Pergamon

PII: S0305-0548(96)00045-7

Computers Ops Res. Vol. 24, No. 2, pp. 141–145, 1997
Copyright © 1997 Elsevier Science Ltd
Printed in Great Britain. All rights reserved
0305-0548/97 $17.00+0.00

JOB SELECTION IN A HEAVILY LOADED SHOP

Jay B. Ghosh[†][‡]
Faculty of Business Administration, Bilkent University, Bilkent, 06533, Ankara, Turkey

(Received March 1996; in revised form June 1996)

Scope and Purpose—In a recent paper, Slotnick and Morton introduce a problem of job selection in a heavily loaded shop. The problem is both practically relevant and theoretically interesting. As Slotnick and Morton point out, this kind of problems have just begun to receive attention in the research literature. For the problem at hand, Slotnick and Morton provide several algorithms (one exact and two approximate). They, however, do not establish the complexity status of the problem; nor do they provide any exact algorithm that is formally efficient or any approximate algorithm that guarantees performance within a specified bound. In this note, we attempt to address some of these unresolved issues.

Abstract—Recently, Slotnick and Morton address a job selection problem in a heavily loaded shop, where a tradeoff is sought between the reward obtained when a job is accepted for processing and the lateness penalty incurred when such a job is actually delivered. They provide a branch and bound algorithm and a couple of heuristics for the problem's solution. They do not, however, resolve the issue of problem complexity. In this note, we first establish that the problem is NP-hard. We then go on to provide two pseudo-polynomial time algorithms which also show that the problem is solvable in polynomial time if either the job processing times or the job weights for the lateness penalty are equal. We further provide a fully polynomial time approximation scheme which always generates a solution within a specified percentage of the optimal. Copyright © 1997 Elsevier Science Ltd

1. INTRODUCTION

In a recent paper, Slotnick and Morton [1] introduce a problem of selecting jobs in a heavily loaded shop, where there is a reward for accepting a job for processing but also a lateness penalty associated with the delivery of an accepted job. The idea is to find a suitable tradeoff by selecting a subset of the available jobs (and subsequently sequencing them) that maximizes the net profit. Slotnick and Morton [1] eloquently advocate the practical relevance of the problem and justify the particular variation studied by them. They also point to the emerging literature on job selection and related problems; they refer, among others, to the earlier works of Pourbabai [2,3], Woodruff [4] and Wester *et al.* [5]. Attention should be drawn as well to the work of De *et al.* [6] which, as we shall see in the sequel, provides a significant lead to the problem's effective solution.

The Slotnick–Morton version of the job selection problem [1] can be formally described as follows. Let N be the index set of n jobs numbered 1 through n that are ready at time 0 for processing on a single machine which is continuously available. Associated with each job i in N, there are its processing time p_i, its due-date d_i, a reward of r_i if it is accepted for processing and a weight of w_i for its lateness penalty. Assume at this point, without loss of generality, that all parameters are integers. Now, letting S be a subset of N, σ a sequence of the jobs in S and c_i the completion time of job i in S as sequenced in σ, define $z(S,\sigma)=\Sigma_{i\in S}[r_i-w_i(c_i-d_i)]$. The objective is to find a legitimate (S,σ) pair which maximizes $z(\cdot,\cdot)$. It is easy to see that in order to achieve this objective we need not consider the possibilities of machine idle time between jobs and job preemption. In other words, the job sequence alone is sufficient to identify the schedule.

Slotnick and Morton [1] propose a branch and bound algorithm which finds the exact optimal solution in $O(2^n)$ time. They also present a beam-search heuristic and a myopic heuristic which find approximate solutions (without any guarantees about their closeness to the optimal) in $O(n^4)$ and $O(n^2)$ times, respectively. But they do not resolve the question whether their job selection problem is NP-hard or not.

