Production Systems and Information Engineering, Miskolc, Vol. 1. (2003), pp. 3-16

NUMERICAL SIMULATION FOR THE DETERMINATION OF THE TEMPERATURE FIELDS AND THE HEAT AFFECTED ZONES IN GRINDING

ATHANASIOS G. MAMALIS
Manufacturing Technology Division, National Technical University of Athens
GR 10682 ATHENS, Greece
mamalis@central.ntua.gr

JÁNOS KUNDRÁK
Department of Production Engineering, University of Miskolc
H-3515 MISKOLC, Hungary
kundrak@gold.uni-miskolc.hu

ANGELOS MARKOPOULOS
Manufacturing Technology Division, National Technical University of Athens
GR 10682 ATHENS, Greece
amark@central.ntua.gr

[Received October 2002 and accepted February 2003]

Abstract. A finite element model is proposed for the simulation of grinding of hardened steels with aluminium oxide wheels. For this task the implicit FEM code MARC is used. The proposed model is a transient state, non-linear problem that can calculate the maximum temperature and the temperature fields on the ground workpiece. The input data required for the analysis are provided through a series of grinding experiments. From the numerical results obtained from the analysis it is possible to predict the size of the heat affected zones of the workpiece.

Keywords: Surface grinding, Finite Element Method, Heat affected zones

1. Introduction

Grinding modelling has been a main concern of the researchers dealing with this process due to the difficulties raised in its experimental studying. A lot of models have been used for the mechanical and thermal simulation of grinding and of its components, mainly the workpiece, the grinding wheel, the chip and the coolant. Especially, the thermal modelling of grinding has been extensively investigated because of the importance of the knowledge of the maximum temperature reached during the process and, consequently, of the thermal damage induced to the workpiece because of excessive heat loading. This heat input is responsible for a number of defects in the workpiece like metallurgical alterations, microcracks and residual stresses. The areas of the workpiece that are affected are described as heat affected zones.

An overview of grinding models can be found in Refs [1, 2], where the former lists the majority of models developed for grinding while the latter focuses on thermal modelling.

Since the publications of these references some numerical models and especially Finite Element Method (FEM) models have enriched the relative literature [3-6]. The advances in computers and the introduction of user-friendly FEM software for PCs have given a new turn on the simulation of grinding.

Numerical two-dimensional (2D) and three-dimensional (3D) thermal models using the implicit Finite Element Method (FEM) code MARC in order to simulate grinding have already been presented by the authors [7, 8]. In the present paper the validation of the 2D model was considered when grinding bearing steel 100Cr6 with aluminium oxide wheels. The calculation of the maximum temperature, the distribution of temperature fields developed during grinding and the prediction of the heat affected zones in the workpiece are presented and discussed.

2. FEM modelling of grinding

Almost all models on the thermal modelling of grinding are based on the publication of Jaeger on moving heat sources [9]. In Jaeger's model the grinding wheel is represented by a heat source moving along the surface of the workpiece with a speed equal to the workspeed, see Fig. 1. A two dimensional model is used, provided that the grinding width is large with respect to its length. However, this model is based on the assumption that the total grinding energy is entirely absorbed by the workpiece, whilst in reality the total grinding energy is distributed in the workpiece, the grinding wheel, the chip and the coolant. The initial model was improved by studying the percentage of the total heat entering the workpiece and the effect of cutting fluid on the maximum temperature [10-13] and by the introduction of models where heat capacity and heat conductivity are temperature dependent [6].

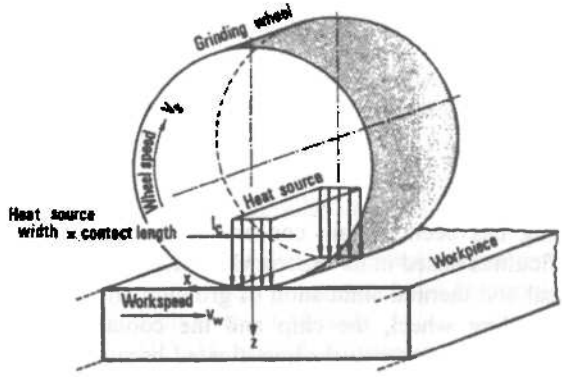


Fig. 1: Jaeger's model [2]

In the present paper a FEM model, based on Jaeger's model, is proposed for the simulation of the grinding process. The model configuration is presented in Fig. 2. On the top surface of the workpiece heat is entering the workpiece, in the form of heat flux, Q, input that moves along this surface. Cooling is simulated by means of convective boundary conditions. All the other sides of the workpiece are considered to be adiabatic, and so no

heat exchange takes place in these sides. The model has a length of 35 mm and a height of 5 mm, sufficient enough for the temperature fields to be fully deployed and observed in full length. A mesh is applied on the proposed model, consisting of 1400 4-noded quadrilateral elements and 1491 nodes. The mesh is denser towards the grinding surface, that is the thermally loaded surface, and, thus the most affected zone of the workpiece, allowing for greater accuracy to be obtained.

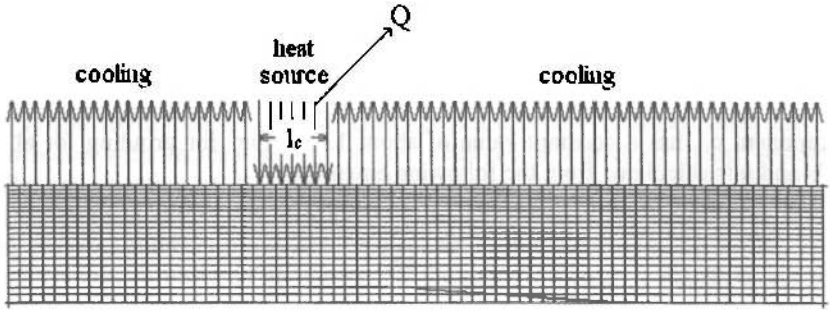


Fig. 2: Suggested thermal finite element model for surface grinding

As mentioned above, the heat source is characterized by a physical quantity, the heat flux, Q, that represents the heat entering the workpiece per unit time and area and it is considered to be of the same density along its length, taken equal to the geometrical contact length, l_c, which is calculated from the relation

$$l_c = \sqrt{a \cdot d_s} \tag{1}$$

where a is the depth of cut and d_s the diameter of the grinding wheel. The real contact length is expected to be bigger due to the deflection of the grinding wheel and the workpiece in the contact area. Nevertheless, the geometrical and real contact lengths are considered to be equal. The heat flux can be calculated from the following equation

$$Q = \varepsilon \frac{F_t' v_s}{l_c} \tag{2}$$

where ε is the percentage of heat flux entering the workpiece, F'_t the tangential force per unit width of the workpiece, v_s the peripheral wheel speed and l_c the geometrical contact length. The proportion of the heat flux entering the workpiece can be calculated by a formula suggested by Malkin for grinding with aluminium oxide wheels, by making assumptions on the partitioning of total specific grinding energy, u, required for grinding [14]. The total specific grinding energy consists of three different components: the specific energy required for the formation and the removal of the chip, u_{ch}, the specific energy required for plowing, i.e. the plastic deformation in the regions where the grains penetrate

the workpiece surface but no material is removed, u_{pl} and the specific energy required for making the flat wear grains slide on the workpiece surface, u_{sl}, thus

$$u = u_{ch} + u_{pl} + u_{sl} \tag{3}$$

It has been analytically and experimentally shown that approximately 55% of the chip formation energy and all the plowing and sliding energy are conducted as heat into the workpiece, i.e.

$$\varepsilon = \frac{0.55 \cdot u_{ch} + u_{pl} + u_{sl}}{u} = \frac{u - 0.45 \cdot u_{ch}}{u} \Rightarrow \varepsilon = 1 - 0.45\frac{u_{ch}}{u} \tag{4}$$

The component u_{ch} has a constant value of about 13.8 J/mm^3 for grinding for all ferrous materials and u is calculated from the following equation:

$$u = \frac{F_t' v_s}{a v_w} \tag{5}$$

where v_w is the workspeed and, consequently, as in Jaeger's model, the speed of the moving heat source. Note that, in both equations (2) and (5) the value of F'_t is needed in order to calculate the heat flux and the total specific grinding energy, respectively; it can be calculated from

$$F_t' = \frac{P_t'}{v_s} \tag{6}$$

where P'_t is the power per unit width of the workpiece, which was measured during the testing of the different grinding wheels, as described in the next paragraph. The kind of modelling suggested in this paper is suitable for grinding process with very small depth of cut, since there is no modelling of the chip.

Note that, in thermal modelling there are two kinds of problems that are being dealt with: the steady-state and the transient problems. In the former it is assumed that the temperature is constant in respect with time, i.e. $\dot{T} = {}^{\partial T}/_{\partial t} = 0$, while in the latter $\dot{T} \neq 0$, as is the case in the presented model. Furthermore, the two coefficients of the workpiece material that are related to temperature, i.e. the thermal conductivity and the specific heat capacity, along with the density of the workpiece need to be inserted as input to the program. Especially the first two were considered to be temperature depended and they were taken from the FEM program material properties data bank. Transient conditions and temperature depended material properties produce non-linear finite element problems, which are more difficult to be solved. The algorithm and the formulae used for the solution of such problems by MARC, which is an implicit FEM software, are given in Appendix 1.

3. Experimental results

Six aluminium oxide grinding wheels of the same diameter, d_s=250 mm and width b_s=20 mm with different bonding were used on a BRH 20 surface grinder. Two depths of cut were used, namely 0.02 mm and 0.05 mm while the workpiece speed was v_w=8 m/min

and the wheel speed v_s=28 m/s kept constant for both sets of experiments, for all wheels. The workpiece material was the 100Cr6 bearing steel. Throughout the process the synthetic coolant Syntilo-4 was applied at 15 ℓ/min. For each grinding wheel, 10 passes of the same depth of cut were performed over the workpiece. The power per unit width of the workpiece was measured for each pass and its average value was calculated. For measuring the power, a precision three-phase wattmeter was used. First, the power of the idle grinding machine was measured and set as the zero point of the instrument. Then, the workpieces were properly ground and the power was registered on the measuring device. After 10 passes were performed the grinding wheel was dressed with a single point diamond dressing tool, with depth a_d=0.02 mm and feed of f_d=0.1-0.2 mm/wheel rev. The measured quantities, as well as the quantities calculated from equations (1)-(6), are tabulated in Table 1 (a) and (b) for each depth of cut. According to the measured data the specific energy increases with decreasing depth of cut. This can be explained by the so-called size effect. The cross section of the chip, which is smaller for smaller depths of cut, possess different mechanical properties at microscale as compared with macroscale, due to the existence of fewer numbers of dislocations, grit faults and inclusions. Therefore, the micro hardness increases, resulting in an increase of the specific energy in grinding, see also Ref [2].

Table 1: Experimental measurements and calculated results

(a) Depth of cut, a=0.02 mm

	Experimental	Calculated				
Grinding wheel	P'_t (W/mm)	F'_t (N/mm)	u (J/mm^3)	ε (%)	l_c (mm)	Q (W/mm^2)
No. 1	143.5	5.13	53.81	0.885	2.24	56.77
No. 2	164.0	5.86	61.50	0.899	2.24	65.94
No. 3	176.0	6.29	66.00	0.906	2.24	71.30
No. 4	132.5	4.73	49.69	0.875	2.24	51.85
No. 5	170.5	6.09	63.94	0.903	2.24	68.84
No. 6	141.0	5.04	52.88	0.883	2.24	55.65

(b) Depth of cut, a=0.05 mm

	Experimental	Calculated				
Grinding wheel	P'_t (W/mm)	F'_t (N/mm)	u (J/mm^3)	ε (%)	l_c (mm)	Q (W/mm^2)
No. 1	277.5	9.95	41.78	0.851	3.54	67.06
No. 2	288.5	10.29	43.20	0.856	3.54	69.75
No. 3	298.5	10.66	44.78	0.861	3.54	72.72
No. 4	256.0	9.14	38.40	0.838	3.54	60.70
No. 5	277.5	9.91	41.63	0.851	3.54	66.78
No. 6	378.5	13.52	56.78	0.891	3.54	95.35

4. Results and discussion

4.1 Temperature fields

The parameters reported on the previous paragraph are applied to the model. In Fig 3 (a) and (b) variations of the temperatures on the surface of the workpiece, for different grinding wheels and with distance x from the left edge of the workpiece are presented, for each depth of cut. The temperature fields appear to be the same for the same depth of cut and the only difference is the maximum temperature reached for each grinding wheel. From the same figures it is revealed that the temperatures are higher in the regions on the back of the wheel, therefore, it seems that, it is more critical to direct the coolant to this side, in order to prevent the damage of the surface integrity due to the temperature rise.

From the same figures it can be concluded that wheel No. 4 has the lower maximum temperature for both depth of cut, whilst when grinding with wheel No. 6 a significant deviation in temperature between the two cases. This irregularity may be attributed either to the wheel specification being not suitable for the cutting conditions and so the grains are overloaded, or to the increase of the specific energy due to friction. The latter can be explained by the bigger radius in the workpiece-wheel contact zone, when the depth of cut is bigger, leading to the increase of the adhesive effect between the workpiece material and the wheel. The particles that are adhered to the wheel increase the friction surfaces, leading to the consumption of more energy. Such a phenomenon is not rare in grinding technology; it can be avoided by using the right wheel or coolant.

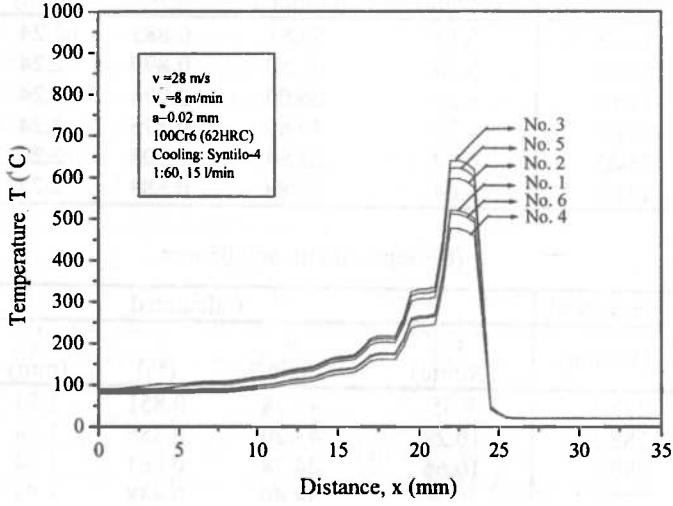


Fig.3 (a): Temperature variation on the surface of the workpiece for depth of cut 0.02 mm

The individual results obtained for grinding wheel No.3, can be seen in Fig 4 (a) and (b) for depth of cut 0.02 mm and 0.05 mm, respectively; the maximum temperature and the temperature fields for a specific step of the analysis are illustrated, in both isothermal bands

within all the workpiece and isothermal lines in the region underneath the grinding wheel, where the maximum temperature appears.

In Fig 5 (a) and (b) the temperatures on the surface (y=0) and underneath it (y>0), for grinding wheel No. 4 are presented. In these diagrams y (mm) is the distance of the nodes of the mesh of the finite element model from the surface, C is the center of the heat source and the region 0-l_{ci} is the current position of the heat source, where l_{ci}, i=1,2, is the geometrical contact length between the grinding wheel and the workpiece.

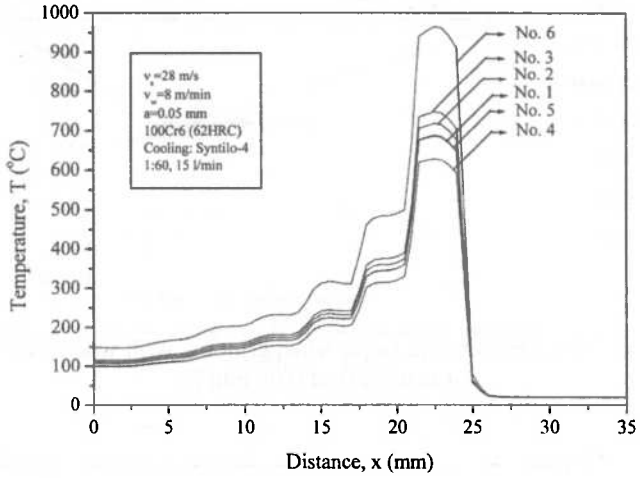


Fig. 3 (b): Temperature variation on the surface of the workpiece for depth of cut 0.05 mm

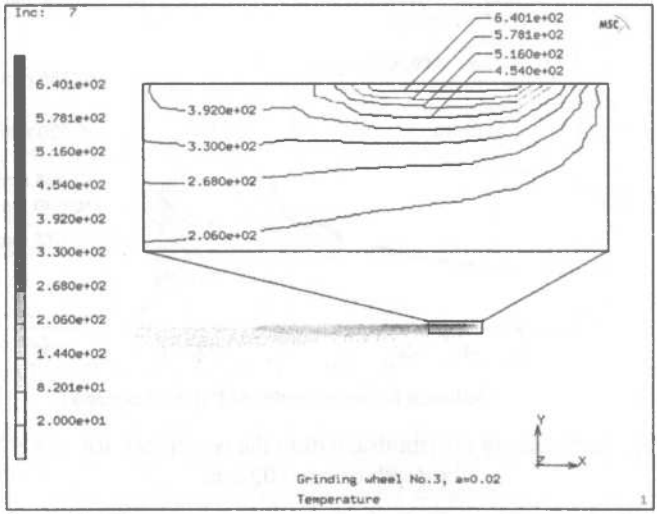


Fig. 4 (a): Temperature fields when grinding with wheel No.3, for depth of cut0.02 mm [7]

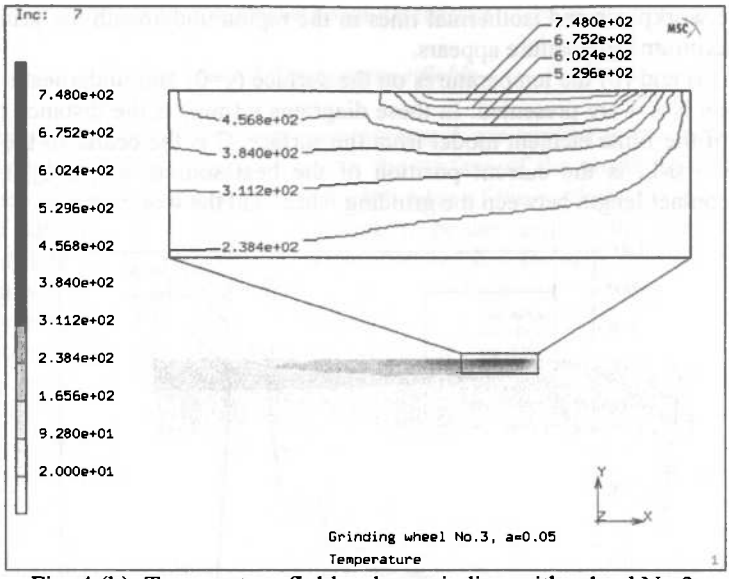


Fig. 4 (b): Temperature fields when grinding with wheel No.3, for depth of cut 0.05 mm [7]

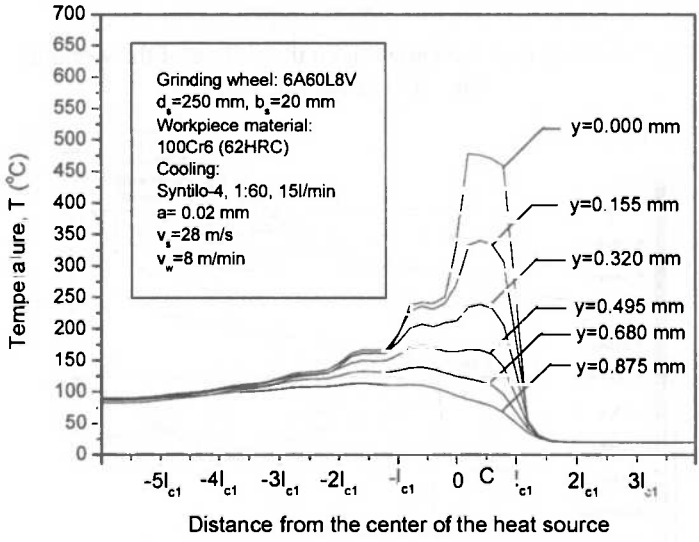


Fig.5 (a): Temperature distribution within the workpiece for wheel No.4, for depth of cut 0.02 mm

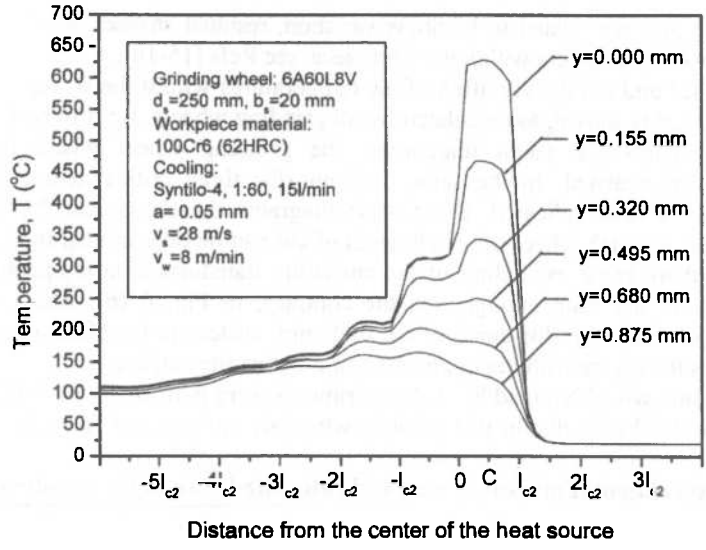


Fig. 5 (b): Temperature distribution within the workpiece for wheel No.4, for depth of cut 0.05 mm

From these figures it can be concluded that the maximum temperature appears to be reported very close to the center of the heat source and that the temperatures below the surface and especially for y>0.320 mm are approximately half than those on the surface of the workpiece. Furthermore, the regions in front of the heat source that are more than 1 l_{ci} away from the center of the heat source are not yet affected, while the regions even at a distance of 5.5 l_{ci} from the same point are still thermally loaded.

4.2 Heat affected zones

As noted before, the high temperatures that usually appear in grinding have a negative effect on the workpiece. The surface of the workpiece and also the layers that are near the surface and have been affected by the heat loading during the grinding process consist the heat affected zones of the workpiece. The excessive temperature in these zones contributes to residual stresses, microcracking and tempering and may cause microstructure changes, which result to hardness variations of the workpiece surface. Steels that cool down quickly from temperatures above the austenitic transformation temperature undergo metallurgical transformations; as a result, untempered martensite is produced which forms the so-called "white layer" in the workpiece. Excessive heat may also lead to metallurgical burn of the workpiece, which produces a bluish color on the surface of the processed material due to oxidation. If the critical temperatures at which these transformations take place are known, the size of the heat affected zones can be also "predicted" from the FEM model. The actual size of these zones and their composition depends on the duration of thermal loading, except the maximum temperature reached. The three critical temperatures for the 100Cr6 steel are [2]: T_T=150 °C for tempering, T_M=250 °C for martensitic and, T_A=800 °C for austenitic

transformation and are related to hardness variation, residual stresses and the formation of untempered martensite layers within the workpiece, see Refs [15-18].

