Handling Airport Ground Processes based on Resource-Constrained Project Scheduling

Jürgen Kuster and Dietmar Jannach

Department of Business Informatics and Application Systems,
University of Klagenfurt, Universitätsstraße 65-67, 9020 Klagenfurt, Austria
{jkuster,dietmar}@ifit.uni-klu.ac.at
http://ifit.uni-klu.ac.at

Abstract. This paper describes how the Resource-Constrained Project Scheduling Problem (RCPSP) can be used as a basis for real-time decision support in the disruption management of the aircraft turnaround, the most typical airport ground process. For this purpose, the RCPSP is extended by the possibility to describe alternative activities, which can be used to model potential process modifications. An evolutionary approach is presented for solving this generalized problem, considering both basic rescheduling possibilities as well as the option of exchanging activities. The work presented in this paper is based on the results of a study conducted in collaboration with Deutsche Lufthansa AG, concerned with the analysis of the elementary requirements of decision support systems for turnaround process management.

1 Introduction

Managing disruptions of airport ground processes is a complex task: This is mainly due to the particularly high level of time and resource dependencies (among processes), the huge amount of parallel and interrelated activities as well as the often incomplete, unstable and deficient information, which forms the basis for the selection of appropriate repair activities. Moreover, the decision on a process intervention typically has to be made within little time, since the airport represents a highly dynamic and volatile environment with continuously changing resource and time availabilities.

In the current situation, it is usually the human operators who are responsible for disruption management (DM, see [1] for example) in the turnaround process. They take their decisions based on the available information and their individual experience. The improvement of the former element is targeted by the concepts of Collaborative Decision Making (CDM, see [2,3] for example), which focus on the augmentation of information awareness and information sharing. However, air traffic organizations are also particularly interested in the enhancement of the latter, the rather subjective foundation of respective decisions. We have therefore studied the elementary requirements of turnaround-related decision support systems (DSS) in collaboration with Deutsche Lufthansa AG. The work presented herein is based on the respective insights and findings.

Basically, the problem of providing decision support in management of disruptions in time- and resource-bound processes (such as the turnaround process but also any other time-critical production process) comes down to the selection of appropriate options of *rescheduling* and *process modification*: The respective set of interventions is required to be actually applicable and is considered optimal if it minimizes the costs associated with both the previous and the pending disruptions. Its execution shall furthermore produce a schedule as similar as possible to the original plan. Whereas the aspect of rescheduling has been discussed extensively in the literature, little work has been done on the options of structural modifications in flexible processes. The principal challenge consists in modelling this flexibility and in considering it in an algorithm which is able to provide good solutions in real-time.

We claim that the conceptual framework of the Resource-Constrained Project Scheduling Problem (RCPSP, see [4] for example) is perfectly suited for the resolution of the part of the problem concerned with rescheduling. The associated model provides possibilities to describe time and resource dependencies on a relatively high level, which makes the RCPSP intuitive and easily describable. Based on the respective elements, schedules for the contained activities can be generated and optimized in an efficient manner: Especially the possibility to use various forms of metaheuristics (as opposed to the application of exact mathematical programming) makes it possible to solve even problems of realistic size in reasonable time.

However, the current versions of the RCPSP provide a relatively weak support for structural flexibility and almost no possibilities to evaluate process modifications or alternative process paths. Even though the Multi-Mode RCPSP (MRCPSP, see [5] for example) introduces elementary forms of flexibility by making it possible to vary resource requirements and duration values, we argue that this is not sufficient for the application of the RCPSP to DM in realistic problems. This is mainly due to the fact that the human operator typically requires (and actually has) more options than the mere temporal shift of starting times and the simple modification of activity execution modes. He might, in particular, want to change the order of activities, insert or remove supporting tasks, or even parallelize activities which have been planned for serial execution (or vice versa).

This paper therefore introduces the *x-RCPSP* (where *x* stands for *ex*tended) as a generalization of the classical RCPSP. It provides possibilities to define alternative activities, which can be used to describe potential process modifications within a comprehensive process model. The remainder of the paper is structured as follows: In Section 2, a simplified version of the turnaround process is introduced along with three exemplary forms of potential modifications. Section 3 describes the *x-RCPSP* approach by defining a formal model and by outlining its application to the problem of disruption management. Section 4 provides an overview on related approaches before, finally, Section 5 summarizes the contribution of this paper and gives an outlook on future work.

