Decision Making in Manufacturing and Services
Vol. 2 • 2008 • No. 1–2 • pp. 5–32

CP-driven Production Process Planning in Multiproject Environment

Zbigniew Banaszak*, Grzegorz Bocewicz**, Irena Bach***

Abstract. The way enterprise capabilities are used decides about its competitiveness among other ones. In that context modeling aimed at production tasks allocation planning plays a crucial role especially at concurrently executed production orders. The introduced reference model employing constraint programming (CP) paradigm describes both an enterprise and a set of project-like production orders. Moreover, encompassing consumer orders requirements and available production capabilities, the model provides the formal framework allowing one to develop a class of decision support systems aimed at interactive production process planning subject to multiproject environment constraints. In that context our contribution is a knowledge-based and CP-driven approach to resource allocation assuming precise character of decision variables. The conditions sufficient for deadlock avoidance are the main goal. The conditions delivered provide formal framework for developing a task oriented Decision Support Tool for Project Portfolio Prototyping (DST4P, Banaszak 2006). The tool provides a prompt and interactive service to a set of routine queries formulated either in straight or reverse way

Keywords: projects portfolio, support system, constraints programming, scheduling

Mathematics Subject Classification: 90B35, 90C10, 91B32, 90B50

Journal of Economics Literature Classification: C60
Revised: 14 December 2008

1. INTRODUCTION

An optimal assignment of available resources to production steps in a multi-product job shop is often economically indispensable. The goal is to generate a plan/schedule of production orders for a given period of time while minimize the cost that is equivalent to maximization of profit. In that context executives want to know how much a particular production order will cost, what resources are needed, what resources allocation can guarantee due time production order completion, and so on. So, a manager needs might be formulated in a form of standard, routine questions, such as: Does the

* Department of Computer Science and Management, Koszalin University of Technology, Śniadeckich 2, 74-453 Koszalin, Poland, E-mail: zbigniew.banaszak@tu.koszalin.pl
** E-mail: bocewicz@ie.tu.koszalin.pl
*** E-mail: bachirena@wp.pl

production order can be completed before an arbitrary given deadline? What is the production completion time following assumed robots operation time? Is it possible to undertake a new production order under given (constrained in time) resources availability while guaranteeing disturbance-free execution of the already executed orders? What values and of what variables guarantee the production order will completed following assumed set of performance indexes?

The problems standing behind of the quoted questions belong to the class of so called project scheduling ones. In turn, project scheduling can be defined as the process of allocating scarce resources to activities over a period of time to perform a set of activities in a way taking into account a given set of performance measures. Such problems belong to NP-complete ones. Therefore, the new methods and techniques addressing the impact of real-life constraints on the decision making is of great importance, especially in case of interactive and task oriented DSSs designing (Banaszak, 2006; Bocewicz et al., 2007).

Several techniques have been proposed in the past fifty years, including MILP, Branch-and-Bound (Beale, 1979) or more recently Artificial Intelligence. The last sort of techniques concentrates mostly on fuzzy set theory and constraint programming frameworks. Constraint Programming/Constraint Logic Programming (*CP*/*CLP*) languages (Beale, 1979; Schutle et al., 1998; Van Hentenryck, 1991) seems to be well suited for modeling of real-life and day-to-day decision-making processes in an enterprise (Bach et al., 2008a). In turn, applications of fuzzy set theory in production management (Banaszak et al., 2005) show that most of the research on project scheduling has been focused on **fuzzy *PERT*** and **fuzzy *CPM*** (Chanas and Komburowski, 1981; Dubois et al., 2003).

In this context, the contribution provides the framework allowing one to take into account distinct data describing modeled object and then to treat them in terms of the constraint satisfaction problem (CSP) (Bach et al., 2008b).The approach proposed concerns of logic-algebraic method *(LAM)* based and *CP*-driven methodology aimed at interactive decision making based on distinct and imprecise data. The paper can be seen as continuation of our former works concerning projects portfolio prototyping (Bach et al., 2008a; Bach et al., 2008b).

The following two classes of standard routine queries are usually considered and formulated in:

a straight way (i.e. corresponding to the question: What results from premises?

- What the portfolio makespan follows from the given project constraints specified by activity duration times, resources amount and their allocation to projects' activities?
- Does a given resources allocation guarantee the production orders makespan do not exceed the given deadline?
- Does the projects portfolio can be completed before an arbitrary given deadline?
- and so on.

a reverse way (i.e. corresponding to the question: What implies conclusion?)

- What activity duration times and resources amount guarantee the given production orders portfolio makespan do not exceed the deadline?

- Does there exist resources allocation such that production orders makespan do not exceed the deadline?
- Does there exist a set of activities' operation times guaranteeing a given projects portfolio completion time will not exceed the assumed deadline?
- and so on.

Above mentioned categories encompass the different reasoning perspectives, i.e. deductive and abductive ones. The corresponding queries can be stated in the same model that can be treated as composition of variables and constraints, i.e. assumed sets of variables and constraints limiting their values. In that context both an enterprise and the relevant production orders can be specified in terms of distinct and/or imprecise variables, discrete and/or continuous variables, renewable and/or non-renewable resources, limited and/or unlimited resources, and so on.

Therefore, an approach proposed assumes a kind of reference model encompassing open structure enabling one to take into account different sorts of variables and constraints as well as to formulate straight and reverse kind of project planning problems. So, the elementary as well as hybrid models can be considered, see the Fig. 1. Of course, the most general case concerns of the hybrid model specified by discrete distinct and/or imprecise (fuzzy) variables and renewable and/or non-renewable resources.

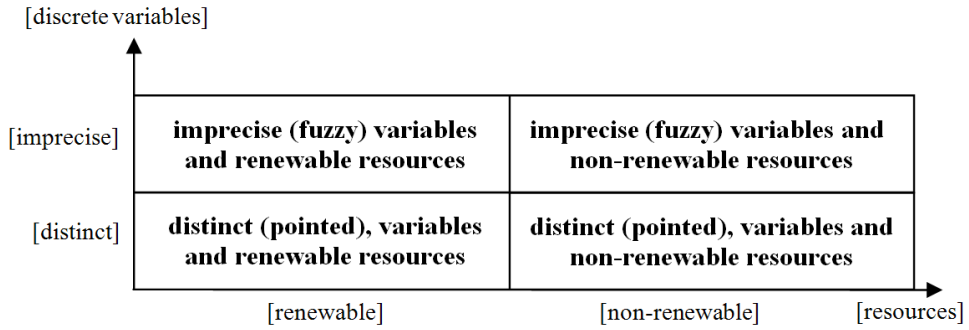

Fig. 1. Elementary decision problems

The assumed model enabling descriptive way of a problem statement encompasses constraint satisfaction problem structure and then allows implementing the problem considered in constraint programming environment. That is because the constraint programming treated as programming paradigm enables to specify both variables and relations between them in the form of constraints and then to implement them in the one of popular constraint logic languages such as: **CHIP V5**, **ECLiPSe**, and **SICStus**, or imperative constraint programming languages (assuming that a statement computation results in a program state change) such as: **Choco**, **ILOG**, and **python-constraint**, or public domain concurrent constraint programming language as **Oz Mozart**.

In order to illustrate the approach proposed let us focus on a reference model of decision problem encompassing equilibrium between possible expectations regarding potential orders completion (e.g. following a set of routine queries) and available production capabilities. The considered decision problem concerns of resources conflict resolution, i.e. conflicts arising in case different activities simultaneously request their access to renewable and non renewable resources of limited quantity.

2. REFERENCE MODEL

2.1. DECISION PROBLEM

Both kinds of queries distinguished in the Section 1 (i.e. concerning the straight and reverse problems formulation) assumes at least one feasible solution there exists. That means, the class of so called decision problems focusing on the question whether any feasible solution there exists should be stated at first. Then, following from the guarantee a set of feasible solutions is not empty the class of so called optimization problems can be considered as well.

In this contribution, we concentrate on the first kind of problems, i.e. decision ones. So, the sets of considered queries are aimed at searching for feasible solutions while are formulated in the straight or reverse ways. Typical queries of both kinds are: Does the given resources allocation guarantee the duration time of considered projects portfolio do not exceed the assumed deadline? What values and of what variables if any guarantee the duration time of considered projects portfolio do not exceed the assumed deadline?

In case the optimal solutions are sought, i.e. optimization problems are considered, the above mentioned questions have to be reformulated, for instance as follow: What resources allocation results in shortest makespan of considered projects portfolio? What are the optimal values and of what variables guarantee the considered projects portfolio completion time is due date?

In that context the problem of production process planning in an small and medium sized enterprise (SME) environment seen as projects portfolio scheduling can be treated either as searching for such resources allocation, or as searching for such adjustment of arbitrarily chosen variables which guarantees the required values of assumed performance indexes hold. Both kinds of problems can be stated and resolved then in terms of so called reference model of decision problem, see the section bellow.

2.2. REFERENCE MODEL OF DECISION PROBLEM

Let us consider the reference model of a decision problem concerning of multi-resource task allocation in a multi-product job shop assuming imprecise character of decision variables. The model specifies both the job shop capability and production orders requirement in a unified way, i.e., through the description of determining them sets of variables and sets of constraints restricting domains of discrete variables. Some

conditions concerning the routine questions are included in the set of constraints. That means in case such conditions hold the response to associated questions is positive. Of course, in order to avoid confusion the constraints guaranteeing the responses DO NOT KNOW are not allowed are also taken into account. In that context, the reference model is aimed at routine questions such as: Does a given job shop capabilities and a given way of resources allocation guarantee the assumed makespan of production orders do not exceed the deadline h? (Bach et al. 2008a,b) Therefore, the reference model considered specifies both SMEs and projects portfolio in terms of describing them variables and constraints.

Decision variables

SME – a job-shop perspective. Given amount lz of renewable discrete resourcesro_i specified by (e.g. workforce, machine tools, AGVs, etc.): $Ro = (ro_1, ro_2, \ldots, ro_z)$. Given amounts $zo_{i,k}$ of available renewable resources $zo_i = (zo_{i,1}, zo_{i,2}, \ldots, zo_{i,h})$, where $zo_{i,k}$ – limited amount of the i-th renewable resource available at the k-th moment of H: $H = \{0, 1, \ldots, h\}$, specified by $Zo = (zo_1, zo_2, \ldots, zo_{lz})$

Given amount ln of non-renewable resources (i.e. money) rn_i specified by: $Rn = (rn_1, rn_2, \ldots, rn_{ln})$. Given amounts zn_i of available non-renewable resources rn_i specified by: $Zn = (zn_1, zn_2, \ldots, zn_{ln})$, where zn_i denotes amount of the resource rn_i being available at the beginning of time horizon H.