In Fig 6 (a) and (b) the variation of the temperatures within the workpiece with depth below the surface is shown, as calculated for all grinding wheels, for different depths of cut. These temperatures are taken underneath the grinding wheel where the maximum temperatures are reached. In the same diagrams the three critical temperatures for the 100Cr6 steel are also indicated. From these diagrams the theoretical depth of the heat affected zones, for each wheel used and depth of cut can be determined. In Fig. 6 (a) it can be seen that there is no exceeding of the austenitic transformation temperature since the temperatures are not high enough. On the contrary, in Fig. 6 (b), when grinding with grinding wheel No. 6 and for depth of cut 0.05 mm, austenitic transformation temperature is exceeded in the layers with depth up to 0.1 mm below the surface.

For grinding wheel No.6 additional experiments were performed for depths of cut 0.01 and 0.03 mm. All the results for this grinding wheel are tabulated in Table 2.

Table 2: Experimental measurements and calculated results for grinding wheel No.6

	Experimental	Calculated				
Depth of cut (mm)	P'_t (W/mm)	F'_t (N/mm)	u (J/mm^3)	ε (%)	l_c (mm)	Q (W/mm^2)
0.01	102.8	3.66	76.88	0.919	1.58	59.59
0.02	141.0	5.04	52.88	0.883	2.24	55.65
0.03	202.5	7.23	50.63	0.877	2.74	64.87
0.05	378.5	13.52	56.78	0.891	3.54	95.35

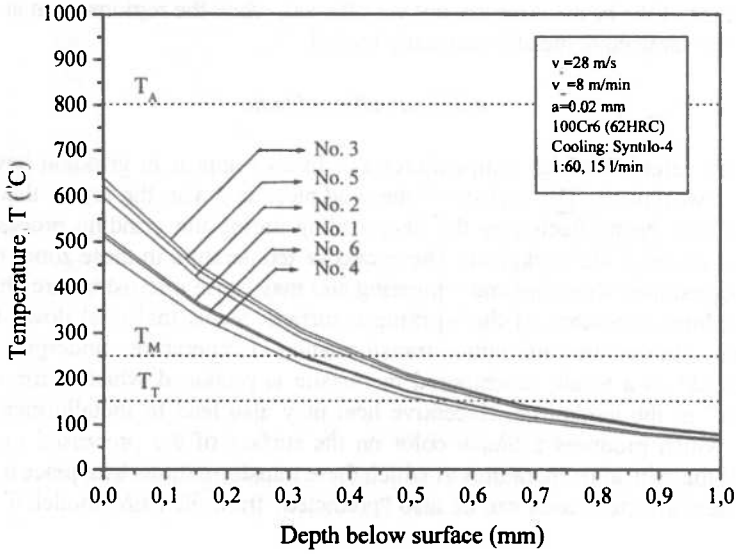


Fig. 6 (a): Variation of temperature with depth below surface when grinding 100Cr6 steel with various grinding wheels for a depth of cut 0.02 mm

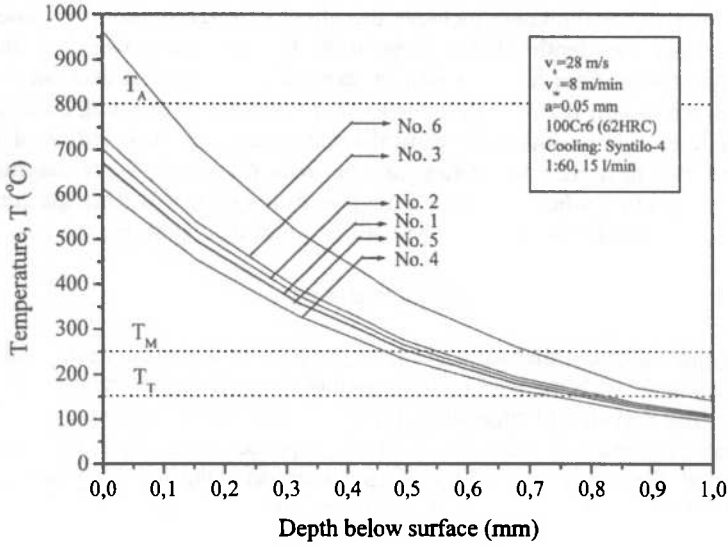


Fig. 6 (b): Variation of temperature with depth below surface when grinding 100Cr6 steel with various grinding wheels for a depth of cut 0.05 mm

By using these data a diagram is presented, see Fig. 7, to approximately predict the temperatures on the surface and within the workpiece for different values of the equivalent chip thickness, h_{eq} (mm), which is calculated as

$$h_{eq} = \frac{v_w}{v_s} a \tag{7}$$

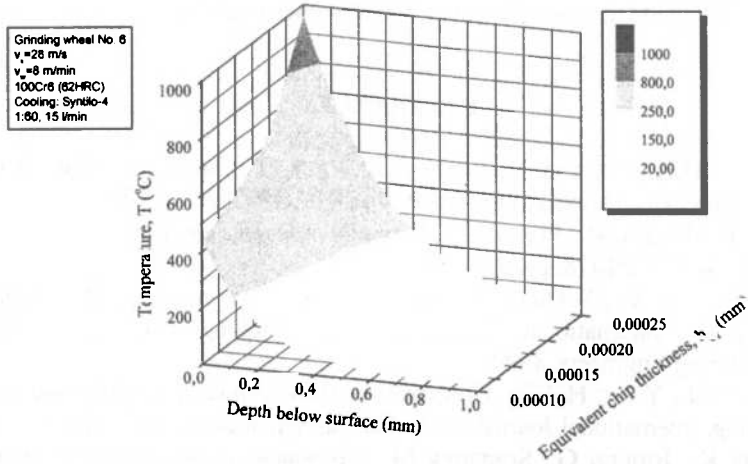


Fig. 7: Variation of temperature with depth below surface and equivalent chip thickness, h_{eq}, when grinding 100Cr6 steel with grinding wheel 6.

The equivalent chip thickness includes the effect of three grinding parameters and is more suitable than the depth of cut to be used for the optimization of the grinding parameters. In order to observe, or to limit, its critical value it is not necessarily needed to decrease the depth of cut; it is also possible to alter suitably the grinding conditions. In the same diagram the regions within the critical temperatures are also indicated by contour bands, so that the heat affected zones can be also predicted. Such diagrams can be constructed for the other wheels as well, from the results extracted from the finite element model and used as a guide for choosing the optimal grinding conditions.

5. Conclusions

From the finite element thermal analysis of grinding of hardened steels with aluminium oxide grinding wheels with the implicit finite element code MARC some useful conclusions may be drawn. The maximum temperature and the distribution of the temperature fields in the workpiece can be successfully calculated with the proposed model, when knowing the power or the tangential force per unit width of the workpiece during the process. From the temperature fields derived from the model, the heat affected zones of the workpiece can be predicted, considering the critical temperatures for tempering, martensitic and austenitic transformation. Furthermore, it is possible to correlate the temperature fields and thus the heat damage induced to the workpiece with kinematical and geometrical parameters of the process in order to find the optimal grinding conditions. Note that, the calculation of temperature is also based on kinematical and geometrical parameters. That allows for the monitoring of the process without using any temperature measurement devices.

The proposed model is relatively simple and very fast, since it takes only a few seconds for running on a modern PC; the total time depends on the number of steps and the parameters applied. It provides a very reliable tool and could replace difficult and time-consuming experiments. Furthermore, it can provide data, like the temperature fields in the workpiece, that it could be very laborious to obtain otherwise.

REFERENCES

1. TÖNSHOFF H.K. PETERS J., INASAKI I., PAUL T.: *Modelling and simulation of grinding processes*, Annals of the CIRP, Vol. 41/2/1992, pp. 677-688.
2. SNOEYS R., MARIS M., PETERS J.: *Thermally induced damage in grinding*, Annals of the CIRP, Vol.27/2/1978, pp. 571-581.
3. HOFFMEISTER H.W. WEBER T.: *Simulation of grinding by means of the finite element analysis*, 3rd International Machining and Grinding Conference, Society of Manufacturing Engineers, 1999.
4. MOULIK P.N., YANG H.T.Y., CHANDRASEKAR S.: *Simulation of thermal stresses due to grinding*, International Journal of Mechanical Sciences, 43, pp. 831-851, 2001.
5. WEINERT K., JOHLEN G., SCNEIDER M.: *Experimental and numerical studies of the grinding process on tool life*, Production engineering, Vol. V/2,1998, pp. 9-14.
6. MAHDI M., ZHANG L.: *The finite element thermal analysis of grinding processes by ADINA*, Computers and Structures Vol. 56, No. 2/3, pp. 313-320, 1995.

7. MAMALIS A.G., KUNDRÁK J., MANOLAKOS D.E. GYÁNI K., MARKOPOULOS A. HORVÁTH M.: *Effect of the workpiece material on the heat affected zones durin grinding: numerical simulation*, International Journal of Advanced Manufacturing Technology 4/10/2002 (In press)
8. MAMALIS A.G., KUNDRÁK J., MANOLAKOS D.E., MARKOPOULOS A., CHRISTOFOROU G.: *Numerical analysis of surface grinding: comparison between 2D and 3D modelling* Proc. of the microCAD 2003 International Computer Science Conference, Miskolc, Hungary, 2003 (In press).
9. JAEGER J.C.: *Moving sources of heat and the temperature at sliding contacts*, Journal and Proceedings of the Royal Society of New South Wales, Vol. 76/3, pp. 263-264, 1942.
10. OUTWATER J.O., SHAW M.C.: *Surface temperatures in grinding*, Transactions of ASME, 74, pp. 73-86, 1952.
11. DES RUISSEAUX N.R., ZERKLE R.D.: *Temperature in semi-infinite and cylindrical bodies subjected to moving heat sources and surface cooling*, Journal of Heat Transfer, 92, pp. 456-464, 1970.
12. GUO C., MALKIN S., *Energy partition and cooling during grinding*, 3rd International Machining and Grinding Conference, Society of Manufacturing Engineers, 1999.
13. ROWE W.B., MORGAN M.N., ALLANSON D.A.: *An advance in the modelling of thermal effects in the grinding process*, Annals of the CIRP, Vol.40/1/1991, pp. 339-342.
14. MALKIN S.: *Burning limit for surface and cylindrical grinding of steels*, Annals of the CIRP, Vol. 27/1/1978, pp. 233-236.
15. SHAW M.C., VYAS A.: *Heat affected zones in grinding steel*, Annals of the CIRP, Vol.43/1/1994, pp. 279-282.
16. ZHANG L., MAHDI M.: *Applied mechanics in grinding – IV. the mechanism of grinding induced phase transformation*, International Journal of Machine Tools and Manufacture, Vol. 35, No. 10, pp. 1397-1409, 1995.
17. CHANG C.C., SZERI A.Z.: *A thermal analysis of grinding*, Wear, 216, pp. 77-86, 1998.
18. ROWE W.B., PETIT J.A., BOYLE A., MORUZZI J.L.: *Avoidance of thermal damage in grinding and prediction of the damage threshold*, Annals of the CIRP, Vol.37/1/1988, pp. 327-330.

Appendix 1

In this appendix a concise description of the mathematical formulation used for heat transfer analysis by MARC is given. The formulae and the algorithm used are taken by the training guide of MARC Mentat, supplied with the program.

The heat transfer problem can be written, as known, as a differential equation

$$[C]\{\dot{T}\}+[K]\{T\}=\{Q\} \tag{8}$$

where $[C]$ is the heat capacity matrix, $[K]$ the conductivity and convection matrix, $\{T\}$ the vector of the nodal temperatures and $\{Q\}$ the vector of nodal fluxes. In the case of a steady state problem, where $\dot{T}=\frac{\partial T}{\partial t}=0$, the solution can be easily obtained by a matrix inversion

$$\{T\}=[K]^{-1}\{Q\} \tag{9}$$

In the case of transient analysis, where $\dot{T}\neq 0$, which is the case described in this paper, the nodal temperature is approximated at discrete points in time as

$$\{T\}^n=\{T\}(t_0+n\Delta t) \tag{10}$$

MARC program is using a backward difference scheme to approximate the time derivative of the temperature

$$\{\dot{T}\}^n\cong\frac{\{T\}^n-\{T\}^{n-1}}{\Delta t} \tag{11}$$

which, when substituted in equation (8), results in the finite difference scheme

$$\left(\frac{[C]}{\Delta t}+[K]\right)\{T\}^n-\frac{[C]}{\Delta t}\{T\}^{n-1}=\{Q\} \tag{12}$$

that gives the solution of the differential equation (8).

Production Systems and Information Engineering, Miskolc, Vol. 1. (2003), pp. 17-28

OPTIMIZATION OF FACE-MILLING CONDITIONS ON THE BASE OF MATERIAL REMOVAL RATE (MRR)

KÁROLY NEHÉZ*
Department of Information Engineering, University of Miskolc
H-3515 MISKOLC, Hungary
nehez@ait.iit.uni-miskolc.hu

TIBOR TÓTH*
Department of Information Engineering, University of Miskolc
H-3515 MISKOLC, Hungary
toth@ait.iit.uni-miskolc.hu

[Received September and accepted November 2002]

* Production Information Engineering Research Team (PIERT) of the Hungarian Academy of Sciences; Department of Information Engineering of the University of Miskolc

Abstract. There is a significant body of literature related to optimization of cutting processes. This paper shows a new approach to face-milling optimization based upon the intensity of stock removal. The state variable Q {cm^3/min} expresses both technological and production management aspects at the same time. An investigation of the effect of relative cost-deviation will also be shown.

Keywords: optimization of cutting conditions, milling, material removal rate (MRR)

1. Introduction

Determination of optimum cutting conditions is a classic task of technology process planning. In a Computer-Aided Process Planning (CAPP) system, one of the important steps is the selection of those machining parameters which yield optimum results. The success of the machining operation will depend on the selection of machining parameters. We may say that technological parameters have most important roles as they control the economical aspects. Determination of the optimum cutting parameters is of great importance especially for NC/CNC machine tools. The paper shows a new approach to face-milling optimization based upon the intensity of stock removal. The state variable Q {cm^3/min} suggested by us expresses both technological and production management aspects at the same time. The presented method is based on the previous results of several authors [1,2,3,5]. Similar suggestion was initiated from *E. Kiliç* [6] and his colleagues. They introduced the $x = v \cdot f$ state variable that made it possible to find the optimized cutting conditions.

2. Intensity parameters

The mathematical model of the optimization task can be formulated as follows: 1]

$$U = \{u_i\}, \quad i = 1,...,I \tag{1}$$

$$S = \{s_j(\mathbf{u})\}, \quad j = 1,...,J, \quad \mathbf{u} \subset U \tag{2}$$

$$C = \{c_k(\mathbf{u},\mathbf{s})\}, \quad k = 1,...,K, \quad \mathbf{s} \subset S \tag{3}$$

$$E = \{e_j(\mathbf{u})\}, \tag{4}$$

$$\Phi = \{\varphi_m(\mathbf{u},\mathbf{s})\}, \quad m = 1,...,M \tag{5}$$

where

U - set of technological parameters,
S - set of state variables,
C - set of production objective functions,
E - set of state equations
Φ - set of constraining relations.

The relationship (1)-(5), in the case of milling, can be written as follows (Figure 1):

$u_1 \Rightarrow d$ [mm] - depth of cut
$u_2 \Rightarrow w$ [mm] - width of cut,
$u_3 \Rightarrow f$ [m/min] - feed rate

Φ set of constraints, for example: maximum allowable feedrate; the cutting power allowed;
C the objective function depending on the parameters to be optimized and state variables (e.g.: cost or productivity of a given operation element);
S : the set of state variables (e.g.: $s_1 \Rightarrow T$, $s_2 \Rightarrow Q$ where T is the tool life [min], Q is material removal rate [cm^3/min]);
E the set of state equations (e.g.: the extended Taylor tool life equation, $T=T(\mathbf{u})$, material removal rate $Q=Q(\mathbf{u})$;).

Depth of cut and width of cut are depending on the current configuration of the allowance to be removed and its quasi-optimum value can be determined with the aid of geometric modelling.

By means of Q state variable both machining time and operation element cost can easily be expressed in case of face milling. For a given milling operation element we have:

$$Q = d\,w\,f = d\,w\frac{f_z\,z\,n}{1000} = d\,w\frac{f_z \cdot z}{1000}\frac{1000\,v}{\pi \cdot D_c} = d\,w\,f_z\,z\frac{v}{\pi\,D_c}, \tag{6}$$

$$t_m = \frac{V}{Q}, \tag{7}$$

$$\frac{1}{T} = \left[\frac{d^{x_v} f_z^{y_v} w^{z_v} z^{p_v} v}{C_v D_c^{w_v}} \right]^{\frac{1}{m}} = \left[\frac{Q\pi}{C_v d^{1-x_v} f_z^{1-y_v} w^{1-z_v} z^{1-p_v} D_c^{w_v-1}} \right]^{\frac{1}{m}} \tag{8}$$

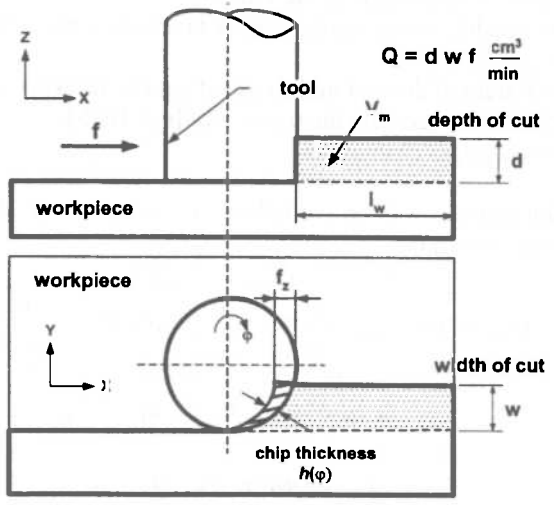


Fig. 1.: End milling cutting geometry

$$\tau = \frac{C_\Sigma}{c_w V_m} = \frac{1}{Q} + \frac{Q^{q-1}}{R^q}, \tag{9}$$

$$R = C_T f_z^{1-y_v} \tag{10}$$

$$C_T = \frac{C_v d^{1-x_v} w^{1-z_v} z^{1-p_v} D_c^{w_v-1}}{\pi \left(\frac{C_T}{c_w N_e} + t_c \right)^m}, \tag{11}$$

$$q = \frac{1}{m}, \tag{12}$$

where:

t_m - machining time [min],
f_z - feed per tooth [mm],
z - number of teeth,
D_c - tool diameter [mm],
V_m - the material volume to be removed [cm^3],

T - tool life [min],
C_v, x_v, y_v, z_v, p_v, w_v - empirical quasi-constants of Taylor equation which are constants for a given workpiece-tool pair;
τ - the specific time of removal (the so-called cost equivalent time) [min/cm^3];
C_Σ - the total cost of the operation [e.g. HUF]
c_w - specific cost of workpiece [e.g. HUF/min];
$R=R(d,w,f_z)$ - a state variable that is a monotonous function of the cutting parameters d, w, f_z [cm^3/min];
C_T - a complex feature of the tool in the case of a given workpiece;
C_t^* - the total cost connected with the use of tool [e.g. HUF],
N_e - number of tool edges changeable.

The optimization process will be carried out in the space of state (Q, R) with the following characteristic constraints:

$$Q_{\min} = d\,w\,f_{z\min}\,z\frac{v_{\min}}{\pi D_c}, \quad Q_{\max} = d\,w\,f_{z\max}\,z\frac{v_{\max}}{\pi D_c},$$

$$R_{\min} = R(f_{z\min}), \quad R_{\max} = R(f_{z\max}),$$

$$Q_{\min} \le Q(d,w,f) \le Q_{\max}, \tag{13}$$

where d and w are previously fixed. Other constraints have similar shape in the space of state (Q,R), but in case of face milling the power constraint has the following form:

$$P_a = F_c\,v, \quad F_c = k_c\,d\,w\,f_z\frac{z}{D_c\,\pi},$$
$$P_a = \frac{k_c\,d\,w\,f_z\,z}{D_c\,\pi\,60\cdot 10^3}\,v = K\,Q \quad [kW], \tag{14}$$

where K is a tool-workpiece specific constant and its value can be found in any tool catalogue [3], P_a is the allowable cutting power, which is represented as a vertical straight line.

Similarly, the *tool deflection* δ_f can be expressed in the plane (Q,R) in the following form:
E - *Young's* modulus of the tool material,
L - tool length,
D_c - cutter diameter,

$$\delta_f = \frac{F_c\,L^3}{3EI}, \quad I = \frac{\pi D_c^4}{64}, \tag{15}$$

$$\delta_f = \frac{64}{3\pi E}\frac{F_c L^3}{D_c^4} = \frac{64}{3\pi^2}\frac{E L^3 k_c d w f_z z}{D_c^3}, \tag{16}$$

where f_z can be expressed as:

$$f_z = \left(\frac{R}{C_s}\right)^{\left(\frac{-1}{y_v-1}\right)} \tag{17}$$

then let us substitute f_z into (16):

$$R = C_T \left(\frac{3\delta_a D_c^3 \pi^2}{64 E L^3 k_c d w z}\right)^{1-y_v} = C_s C_\delta \delta_f^{1-y_v} \tag{18}$$

which can be represented in the plane (Q,R) as a horizontal line.

3. Typical decision making processes in the space of state (*Q, R, τ*)

In the case of face milling the same typical decision making processes can be carried out as in the case of rough turning. The shapes of the allowable power and the tool deflection constraints are vertical and horizontal straight lines. According to definition, the optimization domain is considered regular if the straight lines named valley line and optimum line are within the domain. The Figure 2 shows a regular optimization domain with the main constraints.

