European Journal of Operational Research 244 (2015) 766–777

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Improved integer linear programming formulations for the job Sequencing and tool Switching Problem

Daniele Catanzaro [a,b,*], Luis Gouveia [c,1], Martine Labbé [d,2]

[a] Louvain School of Management, Université Catholique de Louvain, Chausse de Binche 151, bte M1.01.01, 7000 Mons, Belgium
[b] Center for Operations Research and Econometrics (CORE), Université Catholique de Louvain, Voie du Roman Pays 34, L1.03.01, B-1348 Louvain-la-Neuve, Belgium
[c] Department of Statistics and Operations Research, Centro de Investigação Operacional, Universidade de Lisboa, Campo Grande, Bloco C6, 1749-016 Lisboa, Portugal
[d] Graphs and Mathematical Optimization Unit, Computer Science Department, Université Libre de Bruxelles, Boulevard du Triomphe, CP 210/01, B-1050 Brussels, Belgium

ARTICLE INFO

Article history:
Received 7 April 2014
Accepted 10 February 2015
Available online 15 February 2015

Keywords:
Job sequencing
Tool switching
Traveling salesman problem
Combinatorial optimization
Integer programming

ABSTRACT

In this article we investigate the *job Sequencing and tool Switching Problem* (SSP), a $\mathcal{NP}$-hard combinatorial optimization problem arising from computer and manufacturing systems. Starting from the results described in Tang and Denardo (1987), Crama et al. (1994) and Laporte et al. (2004), we develop new integer linear programming formulations for the problem that are provably better than the alternative ones currently described in the literature. Computational experiments show that the lower bounds obtained by the linear relaxation of the considered formulations improve, on average, upon those currently described in the literature and suggest, at the same time, new directions for the development of future exact solution approaches.

1. Introduction

In manufacturing systems it often happens that a set of jobs, each one requiring a specific set of tools, has to be sequentially processed in a flexible machine with a capacitated tool magazine (Crama, 1997; Crama, Moonen, Spieksma, & Talloen, 2007; Tang & Denardo, 1987). The capacity of the magazine is usually suited to store the tools required by any job, but it is usually insufficient to store, at the same time, all of the involved tools. Hence, operations of *tools switching* (i.e., the removal of a tool from the magazine and the insertion of a new one in its place) may be necessary to sequentially process the considered set. This situation often occurs e.g., in metal working industries, where automatic machines are used to manufacture parts. Each machine performs operations by using the tools placed in its limited capacity tool magazine and requires tool switching operations when completing a job and moving to the next one in the sequence (Tang & Denardo, 1987). This situation also occurs in other contexts, such as computer systems. Specifically, a modern computer system has k pages of fast memory and $n-k$ pages of slow memory. The system usually has to satisfy a sequence of requests to pages in their order of occurrence. A request can be addressed only if the required page is in the fast memory. If the request cannot be addressed, a page fault occurs and a page in the fast memory must be selected and moved to the slow memory in order to make room for the requested page (McGeoch & Sleator, 1994).

Tools switching operations can be performed either in groups or one at a time, depending on the specific application. However, they are generally time consuming and/or expensive, hence it is usually convenient to process the jobs in such a way that the overall number of tool switches is minimized.

The switching cost may be tool and/or application dependent as shown in Tang and Denardo (1987), Crama (1997), Crama et al. (2007), Crama, Kolen, Oerlemans, and Spieksma (1994), Balakrishnan and Chakravarty (2001), Belady (1966), Blazewicz and Finke (1994), McGeoch and Sleator (1994), Privault and Finke (2000). In this article we just focus on the simplest (although the main) version of the switching cost, characterized by being uniform and tool independent, and we investigate the problem of determining the job processing sequence inducing the minimum overall switching cost. Such a problem is known in the literature as the *job Sequencing and tool Switching*

* Corresponding author at: Center for Operations Research and Econometrics (CORE), Université Catholique de Louvain, Voie du Roman Pays 34, L1.03.01, B-1348 Louvain-la-Neuve, Belgium. Tel.: +32 65 323 313; fax: +32 65 323 279.
E-mail addresses: daniele.catanzaro@uclouvain.be (D. Catanzaro), legouveia@fc.ul.pt (L. Gouveia), mlabbe@ulb.ac.be (M. Labbé).
[1] Tel.: +351 217 500 409; fax: +351 217 500 081.
[2] Tel.: +32 2 650 38 36; fax: +32 2 650 5970.

http://dx.doi.org/10.1016/j.ejor.2015.02.018
0377-2217/

Table 1
An example of an instance of the SSP (Tang & Denardo, 1987).

Jobs	1	2	3	4	5	6	7	8	9	10
Tools	1	1	2	1	3	1	1	6	3	5
	4	3	6	5	5	2				7
	8	5	7	7	8	4				
	9		8	9						
Magazine capacity: 4										

Table 2
An example of a solution to the instance of the SSP shown in Table 1.

