A tabu search procedure for the resource-constrained project scheduling problem with alternative subgraphs

Tom Servranckx[a], Mario Vanhoucke[a,b,c,*]

[a] *Faculty of Economics and Business Administration, Ghent University, Tweekerkenstraat 2, 9000 Ghent (Belgium)*
[b] *Operations and Technology Management Centre, Vlerick Business School, Reep 1, 9000 Ghent (Belgium)*
[c] *UCL School of Management, University College London, 1 Canada Square, London E14 5AA (UK)*

Abstract

This paper investigates the resource-constrained project scheduling problem with alternative subgraphs (RCPSP-AS). In this scheduling problem, there exist alternative ways to execute subsets of activities that belong to work packages. One alternative execution mode must be selected for each work package and, subsequently, the selected activities in the project structure should be scheduled. Therefore, the RCPSP-AS consists of two subproblems: a selection and a scheduling subproblem. A key feature of this research is the categorisation of different types of alternative subgraphs in a comprehensive classification matrix based on the dependencies that exist between the alternatives in the project structure. As the existing problem-specific datasets do not support this framework, we propose a new dataset of problem instances using a well-known project network generator. Furthermore, we develop a tabu search that uses information from the proposed classification matrix to guide the search process towards high-quality solutions. We verify the overall performance of the metaheuristic and different improvement strategies using the developed dataset. Moreover, we show the impact of different problem parameters on the solution quality and we analyse the impact of distinct resource characteristics of alternatives on the selection process.

Keywords: Project scheduling, Resource-constrained scheduling, Alternative subgraphs, Tabu search

*Corresponding author
Email addresses: tom.servranckx@ugent.be (Tom Servranckx), mario.vanhoucke@ugent.be (Mario Vanhoucke)

1. Introduction

The scheduling of activities subject to resource and technological constraints in order to minimise the total project duration or makespan is the standard problem in the context of project scheduling, known as the resource-constrained project scheduling problem (RCPSP). We refer to multiple survey papers published on the basic principles of the RCPSP (Demeulemeester and Herroelen, 2002; Brucker et al., 1999) and methods for the RCPSP (Hartmann and Kolisch, 2000; Kolisch and Hartmann, 2006; Weglarz et al., 2011; Weglarz, 2012). The problem is known to be NP-hard (Blazewicz et al., 1983). Several researchers have relaxed some of the assumptions of the RCPSP in order to obtain a more general project scheduling problem. We refer to the work of Hartmann and Briskorn (2010) for a broad overview of the different extensions of the RCPSP. However, most research studies retain the assumption that the project structure is deterministic and completely known in advance. Based on discussions with practitioners, Vidal et al. (2011) observed that engineering projects become larger, use more sophisticated technologies and involve a larger number of stakeholders. Due to the ever-increasing complexity of projects (Marle and Vidal, 2016), the above assumption has been rendered obsolete. In case that we relax the assumption of a deterministic project structure, we consider a more general project scheduling problem that consists of alternative, substitutable ways to execute the project. In this research, the RCPSP is therefore extended with alternatives in order to allow a flexible network topology. In this regard, we should mention an extension of the RCPSP related to this research, called multi-mode RCPSP (MRCPSP) (Elmaghraby, 1977). In the MRCPSP, activities can be performed in several modes that correspond to specific resource/time profiles of the activities. However, the main difference between the MRCPSP and the scheduling problem proposed in this research is the level in the work breakdown structure (WBS) at which the alternative execution modes are defined. The alternatives are limited to the activity level in the former case, while the alternatives concern a complete subgraph of activities in the latter case. The problem formulation should also be clearly distinguished from modular projects with alternative technologies as investigated by Creemers et al. (2015). In modular projects, the assumptions typically hold that (1) modules are considered to be independent subparts, (2) different alternatives of a module can be active in parallel and (3) modules can fail to deliver an outcome. In order to differentiate the problem described in this paper from the above problems, we refer to this problem as project scheduling with alternative subgraphs. The need to model alternative

subgraphs in the RCPSP is motivated by several practical examples from project and job-shop scheduling, disruption and portfolio management. Since projects have become highly complex systems through a multitude of mutual interrelations between activities in the project structure, the existence of alternatives is either imposed by the project's reality or sought-after by project managers. For example, Kis (2003) describes a case study in which there exists a flexible production process for wire harnesses using general machines, specialised semi-automated machines or manual effort. These alternative processes are vital to the flexibility required in today's production processes. Similar, Kellenbrink and Helber (2015) identify alternatives in highly complex regeneration processes of complex durable goods such as aircraft engines. Within the boundaries set by legal constraints and engine characteristics, this allows manufacturers to draft customised regeneration schedules for the engines. Moreover, Kuster et al. (2008) discuss the aircraft turnaround process that can be executed in different variations. The authors investigate how these alternative processes can be used to manage frequently occurring disruptions. Finally, the existence of alternatives can be used to model processes that produce multiple types of products on shared resources, i.e. a flexible manufacturing system (FMS) (Lee and DiCesare, 1994). In a FMS, jobs can be completed by executing different sequences of operations. The fact that a process can be carried out in multiple ways is crucial to guarantee the flexibility that typifies a FMS. The above examples provide both theoretical and practical support for the project scheduling problem discussed in this research.

Despite that there has been an increasing volume of research on project scheduling with alternatives in recent years, studies have been developed largely independent and are motivated by small-scaled numerical examples and case studies. Therefore, they should be combined as well as extended to provide a broader perspective on the problem area. Furthermore, the majority of publications focus on a restricted definition of alternative subgraphs that hinder the practical application of the scheduling problem. To the best of our knowledge, no research to date provides a systematic analysis of the impact of the characteristics of alternative subgraphs on the solution quality of project schedules as well as the relationship between resources and alternatives.

Our research efforts result in four main contributions. (1) We extend the existing problem formulation of project scheduling with alternative subgraphs with the concepts of nested and linked alternatives. (2) We introduce a comprehensive theoretical framework in order to unambiguously classify projects with alternative subgraphs. (3) We propose a new benchmark dataset of artificial problem instances that supports the proposed problem

formulation. (4) We develop a metaheuristic solution approach to obtain high-quality solutions for the project scheduling problem with alternative subgraphs.

The outline of this paper is organised as follows. In section 2, we discuss the literature overview and section 3 describes the problem statement. In section 4, the general solution framework that is proposed in this research is presented, while the specific implementation of the tabu search procedure is explained in detail in section 5. Section 6 details the new benchmark dataset that is presented in this research as well as the computational results. Finally, we discuss some general conclusions and suggestions for future research.

2. Literature overview

The existence of alternatives is often expressed by means of logical constraints such as AND-, (exclusive) OR- or BI-constraints. In this section, we discuss the existing literature on the application of such logical constraints in (1) project scheduling without resources, (2) project scheduling with resources and (3) other related fields of research.

Initially, Neumann (1990) presents stochastic project networks in which activities are only selected for implementation throughout the project progress. Tsamardinos et al. (2003) present an approach to conditional planning where the presence of an activity in the final solution depends on external conditions. Instead of enumerating all possible precedence networks, the authors propose to extend the temporal constraint satisfaction problem (TCSP) with observation nodes and labels to indicate which activities will be executed in different situations. Beck and Fox (2000) define the concept of the probability of existence (PEX) in order to model alternative activities. The expected PEX value of an activity represents the probability that the activity will be present in the final solution. Although the previously mentioned studies include logical constraints apart from temporal constraints in scheduling problems, they fail to exploit the integration between both constraints. However, Barták et al. (2008) extend temporal networks to deal with alternatives in such a way that logical and temporal constraints are closely related. Building on prior research, the authors propose a restricted temporal network with alternatives (TNA), called nested TNA. This reduces the complexity of the problem to ensure a consistent selection of activities given prior decisions. Moreover, we should also mention the research of Ranjbar and Davari (2013) with respect to the alternative-technologies project scheduling problem. The authors investigate innovative Research and Development (R&D) projects in which several alternative technologies might exist and, therefore, certain alternatives will be interrupted

or even ignored depending on realised successes within the project.

Based on this aforementioned research on alternatives in project scheduling, several papers propose each a different problem class that extends the RCPSP to include logical constraints. Belhe and Kusiak (1995) observe that design processes comprise of different types of logical relationships between activities: AND-, OR- and exclusive OR-relations. The key difference between those types of relations is that the former relation does not exclude activities during schedule generation, while the latter two relations might. Gillies and Liu (1995) investigate the complexity of the resource-constrained scheduling problem with generalised AND/OR-constraints as well as propose two heuristic solution procedures. Moreover, Kuster et al. (2008) models project structures with mutual exclusion and inclusion relationships between the activities in order to solve disruption management problems. Capek et al. (2012) extend the RCPSP with alternative process plans that differ in terms of activity, precedence and resource characteristics. The authors propose an integrated approach to select a subset of activities that constitute a process plan and schedule the selected activities in order to minimise the total project makespan. Kellenbrink and Helber (2015) introduce an extension of the RCPSP with model-endogenous decisions on the project structure. In contrast to the current approach, they relax the assumption that any dependency between the logical and precedence relations should exist. Moreover, Vanhoucke and Coelho (2016) propose a novel approach to include logical constraints in the RCPSP. The additional OR- and BI-relations are converted into standard AND-relations using a set of transformation rules. However, a satisfiability (SAT) solver should be included in these algorithms to retain the logic of the transformed precedence relations. Finally, Tao and Dong (2017) present an extension of the RCPSP that considers alternative activity chains (RCPSP-AC). The authors introduce a modified project network, so-called AND-OR project network, to represent the alternative structure and they develop a simulated annealing procedure to solve large-scale problem instances of the RCPSP-AC.

An extensive amount of research has been devoted to the inclusion of alternatives in the job-shop scheduling problem (JSP). Kis (2003) studies the JSP with processing alternatives (AJSP). In this extension, the routing of jobs is represented by a directed, acyclic graph that consists of AND- and OR-subgraphs. The former indicates a parallel execution of the operations, while the latter consists of alternative subroutings of which exactly one subrouting should be selected. Another field of research that is closely related to project scheduling and that already considered the inclusion of alternatives is assembly line balancing. Motivated by real-life cases, Capacho and Pastor (2006) study the simple

assembly line balancing problem with alternatives (ASALB) in which certain parts of the assembly line can be executed in different modes that correspond to different assembly variants. In this research, a mathematical model is proposed to minimise the number of workstations given a maximum cycle time in the ASALB. Later, Capacho et al. (2009) extend the ASALB literature with a heuristic method to select assembly variants using priority rules and random choices.

The current paper is founded on research efforts carried out in various fields over the past decades, as discussed above, and was mainly inspired on the work of Kis (2003), Capacho et al. (2009), Capek et al. (2012) and Kellenbrink and Helber (2015).

3. Problem formulation

In section 3.1, we give a brief overview of the general problem that is considered in this study, while the problem formulation is discussed in more detail in section 3.2. Section 3.3 introduces the alternative subgraph matrix and provides formal definitions for the different types of alternative subgraphs that are labelled in this classification matrix. Finally, we provide an example to support the practical relevance of the RCPSP-AS in section 3.4.

3.1. Problem overview

Resource-constrained project scheduling requires a set of activities to be performed on a set of resources, while taking into consideration both the precedence and resource constraints. The project structure is explicitly assumed deterministic in the basic situation, while there might exist multiple alternative ways to execute certain work packages in practice. Each work package is represented by a subproject, i.e. a set of interrelated activities with specific precedence relations and resource requirements. The decision to relax the assumption of a deterministic project structure, therefore, results in two types of work packages. First of all, there exist work packages without alternatives for which the activities in the corresponding subprojects should always be executed. Secondly, there exist work packages that consist of multiple alternatives. This implies that the activities in the subprojects that correspond to these alternatives are considered optional. Furthermore, there might exist dependencies between alternative execution modes of work packages. For example, one alternative might trigger an additional choice amongst alternative execution modes of a work package or different alternatives might be connected. Since each alternative of a work package is characterised by specific precedence relations and resource constraints,

decisions in the presence of such dependencies have a considerable impact on the quality of the resulting project schedule. Hence, we are confronted with a complex selection and scheduling problem. In order to select alternatives for work packages such that the objective of the scheduling problem is optimised, the selection and the scheduling process should be integrated.

3.2. Problem formulation

The traditional RCPSP investigates the situation where a set of activities needs to be performed on a set of resources given precedence relations between those activities. Let $G = (N, A)$ be a directed acyclic graph that represents the activity-on-the-node (AoN) project network, where the project consists of a set N of activities and A represents the set of pairs of activities between which a finish-to-start precedence relation with zero minimum time lags exists. It is typically assumed that the activities in G are topological ordered starting from a dummy start activity 0 to a dummy end activity $n+1$. These dummy activities ensure that the AoN-network has exactly one starting and finishing node, such that each non-dummy activity is precedence related to the dummy start and end activity. The n non-dummy activities 1 to n should be scheduled on a set R^{ρ} of renewable resource types for which the per-period available amount of each resource type k a_k^{ρ} is a priori known and assumed constant over time. Each activity i in the set N is characterised by an activity duration d_i and a resource requirement for each resource type $r_{i,k}$. The dummy activities in the project network are defined by a deterministic activity duration equal to zero and a resource requirement likewise equal to zero. We assume that activities cannot be interrupted, i.e. no pre-emption, which implies that once started an activity i should be completely finished after d_i time units. A schedule S is defined by a vector of start times s_i for all activities in the set N. When such a schedule satisfies all precedence and renewable resource constraints, it is called a feasible schedule. The objective of the RCPSP is to find the feasible schedule with the minimum makespan for a project.

