IAS-8 Proceedings of the 8th International Conference on Intelligent Autonomous Systems
10-13 March 2004, Amsterdam, The Netherlands

Multi-agent repairer of damaged process plans in manufacturing environment

S. Kornienko, O. Kornienko, P. Levi
Institute of Parallel and Distributed Systems,
University of Stuttgart, Universitätsstr. 38, D-70569 Stuttgart, Germany

Abstract. Modern manufacturing is highly dynamical and complex environment. Regular processes in this environment can often be perturbed by different disturbances (failures). Unpredicted nature of some disturbances represents a hard problem for a planning system. To react reasonably to these disturbances, the planning system should possess a huge state space. The reactions should be either included into this state space or generated dynamically. Application of multi-agent technology for such a generator, performing a dynamic replanning (or repairing a damaged plan), is proposed. The main focus of the consideration lies on a distributed algorithms of symbolic tasks decomposition.

1 Introduction

Modern manufacturing operates in quickly changing environment. Reasons for these changes (disturbances) are different: competition in global markets, change of consumer properties, technological innovation, failures and so on . To survive, enterprises are forced, among other arrangements, to react dynamically to disturbences, to have flexible (transformable) structure on all levels of organization [1]. This work focuses on one aspect of this transformability, namely on an adaptable planning of the lowest level, denoted as process planning. This planning is the most sensitive element in the production chain.

Disturbances can be of different types. The *”predicted”* disturbances arise when some parameters are changed, but the planning system contains a mechanism of how to perform a replanning. This replanning often involves into a plan a huge number of states. Typical example is a failure of processing machines. In principle, this failure can be absorbed by rescheduling other machines. However if a manufacturing chain contains reconfigurable machines, the number of possible functional alternatives grows exponentially with a number of such machines. *”Unpredicted”* disturbances arise when there is no replanning mechanism, absorbing them. To repair a damaged plan in this case, a replanning involves new states into a plan.

The mechanism, repairing a damaged plan in both cases, is a part of MaPP (Multi-agent Process Planning) system representing a rapid prototyping system for flexible manufacturing control in turbulent environment [2], [3]. The idea is to apply AI and DAI approaches (DCSP/DCOP, reasoning and so on, see [4], [5]) to manufacturing PPC/APC systems (e.g. [6]). In this way, the replanning remains in the short-term planning horizon (that reduces time and cost) and can react to wide spectra of disturbances.

2 Autonomous systems in manufacturing

Autonomous planning systems, used in modern manufacturing, possess specific properties and differ from other kinds of autonomous systems, e.g. mobile robots, UAV (Unmanned Aerial Vehicle) [7] or stationary autonomous systems [8]. The main difference lies in a complexity of input/output states. Since mobile robots or UAV operate in real (uncertain) environment, the number of input states is huge. They are often of uncertain nature. The output state space is essentially smaller. For example, UAV of WITAS's projects can execute only 6-10 main activities (see description in http://www.ida.liu.se/ext/witas/). In contrast to them, the input state space of manufacturing planning systems has a well defined and limited character (e.g. 150-200 states), but the output space consists often of thousands states. However a real challenge is that some of these states can not be defined in advance, they should be dynamically generated.

To exemplify this affirmation, we briefly consider a process planning for the workpiece shown in fig. 1. Manufacturing of this billet consists of 17 working steps (WS) (see Step NC [9]). Order of these steps is defined by technological network. Each node in this network determines one processing operation (i.e. WS), consisting of geometrical description of required features and technological description of how to manufacture them (fig. 2(a)).