2 The Turnaround Process

In general, the turnaround process combines all activities carried out at an airport while the respective aircraft is on ground. The simplified version considered for exemplary purposes in this paper, is structured as follows: After the plane reaches its final position, first the passengers leave the aircraft before it is fuelled, cleaned and catered simultaneously. After the last of these activities has finished, the outgoing passengers enter the plane which then leaves its position, heading for the runway. Figure 1 illustrates an instance of this process, where a disruption has occurred during deboarding, leading to deviations of the predicted (white) from the planned (gray) process times and finally causing a delay.

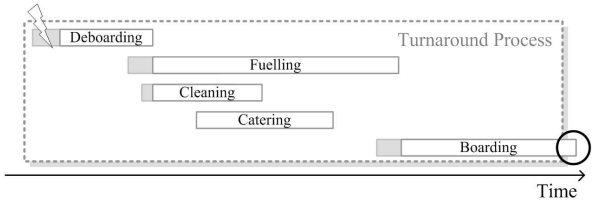

Fig. 1. Simplified Version of the Turnaround Process with Disruption

We assume that – apart from simply rescheduling (i.e. temporally shifting) the process – various forms of structural intervention are available to eliminate the pending delay. First, it is possible to accelerate deboarding by assigning additional busses. Second, it is possible to shorten cleaning, if in exchange the cabin is additionally inspected by the cabin crew prior to boarding. And finally, it is possible to parallelize fuelling and boarding if the fire brigade is available for supervision. Equipped with these options, the task of DM is the identification of a combination of rescheduling and process modification activities, which minimizes the departure delay and (in particular) the associated costs. Under the assumption that the original plan was optimal, the aim is to get back on track: The number of schedule modifications shall therefore also be minimized.

3 Using the RCPSP for Disruption Management

In this section, a generalization of the RCPSP is introduced, which can provide the basis for DSS in the area of process DM by supporting alternative activities and variable process execution paths. After the formal description of the model, an exemplary heuristic approach based on an evolutionary algorithm is sketched.

3.1 Extended Model

The extension proposed in this section is based on the idea of placing an abstract layer atop of the elementary constructs of the classical RCPSP. Instead of having only one fixed set of activities, associated precedence constraints and resource requirements, we distinguish two different levels by introducing respective supersets for these fundamental elements. The supersets contain all elements

which can but do not have to be considered in schedule generation, whereas the subjacent sets of *active* elements are actually and definitively considered: They form the basis for the final schedule.

Accordingly, the Extended Resource-Constrained Project Scheduling Problem *x-RCPSP* can be described as follows: Each project is defined by a set of potential activities $\mathcal{A}^+ = \{0, 1, ..., a, a+1\}$, where 0 and $a+1$ represent abstract start and end activities of the project, having a duration of 0 and no resource requirements associated. The currently *active* activities form a subset $\mathcal{A} \subseteq \mathcal{A}^+$. Correspondingly, $\mathcal{A}^+ \setminus \mathcal{A}$ represents the set of *inactive* activities, which shall not be contained in the final schedule. Let furthermore be $\mathcal{A}^0 \subseteq \mathcal{A}^+$ the set of those activities which jointly form the reference (i.e. standard) process and which shall be executed preferably. For the execution of the activities, a set of non-renewable resource types $\mathcal{R} = \{1, ..., r\}$ is available, with u_k units available for type k. For the definition of various forms of dependencies, the following constructs can be used to describe activity details:

- A duration d_i is associated with an activity i and describes how long its execution lasts. The respective value must not be negative: $d_i \geq 0 \quad \forall i \in \mathcal{A}^+$
- Precedence constraints can be used to order activities: A precedence constraint $p_{i,j}$ states that activity i has to be finished at or before the start of activity j. As regards the grouping of such constraints, a two-leveled approach is used, corresponding to the classification of activities: $\mathcal{P}^+$ contains all potential precedence constraints linking the elements of $\mathcal{A}^+$, whereas *active* precedence constraints form the subset $\mathcal{P} = \{p_{i,j} \in \mathcal{P}^+ \mid i, j \in \mathcal{A}\}$. If we introduce β_i as the starting time of activity i, the following statement has therefore to be true: $\beta_i + d_i \leq \beta_j \quad \forall p_{i,j} \in \mathcal{P}$
- Resource requirements describe the relationship between activities and resources: An activity i requires $q_{i,k}$ units of type k throughout its execution. Again, two sets can be distinguished: $\mathcal{Q}^+$ contains one requirement definition for any combination of an activity $i \in \mathcal{A}^+$ and a resource type $r \in \mathcal{R}$, whereas $\mathcal{Q} = \{q_{i,k} \in \mathcal{Q}^+ \mid i \in \mathcal{A}\}$ groups only the requirements describing *active* activities. Resources are constrained the way that the respective requirements must not exceed the availabilities at any time t. If we suppose $\mathcal{A}_t$ to be the set of activities carried out at t, the following statement has to be true for any t: $\sum_{i \in \mathcal{A}_t} q_{i,k} \leq u_k \quad \forall k \in \mathcal{R}$

