Computers & Operations Research 40 (2013) 1060–1063

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

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

A note on ''event-based MILP models for resource-constrained project scheduling problems''

Christian Artigues [a,b,*], Peter Brucker [c], Sigrid Knust [c], Oumar Koné [d], Pierre Lopez [a,b], Marcel Mongeau [e]

[a] CNRS, LAAS, 7 avenue du Colonel Roche, F-31400 Toulouse, France
[b] Univ de Toulouse, LAAS, F-31400 Toulouse, France
[c] Department of Mathematics/Computer Science, University of Osnabrück, Germany
[d] Laboratoire de Mathématiques et Informatique, UFR-SFA, Université d'Abobo - Adjamé, BP 801 Abidjan 02, Côte d'Ivoire, France
[e] École Nationale de l'Aviation Civile, 7 av. É.-Belin - BP 54005, 31055 Toulouse Cedex 4, France

ARTICLE INFO

Available online 1 November 2012

Keywords:
Resource-constrained project scheduling
MILP formulation
Event

ABSTRACT

Recently, new mixed integer linear programming formulations for the resource-constrained project scheduling problem were proposed by Koné et al. [3]. Unfortunately, the presentation of the first new model (called start/end-based formulation SEE) was not correct. More precisely, a set of necessary constraints representing the relative positioning of start and end events of activities was unintentionally omitted in the paper although it was present in the integer program used for the computational experiments. After presenting a counterexample showing the incorrectness, we provide a disaggregated and an aggregated variant of the set of necessary constraints, the disaggregated formulation yielding in theory a better linear programming relaxation. We present computational results showing that although the linear programming relaxations of both formulations yield equivalently poor lower bounds, the disaggregated formulation shows in average a better performance for integer solving of a well-known set of 30-activity instances.

1. Introduction

The resource-constrained project scheduling problem (RCPSP) may be formulated as follows. Given are n activities $A=\{1,\ldots,n\}$ and m renewable resources $R=\{1,\ldots,m\}$. A constant number of B_k units of resource k is available at any time. Activity $i \in A$ must be processed for p_i time units and occupies b_{ik} units of resource k during this time period. Furthermore, a set E of precedence relations (i,j) is given, where $(i,j) \in E$ means that activity j cannot start before activity i is completed.

The objective is to determine starting times S_i for the activities $i \in A$ such that

- at each time the total resource demand is less than or equal to the resource availability of each resource $k \in R$,
- the given precedence constraints are fulfilled, i.e. $S_i + p_i \le S_j$ for $(i,j) \in E$, and
- the makespan $C_{\max} = \max_{i=1}^{n}\{C_i\}$ is minimized, where $C_i := S_i + p_i$ denotes the completion time of activity i.

Additionally, two dummy activities 0 and $n+1$ are introduced indicating the start and the end of the project, respectively. These dummy activities need no resources and have processing time zero. Furthermore, we have $(0,i) \in E$ for all activities i without any predecessor and $(i,n+1) \in E$ for all activities i without any successor. Then S_{n+1} may be interpreted as the makespan of the project.

Recently, two new mixed integer linear programming (MILP) formulations for the RCPSP based on events were introduced in Koné et al. [3]. In the following, we will show that the start/end-based formulation SEE is not correct by presenting a counterexample. Afterwards we give a corrected version and present some computational results.

2. The SEE-formulation

The start/end event-based formulation SEE uses the notion of events which correspond to times where an activity starts or ends. The SEE-formulation relies on the fact that for the RCPSP always an optimal left-shifted (semi-active) schedule exists in which the start time of any activity is either 0 or coincides with the completion time of another activity. Therefore, for n activities at most $n+1$ events have to be considered. Let $\mathcal{E}=\{0,1,\ldots,n\}$ be

* Corresponding author at: CNRS, LAAS, 7 avenue du Colonel Roche, F-31400 Toulouse, France.
E-mail addresses: christian.artigues@laas.fr (C. Artigues), peter.brucker@uni-osnabrueck.de (P. Brucker), sigrid.knust@uni-osnabrueck.de (S. Knust), mr.okone@gmail.com (O. Koné), pierre.lopez@laas.fr (P. Lopez), marcel.mongeau@enac.fr (M. Mongeau).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.10.018

the index set of the events corresponding to the starting and completion times of the activities.