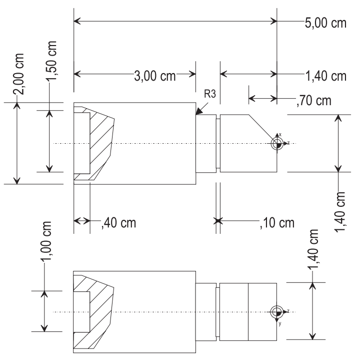
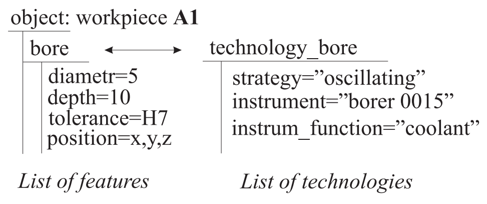
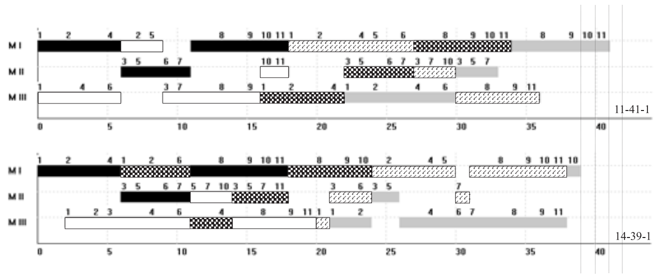


Figure 1: *Example of a workpiece to be manufactured.*

Besides restrictions determined by technological network, there is a lot of organizational, technical and other restrictions. This kind of problem can be formulated and solved as constraint-satisfaction problem (CSP), details of approach and constraints can be found in [2]. The final plan can be thought of as an assignment between low-level jobs and available machines that satisfies all restrictions. As seen from fig. 2(b), this assignment can be of different length and cost (depending on optimization criteria). *In the following consideration we denote this assignment plan as the primary plan, and activities to generate it as the primary activity of the planning system*.



(a)



(b)

Figure 2: **(a)** *Geometrical description of features and corresponding production technology;* **(b)** *Two assignment plans with different length and cost of the same manufacturing process (taken from [2]).*

The produced in small series workpiece, shown in fig. 1, can be changed in customer properties, in a number of pieces, in a time given for production, in priority of different products. Change of customer properties can influence a manufacturing technology, technical basis (replacement or retooling of a machine), can require other resources. Moreover a machine can fail, an instrument can break down, a supply of resources can be interrupted and so on. Several types of these disturbance are collected in table 1(left).

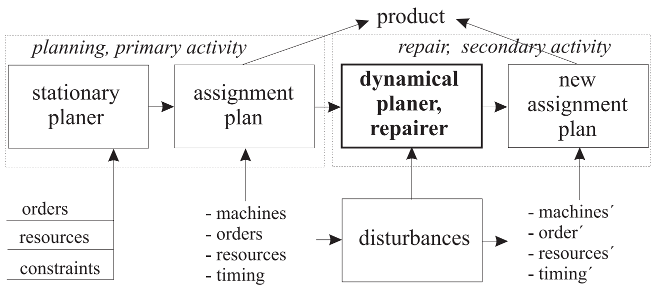
Disturbances	Examples	Horizon
management level	aims, appointm., deadlines	long-term mid.-term
organizational level	orders, lot size, urgent order	short-term
technological level	product-technology process-technology	mid.-term long-term
resources level	machines, supplies	short-term

Type of reaction	N	Examples
time/order oriented	1	rescheduling
	2	reordering
	3	reoptimization
	4	shift of deadline
technical/ technological	5	retooling
	6	oper. replacement
increase of redundancy	7	addit. machines
	8	increase buffer
management	9	extern. manufact.

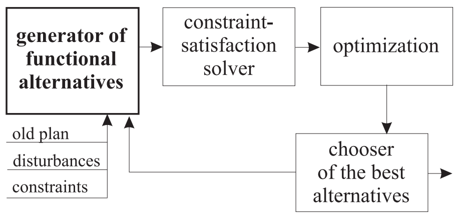
Table 1: **(left)** *Several types of disturbances;* **(right)** *Several types of reaction to disturbances.*

We are mainly interested in sort-term disturbances, that occur during execution of the primary plan. The short-term disturbances deviate the executing conditions, so that the primary plan can not be accomplished and gets damaged by disturbances. Sometimes a damaged plan can anew be generated, but in the most cases it should be repaired. ”Repair” means the system should generate a new plan, that changes the executing conditions of the primary plan so that it can prolong an aborted executing. *This kind of a plan we denote as the secondary plan and activities to generate it as the secondary activity of a planning system*.

