ABDUCTIVE REASONING DRIVEN APPROACH TO PROJECT - LIKE PRODUCTION FLOW PROTOTYPING

Grzegorz BOCEWICZ*, Zbigniew BANASZAK**

*Department of Computer Science and Management
Technical University of Koszalin, 75-453 Koszalin, Poland
bocewicz@ie.tu.koszalin.pl

**Faculty of Management
Warsaw University of Technology, 02-524 Warszawa, Poland
zbigniew.banaszak@tu.koszalin.pl

Abstract: Constraint Programming (CP) is an emergent software technology for declarative description and effective solving of large combinatorial problems especially in the area of integrated production planning. In that context, CP can be considered as an appropriate framework for development of decision making software supporting scheduling of multi-robot in a multi-product job shop. The paper deals with multi-resource problem in which more than one shared renewable and non-renewable resource type may be required by manufacturing operation and the availability of each type is time-windows limited. The problem belongs to a class of NP-complete ones. The aim of the paper is to present a knowledge based and CLP-driven approach to multi-robot task allocation providing a prompt service to a set of routine queries stated both in straight and reverse way. Provided examples illustrate both cases while taking into account an accurate as well as an uncertain specification of robots and workers operation time.

Key words: knowledge engineering, modeling, constraints logic programming, scheduling.

1. Introduction

Some industrial processes simultaneously produce different products using the same production resources. 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 minimizing 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's needs might be formulated in a form of standard, routine questions, such as: Does the 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 performance measure. 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 for interactive and task oriented DSSs designing [4, 8].

Several techniques have been proposed in the past fifty years, including MILP, Branch-and-Bound [6] 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 [6, 18] seems to be well suited for modeling of real-life and day-to-day decision-making processes in an enterprise [5]. In turn, applications of fuzzy set theory in production management [19] show that most of the research on project scheduling has been focused on fuzzy PERT and fuzzy CPM [12, 13].

In this context, the contribution provides the framework allowing one to take into account both: distinct (pointed), and imprecise (fuzzy) data, in a unified way and treated in a unified form of a discrete, constraint satisfaction problem (CSP) [4]. 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 [5, 11].

The following two classes of standard routine queries are usually considered and they are 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?

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?

Above mentioned categories encompass the different reasoning perspectives, i.e. deductive and abductive ones.

The corresponding queries can be stated in different models that in turn may be treated as compositions 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.

Possible problems formulation taking into account commercially available software packages capabilities is shown in the Table 1. So, that is easy to observe that commercially available tools are not able to consider cases assuming imprecise data as well as are not able to state a problem in an reverse way (e.g., looking for values of some input variables guaranteeing the assumed output variables reach required values).

Moreover, the commercially available DSSs are not able to respond in an interactive, i.e. on-line/real-time mode, as well as to support a project-like production flow prototyping (i.e. integrated production planning containing such partial problems as routing, batch-sizing and scheduling).

That disadvantage is our motivation to develop methodology supporting one in the course of designing of an interactive and task oriented decision support systems aimed at projects portfolio prototyping. By projects prototyping we mean a decision process resulting in selection (variables adjustment) both an enterprise and projects portfolio parameters fulfilling assumed requirements, e.g. an admissible solution being a kind of an equilibrium between enterprise capabilities and projects' cost and make span.

Table 1. Possible problems formulation available in commercially available software packages perspective *(source: self study)*

DSS	variables		resources		queries	
	precise	imprecise	renewable	non-renewable	straight	reverse
Primavera	✓	×	✓	×	✓	×
Planiswere	✓	×	✓	×	✓	×
Tracker Suite	✓	×	✓	×	✓	×
Project Net	✓	×	✓	×	✓	×
Team Work	✓	×	✓	×	✓	×
…	…	…	…	…	…	...
MS Project	✓	×	✓	×	✓	×

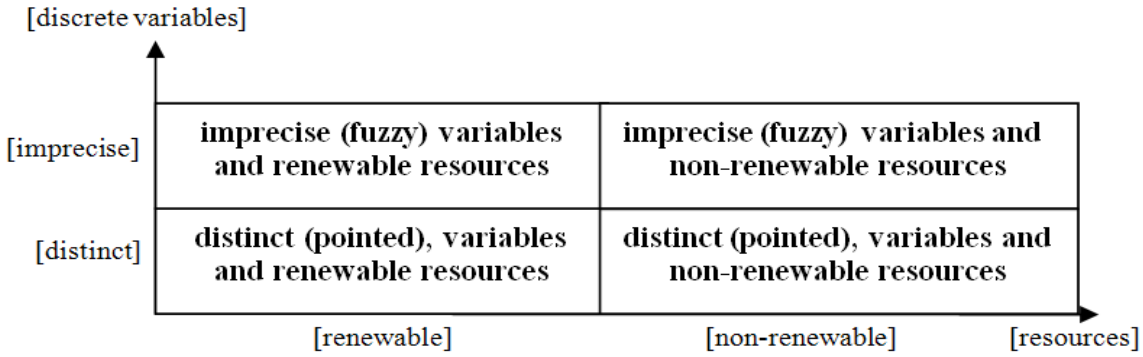

Figure 1. Elementary decision problems *(source: self study)*

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.

Note that 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 that context the methodology proposed consists of the following three stages, see Fig. 2. At the first stage the reference model of constraint satisfaction problem is considered. That means that on the base of available specifications of a small and medium sized enterprise and projects portfolio as well as the assumed routine queries and possible auxiliary (suggested by experts) knowledge a relevant **reference model of constraint satisfaction problem** is designed. The model encompasses technical parameters, experts' experience and user expectations in the form of knowledge base, i.e. as a set of variables, and their domains, and a set of relations (constraints, e.g. time-window resource availability) linking some subsets of constraints. Such model's interpretation allows using the logic-algebraic method as a reference engine.

At the second stage, the knowledge base obtained is examined from the point of view of its future implementation in assumed, implementing CP framework, real-life DSS. Since the CP framework is useless in case variables can be gathered in disjoint clusters and is useless also for queries checking whether a given subset of variables implies other one, thus the knowledge base (KB) consistency (guaranteeing response to the set of assumed queries) and its discrepancy (guaranteeing the unique response to each query) point of view must be examined.

Besides of that, the KB has to be examined also from the point of view of the time efficiency of possible searching strategies (especially variables distribution). That means the searching strategy guaranteeing an interactive DSS operation has to be developed. The above mentioned examinations guarantee the KB specification can be directly implemented in CP framework (that means the straight and reverse problems' formulation and queries such as whether a given subset of variables implies other one can be considered).

Therefore, the third stage transforms the knowledge-based and CP-driven framework into the commercially available CP/CLP platforms (i.e., taking advantage of the fact the decision problems can be friendly formulated in a declarative way, and solved with guarantee the response DO NOT KNOW will not be allowed). So, besides of the right constraint programming

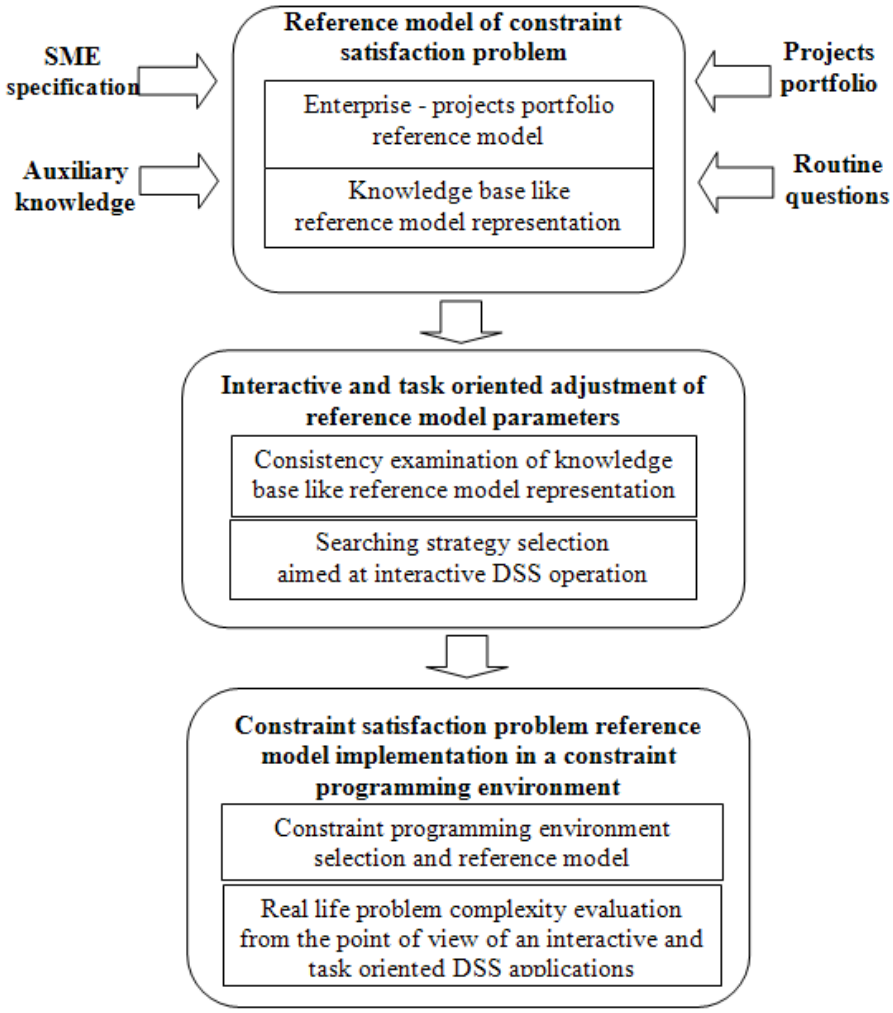

Figure 2. Main stages of an interactive and task oriented DSS designing *(source: self study)*

environment selection and the reference model implementation the complexity of a class of real life problems guaranteeing an interactive DSS application should be estimated.

Of course, in case of imprecise decision variables the one more so called preliminary stage has to be considered. The main aim of the stage is identification of membership functions in case of imprecise decision variables as well as verification of inference fuzzy rules implemented.

2. Modelling

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 aimed at the following routine question: Does a given job shop capabilities and assumed resources allocation guarantee the production orders completion time do not exceed the deadline h and amount of renewable resources is positive in any moment of time horizon H?

Given amount of lz of renewable discrete resources ro_i specified by: Ro = (ro_1, ro_2,…, ro_z). Given amounts $zo_{i,k}$ of available renewable resources zo_i = ($zo_{i,1}$, $zo_{i,2}$, …, $zo_{i,h}$), where $zo_{i,k}$ – limited amount of the i-th renewable resource at the k-th moment of H, specified by Zo = (zo_1, zo_2,…, zo_{lz}).

