*decision support, constraint logic programming, production planning, modeling, scheduling*

Zbigniew BANASZAK*, Jerzy JÓZEFCZYK**

# TOWARDS DEDICATED DECISION SUPPORT TOOLS: CLP-BASED APPROACH

**Abstract**

*Constraint programming (CP) is an emergent software technology for declarative description and effective solving of large combinatorial problems especially in areas of integrated production planning. In that context, the CP can be considered as a well-suited framework for development of decision making software supporting small and medium size enterprises in the course of Production Process Planning (PPP). The problem considered regards of finding of computationally effective approach aimed at scheduling of a new project subject to constraints imposed by a multi–project environment. In other words, we are looking for an answer whether a given production order specified by its cost and completion time can be accepted in a given manufacturing system specified by available production capability, i.e., the time-constrained resources availability. The problem belongs to a class of multi-mode case project scheduling problems, where the problem of finding a feasible solution is NP-complete. The aim of the paper is to present the CP modeling framework as well as to illustrate its application to decision making in the case of a new production order evaluation. So, the contribution emphasizes benefits derived from CP-based DSS and focuses on constraint satisfaction driven decision-making rather than on an optimal solution searching.*

## 1. INTRODUCTION

Today's manufacturing environment can be characterized in terms of many factors - the maturity of manufacturing procedures, technologies and standards; the efficiency of logistic chains and global telecommunication infrastructure; the penetration of artificial intelligence methods in the area of control and decision support; and practically unlimited computing resources. But the key factor for companies confronting the challenge of remaining competitive in an era of globalization is undoubtedly the capability to fast and accurate decision making, especially in project management domain.

* Prof., Technical University of Koszalin, Department of Electronics and Informatics, 75-453 Koszalin, Poland, e-mail: banaszak@tu.koszalin.pl

** Prof., Systems Research Institute of Polish Academy of Sciences, Laboratory of Knowledge Systems and Artificial Intelligence, Podwale St. 75, 50–449 Wroclaw, Poland, e-mail: Jerzy.Jozefczyk@pwr.wroc.pl

Currently, the field of project-oriented management of manufacturing systems is driven primarily by market forces. Some of the most challenging issues that arise in the domain of distributed manufacturing technology and management include manufacturability analysis, validation and evaluation of process plans, partnership in virtual enterprises, process design, and optimization of production plans and schedules. These issues are easily unified within a framework of a project-driven manufacturing concept which is focusing on the Small and Medium size Enterprises (SMEs) where products are manufactured based on a make-to-order or a build-to-order principle.

Most companies, particularly SME have to manage various projects, which share a pool of constrained resources, taking into account various objectives at the same time. Due to the surveys conducted about 80% of companies have to deal with multiple projects, what corresponds to the other data stating that about 90% of all projects occur in the multiproject context. Since the project management problems belong to a class of NP-complete ones, thus the new methods and techniques aimed at real-life constraints imposing on-line decision making are of great importance. Such methods enhancing an on-line project management, and supporting a manager in the course of decision making, e.g. in the course of evaluation whether a new project can be accepted to be processed in a multi-project environment of a manufacturing system at hand or not, could be included into Decision Support Systems (DSSs) tools integrated into standard project management software like MS Project or CA-Super Project [1].

Regardless of its character and scope of business activities a modern enterprise, has to build a project-driven development strategy in order to respond to challenges imposed by growing complexity and globalization. Managers need to be able to utilize a modern DSS as to undertake optimal business decisions in further strategic perspective of enterprise operation. In this context this contribution covers various issues of decision making while providing the concept of Constraint Programming (CP) as well as modeling and designing of decision support tools aimed at management in SMEs and/or the associated Extended Enterprise.

The main objective of DSS aimed at the production flow planning is the coordination of processes and activities related to work order processing, i.e., regarding the transportation, inventory management, warehousing and production. In other words the goal is to achieve a well-synchronized behavior of dynamically interacting components, where the right quantity of the right material is provided in the right place, and at the right time [1]. The decision making regards of the question: Whether a given production order specified by its cost and completion time can be accepted for processing in a SME specified by available production capability, i.e., the time-constrained resources availability (see Fig.1).The problem considered regards of finding of computationally effective approach aimed at scheduling of a new project subject to constraints imposed by a multi–project environment.

That is worth to note that the currently available software tools allow pre-emption; however, they are not designed to cope with company production capability constraints in terms of resource and time availability. Moreover, they do not permit to consider production planning in a unified way to enable an integrated approach to such different tasks as production and transportation routings, production and transportation batch sizing as well as tasks scheduling.

In that context, Constraint Programming/Constraint Logic Programming (CP/CLP) languages [6], [7], employing the constraints propagation concept and by providing unified constraints specification, seem to be well suited for modeling of a company real-life and day-to-day decision-making [4], [5], [9], [10] process.

The rest of the paper is organized as follows: Section 2 describes the modeling framework enabling to state the problem. A concept of the decomposition reference model of constraint satisfaction problem that stands behind searching for a feasible production flow prototyping is then presented in Section 3. The issues of the state space pruning strategies development based on the concept of a reference model of constraint satisfaction problem decomposition are discussed in Section 4. In Section 5, a concept of the CP/CLP-based approach to DSS designing aimed at a SME is investigated. Conclusions are presented in Section 6.

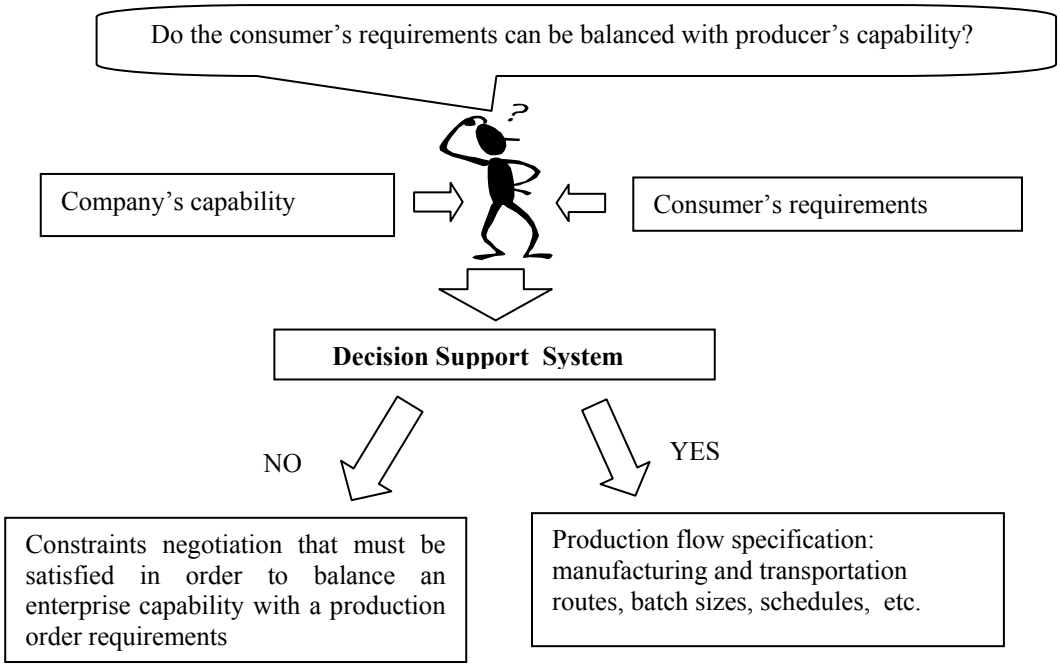



Fig.1. Decision making problem: Do the consumer's requirements can be balanced with producer's capability?

## 2. CP-BASED PROBLEM SPECIFICATION

Constraint programming (CP) is an emergent software technology for declarative description and effective solving of large combinatorial problems especially in 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, hence the idea of CP is to solve problems by stating requirements (constraints) specifying a problem at hand, and then finding a solution satisfying all the constraints [6].

CP is a framework for solving combinatorial problems specified by pairs: **<a set of variables and associated to them domains, a set of constraints restricting the possible combinations of the variables' values>.** Constraints propagation, i.e., reference engine, is based on the idea of using constraints actively to prune the search space. The scope 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 a search tree [7], [15].

