NOTAS DE MATEMATICA

№ 136

BRANCH AND BOUND ALGORITHMS FOR CONTROL OF DISCRETE EVENT PROCESSES-APPLICATION FOR DISCRETE MANUFACTURING PROCESSES.

BY

EWA DUDEK-DYDUCH

PRE-PRINT

UNIVERSIDAD DE LOS ANDES
FACULTA DE CIENCIAS
DEPARTAMENTO DE MATEMATICA
MERIDA-VENEZUELA
1993

BRANCH AND BOUND ALGORITHMS FOR CONTROL OF DISCRETE EVENT PROCESSES -APPLICATION FOR DISCRETE MANUFACTURING PROCESSES

Ewa Dudek-Dyduch
Institute of Automatics,
University of Mining and Metallurgy
al. Mickiewicza 30, 30-059 Kraków, Poland
tel\fax: + 48 12 34 15 68

KEY WORDS: control of discrete event processes, branch & bound method, discrete manufacturing processes.

ABSTRACT

The paper deals with branch & bound (B&B) algorithms appropriated for control optimization of discrete event processes of some class. The definition of multistage decision process is given as a special form of knowledge-based model. It is a basis for general classification of B&B algorithms and for more detailed classification of lower bounds. The domination rules for lower bounds are defined. An example is given.

1. INTRODUCTION

The paper deals with branch and bound method (B&B) applied to control optimization of some class of discrete processes, namely so-called discrete determinable event processes (DDEP). The DDEP are formally defined in the previous paper of the author (Dudek 94) thus the paper is continuation of the other paper.

As a discrete determinable event process we mean all possible sequences of events that occurrences are not spontaneous, but depend on sequence of decisions i.e. depend on control. Both events and decisions can have different interpretation. The examples of control of DDEP include control of discrete manufacturing processes (with no disturbances), determining the optimal tour of salesman, scheduling programs processed by computers and many other combinatorial optimization problems.

There are three kinds of papers connected with the branch and bound method (B&B). The very general ones (Kanal 1988) (artificial intelligence level),

the papers considering the method applied to some class of problems (e.g. defined by means of the discrete programming) and the papers presenting algorithms for fixed, single problems. The papers dealing with B&B applied to optimization of control of discrete events processes, particularly discrete manufacturing processes, are usually of the third kind (Ignal 1965; Held 1971; Schrange 1972; Grabowski 1977, 1978, 1984). It is caused by the fact that there was no general, universal model enabling presenting all these problems.

The particular papers used different models and often used a verbal description only. Different algorithms were frequently presented for the same problems but the lack of uniform formalization makes determination of formal differences between their conceptions impossible.

In contrary to the previous papers this one belongs to the second group and is aimed at generalization and formal discussion of the basic construction of the given there algorithms. A new classification of fundamental conception of B&B algorithms is presented, different types of lower bounds are formally defined and domination rules for lower bounds are defined. The paper is based on the general mathematical model that has been introduced by the author (Dudek 1988,1989) and which modified version is presented in the paper "Discrete determinable processes - Compact knowledge-based model"

2. MULTISTAGE PROCESS AS A SPECIALIZED FORM OF KNOWLEDGE BASED MODEL

The notion of a multistage decision process is widely used in optimization, but its idea has been introduced by means of examples rather than by a formal description. Let us recall the definition of multistage process given in the previous author paper.

Definition 1. Multistage decision (constructing) process is a process P defined as a six-tuple (U,Y,y_o,f,Y_N,Y_G) where the individual elements are defined as follows: U is a decision set, Y is a set of states, y_o, Y_N, Y_G are distinguished initial state and sets of not admissible and goal states, respectively, f is a partial function $f : U \times Y \to Y$ defined by means of the function $g : Y \to 2^U$ in the following way:

$$(u,y) \in \mathrm{Dom}\ f \Leftrightarrow u \in g(y).$$

Function f was defined as a partial function. This is so in order to deal with all the limitations concerning the decisions in the current state in a convenient way. This is done by means of the so-called sets of possible decisions in a state y, denoted as $U_p(y)$ and defined as:

$$U_p(y) = \{u \in U : (u,y) \in \mathrm{Dom}\ f\}.$$

Thus function $g(y)$ determines the decision subset $U_p(y)$, for which function f is determined in the state y.

The sets Y_n, Y_G and U_p are defined by means of logical formulae. Therefore, the complete model constitutes a specialized form of a knowledge-based model.

It is easy to see that the model of DDEP given in the previous paper is a special kind of the multistage decision (construction) process. (The numerical, monotonously increasing coordinate t is assumed for the DDEP model).

Let us recall that the task of optimization of control of DDEP lies in finding of such an admissible decisions sequence $\tilde{u}$ that minimizes a certain criterion Q (thus it is represented by the pair (P,Q)) and a defined problem of control optimization is the set of optimization tasks that have common (parametric) description.

2.1. Types of multistage processes

It should be noticed that there are many problems of which the searched control sequence $\tilde{u}$ can be generated with use of more than one multistage process.

