Available online at www.sciencedirect.com

Operations Research Letters 33 (2005) 603–608

Operations Research Letters

www.elsevier.com/locate/orl

The no-wait flow-shop paradox

Frits C.R. Spieksma[a], Gerhard J. Woeginger[b,*]

[a]Department of Applied Economics, Katholieke Universiteit Leuven, Naamsestraat 69, B-3000 Leuven, Belgium
[b]Department of Mathematics and Computer Science, TU Eindhoven, P.O. Box 513, 5600 MB Eindhoven, The Netherlands

Received 29 September 2004; accepted 31 October 2004
Available online 18 December 2004

Abstract

We discuss a new resource paradox in the area of scheduling: Increasing the speed of some machines in a no-wait flow-shop instance may actually *worsen* the optimal makespan. We construct instances for which the ratio between optimal makespan with improved speed and optimal makespan without improved speed becomes arbitrarily bad.
© 2004 Published by Elsevier B.V.

Keywords: Resource paradox; Scheduling; Flow-shop problem

1. Introduction

1.1. Resource paradoxes

The areas of logistics, production, and planning contain a number of strange anomalies and counter-intuitive paradoxes. In most of these paradoxes, improving the resources of a complex system leads to a degradation of the overall system performance:

- The transportation paradox [1,4,8] demonstrates that decreasing the demands and supplies in a transportation problem might actually yield an increase of the overall transportation cost.
- The Braess paradox [3] shows that by adding new and faster streets to a traffic network, one might actually decrease the throughput of this network in equilibrium state. One variant of the Braess paradox shows up when new connections are added to the internet [7].
- Belady's anomaly [2] concerns memory management systems where the cache (the fast memory) is run under the first-in–first-out page replacement policy. Increasing the size of the cache may actually yield a higher number of page faults for the same sequence of page requests.
- Graham's multiprocessing anomaly [5] concerns parallel machine systems where precedence-constrained processes are assigned according to the list-scheduling policy. By adding new machines to the system, one might actually worsen the makespan.

* Corresponding author.
E-mail addresses: frits.spieksma@econ.kuleuven.ac.be (F.C.R. Spieksma), gwoegi@win.tue.nl (G.J. Woeginger).

0167-6377/$ - see front matter © 2004 Published by Elsevier B.V.
doi:10.1016/j.orl.2004.10.007

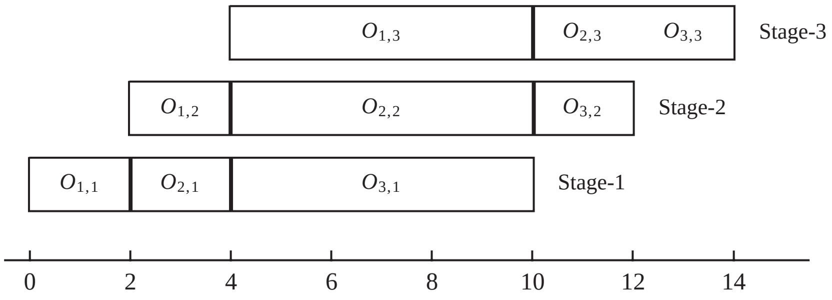

Fig. 1. The optimal schedule for J_1, J_2, J_3. By $O_{j,i}$ we denote the part of job J_j that is processed in stage i.

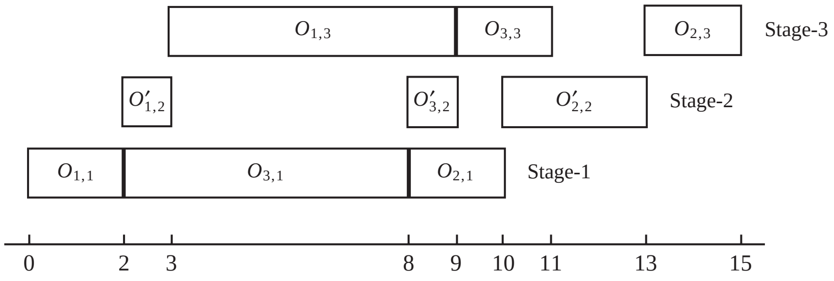

Fig. 2. An optimal schedule for J_1, J_2, J_3 when the second stage works at double speed.

