

# Report

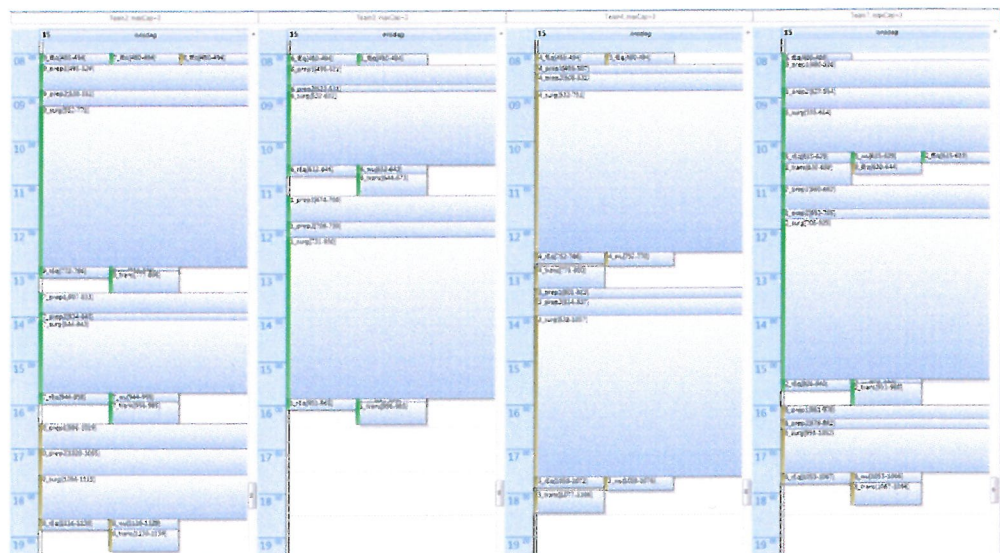
## The Surgery Scheduling Problem

A General Model

### Author(s)

Atle Riise

Carlo Mannino



SINTEF IKT  
SINTEF ICT

Address:  
Postboks 124 Blindern  
NO-0314 Oslo  
NORWAY

Telephone:+47 73593000  
Telefax:+47 22067350

postmottak.ikt@sintef.no  
www.sintef.no  
Enterprise /VAT No:  
NO 948 007 029 MVA

# Report

## The Surgery Scheduling Problem

A General Model

**KEYWORDS:**

Surgery scheduling,  
intervention scheduling,  
resource constrained  
project scheduling  
problem, healthcare

**VERSION**

1

**DATE**

2012-03-30

**AUTHOR(S)**

Atle Riise,  
Carlo Mannino

**CLIENT(S)**

Norges forskningsråd

**CLIENT'S REF.**

**PROJECT NO.**

90A324

**NUMBER OF PAGES/APPENDICES:**

27 + Appendices

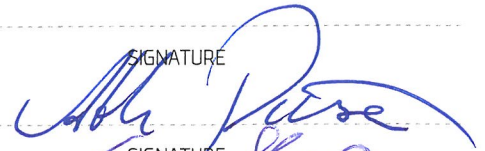
**ABSTRACT**

The term surgery scheduling is used about a variety of strategic, tactical and operational scheduling problems, many of which are critical to an efficient use of hospital resources. Our focus is on operational surgery scheduling problems, which are often NP-hard. The exact problem formulation varies substantially among hospitals, or even hospital departments. In addition, the level of detail vary between different planning situations, ranging from long term patient admission planning to a very detailed planning of the same day's surgeries. This diversity makes it difficult to design scheduling methods and software solutions that are applicable to a wide range of surgery scheduling problems, without extensive customization for each individual application. We approach this challenge by proposing a new generalised model for surgery scheduling problems. The problem can be seen as a rich extension to the resource-constrained project scheduling problem, and we present a structured overview of how our contribution relates to the existing project scheduling literature. We represent this problem by extending the classical disjunctive graph model developed for jobshop scheduling problems. To investigate the power of exact optimization methods in solving generalised surgery scheduling problems, we formulate this disjunctive model as a Mixed Integer Linear Program and solve it by means of a commercial solver. The results show that while it is not capable of solving realistic instances to optimality, the formulation produces good bounds, and promising results were found for interesting sub problems.

**PREPARED BY**

Atle Riise

SIGNATURE



**CHECKED BY**

Geir Hasle

SIGNATURE



**APPROVED BY**

Tomas Nordlander

SIGNATURE



**REPORT NO.**

A22333

**ISBN**

978-82-14-05281-7

**CLASSIFICATION**

Unrestricted

**CLASSIFICATION THIS PAGE**

Unrestricted

# Document history

---

VERSION	DATE	VERSION DESCRIPTION
1	2012-03-30	

# Table of contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
	<b>1.2 An Extended Resource-constrained Project Scheduling Problem</b>	<b>5</b>
	2.1 Definitions and extensions	5
	2.2 Problem statement	6
	2.3 The resource assignment problem	7
	2.4 The sequencing problem and the generalised disjunctive graph	7
	2.4.1 Project subgraphs	7
	2.4.2 Resource subgraphs	8
	2.5 Objective function	12
<b>3</b>	<b>Modelling Surgery Scheduling Problems</b>	<b>12</b>
	3.1 Patients	12
	3.2 Activities	13
	3.3 Resources	13
	3.4 Uncertainty	14
	3.5 Objective components	14
<b>4</b>	<b>Mixed Integer Linear Program formulation</b>	<b>15</b>
<b>5</b>	<b>Computational experiments</b>	<b>18</b>
	5.1 Test instance classes	18
	5.1.1 The "Admission" problem	19
	5.1.2 The "Weekly" problem	19
	5.1.3 The "Daily" problem	19
	5.2 Results	19
<b>6</b>	<b>Conclusion</b>	<b>21</b>
<b>7</b>	<b>References</b>	<b>22</b>
<b>A</b>	<b>Objective Components</b>	<b>24</b>
	A.1 Activity start times	24
	A.2 Use of resources	25
	A.3 Project completion	26
	A.4 Un-scheduled Projects	26

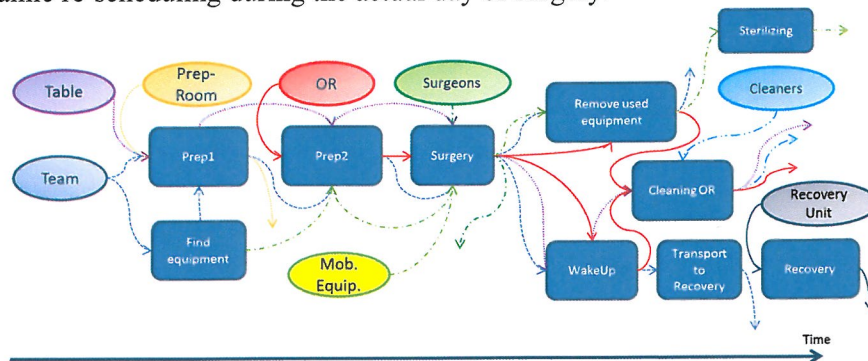
## APPENDICES

A: Objective Components

## 1 Introduction

The term surgery scheduling covers a variety of strategic, tactical and operational scheduling problems (John T. Blake and Carter 2002), many of which are critical to an efficient use of hospital resources. Efficient surgery scheduling on different levels and time scales is also crucial for minimizing patients' waiting time, reducing the number of cancellations, levelling staff work load and improving the overall performance of the hospital (Cardoen et al. 2010). Our focus in this paper is on the operational surgery scheduling problem (SSP), which may be informally described as the task of assigning start times to all surgery related activities for each patient, while reserving capacity for these on a set of constrained renewable resources. Such activities may be, for example, preparation of the patient for surgery, preparation of equipment, removal of un-necessary equipment (if the patient has some infection), surgery, waking the patient, cleaning of the operating room and equipment, transporting the patient to the recovery room, recovery, etc. The involved resources can be, e.g., operating rooms, operation teams, surgeons, equipment, or post-operative beds. Objectives are typically resource overtime, hospitalization costs, intervention costs, operating room utilization, patient's waiting time, and patient or personnel preferences, among others. These scheduling problems are often NP-hard (Hans et al. 2008).

Exact problem formulations vary substantially among hospitals, or even between hospital departments. In addition, the level of detail varies between different planning situations: patient admission planning may consider only one or two kinds of resources, is mainly concerned with allocating a date of admission for each patient, and typically has a long time horizon. Closer to the day of surgery—such as when scheduling surgeries for the next one or two weeks—the number of activities, resources, and choices to make increase. More detailed information about resource availability is also available at this point. Finally, a very detailed schedule is made for the next day, considering all relevant activities and resources in full detail, as illustrated in Fig. 1. This plan is also maintained by dynamic re-scheduling during the actual day of surgery.



**Fig. 1: Example of activities for one patient, with an indication of flows of relevant resources.**

Surgery planning software and associated scheduling algorithms must be able to handle this variation in problem definitions without excessive customisation to each individual hospital and planning situation. So far, however, the SSP literature directly reflects the diversity of the problem domain (see the recent surveys of (Cardoen et al. 2010) and (May et al. 2011)). Different authors use different problem definitions. They also focus on different problem aspects, such as the intensive care unit as a bottleneck resource (Jebali et al. 2006), prediction of hospital bed availability (John T. Blake and Carter 1997), use of mobile equipment (Jebali et al. 2006), uncertainty in surgery durations (Charnetski 1984; Denton et al. 2007; Hans et al. 2008), resource allocation and sequencing in admission planning (Riise and Burke 2011), etc. There is no common ontology for operational surgery scheduling problems, or any common repository of benchmark problems (May et al. 2011). This, of course, makes it difficult to compare algorithmic approaches.

We propose that the great variety of real world SSPs is best handled through a unified, or generalised, formulation of surgery scheduling problems. We view this "generalised surgery scheduling problem" as an extended version of the classical resource-constrained project scheduling problem (RCPSP). The problem includes some new extensions that have not been previously reported in the literature. The proposed modelling framework represents this problem by a generalisation of the classical disjunctive graph representation for jobshop scheduling problems (Roy and Sussmann 1964). This is, as will be discussed in section 2.4, well suited to express a number of problem properties in a straightforward manner. The resulting model provides a basis for the development of a range of different algorithms. Due to the generality of the underlying model these will be applicable across a wide range of real world problems.

To investigate the power of exact optimization methods, we formulate the generalised SSP as a Mixed Integer Linear Program (MILP). We apply a commercial solver to three sets of realistic benchmark instances. These are presented and made available as instances of the (extended) RCPSP, to enable other researchers to develop optimisation methods for the SSP without any knowledge of the application domain. The computational results indicate that while large realistic problems could not be solved, good bounds were found for most test instances. Also, optimal solutions were found for interesting sub-problems. MILP solvers could thus constitute a useful part of a hybrid optimisation method. Section 2 introduces the extended RCPSP problem, and includes a structured overview of how this relates to previous work. In section 3 we explain how our model applies to the generalised SSP. A comprehensive mathematical formulation of the generalised SSP is given as a MILP in section 4. Experimental results are presented in section 5, and we conclude and discuss directions for future research in section 6.

