Constraint-Based Modeling and Scheduling of Clinical Pathways

Armin Wolf

Fraunhofer FIRST, Kekuléstr. 7, D-12489 Berlin, Germany
armin.wolf@first.fraunhofer.de

Abstract. In this article a constraint-based modeling of clinical pathways, in particular of surgical pathways, is introduced and used for an optimized scheduling of their tasks. The addressed optimization criteria are based on practical experiences in the area of Constraint Programming applications in medical work flow management. Objective functions having empirical evidence for their adequacy in the considered use cases are formally presented. It is shown how they are respected while scheduling clinical pathways.

1 Introduction

Increasing cost in health care and payment on the basis of case-based lump sums (Diagnosis Related Groups – DRG) requires a more efficient, reliable and smooth treatment of patients. One popular approach is the standardization of treatments on the basis of medical work flows – so called *clinical pathways*.

The aim of this paper is the presentation of constraint-based modeling approaches and scheduling techniques for clinical pathways. Clinical pathways are predefined work flows similar to those in traditional (i.e. industrial) scheduling thus suitable for Constraint Programming techniques [3]. In this paper activities in clinical processes, their required resources, their temporal relationships are considered as well as the constraints resulting from the clinical infrastructure and hospital organization. The general structure of clinical pathways and their basics are completed by specialized approaches addressing in particular surgeries. The reason is that surgeries play a central role for the economic situation of a hospital. Well-organized and resource efficient surgical treatment of patients increases work quality and patient satisfaction while decreasing costs and thus the risk of financial loss.

In health care there are particular scheduling approaches for nurse and physician rostering [7,16,17], appointment scheduling [10] surgery scheduling [5] – sometimes with integrated rostering. However, a holistic approach for clinical pathways seems to be missing.

2 Modeling Clinical Pathways

A *clinical pathway* is a standardized medical work flow. It consists of a sequence of *medical activities* like diagnosis, preparation, treatment (e.g. surgery), care,

J. Larrosa and B. O'Sullivan (Eds.): CSCLP 2009, LNAI 6384, pp. 122–138, 2011.

training (e.g. physiotherapy) etc. In general, a medical activity has to be processed within an earliest start time and a latest completion time and within a minimal and maximal duration. Furthermore, a medical activity in general requires several kinds of resources to be performed like nurses, doctors (e.g. surgeons, anesthetists), devices (e.g. MRT), rooms, instruments, consumables etc.

2.1 Tasks

In the context of finite-domain Constraint Programming a medical activity will be represented by at least one *task* formally specified by the following definition:

Definition 1 (Task). *A* task *is a non-preemptive activity with a start time, a duration and an end time. Due to the fact that these times are in general restricted but a-priori undetermined they will be represented by finite-domain variables: Each task t has*

- *a* start time variable $t.\mathsf{start}$ *restricted by* $t.\mathsf{start} \in S_t$ *where* S_t *is a finite set of non-negative integer values.*[1]
- *a duration variable* $t.\mathsf{duration}$ *restricted by* $t.\mathsf{duration} \in D_t$ *where* D_t *is a finite set of non-negative integer values.*
- *an* end time variable $t.\mathsf{end}$ *restricted by* $t.\mathsf{end} \in E_t$ *where* E_t *is a finite set of integer values.*

For each task t the constraint $t.\mathsf{start} + t.\mathsf{duration} = t.\mathsf{end}$ *has to be satisfied.*

Furthermore, a task requires some capacity from a resource out of set of resources; thus each task t has

- *a* capacity variable $t.\mathsf{capacity}$ *restricted by* $t.\mathsf{capacity} \in C_t$ *where* C_t *is a finite set of non-negative integer values.*
- *a* resource identifier variable $t.\mathsf{resourceId}$ *restricted by* $t.\mathsf{resourceId} \in R_t$ *where* R_t *is a finite set of resource identifiers (cf. Section 2.2 below).*

In some cases tasks are *optional*, i.e. they are not necessarily performed. This can be modeled either with a possible duration of zero, i.e. $\mathsf{duration} \in \{0, p, \ldots, q\}$ where $0 < p \leq q$ holds or with an additional Boolean flag isOptional deciding whether the corresponding task is optional (1 [true]) or mandatory (0 [false]).

2.2 Resources

Constraint-based scheduling knows several kinds of resources. In the context of clinical pathways a task requires in general either

- an *exclusive resource*: it performs at most one activity at the same time, e.g. a surgeon, an operating room, a clinical device etc.
- or an *alternative exclusive resource*: it offers an assortment of exclusive resources, however, one has to be chosen, e.g. similar operating rooms, a collections of the same medical instruments etc.

[1] Object-oriented notation is used within this article.

- or a *cumulative resource*: it offers a limited quantity of its capacity, i.e. other tasks can be performed concurrently, e.g. a pool of nurses, consumables, power etc.
- or an *alternative cumulative resource*: it offers an assortment of cumulative resources, however, one has to be chosen, e.g. storages of consumables, pools of nurses etc.

It is assumed that each resource r has a unique integral identifier id_r, i.e. a "number" such that there is a bijection mapping resources to identifiers and vice versa.