In this note, we address the issues that are simultaneously of theoretical and practical interest. First, we prove that the Slotnick-Morton version of the job selection problem [1] is NP-hard [7]. Next, we

[†] To whom all correspondence should be addressed (email: ghosh@bilkent.edu.tr)
[‡] Jay B. Ghosh is currently visiting the Bilkent University in Turkey as a Professor of Management. He received his PhD in Industrial Engineering and Operations Research from the University of Arkansas at Fayetteville in 1983. Since then, he has worked extensively on operations planning, scheduling and control problems. His various research papers have been published in this and other respected journals. Dr Ghosh serves on the editorial boards of this journal and of *Production and Operations Management*.

propose two dynamic programs that produce the exact solution to the problem in $O(n\Sigma_{i\in N}p_i)$ and $O(n\Sigma_{i\in N}w_i)$ times, both of which are pseudo-polynomial. This shows that the problem is NP-hard only in the ordinary sense (and is thus solvable efficiently if the p_i or the w_i are agreeable) and also that it can be solved in $O(n^2)$ time if either the p_i or the w_i are all equal. We go on to propose a fully polynomial time approximation scheme [7] which delivers in $O(n^2/\epsilon)$ time a solution to the problem, that is guaranteed to have a value within 100ϵ percent of the optimum. Finally, we conclude with some closing remarks.

2. SOLUTION PROPERTIES AND COMPLEXITY

The first property is rather obvious and stated informally by Slotnick and Morton [1].

Property 1. Given a chosen subset S, the associated optimal sequence σ processes the jobs in S according to the weighted shortest processing time first (WSPT) order.

Because of Property 1, it is possible to represent the pair (S,σ) by S alone and write $z(S,\sigma)$ as simply $z(S)$. We will also assume from now on that the jobs in N are numbered such that $p_1/w_1\leq\ldots\leq p_n/w_n$.

The second property is similar to one in De *et al.* [6] and excludes choices of S that are clearly suboptimal.

Property 2. Given a chosen subset S optimally sequenced, S cannot be optimal if it includes a job i for which $r_i+w_id_i-w_ip_i-w_i(\Sigma_{j\in S,j<i}p_j)-(\Sigma_{j\in S,j>i}w_j)\,p_i\leq 0$.

The third property is a restatement of Theorem 1 of Slotnick and Morton [1] and identifies jobs that will be included in an optimal subset.

Property 3. Given that all jobs are optimally sequenced, job i will be included in an optimal subset if $z(N)\geq z(N-\{i\})$.

The fourth property is in a sense the opposite of Property 3 and identifies jobs that will not be included in any optimal subset. Its proof follows from a straightforward extension of the arguments used in proving Property 2.

Property 4. Job i will not be included in any optimal subset if $r_i+w_id_i-w_ip_i\leq 0$.

Remark 1. We may assume, without loss of generality, that neither Property 3 nor Property 4 applies to the job set N. This leads to a problem instance of full dimension n. Also, this guarantees that the problem instance does not yield a trivial solution given by the null set, where the maximum of $z(\cdot)$ equals 0. Notice that it can be checked, through Property 4, in $O(n)$ time whether an instance has a trivial solution.

Having stated the basic solution properties, we are now ready to prove that the Slotnick–Morton job selection problem [1] is NP-hard [7].

Result 1. The Slotnick–Morton job selection problem is NP-hard.

We use a reduction from the well-known *Partition* problem [7]: given a set N of indexes 1 through n and a set of integers $\{a_i{:}i\in N\}$, is there a subset S of N such that $\Sigma_{i\in S}a_i=\Sigma_{i\in N-S}\,a_i=\frac{1}{2}\,b$, where $\Sigma_{i\in N}\,a_i=b$? This problem is NP-complete.

From this instance of *Partition*, we can create an instance of the Slotnick–Morton job selection problem as follows. For all $i\in N$, set $p_i=w_i=a_i$, and either set $r_i=0$ and $d_i=\frac{1}{2}\,(b+a_i)$ or set $r_i=\frac{1}{2}\,(b+a_i)\,a_i$ and $d_i=0$. This is obviously accomplished in polynomial time and space. Now, consider the associated decision problem: given a job set as above, is there a subset S and an optimal (i.e. WSPT) sequence of the jobs in S such that $z(S)\geq 1/8\;b^2$? We claim that this decision problem has a solution if and only if *Partition* has a solution.