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

Decision variables

Given a set of production routes P = {P_1, P_2,..., P_{lp}}. Each P_i is specified by the set composed of lo_i activities, i.e., P_i = {$O_{i,1}$,…,$O_{i,loi}$}, where [2]:

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

where:

$x_{i,j}$ – 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 $O_{i,j}$-th activity

$Tp_{i,j}$ = ($tp_{i,j,1}$, $tp_{i,j,2}$,... , $tp_{i,j,lz}$) – the sequence of time 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: $0 \le tp_{i,j,k} < t_{i,j}$; k = 1,2,…,lz

$Tz_{i,j}$ = ($tz_{i,j,1}$, $tz_{i,j,2}$,... , $tz_{i,j,lz}$) – the sequence of moments the activity $O_{i,j}$ releases the subsequent resources, $tz_{i,j,k}$ – the time counted since the moment $x_{i,j}$ of 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: $0 < tz_{i,j,k} \le t_{i,j}$ and $tp_{i,j,k} < tz_{i,j,k}$; k = 1, 2,…, lz

$Dp_{i,j}$ = ($dp_{i,j,1}$, $dp_{i,j,2}$,..., $dp_{i,j,lz}$) – 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: $0 \le dp_{i,j,k} \le zo_k$; k = 1, 2,…, lz

$Tr_{i,j}$ = ($tr_{i,j,1}$, $tr_{i,j,2}$, ..., $tr_{i,j,ln}$) – the sequence of moments the determined amounts of subsequent non renewable resources are collected 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: $0 \le tr_{i,j,k} < t_{i,j}$; k = 1, 2,…, ln

$Ts_{i,j}$ = ($ts_{i,j,1}$, $ts_{i,j,2}$, ..., $ts_{i,j,ln}$) - 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-the 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.: $0 \le ts_{i,j,k} < t_{i,j}$; $k = 1, 2,\ldots, ln$, as well as $tr_{i,j,k} \le ts_{,j,k}$; $k = 1, 2, \ldots, ln$

$Cr_{i,j} = (cr_{i,j,1}, cr_{i,j,2},\ldots , cr_{i,j,ln})$ – the sequence of non-renewable resources amount consumed by activity $O_{i,j}$, $cr_{i,j,k}$ – the amount of the k-th resource required by the activity $O_{i,j}$, $cr_{i,j,1} \le 0$; $k = 1, 2,\ldots, ln$, $cr_{i,j,k} = 0$ 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 inflowed by activity $O_{i,j}$, $cs_{i,j,1} \ge 0$; $k = 1,2,\ldots,ln$, $cr_{i,j,k} = 0$ means the activity does not inflow the k-th resource

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

- starting times of activities in the route P_i:

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

- duration of activities in the route P_i:

  $Ti = (ti, 1, ti, 2,\ldots, t_{i,lo_i})$

- starting times the j-th resource is allocated to the k-th activity in the route 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:

  $TZ_{i,j} = (tz_{i,1,j}, tz_{i,2,j},\ldots, tz_{i,lo_i,j})$

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

  $DP_{i,j} = (dp_{i,1,j}, dp_{i,2,j},\ldots, dp_{i,lo_i,j})$

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

  $TR_{i,j} = (tr_{i,1,j}, tr_{i,2,j}, \ldots, tr_{i,lo_i,j})$

- the sequence of moments the j-th non renewable resource is released by activities of the project P_i:

  $TS_{i,j} = (ts_{i,1,j}, ts_{i,2,j}, \ldots, ts_{i,lo_i,j})$

- sequences of amounts of the j-th non-renewable resource consumed by activities of the route P_i:

  $CR_{i,j} = (cr_{i,1,j}, cr_{i,2,j},\ldots, cr_{i,lo_i,j})$

- sequences of amounts of the j-th non-renewable resource inflowed by activities of the route P_i:

  $CS_{i,j} = (cs_{i,1,j}, cs_{i,2,j},\ldots, cs_{i,lo_i,j})$

Assume some of chosen execution times are defined roughly, i.e. are treated as fuzzy variables specified by fuzzy sets. Therefore, the activity $O_{i,j} = (\hat{x}_{i,j}, \hat{t}_{i,j}, Tp_{i,j}, Tz_{i,j}, Dp_{i,j}, Tr_{i,j}, Ts_{i,j}, Cr_{i,j}, Cs_{i,j})$ is specified by the following sequences of:

- starting times of activities in the route P_i:

  $\hat{X}_i = (\hat{x}_{i,1}, \hat{x}_{i,2}, \ldots, \hat{x}_{i,lo_i})$

- duration of activities in the route P_i:

  $\hat{T}_i = (\hat{t}_{i,1}, \hat{t}_{i,2}, \ldots, \hat{t}_{i,lo_i})$

where:

$\hat{X}_i$– is a fuzzy set determining the activity $O_{i,j}$ starting time,

$\hat{T}_i$ – is a fuzzy set specifying the activity time,

$Tp_{i,j}, Tz_{i,j}, Dp_{i,j}, Tr_{i,j}, Ts_{i,j}, Cr_{i,j}, Cs_{i,j}$ – the sequences defined by (1).

Activities order constraints

Let us consider a set of production routes P_i composed of lo_i precedence and resource constrained, non-preemptable activities that require renewable resources. Assume lz renewable discrete resources are available and sequences $r_i = (ro_1, ro_2,\ldots, ro_f)$, $i = 1,\ldots, lo_i$, determines fixed discrete resource requirements of the i-th activity. The total number of units of the discrete resource j, $j = 1,\ldots, lz$, is limited by zo_j. The resource can be allotted (and constant within activity operation time) to activities in arbitrary amount from the set $\{1,\ldots, zo_j\}$. It means two different resources can be allotted to the i-th activity at different also overlapping each other periods of time.

The production routes P_i are represented by activity-on-node networks, where activities state for nodes and arcs determine an order of activities execution. Consequently, assuming discrete decision variables the following activities order constraints are considered [2]:

- the k-th activity follows the *i*-th one :

$$x_{i,j} + t_{i,j} \le x_{i,k} \tag{2}$$

- the *k*-th activity follows other activities:

$$\begin{aligned} &x_{i,j}+t_{i,j} \le x_{i,k,} \\ &x_{i,j+1}+t_{i,j+1} \le x_{i,k} \\ &\ldots \\ &x_{i,j+n}+t_{i,j+n} \le x_{i,k} \end{aligned} \tag{3}$$

- the k-th activity is followed by other activities:

$$\begin{aligned} &x_{i,k} + t_{i,k} \le x_{i,j} \\ &x_{i,k} + t_{i,k} \le x_{i,j+1} \\ &\dots \\ &x_{i,k}+t_{i,k} + \le x_{i,,j+n} \end{aligned} \quad (4)$$

In the case fuzzy value of variables the constraints (2), (3), (4) have following form [3]:

- the k-th activity follows the *i*-th one :

$$\hat{x}_{i,j} \hat{+} \hat{t}_{i,j} \hat{\le} \hat{x}_{i,k} \quad (5)$$

- the k-th activity follows other activities:

$$\begin{aligned} &\hat{x}_{i,j} \hat{+} \hat{t}_{i,j} \hat{\le} \hat{x}_{i,k} \\ &\hat{x}_{i,j+1} \hat{+} \hat{t}_{i,j+1} \hat{\le} \hat{x}_{i,k} \\ &\dots \\ &\hat{x}_{i,j+n} \hat{+} \hat{t}_{i,j+n} \hat{\le} \hat{x}_{i,k} \end{aligned} \quad (6)$$

- the k-th activity is followed by other activities

$$\begin{aligned} &\hat{x}_{i,j} \hat{+} \hat{t}_{i,j} \hat{\le} \hat{x}_{i,k+1} \\ &\hat{x}_{i,j} \hat{+} \hat{t}_{i,j} \hat{\le} \hat{x}_{i,k+2} \\ &\hat{x}_{i,j} \hat{+} \hat{t}_{i,j} \hat{\le} \hat{x}_{i,k+n} \end{aligned} \quad (7)$$

The relevant fuzzy arithmetic operations $\hat{+}$, $\hat{\le}$, are defined in the Appendix A. Due to the formulas (a8), (a12), see the Appendix A, any fuzzy constraint C_i (e.g. $\hat{v}_i \hat{<} \hat{v}_l$) can be characterized by the logic value $E(C_i)$, $E(C_i) \in [0,1]$. In turn, values $E(C_i)$ allow to determine the level of uncertainty DE of reference model's constraints satisfaction, i.e. a kind of uncertainty threshold. For instance, DE = 1 means the all constraints hold, and DE = 0,8 means that they are almost satisfied. The level DE is defined due to the formulae (8):

$$DE = \min_{i=1,2,\dots,lo_c}\{E(C_i)\} \quad (8)$$

where:

lo_c – a number of reference model constraints.

In the course of decision making based on constraints assuming fuzzy variables an uncertainty threshold (e.g. following an operator's experience) should be assumed. That means, the decision maker should be able to decide about the membership functions of the decision variables used as well as uncertainty thresholds of fuzzy constraints employed.

Renewable resource constraints

The constraints avoiding exceeding of resources available limits play a primary role. That means, the relevant constraints taking into account precise/imprecise character of such decision variables as activities operation times $t_{i,j}/\hat{t}_{i,j}$ and the moments of activities beginning $x_{i,j}/\hat{x}_{i,j}$ have to be considered. The approach proposed follows the way applied in case of distinct variables [2]. Note that exceeding available resources limit exceeding may result in bad resources allocation leading to the closed loops of resources requests. So, the constraints allowing one to avoid such cases follow formulas (9), (10):

$$\sum_{i=1}^{lp}\sum_{j=1}^{lo_i}\left[dp_{i,j,k} \cdot \bar{1}(x_{m,n} + tp_{m,n,k}, x_{i,j} + tp_{i,j,k}, x_{i,j} + tz_{i,j,k})\right] \le zo_{k,x_{m,n}+tp_{m,n}-1} \quad (9)$$

$\forall\ (m,n) \in \{(a,b) \mid a = 1,2,\dots,lp,\ b = 1,2,\dots,lo_a\}$, $\forall k \in \{1, 2, \dots, lz\}$

and

$$\sum_{i=1}^{lp}\sum_{j=1}^{lo_i}\left[dp_{i,j,k} \cdot \bar{1}(vg_{k,d}, x_{i,j} + tp_{i,j,k}, x_{i,j} + tz_{i,j,k},)\right] \le zo_{k,vg_{k,d}-1} \quad (10)$$

$\forall d \in \{1,2,\dots,q\}, \forall k \in \{1, 2, \dots, lz\}$

where:

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 $O_{i,j}$,

$\bar{1}(u, a, b)$ – an unary function determining the time of the resource occupation,

$\bar{1}(u, a, b) = 1(u - a) - 1(u - b), 1(u)$ - the unit step function,

$vg_{k,i}$ - the *i*-th characteristic point, it is moment $u \in H$ when amount of renewable resource ro_i changes value,

q – number of the characteristic points.

Similarly to constraints concerning precise variables the relevant ones taking into account imprecise data follow the formula (11) [2, 3]:

$$\sum_{i=1}^{lp}\sum_{j=1}^{lo_i}\left[dp_{i,j,k} \cdot \hat{\bar{1}}\left(\hat{x}_{m,n} \hat{+} tp_{m,n,k}, \hat{x}_{i,j} \widehat{+} tp_{i,j,k}, \hat{x}_{i,j} \widehat{+} tz_{i,j,k}, E_{\hat{1},i,j,m,n}\right)\right] \le zo_k \quad (11)$$

$\forall\ (m,n) \in \{(a,b) \mid a = 1, 2, \dots, lp,\ b = 1, 2, \dots, lo_a\}$ $\forall k \in \{1, 2, \dots, lz\}$

where:

$\hat{\bar{1}}(\hat{g}, \hat{a}, \hat{b}, E_{\hat{1}}) = \hat{1}(\hat{g}, \hat{a}, E_{\hat{1}}) - \hat{1}(\hat{g}, \hat{b}, E_{\hat{1}})$ – an unary fuzzy function determining the time of recourse occupation, $\hat{1}(\hat{v}, \hat{a}, E_{\hat{1}}) \in \{0,1\}$- the unit fuzzy function.

$$\hat{1}(\hat{g}, \hat{a}, E_{\hat{1}}) = 1 - \frac{E_{\hat{1}} - E(\hat{g} \hat{\ge} \hat{a})}{1 - 2E(\hat{g} \hat{\ge} \hat{a})} \quad (12)$$

where:

$\hat{a}, \hat{b}, \hat{g}$ – the fuzzy numbers,

$E_{\hat{1}} \in [0,1]$ – the logic value of unit fuzzy function, $E_{\hat{1},i,j,m,n}$ – the logic value of i,j-th an unary fuzzy function for pair (m,n).