In constraint-based scheduling these resources are represented by adequate constraints. If there are cumulative resources the corresponding constraints that has to be satisfied ar defined accordingly:

Definition 2 (Alternative Cumulative Constraint). *Let a set of tasks T and a set of cumulative resources R be given. Each resource $r \in R$ has a unique identifier id_r and an integral capacity $C_r > 0$. Then, the* alternative cumulative constraint *is satisfied for R, T and $T_r = \{t \in T \mid t.\mathsf{resourceId} = id_r\}$ for any $r \in R$ if the following inequality holds:*

$$\forall i \in [\min_{t \in T} S_t, \max_{t \in T} E_t - 1] \, \forall r \in R \, \forall t \in T_r \mid \sum_{t.\mathsf{start} \leq i < t.\mathsf{end}} t.\mathsf{capacity} \leq C_r.$$

It has to be pointed out, that this definition is a special case of the *cumulatives* constraints presented in [4] but generalizes the *cumulative* constraint originally introduced in [1]:

Definition 3 (Cumulative Constraint). *Let a set of tasks T and a cumulative resource r with integral capacity $C_r > 0$ be given such that $R_t = \{r\}$ holds for each task $t \in T$. Then, the* cumulative constraint *is satisfied for T and r if and only if*

$$\forall i \in [\min_{t \in T} S_t, \max_{t \in T} E_t - 1] \, \forall t \in T \mid \sum_{t.\mathsf{start} \leq i < t.\mathsf{end}} t.\mathsf{capacity} \leq C_r$$

holds.

In general, pruning techniques for cumulative constraints, in particular for the *cumulatives* constraint (cf. [4]) support tasks with possibly zero duration and thus optional tasks.

In the special case where consumables have to be represented, cumulative resources can be used for an adequate modeling (cf. [19]). However, there are alternative approaches using specialized algorithms considering consumers as well as producers of consumables to be stored in so-called *reservoirs* or *inventories* with restricted capacities [14,15].

Alternative exclusive resource constraints are special cases of alternative cumulative constraints and exclusive resource constraints are special cases of cumulative constraints. The restriction is that the capacities of all resources is one:

$C_r = 1$ for each considered resource r. Obviously the capacity requirements of all considered tasks must be one, too. Otherwise, the task will neither require the resource (its capacity is zero) nor the constraint will be satisfied because the capacity of the task is greater than one.

However, other definitions are introduced for exclusive constraints in order to highlight the differences with respect to cumulative constraints, in particular there are specialized algorithms to handle these constraints (cf. [3]).

Definition 4 (Alternative Exclusive Resource Constraint). *Let a set of tasks T and a set of exclusive resources R be given. Each task $t \in T$ requires an unary capacity, i.e. $t.\mathsf{capacity} = 1$ and each resource $r \in R$ has an unique identifier id_r and an unary capacity $C_r = 1$, too. Then, the* alternative exclusive resource constraint *holds for T and R if*

$$\forall s \in T\, \forall t \in T \setminus \{s\}\, \forall r \in R \mid$$
$$(s.\mathsf{resourceId} = r \wedge t.\mathsf{resourceId} = r) \implies (s.\mathsf{end} \leq t.\mathsf{start} \vee t.\mathsf{end} \leq s.\mathsf{start})$$

is satisfied.

This definition generalizes the definition of exclusive resource constraints:[2]

Definition 5 (Exclusive Resource Constraint). *Let a set of tasks T and an exclusive resource R be given such that $t.\mathsf{capacity} = 1$ and $R_t = \{r\}$ holds for each task $t \in T$ and the resource r has an unique identifier id_r and $C_r = 1$ holds. Then, the* exclusive resource constraint *holds for T and r if*

$$\forall s \in T\, \forall t \in T \setminus \{s\} \mid s.\mathsf{end} \leq t.\mathsf{start} \vee t.\mathsf{end} \leq s.\mathsf{start}$$

is satisfied.

In general, pruning techniques for exclusive resource constraints support tasks with possibly zero duration and thus optional tasks. In particular, there are specialized pruning algorithms for optional tasks on exclusive resources [21]. All these algorithms are adopted for alternative exclusive resource constraints (cf. [13,25]).

If a medical activity requires several resources this will be modeled by several tasks – one task for each required (alternative) resource. All these tasks will have identical start times, durations and end times.

Example 1. Within the clinical pathway for a surgery the actual operation *op* requires an operating room, a surgeon, an anesthetist and some nurses. There are three operating rooms or_1, or_2, or_3, two qualified surgeons su_1, su_2 and four anesthetists $an_1, \ldots, an_4$ available. Two of the five available nurses are required. Obviously the operating room the surgeon and the anesthetist are exclusive resources because they will not treat any other patient while operating. The available nurses are considered as one cumulative resource pool *np*

[2] Exclusive resources are also called "single" resources or "one-machine" resources.

offering a capacity of 5 (nurses). Thus this multi-resource activity op will be modeled by four task $op_1, \ldots, op_4$ with $op_1.\mathsf{start} \equiv \cdots \equiv op_4.\mathsf{start}$, $S_{op_1} \equiv \cdots \equiv S_{op_4}$, $op_1.\mathsf{duration} \equiv \cdots \equiv op_4.\mathsf{duration}$, $D_{op_1} \equiv \cdots \equiv D_{op_4}$, $op_1.\mathsf{end} \equiv \cdots \equiv op_4.\mathsf{end}$, and $E_{op_1} \equiv \cdots \equiv E_{op_4}$. Furthermore, $op_1.\mathsf{capacity} \in \{1\}$, $op_1.\mathsf{resourceId} \in \{id_{or_1}, id_{or_2}, id_{or_3}\}$, $op_2.\mathsf{capacity} \in \{1\}$, $op_2.\mathsf{resourceId} \in \{id_{su_1}, id_{su_2}\}$, $op_3.\mathsf{capacity} \in \{1\}$, $op_3.\mathsf{resourceId} \in \{id_{an_1}, \ldots, id_{an_4}\}$, and $op_4.\mathsf{capacity} \in \{2\}$, $op_4.\mathsf{resourceId} \in \{id_{np}\}$.

In Example 1 all considered resources are independent from each other, thus their identifiers are pairwise different by definition. For a practical implementation of this model in a Constraint Programming system, however, resources are identified by their corresponding constraints and an additional relative index ranging from 0 (or 1) to a maximal index in case of alternative constraints. Consequently, a bijection has to realized that maps identifiers to the according constraints and indexes in case of alternatives.