There are different ways to react to each disturbance. For example, as a reaction to a machine failure we can: reorder or reschedule machines, retool or reoperate other machines, increase a redundancy, produce a detail externally and so on. Some examples of these reactions are collected in table 1(right). These reactions compose a tree of alternatives. We distinguish between organizational and functional alternatives. *Functional alternatives* mean different sequences of production activities (e.g. retooling or rescheduling), whereas *organizational alternatives* mean macroscopic activities (e.g. to take from a lager, to produce externally and so on). The alternative, utilized on the next planning steps, depends on the used alternatives on the previous steps, moreover this dependence is nonlinear. *There is no way to know beforehand, which route in this three of alternatives is cost- or time-optimal. In several cases, the number of alternatives is so huge that they can not be preprogrammed in advance*. Let us consider this point on the example of planning system, shown in fig. 3(a).



(a)


generator of functional alternatives
constraint-satisfaction solver
optimization
old plan
disturbances
constraints
chooser of the best alternatives

(b)

Figure 3: **(a)** *Architecture of an autonomous planning system;* **(b)** *Architecture of the repair part from fig. 3(a).*

The goal of this system is to generate, to monitor and to repair a short-term process plan. The system consists of two parts: the planning part and the part that is in charge of replanning and repairing the primary plans. Whereas the first part is already considered in [2], [3], here we focus on the second part of this system. In the field of mobile agents, this subsystem is also known as dynamic planner, whose structure is shown in fig. 3(b).

Before start to describe the dynamic planner, let us estimate *the worst case for a number of functional alternatives*. The workpiece shown in fig. 1 has 17 working steps, containing circa 130 geometrical and technological parameters, that can be perturbed. Adding management and other disturbances from table 1(right), we have circa 200 possible disturbances. In the worst case we assume that, firstly, each of these disturbances can be absorbed in 10 different ways, secondly, these functional alternatives are applicable to all next working steps after disturbance (e.g. to avoid an arisen bottle neck/lack of resources and to retain a deadline). In this way we have circa 2000 functional alternatives for one detail and 10000 for a lot of 5 details. For the average and best cases we assume there are some heuristic approaches reducing this number.

Now we consider *the wost case for a number of planning states*. As an example, let us assume a disturbance changes a diameter of the boring hole shown in fig. 1. As a functional alternative the system chooses remanufacturing. However the exact reaction depends on the exact kind of disturbance. If instead of 5 mm we have really only 1 mm, one can bore once again with a drill diameter 5 mm. However if we get a hole 10 mm instead of 5 mm, there is no way to repair it. Another question, what is to do if a boring angle (or drilling temperature or something else) has been changed ? If we take into account all possible ways to disturb 130 technological and geometrical parameters of the workpiece in fig. 1(a), we achieve an enormously big number of possible reactions (output plan state) and it is obviously that *all these reactions can not be preprogrammed in advance. In this case we can assume the disturbance leads to an appearance of new states in the plan*.

3 Multi-agent planning system

Consider the process planning problem more in detail. The plan P is a step-wise mapping S between available machines M and working steps WS in a time window T bounded by constraints C_g

$$S = \{M \times WS \to T, \{C_g\}\}, \tag{1}$$

where $S \in (\mathbb{S}^{P_1} \cup \mathbb{S}^{P_2} \cup \mathbb{S}^{P_3} \cup ... \equiv \mathbb{S})$ and $\mathbb{S}$ is a state space of all plans and $\mathbb{S}^{P_i}$ are corresponding subspaces. Global constraints C_g define manufacturing technology, e.g. the order of operations, organizational and other requirements, applied to whole sequence of WS. The state space $\mathbb{S}$ possesses an ordered structure and can be represented as sequences of steps .., S^j_{i-1}, S^j_i, S^j_{i+1},..., where subindex is the number of step and superindex is the number of plan. This structure is determined by a transition Tr that connects the state S_i with the state S_{i+1}

$$Tr = \{Tr^j : S^j_i \to S^j_{i+1}\}, \tag{2}$$

where also $Tr \in \mathbb{T}r$, $\mathbb{T}r$ is a space of all transitions. As shown later, Tr is also structured into several domains. The common plan can be written as

$$Pl = \{\mathbb{T}r(\mathbb{S}) \to \mathbb{S}\}. \tag{3}$$

All well-known planning approaches (e.g. MDP [10]) require that $\mathbb{S}$ and $\mathbb{T}r$ are predefined. For a small number of states (even hundreds of states) it does not represent a real problem. However, if the number of states becomes huge or there are new states needed to be introduced into a plan, the planning approach fails. *Therefore our idea is, firstly, to introduce a generator of new states Γ and, secondly, to transform a space $\mathbb{T}r$ so that, at least, a part of it can be dynamically generated by demand*.