## 2 An Extended Resource-constrained Project Scheduling Problem

### 2.1 Definitions and extensions

In order to put the generalised surgery scheduling problem into context, it is useful to consider its relationship with other scheduling problems. We view the generalised SSP as a rich extension to the resource-constrained project scheduling problem (RCPSP) (Artigues et al. 2008). The classical RCPSP considers the scheduling of the activities of a single project, subject to fixed (conjunctive) precedence constraints. Each activity demands a certain amount of each of a set of renewable resources, and each resource has a constant capacity. Because of the resource capacity constraints, the problem also contains disjunctive precedence constraints. A disjunctive precedence constraint between two activities  $i$  and  $j$  states that either  $i$  precedes  $j$ , or vice versa. Activity durations are not sequence dependent, and the problem is entirely deterministic. Pre-emption is not allowed. The objective is to minimize makespan. In the classification notation of (Brucker et al. 1999), the standard RCPSP is labelled  $PS|prec|C_{max}$ . Even in this basic form, the RCPSP can be shown to be NP-hard in the strong sense (Blazewicz et al. 1983). A recent review of RCPSP variants and extensions can be found in (Hartmann and Briskorn 2010). The problem discussed in this paper includes many of these known extensions, as well as —to the best of our knowledge— extensions that have not been previously reported in the literature. The known extensions in the model are:

1. *Multi-project*: We have multiple-projects, represented in the same graph and sharing the same pool of resources, as introduced in (Pritsker et al. 1969). Each project consists of a set of activities, which is topologically ordered by precedence constraints. E.g., in the SSP, each project contains all treatment activities related to one patient referral.
2. *Project release and completion constraints*: Each project has an earliest possible start time. The model also includes the possibility of expressing a latest possible completion time for a project.

3. *Multiple modes*: For each activity, a set of feasible *modes* are defined out of which one must be selected. A mode defines a set of resources that can perform the activity, the demand for each resource, and the activity duration (Elmaghraby 1977).
4. *Setup times*: Our model includes sequence-independent setup times (Mika et al. 2008).
5. *Maximum delay*: The model contains maximum delay (a.k.a. maximum time lag) constraints between project activities. Note that the decision problem associated with RCPSP with minimum and maximum delay (RCPSP/max) is NP-complete (Hartmann and Briskorn 2010).
6. *Activity time windows*: Our model contains both hard (Bomsdorf and Derigs 2008) and soft (Vanhoucke et al. 2006) time windows.
7. *Dedicated resources*: Some resources in the SSP, such as a surgeon, or a surgery table, can only be assigned to one activity at the time (capacity = 1). Indeed, in some versions of the problem (e.g. in a simple patient admission problem) all resources can be considered as such.
8. *Continuous time*: In the traditional RCPSP, time is treated as a discrete variable. We use a continuous representation of time, as e.g. in (Icmeli and Rom 1996).

In addition to those listed above, we introduce the following new extensions to the classical RCPSP:

1. *Resource periods*: A sequence of non-overlapping available periods is defined for each resource. This is a convenient, and to our knowledge new, way of representing time-dependent resource availability, resource capacity, and block constraints in a disjunctive graph model. Further motivation and details of this concept will be given in section 2.4.2.
2. *Inter-mode constraints*: Compatibility constraints between the modes of activities in the same project.
3. *Project disjunctions*: Some resources are required to complete all work with one project before participating in any activity of any other project.
4. *Mode-dependent precedence constraints*: Depending on the modes chosen for the two activities, there may be a precedence constraint between them.

These extensions will all be discussed in detail in the following. In addition, the SSP also typically includes a variety of objective function components (see section 3.5), rather than the single makespan objective of the classical RCPSP.

## 2.2 Problem statement

In section 3 we will explain how the proposed modelling framework can express a wide range of real world SSP variants. First, however, we present the modelling framework in general project scheduling terms, since we believe that it is applicable also to many other real world scheduling applications. This detailed discussion will include some new and useful concepts. One of these is a generalisation of the disjunctive precedence constraint that was briefly mentioned above for the classical RCPSP. This generalisation will be thoroughly discussed in section 2.4.2.

Our scheduling problem can be loosely formulated as follows:

Given a set of resources  $\mathcal{R}$ , a set of resource periods  $\mathcal{K}^r$  for each resource  $r \in \mathcal{R}$ , a set of projects  $\mathcal{P}$ , a set of activities  $\mathcal{N}^p$  for each project  $p \in \mathcal{P}$ , and a set of modes  $\mathcal{M}^i$  for each activity  $i \in \bigcup_p \mathcal{N}^p$ , find:

1. An assignment of modes to all activities
2. An ordering of activities, and a choice of activity start times that respects a set of conjunctive precedence constraints, and a set of generalised disjunctive precedence constraints,

so that some scalar cost function is minimized.

**Fig. 2: An informal definition of the scheduling problem**

The various kinds of precedence constraints mentioned in Fig. 2 will be discussed in detail below. To simplify the discussion, we decompose this problem into two sub problems; a "Resource assignment problem" and a "Sequencing problem", where the latter includes both ordering and scheduling decisions. Please refer to Table 1 on page 15 for an overview of the notation that we use.

### 2.3 The resource assignment problem

As indicated above, the resource assignment problem consists of choosing exactly one mode for each scheduled activity. The choices of mode for different activities are not independent. If one activity uses a specific resource, other activities in the same project may be required to use the same resource. For example, in a typical SSP the team, surgery room, and surgery table will all be the same for all project activities that use such resources. We model this by defining as input to the model a set of modes  $\mathcal{M}_{j,m}^i$  that are feasible for activity  $i$  given the choice of mode  $m$  for another activity  $j$  of the same project. This is a generalisation of the simpler "Same mode constraints" discussed in (Drexel et al. 2000).

Note that for the SSP, as in many other real world scheduling problems, there may be instances where not all activities can be scheduled within the available time horizon. In such cases it is necessary to choose which activities schedule. Modes must be chosen only for scheduled activities. We demand that each project is either completely scheduled (all the project's activities are scheduled), or not at all (no project activities are scheduled). Let  $\mathcal{R}^i$  be the set of resources that is assigned to activity  $i$  through the choice of mode. Some objective components depend only on the resource assignment. However, many objective components can only be evaluated after solving the associated sequencing problem.

### 2.4 The sequencing problem and the generalised disjunctive graph

Once a mode is chosen for each activity, we are left with a single-mode generalisation of the RCPSp with precedence constraints, multiple projects, various forms of disjunctive precedence constraints, time windows, and time dependent resource capacities. These problem features, as well as activity durations and release dates, can all be modelled in a natural manner by expressing the problem as a generalised disjunctive graph,

$$G = (V, E \cup A) \quad (5.1)$$

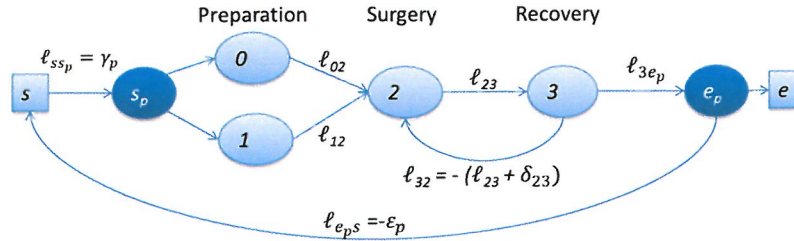
Such a graph representation has been used with various types of algorithms, including exact tree-search based algorithms as well as (meta-) heuristic search methods based on iterative graph modifications (Artigues et al. 2003). In addition to the artificial source ( $s$ ) and target ( $e$ ) nodes, the set of nodes  $V$  contains nodes associated with the project activities as well as nodes that are used to define resource periods. We associate an earliest possible start time  $t_i$  with each node  $i$ . The set of all arcs of the graph represent the precedence constraints of the problem. It can be seen as the union of all project-related arcs,  $E$ , and all resource related arcs,  $A$ .  $G$  can therefore be seen as a union of subgraphs, each associated with a project or a specific resource. These subgraphs share some nodes, but have disjoint arc sets. We discuss these subgraphs in detail in the following.

#### 2.4.1 Project subgraphs

Each project  $p \in \mathcal{P}$  is associated with a positive acyclic subgraph  $G^p = G(V^p, E^p)$  in  $G$  by, where  $V^p = \mathcal{N}^p \cup \{s\} \cup \{e\} \cup \{s_p\} \cup \{e_p\}$ . The artificial nodes  $s_p$  and  $e_p$  represent the "project start" and "project finish" activities (both with zero duration and resource demand) of project  $p$ , respectively. Let the arc set  $E_1^p$  represent all conjunctive precedence constraints in project  $p$ . The lengths of each arc  $(i, j) \in E_1^p$  equals the duration of activity  $i$  in the chosen mode. Let  $E_2^p$  be the set of all arcs from  $s_p$  to any activity of project  $p$  without predecessors. Similarly,  $E_3^p$  is the set of arcs from each activity without a



successor to  $e_p$ . The total arc set of project  $p$  can then be written as  $E^p = E_1^p \cup E_2^p \cup E_3^p \cup \{(s_p, e_p), (s, s_p), (e_p, e)\}$ . The arc set  $E$  that was introduced in Equation (5.1) is the union  $E = \cup_p E^p$ . Fig. 3 shows a simple example from surgery scheduling, with a project subgraph containing preparation, surgery, and recovery activities. The two preparation activities, preparation of the patient and preparation of surgery equipment can be done in parallel. They both have to be completed before the surgery—activity 2—can be performed. The earliest possible starting time of the project start node ( $\gamma_p$ ) represents the project release date, and is modelled as the length of the arc  $(s, s_p)$ , assuming  $t_s = 0$ .  $t_{e_p}$  is the time when all the planned activities of the project are completed.



**Fig. 3: Subgraph for project  $p$ , where arc lengths represent activity durations or time constraints.**

The arc length  $\ell_{3e_p}$  is the duration of activity 3. Note that activity durations are not sequence dependent. The arc  $(3,2)$  in Fig. 3 is given a negative weight to model that activity 3 has to follow activity 2 within a maximum delay,  $\delta_{23}$ . The arc with length  $\ell_{e_p s} = -\epsilon_p < 0$  expresses a deadline for the treatment of the patient. This is useful when the planner wants to enforce some preferences that are not otherwise stated in the model, such as a decision to schedule a project within a restricted time period.

### Mode-dependent precedence constraints

We extend the concept of conjunctive precedence constraints to include “mode-dependent” precedence constraints. These have the form: “if activities  $i$  and  $j$  share resource  $r$ , then  $i$  must precede  $j$ ”. Mode-dependent precedence constraints are similar to the “partially ordered destructive relation” introduced in (Bartels and Zimmermann 2009), which was derived in the context of non-renewable resources in destructive testing projects. Note that in terms of our decomposition into a resource assignment problem and a sequencing problem, all modes are selected at the resource assignment level. For the resulting sequencing problem, the relevant mode-dependent precedence constraints are therefore included in  $E$  just like any other conjunctive precedence constraint.