The traditional RCPSP assumes that the project structure is deterministic. This implies that all activities $i \in N$ should be present in the optimal schedule S^{opt}, i.e. the feasible schedule with minimal makespan, and consequently the precedence and resource constraints of all activities should be satisfied. However, there are practical situations in which activities can be excluded from the schedule, while the project still delivers on the promised outcomes. The resulting problem is referred to as the resource-constrained project scheduling problem with alternative subgraphs (RCPSP-AS). In this problem formulation,

the set N of activities consists of two mutually exclusive and collectively exhaustive subsets of activities: the set N^f of *fixed activities* and the set N^a of *alternative activities*.

Fixed activity This activity should always be executed in order to complete the project.
Alternative activity The presence of this activity in the schedule is optional. Therefore, the precedence and resource constraints of an alternative activity should only be satisfied when the activity is included in the schedule.

3.3. Alternative subgraphs and numerical example

Given the definition of fixed and alternative activities, it is possible to describe project scheduling problems with alternative subgraphs. When we consider how these alternative activities are related to each other in the project network, we can distinguish between different types of alternative subgraphs. First of all, it is possible that a work package that can be executed in different alternative ways is embedded in an alternative of another work package. In this situation, the selection process becomes more complex as the decision to execute a work package in one alternative way might trigger an additional decision in another work package with alternatives. In the remainder of this paper, we refer to projects with such a project structure as *nested projects*. Secondly, it might be that an alternative for one work package is connected to an alternative of the same or a different work package. This implies that decisions made in different work packages with alternatives are interconnected. Projects that consist of this type of project structure are called *linked projects*.

We propose a dichotomous classification of the different types of alternative subgraphs based on the above two dimensions in a 2x2-matrix, the so-called *alternative subgraph matrix* (ASM) (see figure 1). The ASM provides a comprehensive classification of important, practical cases of alternative subgraphs. These cases are already discussed in resource management in general and project scheduling in particular, which provides practical and theoretical justification for the ASM. In general, there exist different common types of resource management decisions: a single preferred alternative, linked choices and decisions within decisions (Gregory et al., 2012). Since resource-constrained project scheduling is part of general resource management, these types of decisions are also covered in the ASM. In particular, several real-life examples and case studies have been presented in the field of project scheduling to support the practical relevance of the ASM. Capek et al. (2012) distinctly exclude an interleaving between alternative branches (i.e. linked alternative branches) in their problem formulation and textually mention the concept of nested

alternatives. Capacho et al. (2009) and Capacho and Pastor (2006) also mention that it is possible and useful to nest alternative subgraphs.

The formal definition of the different types of alternative subgraphs in each of the quadrants of the ASM is given in the remainder of this section. The introduced terminology is derived from various research papers (Beck and Fox, 2000; Kis, 2003; Capacho and Pastor, 2006; Barták et al., 2008; Scholl et al., 2009; Capek et al., 2012; Kellenbrink and Helber, 2015) with the sole purpose of providing a coherent problem formulation. The problem-specific parameters of the RCPSP-AS are summarised in table 1. For the sake of clarity, the definitions described below are illustrated based on the examples shown in each of the four quadrants of figure 1 (see table 2). Each curved line '**)**' in figure 1 marks a choice between alternative branches in an alternative subgraph.

In order to provide a clear explanation for each of the different types of alternative subgraphs, we define three **types of activities**: the *principal activity* $\rho \in N^p$, the *branching activity* $b_{\rho,z} \in N^b$ and the *terminal activity* $\tau_\rho \in N^t$.

Principal activity $\boldsymbol{\rho}$ A principal activity causes the decision amongst different choices in the project structure. Therefore, this fixed or alternative activity should have at least two mutually exclusive direct, alternative successors of which, respectively, exactly or at most one should be selected.

Branching activity $\boldsymbol{b_{\rho,z}}$ A branching activity is an alternative activity that is the direct successor of a principal activity ρ. The z branching activities represent the start of each of the z choices that are caused by ρ.

Terminal activity $\boldsymbol{\tau_\rho}$ A terminal activity terminates the decision amongst different choices in the project structure caused by ρ. Therefore, this fixed or alternative activity should have z direct, alternative predecessors that correspond with the z choices that are caused by ρ.

Based on this terminology, we will define three **types of graphs**: the set of *alternative subgraphs* $\mathscr{P}$, the set of *alternative branches* $\mathscr{Q}$ and the set of *alternative paths* $\mathscr{U}$. We identify alternative subgraphs in order to provide a formal definition for a work package with alternatives, where each alternative way to execute this work package is considered an alternative branch of the corresponding alternative subgraph. Given the different alternative branches in each of the alternative subgraphs, a unique way to execute the project is formally defined by an alternative path.

Alternative subgraph $\boldsymbol{P_\rho}$ An alternative subgraph is an induced subgraph of G consist-

	Symbol	Definition
Activities	N^f	Set of fixed activities ($N^f \subseteq N$)
	N^a	Set of alternative activities ($N^a \subset N$)
	N^p	Set of principal activities ($N^p \subset N$)
	N^b	Set of branching activities ($N^b \subseteq N^a$)
	N^t	Set of terminal activities ($N^t \subset N$)
	N^s	Set of selected alternative activities ($N^s \subseteq N^a$)
Graphs	$\mathscr{P}$	Set of alternative subgraphs ($\|\mathscr{P}\|$ = Number of alternative subgraphs)
	P_ρ	Alternative subgraph ($P_\rho \in \mathscr{P}$ and $P_\rho \subset N$)
	$\mathscr{Q}$	Set of alternative branches ($\|\mathscr{Q}\|$ = Number of alternative branches)
	$Q_{\rho,z}$	Alternative branch ($Q_{\rho,z} \in \mathscr{Q}$, $z \in \{1, ..., \|P_\rho\|\}$ and $Q_{\rho,z} \subseteq P_\rho$) with $\|P_\rho\|$ = Number of alternative branches in alternative subgraph P_ρ
	$\mathscr{U}$	Set of alternative paths ($\|\mathscr{U}\|$ = Number of alternative paths)
	u	Alternative path ($u \in \mathscr{U}$ and $u \subseteq N$)

Table 1: Specific terminology of the RCPSP-AS

ing solely of alternative activities that originate from the same principal activity ρ with the inclusion of the corresponding terminal activity τ_ρ.

Alternative branch $Q_{\rho,z}$ An alternative branch $Q_{\rho,z}$ is a subset of activities in the alternative subgraph P_ρ that consist of the branching activity $b_{\rho,z}$ as well as all its transitive successors in P_ρ.

Alternative path u An alternative path u is a subset of N that consists of the set of fixed activities N^f and the logical feasible set of selected alternative activities N^s, i.e. $u = \{N^f \cup (N^s \subseteq N^a)\}$. The set N^s should satisfy the following two conditions. Condition (1) guarantees that exactly one branching activity is selected when the corresponding principal activity is selected. Condition (2) ensures the correct selection of non-branching activities.

1. $p \in N^p \cap (N^f \cup N^s) \implies \exists! b_{p,z} \in N^s$
2. $k \in N^f \cup N^s \setminus N^p \& i \in N^a \& (k,i) \in A \implies i \in N^s$

Given this terminology, we propose the following definitions for the different types of alternative subgraphs:

- $G(N, A)$ is a non-linked non-nested project $\Leftrightarrow$ (1)&(3)
- $G(N, A)$ is a non-linked nested project $\Leftrightarrow$ (2)&(3)

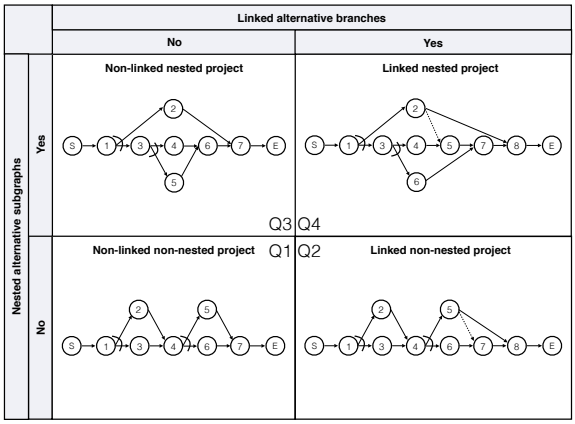


Figure 1: Alternative subgraph matrix

- $G(N, A)$ is a linked non-nested project $\Leftrightarrow$ (1)&(4)
- $G(N, A)$ is a linked nested project $\Leftrightarrow$ (2)&(4)

with

$$N^p \cap N^a = \emptyset \tag{1}$$

$$N^p \cap N^a \neq \emptyset \tag{2}$$

$$\forall k \in N^a \setminus N^p \& \forall i \in N^a \setminus N^t \& \nexists Q_{p,z} \in \mathcal{Q} \& (k,i) \in A \& k \notin Q_{p,z} \& i \in Q_{p,z} \tag{3}$$

$$\exists k \in N^a \setminus N^p \& \exists i \in N^a \setminus (N^b \cup N^t) \& \exists Q_{p,z} \in \mathcal{Q} \& (k,i) \in A \& k \notin Q_{p,z} \& i \in Q_{p,z} \tag{4}$$

In a **non-linked non-nested project**, all principal activities are fixed activities and any two alternative activities that are precedence related should belong to the same alternative branch. A **non-linked nested project** has at least one principal activity that belongs to the set of alternative activities. Any two precedence-related, alternative activities should belong to the same alternative branch. However, this rule should not hold in case that the first (second) alternative activity in the precedence relation is a principal (terminal) activity. In a **linked non-nested project**, all principal activities are fixed activities, similar to a non-linked non-nested project. Also, there exists at least one precedence-related pair of alternative activities in which each alternative activity belongs to a different alternative branch. This part of the definition allows the existence of linked alternative branches. A **linked nested project** has at least one principal activity that belongs to the set of alternative activities, similar to a non-linked nested project. Also, there is at

least one precedence-related pair of alternative activities such that there exists at least one alternative branch that contains the second but not the first alternative activity. This indicates the existence of linked alternative branches. However, this rule should not hold in case that the first (second) alternative activity in the precedence relation is a principal (terminal) activity. In order to avoid poorly configured alternative subgraphs, the second alternative activity cannot be a branching activity in the linked (non-) nested projects. Note that condition (2) is a negation of condition (1) for the definition of the different types of alternative subgraphs, while this is not the case for conditions (3) and (4). Consequently, a project structure that does not satisfy condition (3) nor (4) does not result in a valid project.

The objective of the RCPSP-AS is to select for each alternative subgraph P_ρ at most one alternative branch $Q_{\rho,z}$ such that the corresponding logical feasible alternative path u results in a precedence and resource feasible schedule with minimal project makespan. In table 2, the discussed terminology is applied to the examples in figure 1 for the sake of clarification. This terminology establishes one of the contributions of the presented paper since it provides a unique problem definition for the RCPSP-AS.

3.4. Practical relevance

In this section, we will present a practical example to illustrate the idea behind nested alternative subgraphs and linked alternative branches. Figure 2 shows a simplified representation of part of a residential apartment building project in which the nodes resemble work packages that consist of multiple activities. The project can be classified as a linked

Non-linked nested project						**Linked nested project**					
N^f	{S,1,7,E}	N^a	{2,3,4,5,6}	N^p	{1,3}	N^f	{S,1,8,E}	N^a	{2,3,4,5,6,7}	N^p	{1,3}
N^b	{2,3,4,5}	N^t	{6,7}			N^b	{2,3,4,6}	N^t	{7,8}		
P_1	{2,3,4,5,6,7}	$Q_{1,1}$	{2,7}	$Q_{1,2}$	{3,4,5,6,7}	P_1	{2,3,4,5,6,7,8}	$Q_{1,1}$	{2,5,7,8}	$Q_{1,2}$	{3,4,5,6,7,8}
P_3	{4,5,6}	$Q_{3,1}$	{4,6}	$Q_{3,2}$	{5,6}	P_3	{4,5,6,7}	$Q_{3,1}$	{4,5,7}	$Q_{3,2}$	{6,7}
$\mathscr{U}$	{{S,1,2,7,E},{S,1,3,4,6,7,E},{S,1,3,5,6,7,E}}					$\mathscr{U}$	{{S,1,2,5,7,8,E},{S,1,3,4,5,7,8,E},{S,1,3,6,7,8,E}}				
Non-linked non-nested project						**Linked non-nested project**					
N^f	{S,1,4,7,E}	N^a	{2,3,5,6}	N^p	{1,4}	N^f	{S,1,4,8,E}	N^a	{2,3,5,6,7}	N^p	{1,4}
N^b	{2,3,5,6}	N^t	{4,7}			N^b	{2,3,5,6}	N^t	{4,8}		
P_1	{2,3,4}	$Q_{1,1}$	{2,4}	$Q_{1,2}$	{3,4}	P_1	{2,3,4}	$Q_{1,1}$	{2,4}	$Q_{1,2}$	{3,4}
P_4	{5,6,7}	$Q_{4,1}$	{5,7}	$Q_{4,2}$	{6,7}	P_4	{5,6,7,8}	$Q_{4,1}$	{5,7,8}	$Q_{4,2}$	{6,7,8}
$\mathscr{U}$	{{S,1,2,4,5,7,E},{S,1,2,4,6,7,E},{S,1,3,4,5,7,E}, {S,1,3,4,6,7,E}}					$\mathscr{U}$	{{S,1,2,4,5,7,8,E},{S,1,2,4,6,7,8,E},{S,1,3,4,5,7,8,E}, {S,1,3,4,6,7,8,E}}				

Table 2: Illustration of terminology in table 1 based on figure 1

nested project and consists of two sequential phases. Phase 1 focuses on the customised construction and on-site installation of the terrace sliding door frames. In phase 2, the water tightness of the metal frame should be tested as it might deform after the installation due to climatological factors.