In clinical practice, often the situation arises that the resources in alternatives are also considered individually or alternative resources share some of their individuals. In both cases, it has to be ensured that all tasks competing for an individual resource in different contexts are related by the according constraints as demonstrated in the following example:

Example 2. Within the clinical pathway for a surgery the actual operation requires that the (a-priori known) leading surgeon is supported by any other (a-priori unknown) assisting surgeon out of a pool of assistants. In general, this pool of assistants consists of surgeons having also leading positions but never both at the same time.

Using the presented modeling approach it is easy to master situations with shared resources:

- use an "all-purpose" alternative cumulative constraint for all cumulative resources and
- use another "all-purpose" alternative exclusive resource constraint for all exclusive resources.

However, it is recommended to use as much as possible "special-purpose" constraints for alternative resources for highest flexibility, i.e. in order to adapt the used pruning algorithms individually or reduce their overall runtime (see below).

Thus we suggest to apply the following rules while they are applicable:

- for each individual resource occurring in alternatives use the same identifier and ignore the constraint for the individual resource.[3]
- for any pair of alternatives sharing individual resources use the same identifier for each shared individual and replace the constraints for the pair by one alternative resource constraint covering the individual resources of the pair.

[3] It is not really necessary to ignore the redundant constraint.

More formally, let $R_1, \ldots, R_n$ be the possible domains of any resource variables. Then, let $P = P_1 \uplus \ldots \uplus P_m$ be *the partition*[4] of $R_1 \cup \ldots \cup R_n$ such that

- if $R_i \cap P_k \neq \emptyset$ holds for any $i \in \{1, \ldots, n\}$ and any $k \in \{1, \ldots, m\}$ then $R_i \subseteq P_k$ holds, too,
- for each $k \in \{1, \ldots, m\}$ and each non-empty subset $P' \subsetneq P_k$ there is a R_i with $i \in \{1, \ldots, n\}$ such that $R_i \cap P_k \neq \emptyset$ holds, however,$R_i \not\subseteq P'$ holds.

A *Union-Find* algorithm – the classical algorithm was introduced by Tarjan [20] – will compute this partition efficiently. Therefore, we start with the singular sets $\{r_1\}, \ldots, \{r_k\}$ of all considered resources: $\{r_1, \ldots, r_k\} = R_1 \cup \ldots \cup R_n$. Then, for each pair of different resources r_p and r_q the sets containing them are united if there is a set R_i such that $r_p, r_q \subseteq R_i$ holds for $1 \leq i \leq n$ and $1 \leq p < q \leq k$.

Obviously, it holds

$$|T_1|^n + \cdots + |T_m|^n < (|T_1| + \cdots + |T_m|)^n$$

if $T_1, \ldots, T_m$ are the non-empty task sets to be scheduled independently on the resources in $P_1, \ldots, P_m$.[5] Assuming polynomial runtime of the pruning algorithms it means that the overall runtime of the pruning for the "special-purpose" alternative resource constraints is less than the runtime of the pruning for an "all-purpose" alternative resource constraint.

The application of these rules in the situation presented in Example 2 results in the following constraint model:

Example 3 (Continuation of Example 2). Let the clinical pathways for surgeries require an (a-priori known) leading surgeon and sometimes require any other (a-priori unknown) assisting surgeon. Then all potentially leading or assisting surgeons will be considered commonly in an alternative exclusive resource constraint.

2.3 Temporal Relationships

The activities of clinical pathways are in general temporally related. There are absolute/relative and intra-/inter-pathway relationships. In the absolute case tasks can take up to three different roles:

- *Successors*: each successor s of a task t starts after the end of t within a given offset. It holds $t.\mathsf{end} + \mathsf{offset}_{t,s} = s.\mathsf{start}$ where the delay $\mathsf{offset}_{t,s}$ is either an integer value or a finite-domain variable. Choosing the delay appropriately it is possible to model several situations, e.g.
  - $\mathsf{offset}_{t,s} \in [0, \mathsf{MAX\_HORIZON}]$: t must be finished before s will start, which is equivalent to $t.\mathsf{end} \leq s.\mathsf{start}$ if the value $\mathsf{MAX\_HORIZON}$ is sufficiently large.

[4] Such a partition is uniquely defined.
[5] N.B. $T_1, \ldots, T_m$ define a partition as well.

  - $\mathsf{offset}_{t,s} \in [-5, 5]$: t must be finished at most five time units before/after t will start.
- *Predecessors*: for each predecessor p of a task t it holds that t is a successor of p.
- *Concurrent activities*: for any two concurrent tasks c an d it holds $c.\mathsf{start} + \mathsf{startOff}_{c,d} = d.\mathsf{start}$ and/or $c.\mathsf{end} + \mathsf{endOff}_{c,d} = d.\mathsf{end}$ where the delays $\mathsf{startOff}_{c,d}$ and $\mathsf{endOff}_{c,d}$ are either integer values or finite-domain variables. Choosing the delays appropriately it is possible to model several situations:
  - $c.\mathsf{start} - 5 = d.\mathsf{start}$: c starts five time units later than d
  - $c.\mathsf{end} + \mathsf{endOff}_{c,d} = d.\mathsf{end}$ with $\mathsf{endOff}_{c,d} \in [0, \mathsf{MAX\_HORIZON}]$: c finishes not-after the end of d.

