Reports in
operations research
and systems theory

The complexity of scheduling short tasks with few starting times

Frits C.R. Spieksma and Yves Crama

Report M 92-06

University of Limburg
Mathematics
PO Box 616
6200 MD Maastricht
The Netherlands

The complexity of scheduling short tasks with few starting times

Frits C.R. Spieksma*and Yves Crama†

December 1992

Abstract

The following problem is proved to be $\mathcal{NP}$-complete: given n tasks, such that each task has processing time $\tau = 2$, and has no more than $k = 3$ possible starting times, does there exist a feasible schedule for these tasks on a single processor? This result establishes a sharp borderline between $\mathcal{NP}$-complete and polynomially solvable versions of this problem with respect to the parameters τ and k.

Keywords: scheduling, complexity, discrete starting times.

*Department of Mathematics, University of Limburg, P.O. Box 616, 6200 MD Maastricht, The Netherlands (to whom all correspondence should be addressed).

†Department of Quantitative Economics, University of Limburg, P.O. Box 616, 6200 MD Maastricht, The Netherlands.

Acknowledgements. The second author was partially supported in the course of this research by grants AFOSR-89-0512B and F49620-93-1-0041 to Rutgers University.

1 Introduction

Consider the following problem. Given are n tasks T_i, $i = 1, \ldots, n$, which have to be scheduled on a single processor without preemption. The processing time of each task equals τ_i, $\tau_i \in \mathbb{N}$, and each task T_i is only allowed to start for execution at a time-unit from a given set of possible starting times $S_i = \{s_{i1}, s_{i2}, \ldots, s_{ik_i}\}$, $k_i \in \mathbb{N}$, for $i = 1, \ldots, n$. The problem is to decide whether there exists a feasible schedule for all tasks. We will refer to this problem as problem P.

Applications of this problem are mentioned in Nakajima and Hakimi [5] in the area of computation in real-time environments. Another source of instances of problem P can be found in the manufacturing of printed circuit boards: in Crama and Spieksma [1] it is described how the assignment of so-called feeders to feederslots gives rise to instances of an optimization version of problem P.

The complexity of problem P (see Garey and Johnson [2] for an introduction to complexity theory) depends on assumptions on the parameters $k \stackrel{\text{def}}{=} \max_i k_i$ and τ_i, $i = 1, \ldots, n$. Concerning the parameter k, Nakajima and Hakimi [5] proved that problem P is $\mathcal{NP}$-complete for $k = 3$, even when $\tau_i \in \{\tau_1, \tau_2\}$, $\tau_1, \tau_2 \in \mathbb{N}$, for $i = 1, \ldots, n$. These authors also asked what happens when all tasks have the same processing time, say $\tau_i = \tau$ for $i = 1, \ldots, n$. Recently, Keil [4] answered this question by showing that problem P is $\mathcal{NP}$-complete when $k = 3$ and $\tau = 3$. Moreover, Keil [4] also gave a polynomial-time algorithm for the case $k = 2$ and arbitrary τ_i, $i = 1, \ldots, n$. Regarding the parameter τ, problem P was shown to be $\mathcal{NP}$-complete for $\tau = 2$ and arbitrary k_i, $i = 1, \ldots, n$, in Crama and Spieksma [1]. Also, it is readily verified that for $\tau = 1$ and arbitrary k_i, $i = 1, \ldots, n$, problem P reduces to bipartite matching, and hence is solvable in polynomial time.

In this paper we consider the complexity of the 'remaining' case, that is problem P restricted to instances with $\tau = 2$ and $k = 3$. In Section 2 we show that this problem is $\mathcal{NP}$-complete, thereby generalizing the results of [1,2], and establishing the borderline between $\mathcal{NP}$-complete and polynomially solvable versions of the problem with respect to k and τ.