Nested alternative subgraphs. There exist two alternative ways to execute the work in the first phase. In work package A, the metal frames are mass-produced for the entire building and thus the potential reformation of the metal is neglected. A second option, represented as work package B, is to construct a single frame, which is then subjected to climatological factors such that the reformed frame can be used as a cast during mass production. In this way, the frame will no longer reform after installation. In order to proactively reform the frame, it can be placed onto the construction site (C) or reformed under controlled settings in a laboratory (D). One can observe that the choice between nodes C and D is nested in the choice for node B.

Linked alternative branches. There exist two alternative ways to execute stage 2 as either all apartment units (E, F and G) or a random sample of all apartment units (H, I, J and G) is tested to guarantee water tightness. The testing requires pumping water onto the terrace and checking the installed frame for any leaks. The first option to test the water tightness is to perform a 100% testing of the window frames in the apartment units. Therefore, one has to measure the pressure equalisation (E), analyse the drainage design (F) and perform a quality check of the metal frame (G). The second option is to perform a random sampling testing and thus only test the water tightness in a selected number of apartment units. In this case, the measurement of pressure equalisation (H) and the analysis of drainage design (I) are again performed, however, on a limited number of window frames. Subsequently, a sensitivity analysis (J) should be conducted to guarantee an overall acceptable level of water tightness in the entire apartment building. In case that the frames are mass-produced without cast (A), a 100% analysis of the drainage design (F) is required since no proactive actions were taken to avoid reformation of the metal frames. We incorporate the above situation in the project through a link between work packages A and F. The quality check of work package G should also be executed in case that the pressure equalisation is measured through a random sampling technique (H). We model this situation through a link between work packages H and G.

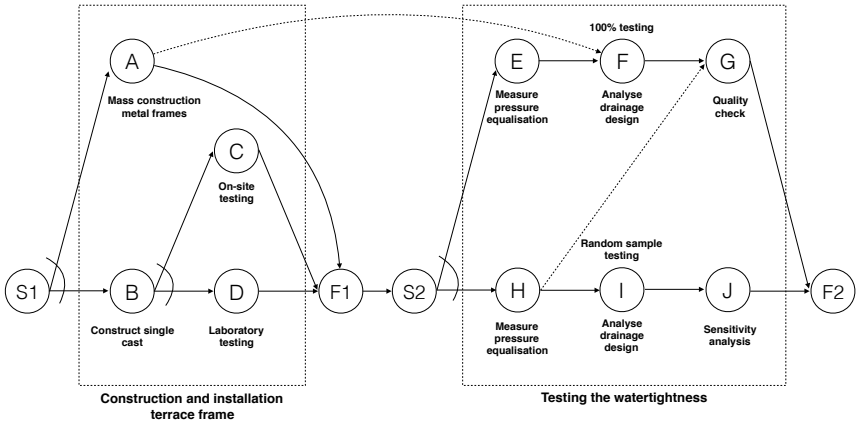


Figure 2: Illustrative example of linked nested project

4. Solution procedure

In section 4.1, we describe our general solution approach with building blocks that are customised to either the selection or the scheduling subproblem. We provide a detailed discussion of the different strategies for these building blocks in section 5. In section 4.2, we explain in detail the dynamic quadrant selection procedure that positions problem instances in the ASM as this is a key feature of the proposed solution approach. For the sake of clarity, table 3 provides a summary of the abbreviations and their explanation used in this paper.

4.1. General approach for the RCPSP-AS

In this research, we implement a tabu search (TS) metaheuristic (Glover, 1986, 1989, 1990) in order to solve the RCPSP-AS. An initial solution is generated to start the TS and, subsequently, we proceed to the first iteration of the neighbourhood (NH) search. The best, non-tabu move in the NH, i.e. the move that results in the lowest project makespan, determines the new solution. However, a tabu move that results in a solution with a better objective value than the overall best solution encountered so far is accepted nevertheless. This is a widely applied aspiration criterion in TS procedures to allow further investigation of interesting areas of the solution space. The best move is subsequently added to the top of the tabu list and, in case the tabu list is completely stocked, the move at the bottom of the tabu list is made admissible again. Before the TS moves to the next iteration, the current solution is further improved using a local search (LS). In case that the current

Abbreviation	Explanation	Abbreviation	Explanation
General problem		*Representation*	
ASM	Alternative subgraph matrix	CL	Choice list
%flex	Degree of flexibility	RK	Random key
%nested	Degree of nested alternative subgraphs	*Initial solution*	
%linked	Degree of linked alternative branches	IN-1	Weight-based strategy
RC	Resource constraindness	IN-2	Random strategy
SP	Serial-parallel indicator	IN-3	Relaxation strategy
General procedure		IN-4	Priority-rule strategy
TS	Tabu search	*Neighbourhood*	
NH	Neighbourhood	NH-1	Improved alternative swap
LS	Local search	NH-2	Inserted alternative swap
MSILS	Multi-start iterated local search	NH-3	Random alternative swap
SOD	Sum of durations	NH-4	Adjecent pairwise interchange swap
TWC	Total work content	*Local search*	
PEX	Probability-of-existence	LS-1	CL adjustment
CPLB	Critical path lower bound	LS-2	Forward-backward scheduling
API	Adjecent pairwise interchange operator		
TL-SCH	Tabu list - scheduling subproblem		
TL-SEL	Tabu list - selection subproblem		

Table 3: Summary list of abbreviations

solution is improved, the new solution is passed on to the next iteration of the TS. The metaheuristic generates a pre-specified number of schedules after which the overall best solution is returned as the final solution.

An important aspect of any metaheuristic is the solution representation. In our research, a schedule is represented by a solution vector that consists of two lists: the random key (RK) and the choice list (CL). The RK is a popular schedule representation in the RCPSP literature that determines the scheduling priority of activities. The binary CL controls whether an activity will be present or not (respectively a value equal to 1 or 0) in the final schedule, similar to the mode list frequently used in the MRCPSP. We refer to the work of Kolisch and Hartmann (1998) for a detailed explanation of the different solution representations that are used for the RCPSP.

We propose a solution procedure that alternates between two search spaces: solutions for the selection subproblem will focus on the selection of alternatives, while solutions for the scheduling subproblem will optimise the scheduling priorities of the selected activities. Since changes in the set of selected alternatives disrupt the current solution, much information that is captured in this solution is lost. In order to avoid the depreciation of our solution procedure to a simple random walk, we propose to focus less frequently on the selection subproblem than on the scheduling subproblem. Consequently, a pre-specified

number of iterations designated to the scheduling run are executed for each iteration of the selection run to intensify the search for a high-quality schedule. An overview of the procedure is given in figure 3. We initiate the procedure with a selection run, which consists of a NH and a LS, to improve the set of selected alternatives in the project. After each iteration of the selection run, we execute a pre-specified number of iterations of the scheduling run. Each scheduling run again consists of two components: a NH and a LS. Afterwards, the solution that results from the scheduling run is passed on to the next iteration of the selection run. Hence, we will further investigate whether there exists a better set of selected activities given the new scheduling priorities. After each selection and scheduling run, we determine the quality of the obtained solution by scheduling the selected activities in the CL given the scheduling priorities in the RK using the serial schedule generation scheme (SSGS). The SSGS will consider each selected activity one at a time based on the RK and schedule them as soon as possible within the precedence and resource constraints (Kelley Jr., 1963). The procedure visits 5,000 schedules after which the overall best solution is reported.

4.2. Dynamic quadrant selection

In our solution approach, we propose to guide the search towards a favourable set of alternatives based on the position of the problem instance in the ASM. The general idea can be explained as follows. Initially, each alternative branch is assigned a value, which is referred to as *weights* in this paper. Subsequently, operators to select the best possible alternative branch will use these weights as input during initialisation, neighbourhood and local search. However, different operators exist for the different quadrants in the ASM and a project can be positioned in each quadrant with a certain probability. In the search procedure, a project instance will be assigned to a specific quadrant based on these probabilities and the corresponding operator will be applied. Each time that alternative branches need to be selected, the quadrant selection will be reconsidered and, hence, it is called *dynamic quadrant selection*.

Calculation of weights. First of all, weights are assigned to each of the alternative branches based on a measure that either captures information from the network, i.e. the sum of durations (SOD), or from the resource usage, i.e. the total work content (TWC) (Boctor, 1993). These weights guide the selection of alternative branches in the proposed solution procedure. However, the calculations of the SOD and TWC differ between the quadrants in the ASM:

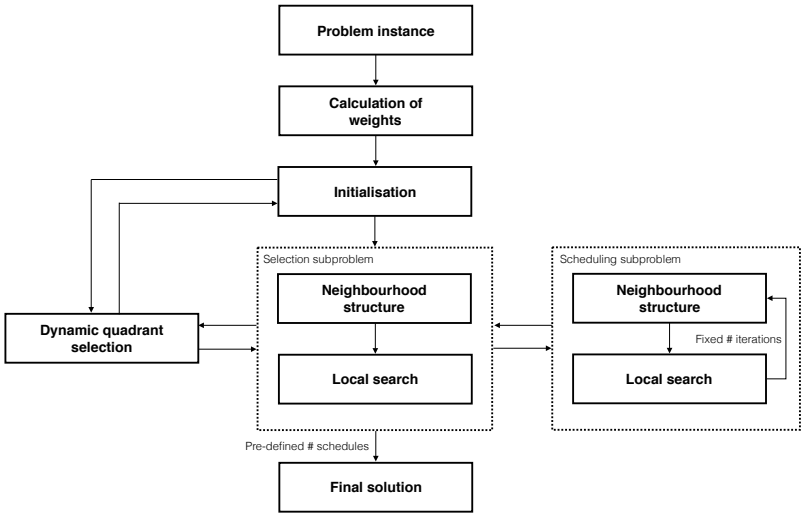


Figure 3: Overview of solution approach for the RCPSP-AS

Q1 For non-linked non-nested projects, these weights can easily be applied to prioritise alternative branches. The SOD (TWC) of an alternative branch equals the sum of the duration (work content) of the alternative activities that belong to this alternative branch.
Q2 In case of linked non-nested projects, the SOD (TWC) of an alternative branch is extended with the SOD (TWC) of all its linked alternative activities. This is because the SOD (TWC) of linked alternatives should be accounted for in the selection process.
Q3 In case of non-linked nested projects, the average SOD (TWC) of the alternative branches of a nested alternative subgraph are added to the weight of the enclosing alternative branch. This is because one of the alternative branches in the enclosed alternative subgraph needs to be scheduled, however, it is uncertain which one.
Q4 For linked nested projects, the calculations of the SOD (TWC) for linked non-nested projects (Q2) and non-linked nested projects (Q3) are combined.

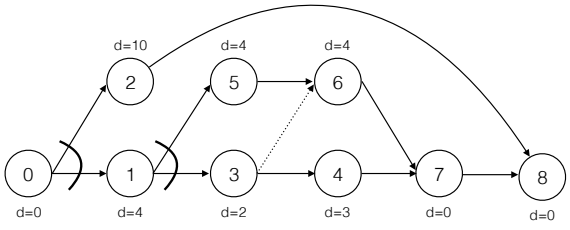
Dynamic use of weights. Subsequently, we position the problem instance with alternative subgraphs in the ASM. However, we will not consider the two dimensions of the ASM as binary parameters that mark the presence or absence of nested alternative subgraphs and linked alternative branches. On the contrary, the two dimensions will express a degree of nested alternative subgraphs (%nested) and linked alternative branches (%linked) in the range [0,1]. In this way, we convert the four-quadrant ASM into a two-dimensional grid, such that a problem instance can be positioned in each quadrant of the ASM with

a certain probability. For example, a problem instance with alternative subgraphs that is characterised by a %nested and %linked equal to, respectively, 0.75 and 0.25 will be positioned in each of the four quadrants of the ASM with the following probabilities:

- Non-linked non-nested project = $(1-\%nested)*(1-\%linked) = (1-0.75)*(1-0.25) = 0.1875$
- Linked non-nested project = $(1 - \%nested) * \%linked = (1 - 0.75) * 0.25 = 0.0625$
- Non-linked nested project = $\%nested * (1 - \%linked) = 0.75 * (1 - 0.25) = 0.5625$
- Linked nested project = $\%nested * \%linked = 0.75 * 0.25 = 0.1875$

The values for %nested and %linked are specific to each project instance and can be easily calculated in advance. Details are provided in section 6.1. Based on the above probabilities, it is decided which operator will be used to select the alternative branches in the building blocks of the procedure. However, the assignment of a problem instance to a quadrant of the ASM, and thus the selection of the appropriate operator, is reconsidered each time an alternative branch needs to be selected. Observe that the illustrative example discussed above can be positioned in each of the four quandrants with a non-zero probability. This implies that the project can be classified as non-linked and/or non-nested despite that it contains nested alternative subgraphs (%flex=75%) and linked alternative branches (%linked=25%). This misclassification is crucial to include randomness in the solution procedure, however, it is less probable for project instances with more distinct characteristics (i.e. very high or low %linked and %nested). Due to this misclassification, the formulae presented in the previous paragraph cannot be applied in a straightforward way. On the one hand, the formulae are self-evident in case that linked alternative branches and nested alternative subgraphs are absent in the project, but considered in the formulae. On the other hand, the linked alternative branches and nested alternative subgraphs are neglected in the project in case that they are not considered in the respective formula. This implies that the activities in nested alternative subgraphs or linked alternative branches are assumed dummy activities in case that the nested alternative subgraphs or linked alternative branches are neglected. In order to clarify the calculations of the weights, we have included a simplified example of a linked nested project in figure 4. The project consists of two alternative subgraphs 1-2-3-4-5-6-7-8 and 3-4-5-6-7 that consist of two alternative branches each, respectively, alternative branch 1-3-4-5-6-7-8 and alternative branch 2-8 in the first alternative subgraph and alternative branch 3-4-6-7 and alternative branch 5-6-7 in the second alternative subgraph. The computations of the weigths based on the SOD

for the different alternative branches in case that the project is classified in each of the four quadrants is detailed in the corresponding table.