Jobs	8	1	6	4	2	5	10	3	9	7
Tools	④	4	4	5	5	5	5	2	3	1
	6	1	1	1	1	⑦	7	7	⑦	⑦
	⑧	8	2	7	3	3	⑥	6	⑥	⑥
	⑨	9	⑨	9	⑧	8	⑧	8	⑧	⑧
Magazine capacity: 4										

Problem (SSP) (Crama et al., 1994; Ghiani, Grieco, & Guerriero, 2010; Laporte, Salazar-Gonzáles, & Semet, 2004; Tang & Denardo, 1987). An example of an instance of the SSP is shown in Table 1. In particular, the first row in the table provides the labels of the jobs and the corresponding processing order; the last row provides the capacity of the magazine. Each column in the table refers to a particular job in the considered set and specifies the tools necessary to its processing. The empty entries in each column denote jobs requiring a number of tools smaller than the capacity of the magazine. A solution to an instance of the SSP is obtained by specifying both the sequence in which the jobs have to be processed and the tools that have to be in the magazine when a given job is processed. For example, Table 2 shows a feasible solution to the instance shown in Table 1. In this case, the job processing sequence is {8, 1, 6, 4, 2, 5, 10, 3, 9, 7}; the circled tools represent tools that are unnecessary to process a specific job, but that may prove useful to decrease the overall number of switches induced by a specific job sequence. The circled tools can be added only if a job requires a number of tools smaller than the capacity of the magazine. The solution induces an overall number of tool switches equal to 15, in particular: 4 tool switches to load the tools required by job 8; 1 tool switch in the transition from job 8 to job 1; 1 tool switch in the transition from job 1 to job 6; 2 tool switches in the transition from job 6 to job 4; 2 tool switches in the transition from job 4 to job 2; and 1 tool switch in each of the remaining transitions. The size (i.e., the number of jobs and tools) of an instance of the SSP depends on the nature of the considered application and can easily range from dozens (e.g., in manufacturing systems) to hundreds (e.g., in computer systems) of jobs and tools.

Although numerous applications of the SSP appeared in the literature since 1966 (see e.g., Balakrishnan & Chakravarty, 2001; Belady, 1966; Blazewicz & Finke, 1994; Crama et al., 1994, 2007; McGeoch & Sleator, 1994; Privault & Finke, 2000), the problem was formally stated only in 1987 by Tang and Denardo (1987). The authors first observed that the optimal solution to an instance of the SSP does not change if one removes from the instance a job whose tool set is a subset of the tool set required by another job. The authors called this property *the dominance rule*. For example, job 10 in Table 1 requires tools 5 and 7 i.e., a subset of the tool set required by job 4. Hence, job 10 can be removed from the instance without affecting the corresponding optimal solution. If we apply the dominance rule also to jobs 7, 8 and 9, we obtain a new instance of the SSP (see Table 3) characterized by a smaller size. Unless not explicitly stated, in this article we will always assume to work with instances of the SSP containing only jobs requiring non-dominated tool sets.

Tang and Denardo (1987) also investigated a number of practical aspects of the problem and first pointed out the relationship between the SSP and the *Traveling Salesman Problem* (TSP) (see Garey & Johnson, 2003). In particular, the authors observed that an instance of the SSP degenerates into an instance of the TSP when (i) the tool set of each job has cardinality constant and equal to the capacity of the magazine and (ii) the switching cost is uniform and tool independent. This observation constitutes the earliest proof of NP-hardness of the SSP; a more formal and thorough analysis of the complexity of the SSP was performed seven years later by Crama et al. (1994).

Table 3
The instance of SSP can be reduced in size by applying the dominance rule described in Tang and Denardo (1987).

Jobs	1	2	3	4	5	6
Tools	1	1	2	1	3	1
	4	3	6	5	5	2
	8	5	7	7	8	4
	9		8	9		
Magazine capacity: 4						

Interestingly, Tang and Denardo showed that the SSP can be decomposed into two nested subproblems called:

1. *the Sequencing Problem (SP)*, consisting of identifying an (optimal) job processing sequence;
2. *the Tooling Problem (TP)*, consisting of determining, for a given job processing sequence, what tools should be loaded in the tool magazine at each position of the sequence, in order to minimize the total number of tool switches.

The authors showed that the TP can be solved in polynomial time via a greedy algorithm called *Keep Tool Needed Soonest* (KTNS); in contrast, the SP constitutes the hardest part of the problem. Tang and Denardo developed an integer linear programming formulation to exactly solve the problem and a constructive heuristic to approximate its optimal solution. However, the performances of both approaches proved to be rather disappointing in practice (Laporte et al., 2004). Hence, numerous research efforts focused on developing solution approaches aiming at solving, at least heuristically, instances of the SSP, see e.g., Crama (1997), Crama et al. (2007), Crama et al. (1994), Askin, Sodhi, and Sen (1994), Avci and Akturk (1996), Bard (1988), Gray, Seidmann, and Stecke (1993), Hertz and Widmer (1993), Hertz, Laporte, Mittaz, and Stecke (1998). Surprisingly enough, however, in the last two decades only a limited number of exact solution approaches have been proposed in the literature, namely Laporte et al. (2004) and Ghiani et al. (2010).

Laporte et al. (2004) proposed an Integer Linear Programming (ILP) formulation and a branch-and-bound algorithm to exactly solve the SSP. The authors showed that the proposed ILP formulation was tighter than Tang and Denardo's but, at the same time, unable to solve instances containing more than 10 jobs within 1 hour of computing time. In contrast, the branch-and-bound algorithm and the corresponding combinatorial lower bounds proved to be more successful, allowing the solution of instances of the SSP containing up to 25 jobs within the same runtime limit. Ghiani et al. (2010) proposed a different exact solution approach for the SSP based on modeling the problem as a nonlinear TSP. The authors improved the KTNS greedy algorithm for the TP and developed a branch-and-cut algorithm for the SSP exploiting a number of symmetry breaking techniques, dominance rules, and specific lower bounds. Their computational experiments showed that the proposed algorithm outperforms the performance of Laporte et al., being able to solve instances of the SSP containing up to 45 jobs within 1 hour of computing time.

The weak lower bounds provided by the linear programming relaxations of the ILP formulations currently described in the literature on the SSP possibly are one of the main reasons for explaining the