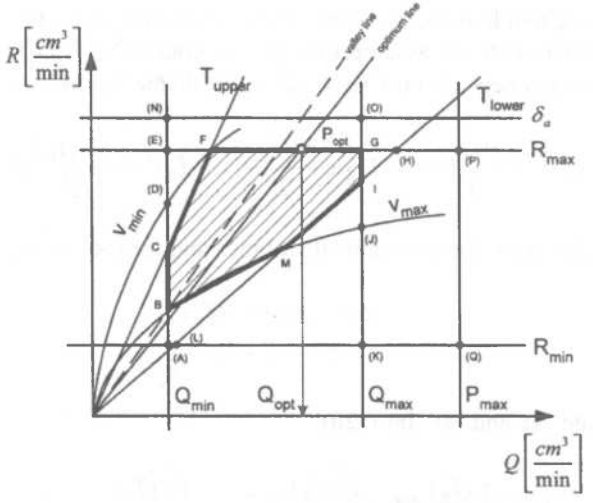


Fig. 2.: The constraints and the domains having a chance to be optimal in the space of state (τ, Q, R)

The direct task of operation element planning can be characterized by the following algorithm:

(1) Tool selection;
(2) Determination of the objective function and the constraints (fixing the depth of cut and width of cut in a heuristic way is also included);
(3) Optimization (Computing Q_{opt}, R_{opt});
(4) Planning the tool motions;
(5) Time computations;
(6) Post-processing (determining f_z).

4. Cost influence of tool-life's dispersion

On account of inhomogeneity of workpiece materials, in practice tool life has some dispersion. We prescribed an optimal tool life, but shop floor practice may experience difference between the prescribed and real tool-life. This dispersion may effect more or less cost.

Let us check that how this dispersion influences the total cost function (9). Let us express the total cost-equivalent time function as a function of Q and T:

$$\tau(Q,T) = \frac{1}{Q} + \frac{t_t^* + t_c}{TQ}, \tag{19}$$

where,

t_t^* - the so-called "tool cost equivalent time" [min],
t_c - the average time necessary for changing a tool edge [min].

We will examine that how the cost does change if the real tool life is not the prescribed T_0 but different. Therewith we assume that Q is a constant ($Q=Q_0$) therefore the cost-equivalent time function depends on T only. Let us create the Taylor's expansion of τ at T_0:

$$\tau(T) = \tau(T_0) + \frac{\tau'(T_0)}{1!}(T-T_0) + \frac{\tau''(T_0)}{2!}(T-T_0)^2 + \frac{\tau'''(T_0)}{3!}(T-T_0)^3 + \ldots \tag{20}$$

The absolute deviation for cost-equivalent times and for tool lives, respectively, are as follows:

$$\Delta\tau = \tau(T) - \tau(T_0), \tag{21}$$

$$\Delta T = T - T_0 \tag{22}$$

Let us substitute $\Delta\tau$ and ΔT into (20):

$$\Delta\tau = \frac{\tau'(T_0)}{1!}\Delta T + \frac{\tau''(T_0)}{2!}(\Delta T)^2 + \frac{\tau'''(T_0)}{3!}(\Delta T)^3 + \ldots . \tag{23}$$

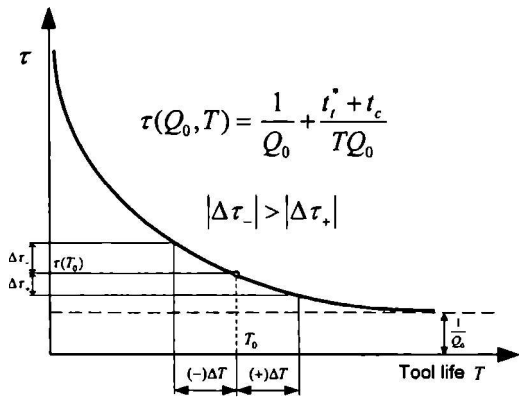


Fig. 3.: The cost equivalent time function as a one variable function of T (tool life)

We are finding $\frac{\Delta\tau}{\tau}$ relative difference of cost-equvivalent times as a function of $\frac{\Delta T}{T_0}$

After executing derivations, (22) can be rearranged as follows: (the mathemetical details can be found in the Appendix A):

$$\frac{\Delta\tau}{\tau}=-\frac{1}{1+\frac{T_0}{t_t^*+t_c}}\cdot\frac{\frac{\Delta T}{T_0}}{1+\frac{\Delta T}{T_0}}. \tag{24}$$

If the optimization domain is regular, we can do further reduction, because in this case $T_0=\left(t_t^*+t_c\right)(q-1)$. After the substitution we will get the following: (the mathemetical details can be found in the Appendix B)

$$\frac{\Delta\tau}{\tau}=-\frac{1}{q}\cdot\frac{\frac{\Delta T}{T_0}}{1+\frac{\Delta T}{T_0}}. \tag{25}$$

According to (24) we can claim that if the optimization domain is regular then relative cost deviation does not depend on t_t^* and t_c but it depends on $q=\frac{1}{m}$ (empirical quasi-constants of Taylor equation) only. Therefore there is a direct proportionality between m and relative difference of cost-equivalent time. We can represent (24) easily by means of MatLab software package. Fig. 4 shows that, if the relative tool life dispersion 40% then cost-equivalent time difference is 18% only! (10% tool life dispersion causes 2% cost-equivalent time difference only.)

The main objective of this investigation is that if we have an allowable relative cost deviation domain, we can also define a relative tool life deviation domain. If at shop floor level we experienced more tool life deviation than the prescribed one during machining, we can not use any static optimization, we must use dynamic optimization strategies.

5. Conclusions

A new approach to face milling optimization based upon the intensity of stock removal has been presented. The applicability limits of the new method were described with mathematical formulas considering the stochastical characteristic of the tool life. Continuing work will focus on finishing and semi-finishing of the end milling operation and determination of the cutting force.

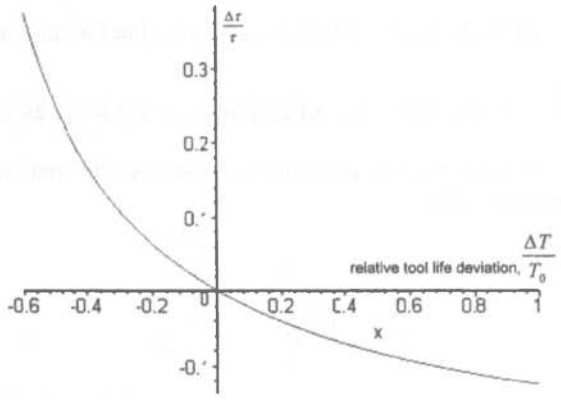


Fig. 4.: Relative cost equivalent time function difference ($\Delta\tau/\tau$) versus relative tool life deviation ($\Delta T/T$) , where m= 0.25

6. Acknowledgements

The scientific investigation, summarized briefly in the paper, are closely connected with the research project of Research and Development Competition for Higher Education in Hungary (FKFP) entitled "Object Oriented Modelling of Manufaturing Processes" (Id. no.: 0275, headed by F. Erdelyi). The projects of Production Information Engineering Research Team (PIERT) established at the Department of Information Engineering and supported by the Hungarian Academy of Sciences have also been used as additional resources. The financial support of the research by the two mentioned sources is gratefully acknowledged.

REFERENCES

1. TÓTH, T.: *Design and Planning Principles, Models and Methods in Computer Integrated Manufacturing*, Publisher of the University of Miskolc, 1998, 252 p. (in Hungarian)
2. RAYEGANI, F.: *Computerized Process Planning and Production Scheduling for Discrete Manufacturing*, Ph.D. Thesis, University of Miskolc, Institute of Information Science, Hungary, 1999, 154 p.
3. SANDVIK COROMANT: *Tools for Die and Mould Makers* – Tool Catalogue 2000, p.114.
4. TÓTH, T., RAYEGANI, F., ERDÉLYI, F.: *Integration of Process Planning and Production Planning Activities in CIM Environment*, Preceedings of the third international symposium TMCE 2000, Delft, The Netherlands, April 18-21, 2000, pp. 717-729.
5. DETZKY, I., FRIDRIK, L., TÓTH, T.: *On a New Approach to Computerized Optimization of Cutting Conditions*, Proceedings of the 2nd World Basque Congress, Advanced Technology and Manufacturing Conference, Bilbao, 1998, Vol. 1. pp. 129-141.
6. KILIÇ, S.E., ÇOGUN, C., SEN, D.T.: *A Computer-aided Graphical Technique for the Optimization of Machining Conditions*, Computers in Industry (22), 1993, pp. 319-326.

APPENDIX A

Mathematical details of relative cost difference in general case

Let us execute the derivation in (22) then we will have the following:

$$\alpha = \frac{t_l^* + t_c}{Q_0},$$

$$\Delta\tau = -\frac{\alpha}{T_0^2}\Delta T + \frac{\alpha}{T_0^3}\Delta T^2 - \frac{\alpha}{T_0^4}\Delta T^3 + \ldots + \frac{\alpha}{T_0^{n+1}}\Delta T^n, \tag{26}$$

$$\Delta\tau = \frac{\alpha}{T_0}\left[-\left(\frac{\Delta T}{T_0}\right) + \left(\frac{\Delta T}{T_0}\right)^2 - \left(\frac{\Delta T}{T_0}\right)^3 + \ldots + (-1)^n\left(\frac{\Delta T}{T_0}\right)^n\right],$$

$$\Delta\tau = \frac{\alpha}{T_0}\left[\sum_{i=1}^{n}(-1)^n\left(\frac{\Delta T}{T_0}\right)^n\right].$$

Let us determine the relative cost difference, we have:

$$\frac{\Delta\tau}{\tau} = \frac{\frac{\alpha}{T_0}}{\frac{1}{Q_0} + \frac{\alpha}{Q_0}}\left[\sum_{i=1}^{n}(-1)^n\left(\frac{\Delta T}{T_0}\right)^n\right] = \frac{t_l^* + t_c}{T_0 + t_l^* + t_c}\left[\sum_{i=1}^{n}(-1)^n\left(\frac{\Delta T}{T_0}\right)^n\right], \tag{27}$$

which can be simplified to the next form:

$$\frac{\Delta\tau}{\tau} = \frac{1}{\frac{T_0}{t_l^* + t_c} + 1}\left[\sum_{i=1}^{n}(-1)^n\left(\frac{\Delta T}{T_0}\right)^n\right]. \tag{28}$$

We will limit our further investigations that case where $\left|\frac{\Delta T}{T_0}\right| < 1$, which means $-1 < \frac{\Delta T}{T_0} < 1$. The summation in (27) is a convergent, alternating geometrical series therefore, if n goes to infinity we will get:

$$\sum_{i=1}^{n}(-1)^n\left(\frac{\Delta T}{T_0}\right)^n = -\frac{\Delta T}{T_0} + \left(\frac{\Delta T}{T_0}\right)^2 - \left(\frac{\Delta T}{T_0}\right)^3 + \left(\frac{\Delta T}{T_0}\right)^4. \tag{29}$$

According to the summation rule of geometrical series $a_1 = -\frac{\Delta T}{T_0}$ and $q = -\frac{\Delta T}{T_0}$,

therefore the addition formula will be as follows:

$$\sum_{i=1}^{n\to\infty} (-1)^n \left(\frac{\Delta T}{T_0}\right)^n = \frac{a}{1-q} = \frac{-\frac{\Delta T}{T_0}}{1+\frac{\Delta T}{T_0}}. \tag{30}$$

Let us substitute (29) into 27:

$$\frac{\Delta\tau}{\tau} = -\frac{1}{1+\frac{T_0}{t_t^* + t_c}} \frac{\frac{\Delta T}{T_0}}{1+\frac{\Delta T}{T_0}}. \tag{31}$$

APPENDIX B

Reduction possibilities in case of regular optimization domain

The optimization domain is regular, if the straight lines of valley line and optimum line are within the regular domain. In this case the optimum tool life can be expressed as:

$$R = MIN(\partial\tau/\partial Q)$$

$$\frac{\partial\tau}{\partial Q} = \frac{1}{Q^2} + \frac{(q-1)Q^{q-2}}{R^q} = 0; \tag{32}$$

from which:

$$R = (q-1)^{\frac{1}{q}} Q \tag{33}$$

The gradient of optimum line is $(q-1)^{\frac{1}{q}}$ and $C_T = \left(\frac{T_0}{t_t^* + t_c}\right)^{\frac{1}{q}}$ for tool life. In case of regular optimization domain this two gradient will be equal:

$$\left(\frac{T_0}{t_t^* + t_c}\right)^{\frac{1}{q}} = (q-1)^{\frac{1}{q}}, \tag{34}$$

and we can express T_0 for it:

$$T_0 = \left(t_i^* + t_c\right)(q-1) \tag{35}$$

Let us substitute it into (23):

$$\frac{\Delta\tau}{\tau} = -\frac{1}{1+\dfrac{\left(t_i^* + t_c\right)(q-1)}{\left(t_i^* + t_c\right)}} \frac{\dfrac{\Delta T}{T_0}}{1+\dfrac{\Delta T}{T_0}}, \tag{36}$$

$$\frac{\Delta\tau}{\tau} = -\frac{1}{q} \frac{\dfrac{\Delta T}{T_0}}{1+\dfrac{\Delta T}{T_0}}. \tag{37}$$

Production Systems and Information Engineering, Miskolc, Vol. 1. (2003), pp. 29-39

CUTTING FORCE MODELING POSSIBILITIES IN OPENGL BASED MILLING SIMULATORS

KÁROLY NEHÉZ*
Department of Information Engineering, University of Miskolc
H-3515 MISKOLC, Hungary
nehez@ait.iit.uni-miskolc.hu

TIBOR CSÁKI
Department of Machine Tools, University of Miskolc
H-3515 MISKOLC, Hungary
csaki@szgtirix.szgt.uni-miskolc.hu

[Received April and accepted October 2002]

* Production Information Engineering Research Team (PIERT) of the Hungarian Academy of Sciences; Department of Information Engineering of the University of Miskolc

Abstract. In the past years, our research activities were focused on several fields in machining simulation and verification. A new milling simulation method and a software application was developed based on pure OpenGL calculating and rendering techniques [1]. As a next step, we tried to investigate the possibilities and practical applications of cutting force modeling in the simulator. This paper examines the developing possibilities of this new feature and describes methods found in the literature [4-11] to determine the cutting force and helps to calculate average or instantaneous cutting force in each tool motion during milling process using the simplest 'volumetric' and an advanced mechanistic model. Using these methods, feedrate can be adjusted more precisely. Since the tool is modeled with a set of polygons in the simulator, therefore it is easy to approximate the instantaneous contact area between tool and workpiece in each tool motion

Keywords: milling simulation, OpenGL, cutting force calculation

1. Introduction

In the early 20th century, *F.W Taylor* performed experiments for 26 years to create a simple solution for the intricate problem of setting safe and efficient cutting conditions. Despite of the progress in cutting tool technology during the last century it is still not easy to determine the optimum spindle speed and federates for metal removal processes. A direct relationship can be found between the cutting conditions and the economics of the metal removal process. The goal is to obtain the least machining time or machining cost while maintaining the safe cutting conditions and high part quality.

Currently, most machining shops employ the traditional method of constant federate cutting for sculptured surface parts. This can result in significant tolerance deviations. [3] The different machining stages have very different requirements. In case of finishing the main aim to ensure constant tool deflection, resulting better tolerances. In roughing, we try to reduce breaking strength of the tool shank, note that in roughing the removable volume is larger than in case of finishing.

2. Calculation models

As it was mentioned in the abstract, there are number of different approaches for optimization and calculation of cutting conditions. In this paper, the most important methods will be showed: volumetric and mechanistic models. Simulators of commercial CAD/CAM systems support geometrical simulation and verification only.

Simulators with cutting force determination capability should answer the following questions:

- Does the local tool stress exceed the strength of the cutter material?
- Can the machine tool supply enough power to cut the workpiece?
- How much is the maximum bending of the current cutting tool?

Other requirement:

- The algorithm should be very efficient and quick. Note that SSM (sculptured surface machining) data files (e.g. CLData) often consist of thousand of tool motions and we must examine the cutting force for each tool step.

2.1. Volumetric model

Let us introduce the material removal rate (Q) [cm^3/min] and assume that, this quantity is proportion to the power required to cut (P):

$$P = Q \cdot K \text{ , [kW]} \tag{1}$$

where K constant depends on the workpiece material, cutting tool geometry and cutting conditions. The spindle motor power is equal to the tangential cutting force times the tooth velocity. Therefore, the tangential cutting force F_t is easily found by dividing the right side of (1) by the velocity. The radial force is calculated by multiplying the tangential force by a constant K_r:

$$F_t = \frac{Q \cdot K}{v_t} \quad , \quad F_r = K_r F_t \text{ . [N]} \tag{2}$$

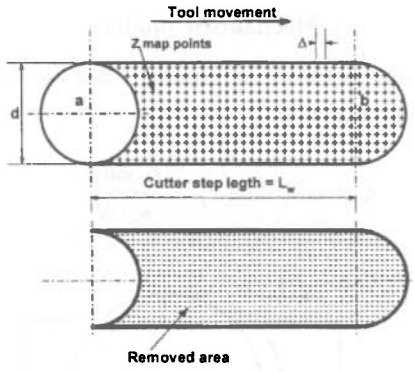


Fig. 1: Tool movement above dexel space

To determine this value for each tool movement we must calculate the volume removed: (see Figure 1)

$$V_m = \sum_{i=1}^{n_f} \left((z_2 - z_1)_i \cdot A_i \right) \qquad [\text{cm}^3], \tag{3}$$

where A_i is the dexel area, z_2 and z_1 are the z values before and after the tool movement, n_f is the number of dexels modified by this particular tool movement:

$$Q = \frac{V_m}{time} = \frac{V_m}{\frac{L_w}{f}} \qquad [\text{cm}^3/\text{min}], \tag{4}$$

where f is the feedrate. The maximum possible error is easily calculated by looking at Figure 1. Unfortunately, this method is probably not useful for finishing. The basic shortcoming of this approach is that it only estimates average forces, not peak or instantaneous forces. (Fig 2)

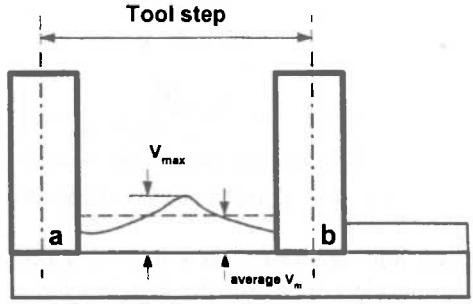


Fig. 2: Average removed volumes can be significantly different from the local maximum volume

2. Mechanistic model

The chip thickness $h(\vartheta)$ varies as shown in Figure 3 and Figure 4. For simple two axis cutting can be used the following equation:

$$h(\vartheta) = f_t \cdot \sin(\vartheta) = \frac{f \cdot \sin \vartheta}{n_t \cdot N} \tag{5}$$

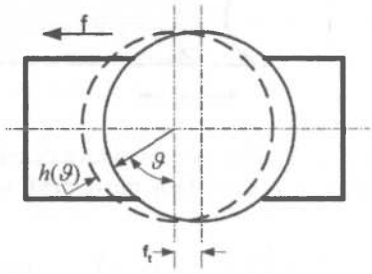


Fig. 3: Chip Thickness

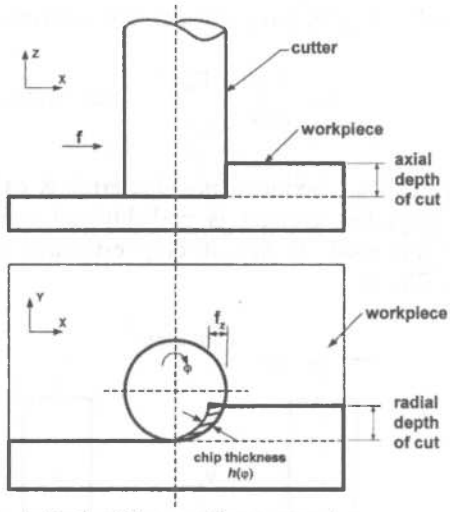


Fig. 4: End milling cutting geometry

In case of three axis milling and five-axis milling, particularly, if the cutting tool is ball-end mill, the equation (5) is not useable. In this case, a more complex equation is appropriate.

$$h(\vartheta,\varphi) = \frac{f(\vartheta,\varphi) \bullet N(\vartheta,\varphi)}{n_t \quad N}, \tag{6}$$

where the scalar product of the feed velocity $f(\varphi,z)$ and the cutting tool surface normal $N(\vartheta,z)$ determine the chip thickness. Our pure OpenGL based machining simulator can easily determine the contact-area by means of its stencil-buffer. The normal vectors can be coded as color values and easily rereadable from the color-buffer for further calculation.

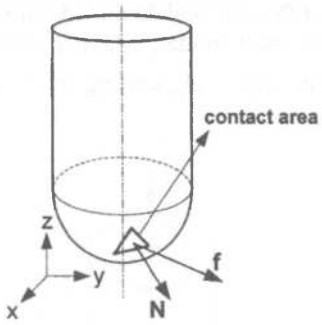


Fig. 5: Contact area on the ball-end cutter

2.2. Empirical cutting force model

Figure 6 shows a modell of the ball-end mill, where ϑ means the cutter rotation angle. There is a direct proportionality between chip thickness and cutting force. Tangential and radial components of the cutting force can be formulated as: [4]:

$$\begin{aligned} \Delta F_t &= K_t\, h(\varphi,z) \\ \Delta F_r &= K_r\, \Delta F_t\,. \end{aligned} \tag{7}$$

K_t and K_r depend on workpiece material and tool and cutting temperature and tool geometry. The force on the cutting edge depends on the cutter rotation angle and the helix angle. There are number of studies about the empirical evaluation of these constants [4-11]. In our special situation (7) can be expressed as:

$$\begin{aligned} dF_T &= K_T(\varphi)\, h(\vartheta,\varphi)^{m_T}\, dz \\ dF_R &= K_R(\varphi)\, h(\vartheta,\varphi)^{m_R}\, dz\,, \end{aligned} \tag{8}$$

where $0< m_R <1$ és $0< m_T <1$. K_T and K_R are function of φ angle. It is verifiable with lot of cutting test that $K_R(\varphi)$ has maximum at $\varphi = \frac{\pi}{4}$ and $K_T(\varphi)$ is a monoton function of φ. According to [5] these parameters we should approximate on the following way:

$$K_T(\varphi) = a_0 + a_1\varphi + a_2\varphi^2 + a_3\varphi^3$$
$$K_R(\varphi) = b_0 + b_1\varphi + b_2\varphi^2 + b_3\varphi^3 \qquad (9)$$

These eight constant values of (9) can be determined with a lot of test cuts. Figure 7 shows the coordinate system of the used model, where ϑ cutter rotation angle, φ means the cutting edge element position angle, ϑ_{pi} starting angle of the helix, $\psi(\varphi)$ cutting edge position angle.

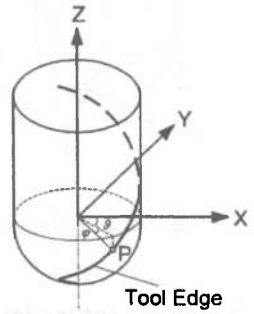


Fig. 6: Geometrical modell of a Ball-End cutter. An arbitrary element of the cutting edge can be experssed with φ, ϑ angles.