These relations are complete in the sense of Allen [2] because any temporal relationship between the start times and end times of any two a-priori known activities are specifiable. Alternatively, the temporal relationships of start times and end times are representable as a *Simple Temporal Problem* according to [8]:

Definition 6 (Temporal Constraints). *Let a set of start and end time variables E be given. Then the* temporal constraints *$T(E)$ on E are constituted by a set of of inequalities $e - e' \leq d$ where d is an integer constant and e, e' are in E.*

The situation becomes more complicated if the (temporal) relations between two medical activities in different clinical pathways depend on the chronological order of the "crucial" tasks of these pathways, e.g. the actual operations in the clinical pathway of two different surgeries. In practice, often the situation arises that a task x_a of a clinical pathway a is related to a task y_b of another clinical pathway b by a constraint $c_{a,b}$, e.g. $c_{a,b} \equiv x_a.\mathsf{end} + \mathsf{offset}_{x_a,y_b} = y_b.\mathsf{start}$, if the operating task op_a of pathway a is the direct predecessor of the operating task op_b of pathway b in the commonly used operating room. This means that the conditional constraint between x_a and y_b has to be satisfied only if the operation of a is scheduled directly before the operation of b.

The proposed constraint-based approach to handle this situation adequately uses a sequence dependent setup costs constraint:

Definition 7 (Sequence Dependent Setup Costs). *Let a sequence of tasks $T = t_1, \ldots, t_n$ with $n > 1$ be given. It is assumed that these tasks have to be scheduled in linear order, i.e. it holds*

$$t_i.\mathsf{end} \leq t_j.\mathsf{start} \vee t_j.\mathsf{end} \leq t_i.\mathsf{start} \quad \textit{for } 1 \leq i < j \leq n.$$

It is further assumed that for each pair of tasks (t_i, t_j) with $1 \leq i, j \leq n$ there is an non-negative integer cost value $\mathsf{cost}_{i,j}$.

Then, for each task t_j in the sequence T $(1 \leq j \leq n)$ the finite domain variable $\mathsf{setupCost}(t_j, T)$ defines the sequence dependent setup cost *of t_j (with respect to T) if this variable is constrained to*

$$\mathsf{setupCost}(t_j, T) = \begin{cases} \mathsf{cost}_{i,j} & \textit{if there is a task } t_i \textit{ with } 1 \leq i \leq n \textit{ such that} \\ & t_i.\mathsf{end} \leq t_j.\mathsf{start} \textit{ holds and for each task } t_k \\ & \textit{with } 1 \leq k \leq n, k \neq i, k \neq j \textit{ it holds} \\ & t_k.\mathsf{start} < t_i.\mathsf{end} \textit{ or } t_j.\mathsf{start} < t_k.\mathsf{end}, \\ 0 & \textit{otherwise.} \end{cases}$$

This means that the setup cost of a task t_j is determined by its direct predecessor t_i – if there is any: t_i is scheduled before t_j and there is not any other task scheduled between t_i and t_j.

Now, let $op_1, \ldots, op_n$ be the operations to be performed concurrently in the operating room of op_a and op_b, i.e. $1 \leq a, b \leq n$ holds. Further, let $\mathsf{costValue}_{i,j} = i$ for $1 \leq i, j \leq n$.

Then, it is possible to trigger the constraint $c_{a,b}$ conditionally using *reified constraints*, i.e. logical connections (implication, disjunction etc.) of possibly negated constraints:

$$(\mathsf{setupCost}(op_b) = a) \implies c_{a,b}.$$

A rather similar situation is given, if the preparation time for one task in a clinical pathway depends on another task in another pathway. This occurs, e.g. due to cleaning and autoclaving times resulting from a previously performed treatment. An adequate constraint-based model of this situation is usually based on *sequence dependent setup times constraints*. (cf. [9] for a survey and a definition and [24] for a modeling and pruning algorithms).

2.4 Infrastructural and Organizational Relationships

Among the presented treatment-specific relationships there are additional restrictions on the activities of clinical pathways. Some of them are of organizational nature while others are caused by the infrastructure of the medical institution:

- time slots for some activities, sometimes depending on the used resources, e.g. working or operating hours etc.
- (sequence dependent) setup or transfer times, e.g. for autoclaving surgical instruments or for the transportation of the patients from one location to another.
- the location of stations, devices, treatment rooms etc. and the connections between them, e.g. corridors, entrance and exit doors etc.

2.5 Relating Times and Locations

In general, for some tasks there are alternative time slots available in order to process these tasks. However, some of the alternative time slots are only available for some alternative resources and vice versa.

Example 4. On Tuesdays the orthopedic surgeries either has to be performed between 08:00 and 10:30 or between 16:00 and 20:30. However, in the morning there are only the operating rooms numbered 4 and 5 available for orthopedic surgeries and in the afternoon only the rooms with number 1 and 3.

In Constraint Programming it is suitable to model these dependencies between time slots and resources using so called *element* constraints.

Definition 8 (Element Constraint). *Let a sequence of integers values and/or finite-domain variables* $\mathsf{element}_0, \ldots, \mathsf{element}_{n-1}$ $(n > 0)$ *be given. Further, let* value *and* index *be two finite domain variables. Then, the* element constraint

$$\mathsf{value} = \langle \mathsf{element}_0, \ldots, \mathsf{element}_{n-1} \rangle [\mathsf{index}]$$

holds for these entities if $\mathsf{value} = \mathsf{element}_{\mathsf{index}}$ *is satisfied.*

Example 5 (Continuation of Example 4). Let $s_1, \ldots, s_k$ be the orthopedic surgery tasks to be scheduled next Tuesday according to the spatiotemporal relationships presented in Example 4. The time granularity for scheduling is 5 minutes per time unit. Thus the whole day (0:00 to 24:00) is represented by the interval [0, 288] and e.g. 8 o'clock in morning by 96. It is assumed that each surgery task is additionally defined by

- its variable slot start $\mathsf{slotStart} \in \{96, 192\}$,
- its variable slot end $\mathsf{slotEnd} \in \{126, 246\}$,
- an a.m./p.m. selector variable $\mathsf{ampm} \in \{0, 1\}$,
- a finite-domain variable for the operating rooms available during the morning slot $\mathsf{amOR} \in \{4, 5\}$,
- another finite-domain variable for the operating rooms available during the afternoon slot $\mathsf{pmOR} \in \{1, 3\}$.