A constraint satisfaction problem $CSP = ((X,D),C)$, can be stated as follows. Consider a finite set of n variables $X = \{x_1,x_2,...,x_n\}$, their finite and discrete domains $D = \{D_1,D_2,...,D_n\}$, where $D_i = \{d_{i1},d_{i2},...,d_{ir}\}$, and a finite set of constraints $C = \{c_1,c_2,...,c_m\}$. Each constraint treated as a predicate can be seen as an n-ary relation defined on the Cartesian product $D_1 \times D_2 \times ... \times D_n$.

The solution to the *CSP* is a vector $(d_{1i},d_{2k},...,d_{nj})$ such that the entries assignments satisfy all the constraints *C*. So, the task is to find values of variables satisfying all the constraints, i.e., a feasible valuation. In general case, however, the constraints can express any arbitrary analytical and/or logical formulae as well as bind variables with different non-numerical event domains.

An inference engine employed consists of the following two stages: constraints propagation and variables distribution, respectively. In order to illustrate its mechanism let us consider the $CSP = ((X,D),C)$, where $X = \{x_1, x_2, x_3\}$ is the set of variables and $D = \{D_{x1},D_{x2},D_{x3}\}$ is the set of their domains: $D_{x1} = \{1,2,3,4,5,6\}$, $D_{x2} = \{1,2,3,4,5,6\}$, $D_{x3} = \{1,2,3,4,5,6\}$. Suppose the following set of constraints $C = \{\alpha,\beta,\gamma\}$, where $\alpha: x_1 \geq x_2 + 1$, $\beta: x_2 \geq x_3 + 2$, $\gamma: x_3 < x_1 - x_2$.

Due to the first stage one of possible ways of constraints propagation is shown below:

$D_{x1} = \{1,2,3,4,5,6\}$
$D_{x2} = \{1,2,3,4,5,6\}$ $\xrightarrow{\alpha}$ $D_{x2} = \{1,2,3,4,5\}$
$D_{x3} = \{1,2,3,4,5,6\}$ $D_{x3} = \{1,2,3,4,5,6\}$

$D_{x1} = \{2,3,4,5,6\}$ $D_{x1} = \{4,5,6\}$
$D_{x2} = \{1,2,3,4,5\}$ $\xrightarrow{\beta}$ $D_{x2} = \{3,4,5\}$
$D_{x3} = \{1,2,3,4,5,6\}$ $D_{x3} = \{1,2,3\}$

$D_{x1} = \{4,5,6\}$ $D_{x1} = \{5,6\}$
$D_{x2} = \{3,4,5\}$ $\xrightarrow{\gamma}$ $D_{x2} = \{3,4\}$
$D_{x3} = \{1,2,3\}$ $D_{x3} = \{1,2\}$.

Note, the initial space containing of 6 x 6 x 6 = 216 potential solutions is reduced to its subspace size of 2 x 2 x 2 = 8.

In order to select a subset of feasible solutions the another stage regarding the variables distribution has to be performed, e.g., for each element of the domain $D_{x1} = \{5,6\}$ the phase of constraints propagation is once more applied. So, the resultant set of feasible solutions consists of the following combinations of variables value assignment:

$x_1, x_2, x_3$ $x_1, x_2, x_3$ $x_1, x_2, x_3$
(5 , 3 , 1), (6, 3, 1), (6, 4, 1).

The illustration of searching tree provided by **OZ Explorer** is shown in Fig.2. Of course, the stage of variables distribution my start with other domains as well, e.g., from $x_2$ or $x_3$. In case such distributions do not result in unique variables' domains, the stage of variables distribution (for example for $x_2$) has to be performed again. The situation when constraints propagation and/or variables distribution result in an empty set of feasible solutions corresponds to discrepancy of the CSP considered.

In that context, the CP can be considered as a well-suited framework for development of decision making software aimed at support of the SMEs in the course of the Production

Process Planning (PPP). Because of its declarative nature, for a use that is enough to state *what* has to be solved instead *how* to solve it [4].

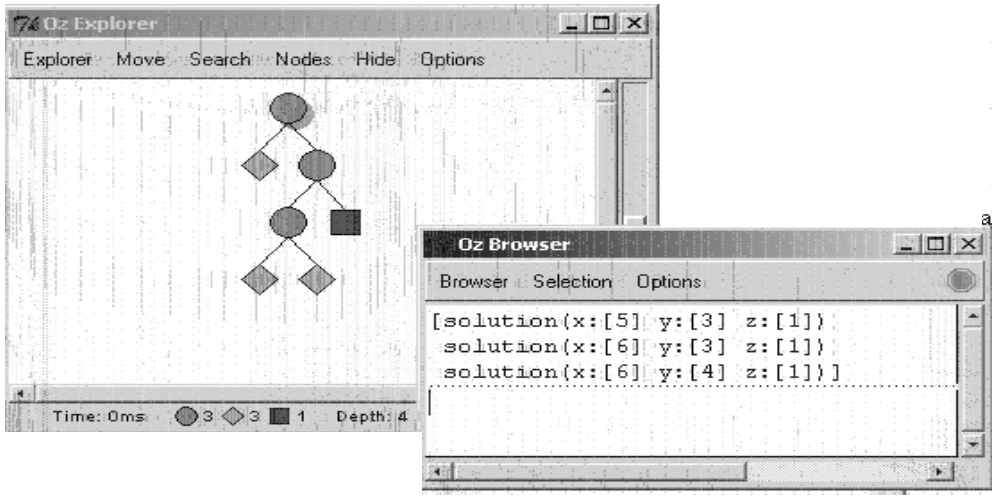



Fig.2. Searching tree and the set of feasible solutions

The aim of the contribution is to present the CP modeling framework as well as to illustrate its application to decision making in the case of a new production order evaluation, i.e. PPP. Finding an answer to the question whether a given work order can be accepted to be processed in a production system seems to be a fundamental from the customer-driven, and highly competitive market point of view. In that context decision making regards to the question whether enterprise's capability allows to fulfill constraints imposed by the production order requirements, i.e. whether its completion time, batch size, and its delivery period satisfy the customer requirements while satisfying constraints imposed by the enterprise configuration taking into account available resources, know how, experience, and so on. In the case of the response to this question being positive, i.e. there exist a way guaranteeing to complete a production order, the next question regards of finding of the most efficient one (e.g. as to be competitive on the market) [15].

## 3. MODELLING FRAMEWORK

Consider $CSP = ((X,D),C)$, such that $X = \{A,B\}$, $D = \{D_A,D_B\}$, where $D_A = \{1,2,3\}$, $D_B = \{3,4,...,9\}$, and $C = \{c_1,c_2\}$, where $c_1 = P[\, B \geq 3 \cdot A \,]$, and $c_2 = P[\, A + B > 9 \,]$. Depends on the order in which variables are distributed the time required to obtain a set of feasible solutions may differ dramatically. In the case considered, starting with variable $A$ requires twice less searching than in the case when variables distribution begin from variable $B$, see Fig.3.

It's easy to notice, that efficiency of a searching strategy can be evaluated in advance on the base of domains' sizes. In order to discuss this possibility let us introduce a concept of a reference model of the CSP decomposition. Consider the $CSP = ((X,D),C)$, where $X = \{x_1,x_2,...,x_{12}\}$, $D = \{D_1,D_2,...,D_{12}\}$, $C = \{c_1,c_2,...,c_8\}$, and where:

$c_1 = P[x_1,x_2,x_3]$, $c_2 = P[x_2,x_4,x_5]$, $c_3 = P[x_4,x_6]$, $c_4 = P[x_7,x_8]$, $c_5 = P[x_4,x_7]$,
$c_6 = P[x_9,x_{10}]$, $c_7 = P[x_8,x_9]$, $c_8 = P[x_{11},x_{12}]$.