According to Figure 7, we can calculate the rotation angle of cutting edge element $\vartheta_i(\varphi)$ of th ith flute as follows [9]:

$$\vartheta_i(\vartheta,\varphi) = \vartheta + \vartheta_{pi} - \psi(\varphi) = \vartheta + (i-1)\frac{2\pi}{n} - \sin(\varphi)\tan(\beta), \qquad (10)$$

where β is the helix angle, n is the number of flutes, i means the current flute. Let us substitute (10) into the equation of a helix and we will get:

$$x = R(\varphi)\cos\vartheta_i(\vartheta,\varphi)$$
$$y = R(\varphi)\sin\vartheta_i(\vartheta,\varphi), \qquad (11)$$

where $R(\varphi) = \sqrt{R-(R-R\cos(\varphi))^2}$ on the sphere and $R(\varphi) = R$ on the cylinder part of cutting tool. We can easily represent (11) with Matlab software. (see: Fig 7) According to

Martelloti [11] the chip thickness can be approximate with (5) but Engin [9] recomended a more general formula:

$$h(\vartheta,\varphi) = f_z \sin\vartheta \sin\varphi\,. \tag{12}$$

In our model it is more natural to apply an other formula to calculate chip thickness as we have shown in equation (6).

If the rotating axis of the tool is non perpendicular to the feed direction (e.g.: real 3 axis machining) then XYZ coordinate system will rotate about X axis by ξ angle. (see: Fig 7) Differential height of the chip segment dz can be expressed with the diameter of the cutter and φ angle:

$$dz = R\cos\varphi\, d\varphi\,. \tag{13}$$

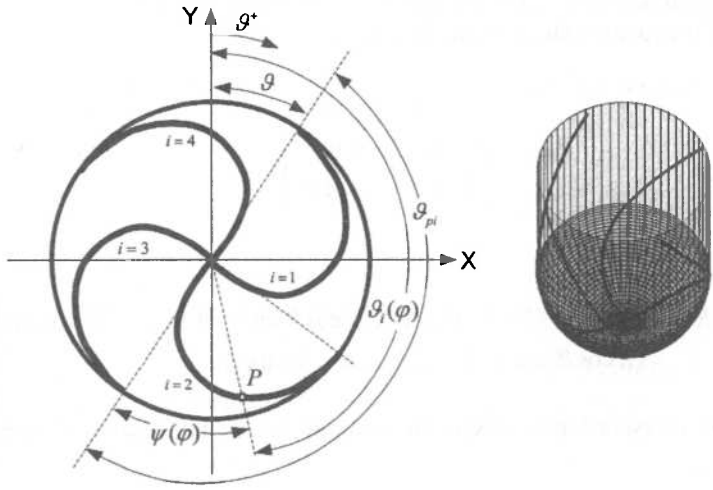


Fig. 7: Left side of the picture shows geometric model of the ball-end cutter on the right side can be seen MATLAB representation of (11).

The coordinates of cutting edge element P in the XYZ coordinate system:

$$P = (R\cos\vartheta\cos\varphi, R\cos\vartheta\sin\varphi, -R\sin\varphi)\,. \tag{14}$$

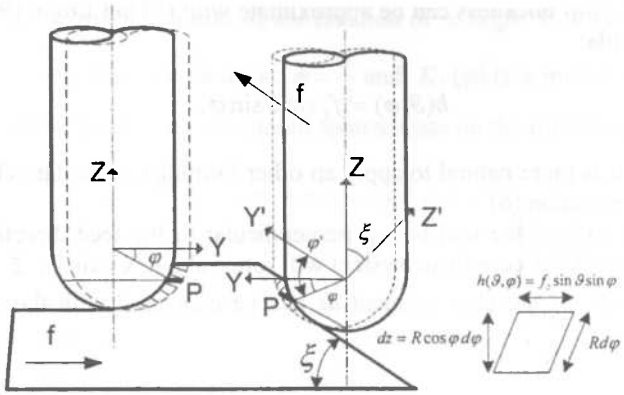


Fig. 8: In the case of upward/downward ramping the chip geometrycan be expressed in the new X'Y'Z' coordinate system

Matrix form of the rotation about X axis by ξ angle:

$$\mathfrak{R}_{Xrot} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\xi & -\sin\xi \\ 0 & \sin\xi & \cos\xi \end{bmatrix} \tag{15}$$

P in XYZ coordinate system:

$$\begin{aligned} P &= (R\cos\vartheta\cos\varphi, R\cos\vartheta\sin\varphi\cos\xi + R\sin\varphi\sin\xi, R\cos\vartheta\sin\varphi\sin\xi - R\sin\varphi\cos\xi) = \\ &= (R\cos\vartheta'\cos\varphi', R\cos\vartheta'\sin\varphi', -R\sin\varphi'). \end{aligned} \tag{16}$$

We can find out the correspondence between the new and the old coordinates, if we express φ' and ϑ' from (16):

$$\varphi' = -\sin^{-1}(\cos\vartheta\sin\varphi\sin\xi - \sin\varphi\cos\xi), \tag{17}$$

$$\vartheta' = \cos^{-1}\left(\frac{\cos\vartheta\sin\varphi\cos\xi - \sin\varphi\sin\xi}{\sin\varphi'}\right). \tag{18}$$

Tangential and radial elemental forces can be calculated from (8) and (6):

$$\begin{aligned} dF_T &= K_T(\varphi)\left[f_z \bullet N(\vartheta(\vartheta,\varphi),\varphi)\right]^{m_T} R\cos\varphi'\, d\varphi' \\ dF_R &= K_R(\varphi)\left[f_z \bullet N(\vartheta(\vartheta,\varphi),\varphi)\right]^{m_R} R\cos\varphi'\, d\varphi'. \end{aligned} \tag{19}$$

dF_x and dF_y can be calculated with the following formula:

$$\begin{bmatrix} dF_X \\ dF_Y \\ dF_Z \end{bmatrix} = R_Z(\vartheta) \begin{bmatrix} dF_t \\ dF_r \\ dF_A \end{bmatrix}, \tag{20}$$

where:

$$R_Z(\vartheta) = \begin{bmatrix} \cos\vartheta & \sin\vartheta & 0 \\ -\sin\vartheta & \cos\vartheta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{21}$$

Executing the operation in (20) and summarizing along the cutting edge:

$$F_x(\vartheta) = \int \rho\, R \{-K_T(\varphi) [f_z \bullet N(\vartheta(\vartheta,\varphi),\varphi)]^{m_T} \cos\vartheta - K_R(\varphi) [f_z \bullet N(\vartheta(\vartheta,\varphi),\varphi)]^{m_R} \sin\vartheta\} \cos\varphi'\, d\varphi'$$

$$F_y(\vartheta) = \int \rho\, R \left\{K_T(\varphi) [f_z \bullet N(\vartheta(\vartheta,\varphi),\varphi)]^{m_T} \sin\vartheta - K_R(\varphi) [f_z \bullet N(\vartheta'(\vartheta,\varphi),\varphi)]^{m_R} \cos\vartheta\right\} \cos\varphi'\, d\varphi'$$

$$\rho = \begin{cases} 1 & \textit{engaged} \\ 0 & \textit{not engaged} \end{cases} \tag{22}$$

The 3 dimensional contact area, which is obtained from the stencil buffer, is projected onto the cutter plane. The cutting edge elements are also projected onto the cutter plane. The cutting edge element engages in the cutting process if its position is on the contact area. If the cutting edge element engages then the elemental cutting forces are calculated from (8). By numerical integration instanteous cutting forces are calculated from (22).

In Figure 10, you can see a screen-shot of our pure OpenGL based milling simulator, in which we are planning to apply this new cutting force model.

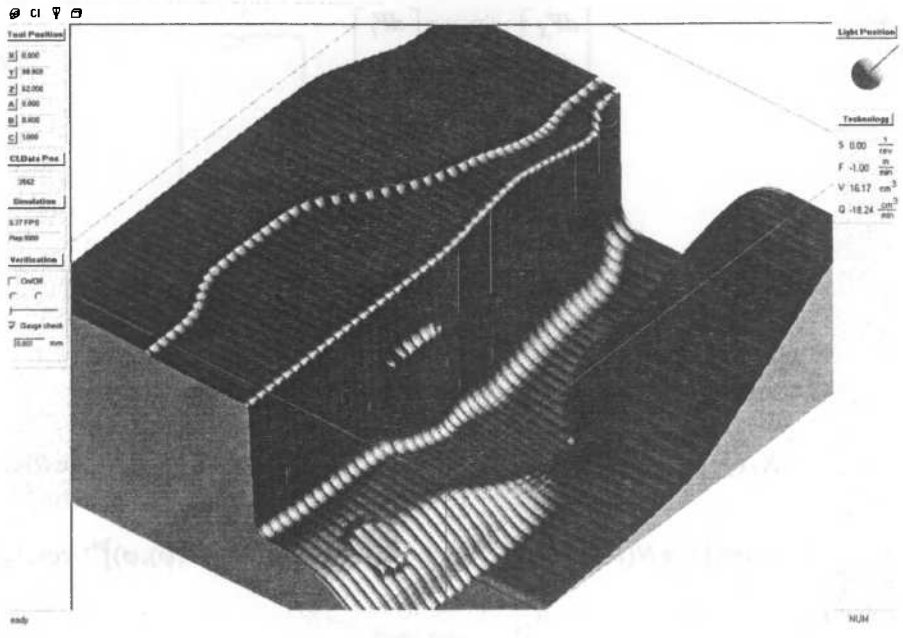


Fig. 9: Our pure OpenGL based milling simulator in which we are planning to apply this new cutting force model

4. Acknowledgements

The author is grateful to the colleagues of the Department of Information Engineering at University of Miskolc. Special thanks are due to Professor Tibor Tóth for his constructive comments and advice.

REFERENCES

NEHÉZ, K., VELEZDI, Gy., CSÁKI, T.: *Efficient Simulation of 3-5 Axis Milling*, MicroCAD International Conference 2001 Miskolc, Hungary.

2. CHOI, B., JERALD, R.: *Sculptured Surface Machining – Theory and Applications*, 1998, ISBN 0-412-78020-8, Kluwer Academic Publisher.

3. JERALD, B., FUSSEL, K., T.: *On-line Optimization of Cutting Conditions for NC Machining*, 2001 NSF Design, Manufacturing & Industrial Innovation Research Conference, Jan 7-10, Tampa, Florida.

4. LEE, P., ALINTAS, Y *Prediction of Ball-end Milling Forces from Orthogonal Cutting Data*, Int. J. Mach. Tools Manufacture, 1995,Vol. 36. No.9, pp. 1059-1072.

5. FENG, H.Y., MENQ, C. H.: *The Prediction of Cutting Forces in the Ball-End Milling Process-I Model Formulation and Model Building Procedure*, Int. J. Mach. Tools Manufacture, 1994, Vol. 34. No.5, pp. 697-710.

6. FENG, H. Y., MENQ, C. H.: *The Prediction of Cutting Forces in the Ball-End Milling Process-II. Cut Geometry Analysis and Modell Verification*, Int. J. Mach. Tools Manufacture, 1994, Vol. 34. No.5, pp. 711-719.

7. TAI, C. C., FUH, K. H.: *Model for Cutting Forces Prediction in Ball-End Milling*, Int. J. Mach. Tools Manufacture, 1994, Vol. 35. No.4, pp. 511-534.

8. LI, H.Z., LI, X.P.: *Milling Force Prediction Using a Dynamic Shear Length Model*, Int. J. Mach. Tools Manufacture, 2001, Vol. 42. pp. 277-286.

9. ENGIN, S., ALTINTAS, Y.: *Mechanics and Dynamics of General Milling Cutters. Part I: Helical End Mills*, Int. J. Mach. Tools Manufacture, 2001, Vol. 41. pp. 2195-2212.

10. ENGIN, S., ALTINTAS, Y.: *Mechanics and Dynamics of General Milling Cutters. Part II: Inserted cutters*, Int. J. Mach. Tools Manufacture, 2001, Vol. 41. pp. 2213-2231. 2001.

11. MARTELLOTI, M.E.: *An analysis of the milling process*, 1941, Trans. ASME 63 677-700.

Production Systems and Information Engineering, Miskolc, Vol. 1. (2003), pp. 41-53

ADVANCED SIMULATION OF NC TURNING OPERATIONS

FERENC ERDÉLYI*
University of Miskolc, Dept. of Information Engineering
H-3515 MISKOLC, Hungary
erdelyi@ait.iit.uni-miskolc.hu

OLIVÉR HORNYÁK*
University of Miskolc, Dept. of Information Engineering
H-3515 MISKOLC, Hungary
hornyak@ait.iit.uni-miskolc.hu

[Received May 2002 and accepted November 2002]

* Production Information Engineering Research Team (PIERT) of the Hungarian Academy of Sciences; Department of Information Engineering of the University of Miskolc.

Abstract. Modern Manufacturing Execution Systems (MESs) provide new possibilities for employing robust process plans including technological alternatives. Decision making activities on MES level can be based on alternatives which had been created at the process planning stage. This requires the improvement of computer aided NC Programming (NCP) and simulation capabilities. This paper deals with the complex simulation and modelling problem of cutting processes and summarises the technical-economic model of turning operations. The operation level model of turning is discussed. The simulation software layout is also presented.

Keywords: MES, NC Simulation, Turning, Artificial Neural Networks

1. Introduction

In modern manufacturing the 80 % of metal parts are machined by means of cutting. The reason of this fact is that in most cases the geometry of cutting tools are relatively simple; and the shaping process on the machine tools is based on NC path generation with the part geometric model. In general, other types of manufacturing cannot compete with cutting on required geometric accuracy and smoothness. In the last years, cutting operations have improved their efficiency. The main directions of improvement are as follows:

- improvement of the real time functions provided by the CNC controllers,
- high concentration of machining operations on machining centres,
- improvement of machine tool materials and tool life,
- widespread use of Flexible Manufacturing Systems and Cells,
- more applications of Computer Aided NC Programming and MES systems.

The manufacturers using CNC technology require more efficient tools for planning and controlling cutting operations. Their needs can be summarized as follows:

ability to machine sophisticated sculptured surfaces,
high utilization rate of automated CNC machines,
reduction of operation times and increasing productivity,
increased quality requirements,
new computer tools to support process planning (CAPP, NCP, CAD/CAM),
more reliable process modelling for optimisation of operation elements,
more precision prediction of operation times, costs and resources consumption ,
more accurate forecasting of quality, dimensional accuracy and risk of waste production, and
creation of alternative process plans.

Computer aided NC programming tools play significant role to meet these demands. The NCP and CAD/CAM systems are the most frequently used tools of Computer Aided Process Planning (CAPP) applications. Conventionally, NCP systems are used to design the cutter location path and then to transform it into NC program. The execution of this program on NC machine tool by means of controlled positioning results the required work piece shapes.

Majority of the modern NC programming systems have their own built-in simulator. The main services of these simulators include the syntactical checking of the part program and the generation of the geometrical model of cutting. By means of the geometrical model, collisions, undercuts and interferences can be detected; estimation of operation times, etc. can be performed. In most cases, however, the physical aspect of cutting process is not modelled at all, thus technical-economic parameters that describe the effectiveness of the cutting process cannot be predicted.

The problem stems from the extreme complexity of cutting processes [3]. Significant research work has been carried out in the field of cutting theory since the 1930's and accumulated in the form of engineering handbooks and electronic databases. However, it is remarkable that even the most widely used NCP applications have no underlying physical models. The research results have not turned into industrial practice. The heuristics of mature technicians and operators are still playing their important role in decision-making.

2. Integration of planning and execution

The improvements in the field of computer networks (LAN) provide opportunity to integrate Process Planning (CAPP), Production Planning and Scheduling (PPS) and Manufacturing Execution Systems. Nevertheless, the existing computer applications which fall into the three major areas of applications mentioned before, do not provide the appropriate services to meet the demands discussed in the introduction.

At MES level numerous criterions are used to assess the goodness of process control. These are as follows:

1. Manufacturing costs;
2. Makespan and lead times;
3. Readiness for delivery (or: delivery capability);
4. Level of work in progress;
5. Utilization rate of resources;
6. Rate of waste products;
7. Flexibility and stability of operations.

Any re-scheduling at the MES level is a dynamic problem which requires the decision making to be supported.

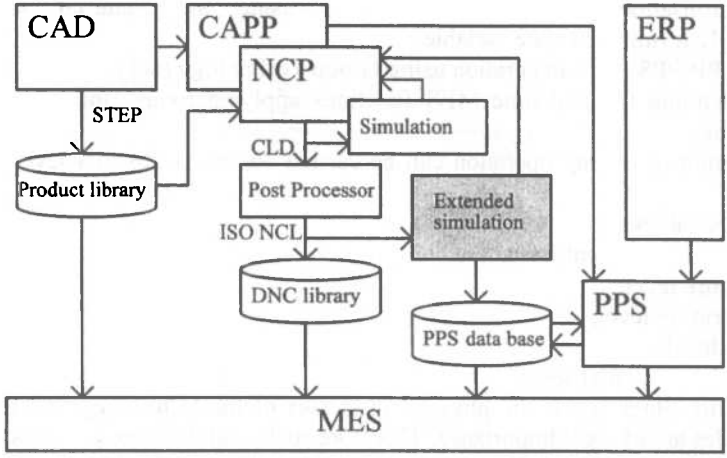


Fig. 1: The role of extended simulation in CIM

Consider the following MES task. Suppose a product assembly where priorities of some orders have been changed due to business reasons. New deadlines have been set which require certain parts to be produced or purchased faster than it does usual or planned originally. At MES level this could be achieved by "make or buy" decisions, rescheduling of jobs, releasing assigned resources or disrupting batch sizes. This would require the knowledge about the range of parameters in which the makespan time of cutting processes can be reduced guaranteeing the same quality, using the cutting tools more intensively which is often coupled with higher probability of waste production. It can be accomplished by having alternative CNC programs archived on the DNC server which had been tested with simulators and ready to download and use. Nevertheless, such alternatives can only be computed on the base of reliable technical- economic models.

The goal of using robust and alternative process plans is to increase the flexibility of MES systems. The supervised amendment of operation times can be successful if the alternatives of the operations are created at design (pre-manufacturing) stage. The synthesis of alternative solutions requires computer aided modelling and simulation techniques. The simulation should include the technical and geometrical aspects of machining as well as the aspect of utilization of resources and the production quality.

3. Technical-economic modelling of turning

The importance of technical-economic modelling of cutting processes is reflected in the fact that CIRP established the "Modelling of Machining Operations" working group in 1997 which presented its comprehensive keynote paper in 1998. [4]. Research work has been carried out in the University of Miskolc, Department of Information Engineering related to the general and specific modelling of cutting includes as follows:

Solving process-monitoring issues by means of Artificial Intelligence (AI) techniques.

Optimisation of cutting parameters using the Material Removal Rate (Q, [cm^3/min]) as state variable.

CAPP-PPS-MES integration using Group Technology (GT).

Extending the real time MES functions applying robust and alternative process plans.

The simulation of cutting operation can be carried out at six abstract levels. They are as follows:

physical level,
operation element level,
feature level,
operation level,
job level,
production order level.

At the first three levels the physical processes of the technology and the continuous state variables are of great importance. Therefore such models are required, which represent the geometrical, kinematical, dynamical and physical characteristics of cutting. The latter three levels belong to the scope of Event Driven Discrete Modelling (EDDM), which does not correspond to the objective of this paper. The conventional methods of cutting, process planning and CNC programming are able to solve certain modelling tasks. These are the geometrical and physical models. However, they cannot support the technical-economic modelling and simulation of cutting operations. The reason of this phenomenon can be found in the difficulties of modelling of machining operations. They are as follows:

The models should handle great number of inputs and outputs.

The relationship between the parameters is extremely complex.

Consideration of material properties cannot be done without empirical knowledge.

Some of the sub-processes can only be described with stochastic variables.

Some computerized methods, which are promising to solve the difficulties listed above, are as follows:

using database of cutting parameters,
AI methods to handle non-linearities,
state equations to describe the dynamic behaviour of processes,
object oriented programming methods,
component based software engineering, as well as
graphical representation and interactive human-machine interface.

The feature level has a significant role among the other levels. It is possible to create a model at this level which can utilize the advances of computer technology listed above and, on the other hand, can facilitate the construction of appropriate models at lower and higher levels.

4. Operation level model of turning

The operation level model of turning, which has the capability to support management decisions, can be summarized as follows:

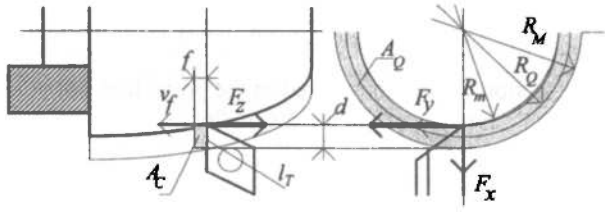


Fig. 2: Geometrical model of turning

4.1. Geometrical relations. The mean diameter of turning is:

$$D_Q(t) = \frac{1}{2}.(D_M + D_m), \text{ where} \tag{1}$$

$D_m(t)$ is the smallest diameter swept by the generating surface of the tool at a given time;

$D_M(t)$ is the largest diameter of the actual pass. It depends on the workpiece geometry and is calculated by the simulator.

The current depth of cut calculated by the simulator as:

$$d(t) = \frac{1}{2}(D_M - D_m). \tag{2}$$

The active cross-section is:

$$A_Q(t) = D_Q \pi.d = \frac{\pi}{4}.(D_M^2 - D_m^2). \tag{3}$$

The feedrate:

$$f(t) = \frac{v_f}{n} \text{ [mm/rev]}. \tag{4}$$

The current (active) cross section of the chip:

$$A_c(t) = d(t).f(t) \text{ [mm}^2\text{]}. \tag{5}$$

The current average thickness of chip is:

$$h_c(t) = \frac{A_c}{l_T}, \text{ where} \tag{6}$$

$l_T(t)$ is the length of tool edge being in cut (dependent to the geometry).

4.2. Kinematical relations. The mean cutting speed can be evaluated as:

$$v(t) = D_Q(t)\pi.n, \text{ where} \tag{7}$$

n [rev/min] is the spindle (rotation) speed.

The feed speed is:

$$v_f = n.f \text{ [mm/min]}, \text{ where} \tag{8}$$

f [mm/rev] is the feedrate.

Here we introduce the most important technical-economic variable, Q the Material Removal Rate in general form.

Definition: In rough cutting operation the technological can be characterised by the material removal rate, which is the product of the active cross section of the cutting process and the feed speed.