2 The result

Theorem *Problem P is strongly $\mathcal{NP}$-complete, even when $k = 3$ and $\tau = 2$.*

Proof Evidently, problem P belongs to the class $\mathcal{NP}$. We use a reduction from a special case of the strongly $\mathcal{NP}$-complete 3-dimensional matching (3DM) problem (Karp [3]). An instance of 3DM is given by three mutually disjoint sets $A_1 = \{a_1, a_2, \ldots, a_q\}$, $A_2 = \{a_{q+1}, a_{q+2}, \ldots, a_{2q}\}$ and $A_3 = \{a_{2q+1}, a_{2q+2}, \ldots, a_{3q}\}$ and a set $R \subseteq A_1 \times A_2 \times A_3$ with $|R| = m$. Let an element r of R be called a triple r, $1 \leq r \leq m$, and let l_i denote the number of occurrences of element a_i in triples of R, $i = 1, \ldots, 3q$. Also, let the triples $r \in R$ containing a_i be indexed $r_1^{a_i}, r_2^{a_i}, \ldots, r_{l_i}^{a_i}$ for $i = 1, \ldots, 3q$. An instance of 3DM is called feasible if and only if there exists a set $M \subseteq R$ such that each element of $A_1 \cup A_2 \cup A_3$ occurs exactly once in a triple of M. This problem remains $\mathcal{NP}$-complete if no element of $A_1 \cup A_2 \cup A_3$ occurs in more than three triples of R, that is $l_i \leq 3$ for $i = 1, \ldots, 3q$ (see

Garey and Johnson, page 221 [2]). Given an instance I of 3DM satisfying this requirement, we now construct an instance $\mathcal{T}$ of problem P with $\tau = 2$ and $k = 3$.

For each element $a_i \in A_1 \cup A_2 \cup A_3$, $1 \leq i \leq 3q$, we construct l_i tasks. One of these l_i tasks is called the odd task, the remaining $l_i - 1$ tasks are called even tasks. (Notice that there may be either 0, 1 or 2 even tasks associated with each element a_i, $1 \leq i \leq 3q$). Hence $n = \sum_{i=1}^{3q} l_i$. With each triple $r \in R$ we associate 7 consecutive time-units $7(r-1)+1, 7(r-1)+2, \ldots, 7(r-1)+7$, for $r = 1, \ldots, m$, referred to as block r. Now, the sets of possible starting times for each task are defined as follows.

Let S_i^{odd} denote the set of possible starting times for the odd task associated with element a_i, $1 \leq i \leq 3q$; and let S_i^{even} denote the set of possible starting times for all even tasks associated with element a_i (all $l_i - 1$ even tasks associated with a_i will have identical sets of possible starting times). We have:

$$
\begin{aligned}
S_i^{\text{odd}} &= \{7(r_1^{a_i} - 1) + 1, \ldots, 7(r_{l_i}^{a_i} - 1) + 1\} \text{ for } i = 1, \ldots, q, \\
S_i^{\text{odd}} &= \{7(r_1^{a_i} - 1) + 3, \ldots, 7(r_{l_i}^{a_i} - 1) + 3\} \text{ for } i = q+1, \ldots, 2q, \\
S_i^{\text{odd}} &= \{7(r_1^{a_i} - 1) + 5, \ldots, 7(r_{l_i}^{a_i} - 1) + 5\} \text{ for } i = 2q+1, \ldots, 3q.
\end{aligned}
$$

The set of possible starting times for each of the $l_i - 1$ even tasks associated with a_i is formed by the time-units which directly follow the time-units in S_i^{odd}, $i = 1, \ldots, 3q$. More formally, the set of possible starting times for each even task associated with a_i is:

$$
\begin{aligned}
S_i^{\text{even}} &= \{7(r_1^{a_i} - 1) + 2, \ldots, 7(r_{l_i}^{a_i} - 1) + 2\} \text{ for } i = 1, \ldots, q, \\
S_i^{\text{even}} &= \{7(r_1^{a_i} - 1) + 4, \ldots, 7(r_{l_i}^{a_i} - 1) + 4\} \text{ for } i = q+1, \ldots, 2q, \\
S_i^{\text{even}} &= \{7(r_1^{a_i} - 1) + 6, \ldots, 7(r_{l_i}^{a_i} - 1) + 6\} \text{ for } i = 2q+1, \ldots, 3q.
\end{aligned}
$$

See Figure 1 for a piece of the graph corresponding to some a_i with $q+1 \leq i \leq 2q$, $l_i = 3$.