Projects portfolio. Given a set of projects $P = \{P_1, P_2, \ldots, P_{lp}\}$, where P_i is specified by the set composed of lo_i activities, i.e., $P_i = \{O_{i,1}, \ldots, O_{i,loi}\}$, where:

$$O_{i,j} = (x_{i,j}, t_{i,j}, Tp_{i,j}, Tz_{i,j}, Dp_{i,j}, Tr_{i,j}, Ts_{i,j}, Cr_{i,j}, Cs_{i,j}) \tag{1}$$

$x_{i,j}$ – means the starting time of the activity $O_{i,j}$, i.e., the time counted from the beginning of the time horizon H,

$t_{i,j}$ – the duration of the activity $O_{i,j}$,

$Tp_{i,j} = (tp_{i,j,1},\ tp_{i,j,2}, \ldots, tp_{i,j,lz})$ – the sequence of moments the activity $O_{i,j}$ requires new amounts of renewable resources: $tp_{i,j,k}$ – the time counted since the moment $x_{i,j}$ of the $dp_{i,j,k}$ amount of the k-th resource allocation to the activity $O_{i,j.}$ That means a resource is allotted to an activity during its execution period: $0 \leq tp_{i,j,k} < t_{i,j}$; $k = 1, 2, \ldots, lz$.

$Tz_{i,j} = (tz_{i,j,1},\ tz_{i,j,2}, \ldots, tz_{i,j,lz})$ – the sequence of moments the activity $O_{i,j}$ releases the subsequent resources, $tz_{i,j,k}$ – the time counted since the moment $x_{i,j}$ the $dp_{i,j,k}$ amount of the k-th renewable resource was released by the activity $O_{i,j}$. That is assumed a resource is released by activity during its execution: $0 < tz_{i,j,k} \leq t_{i,j}$ and $tp_{i,j,k} < tz_{i,j,k}$; $k = 1, 2, \ldots, lz$.

$Dp_{i,j} = (dp_{i,j,1}, dp_{i,j,2}, \ldots, dp_{i,j,lz})$ – the sequence of the k-th resource amounts $dp_{i,j,k}$ are allocated to the activity $O_{i,j}$, i.e., $dp_{i,j,k}$ – the amount of the k-th resource allocated to the activity $O_{i,j}$. That assumes: $0 \leq dp_{i,j,k} \leq zo_k$; $k = 1, 2, \ldots, lz$.

$Cr_{i,j} = (cr_{i,j,1}, cr_{i,j,2}, \ldots, cr_{i,j,ln})$ – the sequence of non-renewable resources amount required by activity $O_{i,j}$, : $cr_{i,j,k}$ – the amount of the k-th resource

required by the activity $O_{i,j}$, $cr_{i,j,1} \leq 0$; $k = 1, 2, \ldots, ln$, $cr_{i,j,k} = 0$ means the activity does not consume the k-th resource.

$Cs_{i,j} = (cs_{i,j,1}, cs_{i,j,2}, \ldots, cs_{i,j,ln})$ – the sequence of amounts of non-renewable resources released by activity $O_{i,j}$, $cs_{i,j,k}$ – the amount of the k-th resource involved by activity $O_{i,j}$, $cs_{i,j,1} \geq 0$; $k = 1, 2, \ldots, ln$, $cr_{i,j,k} = 0$ means the activity does not inflow the k-th resource.

$Tr_{i,j} = (tr_{i,j,1}, tr_{i,j,2}, \ldots, tr_{i,j,ln})$ – the sequence of moments the determined amounts of subsequent non renewable resources are required by activity $O_{i,j}$: $tr_{i,j,k}$ – the time counted since the moment $x_{i,j}$ the $dp_{i,j,k}$ amount of the k-th non renewable resource was released by the activity $O_{i,j}$. That is assumed a resource is collected by activity during its execution: $0 \leq tr_{i,j,k} < t_{i,j}$; $k = 1, 2, \ldots, ln$,

$Ts_{i,j} = (ts_{i,j,1}, ts_{i,j,2}, \ldots, ts_{i,j,ln})$ – the sequence of moments the determined amounts of subsequent non renewable resources are generated (released) by activity $O_{i,j}$: $ts_{i,j,k}$ – the time counted since the moment $x_{i,j}$ the $cs_{i,j,k}$. amount of the k-th non renewable resource was generated by the activity $O_{i,j}$. That is assumed the resource is generated during activity execution, however not earlier than beginning of its collection, i.e.: $0 \leq ts_{i,j,k} < t_{i,j}$; $k = 1, 2, \ldots, ln$, as well as $tr_{i,j,k} \leq ts_{j,k}$; $k = 1, 2, \ldots, ln$.

Consequently, each activity $O_{i,j}$ is specified by the following sequences of:

- starting times of activities in the activity network P_i:

$$X_i = (x_{i,1}, x_{i,2}, \ldots x_{i,lo_i}), \quad 0 \leq x_{i,j} < h, \quad i = 1, 2, \ldots, lp; \quad j = 1, 2, \ldots, lo_i,$$

- duration of activities in the activity network P_i:

$$T_i = (t_{i,1}, t_{i,2}, \ldots, t_{i,lo_i}).$$

Elements of sequences: $Tp_{i,j}$, $Tz_{i,j}$, $Dp_{i,j}$, $Cr_{i,j}$, $Cs_{i,j}$, $Tr_{i,j}$, $Ts_{i,j}$ specify the activity network P_i:

- starting times the$j$-th resource is allocated to the k-th activity in the activity network P_i:

$$TP_{i,j} = (tp_{i,1,j}, \ldots, tp_{i,k,j}, \ldots, tp_{i,lo_i,j}),$$

- starting times the j-th resource is released by the k-th activity in the P_i:

$$TZ_{i,j} = (tz_{i,1,j}, \ldots, tz_{i,k,j}, \ldots, tr_{i,lo_i,j}),$$

- the sequence of moments the j-th non-renewable resource is collected by activities of the projects P_i:

$$TS_{i,j} = (ts_{i,1,j}, \ldots, ts_{i,k,j}, \ldots, ts_{i,lo_i,j}),$$

- amounts of the j-th resources allotted to the k-th activity in the route P_i:

$$DP_{i,j} = (dp_{i,1,j}, \ldots, dp_{i,k,j}, \ldots, dp_{i,lo_i,j}).$$

- sequences of amounts of the $j-$th non-renewable resource consumed by activities of the route P_i:
$$CR_{i,j} = (cr_{i,1,j}, \ldots, cr_{i,k,j}, \ldots, cr_{i,lo_i,j}),$$
- sequences of amounts of the $j-$th non-renewable resource involved by activities of the route P_i:
$$CS_{i,j} = (cs_{i,1,j}, \ldots, cs_{i,k,j}, \ldots, cs_{i,lo_i,j}).$$

Constraints

Given projects portfolio and available amounts of renewable and non-renewable resources as well as the above mentioned sequences: T_i, $TP_{i,j}$, $TZ_{i,j}$, $DP_{i,j}$.

Given the time horizon $H = \{0, 1, \ldots, h\}$, the projects portfolio should be completed. That is assumed the activities cannot be suspended during their execution, and moreover:

- each activity can request any kind and quantity (not exceeding the resource's limited amount) of any resource
- each resource can be uniquely used by an activity,
- the quantity of renewable resource used by an activity cannot be changed or allotted to other activity,
- an activity can start its execution only if required amounts of renewable and non-renewable resources are available at the moments given by $Tp_{i,j}$, $Ts_{i,j}$.

The project P_i is represented by activity-on-node networks, where nodes represent activities and arcs determine an order of activities execution. Consequently, the following activities order constraints are considered:

- the k-th activity follows the i-th one:
$$x_{i,j} + t_{ij} \leq x_{i,k} \tag{2}$$
- the k-th activity follows other activities:
$$x_{i,j} + t_{i,j} \leq x_{i,k}, \quad x_{i,j+1} + t_{i,j+1} \leq x_{i,k}, \quad x_{i,j+2} + t_{i,j+2} \leq x_{i,k}, \quad \ldots, \quad x_{i,j+n} + t_{i,j+n} \leq x_{i,k} \tag{3}$$
- the k-th activity is followed by other activities:
$$x_{i,k} + t_{i,k} \leq x_{i,j}, \quad x_{i,k} + t_{i,k} \leq x_{i,j+1}, \quad x_{i,k} + t_{i,k} \leq x_{i,j+2}, \quad \ldots, \quad x_{i,k} + t_{i,k} + \leq x_{i,j+n} \tag{4}$$

In general case, depending on the routine question context the different (so called query oriented) reference models of decision problems can be considered. For instance, such models can be aimed at:

- a goal function values implied by values of assumed decision variables,
- the values of assumed decision variables guaranteeing expected values of the goal functions,
- the variables and/or constraints guaranteeing the given values of decision variables imply assumed values of the goal functions considered (Bach et al., 2008b).

3. CONSTRAINT SATISFACTION PROBLEM

Constraint programming (*CP*) is an emergent software technology for declarative description and effective solving of large combinatorial problems, especially in the areas of integrated production planning. Since a constraint can be treated as a logical relation among several variables, each one taking a value in a given (usually discrete) domain, the idea of *CP* is to solve problems by stating the requirements (constraints) that specify a problem at hand, and then finding a solution satisfying all the constraints (Banaszak et al., 2005). Because of its declarative nature, it is particularly useful for applications where it is enough to state *what* has to be solved instead *how* to solve it (Banaszak et al., 2005). More formally, *CP* is a framework for solving combinatorial problems specified by pairs: **a set of variables and associated domains, a set of constraints restricting the possible combinations of the values of the variables.** So, the constraint satisfaction problem (*CSP*) (Banaszak et al., 2005) is defined as follows: $CS = ((A, D), C)$, where: $A = \{a_1, a_2, \ldots, a_g\}$ – a finite set of discrete decision variables, $D = \{D_i | D_i = \{d_{i,1}, d_{i,2}, \ldots, d_{i,,j}, \ldots, d_{i,,ld}\}$, $i = 1, \ldots, g\}$ – a family of finite variable domains and the finite set of constraints $C = \{C_i | i = 1, \ldots, L\}$ – a finite set of constraints limiting the variables domain. The solution to the CS is a vector $(d_{1,i}, d_{2,k}, \ldots d_{n,j})$ such that the entry assignments satisfy all the constraints C. So, the task is to find the values of variables satisfying all the constraints, i.e., a feasible valuation.

The inference engine consists of the following two components: constraint propagation and variable distribution. Constraints propagation uses constraints actively to prune the search space. The aim of propagation techniques, i.e., local consistency checking, is to reach a certain level of consistency in order to accelerate search procedures by drastically reducing the size of the search tree (Banaszak, 2006). The constraints propagation executes almost immediately. What limits the size of the problem in practical terms is the variable distribution phase, which employs the backtracking-based search and is very time consuming as a result.

The declarative character of *CP* languages and their high efficiency in solving combinatorial problems offer an attractive alternative to the currently available DSSs that employ operation research techniques.