In case considered, see (11) the amounts $zo_{k,i}$ of available renewable resources are assumed to be constant in whole horizon H: $zo_{k,1} = zo_{k,2} = \ldots = zo_{k,h} = zo_k$ [3].

If for any moment $\hat{x}_{m,n}$ (where: $(m,n) \in \{(a,b) \mid a = 1, 2,\ldots, lp, b = 1, 2,\ldots, lo_a\}$) and for each ro_k-th renewable resource $k \in \{1, 2,\ldots, lz\}$, conditions (11) hold, then projects portfolio execution will deadlock free (and conflict-free) with the uncertainty level $DEf = \min\{E_{\hat{1},i,j,m,n}\}$.

Non-renewable resource constraints

Because of limited amount of available discrete non-renewable resources the constraints protecting against their allocation exceeding available outflows should also be considered. Moreover, because non-renewable resources can be allotted at the same time to different activities in a way causing in occurrence of closed loop resources requests, i.e. the deadlocks. In order to avoid them the relevant constraints should be imposed. By analogy to renewable resources allocation the constraints guaranteeing deadlock-free execution of activities (treated as precise variables) are considered [2, 3]:

$$zn_k - \sum_{i=1}^{lp}\sum_{j=1}^{lo_i}\left[cr_{i,j,k} \cdot \bar{1}\left(x_{m,n} - x_{i,j} - tr_{i,j,k}\right)\right] + \sum_{i=1}^{lp}\sum_{j=1}^{lo_i}\left[cs_{i,j,k} \cdot \bar{1}\left(x_{m,n} - x_{i,j} - ts_{i,j,k}\right)\right] \geq 0 \quad (13)$$

$\forall (m,n) \in \{(a,b) \mid a = 1 , 2,\ldots, lp; b = 1, 2,\ldots, lo_a\}$

$\forall k \in \{1, 2,\ldots, ln\}$

where:

lp – the number of projects,

lo_i – the number of the i-th project's activities,

1(v) – the unit step function,

$cr_{i,j,k}$ – the amount of the k-th resource required by the activity $O_{i,j}$,

$cs_{i,j,k}$ – the amount of the k-th resource flowed by activity $O_{i,j}$, zn_k denotes amount of the resource rn_k being available at the beginning of time horizon H.

In case of imprecise variables the constraints guaranteeing deadlock-free execution of activities follow formulae (13), see [3]:

$$zn_k - \sum_{i=1}^{lp}\sum_{j=1}^{lo_i}\left[cr_{i,j,k} \cdot \hat{1}\left(\hat{x}_{m,n}, \hat{x}_{i,j} \hat{+} tr_{i,j,k}, E_{\hat{1},i,j,m,n}\right)\right] + \sum_{i=1}^{lp}\sum_{j=1}^{lo_i}\left[cs_{i,j,k} \cdot \hat{1}\left(\hat{x}_{m,n}, \hat{x}_{i,j} \hat{+} ts_{i,j,k}, E_{\hat{1},i,j,m,n}\right)\right] \geq 0$$

$\forall (m,n) \in \{(a,b) \mid a = 1, 2, \ldots, lp; b = 1, 2, \ldots, lo_a\}$

$\forall k \in \{1, 2, \ldots, ln\}$ (14)

where:

lp – the number of projects,

lo_i – the number of the i-th project's activities,

$\hat{1}\left(\hat{v}, \hat{a}, E_{\hat{1},i,j,m,n}\right)$ – the fuzzy unit function (12),

zn_k – denotes amount of the resource rn_k being available at the beginning of time horizon H.

If at any moment $\hat{x}_{m,n}$ (where: $(m,n) \in \{(a,b) \mid a = 1, 2,\ldots, lp, b = 1, 2,\ldots, lo_a\}$) for each rn_k-th nonrenewable resource $k \in \{1, 2, \ldots, ln\}$ conditions (11) hold, then projects portfolio execution is deadlock-free with the uncertainty level $DEf = \min\{E_{\hat{1},i,j,m,n}\}$.

2.3. 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 [4]. 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 [4].

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) [4] is defined as follows: CS = ((A, D), C), where: $A = \{a_1, a_2,..., a_g\}$ – a finite set of discrete decision variables, $D = \{D_i \mid D_i = \{d_{i,1}, d_{i,2},..., d_{i,j},..., d_{i,ld}\}, i = 1,..., g\}$ – a family of finite variable domains and the finite set of constraints $C = \{C_i \mid i = 1,\ldots, L\}$ – 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 [3]. The constraints propagation exe-

cutes 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.

3. Decision support tool for project portfolio prototyping

The considered Decision Support Tool for Project Portfolio Prototyping (DST4P^3) aimed at project planning in small and medium sized enterprises (SME) has been developed in Oz Mozart [17] and Delphi languages environment. The main components of the DST4P^3 structure are shown in Fig. 3. The system considered is composed of two modules serving for computations and interfacing, respectively. Of course, the main role plays the first module responsible for implementation of the reference model (see chapter 2) specified in terms of the fuzzy constraint satisfaction problem [9] (implementing fuzzy variables and fuzzy constraints (5), (6), (7), (11), (14)) and operation of an inference engine (implementing the logic-algebraic method) operation [8, 10].

Moreover, the module employs procedures enabling constraint compression and time effective searching strategies [6] as well as a newly introduced algebraic and logic operations allowing to calculate fuzzy constraints including fuzzy numbers [3, 9].

The second module of the DST4P^3 enables problems specification, i.e. input data insertion, and queries selection, as well as an output data visualization and documentation. The following kinds of project planning problems are allowed:

- "straight" with distinct variables specifying the SME at hand,
- "straight" with imprecise variables specifying the SME at hand,
- "reverse" with distinct variables specifying the SME at hand,
- "reverse" with distinct variables specifying the SME at hand.

Illustrative examples of the DST4P^3 (fig. 3) application to the above mentioned problems provide the section below.

4. Illustrative examples

Example 1 – „straight"/distinct variables

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 [2] are shown on the following figures: Fig. 4, Fig. 5, Fig. 6, and Fig. 7.

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:

$T_1 = (1, 2, 3, 4, 4, 8, 3, 2, 1, 6)$

$T_2 = (3, 1, 6, 3, 2, 5, 1, 5, 2, 4, 2, 1)$

$T_3 = (3, 7, 2,7, 2, 1,8, 3, 3, 4, 8)$

$T_4 = (3, 3, 2, 8, 3, 1, 4, 1, 8, 4, 3, 3, 8)$

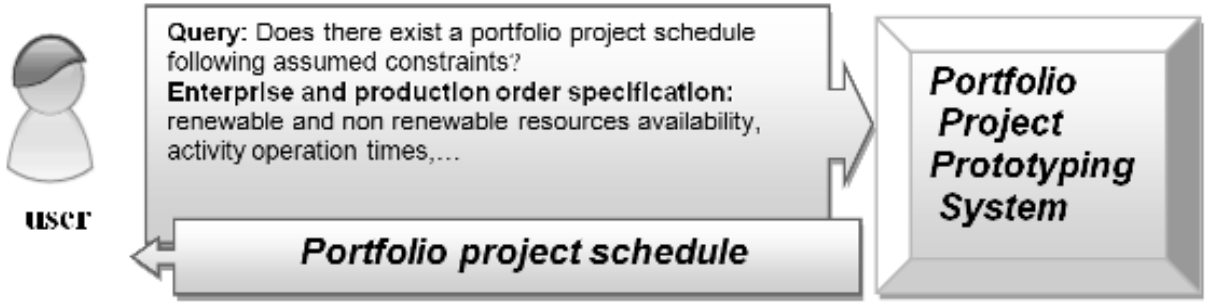

Figure 3. Mains components of DST4P^3 structure *(source: self study)*

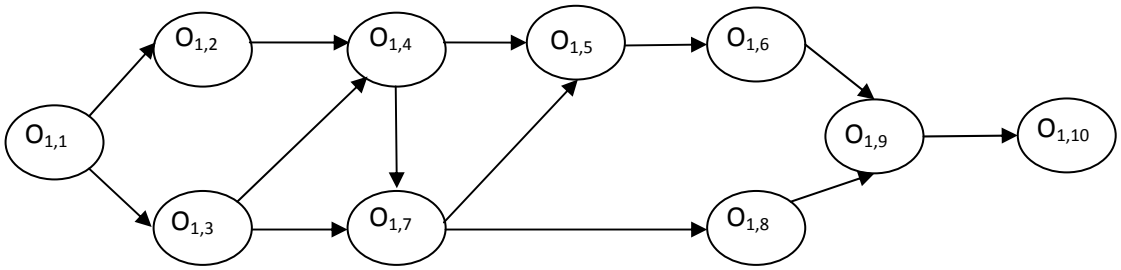

Figure 4. Activity network for the project P_1 *(source: self study)*

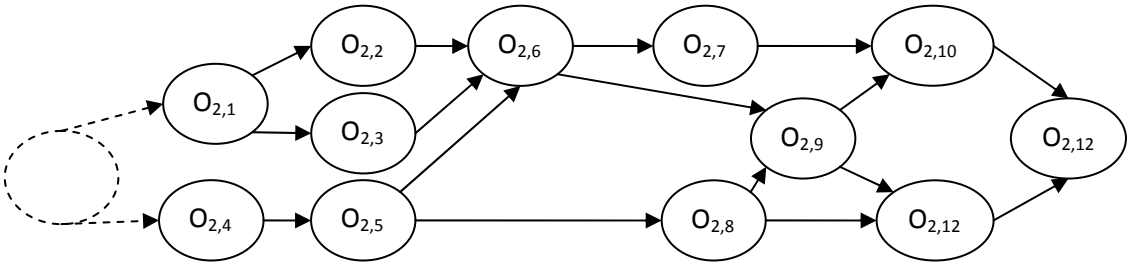

Figure 5. Activity network for the project P_2 *(source: self study)*

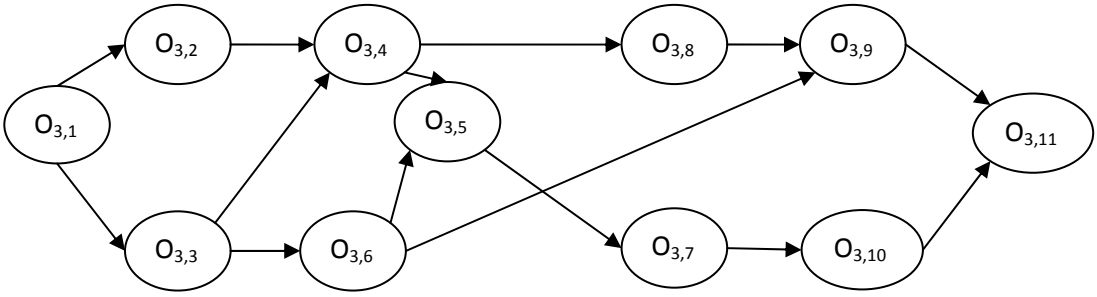

Figure 6. Activity network for the project P_3 *(source: self study)*

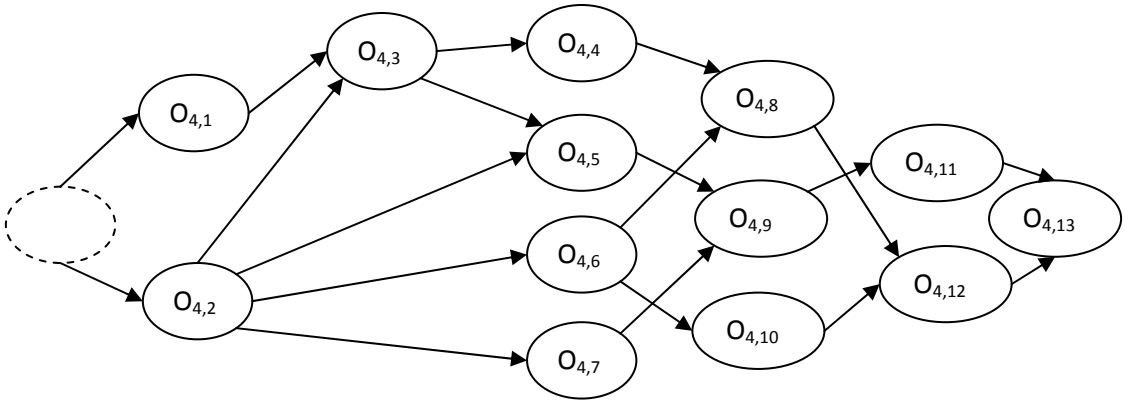

Figure 7. Activity network for the project P_4 *(source: self study)*

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 the relevant amount of resources required by particular activity 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 2, Table 3, Table 4, and Table 5.

That is assumed some activates besides of renewable resources require also non-renewable resources. 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. That 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 6, Table 7, Table 8, and Table 9.

Table 2. Amounts of resources required by activities of the project P1 *(source: self study)*

	$O_{1,1}$	$O_{1,2}$	$O_{1,3}$	$O_{1,4}$	$O_{1,5}$	$O_{1,6}$	$O_{1,7}$	$O_{1,8}$	$O_{1,9}$	$O_{1,10}$
$DP_{1,1}$	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 3. Amounts of resources required by activities of the project P2 *(source: self study)*

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

Table 4. Amounts of resources required by activities of the project P3 *(source: self study)*

	$O_{3,1}$	$O_{3,2}$	$O_{3,3}$	$O_{3,4}$	$O_{3,5}$	$O_{3,6}$	$O_{3,7}$	$O_{3,8}$	$O_{3,9}$	$O_{3,10}$	$O_{3,11}$
$DP_{3,1}$	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 5. Amounts of resources required by activities of the project P4 *(source: self study)*

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

Table 6. Amount of used up (CR) and generated (CS) non-renewable resources required by activities of the project P1 *(source: self study)*

	$O_{1,1}$	$O_{1,2}$	$O_{1,3}$	$O_{1,4}$	$O_{1,5}$	$O_{1,6}$	$O_{1,7}$	$O_{1,8}$	$O_{1,9}$	$O_{1,10}$
$CR_{1,1}$	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 7. Amount of used up (CR) and generated (CS) non-renewable resources required by activities of the project P2 *(source: self study)*

	$O_{2,1}$	$O_{2,2}$	$O_{2,3}$	$O_{2,4}$	$O_{2,5}$	$O_{2,6}$	$O_{2,7}$	$O_{2,8}$	$O_{2,9}$	$O_{2,10}$	$O_{2,11}$	$O_{2,12}$
$CR_{2,1}$	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 8. Amount of used up (CR) and generated (CS) non-renewable resources required by activities of the project P3 *(source: self study)*

	$O_{3,1}$	$O_{3,2}$	$O_{3,3}$	$O_{3,4}$	$O_{3,5}$	$O_{3,6}$	$O_{3,7}$	$O_{3,8}$	$O_{3,9}$	$O_{3,10}$	$O_{3,11}$
$CR_{3,1}$	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 9. Amount of used up (CR) and generated (CS) non-renewable resources required by activities of the project P4 *(source: self study)*

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

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:

CFt – 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].

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?

Solution to the problem results in determination of moments the activities start their execution $x_{i,j}$ [8]. So, the solution we are searching for has the form of the following sequences: $X_1 = (x_{1,1},\ldots, x_{1,10})$, $X_2 = (x_{2,1},\ldots, x_{2,12})$, $X_3 = (x_{3,1},\ldots, x_{3,11})$, $X_4 = (x_{4,1},\ldots, x_{4,13})$.

The graphical representation of the projects portfolio schedule is shown in the Fig. 8. The schedule obtained follows all constrains imposed by an enterprise capacity 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. 9. It can be observed the assumed resource's limit was not exceeded, the same regards of resources zo_2, zo_3.The Fig. 10 in turn 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 main capabilities of the DST4P[3] package possessing capability of multi-criteria project planning (e.g. taking into account a particular project deadline, projects portfolio deadline, resources limits, and so on) and an interactive 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).

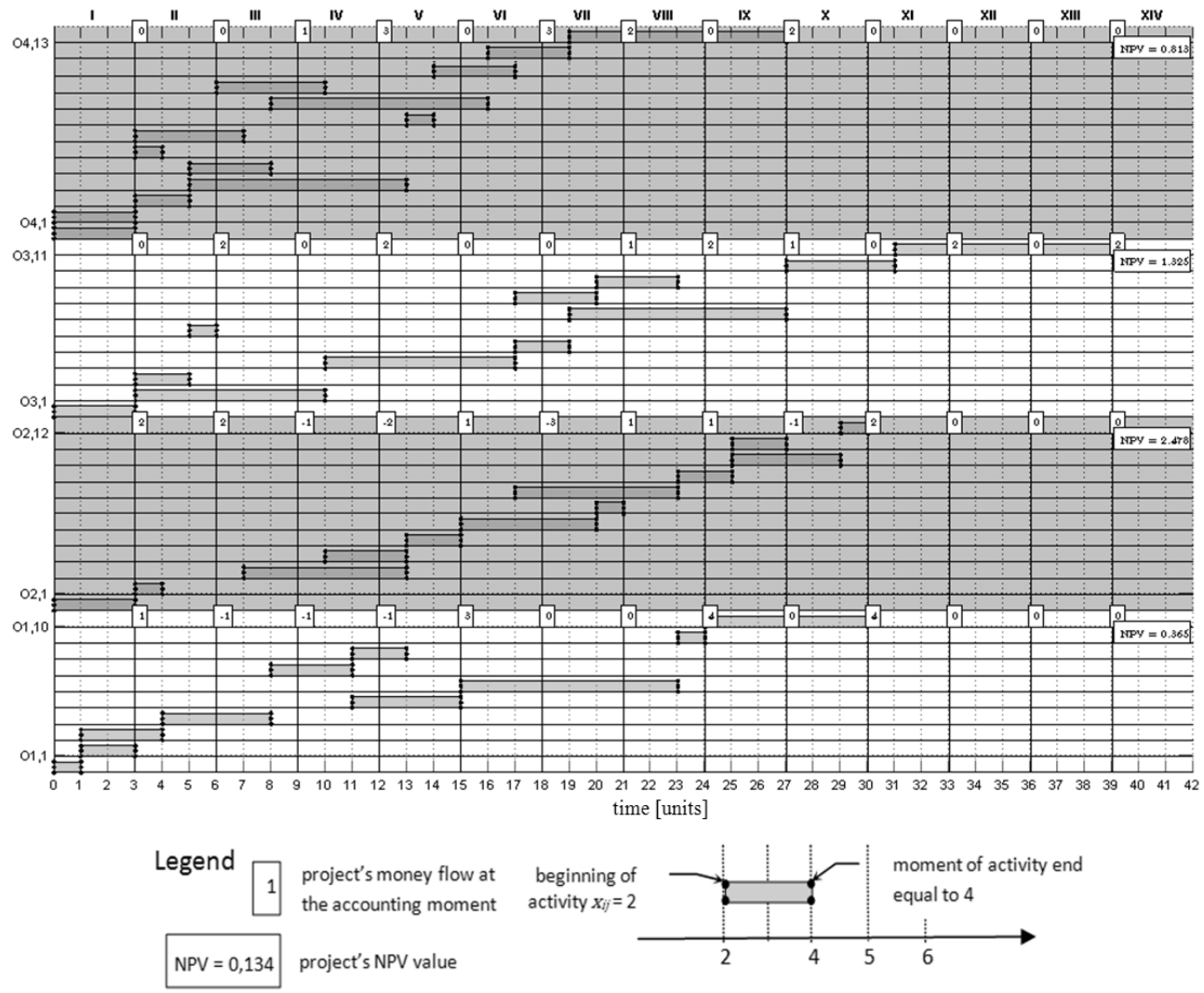

Figure 8. Projects portfolio schedule *(source: self study)*

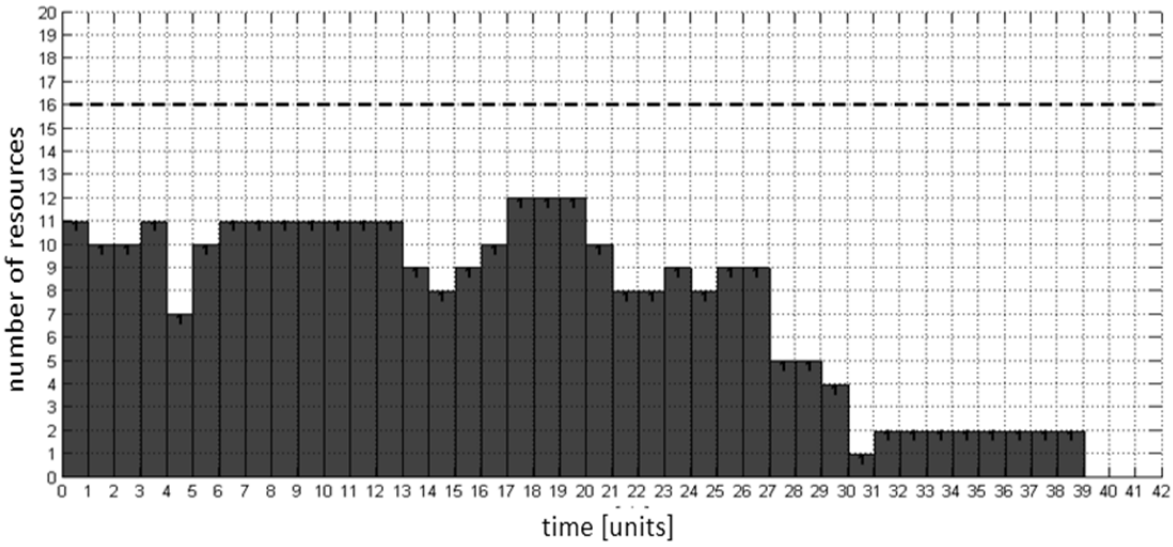

Figure 9. Gantt's-like chart of the renewable resource zo1 usage *(source: self study)*