Note that in all these examples (and throughout this paper), the word ‘paradox’ is used to denote a “*startling, perplexing, and mind-boggling situation*”, but not to denote a classical paradox in the context of formal logics.

1.2. The flow-shop paradox

This note adds another paradox to the list of resource paradoxes, the flow-shop paradox. The simplest version of this paradox consists of three jobs J_1, J_2, J_3 in a three-stage production system. Since there is no intermediate storage between the stages, every job first enters the system, then goes through the first stage, then immediately moves on to the second stage, then immediately moves on to the third stage, and finally leaves the system. Each of the three jobs spends a total of 10 time units in the system: Job J_1 spends, respectively, $2, 2, 6$ time units in the three stages, job J_2 spends $2, 6, 2$ time units, and job J_3 spends $6, 2, 2$ time units. At any moment in time, at most one job can be processed in any fixed stage. All jobs are available from time 0 onwards. The objective is to find an ordering of the jobs that minimizes the makespan, that is, the maximum job completion time. In the scheduling literature, this problem is called *makespan minimization in a no-wait* 3*-stage flow-shop*. By processing the jobs in the ordering $\langle J_1, J_2, J_3\rangle$ one reaches a makespan of 14, and it is easily verified that 14 is the optimal makespan; see Fig. 1.

Here is the paradox: Suppose that we *increase* the processing power in the second stage, so that the processing times in this stage all go down by a factor of two. Then the new processing times of the three jobs become, respectively, $2, 1, 6$, and $2, 3, 2$, and $6, 1, 2$. The optimal makespan, however, then *deteriorates* from 14 to 15: All in all, there are six possible schedules. If we process the jobs in one of the three orderings $\langle J_1, J_2, J_3\rangle$, $\langle J_1, J_3, J_2\rangle$, and $\langle J_2, J_1, J_3\rangle$, then we get a makespan of 15; see Fig. 2 for one possible schedule. If we process the jobs in one of the two orderings $\langle J_2, J_3, J_1\rangle$ and $\langle J_3, J_1, J_2\rangle$, then we get a makespan of 17. And if we process the jobs in the ordering $\langle J_3, J_2, J_1\rangle$, then we get a makespan of 19.

1.3. Variations of the flow-shop paradox

In a variant of the flow-shop paradox, the optimal makespan increases as a consequence of decreasing the processing times of *one* job in *all* stages by the same factor: Consider the above basic instance with three jobs and optimal makespan 14, and reduce the processing times 2, 6, 2 of job J_2 all by 25%. In the resulting instance with processing times 2, 2, 6, and 1.5, 4.5, 1.5, and 6, 2, 2, the optimal makespan has increased to 15.5.

In another variant, the optimal makespan increases as a consequence of decreasing the processing time of *one* job in *one* stage: Consider the above basic instance with three jobs and optimal makespan 14. If we decrease the processing time of the old job J_2 in the second stage from 6 to 3 (so that the new instance has processing times 2, 2, 6, and 2, 3, 2, and 6, 2, 2), then the optimal makespan increases from 14 to 15.

2. Formal definitions and the main result

In a general m-stage no-wait flow-shop system, there are n jobs $J_1, \dots, J_n$. Every job J_j is a chain $O_{j,1}, \dots, O_{j,m}$ of m operations that have to be processed in the m stages; the processing time of operation $O_{j,i}$ (of job J_j in stage i) equals $p_{i,j}$. All jobs are available from time 0 onwards, and at any moment in time at most one job can be processed in any fixed stage. Every job J_j must move through the stages without intermediate waiting time. The objective is to find a schedule that minimizes the makespan. In the standard scheduling notation, this problem is denoted as $F\,|\,\text{no-wait}\,|\,C_{\max}$. The flow-shop paradox occurs, if by speeding up the processing in some of the stages the optimal makespan increases.

It is easy to see that the flow-shop paradox cannot occur in no-wait flow-shop instances with $m=2$ stages (and an arbitrary number of jobs), and that this paradox also cannot occur if there are only $n=2$ jobs (and an arbitrary number of stages). The basic instance discussed in the introduction has $m=3$ stages and $n=3$ jobs, and thus yields a minimum size example for the paradox.

In the instance in the introduction, the paradox increases the optimal makespan by a factor of 15/14, from 14 up to 15. The main result of this paper shows that this factor may become arbitrarily large.