4. SUFFICIENT CONDITIONS FOT RESOURCES ALLOCATION

4.1. CONSTRAINTS GUARANTEEING THE DEADLOCKS AVOIDANCE IN RENEWABLE RESOURCES ENVIRONMENT

Due to the earlier mentioned, the resources conflict may be observed in cases the concurrently executed processes (activities) compete with access to the limited amount of common shared resources. In dependence on priorities determining an order in which the processes access to shared resources as well as a way of resources allocation a deadlock phenomenon may occur or not. In order to illustrate such possibility let us consider three activities $O_{1,1}$, $O_{1,2}$ and $O_{1,3}$ execution of which requires some amounts

of two among three different resources ro_1, ro_2, ro_3, i.e. the activity $O_{1,1}$ requires the resources ro_1, ro_2, the activity $O_{1,2}$ requires the resources ro_2, ro_3, and the activity $O_{1,3}$ requires the resources ro_3, ro_2 (see Fig. 2a)). Assumed limits determining the amount of available resources equals to 4 units and are the same within the all time horizon H, i.e., $zo_{1,u} = zo_{2,u} = zo_{3,u} = 4$, for $u = 1, 2, \ldots, h+1$.

All activities start at the same moment u. Each activity $O_{1,1}$, $O_{1,2}$ and $O_{1,3}$, uses 4 resources units at its beginning. At the u+1-th moment the activity $O_{1,1}$, requires 4 units of resource ro_2 (actually occupied by the activity $O_{1,2}$) necessary for continuation of its execution. So, the activity $O_{1,1}$ has to suspend its execution till the activity $O_{1,2}$ releases the resource ro_2. In turn, at the $u+2$-th moment the activity $O_{1,2}$ requests 4 units of the resource ro_3 (at this moment occupied by the operation $O_{1,3}$) necessary for continuation of its execution. Similar to the case of activity $O_{1,1}$, the activity $O_{1,2}$, has to suspend its execution till the activity $O_{1,3}$releases the resource ro_3. Finally, at the $u+3$-th moment the activity $O_{1,3}$, requests 4 units of the resource ro_1occupied by the activity $O_{1,1}$. Consequently, the activity $O_{1,3}$, waits for the end of the activity $O_{1,2}$ which is waiting for completion of the activity $O_{1,3}$. So, the observed at the u+3-th moment, the closed loop of resources requests results in deadlock activities (see Fig. 2b).

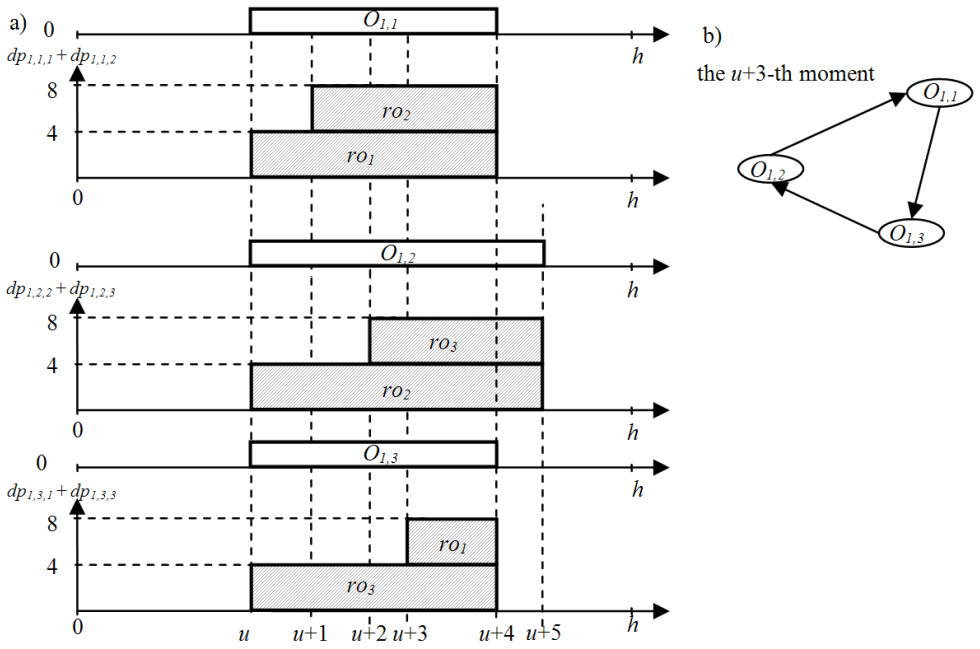

Fig. 2. Illustration of the deadlock occurrence following constraints imposed by the renewable resources allocation.

Presented illustration shows that the deadlock occurrence at the $u+3$-th moment follows from activities requirements exceeding amount of available resources. So, in order to be able to develop the constraints allowing one to avoid the exceeding of assumed limits imposed on available amount of renewable resources, let us consider the following functions $f_k(u, X)$ and$g_k(u)$: determining available limits of the k-th resource units required at the moment u:

- $f_k(u, X) \geq 0$, $u \in H$ – determines the required number of the k-th resource units at the moment, which depends on assumed moments of activities beginning $X = (X_1, X_2, \ldots, X_{lp})$;
- $g_k(u) \geq 0$, $u \in H$ – determines the available amount of the k-th resource at the moment;

The changes of functions f_k, g_k values following execution of activities from the Fig. 2 are shown in Fig. 3. In case considered, at the moments i+1, i+2, i+3, resources requests (encompassed by functions f_2, f_1, f_3) exceed assumed limits of resources ro_2, ro_1, ro_3, respectively. Periods in which the quantities of required resource exceed an available resources amount are distinguished by shadowed area. That is easy to note the deadlock state follows from the situation where for each resource the function $f_k(u,X)$ values (in general f_1, f_2, f_3) are greater than values of the $g_k(u)$ function, i.e. $f_k(u, X) > g_k(u)$ holds. Such cases lead to the closed loop of resources request. In lour case, such situation takes place at the $i + 3$-th moment.

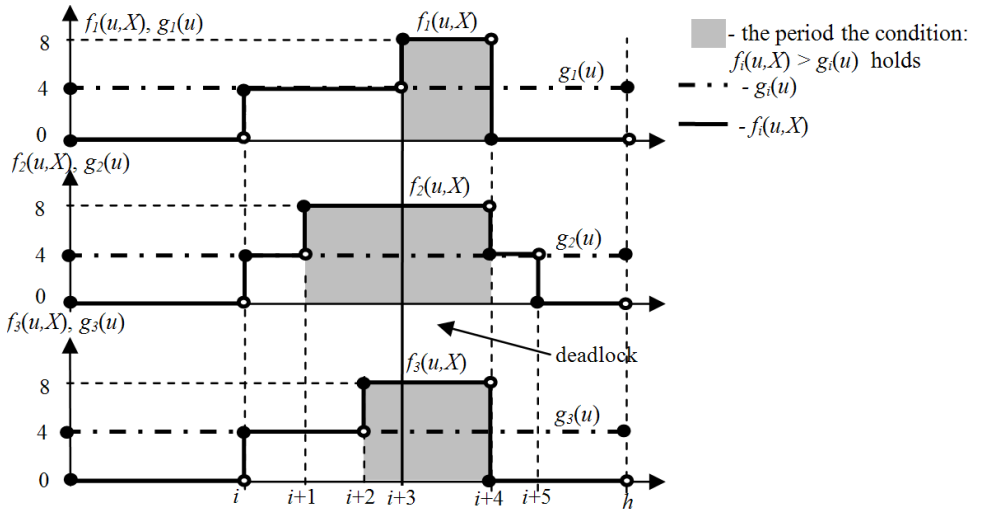

Fig. 3. Illustration of the function g_k and f_k changes following activities from Fig. 2.

The above observation leads to conclusion that occurrence of the closed loop of resources request implies the following condition holds $f_k(u_b, X) > g_k(u_b)$. That results in the following Property 1.

Property 1. *The inequality* $f_k(u_b, X) > g_k(u_b)$ *is a necessary condition for the occurrence of closed loops of resources request.*

Moreover, assuming that activities cannot be stopped (suspended) during their execution, the following Lemma 1 holds.

Lemma 1. *If resources allocation to activities in the projects portfolio* P *at the moment ufollow the condition* $f_k(u, X) \leq g_k(u)$, $\forall k \in \{1, 2, \ldots, lz\}$, *for assumed* X, T_i, $TP_{i,j}$, $TZ_{i,j}$, $DP_{i,j}$, H, *then activities execution does not lead to the deadlocks.*

Proof. The proof directly follows from the Property 1. Due to the Property 1 a deadlock cannot occur at the moment v only in the case when the condition $f_k(v, X) > g_k(v)$ holds at least for one resource. So, in case for each resource ($\forall k \in \{1, 2, \ldots, lz\}$) the Property 1 does not hold (i.e., the condition $f_k(v, X) > g_k(v)$ does not hold), i.e. the closed cycle does not occur. The case the Property 1 does not hold means the condition $f_k(v, X) \leq g_k(v)$ holds. So, the closed cycle (i.e. a deadlock) does not occur, in case the condition $f_k(v, X) \leq g_k(v)$ holds for each k-th resource, i.e. $\forall k \in \{1, 2, \ldots, lz\}$. □

Property 2. *If at any moment* u *within the time horizon* H *considered the following condition holds* $f_k(u, X) \leq g_k(u)$, $\forall k \in \{1, 2, \ldots, lz\}$, *then activities execution is deadlock-free.*

Due to the above introduced assumptions the functions $f_k(u, X)$ and $g_k(u)$ have the following form:

- $$f_k(u, X = \sum_{i=1}^{lp} \sum_{j=1}^{lo_i} \left[dp_{i,j,k} \cdot \overline{1}(v, x_{i,j} + tp_{i,j,k}, x_{i,j} + tz_{i,j,k}) \right] \tag{5}$$

  where: $tp_{i,j,k} < tz_{i,j,k}$, lp – the number of projects,
  lo_i – the number of activities in the i-th project,
  $dp_{i,j,k}$ – the number of resources of the k-th resource used by the activity $O_{i,j}$,
  $\overline{1}(u, a, b)$– an unary function determining the time of the resource occupation,

  $$\overline{1}(u, a, b) = 1(u - a) - 1(u - b)$$

  where: $1(u)$- the unit step function.

  Let us assume that the parameter a is the characteristic point of the function $\overline{1}(u, a, b)$. In the formulae (5), the characteristic point determines the moments $(x_{i,j} + tp_{i,j,k})$ at which a number of the k-th resource units become allotted to an activity. In further considerations such points will called the characteristic points of the function $f_k(u, X)$. Note that increasing of the function $f_k(u, X)$ value can be done only in characteristic points of this function.
- in case $H = \{0,1,\ldots,h\}$ the function $g_k(u)$ can be determined by the$zo_k = (zo_{k,1}, zo_{k,2},\ldots,zo_{k,h+1})$. That means the value of the function $g_k(u)$ corresponds to the $u + 1$-th element of the sequence zo_k:

  $$g_k(u) = zo_{k,u+1} \tag{6}$$

Therefore, since (5) and (6) hold, hence the following Theorem 1 is true also.