The active cross section can be considered as the projection of the surface contacting the tool and the parts per one rotation of the main spindle.

$$Q = A_Q.v_f \ [\text{cm}^3/\text{min}] \tag{9}$$

In case of turning:

$$Q = D_Q.\pi.d.v_f = D_Q.\pi.d.n.f = v.f.d. \tag{10}$$

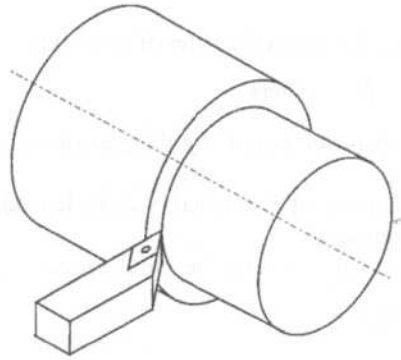


Fig. 3: Active cross section o outer turning operation

4.3. Dynamic relations The mean cutting force can be calculated as:

$$F_y(t) = k_q(h_c, \text{workpiece material}).A_c(t) \ [\text{N}], \text{ where} \tag{11}$$

k_q is the unit force. In industrial practice empirical relations are very popular:

$$F_y(t) = C_F(\text{workpiece material}).v^{y_F}.f^{x_F}.d.\prod_i K_i, \text{ where} \tag{12}$$

K_i is the cutting coefficients describing the lubrication, rough material, clamping, etc.

The force component at the feed direction is:

$$F_x = \lambda_x(\text{cutter angles}).F_y \ [\text{N}] \tag{13}$$

The orthogonal force is:

$$F_z = \lambda_z(\text{cutter angles}).F_y \ [\text{N}], \text{ where} \tag{14}$$

λ_x and λ_z are experimented parameters and dependent on tool geometry. The spindle torque is:

$$M(t) = \frac{1}{2} D_Q.F_y.10^{-3} \ [\text{Nm}], \tag{15}$$

and the cutting power is:

$$P(t) = M.\frac{2\pi}{60}n \text{ [Nm/s].} \tag{16}$$

4.4. Technological relations. The technological relations can be modelled using empirics. The most important state variable is the tool life. In stationary cutting the Taylor equation is the most applicable if the cutting parameters fall into certain range. In non-stationary cutting a load-dependent linear model based on experiments can be used. This model uses the tool wear speed as state variable (v_Δ) which dependent on the tool material and load. To describe the load of coated inserts the following state variable can be used:

$$T^m = \frac{C_v}{d^{x_v} f^{y_v} v}. \tag{17}$$

$$L_T = (d^{x_v} \quad f^{y_v} \cdot v)^q \tag{18}$$

$$q = 1/m \approx 4. \tag{19}$$

According to the model:

$$v_\Delta = k_\Delta(\text{tool} \quad \text{material}).L_T \text{ [mm/min], where} \tag{20}$$

$$k_\Delta = \frac{1}{C_v^q} \tag{21}$$

$$\delta(t) = \frac{\Delta(t)}{\Delta_{ref}}, \quad 0 \le \delta \le 1, \quad \Delta(t) = \Delta_0 + \int_0^t v_\Delta(t)\ dt, \tag{22}$$

This model uses the cumulative wear theory, which gives the Taylor equation in a stationary case. The statistical modelling is also feasible when the k_Δ variable has an exponential (or other) distribution. The tendency to self-induced oscillation is also belongs to the technological modelling. It is dependent on the workpiece and tool geometry and the characteristics of the machine-clamping-workpiece-tool system, which is regarded as a flexible mechanical system. The possible methods for modelling this can be as follows:

Setting up a dynamic model at simulation time based on measurements.
Transferring the stability factor into the model.
Using Neural Networks estimation.

The cutting energy consumption is

$$E_c(t_c) = \int_0^{t_c} P(t)\ dt \tag{23}$$

The cutting time is:

$$t_c = \int_0^s \frac{ds}{v_f(ds)} \text{ where} \tag{24}$$

d_s: cutter path incremental length.

The modelling of the average surface roughness ($\overline{R}_a$), the dimensional accuracy ($\overline{\delta}_m$), the geometrical trueness ($\overline{\delta}_a$) and the rate of waste products (p_W) is extremely difficult. Using AI methods based on the measured data on existing machines could provide usefull models.

4.5. Technical-economic relations. Some technical-economic state variables, and their integrated or average values are required to assess turning processes. The operation time can be evaluated for each operation element as:

$$t_m = t_c + t_r + t_i \text{, where} \tag{25}$$

t_r is the time spent with rapid feeding , t_c: is the cutting (feed) time and t_i is the time consumed without movements (e.g. insert replacement or tool change time). The operation times are easy to calculate using the NC program as input. The expected cost of an operation element can be calculated with special regard to the circumstances, i.e. based on the operational data and conventions of the given firm. The cost can be evaluated as:

$$C_\Sigma = C_m + C_t = c_w t_m + \frac{t_m}{T}(c_w t_{ch} + C_T) \text{, where} \tag{26}$$

c_w is the cost of one work minute, C_T is cost of tool insert replacement, t_{ch} is the tool change time, C_t is the portion of tool insert replacement cost per operation element.

The average Material Removal Rate (often regarded as technological intensity):

$$\overline{Q} = \frac{V}{t_c} \text{, where} \tag{27}$$

V is the material volume to be removed [cm^3].

The optimal MRR can be calculated considering the technological constraints. According to the model described in [7]:

$$\tau = \frac{C_\Sigma}{V.c_w} = \frac{1}{\overline{Q}} + \frac{\overline{Q}^{q-1}}{R^q} \tag{28}$$

$$R = \frac{d^{1-x_v}.f^{1-y_v}C_v}{t_H^m} = \frac{Q.T^m}{t_H^m} \tag{29}$$

$$t_H = \frac{C_t}{c_w} + t_{ch}. \tag{30}$$

The technological intensity optimised for cost when the depth of cut is constant and a force (feedrate) limit R_M is given can be calculated as

$$Q^* = \frac{R_M}{(q-1)^m} \tag{31}$$

$$R_M = \frac{d^{1-x_v} \cdot f_{\max}^{1-y_v} \cdot C_v}{t_H} \tag{32}$$

(Here we assumed that the active constraint is the feed limit). We can define the efficiency of technological intensity:

$$\eta = \frac{\overline{Q}}{Q^*} \text{,where} \tag{33}$$

$$Q_m \le \overline{Q} \le Q_M. \tag{34}$$

The result should be compared with the average technological intensity coded into the NC program. The maximum intensity (Q_M) which is still feasible can be derived from the fact that the tool life of the current tool cannot be smaller than the operation time of the operation elements. To save (tool) cost, the technological intensity can be reduced when the

capacity of the machine tool allows it. The minimum intensity can be derived from the τ_{Max} value as:

$$Q_M = \frac{V}{T} = V^{\frac{m}{m-1}} \cdot \frac{1}{R_M^{\frac{1}{m-1}} \cdot t_H^{\frac{m}{1-m}}} \tag{35}$$

$$Q_m \cong \frac{1}{\tau_{Max}} \tag{36}$$

$$t_{c,\min} = \frac{V}{Q_M} \tag{37}$$

$$t_{c,Max} = V.\tau_{Max} . \tag{38}$$

Having knowledge about the boundaries of Q makes it possible to measure the current and maximum values of cost reserve, minimum operation element time, time reserve and the related MRR, efficiency of the synchronisation of tool life. If the minimum value of tool life is prescribed or the power of the machine is limited then the current $\overline{Q}$ value must be compared to the $Q_{T,Max}$ and $Q_{P,Max}$. This can be achieved using the model discussed above.

This model is easy to aggregate with the upper levels of modelling. Aggregation is referred as a complex modelling function which based on the data of lower levels creates the upper level model parameters. The aggregation of operation elements is an additive function in the aspect of time and cost.

In the direction of lower levels certain decomposition is required, which should be carefully considered due to the non-linear internal relationships of the model. It is difficult to predict the accuracy of the model as a whole. A significant factor is the measurability of the sub-models. The final validation must be accomplished by laboratory experiments.

5. Hierarchical layout for extended simulation software

The hierarchical system layout for extended simulation software is shown in Figure 4 This architecture allows mixed models to be used. At the first layer some basic system variables are evaluated. As the simulator software maintains the actual parameters the evaluation of these is based on the mathematical relations (1)-(5).

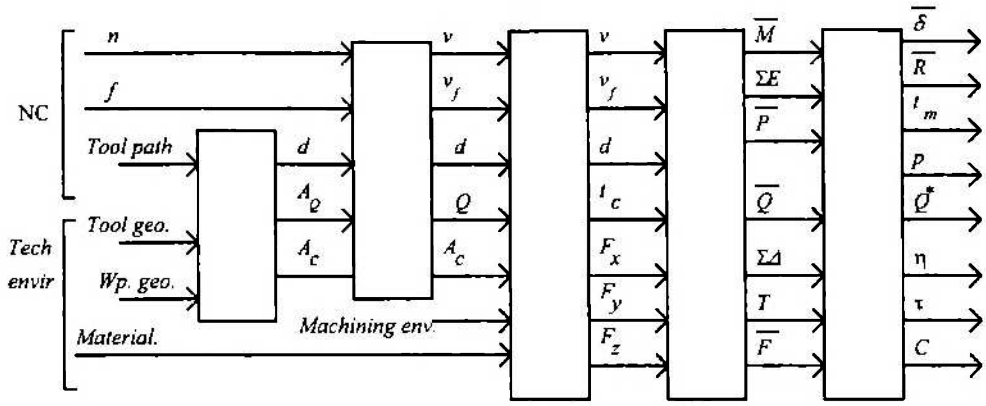


Fig. 4: Chain layout of sub-models

5.1. Cutting force modelling by means of ANN. At the second layer the MRR and the cutting and other additional times are calculated. At this level a backpropagation (BP) feed forward neural network is used to predict the cutting force.

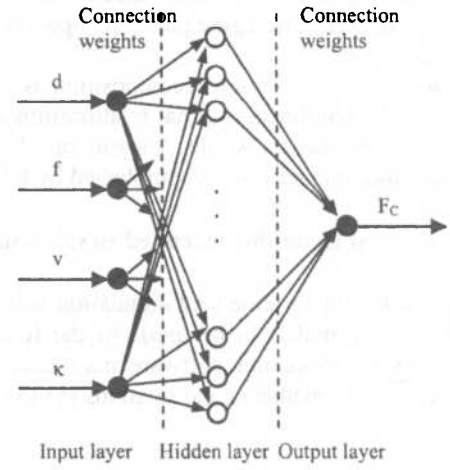


Fig. 5: Single hidden layer ANN for cutting force prediction

The BP network is a supervised continuous valued network. The scaled conjugate gradient (SCG) algorithm had been used to train the network. Compared with the basic BP algorithm, which alters the weights in the steepest descent direction (i.e. the direction in which the performance function is decreasing most rapidly) the SCG algorithm provides faster convergence. [8]. A single hidden layer ANN had been used having 30 neurons. At the input layer the hyperbolic sigmoid transfer function had been used while on the output layer a pure linear transfer function had been applied. The training set was a set of 75 input

samples generated as it is described in [9]. Figures 6 (a) and (b) show the testing accuracy of the Neural Network model.

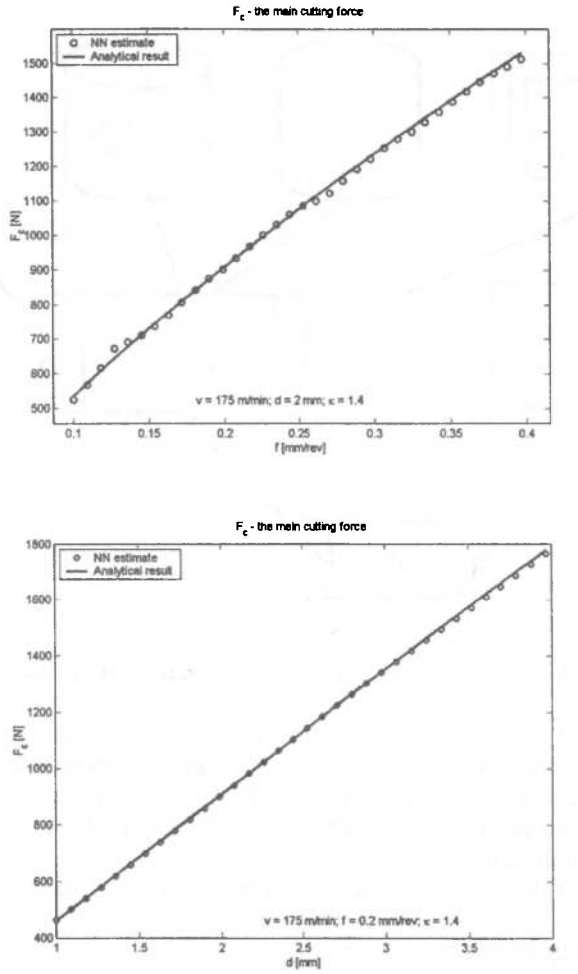


Fig. 6 (a) (b): Testing stage of ANN performance for cutting force

It is well known that the *training session* having multiple epochs of the ANN requires much more computing effort that that of single pass *simulating* the network. Thus it is expedient to separate the training from the simulation. This allows a client-server ANN approach where a server application performs the training sessions and stores the network layout (i.e. number of input, output neurons, number of hidden layers with the number of neurons it has, the calculated weights for each connections and biases of each neuron and the transfer function for each layer) and the client (i.e. the NC simulator) implements only

the propagation passes. This requires an Application Program Interface (API) between the client and server application to be implemented as shown in Figure 7.

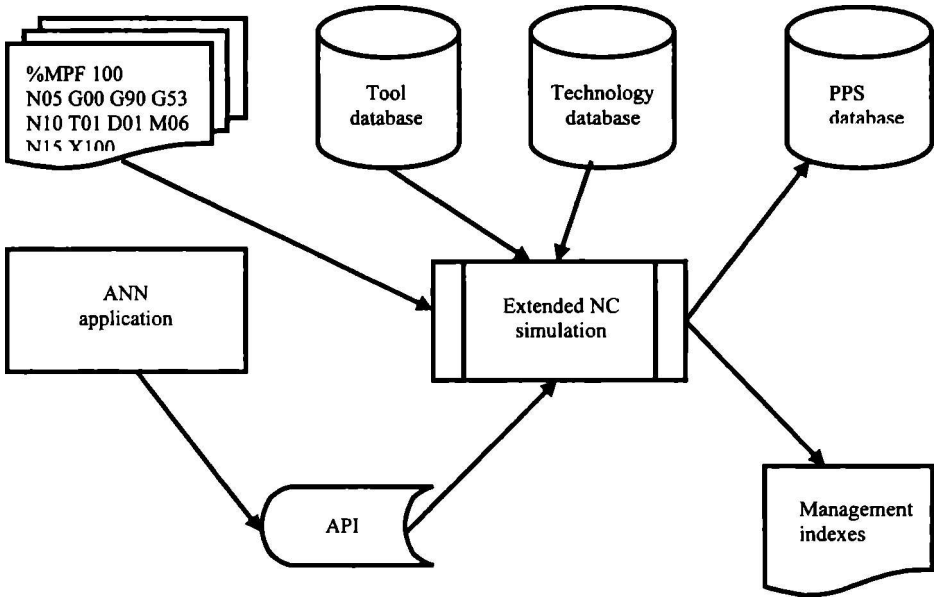


Fig. 7: Architecture of extended simulator software

5.2. Subsequent models. The subsequent models can go on the cutting force model. The required power, the total energy consumption an the torque can be evaluated according to the equation (15) and (16) respectively. The cutting time is evaluated as (24). According to the tool life constraint the time spent with cutting for certain tools must be les than the allowed time limit of the cutter.

The modelling of the workpiece quality can be again an AI model where the relevant input variables are:

the cutting force,
the torque,
the time spent with cutting so far ,
cutting speed,
federate,
cutting edge radius.

5.3. Software engineering approach. The advantage of this chain layout is that the sub models can be implemented in separated classes or components utilizing the advantages of Object Oriented Programming (OOP). This also allows a mixture of the sub models (analytical, numerical, AI-based) to be used. However the major disadvantage is that any error introduced at the beginning of the chain will be rolled over the whole system.

6. Conclusions

Creating alternative NC programs by varying the technological parameters can be an essential tool for modern production management. The extended NC simulators over geometric simulation could provide the required data for supporting production planning and MES systems. This could be a step for realising Virtual Manufacturing in the field of NC turning.

7. Acknowledgements

The research summarized in the paper has been carried-out within the framework of the project entitled "Object Oriented Modelling of Manufacturing Processes" (Id. no.: 0275, headed by F. Erdélyi) supported by Research and Development Competition for Higher Education in Hungary (FKFP). The financial support of the research is gratefully acknowledged.

REFERENCES

1. BALI, J.: *Forgácsolás*. Tankönyv. Tankönyvkiadó, Budapest, 1988.
2. ERDÉLYI, F., HORNYÁK, O.: *Overview of the Possibilities of NC Simulation of CAM*, MicroCAD '01, International Conference, 2001, pp. 18-24.
3. ERDÉLYI, F., HORNYÁK, O.: *NC Program Simulation with the Capability of Generating Alternative Process Plan for Flexible Manufacturing*, 11th PROLAMAT 2001 Conference on Digital Enterprise, Budapest, Nov. 7-10, 2001. Kluwer Academic Publishers, 2001, pp. 43-50.
4. VAN LUTTERVEIT, C. A., CHILDS, T. H. C., JAWAHIR, I. S., KLOCKEM F. VENUVINOD, P. K.: *Present Situation and Future Trends in Modelling of Machining Operations*, CIRP keynote papers, 1998, pp. 1-42.
5. EL MARAGHY, H. A: *Evolution and Future Perspectives of CAPP*, Annals of the CIRP, Vol.42/2, 1993, pp.739-751.
6. TÓTH, T.: *Automatizált műszaki tervezés a gépgyártástechnológiában*, Akadémiai doktori értekezés [Computer Aided Design and Planning in Production Engineering].(in Hungarian), DSc Dissertation, Hungarian Academy of Sciences, Budapest, 1988.
7. TÓTH, T., ERDÉLYI, F.: *The Role of Optimization and Robustness in Planning and Control of Discrete Manufacturing Processes*. Proceedings of the 2nd World Congress on Intelligent Manufacturing Processes & Systems. Springer, Budapest, Hungary, 1997, pp. 205-210.
8. MOLLER, M. F.: *A scaled conjugate gradient algorithm for fast supervised learning*, Neural Networks, Vol. 6, 1993, pp. 525-533.
9. VIHAROS, ZS. J., MONOSTORI, L: *Automatic input-output configuration of ANN-based process models and its application in machining*. In: Lecture Notes of Artificial Intelligence - Multiple Approaches to Intelligent Systems, Conference, Cairo, Egypt, May 31-June 3, 1999. Springer Computer Science Book, Springer-Verlag Heidelberg, 1999, pp. 659-668.

Production Systems and Information Engineering, Miskolc, Vol. 1. (2003), pp. 55-69

A MULTIPLE (EXTENDED) APPLICATION OF THE JOHNSON ALGORITHM FOR THE TWO-MACHINE MANUFACTURING CELL SCHEDULING BASED ON GROUP TECHNOLOGY

SÁNDOR RADELECZKI*
Institute of Mathematics, University of Miskolc
H-3515 MISKOLC, Hungary
matradi@uni-miskolc.hu

TIBOR TÓTH*
Department of Information Engineering, Institute of Information Science,
University of Miskolc
H-3515 MISKOLC, Hungary
toth@iit.uni-miskolc.hu

JÁNOS GÖNDRI-NAGY*
Department of Information Engineering, Institute of Information Science,
University of Miskolc
H-3515 MISKOLC, Hungary
iitgnagy@uni-miskolc.hu

[Received September 2002 and accepted March 2003]

* Production Information Engineering Research Team (PIERT) of the Hungarian Academy of Sciences; Department of Information Engineering of the University of Miskolc

Abstract. As is known, the *Johnson* algorithm is an exact solving method of the two-machine, one-way, no-passing scheduling tasks [1], [6], which serves as a basis for many heuristic algorithms. This paper presents the extension of Johnson's algorithm for Group Technology (GT). The task is as follows: Let us assume that in a two-machine manufacturing cell, in which two machines (A and B) of high automation and environment degree are working together in such a way that machine A is always ready to perform jobs, and machine B is working or waiting according to the timing of the work-pieces transferred from machine A to machine B; the work-pieces are arranged in groups $G_1, G_2, ..., G_i, ..., G_m$ Because of the similarities of the work-pieces in group G_i, retooling and other setups (e. g.: change of equipment) of the machines in the manufacturing cell are not necessary. Consecutive machining of the groups G_i and G_j requires retooling and other setups in the manufacturing cell. An ordering of the groups is to be determined considering all the groups so that the sum of the setup times is to be minimum for all groups.
In the paper the authors prove that the solving of this task can be traced back to the extended application of the Johnson algorithm and results an exact, closed-form optimum.
Keywords: Johnson algorithm, scheduling, Group Technology, Manufacturing Cell

1. Base Model of the Two-Machine, One-Way, No-Passing Scheduling Task

The Johnson algorithm enables the solution of sequencing tasks (here: one-way, no-passing), in which n different jobs are to be allocated to *two* consecutive workplaces (machines, equipment, $m = 2$)[7]. In certain special cases, the task can be extended also to $m = 3$. Although it cannot be used in case of $m > 3$, yet it is an important procedure, because it constitutes the basis of the heuristic methods developed for large-sized tasks. Let us consider Figure 1:

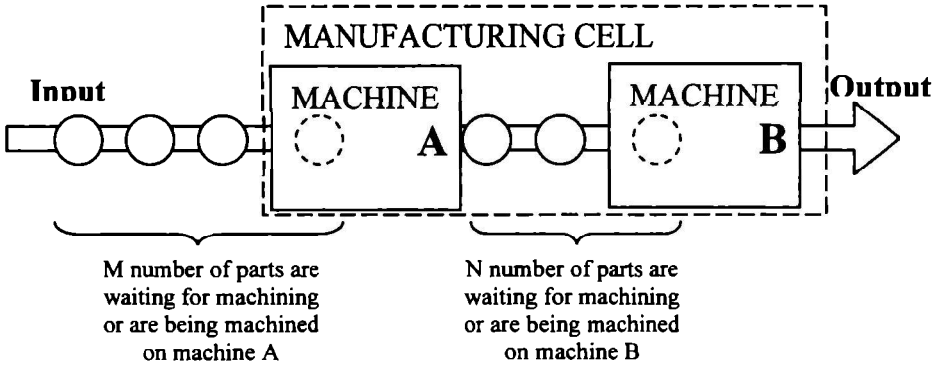


Fig. 1: Model of the two-machine manufacturing cell for deducing Johnson's algorithm

In Figure 1 a manufacturing cell consisting of two machines is demonstrated, on which the machining processes, in accordance with the machines are as follows:

$A \rightarrow$ rough boring,

$B \rightarrow$ finish boring.