Let us first consider the generator Γ of new states. Per definition, there are no equal mappings S in $\mathbb{S}$, however the question is whether there states are really unique or they can be decomposed on some atomic constructions ? Looking at (1), we see that the time window T and machines M are of elementary nature, only a working step WS is composed from other terms. As followed from a definition of working step in Sec. 2, it consists of activity A parameterized by technology (modality) D, geometrical descriptions of features F bounded by constraints C_l $\mathbb{W}S = \{A(D) \rightarrow F, \{C_l\}\}$, where $WS \in \mathbb{W}S$. Local constraints C_l define manufacturing technology (see the right side of fig. 2(a)), applied only to one WS. If we consider e.g. $WS5$ "mill" and $WS6$ "fine mill", the difference between them lies primarily in manufacturing technology (the right part of fig. 2(a)) and only then in geometrical descriptions of features (the left part of fig. 2(a)). Manufacturing technology for WS "mill" consists of 9 steps, preparing a machine for performing this operation. In several cases, e.g. a transportation of a workpiece from one machine to another (see e.g. an assignment plan shown in fig. 2(b)) it requires even more steps. In this way, each WS is composed from other activities, being of elementary nature. Specifying a complete set of these atomic activities, we suppose that in fact each arbitrary state of a process plan can be composed from them.

What kind of generator can be applied to this problem ? Analyzing the structure of this problem, we remark one interesting intersection with the problem solving in multi-agent (MA) systems. Speaking in a combinatoric language, the MA system of m agents each with n internal states for time interval t is able to emerge n^{m^t} combinations of internal states. Transforming it to the manufacturing problem, we have 4 retooling activities (for reconfigurable machines): *set mill, set drill, set bore, set grind*; 8 basic activities: *grinding, boring, milling, drilling, shift, transport, load, upload*; 9 preparing activities (example for milling): *choose a type, prepare tools, set rate of feed, set a cutting speed, set a machine function, set depth of cut, processing strategy, set mill overlapping, set oversize* for each basic function. By different combination of these activities is possible e.g. to mill by normal machine function, by retooling or by applying other operations (e.g. drilling with correspondingly prepared machine), i.e. $S = \{M \times (\Gamma \rightarrow F, \{C_l\}) \rightarrow T, \{C_g\}\}$.

For instance, if the maximal length of a repairing plan is equal to 10 steps, this MA generator Γ is able to emerge maximal 21^{10} different combinations of atomic activities (i.e. states S). It is much more than the disturbances, described in Sec. 2, can ever cause. The point is that, firstly, not all of these combinations have a sense for manufacturing problem, secondly, how to force the agents to find to required combinations. The last problem can be solved by specifically chosen negotiations among agents [5], e.g. CSP-based negotiations used in the generation of primary plans (see e.g. [2]).

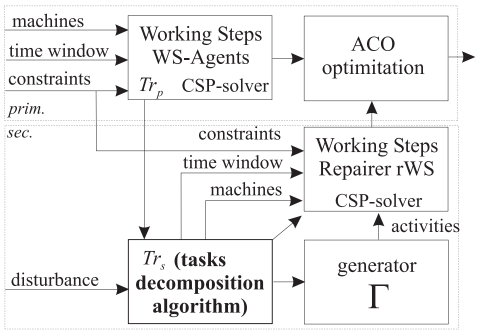
Now we rewrite (2), taking into account the made assumptions about the generator Γ

$$Tr = \{Tr^j : S_i^j \rightarrow (M \times (\Gamma \rightarrow F, \{C_l\}) \rightarrow T, \{C_g\})\}. \tag{4}$$