Theorem 1. *Given the projects portfolio. Consider assumptions imposed by the reference model regarding activities specification:* X, T_i, $TP_{i,j}$, $TZ_{i,j}$, $DP_{i,j}$, H, *and functions* $f_k(u, X)$ *and* $g_k(u)$ *following formulas* (5), (6). *If for any moment in assumed time horizon* H *and for each* k*-th resource* $k \in \{1, 2, \ldots, lz\}$, *conditions* (7) *and* (8) *hold, then projects portfolio execution is a deadlock-free.*

$$\sum_{i=1}^{lp}\sum_{j=1}^{lo_i}\left[dp_{i,j,k}\cdot\overline{1}(x_{m,n}+tp_{m,n,k},\, x_{i,j}+tp_{i,j,k},\, x_{i,j}+tz_{i,j,k})\right] \le zo_{k,x_{m,n}+tp_{m,n}-1} \quad (7)$$
$$\forall (m,n)\in\{(a,b)|a=1,2,\ldots,lp,\ b=1,2,\ldots,lo_a\}$$

$$\sum_{i=1}^{lp}\sum_{j=1}^{lo_i}\left[dp_{i,j,k}\cdot\overline{1}(vg_{k,d},x_{i,j}+tp_{i,j,k},\, x_{i,j}+tz_{i,j,k})\right] \le zo_{k,vp_{k,d}-1} \quad (8)$$
$$\forall d\in\{1,2,\ldots,q\}$$

where: $vg_{k,i}$ *– the* i*-th characteristic point of the function* $g_k(u)$ *(value of argument* u *where function* g_k*,changes value),* q*– number of the characteristic points.*

Proof. Due to the functions $f_k(v, X)$ and $g_k(v)$ (see formulas (5), (6)) the condition $f_k(u, X) \le g_k(u)$ can be stated in the following form:

$$\sum_{i=1}^{lp}\sum_{j=1}^{lo_i}\left[dp_{i,j,k}\cdot\overline{1}(v,x_{i,j}+tp_{i,j,k},\, x_{i,j}+tz_{i,j,k})\right] \le zo_{k,v+1}, \quad \forall v\in H \quad (9)$$

Moreover, due to the Property 2, in order to avoid deadlocks the condition (9) has to hold for any moment v within the time horizon H. Note, the conditions (7) and (8) are generalizations of the inequality (9) for cases the variable follows vaules of characteristic point of the function $f_k(v, X)$ and $g_k(v)$. So, due to the formulae (5), (6) the change of the projects portfolio states can occur. So, if for each k-th resource $k \in \{1, 2, \ldots, lz\}$ the formulae (9) holds in the characteristic points, then it holds for every $v \in H$. That observation due to the Property 2 and the Lemma 1, guaranties the operations execution in whole time horizon (i.e. the projects portfolio execution) is deadlock-free. Therefore, taking into account conditions (7) and (8) there is a guarantee the solution (i.e. admissible projects portfolio execution) of constraint satisfaction problems is free of deadlock. □

4.2. CONSTRAINTS GUARANTEEING THE DEADLOCK AVOIDANCE IN NON-RENEWABLE RESOURCES ENVIRONMENT.

Besides of renewable resources the non-renewable ones (e.g. money) play a crucial role in course of projects portfolio planning. In order to avoid the financial liquidity losses the currently available amount of such resources cannot fall below an assumed level.

Let us assume the activities requiring non-renewable resources my collect (i.e. decreasing their available amount) as well as generate them (i.e. increasing their amount). That means the beginning of some activities results in some costs (e.g. required for their execution) as well as benefits (e.g. following from their execution). In that context the financial liquidity of projects portfolio execution follows from such allocation of non-renewable resources (e.g. money) which guarantees their amount not fall below an assumed level (e.g. equal to 0). Note that if amounts of available resources fall below an assumed level then may not allow either to start or to complete other activities execution.

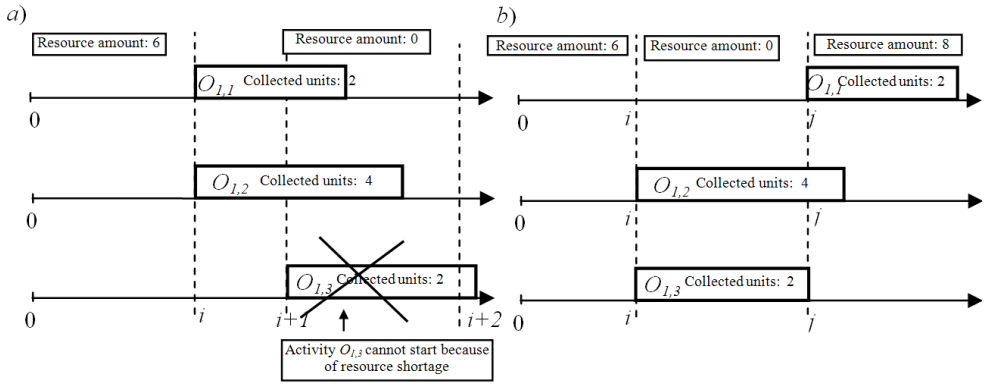

Fig. 4. Illustration of a deadlock leading (a), and a deadlock-free (b) strategies of resource allocation

For illustration let us consider activities $O_{1,1}$, $O_{1,2}$, and $O_{1,3}$, requiring at the i-th moment the following quantities 2, 4, 2 of the resource units (see Fig. 4a). Let us assume the amount of the considered non-renewable resource is equal to 6 units. The activity $O_{1,3}$ generates at its end 10 resource units while the rest of all resources do not generate any benefits. Let us consider two cases assuming the activities may start their execution at the same moment i. In case, the activities $O_{1,1}$, $O_{1,2}$, begin their execution at the i-th moment, the execution of the activity $O_{1,3}$become suspended for ever. However, in case the resource units are allocated to activities $O_{1,2}$, $O_{1,3}$, the activity $O_{1,1}$ can start its execution at the j-th moment, i.e. after the completion of the activity $O_{1,3}$. That means, the constraints allowing one to avoid decisions leading to such kind of deadlock are of main importance.

By analogy to renewable resources the constraints determining a way of non-renewable resources allocation guaranteeing deadlock-free execution of activities can be considered. Let us assume the number of required and generated units of the k-th non-renewable resource is determined by functions $b_k(v, X)$,$m_k(v, X)$ respectively:

- $b_k(v, X) \geq 0$, $\forall v \in H$ – the function determining total amount of required units

of the k-th resource at the moment v, taking into account moments of of activities beginning $X = (X_1, X_2, \ldots, X_{lp})$.

- $m_k(v, X) \geq 0$, $\forall v \in H$ – the function determining total amount of used up units of the$k$-th resource at the moment v, taking into account moments of activities beginning $X = (X_1, X_2, \ldots, X_{lp})$.

The illustration of functions b_k and m_k changes following the activities execution from Fig. 4 is shown in Fig. 5. The periods distinguished by shadowed areas contains the moments the quantity of required resources exceed currently available amount of non-renewable resources. So, the states of deadlock correspond to moments when the values of the function $b_k(v,\ X)$ are greater than the relevant values of the function $m_k(v, X)$, i.e. $b_k(v, X) > m_k(v, X)$. The case considered is shown in Fig. 5 a), see the $i + 1$-th moment. In turn, the situation shown in Fig. 5 b) (i.e. corresponding to the resources allocation from the Fig. 4 b) an order the activities are executed is deadlock-free, i.e. $b_k(v, X) \leq m_k(v, X)$ for every $v \in H$. That means the deadlock state occurrence in the non-renewable resources environment implies $b_k(v_b, X) > m_k(v_b, X)$ for $v_b \in H$. That observation leads to the following Property 3.

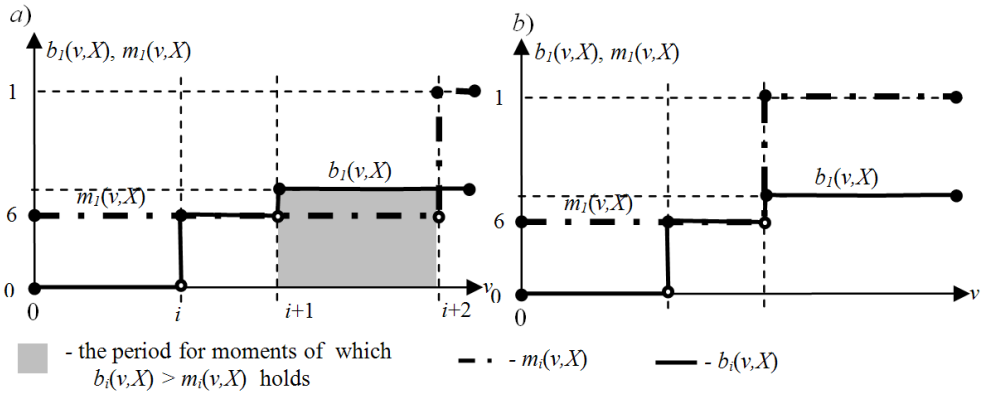

Fig. 5. Illustration of functions b_k and m_k changes following the activities execution from Fig. 4

Property 3. *The inequality $b_k(v_b, X) > m_k(v_b, X)$ is a necessary condition for the deadlocks occurrence is a necessary condition for the occurrence of closed loops of non-renewal resources request (i.e. the deadlocks).*

The Property 3 allows proving the following Lemma 2.

Lemma 2. *Assume the reference model is specified by X, T_i, $TP_{i,j}$, $TZ_{i,j}$, $DP_{i,j}$, H. If for the allocation of the non-renewable resources to the projects P activities at the moment v the following condition holds $b_k(v, X) \leq m_k(v, X)$, $\forall k \in \{1, 2, \ldots, lz\}$, then those activities execution is deadlock-free.*

Proof. The proof follows directly from the Property 3. Due to the Property 3 a deadlock can occur, only in case for at least k-th resource the following condition $b_k(v, X) > m_k(v, X)$ holds, at the moment v. That means, in case if for any k-th resource ($\forall k \in \{1, 2, \ldots, lz\}$) the Property 3 does not hold (i.e. the condition $b_k(v, X) > m_k(v, X)$ does not hold), then the activates execution is deadlock-free. The case the Property 3 does not hold means the following condition $b_k(v, X) \leq m_k(v, X)$, holds. That implies, the activities execution is deadlock-fee if for each k-th resource ($\forall k \in \{1, 2, \ldots, lz\}$) the following condition $b_k(v, X) \leq m_k(v, X)$ holds. □

Property 4. *If at any moment v in the time horizon H considered the following condition holds $b_k(v, X) \leq m_k(v, X)$ for $\forall k \in \{1, 2, \ldots, lz\}$, then activities execution is deadlock-free.*

Due to the assumptions imposed on the reference model the functions contained by the inequality $b_k(v, X) \leq m_k(v, X)$ are of the following form:

- the function $b_k(v, X)$:

$$b_k(v, X) = \sum_{i=1}^{lp} \sum_{j=1}^{lo_i} [cr_{i,j,k} \cdot 1(v - x_{i,j} - t\ r_{i,j,k})] \tag{10}$$

  where: $tr_{i,j,k}$ – the moment of the k-th non-renewable resource allocation to an activity, lp – the number of projects, lo_i – the number of the i-th project's activities, $cr_{i,j,k}$ – the number of the k-the non-renewable resource units used by the activity $O_{i,j}$, $1(v)$ – the unit step function.