It is assumed that dividing the bore machining into two phases using two machines (with different accuracy) is reasonable because of the tolerances.

The two different machining operations are to be carried out on n different parts whose characteristics are similar but their dimensions can significantly differ from one another; in addition, their arrival sequence is optional. Consequently, the throughput time for a series optional but fixed after selection can significantly differ from another variant of the same series manufactured in a different sequence.

Let us denote the time needed for machining the ith part on machine A and B with A_i and B_i, respectively. *The task is to minimize the idle time of machine B.* ("Idle time" stands for the time that elapses between the completion of job p_{i-1} and the start of job p_i.) That is, we want to determine a sequence $p_1, p_2, ..., p_i, ..., p_n$, for which the sum of idle times between finish boring parts p_j and p_{j+1} will be minimum, computing the sum for consecutive j values.

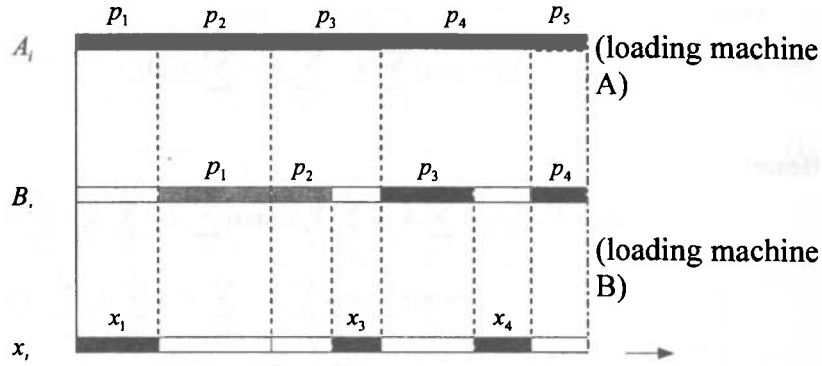


Fig. 2: Default Gantt chart for the two-machine cell model

Let us denote the time that elapses between the start of rough boring the first part and the completion of finish boring the last part by T. Let X_i be the idle time between the completion of job p_{i-1} and the start of job p_i. According to Figure 2, we can write:

$$T = \sum_{i=1}^{n} B_i + \sum_{i=1}^{n} X_i ,$$

For $\sum_i B_i$ is given and known, only $\sum_i X_i$ is to be minimized.

From Figure 2 it can be seen that:

$$X_1 = A_1 , \quad X_2 = 0, \qquad \text{if } A_1 + A_2 < B_1 + X_1 ,$$
$$X_2 = A_1 + A_2 - B_1 - X_1 , \quad \text{if } A_1 + A_2 > B_1 + X_1 .$$

(The equality sign is taken into consideration in the second case, because the transition times between the machines to a first approximation are ignored.) Therefore, such an X_2 is to be determined, for which the following applies:

$$X_2 = \max(A_1 + A_2 - B_1 - X_1, 0) = \max(\sum_{i=1}^{2} A_i - \sum_{i=1}^{1} B_i - \sum_{i=1}^{1} X_i, 0).$$

Let us examine the sum $X_1 + X_2$. We can write that

$$X_1 + X_2 = X_1 + \max(A_1 + A_2 - B_1 - X_1, 0) =$$
$$= \max(A_1 + A_2 - B_1, X_1) = \max(A_1 + A_2 - B_1, A_1) =$$
$$= \max(\sum_{i=1}^{2} A_i - \sum_{i=1}^{1} B_i, A_1).$$

Similarly:

$$X_3 = \max(\sum_{i=1}^{3} A_i - \sum_{i=1}^{2} B_i - \sum_{i=1}^{2} X_i, 0).$$

Hence:

$$X_1 + X_2 + X_3 = \sum_{i=1}^{3} X_i = \sum_{i=1}^{2} X_i + \max(\sum_{i=1}^{3} A_i - \sum_{i=1}^{2} B_i - \sum_{i=1}^{2} X_i, 0) =$$

$$= \max(\sum_{i=1}^{3} A_i - \sum_{i=1}^{2} B_i - \sum_{i=1}^{2} X_i + \sum_{i=1}^{2} X_i, \sum_{i=1}^{2} X_i) =$$

$$= \max(\sum_{i=1}^{3} A_i - \sum_{i=1}^{2} B_i, \sum_{i=1}^{2} X_i) =$$

$$= \max(\sum_{i=1}^{3} A_i - \sum_{i=1}^{2} B_i, \sum_{i=1}^{2} A_i - B_1, A_1).$$

This formula can easily be extended for n-number idle times assuming a certain sequence $(S) = (p_1, p_2, \ldots, p_i, \ldots, p_n)$:

$$D_n(S) = \sum_{i=1}^{n} X_i = \max\left[\sum_{i=1}^{n} A_i - \sum_{i=1}^{n-1} B_i, \sum_{i=1}^{n-1} A_i - \sum_{i=1}^{n-2} B_i, \ldots, A_1\right].$$

This formula can be written in a more concise form in the following manner:

$$D_n(S) = \max_{1 \le r \le n}\left[\sum_{i=1}^{r} A_i - \sum_{i=1}^{r-1} B_i\right]$$

or in another way:

$$D_n(S) = \max_{1 \le r \le n} L_r, \quad \text{where} \quad L_r = \sum_{i=1}^{r} A_i - \sum_{i=1}^{r-1} B_i$$

Let us have some kind of series $(S^{(1)})$:

$$(S^{(1)}) = (p_1, p_2, \ldots, p_{k-1}, p_k, p_{k+1}, p_{k+2}, \ldots, p_n)$$

and a series $(S^{(2)})$, that can be obtained from $(S^{(1)})$ exchanging k and $k+1$ with each other:

$$(S^{(2)}) = (p_1, p_2, \ldots, p_{k-1}, p_{k+1}, p_k, p_{k+2}, \ldots, p_n),$$

and let us define the sums $L_r^{(1)}$ and $L_r^{(2)}$ for the first r members of $(S^{(1)})$ and $(S^{(2)})$ similarly.

It is easy to see that $L_r^{(1)}$ and $L_r^{(2)}$ are the same for all $1 \le r \le n$ in the cases of series $(S^{(1)})$ and $(S^{(2)})$, except maybe the cases of $r = k$ and $r = k+1$.

Hence $D_n(S^{(1)}) = D_n(S^{(2)})$, whenever $\max(L_k^{(1)}, L_{k+1}^{(1)}) = \max(L_k^{(2)}, L_{k+1}^{(2)})$.

If $\max(L_k^{(1)}, L_{k+1}^{(1)}) \neq \max(L_k^{(2)}, L_{k+1}^{(2)})$, then one of the two series $(S^{(1)})$ and $(S^{(2)})$ is more advantageous than the other. Series $(S^{(1)})$ – in which $k+1$ follows k – is more advantageous than series (S^2), in which $k+1$ *precedes* k, if

$$\max(L_k^{(1)}, L_{k+1}^{(1)}) < \max(L_k^{(2)}, L_{k+1}^{(2)}). \tag{1}$$

In detail:

$$\max(L_k^{(1)}, L_{k+1}^{(1)}) = \max(\sum_{i=1}^{k} A_i - \sum_{i=1}^{k-1} B_i, \sum_{i=1}^{k+1} A_i - \sum_{i=1}^{k} B_i),$$

$$\max(L_k^{(2)}, L_{k+1}^{(2)}) = \max\left[\sum_{i=1}^{k-1} A_i + A_{k+1} - \sum_{i=1}^{k-1} B_i, \sum_{i=1}^{k+1} A_i - \sum_{i=1}^{k-1} B_i - B_{k+1}\right]$$

Hence, we obtain:

$$\sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k+1} A_i + \max(L_k^1, L_{k+1}^1) =$$

$$= \max\left[\sum_{i=1}^{k} A_i - \sum_{i=1}^{k+1} A_i + \sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k-1} B_i, \sum_{i=1}^{k+1} A_i - \sum_{i=1}^{k+1} A_i + \sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k} B_i\right] =$$

$$= \max(-A_{k+1}, -B_k) = -\min(A_{k+1}, B_k).$$

And similarly:

$$\sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k+1} A_i + \max(L_k^2, L_{k+1}^2) =$$

$$= \max\left[\sum_{i=1}^{k-1} A_i + A_{k+1} - \sum_{i=1}^{k+1} A_i - \sum_{i=1}^{k-1} B_i + \sum_{i=1}^{k-1} B_i, \sum_{i=1}^{k+1} A_i - \sum_{i=1}^{k+1} A_i + \sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k-1} B_i - B_{k+1}\right] =$$

$$= \max\left[\sum_{i=1}^{k-1} A_i + A_{k+1} - (\sum_{i=1}^{k-1} A_i + A_k + A_{k+1}), -B_{k+1}\right] =$$

$$= \max(-A_k, -B_{k+1}) = -\min(A_k, B_{k+1}).$$

Hence, relation (1) is equivalent to the following form:

$$-\min(A_{k+1}, B_k) < -\min(A_k, B_{k+1}),$$

that is, to:

$$\min(A_{k+1}, B_k) > \min(A_k, B_{k+1})$$

Now, we can draw the conclusion that the sequence $(..., p_k, p_{k+1}, ...)$ is more advantageous than the sequence $(..., p_{k+1}, p_k, ...)$ if:

$$\min(A_k, B_{k+1}) < \min(A_{k+1}, B_k) \quad . \tag{2}$$

Let $1 \le k < l \le n$ and let us consider now the sequence $(S') = (p_1, p_2, ..., p_l, ..., p_k, ..., p_n)$ different from *(S)* only in the fact that in (S') the job p_l is staying in the position number k, and it is the job p_k in the position k.

The sequence *(S)* is more advantageous than (S'), if and only if

$$\min(A_k, B_l) \le \min(A_l, B_k), \tag{3}$$

which holds either $A_k \le B_l$ & $A_k \le A_l$ & $A_k \le B_k$ is true,

or $B_l \le A_k$ & $B_l \le A_l$ & $B_l \le B_k$ is true.

The first case can be expressed also in the form:

$$\min(A_k, B_k) \le \min(A_l, B_l). \tag{4}$$

Therefore, if we find a time A_k in the table of times which is less than all other A_l and B_l at the same time, then it has to be begun the sequence with p_k. If time A_k – although is one of the smallest times – is equal to another A_l or B_l, the sequence can also be begun with p_k.

The second case is equivalent to:

$$\min(A_l, B_l) \le \min(A_k, B_k). \tag{4'}$$

Consequently, if we find a time B_l in the table of times, that is less than all other A_k or B_k at the same time, then the sequence to be determined must be ended with p_l. If time B_l although is one of the shortest times – is equal to another A_k or B_k, the sequence can also be ended with it.

It can be seen from the detailed deduction, that the sequence can be determined step-by-step by means of the *Johnson algorithm*. In Operations Research, the mathematical method that optimizes a series of decisions depending on one another is named *Dynamic Programming*. *Johnson's* algorithm solves actually a dynamic programming problem.

Extension of the Johnson algorithm for three machines

The *Johnson* algorithm can also be used in the following two special cases:

$$\min A_i \ge \max B_i \quad \text{or} \quad \min C_i \ge \max B_i,$$

if the three machines *A, B* and *C* with *n* jobs to be done are given. In such a case the examination of the times is executed with the sums $A_i + B_i$ and $B_i + C_i$. These "virtual machining times" are handled exactly as real machining times of two fictive machines [7].

Let us consider the following task:

The turning, milling and grinding operations are defined by periods A_i, B_i and C_i for the parts denoted by $p_1, ..., p_5$.

Let us start from the following table:

	Turn (A_i)	Mill (B_i)	Grind (C_i)
p_1	8	6	7
p_2	12	3	10
p_3	9	5	4
p_4	15	4	18
p_5	11	5	10

Table I

It is valid that $min\ A_i \geq max\ B_i$, because: $8 > 6$. We can compile the second table, too:

	$A_i + B_i$	$B_i + C_i$
p_1	14	13
p_2	15	13
p_3	14	9
p_4	19	22
p_5	16	15

Table II

Applying the Johnson algorithm, we get:

1.

p_1	14	13
p_2	15	13
p_4	19	22
p_5	16	15
p_3	14	9

2.

p_1	14	13
p_4	19	22
p_5	16	15
p_2	15	13
p_3	14	9

3.

p_4	19	22
p_5	16	15
p_1	14	13
p_2	15	13
p_3	14	9

The algorithm can be well followed through Table II, 1. 2. and 3.

1. We examine, which is the smallest time in the Table II; this time, it is 9 minutes.
2. If this value belongs to the *first* column of the table, then we start with the part corresponding to it.; if it belongs to the *second* column, then we end machining with the part corresponding to it.
3. We separate the corresponding row of the table, and we follow the steps 1.-2.-3. with the remaining part of the table. For the optimal scheduling sequence we get, that:

$$S = (p_4, p_5, p_1, p_2, p_3).$$

Let us take the original *ad hoc* sequence:

$$S_1 = (p_1, p_2, p_3, p_4, p_5).$$

Representing it (Figure 3):

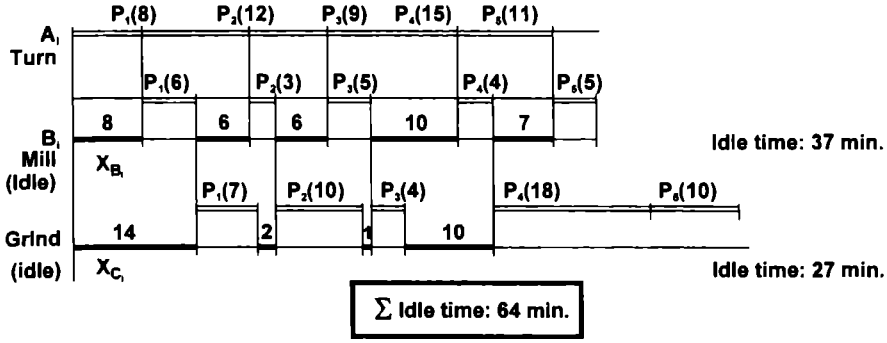


Fig. 3: Demonstrating the extension of the Johnson algorithm through a concrete example (Gantt chart, *ad hoc* sequence)

According to the *Johnson* algorithm (Figure 4):

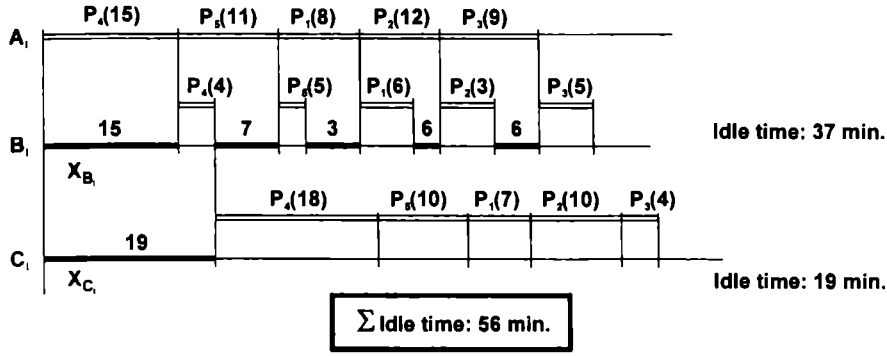


Fig. 4: Demonstrating the extension of Johnson's algorithm through a concrete example (Gantt chart, *optimal* sequence)

From the two Gantt chart it can be seen, that the Johnson algorithm decreases the idle times by 8 minutes.

2. Modification of the Base Model Considering Retooling (Setup) Times

Let $p_1, p_2, ..., p_n$ be the pieces to be machined on machine A and B, and let us denote their machining times by $A_1, A_2, ..., A_n$ and $B_1, B_2, ..., B_n$, respectively [5]. Let $x_1, x_2, ..., x_n$ denote the idle times during (or rather prior to) the machining of pieces $p_1, p_2, ..., p_n$ on machine B. Let $\pi = (i_1, i_2, \quad i_n)$ denote an arbitrary permutation of the indexes $(1, 2, \quad n)$ and let ρ_n stand for the set of all permutations of the index set $\{1, 2, ..., n\}$.

It is known, that in case of a $p_{i_1}, p_{i_2}, ..., p_{i_n}$ permutation of the pieces, idle times $x_{i_1}^{(\pi)}, x_{i_2}^{(\pi)}, ..., x_{i_n}^{(\pi)}$ that occur on machine B differ – Let us denote the sum of these with X_π.

If the pieces are ordered in the sequence $p_{j_1}, p_{j_2}, ..., p_{j_n}$ that is given by Johnson's algorithm, i.e. we consider the $\pi_J=(j_1, j_2, ..., j_n)$ permutation of the indexes $(1, 2, ..., n)$, then idle time X_{π_J} will be the minimum possible, that is

$$X_{\pi_J} = \min\{X_\pi | \pi \in \rho_n\}.$$

(1) First of all, we would like to show that if machine A does not start to operate at the $t=0$ point of time but at the point of time t (where t can also be negative, which means that machining on machine A had already been started before 0 point of time), the optimal sequence of machining – i.e. the sequence when the sum of idle times x_i on machine B is minimum – remains the one that is determined by the Johnson algorithm, that is the sequence corresponding to the index permutation π_J. If $t > 0$ then this case can also be viewed as if, the machining time of piece p_{i_1} on machine A that stands in the first position would grow from A_{i_1} to $A_{i_1} + t$, for the machining sequence $p_{i_1}, p_{i_2}, ..., p_{i_n}$ given by the permutation $\pi=(i_1, i_2, ..., i_n)$, and the machining time on machine B would remain B_{i_1}

As is known, the idle time (in case of starting at the point of time 0) for permutation π was so far given by the formula $X_\pi = \max_{1 \le r \le n} \left(\sum_{j=1}^{r} A_{i_j} - \sum_{j=1}^{r-1} B_{i_j} \right)$, now we get the following:

$$X'_\pi = \max_{1 \le r \le n} \left(\sum_{j=1}^{r} A_{i_j} + t - \sum_{j=1}^{r-1} B_{i_j} \right) = t + \max_{1 \le r \le n} \left(\sum_{j=1}^{r} A_{i_j} - \sum_{j=1}^{r-1} B_{i_j} \right) = X_\pi + t$$

Now:

$$\min\left(X'_\pi | \pi \in \rho_n\right) = \min\left(t + X_\pi | \pi \in \rho_n\right) = t + \min\left(X_\pi | \pi \in \rho_n\right).$$

Since the minimum involved in the latter sum turns up just in the permutation relevant to the Johnson algorithm, therefore the idle time X'_π will also be minimum, if $\pi = \pi_J$; *i.e. if the pieces are further ordered according to the Johnson algorithm.* Moreover, the idle time that results in this manner is exactly $t + X_{\pi_J}$ – i.e. it is longer exactly by t time units (than in case of starting at *0* point of time).

If $t<0$, then this situation can also be seized as if we started measuring the time earlier than at *0*, but at t point of time; yet machine B would come into operation only with $|t|$-time later.

And this latter case can be considered, as if the machining time of the first one, part p_{i_1} on machine B increased from B_{i_1} to $B_{i_1} + |t| = B_{i_1}$ for the machining sequence concerning the permutation of the indexes $\pi=(i_1, i_2, \ldots, i_n)$, and the machining time on A did not alter.

Now $\sum_{j=1}^{r-1} B'_{i_j} = B_{i_1} - t + \sum_{j=2}^{r-1} B_{i_j} = \sum_{j=1}^{r-1} B_{i_j} - t$, thus now the total idle time X'_π is as follows:

$$X'_\pi = \max\left(\max_{1\le r\le n}\left(\sum_{j=1}^{r} A_{i_j} - \sum_{j=1}^{r-1} B_{i_j} + t\right);0\right) = \max\left(t + \max_{1\le r\le n}\left(\sum_{j=1}^{r} A_{i_j} - \sum_{j=1}^{r-1} B_{i_j}\right);0\right) = \max(t + X_\pi;0)$$

– considering the fact that the idle time X'_π (for any $t<0$) can only be positive or *0*. Since for the index-permutation π_J determined by the Johnson-algorithm it is true that $X_{\pi_J} \le X_\pi$ for all $\pi \in \rho_n$, therefore $\max(t + X_{\pi_J};0) \le \max(t + X_\pi;0)$, i.e. the Johnson algorithm gives still the least idle time possible, which idle time is $t + X_{\pi_J}$ if this latter number is positive, and *zero* if $t + X_\pi < 0$.

Summing up the $t>0$ and the $t<0$ cases we can say that the Johnson algorithm derives in both cases such a sequence of the pieces, beside which the total idle time is the minimum possible and both cases fit the relation that is follows: $X'_{\pi_J} = \max(X_{\pi_J} + t;0)$.

(2) As an application of the above-mentioned ones, let us consider the situation in which the machining process on machine A and machine B is preceded by the setup times S_A and S_B. This case can also be viewed as if machine A in comparison with machine B were available only in the point of time S_A-S_B. According to the above model, now the time shift on machine A is $t=S_A-S_B$. To this, according to item (1), the optimal sequence of pieces remains the one that is determined by the Johnson algorithm; denoting the given optimal total times with X or in case of considering setup times with X', the relation between them is as follows:

$$X' = \max(X + t;0) = \max(X + S_A - S_B;0).$$

(Note: Here the setup time S_B of machine B is not considered an idle time!)

3. Extension of the Modified Base Model for Group Technology

Hereafter, let us make an attempt at an optimal fitting of piece groups [3], [4] by means of tracing back it to Johnson's algorithm.

It is assumed that it is in groups G_1, G_2, G_m advantageous for the pieces to be machined (on machine A and B) because of some economic requirements, where the setup times S_{A_i} and S_{B_i} on machine A and B belong to each group G_i (that is the setup time of the machines to be prepared for the machining of group G_i). If the parts $p_{i_1}, p_{i_2}, ..., p_{i_{N_i}}$ belong to group G_i, then let us denote their times needed to be machined on machine A and B with $A_{i_1}, A_{i_2}, ..., A_{i_{N_i}}$, and with $B_{i_1}, B_{i_2}, ..., B_{i_{N_i}}$

Moreover, let us introduce the notations $A_i = S_{A_i} + \sum_{j=1}^{N_i} A_{i_j}$ and $B_i = S_{B_i} + \sum_{j=1}^{N_i} B_{i_j}$. According to item (2), the total idle time for group G_i will be minimum, if the pieces i.e. the corresponding number pairs $(A_{i_1}, B_{i_1}), (A_{i_2}, B_{i_2}), ..., (A_{i_{N_i}}, B_{i_{N_i}})$ are ordered in accordance with the Johnson algorithm. Let us denote the so arisen total idle time (at which the setup times have not been considered yet) for group G_i *with* X_i. (i.e. $X_i = \sum_{j=1}^{N_i} x_{i_j}$, if the pieces are ordered in the group according to the Johnson algorithm).