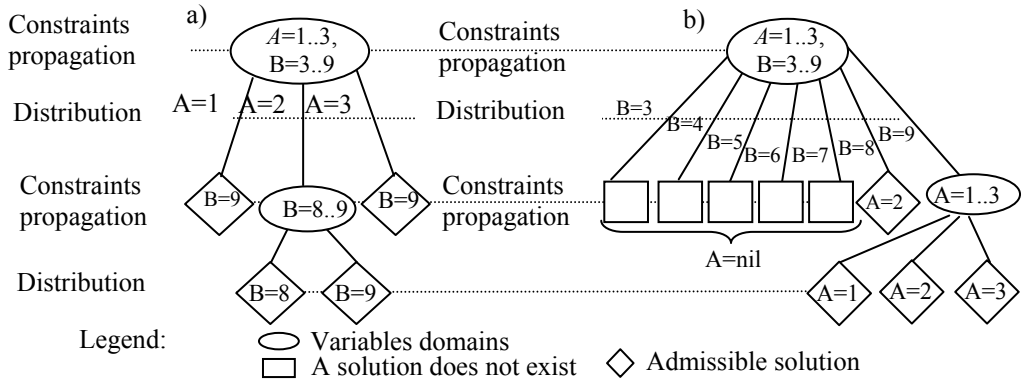



Fig.3. Decision variables distribution, a) B follows A, b) A follows B

The problem in natural way decomposes into subproblems, in particular to elementary subproblems, which are not further decomposed. The elementary problems can be seen as problems encompassed by constraints, for instance the elementary problem associated to the constraint $c_8 = P[x_{11},x_{12}]$ can be stated as follows $CSP_8 = (\{\{x_{11},x_{12}\},\{d_{11},d_{12}\}\},\{c_8\})$.

In general case any *CSP* may be decomposed, however, either into a set of loosely coupled problems or into a set of strongly coupled problems.

The problems $CSP = ((X,D),C)$ and $CSP' = ((X',D')C')$ are loosely coupled ones if the following conditions hold:

$$\begin{aligned} &\text{i)} \quad X \cap X' = \varnothing \\ &\text{ii)} \quad \forall\, c \in C : D(c) \cap X' = \varnothing \\ &\text{iii)} \quad \forall\, c' \in C' : D(c') \cap X = \varnothing \end{aligned} \tag{1}$$

where:

$D(c)$ – is the set of variables included in the constraint $c$.

In turn any element of a set of loosely coupled problems is a strongly coupled problem following the condition below

$$\forall X_j^*, X_i^* \subset X^*,\ \exists X_a^*, X_b^*, \ldots, X_z^* \subset X^* : X_j^* \cap X_a^* \neq \varnothing \wedge X_a^* \cap X_b^* \neq \varnothing \wedge \ldots \wedge X_z^* \cap X_i^* \neq \varnothing \tag{2}$$

where:

$SP^* = ((X^*,D^*)C^*)$ – a strongly coupled problem composed a set of elementary problems $\{SP_1^*, SP_2^*, \ldots, SP_k^*\}$

It means, that for any two pairs of elementary problems of a strongly coupled problem there exists either nonempty subset of common variables or there exists a set of elementary

subproblems constraints of which provide a chine of nonempty subsets of variables (following the pairs of elementary problems while linking the considered pair).

In turn, a strongly coupled problem may be decomposed into a set of so called dependent problems which are strongly coupled ones. It is assumed, however, that any pair of dependent problems follows the condition (3).

$$\forall X_j\hat{},X_i\hat{} \subset X,\ \exists X_k{*} \subset X \mid X_j\hat{} \cap X_i\hat{} = \varnothing \wedge X_k{*} \cap (\cup\{X'{*} \mid X'{*} \in X_j\hat{}\}) \neq \varnothing \wedge \ \ldots \quad (3)$$
$$\ldots \ \wedge \ X_k{*} \cap (\cup\{X{*} \mid X{*} \in X_i\hat{}\}) \neq \varnothing$$

where:

$CSP_j\hat{} = ((X_j\hat{},D_j\hat{}),C_j\hat{})$, $CSP_i\hat{} = ((X_i\hat{},D_i\hat{}),C_i\hat{})$ - are the strongly coupled subproblems of the strongly connected problem $CSP = ((X,D),C)$,

$CSP{*} = ((X{*},D{*}),C{*})$ – elementary subproblem of the problem $CSP = ((X,D)C)$.

Illustration of the $CSP = ((X,D),C)$ decomposition into the sets of loosely and strongly coupled as well as dependent subproblems is shown in Fig.4.

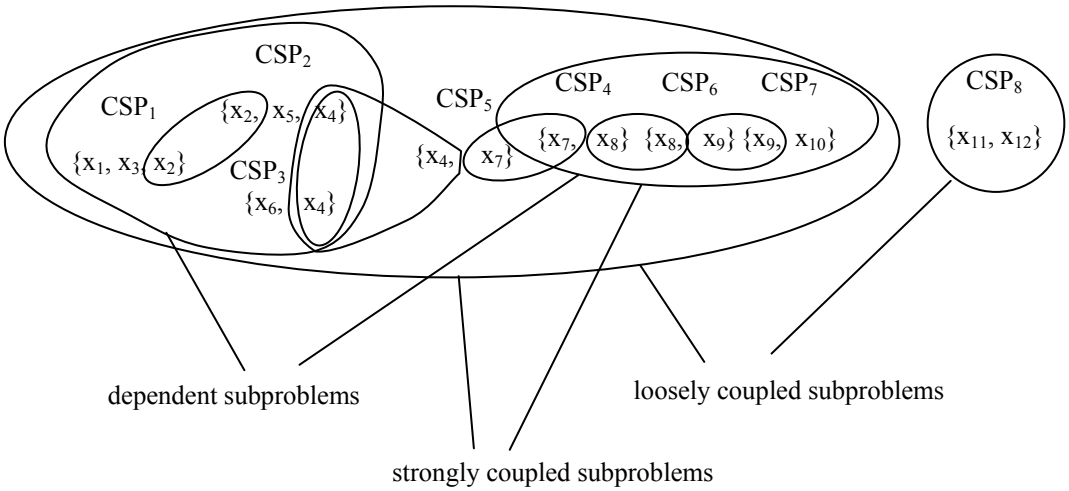



Fig.4. Decomposition of CSP = ((X,D),C) into losely and strongly coupled as well as dependent subproblems.

In order to summarize the above considerations it should be noted that a $CSP = ((X,D),C)$ can be decomposed into a set of:

- elementary subproblems,
- loosely coupled subproblems,
- dependent subproblems of a strongly coupled problems.

Instead of the first two ways of possible *CSP* decompositions the third one does not lead to a unique decomposition. For instance, besides of the possible decomposition shown in Fig. 5, an alternative the more detailed one can be considered as shown in Fig. 6. Such observation enables to consider a tree of all the potentially available decompositions in a form of a AND/OR –like digraph, see Fig. 7.

Different possibilities of *CSP* decomposition enable one to take into account the real life constraints that follow from:

- a way of a problem specification (i.e., a set of elementary problems recognized)
- a programming language implementation (some structures of dependent problems may or may not be accepted by CP/CLP packages)
- a way of a *CSP* resolution (e.g., the loosely coupled subproblems can be computed parallel within an multiprocessor environment)
- a searching strategy applied (the order of subproblems resolution results in a *CSP* makespan).

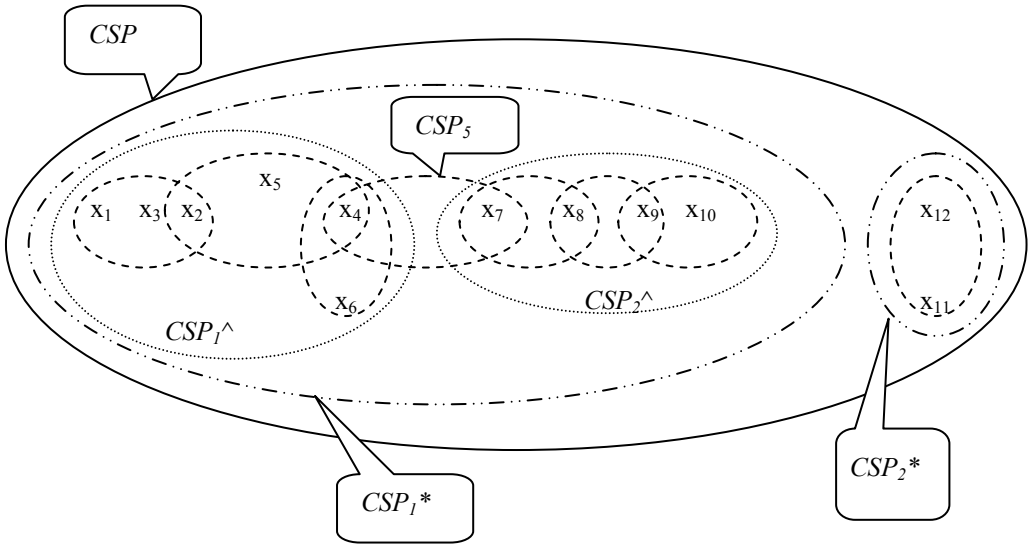



Fig.5. The possible decomposition of the CSP problem into the two loosely coupled subproblems $CSP_1$*, $CSP_2$*, and decomposition of the $CSP_1$* into the two dependent subproblems $CSP_1$^ and $CSP_1$^