  The parameter a is the characteristic point of the unit function $1(v\text{-a})$. In the formulae (10) where the unit functions are added, the characteristic points determines the moments $x_{i,j} + tr_{i,j,k}$ the activities require presumed quantities of the $k-$th resource units. Such points are called the characteristic points of the function $b_k(v, X)$. By analogy to the case of renewable resources the function $b_k(v, X)$ changes may occur only in such characteristic points.
- the function $m_k(v, X)$:

$$m_k(v, X) = \sum_{i=1}^{lp} \sum_{j=1}^{lo_i} [cs_{i,j,k} \cdot 1(v - x_{i,j} - t\ s_{i,j,k})] + zn_k \tag{11}$$

  where: $ts_{i,j,k}$ – the moment of the k-th non-renewable resource allocation to the activity $O_{i,j}$, lp – the number of projects, lo_i – the number of the i-th project's activities, $cs_{i,j,k}$ – the number of the k-the non-renewable resource units used by the activity $O_{i,j}$, $1(v)$ – the unit step function.

  By analogy to $b_k(v, X)$ the values of$x_{i,j} + ts_{i,j,k}$ are called the characteristic points of the function $m_k(v, X)$.

Due to the functions $b_k(v,X)$, $m_k(v,X)$ (see the formulas (10), (11)) and the Property 4 the following condition $b_k(v,X) \leq m_k(v,X)$ can be transformed to the inequality (12):

$$\begin{aligned} zn_k - \sum_{i=1}^{lp}\sum_{j=1}^{lo_i}[cr_{i,j,k}\cdot 1(v - x_{i,j} - tr_{i,j,k})] + \\ + \sum_{i=1}^{lp}\sum_{j=1}^{lo_i}[cs_{i,j,k}\cdot 1(v - x_{i,j} - ts_{i,j,k})] \geq 0, \qquad \forall v \in H \end{aligned} \tag{12}$$

where: lp – the number of projects, lo_i – the number of the i-th project's activities, $1(v)$ – the unit step function.

That follows from the Lemma 2, the activities execution is deadlock-free, in case the condition (12) holds for each moment v in the time horizon H. Values of the functions $b_k(v,X)$, $m_k(v,X)$ (see the formulas (10), (11)) can change only for variables corresponding to the characteristic points. The variable v in the formulae (12) can be then replaced by a set of relevant characteristic points. Consider the characteristic points of the function $b_k(v,X)$, for which the left side of inequality (12) decreases. Therefore, the following Theorem 2 holds.

Theorem 2. *Given the projects portfolio P. Consider assumptions imposed by the reference model regarding activities specification X, T_i, $TP_{i,j}$, $TZ_{i,j}$, $DP_{i,j}$, H and functions $b_k(v,X)$, $m_k(v,X)$ following formulas* (10), (11). *If for any moment v in assumed time horizon H and for each k-th resource $k \in \{1,2,\ldots,ln\}$ (ln – the number of non-renewable resources), conditions* (13) *hold, then projects portfolio execution is deadlock-free.*

$$\begin{aligned} zn_k - \sum_{i=1}^{lp}\sum_{j=1}^{lo_i}[cr_{i,j,k}\cdot 1(x_{m,n} - x_{i,j} - tr_{i,j,k})] + \\ + \sum_{i=1}^{lp}\sum_{j=1}^{lo_i}[cs_{i,j,k}\cdot 1(x_{m,n} - x_{i,j} - ts_{i,j,k})] \geq 0, \\ \forall (m,n) \in \{(a,b) | a = 1,2,\ldots,lp; b = 1,2,\ldots,lo_a\} \end{aligned} \tag{13}$$

Proof. The conditions (13) can be seen as a generalization of the inequality (12) at moments v corresponding to the characteristic points of functions $b_k(v,X)$,$m_k(v,X)$. Due o the formulas (10), (11), the change of projects portfolio P state can arise only in such characteristic points. Therefore, in the case the inequality (12) holds for every characteristic points (see the function $b_k(v,X)$, such that the left side of this inequality decreases its value, then it holds for any moment v in the time horizon H. That observation due to the Property 4 (and the Lemma 2) provides the guarantee the operations execution in whole time horizon (i.e. the projects portfolio execution taking into account non-renewable resources) is deadlock-free. □

Therefore, the non-renewable resources allocation guaranteeing the assumed level of resources amount will not decreased, i.e. activities execution will deadlock-free, follows from conditions (13).

The constraints developed in this section can be added to the set of constraints of the relevant CSP based reference model. In case considered the constraints enhance the model by a knowledge guaranteeing the solutions obtained encompass assumed qualitative features, i.e. deadlock-free execution of projects portfolio. In general case, the conditions aimed at other features such as resources allocation periodicity and starvation-free activities execution can be developed in similar way. Therefore, the open structure of the reference model provides a natural framework supporting a decision maker in the process of projects portfolio prototyping.

5. ILLUSTRATIVE EXAMPLE

In order to illustrate such possibility let us concentrate on the decision problems formulated for resources allocation conflict resolution. Let us assume the activities compete with the access to the discrete resources of the both kind's renewable and non- renewable ones. As a programming environment enabling the reference model implementation the OzMozart language is used (Schutle et al., 1998).

5.1. ROUTINE QUERIES FORMULATED IN THE STRAIGHT WAY.

Example 1

Given the following projects portfolio, i.e. the set of projects $P = \{P_1,P_2,P_3,P_4\}$. Activities $O_{i,j}$ of projects are specified by corresponding sets: $P_1 = \{O_{1,1},\ldots,O_{1,10}\}$, $P_2 = \{O_{2,1},\ldots,O_{2,12}\}$, $P_3 = \{O_{3,1},\ldots,O_{3,11}\}$, $P_4 = \{O_{4,1},\ldots,O_{4,13}\}$. The relevant activity networks (Bach et al., 2008b) are shown on the following figures: Fig. 6, Fig. 7, Fig. 8, and Fig. 9.

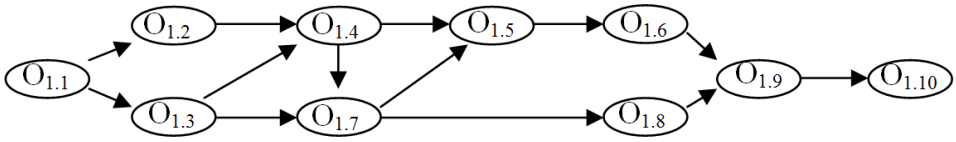

Fig. 6. Activity network of the project P_1

Given the time horizon $H = \{0,1,\ldots,40\}$. Operation times for particular projects P_1, P_2, P_3, P_4 are determined by the following sequences:

$$T_1 = (1,2,3,4,4,8,3,2,1,6), \qquad T_2 = (3,1,6,3,2,5,1,5,2,4,2,1),$$
$$T_3 = (3,7,2,7,2,1,8,3,3,4,8), \qquad T_4 = (3,3,2,8,3,1,4,1,8,4,3,3,8).$$

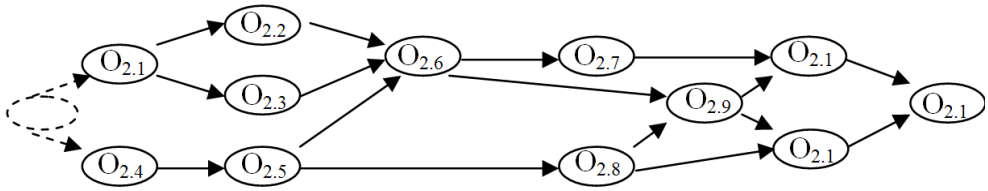

Fig. 7. Activity network of the project P_2

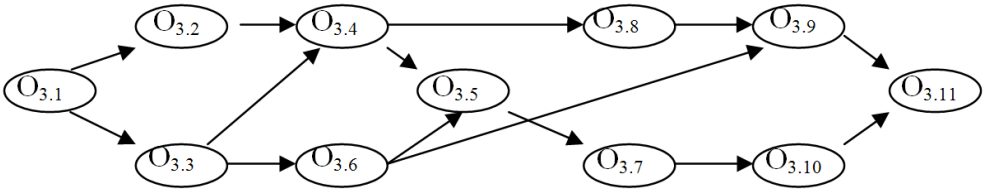

Fig. 8. Activity network of the project P_3

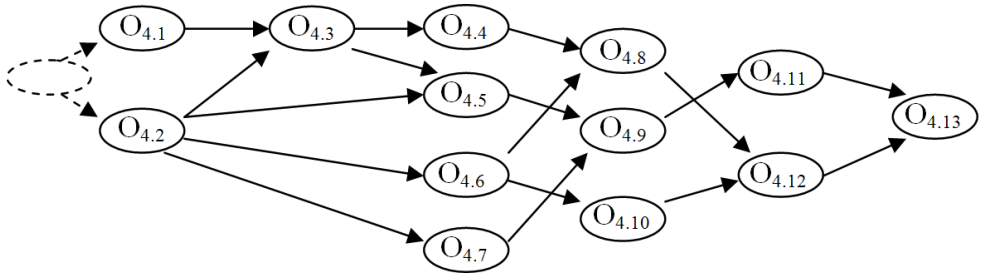

Fig. 9. Activity network of the project P_4

Given are three kinds of renewable resources ro_1, ro_2, ro_3. Resources amounts are limited by following units number: 11, 14, 12 , respectively. Resource amounts are constant in whole time horizon H. That is assumed an amount of resources allocated to the activity at the moment of its beginning can be released only by this activity and only at the moment of its completion. The amounts of particular resources required by projects' P_1, P_2, P_3, P_4 activities are given in the following tables: Table 1, Table 2, Table 3, and Table 4.