Then, for $1 \leq i \leq n$ the constraints

$$\begin{aligned} s_i.\mathsf{slotStart} &= \langle 96, 192 \rangle [s_i.\mathsf{ampm}] \\ s_i.\mathsf{slotEnd} &= \langle 126, 246 \rangle [s.i.\mathsf{ampm}] \\ s_i.\mathsf{resourceId} &= \langle s_i.\mathsf{amORs}, s_i.\mathsf{pmORs} \rangle [s_i.\mathsf{ampm}] \\ s_i.\mathsf{slotStart} &\leq s_i.\mathsf{start} \\ s_i.\mathsf{end} &\leq s_i.\mathsf{slotEnd} \end{aligned}$$

model the spatiotemporal relationships presented in Example 4 adequately. Evidence of this modeling is given by examination of the two cases: $\mathsf{ampm} = 0$ and $\mathsf{ampm} = 1$.

In general, *element* constraints are very appropriate to relate times and/or locations, e.g. to assign the induction and recovery rooms to their according operating room, to assign the transportation times from the hospital wards to the treatment locations etc.

2.6 Room-Specific Relationships

In particular, spatiotemporal relationships in hospitals are sometimes very specific: In one practical application, the situation arises that a newly operated patient treated in one specific operating room has to pass the related induction room in order to end up in the related recovery room. For several reasons (infection risk, privacy etc.) it has to be ensured that there is no other patient in the induction room while moving the newly operated patient from the operating room into the recovery room. – The following constraint-based approach is proposed to consider this "design-specific" situation adequately:

For each surgery i there is a task for induction ind_i and another task for recovery rec_i in its clinical pathway. Obviously, the induction task is before the real surgery task s_i and the recovery task is afterwards:

$$ind_i.\mathsf{start} < ind_i.\mathsf{end} \leq s_i.\mathsf{start} < s_i.\mathsf{end} \leq rec_i.\mathsf{start}.$$

Then, for any two surgeries a and b potentially performed in this special operating room numbered id_{or}, i.e. $R_a \ni id_{or}, R_b \ni id_{or}$, the reified constraint

$$\begin{aligned} & s_a.\mathsf{resourceId} = id_{or} \wedge s_b.\mathsf{resourceId} = id_{or} \\ & \quad \Longrightarrow\ (rec_b.\mathsf{start} - ind_a.\mathsf{start}) \cdot (rec_a.\mathsf{start} - ind_b.\mathsf{start}) < 0 \end{aligned}$$

is stated. Alternatively, e.g. if the used Constraint Programming system does not support such reified constraints, the following approach

$$\begin{aligned} & (rec_b.\mathsf{start} - ind_a.\mathsf{start}) \cdot (rec_a.\mathsf{start} - ind_b.\mathsf{start}) \\ & \quad < ((s_a.\mathsf{resourceId} - id_{or})^2 + (s_b.\mathsf{resourceId} - id_{or})^2) \cdot \mathsf{BIG\_INTEGER}. \end{aligned}$$

is suitable, too, if the integer value BIG_INTEGER is sufficiently large. In the case that non-linear arithmetics are not supported, another alternative modeling is suggested using the *Kronecker* operator:

Definition 9 (Kronecker Operator). *Let an integer value* val *and a finite-domain variable* var *be given. Then, the* Kronecker *operator is defined by*

$$\delta_{\mathsf{val}}(\mathsf{var}) = \begin{cases} 1 & \textit{if } \mathsf{var} = \mathsf{val}, \\ 0 & \textit{otherwise.} \end{cases}$$

Using the Kronecker operator, the following simplified alternative

$$\begin{aligned} & (rec_b.\mathsf{start} - ind_a.\mathsf{start}) \cdot (rec_a.\mathsf{start} - ind_b.\mathsf{start}) \\ & \quad < (2 - \delta_{id_{or}}(s_a.\mathsf{resourceId}) - \delta_{id_{or}}(s_b.\mathsf{resourceId})) \cdot \mathsf{BIG\_INTEGER}. \end{aligned}$$

is also suitable, if the integer value BIG_INTEGER is sufficiently large.

All three approaches models the situation correctly. Evidence is given by the following considerations: If at least one of both surgery tasks is not performed in the considered operating room, then the implication is satisfied and the right-hand-sides of the inequalities become large enough such that they dominate

the left-hand-sides of the inequalities. Now, it is assumed that both surgeries are performed in the considered operating room. Then, the *condition* of the implication is satisfied and its *conclusion* is identical with both inequalities, because their left-hand-sides become zero.

Without loss of generality, it is further assumed that surgery a is performed before b.[6] Due to the order of the tasks in the clinical pathways of surgeries it holds $ind_a.\mathsf{start} < rec_b.\mathsf{start}$. It follows immediately that

$$rec_b.\mathsf{start} - ind_a.\mathsf{start} > 0 \text{ and thus } rec_a.\mathsf{start} - ind_b.\mathsf{start} < 0.$$

This means that the recovery of surgery a has already started (in the recovery room after leaving the induction room) before the induction of surgery b starts: $rec_a.\mathsf{start} < ind_b.\mathsf{start}$.

3 Scheduling Clinical Pathways

The challenge in constraint-based scheduling of clinical pathways is the determination of values for the start, duration, and end variables of all tasks in the pathways such that all all inter- and intra-pathway constraints (see Section 2) are satisfiable, e.g. there are values for the remaining variables such that all constraints are satisfied.