	Weight 1-3-4-5-6-7-8	Weight 2-8	Weight 3-4-6-7	Weight 5-6-7	Selected alternative path using SOD
Non-linked non-nested	4	10	5	8	Alternative path 0-1-3-4-6-7-8
Linked non-nested	4	10	9	8	Alternative path 0-1-5-6-7-8
Non-linked nested	10,5	10	5	8	Alternative path 0-2-8
Linked nested	12,5	10	9	8	Alternative path 0-2-8

Figure 4: Illustration of weigth calculations based on the SOD for the different quadrants

5. Tabu search

This section provides a detailed discussion of the different strategies for the three building blocks in the proposed TS: initial solution (section 5.1), neighbourhood structure (section 5.2) and local search (section 5.3). The generic procedure of the TS is shown in figure 5, which presents a more detailed implementation of the solution approach in figure 3.

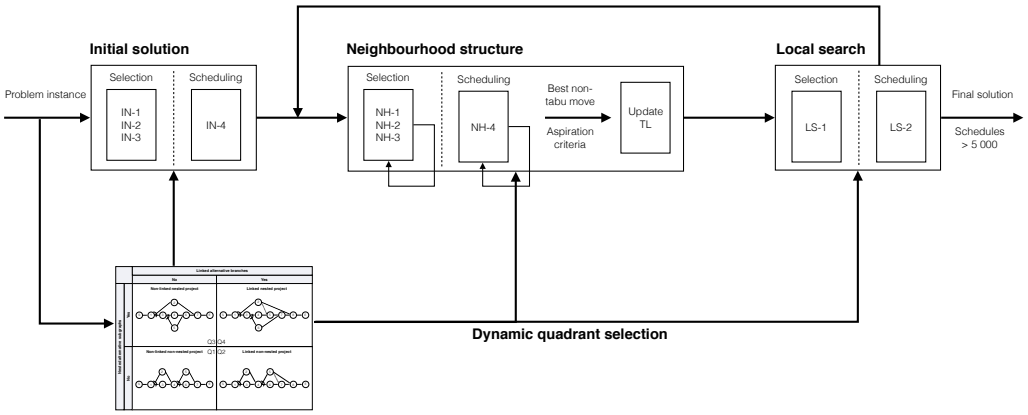


Figure 5: Generic tabu search procedure

5.1. Initial solution

In this research, two approaches can be followed to obtain a good initial solution for the RCPSP-AS. Either the set of selected alternatives is controlled and the RK is randomly generated, i.e. a controlled CL, or the generation of the RK is controlled and based on these

scheduling priorities the CL is determined, i.e. a controlled RK. An experiment is designed to determine the best approach for the generation of a high-quality initial solution.

Three controlled CL approaches to prioritise the alternative branches in an alternative subgraph are compared in this research. The RK is each time randomly generated.

Weight-based strategy (IN-1) The first approach assigns a duration- or resource-based weight to each alternative branch. Consequently, the alternative branches with the lowest weight is selected for each alternative subgraph in such a way that the logical feasibility is guaranteed. The dynamically adjusted weights are calculated as shown in section 4.2, which results in four duration (resource)-based weights, labeled IN-1.1 (IN-1.5) up to IN-1.4 (IN-1.8) for Q1 up to Q4.

Random strategy (IN-2) The random approach for the controlled CL implies that for each alternative subgraph one alternative branch is selected at random bearing in mind the logical feasibility.

Relaxation strategy (IN-3) This approach constructs a controlled CL based on information obtained from the optimal solution of the unconstrained version of the problem. This approach is already applied in literature to derive priority rule-based heuristics (Russell, 1986). The unconstrained version of the RCPSP-AS assumes that an infinite amount of renewable resources are available and the project structure is classified as a non-linked (non-) nested project. Under these assumptions, the RCPSP-AS can be solved in polynomial time using the procedure described by Capek et al. (2012). The set of selected alternatives in the relaxation is used to construct the CL for the general problem.

In the controlled RK approach, priority values are systematically assigned to each activity. Since a schedule should also be logical feasible, we propose to construct the CL based on the RK as follows: an activity cannot be selected (i.e. cannot assign a value equal to 1 in the CL) based on the RK when this causes a logical conflict with a higher-priority, selected activity.

Priority-rule strategy (IN-4) The priority-rule strategy assigns a priority to each activity that increases as the activity has a lower earliest start time (EST) according to the critical path calculations (Kelley Jr., 1963) and the length of the alternative branch that encompasses the activity decreases. This priority rule is derived from Capek et al. (2012). Similar to this author, we put more emphasis on the first than on the second component of the calculation as we weigh the components with a value equal to, respectively, 5 and 3.

5.2. Neighbourhood structure

At each iteration, the TS procedure searches for the best, non-tabu neighbour of the current solution. The NH of a solution is defined by the set of solutions that can be accessed from the current solution by means of a single move. We propose four types of moves, which result in four distinct NH structures. The first group of moves act on the CL (NH-1, NH-2 and NH-3), while the second group of moves act on the RK (NH-4). The first group of moves swap every selected alternative branch with each other alternative branch in the same alternative subgraph. In case that a selected or deleted alternative branch consists of nested alternative subgraphs or linked alternative branches, all the related alternative activities need to be, respectively, selected or deleted as well. When an alternative branch that contains a nested alternative subgraph is selected, it goes without saying that only one of the alternative branches in the enclosed alternative subgraph can be selected. However, there exist a potentially large number of neighbours in this NH. In a project with x alternative subgraphs and y alternative branches for each alternative subgraph, the NH size grows in the worst case to $(x^y - 1)$. Therefore, we restrict the size of the NH in the approaches NH-1, NH-2 and NH-3 in the following way. Each selected alternative branch is swapped with each unselected alternative branch in the same alternative subgraph, but we consider the alternative subgraphs independent of each other. This implies that the alternative branches in one alternative subgraph are swapped, while the selection of alternative branches in the other, unrelated alternative subgraphs is retained.

Improved alternative swap (NH-1) In this NH, a selected alternative branch is only swapped with an unselected alternative branch that has an improved dynamically adjusted weight calculated as shown in section 4.2.

Inserted alternative swap (NH-2) In this NH, a quick check is performed to ensure that the considered neighbour has the potential to improve the current solution without the need to generate a complete schedule. Therefore, a lower bound on the duration of the neighbouring alternative branch is computed (section 6.2.1). Subsequently, we delete the selected alternative branch from the current schedule and check whether the neighbour can be inserted in the schedule based on the lower bound. Therefore, we consider the start time of the branching activity and the finish time of the terminal activity of the selected alternative branch in the schedule. However, we use the position of the problem instance in the ASM to determine the terminal activity as we regard or disregard the linked and nested alternatives. In case that the lower bound is higher than or equal to the time interval in

which the selected alternative branch was scheduled, the quality of the solution cannot be improved through the proposed swap and the swap is excluded from the NH. In the other case, the swap will be executed.

Random alternative swap (NH-3) In this NH, each selected alternative branch is swapped with one random, unselected alternative branch in the same alternative subgraph.

Adjacent pairwise interchange (NH-4) The NH of a solution consists of the set of solutions that can be obtained from the current solution by swapping two activities in the schedule that are adjacent, but not precedence related (Morton and Pentico, 1993). We limit the NH by swapping only the critical activities with their adjacent activities.

5.3. Local searches

We again consider two types of LS procedures: one procedure attempts to improve the choices amongst alternatives with given scheduling priorities (i.e. LS-1), while the other procedure attempts to improve the schedule given a rigid set of decisions (i.e. LS-2).

CL adjustment (LS-1) For each selected alternative branch in the current solution, there is a probability that the CL adjustment will be executed. This probability is derived from the concept of the PEX (Beck and Fox, 2000). In this paper, however, the PEX calculations are considered at the level of the alternative branches rather than the activity level. The probability of existence (PEX) of an alternative branch is determined based on the number of alternative branches in the alternative subgraph and whether or not the alternative subgraph is nested. In case that there exist five alternative branches in one stage, the PEX of each alternative branch is equal to 20%. Let us assume now that there exist two stages with each five alternative branches, however, the five alternative branches in the second stage are nested in one of the alternative branches in the first stage. In this case, the PEX of each alternative branch in the second stage is equal to 4% ($= 20\% * 20\%$). For each non-selected alternative branch in an alternative subgraph with a selected alternative branch, we draw a value uniformly at random from the interval [0,1]. In case that the value is lower than the PEX-value of the alternative branch, this alternative branch is selected given that its dynamically adjusted weight is higher than the currently selected alternative branch to avoid cycles in the search. The general idea of this LS is to invest less time in searching for a better solution by selecting alternative branches with a lower PEX, while at the same time more effort is invested in selecting alternative branches with a higher PEX.

Forward-backward scheduling (LS-2) This multi-pass scheduling method is well-know in the RCPSP literature (Li and Willis, 1992). Since the forward-backward scheduling method is implemented in this research as a LS in an iterative procedure instead of a multi-pass scheduling method, we do not compile the initial activity loading list using the heuristics proposed by Li and Willis (1992). Rather, we prefer to exploit the knowledge that is stored in the iterative procedure by using the priority values in the RK of the current solution for the creation of the initial forward and backward schedules.

6. Computational results

In this section, we test the different parameter settings and strategies of the TS and evaluate the overall performance of the fine-tuned procedure. Section 6.1 presents the novel, problem-specific dataset that is generated in this paper based on the ASM. In section 6.2, we validate the performance of the different building blocks of the TS and we report on the overall performance of the procedure. Finally, the impact of the problem parameters on the solution quality is investigated in section 6.3. All tests were carried out on a computer with an Intel Core i5 processor 2.5 GHz and 8 Gb RAM.

6.1. Benchmark dataset

6.1.1. Generation process

Several datasets have been developed based on different network generators for the RCPSP (Vanhoucke et al., 2016). Since the RCPSP is extended with a selection subproblem in this research, the datasets in these studies are insufficient to test the presented solution approach for the RCPSP-AS. Therefore, we provide an overview of the datasets that are used in the distinct problems with alternatives that support this research (see table 4).

Based on this overview, we can conclude that most datasets related to the RCPSP-AS are small-scale and problem-specific. Moreover, the majority of the datasets are randomly generated without the use of a network generator with the exception of Kellenbrink and Helber (2015). However, Elmaghraby and Herroelen (1980) have indicated the importance of using a dataset with problem instances that span the full range of problem complexity. This could be considered a drawback of the existing datasets for optimisation problems with a selection subproblem. For example, most research papers discussed above neglect the links that might exist between alternatives (see column *flexibility parameters* in table 4). Again, the dataset of Kellenbrink and Helber (2015) is considered the most suitable