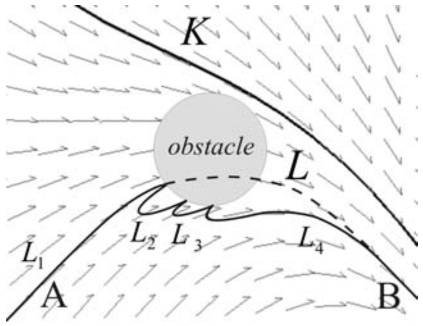
As seen from this expression, the S_i^j is known, but $S_{i+1}^j = \Gamma(S_i)$ is yet unknown. Therefore Tr, beside transition, has in this context a role of a decomposition algorithm. It tells Γ which new state is required on the next step. The problem is that a transition (per definition) determines an order of steps and performs parameterization of WS by geometry. Changing Tr, we change a global technology and as a result a final product. Therefore it makes sense to divide the space $\mathbb{T}r$ into two subspaces of primary $\mathbb{T}r_p$ and of secondary $\mathbb{T}r_s$ activities. The first one defines a manufacturing technology for a product and the second one is directed to repair a primary activity. $\mathbb{T}r_p$ is predefined and may not be changed, whereas $\mathbb{T}r_s$ is in charge of reactions to disturbances, includes functional decomposition and should be so flexible as possible. Rewriting finally (3), we get

$$Pl = \begin{cases} prim. : Tr_p(M \times WS \to T, \{C_g\}), \\ sec. : Tr_s(M \times (\Gamma \to F, \{C_l\}) \to T, \{C_g\}). \end{cases} \tag{5}$$

Transforming this functional expression into graphical form, we obtain the following structure of MA generator of functional alternatives for damaged process plans, shown in fig. 4(a). Basic element of the primary structure in this figure is a WS-planning agent, described e.g.



(a)



(b)

Figure 4: **(a)** *Structure of MA generator;* **(b)** *Motion in a vector field with (L) and without (K) obstacle.*

in [2]. The kernel of the secondary structure is a task decomposition algorithm, described in the next section.

4 Algorithm of symbolic tasks decomposition (ASTD)

The algorithm of task decomposition is perhaps the most challenging problem not only in the distributed planning approaches, but also in domain of distributed problem solving, coordination and so on. There is a lot of known solutions (see e.g. [5]), however these approaches are applicable only to a limited number of specific problems and can not be generalized even to common problem-oriented domain.

Our idea of task decomposition originates from nonlinear dynamics and synergetics (see e.g. [11]). As known, a motion of a dynamical system is defined by a vector field. If this field contains an attracting manifold, the system from arbitrary initial state (in attracting area) will land on this attractor. The attracting dynamics has many applications not only in bifurcation analysis, in systems's control, but also in robot navigation and agent coordination (e.g. [12]). The attracting dynamics has one interesting effect, shown in fig. 4(b). If we perturb nonlinear field (by putting some obstacle on a motion trajectory), a system finds a bypass by all alone. It is especially evident in time-discrete systems, e.g. the way L from the point "A" to "B" will be automatically decomposed on small parts L_1, L_2, L_3, L_4 approximating a bypass.

The question is whether this analytical approach can be applied by analogy to algorithmic problems ?

We start from several basic thoughts, being motivated by the example with a motion in a vector field. Creating a bypass consists of three phases: detection of obstacle and interaction with it, reaction to this interaction, and, finally, a motion in the field so that to achieve the final goal. Consider these phases from the viewpoint of MA system.

I. Detection of disturbance. This is a complex problem that is widely discussed in the corresponding communities. However in the manufacturing environment this problem is simplified, firstly, by certain and limited sensor input, secondly, by a construction of the system, where the agent, that performs activity, performs also a monitoring of this activity. Forasmuch as each disturbance perturbs a formalized primary plan, detection of this deviation does not represent, at least in principle, a problem (see more about this point in the next sections).

II. Reaction on disturbance. We can intuitively say, if the disturbance changes some feature, to change this feature backwards, we need an activity, being able to change this feature. For instance, if the feature ”position” has been modified by a disturbance, we need an agent ”transporter” that can modify a position of objects. Speaking more strongly, we suppose that a reaction to disturbance is determined by some equivalent to vector field, which is a media for propagation of interactions. The most simple way to create this media is to connect source and receiver of activities (agents and objects), so that each perturbation can be propagated further (modifying attributes and activities), till this perturbation will be absorbed. More exactly, the features of objects have to be connected with corresponding activities of agents, e.g. feature ”position” has to be connected with activity ”to move” of an agent ”transporter” (see fig. 5(a)). We denote a network of coupled features-activities as FA-network N_{FA}.