3.1 Optimization

Practical experiences show that optimized scheduling of clinical pathways is rather difficult. Beyond the intrinsic complexity of optimization, the reasons are that the criteria in a clinical context are manifold, informal, and sometimes contradicting each other. There is currently neither a generally accepted objective function nor an according scheduling strategy to be customized via appropriate parameter setting. Different contexts require specialized solutions. Some evidence is given by the following two cases:

Workload Balancing in Surgery Slots. Practical experiences in surgical pathway scheduling have shown that a balancing of the workload within the time slots of the actual surgery tasks is often favored.

The chosen scheduling approach uses the work load factor in percent: $\mathsf{workloadFactor} \in [0, 100]$. For a proper consideration of this objective the constraint-based model has to be extended: for each slot s_i with maximal slot duration msd_i (e.g. the difference between the latest slot end and the earliest slot start), with variable slot start $\mathsf{slotStart}_i$ and variable slot end $\mathsf{slotEnd}_i$ the constraint

$$\mathsf{slotEnd}_i \leq \mathsf{slotStart}_i + \frac{msd_i}{100} \cdot \mathsf{workloadFactor}$$

[6] The factors of the product on the left-hand-side are symmetric with respect to the surgeries.

has to be added. More weak criteria like "the surgery tasks should be – if possible – in the chronological order of their requests" following the principle: "first-come, first-served" are difficult to model. Even if penalties for offending this principle are not quantified. In such a case, it is suggested to use an appropriate heuristic search strategy instead of a (strong) branch-and-bound optimization.

This scheduling strategy distributes the surgery tasks over the available slots according to their time and location restrictions (cf. Section 2.5). The algorithm is based on the invariant that the already distributed surgery tasks are constrained to be in linear order in their slots and realizes a depth-first chronological backtracking search:

While there are not yet distributed surgery tasks

- sort the slots according to their work load – lowest load first.
- select the next task according to "first-come, first-served".
- while there is a next slot, try to insert the task into the linear order already established there without causing an inconsistency:
  - start with the last position (after all tasks) first.
  - finish either with success if such position is found or continue with the next slot otherwise.
- if scheduling fails, backtrack to the last recently scheduled task and try another position and/or slot. Otherwise continue with the next task.

This scheduling strategy is completed by a depth-first chronological backtracking labeling strategy. The labeling tries to determine the values of the necessary variables such that all remaining variables will be fixed, too. If the labeling fails, search backtracks to the task distribution until either another distribution of tasks is found or finitely fails because there is no admissible schedule.

A workload balancing distribution of tasks (without linear task orders) can be seen as a preprocessing for a more "local" optimization strategy. Such a strategy on individual operating slots and rooms is addressed in the following:

Optimized Sequencing of Surgeries. In one practical application the challenge is an optimized sequencing of the actual surgery tasks in surgical pathways. In this context the operating rooms and the time slots are already allocated to each surgery task (cf. Section 3.1). Furthermore, there are two operating tables in alternating use: while one table is currently used in the operating room the other is altered for the next surgery. Additionally, all surgeries have priority scores (e.g. high scores for surgeries with a high anesthetic risk) and they have to be performed without break: the next surgery starts directly after the current.

The optimization criteria are manifold:

- minimize the delays of the surgeries according to its scores, i.e. apply the principle: "the higher the score the earlier the surgery",
- reduce the costs for the preparations and alterations of the tables,
- reducing the delays of the surgeries is the primary optimization criterion while the cost reduction has secondary importance.

It is proposed to use a weighted sum to represent all these criteria adequately in an objective function. Therefore, for each surgery s let

- its *priority score* $s.\mathsf{score}$ be an integer value,
- its *bedding type* $s.\mathsf{type}$ be an integer value determining the sequence dependent preparation/alternation cost for this surgery.

For each bedding type t let $\mathsf{initalCost}_t$ be a non-negative integer value determining the cost for the initial preparation of an operating table for a surgery of type t. For each pair of bedding types s, t let $\mathsf{alterationCost}_{s,t}$ be a non-negative integer value determining the cost for the alteration of an operating table from type s to type t.

Thus, for the sequence of surgeries $S = s_1, \ldots, s_n, (n > 1)$ to be scheduled a permutation $\sigma : \{1, \ldots, n\} \to \{1, \ldots, n\}$ and a labeling of the start times $s_1.\mathsf{start}, \ldots, s_n.\mathsf{start}$ has to be found such that

$$s_{\sigma(i)}.\mathsf{start} = s_{\sigma(i-1)}.\mathsf{start} + s_{\sigma(i-1)}.\mathsf{duration}$$

is satisfied for $1 \leq i \leq n$ and the sum

$$\mathsf{initalCost}_{s_{\sigma(1)}.\mathsf{type}} + \mathsf{initialCost}_{s_{\sigma(2)}.\mathsf{type}} + \sum_{i=1}^{n-2} \mathsf{alterationCost}_{s_{\sigma(i)}.\mathsf{type}, s_{\sigma(i+2)}.\mathsf{type}} + \sum_{j=1}^{n} \alpha \cdot s_{\sigma(j)}.\mathsf{score} \cdot s_{\sigma(j)}.\mathsf{start}$$

is minimal. – Here, the value of the parameter α adjusts the importance of the delay of surgeries with respect to the costs: the larger the value the higher the importance.

Due to the fact that the order of the surgeries is a-priori unknown, i.e. it has to be determined during an optimized scheduling, appropriate constraints have to be used to model this objective adequately in Constraint Programming. First of all there is a specialized cost constraint required taking the alteration of the operation tables into account:

Definition 10 (Sequence Dependent Alternating Setup Costs). *Let a sequence of tasks $T = t_1, \ldots, t_n$ with $n > 1$ be given. It is assumed that these tasks have to be scheduled in linear order, i.e. it holds*

$$t_i.\mathsf{end} \leq t_j.\mathsf{start} \vee t_j.\mathsf{end} \leq t_i.\mathsf{start} \quad \textit{for } 1 \leq i < j \leq n.$$

It is further assumed that each task t_i with $1 \leq i \leq n$ has a specific type $t.\mathsf{type}$ (an integer value) and there is a non-negative integer cost value $\mathsf{initalCost}_{t.\mathsf{type}}$ and that for each pair of tasks (t_i, t_j) with $1 \leq i, j \leq n$ there is a non-negative integer cost value $\mathsf{alterationCost}_{t_i.\mathsf{type}, t_j.\mathsf{type}}$.