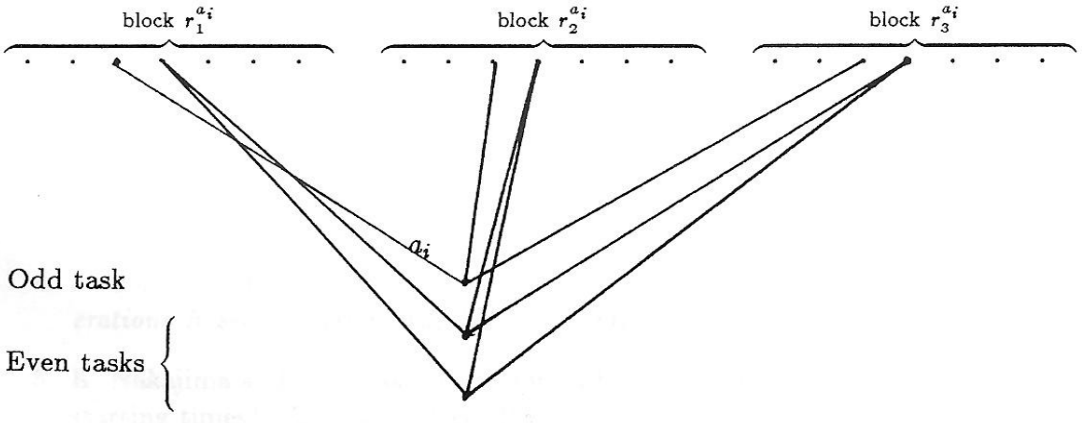

Figure 1

The processing time of each task equals 2. Notice also that in the above described transformation, for an odd task as well as for the even tasks associated with element a_i, $1 \leq i \leq 3q$, the number of possible starting times equals l_i. Thus, $k = 3$ for this instance $\mathcal{T}$ of problem P.

Let us now show that the instance I of 3DM is feasible if and only if there is a feasible schedule for $\mathcal{T}$.

Suppose I has a 3-dimensional matching M. Consider then the blocks r, $1 \leq r \leq m$, corresponding to a triple in M. By scheduling the odd tasks associated to all such blocks r at their starting times $7(r-1)+1, 7(r-1)+3$ and $7(r-1)+5$, we find a schedule for all odd tasks. To schedule the even tasks consider the following. An odd task associated to element a_i has starting times in l_i blocks. Now, since precisely one of these blocks must be in the matching M, the $l_i - 1$ even tasks corresponding to that odd task can be scheduled arbitrarily as long as each of them has its starting time in one of the $l_i - 1$ blocks not used in the matching.

Suppose $\mathcal{T}$ has a feasible schedule F. A simple double-counting argument establishes that $3m = \sum_{i=1}^{3q} l_i$ (this is true for any 3DM instance). Since no more than 3 tasks can start in any of the m blocks, and $3m = \sum_{i=1}^{3q} l_i = n$, it follows that, in F, exactly 3 tasks start in each block. Consider now some element $a_i \in A_3$ $(2q+1 \leq i \leq 3q)$ and the corresponding odd task. Say that, in F, this task starts at time $7(r-1)+5$, where $1 \leq r \leq m$. It is easy to see that the other two tasks starting in block r must start respectively at time $7(r-1)+1$ and $7(r-1)+3$. Hence, the tasks starting in block r are precisely the odd tasks associated with the elements of triple r. This reasoning shows that there exist q blocks accommodating all odd tasks, and that these blocks correspond to q triples forming a matching for the instance I of 3DM. □

References

1. Y. Crama and F.C.R. Spieksma, "Scheduling jobs of equal length: complexity and facets", Research Report M91-13, University of Limburg, submitted for publication (1991).

2. M.R. Garey and D.S. Johnson, *Computers and Intractability: a Guide to the Theory of $\mathcal{NP}$-Completeness*, Freeman, San Francisco, 1979.

3. R. Karp, "Reducibility among combinatorial problems", in R.E. Miller and J.W. Thatcher (eds.), *Complexity of Computer Computations*, Plenum Press, New York, 85–103, 1972.

4. J.M. Keil, "On the complexity of scheduling tasks with discrete starting times", *Operations Research Letters* **12**, 293–295 (1992).

5. K. Nakajima and S.L. Hakimi, "Complexity results for scheduling tasks with discrete starting times", *Journal of Algorithms* **3**, 344–361 (1982).