Problem	General information					Flexibility parameters		Project parameters			Problem-specific information
	Paper	Dataset	Generator	Subsets	#Inst	Nested	Linked	#Act	#RR	#NR	
Alternative process plan (resource) scheduling problem	Beck and Fox (2000)	-	Notional data	Set 1	480	Y	N	10	10	n.a.	Max. number of alternatives = $\{1, 3, 5, 7\}$ Makespan factor = $\{1.0, 1.1, 1.2, 1.3, 1.4, 1.5\}$
				Set 2	480	Y	N	$\{5, 10, 15, 20\}$	$\{5, 10, 15, 20\}$	n.a.	Fixed number of alternatives = 5 Makespan factor = $\{1.0, 1.1, 1.2, 1.3, 1.4, 1.5\}$
				Set 3	480	Y	N	10	10	n.a.	Set 1 with alternative resources [1,6]
Job-shop scheduling with processing alternatives	Kis (2003)	-	Notional data	n.a.	40	N	N	$\{5, 10\}$	$\{5, 10, 15, 20\}$	n.a.	Number of OR-subgraphs = $\{1, 2, 3\}$
		MPM	Notional data	edata	43	n.a.	n.a.	$\{5, 6, 10, 15\}$	$\{6, 10, 15, 20, 30\}$	n.a.	Avg. $M_{i,j} = 1.15$; Max. $M_{i,j} = \{2, 3\}$
				rdata	43	n.a.	n.a.	$\{5, 6, 10, 15\}$	$\{6, 10, 15, 20, 30\}$	n.a.	Avg. $M_{i,j} = 2$; Max. $M_{i,j} = 3$
				vdata	43	n.a.	n.a.	$\{5, 6, 10, 15\}$	$\{6, 10, 15, 20, 30\}$	n.a.	Avg. $M_{i,j} = \frac{1}{2}m$; Max. $M_{i,j} = \frac{4}{5}m$
Alternative subgraphs assembly line balancing problem	Capacho and Pastor (2006)	-	Notional data	n.a.	82	N	N	[8, 111]	n.a.	n.a.	CT = [20, 4676] Number of subassemblies = $\{2, 3, 4\}$
	Capacho et al. (2009)	-	Notional data	n.a.	150	N	N	[37, 305]	n.a.	n.a.	CT = [28, 11570] Number of subassemblies = $\{2, 3, 4\}$
	Scholl et al. (2009)	-	Notional data	Set 1	269	Y	N	n.a.	n.a.	n.a.	Portion of variable sub-processes = 2
				Set 2	269	Y	N	n.a.	n.a.	n.a.	Portion of variable sub-processes = 5
				Set 3	269	Y	N	n.a.	n.a.	n.a.	Portion of variable sub-processes = 7
				Set 4	269	Y	N	n.a.	n.a.	n.a.	Portion of variable sub-processes = 10
Production scheduling with alternative process plans	Capek et al. (2012)	-	Notional data	n.a.	1040	Y	N	[10, 2000]	[1, 5]	n.a.	Processing time = $[1, p_{max}]$ with $p_{max} = 15$ or 100 Number of time-lags = $\lvert E(G^{prec})\rvert + \frac{n}{3}$ for positive values Number of time-lags = $\frac{n}{5}$ for negative values
		Boctor (1993)	-	boct50	120	n.a.	n.a.	50	[1, 4]	n.a.	We refer to Boctor (1993)
				boct100	240	n.a.	n.a.	100	[1, 4]	n.a.	We refer to Boctor (1993)
		Kis (2003)	Notional data	n.a.	40	N	N	$\{5, 10\}$	$\{5, 10, 15, 20\}$	n.a.	See problem-specific information of Kis (2003)
Resource-constrained project scheduling problem with model-endogenous decisions on the project structure	Kellenbrink and Helber (2015)	-	ProGen	Set 1	1536	Y	Y	30	2	2	Nummber of choices = $\{2, 4\}$ and number of act. that causes further act. = $\{1, 2\}$ Number of optional act. per choice = $\{2, 4\}$ and number of dependent activities caused by optional activity = $\{1, 3\}$ $RF_R = \{0.5, 1\}$; $RS_R = \{0.25, 0.5, 0.75, 1\}$; $RF_N = \{0.5, 1\}$; $RS_N = \{0.75, 1\}$; NC = $\{1.5, 1.8, 2.1\}$;
				Set 2	1536	Y	Y	60	2	2	Number of choices = $\{3, 6\}$ and number of act. that causes further act. = $\{2, 4\}$ Number of optional act. per choice = $\{2, 4\}$ and number of dependent activities caused by optional activity = $\{1, 3\}$ $RF_R = \{0.5, 1\}$; $RS_R = \{0.25, 0.5, 0.75, 1\}$; $RF_N = \{0.5, 1\}$; $RS_N = \{0.75, 1\}$; NC = $\{1.5, 1.8, 2.1\}$;
				Set 3	1536	Y	Y	90	2	2	Number of choices = $\{4, 8\}$ and number of act. that causes further act. = $\{3, 6\}$ Number of optional act. per choice = $\{2, 4\}$ and number of dependent activities caused by optional activity = $\{1, 3\}$ $RF_R = \{0.5, 1\}$; $RS_R = \{0.25, 0.5, 0.75, 1\}$; $RF_N = \{0.5, 1\}$;$RS_N = \{0.75, 1\}$; NC = $\{1.5, 1.8, 2.1\}$;
				Set 4	1536	Y	Y	120	2	2	Number of choices = $\{5, 10\}$ and # of act. that causes further act. = $\{4, 8\}$ Number of optional act. per choice = $\{2, 4\}$ and number of dependent activities caused by optional activity = $\{1, 3\}$ $RF_R = \{0.5, 1\}$; $RS_R = \{0.25, 0.5, 0.75, 1\}$; $RF_N = \{0.5, 1\}$; $RS_N = \{0.75, 1\}$; NC = $\{1.5, 1.8, 2.1\}$;

legend	
$M_{i,j}$	Number of machines j available for processing operations i
m	Number of different machines
n	Number of activities
CT	Cycle time
p_{max}	max. value of the activity processing time
$\lvert E(G^{prec})\rvert$	Number of edges in graph G^{prec} defined in Capek et al. (2012)
RF_R	Resource factor of the renewable resources
RS_R	Resource strength of the renewable resources
RF_N	Resource factor of the non-renewable resources
RS_N	Resource strength of the non-renewable resources
NC	Network complexity

Table 4: Literature on benchmark datasets for scheduling problems related to the RCPSP-AS

dataset for the RCPSP-AS. However, a straightforward application in our research is rendered impossible due to some limitations. First of all, the flexibility parameters proposed in section 3.3 are not used in the controlled design of this dataset. Furthermore, the complexity of the selection subproblem is kept rather low as the number of choices as well as the number of activities per choice is limited in all problem instances to, respectively, $\{2, 10\}$ and $\{2, 4\}$ (see column *problem-specific information* in table 4). Based on this analysis, we observe the need for a new benchmark dataset for the RCPSP-AS. Therefore, we develop a large-scale dataset that is made available for the evaluation of algorithms and the facilitation of future research. Moreover, the data generation process is executed in a controlled way and supported by the RanGen 2 network generator (Vanhoucke et al., 2008). We can distinguish between problem parameters that are related to the alternative subgraphs (i.e. *flexibility parameters*) and problem parameters that capture information about the network topology and resource profile of the project (i.e. *project parameters*). Note that the flexibility parameters determine the complexity of the selection subproblem regarding the selection of alternatives. Given a rigid set of alternatives, the project parameters will impact the complexity of the scheduling subproblem. In the remainder of this section, we present several project and flexibility parameters and we briefly illustrate our procedure to construct the dataset using a controlled design based on these parameters.

To the best of our knowledge, none of the existing network generators can deal with the selection subproblem of the RCPSP-AS. Neither, we are aware of any research on the relationship between the characteristics of alternative subgraphs and the complexity of problem instances of the RCPSP-AS. Therefore, we introduce four flexibility parameters: the degree of flexibility (%flex), the degree of nested alternatives (%nested), the degree of linked alternatives (%linked) and the flexibility distribution. The %flex parameter gives an indication of the number of alternatives in the project network. While the %flex indicator determines the size of the alternative subgraphs in the RCPSP-AS, the %nested and %linked indicators imply its complexity. The %nested parameter gives an indication of the relative number of alternatives that are embedded in another alternative, while the %linked parameter marks the relative number of links between alternatives in a problem instance. Moreover, the flexibility distribution indicates where these alternatives are allocated in the project network.

The data generation procedure to construct our dataset consists of four major steps that are illustrated in table 5.

In the **first step**, the procedure is initiated with a serial project structure that consists

of a pre-specified number of nodes, which reflect the different stages in the project. The overall maximum number of alternatives in the project is defined by two metrics: the *number of stages* and the *maximum number of alternatives per stage*. Both parameters are assumed equal to 5, which results in a maximum number of alternatives equal to 25.

In the **second step**, we decide on the *actual number of alternatives* in the project and the *flexibility distribution* of these alternatives over the different stages in the project. We include the alternatives as parallel nodes at each stage in the project structure. The actual number of alternatives in the project can be calculated as the %flex $\in [0, 1]$ multiplied by the maximum number of alternatives in the project imposed in the first step of the data generation. Once the number of alternatives has been defined, the proposed dataset covers four flexibility distributions $\in \{1, 2, 3, 4\}$ in order to assign these alternatives to the different stages in the project:

Early focus (1) We start with the allocation of alternatives to the first stage in the project and repeat this process for the following stages until all alternatives have been allocated.
Middle focus (2) We start with the allocation of alternatives to the middle stage in the project and consecutively assign the remaining alternatives in an alternating manner to the next and the previous stage in the project.
Late focus (3) The alternatives are distributed in a similar way than in the early focus scenario, however, in descending order of stage number.
Uniform distribution (4) The alternatives are added one alternative at a time starting from the first up to the last stage in the project, after which the procedure is repeated.

In the **third step** of the data generation procedure, the project structure obtained after the second step is transformed into a project structure with nested alternative subgraphs and linked alternative branches. Both extensions can be integrated in the dataset through the addition of arcs between the nodes in the project structure. The *number of nested alternatives* can be computed as the %nested $\in [0, 1]$ multiplied by the actual number of alternatives that can potentially be nested. In order to include the nested alternatives, we add arcs between a single node in one stage and a set of nodes in the succeeding stage. The *number of linked alternatives* can be computed as the %linked $\in [0, 1]$ multiplied by the actual number of alternatives that can potentially be linked. In order to include the linked alternatives, we connect two nodes in the alternative subgraphs without the unwanted creation of more nested alternatives. With respect to the number of potentially linked

nodes, each node can only be linked to a node with a higher rank according to a ranked list from 1 to the total number of nodes (e.g. [1,19] in the example of table 5). This ensures a unique assignment of linked alternatives as we want to avoid that, for example, node 18 is linked to node 19 and visa versa. Consequently, the number of potentially linked nodes is equal to 18 in this example since the last non-dummy node (i.e. node 19) cannot be linked to a node with a higher rank. Since there might still exist some floating nodes in the project structure after the inclusion of the required arcs, the unconnected nodes are connected with additional arcs between the stages. This is represented in table 5 by □.

In the **fourth step**, the generation procedure will replace each node in the project structure, which corresponds to an execution mode of a work package, by a subproject using the network generator RanGen 2 (Vanhoucke et al., 2008). We refer to existing research (Herroelen and De Reyck, 1999) that focuses on the topological structure and the resource allocation to determine the scheduling complexity. First of all, the topological indicators provide information on the structure and size of the project network (Vanhoucke et al., 2008; Demeulemeester et al., 2003). In our dataset, we measure the topological complexity of the subprojects by means of the *serial/parallel indicator* (SP) (Tavares, 1999). Secondly, several measures have been introduced to link the resource characteristics of a project with the scheduling complexity (Demeulemeester et al., 2003). For the proposed dataset, we compute the resource requirements of the activities by means of the *resource use* (RU) (Demeulemeester et al., 2003) and generate the resource tightness using the *resource-constrainedness* (RC) (Patterson, 1976). The project parameters define the project network that is included in each alternative execution mode of a work package and thus impact the complexity of the scheduling subproblem of the RCPSP-AS.

6.1.2. Activity networks of step 4

In step 4, the work packages in the project structure are implemented by means of a set of interrelated activities in a subproject. As a result, we have to create activity networks for the subprojects that correspond with the alternatives in the project structure. In this section, we will briefly discuss the number of activities and resources in the activity networks. The number of activities in each alternative (i.e. each node) in each stage is not equal for all alternatives, since the data generation procedure creates shorter (e.g. alternative path 5-7-14-20 in table 5) and longer (e.g. alternative path 5-6-11-16-20 in table 5) alternative paths in the project structure. Consequently, some alternatives would always be preferred over others in case that the scheduling complexity of all alternatives

would be assumed equal. Instead, an approach to generate alternative paths that are more or less equal to each other is proposed. More precisely, the number of activities in each node is directly proportional to the number of stages each node spans. For nodes that are connected to the next stage, the number of activities is set low (e.g. 10 activities in nodes 7, 10 and 11 in table 5), while nodes that are connected with further stages in the project are set proportionally higher (e.g. 20 activities in nodes 12,13,14 and 15 in table 5). In doing so, the choices between alternative paths result in similar projects with respect to the overall number of activities that need to be scheduled. In order to analyse the impact of resource characteristics on the selection process, we have used a similar approach to create alternatives with different resource profiles that are similar in scheduling complexity. More precisely, the number of resources in each of the subprojects is directly proportional to the number of stages each node spans, while the RU equals to the number of resources. For nodes that are connected to the next stage, the number of resources is set low (e.g. one resource in nodes 7, 10 and 11 in table 5), while nodes that are connected with further stages in the project are set proportionally higher (e.g. two resources in nodes 12,13,14 and 15 in table 5). In order to allow a fair comparison between the alternatives, the work content divided by the resource availability is assumed equal in all subprojects.

6.1.3. Details of linked alternative branches

The project structure obtained after the first three steps of the data generation process consists of nodes that reflect alternative execution modes of work packages, while the fourth step focusses on the activities as the work packages are replaced by activity networks. This distinction between the work package level (steps 1-2-3) and the activity level (step 4) is important to accurately model the linked alternative branches. For example, each node in table 5 corresponds to an execution mode of a work package and not a single activity. Consequently, the links between nodes in table 5 represent links between alternatives rather than links between activities in the activity network. The links between activities can only be implemented when the nodes are replaced by subprojects in step 4. Given that we have generated a subproject in each node of the project structure, we still need to establish the links between the individual activities. In this research, we model a link by selecting an activity in each of the two subprojects making sure that the selected activity in the subproject that is connected by the incoming arc is not a branching activity. The restriction that prohibits activities to be linked to a branching activity is stipulated in equation 4 in section 3.3. For example, there exists a link between nodes 2 and 7 in table

5. Assume that both nodes 2 and 7 are replaced by a small project network that consists of four activities. In figure 6, we graphically illustrate the different options to implement the link between an activity in node 2 and 7. In the above situation, we observe an infeasible link between activity 2 in node 2 and activity 5, which is the branching activity in node 7. In the below situation, a feasible link is implemented between activities 2 and 6 in, respectively, nodes 2 and 7.