## 2.4.2 Resource subgraphs

Most activities will use modes that include one or several resources. The problem graph  $G$  includes a resource subgraph,  $G_r = (V^r, A^r)$ , for each resource  $r$ .  $V^r \subset V$  contains all activities whose chosen mode contains resource  $r$ .  $V^r$  also contains a set of nodes that define the available resource periods, as will be explained in the following. The set of all resource related arcs in the project graph (5.1) is the union  $A = \cup_r A^r$ .

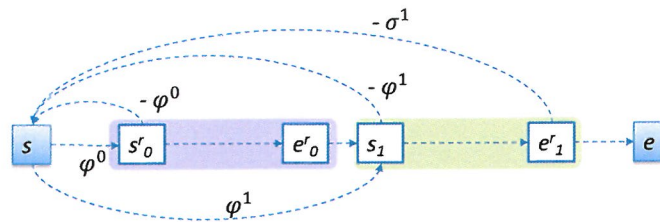
### Resource periods

Many real world scheduling problems run over several days, while many common constraints and objective components are naturally formulated on a daily basis. At the same time, the availability of resources is often time dependent. In the literature on RCPSP, this has been modelled in different ways, including the use of resource availability calendars, and insertion of “forbidden periods”, or artificial “away” activities. In the context of the disjunctive graph model, we find that these aspects of the problem can be modelled in a more flexible way by using the concept of “resource periods”. A

resource period is an interval in time in which a resource is available with a certain capacity. We let  $\mathcal{K}^r = \{1, 2, \dots\}$  be the set of successive resource periods associated with the resource  $r$ . To find a feasible solution to the problem, exactly one resource period  $k \in \mathcal{K}^r$  has to be chosen for each activity and for each resource  $r$  that is assigned to the activity through the choice of mode. As will be shown below, this approach has several advantages:

1. It enables a natural modelling of hard resource availability time windows in terms of lengths of arcs in the resource graph.
2. It facilitates an efficient evaluation of any objectives and constraints that are connected with the end time of each period, which will be a direct result of time propagation through the graph.
3. It offers a convenient way of modelling time dependent resource capacity, as well as block constraints.

Fig. 4 shows a small example, with a subgraph for one resource  $r$  that is available in two resource periods.



**Fig. 4: A resource subgraph for the resource  $r$ , with two resource periods. The arc lengths signify time, and un-labelled arcs have length zero. See the text for details.**

Each period  $k \in \mathcal{K}^r$  is represented in the subgraph of resource  $r$  by a pair of start and end nodes,  $s^r_k$  and  $e^r_k$ . Each resource period is subject to time constraints, which are represented in the resource subgraph by a set of fixed arcs. The start time of each resource period  $k$  is fixed to  $\varphi^k$ . This is modelled by the arcs  $(s, s^r_k)$  and  $(s^r_k, s)$  with lengths  $\varphi^k$  and  $-\varphi^k$ , respectively, as illustrated in the example in Fig. 4. The end time of the resource period  $k$  is bounded by the start time of the following resource period  $k+1$ . These bounds are modelled by the arc  $(e^r_k, s^r_{k+1})$  with zero length (whereas the zero length arc  $(s^r_k, e^r_k)$  is used to bound the end time from below). An additional upper bound  $\sigma^k$  on the resource period end time may, when relevant, be expressed by adding an arc  $(e^r_k, s)$  with length  $-\sigma^k$ . Such arcs can be used to express for example hard constraints on the end of staff working hours, or other resource availability restrictions. All activities that use a given resource period must start and finish within the corresponding time window. This is similar to the time-switch constraint used in (Chen et al. 1997), but in our case the available period is specific to the individual resource. In a feasible solution, the earliest start times of the nodes  $e^r_k$  equals the latest completion time of any activities that are scheduled in resource period  $k$ . In Appendix A, we show that it is straightforward to express some objective components, such as the "overtime" objective, as a function of  $t_{e^r_k}$ .

Different resources may have different period definitions. For example, in surgery scheduling, one finds that two surgery teams may have different—possibly overlapping—working hours. Also, some surgery resources may be continuously available, such as intensive care beds, operating rooms, and equipment. For simplicity and consistency, we model these resources as having one resource period. So called "block constraints" are represented by letting a resource period be available only to a subset of activities. In this aspect, the concept of resource periods is similar (but reverse) to that of "forbidden periods" used in (Drexel et al. 2000).

In the following, we will describe in detail various important problem constraints, and how they can be expressed in our graph model. First, however, we must define a generalisation of the classical concept of disjunctive precedence constraints.

### Generalised disjunctive precedence constraints

In the standard RCPSp, disjunctive precedence constraints are used to model that when a resource has limited capacity, not all activities can be scheduled concurrently. One therefore has to select an ordering among some pairs of activities  $i$  and  $j$ , to obtain a feasible schedule<sup>1</sup>. In classical disjunctive graph models (Roy and Sussmann 1964) this is modelled by pairs of directed arcs  $\{(i,j), (j,i)\}$ , where exactly one arc must be selected from each pair to establish an ordering. The union of selected arcs from all such pairs is called a *selection*. Making a feasible selection is equivalent to removing those arcs that were not selected from the original problem graph, in such a way that the resulting solution graph is positive acyclic.

More complicated disjunctive constraints are also necessary to model our problem. We therefore generalise the traditional concept of disjunction by generalising an arc to a set of arcs, and the pair of arcs to a set of such sets. We use the following definition:

*A disjunctive set of size  $n_j$  is defined as a set of sets of arcs,  $D_j = \{D_j(1), \dots, D_j(n_j)\}$ .*

Obviously, the classical disjunctive pair of directed arcs between activities  $a$  and  $b$  can be seen as a special case of a disjunctive set  $D_j$ , where  $n_j = 2$ , and  $D_j(1) = (a, b)$  and  $D_j(2) = (b, a)$ . Our model also contains disjunctive pairs of arcs that are between different activities; i.e. where  $D_j(1) = (a, b)$  and  $D_j(2) = (c, d)$  for some activities  $a, b, c$ , and  $d$ . The above definition is also a generalisation of the "family of disjunctive sets" used in (Gröflin and Klinkert 2007), where each set in the disjunctive set contains only one arc. Another example of previous generalisations of disjunctive graphs can be found in (Mannino and Mascis 2009), in the context of train scheduling.

Below, we will introduce several "complicated" disjunctive constraints. The set of all disjunctive constraints in our model are represented as a collection of disjunctive sets,  $D = \{D_1, \dots, D_m\}$ <sup>2</sup>. Again, from each disjunctive set  $D_j \in D$  we want to select precisely one of its sets of arcs. We denote this chosen set of arcs by  $S(D_j)$ , and generalize the concept of selection as follows:

*Let  $D = \{D_1, \dots, D_m\}$  be a collection of disjunctive sets. A selection is then defined as*

$$S(D) = \bigcup_i S(D_j)$$

A selection of the disjunctive sets in the problem represented by the graph in Equation (5.1), is feasible if it produces a positive acyclic solution graph:

$$\bar{G} = (V, E \cup \bar{A}), \quad (5.2)$$

where  $\bar{A} = (A \setminus D) \cup S(D)$ . Correspondingly, each resource subgraph  $G_r \subset G$  is transformed by the selection  $S(D)$  into a subgraph  $\bar{G}_r \subset \bar{G}$ . The lengths of each arc  $(i, j) \in \bar{A}$  either equals the duration of the predecessor activity  $i$  plus any setup time for the resource  $r$ , or express some other time constraint. Calculating the longest path in  $\bar{G}$  between  $s$  and all other nodes thus gives the earliest start time of each node, which in turn forms the basis for evaluating the cost of the solution (Appendix A).

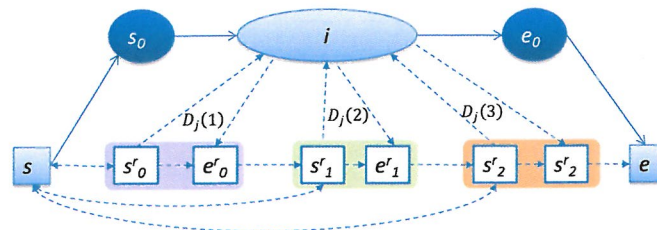
We now go on to describe the most important disjunctive constraints of our problem.

### Disjunctive constraints between activity nodes and resource period nodes

As mentioned above, exactly one resource period must be chosen for each activity, and for each of the activity's assigned resources.

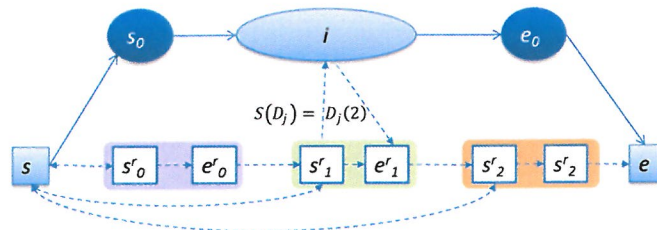
<sup>1</sup> Note that in our model, a disjunctive constraint between the activities needs only be included in the model if they can both be assigned to the same resource period.

<sup>2</sup> Note that the collection  $D$  corresponds to the "disjunctive constraints" mentioned in Fig. 2.



**Fig. 5:** Activity “ $i$ ” can be executed in any one of three available periods for the single resource. Here, and in the following figures, solid arcs belong to a project graph, while dotted arcs belongs to a resource graph.

This choice is modelled by defining a disjunctive set  $D_j = \{D_j(1), D_j(2), \dots\}$  for each activity  $i$  and each resource  $r \in \mathcal{R}^i$ . Each set  $D_j(k)$  contains two arcs linking the activity to resource period  $k \in \mathcal{K}^r$ . This is illustrated in Fig. 5. Selecting a resource period for activity  $i$  corresponds to choosing a set  $S(D_j) = D_j(k)$ , for some  $k \in \{1, \dots, |D_j|\}$ . In our simple example, choosing  $S(D_j) = D_j(2)$  will give the graph in Fig. 6.

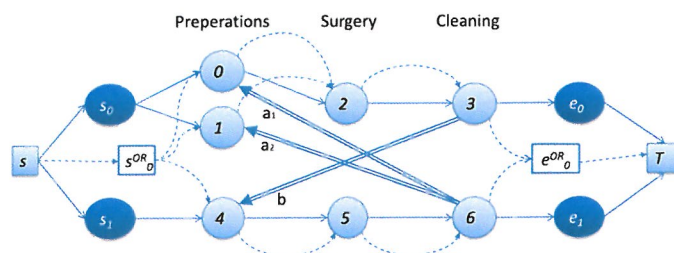


**Fig. 6:** The example of Fig. 5, after a selection has been made.

### Project disjunctions

It is a property of the surgery scheduling problem that some resources, such as the operating room, can only be used in one project (patient) at the time. All activities of one project must complete their use of the resource before it can be used in any activity of any other project. This comes from the fact that as long as activities for one patient are not completed in the operating room, no activities concerning other patients can happen in the same room. We can think of this constraint as a disjunction between projects. To our knowledge, such a constraint has not been previously studied, although it bears some resemblance to constraints that were used in (Bomsdorf and Derigs 2008; Nonobe and Ibaraki 2002), which demand that no other activities can be scheduled between certain pairs of precedence-related activities.