Theorem 1. *For every real number $r \geqslant 1$, there exists an instance of the no-wait flow-shop problem $F\,|\,\text{no-wait}\,|\,C_{\max}$ with optimal makespan C^*, and a certain way of speeding up the processing in some of the stages, such that in the resulting instance with faster machines the optimal makespan is at least $r\,C^*$.*

3. The proof

In this section, we will prove Theorem 1. Let $z \geqslant 2$ be an integer. Consider the following instance I' with $m=z^2$ stages and $n=z^2$ jobs: Job J'_j $(j=1,\dots,n)$ has a processing time of $p'_{j,j}=z$ time units in the jth stage, and a processing time of 0 time units in all the other stages. (The processing times of 0 time units can be simulated by processing times of ε time units, and by having ε go to 0. Hence, these operations of length 0 block the corresponding stage for an infinitesimally small amount of time, and do not allow the simultaneous processing of another job in the same stage.) The optimal schedule for instance I' processes the jobs in the order of decreasing index, and the optimal makespan equals z. Next, consider the following instance I that results from instance I' by speeding up the processing in the stages such that job J_j $(j=1,\dots,n)$ has a processing time of

$$q_j \in \{1, 2, \dots, z\} \quad \text{with } q_j = j \,(\text{mod}\, z)$$

in the jth stage, and a processing time of 0 time units in all the other stages. Note that the lengths of the non-zero operations of jobs $J_1, \dots, J_n$ form a periodic sequence of the form $1, 2, \dots, z, 1, 2, \dots, z, \dots, 1, 2, \dots, z$.

In the rest of this section, we will show that the optimal makespan of instance I is at least $\frac{1}{2}z(z+1)$. As a consequence, the ratio between the optimal makespan of I and the optimal makespan of I' is at least $(z+1)/2$. Since z can be made arbitrarily large, this will yield Theorem 1. We start with a small observation that essentially restates the standard TSP-formulation of the no-wait flow-shop problem; see for instance [9]:

Lemma 1. *If the processing of job J_k starts at time t, and if job J_ℓ is processed immediately after job J_k,* *then the starting time of J_ℓ equals $t + d(k,\ell)$ where*

$$d(k,\ell) = \begin{cases} q_k & \text{if } k < \ell, \\ \max\{0, q_k - q_\ell\} & \text{if } k > \ell, \\ \infty & \text{if } k = \ell. \end{cases} \tag{1}$$

The optimal makespan of the no-wait flow-shop instance I equals the length of the shortest travelling salesman tour through the cities $1, \ldots, n$ with distances $d(k,\ell)$.

The linear assignment relaxation (2) of the travelling salesman problem provides a lower bound on the length of the shortest TSP tour:

$$\begin{array}{lll} \text{minimize} & \displaystyle\sum_{k=1}^{n}\sum_{\ell=1}^{n} d(k,\ell)x_{k,\ell} & \\ \text{such that} & \displaystyle\sum_{\ell=1}^{n} x_{k,\ell} = 1 & \text{for } k = 1,\ldots,n, \\ & \displaystyle\sum_{k=1}^{n} x_{k,\ell} = 1 & \text{for } \ell = 1,\ldots,n, \\ & x_{k,\ell} \geqslant 0 & \text{for } k = 1,\ldots,n, \\ & & \quad \ell = 1,\ldots,n. \end{array} \tag{2}$$

The dual linear program for (2) is given by

$$\begin{array}{lll} \text{maximize} & \displaystyle\sum_{k=1}^{n} u_k + \sum_{\ell=1}^{n} v_\ell & \\ \text{such that} & u_k + v_\ell \leqslant d(k,\ell) & \text{for } k = 1,\ldots,n, \\ & & \quad \ell = 1,\ldots,n, \\ & u_k, v_\ell \text{ real} & \text{for } k = 1,\ldots,n, \\ & & \quad \ell = 1,\ldots,n. \end{array} \tag{3}$$

We will now define and investigate one particular feasible solution of the dual linear program (3). For fixed k and ℓ with $1 \leqslant k \leqslant n = z^2$ and $1 \leqslant \ell \leqslant n = z^2$, let α and β be the unique integers that satisfy

$$k = \alpha z + \beta \quad \text{with } 0 \leqslant \alpha \leqslant z-1 \text{ and } 1 \leqslant \beta \leqslant z \tag{4}$$

and let γ and δ be the unique integers that satisfy

$$\ell = \gamma z + \delta \quad \text{with } 0 \leqslant \gamma \leqslant z-1 \text{ and } 1 \leqslant \delta \leqslant z. \tag{5}$$

We define

$$u_k^* = \min\{\beta, z-\alpha\},$$

$$v_\ell^* = \begin{cases} -\min\{\delta, z-\gamma-1\} & \text{if } \gamma \leqslant z-2, \\ 0 & \text{if } \gamma = z-1. \end{cases}$$

Note that $q_k = \beta$ and that $q_\ell = \delta$.