It is assumed some activities besides of renewable resources require also non-renewable ones. Given are two kinds of non-renewable resources rn_1, rn_2. Initial amount of the resource rn_1 is equal to 10 units, and of the resource rn_2 is equal to 7 units. Activities may use up and generate some number of resources rn_1, rn_2 units. It

Table 1. Amounts of resources required by activities of the project P_1 (the sequences $DP_{1,1}$, $DP_{1,2}$, $DP_{1,3}$)

	$O_{1,1}$	$O_{1,2}$	$O_{1,3}$	$O_{1,4}$	$O_{1,5}$	$O_{1,6}$	$O_{1,7}$	$O_{1,8}$	$O_{1,9}$	$O_{1,10}$
$DP_{1,1}$	3	1	1	1	1	1	2	1	2	1
$DP_{1,2}$	2	1	2	1	1	2	3	3	1	1
$DP_{1,3}$	2	2	3	1	1	1	1	1	2	1

Table 2. Amounts of resources required by activities of the project P_2 (the sequences $DP_{2,1}$, $DP_{2,2}$, $DP_{2,3}$)

	$O_{3,1}$	$O_{2,2}$	$O_{2,3}$	$O_{2,4}$	$O_{2,5}$	$O_{2,6}$	$O_{2,7}$	$O_{2,8}$	$O_{2,9}$	$O_{2,10}$	$O_{2,11}$	$O_{2,12}$
$DP_{3,1}$	4	3	2	2	1	1	1	3	1	2	2	2
$DP_{3,2}$	1	2	3	1	2	1	2	1	1	2	1	1
$DP_{3,3}$	2	1	1	1	3	1	2	2	2	1	1	1

Table 3. Amounts of resources required by activities of the project P_3 (the sequences $DP_{3,1}$, $DP_{3,2}$, $DP_{3,3}$)

	$O_{3,1}$	$O_{3,2}$	$O_{3,3}$	$O_{3,4}$	$O_{3,5}$	$O_{3,6}$	$O_{3,7}$	$O_{3,8}$	$O_{3,9}$	$O_{3,10}$	$O_{3,11}$
$DP_{3,1}$	2	4	1	2	2	2	1	2	2	1	3
$DP_{3,2}$	2	1	3	2	2	2	1	1	1	2	2
$DP_{3,3}$	2	4	1	2	2	2	1	2	2	1	3

Table 4. Amounts of resources required by activities of the project P_4 (the sequences $DP_{4,1}$, $DP_{4,2}$, $DP_{4,3}$)

	$O_{4,1}$	$O_{4,2}$	$O_{4,3}$	$O_{4,4}$	$O_{4,5}$	$O_{4,6}$	$O_{4,7}$	$O_{4,8}$	$O_{4,9}$	$O_{4,10}$	$O_{4,11}$	$O_{4,12}$	$O_{4,13}$
$DP_{4,1}$	1	2	3	4	3	2	2	1	1	1	3	1	4
$DP_{4,2}$	1	1	1	2	1	2	1	3	2	2	2	1	2
$DP_{4,3}$	1	2	2	1	1	2	4	1	2	2	2	1	2

is assumed each activity uses up some resource units at the beginning and generates some resource units at the activity's end. The amounts of used up and generated resource rn_1 units determine sequences: $CR_{i,j}$, $CS_{i,j}$ respectively in the following tables: Table 5, Table 6, Table 7, and Table 8.

Let us assume each project's efficiency is measured by Net Present Value (NPV) performance index calculated due to the following formulae:

$$NPV = \sum_{t=0}^{n} \frac{CF_t}{(1+k)^t},$$

where:

CF_t – the money netto flow expected in the year t,

k – the discount rate (alternative capital investment cost),

n – the period of a project exploitation [years].

Table 5. Amount of used up (CR) and generated (CS) non-renewable resources required by activities of the project P_1 (the sequences $CR_{1,1}$, $CR_{1,2}$, $CS_{1,1}$, $CS_{1,2}$)

	$O_{1,1}$	$O_{1,2}$	$O_{1,3}$	$O_{1,4}$	$O_{1,5}$	$O_{1,6}$	$O_{1,7}$	$O_{1,8}$	$O_{1,9}$	$O_{1,10}$
$CR_{1,1}$	1	1	2	1	2	1	3	1	1	1
$CR_{1,2}$	1	2	1	1	1	0	1	0	1	1
$CS_{1,1}$	3	2	0	2	4	4	2	0	2	4
$CS_{1,2}$	1	2	3	2	2	2	0	2	1	2

Table 6. Amount of used up (CR) and generated (CS) non-renewable resources required by activities of the project P_2 (the sequences $CR_{2,1}$, $CR_{2,2}$, $CS_{2,1}$, $CS_{2,2}$)

	$O_{2,1}$	$O_{2,2}$	$O_{2,3}$	$O_{2,4}$	$O_{2,5}$	$O_{2,6}$	$O_{2,7}$	$O_{2,8}$	$O_{2,9}$	$O_{2,10}$	$O_{2,11}$	$O_{2,12}$
$CR_{2,1}$	1	0	1	2	1	1	1	3	1	0	1	1
$CR_{2,2}$	3	2	1	2	0	2	3	2	2	2	1	2
$CS_{2,1}$	3	2	0	2	1	2	0	2	0	2	0	1
$CS_{2,2}$	3	2	1	2	0	2	3	2	2	2	1	2

Table 7. Amount of used up (CR) and generated (CS) non-renewable resources required by activities of the project P_3 (the sequences $CR_{3,1}$, $CR_{3,2}$, $CS_{3,1}$, $CS_{3,2}$)

	$O_{3,1}$	$O_{3,2}$	$O_{3,3}$	$O_{3,4}$	$O_{3,5}$	$O_{3,6}$	$O_{3,7}$	$O_{3,8}$	$O_{3,9}$	$O_{3,10}$	$O_{3,11}$
$CR_{3,1}$	1	1	2	1	1	1	0	1	3	1	1
$CR_{3,2}$	0	1	1	0	2	1	1	1	3	1	0
$CS_{3,1}$	2	3	2	0	2	1	2	2	2	3	2
$CS_{3,2}$	3	2	1	2	0	2	3	2	2	2	1

Table 8. Amount of used up (CR) and generated (CS) non-renewable resources required by activities of the project P_4 (the sequences $CR_{4,1}$, $CR_{4,2}$, $CS_{4,1}$, $CS_{4,2}$)

	$O_{4,1}$	$O_{4,2}$	$O_{4,3}$	$O_{4,4}$	$O_{4,5}$	$O_{4,6}$	$O_{4,7}$	$O_{4,8}$	$O_{4,9}$	$O_{4,10}$	$O_{4,11}$	$O_{4,12}$	$O_{4,13}$
$CR_{4,1}$	1	1	2	1	1	1	0	1	3	1	1	1	1
$CR_{4,2}$	0	1	1	0	2	1	1	1	3	1	0	1	1
$CS_{4,1}$	2	3	2	0	2	1	2	2	2	3	2	3	2
$CS_{4,2}$	3	2	1	2	0	2	3	2	2	2	1	2	2

The problem considered belongs to the class of „straight" ones where for a given parameters describing the "enterprise - portfolio projects" system the activities schedule following assumed makespan limits is sought. It reduces to the following question: Does there exist a schedule following constraints assumed on availability of renewable and non-renewable resources and $NPV > 0$ such that production orders completion time not exceeds the deadline h? Note that the schedule we are looking for is determined by moments $x_{i,j}$ the activities start their execution (Beale, 1979; Bocewicz et al., 2007).

Solution to the problem results in determination of moments the activities start their execution $x_{i,j}$ (Linderoth et al., 1999). So, the solution we are searching for has the form of the following sequences: $X_1 = (x_{1,1}, \ldots, x_{1,10})$, $X_2 = (x_{2,1}, \ldots, x_{2,12})$, $X_3 = (x_{3,1}, \ldots, x_{3,11})$, $X_4 = (x_{4,1}, \ldots, x_{4,13})$.

Of course, the elements of sequences X_1, X_2, X_3, X_4 have to follow activities order constraints from Fig. 6 – Fig. 9 as well as constraints assumed on renewable (see tables, Table 1 – Table 4) and non-renewable (see tables Table 5 – Table 8) resources allocation (guaranteeing deadlock avoidance). Constraints have a form similar to the formulas (2), (3), (4), (7), (8), and (13). The obtained solution, i.e. sequences X_1, X_2, X_3, X_4, follows from model implementation in the CSP-based reference model and programmed in OzMozrat (Puget, 1994).

The first admissible solution has the following form:

$$X_1 = (0, 1, 1, 4, 11, 15, 8, 11, 23, 24), \qquad X_2 = (0, 3, 7, 10, 13, 15, 20, 17, 23, 25, 25, 29),$$
$$X_3 = (0, 3, 3, 10, 17, 5, 19, 17, 20, 27, 31), \quad X_4 = (0, 0, 3, 5, 5, 3, 3, 13, 8, 6, 14, 16, 19)$$

The NPV index value calculated for projects: P_1, P_2, P_3, P_4 follows the requirement $NPV > 0$, i.e. $NPV_{P1} = 0,3649$, $NPV_{P2} = 2,4775$, $NPV_{P3} = 1,3248$, $NPV_{P4} = 0,8134$.

The graphical representation of the projects portfolio schedule is show in the Fig. 10. The schedule obtained follows all constrains imposed by an enterprise capability and projects execution requirements. The system considered allows one to obtain the Gantt's-like chart illustrating the rates of resources usage both renewable and non-renewable ones. An example of graphical representation of the resource zo_1 usage rate containing assumed resource's limit in whole time horizon is shown on Fig. 11. It can be observed the assumed resource's limit was not exceeded, the same regards of resources zo_2, zo_3.

In turn, the Fig. 12 illustrates changes regarding the rate of resource usage concerning of the non-renewable resource zn_2. That is easy to note that the assumed minimal level of resource usage equal to 0 was never exceeding in whole time horizon. The same remark concerns the resource zn_1.

Therefore, the example presented illustrates the capability of an interactive multi-criteria project planning (e.g. taking into account a particular project deadline, projects portfolio deadline, resources limits, and so on) approach to projects prototyping problems formulated either in a straight or in a reverse way. The problem of the size just considered took less than 5 minutes (the AMD Athlon(tm)XP 2500 + 1.85 GHz, RAM 1,00 GB platform has been used).

