European Journal of Operational Research 246 (2015) 379–391

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Discrete Optimization

# Scheduling resource-constrained projects with a flexible project structure

Carolin Kellenbrink[1], Stefan Helber*

*Department of Production Management, Leibniz Universität Hannover, Königsworther Platz 1, D-30167 Hannover, Germany*

ARTICLE INFO

*Article history:*
Received 4 March 2013
Accepted 1 May 2015
Available online 11 May 2015

*Keywords:*
Project scheduling
Genetic algorithms
RCPSP
Flexible projects

ABSTRACT

In projects with a flexible project structure, the activities that must be scheduled are not completely known in advance. Scheduling such projects includes deciding whether to perform particular activities. This decision also affects precedence constraints among the implemented activities. However, established model formulations and solution approaches for the resource-constrained project scheduling problem (RCPSP) assume that the project structure is provided in advance. In this paper, the traditional RCPSP is extended using a highly general model-endogenous decision on this flexible project structure. This extension is illustrated using the example of the aircraft turnaround process at airports. We present a genetic algorithm to solve this type of scheduling problem and evaluate it in an extensive numerical study.



## 1. Introduction

In the classical resource-constrained project scheduling problem (RCPSP), the project structure is provided exogenously, i.e., all activities and precedence constraints are known, and all activities must be implemented. In this paper, the RCPSP is extended using a model-endogenous decision on the project structure. For projects with a flexible project structure, scheduling includes deciding whether to implement specific optional activities and impose the related precedence constraints.

Because of these constraints, any predecessor activity must be completed before a directly succeeding activity can be started. In addition, resource constraints must be considered for renewable and/or non-renewable resources. Renewable resources, e.g., machines or human resources, are available in a given quantity in each period. By contrast, non-renewable resources are limited for the entire planning horizon. An example of this latter type of resource is the budget available for the entire project. It is not necessary to consider this type of resource in the classical RCPSP. However, for projects with a flexible project structure, the resource requirements for a non-renewable resource may vary because of the (non-)implementation of some activities so that a particular project structure may be infeasible due to a non-renewable resource. The typical aim of the RCPSP is to create a schedule that minimizes the total makespan of the project, i.e., the completion time of the last activity. The established multi-mode extension of the RCPSP (MRCPSP, cf., e.g., Talbot, 1982) can be interpreted as a special case of the problem studied in this paper. In the MRCPSP, each activity can be performed in one or more alternative modes, one of which must be selected, while all precedence constraints must be respected, regardless of the chosen activity modes. In our approach, we would introduce a specific activity that corresponds to each mode of the MRCPSP and impose the same set of precedence constraints. Thus, the MRCPSP is included in the problem studied in this paper. However, as observed below, the modeling flexibility that our approach achieves notably exceeds that of the MRCPSP.

The remainder of this paper is organized as follows. In Section 2, the assumptions for the RCPSP with model-endogenous decision on the project structure (RCPSP-PS) are stated, and a practical example is provided. In Section 3, we develop a mathematical model for the RCPSP-PS. The genetic algorithm to solve the problem is presented in Section 4. We report the numerical results in Section 5. The paper ends with some conclusions and suggestions for further research in Section 6.

## 2. Problem statement

### 2.1. Projects with a flexible project structure

In projects with a flexible project structure, decisions must be made whether to implement specific activities and impose specific precedence constraints, which leads to the question of how to model this flexibility of the project structure.

Even in flexible projects, some activities and the precedence constraints among those activities might be mandatory, i.e., they must

* Corresponding author. Tel.: +49 511 7625650.
*E-mail addresses:* carolin.kellenbrink@prod.uni-hannover.de (C. Kellenbrink), stefan.helber@prod.uni-hannover.de (S. Helber).
[1] +49 511 7628002.

http://dx.doi.org/10.1016/j.ejor.2015.05.003
0377-2217/

always be implemented, as in a classical RCPSP. In addition, to develop the RCPSP-PS, we assume that

(i) Choices among alternative activities must be made, which (eventually)
(ii) Cause the (non-)implementation of further activities, and/or
(iii) Trigger further choices.

Because of assumptions (ii) and (iii), such a RCPSP-PS with a flexible structure differs from the multi-mode MRCPSP with a rigid project structure. In the RCPSP-PS, a set of potential precedence constraints is defined in the same manner as the precedence constraints in the RCPSP. However, this potential precedence constraint is only enforced if both the preceding and the succeeding activities that are connected via this constraint are actually implemented in a schedule. In this schedule, the starting and finishing times of the implemented activities are determined. Thus, the timing of the implemented activities and the decision on the project structure are interdependent. In addition, the decision about the project structure, which addresses topics (i)–(iii), can be combinatorial. Therefore, it is unreasonable to separate these two planning steps from each other. Instead, the decision about the project structure and the timing of activities should be made model-endogenously and simultaneously.