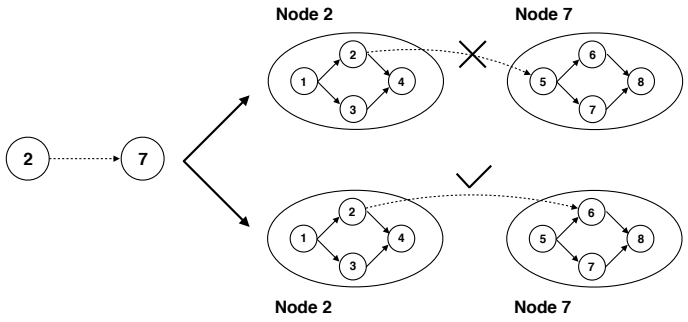


Figure 6: An illustration of links between work packages and links between activities

6.1.4. Benchmark dataset

The result of the stepwise data generation is a complete project network with alternative subgraphs that can be classified based on a set of flexibility and project parameters. The test instances in our dataset have been generated given the problem settings displayed in table 6. Using 10 instances for each setting, we obtain a dataset of 10*4*4*5*5*3*3 = 36,000 instances. Table 7 provides an overview of the exact number of activities in each problem instance that corresponds with a given problem setting. The test instances in this dataset can be downloaded from the website www.projectmanagement.ugent.be.

6.2. Performance of tabu search procedure

6.2.1. Initial solution

The four different initial solution procedures that are described in section 5.1 are investigated to find the best approach. Table 8 summarises the average project makespan (Avg. makespan) over all problem instances in the dataset obtained using each of these strategies as well as the percent deviation from the best strategy (Dev. best). We also compare the performance of the strategies with a proposed lower bound (Dev. CPLB-A). This lower bound is derived from the well-known critical path lower bound (CPLB) for the RCPSP.

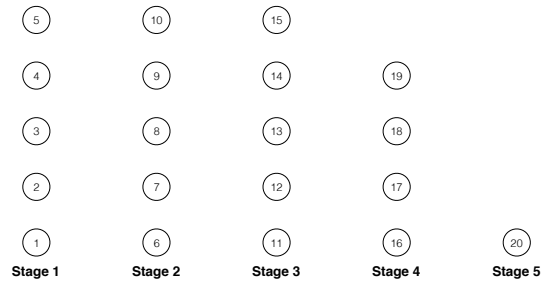
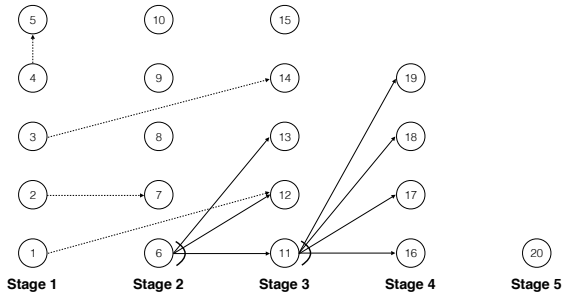
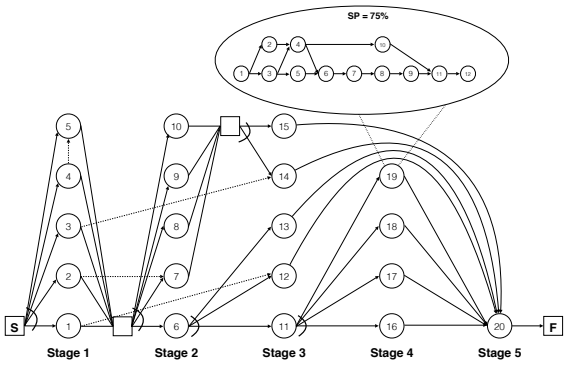
Generation procedure		Illustrative project network	Parameter setting
	Step 1: Initialisation		# stages = 5 Max # of alternatives $per stage = 5$ Max # of alternatives = (5*5) = 25
Selection subproblem	Step 2: Alternative work packages		%flex = 75% Flexibility distribution = 1 $75\% * (25 - 5) = 15$ 1. Add 4 nodes in the first, second and third stage (max. # nodes per stage = 5) 2. Add 3 nodes in stage 4 up to a total of 15 additional nodes
	Step 3: Alternative project structure		%nested = 50% %linked = 25% # of potential nested nodes = 14 (i.e. nodes 6 to 19) $50\% * 14 = 7$ 1. Link node 11 with nodes 16, 17, 18 & 19 2. Link node 6 with nodes 11, 12 & 13 # of potential linked nodes = 18 (i.e. nodes 1 to 18) 25%*18=4 1. Link node 1 with node 12 2. Link node 2 with node 7 3. Link node 3 with node 14 4. Link node 4 with node 5
Scheduling subproblem	Step 4: Deterministic subproject		SP = 75% RC = 50% Number of activities/resources: Node 1, 2, 3, 4 and 5: 10/1 Node 6, 7, 8, 9 and 10: 10/1 Node 11: 10/1 Node 12, 13, 14 and 15: 20/2 Node 16, 17, 18 and 19: 10/1 Node 20: 10/1

Table 5: An illustration of the data generation procedure to construct a project instance

		Problem parameters	
		Flexibility parameters	**Project parameters**
	Number of instances	10	
	Number of activities	10*number of connected stages	
	Number of resources	number of connected stages	
Step 1	Number of stages	5	
	Max. number of alternatives per stage	5	
Step 2	Degree of flexibility	0.25, 0.50, 0.75, 1	
	Flexibility distribution	1, 2, 3, 4	
Step 3	Degree of nested alternative subgraphs	0, 0.25, 0.50, 0.75, 1	
	Degree of linked alternative branches	0, 0.25, 0.50, 0.75, 1	
Step 4	SP		0.25, 0.50, 0.75
	RC		0.25, 0.50, 0.75

Table 6: Parameter settings used to generate the test instances for the RCPSP-AS

However, the CPLB calculations cannot be applied in a straightforward manner in the RCPSP-AS since it consists of both fixed and alternative activities. Therefore, we propose the following adjustments to the CPLB calculations: (1) neglect the links between alternative branches and (2) select the shortest alternative branch in each alternative subgraph of the project. We refer to this lower bound as the CPLB with alternatives (CPLB-A). Recall that eight different variants for IN-1 exist depending on which adjusted weight is used for the selection of alternative branches. It can be seen that IN-1.8 yields on average the lowest project makespan. The relative difference in solution quality with a random initial solution (IN-2) mounts to 15.89% on average. Moreover, we also observe that even the variants of IN-1 with the lowest performance result on average in a significant lower project makespan than IN-2. The approach that uses information from the optimal solution of the relaxation of the problem (IN-3) results in a good overall performance and the controlled RK approach (IN-4) results on average in better initial solutions than IN-2, but worse than IN-1 and IN-3. This can be explained by the fact that IN-4 applies a priority rule that accounts for the presence of alternatives but neglects the specific characteristics of the alternative subgraphs. For the sake of completeness, we have subdivided the results in table 8 based on the position of the problem instance in the ASM. We should note that problem instances with a %linked or %nested $< 50\%$ ($\geq 50\%$) are classified, respectively, as low (high) linked or nested.

		%flex															
		0.25				0.5				0.75				1			
		Flexibility distribution															
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
%nested	0	122	122	123	126	183	183	184	186	244	246	245	246	306	306	306	306
	0.25	122	130	162	135	201	212	182	205	263	285	243	275	345	345	345	345
	0.50	161	130	162	154	240	222	223	244	342	304	282	334	424	424	424	424
	0.75	161	120	152	183	270	241	211	303	371	483	321	423	543	543	543	543
	1	161	131	132	222	301	281	262	382	481	562	452	542	702	702	702	702

Table 7: Number of activities for different combinations of parameter settings

		IN-1.1	IN-1.2	IN-1.3	IN-1.4	IN-1.5	IN-1.6	IN-1.7	IN-1.8	IN-2	IN-3	IN-4
Avg. makespan		204.96	205.58	211.15	207.23	205.43	202.98	205.48	**201.44**	233.44	213.85	222.42
Dev. best (%)		1.75	2.06	4.82	2.88	1.98	0.77	2.01	-	15.89	6.16	10.42
Dev. CPLB-A (%)		64.81	65.31	69.78	66.63	65.18	63.21	65.22	61.97	87.70	71.95	78.85
Q1	Avg. makespan	186.67	187.18	**185.42**	185.82	187.32	187.56	187.42	187.64	211.76	189.97	191.71
	Dev. best (%)	0.68	0.95	-	0.22	1.03	1.16	1.08	1.20	14.21	2.02	3.40
Q2	Avg. makespan	195.87	200.53	196.16	**195.60**	196.62	195.91	196.50	195.63	221.10	200.16	213.95
	Dev. best (%)	0.14	2.52	0.28	-	0.52	0.16	0.46	0.02	13.03	2.33	9.38
Q3	Avg. makespan	201.96	202.27	207.92	204.87	202.34	201.52	202.23	**200.81**	228.32	207.35	216.12
	Dev. best (%)	0.57	0.73	3.54	2.02	0.76	0.35	0.70	-	13.70	3.26	7.62
Q4	Avg. makespan	221.16	219.34	234.74	226.07	221.41	215.52	221.66	**211.85**	254.72	238.28	245.92
	Dev. best (%)	4.39	3.54	10.81	6.71	4.51	1.73	4.63	-	20.23	12.48	16.08

Table 8: Performance of the different initial solution generation strategies

6.2.2. Multi-start iterated local search

In order to validate the performance of the TS procedure, we have initially solved each problem instance in the dataset using a multi-start iterated LS (MSILS). The MSILS approach will iteratively call a LS routine, each time starting from a different initial solution, to obtain a local optimum for the optimisation problem. In this research, we apply the API-operator (NH-4) to improve a random initial solution (IN-2) until no more improvements can be obtained, after which the procedure is relaunched using another random initial solution (i.e. MSILS-1). The average overall best makespan obtained after 5,000 visited schedules using MSILS-1 is presented in table 9. We observe that the solution quality is only 1.69% better than the average, random initial solution (Dev. IN-2) and even 13.92% worse than the average, best initial solution (Dev. IN-1.8). This shows that a single solution that corresponds to a good set of alternatives outperforms an iterative search without a clever selection of alternatives. Subsequently, we investigate the performance of the MSILS in case that we start each LS routine with a good initial solution (IN-1.X) rather than a random initial solution (i.e. MSILS-2). The specific variant of IN-1 is determined based on the dynamic quadrant selection in order to avoid that the same initial solution is used in each LS routine, which would contradict the iterative nature of the MSILS process. We observe that the average makespan obtained after 5,000 schedules decreases with 10.99% in case that a good rather than a random initial solution is used (Dev. MSILS-1). In order to further test the potential of MSILS-2, we include the iterative forward-backward LS (LS-2) after the exploration of the NH in each LS routine (i.e. MSILS-3). Table 9 shows that MSILS-3 further improves the performance of MSILS-2.

6.2.3. Tabu search procedure

In order to determine the best parameter settings for the TS, a number of experiments have been carried out. Initially, we should determine the length of the tabu list and the

	MSILS-1	MSILS-2	MSILS-3
Avg. makespan	229.48	204.27	200.56
Dev. IN-2 (%)	-1.69	-12.50	-14.08
Dev. IN-1.8 (%)	13.92	1.41	-0.43
Dev. MSILS-1 (%)	-	-10.99	-12.60

Table 9: Relative performance of the different scenarios for the MSILS approach

number of iterations of the scheduling run within an iteration of the selection run. Furthermore, we test the added value of the different NH and LS procedures. The experiments to fine-tune the performance of the proposed TS are performed on a training set of 100 test instances. The results of these experiments are summarised in tables 10-12.

Parameter settings. Since the proposed TS procedure consists of a tabu list for the scheduling (TL-SCH) and selection (TL-SEL) subproblem, we run two independent experiments in order to determine the length of these tabu lists. In these experiments, the size of the tabu list is expressed in terms of the number of activities and the number of alternative branches in the problem instances, respectively, for TL-SCH and TL-SEL. This is important as the number of activities and alternatives can vary widely between the instances and therefore an absolute size of the tabu lists is undesired. We observe from table 10 that a list size equal to 0.50 times the number of activities for the TL-SCH and 0.25 times the total number of alternative branches for the TL-SEL result in the best outcome. We also note that it is optimal to iterate two scheduling runs within one iteration of the selection run.