Li et al. [6] used several indices to describe alternative *resources* in scheduling problems: Apart from the description of exchange possibilities, they also considered mutual dependencies in their model. We adapt this methodology for the description of alternative *activities* by introducing the following constructs:

- Alternative Activity Index $\mathcal{X}^+$. This index describes potential substitutes for an activity: $x_{i,j} \in \mathcal{X}^+$ states that activity i can be *deactivated* (i.e. removed from $\mathcal{A}$) if activity j is *activated* (i.e. added to $\mathcal{A}$). Note, that the matrix formed this way is not necessarily symmetric, since the possibility to replace activity i with activity j does not automatically imply the possibility to substitute j with i.

– Mutual Dependency Index $\mathcal{M}^+$. This index groups all definitions of activity interdependency. Basically, the following two types of binary and non-commutative relationship between elements of $\mathcal{A}^+$ can be distinguished:
  – Mutual Exclusion. $m^{\ominus}_{i,j} \in \mathcal{M}^+$ implies that activity j can and shall be removed from the schedule upon *activation* of activity i.
  – Mutual Inclusion. $m^{\oplus}_{i,j} \in \mathcal{M}^+$ states that i and j are linked and that activity j shall also be added to $\mathcal{A}$ if activity i is *activated*.

Equipped with these constructs, the *x-RCPSP* can be used to describe potential *structural modifications*. Basically, each possible form of respective intervention introduces a choice point into the process model, where it is possible to select either the activity of the reference process or a potential alternative. In this context, the switch from a previously chosen execution path to a valid alternative corresponds to a structural modification. For the simplified version of the turnaround process and the associated forms of intervention (see Sect. 2), the resulting model is illustrated in Figure 2: Solid lines visualize the structure of the reference process whereas dashed lines show alternative activities and respective precedence constraints. The encircled *or*-nodes symbolize choice points.

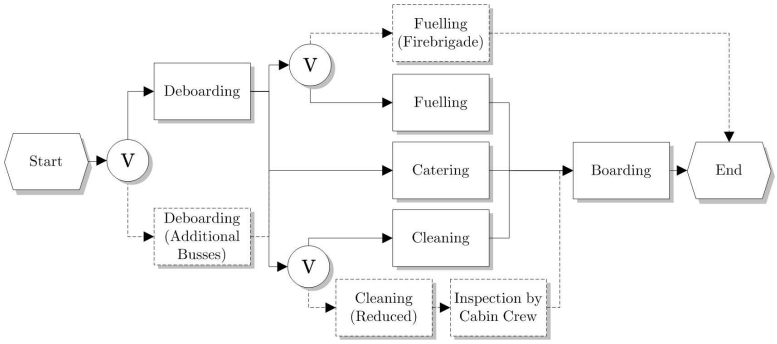

Fig. 2. Turnaround Process with Modification Possibilities

Correspondingly, we might define the elements of the *x-RCPSP* as given in Table 1. Note, that for improved readability a simplified form of notation is used in the following: $i \rightarrow j$ defines that $p_{i,j} \in \mathcal{P}^+$, $i \triangleright n \times k$ defines that $q_{i,k} = n \in \mathcal{Q}^+$, $i \Rightarrow j$ defines that $x_{i,j} \in \mathcal{X}^+$, $i \Leftrightarrow j$ defines that $x_{i,j}, x_{j,i} \in \mathcal{X}^+$, $i \oplus j$ defines that $m^{\oplus}_{i,j} \in \mathcal{M}^+$ and $i \ominus j$ defines that $m^{\ominus}_{i,j} \in \mathcal{M}^+$.