In order to illustrate it, let us present two multistage processes for the salesman (salesperson) problem. It is one of the oldest problems in network optimization. The statement of the traveling salesman problem is simple.

A salesman must visit every city in his territory exactly once and then return to his starting point. Given the cost of travel between the pairs of cities, how should he plan his itinerary so that he visits each city exactly once and so that the total coast of his entire tour is minimum? In network theory terms, the problem is to find a minimum-weight cycle of lenght n in a given weighted graph of n nodes (minimum-weight Hamiltonian cycle).

The problem, presented by means of production terms, is as follows.
The set of jobs $\mathbb{Z} = \{1,2,..n\}$ is to be performed by means of one machine but retooling is required whenever a job is changed.
Relation $R \subset \mathbb{Z}\times\mathbb{Z}$ determines the poosible direct sequence of two jobs. The function τ determines for each pair $(i,j) \subset R$ the sum of retooling time and time of performing the job j. Time of retooling depends on the both jobs. One should find the admissible sequence (permutation) of the jobs so that the total time is minimal. This permutation corresponds to the searched control sequence $\tilde{u}$.

The very same permutation can be specified in different ways e.g. by giving its succeeding elements or by giving the set of pairs of neighbouring elements. Thus we can propose at least two multistage processes for constructing the permutations. In the first process, a value of a decision is the succeeding element of the generated permutation (name of the succeeding job). A value of decision of the second process is a pair $(k,l) \in R$, and each goal state of the process defines any admissible permutation.
Let us give the processes for the selesman problem

Process I (natural process)

Let us assume the additional job, denoted as 0-th job, and enlarge the relation R and the function τ in the following way:

$$\forall j\in\mathbb{Z} \quad (0,j)\in R \ , \quad \forall j\in\mathbb{Z} \quad \tau(0,j) = 0$$

(for the sake of simplicity the performing time of the first job is taken into account at the last step)

The decision consists in determinimg the next job to be performed. Thus, the decision value is the name (number) of the chosen job.
The proper state x is defined by the set of completed jobs, the job performed as the last one, and the job performed as the first one.
Formally:
the decision set $U = \mathbb{Z} \cup \{0\}$
the state set $X = X^o \times X^1 \times X^2$ $\quad x = (x^o, x^1, x^2)$ where
a value of x^1 is the number of the last job, a value of x^o is the set of all completed jobs apart from the last one, and a value of x^2 is the number of the first job,
the initial state: $x_o = (\emptyset, 0, 0)$ $\quad t_o = 0$

the set of possible decisions $U_p(x) = \{j \in \mathbb{Z} \setminus (x^o \cup \{x^1\}) : (x^1, j) \in R\}$,

the algorithm of transition function is as follows:

$$f_x(u_i, x_i) = \begin{cases} x^o_{i+1} = \emptyset \ , \ x^1_{i+1} = u_i \ , \ x^2_{i+1} = u_i & \text{if} \quad i = 0 \\ x^o_{i+1} = x^o_i \cup \{ x^1_i \} \ , \ x^1_{i+1} = u_i \ , \ x^2_{i+1} = x^2_i & \text{if} \quad i \neq 0 \end{cases}$$

$$\Delta t(u_i, x_i) = \tau(x^1_i, u_i)$$

the definition of the set of goal states does not depend of time, thus

$S_G = \{(x,t) : x \in X_F)$ where $X_G = \{(x^o, x^1, x^2) : |x^o| = |\mathbb{Z}|$ and $(x^1, x^2) \in R\}$,

the definition of the not admissible states S_N also does not depend on time thus it is enough to define the set of not admissible proper states

$$X_N = \{x^o, x^1, x^2) : |x^o| = |\mathbb{Z}| \text{ and } (x^1, x^2) \notin R\}$$

Proces II (loose process)

Let us define the elements of another miltistage process $P = (U, Y, y_o, Y_N, Y_G)$. Let us assume that the value of a decision u is a pair of jobs (i,j) such that j-th job is performed just after the i-th one. Thus $U = \mathbb{Z} \times \mathbb{Z}$. A state y is defined by a subset of the fixed (by earlier decisions) pairs of jobs. Thus the initial state $y_o = \emptyset$.
The next state is defined $y_{i+1} = y_i \cup \{u_i\}$.
A goal state corresponds to the set of n-1 pairs such that define one cycle.
A not admissible state corresponds to a set of pairs that constitue any subcycles, or the pairs such that the first elements or the last elements of some pairs are the very same.

Now, Let us distinguish two basic ways of constructing the searched control sequence. The first one lies in recurrence generation of control sequence according to the model of DDEP, (see the first process) the second one - in the construction of the sequence by the determination of its elements in

any order different than the order of occurrence in the sequence $\tilde{u}$ (see the second process). The multistage process of control sequence generation (following the DDEP model) will be called a natural process, whereas other multistage construction processes will be called multistage loose processes.