Then, for each task t_j in the sequence T $(1 \leq j \leq n)$ the finite domain variable $\mathsf{alternatingSetupCost}(t_j, T)$ defines the sequence dependent alteration setup cost *of t_j (with respect to T) if this variable is constrained to*

$$\mathsf{alternatingSetupCost}(t_j, T)$$
$$= \begin{cases} \mathsf{alterationCost}_{t_h.\mathsf{type}, t_j.\mathsf{type}} & \textit{if there are two tasks } t_h, t_i \textit{ with } 1 \leq h, i \leq n \\ & \textit{such that } t_h.\mathsf{end} \leq t_i.\mathsf{start} \wedge t_i.\mathsf{end} \leq t_j.\mathsf{start} \\ & \textit{holds and for each task } t_k \textit{ with } 1 \leq k \leq n, \\ & k \neq i, k \neq h, k \neq j \textit{ it holds } t_k.\mathsf{end} \leq t_h.\mathsf{start} \\ & \textit{or } t_j.\mathsf{end} \leq t_k.\mathsf{start}, \\ \mathsf{initialCost}_{t_j.\mathsf{type}} & \textit{otherwise.} \end{cases}$$

This means that the alternating setup cost of a task t_j is determined by the direct predecessor t_h of its direct predecessor t_i – if there are any: t_h is scheduled before t_i, t_i is scheduled before t_j and there is neither a task scheduled between t_h and t_i nor between t_i and t_j.

This constraint is sufficient for a constraint-based modeling of the proposed objective

$$\sum_{i=1}^{n} \mathsf{alternatingSetupCost}(s_i, S) + \alpha \cdot s_i.\mathsf{score} \cdot s_i.\mathsf{start}$$

which is independent of the a-priori unknown permutation and correct by definition of the alternating setup costs constraint and because any permutation of the weighted start times will not change their sum.[7]

3.2 Implementation

The realized scheduling of clinical pathways is based on the presented modeling approaches (cf. Section 2). In particular, schedulers are realized for surgical pathways. The schedulers are implemented in Java combining object-orientation of the host language with Constraint Programming supported by the constraint solver library `firstCS` [11,23].

The pruning algorithms for the sequence dependent setup costs constraints are based on adjacency matrices. These matrices are used to maintain the transitive closure of the task order relation. For instance, possible predecessors (or pre-predecessors in the case of alternating costs) are used to restrict the domain of a task's cost variable or the maximal gap between two task is used to determine whether a task is a direct predecessor of another task thus determining the cost value of the direct successor task. For sequence dependent setup *times*, however, the implementation of the according constraint is based on the results presented in [6,24].

For the heuristic workload balancing the combined scheduling and labeling presented in Section 3.1 is implemented as described there. For efficiency reasons – this strategy is applied on-line while the surgeries are agreed between the

[7] Addition is associative and commutative.

physicians and their patients – this "heuristic optimization" is chosen. The computed schedules are accepted by the physicians and the patients because they are (mostly) conform with their expectations.

The optimized sequencing of surgeries presented in Section 3.1 uses in the first version a weighted task sum constraint – cf. e.g. [12,18] for its definition and pruning algorithms – and a task labeler which labels the start times of the considered tasks after finding a linear task order with depth-first, chronological backtracking search (cf. [12, Section 13.3]). Optimization uses a dichotomic branch-and-bound approach. However, this first approach performs rather bad in some use cases. For improving the runtime of this scheduler a specialized *weighted task sum constraint* [26] is investigated addressing the delay of the surgery start times. The improved pruning algorithms takes into account that the weighted addends are the start times of tasks to be scheduled in linear order yielding better approximations of the sum's lower bound. Furthermore, the labeling is replaced by a simplified search algorithm appropriate for task scheduling without breaks [22]. These modifications performs well even with the chosen dichotomic branch-and-bound optimization of the weighted sum.

Practical applications of the realized scheduling in hospitals have shown that the implementation performs well. In general – even optimal – schedules are computed within a few seconds on a modern PC, however in rare cases the branch-and-bound optimization is interrupted after a individually set time limit yielding sub-optimal but acceptable solutions.

4 Conclusion

There are modelings for clinical pathways presented. The considered modeling aspects reach from intra- to inter-path relationships addressing spatiotemporal and sequence dependent constraints. It is shown how these modelings are applied in practical applications to schedule clinical tasks on medical resources while balancing workload or minimizing costs and delays.

References

1. Aggoun, A., Beldiceanu, N.: Extending CHIP in order to solve complex scheduling and placement problems. Journal of Mathematical and Computer Modelling 17(7), 57–73 (1993)
2. Allen, J.F.: Maintaining knowledge about temporal intervals. Communications of the ACM 26(1), 832–843 (1983)
3. Baptiste, P., le Pape, C., Nuijten, W.: Constraint-Based Scheduling: Applying Constraint Programming to Scheduling Problems. International Series in Operations Research & Management Science, vol. 39. Kluwer Academic Publishers, Dordrecht (2001)
4. Beldiceanu, N., Carlsson, M.: A new multi-resource cumulatives constraint with negative heights. In: Van Hentenryck, P. (ed.) CP 2002. LNCS, vol. 2470, pp. 63–79. Springer, Heidelberg (2002)