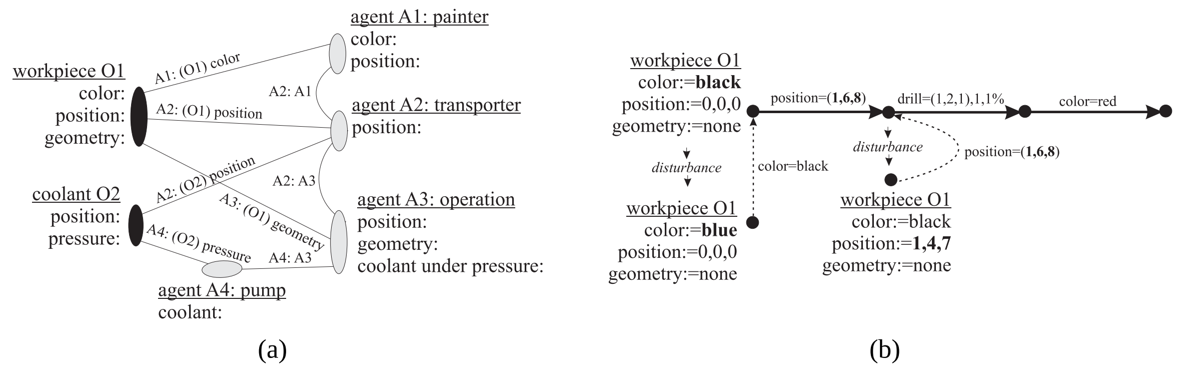


Figure 5: **(a)** *Simple example of FA-network with two objects and four agents;* **(b)** *Executing of primary plan (bold), where the FA-network is applied to each disturbance (dotted) to absorb it.*

III. Achievement of a final goal. This is the most important question that is closely related with reversibility of processes and can be reformulated in the following way: Can each arbitrary disturbance be absorbed ? or Can each arbitrarily damaged plan be repaired ? We intuitively so answer this question: if disturbance breaks one of technological or other restrictions, the primary plan cannot be repaired (e.g. if instead of 5 mm drill hole, we get 10 mm, there is no operation that can drill -5 mm). However this point needs more detailed investigation. By analogy, a motion in the vector field is an equivalent to executing a primary plan, where the FA-network is applied to each disturbance to absorb it, as shown in fig. 5(b). Moreover if all constraints are satisfied the system can accomplish this plan, i.e. achieve a final goal.

Now we formalize these intuitive propositions. Let us introduce a new transition Tr_{dam}, representing a disturbance $Tr_{dam} = \{Tr^j_{dam} : S^j_i \rightarrow S'^j_{i+1}\}$, where the state S'^j_{i+1} is a new perturbed state, deviating from the desired state S^j_{i+1}. If the transition Tr_{dam} perturbs only one feature of an object, we speak about single Tr^s_{dam}, if Tr_{dam} perturbs simultaneously several features of one or more objects, we speak about multiple Tr^m_{dam}. In this work we generally focus only on Tr^s_{dam}. The aim of planning system is to create a repairing plan Pl_{sec} that returns the system into the state S^j_{i+1}

$$Pl_{sec} = \{Tr^j_s(S'^j_{i+1}) \rightarrow S^j_{i+1}\} \tag{6}$$

Considering this expression, we claim there are one-step plans and many-step plans satisfying (6).

Statement 1 *Let N_{FA} be a FA-network and Tr^s_{dam} is a single perturbing transition. If Pl_{sec} is defined as one-step plan, there is always Tr_s in sense of (6), if and only if the corresponding C_l are satisfied.*

By construction of the N_{FA} network there is always an activity that is able to modify the perturbed attribute in S'^j_{i+1}. Limitation of this activity is determined by the local constraints C_l. Therefore if all C_l are satisfied by Tr_s, the system can achieve the state S^j_{i+1}.