To see that the claim is correct, notice that $p_i/w_i=1$ for all $i\in N$ and thus that the jobs can be sequenced in any order in an optimal subset. Notice further that, for any subset S of N, we get after some algebra: $z(S)=\frac{1}{2}\,b\,(\Sigma_{i\in S}\,a_i)\;-\;\frac{1}{2}\,(\Sigma_{i\in S}\,a_i)^2$. It is not difficult to see at this juncture that $z(S)$ takes on its maximum value ($=1/8\;b^2$) if and only if $\Sigma_{i\in S}\,a_i=\frac{1}{2}\,b$, i.e. if and only if *Partition* has a solution. This proves the claim. Noting now that the decision problem can easily be shown to be in NP, it immediately follows that it is NP-complete. The associated optimization problem (viz. the Slotnick–Morton job selection problem) is therefore NP-hard.

Remark 2. A couple of points deserve to be mentioned. Because of the special way in which we have constructed the problem instance used in the proof given above, it should be clear that the total weighted lateness problem in the job selection context (i.e., the Slotnick–Morton problem with the reward r_i set equal to 0) is NP-hard as well. Similarly, it should be clear that another variation of the Slotnick–Morton problem, where a leadtime penalty (which is based exclusively on c_i and is obtained by setting $d_i=0$ for all $i \in N$ in the Slotnick–Morton objective function) is imposed in place of the lateness penalty, is also NP-hard.

3. EXACT AND APPROXIMATE ALGORITHMS

We now provide two dynamic programming algorithms for the exact solution of the Slotnick–Morton job selection problem. We also provide a fully polynomial time approximation scheme which guarantees a solution with a value within 100ϵ percent of the optimum, where ϵ is prespecified. These algorithms are developed along lines of and are structurally quite similar to those presented by De *et al.* [6] for the solution of a different job selection problem.

Let S be a chosen subset in which the jobs are sequenced optimally (i.e. in the WSPT order). Assume that S is partitioned into subsets F and B such that any job in F precedes all jobs in B. The key to algorithm development is our ability to express $z(S)$ as: $z(S)=z(F)+z(B)-\pi(F)\varpi(B)$ where $\pi(F)=\Sigma_{i\in F}p_i$ and $\varpi(B)=\Sigma_{i\in B}w_i$. We will use this expression (which is obtained through straightforward algebra) to derive the dominance rules for our dynamic programs. We now describe the dynamic programs in an enumerative form [8].

The first algorithm builds on an existing subset S by choosing to include or not include jobs with successively higher indexes and by placing the chosen jobs at the back of the associated sequence σ. Thus, at the end of stage k in our enumeration, S represents a subset of the first k jobs in N. The following is a dominance rule that allows us to eliminate a subset from further consideration if there is another that promises to be at least as good.

Rule 1. Given subsets S and S' at a stage such that $z(S)\geq z(S')$ and $\pi(S)\leq\pi(S')$, retain only S for further expansion.

The rule follows from the observation that if S and S' are identically completed, then under the stated condition S' cannot lead to a solution that is better than one obtained from S (the partitioning expression given above can be used to verify this). We can now outline the first dynamic program which we call DP_F.