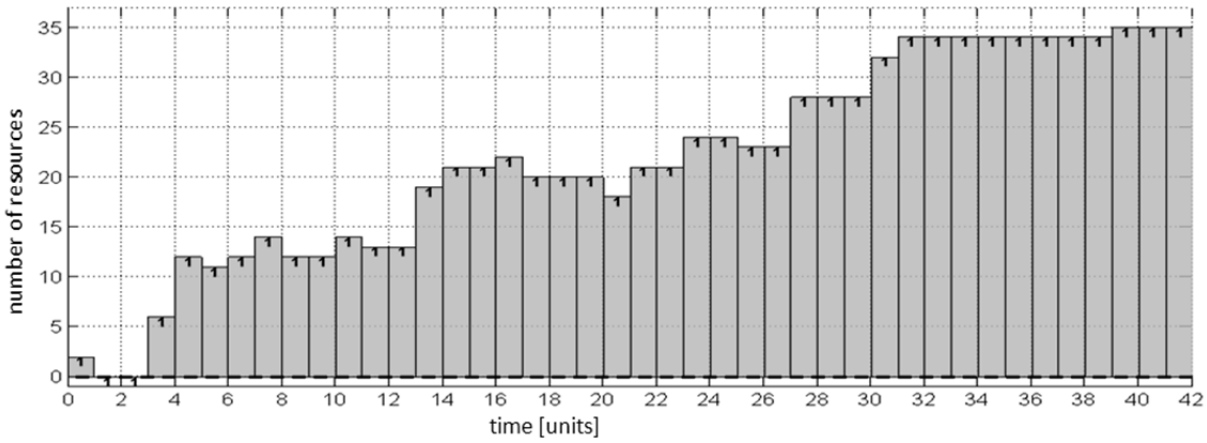

Figure 10. Gantt's-like chart of the non-renewable resource zn2 usage *(source: self study)*

Example 2 – „straight"/distinct variables

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. 4, Fig. 5, Fig. 6 and Fig. 7) and resources allocations (see the Table 2 – Table 9) as in the Example 1. However, the new time horizon H = {0, 1,…, 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.

In the case considered, in 2 seconds, the DST4P^3 package's response was: *Lack of any solutions.* That means no schedule there exists. In such situation, however there is still a possibility to reformulate the problem considered by stating it in terms of imprecise variables, i.e. looking for a solution specified by an uncertainty measure. Such the case is just considered below.

Example 3 – „straight"/imprecise variables

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. 4, Fig. 5, Fig. 6 and Fig. 7) and resources allocations (see Table 2 – Table 9) as in the Example 1. In case considered the industrial robots guaranteeing distinct operation times and workers responsible for imprecise operation times are treated as available renewable resources. However, the new time horizon H = {0,1,…,36} is considered.

Given the uncertainty threshold value DE ≥ 0,7 limiting uncertainty of constraints specifying projects portfolio. So, DE determines the minimal grade value guaranteeing the all constraints hold. For instance, DE = 0,9 means that the makespan of the projects portfolio considered will hold within the given time horizon H. Moreover, the operations times of activities: $O_{1,6}$, $O_{1,10}$, $O_{2,3}$, $O_{2,6}$, $O_{2,8}$, $O_{3,2}$, $O_{3,4}$, $O_{3,7}$, $O_{3,11}$, $O_{4,4}$, $O_{4,9}$, $O_{4,13}$ have an imprecise character. So, the relevant sequences of activities' operation times are as follow:

$_1$ = (1, 2, 3, 4, 4, "about 6", 3, 2, 1, "about 4")

$_2$ = (3, 1, "about 4", 3, 2, "about 3", 1, "about 4", 2, 4, 2, 1)

$_3$ = (3, "about 5", 2, "about 5", 2, 1, "about 6", 3, 3, 4, "about 6")

$_4$ = (3, 3, 2, „about 6", 3, 1, 4, 1, „about 6", 4, 3, 3,"about 6")

For instance, in the case the activity's $O_{1,6}$ operation time is „about 6" (see Fig. 11) that means the activity can be executed within the time period of 4 till 8 units of time. In order to be able to distinguish crispy and imprecise variables the following symbol „^" is used.

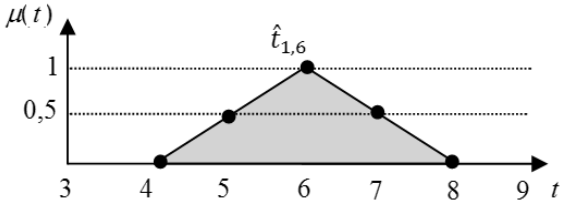

Figure 11. *(source: self study)*

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 with the uncertainty threshold DE ≥ 0,7 ?

The proposed problem formulation assumes the solutions obtained are imprecise, so their implementation can be risky because of such uncertainty. Moreover we assume some variables e.g. operations times, and uncertainty threshold value DE are imprecise.

Similarly to the Example 1 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, as in the Example 2.

The first admissible solution provided by DST4P[3] (obtained in 10 s) has the following form:

X_1 = (0, 1, 1, 4, 11, 15, 8, 11, 22, 23)

X_2 = (0, 3, 10, 10, 13, 15, 19, 18, 23, 25, 25, 29)

X_3 = (0, 3, 3, 9, 15, 5, 17, 15, 18, 24, 28)

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 follow the requirement NPV > 0, i.e. NPV_{P1} = 0,3649, NPV_{P2} = 2,6024, NPV_{P3} = 1,6177, NPV_{P4} = 0,8165.

The graphical representation of the projects portfolio schedule is show in the Fig. 12. The schedule obtained follows all constrains imposed by an enterprise capacity and projects execution requirements.

Obtained schedule provides the plan for projects portfolio execution, where uncertainty threshold level for all constraints is equal or less than 0,7 (that one may interpret as a risk of due time completion on the level equal to 0,3).

The level the planed schedule fits its real live execution depends on a decision maker. The way such fitting can be adjusted is show in the Example 5.

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 renewable resource zo_4 usage rate containing assumed resource's limit (equal to 12 units) in whole time horizon is shown on Fig. 14.

Assumed resource's limit is distinguished by bold and dashed line. The chart considered provides information about the number of currently used resources units. For instance (see Fig. 13), between the first and the second unit of time there are used six resource units (i.e. with certainty equal to 1).

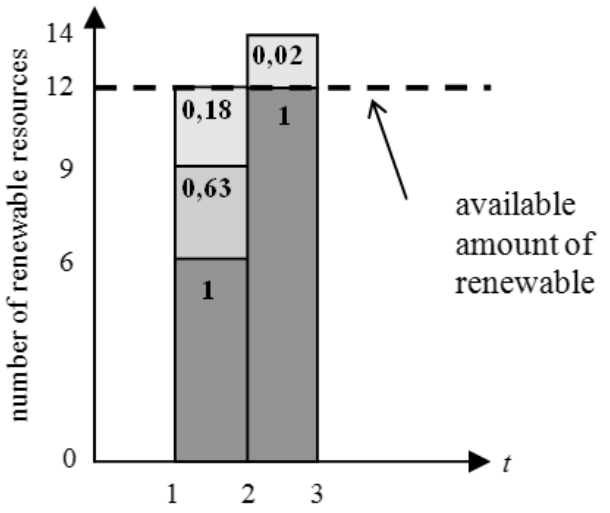

Figure 13. Illustration of the renewable resource usage rate estimation *(source: self study)*

That number changes, however with uncertainty level, for instance with the uncertainty level equal to 0,18, the 12 resource units are required. In turn, in the period between the second and the third units of time a risk level equal to 0,02 resulting in exceeding the assumed resource limit (by two units) is observed.

The similar cases concerning the resource limits exceeding can be observed for the resource zo_3 (see Fig. 14). That means, due to the schedule from Fig. 12, resource shortage may occur at the following time units 12, 15 and 19.

Quite similar observations regards of the non-renewable resource zn_2 see Fig. 15. In this case, however any risk of the resource shortage does not occur. In turn, the constraint determining the minimal, i.e. equal to 0, amount of renewable resources holds in both kinds of resources in whole time horizon.

The problem of the size just considered took less than 1 minute (the AMD Athlon(tm)XP 2500 + 1,85 GHz, RAM 1,00 GB platform has been used).

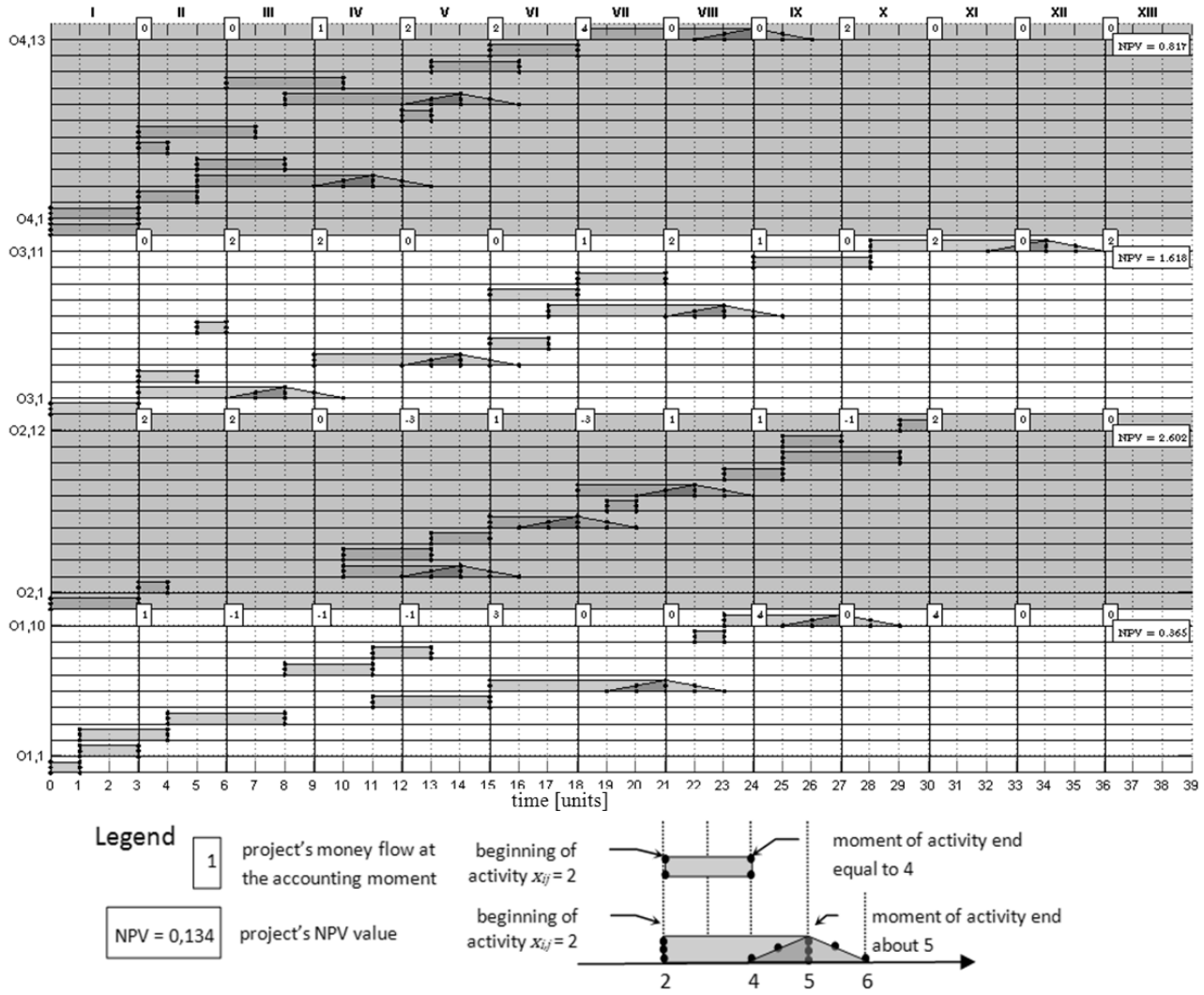

Figure 12. Projects portfolio schedule with the uncertainty threshold ≥ 0,7 *(source: self study)*