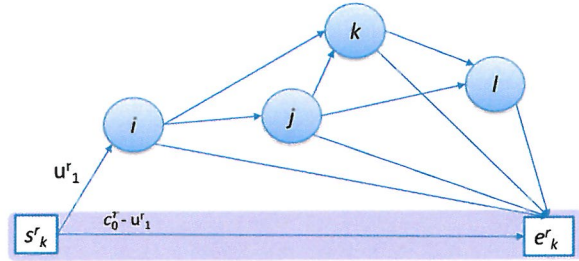
Project disjunctions are modelled by creating one disjunctive set  $D_j$  for each pair of projects  $(p, p')$ , and for each relevant, “project-disjunctive”, resource  $r$ . The situation is illustrated in the example of Fig. 7, where a disjunctive set  $D_j = \{\{a_1, a_2\}, \{b\}\}$ . That is, exactly one of the arc sets  $\{a_1, a_2\}$  or  $\{b\}$  must be selected.



**Fig. 7:** Project disjunction as a disjunctive set between two projects, on the resource “OR”. The double arcs represent the disjunctive set, from which either  $\{a_1, a_2\}$  or  $\{b\}$  must be selected.

### Resource capacity and flow

So far we have ignored resource demand and capacity constraints. In our model, both the capacity of resource  $r$  and the corresponding demand  $d_i^r$  of activity  $i$ , can be larger than one. If the activities that are assigned to a resource period  $k \in \mathcal{K}^r$  have a total demand that is larger than the resource capacity in the period,  $c_k^r$ , they cannot all be scheduled concurrently. To avoid this, a partial ordering of the activities must be imposed by suitable decision variables (see section 4). Such a partial ordering can be represented by the arcs in a graph for each resource period. The nodes are the resource period's start and end nodes, as well as the activity nodes (Fig. 8).



**Fig. 8** The flow network associated with a single resource period. Arc labels refer to flow.

Like in (Artigues et al. 2003), we model the use and capacity of the resource by a flow in this network. The start node  $s_k^r$  of resource period  $k$  can be seen as a source node with total (positive) divergence  $c_k^r$ . Similarly, the end node  $e_k^r$  is a sink node with divergence  $-c_k^r$ . The divergence of all other nodes is zero. In addition to the standard flow conservation constraints, we also introduce activity demand constraints which state that the total flow into—and out of—each activity node  $i$  must equal  $d_i^r$ . Note that this model can easily be reduced to the classical network flow model by representing each activity by two nodes, one source and one sink node, both with divergence of magnitude equal to  $d_i^r$ .

## 2.5 Objective function

The traditional objective in project scheduling is to minimize only makespan or project lateness. Several other objective components are necessary to model the generalised SSP (see section 3.5).

## 3 Modelling Surgery Scheduling Problems

Now that we have presented the general reference model, we can discuss how it can express real world SSP concepts. For completeness we also include the relevant mathematical notation for each concept, which will be used in the MILP formulation in section 4. A summary of this notation may be found in Table 1.

Note that the model can easily express re-scheduling based on an existing solution. This is the norm in practical surgery scheduling, since an existing plan must usually be respected to some degree. So must manually made planning decisions. This is handled by adding appropriate constraints or preferences when the SSP is mapped into the generalised model

### 3.1 Patients

The SSP concerns the scheduling of activities associated with specific referrals for surgery for each patient. Model parameters are set based on patient properties as follows:

- $\mathcal{P}$ : The set of projects, each representing a patient referral. Note that compared to many other RCPSP variants, these projects are typically quite small, with less than, say, 10 or 15 activities. Also, not many of these activities can be performed in parallel, as most of them involve the patient.
- $\mathcal{N}^p$ : This set represents the real world hospitals activities associated with project  $p$ .

- $\omega_p$ : The due date for the project is set to the guaranteed deadline date for the corresponding surgery. In Norway, at least, violations of this deadline is a common measure of the hospital's performance.
- $\chi_p$ : The reference date for the project is typically set to the date at which the referral was written. This is the date from which waiting time is counted.

### 3.2 Activities

Each hospital activity is modelled with the following properties:

- $\vartheta_i^m$ : The duration of activity  $i$  depends on which resources that are used, i.e. the chosen mode. E.g., the duration of the surgery itself depends on the surgeon, whether surgeons in training are present, the type of surgery, etc. All durations are treated as deterministic; see section 3.4.
- $\mathcal{G}_i, [\alpha_i, \beta_i]$ : All relevant precedence constraints between activities, as well as any hard time constraints are included by adding weighted directed arcs to the corresponding project graph,  $G^P$ .
- $[\zeta_i, \eta_i]$ : Preferred time windows are set either to reflect patient preferences or based on previous planning to achieve a minimum disruption with respect to the existing schedule.
- $\delta_{ij}$ : The maximum delay between activities  $i$  and  $j$  is used to model how long the patient can wait between activities. For most pairs of successive activities in the operating room, this parameter is set to zero, as no delay is permitted. For SSP instances with many activities per project, these activities are therefore very tightly linked through such constraints. This has consequences for iterative improvement algorithms such as local search, since it severely limits the freedom of modifying the mode or the starting time of single activities, unless the search is allowed to move into infeasible parts of the search space.
- $\mathcal{M}^i$ : An activity  $i$  may need one or several resources to be present. A mode is added to this set for each feasible combination of resources that can execute activity  $i$ .
- $\mathcal{M}_{j,m}^i \subseteq \mathcal{M}^i$ : A SSP is typically tightly constrained by mode consistency constraints. The set  $\mathcal{M}_{j,m}^i$  contains those modes that are legal for activity  $i$  when activity  $j$  uses mode  $m$ . For the example in Fig. 1, this is used to express that if the activity "Surgery" uses a specific operating room, "OR", then so must the activities "Prep2", "Wake Up", "Remove used equipment", and "Cleaning OR".
- $\tau_i$ : When relevant, e.g. in the case of children or patients with diabetes, this target activity start time is set to the earliest possible time in the day.
- $\Pi^r$ : This set of mode-dependent precedence relationships are set to express (e.g.) that each "Prep1" activity<sup>3</sup> must precede the corresponding "Remove superfluous equipment" activity<sup>4</sup> if they are both assigned to the operation room resource  $r$ . However, if the "Prep1" activity is chosen to use a separate preparation room, the two activities can happen in parallel.
- $\nu_i^r$ : This penalty is used to express a preferred choice of surgeon,  $r$ , for a given surgery activity,  $i$ .

### 3.3 Resources

The number of resources that are included varies between hospitals and planning situations. In admission planning, which is typically performed quite some time before the date of surgery, only the most critical resources are included. Often, even if a resource is actually critical it may be omitted simply because no information exists about its availability. Closer to the day of surgery more information is available, and more resources are included in the planning. A detailed SSP may e.g. include resources such as

<sup>3</sup> The first, pre-anesthesia, step of preparation of a patient for surgery

<sup>4</sup> Equipment that is not needed in the OR is removed if the patient is infected, so that it does not have to be sterilized after the surgery.

operating rooms, intensive care units, stationary equipment, surgeons, anaesthesiologists, surgery teams, cleaning staff, surgery tables, and other mobile equipment. These constitute the model's set of *resources*,  $\mathcal{R}$ . For each  $r \in \mathcal{R}$ :

- $\mu_r^m$ : This parameter is set to reflect how much of each resource the mode  $m$  demands, in terms of resource units. E.g., transporting the patient to the recovery unit after surgery may require only one team nurse. If the team has three nurses, the team will have capacity 3, while the transport activity will have modes that require only 1 unit of the team resource.  $\mu_r^m = 0$  means that mode  $m$  does not involve resource  $r$ .
- $\mathcal{K}^r$ ,  $\varphi^k$ ,  $\sigma^k$  and  $c_k^r$ : In the SSP problem definition, each resource has a calendar that defines periods in which the resource is available (e.g. each day's working hours for employees, or allocated time blocks for operating rooms). Based on this, a corresponding set of non-overlapping resource periods,  $\mathcal{K}^r$ , is defined in the corresponding resource graph,  $G_r$ . The start and end times ( $\varphi^k$  and  $\sigma^k$ , respectively) of each  $k \in \mathcal{K}^r$  are modelled as weighted arcs in  $G_r$ . A flow is defined on the arcs of  $G_r$  to reflect the capacity ( $c_k^r$ ) of each resource period, in "resource units". E.g., a "team" resource may have capacity 3, which enables it to perform three activities in parallel, each of which requires one team unit. Another example is if a problem instance does not define individual surgeon resources, but rather surgeon groups with time dependent capacities in terms of the number of parallel surgeries the group can perform.
- $\mathcal{K}_i^r \subseteq \mathcal{K}^r$ : Allocation of surgery time, on resource  $r$ , to medical specialties<sup>5</sup> are expressed as block constraints by defining these sub sets of resource periods that are available to each activity,  $i$ . Resource skills and other resource/activity compatibility parameters are also considered.
- $\mathcal{C} \subseteq \mathcal{R}$ : This set is used to define those resources, usually operating rooms, that can only participate in the treatment of one patient at the time (without any interlacing of activities concerning other patients).
- $\rho^r$ : Setup time is set for equipment that needs to be sterilized between surgeries.
- $\zeta^k$ : The preferred finish time is used to express the end of working hours for staff, in the case where a wider resource period is defined to allow for overtime.

### 3.4 Uncertainty

In the presented model, activity durations are treated as deterministic. In practical use, the durations are set based on statistical information from daily operations. They include a scalable buffering to model robustness, or alternatively, encourage over-booking of resources. The model can be extended to include a more rigorous treatment of activity durations, but this is outside the scope of this paper.

In admission planning, the uncertainty in the number of future arrivals of patients must be taken into account. This can be done by limiting future resource availability.

We do not consider robustness with respect to short term disruptions, e.g. in the form of arriving emergency care patients or unexpected cancellations. The disruptions caused by such events are very large, and may be best handled by a combination of strategic dimensioning of resource capacities, and efficient dynamic re-scheduling.

### 3.5 Objective components

In the review in (Cardoen et al. 2010), the authors give an overview of SSP objective components, and classify them according to how they are treated in the literature. Common SSP objective components are patient waiting time, makespan, time of day preferences, soft time windows, minimum disruption, resource overtime, resource utilization, minimum resource activity, and choice of surgeon preferences. Finally, it is common that surgery scheduling with short time horizons must include the option not to schedule some of the patients. These are then postponed to the next time period. The SSP

---

<sup>5</sup> E.g. "General Surgery", or "Orthopedics"

therefore often includes an objective component that encourages scheduling of as many patients as possible, and gives a priority between patients when some must be postponed. In Appendix A we describe how these SSP objective components may be expressed in the proposed model.