Statement 2 *Let N_{FA} be a FA-network and Tr^s_{dam} is a single perturbing transition. If Pl_{sec} is defined as many-step plan, there is always a sequence of Tr_s in sense of (6) if and only if*
- (1) all local constraints C_l are satisfied;
- (2) all global constraints C_g are satisfied;
- (3) Tr^s_{dam} and Tr_s never intersects.

In this case we follow the previous statement. If Tr^s_{dam} perturbs only one feature, we have a global technology (determined by C_g) of how to change it. If these conditions are satisfied, then we can apply step-by-step the statement 1. Important is that Tr_{dam} does not cause additional perturbation during executing of Pl_{sec}, because an accumulation of several Tr_{dam} may have a nonlinear influence on one another and lead to multiple Tr^m_{dam}. Therefore we require that Tr_{dam} and Tr_s never intersects in this sense. If (1)-(3) are satisfied, and N_{FA} is closed (all features are connected with activities), any arisen perturbations will be propagated in this network, till it will be absorbed.

Now the question is of how to derive Tr_s. From (1) we have

$$S = \{M \times WS \rightarrow T, \{C_g\}\}, \quad S' = \{M' \times WS' \rightarrow T', \{C_g\}\}. \tag{7}$$

Let us define a difference between S and S' as $\triangle S'$. We assume that modifications of machines M and time T can be absorbed by rescheduling. Therefore functional decompositions from $\triangle S'$ concern only working steps (denoted as $\triangle WS'$). The goal of Tr_s is to minimize $\triangle S'$, i.e. we can write

$$Tr_s(M \times WS' \rightarrow T, \{C_g\}) = \triangle S' \tag{8}$$

or with generator Γ: $Tr_s(M \times (\Gamma \rightarrow F, \{C_l\}) \rightarrow T, \{C_g\}) = \triangle S'$, where

$$(\Gamma \rightarrow F, \{C_l\}) = \triangle WS'. \tag{9}$$

Expressions (8) and (9) give us a practical way to derive Tr_s. Thus, a *task decomposition represents a systematic way to find a difference between real state and desired state in a form of* *working steps, generated by* Γ*. For many-step plans this rule should be applied on each step of executing, moreover this sequence of generated working states converges in sense of (6).* So far as the problem, before decomposition, should be first formulated in a symbolic form (as FA-network), we call this algorithm as the symbolic tasks decomposition. Complexity of the FA-network can be estimated from different viewpoints: Kolmogorov-Sinai's complexity, Kullback-Leibler's cross-entropy or information capacity of interconnections. However it represents a future work.

The most simple algorithm that implements (8), (9) in agent-based executing has the following form: This algorithm assumes the FA-network is already constructed, moreover

```
agent:role (monitoring)
  do always monitor attributes
                     of the plan
  activate if  not equal do
        {call FA-connected agent,
        role=change attribute}
     finish if receive cost
  endactivate
endrole
```

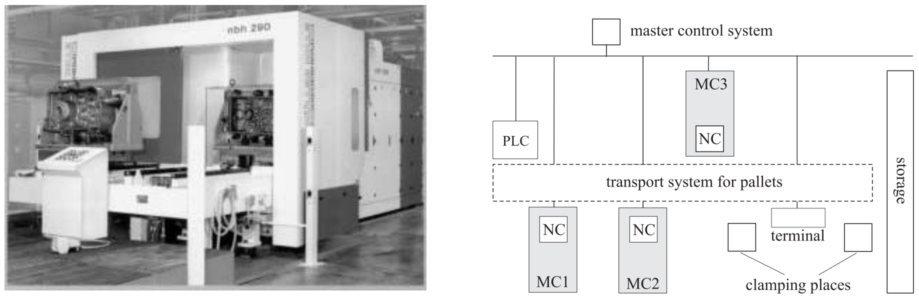
```
agent:role (change attribute)
  do always monitor attributes of the plan
  activate if  not equal do take role=monitoring
     finish if receive cost
  endactivate
  activate if  equal do change attribute
     finish if calculate cost
  endactivate
endrole
```

there is only Tr^{s}_{dam}. The role "monitoring" detects a deviation from a plan and calls the role "change attribute" of a connected agent. The connected agent (from FA-network) compares the executing conditions, in the case of mismatch, it calls the role "monitoring", otherwise it changes an attribute, calculates cost and returns this cost to a parent activity. Application this algorithm is considered in the next section.

