \lambda_{\sigma(j)}.$$

As soon as the value of π_1 has grown to the value of p_2 the jobs 1 and 2 exchange in the schedule, producing a new optimal schedule and the function $z(\pi_1)$ on the interval $p_2 \le \pi_1 \le p_3$ is found as

$$z(\pi_1) = nv + p^+ + \pi_1 \lambda_{\sigma(2)} + p_2 \lambda_{\sigma(1)} + \sum_{j=3}^{n} p_j \lambda_{\sigma(j)}.$$

In general for $\ell = 3, ..., n$ as soon as the value of π_1 has grown to the value of p_ℓ the place of jobs $\ell - 1$ and ℓ in the schedule are exchanged to obtain a corresponding optimal schedule and the function $z(\pi_1)$ on the interval $p_{\ell-1} \le \pi_1 \le p_\ell$ $(\ell = 3, ..., n)$ is found as

$$z(\pi_1) = nv + p^+ + \pi_1 \lambda_{\sigma(\ell)} + \sum_{j=1}^{\ell-1} p_{j+1} \lambda_{\sigma(j)} + \sum_{j=\ell+1}^{n} p_j \lambda_{\sigma(j)}.$$

Because the $\lambda_{\sigma(j)}$-values are sorted from large to small, the growth of $z(\pi_1)$ diminishes when the value of π_1 goes from an interval to the next interval.

Theorem The complexity of the sensitivity analysis of the processing time of a job is $O(n)$.

Proof

– the z-value on the first interval and a related optimal schedule is computed in $O(n)$;

– each next z-value is obtained by a $O(1)$ operation from the previous value;

– each corresponding optimal schedule is also obtained by a $O(1)$ operation from the previous optimal schedule.

5 Example

We illustrate the improved MSP algorithm by an example on 3 jobs, see Choi et al. (2007), with $v = 10$, $\alpha = 2$, $\beta = 1$, $p_1 = 2$, $p_2 = 1$ and $p_3 = 4$, so $p^+ = 7$ and $nv + p^+ = 37$. As a result $k^* = \lfloor 3 / 2 + 1 \rfloor = 2$, and an optimal schedule follows from solving P(2) by the north west corner rule.

For ease of presentation we renumber the jobs such that $p_1 = 1 < p_2 = 2 < p_3 = 4$. It is easy to check that $\lambda_1 = 1$, $\lambda_2 = 1$ and $\lambda_3 = 0$. Thus an optimal schedule with no intermittent idle time is (J_1, J_2, J_3) with $r^*(1) = 10$. As $\lambda_1 = \lambda_2$ the schedule (J_2, J_1, J_3) is also optimal. In the original indices of the jobs the optimal schedules are (J_2, J_1, J_3) and (J_1, J_2, J_3). To easily compare it with the solution method of Choi et al. (2007), we note that the cost matrix $Q(2) = \begin{pmatrix} 2 & 2 & 1 \\ 4 & 4 & 2 \\ 8 & 8 & 4 \end{pmatrix}$ is after reduction $\begin{pmatrix} 1 & 1 & 0 \\ 2 & 2 & 0 \\ 4 & 4 & 0 \end{pmatrix}$ + reduction sum 7.

We illustrate the sensitivity analysis on p_1:
The optimal schedules can be given on the intervals 0 – 2 ($= p_2$), 2 – 4 ($= p_3$), 4 – 4 (the value 10 of v limits the value of π_1 in the analysis). The values of $z(\pi_1)$ on an interval, say ℓ, are:

$$z(\pi_1) = nv + p^+ + \pi_1\lambda_\ell + \textstyle\sum_{j=1}^{\ell-1} p_{j+1}\lambda_j + \sum_{j=\ell+1}^{n} p_j\lambda_j .$$

The λ-values are $\lambda_1 = 1$, $\lambda_2 = 1$ and $\lambda_3 = 0$. We have

$0 \le \pi_1 \le 2$: $z(\pi_1) = 36 + 2\pi_1 + 1 \times 2 = 2\pi_1 + 38$;

the optimal schedule is (J_1, J_2, J_3);

$2 \le \pi_1 \le 4$: $z(\pi_1) = 36 + \pi_1 + 1 \times 2 + 1 \times \pi_1 = 2\pi_1 + 38$;

the optimal schedule is (J_2, J_1, J_3).