#### 4 Mixed Integer Linear Program formulation

We now go on to show how our general project scheduling model can be formulated as a mixed integer linear program (MILP). This provides a precise and comprehensive description of the generalised SSP. It also enables us to assess the solving power of a commercial MILP solver, when applied to our formulation. The level of generality makes the model rather complex, but some simplification is possible for specific SSP variations. Note that while we have previously de-composed the problem into a resource assignment problem and a sequencing problem, the following MILP formulation includes the whole unified scheduling problem. Table 1 summarizes the notation that we need.

**Table 1: Summary of notation.**

Definitions	
$\mathcal{P}$	The set of all projects
$\mathcal{N}^p$	The set of activities belonging to project $p$
$\mathcal{N}$	The set of all activities. $\mathcal{N} = \bigcup_{p \in \mathcal{P}} \mathcal{N}^p$ .
$\mathcal{M}^i$	The set of modes for activity $i$ .
$\mathcal{M}_{j,m}^i \subseteq \mathcal{M}^i$	The set of all modes that are feasible for activity $i$ when activity $j$ uses mode $m$ .
$\mathcal{R}$	The set of all resources, $\mathcal{R} \neq \emptyset$ .
$\mathcal{R}^i$	The resources assigned to activity $i$ , through the choice of mode.
$\mathcal{K}^r$	The set of resource periods for resource $r$ .
$\mathcal{K}_i^r \subseteq \mathcal{K}^r$	The set of resource periods for resource $r$ that are available to activity $i$ .
$\mathcal{G}_i$	The set of immediate predecessor activities of activity $i$ , in the project graph.
$\mathcal{C}$	The set of all resources that can only participate in one project at the time, meaning that all activities from a project has to be completed before the resource takes part in any other activity from any other project. $\mathcal{C} \subseteq \mathcal{R}$
$\mathcal{S}_p^r$	The set of all activities of the project $p$ that does not have any predecessors on resource $r$ . $\mathcal{S}_p^r \subseteq \mathcal{N}^p$ . Defined for each $r \in \mathcal{C}$ .
$\mathcal{E}_p^r$	The set of all activities of the project $p$ that does not have any followers on resource $r$ . $\mathcal{E}_p^r \subseteq \mathcal{N}^p$ . Defined for each $r \in \mathcal{C}$ .
$s_k^r, e_k^r$	The start and finish nodes of resource period $k \in \mathcal{K}^r$ , respectively.
Problem parameters	
$H$	The planning horizon, or planning period length.
$\mathfrak{z}$	The number of time units in a day.
$[\zeta_i, \eta_i]$	Preferred (soft) time window for activity $i$ .
$[\alpha_i, \beta_i]$	Hard time window constraint for activity $i$ .
$\omega_p$	Due date for project $p$ , given in days. Preference, not a hard constraint.
$[\gamma_p, \varepsilon_p]$	Hard time window constraint for project $p$ . $\gamma_p$ is the project release date, while $\varepsilon_p$ can be set to model any absolute upper limit on project completion (or to $H$ if no such limit exists).
$\chi_p$	Reference date for project $p$ , given in days.
$c_k^r$	The capacity of the resource period $k$ of resource $r$ , in <i>resource units</i> .
$\mu_r^m$	Mode $m$ 's use of resource $r$ , given in resource units.
$u_r^m$	= 1 if mode $m$ uses resource $r$ ( $\mu_r^m > 0$ ), zero otherwise.
$\delta_{ij}$	Maximum delay between the completions of activity $i$ and the start of activity $j$ .
$\vartheta_i^m$	The duration of activity $i$ in mode $m$ .
$\rho^r$	The setup time for resource $r$ . $\rho^r \neq 0$ only for resources with maximum capacity 1.
$\varphi^k$	Fixed starting time of resource period $k$ . This is a hard constraint.
$\sigma^k$	Latest end time of resource period $k$ . The maximum value for this input parameter



	must be the start time for the following resource period on the same resource. This is a hard constraint.
$\zeta^k$	Preferred finish time for resource period $k$ , which is typically used in objective components concerning resource utilisation.
$\Pi^r$	The set of ordered pairs of activities, $(i,j)$ that are associated through “mode-dependent” precedence constraints in such a way that if they both use resource $r$ , then $i$ must precede $j$ .
$\tau_i$	Preferred time of day, given in time units since midnight, for activity $i$ . If no preference exists, the corresponding weight in the “Preferred time of day objectives” (Equation (10.2)) will be zero, and the value of $\tau_i$ will be ignored.
$v_i^r$	= 0 if resource $r$ is a preferred resources of activity $i$ , and 1 otherwise.
$\lambda^r, \Lambda^r$	Resource $r$ prefers to participate in at least $\lambda^r$ activities, from the set of desired activities, $\Lambda^r$ . These parameters are 0 and empty, respectively, for resources without such preferences.
Decision variables	
$x_i^m$	= 1 if activity $i$ uses mode $m \in \mathcal{M}^i$ and 0 otherwise.
$q_i^k$	= 1 if activity $i$ uses resource period $k \in \mathcal{K}^r$ , for resource $r$ , and 0 otherwise.
$t_i$	The non-negative starting time of activity $i$ .
$f_{ij}^r$	The non-negative flow of units of resource $r$ between activity $i$ and activity $j$ .
$z_{ij}$	= 1 if activity $i$ precedes activity $j$ , and 0 otherwise.
$d_{p,p'}^r$	= 1 if $p$ precedes $p'$ on $r$ , and 0 if $p'$ precedes $p$ on $r$ , where $r \in \mathcal{C}$ and $p, p' \in \mathcal{P}$ .
Derived variables	
$y_i$	The completion time of activity $i$ .
$\epsilon_k$	The earliest start time of the artificial “end activity” of resource period $k$ , which is the same as the maximum completion time over all activities that are scheduled in resource period $k$ .
$C_p$	Completion time of project $p$
$d_i^r$	The demand of activity $i$ for resource $r$ in the chosen mode.
$g_i^r$	= 1 if activity $i$ uses resource $r$ in the chosen mode, and 0 otherwise.
$h^p$	Binary variable expressing whether project $p$ is un-scheduled ( $h^p = 1$ ), or scheduled ( $h^p = 0$ ).

Unless otherwise specified, all time-related parameters or variables above are given in time units since midnight on the first day of the planning period. For the SSP the natural time units are minutes ( $\beta = 1440.0$ ). We therefore continue to use continuous time, since a time indexed formulation would need to have many more variables. The MILP formulation is defined by equations (7.1) through (7.34) ( $M$  represents a large number).

First we define the decision variables and some derived variables:

$$x_i^m \in \{0,1\}; \forall i \in \mathcal{N}, \forall m \in \mathcal{M}^i \quad (7.1)$$

$$z_{ij} \in \{0,1\}; \forall i, j \in \mathcal{N} \quad (7.2)$$

$$t_i \in \mathbb{R}_+; \forall i \in \mathcal{N} \quad (7.3)$$

$$q_i^k \in \{0,1\}; \forall r \in \mathcal{R}, \forall k \in \mathcal{K}_i^r \quad (7.4)$$

$$f_{ij}^r \in \mathbb{R}_+; \forall r \in \mathcal{R}, \forall i, j \in \mathcal{N} \cup \left( \bigcup_{k \in \mathcal{K}^r} \{s_k^r, e_k^r\} \right) \quad (7.5)$$

$$d_{p,p'}^r \in \{0,1\}; \forall r \in \mathcal{C}, \forall p, p' \in \mathcal{P} \quad (7.6)$$

$$h^p \in \{0,1\}; \forall p \in \mathcal{P} \quad (7.7)$$

$$y_i \in \mathbb{R}_+; \forall i \in \mathcal{N} \quad (7.8)$$

$$g_i^r \in \{0,1\}; \forall r \in \mathcal{R}, \forall i \in \mathcal{N} \quad (7.9)$$

$$y_i = t_i + \sum_{m \in \mathcal{M}^i} \vartheta_i^m x_i^m; \forall i \in \mathcal{N} \quad (7.10)$$

$$g_i^r = \sum_{m \in \mathcal{M}^i} u_r^m x_i^m; \forall r \in \mathcal{R}, \forall i \in \mathcal{N} \quad (7.11)$$

$$d_i^r = \sum_{m \in \mathcal{M}^i} \mu_r^m x_i^m; \forall r \in \mathcal{R}, \forall i \in \mathcal{N} \quad (7.12)$$

Equation (7.10) defines the completion times for each activity, while (7.11) defines the variable  $g_i^r$  that indicates whether or not activity  $i$  uses resource  $r$  in the chosen mode. (7.12) defines resource demands for each activity in the chosen mode.

$$\sum_{m \in \mathcal{M}^i} x_i^m = 1 - h^p; \forall i \in \mathcal{N} \quad (7.13)$$

$$\sum_{k \in \mathcal{K}_i^r} q_i^k = g_i^r; \forall i \in \mathcal{N}, \forall r \in \mathcal{R} \quad (7.14)$$

$$x_j^m \leq \sum_{m' \in \mathcal{M}_{j,m}^i} x_i^{m'}; \forall i, j \in \mathcal{N}^p, \forall p \in \mathcal{P} \quad (7.15)$$

(7.13) says that exactly one mode must be chosen for each activity of a scheduled project. Equation (7.14) states that exactly one resource period must be chosen for each resource that is assigned to an activity. This period must be selected amongst those available to the activity; thus the equation also expresses block constraints. Consistency between modes of different activities is ensured by (7.15).

$$z_{ij} + z_{ji} \leq 1; \quad \forall i, j \in \mathcal{N} \quad (7.16)$$

$$z_{ij} = 1; \quad \forall i \in \mathcal{G}_j, \forall j \in \mathcal{N} \quad (7.17)$$

$$z_{ij} \geq (g_i^r + g_j^r - 1); \quad \forall r \in \mathcal{R}, \forall (i, j) \in \Pi^r \quad (7.18)$$

$$t_j - y_i - \rho^r \text{Max}(0, g_j^r + g_i^r - 1) \geq (z_{ij} - 1) M; \quad \forall i, j \in \mathcal{N}, \forall r \in \mathcal{R} \quad (7.19)$$

$$d_{p,p'}^r + d_{p',p}^r = 1; \quad \forall p, p' \in \mathcal{P} \quad (7.20)$$

$$z_{ij} \geq d_{p,p'}^r + g_j^r + g_i^r - 2; \quad \forall r \in \mathcal{C}, \forall i \in \mathcal{E}_p^r, \forall j \in \mathcal{S}_{p'}^r, \forall p, p' \in \mathcal{P} \quad (7.21)$$

$$z_{ji} \geq -d_{p,p'}^r + g_j^r + g_i^r - 1; \quad \forall r \in \mathcal{C}, \forall i \in \mathcal{S}_p^r, \forall j \in \mathcal{E}_{p'}^r, \forall p, p' \in \mathcal{P} \quad (7.22)$$