2.2. Properties of multistage processes

Now we define some properties that are vital for synthesis of B&B algorithms. The properties refer both to loose and natural process. Let us denote: $\tilde{y}$ - a trajectory of considered multistage process, n - a number of the last state of a trajectory, $\tilde{U}$ - set of all decision sequences (admissible or not admissible ones) of process P.

Definition 2. The criterion Q is separable for process P if for each control sequence $\tilde{u} \in \tilde{U}$ it can be computed in the following way:

Q_o = constant, in particular $Q_o = 0$

$Q_{i+1} = f_Q(Q_i, u_i, y_i)$ for i=0,1,..n-1, and $Q_n = Q$, where Q_i, for $i > 0$ denotes the partial criterion value computed for i-th state of considered trajectory and defined as follows:

$Q_i = Q(\tilde{u}')$ where $u = (u_0, \ldots u_{i-1})$ is the first part of sequence $\tilde{u}$, f_Q is some partial function $f_Q : \mathbb{R} \times U \times Y \to\to \mathbb{R}$ determined for the same pairs (u,y) as the transition function, that is:

Dom $f_Q = \{(a,u,y) \in \mathbb{R} \times U \times Y : y \in Y^P , \ u \in U_p(y) , \ a \in \mathbb{R}\}$, where Y^P is a set of states of all trajectories of process P.

Let ΔQ_i denotes the increase of criterion value in i-th state of a fixed trajectory of process P : $\Delta Q_i = f_Q(Q_i, u_i, y_i) - Q_i$

Definition 3. The separable criterion Q is additive if ΔQ_i depends only on u_i and y_i (does not depend on Q_i) for each i=0,1, i.e.. $Q_{i+1} = Q_i + \Delta Q_i(u_i, y_i)$.

Definition 4. The separable criterion Q is multiplicative if it can be recurrently calculated for consecutive states of trajectory $\tilde{y}$ by means of formula $Q_i = \delta Q_1 \cdot \delta Q_2 \cdot \ldots \cdot \delta Q_i$ where $\delta Q_i = \delta Q(u_{i-1}, y_{i-1})$ is given function for $i = 1,2,.n$ and $Q_o = 1$.

Definition 5. The separable criterion Q is monotonically increasing along each trajectory of process P if $Q_{i+1} \geq Q_i$ for each control sequence $\tilde{u} \in \tilde{U}$.
In case of strong inequality, we say about exact monotonicity.

It results from the above that an additive criterion is exact monotonically increasing iff $\Delta Q > 0$ and a multiplicative criterion is exact monotonically increasing iff $\delta Q > 1$ (if $\Delta Q<0$ and $0<\delta Q<1$ then the criterion is monotonically decreasing).

Each of the definitions refers to whole problem $(\mathbb{P}, \mathbb{Q})$ if the property occurs for all the tasks $(P,Q) \in (\mathbb{P},\mathbb{Q})$.

Definition 6. Process $\mathbb{P}$ is static with respect to optimal solution if for each individual process $P \in \mathbb{P}$ the following condition occurs: if $\tilde{u}$ is an admissible decision sequence then the sequence $\tilde{u}'$, obtained as a result of transposition of any two decisions in the sequence $\tilde{u}$, is also the admissible sequence for the process P, and the criterion value for $\tilde{u}$ is equal to the criterion value for $\tilde{u}'$.

The loose process for salesman problem, which is presented former, is an example of a static process.

3. CLASSIFICATION OF ALGORITHMS

Generally speaking, B&B method lies in the construction of a decision tree, the nodes of which correspond to the sets of objects of the same type as the searched solution of the problem. By way of eliminating the nodes, the successors of which do not contain admissible or better than the best solution, the number of calculations is reduced.

The necessity of introducing algorithms classification within the branch and bound method has been observed by several authors (Mitten 1970; Kanal 1988). Usually the classification is based on the following elements: selection rule applied to the choice of a next node for branching, branching rule, lower bound, upper bound, elimination rules of nodes and computational parameters determining maximal error, maximal computing time, maximal size of the set of active nodes etc.

According to the author, to compare and analyse the idea of algorithms, the definition of the set of objects assigned to the nodes (and the connected with it branching rules) is most important. Distinguishing among these sets makes the algorithms classification more legible.

Let us introduce the general classification of B&B algorithms optimizing control of DDEP. The classification is presented in Fig.1. It utilizes the former division of multistage processes.