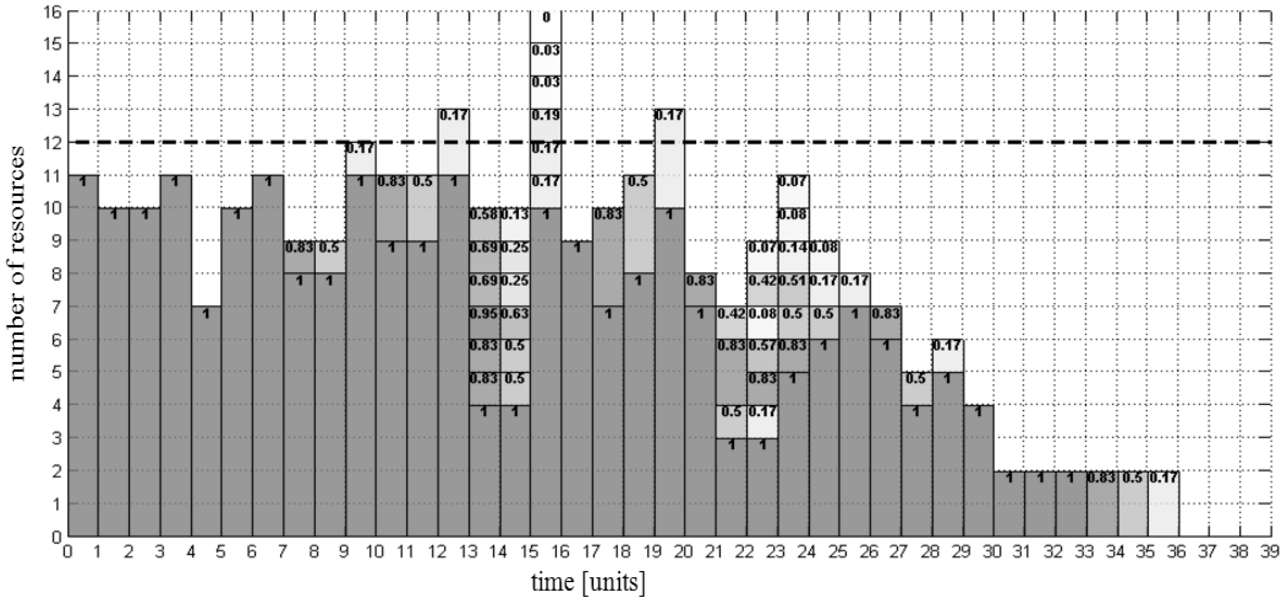

Figure 14. Usage rate of renewable resource zo3 *(source: self study)*

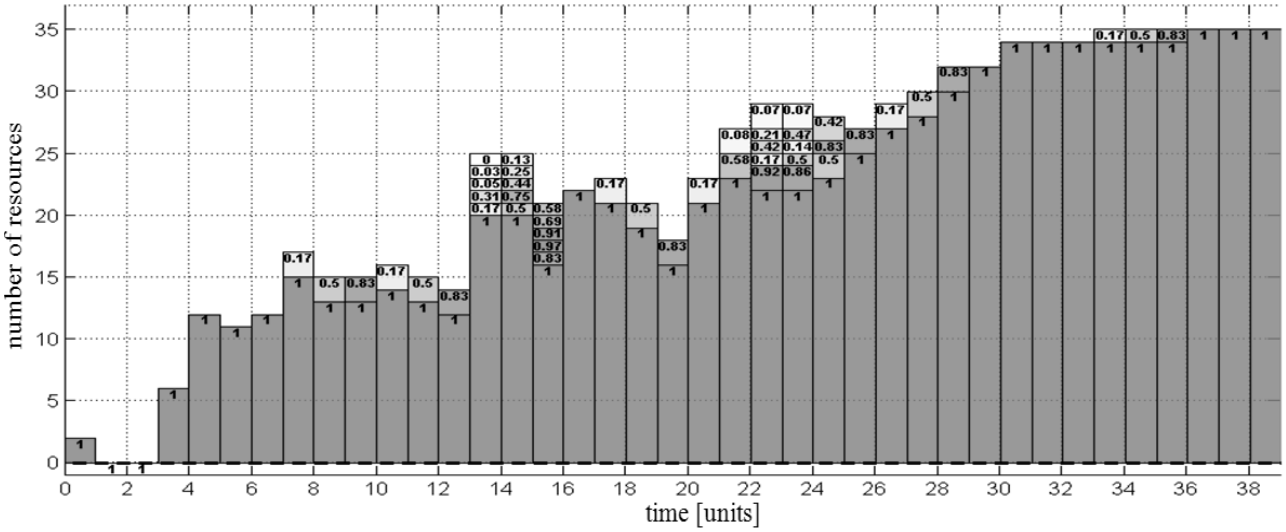

Figure 15. Usage rate of non-renewable resource zn2 *(source: self study)*

Example 4 – „straight"/imprecise variables

Given the following projects portfolio, i.e. the set of projects P = {P_1, P_2, P_3, P_4} as in the Example 3. However, the new, shorter time horizon H = {0, 1,…, 33} 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 with the uncertainty threshold DE ≥ 0,7 ?

In the case considered, in 3 seconds, the DST4P^3 package's response was: *Lack of any solutions***.** That means no schedule there exists under assumption of the uncertainty threshold DE ≥ 0,7. In such situation, however there is still a possibility to reformulate the problem considered by assuming greater uncertainty level (less value of DE). In case the increasing of an uncertainty level is not acceptable the decision maker can reformulate the problem statement, e.g. stating it in a reverse way. In such a new formulation the decision maker is looking for the values of a subset of assumed decision variables guaranteeing the makespan of considered projects portfolio will not exceed the deadline h. Such the case is just considered below.

Example 5 – „reverse"/imprecise variables

Given the following projects portfolio, i.e. the set of projects P = {P_1, P_2, P_3, P_4} as in the Example 4. H = {0,1,…,33} is the time horizon considered.

Given the uncertainty threshold value DE ≥ 0,7 limiting uncertainty of constraints specifying projects portfolio. Similarly to the Example 4, the operations times of activities: $O_{1,6}$, $O_{1,10}$, $O_{2,3}$, $O_{2,6}$, $O_{2,8}$, $O_{3,2}$, $O_{3,4}$, $O_{3,7}$, $O_{3,11}$, $O_{4,4}$, $O_{4,9}$, $O_{4,13}$ have an imprecise character. Moreover, let us assume that besides of operation times of the following activates $O_{3,7}$ and $O_{3,11}$ the values of all rest parameters are known. Given is the following relationship linking operation times of activates $O_{3,7}$ and $O_{3,11}$: $\hat{t}_{3.7} + \hat{t}_{3.11} = \text{about } 8$.

The problem considered belongs to the class of „reverse" ones and reduces to the following question: Do there exist such activities operation times guaranteeing production orders completion time not exceeds the deadline h with the uncertainty threshold DE ≥ 0,7 while constraints assumed on availability of renewable and non-renewable resources and NPV > 0 hold?

In such problem formulation the searched solution regards of the values of some decision variables guaranteeing the makespan of considered projects portfolio do exceed the deadline h. So, in the particular case the considered variables are activities $O_{3,7}$ and $O_{3,11}$, and the searched values concerns of activities operation times $\hat{t}_{3,7}, \hat{t}_{3,11}$ and moments of activities $x_{i,j}$ beginning, i.e. the components of the following sequences: X_1 = ($x_{1,1}$, …, $x_{1,10}$), X_2 = ($x_{2,1}$, …, $x_{2,12}$), X_3 = ($x_{3,1}$,…, $x_{3,11}$), X_4 = ($x_{4,1}$,…, $x_{4,13}$).

The first admissible solution provided by DST4P^3 (obtained in 15 s) has the following form concerning

the operation times: $\hat{t}_{3,7} = 6$, $\hat{t}_{3,11}$ = "about 3" (see Fig. 16) and the following form concerning the moments of activities start:

X_1 = (0, 1, 1, 4, 11, 15, 8, 11, 22, 23)

X_2 = (0, 3, 10, 10, 13, 15, 19, 18, 23, 25, 25, 29)

X_3 = (0, 3, 3, 9, 15, 5, 17, 15, 18, 24, 28)

X_4 = (0, 0, 3, 5, 5, 3, 3, 12, 8, 6, 13, 15, 18)

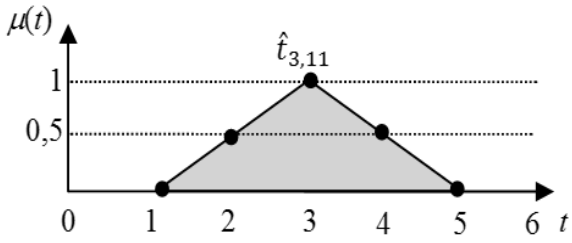

Figure 16. $\hat{t}_{3,11}$ = "about 3" *(source: self study)*

The NPV index value calculated for projects: P_1, P_2, P_3, P_4 follow the requirement NPV > 0, i.e. NPV_{P1} = 0,3649, NPV_{P2} = 2,6024, NPV_{P3} = 1,6014, NPV_{P4} = 0,8165.

The graphical representation of the projects portfolio schedule is show on the Fig. 17. The schedule obtained assumes operation times of activities $O_{3,7}$, $O_{3,11}$ equal to "about 6" and "about 3", respectively, and follows all constrains imposed by an enterprise capacity and projects execution requirements.

Obtained schedule provides the plan for projects portfolio execution, and provides a base for further adjustment aimed at fitting to real live execution [3]. The adjustment process consist in narrowing down the periods of operation times (by changing the beginning and ending moments of activities executions) as to avoid their overlapping, i.e. removing the confusion regarding the cases where an activity's ending exceeds its begging. It that context the schedule fitting leads to a minimal, confusion-free periods of operation times. The illustration of such fitting is shown on Fig. 18 presenting the projects portfolio schedule assuming the uncertainty threshold value DE ≥ 0,5. The new schedule has been obtained from the former one shown in the Fig. 17 under assumption the uncertainty threshold value DE ≥ 0,7.