Neighbourhood structure. The three NH structures that focus on the selection subproblem (NH-1, NH-2 and NH-3) are tested on the same training set to find the best approach. We have investigated the performance of NH-1 for each of the four duration- and resource-based weights (NH-1.1 up to NH-1.8) as well as for the dynamic quadrant selection (NH-1.X). Since NH-2 also depends on the position of the problem instance in the ASM, we also test the performance of this NH in case that each of the four quadrants are assumed (NH-2.1 up to NH-2.4) and the quadrants are dynamically selected (NH-2.X). According to the %nested and %linked, an instance of the RCPSP-AS can be uniquely classified in a single quadrant of the ASM. In this research, we also test the performance of NH-1 and NH-2 in case that the operators are selected based on a misclassification of the project instances (e.g. NH-1.1 is used in case that the instance should be positioned in Q2). We refer to section 4.2 for a detailed explanation of the calculation of the weights. The average project makespan and run time of each NH structure is shown in table 11. We observe that NH-1 and NH-2 outperform NH-3, independent of the configuration and that NH-1.X is deemed the overall best NH. Furthermore, we notice that the dynamic selection of quadrants outperforms a more rigid position of the problem instances in the ASM for both NH-1 and NH-2. In summary, we will apply NH-1.X as NH structure in the final version of the TS. In order to analyse the relation between the position of the problem

instances in the ASM and the performance of the different NH structures, we subdivide the results in table 11 based on the position of the problem instance in the ASM. The two following contributions can be drawn from this additional analysis. First of all, NH-1.5 up to NH-1.8 is the technique with on average the best solution quality in case that the variant of the NH is used that corresponds to the actual position of the problem instance in the ASM (i.e. NH-1.5 is used for Q1, NH-1.6 for Q2, etc.). The characteristics of the alternative subgraphs are not as clearly captured in the other NH structures. Secondly, the dynamic quadrant selection procedure pays off in all four quadrants, although its potential is highest in the most complex quadrant Q4. The NH-1.X structure has two important properties. First of all, it facilitates an increased exploration in the TS as different quadrants are selected throughout the procedure for the calculation of weights. Secondly, the dynamic quadrant selection is guided by the position of the problem instance in the ASM based on %nested and %linked, which increases the randomness in the TS procedure. In order to illustrate that the improved performance of NH-1.X is not merely due to the increased randomness, we execute an additional experiment in which we perform a dynamic quadrant selection that selects the quadrant uniformly at random (NH-1.U). According to NH-1.U, each quadrant in the ASM is selected with a probability equal to 25% in the dynamic quadrant selection. This allows us to show the relevance of the dynamic quadrant selection. The experiments show that NH-1.U results on average in a similar project makespan as the NH-1.X. The superior performance of NH-U and NH-1.X clearly illustrates that the dynamic quadrant selection works better than the NH-1.1 to NH-1.8 methods, which focus only at a single quadrant for the selection of a NH. In the remainder of the experiments, we have used NH-1.X instead of NH-U since the NH-U results in a slightly higher project makespan (0,16%) than NH-1.X.

Local searches. We also test different configurations of the TS in order to investigate the added value of both LS procedures. The independent inclusion of LS-1 and LS-2 in the TS yields a decrease of the total makespan of, respectively, 0.90% and 1.19%. With respect to LS-1, we also check the performance of the TS with the inclusion of a LS-1 after each scheduling run (LS-1.2) rather than after two iterations of the scheduling run (LS-1.1). The experiments show a slight improvement of the solution quality compared to LS-1.1. The impact of the LS procedures is summarised in table 12. As expected, the inclusion of both LS procedures increases the run time of the TS. However, we note that the increase is more significant for both LS-1.1 and LS-1.2 than for LS-2 although the relative improvement of

LS-2 is higher. Based on the above analysis, we decide to include LS-1.2 and LS-2 in the fine-tuned TS discussed below.

Overall performance of TS. Given the best combination of parameter settings as well as the preferred NH and LS procedures, the TS was run with a stopping criterion of 5,000 schedules for all instances in the dataset. The results are summarised in table 13. The average project makespan that is obtained through the TS over all problem instances equals 182.49, while the project makespan obtained through the MSILS-3 procedure is on average 9.90% (Dev. TS) higher. This implies that the memory structures of the TS pay off for the RCPSP-AS as a much better solution can be obtained within a limited number of generated schedules compared to the MSILS search strategy. Also, we compare the quality of the solution obtained through the TS and MSILS-3 with a single (initial) schedule in order to show the improvement potential of both approaches. One can observe that a single schedule with a focus on the selection subproblem (IN-1.8) results in a project makespan that is only slightly higher than the MSILS-3 solution. This shows the importance of a good initial solution that considers the complexity of the selection subproblem, rather than a random initial solution that is iteratively improved over multiple scheduling steps. The project makespan obtained through the TS improves on average the solution quality of the best initial solution IN-1.8 with 10.38%. This relative deviation even increases to 27.92% in the case that a random initial solution (IN-2) is considered. On average, the project makespan deviates 31.85% from the CPLB-A. Furthermore, we have conducted an additional analysis to investigate the improvement potential of the TS. Therefore, we have investigated the number of schedules that should be visited in the TS before the best solution after 5,000 schedules in the MSILS-3 approach was obtained. The results are shown in figure 7. We

Tabu size	0.01	0.05	0.1	0.25	0.5	0.75
Average makespan TL-SCH	188.92	188.92	188.92	**185.74**	**185.74**	185.94
Dev. best size TL-SCH (%)	1.71	1.71	1.71	-	-	0.11
Average makespan TL-SEL	183.77	183.70	183.80	**183.19**	183.40	183.31
Dev. best size TL-SEL (%)	0.32	0.28	0.33	-	0.11	0.07
# iterations	1	2	5	10	25	50
Avg. makespan	182.02	**182**	182.13	182.37	185.26	189.75
Dev. best (%)	0.01	-	0.07	0.20	1.79	4.26

Table 10: Parameter settings

Neighbourhood		NH-1.1	NH-1.2	NH-1.3	NH-1.4	NH1-5	NH-1.6	NH-1.7	NH-1.8	NH-1.X	NH-2.1	NH-2.2	NH-2.3	NH-2.4	NH-2.X	NH-3
Avg. makespan		183.93	183.69	188.24	191.14	184.46	183.89	184.46	183.77	**182.93**	183.18	187.84	183.15	187.44	183.55	185.46
Dev. best (%)		0.55	0.42	2.90	4.49	0.84	0.52	0.84	0.46	-	0.14	2.68	0.12	2.46	0.34	1.38
Run time (s)		0.89	0.86	0.86	0.82	0.89	0.85	0.89	0.84	0.85	0.89	0.86	0.88	0.85	0.86	0.87
Q1	Avg. makespan	172.31	172.03	172.63	172.59	172.34	172.31	172.34	172.31	171.63	172.03	174.50	172.25	174.00	**170.69**	
	Dev. best (%)	0.95	0.79	1.14	1.12	0.97	0.95	0.97	0.95	0.55	0.79	2.23	0.92	1.94	-	
Q2	Avg. makespan	184.94	185.10	186.69	186.21	185.58	185.63	185.58	185.63	**183.54**	183.96	193.81	183.60	193.69	187.54	
	Dev. best (%)	0.76	0.85	1.71	1.45	1.11	1.14	1.11	1.14	-	0.23	5.60	0.03	5.53	2.18	
Q3	Avg. makespan	185.67	186.35	189.23	190.60	185.85	186.56	185.85	185.46	185.63	185.60	186.98	184.83	185.92	**184.21**	
	Dev. best (%)	0.79	1.16	2.73	3.47	0.89	1.28	0.89	1.22	0.77	0.76	1.50	0.34	0.93	-	
Q4	Avg. makespan	187.26	186.15	195.56	203.03	188.17	186.08	188.17	185.82	**185.75**	186.00	190.36	186.56	190.25	186.17	
	Dev. best (%)	0.82	0.22	5.28	9.30	1.30	0.18	1.30	0.04	-	0.13	2.48	0.43	2.42	0.22	

Table 11: Performance of the different neighbourhood structures

Local searches	no LS-1	LS-1.1	LS-1.2	no LS-2	LS-2
Avg. makespan	214.30	212.37	212.23	183.19	181.01
Dev. basic scenario (%)	-	-0.90	-0.97	-	-1.19
Run times (s)	0.89	0.93	1.01	0.85	0.85

Table 12: Performance of the different local searches

observe that for a large number of instances (more than 19,000 instances) a lower project makespan can be obtained after less than 100 visited schedules in the TS compared to the best solution obtained through the MSILS-3 after 5,000 schedules. Moreover, for 95% of the instances, for which the TS results in a lower project makespan compared to the MSILS-3 approach, a better solution is obtained already within 2,000 visited schedules. Figure 8 shows the average makespan obtained in the TS for different numbers of visited schedules as a stopping criterion. This clearly shows that the lowest project makespan found in the TS decreases drastically during the first visited schedules, where further improvements are harder to come across in the later visited schedules as the solution quality converges to the best found solution. This behaviour is in line with the expectations as improvements are typically more radical in the beginning of a metaheuristic procedure, but more incremental at the end of the procedure.

6.3. Impact of problem parameters

6.3.1. Impact of flexibility parameters

In this section, we examine the impact of the four parameters that uniquely define the alternative subgraphs (see section 6.1) on the solution quality. Table 14 summarises the results. We observe that an increased number of alternatives in the project structure result in a significant improvement of the solution quality. The project makespan deviates

	Procedure		Initial solution		LB
	TS	MSILS-3	IN-2	IN-1.8	CPLB-A
Avg. makespan	182.49	200.56	233.44	201.44	124.37
Dev. TS (%)	-	9.90	27.92	10.38	-31.85
#schedules	5,000	5,000	1	1	-
run time (s)	0.88	0.88	-	-	-

Table 13: Performance of TS

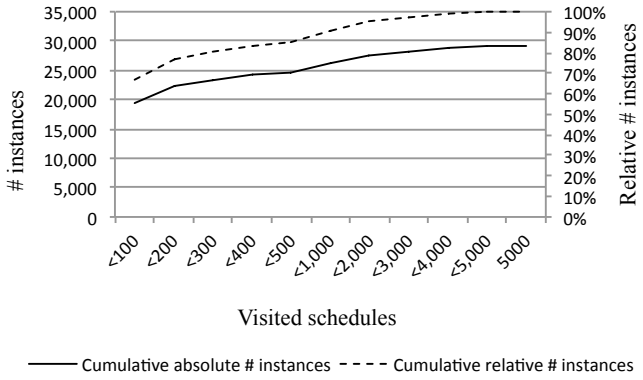


Figure 7: Improvement potential of the TS compared to the best found solution in the MSILS-3

with 7.97% when we compare the scenarios with the lowest and the highest %flex. This observation can be explained by the fact that no marginal costs of additional alternatives are explicitly included in this research. As a result, more flexibility in the project structure provides more opportunities to create a schedule with a better overall project makespan. However, this increased degree of freedom comes at the cost of an increased computational time. The results also show that an increased number of linked alternative branches and nested alternative subgraphs has on average a negative impact on the solution quality. The average project makespan increases, respectively, with up to 4.70 % and 10.43% in case that we move from a low to a high %linked and %nested. These results could be expected since the complexity of the problem increases as more dependencies exist between alternatives, which is also supported by the increased run times for higher levels of %linked and %nested. Note that the impact of a high %nested on the computational complexity is more distinct than a high %linked. We observe on average a slightly lower project makespan in case that the alternatives are equally distributed over the project horizon (i.e. strategy 4 compared

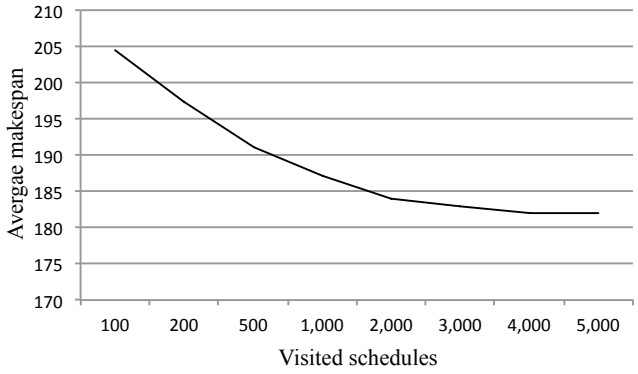


Figure 8: Average makespan after different numbers of visited schedules in the TS

to strategies 1, 2 and 3). This implies that a lower project makespan can be expected when the alternatives are distributed rather than concentrated in the project.

6.3.2. Impact of project parameters

We briefly describe the impact of the project parameters SP and RC on the solution quality. The results are provided in table 15. As expected, the solution quality deteriorates as the resource constrainedness increases and the network topology of the subprojects in the project structure becomes more sequential. At the same time, the run time of the TS decreases with an increased SP and RC level due to a decreased degree of freedom during scheduling.

	%flex				**%linked**				
	0.25	0.5	0.75	1	0	0.25	0.5	0.75	1
Avg. makespan	190.49	184.05	178.98	**176.43**	**179**	180.61	181.97	183.44	187.41
Dev. best (%)	7.97	4.32	1.45	-	-	0.90	1.66	2.48	4.70
Run time (s)	0.41	0.68	1.05	1.41	0.80	0.86	0.90	0.93	0.93
	flexibility distribution				**%nested**				
	1	2	3	4	0	0.25	0.5	0.75	1
Avg. makespan	182.68	183.13	182.58	**181.56**	**175.03**	176.99	181.16	185.97	193.29
Dev. best (%)	0.62	0.87	0.56	-	-	1.12	3.51	6.25	10.43
Run time (s)	0.86	0.89	0.83	0.96	0.61	0.70	0.83	1.02	1.26

Table 14: Impact of the flexibility parameters on the soluton quality

		RC		
		0.25	0.50	0.75
SP	0.25	89.37 (1.00)	158.88 (0.95)	228.87 (0.92)
	0.50	126.85 (0.89)	178.53 (0.88)	235.20 (0.86)
	0.75	176.96 (0.82)	204.56 (0.82)	243.16 (0.81)

Table 15: Impact of the project parameters on the solution quality (run times (s))

6.3.3. Impact of resource characteristics of alternatives

As already stated in section 6.1, the problem instances in our dataset are constructed in such a way that the number of resources required to complete a work package is directly proportional to the number of stages that each work package spans. Recall that a non-nested alternatives that spans 3 stages consists of 30 activities and requires 3 resource types, while three consecutive nested alternatives each consist of 10 activities and require 1 resource type. In this way, we can distinguish between complete, global alternative execution modes of a work package that call for a higher number of general resources and partial, specialised alternative execution modes of a work package that call for a lower number of dedicated resources. The former type of alternatives allow for a global optimisation of the project schedule, but correspond to a more complex scheduling problem. The latter type of alternatives consist of multiple, less complex local optimisation problems. In this section, we analyse the impact of the different resource profiles of the alternatives on the outcome of the selection process. Since the trade-off discussed above is most apparent in problem instances with a high %flex and %nested, we focus our analysis on the 360 problem instances in the dataset that are characterised by a %flex=1, %nested=1 and %linked=0. Note that the conclusions for other problem settings can be derived from the presented analysis as they propose special cases of the tested instances. We run an experiment to gather the average makespan of the different alternative paths that span, respectively, 1, 2, 3, 4 and 5 stages for each of the 360 instances. This information reflects the expected solution quality for a given trade-off between general and dedicated resources. In our experiments, we could partially confirm our conjectured statements explained in this section. We saw that in case of low RC values, the lowest makespan was obtained with two connected stages. This is not equal to the lowest value (i.e. one connected stage), but nevertheless relatively low. For a high RC value, this number of connected stages is equal to five (i.e. the highest possible value in our data). Obviously, this effect would be more

profound if we should generate our data with more than five connected stages.