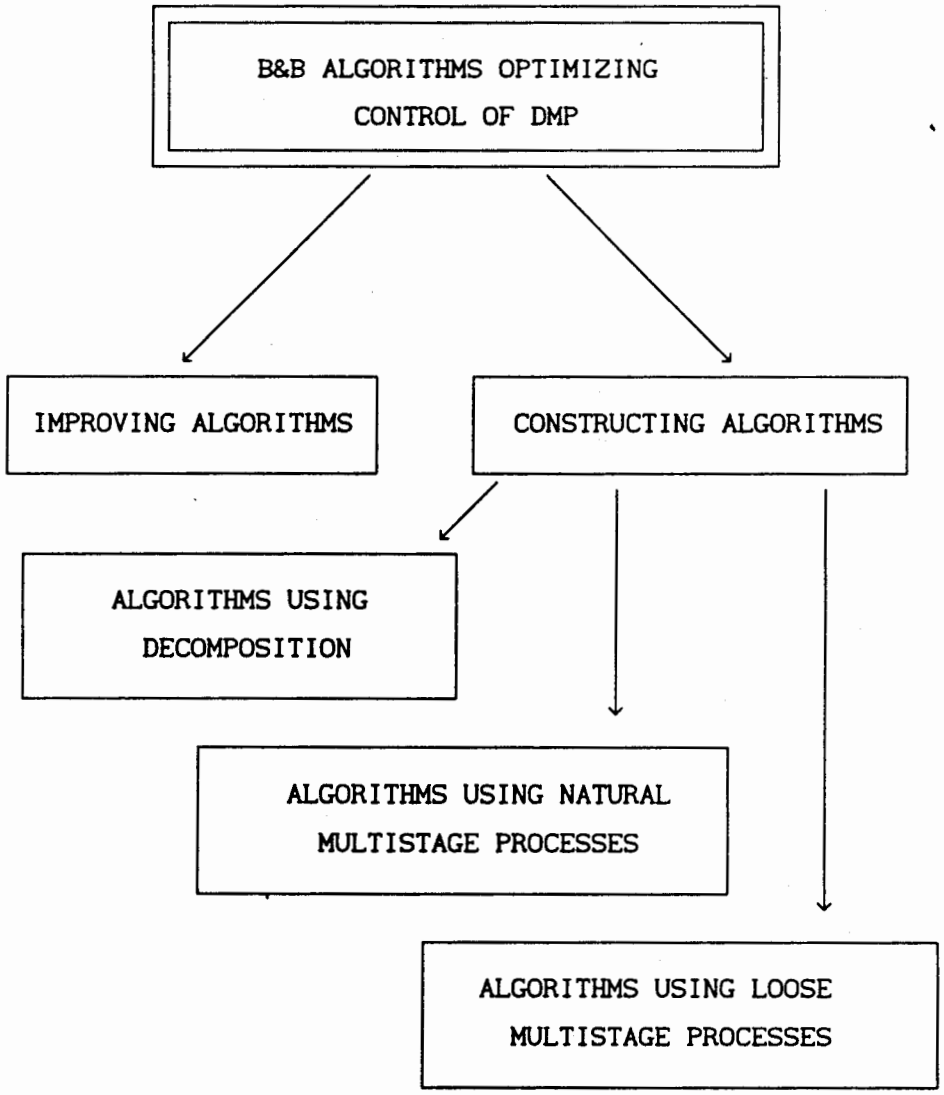

Fig. 1 Division of B&B algorithms appriopriated for optimization of DMP control.

The improving algorithms start from a certain admissible solution (or a certain object of the same type as the admissible solution), the elements of which are changed in the succeeding iterations. Algorithms adding some solution elements, without initial settings, are called constructing algorithms.

Let us denote $A(\omega)$ - a set of objects (solutions) connected with the node ω of a decision tree of a B&B algorithm. It should be noted that there is often more than one way of defining the set of objects $A(\omega)$. On one hand, this definition may be connected with the way in which we try to reach the searched object, and so depends on whether the algorithm belongs to the group improving or constructing the solution; in the latter case also on the definition of the used type of multistage process and its properties. On the other hand, however, thanks to the common DDEP model we may uniformly define the objects connected with each node of the tree, regardless of the algorithm. Since the searched object is a control sequence of some DDEP, a subset of control sequences of the process is connected with each node. Therefore, each definition of set $A(\omega)$ may be referred to the universal definition expressed by means of decisions sequences subset. Thanks to this it is possible to carry out purely formal comparison of various algorithms designed for the same problem.

Let us consider constructing algorithms using a multistage process. Some initial part of decisions sequence of the constructing process is connected with each node of decision tree. This sequence, now called a **characteristic sequence of the node**, determines the set $A(\omega)$. In the case of natural process the set $A(\omega)$ contains certain control sequences with the first part the same as the characteristic sequence. In the case of loose process the set $A(\omega)$ contains control sequences which **may be constructed** by the enlongation of the characteristic sequence.

Let us consider the branching rules. There are two basic situations. Let us discuss them for the natural process.

1. The set $A(\omega)$ contains all decision sequences which the initial part overlapes the characteristic sequence for a given node. Obviously, there is only one branching rule preserving the definition of the set $A(\omega)$ of this type. According to the rule, a set of sequences $\tilde{u} \in \tilde{U}$ that have common initial part longer by one element is assigned to a successor of the node

ω (Ignal 1965). The additional elements (decisions) are different for the different successors.