5 Experiments in TMS-scenario

The ideas described in this paper have been implemented in the TMS-scenario (Transformable Manufacturing Systems). In this scenario there are three machine shops, each of them contains three-five reconfigurable and non-reconfigurable processing machines. Each of non-reconfigurable machines is able to perform two different (of four required) processing operations. These machines are connected by a conveyer belt, so that details can be transported from one machine to another within one machine shop (see fig. 6). There is also a transporter being able to transport details from one shop to another. One shop consists completely of reconfigurable machines, one shop has only one reconfigurable machine and the last shop consists only of non-reconfigurable machines. The planning system uses Agent-Based Scheduling Engine ABSE, with CSP-solver and MAS optimization module based on ACO algorithm [3]. The ASTD module as well as the generator Γ are implemented by OpenCybele (see http://www.opencybele.org) in Java, the time-sensitive optimization module is written in C++. The whole approach for 55-nodes in CSP- and FA-networks executes in real time.

Different disturbances are composed into several scenarios. Development of these scenarios has been motivated to find the worst case for a planning system, where the human assistance can lead to better solution. As shown by experiments, ABSE-ASTD-Γ engine perform in the best way a functional decomposition, but cannot perform completely autonomously an organizational decomposition. Here can the human assistance essentially improve the plan.



(a) (b)

Figure 6: **(a)** *Example of a reconfigurable machine (taken from www.hueller-hille.com);* **(b)** *Layout of one of machine shops, where NC is a numerical control, PLC is a programmable logic controller, and MC is a machine center (taken from [2]).*

6 Conclusion

The suggested ASTD allows solving the problem of huge state space in manufacturing planning systems: *typical regular processes can be preprogrammed, the untypical irregular processes (e.g. reactions on disturbances) can be dynamically generated.* This approach can also be applied in other environments, where the number of state space is huge or an appearance of new states in a plan is possible. Adaptation of ASTD to MDP or similar approaches represents a further task.

This work has been made in the framework of project SFB 467 ”Transformable Business Structures for Multiple-Variant Series Production ”, supported by DFG. We are grateful to Jörg Priese for details of a workpiece shown in fig. 1.

References

[1] B. J. Pine. *Mass Customization. The New Frontier in Business Competition.* Harvard Business School Press, Boston, Mass, 1999.

[2] S. Kornienko, O. Kornienko, and J. Priese. Application of multi-agent planning to the assignment problem. accepted for publication in ‘Computers in Industry‘, 2003.

[3] S. Kornienko, O. Kornienko, and P. Levi. Flexible manufacturing process planning based on the multi-agent technology. In *Proc. of 21st Int. Conf. AIA’03, Innsbruck, Austria*, pages 156–161, 2003.

[4] S.J. Russell. *Artificial intelligence: a modern approach*. Prentice-Hall, 1995.

[5] G. Weiss. *Multiagent systems. A modern approach to distributed artificial intelligence*. MIT Press, 1999.

[6] A. Kusiak. *Intelligent manufacturing systems*. Prentice-Hall, Englewood Cliffs, NJ, 1990.

[7] V. Engelson. Simulation and visualization of autonomous helicopter and service robots. *Linköping Electronic Articles in Computer and Information Science, ISSN 1401-9841*, 5(013), 2000.

[8] B. C. Williams and P. Nayak. A model-based approach to reactive self-configuring systems. In *Proc. of 13th AAAI’96 / 8th IAAI’96*, volume 2, pages 971–978, 1996.

[9] ISO/DIS14649-1. *Part 1: Overview and fundamental principles*. Genf Final DIS, 2000.

[10] G.E. Monahan. A survey of partially observable markov decision processes: Theory, models, and algorithms. *Management Science*, 28(1):1–16, 1982.

[11] H. Haken. *Advanced synergetics*. Springer-Verlag, Berlin, 1983.

[12] O. Kornienko, S. Kornienko, and P. Levi. Collective decision making using natural self-organization in distributed systems. In *Proc. of Int. Conf. CIMCA’01, Las Vegas*, pages 461–471, 2001.