In the SEE-formulation, two types of binary variables and two types of continuous variables are used. There are binary variables x_{ie}, $y_{ie} \in \{0,1\}$ for $i \in A$, $e \in \mathcal{E}$ with

$$x_{ie} := \begin{cases} 1, & \text{if activity } i \text{ starts at event } e \\ 0, & \text{otherwise} \end{cases}$$

and

$$y_{ie} := \begin{cases} 1, & \text{if activity } i \text{ ends at event } e \\ 0, & \text{otherwise} \end{cases}$$

The continuous variables t_e for $e \in \mathcal{E}$ represent the dates of the events e. It is assumed that the events are enumerated such that $t_0 \le t_1 \le \cdots \le t_n$ holds. The continuous (auxiliary) variable r_{ek} for $e \in \mathcal{E}, k \in R$ represents the quantity of resource k required immediately after event e.

Using these variables, the SEE-formulation was introduced as follows:

$$\min \quad t_n \tag{1}$$

$$\text{s.t.} \quad t_0 = 0 \tag{2}$$

$$t_{e+1} - t_e \ge 0 \quad (e \in \mathcal{E} \setminus \{n\}) \tag{3}$$

$$t_f - t_e - p_i x_{ie} + p_i(1 - y_{if}) \ge 0 \quad (i \in A;\ e,f \in \mathcal{E}, e < f) \tag{4}$$

$$\sum_{e \in \mathcal{E}} x_{ie} = 1 \quad (i \in A) \tag{5}$$

$$\sum_{e \in \mathcal{E}} y_{ie} = 1 \quad (i \in A) \tag{6}$$

$$\sum_{e'=e}^{n} y_{ie'} + \sum_{e'=0}^{e-1} x_{je'} \le 1 \quad ((i,j) \in E;\ e \in \mathcal{E}) \tag{7}$$

$$r_{0k} - \sum_{i \in A} b_{ik} x_{i0} = 0 \quad (k \in R) \tag{8}$$

$$r_{ek} - r_{e-1,k} + \sum_{i \in A} b_{ik}(y_{ie} - x_{ie}) = 0 \quad (e \in \mathcal{E} \setminus \{0\};\ k \in R) \tag{9}$$

$$r_{ek} \le B_k \quad (e \in \mathcal{E}; k \in R) \tag{10}$$

$$x_{ie}, y_{ie} \in \{0,1\} \quad (i \in A;\ e \in \mathcal{E}) \tag{11}$$

$$t_e \ge 0 \quad (e \in \mathcal{E}) \tag{12}$$

$$r_{ek} \ge 0 \quad (e \in \mathcal{E};\ k \in R) \tag{13}$$

Additionally, the following inequalities were integrated taking into account time windows $[ES_i, LS_i]$ for the activities $i \in A$, where ES_i and LS_i denote the earliest and latest starting time for i, respectively:

$$ES_i x_{ie} \le t_e \le LS_i x_{ie} + LS_{n+1}(1 - x_{ie}) \quad (i \in A; e \in \mathcal{E}) \tag{14}$$

$$(ES_i + p_i) y_{ie} \le t_e \le (LS_i + p_i) y_{ie} + LS_{n+1}(1 - y_{ie}) \quad (i \in A; e \in \mathcal{E}) \tag{15}$$

$$ES_{n+1} \le t_n \tag{16}$$

The objective function (1) consists in minimizing the completion time t_n of an activity processed last. Constraint (2) indicates that the first event starts at time 0, (3) takes care of the ordering of the events. Inequalities (4) ensure that if $x_{ie} = y_{if} = 1$ (i.e. i starts at event e and completes at event f), then $t_f \ge t_e + p_i$ holds. For all other combinations of values for x_{ie} and y_{if} we have either $t_f \ge t_e$ or $t_f \ge t_e - p_i$, which are covered by (3). Constraints (5) and (6) guarantee that each activity starts and ends exactly once. Constraint (7) ensures that the given precedence constraints are respected: If a predecessor i of j ends at event e or later (i.e. $\sum_{e'=e}^{n} y_{ie} = 1$), then $\sum_{e'=0}^{e-1} x_{je'}$ must be zero, i.e. j cannot start before event e. Equalities (8) set the initial quantities of each resource k needed immediately after time 0. Equalities (9) describe the recursion for calculating the r_{ek}-values for the other events, namely the quantity of resource k needed immediately after time t_e is equal to the quantity of resource k needed immediately after time t_{e-1} plus the quantity of resource k needed by the activities starting at time t_e minus the quantity of resource k needed by the activities completing at time t_e. Constraints (10) limit the quantity of resource k needed immediately after time t_e to the availability of resource k. Inequalities (14) and (15) ensure that an activity cannot start before its earliest starting time nor after its latest starting time. Furthermore, (16) says that the project cannot end before the earliest start time of the dummy finishing activity $n+1$.