2. The set $A(\omega)$ does not contain all above defined sequences. It is additionally defined by giving the subset of decisions which cannot appear in the further part of the construction. The branching rule distinguishes two successors: first - in which characteristic decision sequence is broadened by a new decision u' , and the second one - containing sequences, the initial part of which overlaps the characteristic sequence, but in which the decision u' will not appear. Such a branching rule can be applied only for the process that is static with respect to optimal solution.

The above rules may be expressed analogously for the loose processes. The algorithm for salesman problem presented in (Sysło 1983) is an illustration of application of the second rule to the loose processes. Admissible solution, if exists, are ascribed to the nodes that are leaves of the decision tree.

Improving algorithms (Grabowski 1977, 1978, 1984) lie in the modification of an admissible solution. The set $A(\omega)$ is determined by:
- some admissible solution that is characteristic of the node,
- rules determining possible modifications of this solution, in most cases defined in the same way for all the nodes,
- information limiting the modifications and preventing new generations of the same nodes.

4. LOWER BOUNDS

4.1. Classification of Lower Bounds

The concept of creating the lower bounds significantly depends on the fact to which of the above mentioned types belongs the analysed algorithm. Taking into account the way of creating, the following groups of lower bounds can be distinguished:

1) bounds obtained as a partial value of monotonously increasing criterion, calculated for the some part of the control or decision sequence of a suitable multistage process (in particular, for the initial part of the

sequence),

2) bounds obtained as a result of solution of another problem or task,

3) bounds obtained from the combination of two former types.

Now, let us give some theorems referring to the first group. The similar statements are presented in (Pearl 1984) but for the state graph model only.

Let $L(\omega)$ denotes the lower bound for the node ω of the decision tree.

Theorem 1 For each problem $(\mathbb{P},\mathbb{Q})$, where $\mathbb{P}$ is a multistage process and $\mathbb{Q}$ - minimized criterion exactly monotonously increasing along the trajectory of the process, there may be applied a B&B construction algorithm with lower bound equal to the partial value of criterion $\mathbb{Q}$ (in case of maximization the criterion must decrease).

Proof. The theorem refers both to an additive and to a multiplicative criterion. $L(\omega) = Q_i < Q_i+\Delta Q_{i+1}+\Delta Q_{i+2}+\ldots\Delta Q_{n-1}$ in the first case and $L(\omega) = Q_i < Q_i\cdot\delta Q_{i+1}\cdot\delta Q_{i+2}\cdot\ldots\cdot\delta Q_n$ in the second case, because of exact monotonicity of criterion ($\Delta Q > 0$ and $\delta Q > 1$).

Although in the case of non-exact monotonicity the theorem holds true, yet the efficiency of recurrent algorithm lowers because for each succesor ω' of the node ω in the decision tree the relation $L(\omega') \geq L(\omega)$ holds and the elimination activity of the lower bound is much weaker. For this reason the construction algorithm with lower bound of this type should not be applied. The algorithm defined in theorem 1 is the simplest one in the analysed class. It uses minimum information. Obviously, using more information we obtain more efficient algorithms.

Theorem 2. For each problem $(\mathbb{P},\mathbb{Q})$, where $\mathbb{P}$ is a multistage process and $\mathbb{Q}$ - minimized criterion additively separable for the process, there may be applied B&B algorithm with lower bound equal to the sum of partial value of criterion $\mathbb{Q}_i$ for the state y_i and lower estimation of criterion for the final sections of trajectories starting in the state y_i .

Proof, as obvious, will be ignored.

Theorem 3. Let $(\mathbb{P},\mathbb{Q})$ be a problem where $\mathbb{Q}$ is the minimized criterion multiplicatively separable for the process $\mathbb{P}$ and $\delta Q_i > 0$ for each pair (u,y) such that the transition function is determined for it. The construc-

tion B&B algorithm exists for the problem, where the lower bound is equal to product of the partial value of criterion Q_i for the state y_i and lower estimation of criterion for the final section of trajectories starting in the state y_i.

Proof. Let $R(y_i)$ denotes the lower estimation. The theorem follows from the fact that the value of partial criterion Q_i is positive for each state. Thus:

$L(\omega) = Q_i \cdot R(y_i) \leq Q_i \cdot \delta Q_{i+1} \cdot \ldots \cdot \delta Q_n$ because $R(y_i) \leq \delta Q_{i+1} \cdot \delta Q_{i+2} \cdot \ldots \cdot \delta Q_n$.

Above theorems were formulated for an arbitrary multistage process. It must be stressed, however, that even if for the construction of admissible control sequences of some DDEP we may use a loose process, the criterion may be not separable for this loose process. Therefore such a multistage loose process can be used only as auxiliary one.

4.2 Second Group of Lower Bounds

Let us consider the second group of lower bounds. In this case, one must pay attention that the lower bounds for a task differ from the lower bounds for a whole problem. Fig. 2 presents the types of lower bounds obtained as a result of change of model but only for a task.

Let us introduce some formal distinctions of relaxation types. We say that the task A' is relaxed task in relation to the task A if the set of admissible solutions of the task A' contains the set of admissible solutions of task A. As we consider only the problems in which the searched object is a decision sequence of a multistage (natural or loose) process, so in the discussion over relaxation of the task we will employ the following definition of process relaxation.