Algorithm DP_F:

```
Step 1. Start with a null set.
Step 2. For $k=1$ through $n$:
    (a) From an existing subset $S$ of the first $k-1$ jobs, create a new subset by adding job $k$ to the
        back of the jobs in $S$ only if $r_k+w_kd_k-w_kp_k-w_k\pi(S)>0$ (cf. Property 2).
    (b) Use Rule 1 to retain only a minimal set of nondominated subsets.
Step 3. Identify an optimal subset and the associated sequence.
```

In an actual implementation of DP_F, a subset at stage k may implicitly be represented by a pair (z,π) augmented with an indication if job k is included in the subset or not. Note that the null set will be given by $(0,0)$, and in Step 2(a) of stage k the pair (z,π) may give rise to a new pair $(z+r_k+w_kd_k-w_kp_k-w_k\pi, \pi+p_k)$. Let $\bar{z}$ be an upper bound on the maximum value of z at the end of stage n.

Result 2. Algorithm DP_F provides a correct solution in $O\left(n\min\left\{\bar{z},\Sigma_{i\in N}p_i\right\}\right)$ time.

Algorithm DP_F enumerates over a completely representative set of all nondominated subsets of N. It is thus guaranteed to produce an optimal solution at the end of the enumeration. The number of active subsets, or equivalently (z,π) pairs, retained at the end of any stage is bounded by the number of distinct z or π values possible which in turn are bounded by $\bar{z}$ and $\Sigma_{i\in N}p_i$, respectively. Over n stages, this translates into the reported time (and space) complexity.

Remark 3. It should be clear that if the p_i are all equal, then the number of distinct π values at any stage is bounded by n. The complexity of the algorithm in this case thus becomes $O(n^2)$.

We now turn to the second dynamic program which builds on an existing subset S by choosing to include or not include jobs with successive lower indexes and by placing the chosen jobs in front of the associated sequence σ. In this case, S at the end of stage k thus represents a subset of the last k jobs in

N. We can now state an appropriate dominance rule for this case.

Rule 2. Given subsets S and S' at a stage such that $z(S) \geq z(S')$ and $\varpi(S) \leq \varpi(S')$, retain only S for further expansion.

This rule can also be proved with the same kind of arguments used in the proof of Rule 1. We outline below the second dynamic program called DP_B.

Algorithm DP_B:

```
Step 1. Start with a null set.
Step 2. For $k=1$ through $n$:
        (a) From an existing subset $S$ of the last $k-1$ jobs, create a new subset by adding job $n-k+1$
            in front of the jobs in $S$ only if $r_{n-k+1}+w_{n-k+1}d_{n-k+1}-w_{n-k+1}p_{n-k+1}-\varpi(S)p_{n-k+1}>0$ (cf.
            Property 2).
        (b) Use Rule 2 to retain only a minimal set of nondominated subsets.
Step 3. Identify an optimal subset and the associated sequence.
```

Algorithm DP_B is implemented similar to DP_F; only this time a subset is represented by the pair (z,ϖ) which in Step 2(a) of stage k may give rise to a new pair $(z+r_{n-k+1}+w_{n-k+1}d_{n-k+1} - w_{n-k+1}p_{n-k+1} - \varpi p_{n-k+1}, \varpi + w_{n-k+1})$.

Result 3. Algorithm DP_B gives a correct solution in $O\left(n \min\left\{ \bar{z}, \Sigma_{i\in N} w_i \right\}\right)$ time.

The correctness proof for DP_B is similar to that for DP_F. The number of active subsets or (z,ϖ) pairs at any stage in this case is determined by the number of distinct z or ϖ values. This implies the reported complexity.

Remark 4. When the w_i are all equal, it is apparent that the number of distinct ϖ values at any stage is bounded by n. This leads to an overall complexity of $O(n^2)$ in this case also.

Finally, we are ready to give the ϵ-approximation scheme (call it DP_ϵ) which uses the same enumeration framework as DP_F or DP_B but uses a different rule in Step 2(b); see [8] for examples of this kind of a scheme. Let $\bar{z}_k$ be the maximum z value observed during stage k of the enumeration. Also, let ϵ be the maximum permissible relative error. Assume that the interval $[0,\bar{z}_k]$ is split into subintervals of width Δz_k, where $\Delta z_k = \epsilon\bar{z}_k/n$. The following rule is used.