5. Belien, J., Demeulemeestera, E.: A branch-and-price approach for integrating nurse and surgery scheduling. European Journal of Operational Research 189(3), 652–668 (2008)
6. Bouquard, J.-L., Lenté, C.: Equivalence to the sequence dependent setup time problem. In: Frank, J., Sabin, M. (eds.) CP 1998 Workshop on Constraint Problem Reformulation (October 30, 1998), `http://ti.arc.nasa.gov/ic/people/frank/bouquard.cp98.FINAL.ps` (last visited: 2008/08/14)
7. Burke, E.K., De Causmaecker, P., Vanden Berghe, G., Van Landeghem, H.: The state of the art of nurse rostering. The Journal of Scheduling 7(6), 441–499 (2004)
8. Dechter, R., Meiri, I., Pearl, J.: Temporal constraint networks. Artificial Intelligence 49(1-3), 61–95 (1991)
9. Gagné, C., Price, W.L., Gravel, M.: Scheduling a single machine with sequence dependent setup time using ant colony optimization. Technical Report 2001-003, Faculté des Sciences de L'Administration, Université Laval, Québec, Canada (April 2001)
10. Hannebauer, M., Müller, S.: Distributed constraint optimization for medical appointment scheduling. In: Proceedings of the Fifth International Conference on Autonomous Agents, AGENTS 2001, pp. 139–140. ACM, New York (2001)
11. Hoche, M., Müller, H., Schlenker, H., Wolf, A.: `firstCS` - A Pure Java Constraint Programming Engine. In: Hanus, M., Hofstedt, P., Wolf, A. (eds.) 2nd International Workshop on Multiparadigm Constraint Programming Languages – MultiCPL 2003 (September 29, 2003), `http://uebb.cs.tu-berlin.de/MultiCPL03/Proceedings.MultiCPL03.RCoRP03.pdf` (last visited: 2009/05/13)
12. Hofstedt, P., Wolf, A.: Einführung in die Constraint-Programmierung, eXamen.press. Springer, Heidelberg (2007) ISBN 978-3-540-23184-4
13. Kuhnert, S.: Efficient edge-finding on unary resources with optional activities. In: Seipel, D., Hanus, M., Wolf, A., Baumeister, J. (eds.) Proceedings of the 17th Conference on Applications of Declarative Programming and Knowledge Management (INAP 2007) and 21st Workshop on (Constraint) Logic Programming (WLP 2007), Institut für Informatik, Am Hubland, 97074 Würzburg, Germany. Technical Report, vol. 434, pp. 35–46. Bayerische Julius-Maximilians-Universität Würzburg (October 2007)
14. Laborie, P.: Algorithms for propagating resource constraints in AI planning and scheduling: Existing approaches and new results. Artifical Intelligence 143, 151–188 (2003)
15. Neumann, K., Schwindt, C.: Project scheduling with inventory constraints. Mathematical Methods of Operations Research 56, 513–533 (2002)
16. Rousseau, L.-M., Pesant, G., Gendreau, M.: A general approach to the physician rostering problem. Annals of Operations Research 115(1), 193–205 (2002)
17. Schlenker, H., Goltz, H.-J., Oestmann, J.-W.: Tame. In: Quaglini, S., Barahona, P., Andreassen, S. (eds.) AIME 2001. LNCS (LNAI), vol. 2101, pp. 395–404. Springer, Heidelberg (2001)
18. Schulte, C., Stuckey, P.J.: When do bounds and domain propagation lead to the same search space. In: Sørgaard, H. (ed.) Third International Conference on Principles and Practice of Declarative Programming, Florence, Italy, pp. 115–126. ACM Press, New York (September 2001)
19. Simonis, H., Cornelissens, T.: Modelling producer/consumer constraints. In: Montanari, U., Rossi, F. (eds.) CP 1995. LNCS, vol. 976, pp. 449–462. Springer, Heidelberg (1995)
20. Trajan, R.E., van Leeuwen, J.: Worst-case analysis of set union algortihms. Journal of the ACM 31(2), 245–281 (1984)

21. Vilím, P., Barták, R., Čepek, O.: Unary resource constraint with optional activities. In: Wallace, M. (ed.) CP 2004. LNCS, vol. 3258, pp. 62–76. Springer, Heidelberg (2004)
22. Wolf, A.: Reduce-to-the-opt – a specialized search algorithm for contiguous task scheduling. In: Apt, K.R., Fages, F., Rossi, F., Szeredi, P., Váncza, J. (eds.) CSCLP 2003. LNCS (LNAI), vol. 3010, pp. 223–232. Springer, Heidelberg (2004)
23. Wolf, A.: Object-oriented constraint programming in Java using the library `firstCS`. In: Fink, M., Tompits, H., Woltran, S. (eds.) 20th Workshop on Logic Programming, Vienna, Austria, February 22-24. INFSYS Research Report, vol. 1843-06-02, pp. 21–32. Technische Universität Wien (2006)
24. Wolf, A.: Constraint-based task scheduling with sequence dependent setup times, time windows and breaks. In: Im Fokus das Leben, INFORMATIK 2009. Lecture Notes in Informatics (LNI) - Proceedings Series of the Gesellschaft für Informatik (GI), vol. 154, pp. 3205–3219. Gesellschaft für Informatik e.V (2009)
25. Wolf, A., Schlenker, H.: Realizing the alternative resources constraint. In: Seipel, D., Hanus, M., Geske, U., Bartenstein, O. (eds.) INAP/WLP 2004. LNCS (LNAI), vol. 3392, pp. 185–199. Springer, Heidelberg (2005)
26. Wolf, A., Schrader, G.: Linear weighted-task-sum – scheduling prioritised tasks on a single resource. In: Seipel, D., Hanus, M., Wolf, A. (eds.) INAP 2007. LNCS (LNAI), vol. 5437, pp. 21–37. Springer, Heidelberg (2009)