Let $P = (U, Y, y_o, f, Y_N, Y_G)$ be an individual multistage process and $\tilde{U}_d, \tilde{Y}_d$ be the set of admissible control sequences and the set of admissible trajectories respectively. Let us recall that the transition function f is defined by means of function $g : Y \to 2^U$, defining a subset of possible decisions in the state y.

Definition 7. Relaxation of an individual process P is a modification of the process lying in such a change of definition of any of U, Y_N, Y_G sets or

function g, that the set of admissible control sequences $\tilde{U}'_d$ of the new process P' is not less than the original set $\tilde{U}_d$ and, moreover, $Y'_N \subseteq Y_N$, $Y_G \subseteq Y_G$, $U_p(y) \subseteq U'_p(y)$ for each $y \in Y$ (prim values denote respective modified elements), whereas the remaining elements defining the process stay unchanged.

As the individual process P uniquely defines the task of searching for the admissible solution, therefore defining the types of process relaxation we define the types of task relaxation.

a. Relaxation consisting in the reduction of a set of not admissible states. Let the set of not admissible states be defined as an alternative of predicates $Y_N = \{y \in Y : \psi_1(y) \cup \psi_2(y) \cup \ldots \psi_N(y)\}$. This set may be decreased by omitting at least one predicate in the definition or by a suitable change of its data.

b. Relaxation consisting in in enlargement the set of final states. Let the set of final states is defined by means of predicates conjunction $Y_G = \{y \in Y : \phi(y) \cap \phi(y) \cap \ldots \phi(y)\}$. This set may be enlarged by omitting at least one of the predicates in the definition or by a suitable change of its data.

c. Relaxation lying in the modification of definition of sets U_p. Let the sets U_p are defined by means of predicates conjunction $U_p(y) = \{u \in U : \mu_1(u,y) \cap \mu_2(u,y) \cap \ldots \mu_p(u,y)\}$. This sets may be enlarged by omitting at least one of the predicates in the definition or by a suitable change of its data.

Relaxation can be also made by means of adding some predicates but it cannot be done automatically.
In particular, when the data are changed, some predicate may happen never to fulfill for the process. It may be removed as a redundant then. In this case the change of data causes the change of form (scheme) of definition of a given set. This has been represented on Fig. 2 by the broken line (1).

Let us notice that, due to the modification of the definitions of sets Y_N, Y_G and U_p, some state coordinates appearing in the deleted predicates may turn out to be redundant in the model of relaxed process. For this reason, the definition of the set of states, and consequently the definition of the transition function may be also changed. This has been represented in Fig. 2 by the broken line (3).

If the control sequence my be constructed by means of various multistage processes, then the type of relaxation depends on the used process. Therefore we may distinguish:

- task relaxation resulting from the relaxation of multistage natural process,
- task relaxation resulting from the relaxation of multistage loose process; here the type of relaxation additionally depends on the chosen loose process.

Now let us define the problem relaxation.

Let us recall that the process $\mathbb{P}$ is defined with use of paremeters. When the parameters are replaced by data, then we obtain an individual process P (instance of problem). Let P(D) denotes the individual process $P \in \mathbb{P}$ obtained for the established data D (data assignment).

Definition 8. Process $\mathbb{P}'$ is relaxed in relation to process $\mathbb{P}$ if for each data assignment D , the individual processes $P'(D) \in \mathbb{P}'$ and $P(D) \in \mathbb{P}$ are the pair of processes such that P' is relaxed in relation to P.

Definition 9. Problem $\mathbb{A}'$ is relaxed in relation to problem $\mathbb{A}$ if there exists a pair of processes $\mathbb{P}'$ and $\mathbb{P}$ defining the sets of admissible solutions of problems $\mathbb{A}'$ and $\mathbb{A}$, respectively, and such that process $\mathbb{P}'$ is relaxed in relation to $\mathbb{P}$.

To sum up, we may distinguish the types of relaxation taking into account:

a) a kind of the used multistage process,

b) kind of limitation modified in the process.

4.3 Lower Bounds Domination

A good lower bound should possibly best approximate the criterion value and not require much calculation. Different lower bounds can be proposed for a given problem. Their domination is usually presented by, so-called, graph of domination relation, i.e. such a graph that particular lower bounds are ascribed to its nodes. The lower bounds are coded by means of graphical symbols. The symbols are set specially for the considered problem (Potts 1980; Grabowski 1984). Domination graphs are constructed separately for the particular problems.