6.4. Impact of modified dataset

The data generation procedure described in section 6.1 consists of two layers. In the high-level layer, the project structure with alternative subgraphs is constructed, while each node is replaced by a project network in the low-level layer. In order to generate a dataset that covers all possible structures and complexities, we generate an alternative dataset in which the two layers are interchanged. This implies that the high-level layer considers the creation of a project network, while each node in the project network is then replaced by an alternative subgraph in the low-level layer. In this case, the steps discussed above remain valid, however, we first execute step 4 followed by steps 1 to 3. In the modified dataset, each activity consists of subactivities that can be executed in different alternative ways, represented by the alternative subgraphs. The characteristics of these low-level subactivities are inherited from the higher-level activities. In order to construct the modified dataset, we generate 400 distinct alternative subgraphs based on the flexibility parameters %flex, %nested, %linked and the flexibility distribution. Also, we generate 90 (i.e. 10 instances for each setting) project networks with 10 activities based on the project parameters SP and RC. Consequently, we obtain a modified dataset with 36,000 instances, which combined with the initial dataset results in an overall problem-specific dataset with 72,000 instances. The test instances in the modified dataset can also be downloaded from the website www.projectmanagement.ugent.be. In the remainder of this section, we will analyse the performance of the TS procedure and evaluate the impact of the problem parameters for the modified instances. However, the instances in both datasets differ significantly with respect to their overall project structure. Therefore, we have to create a new training set from the modified dataset in order to fine-tune the TS procedure. Overall, the MSILS-3 results on average in a project makespan equal to 227.50, while an average makespan equal to 193.82 (-14.81%) is obtained through the use of the fine-tuned TS procedure. The additional experiments show that most conclusions obtained in section 6.3 can be generalised to the modified dataset. However, we will briefly discuss the three main differences:

1. The effect of the flexibility parameters %nested and %linked on the average project makespan is larger in the modified dataset as the alternative subgraphs are more deeply embedded. Consequently, the relative impact of the selection subproblem is

larger. We observe that an increase in %nested and %linked from 0% to 100% results in an increase in project makespan, respectively, equal to 16% and 54.86%.

2. When no nested and linked alternative subgraphs are included in the network (i.e. %nested and %linked = 0), a higher number of alternative branches (i.e. high %flex) leads to a lower makespan. When these alternative branches are nested and/or linked (i.e. %nested and %linked > 0), however, adding more alternative branches does not always reduce the project makespan. The lowest makespan is then obtained at %flex values around 50%.

3. The preference for an equal flexibility distribution (i.e. strategy 4) is not supported in the modified dataset. The overall best flexibility distribution is strategy 1, followed by strategies 3 (+3.21%), 4 (+4.81%) and 2 (+5.90%). This discrepancy can be explained by the way the modified dataset is constructed. Since the low-level alternative subgraphs are embedded in the nodes of the high-level project network, the alternatives are, to a certain extent, equally distributed over the project horizon, even in strategies 1, 2 and 3.

7. Conclusions

This paper considers the problem of scheduling a project with fixed and alternative activities. Each alternative activity belongs to one or multiple alternative branches and subsets of these alternative branches are combined into alternative subgraphs. The objective of the problem is to select (at most) exactly one (nested) non-nested alternative branch for each alternative subgraph (i.e. selection subproblem) and assign start times to all fixed and selected alternative activities (i.e. scheduling subproblem), such that a logical, precedence and resource feasible project schedule is obtained with a minimal project makespan. We have proposed to uniquely classify problem instances with alternative subgraphs based on two dimensions in the ASM: the absence/presence of nested alternative subgraphs and/or linked alternative branches. In this paper, we have also generated a dataset in which problem instances with alternative subgraphs are constructed using a controlled design. As the RCPSP-AS involves two subproblems, we present a TS that focuses on the scheduling and the selection subproblem. The results show that the proposed TS outperforms the MSILS on the problem instances in the dataset. The computational results further demonstrate the positive impact of the %flex as well as the negative impact

of the %linked and %nested on the project makespan. For a given number of alternatives, a slightly better solution quality can be on average obtained when the alternatives are uniformly allocated to different project stages rather than concentrated in a limited number of project stages. Furthermore, it is shown that the selection process is influenced by the resource availability and requirements in the project. As the alternatives that span a larger number of stages require multiple, generalised resources, they underperform in case of a high RC. The opposite is true for a sequence of alternatives that each span a smaller number of stages and require less, dedicated resources. In the case of a low RC, the decision for such alternatives does not warrant a global optimisation. Future research will focus on closing small-scale instances of the RCPSP-AS and providing optimal benchmarks to investigate the performance of heuristic procedures. Furthermore, better lower bounds that incorporate some of the specifications of scheduling problems with alternatives should be developed. More research efforts will also be devoted to the inclusion of the cost of flexibility in the framework of alternative subgraphs in order to further improve the practical applicability of the problem area. First of all, a cost can be induced in the scheduling problem for the inclusion of alternative branches in an alternative subgraph. The penalty factor can increase (non-) linearly with the number of alternative branches that are added to an alternative subgraph and/or the number of alternative subgraphs that are added to the project. From a practical point of view, the existence of nested alternative subgraphs and linked alternative branches might also impact, respectively, the cost of including an alternative subgraph and an alternative branch.

Acknowledgement

We acknowledge the support provided by the Nationale Bank van België (NBB) and the Bijzonder Onderzoeksfonds (BOF) for the project, under contract number BOF12GOA021.

References

Barták, R., Čepek, O., Hejna, M., 2008. Temporal Reasoning in Nested Temporal Networks with Alternatives. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 17–31.

Beck, J., Fox, M. S., 2000. Constraint-directed techniques for scheduling alternative activities. Artificial Intelligence 121 (1), 211 – 250.

Belhe, U., Kusiak, A., 1995. Resource constrained scheduling of hierarchically structured design activity networks. Engineering Management, IEEE Transactions on 42 (2), 150–158.

Blazewicz, J., Lenstra, J., Rinnooy Kan, A., 1983. Scheduling subject to resource constraints: Classification and complexity. Discrete Applied Mathematics 5, 11–24.

Boctor, F., 1993. Heuristics for scheduling projects with resource restrictions and several resource-duration modes. International Journal of Production Research 31, 2547–2558.

Brucker, P., Drexl, A., Möhring, R., Neumann, K., Pesch, E., 1999. Resource-constrained project scheduling: notation, classification, models, and methods. European Journal of Operational Research 112, 3–41.

Capacho, L., Pastor, R., 2006. The ASALB Problem with Processing Alternatives Involving Different Tasks: Definition, Formalization and Resolution. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 554–563.

Capacho, L., Pastor, R., Dolgui, A., Guschinskaya, O., 2009. An evaluation of constructive heuristic methods for solving the alternative subgraphs assembly line balancing problem. Journal of Heuristics 15 (2), 109–132.

Capek, R., Šucha, P., Hanzálek, Z., 2012. Production scheduling with alternative process plans. European Journal of Operational Research 217 (2), 300 – 311.

Creemers, S., De Reyck, B., Leus, R., 2015. Project planning with alternative technologies in uncertain environments. European Journal of Operational Research 242 (2), 465 – 476.

Demeulemeester, E., Herroelen, W., 2002. Project scheduling: A research handbook. Kluwer Academic Publishers.

Demeulemeester, E., Vanhoucke, M., Herroelen, W., 2003. Rangen: A random network generator for activity-on-the-node networks. Journal of Scheduling 6, 17–38.

Elmaghraby, S., 1977. Activity networks: Project planning and control by network models. New York: John Wiley and Sons, Inc.

Elmaghraby, S., Herroelen, W., 1980. On the measurement of complexity in activity networks. European Journal of Operational Research 5, 223–234.

Gillies, D., Liu, J. W., 1995. Scheduling tasks with and/or precedence constraints. SIAM Journal on Computing 24, 797–810.

Glover, F., 1986. Future paths for integer programming and links to artificial intelligence. Computers & Operations Research 13 (5), 533 – 549.

Glover, F., 1989. Tabu search–part 1. ORSA Journal on Computing 1, 190–206.

Glover, F., 1990. Tabu search–part 2. ORSA Journal on Computing 2, 4–32.

Gregory, R., Failing, L., Harstone, M., 2012. Structured Decision Making: A Practical Guide to Environmental Management Choices. Wiley.

Hartmann, S., Briskorn, D., 2010. A survey of variants and extensions of the resource-constrained project scheduling problem. European Journal of Operational Research 207, 1–15.

Hartmann, S., Kolisch, R., 2000. Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem. European Journal of Operational Research 127, 394–407.

Herroelen, W., De Reyck, B., 1999. Phase transitions in project scheduling. Journal of the Operational Research Society 50, 148–156.

Kellenbrink, C., Helber, S., 2015. Scheduling resource-constrained projects with a flexible project structure. European Journal of Operational Research 246 (2), 379 – 391.

Kelley Jr., J., 1963. The critical-path method: Resources planning and scheduling. Prentice-Hall, New Jersey,.

Kis, T., 2003. Job-shop scheduling with processing alternatives. European Journal of Operational Research 151 (2), 307 – 332.

Kolisch, R., Hartmann, S., 1998. Heuristic algorithms for solving the resource-constrained project scheduling problem: Classification and computational analysis.

Kolisch, R., Hartmann, S., 2006. Experimental investigation of heuristics for resource-constrained project scheduling: An update. European Journal of Operational Research 174, 23–37.

Kuster, J., Jannach, D., Friedrich, G., 2008. Extending the RCPSP for modeling and solving disruption management problems. Applied Intelligence 31 (3), 234.

Lee, D., DiCesare, F., 1994. Scheduling flexible manufacturing systems using petri nets and heuristic search. IEEE Transactions on Robotics and Automation 10, 123–132.

Li, K., Willis, R., 1992. An iterative scheduling technique for resource-constrained project scheduling. European Journal of Operational Research 56, 370–379.

Marle, F., Vidal, L.-A., 2016. Managing Complex, High Risk Projects: A Guide to Basic and Advanced Project Management. Springer-Verlag London.

Morton, T., Pentico, D., 1993. Heuristic Scheduling Systems: With Applications to Production Systems and Project Management. No. v. 3 in A Wiley-Interscience publication. Wiley.

Neumann, K., 1990. Stochastic project networks: temporal analysis, scheduling and cost minimization. Springer Science & Business Media.

Patterson, J., 1976. Project scheduling: The effects of problem structure on heuristic scheduling. Naval Research Logistics 23, 95–123.

Ranjbar, M., Davari, M., 2013. An exact method for scheduling of the alternative technologies in r&d projects. Computers & Operations Research 40 (1), 395 – 405.

Russell, R. A., 1986. A comparison of heuristics for scheduling projects with cash flows and resource restrictions. Management Science 32 (10), 1291–1300.

Scholl, A., Boysen, N., Fliedner, M., 2009. Optimally solving the alternative subgraphs assembly line balancing problem. Annals of Operations Research 172 (1), 243.

Tao, S., Dong, Z. S., 2017. Scheduling resource-constrained project problem with alternative activity chains. Computers & Industrial Engineering 114, 288 – 296.

Tavares, L., 1999. Advanced models for project management. Kluwer Academic Publishers, Dordrecht, 1999.

Tsamardinos, I., Vidal, T., Pollack, M. E., 2003. CTP: A new constraint-based formalism for conditional, temporal planning. Constraints 8, 365–388.

Vanhoucke, M., Coelho, J., 2016. An approach using SAT solvers for the RCPSP with logical constraints. European Journal of Operational Research 249 (2), 577 – 591.

Vanhoucke, M., Coelho, J., Batselier, J., 2016. An overview of project data for integrated project management and control. The Journal of Modern Project Management 3 (3), 6–21.

Vanhoucke, M., Coelho, J., Debels, D., Maenhout, B., Tavares, L., 2008. An evaluation of the adequacy of project network generators with systematically sampled networks. European Journal of Operational Research 187, 511–524.

Vidal, L.-A., Marle, F., Bocquet, J.-C., 2011. Using a delphi process and the analytic hierarchy process (ahp) to evaluate the complexity of projects. Expert Systems with Applications 38 (5), 5388 – 5405.

Weglarz, J., 2012. Project Scheduling: Recent Models, Algorithms and Applications. International Series in Operations Research & Management Science. Springer US.

Weglarz, J., Józefowska, J., Mika, M., Waligóra, G., 2011. Project scheduling with finite or infinite number of activity processing modes - a survey. European Journal of Operational Research 208, 177–205.