Let us have a look at the total idle time X_i' that is arisen during the machining of the pieces of group G_i, if the group machining is executed in the sequence of G_1, G_2, G_{i-1}, G_i, G_m.

The total idle time for group G_i will namely be influenced by the fact which point of times $T_{A_{i-1}}, T_{B_{i-1}}$ machine A after completing the pieces of the groups G_1, G_2, G_{i-1}, and machine B after completing the pieces of the groups G_1, G_2, G_{i-1} are available at.

Accurately, the reason for the idle time X_i to be altered is that machine A in comparison to machine B is available with a time shift of $t_{i-1} = T_{A_{i-1}} - T_{B_{i-1}}$ Thus, the new idle time X_i' is as follows: $X_i' = \max(X_i + t_{i-1}, 0) = \max\left(X_i' + T_{A_{i-1}} - T_{B_{i-1}}, 0\right)$. The time $T_{A_{i-1}}$ means evidently the sum $A_1+A_2+...+A_{i-1}$ of the times (where $A_k = S_{A_k} + \sum_{j=1}^{N_k} A_{k_j}$ – so it contains the setup times $S_{A_1}, S_{A_2}, ..., S_{A_{i-1}}$ too.), whilst the time $T_{B_{i-1}}$ is the sum of the times $B_1 + X_1' + B_2 + X_2' + ... + B_{i-1} + X_{i-1}'$ (where: $B_k = S_{B_k} + \sum_{j=1}^{N_k} B_{k_j}$), so

$$t_{i-1} = \sum_{k=1}^{i-1} A_k - \sum_{k=1}^{i-1} B_k - \sum_{k=1}^{i-1} X_k' .$$

Let us observe that in case of group G_1, which is preceded by no other groups but the setup times S_{A_1} and S_{B_1}, $T_{A_0} = S_{A_1}$, $T_{B_0} = S_{B_1}$, thus it can be said that $t_0 = S_{A_1} - S_{B_1}$, and for X_1' we get that $X_1' = \max(X_1 + t_0;\ 0) = \max(X_1 + S_{A1} - S_{B1};\ 0)$.

Similarly, $X_2' = \max(X_2 + t_1;\ 0) = \max(X_2 + A_1 - B_1 - X_1';0)$.

Thus $X_1' + X_2' = \max(X_2 + A_1 - B_1;\ X_1') = \max(X_2 + A_1 - B_1, X_1 + S_{A_1} - S_{B_1};\ 0)$

$X_3' = \max(X_2 + t_2;\ 0) = \max(X_3 + (A_1 + A_2) - (B_1 + B_2) - (X_1' + X_2');\ 0)$

Thus

$X_1' + X_2' + X_3' = \max(X_3 + (A_1 + A_2) - (B_1 + B_2);\ X_1' + X_2') =$

$+ \max(X_3 + (A_1 + A_2) - (B_1 + B_2);\ X_2 + A_1 - B_1; X_1 + S_{A_1} - S_{B_1};0).$

In general

$$X_i' = \max(X_i + t_{i-1},0) = \max\left(X_i + \sum_{k=1}^{i-1} A_k - \sum_{k=1}^{i-1} B_k - \sum_{k=1}^{i-1} X_k';\ 0\right)$$

and

$$X_1' + X_2' + ...,+X_i' = \max\left(X_i + \sum_{k=1}^{i-1} A_k - \sum_{k=1}^{i-1} B_k;\ \sum_{k=1}^{i-1} X_k'\right),$$

therefore for the total idle time $X = X_1' + X_2' + ... + X_m'$ the following formula is obtained:

$$X = \max\left(X_m + \sum_{k=1}^{m-1} A_k - \sum_{k=1}^{m-1} B_k;\ X_{m-1} + \sum_{k=1}^{m-2} A_k - \sum_{k=1}^{m-2} B_k;...; X_2 + A_1 - B_1;\ X_1 + S_{A_1} - S_{B_1};\ 0\right)$$

that can be written in a more concise manner as is follows:

$$X = \max_{0 \le r \le m}(T_r), \text{ where } T_0 = 0 \text{ and } T_r = X_r + \sum_{i=1}^{r-1} A_i - \sum_{i=1}^{r-1} B_i$$

(Note: A_0 is regarded as S_{A_1}, and B_0 as S_{B_1}).

Let us consider now the group sequences

S_1: $G_1, G_2, ..., G_k ..., G_l ..., G_m$ and

S_2: $G_1, G_2, ..., G_l ..., G_k ..., G_m$

that differ from each other only in the fact that in the second series it is the group G_l staying in the position number k, and it is the group G_k. in the position number l.

Since in this case the computations can be fulfilled in a very analogous way as in the case when k and l are consecutive numbers, for the sake of clarity we will confine ourselves only to the latter case.

Let the series of groups be the followings:

S_1: $G_1, G_2, \ldots, G_{k-1}, G_k, G_{k+1}, \ldots, G_m$ and

S_2: $G_1, G_2, \ldots, G_{k-1}, G_{k+1}, G_k \ldots, G_m$,

Let us denote the part sums T_r that belong to the first series of groups with T_r^1, and the ones that belong to the second series of groups with T_r^2

Obviously, the machining sequence that is given by series S_1 is more efficient than the machining sequence corresponding to S_2 *if and only if*

$$\max\left(T_k^1, T_{k+1}^1\right) < \max\left(T_k^2, T_{k+1}^2\right). \qquad (*)$$

Now: $$\max\left(T_k^1, T_{k+1}^1\right) = \max\left(X_k + \sum_{i=1}^{k-1} A_i - \sum_{i=1}^{k-1} B_i;\ X_{k+1} + \sum_{i=1}^{k} A_i - \sum_{i=1}^{k} B_i \right),$$

thus: $$\sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k-1} A_i + \max\left(T_k^1, T_{k+1}^1\right) = \max(X_k;\ X_{k+1} + A_k - B_k).$$

On the other hand:

$$\max\left(T_k^2, T_{k+1}^2\right) = \max\left(X_{k+1} + \sum_{i=1}^{k-1} A_i - \sum_{i=1}^{k-1} B_i;\ X_k + \sum_{i=1}^{k-1} A_i + A_{k+1} - \sum_{i=1}^{k-1} B_i - B_{k+1} \right),$$

– taking into consideration that the total idle time during the machining of group G_{k+1} is X_{k+1}, and during the machining of group G_k is X_k.

Thus

$$\sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k-1} A_i + \max\left(T_k^2, T_{k+1}^2\right) = \max\left(X_{k+1};\ X_k + A_{k+1} - B_{k+1}\right).$$

According to the researches, the inequality (*) is equivalent to the inequality as follows:

$$\max(X_k;\ X_{k+1} + A_k - B_k) < \max(X_{k+1};\ X_k + A_{k+1} - B_{k+1})$$

Adding $-X_k - X_{k+1}$ to both side of the inequality, we get

$$\max\left(-X_{k+1};\ A_k - B_k - X_k\right) < \max\left(-X_k;\ A_{k+1} - B_{k+1} - X_{k+1}\right), \text{ i.e.}$$

$$\max\left(-X_{k+1};\ -\left(B_k + X_k - A_k\right)\right) < \max\left(-X_k;\ -\left(B_{k+1} + X_{k+1} - A_{k+1}\right)\right).$$

Since $X_i \geq 0$ for all $i=1,2,\ldots,n$ and since $B_i + X_i - A_i \geq 0$ is true, all four numbers in parentheses are negative or *0*, we can write:

$$-\min\left(X_{k+1};\ B_k + X_k - A_k\right) < -\min\left(X_k;\ B_{k+1} + X_{k+1} - A_{k+1}\right),$$

wherefrom we get that (*) is equivalent to the inequality

$$\min\left(X_{k+1};\ B_k + X_k - A_k\right) > \min\left(X_k;\ B_{k+1} + X_{k+1} - A_{k+1}\right)$$

therefore in this case serial S_1 means a more advantageous sequence than series S_2.

Now returning to the general case when

S_1: $G_1, G_2, \ldots, G_k \ldots, G_l, \ldots, G_m$ and

S_2: $G_1, G_2, \ldots, G_l \ldots, G_k \ldots, G_m$ (where $k<l$),

we would get in analogous way that S_1 is more advantageous than sequence S_2 if and only if

$$\min(X_1, B_k + X_k - A_k) \geq \min(X_k, B_l + X_l - A_l),$$

namely, if

$$\min(X_k;\ B_l + X_l - A_l) \leq \min(X_l;\ B_k + X_k - A_k).$$

Apparently, this same criterion applies to such an abstract Johnson-sequence that consists of the following number pairs:

$$(X_1;\ B_1 + X_1 - A_1), \ldots, (X_k;\ B_k + X_k - A_k), \ldots, (X_m;\ B_m + X_m - A_m).$$

Now in order to trace back the optimal sequencing problem of the groups G_1, G_2, ..., G_k, ..., G_m to the sequencing problem of these number pairs according to the Johnson algorithm, it is sufficient to show that were such fictive parts $F_1, \ldots, F_k, \ldots, F_m$ considered instead of the groups $G_1, \ldots, G_k, \ldots, G_m$, whose work time on machine A are $X_1, \ldots, X_k, \ldots, X_m$, and whose work time counted for machine B are $B_1 + X_1 - A_1, \ldots, B_k + X_k - A_k, \ldots, B_m + X_m - A_m$, then we would obtain the same idle time and the same optimal sequence as in case of the machining of the piece groups $G_1, \ldots, G_k, \ldots, G_m$.

Let us denote the idle times for the fictive serial with $X_1^*, \ldots, X_k^*, \ldots, X_m^*$, and their sums with X^*

According to the theory of Johnson's algorithm:

$$X^* = \sum_{i=1}^{m} X_i^* = \max_{1 \leq r \leq m} L_r, \text{ where } L_r = \sum_{i=1}^{r} X_i - \sum_{i=1}^{r-1}\left((B_i + X_i) - A_i\right) = X_r + \sum_{i=1}^{r-1} A_i - \sum_{i=1}^{r-1} B_i,$$

hence $$X^* = \max_{1 \leq r \leq m}\left(X_r + \sum_{i=1}^{n-1} A_i - \sum_{i=1}^{n-1} B_i\right) = \sum_{i=1}^{m} X_i^* = X^{\pi}$$

Because as for an optional permutation $\pi = (i_1, i_2, \ldots, i_n)$ of the indexes the total idle times of the group series $G_1, G_2, \ldots, G_k \ldots, G_m$ and the fictive series of pieces $F_1, F_2, \ldots, F_n \ldots, F_m$ are the same, i.e. $X_{\pi}^{'} = X_{\pi}^{*}$, therefore it will be the same permutation $i_1, i_2, \ldots, i_m$ of the indexes that will provide the smallest total idle time for the series $G_1, G_2, \ldots, G_k \ldots, G_m$, as for the abstract series $F_1, F_2, \ldots, F_k \ldots, F_m$; and this is exactly the same that we have obtained for the latter series by means of the Johnson algorithm.

Therefore, first and last, it can be said that the group series G_1, G_2, ..., G_m can truly be substituted for the fictive piece series $F_1, F_2, \ldots, F_m$, where the machining time of an F_k "piece" on machine A is treated as X_k, and its machining time on machine B is treated as $B_k + X_k - A_k$.

Purely and simply, these quantities can be explicated as:

$$X_k = \sum_{j=1}^{N_k} x_{k_j} \text{ and } B_k + X_k - A_k = S_{B_k} + \sum_{j=1}^{N_k} B_{k_j} + \sum_{j=1}^{N_k} X_{k_j} - \left(S_{A_k} + \sum_{j=1}^{N_k} A_{k_j} \right).$$

Acknowledgements: The work this paper is based on has been supported by the OTKA (Hungarian Scientific Research Fund) project entitled ***„Application of Concept Lattices and Fuzzy Methods in Group Technology"*** (No.: T030243, headed by Tóth, T.). In addition, a part of this research work has been done within the framework of ***Production Information Engineering Research Team*** (PIERT) established at the Department of Information Engineering by the Hungarian Academy of Sciences in 1999. The financial support of the research by the two sources is gratefully acknowledged. Moreover, the support provided by the OTKA (Grant No. T034137) is appreciatively acknowledged.

REFERENCES

1. BRUCKER, P.: *Scheduling Algorithms,* Springer-Verlag Berlin, Heidelberg, 1998.
2. CORMEN, T. H.; LEISERSON, C E.; RIVEST, R. L.: *Introduction to Algorithms,* MIT Press, London, 2001.
3. KAMRANI, A. K.; LOGENDRAN, R.: *Group Technology and Cellular Manufacturing: Methodologies and Applications,* Gordon and Breach Science Publishers, Amsterdam, 1998.
4. NC/CIM GUIDEBOOK (1998), MODERN MACHINE SHOP: *Group Technology and Manufacturing Cells*, pp. 230-242.
5. RADELECZKI, S., and TÓTH, T.: *Connection of Concept Lattices and Fuzzy Methods and Their Application in Group Technology* (in Hungarian), Research Report. University of Miskolc, 1999.
6. ROSEN, K. H.: *Discrete Mathematics and Its Applications,* McGraw-Hill, New York, 1988.
7. TÓTH, T.: *Design and Planning Principles, Models and Methods in Computer Integrated Manufacturing* (in Hungarian), Publisher of the University of Miskolc, 1998.

Production Systems and Information Engineering, Miskolc, Vol. 1. (2003), pp. 71-81

COMPUTERIZED SIMILARITY ANALYSIS OF PARTS FOR SUPPORTING GROUP TECHNOLOGY

TIBOR TÓTH*
Department of Information Engineering, University of Miskolc
H-3515. MISKOLC- Hungary
toth@iit.uni-miskolc.hu

MIHÁLY MOLNÁR*
Department of Information Engineering, University of Miskolc
H-3515. MISKOLC- Hungary
molnar@iit.uni-miskolc.hu

[Received March and accepted October 2002]

* Production Information Engineering Research Team (PIERT) of the Hungarian Academy of Sciences; Department of Information Engineering of the University of Miskolc

Abstract. Group Technology (GT) is a comprehensive organizing principle of part manufacturing which can be utilized both in design and technology processes of parts as well as in the course of real manufacturing processes, too. The base of GT is *similarity* of the most important macro-geometrical and technological features of parts. The paper deals with grouping the parts according to similarity which is the first major activity area indispensable for initiating GT. It shows an algorithm suitable for solving two basic tasks: (1) To form groups from a given part set automatically such a way that within each group (class) the deviation between the numerical values allocated to two optional parts on the base of the same similarity parameters should be less than a given value; (2) To insert a new part into that group (class) which is the most adequate to it on the base of its properties taking into consideration a previously ordered part set and to create a new group (class) if inserting is not successful, respectively.The paper also outlines the fuzzy method constituting the mathematical base of the algorithm and gives a brief summary about the computer program realizing the algorithm in the present phase of development.

Keywords: Group Technology (GT), part classification, similarity matrix, fuzzy method

1. Introduction

Industrial, commercial and service changes in the last quarter of the twentieth century stimulated enterprises to increase the flexiblility of production. The adequate reply to this challenge was the unification and systems approach based integration of the means of production engineering and information technology. This kind of integration results in development of Computer Integrated Manufacturing (CIM) systems. Group Technology, as a comprehensive organizing principle and methodological tool of part manufacturing, plays a significant role in CIM systems.

Group Technology (GT), according to the definition of Ryerson Polytechnic (Toronto, Canada), is „... the philosophy of recognizing the similarities between entities (problems,

parts, machines, designs, processes etc) and utilizing this knowledge." [1]. Another useful definition: „Group Technology is the idea of studying a great population of apparently different items and then dividing them into groups of items having the same or similar characteristics." [2].

This paper only deals with dividing machined parts into groups of parts on the base of similarity characteristics. In production process, starting from production planning and going through design phase, process planning as well as the different phases of real manufacturing process (e.g.: prefabrication, part manufacturing, assembly, storage) to the final quality control, the activities are oriented to more or less similar individual parts [2], [3]. These parts can be ordered in a systematical way and the ordered sets, with finite number of elements for each, make it possible to apply procedures suitable for rationalization and optimization.

The group is a set of parts similar to each other to a certain extent and in some kind of sense and the individuals, because of their similar characteristics, make common design, planning, recording, storing, moving, machining etc. activities possible.

The method of Group Technology could not be very much used before seventies of the last century. The reason was the lack of the computer background suitable for supporting computational tasks. The computer-based methods spreading since that time, however, have improved efficiency and usefulness of the GT method to a considerable extent [2], [4]:

- creating the complex part models has accelerated significantly;
- decomposition of the part family trees can be executed in an efficient and fast way;
- velocity of the searching procedures based on similarity relations has increased to a great extent;
- the principles of grouping, building and decomposing can be extended to the manufacturing process as a whole;
- data bases and processing procedures can be unified, therefore compatibility of the interfaces can be ensured;
- GT applications have been supported by effectual software modules of general purpose and there has been a possibility to co-operate with them, respectively (e.g.: CAD systems, database handling systems).

Spreading of GT has been promoted by the essential changes in environmental conditions. They are as follows [2], [4]:

- product change has accelerated, demand for the modified, variable products has been increasing, i.e. demand for parts design has been increasing;
- flexible manufacturing requires frequented and fast change-over of series, therefore demand for the technology process planning of parts has also beenincreasing;
- the shortening through-put times require a technical preparation of higher and higher quality level, thus, in accordance with this fact, the demand for design of manufacturing tools, fixtures and jigs has been increasing, too;
- economical manufacturing needs an analytical supervising all the circumstances in connection with the manufacturing process itself to reveal reserves. To carry out this task, GT applications can provide a significant aid;
- GT-applications have been supported by computational technology, especially spreading of the local area networks and workstations. This support is more and more effective and is of smaller and smaller specific cost.

The advantages of GT, taking into consideration the aforementioned aspects, are as follows [2], [4]:

- GT promotes the integration of design, planning and manufacturing;
- GT guarantees a higher level of typifying, standardizing and harmonizing the cost calculation and loading procedures in comparison with the previous situation;
- GT decreases the costs of set-up, tooling and fixturing;
- GT makes it possible to decrease the through-put times and costs of parts design and process planning;
- GT contributes to standardization and decreases redundant redesigning the elements and parts with the same functions;
- GT helps to solve very complicated and hard-to-survey problems both in design and planning as well as manufacturing, on the base of systems approach.

2. Part classification by means of fuzzy methods

The algorithm and computer program to be shown in the paper should be suitable for solving two tasks connecting with each other. As an initial siutation let us consider the following:

It is given parts n $(n \in N)$ having features p $(p \in N)$ previously defined. Let $X_1, X_2, ..., X_n$ the identifiers of the parts and let us consider that the features of each part are expressed by real numbers p. So we can represent the part $X_i (1 \le i \le n)$ by means of a vector $(x_{i,1}, x_{i,2}, ..., x_{i,p}) \in R^P$ the components of which are the values of the features $1, 2, .., p$ concerning the part X_i. In the course of classification of the parts n the following tasks are to be executed:

Task1 (grouping task): The parts, i.e. the vectors should be grouped such a way that in a group (class) the deviation between the same feature values of two optional parts be under a given limit, i.e. it is to be determined how many groups (classes) can be formed and which parts are in the detached groups (classes).

Task2 (inserting task): Suppose that a new member X_{n+1} will be added to the parts set ordered into the groups (classes) $X_1, X_2, ..., X_n$. The purpose is to insert the part X_{n+1} in the group (class) adequate to the features of it to the greatest extent, or, if this is not possible, to put it into a separated group (class).

The authors suggest a fuzzy method for ordering the parts into groups (classes). Conventional and fuzzy sets are differ from each other in that in the case of conventional sets a certain element either belongs to the set or not, however, in the case of fuzzy sets the measure of belonging to is expressed by means of a real number between *0* and *1*. Let U be a set well-defined in conventional sense. The fuzzy set A is formed from the set U by means of a function $\mu_A : U \to [0,1]$ named *membership function*. (Accordingly, μ_A allocates to the elements of U real numbers in the interval [0,1]).

Let $\alpha \in [0,1]$ a fixed real number. In the case of a fuzzy set $A=(U, \mu_A)$, the totality of those elements $x \in U$ for which $\mu_A(x)=\alpha$, is named a *level* of the fuzzy set. The name of the set $A_\alpha = \{x \in U | \mu_A(x) \ge \alpha\}$ is a section α of the fuzzy set in question. We introduce two methods. The ***Method1*** examines how many groups (classes) can be formed on the base of the ***Task1***. For each part $X_i (1 \le i \le n)$ and feature $j (1 \le j \le p)$ we define a real number $\mu_{i,j} \in [0,1]$ to express that the part X_i has the feature j to what extent, i.e. we allocate to the parts set

$\{X_1,X_2,...,X_n\}$ known vectors $(\mu_{1,1},...,\mu_{1,p}),...,(\mu_{n,1},...,\mu_{n,p})$ where the components of the vectors can only receive real values between *0* and *1*.

We can interprete the „distance" of two optional parts X_i and X_j according to the next formula [4], [5]:

$$S_{i,j} = \frac{\sum_{k=1}^{p} \min(\mu_{i,k}, \mu_{j,k})}{\frac{1}{2}\sum_{k=1}^{p} (\mu_{i,k} + \mu_{j,k})} \tag{1}$$

From these distances we can form the *similarity matrix* of the parts in the following manner:

$$\mathbf{S} = \begin{bmatrix} 1 & S_{1,2} & S_{1,n} \\ S_{2,1} & 1 & S_{2,n} \\ S_{n,1} & S_{n,2} & 1 \end{bmatrix}. \tag{2}$$

The similarity matrix **S** is symmetric to the principal diagonal and the value of all the elements in the principal diagonal is *1*.

To carry out ordering the parts into groups (classes) on the base of the similarity matrix **S** we have to introduce a new mathematical operation (°). If **R** and **T** are fuzzy relations interpreted in the same set of elements *n* then the matrix of the *product relation* **R°T** can be obtained from the matrices **R** and **T** by means of multiplying, where this mathematical operation is not the conventional matrix multiplying but an operation defined as follows:

$$\underset{\mathbf{R}}{\begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,n} \\ r_{2,1} & r_{2,2} & r_{2,n} \\ r_{n,1} & r_{n,2} & r_{n,n} \end{bmatrix}} \circ \underset{\mathbf{T}}{\begin{bmatrix} t_{1,1} & t_{1,2} & t_{1,n} \\ t_{2,1} & t_{2,2} & t_{2,n} \\ t_{n,1} & t_{n,2} & t_{n,n} \end{bmatrix}} = \underset{\mathbf{E}}{\begin{bmatrix} e_{1,1} & e_{1,2} & e_{1,n} \\ e_{2,1} & e_{2,2} & e_{2,n} \\ e_{n,1} & e_{n,2} & e_{n,n} \end{bmatrix}} \tag{3}$$

where we can get to an element $e_{i,j}$ of the result matrix in the following way:

$$e_{i,j} = \max[\min(r_{i,k}, t_{k,j})]. \quad (k=1,...,n) \tag{4}$$

The matrix $\mathbf{R_{n-1}}$ is derived from the similarity matrix **S** as follows:

$$\mathbf{R_{n-1}} = \underbrace{\mathbf{S} \circ \cdots \circ \mathbf{S}}_{(n-1)-times} \tag{5}$$

where *n* is the number of the parts.