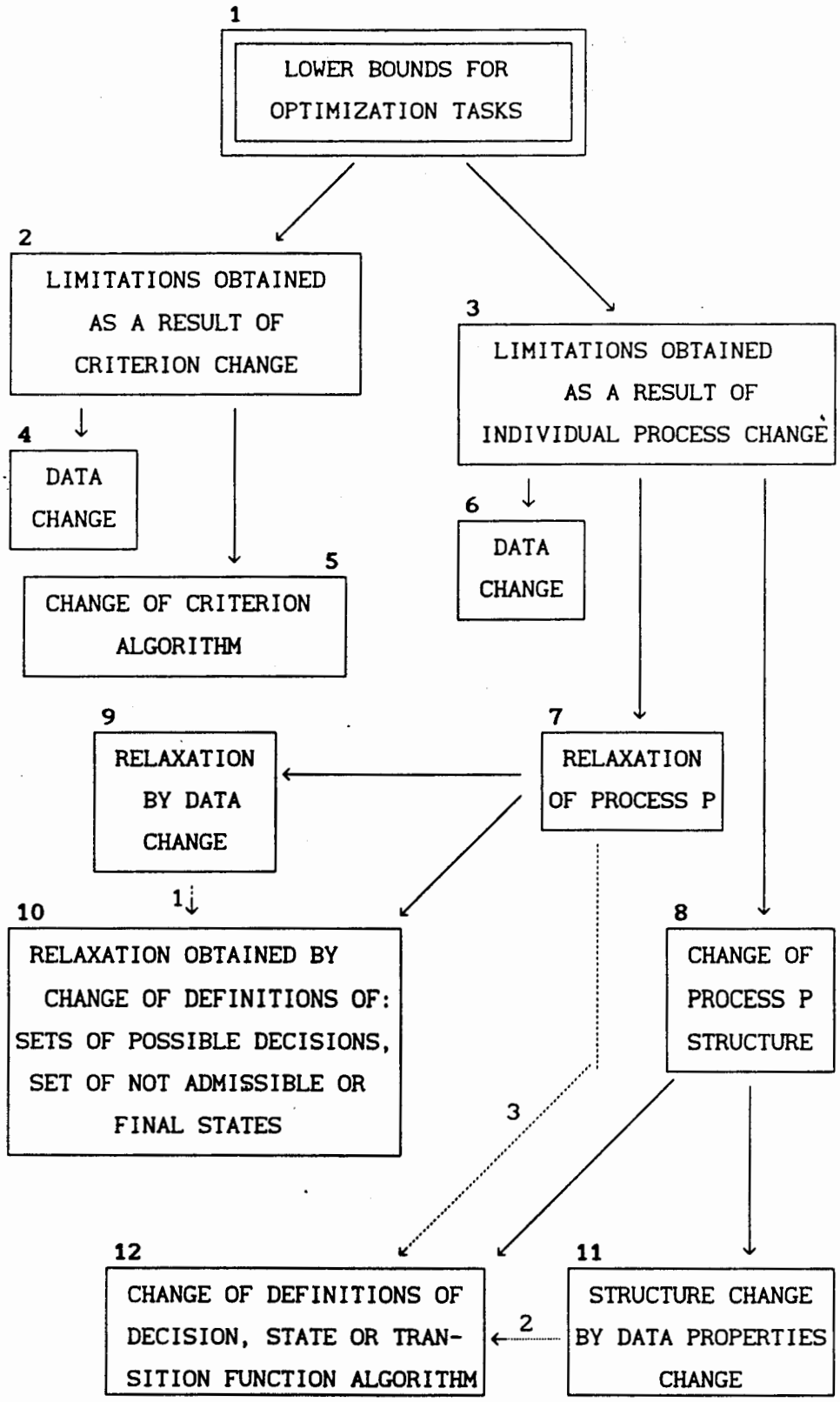

Fig.2 Types for lower bounds for a task, obtained as a result of its model change. The broken lines indicate additional consequences of some changes.

Thanks to the introduced classification of lower bounds, we may formulate general rules satisfied by the domination relations regardless of the analysed problem.

Let us consider a task (P,Q). Let $L(\omega)$ denotes the lower bound for the node ω of the decision tree of B&B algorithm. We say that the lower bound L' dominates lower bound L if for each node ω of the decision tree the following holds: $L'(\omega) \geq L(\omega)$.

We can distinguish the following domination rules. When discussing the change of process, criterion and data we will mean change in the desired direction.

a) The bound obtained as a result of a change of the criterion and a change of the process is dominated by the bound obtained as a result of a change (the same) of criterion only or a change (the same) of the process only.

b) The bound L' obtained as a result of a change of the process structure and a relaxation of process (meaning def. 7) is dominated by the bound L obtained from a change (the same) of the process structure only or as a result of only a relaxation of process.

c) If the bound L is obtained as a result of omission of some predicates in the definition of sets Y_N, Y_G, U_p and the bound L' is obtained as a result of omission besides these predicates still further predicates, then bound L dominates bound L' (speaking about definitions of the mentioned sets we mean the previously given alternative form in case of set Y_N and conjunction form in the case of sets Y_G and U_p).

d) The lower bound obtained as a result of relaxation of the individual process P by missing a predicate in the definition of sets Y_N, Y_G or U_p is dominated by the bound obtained as a result of change of data only, obviously the data appearing in this predicate.