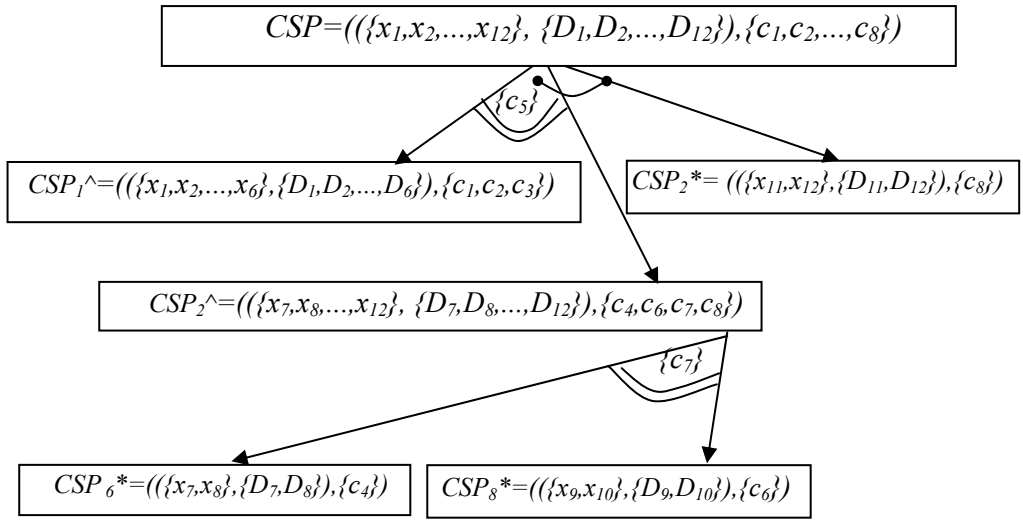
The above observation leads to a concept of a reference model of a *CSP* decomposition, i.e., the model encompassing an object-like nature of the *CSP* structure [12], [15]. So, since each subproblem corresponds to a standard constraint problem structure: *(({a set of decision variables,} (a set of variable domains}), {a set of constraints})*, hence some AND/OR – like graph representation can be used both in the course of analysis of the *CSP* programming (i.e. CP/CLP problem specification) and its resolution.

The concept of the *CSP* decomposition reference model provides a well suited framework for preliminary evaluation of search trees pruning strategies.

## 4. STATE SPACE PRUNING STRATEGIES

Consider the $CSP = ((X,D),C)$, where $X = \{x_1,x_2,x_3\}$, $D = \{D_1,D_2,D_3\}$, $C = \{c_1,c_2,c_3\}$, and where: $D_1 = D_2 = D_3 = \{1,2\}$ , $c_1 = P[x_1]$, $c_2 = P[x_2]$, $c_3 = P[x_3]$. The CSP consists of the

following three elementary problems: $CSP_1 = (\{\{x_1\},\{D_1\}\},\{c_1\})$, $CSP_2 = (\{\{x_2\},\{D_2\}\},\{c_2\})$, $CSP_3 = (\{\{x_3\},\{D_3\}\},\{c_3\})$.




Legend:
$CSP_2^*$, $CSP_6^*$, $CSP_8^*$ - elementary subproblems,
$CSP_1^\wedge$, $CSP_2^\wedge$, $CSP_2^*$, $CSP_6^*$, $CSP_8^*$ - strongly coupled subproblems
$CSP_1^*$, $CSP_2^* = (\{\{x_1\text{-}x_{10}\},\{D_1\text{-}D_{10}\}\},\{c_1\text{-}c_7\})$ - loosely coupled subproblems,
- decomposition into dependent subproblems
- decomposition into loosely coupled subproblems

Fig.6. Alternative way of the CSP problem decomposition

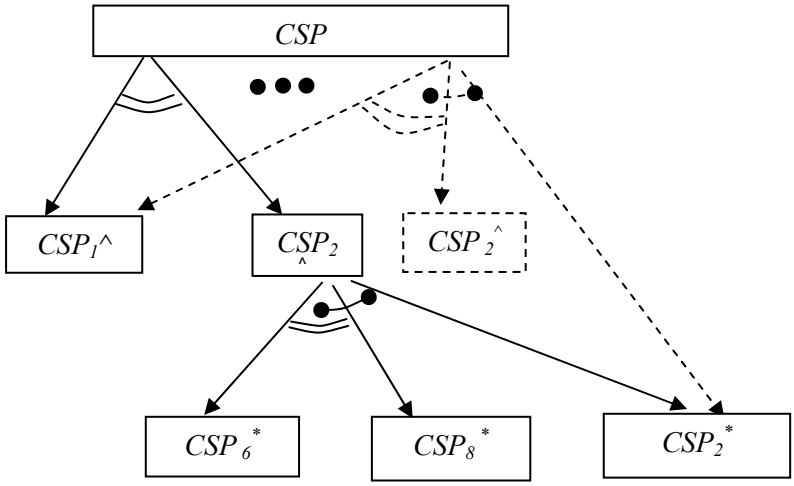



Fig.7. AND/OR-like graph representation of the CSP possible decompositions

The number of possible solutions is equal to 2 ∗ 2 ∗ 2 = 8, however the number of backtrackings required to check their feasibility is greater, and equals to 11 (see Fig. 8). The number of backtrackings can be estimated by the formulae (4).

$$N(j) = \sum_{i=1}^{L} \left(\prod_{h=1}^{i} Z_h^j - 1\right) \tag{4}$$

where:

$k$ – the index of the $k$-th elementary problem of a *CSP*,

$L$ – the number of elementary problems

$j$ – the $j$-th permutation of the set of elementary problem,

$h$ – the index of an elementary problem placed at the h-th position in the $j_k$-th permutation of the set of elementary problem obtained from the *CSP*

$Z_h^j(k)$ – a number of potential assignments of the decision variables of the $k$-th elementary problem placed at the $h$-th position in the $j_k$-th permutation.

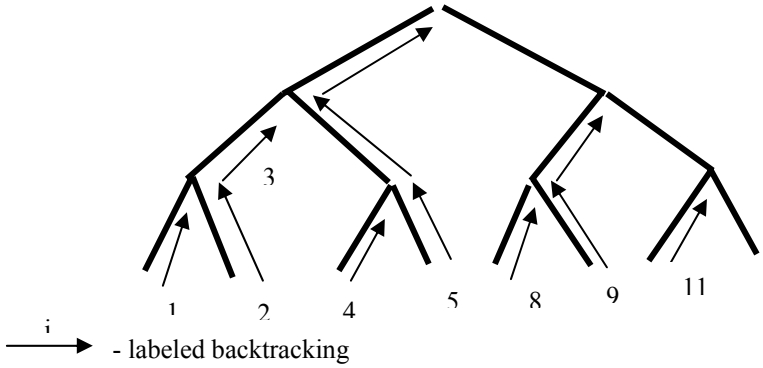



Fig.8. Searching tree encompassed by backtrackings

So, assuming *(PSO₂,PSO₁ ,PSO₃)* as the *j*-th permutation of elementary problems *{PSO₁, PSO₂ ,PSO₃ }* as well as $Z_h^j(2) = Z_h^j(1) = Z_h^j(3) = 2$, the number of backtrackings *N(j)* equals to: 1 + 2∗2 – 1 + 2∗2∗2 – 1 = 11

In general case, however, since the cardinality of a set of possible solutions of each elementary problem *{CSP₁,CSP₂,...,CSP_L}*of CSP can be seen as a multiple of its variables domains, hence the possible orders of CSP resolution are determined by *L!* permutations of the set of elementary problems. Of course, the different permutations lead to the different results, i.e. different numbers of backtrackings.