3.2 An Evolutionary Algorithm for Disruption Management

In the context of time- and resource-bound processes, disruption management is concerned with the continuous adaptation of a schedule (defining the starting times of future activities) to the dynamic and stochastic real-world environment. In a comprehensive view, not only rescheduling (i.e. the rearrangement of contained activities) but also potential structural interventions (i.e. the exchange of alternative activities) have to be considered. This section illustrates how the model described above can serve as the basis for respective DM.

Table 1. Formal Description of the Exemplary Turnaround Process

Set	Content
$\mathcal{R}$	Bus, Firebrigade
$\mathcal{A}^0$	Start, Deb, Fue, Cat, Cle, Boa, End
$\mathcal{A}^+$	Start, Deb, Deb^{Bus}, Fue, Fue^{Par}, Cat, Cle, Cle^{Red}, Ins, Boa, End
$\mathcal{P}^+$	Start $\rightarrow$ Deb, Start $\rightarrow$ Deb^{Bus}, Deb $\rightarrow$ Fue, Deb $\rightarrow$ Fue^{Par}, Deb $\rightarrow$ Cat, Deb $\rightarrow$ Cle, Deb $\rightarrow$ Cle^{Red}, Deb^{Bus} $\rightarrow$ Fue, Deb^{Bus} $\rightarrow$ Fue^{Par}, Deb^{Bus} $\rightarrow$ Cat, Deb^{Bus} $\rightarrow$ Cle, Deb^{Bus} $\rightarrow$ Cle^{Red}, Fue $\rightarrow$ Boa, Fue^{Par} $\rightarrow$ End, Cat $\rightarrow$ Boa, Cle $\rightarrow$ Boa, Cle^{Red} $\rightarrow$ Ins, Ins $\rightarrow$ Boa, Boa $\rightarrow$ End
$\mathcal{Q}^+$	Deb $\triangleright$ 1 $\times$ Bus, Deb^{Bus} $\triangleright$ 2 $\times$ Bus, Fue^{Par} $\triangleright$ 1 $\times$ Firebrigade
$\mathcal{X}^+$	Deb $\Leftrightarrow$ Deb^{Bus}, Fue $\Leftrightarrow$ Fue^{Par}, Cle $\Leftrightarrow$ Cle^{Red}
$\mathcal{M}^+$	Cle^{Red} $\oplus$ Ins, Cle $\ominus$ Ins

For this purpose we apply an evolutionary algorithm, where optimization is accomplished in the continuous evolution of a population: The fittest individuals survive and their children are generated through recombination and mutation. As a metaheuristic approach, it provides good results in shorter time than exact optimization approaches of mathematical programming. In the following, an appropriate form of representation, an approach for the generation of an initial population, a potential fitness function and selection scheme and, particularly, specific crossover and mutation operators are described. As regards respective interrelations, a basic understanding of evolutionary algorithms is assumed.

Encoding A schedule causing higher costs than originally intended represents the starting point for the evolutionary algorithm. As opposed to direct representation, where a solution itself represents an individual, we use the indirect form of activity list representation: The reason for this is the difficulty of directly representing and modifying the schedule's time values [7]. An activity list λ corresponds to a correctly (with respect to precedence constraints) sorted vector of all elements in $\mathcal{A}$. It defines the order in which activities shall be added to the schedule and can therefore be converted into a final set of starting times unambiguously. Respective Schedule Generation Schemes (SGS) have been described by Kolisch et al. [8] and Hindi et al. [7], for example. Note, that the scheduling process is thus split into two steps: λ respects all precedence relations, whereas only the finally derived schedule takes resource constraints into consideration.

Initial Population The original schedule shall represent the progenitor of all members of the initial population. Since it is assumed to respect any precedence constraint, the respective timetable can be converted into an activity list λ by simply sorting all elements $i \in \mathcal{A}^+$ which have been considered in the schedule according to their starting times. All other members of the initial population are deduced from this original solution (which corresponds to the option of *not* intervening at all) through the application of the mutation operator (as discussed below). For the considered example, $\langle$Deb, Fue, Cle, Cat, Boa$\rangle$ represents a potential first member, from which other activity sequences such as $\langle$Deb, Cle, Fue, Cat, Boa$\rangle$ or $\langle$Deb, Fue, Cle^{Red}, Ins, Cat, Boa$\rangle$ can be generated.