Unfortunately, this formulation is not correct since an activity may end at the same time or before it is started (i.e. we may set $x_{ie} = y_{if} = 1$ for events $f < e$). For simplicity, we consider the formulation without the inequalities (14)–(16) for the time windows since they do not influence correctness.

For a counterexample, consider a project with $n=4$ activities, $m=2$ resources with capacities $B_1 = 5$, $B_2 = 7$, a precedence relation (2,3), and the following data:

i	1	2	3	4
p_i	4	3	5	8
b_{i1}	2	1	2	2
b_{i2}	3	5	2	4

An optimal schedule with makespan $C_{\max} = 12$ is shown in Fig. 1.

Let us consider the solution

$$t_0 = t_1 = t_2 = t_3 = t_4 = 0$$
$$x_{14} = x_{21} = x_{34} = x_{41} = 1$$
$$y_{10} = y_{21} = y_{31} = y_{40} = 1$$
$$r_{41} = 4, \quad r_{12} = r_{22} = r_{32} = 2, \quad r_{42} = 7$$

where all other values are equal to zero. In the following, we show that this solution is feasible for the SEE-formulation (1)–(13): That the constraints (2), (3), (5), (6), and (11)–(13) are satisfied, can be seen immediately. Conditions (4) hold because for all $e < f$ we have $(x_{ie}, y_{if}) \ne (1,1)$ for $i = 1,2,3,4$. Therefore, (4) is equivalent to $t_f \ge t_e$ or $t_f \ge t_e - p_i$, which is satisfied due to (3). Condition (7) is fulfilled because for $(i,j) = (2,3)$ we have $\sum_{e'=1}^{n} y_{2e'} + x_{30} = 1 + 0 = 1$ and $\sum_{e'=e}^{n} y_{2e'} = 0$ for $e > 1$. That (8)–(10) are satisfied can be seen by evaluating (8) and (9). For example, for the

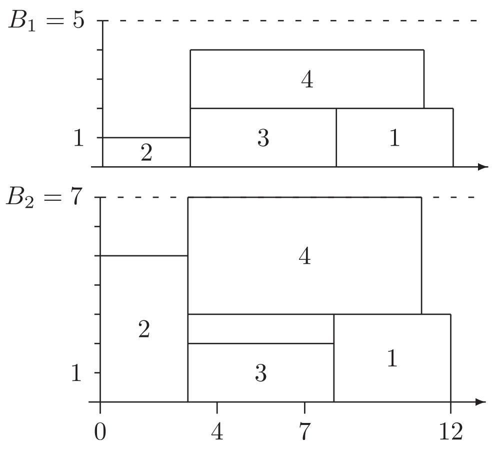

Fig. 1. An optimal schedule for the example.

resource $k=2$ we get

$$r_{02}=0$$
$$r_{12}=r_{02}+\sum_{i\in A}b_{i2}(x_{i1}-y_{i1})=0+3(0-0)+5(1-1)+2(0-1)$$
$$+4(1-0)=2r_{22}=r_{12}+\sum_{i\in A}b_{i2}(x_{i2}-y_{i2})=2+3(0-0)+5(0-0)$$
$$+2(0-0)+4(0-0)=2r_{32}=r_{22}+\sum_{i\in A}b_{i2}(x_{i3}-y_{i3})=2+3(0-0)$$
$$+5(0-0)+2(0-0)+4(0-0)=2r_{42}=r_{32}$$
$$+\sum_{i\in A}b_{i2}(x_{i4}-y_{i4})=2+3(1-0)+5(0-0)+2(1-0)+4(0-0)=7$$

and all these values are not larger than $B_2=7$. Similarly, for the resource $k=1$ we have $r_{01}=r_{11}=r_{21}=r_{31}=0$ and $r_{41}=4$, which are all smaller than $B_1=5$.