(7.16) says that if activity  $i$  precedes activity  $j$ , then activity  $j$  cannot precede activity  $i$ . (7.19) expresses the link between the precedence variable and activity start and completion times, taking resource setup time into account. Note that such setup time only has meaning for resources with capacity one; for all other resources  $\rho^r = 0$ . We can use  $M=H$ . Equations (7.17) and (7.18) express project graph and mode dependent precedence constraints, respectively. (7.20), (7.21) and (7.22) enforce project disjunctions. The definition of  $\mathcal{S}_p^r$  and  $\mathcal{E}_p^r$  assumes that the feasible set of modes for all activities in  $p$  are bound through mode compatibility in a way that ensure that if resource  $r$  is used, then  $\mathcal{S}_p^r$  and  $\mathcal{E}_p^r$  contains the set of first and last activities of  $p$  on  $r$ , respectively. For projects for which this assumption does not hold, the constraints must be taken between all pairs of activities, i.e.  $\mathcal{S}_p^r = \mathcal{E}_p^r = \mathcal{N}^p$ .

$$t_i \geq \text{Max}(\alpha_i, \gamma_p); \quad \forall p \in \mathcal{P}, \forall i \in \mathcal{N}^p \quad (7.23)$$

$$y_i \leq \text{Min}(\beta_i, \varepsilon_p); \quad \forall p \in \mathcal{P}, \forall i \in \mathcal{N}^p \quad (7.24)$$

Equations (7.23) and (7.24) express activity and project time windows

$$f_{ij}^r \leq z_{ij} M; \quad \forall r \in \mathcal{R}, \forall i, j \in \mathcal{N} \quad (7.25)$$

$$\sum_{j \in \mathcal{N}} f_{ij}^r + \sum_{k \in \mathcal{K}_i^r} f_{ie_k}^r = d_i^r; \quad \forall i \in \mathcal{N}, \forall r \in \mathcal{R} \quad (7.26)$$

$$\sum_{i \in \mathcal{N}} f_{ij}^r + \sum_{k \in \mathcal{K}_j^r} f_{s_k i}^r = d_i^r; \quad \forall i \in \mathcal{N}, \forall r \in \mathcal{R} \quad (7.27)$$

$$f_{s_k i}^r \leq c_k^r q_i^k; \quad \forall i \in \mathcal{N}, \forall k \in \mathcal{K}^r, \forall r \in \mathcal{R} \quad (7.28)$$

$$f_{ie_k}^r \leq c_k^r q_i^k; \quad \forall i \in \mathcal{N}, \forall k \in \mathcal{K}^r, \forall r \in \mathcal{R} \quad (7.29)$$

$$\sum_{i \in \mathcal{N}} f_{s_k i}^r + f_{s_k e_k}^r = c_k^r; \quad \forall k \in \mathcal{K}^r, \forall r \in \mathcal{R} \quad (7.30)$$

$$\sum_{i \in \mathcal{N}} f_{ie_k}^r + f_{s_k e_k}^r = c_k^r; \quad \forall k \in \mathcal{K}^r, \forall r \in \mathcal{R} \quad (7.31)$$

(7.25) ensures that the flow associated with resource  $r$  is zero if activity  $i$  does not precede activity  $j$ , and, vice versa, forces  $z_{ij} = 1$  if there is any such flow between the two activities.  $M = \min(\max_{m \in \mathcal{M}^i} \mu_r^m x_i^m, \max_{m \in \mathcal{M}^j} \mu_r^m x_j^m, \max_{k \in \mathcal{K}_i^r \cup \mathcal{K}_j^r} (c_k^r))$ .

Flow conservation and demand satisfaction is ensured through the constraints in (7.26) and (7.27). Equations (7.28) and (7.29) ensures that the flow from any resource period start (end) node to (from) any activity  $i$ , is zero if the activity is not scheduled in that resource period. Resource capacity for each resource period is given by (7.30) and (7.31).

$$t_i - \varphi^k q_i^k \geq 0; \quad \forall i \in \mathcal{N}, \forall r \in \mathcal{R}, \forall k \in \mathcal{K}_i^r \quad (7.32)$$

$$y_i - \sigma^k q_i^k - (1 - q_i^k) M \leq 0; \quad \forall i \in \mathcal{N}, \forall r \in \mathcal{R}, \forall k \in \mathcal{K}_i^r \quad (7.33)$$

$$t_j - y_i - \delta_{ij} \leq 0; \quad \forall i, j \in \mathcal{N} \quad (7.34)$$

Equations (7.32) and (7.33) force activity start times to lie within hard time windows for the chosen resource period. In (7.33), we can use  $M=H$ . Finally, (7.34) is the max-delay constraint between project activities.

Note that if a given resource has a constant capacity = 1, and the maximum demand for this resource for any activity is also one, then we can omit the flow formulation in equations (7.26) through (7.31). For such a resource  $r$ , we can use the simpler classical resource disjunctions instead. Together with (7.16), this can be expressed by:

$$z_{ij} + z_{ji} \geq g_i^r + g_j^r - 1 \quad \forall i, j \in \mathcal{N} \quad (7.35)$$

## 5 Computational experiments

As the proposed model aims to cover a large majority of the real world SSPs, it is necessarily general and not tuned to any specific problem variant. All the same, it is useful to investigate how a commercial MILP solver will perform on the model.

### 5.1 Test instance classes

The problem class's inherent diversity makes it impossible to test it for the entire variety of SSP problems. However, we present a representative set of realistic test instances that together should give some indication of the model's performance in typical planning situations. The instances can be ordered in three classes, characterized by their planning horizon and the level of detail in resource information. All the test problems are based on the basic configuration of a surgery department in a medium sized Norwegian hospital. The test data are available online in the XML format of the proposed general project scheduling model. This will enable researchers in project scheduling to develop and test

algorithms for the generalised SSP without the need for any understanding of the surgery scheduling domain.

For all instances, the objective function can be written as a weighted sum of objective components as defined in Appendix A. The choice of objective components for each respective class of test instances is given below.

### 5.1.1 The "Admission" problem

This is a typical admission planning problem, with surgeon and operating room resources, both with constant unit capacity. The surgery is the only project activity. Our test cases have a planning horizon of about 4 or 5 months. Each surgery that was already scheduled in the existing plan are constrained by time windows to the day (resource period) on which they were originally scheduled. There are four objective components of the admission problem; patient waiting time, surgeon overtime and "children early in the morning", and the "un-scheduled" objective. The internal weighting between patients in "children early" objective is inversely proportional to the patient's age, and zero for patients above a certain age (adults). Weights for the "un-scheduled" objective are calculated based on the relative waiting time that the patient would have if her surgery was scheduled at the end of the planning horizon<sup>6</sup>.

### 5.1.2 The "Weekly" problem

In the "weekly" surgery scheduling problem, we include three activities per project: surgery, recovery, and the cleaning of operating rooms. The involved resources are surgeons, operating rooms, a recovery unit, and cleaning personnel. The planning horizon is one week. The objective components include those of the "Admission problem". In addition, a violation of soft time windows is added, where each time window defines the day on which each surgery was scheduled in the previous existing schedule. This replaces the hard time window constraints that were used in the admission planning problem.

### 5.1.3 The "Daily" problem

This is a very detailed SSP, with a planning horizon of one day. The considered resources are surgeons, operating rooms, teams, cleaning personnel, and the recovery unit. The project activities are preparation of the patient (pre- and post-anesthesia as separate activities), finding the necessary equipment, removing superfluous equipment in cases with infected patients, surgery, waking the patient, removing used equipment, transporting the patient to the recovery unit, cleaning the operating room, and recovery. The objective function is the same as for the "Weekly" problem, except that the waiting time component and the soft time windows are excluded.

## 5.2 Results

All experiments were carried out using 64-bit CPLEX 12 on a Dell Precision M4500 laptop computer with an Intel Core i7 CPU with four 1.73 GHz cores, and with 8Gb RAM. The CPLEX setup is standard, except that we specify that it should use parallel mode with up to 6 threads. In all experiments, CPLEX was given an initial feasible solution that was generated by a simple construction heuristic, applying a suitably modified version of the sequential schedule generation scheme (Artigues et al. 2008). The heuristic schedules one project at the time, in the order of increasing project due date. Each CPLEX run had a timeout that was chosen to reflect a realistic level of patience on the part of the planner in each planning situation: 5 minutes for the daily planning problem, 60 minutes for admission planning, and 15 minutes for weekly planning.

---

<sup>6</sup> These test instances correspond to those that were used in (Riise and Burke 2011), but with the modification that the overtime objective is replaced by the linear formulation of Equation (10.10), that the choice of surgeon is no longer pre-determined, and that the weights for the "un-scheduled" objective is set individually for each patient. Also, the normalisation of each objective component is changed.

Computational results for these three problem classes are given in Table 2, Table 3, and Table 4. The instance names indicate the number of projects in each case; e.g. "w32a" is a "weekly" problem instance with 32 projects). The columns "#var" and "#constr" contains the number of variables and constraints, respectively, in the MILP formulation. We report the objective value of the initial heuristic solution ("Initial"), and the value of the best found solution ("Result"). If this was proven optimal, we report the wall time ("T") used to supply this proof. The column "Bound" contains the best lower bound produced by CPLEX in any of our runs. Bold face indicates improvement with respect to the initial solution, or proof of optimality. Result values in italic indicate that the search stopped because of lack of memory.

**Table 2 Results for the "daily" class**

Case	#var	#constr	Initial	Result	T(s)	Bound
d10a	47051	131887	0.0387	0.0387	-	0
d10b	48325	135124	4.0792	4.0792	-	3.9646
d10c	48325	135124	0.4472	0.4472	-	0.3785
d10d	49614	138392	45350.5330	45350.5330	-	0.3646
d10e	47057	131895	4.1875	4.1875	-	2.8298
d5a	13405	39529	3.9861	3.9861	-	3.8819
d5b	14087	41386	4.6715	4.6715	-	4.0660
d5c	14081	41378	0	0	2.17	0
d5d	12747	37715	4.4875	4.4875	-	4.3792
d5e	12741	37707	0	0	2.42	0

For the "daily" instances, optimality was only proven when it was trivial, i.e. in the cases where the initial objective value was zero (no children and no overtime). No improvements were found on the initial solution for any of the cases. Running each case for one hour failed to provide any better results, except for the instant d5b, for which a value of 4.2757 was found after

about 10 minutes. Note that the initial solution values are not very far from the lower bounds, except for the case d10d, where the initial solution contained some un-scheduled projects.

None of the "admission" instances (Table 3) were solved to proven optimality and for only one of them were improvements found over the heuristic starting solution. Again, the lower bounds are reasonably good.