In order to illustrate this fact let us consider the *CSP = ((X,D),C),* where *X = {x₁,x₂,x₃,x₄,x₅,x₆}, D = {d₁,d₂,d₃,d₄,d₅,d₆}, C = {c₁,c₂,c₃},* and where: $D_1 = D_2 = D_3 = \{1,2\}$ , $D_4 = D_5 = D_6 = \{1,2,3\}$ $c_1 = P[x_1,x_2]$, $c_2 = P[x_3,x_4]$, $c_3 = P[x_5,x_6]$. The CSP considered consists of the following three elementary problems: $CSP_1 = (\{x_1,x_2\},\{D_1.D_2\}),\{c_1\})$, $CSP_2 = (\{x_3.x_4\},\{D_3,D_4\}),\{c_2\})$, $CSP_3 = (\{x_5,x_6\}),\{D_5,D_6\}),\{c_3\})$. Among the possible 3! permutations let us focus on the following two ones: *(PSO₁,PSO₂ ,PSO₃), and (PSO₃,PSO₁,PSO₂).*

Since $CSP_1$ results in possible $2 * 2 = 4$ solutions (assignments), and $CSP_2$ results in possible $2 * 3 = 6$ solutions, and $CSP_3$ results in possible $3 * 3 = 9$ solutions, hence first permutation result in $4 - 1 + 4*6 - 1 + 4*6*9 - 1 = 241$ backtrackings, and the second one in $9 - 1 + 9*4 - 1 + 9*4*6 - 1 = 258$ backtrackings.

In the considered case, however, the way of backtrackings estimation suffers from omitting the number of possible backtrackings at elementary problem levels. Note, that for the case when elementary problem consists of three and more variables a number of required backtracking is bigger than number of possible solutions to the problem.

In order to overcome this disadvantage the modified formulas are proposed (5), (6).

$$N(j) = \sum_{k=1}^{L} \left(\prod_{r=1}^{k} N_r^j(j_k, k) - 1\right) \tag{5}$$

where:

$L$ – the number of elementary problems of a *CSP,*

$j$ – the j-th permutation of a set of the elementary problems of a *CSP*,

$r$ – the index of an elementary problem placed at the r-th position in the $j$-th permutation,

$k$ – the index of the k-th elementary problem,

$j_k$ – the $j_k$-th permutation of a set of the $k$-th elementary problem variables,

$N_r^j(j_k, k)$ – the number of potential backtrackings of the k-th elementary problem resolved due to the $j_k$-permutation of variables, the k-th elementary problem is placed at the r-th position in the j-th permutation

$$N(j_k, k) = \sum_{i=1}^{L(k)} \left(\prod_{h=1}^{i} Z_h^{j_k}(i, k) - 1\right) \tag{6}$$

where:

$k$ – the index of the k-th elementary problem of a *CSP*,

$L(k)$ – the number of the variables of the $k$-th elementary problem,

$i$ – the index of the $i$-th variable of the $k$-th elementary problem,

$j_k$ – the $k$-th permutation of the variables of the $k$-th elementary problem,

$h$ – the index of the variable placed at the $h$-th position in the $j_k$-th permutation of the $k$-th elementary problem variables,

$Z_h^{j_k}(i, k)$ – the cardinality of the i-th variable domain, i.e., the variable placed at the h-the position in the $j_k$-th variables permutation of the $k$-th elementary problem

So, in order to obtain a correct evaluation of the backtracking number required to find a set of admissible solutions of a *CSP*, the permutation of elementary problems as well as variables permutation in each elementary problem have to be assumed. The variables and elementary problems permutation determine the order of variables substitution and elementary problems resolution, respectively.

In the case considered, for the set of elementary problems $\{CSP_1, CSP_2, CSP_3\}$ let us consider:

- the permutation $(CSP_1, CSP_2, CSP_3)$ and the following variables permutation: $(x_1, x_2)$ for $CSP_1$, $(x_3, x_4)$ for $CSP_2$, and $(x_5, x_6)$ for $CSP_3$,
- the permutation $(CSP_3, CSP_1, CSP_2)$ and the following variables permutation: $(x_1, x_2)$ for $CSP_1$, $(x_3, x_4)$ for $CSP_2$, and $(x_5, x_6)$ for $CSP_3$.

In the first case, since $CSP_1$ results (see formulae (5)) in possible $1 + 2 * 2 - 1 = 4$ solutions (assignments), and $CSP_2$ results in possible $1 + 2 * 3 - 1 = 6$ solutions, and $CSP_3$ results in possible $2 + 3 * 3 - 1 = 10$ solutions, hence due to the formulae (6) the number of backtrackings equals to $4 - 1 + 4*6 - 1 + 4*6*10 - 1 = 275$ backtrackings. In the second case, however, the number of backtrackings equals to $10 - 1 + 10*4 - 1 + 10*4*6 - 1 = 287$ backtrackings.

Assuming a new variables permutation: $(x_1,x_2)$ for $CSP_1$, $(x_3,x_4)$ for $CSP_2$, and $(x_5,x6)$ for $CSP_3$, the relevant numbers of backtrackings equal to: 309, and 357, respectively.

The above observation providing a way of pruning strategies evaluation can be generalized for the case of loosely and strongly coupled subproblems of *CSP*. The formulas allowing one to evaluate the pruning strategies for a given *CSP* decomposition as well as for assumed subproblems and variables permutation see the formulae (7), and (8), respectively.

$$N(j_d) = \sum_{k=1}^{L} \left(\prod_{r=1}^{L} N_r^{j_d}(j_k,k) - 1\right) \tag{7}$$

where:

$L$ – the number of the subproblems obtained due to the $d$-th decomposition of a *CSP*,

$j_d$ – the $j_d$-th permutation of a set of subproblems obtained due to the $d$-th decomposition of a *CSP*,

$r$ – the index of the subproblem placed in the $r$-th position in the $j_d$-th permutation,

$k$ – the $k$-th subproblem of a set of subproblems obtained due to the $d$-th decomposition of a *CSP*,

$j_k$ – the $j_k$-th permutation of a set of the $k$-th subproblem variables,

$N_r^{j_d}(j_k,k)$ – the number of potential backtrackings of the $k$-th subproblem placed at the $r$-th position in the $j_d$-th permutation

$$N(j_k,k) = \sum_{i=1}^{L(k)} \left(\prod_{h=1}^{L(k)} Z_h^{j_k}(i,k) - 1\right) \tag{8}$$

where:

$k$ – the $k$-th subproblem of a set of subproblems obtained due to the $d$-th decomposition of a *CSP*,

$L(k)$ – the number of the variables of the $k$-th subproblem,

$i$ – the index of the $i$-th variable of the $k$-th subproblem,

$j_k$ – the $k$-th permutation of the variables of the $k$-th subproblem,

$h$ – the index of the variable placed at the $h$-th position in the $j_k$-th permutation of the $k$-th subproblem variables,

$Z_h^{j_k}(i,k)$ – the cardinality of the $i$-th variable domain, i.e., the variable placed at the $h$-th position in the $j_k$-th variables permutation of the $k$-th subproblem

In order to summarize the section it should be noticed that since with arcs of a AND/OR graph it is possible to bind weight factors determining the necessary number of searches, hence such representation provides a way to chose the best searching strategy, i.e. a variant with least number of backtrackings. In the case of a *CSP* decomposition into a set of $\{CSP_1,CSP_2,...,CSP_L\}$ the relevant searching tree is shown in Fig. 9.