Fitness and Selection The fitness function evaluates the quality of an activity list through the assessment of the associated interventions and the analysis of the implied schedule. In the reduction of a potential solution to a simple numeric quality value, the function has to consider all objectives for schedule optimization: Whereas most scheduling approaches for the classical RCPSP focus on the makespan property, disruption management is rather concerned with the implications of earliness and tardiness, costs for interventions as well as the dissimilarity to the original plan. Accordingly, a turnaround-related fitness function typically combines the costs associated with predicted departure delays and the costs of intervening, considering the fact that higher costs imply lower quality.

If the members of an existing population do not fulfill specific optimization criteria, the best solutions form the next generation along with the offspring created through recombination and mutation (as discussed below): Respective parents are selected with a probability proportional to their (relative) fitness.

Crossover Operator A method for the combination of two parent solutions is summarized in Alg. 1. Whereas the case of both elements being based on the same set of activities can be handled by the operators which have been defined for the classical RCPSP [7], we particularly focus on the description of how crossover can look like if the activity sets are distinct. The basic idea for our approach is that one parent λ_a prescribes which activities shall be contained in the child and the other one, λ_b, defines their order. To cope with different elements, we use a transition set $\mathcal{T} \subseteq \mathcal{X}^+$ which describes the conversion from activity list λ_b to λ_a. If $\mathcal{X}_a$ is assumed to be the set of modifications which led from the original sequence to a solution a, $\mathcal{T}$ is intended to define one transition for every element which exists only in either $\mathcal{X}_a$ or $\mathcal{X}_b$. Note, that respective elements of $\mathcal{X}_a$ can be applied directly, whereas the direction of the transition has to be inverted for elements of $\mathcal{X}_b$: Unless all elements can be converted this way, the parent solutions are considered incompatible and the algorithm returns without a result (line 5). Otherwise, each activity $i \in \lambda_b$ is appended to the child sequence λ if it is also contained in λ_a: Precedence constraints are considered in the way that all successors of i (as well as their successors) are shifted to the end of the vector. If $i \notin \lambda_a$ and a valid transition is defined in $\mathcal{T}$, the respective exchange operation is executed, taking into account all mutual dependencies: Afterwards the loop is resumed considering j in the next iteration. If i neither is contained in λ_a nor can be converted (consider activities which will be removed later on due to an exclusive dependency), the method proceeds.

As regards the turnaround process, consider two parent nodes $\lambda_1 = \langle \text{Deb}^{Bus}, \text{Fue}, \text{Cle}, \text{Cat}, \text{Boa} \rangle$ and $\lambda_2 = \langle \text{Deb}^{Bus}, \text{Cat}, \text{Cle}^{Red}, \text{Ins}, \text{Boa}, \text{Fue}^{Par} \rangle$. If $\lambda_a \leftarrow \lambda_1$ and $\lambda_b \leftarrow \lambda_2$, the transition set $\mathcal{T} = \{\text{Cle}^{Red} \Rightarrow \text{Cle}, \text{Fue}^{Par} \Rightarrow \text{Fue}\}$ groups the inversions of the elements exclusively contained in $\mathcal{X}_b$. According to the order prescribed by λ_b, activity list $\lambda = \langle \text{Deb}^{Bus}, \text{Cat}, \text{Cle}, \text{Fue}, \text{Boa} \rangle$ is created as a child. If alternatively $\lambda_a \leftarrow \lambda_2$ and $\lambda_b \leftarrow \lambda_1$, the sequence $\lambda = \langle \text{Deb}^{Bus}, \text{Fue}^{Par}, \text{Cle}^{Red}, \text{Ins}, \text{Cat}, \text{Boa} \rangle$ can be generated (under the assumption that mutually inclusive activities are placed at their earliest possible positions).