Lemma 2. *The values u_k^* and v_ℓ^* constitute a feasible solution for* (3).

Proof. We must show that $u_k^* + v_\ell^* \leqslant d(k,\ell)$ holds for all k and ℓ. Since $d(k,k) = \infty$, this inequality trivially holds for $k = \ell$. For $k \neq \ell$, we distinguish four cases.

- In the first case, $k < \ell$ holds. Then $d(k,\ell) = q_k = \beta$. Since $u_k^* = \min\{\beta, z-\alpha\} \leqslant \beta$ and $v_\ell^* \leqslant 0$, we get the desired inequality $u_k^* + v_\ell^* \leqslant d(k,\ell)$.
- In the second case, $k > \ell$ and $\gamma = z-1$. Then (4) and (5) imply $\alpha = z-1$ and $\beta > \delta$. In this case $u_k^* = \min\{\beta, 1\} = 1$ and $v_\ell^* = 0$, and the desired inequality follows from

$$\begin{aligned} u_k^* + v_\ell^* &= 1 \leqslant \beta - \delta \\ &= \max\{0, q_k - q_\ell\} = d(k,\ell). \end{aligned}$$

- In the third case, $k > \ell$, and $\gamma \leqslant z-2$, and $\beta \leqslant \delta$. Then (4) and (5) imply $\alpha \geqslant \gamma + 1$. Under these conditions we have $d(k,\ell) = \max\{0, \beta - \delta\} = 0$. Then the desired inequality is equivalent to $u_k^* \leqslant -v_\ell^*$, which is

$$\min\{\beta, z-\alpha\} \leqslant \min\{\delta, z-\gamma-1\}.$$

The displayed inequality follows from $\beta \leqslant \delta$ and $z - \alpha \leqslant z - \gamma - 1$.
- In the fourth case, $k > \ell$, and $\gamma \leqslant z-2$, and $\beta > \delta$. This implies $\alpha \geqslant \gamma$. Under these conditions we have $d(k,\ell) = \max\{0, \beta-\delta\} = \beta - \delta$. Furthermore, $u_k^* = \min\{\beta, z-\alpha\} \leqslant \min\{\beta, z-\gamma\}$, and $v_\ell^* = -\min\{\delta, z-\gamma-1\}$. If $\delta \leqslant z-\gamma-1$, then

$$\begin{aligned} u_k^* + v_\ell^* &\leqslant \min\{\beta, z-\gamma\} - \min\{\delta, z-\gamma-1\} \\ &\leqslant \beta - \delta = d(k,\ell). \end{aligned}$$

If $\delta \geqslant z - \gamma$, then $\beta > z - \gamma$, and

$$\begin{aligned} u_k^* + v_\ell^* &\leqslant \min\{\beta, z-\gamma\} - \min\{\delta, z-\gamma-1\} \\ &= (z-\gamma) - (z-\gamma-1) \\ &= 1 \leqslant \beta - \delta = d(k,\ell). \end{aligned}$$

Summarizing, in all four cases we ended up with the desired inequality $u_k^* + v_\ell^* \leqslant d(k,\ell)$. □

Next, let us evaluate the objective value of the feasible solution u_k^* and v_ℓ^* for (3). Note that for $\ell = 1, \ldots, z^2 - z$ we have $u_{\ell+z}^* + v_\ell^* = 0$, and that for

$\ell = z^2 - z + 1, \ldots, z^2$ we have $v_\ell^* = 0$. Therefore,

$$\begin{aligned}\sum_{k=1}^{n} u_k^* + \sum_{\ell=1}^{n} v_\ell^* &= \sum_{k=1}^{z} u_k^* + \sum_{\ell=1}^{z^2-z} (u_{\ell+z}^* + v_\ell^*) + \sum_{\ell=z^2-z+1}^{z^2} v_\ell^* \\ &= \sum_{k=1}^{z} u_k^* = \sum_{k=1}^{z} \min\{k, z\} = \frac{1}{2} z(z+1).\end{aligned}$$