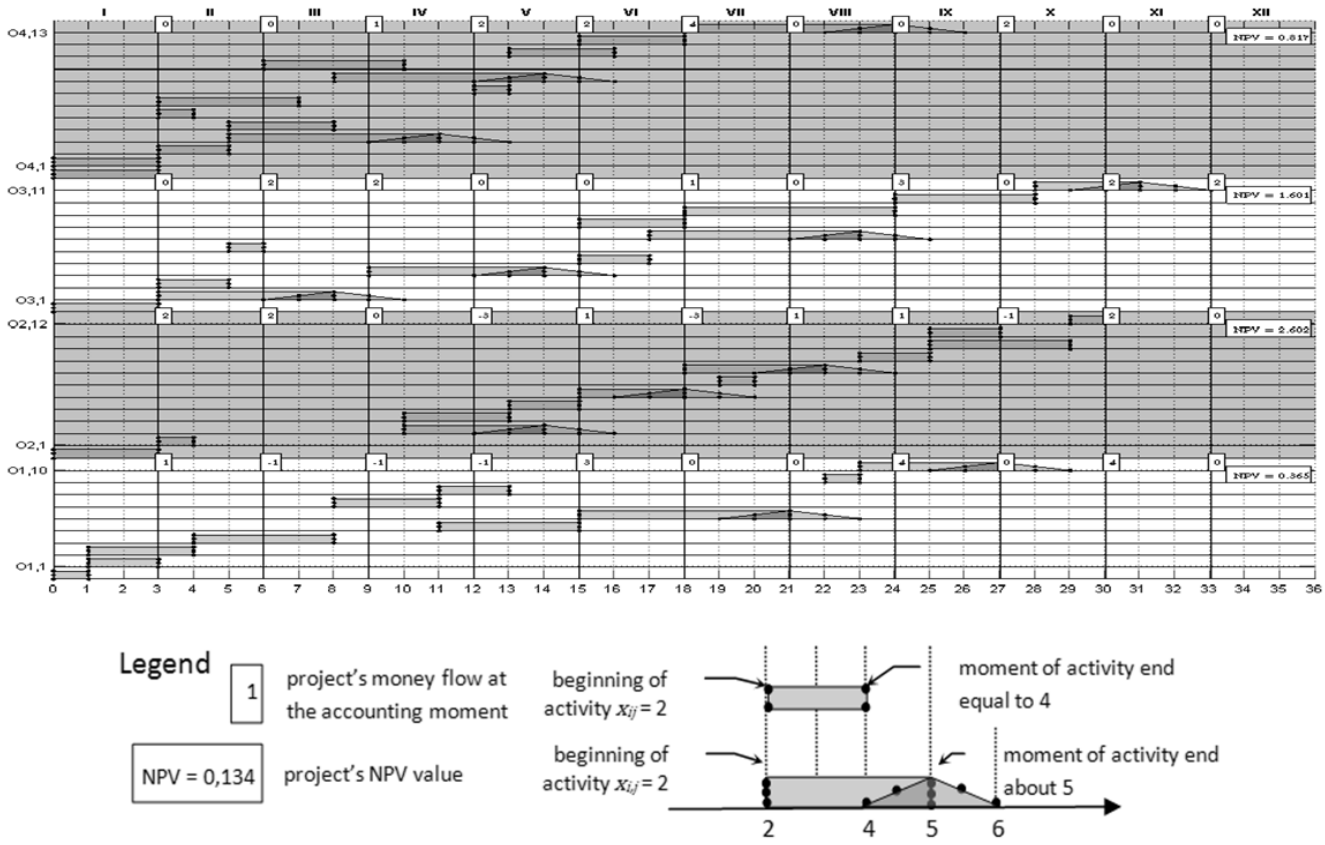

Figure 17. Projects portfolio schedule with the uncertainty threshold ≥ 0,7 *(source: self study)*

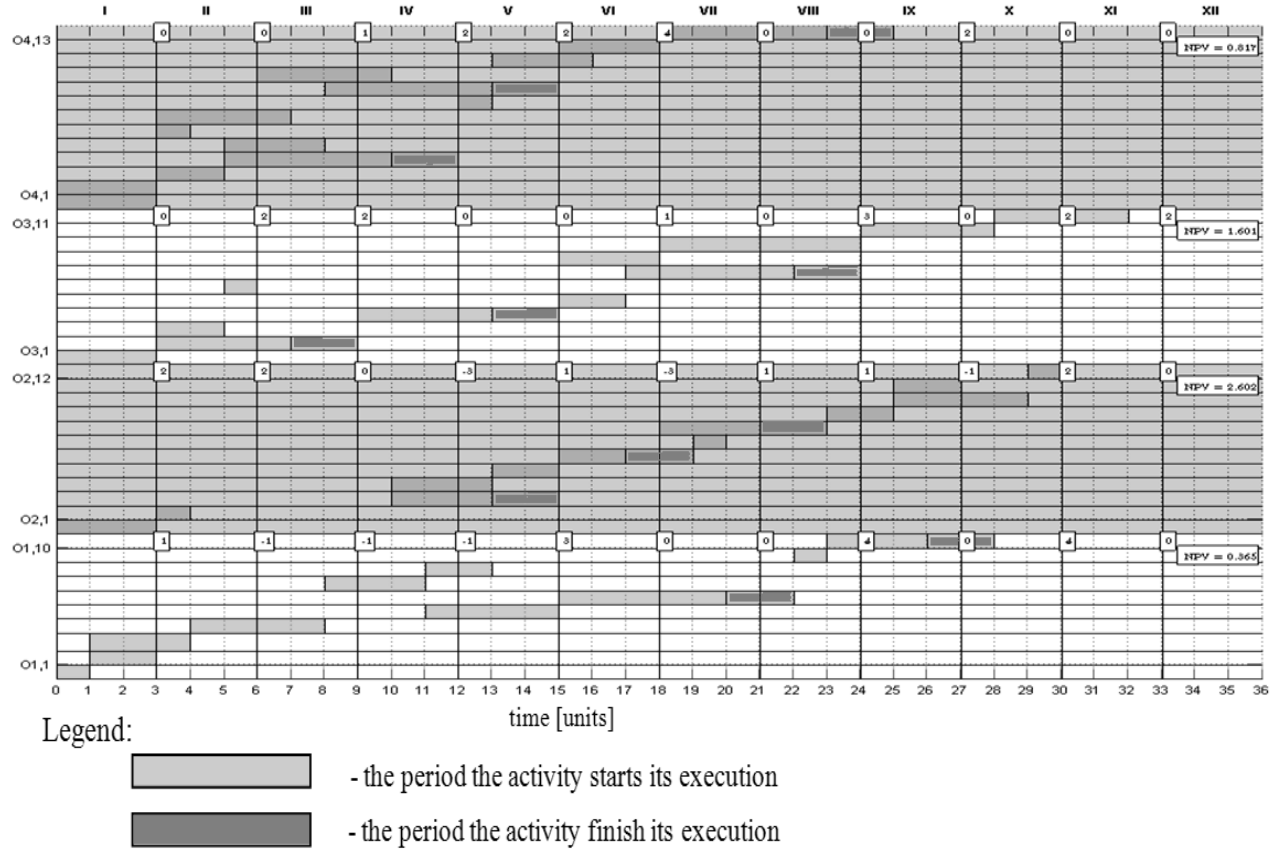

Figure 18. Projects portfolio schedule with the uncertainty threshold $\geq 0{,}5$ *(source: self study)*

5. Concluding remarks

Our approach to an interactive task oriented decision support tools provides the framework allowing one to take into account both: straight and reverse problems formulation. This advantage can be seen as a possibility to response (besides of such standard questions as Is it possible to complete a given set of production orders at a scheduled project deadline?) to the questions like: What variables value guarantee the production orders makespan follows the assumed deadline? Constraint programming paradigm standing behind of the methodology aimed for such tools designing allows to take into account both distinct and imprecise character of the decision variables as well as to consider of multi-criteria decision problems.

The methodology developed is based on the concept of the decision problem reference model [3]. The model considered can be seen as a knowledge base encompassing the structure of a constraint satisfaction problem, where the logic-algebraic method plays a role of inference engine. So, the main idea standing behind of the methodology lies in a way the knowledge base is "adjusted", i.e. adding the conditions guaranteeing the responses to the standard queries there exist as well as conditions guaranteeing the employed them searching strategies can be used in an on-line mode for the real-life size of project planning problems.

Provided multiple examples illustrate a way some arbitrary selected cases can be managed by DST4P^3 package. Its current version is aimed at an interactive projects portfolio prototyping aimed at SMEs where the number of simultaneously considered projects do not exceeds 5 and whole number of activities do not exceeds 80. In that context the approach presented can be considered as a new alternative contribution to project-driven production flow management, and its DSS implementation can be applied in make-to-order manufacturing as well as for prototyping of the virtual organization structures.

6. References

[1] Parcher N., Chasemzadech F. - *An integrated framework for project portfolio selection* [in] Int. Journal of Project Management, Vol. 17, No. 4, pp. 207-216, 1999.

[2] Bach I., Bocewicz G., Banaszak Z. - *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 (ed. Nguyen N.T.). *Lecture Notes in Artificial Intelligence* 5027, pp. 767–776, Springer-Verlag, Berlin, Heidelberg 2008.

[3] Bach I., Wójcik R., Bocewicz G. - *Projects portfolio prototyping subject to imprecise activities speci-fication* [in] Proceedings of 14th International Congress of Cybernetics and Systems of WOSC – ICCS'08, pp. 261-272, Wroclaw 2008.

[4] Banaszak Z. - *CP-based decision support for project-driven manufacturing* [in] Perspectives in Modern Project Scheduling, (ed. Józefowska J., Węglarz J.). International Series in Operations Research and Management Science, Vol. 92, pp. 409-437, Springer Verlag, New York 2006.

[5] Banaszak Z., Zaremba M., Muszyński W. - *CP-based decision making for SME* [in] Preprints of the 16th IFAC World Congress (ed. Horacek P., Simandl M., Zitek P.). DVD, Prague 2005.

[6] Barták R. - *Incomplete Depth-First Search Techniques: A Short Survey* [in] Proceedings of the 6th Workshop on Constraint Programming for Decision and Control (ed. Figwer J.), pp. 7-14, 2004.

[7] Beale E.M.L. - *Branch and bound methods for mathematical programming systems* [in] Discrete Optimization II (ed. Hammer P. L., Johnson E. L., Korte B. H.), pp. 201–219. North Holland Publishing Co., 1979.

[8] Bocewicz G., Banaszak Z., Wójcik R. - *Design of admissible schedules for AGV systems with constraints: a logic-algebraic approach* [in] Agent and Multi-Agent Systems: Technologies and Applications (ed. Nguyen N.T., Grzech A., Howlett R.J., Jain L.C.), pp. 578-587. Lecture Notes in Artificial Intelligence 4496, Springer-Verlag, Berlin, Heidelberg 2007.

[9] Bocewicz G., Wójcik R., Bzdyra K. - *Fuzzy Logic And Logic-Algebraic Method For Constraint Programming - Driven Project Prototyping* [in] Zarządzanie wiedzą i technologiami informatycznymi (ed. Orłowski C., Kowalczuk Z., Szczerbicki E.), pp. 317-326. PWNT, Gdańsk 2008.

[10] Bubnicki Z. - *Logic-algebraic method for a class of knowledge based system* [in] Computer Aided Systems Theory. Lecture Notes in Computer Science (ed. Picher F., Moreno Diaz R.). Springer-Verlag, Berlin 1997.