**Table 3 Results for the "admission" class**

Case	#var	#constr	Initial	Result	Bound	Case	#var	#constr	Initial	Result	Bound
WOT_1	52280	217150	45.6763	45.6763	35.8683	WOT_6	72264	275121	48.9564	48.9564	34.2245
WOT_2	59916	238259	46.8723	46.8723	34.3954	WOT_7	76049	289227	50.2774	50.2774	35.1558
WOT_3	61095	242054	48.1108	47.3801	35.7004	WOT_8	82463	314303	50.3108	50.3108	34.3746
WOT_4	67806	256402	50.4721	50.4721	37.0322	WOT_9	93086	337288	51.3888	51.3888	34.4294
WOT_5	67916	260364	49.9758	49.9758	36.7373	WOT_10	98775	359556	52.8344	52.8344	34.7907

Similar observations can be made for the "weekly" problems. It is evident from Table 4 ("Full problem") that for realistically sized cases (w32\*), CPLEX cannot improve the initial solution within the allocated time, or prove optimality. Even using a one hour time limit did not yield any improvements. The only improvements that were found over the given start solutions were for the instance w16b, and for the completely unrealistic 8-project instances. For these, the improved solutions were found and proven optimal in 0.11 – 1.22 seconds. Note from Table 4, however, that again our formulation provides fairly good lower bounds ("Bound" for the full problem).

**Table 4: Results for the "Weekly" problem class, for the full problem, and for sub problems with fixed modes as well as with both fixed modes and fixed resource periods.**

Case	#var	#constr	Initial	Full problem			Fixed mode		Fixed mode & RP	
				Result	T(s)	Bound	Result	T(s)	Result	T(s)
w32a	33396	100381	3.1044	3.1044	-	2.1881	3.1044	-	3.0810	78.23
w32b	34659	101908	10890.8376	10890.8376	-	2.1524	10890.8376	-	10890.8376	-
w32c	33672	100741	13329.1240	13329.1240	-	1.6816	13329.1240	-	13329.1240	-
w32d	32889	99821	2.6379	2.6379	-	1.7059	2.5200	527.33	2.5761	390.20
w32e	33780	100895	2.4877	2.4877	-	1.5895	2.4606	-	2.4808	55.82
w24a	19749	57926	4.9931	4.9931	-	3.7798	4.6662	13.34	4.6662	0.72
w24b	19697	57665	4.5797	4.5797	-	4.5619	4.5753	82.57	4.5782	14.85
w24c	19632	57731	3.6967	3.6967	-	3.1800	3.6793	34.63	3.6793	2.06

w16a	10083	27308	3.6851	3.6851	-	3.4576	<b>3.5422</b>	7.6	<b>3.5422</b>	0.64
w16b	8939	25891	3.0673	<b>3.0673</b>	1.22	3.0673	<b>3.0673</b>	0.01	<b>3.0673</b>	0.03
w16c	8636	25559	3.2711	<b>3.2641</b>	-	3.2507	<b>3.2711</b>	0.02	<b>3.2711</b>	0.02
w8a	2513	6915	1.9252	<b>1.9252</b>	0.11	1.9252	<b>1.9252</b>	0.02	<b>1.9252</b>	0.00
w8b	2648	7078	0.9629	<b>0.9629</b>	0.39	0.9629	<b>0.9629</b>	0.00	<b>0.9629</b>	0.00
w8c	2771	7184	0.4328	<b>0.4328</b>	0.14	0.4328	<b>0.4328</b>	0.02	<b>0.4328</b>	0.02

In certain planning situations, or in the context of a hybrid combination of algorithms, it may be interesting to solve specific sub-problems. We therefore investigated how CPLEX performed on the single-mode version of each instance, by fixing the mode to that given in the starting solution. Table 4, in the columns labelled "Fixed mode", illustrates the results for the "Weekly" instances. As expected, this simpler problem is easier to solve, and optimal solutions are found for all of the smaller instances. We also tried to fix both mode and the choice of resource period for each activity ("Fixed mode & RP" in Table 4). In that case, optimality was proven for all instances, except for w32b and w32c. Notably, these were the only instances for which the input start solution contained some un-scheduled projects. These results demonstrate that CPLEX can be used to solve such small sub problems, possibly in a heuristic way, if a reasonably good start solution is provided. As such, solving our MILP formulation could be usefully included in a hybrid combination of collaborating solvers for the SSP. However, it is apparent from Table 4 that our MILP formulation alone will not be competitive with heuristic methods for the full multi-modal problem.

## 6 Conclusion

The great diversity of real life surgery scheduling problems (SSP) implies a need for generalisation. A unified SSP model would greatly facilitate the development of algorithms that are robust across problem variations, and thus applicable in commercial planning software.

To this end, we have presented a general project scheduling model as a rich extension of the classical resource constrained project scheduling problem (RCPSp). We have discussed the involved extensions in detail, with particular attention to some new extensions that have not been previously used in the literature. These are mainly: resource periods, project disjunctions, inter-mode compatibility constraints and mode-dependent precedence constraints.

The proposed model is based on a generalisation of disjunctive graphs, for which we have introduced the concept of "disjunctive sets" as a basis for several types of problem constraints.

While the proposed model can express a range of complicated scheduling problems, we have focused our attention on the SSP, and have explained how real world SSPs can be expressed using this general modelling framework.

Having thus demonstrated the generality and expression power of the model, we wanted to investigate the practicality of solving it. Although the graph based model can be expected to be a suitable basis for a range of exact and heuristic methods (Artigues et al. 2008), we chose to this end to formulate the problem as a mixed integer linear program (MILP). This formulation also provides a precise mathematical description of the generalised SSP. Three different classes of realistic surgery scheduling problems were presented, and computational experiments were performed using CPLEX. The results show that realistically sized problem instances could not be solved within a practical time frame. This, and the fact that even the much simpler classical RCPSp is NP-hard in the strong sense, suggests that meta-heuristic approaches will be more promising for realistic problem instances. At the time of writing, we are therefore developing a meta-heuristic solution approach based on the presented modelling framework. The aim is to provide a high performance heuristic solver that is robust across most real world SSP applications. This does not mean, however, that the proposed MILP formulation is not useful. Our results show that the formulation, despite its generality, produces good bounds for most test instances. Also, promising solutions were found for interesting sub problems. It is therefore probable that our MILP formulation can be applied as a part of a hybrid

framework of collaborative solvers, together with meta-heuristic methods, to improve solutions and/or produce bounds for sub problems. Observe also that our test results were produced without any strengthening of the formulation with valid inequalities, or any use of additional algorithmic mechanisms such as column generation. Further improvements along these lines are therefore also an interesting direction for further research.

Finally, note that the proposed model is aimed at generality, rather than performance for any specific SSP. One can therefore expect that MILP-formulations with better performance may be found for selected problem variants.

### Acknowledgements

We wish to thank Professor Edmund Burke and Dr. Geir Hasle for interesting discussions and valuable input. This work is supported by the Research Council of Norway, through the HOSPITAL project.

## 7 References

- Artigues, C., Demasse, S., & Néron, E. (Eds.). (2008). *Resource-constrained Project Scheduling: Models, Algorithms, Extensions and Applications* (Control Systems, Robotics and Manufacturing). London, UK: ISTE.
- Artigues, C., Michelon, P., & Reusser, S. (2003). Insertion techniques for static and dynamic resource-constrained project scheduling. *European Journal of Operational Research*, 149(2), 249-267.
- Bartels, J. H., & Zimmermann, J. (2009). Scheduling tests in automotive R&D projects. *European Journal of Operational Research*, 193(3), 805-819.
- Blake, J. T., & Carter, M. (1997). Surgical process scheduling: a structured review. *J Soc Health Syst*, 5(3), 17-30.
- Blake, J. T., & Carter, M. W. (2002). A goal programming approach to strategic resource allocation in acute care hospitals. *European Journal of Operational Research*, 140(3), 541-561.
- Blazewicz, J., Lenstra, J. K., & Kan, A. H. G. R. (1983). Scheduling subject to resource constraints: classification and complexity. *Discrete Applied Mathematics*, 5(1), 11-24.
- Bomsdorf, F., & Derigs, U. (2008). A model, heuristic procedure and decision support system for solving the movie shoot scheduling problem. *OR Spectrum*, 30(4), 751-772.
- 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(1), 3-41.
- Cardoen, B., Demeulemeester, E., & Beliën, J. (2010). Operating room planning and scheduling: A literature review. *European Journal of Operational Research*, 201(3), 921-932.
- Charnetski, J. R. (1984). Scheduling Operating Room Surgical Procedures With Early and Late Completion Penalty Costs. *Journal of Operations Management*, 5(1), 91-102.
- Chen, Y.-L., Rinks, D., & Tang, K. (1997). Critical path in an activity network with time constraints. *European Journal of Operational Research*, 100(1), 122-133.
- Denton, B., Viapiano, J., & Vogl, A. (2007). Optimization of surgery sequencing and scheduling decisions under uncertainty. [10.1007/s10729-006-9005-4]. *Health Care Management Science*, 10(1), 13-24.
- Drexl, A., Nissen, R., Patterson, J. H., & Salewski, F. (2000). ProGen/[pi]x - An instance generator for resource-constrained project scheduling problems with partially renewable resources and further extensions. *European Journal of Operational Research*, 125(1), 59-72.
- Elmaghraby, S. E. (1977). *Activity networks: Project planning and control by network models* (Network analysis (Planning)). New York: Wiley.

- Gröflin, H., & Klinkert, A. (2007). Feasible insertions in job shop scheduling, short cycles and stable sets. *European Journal of Operational Research*, 177(2), 763-785, doi:10.1016/j.ejor.2005.12.025.
- Hans, E., Wullink, G., van Houdenhoven, M., & Kazemier, G. (2008). Robust surgery loading. *European Journal of Operational Research*, 185(3), 1038-1050.
- 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), 1-14.
- Icmeli, O., & Rom, W. O. (1996). Solving the resource constrained project scheduling problem with optimization subroutine library. *Computers & Operations Research*, 23(8), 801-817.
- Jebali, A., Hadj Alouane, A. B., & Ladet, P. (2006). Operating rooms scheduling. *International Journal of Production Economics Control and Management of Productive Systems*, 99(1-2), 52-62.
- May, J. H., Spangler, W. E., Strum, D. P., & Vargas, L. G. (2011). The Surgical Scheduling Problem: Current Research and Future Opportunities. *Production and Operations Management*, 20(3), 392-405, doi:10.1111/j.1937-5956.2011.01221.x.
- Mika, M., Waligóra, G., & Weglarz, J. (2008). Tabu search for multi-mode resource-constrained project scheduling with schedule-dependent setup times. *European Journal of Operational Research*, 187(3), 1238-1250.
- Nonobe, K., & Ibaraki, T. (2002). Formulation and tabu search algorithm for the resource constrained project scheduling problem. In C. C. Ribeiro, & P. Hansen (Eds.), *Essays and Surveys in Metaheuristics* (pp. 557-588): Kluwer Academic Publishers.
- Pritsker, A. A. B., Watters, L. J., & Wolfe, P. M. (1969). Multiproject Scheduling with Limited Resources: A Zero-One Programming Approach. [Theory Series]. *Management Science*, 16(1), 93-108.
- Riise, A., & Burke, E. (2011). Local search for the surgery admission planning problem. *Journal of Heuristics*, 17(4), 389-414, doi:10.1007/s10732-010-9139-x.
- Roy, B., & Sussmann, B. Les problèmes d'ordonnancement avec contraintes disjonctives. In *Note DS 9 bis, SEMA, Montrouge, 1964*
- Sprecher, A., Kolisch, R., & Drexl, A. (1995). Semi-active, active, and non-delay schedules for the resource-constrained project scheduling problem. *European Journal of Operational Research*, 80(1), 94-102, doi:10.1016/0377-2217(93)e0294-8.
- Vanhoucke, M., Gavrilova, M., Gervasi, O., Kumar, V., Tan, C., Taniar, D., et al. (2006). Scheduling an R&D Project with Quality-Dependent Time Slots. In *Computational Science and Its Applications - ICCSA 2006* (Vol. 3982, pp. 621-630, Lecture Notes in Computer Science): Springer Berlin / Heidelberg.