The optimal makespan of the no-wait flow-shop instance I equals the length of the shortest TSP tour with distances (1). The length of the shortest TSP tour is at least the optimal objective value of the LP relaxation (2). By the fundamental theorem of linear programming, the optimal objective value of the primal LP (2) is greater or equal to the objective value of the feasible solution u_k^* and v_ℓ^* for the dual LP (3). Hence, it is at least $\frac{1}{2}z(z+1)$. This completes the proof of Theorem 1. □

4. Discussion

The construction of the bad instance in Section 3 used *non-uniform* speed-ups for the various stages; for instance, the machine in stage j with $1 \leqslant j \leqslant z$ became faster by a machine-dependent factor of z/j. In case *all* stages receive the same speed-up by the same factor f, then of course also the optimal makespan improves by this factor f, and the flow-shop paradox cannot occur. What about the case where *some* stages get the same speed-up f, whereas the remaining stages remain unchanged? (This would correspond to the natural setting where a subset of the machines in the flow-shop is replaced by a new and better generation of machines.)

Our construction in Section 3 can be slightly modified and changed so that it also yields bad instances in this uniform setting: Each of the z^2 stages in the old construction is replaced by a corresponding sequence of z consecutive stages in the new construction; these z new stages will simulate the corresponding old stage. Every old job J_j' (that had a processing time of z in the jth old stage and a processing time of 0 time units in all the other old stages) is replaced by a new job J_j' that has processing times of 1 in the z new stages that correspond to the jth old stage and processing times of 0 in all the other new stages. If an old stage in the old construction is sped up by a factor of z/j, then in the new construction $z - j$ of the new corresponding stages get an infinite speed-up, whereas the j remaining new corresponding stages get no speed-up at all. We leave all further details to the reader.

The computational complexity of the following problem FLOW-SHOP-PARADOX remains unclear: "*Given an instance I of the no-wait flow-shop problem* $F \mid \text{no-wait} \mid C_{\max}$, *does there exist a way of speeding up the processing in some* (*arbitrarily chosen*) *subset of the stages, such that the optimal makespan of the resulting instance* I' *with faster machines is strictly larger than the optimal makespan of instance I*?" There is absolutely no reason to assume that problem FLOW-SHOP-PARADOX is contained in the class NP: To demonstrate the occurrence of the paradox in a straightforward way, we have to show that the optimal makespan of I is small (NP-certificate), we have to show that an appropriate instance I' with faster machines exists (NP-certificate), and we have to show that the optimal makespan of instance I' is huge (coNP-certificate). This mixture of NP- and coNP-certificates suggests that FLOW-SHOP-PARADOX might be located in one of the complexity classes above NP (see for instance Chapter 17 in Papadimitriou's book [6]); the complexity class DP might be a reasonable guess.

Acknowledgements

The authors thank Han Hoogeveen for a number of useful comments.

References

[1] G.M. Appa, The transportation problem and its variants, Oper. Res. Quart. 24 (1973) 79–99.
[2] L.A. Belady, R.A. Nelson, G.S. Shedler, An anomaly in space-time characteristics of certain programs running in a paging machine, Commun. ACM 12 (1969) 349–353.
[3] D. Braess, Über ein paradoxon aus der verkehrsplanung, Unternehmensforschung 12 (1968) 258–268.
[4] A. Charnes, D. Klingman, The more-for-less paradox in the distribution model, Cah. Cent. d'Etud. Rech. Oper. 13 (1971) 11–22.
[5] R.L. Graham, Bounds on multiprocessing timing anomalies, SIAM J. Appl. Math. 17 (1969) 416–429.

[6] C.H. Papadimitriou, Computational Complexity, Addison-Wesley, Reading, MA, 1994.
[7] T. Roughgarden, E. Tardos, How bad is selfish routing?, J. ACM 49 (2002) 407–433.
[8] W. Szwarc, The transportation paradox, Nav. Res. Logist. Quart. 18 (1971) 185–202.
[9] D.A. Wismer, Solution of the flowshop scheduling problem with no intermediate queues, Oper. Res. 20 (1972) 689–697.