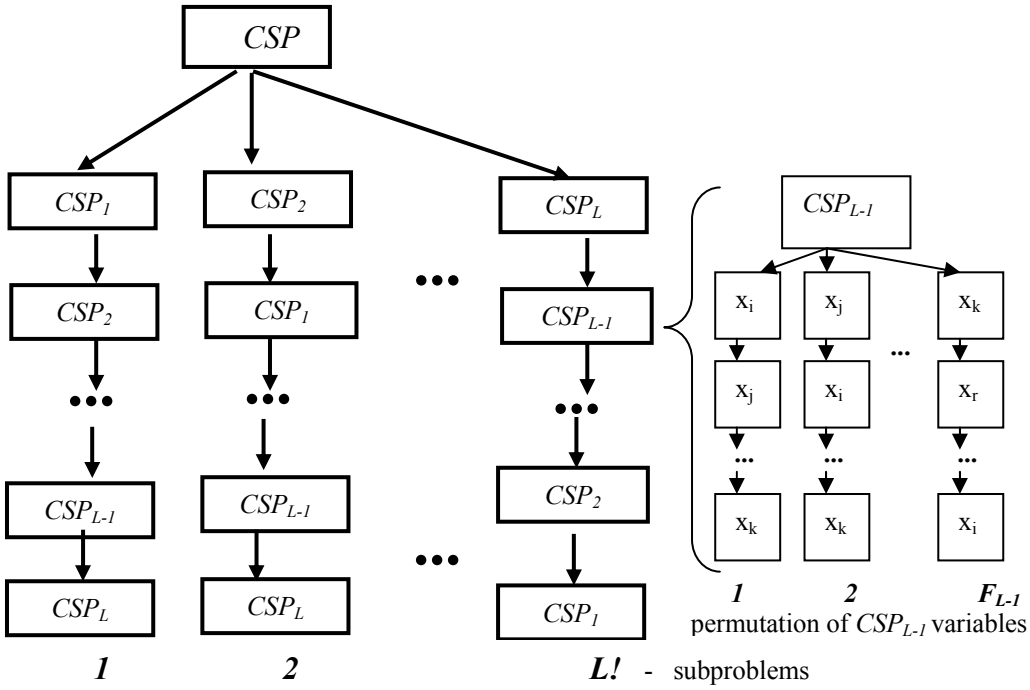



Fig.9. Searching tree for CSP decomposed into the set of elementary problems

In the case considered the number $W$ of possible searching strategies for the *CSP* composed of $L$ elementary problems consisting of the $K_1, K_2,...,K_L$ variables can be estimated due to the upper bound stated by the formulae (9). It means that for the *CSP* consisting of $L$ = 10 elementary problems each of them containing $K_i$ = 2 variables the number of possible searching is equal to $SS = 2^{10}10! \approx 3{,}6 * 10^7$. Note that the estimation formula does not take into account the number of possible *CSP* decompositions!

$$W = KM^{L}L! \tag{9}$$

where

$$W = \max_{i\in\{1,...,L\}}\{K_i\}$$

Therefore the problem of selection of the optimal (i.e. requiring the less backtrackings) state space pruning strategy seems to be at least NP-complete one. So, the branch and bound method can be considered in the course of the best searching strategy selection. The idea standing behind of this concept is shown in Fig. 10.

For a given *CSP* decomposition, i.e., for a given set of subproblems $\{CSP_1, CSP_2, ..., CSP_L\}$ the set of upper bound values $\{W_1, W_2, ..., W_L\}$ is calculated (due to the formulae (7) and/or (8). Than for the subproblem to which the lowest value of the upper bound is assigned the next subproblem is selected as to find the order in which the subproblems should be resolved while requiring the lowest number of backtrackings.

In general case, instead of the upper bound considered till now the other measures (heuristics) could be taken into account. For the illustration of such possibility let us consider the following example.

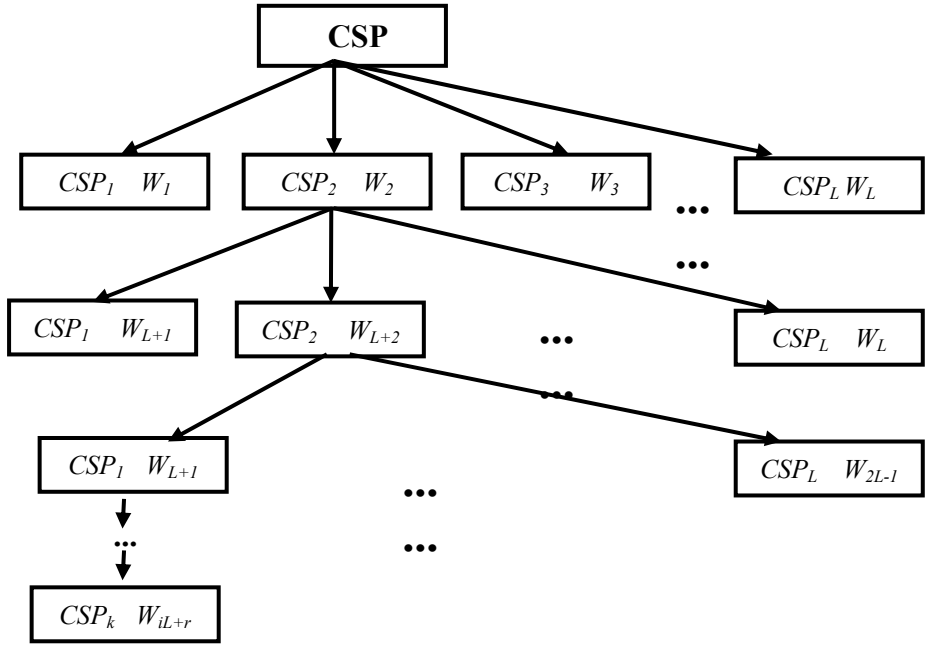



Fig.10. Branch and bound method approach to pruning strategy selection

Given a *CSP = ((X,D),C)* such that $X = \{x_1, x_2, x_3, x_4, x_5\}$, $D = \{D_1, D_2, D_3, D_4, D_5\}$, $D_1 = D_2 = D_3 = D_4 = D_5 = \{1,2,3,...,100\}$, $C = \{c_1, c_2, c_3, c_4\}$ , $c_1 = P[2*x_1+x_2 \leq x_3]$ , $c_2 = P[2*x_2 = x_4]$ , $c_3 = P[x_4*x_5 \leq x_1*x_3]$ , $c_4 = P[x_3*x_4*x_5 \leq 300]$.

As result of constraints propagation, i.e., the reduction of the domains $D_1=\{1,2,...,36\}$, $D_2=\{1,2,...,24\}$, $D_3= \{3,4,...,74\}$, $D_4= \{4,5,...,96\}$, $D_5= \{1,2,...,24\}$ the state space size of $100^5$ is reduced to the size equal to 138 848 256. The constraint influence on the state space size reduction is shown in the Table 1.

Using the results obtained one may consider a searching strategy employing the order of constraints propagation. Such strategy assumes step by step elementary problems resolution emphasizing a dynamic of state space reduction (i.e., a heuristics assuming: "faster state space reduction, shorter searching time"). The evaluation of the possible strategies is shown in the Table 2.

Tab.1. The state space size reduction influenced by constraints.

| Constraints | Domains of decision variables | | | | | The rate of the state space reduction |
|---|---|---|---|---|---|---|
| | $D_1$ | $D_2$ | $D_3$ | $D_4$ | $D_5$ | |
| $c_1$ | 1 - 49 | 1 - 98 | 3 - 100 | 1 - 100 | 1 - 100 | 52,94 % |
| $c_2$ | 1 - 100 | 1 - 25 | 1 - 100 | 4 - 100 | 1 - 100 | 75,75 % |
| $c_3$ | 1 - 100 | 1 - 100 | 1 - 100 | 1 - 100 | 1 - 100 | 0% |
| $c_4$ | 1 - 100 | 1 - 100 | 1 - 100 | 1 - 100 | 1 - 100 | 0% |