Algorithm 1 Crossover (λ_a, λ_b)

if $\mathcal{A}_a = \mathcal{A}_b$ **then**
  generate λ through the application of an RCPSP-related crossover operator
else
  $\mathcal{T} \leftarrow (\mathcal{X}_a \setminus \mathcal{X}_b) \cup \{x_{i,j} \in \mathcal{X}^+ | x_{j,i} \in (\mathcal{X}_b \setminus \mathcal{X}_a)\}$
  if $|\mathcal{T}| < |\mathcal{X}_a \triangle \mathcal{X}_b|$ **then** return *false*
  for all i in λ_b **do**
    if $i \in \lambda_a$ **then** append(λ, i)
    elseif $\exists\ x_{i,j} \in \mathcal{T}$ **then** replace i with j in λ_b and **proceed** with j
    else proceed
  end for
end if
return λ

Mutation Operator In order to avoid early convergence to local minima, a mutation operator is used (with a certain probability) to slightly modify a generated child and to extend the space of considered options thereby. A potential realization is summarized in Alg. 2: Given an activity list λ, mutation means that activities are either shifted (i.e. rescheduled) or exchanged: θ corresponds to the probability of applying the former, $1-\theta$ to the probability of applying the latter form of modification. Methods for mutating an activity list by shifting the contained elements have been described in the context of the RCPSP (see [7], for example). We therefore focus on the exchange option herein: For the respective modification, first an arbitrary $x_{i,j} \in \mathcal{X}^+$ is selected for any $i \in \lambda$. Then, i and all elements excluded by j are removed from the activity list, before finally j and all linked activities are added at the former position of i or after their last predecessor: All associated successors (with subsequent successors) have to be shifted to the right-hand side of the inserted element.

Algorithm 2 Mutate (λ)

if a randomly generated value $\leq \theta$ **then**
  rearrange λ through the application of an RCPSP-related mutation operator
else
  select an arbitrary $x_{i,j} \in \mathcal{X}^+ | i \in \lambda$
  remove i and $\{k | m^{\ominus}_{j,k} \in \mathcal{M}^+\}$ from λ
  insert j and $\{k | m^{\oplus}_{j,k} \in \mathcal{M}^+\}$ at the former position of i or after the last predecessor, and shift all successors to the right-hand side
end if
return λ

As far as the example of the turnaround process is concerned, consider the activity list $\lambda = \langle \text{Deb}^{Bus}, \text{Cat}, \text{Cle}, \text{Fue}, \text{Boa} \rangle$: A simple rescheduling operation can transform this sequence into $\langle \text{Deb}^{Bus}, \text{Cle}, \text{Cat}, \text{Fue}, \text{Boa} \rangle$ or $\langle \text{Deb}^{Bus}, \text{Fue}, \text{Cle}, \text{Cat}, \text{Boa} \rangle$, for example. The exchange operator defined before, might mutate λ into $\langle \text{Deb}^{Bus}, \text{Cat}, \text{Cle}, \text{Fue}^{Par}, \text{Boa} \rangle$ if $\text{Fue} \Rightarrow \text{Fue}^{Par}$ is chosen from $\mathcal{X}^+$, or into $\langle \text{Deb}^{Bus}, \text{Cat}, \text{Cle}^{Red}, \text{Ins}, \text{Fue}, \text{Boa} \rangle$ upon the selection of $\text{Cle} \Rightarrow \text{Cle}^{Red}$.

4 Related Work

As an individual research area, disruption management originally and traditionally focused on airline aircraft and crew scheduling (see [1,9] for example): The Descartes project, which is concerned with the efficient rescheduling of aircrafts, crews and passengers in case of disruptions, represents the main contribution to the development and application of respective concepts [10,11]. Currently, DM is also being adapted for various other areas: Production planning [12,13] as well as supply chain management [14] represent examples of respective domains of application.

Disruption management for the RCPSP has been described by Zhu et al. [15]: They confine the possibilities of interventions to the classical options such as rescheduling as well as a modification of durations and resource assignments. For the resolution of the mathematically formulated problem a hybrid mixed-integer programming/constraint programming procedure is applied. Thus, Zhu et al. focus on the identification of an optimal solution for classical forms of modification, whereas the approach described herein is intended to identify (only) good solutions in a wider space of options and within a shorter time horizon.

As regards the idea of providing the RCPSP with additional flexibility, little research has been done so far. To the best of our knowledge, only Artigues et al. [16] and Elkhyari et al. [17] have made respective proposals. The former is concerned with the dynamic insertion of activities, where each occurrence of an unexpected activity corresponds to a disruption. The latter provides possibilities to handle over-constrained networks based on the excessive use of so-called explanations. And also the concept of alternative activities has received only little attention in the scheduling domain: Only Beck et al. [18] proposed an approach, in which a Probability of Existence (PEX) can be defined for any activity.