Rule 3. At any stage of the enumeration, retain one subset with the minimum π (if using DP_F) or the minimum ϖ (if using DP_B) from among those whose z value belong to the same interval.

Result 4. Algorithm DP_ϵ produces an ϵ-approximate solution in $O(n^2/\epsilon)$ time.

Since Rule 3 is an extension of Rule 1 or 2 (depending upon whether we are using DP_F or DP_B for the enumeration), it is easy to see that the maximum error introduced by it at stage k is limited to Δz_k. The error being additive over the stages, the total error due to DP_ϵ in the worst case is $\Sigma_{1\leq k\leq n}\Delta z_k$ $(=\epsilon/n\ \Sigma_{1\leq k\leq n}\bar{z}_k)$. Let z^H be the solution value delivered by the approximation scheme and z^* be the optimal solution value. Since $\bar{z}_k \leq z^*$ for $k, 1\leq k\leq n$, we have: $z^* - z^H \leq \epsilon/n\ \Sigma_{1\leq k\leq n}\bar{z}_k \leq \epsilon/n\ (n\ z^*) \leq \epsilon z^*$. This implies that $(z^* - z^H)/z^* \leq \epsilon$ as claimed.

At stage k, the number of subintervals and therefore the maximum number of active subsets is limited to $\left\lceil \frac{\bar{z}_k}{\Delta z_k} \right\rceil$, where $\lceil x \rceil$ stands for the smallest integer greater than or equal to x. From the definition of Δz_k, it follows that this number is $O(n/\epsilon)$. Consequently, the overall complexity of DP_ϵ is $O(n^2/\epsilon)$ as claimed.

4. CONCLUSION

We have taken a second look at the job selection problem introduced by Slotnick and Morton [1] and have proved that the problem is NP-hard in the ordinary sense. We have also adapted the algorithms given by De *et al.* [6] for a different problem for the solution of the Slotnick–Morton problem. In particular, we have provided two pseudo-polynomial time dynamic programs that will be formally efficient when the job processing times or the job weights for the lateness penalty are agreeable. We have also provided a fully polynomial time approximation scheme that will always produce an ϵ-optimal solution efficiently.

Past experience shows that, for n as large as 100, our dynamic programs will execute within a few CPU seconds on a machine such as the VAX 4000 provided that the processing times and the weights are reasonably valued, for example, when they are independently sampled from a discrete uniform distribution over [1, p. 100]. Thus, for real use or for use as benchmarks in heuristic testing, the dynamic programs proposed here appear to be quite attractive.

REFERENCES

1. Slotnick, S.A. and Morton, T.E., Selecting jobs for a heavily loaded shop with lateness penalties. *Computers Ops Res.*, 1996, **23**, 131–140.
2. Pourbabai, B., A short term production planning and scheduling model. *Engng Costs Product. Econ.*, 1989, **18**, 159–167.
3. Pourbabai, B., Optimal selection of orders in a just-in-time manufacturing environment: a loading model for a computer integrated manufacturing system. *Int. J. Comput. Integrated Manufact.*, 1992, **5**, 38–44.
4. D. L. Woodruff, Subcontracting when there are setups, deadline and tooling costs.*Proceedings of Intelligent Scheduling Systems Symposium*, eds W. T. Scherer and D. E. Brown, pp. 337-353, 1992.
5. Wester, F.A.W., Wijngaard, J. and Zijm, W.H.M., Order acceptance strategies in a production-to-order environment with setup times and due-dates. *Int. J. Product. Res.*, 1992, **30**, 1313–1326.
6. De, P., Ghosh, J.B. and Wells, C.E., Job selection and sequencing on a single machine in a random environment. *Eur. J. Opl Res.*, 1993, **70**, 425–431.
7. Garey, M. R. and Johnson, D. S., *Computers and Intractability.* Freeman, San Francisco, 1979.
8. Horowitz, E. and Sahni, S., *Fundamentals of Computer Algorithms.* Computer Science Press, Potomac, 1978.