[11] Bubnicki Z. - *Learning processes and logic-algebraic method for the systems with knowledge representation* [in] Systems Analysis and Management. PAS, Warsaw 1999.

[12] Chanas S., Komburowski J. - *The use of fuzzy variables in PERT* [in] Fuzzy Sets and Systems, No. 5(1), pp. 11-19, 1981.

[13] Dubois D., Fargier H., Fortemps P. - *Fuzzy scheduling: Modeling flexible constraints vs. coping with incomplete knowledge* [in] European Journal of Operational Research, No. 147, pp. 231 – 252, 2003.

[14] Linderoth T., Savelsbergh. M.W.P. - *A computational study of search strategies in mixed integer programming* [in] INFORMS Journal on Computing, No. 11, pp.173–187, 1999.

[15] Martinez, E.C., Duje D., Perez G.A. - *On performance modeling of project-oriented production* [in] Computers and Industrial Engineering, Vol. 32, pp. 509-527, 1997.

[16] Piegat A. - *Fuzzy modeling and control.* Exit, Warsaw 1999.

[17] Schutle H., Smolka G., Wurtz J. - *Finite Domain Constraint Programming in Oz.* German Research Center for Artificial Inteligence, Saarbrucken 1998.

[18] Van Hentenryck P. - *Constraint Logic Programming* [in] Knowledge Engineering Review, No. 6, pp. 151–194, 1991.

[19] Zimmermann H.J. - *Fuzzy sets theory and its applications.* Kluwer Academic Publishers, London 1994.

APPENDIX A

Imprecise variables specified by fuzzy sets and determined by convex membership function can be characterized by α - cuts (Piegat, 1999), and then defined by pairs (a1):

$$(A_i, \alpha) \tag{a1}$$

where:

$A_i = \{A_{z_i,1}, A_{z_i,2}, \ldots, A_{z_i,lz}\}$ finite set of so called z - cuts,

$\alpha_{i,j} = \{\alpha_{i,1}, \alpha_{i,2}, \ldots, \alpha_{i,lz}\}$ – is a set $A_{z_i,1}, A_{z_i,2}, \ldots, A_{z_i,la}$ of values corresponding to α - cuts at levels $\alpha_{i,j}$,

lz – a number of z-cuts

and (a2)

$$A_{z_i,k} = [a_{i,k}, b_{i,k}]_N \tag{a2}$$

where:

$a_{i,k}$, $b_{i,k}$ – is the smallest and the highest value of the k-th α - cut, $a_{i,k}, b_{i,k} \in N$.

The z-cut can be seen as a discretized form of the α - cut, i.e. $A_{z_i,k} = A_{\alpha_{i,k}} \cap N$, see Fig. 1a.

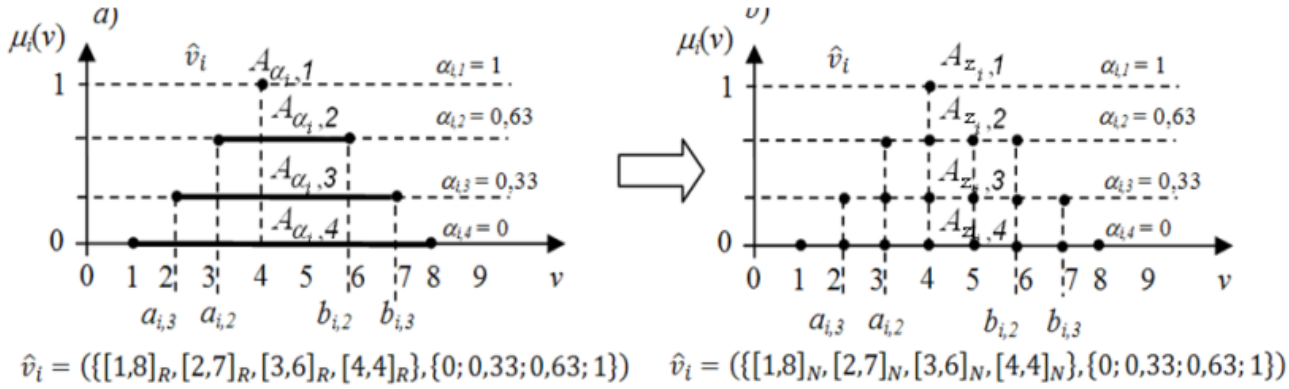

Figure 1a. Fuzzy set $\hat{v}_i$ specified by: a) α - cuts, b) discretized α - cuts, i.e., z-cuts
(source: self study)

Note, that in the assumed specification distinct values are represented by relevant singletons.

Imprecise character of decision variables, e.g., $\hat{x}_{i,j}$, $\hat{t}_{i,j}$, implies imprecise character of employing them constraints, which in turn can be considered as a consequence of implementation of assumed operations. Therefore, consider the set of fuzzy operations: **„≙", „≺", „≻",** encompassing standard algebraic operations such as: =, ≠, <, >, ≥, ≤. Of course, the considered fuzzy operations linking two fuzzy variables $\hat{v}_i$, $\hat{v}_l$ have to follow the condition (a3):

$$E(\hat{v}_i \prec \hat{v}_l) + E(\hat{v}_i \triangleq \hat{v}_l) + E(\hat{v}_i \succ \hat{v}_l) = 1 \quad (a3)$$

where:

E(a) – the fuzzy logic value of the proposition a, $E(a) \in [0,1]$.

In order to define fuzzy operations used for description of the deadlock avoidance conditions (a10) the following auxiliary sets v_i^L, v_i^*, v_i^P and v_l^L, v_l^*, v_l^P are defined as well as the concept of a size of fuzzy variable S_i and the size of subsets S_i^L, S_i^P, S_l^L, S^*, S_i^P of S_i.

For each pair of fuzzy variables $\hat{v}_i$, $\hat{v}_l$ defined by $\{(\mu_i(v), v)\}, \forall v \in K_i$, where: K_i is the domain of the variable $\hat{v}_i$, the following sets can be distinguished: v_i^L, v_i^*, v_i^P i v_l^L, v_l^*, v_l^P. For instance, for the set $\hat{v}_l$ the following subsets can be determined:

v_i^L – the set composed of elements v being less (smaller) than all elements from $\hat{v}_l$,

$v_{i,j}^*$ – the set of elements shared with $\hat{v}_l$,

v_i^P – the set composed of elements v being greater (bigger) than all elements from $\hat{v}_l$.

The sets v_i^L, v_i^*, v_i^P are defined as follows:

$$v_i^L = \{(\mu_i^L(v), v)\}, \forall v \in K_i, \quad (a4)$$

where:

$$\mu_i^L(v) = \begin{cases} \mu_i(v) - \mu_l(v) \text{ if } \mu_i(v) \geq \mu_i(v), v < w_{min} \\ 0 \quad \text{if } \mu_i(v) < \mu_i(v), v < w_{min} \text{ or } v \geq w_{min} \end{cases}$$

$$w_{min} = \min\{K_w\}, K_w = \{v: v \in K_i, \mu_l(v) = 1\},$$

$$v_i^* = \{(\mu_i^*(v), v)\}, \forall v \in K_i, \quad (a5)$$

where:

$$\mu_i^*(v) = \min\{\mu_i(v), \mu_l(v)\}$$

$$v_i^P = \{(\mu_i^P(v), v)\}, \forall v \in K_i, \quad (a6)$$

where:

$$\mu_i^P(v) = \begin{cases} \mu_i(v) - \mu_l(v) \text{ if } \mu_i(v) \geq \mu_i(v), v > w_{max} \\ 0 \quad \text{if } \mu_i(v) < \mu_i(v), v > w_{max} \text{ or } v \leq w_{max} \end{cases}$$

$$w_{max} = \max\{K_w\}, K_w = \{v: v \in K_i, \mu_l(v) = 1\}.$$

Subsets v_l^*, v_l^P corresponding to the fuzzy variable $\hat{v}_l$ are defined in the same way.

To each fuzzy variable $\hat{v}_i$, $\hat{v}_l$ and the corresponding subset v_i^L, v_i^*, v_i^P, v_l^L, v_l^*, v_l^P an associated size value can be determined. For instance, the size value S_i

corresponding to the fuzzy variable v_i, and specified in terms of z-cuts can be defined as (a7):

$$S_i = \sum_{k=1}^{lz} \left\| A_{z_i,k} \right\|, \tag{a7}$$

where:

$\left\| A_{z_i,k} \right\|$ – a number of elements of the set $A_{z_i,k}$.

In the similar way the values S_i^L, S_l^*, S_i^P, S_l^L, S_i^*, S_l^P corresponding to the sets v_i^L, v_i^*, v_i^P, v_l^L, v_l^*, v_l^P are defined.

In the case considered, the equation $S_l^* = S_i^*$ holds for the given v_i^*, v_l^* because the decision variables $\hat{v}_i$, $\hat{v}_l$ belong to the time domain. Therefore, for the sake of simplicity, in further considerations the sizes S_l^*, S_i^* are denoted by the same symbol S^*.

Given fuzzy variables $\hat{v}_i$, $\hat{v}_l$. Consider algebraic-like fuzzy operations following the condition (a3). Fuzzy logic value of the proposition $\hat{v}_i \widehat{=} \hat{v}_l$ is defined by (a8):

$$E(\hat{v}_i \widehat{=} \hat{v}_l) = \frac{2S^*}{S_i + S_l} \tag{a8}$$

where:

S_i – the size of $\hat{v}_i$,

S_l – the size of $\hat{v}_l$,

S^* – the size of the common part of sets $\hat{v}_i$, $\hat{v}_l$.

Fuzzy logic value of the proposition $\hat{v}_i \widehat{<} \hat{v}_l$ is defined by (a9):

$$E(\hat{v}_i \widehat{<} \hat{v}_l) = \frac{S_i^L + S_l^P}{S_i + S_l} \tag{a9}$$

where:

S_i – the size of $\hat{v}_i$, S_l – the size of $\hat{v}_l$,

S_i^L– the size of v_i^L,

S_l^P– the size of v_i^P.

Fuzzy logic value of the proposition $\hat{v}_i \widehat{>} \hat{v}_l$ is defined by (a10):

$$E(\hat{v}_i \widehat{>} \hat{v}_l) = \frac{S_i^P + S_l^L}{S_i + S_l} \tag{a10}$$

Fuzzy logic value of the proposition $\hat{v}_i \widehat{\geq} \hat{v}_l$ is defined by (a11):

$$E(\hat{v}_i \widehat{\geq} \hat{v}_l) = \frac{2S^* + S_i^P + S_l^L}{S_j + S_l} \tag{a11}$$

Fuzzy logic value of the proposition $\hat{v}_i \widehat{\leq} \hat{v}_l$ is defined by (a12):

$$E(\hat{v}_i \widehat{\leq} \hat{v}_l) = \frac{2S^* + S_i^L + S_l^P}{S_i + S_l} \tag{a12}$$

Formulaes (a8), (a9), (a10), (a11), (a12) allow one to design constraints describing basic relations among two fuzzy variables, such as equality, less than, greater than, less or equal, and greater or equal.

In order to allow one to consider other constraints, e.g., taking into acount distinct variables, the fuzzy operations such as fuzzy addition and fuzzy subtraction have to be employed as well. The relevant operations „$\hat{+}$”, „$\hat{=}$” can be found in [17].