5 Conclusions and Future Work

This paper illustrated how the concept of the RCPSP can be used as a basis for turnaround process disruption management. For this purpose, a generalization has been introduced, which is able to cope with alternative activities and potential forms of process modifications. The *x-RCPSP* provides several advantages: First, it supports comprehensive flexibility in complex process structures. Second, the underlying concept is intuitive, which makes it easy to define and maintain respective models. And finally, the possibility to apply metaheuristic search methods makes it possible to use the generalization of the RCPSP for real-time decision support in realistic disruption management.

An unoptimized version of the presented algorithms has been implemented in a Java-based prototype: It illustrates the effectiveness of the proposed methods and their practical applicability to DSS in turnaround process DM. Future tasks include the further optimization of the respective procedures, research on potential reductions of the search space (following Bean et al. [19], for example) and the generation of benchmarkable results for realistic problem sizes.

References

1. Clausen, J., Hansen, J., Larsen, J., Larsen, A.: Disruption Management. ORMS Today **28** (2001) 40–43
2. Wambsganss, M.: Collaborative Decision Making through Dynamic Information Transfer. Air Traffic Control Quarterly **4** (1997) 107–123
3. Hoffman, R., Ball, M., Odoni A., Hall W., Wambsganss M.: Collaborative Decision Making in Air Traffic Flow Management. Technical Report, UC Berkeley (1999)
4. Brucker, P., Drexl, A., Mohring, R., Neumann, K., Pesch, E.: Resource-constrained project scheduling: Notation, classification, models, and methods. European Journal of Operational Research **112** (1999) 3–41
5. Hartmann, S.: Project Scheduling with Multiple Modes: A Genetic Algorithm. Annals of Operations Research **102** (2001) 111–135
6. Li, R.K-Y., Willis, R.J.: Alternative resources in project scheduling. Computers and Operations Research **18** (1991) 663–669
7. Hindi, K.S., Yang, H., Fleszar, K.: An Evolutionary Algorithm for Resource-Constrained Project Scheduling. IEEE Transactions on Evolutionary Computation **6** (2002) 512–518
8. Kolisch, R., Hartmann, S.: Heuristic Algorithms for solving the Resource-Constrained Project Scheduling Problem: Classification and Computational Analysis. In: Weglarz, J. (Ed.): Project scheduling: Recent models, algorithms and applications (1999) 147–178
9. Thengvall, B., Bard, J.F., Yu, G.: Balancing user preferences for aircraft schedule recovery during irregular operations. IIE Transactions on Operations Engineering **32** (2000) 181–193
10. Kohl, N., Larsen, A., Larsen, J., Ross, A., Tiourine, S.: Airline Disruption Management - Perspectives, Experiences and Outlook. Technical Report 2004-16, IMM, Technical University of Denmark (2004)
11. Clausen, J., Larsen, A., Larsen, J.: Disruption Management in the Airline Industry - Concepts, Models and Methods. Technical Report 2005-01, IMM, Technical University of Denmark (2005)
12. Yang, J., Qi, X., Yu, G.: Disruption management in production planning. Naval Research Logistics **52** (2005) 420–442
13. Xia, Y., Yang, M.H., Golany, B., Gilbert, S., Yu, G.: Real-time disruption management in a two-stage production and inventory system. IIE Transactions **36** (2004) 111–125
14. Xu, M., Qi, X., Yu, G., Zhang, H., Gao, C.: The demand disruption management problem for a supply chain system with nonlinear demand functions. Journal of Systems Science and Systems Engineering **12** (2003) 82–97
15. Zhu, G., Bard, J.F., Yu, G.: Disruption management for resource-constrained project scheduling. Journal of the Operational Research Society **56** (2005) 365–381
16. Artigues, C., Michelon, P., Reusser, S.: Insertion techniques for static and dynamic resource constrained project scheduling. European Journal of Operational Research **149** (2003) 249–267
17. Elkhyari, A., Guéret, C., Jussien, N.: Constraint Programming for Dynamic Scheduling Problems. In: ISS'04 Int. Scheduling Symposium (2004) 84–89
18. Beck, J.C., Fox, M.S.: Constraint Directed Techniques for Scheduling with Alternative Activities. Artificial Intelligence **121** (2000) 211–250
19. Bean, J.C., Birge, J.R., Mittenthal, J., Noon, C.E.: Matchup Scheduling with Multiple Resources, Release Dates and Disruptions. Operations Research **39** (1991) 470–483