Since this solution does not correspond to a feasible schedule for the RCPSP, it is shown that the SEE-formulation is incorrect.

If we consider the formulation with the time window inequalities (14)–(16), we still get incorrect solutions. For the above example with $ES_1=ES_2=ES_4=0$, $ES_3=3$, $LS_1=LS_3=LS_4=15$, $LS_2=10$ we get the solution

$$t_0=t_1=t_2=0,\quad t_3=4,\quad t_4=8$$
$$x_{14}=x_{23}=x_{34}=x_{42}=1$$
$$y_{13}=y_{23}=y_{34}=y_{44}=1$$
$$r_{21}=2,\quad r_{22}=4,\quad r_{32}=1$$

where all other values are equal to zero.

In order to correct the above formulation, we must ensure that an activity does not end at the same time or before it is started.

In the computational experiments presented in [3], this was actually done by including the following constraints.

$$\sum_{v=0}^{e}y_{iv}+\sum_{v=e}^{n}x_{iv}\le 1\quad (i\in A,e\in\mathcal{E})\tag{17}$$

These constraints state that an activity cannot simultaneously start at event e or after, and finish at event e or before.

Alternatively, this can be guaranteed by adding the following smaller set of inequalities:

$$\sum_{f\in\mathcal{E}}fy_{if}-\sum_{e\in\mathcal{E}}ex_{ie}\ge 1\quad (i\in A)\tag{18}$$

An analogy can be made between the disaggregated precedence constraints of the standard time-indexed formulation [1] and constraints (17), as well as between the aggregated precedence constraints and constraints (18).

In particular, if for all $i\in A$ and for all $e\in\mathcal{E}$, x_{iv}, $y_{iv}\ge 0$, we have (5), (6) and (17), which implies (18). It follows that the LP relaxation of (1)–(13), (17) cannot be worse than the LP relaxation of (1)–(13), (18). However, as remarked in [4], inequalities (17) are not strong enough to define the convex hull of feasible solutions, contrarily to the equivalent inequalities in the time-indexed formulation. This means that, even with the tighter constraints, the LP relaxation is likely to be of poor quality, as experienced in [3].

If we add either inequalities 17 or inequalities 18 to (1)–(13), for the previous example we get the optimal solution

$$t_0=0,\quad t_1=3,\quad t_2=8,\quad t_3=11,\quad t_4=12$$
$$x_{12}=x_{20}=x_{31}=x_{41}=1$$
$$y_{14}=y_{21}=y_{32}=y_{43}=1$$
$$r_{01}=1,\quad r_{11}=r_{21}=4,\quad r_{31}=2,$$
$$r_{02}=5,\quad r_{12}=6,\quad r_{22}=7,\quad r_{32}=3$$

which corresponds to the optimal schedule shown in Fig. 1.

It remains to show the correctness of the extended SEE-formulations in general, i.e. we have to prove that an optimal solution of the extended SEE-formulation (i.e. (1)–(13) with, in addition, either (17) or (18)) always provides an optimal schedule for the RCPSP.

Theorem 1. *An optimal solution (t,x,y) of MILP (1)–(13) with either (17) or (18) provides an optimal RCPSP schedule $S=(S_i)_{i\in A}$ where $S_i=\sum_{e\in\mathcal{E}}t_ex_{ie}$ for $i\in A$.*

Proof. We have already shown that any feasible solution for the RCPSP provides a feasible solution of the MILP-formulation and t_n corresponds to the makespan. Therefore, it remains to show that if (t,x,y) is a feasible solution for (1)–(13), (17) or (1)–(13), (18), then the schedule S with $S_i=\sum_{e\in\mathcal{E}}t_ex_{ie}$ is a feasible solution for the RCPSP. Note that due to (5) for any activity $i\in A$ exactly one x_{ie} is equal to one, i.e. the starting time S_i is determined by exactly one event. Furthermore, the interpretation of $y_{if}=1$ is slightly different: if $y_{if}=1$ holds, then t_f is only an upper bound for the completion time of activity i (it may be larger than the starting time of activity i plus p_i). But, since the y_{if}-values are ignored in the definition of S, this causes no problem.