Because $\lambda_1 = \lambda_2$ there are alternative optimal schedules, found by exchanging the first two jobs in the schedules.

It is easy to check that for another job, say job 3, the analysis is:

$0 \le \pi_3 \le 1$: $z(\pi_3) = 33 + \pi_3 + \pi_3 + 1 \times 1 = 2\pi_3 + 34$; the optimal schedule is (J_3, J_1, J_2);

$1 \le \pi_3 \le 2$: $z(\pi_3) = 33 + \pi_3 + \pi_3 + 1 \times 1 = 2\pi_3 + 34$; the optimal schedule is (J_3, J_1, J_2);

$2 \le \pi_3 \le 7$: $z(\pi_3) = 33 + \pi_3 + 1 \times 1 + 1 \times 2 = \pi_3 + 36$; the optimal schedule is (J_1, J_2, J_3).

6 Final remarks and conclusions

We make some remarks and conclusions:

1. We did some computations on problem instances with α = 2, β = 1 and integer *p*-values randomly generated from the interval [1, 10]. We used the LAP algorithm of Jonker and Volgenant (1987) in Delphi Pascal on a personal computer (Intel Core 2 Duo CPU P7350, Windows Vista 64 bit, 4 Gb RAM). We found average computing times of about 1 sec for the Choi et al (2007) algorithm on test instances of 100 jobs and about 0.1 sec for the improved algorithm on instances of 10,000 jobs.

2. Choi et al (2007) introduced also a related problem to (2) defined as (*E* a threshold value)

$$\min \textstyle\sum_{j=1}^{n} f(r_{\sigma(j)}) = \sum_{j=1}^{n} \max\{v - r_{\sigma(j)}, 0\} \tag{3a}$$

subject to

$$\textstyle\sum_{j=1}^{n} (r_{\sigma(j)} + p_{\sigma(j)}) \le E. \tag{3b}$$

They proved that it is NP-hard. If one introduces the Lagrangean function of this problem, with β as the Lagrangean multiplier, then the problem to minimize this function is just the problem (2) with α = 1 and a constant term –β*E*, see, e.g., Ahuja, Magnanti and Orlin (1993). The minimal criterion value is a lower bound for the optimum of (3). In a branch-and-bound algorithm such a bound can be useful, given the low complexity order of the algorithm derived in the previous sections.

3. It is still open whether sensitivity analysis on the value of α can be done in O(*n* log *n*).

We conclude that we can solve MSP in O(*n* log *n*); sensitivity analysis of a processing time can be performed in the same complexity. This complexity can be reduced if one applies the sorting algorithm in O(*n* log log *n*) as described by Han (2004).

References

Ahuja, R.K., Magnanti, J.B., Orlin, J.B. 1993. Network Flows: Theory, Algorithms, and Applications. Prentice Hall.

Burkard, R., Dell'Amico, M., Martello, S. 2009. *Assignment Problems*. Society for Industrial and Applied Mathematics, Philadelphia.

Burkard, R.E., Klinz, B., Rudolf, R. 1996. Perspectives of Monge properties in optimization. Discrete Applied Mathematics 70, 95-161.

Choi, B.-C., Yoon, S.-H., Chung, S.-J. 2007. Single machine scheduling problems with resource dependent release times. Computers & Operations Research 34, 1988-2000.

Han, Y. 2004. Deterministic sorting in O(*n* log log *n*) time and linear space. Journal of Algorithms 50, 96-105.

Jonker, R., Volgenant, A. 1987. A shortest augmenting path algorithm for dense and sparse linear assignment problems. Computing 38, 325-340.