## A Objective Components

We write the objective function of our model as a linear combination of independent objective components, and assume that it is to be minimized:

$$O = \sum_l w_l O^l \quad (10.1)$$

In this appendix we give a comprehensive mathematical description of each objective component in the model, using the notation in Table 1 (page 15). We also explain how each component can be used in surgery scheduling. We would like the objective weights ( $w_l$ ) in (10.1) to intuitively reflect the relative importance of the objective components, as these weights are typically set by the user of some decision support system in which the model will be used. We therefore require that the objective components,  $O^l$ , are scaled to approximately the same order of magnitude.

We assume that choosing the earliest possible start time is always optimal for a given solution graph. In other words, we assume that the objective function is *regular* in the sense defined in (Sprecher et al. 1995). This is normally the case, but does not hold for all components individually. For example, for the soft time window components, (10.4) or (10.5), some later activity start time may give a lower value. For the SSP, however, surgery planners typically prefer to plan all activities as early as possible in the day, to keep any spare time towards the end of the day in case of unforeseen delays or arrival of emergency care patients. This can be easily modelled by adding an overtime objective on some critical resources, as e.g. the operating rooms, with a suitable preferred end time. This component will normally ensure that the total objective function is regular per resource period, and thus regular overall for any feasible solution.

The relevant objective components can be classified according to the basis for their evaluation; activity start times, resource use, or project completion time.

### A.1 Activity start times

Some objectives can be calculated based on the time of day of activity start times. Consider first the "preferred time of day" objective:

$$O_T = \frac{1}{\mathfrak{z} |\mathcal{N}^T|} \sum_{p \in \mathcal{P}} (1 - h^p) \left( \sum_{i \in \mathcal{N}^p \cap \mathcal{N}^T} w^i \Gamma(i) \right) \quad (10.2)$$

$$\Gamma(i) = |\text{mod}(t_i, \mathfrak{z}) - \tau_i| \quad (10.3)$$

Here,  $\tau_i$  is the target time of day for activity  $i$  and  $\mathfrak{z}$  is the number of time units in a day.  $\Gamma(i)$  is the absolute value of the difference between the planned time of day associated with the start time  $t_i$  of activity  $i$ , and  $\tau_i$ .  $\mathcal{N}^T \subseteq \mathcal{N}$  is the set of activities that are involved in the objective, and relative importance is weighted with  $0 < w^i \leq 1$ . The variable  $h^p = 0$  if the project is scheduled and 1 otherwise.

For the SSP, (10.2) may be used to express a need to operate on certain patients on preferred times of the day. For example it is preferred to operate on children, patients with diabetes, outpatients<sup>7</sup>, or patients with high demand for post-surgical care as early as possible in the day. A useful time resolution for the SSP is minutes, which means that  $\mathfrak{z} = 1440$  in the equation. Only surgery activities are included in  $\mathcal{N}^T$ , and the weights  $w^i$  are set according to the patient's age, and/or medical condition.

We express the degree of violations of soft time windows (Vanhoucke et al. 2006), as:

$$O_{TW} = \frac{1}{|\mathcal{N}|} \sum_{p \in \mathcal{P}} (1 - h^p) \sum_{i \in \mathcal{N}^p} \frac{w^i \Delta_i}{\eta_i - \zeta_i} \quad (10.4)$$

where  $[\zeta_i, \eta_i]$  is the soft time window of activity  $i$ , and  $\Delta_i = \text{Max}(y_i - \eta_i, \zeta_i - t_i, 0)$ . Similarly, a time window violation count objective can then be written as:

<sup>7</sup> In some hospitals, some outpatients may be included in the surgery schedule for inpatients for various reasons. If so, they should be treated early in the day so that they can return home on the same day.

$$O_{TWC} = \frac{1}{|\mathcal{N}|} \sum_{p \in \mathcal{P}} (1 - h^p) \sum_{i \in \mathcal{N}^p} w^i q_i \quad (10.5)$$

$$q_i = \begin{cases} 1; & \text{if } \Delta_i > 0 \\ 0; & \text{otherwise} \end{cases} \quad (10.6)$$

In both (10.4) and (10.5), the weights  $0 \leq w^i \leq 1$  express the relative importance between activities. In admission planning, (10.4) or (10.5) may be used to express the reluctance to move a surgery from the already planned date to another, or the preference of a certain day or week in which the patients want to be admitted. Narrow soft time windows may also be used to minimize the amount of schedule disruptions in a dynamic SSP problem, such as planning for the next day.

## A.2 Use of resources

The problem definition may include preferences concerning resource assignments. We model this by assigning a penalty  $v_i^r$  to all non-desired resource assignments:

$$O_P = \frac{1}{\mathcal{N}|\mathcal{R}|} \sum_{i \in \mathcal{N}} w^i \sum_{r \in \mathcal{R}} v_i^r g_i^r \quad (10.7)$$

In the SSP, (10.7) may be used to model a strong preference—but still not hard constraint—on who the main surgeon should be for each intervention.

From the point of view of the resource, some activities may be more attractive than others:

$$O_V = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \frac{w_r}{\lambda^r} \text{Max} \left( 0, \lambda^r - \sum_{i \in \Lambda^r} g_i^r \right) \quad (10.8)$$

Here,  $\lambda^r$  is the desired number of activities from the set  $\Lambda^r$ , for resource  $r$ . For resources without any such upper limit, we use  $\lambda^r = |\Lambda^r|$ .  $w_r \in [0,1]$  expresses relative importance between resources. In the SSP, (10.8) expresses the consideration that a surgeon in training requires a minimum number of surgeries of a given type during the planning period.  $\Lambda^r$  is then the set of relevant surgeries for surgeon  $r$ , and  $\lambda^r$  is the minimum required number of these.

Resource overtime can be calculated based on the violation of (soft) resource period end time limits. Typically, one resource period is defined per working day. The latest completion time  $\epsilon_k$  for any activity scheduled in resource period  $k$  can be extracted directly from the graph as the earliest start time of the resource period's end node. Let  $\zeta^k$  be the preferred end time of period  $k$ , and let resource period's "overtime" be defined as:

$$B_k = \begin{cases} \epsilon_k - \zeta^k, & \text{if } \epsilon_k > \zeta^k \\ 0, & \text{otherwise} \end{cases}; \quad \forall r \in \mathcal{R}, k \in \mathcal{K}^r \quad (10.9)$$

We can then write the *resource overtime objective* as

$$O_R = \frac{1}{|\mathcal{R}^o|} \sum_{r \in \mathcal{R}^o} \frac{w_r}{|\mathcal{K}^r|} \sum_{k \in \mathcal{K}^r} \frac{B_k}{(\sigma^k - \varphi^k)} \quad (10.10)$$

Here,  $\mathcal{R}^o \subseteq \mathcal{R}$  is the set of resources for which the objective is defined, and the  $w_r$  reflect the relative importance of each resource  $r \in \mathcal{R}^o$ . In the SSP, (10.10) is used to model staff overtime; usually for the surgeons.

We express a resource utilisation objective as the time that the resource is used in each resource period, relative to the maximum period duration  $(\sigma^k - \varphi^k)$ :

$$O_U = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} w_r \sum_{k \in \mathcal{K}^r} \frac{\sum_{i \in \mathcal{N}} q_i^k \sum_{m \in \mathcal{M}^i} x_{im} \vartheta_i^m}{(\sigma^k - \varphi^k)} \quad (10.11)$$

Again,  $w_r \geq 0$  expresses the relative importance between resources. Some authors use an “Undertime” measure for the SSP (Cardoen et al. 2010), but we take the position that under-utilisation of resources is better measured by a specific measure of resource utilisation, in the form of (10.11). When applied to surgeons, minimizing this utilisation measure is sometimes called minimization of surgeons waiting time.

### A.3 Project completion

The earliest possible project completion times<sup>8</sup> can be used to calculate several common objective components. One of these is the *relative waiting time*, by which we mean the number of days between the project reference date and the project completion time, relative to the number of days between the reference and due dates:

$$O_W = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \left( \frac{C_p - \chi_p}{\omega_p - \chi_p} \right)^2 \quad (10.12)$$

Here  $C_p = \lceil \max_{i \in N^p}(y_i) / \mathfrak{z} \rceil$  is the completion date of project  $p$ , given in days. Note that if a project is completed past its due date, the corresponding project’s contribution to the sum above will be larger than one. The power two means that it is better to have a well distributed waiting time, than to assign all the waiting time to only a few projects. Equation (10.12) can be used to model one of the most objectives in long term surgery scheduling; the minimisation of the time that a patient has to wait for surgery. The reference date  $\chi_p$  is set to patient  $p$ ’s referral date, and the due date  $\omega_p$  is set to the guaranty date for the patient’s surgery. Equation (10.12) promotes a fair distribution of waiting time among patients, since  $(\omega_p - \chi_p)$  is strongly related to the urgency of the surgery. The waiting time objective correlates with patient throughput measures (Cardoen et al. 2010), which can be based on the number of patients that will have an earliest start time of their “Project Finish” nodes within a given time interval.

One may also want to formulate a separate objective that states the degree of violation of due dates, such as the weighted project tardiness objective:

$$O_L = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \frac{w_p}{\omega_p} \text{Max}(0, C_p - \omega_p) \quad (10.13)$$

Here,  $w_p$  expresses the relative importance between projects. Another correlated objective is the classical makespan objective, but taken over the totality of all projects:

$$O_M = \frac{\mathfrak{z}}{H} \max_{p \in \mathcal{P}} C_p \quad (10.14)$$

In admission planning, (10.14) can express the goal of reducing the queues of elective patient for each specialty. This is obviously correlated with the waiting times for patients, as well as to resource utilisation objectives.

### A.4 Un-scheduled Projects

In many variations of the SSP, one considers a limited planning horizon. This means that a feasible solution may include a set of projects that are left un-scheduled, as discussed in section 2.3. In order to force as many projects as possible to be scheduled, the model contains an “Un-scheduled projects” objective:

$$O_{USP} = \frac{1}{|\mathcal{P}|} \sum_p w_p h^p \quad (10.15)$$

The weights  $w_p$  express the relative importance of scheduling each project.

<sup>8</sup> i.e. when all activities of a project are completed.



Technology for a better society  
[www.sintef.no](http://www.sintef.no)