For example, these flexible projects can describe the passenger aircraft turnaround process at an airport, which is explained in the next subsection to illustrate the problem. However, flexible projects are also found in other fields. The basic idea for the model presented in this paper evolved from the interdisciplinary collaborative research center "Regeneration of complex durable goods", in which the often highly individual overhaul of complex capital goods such as aircraft engines is investigated, see www.sfb871.de. For a given wear pattern of an engine, alternative regeneration processes may be legally possible within the airworthiness regulations and the engine manual of the engine producer, leading to a flexible project structure in which alternative activities reflect alternative methods to regenerate the aircraft engine. However, because of the high complexity of those regeneration processes, we use the example of the much less complex but project-type structured passenger aircraft turnaround process for illustrative purposes, although this process is typically not called a "project".

### 2.2. Practical example: the aircraft turnaround process

The aircraft turnaround process consists of the steps through which a passenger aircraft passes between its arrival at an airport and its next departure. This turnaround process can be interpreted as a (small) project and organized in different ways, cf. Kuster, Jannach, and Friedrich (2009).

Table 1 presents a strongly simplified version of the flexible turnaround process. Some activities are mandatory, e.g., cleaning the aircraft, catering and boarding. The choice among alternative arrival options affects the (de-)boarding of the aircraft. The aircraft can arrive either at the apron of the airfield or at the terminal. If the aircraft arrives at the terminal, this choice activates two other activities. First, passenger deboarding must be performed using a bridge. Second, a push-back operation must be performed using a tow vehicle to transport the aircraft back to the airfield. If the aircraft arrives at the apron, this triggers another choice related to deboarding: after leaving the aircraft via stairs, the passengers can (in principle) either walk to the terminal building or be transported by a bus. Finally, there is a choice of fueling. If firefighters supervise the fueling, passengers may board the aircraft while it is being fueled. Without the supervision of firefighters, fueling must be completed before passengers may board the aircraft. (To simplify the example, we intentionally abstract from different boarding operations and possible aircraft relocation operations.) This tiny and simplified example of a flexible project contains

**Table 1**
Modeling aspects of the flexible project structure for a turnaround.

| Modeling aspect | Example(s) |
|---|---|
| Mandatory implementation of some activities | Cleaning the aircraft<br>Catering<br>Boarding |
| Choices among alternative activities | Arrival at the apron of the airfield **or** at the terminal<br>Deboarding by foot **or** by bus<br>Fueling with **or** without firefighters |
| Activities caused by choices made | Arrival at the terminal **causes** deboarding by bridge<br>Arrival at the terminal **causes** push-back |
| Choices triggered by choices made | Arrival at the apron of the airfield **triggers** choice on deboarding mode |
| Precedence constraints caused by optional activities | Fueling without supervision by firefighters must be completed before boarding can start |

all problem aspects introduced in Subsection 2.1. We will return to the example in Subsection 3.1.

### 2.3. Related literature

There is a broad body of literature on resource-constrained project scheduling. Demeulemeester and Herroelen (2002) provide a broad overview of the basic principles and approaches in this field. Extensive literature surveys are provided by Hartmann and Briskorn (2010), Kolisch and Padman (2001), Herroelen, De Reyck, and Demeulemeester (1998), Özdamar and Ulusoy (1995) as well as Brucker, Drexl, Möhring, Neumann, and Pesch (1999). Thus, only the most important research publications regarding execution modes of activities are addressed here.

In the multi-mode extension of the RCPSP, the MRCPSP (cf., e.g., Talbot (1982), or for a recent survey Węglarz, Józefowska, Mika, & Waligóra, 2011), different execution modes can be available for an activity. Although the capacity load and duration vary over these different modes, each activity still must be implemented. Thus, the precedence relations are fixed, and the set of activities to be implemented, i.e., the project structure, does not vary in the MRCPSP. Therefore, even if a dummy-mode with a duration of 0 units is implemented, which indicates that an activity is not implemented, it is not possible to remove the precedence restrictions that accompany fueling without firefighters in the above example. Another difference between the RCPSP-PS studied in this paper and the MRCPSP is the independent choice of modes of the MRCPSP, i.e., a selected mode for one activity does not imply a specific mode for another activity. Indeed, some papers consider that all activities in a predefined set must be executed in the same mode (cf., e.g., Salewski, Schirmer, and Drexl (1997) and Drexl, Nissen, Patterson, and Salewski (2000), pp. 62–64). However, it is not possible to assure that, e.g., only one activity is allowed to be performed in the dummy-mode. Naber and Kolisch (2014) have treated a specific different type of flexibility with respect to resource profiles such that the more of a resource is allocated to an activity at a moment in time, the shorter its duration is.

Tiwari, Patterson, and Mabert (2009) extended the MRCPSP using rework activities. Rework is required if the original activity is implemented in a specific predefined mode. In this approach, rework always consists of a single activity that is a direct successor of the original activity causing the rework activity. This requirement results in variation in the project structure, although only to a notably limited extent because only single rework activities can be activated.

Belhe and Kusiak (1995) present the so-called "design activity network" to include logical dependencies among some activities. For a logical "or" dependency, a decision must be made regarding