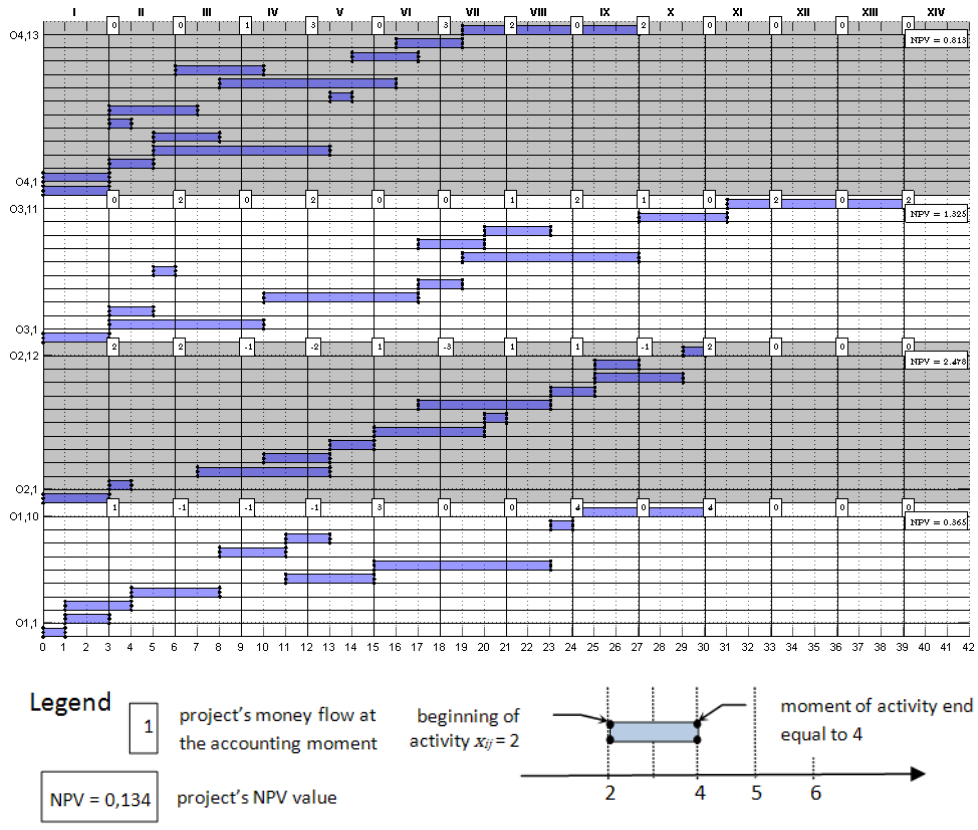

Fig. 10. Projects portfolio schedule

Example 2

Given the following projects portfolio, i.e. the set of projects $P = \{P_1, P_2, P_3, P_4\}$ specified by the same activity networks (see Fig. 6, Fig. 7, Fig. 8 and Fig. 9) and resources allocations (see the Table 2 – Table 9) as in the Example 1. However, the new time horizon $H = \{0, 1, \ldots, 36\}$ is considered.

The problem considered belongs to the class of „straight" ones and reduces to the following question: Does there exist a schedule following constraints assumed on availability of renewable and non-renewable resources and $NPV > 0$ such that production orders completion time not exceeds the deadline h?

Similarly to the previous case the solution to the problem results in determination of the moments activities start their execution $x_{i,j}$. So, the solution we are searching for has the same form of the following sequences: $X_1 = (x_{1,1}, \ldots, x_{1,10})$, $X_2 = (x_{2,1}, \ldots, x_{2,12})$, $X_3 = (x_{3,1}, \ldots, x_{3,11})$, $X_4 = (x_{4,1}, \ldots, x_{4,13})$, however regards of the shorter deadline.

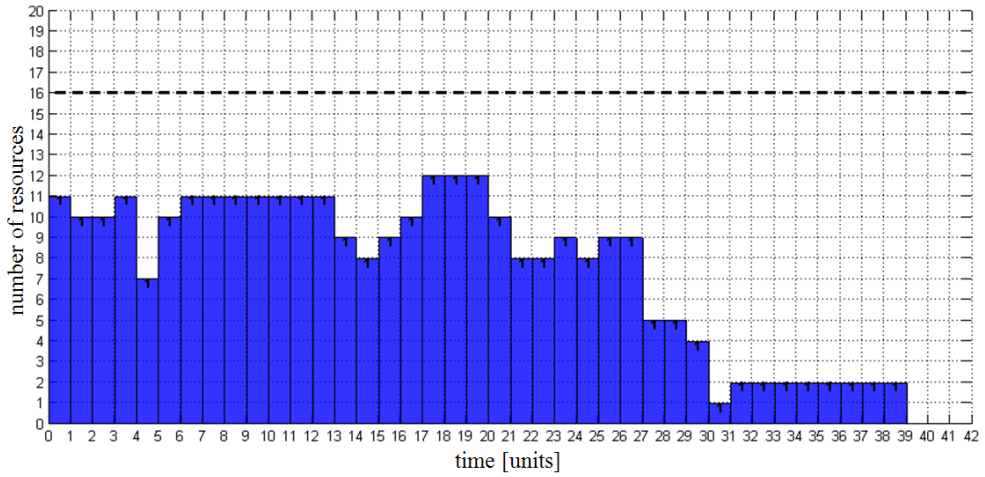

Fig. 11. Gantt's-like chart of the renewable resource zo_1 usage

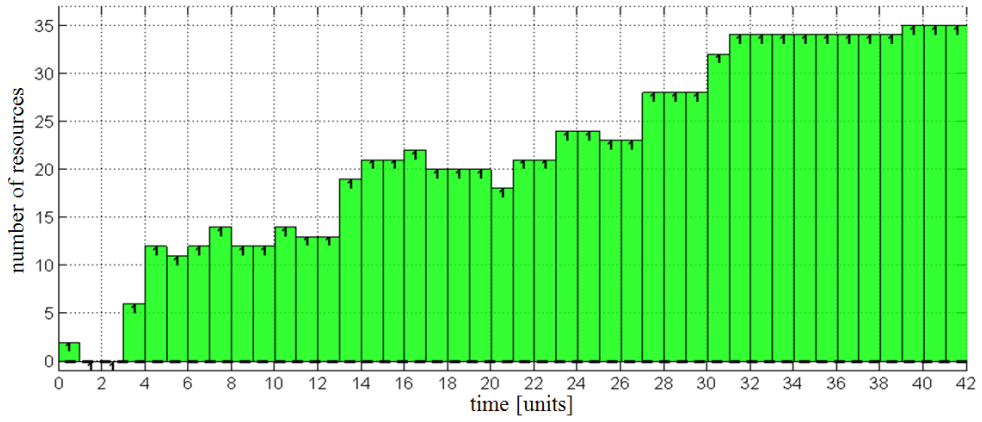

Fig. 12. Gantt's-like chart of the non-renewable resource zn_2 usage

The reference model encompassing assumption of the example considered was implemented in OzMozart programming environment. The solution obtained in 2 s was: „The set of admissible solutions is empty".

That means no schedule there exists. In such situation, however there is still a possibility to reformulate the problem considered by stating it in a reverse way, i.e. the way aimed at searching for decision variables (e.g. activities operation times) guaranteeing the completion time of the considered projects portfolio will not exceed the assumed deadline h. Such case is just considered in the secion below.

5.2. ROUTINE QUERIES FORMULATED IN THE REVERSE WAY.

Consider once again the case from the Example 2. Given the projects port folio containing the following projects P_1, P_2, P_3, P_4. The makespan admissible cannot exceed 36 units of time. Activities operations times are not known, however constraints determining their execution constraints are given. For instance, the following constraint:

$$t_{3,7} + t_{3,9} = 11$$

means the activities operation times are tightly linked, i.e. increasing of activity time associated to one operation (for example to $O_{3,7}$) results in decreasing of the another one (in this case $O_{3,9}$). The set of constraints considered are contained In the Table 9.

Therefore, taking into account above mentioned assumptions as well as the other ones used in the Example 1, the problem considered now reduces to the question: What values and of what variables T_1, T_2, T_3, T_4 guarantee the makespan of the projects portfolio does not exceed a given deadline subject to limits imposed on available amounts of renewable and non-renewable resources as well as $NPV > 0$?

Table 9. Constraints linking activities $O_{i,j}$ execution times

	Constraint		Constraint
C_1	$t_{3,7} + t_{3,9} = 11$	C_6	$t_{3,3} + t_{3,4} = 9$
C_2	$t_{4,12} + t_{4,13} = 11$	C_7	$t_{2,3} + t_{2,4} = 9$
C_3	$t_{4,3} + t_{4,4} = 11$	C_8	$t_{2,3} + t_{2,4} = 9$
C_4	$t_{1,5} + t_{1,6} = 12$	C_9	$2t_{2,5} + t_{3,3} = 8$
C_5	$t_{1,9} + t_{1,10} = 7$	C_{10}	$2t_{4,1} + t_{2,8} = 12$

In order to response to this question the values of the following sequences are sought: $T_1 = (t_{1,1}, \ldots, t_{1,10}), T_2 = (t_{2,1}, \ldots, t_{2,12}), T_3 = (t_{3,1}, \ldots, t_{3,11})$, $T_4 = (t_{4,1}, \ldots, t_{4,13})$ and $X_1 = (x_{1,1}, \ldots, x_{1,10})$, $X_2 = (x_{2,1}, \ldots, x_{2,12})$, $X_3 = (x_{3,1}, \ldots, x_{3,11})$, $X_4 = (x_{4,1}, \ldots, x_{4,13})$.

The OzMozart based implementation of the reference model considered results in the following solution obtained in 250 s. So, the sequences of activities operation times are as follows:

$$T_1 = (1,2,3,4,6,6,3,2,3,4), \qquad T_2 = (3,1,6,3,2,5,1,6,2,4,2,1),$$
$$T_3 = (3,7,4,5,2,1,6,3,5,4,8), \qquad T_4 = (3,3,2,5,6,1,4,1,8,4,3,5,6).$$

In turn, the sequences of the moments of activities beginning are as follows:

$$X_1 = (0,1,1,4,11,17,8,11,23,26), \qquad X_2 = (0,3,8,10,13,15,20,15,21,23,23,27),$$
$$X_3 = (0,3,3,10,15,7,17,15,18,23,27), \quad X_4 = (0,0,3,5,5,3,3,10,11,4,11,19,24,).$$

The NPV index value calculated for projects: P_1, P_2, P_3, P_4 follow the requirement $NPV > 0$, i.e $NPV_{P1} = 0{,}262$, $NPV_{P2} = 2{,}386$, $NPV_{P3} = 0{,}86$, $NPV_{P4} = 1{,}339$.