Tab.2. The state space reduction pruning strategies based on the step by step constraints propagation

| | Constraints and corresponding state space reduction rate Searching strategy | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Searching Strategy | | % | | % | | % | | % |
| 1 | $c_1$ | 52,49 | $c_2$ | 88,35 | $c_3$ | 88,35 | $c_4$ | 98,61 |
| 2 | $c_1$ | 52,49 | $c_2$ | 88,35 | $c_4$ | 98,61 | $c_3$ | 98,61 |
| 3 | $c_1$ | 52,49 | $c_3$ | 52,94 | $c_2$ | 88,35 | $c_4$ | 98,61 |
| 4 | $c_1$ | 52,49 | $c_3$ | 52,94 | $c_4$ | 53,87 | $c_2$ | 98,61 |
| 5 | $c_1$ | 52,49 | $c_4$ | 53,87 | $c_2$ | 98,61 | $c_3$ | 98,61 |
| 6 | $c_1$ | 52,49 | $c_4$ | 53,87 | $c_3$ | 53,87 | $c_2$ | 98,61 |
| 7 | $c_2$ | 75,75 | $c_1$ | 88,35 | c) | 88,35 | $c_4$ | 98,61 |
| 8 | $c_2$ | 75,75 | $c_1$ | 88,35 | $c_4$ | 98,61 | $c_3$ | 98,61 |
| 9 | $c_2$ | 75,75 | $c_3$ | 75,75 | $c_1$ | 88,35 | $c_d$ | 98,61 |
| 10 | $c_2$ | 75,75 | $c_3$ | 75,75 | $c_4$ | 86,72 | $c_1$ | 98,61 |
| 11 | $c_2$ | 75,75 | $c_4$ | 86,72 | $c_1$ | 98,61 | $c_3$ | 98,61 |
| 12 | $c_2$ | 75,75 | $c_4$ | 86,72 | $c_3$ | 86,72 | $c_1$ | 98,61 |

The elements of the third, fifth, seventh and the last column in the Table 2 are calculated as follows:

- the third column $[1-(49 \cdot 98 \cdot 98 \cdot 100 \cdot 100)/(100^5)] \cdot 100\% = 52{,}49\%$ for the constraint $c_1$
- the fifth column $[1-(49 \cdot 25 \cdot 98 \cdot 97 \cdot 100)/(100^5)] \cdot 100\% = 88{,}35\%$ for the constraint $c_3$
- the seventh column $[1-(49 \cdot 25 \cdot 98 \cdot 97 \cdot 100)/(100^5)] \cdot 100\% = 88{,}35\%$ for the constraint $c_3$
- the last column $[1-(49 \cdot 25 \cdot 98 \cdot 97 \cdot 100)/(100^5)] \cdot 100\% = 88{,}35\%$ for the constraint $c_4$

Note that nonlinear constraints are less effective in the state space size reduction than the linear ones.

## 5. TASK ORIENTED DECISION SUPPORT TOOLS

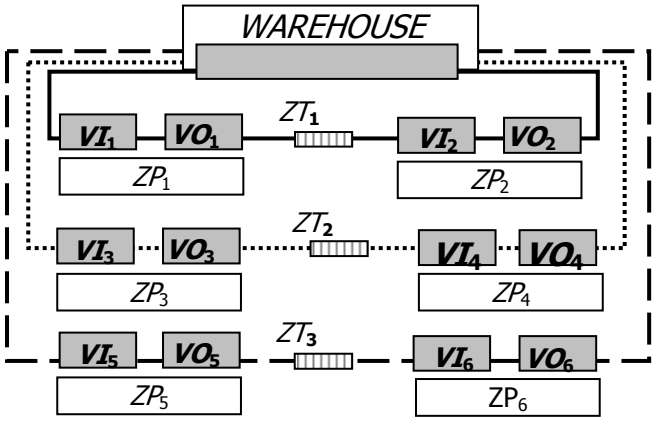
Since the reference model, i.e. an object-like AND/OR graph framework, provides the possibility to estimate the number of decision variables domains values substitution, hence the influence of data structure, sequence of elementary subproblems solution and domain size on decision making time may be evaluated as well. In other words, the model considered provides a well suited framework for development (taking into account the ways of possible problem specification, available CP/CLP languages, and searching strategies) of a CP-based programming methodology as well as the development of the task oriented software tools aimed at the SMEs decision support, e.g. regarding production flow planning.

Many often repeating question regards of the question: Whether in a given shop employed with the machine tools, automated guided vehicles (AGVs), and buffers and warehouses a production order submitted can be completed due assumed period? A production order includes the type of final product to be manufactured, the required quantity and defined due date as well as a final cost a customer has to pay. The product type is usually defined as the combination of components that a machine toll is capable of handling. The product type is

specified by an execution route (production routing) that includes one or more execution steps. In general case, however, a given final product can be produced differently due to alternative execution routes that differ in the execution steps.

The question stated above is usually considered under assumption in the enterprise executes several project-like production orders, however some production capability are still available. So, the question is whether the production order at hand can be accepted for execution in an enterprise where some resources availability is constrained in time?

The typical workshop layout (see Fig. 11) enables to distinguish two kinds of flows regarding production and transportation routings, respectively. These flows interact each other (e.g., via production and batch sizes, AGVs, buffers and warehouses capacities, and so on) and falls into the following two main subproblems: manufacturing and transportation that may be resolved in two alternative orders (see Fig.12).




Legend:
$ZP_j$ – the i-th machine tool, $VI_j$/$VO_j$– the i-th input/output buffer,
$ZT_i$ – *the i-th* AGV.

Fig.11. The workshop layout

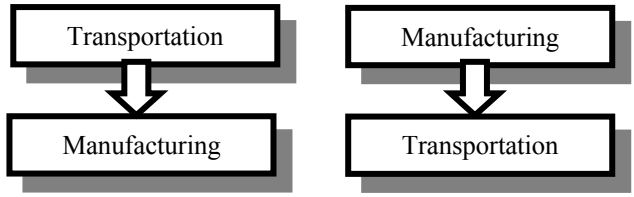



Fig.12. The possible orders of production flow subproblems resolution

The possible decomposition of the production flow planning is shown in Fig 13. So, the relevant *CSP* problem consists of two subproblems corresponding to the manufacturing (denoted by $CSP_1$) and transportation (denoted by **G**) subflows. In turn, the subproblem of manufacturing falls into production routing (denoted by **A**) and production batch planning and scheduling (denoted by $CSP_2$). The last subproblem falls into subproblem of scheduling (denoted by **F**), and production batch planning (denoted by $CSP_{2,1}$). Finally, the production batch planning subproblems falls into the calculation of the number of production batches (denoted by **B**) and production batch sizing (denoted by **E**).

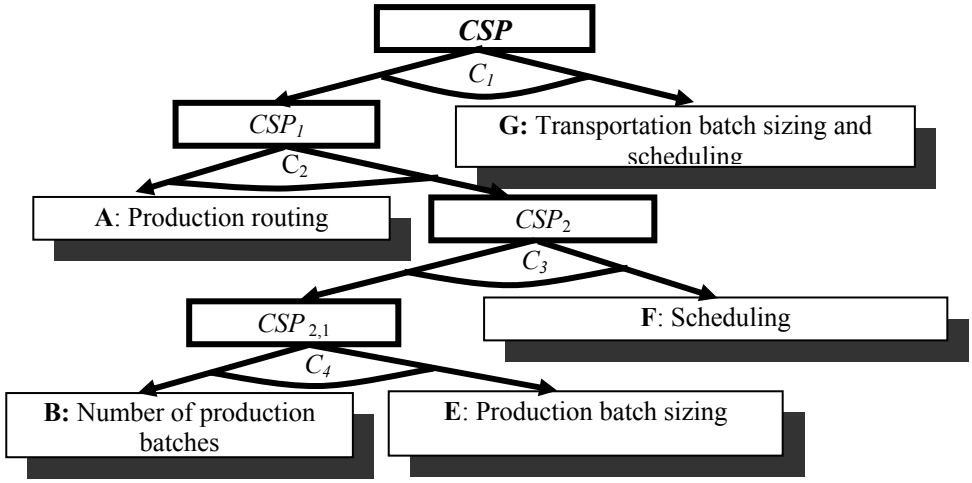



Fig.13. Decomposition of the production flow planning problem

There are growing needs for decision support tools capable to assist a decision maker in the course of a new production order evaluation. The possible approach based on CP/CLP paradigm assumes possibility to adjust the ways of problem specification, its programming language implementation as well as a searching strategy selection to a class of production planning problems (specified by scale, production type, e.g. assembly or machining, and so on) as to respond in on line mode.

Programming methodology proposed takes into account the constraints imposed by a programmer experience (possible problem statements), by a set of available software tools (CP/CLP languages), and a set of searching strategies (build-in the software tools as well as those proposed by programmers) (see Fig.14). The idea standing behind of this approach assumes that a decision maker has to be supported in the course of a standard requests and regarding known in advance a class of situations that may occur in the SME at hand.