5. EXAMPLE

To illustrate the introduced division of lower bounds, let us refer it to the exemplary limitations for the flow-shop problem with the general minimax objective function, given in Grabowski (1984). The flow-shop problem in

the form $n/m/P,\succ/f_{max}$ may be formulated in the following way. There is given n-elements set of jobs which are to be performed by means of m-element sequence of machines M. Each jobs consists of m-element sequence of operations which are to be performed on the subsequent machines. Times of performing subsequent operations of each job are known. Relation $\succ$ of partial order in the set of jobs is known. For each job, e.g. i-th one, there is determined a non-decreasing function $f_i : C_{im} \to \mathbb{R}$ representing the cost connected with completion the job in time C_{im}. The sequence of the jobs fulfilling the requirements of partial order should be find, such that $f_{max} = \max f_i(C_{im})$ reaches minimum.

Grabowski (1984) has given an algorithm which may be treated as an improving algorithm. Basing on the verbal description of the problem, there were determined the following lower bounds defined by:

a) relaxation of operational capacities of machines,
b) relaxation of time of operation,
c) relaxation of the function of job costs,
d) relaxation of sequential bounds (of partial order),
e) relaxation of the number of jobs,

Now let us present their position in the scheme of lower bounds given in Fig. 2.

ad.a)
Relaxation of operational capacities of machines consists in the admittance that for certain selected machines (with limited flow capacity equal to 1), the number of operations performed simultaneously may be arbitrary. It causes a change of the algorithm of transition function, therefore corresponds to block 12.

ad.b)
Relaxation of time of operation consists in the assumption of an uniform (minimum) time of duration of all operations performed on certain machines. It signifies the introduction of an additional property fulfilled by data, and it causes a change of the algorithm of the transition function. It corresponds to block 11.

ad.c)
Relaxation of the function of job cost lying in the assumption that the function of cost of all jobs are the same and it is equal to

$$f^{*}(t) = \min_{1 \le i \le n} f_i(t) \qquad 0 \le t < \alpha$$

corresponds to block 2.

ad.d)

Relaxation of partial order i.e. assumption that $\succ = \emptyset$ corresponds to the relaxation of process. It causes a change of definition of the sets U_p, and it is represented by block 10.

ad.e)

Relaxation of the number of jobs lying in determination of lower bound for a certain subset of jobs $\mathbb{Z}' \subset \mathbb{Z}$ corresponds to the process relaxation lying in the change of definition of the final states set, and is represented by block 10.

6. CONCLUSIONS

In the paper the formal discussion of B&B algorithms optimizing the control of discrete determinable processes are discussed. Two main definitions constitute the formal basis for presented consideration: the definition of discrete determinable event processes given in the previous lecture and the definition of multistage decision process. Both the notions are defined in terms of knowledge-based models.

Then a classification of lower bounds is given. With regard to the logical approach, the classification can be carried out on the more detailed level and it is utilized for defining the rules of the lower bounds domination. It is very essential for computer decision support systems, especially with respect to the possibility of automatic generation of lower bounds.

REFERENCES

1. Dudek-Dyduch, E. (1988). Scheduling some class of discrete processes. *Proc.of 12oh IMACS World Congress*, Paris.

2. Dudek-Dyduch, E. (1992). Control of discrete event processes - branch and bound method. Prepr. of IFAC/IFORS/IMACS Symposium Large scale systems: Theory and applications, Chinese Association of Automation, vol 2, 573-578.

3. Dudek-Dyduch E. (1994). Discrete determinable processes - compact knowledge-based model. Proposed to this conference.

4. Grabowski, J. (1977). A new formulation and solution of the sequencing problem: algorithm. *Zastosowania Matematyki* Nr 15, 463-474.

5. Grabowski, J. (1978). Formulation and solution of the sequencing problem with parallel machines. *Zastosowania Matematyki* Nr 16.2.

6. Grabowski, J. and C. Smutnicki. (1984). Minimizing maximum cost in flow shop problem, part I,II. *Archiwum Automatyki i Telemechaniki*, t.XXIX, z.1-2, 35-74.

7. Held, M. and R.M.Karp. (1971). The travelling salesman problem and minimum spanning trees. *Mathematical Programming* vol.1, pp.6-25.

8. Ignall, E. and L.Schrange (1965). Application of the branch and bound technique to some flow-shop scheduling problems. *Operations Res.* vol.13, No 3, 400-412.

9. Kanal, L. and V.Kumar (eds) (1988). *Search in artificial intelligence.* Springer Verlag New York Inc.

9. Mitten, L.G. (1970). Branch & bound methods. General formulation and properties. *Oper. Res.* vol.18, No.1, 24-34.

10. Schrange, L. (1972). Solving resource-constrained network problems by implicit enumeration-preemptive case. *Operations Res.* vol.20, No.3, 668-677.

11. Sysło, M.M., N.Deo and J.S.Kowalik. (1983). *Discrete Optimization Algorithms*. Prentice-Hall, Englewood C