The graphical illustration of the solution obtained is shown in Fig. 13, where on the base of the Gantt's chart an admissible activities schedule, i.e. the beginning and

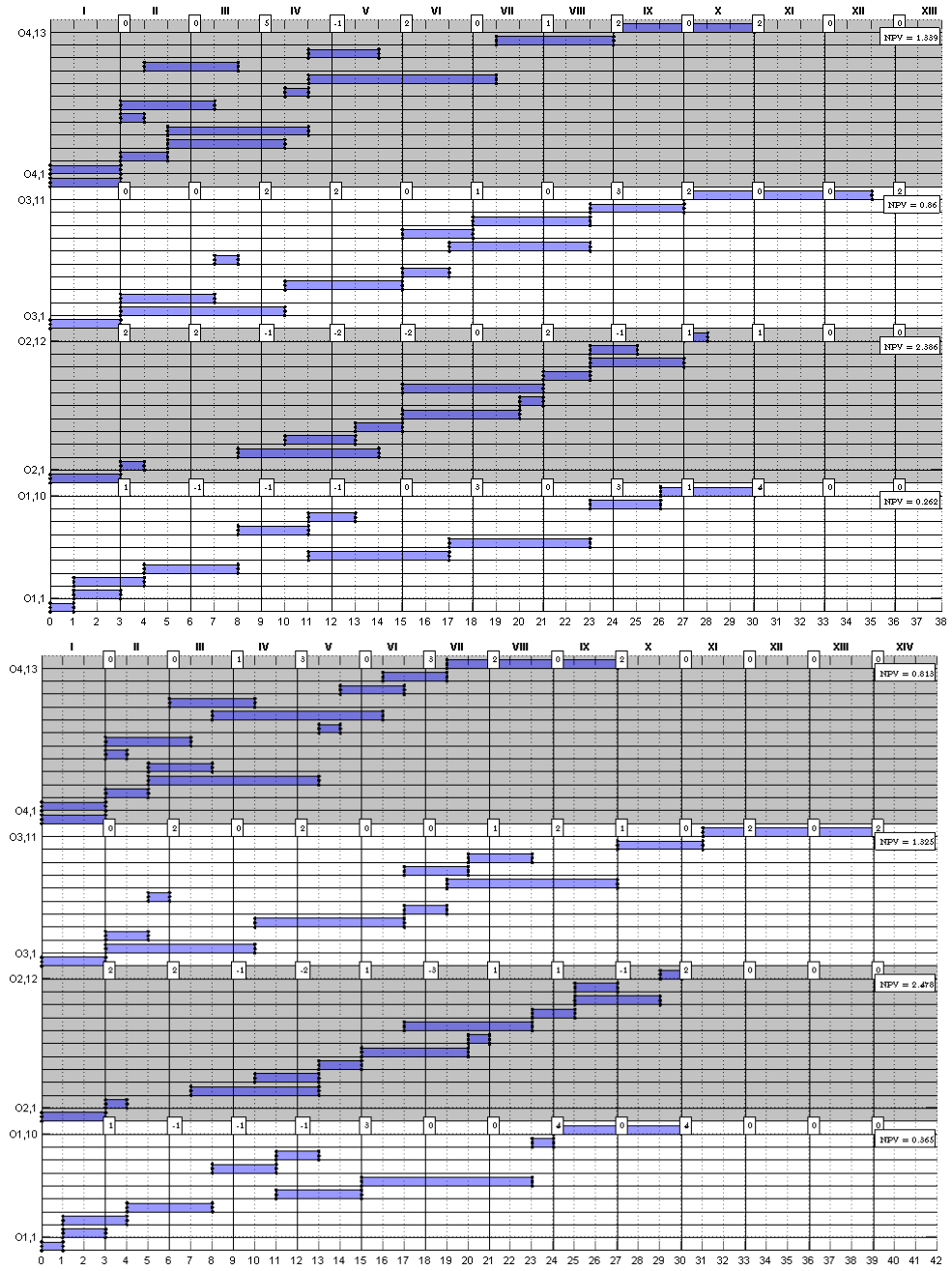

Fig. 13. The Gantt's chart of the projects portfolio execution

duration of activities following all the constraints imposed by availability as well as allocation of renewable and non-renewable resources is provided. The corresponding charts illustrating the changes of resources $zo_1$and zn_2 availability are shown in Fig. 14 and Fig. 15, respectively.

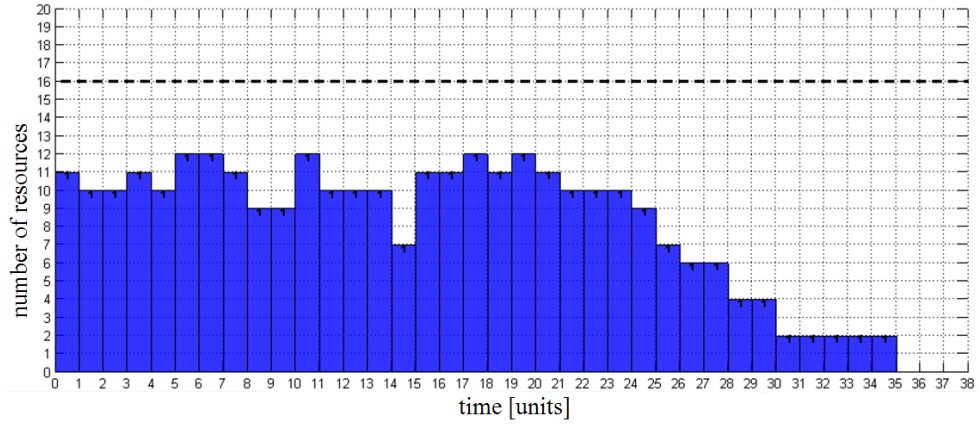

Fig. 14. Illustration of the resource zo_1 changes

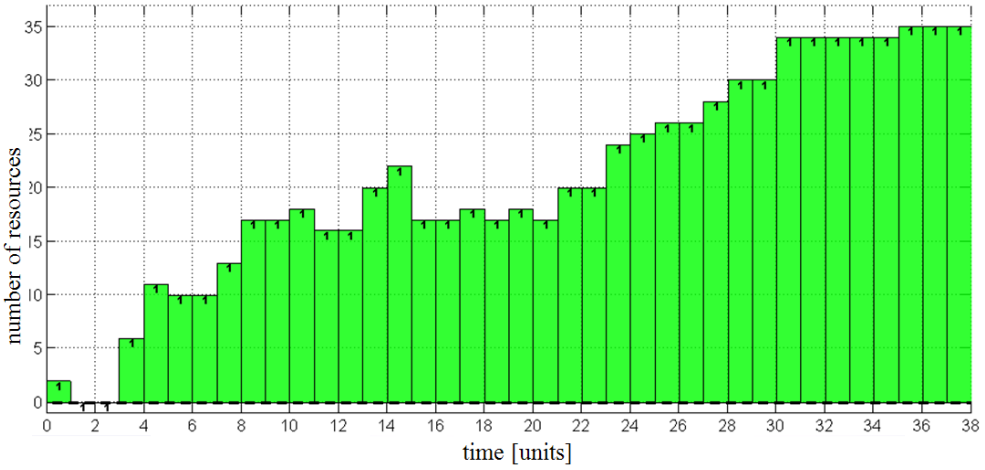

Fig. 15. Illustration of the resource zn_1 changes

The above example shows the possibility of the reference model application to the problem stated in a reverse way.

6. CONCLUDING REMARKS

The introduced CP-based reference model provides a formal framework allowing one to formulate the projects portfolio planning problems in direct and reverse way. In other words it provides a base to an interactive task oriented decision support tools designing. More precisely it provides the framework allowing one to take into account both direct and inverse problem formulation. These offers a possibility to respond to the questions like: What values and of what variables guarantee the production orders will complete due to assumed values of performance indexes? (beside such standard questions as: Is it possible to complete a given set of production orders at a scheduled project deadline?).

Two kinds of resources, i.e. renewable and non-renewable ones can be allocated to projects activities. Besides of constraints following from the SME and projects portfolio specification the constraints proposed by experts and/or developed by a decision makers can be easily introduced (e.g. see conditions guaranteeing deadlock-free activities execution). The direct and reverse formulations of decision problems (size of that typical for SMEs) can be easily implemented in OzMozart language and then resolved in an on-line mode.

Moreover, above mentioned kinds of problems can be formulated in an uniform way, i.e. by distinguishing discrete, distinct and imprecise variables, domains of variables, and constraints linking subsets of variables.

The reference model can be seen as a knowledge base encompassing the structure of a constraint satisfaction problem, where the logic-algebraic method (Bach et al., 2008a; Bocewicz et al., 2007) plays a role of the inference engine. The main idea standing behind of this approach lies in searching for the conditions guaranteeing the existence of responses to the standard queries as well for conditions guaranteeing the employed search strategies cab be used in on-line mode for the given size of project planning problems. Therefore, the reference model of decision problems can be seen as a kernel of a methodology aimed at designing of dedicated and interactive decision support systems.

REFERENCES

Bach I., Bocewicz G., Banaszak Z., 2008a: Constraint Programming Approach to Time-window and Multiresource-Constrained Projects Portfolio Prototyping. In: *Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2008*, N.T. Nguyen et al. (Eds.):, *Lecture Notes in Artificial Intelligence 5027*, Springer-Verlag, Berlin, Heidelberg, pp. 767–776.

Bach I., Wójcik R., Bocewicz G, 2008b: Projects Portfolio Prototyping Subject to Imprecise Activities Specification, In: *Conference proceedings of 14^{th} International Congress of Cybernetics and Systems of WOSC – ICCS'08, Wroclaw*, 261–272.

Banaszak Z., 2006: CP-based Decision Support for Project-driven Manufacturing. In: *Perspectives in Modern Project Scheduling*, (Józefowska J. and J. Węglarz (Ed)), Interna-

tional Series in Operations Research and Management Science, Vol. 92, Springer Verlag, New York, 409–437.

Banaszak Z., Zaremba M., Muszyński W., 2005: CP-based Decision Making for SME. In: *Preprints of the 16th IFAC World Congress* (Eds P. Horacek, M. Simandl), P. Zitek, DVD, Prague, Czech Republic.

Barták R., 2004: Incomplete Depth-First Search Techniques: A Short Survey, *Proceedings of the 6th Workshop on Constraint Programming for Decision and Control*, Ed. Figwer J., p. 7–14.

Beale E.M.L., 1979: Branch and Bound Methods for Mathematical Programming Systems. In: P. L. Hammer, E. L. Johnson, and B. H. Korte, editors, *Discrete Optimization II*, North Holland Publishing Co., 201–219.

Bocewicz G., Banaszak Z., Wójcik R., 2007: Design of Admissible Schedules for AGV Systems with Constraints: a Logic-algebraic Approach, *Agent and Multi-Agent Systems: Technologies and Applications*, Nguyen N.T., Grzech A., Howlett R.J., Jain L.C. (Eds.), *Lecture Notes in Artificial Intelligence* 4496, Springer-Verlag, Berlin, Heidelberg, 578–587.

Linderoth T., Savelsbergh M.W.P., 1999: A Computational Study of Search Strategies in Mixed Integer Programming. *INFORMS Journal on Computing*, 11:173–187.

Chanas S., Komburowski J., 1981: *The Use of Fuzzy Variables in PERT*, Fuzzy Sets and Systems, 5(1), 11–19.

Dubois D., Fargier H., Fortemps P., 2003: Fuzzy scheduling: Modeling Flexible Constraints vs. Coping with Incomplete Knowledge, *European Journal of Operational Research* 147, 231–252.

Martinez, E.C., D., Duje G.A. Perez, 1997: On Performance Modeling of Project-oriented Production. *Computers and Industrial Engineering*, Vol. 32, 509–527.

Puget J-F., 1994: A C++ Implementations of CLP, Proceeding of SPICS 94.

Schutle H., Smolka G., Wurtz J., 1998: Finite Domain Constraint Programming in Oz. *German Research Center for Artificial Intelligence*, Germany, D-66123 Saarbrucken.

Sung. C.S., 1985: A Production Planning Model for Multi-Product Facilities. *Journal of the Operations Research Society of Japan*, Vol. 28, No. 4, pp. 345–385.

Van Hentenryck P., 1991: Constraint Logic Programming, *Knowledge Engineering Review*, 6, 151–194