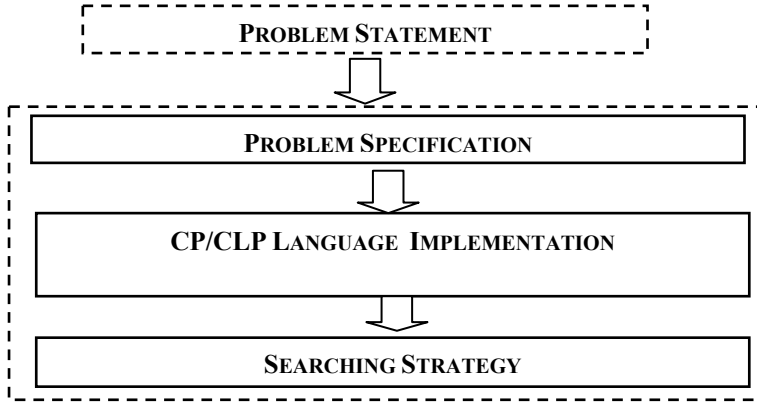



Fig.14. Stages of the CP-based programming methodology

Following these requirements a programmer has to develop a well adjusted CP/CLP based decision support tool encompassing a specific of an enterprise and production orders considered.

The illustration of the implementation of the methodology considered into the task oriented software tools supporting the SME in the course of decision making is shown in Fig. 15. The way of admissible problem resolution is underlined by the bold frames and arcs.

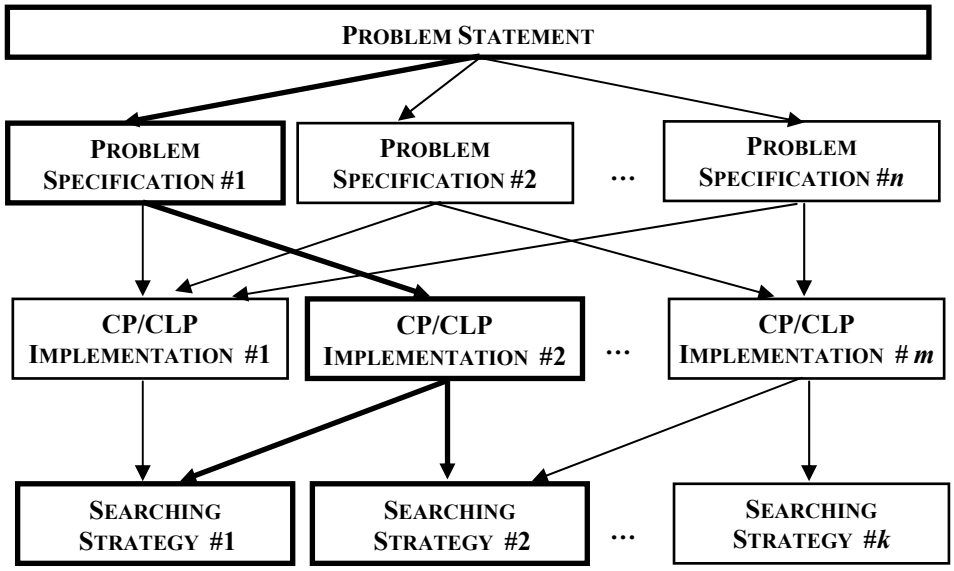



Fig.15. The tree of possible ways of a CSP programming

Since efficiency of programming (i.e. a size of resolved problem) depends on selected problem specification, a software tool employed and a chosen searching strategy, hence designing of the task oriented tools seems to be the only reasonable approach to set-up a CP-based decision support software. It means, for a given class of *CSP*s (e.g. production flow planning ones) and a CP/CLP language assumed, on the base of an experience gathered both from the analysis of the reference model of *CSP* decomposition, and multiple experiments that is possible to develop searching strategy optimal in the sense of minimal number of potential backtrackings.

## 6. CONCLUDING REMARKS

A CP/CLP – based modeling framework provides a good platform for consistency checking between the production order completion requirements and a workshop capability offered. The CP/CLP methodology presented here seem s to be a promising alternative for commercially available tools based on other technologies, such as a class of ERP systems. Their application in solving a real-life problem is quite limited [12], [13].

Also, the proposed approach can be considered as a contribution to project-driven production flow management applied in make-to-order companies as well as for prototyping of

the virtual organization structures. That is especially important in the context of a cheap and user-friendly decision support for the SMEs. Further research is aimed at developing task oriented searching strategies, implementation of which will support the SME's decision making process.

## References


[1] BANASZAK Z., JÓZEFOWSKA J. (red.): *Project-driven manufacturing*, WNT, Warszawa, 2003.

[2] BANASZAK Z., TOMCZUK I.: *Harmonogramowanie przedsięwzięć z wykorzystaniem technik programowania z ograniczeniami*, Red. Knosala R., Komputerowo Zintegrowane Zarządzanie, WNT, Warszawa, 2004, pp. 38-47.

[3] BANASZAK Z., BZDYRA K.: *Programowanie z ograniczeniami w systemach wspomagania decyzji MŚP*. W: Metody sztucznej inteligencji w zarządzaniu i sterowaniu, II tom serii Zarządzanie i technologie informacyjne. J. Józefowska red. Wyd. Uniw. Śląskiego, Gliwice, 2005. (w druku).

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

[5] BARTÁK R.: *Visopt shopfloor: a technology overview*, Proceedings of the Workshop on Constraint Programming for Decision and Control, June, Gliwice, Poland, 2002, pp. 7-15.

[6] BARTÁK R.: On-line guide to constraint programming, Prague, 1998, http://kti.mff.cuni.cz/~bartak/constraints/

[7] BARTÁK R.: Constraint programming: In pursuit of the holy grail. http://kti.mff.cuni.cz/~bartak/constraints/

[8] Ilog Solver, *Object oriented constraint programming*, Ilog S.A., 12, Av. Raspail, BP 7, 94251 Gentilly cedex, France, 1995.

[9] KIS, T.,ËRDOS, G., MÀRKUS, A., VÀNCZA, J.: A Project-Oriented Decision Support Systems for Production Planning in Make-to-Order Manufacturing. ERCIN News, Bo.58, July 2004.

[10] ROSSI F.: *Constraint (Logic) programming: A Survey on Research and Applications*, K.R. Apt et al. (Eds.), New Trends in Constraints, LNAI 1865, Springer-Verlag, Berlin, 2000, pp. 40-74.

[11] TOMCZUK I., BANASZAK Z., JAKUBOWSKI J., BZDYRA K.: *Planning of goods transportation in distribution networks*, Logistics and urban infrastructure, Wrocław, 2004, pp. 150-157.

[12] TOMCZUK I., BANASZAK Z.: *Production flow planning based on CLP approach,* ed. Knosala R., Computer Integrated Management, WNT, Warszawa, 2005, pp. 589-600.

[13] TOMCZUK I., BANASZAK Z.: *Constraint programming approach for production flow planning*, Proceedings of the 6th Workshop on Constraint Programming for Decision and Control, 2004, pp. 47-54.

[13] TOMCZUK I., BOROWIECKI T., BANASZAK Z.: *Project-driven decision support: a CLP approach.*, Automatyzacja-Nowości i Perspektywy, Automation 2004, pp. 333-343.

[14] TOMCZUK I., BZDYRA K.: *Reference model of CSP decomposition*. Materiały konferencyjne, Automatyzacja-Nowości i Perspektywy, Automation 2005, pp. 225-232.

[15] TOMCZUK I., BZDURA K., BANASZAK Z.: *Towards CLP-based task-oriented DSS for SME*. Applied Computer Science and Production Management. Vol.1, No 1, 2005, pp.181-200.
[16] VAN HENTENRYCK P., PERRON L., PUGET J.: *Search and Strategies in OPL*. ACM Transactions on Computational Logic, Vol. 1, No. 2, 2000, pp. 1-36.
[17] VAN HENTENRYCK, P.: *Constraint Logic Programming*, Knowledge Engineering Review, Vol. 6, 1991, pp. 151–194.
[18] WALLACE M.: *Constraint Logic Programming*, Ed. Kakas A.C., Sadri F., Computat. Logic, LNAI 2407, Springer-Verlag, Berlin, Heidelberg 2000, pp. 512-532.