In the following, we show that the schedule S is feasible, i.e. satisfies all precedence and resource constraints:

- Let $(i,j)\in E$, i.e. j is a successor of i. If $x_{ie}=1$, then by constraints (17) or (18) we must have $y_{if}=1$ for some $f>e$. Thus, due to (4) we have

  $$0\le t_f-t_e-p_ix_{ie}+p_i(1-y_{if})=t_f-t_e-p_i,$$

  which implies $t_f\ge t_e+p_i$. Furthermore, with $y_{if}=1$ constraints (5) and (7) induce that $x_{jg}=1$ for some event $g\ge f$. Therefore, by (3) we get $t_g\ge t_f$, which implies $t_e+p_i\le t_f\le t_g$, i.e. activity j cannot start before the completion time of activity i.
- Due to (4) for each activity i and events e,f with $x_{ie}=y_{if}=1$ the inequality $t_f\ge t_e+p_i$ must hold. Therefore, by (8)–(10) the capacity constraints are satisfied even when (in the MILP solution) activity i completes at a time $t_f\ge t_e+p_i$ instead of time t_e+p_i. This implies that all resource constraints are fulfilled. □

3. Computational results

We performed a series of tests to compare the aggregated and the disaggregated variant as the results presented by Koné et al. [3] only presented the results of the disaggregated variant.

As MILP solver we used CPLEX 12.3 on a 4-core Intel Xeon W3520 processor with 2.66 GHz and 5.83 GB main storage, under Linux Ubuntu 10.04. We took the 480 instances KSD30 with $n=30$ activities for the formulation (1)–(18). Time windows $[ES_i,LS_i]$ and an upper bound T for the time horizon were calculated as described in [3].

At first, we considered the continuous LP relaxation which could be solved fast for all instances. For both formulations and for all instances the calculated lower bound value was equal to the trivial lower bound ES_{n+1}, i.e. as reported in [3] the LP relaxations of both SEE formulations are very poor.

Afterwards, we tried to compare the MILP-formulations for exact (integer) solving with a time limit of 10 min. Table 1 shows summarized results for the 480 instances. Column "int" gives the

Table 1
Comparison of the disaggregated and aggregated formulations.

Formulation	int	opt	better	impr (%)	av CPU (s)
Disaggregated	319	249	12	3.03	316
Aggregated	251	225	6	2.19	368

number of times a feasible (integer) solution was found. Column “opt” gives the number of times the solution was proven to be optimal by the IP solver. Column “better” reports the number of times the formulation obtained strictly better solutions than the other when both formulations found a feasible solution. Column “impr” gives the average improvement (in relative value) on the makespan for each instance counted in column “better”. Last, column “av CPU (s)” gives the average CPU time (including the cases where the time limit is reached). The results clearly show a dominance (in general) of the disaggregated formulation.

4. Concluding remarks

In this note, we have shown that the start/end-based MILP-formulation SEE given by Koné et al. [3] is incorrect since a set of necessary constraints representing the relative positioning of start and end events of activities was unintentionally omitted in the paper, although it was present in the integer program used for the computational experiments.

Additionally, we provided a disaggregated and an aggregated variant of a set of necessary constraints. Although (in theory) the disaggregated formulation yields a better linear programming relaxation, the lower bounds obtained by both formulations on the KSD30 set are equivalently poor. However, for integer solving, the disaggregated formulation obtains in general better results, although it involves a larger set of constraints. This could mean that CPLEX is able to derive better cuts from the disaggregated formulation, or that the formulation becomes tighter as variables are fixed in the branch-and-bound tree. In both cases, investigating additional cuts for event-based formulations appears as a promising research topic.

On the other hand, recently, very good computational results were obtained with standard SAT solvers applied to an RCPSP formulated as a SAT problem (see [2]). Hence, for further research it would be interesting to study whether the event-based formulations can be used in such an approach.

References

[1] Christofides N, Alvarez-Valdes R, Tamarit JM. Project scheduling with resource constraints: a branch and bound approach. European Journal of Operational Research 1987;29(4):262–73.
[2] Horbach A. A Boolean satisfiability approach to the resource-constrained project scheduling problem. Annals of Operations Research 2010;181:89–107.
[3] Koné O, Artigues C, Lopez P, Mongeau M. Event-based MILP models for resource-constrained project scheduling problems. Computers & Operations Research 2011;38:3–13.
[4] Queyranne M, Schulz AS. Polyhedral approaches to machine scheduling. Technical Report No. 408/1994. Technische Universität Berlin, 1994.