The elements of the matrix $\mathbf{R}_{n-1}$ express similarity between the parts. If $R_{ij}=0$ then part i and part j are completely different and else if $R_{ij}=1$ then part i and part j are completely alike.

After having constructed the matrix $\mathbf{R}_{n-1}$, we can give different treshold values for representing the similarity of the parts (e.g. the parts will be ranged in the same class if they are similar to each other in 80 % at least, i.e. $\alpha=0.8$). In this case the elements can be ranged into classes through an immediate reading-out of the matrix $\mathbf{R}_{n-1}$ (***Method1***).

Example: Let the parts set $X=\{X_1,X_2,X_3,X_4,X_5,X_6\}$ is given. The matrix $\mathbf{R}_{n-1}$ derived from the similarity matrix **S** is as follows:

$$\mathbf{R}_{n-1}=\begin{bmatrix} 1 & 0.2 & 1 & 0.6 & 0.2 & 0.6 \\ 0.2 & 1 & 0.2 & 0.2 & 0.8 & 0.2 \\ 1 & 0.2 & 1 & 0.6 & 0.2 & 0.6 \\ 0.6 & 0.2 & 0.6 & 1 & 0.2 & 0.8 \\ 0.2 & 0.8 & 0.2 & 0.2 & 1 & 0.2 \\ 0.6 & 0.2 & 0.6 & 0.8 & 0.2 & 1 \end{bmatrix}. \tag{6}$$

Taking different values for $\alpha\in[0,1]$ we have different partitions of the set X on the base of the matrix $\mathbf{R}_{n-1}$. If $\alpha=1$ then the number of the classes will be the highest one, because the completely identical elements can only be ranged into the same class. In the case of example (6) the set X will consist of 5 partitions. They are as follows: $\{X_1,X_3\};\{X_2\};\{X_4\};\{X_5\};\{X_6\}$. If $\alpha=0$ then the number of the classes will be 1 exactly, since all the elements will be ranged into one class. It is easy to see that in the other cases the number of the classes will be equal to the maximum number of the classes or less than the maximum number. E.g. in case of $\alpha=0.8$ the elements of the set X can be arranged in 3 classes. They are as follows: $\{X_1,X_3\};\{X_4,X_6\};\{X_2,X_5\}$.

Solving the ***Task1*** does not make it possible to order the elements in optional number of classes therefore we have to modify the ***Task1*** such a way that we give not the similarity trashold value (α) but an integer number $c\in N$ which indicates that the elements $X_1,X_2,\ldots,X_n$ are to be grouped into how many classes (***Task1****).

The ***Method2*** which is suitable for solving the ***Task1**** is based on the concept of fuzzy-classification and an iterative procedure. As is known that in the case of conventional (non-fuzzy) classification an element either belongs to a given class or not, there exists no other case. However, the fundamental principle of fuzzy classification is that an element belongs to a given class in a certain percentage which can be expressed by means of a real number within the interval [0,1]. In accordance with this, we define the number u_{ij} which means that the element X_j belongs to the i-th class to what extent. By this means we have the matrix **T** as follows:

$$\mathbf{T} = \begin{bmatrix} & X_1 & X_2 & & X_n \\ 1 & u_{1,1} & u_{1,2} & & u_{1,n} \\ 2 & u_{2,1} & u_{2,2} & & u_{2,n} \\ \dots & & & & \\ c & u_{c,1} & u_{c,2} & & u_{c,n} \end{bmatrix}. \tag{7}$$

For the matrix **T** the following constraints are valid:

$$\begin{aligned} &0 \le u_{i,j} \le 1, \quad i = 1,2,...,c \quad j = 1,2,...,n \\ &\sum_{i=1}^{c} u_{i,j} = 1, \quad j = 1,2,...,n \\ &\sum_{j=1}^{n} u_{i,j} > 0, \quad i = 1,2,...,c \end{aligned} \tag{8}$$

For solving the ***Task1**** we arrange the known vectors $(\mu_{1,1},\mu_{1,2}...,\mu_{1,p}),...(\mu_{m,1}, \mu_{m,2}, ...,\mu_{m,p})$ in an *nxp* type matrix **M**:

$$\mathbf{M} = \begin{bmatrix} \mu_{1,1} & \mu_{1,2} & & \mu_{1,p} \\ \mu_{2,1} & \mu_{2,2} & & \mu_{2,p} \\ & & & \\ \mu_{n,1} & \mu_{n,2} & & \mu_{n,p} \end{bmatrix} \tag{9}$$

Let us denote the classes of the decomposition to be obtained by $C_1,C_2,...,C_c$. If we would know the matrix **T** of *cxn* size then we could determine the fictitious feature values $1,2,...,p$ correspond to the classes $C_1,C_2,...,C_c$ in the following manner:

$$\begin{aligned} &v_{i,k} = \frac{\sum_{j=1}^{n} u_{i,j}^{m} \ \mu_{j,k}}{\sum_{j=1}^{n} u_{i,j}^{m}} \\ &i = 1,...,c \\ &k = 1,...,p \end{aligned} \tag{10}$$

where $m>1$ is a rational number previously fixed.

On the base of formula (10) we would get a vector $V_i=(v_{i,1}, v_{i,2}, \quad v_{i,p})$ that would express to what extent class C_i has the features $1,2,...,p$.

Hence, the steps of the procedure needed can be summarized as follows [4], [5]:

(1) Let us define a matrix

$$\mathbf{T}_0 = \left\lfloor \mu_{i,j}^0 \right\rfloor \qquad (11)$$
$$i = 1,...,c \quad \text{and} \quad j = 1,2,...,n$$

such a way that the elements of which as optional values should meet the constraints (8).

(2) On the base of formula (10) we determine the values $v_{i,k}$ for every i=1,2,…,c and k=1,2,…,p, i.e. we determine the vectors V_i.

(3) By using formula (12), see below, we determine the new value $u'_{i,j}$ of the marix **T** and fill up matrix $\mathbf{T}_0$ with the $u_{i,j}$ values before the iterative step. In the course of iterative process, from the beginning to end, we need two matrices **T** and $\mathbf{T}_0$ such a way that $\mathbf{T}_0$ includes always the values of the previous step.

$$u'_{i,j} = \frac{1}{\sum_{l=1}^{c} \left[\frac{\sum_{k=1}^{p} (\mu_{j,k} - v_{i,k})^2}{\sum_{k=1}^{p} (\mu_{j,k} - v_{l,k})^2} \right]^{\frac{1}{m-1}}}. \qquad (12)$$

(4) If $K = \{\max | u_{i,j} - u_{i,j}^0 |,\ i = 1,...,c\ ,j = 1,...,n\} \le \varepsilon$, where ε is an error-limit given by us, we stop else return to the step (2). The process will be repeated until $K \le \varepsilon$, i.e. until determination of optimum u_{ij} ($i=1,...,c$, $j=1,...,n$) with ε error tolerance.

(5) Element X_j will be ranged into class C_i if $u_{i,j}$ is the maximum of $u_{1,j}, u_{2,j}, ..., u_{c,j}$ values, i.e.

$$u_{i,j} = \max\{u_{k,j} \mid 1 \le k \le c\} \qquad (13)$$

After these we determine the last $V_1, V_2, ..., V_c$ vectors featuring the classes C_1, $C_2, ..., C_c$ on the base of formula (10).

The ***Method1*** does not give an acceptable result if the number of parts n is large, at the same time, it is more flexible than ***Method2.*** Therefore it is expedient to combine the two methods. Firstly, on the base of ***Method1***, an adequate number of classes will be determined, i.e. the number of classes c will be calculated by means of matrix **S** for a certain similarity threshold value α previously given. After this ***Method2*** will be applied which results in much more correct fuzzy partition.

To solve ***Task2***, i.e. to insert the part X_{n+1} into the adequate group, first the parts $X_1, X_2, ..., X_n$ should be arranged into groups (classes) according to ***Method2***.

Let us allocate the vector $(\mu_{n+1,1}, ..., \mu_{n+1,p})$ to the part X_{n+1} and the vectors

$$V_1 = (v_{1,1}, v_{1,2}, ..., v_{1,p})$$
$$V_2 = (v_{2,1}, v_{2,2}, ..., v_{2,p})$$

$$V_c = (v_{c,1}, v_{c,2}, ..., v_{c,p})$$

to the classes $C_1,C_2,...,C_c$ according to the formula (10) which expresses that the classes $C_1,C_2,...,C_c$ have the features $1,2,...,p$ to what extent.

Let us denote a threshold value by t which shows above what distance the part in question can be allocated to a given class. This threshold value is constant for every classes. If

$$(X_{n+1},V_i) = \max\{(X_{n+1},V_k) \mid 1 \le k \le c \text{ and } (X_{n+1},V_i) \ge t\} \tag{15}$$

then X_{n+1} can be inserted into the class C_i featured by the vector V_i because this class is the nearest one taking into consideration all the features $1,2,...,p$.

If X_{n+1} can be inserted into neither of classes it can be arranged in a new class or the equivalence analysis can be repeated for the part set of $(n+1)$ elements $\{X_1,X_2,...,X_n,X_{n+1}\}$ because it is possible that the „distances" of X_{n+1} from the other $X_1,X_2,...,X_n$ elements are not the same. So, in the course of a new classification, it would be possible to insert it into a certain class.

1. A computer program for part classification and inserting a new part

The program suitable for solving ***Task1*** and ***Task2*** is not an idependent one but it is a part of a larger application. Therefore the program has not yet own user interface at present but operation of the algorithm and computing can be traced by means of a „test-option" built-in the program and the input data as well as the results are sstored in text files readable in an easy way.

Starting the program can be carried-out in four kinds of way:

„fuzzy_gt" (arrangement of n parts into groups)

„fuzzy_gt-t" (arrangement of n parts into groups and writing the partial results on the screen)

„fuzzy_gt new" (inserting the $(n+1)$-th part)

„fuzzy_gt new-t" (inserting the $(n+1)$-th part and writing the partial results on the screen).

Solving the Task1 and Task1*

Solving the tasks is carried-out on the base of the process chart seen in Fig.1. The input data have to be given in the file *fuzzy_in.-dat* as follows:

Number of rows:		*Data:*
1	N	the number of the parts to be arranged into groups
2	p	the number of features for a part
3	α	similarity threshold value
4	c	the number of groups (classes)
5	0	
6	0	
7	0	
8	0	
9	0	
10	0	
11	$\mu_{1,1}$	The 1st feature of the 1st part ($\mu_{1,1} \in [0,1]$).
	$\mu_{1,2}$	The 2nd feature of the 1st part ($\mu_{1,2} \in [0,1]$)

$\mu_{2,1}$ The 1st feature of the 2nd part ($\mu_{2,1} \in [0,1]$)
$\mu_{2,2}$ The 2nd feature of the 2nd part ($\mu_{2,2} \in [0,1]$)

$\mu_{n,1}$ The 1st feature of the n-th part ($\mu_{n,1} \in [0,1]$)

$\mu_{n,p}$ The p-th feature of the n-th part ($\mu_{n,p} \in [0,1]$).

If $c=0$ then the program determines the number of groups (classes) c on the base of the similarity matrix **S** and the threshold value α.

The result of the analysis can be found in the file *fuzzy-ou.dat* the structure of which is as follows:

Numbers of rows:		*Data:*
1	c	the number of groups (classes)
2	$X_{1,1}$	The 1st part is in the 1st group (0 or 1)
	$X_{1,2}$	The 2nd part is in the 1st group (0 or 1)
	$X_{1,n}$	The n-th part is in the 1st group (0 or 1)
	$X_{2,1}$	The 1st part is in the 2nd group (0 or 1)
	$X_{2,n}$	The n-th part is in the 2nd group (0 or 1)
	$X_{c,1}$	The 1st part is in the c-th group (0 or 1)
	$X_{c,n}$	The n-th part is in the c-th group (0 or 1)

Solving the Task2

To solve ***Task2*** we have to know the data of $(n+1)$-th part. Giving the data of n part is the same as it was previously. Data of the $(n+1)$-th part have to be given in the file *fuzzy_i2.dat* as follows:

Number of rows:		*Data:*
1	$\mu_{n+1,1}$	The 1st feature of the $(n+1)$-th part ($\mu_{n+1,1} \in [0,1]$)
2	$\mu_{n+1,2}$	The 2nd feature of the $(n+1)$-th part ($\mu_{n+1,2} \in [0,1]$)
	$\mu_{n+1,p}$	The p-th feature of the $(n+1)$-th part ($\mu_{n+1,p} \in [0,1]$).

Computing is carried-out on the base of the flow chart seen in Fig.1 and the result will be stored in the file *fuzzy_02.dat* which has the following structure:

Number of rows:		*Data:*
1	$X_{n+1,1}$	The $(n+1)$-th part is in the 1st group (0 or 1)
2	$X_{n+1,2}$	The $(n+1)$-th part is in the 2nd group (0 or 1)
	$X_{n+1,c}$	The $(n+1)$-th part is in the group c (0 or 1)
	$X_{n+1,c+1}$	The $(n+1)$-th part has to be arranged into a new group $(c+1)$ (0 or 1).

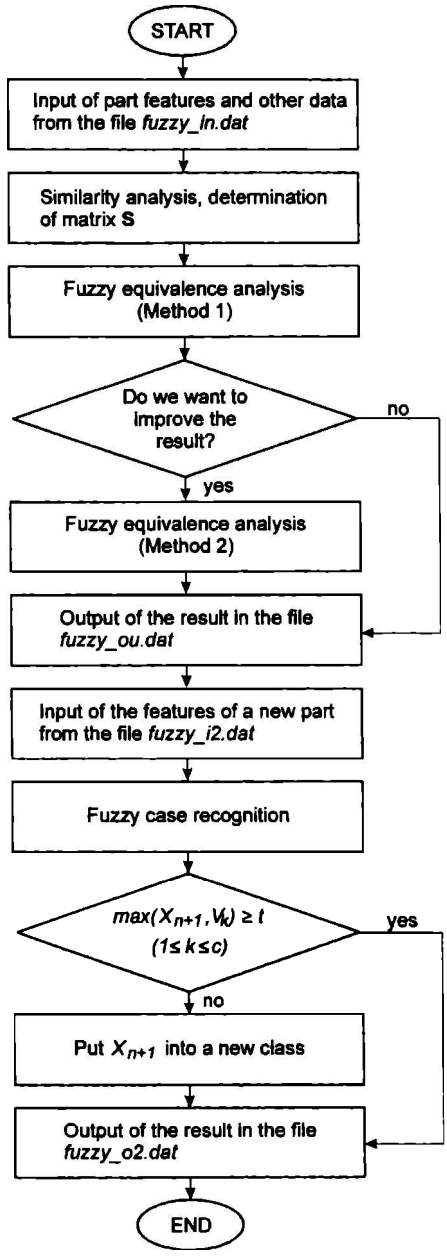


Fig.1.: The flow chart of the computer program suitable for similarity analyis

2. Acknowledgements

The work on which this paper is based has been supported by the OTKA (Hungarian Scientific Research Fund) project entitled ***„Application of Concept Lattices and Fuzzy Methods in Group Technology"*** (Id. No.: T03243, headed by Tóth, T.). In addition, the research has also been supported by the Hungarian Academy of Sciences (HAS) within the framework of ***Production Information Engineering Research Team*** (PIERT) established at the Department of Information Engineering in 1999.
This work connected with the project **„Digital Factory"** therefore it has has also been supported within the framework of ***National Research & Development Program*** (NRDP) grant No. 2/040/2001.

The financial support of the research by the three sources is gratefully acknowledged.

REFERENCES

1. WILLIAMS, P.J.: *Computer Integrated Manufacturing (CIM): A Management Perspective.* Ryerson Polytechnical Institute CATE, Toronto, 1985.
2. *Group Technology and Manufacturing Cells.* In: Modern Machine Shop, NC/CIM Guidebook, 1988., p.230.
3. GALLAGHER, C.C. and KNIGHT, W.A.: *Group Technology Production Methods in Manufacture.* Ellis Horwood Ltd 1986.
4. RADELECZKI, S. and TÓTH, T.: *Planning Discrete Technology Processes by means of Artificial Intelligence Methods: Application of Fuzzy Methods in Group Technology.* Research Report, OTKA 2348/91., 1992. (in Hungarian).
5. XU, H. and WANG, H.P.: *Part Family Formation for GT Applications Based on Fuzzy Mathematics.* Int. J. Prod. Res. Vol 27., No.9., 1989., pp.1637-1651.

Production Systems and Information Engineering, Miskolc, Vol. 1. (2003), pp. 83-90

FLOW-SHOP SCHEDULING BASED ON REINFORCEMENT LEARNING ALGORITHM

PÉTER STEFÁN
Computer and Automation Research Institute, Hungarian Academy of Sciences,
Victor Hugo u. 18-22, H-1132 Budapest, Hungary,
Phone: (+36-1) 4503075, Fax: (+36-1) 2709650
stefan@sztaki.hu

[Received September 2002 and accepted May 2003]

Abstract. In the paper a machine learning based method will be proposed to give a quasi-optimal solution to the *m*-machine flow-shop scheduling problem. Namely, given a set of parts to be processed and a set of machines to carry out the process and the sequence of machines is fixed, each part should have the same technological path on all machines; the order of jobs can be arbitrary. The goal is to find appropriate sequence of jobs that minimizes the sum of machining idle times.

1. Introduction

Recent research has put increasing emphasis on scheduling in all production phases. The goal of scheduling is defined as finding the best sequence of different activities (processing operations, delivering the goods) given a set of constraints imposed by the real world processes. These constraints can cover physical laws as well al rising costs that can make production unrealistic or uneconomic.

Basically, there are three main scheduling concepts: mathematically grounded algorithms, heuristic approaches and algorithms supported by machine learning (ML). The first concept can be adapted to small-sized scheduling problems. Johnson's algorithm, e.g., solves a two-machine flow-shop scheduling task that is established by classical algebraic and dynamic programming ways as well. The advantage of the algorithm is that it is well defined, exact and can be generally applied to the wide range of two-machine scheduling tasks. The price is lack of scalability: i.e. no mathematical proof can be given for a larger number of machines.

As a potential improvement, there are two directions of research to overcome the restrictions of mathematical formulations: using heuristics, and/or machine-learning. Both directions try to set up some model of human being problem-processing capability but in different ways. While heuristic approaches provide direct rules of thumb to follow, but no algorithm to find the solution in a modified decision environment. ML methods give a model of a mental process itself. As the knowledge of the learning agent improves the method results in solutions that are more and more close to the optimal solution, even in changing environment.

In the paper an algorithm will be shown that is capable of "learning from scratch" using a reward-punishment procedure, called reinforcement learning (RL) [4].

2. Scheduling task

The scheduling problem, under consideration, is called flow-shop scheduling where given a set of parts to be processed (jobs) and a set of machines for processing. Each part has the same technological path on all machines; the order of jobs is arbitrary. The goal is to find the appropriate sequence of jobs that minimizes the sum of idle times.

2.1. Johnson's algorithm

Let jobs be denoted by $j_1, j_2, \ldots, j_n$ while the two machines by A and B. If there are no precedence restrictions among the jobs, there is $n!$ (n factorial) number of possible job-sequencing on the two equipment, which yields non-polynomial (NP) hard task. Figure 1 illustrates the Gantt diagram of one possible job sequence.

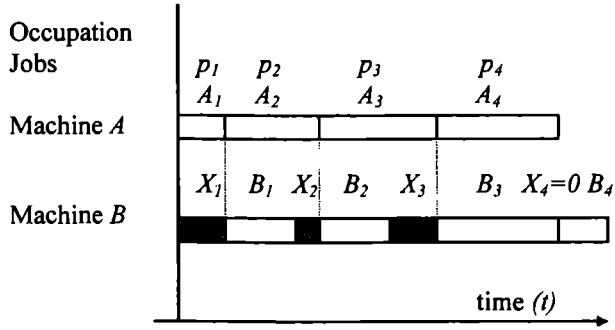


Figure 1 Job sequence on two-machines. p_i denote parts, A_i machining times on A, B_i machining times on B and X_i are idle times on B.

In the figure, $X = \sum_{j=1}^{n} X_i$ indicates the total off-machining time on machine B, as well as the total idle time of the scheduling task with two-machines. The goal of optimization is to find a job-order, which minimizes X.

The possible data structure of the algorithm, for example, is an $n \times 2$ matrix which can be seen in Figure 2. The scheduling method itself is illustrated in Figure 3. The proof of the algorithm can be found, e.g., in [1].

2.2. Three-machine extensions: Palmer's and Dannenbring's methods

The two-machine scheduling algorithm can be extended to three-machine scheduling by imposing additional restrictions on machining times.

A method that was first published by Palmer uses job priorities to set up job sequences. Single priority value compresses the following concept in a single rule: jobs that produce

shorter execution on the first machine are sorted to the beginning; jobs that have shorter execution times on the third machine are left behind [5].
Dannenbing's algorithm decomposes the m-machine scheduling task to m-1 two-machine tasks compromising quasi-optimal values [6].

	A (minutes)	B (minutes)
Job1	10	15
Job2	20	15
Job3	30	65
Job4	20	10
Job5	45	100

Figure 2 Machining times of different jobs on machines *A* and *B*. The length of machining is measured e.g. in minutes

```
function Johnson (table of machining times)
                                        return optimal sequence
  let Q denote the queue of jobs
  initialize Q with empty set
  for each j cell of the table {
    find the minimal machining time scanning in both columns
    if the time found occurs in column A then
      add the job_j to the beginning of the queue Q
    else add job_j to the end of the queue Q
      delete the slot found
    endif
  return Q as the optimal sequence
end
```

Figure 3 Johnson's algorithm

3. Machine learning approach

In order to implement any machine-learning algorithm, first, the concept of learning should be defined. Learning, in its original sense, means that a system is capable of modifying its internals (structure or parameters) to satisfy the requirements of the "evaluator" (or teacher, or external environment).

3.1. Determination of the optimality criterion

For translating the definition into scheduling terms, some evaluation process has to be developed to be capable of distinguishing among different schedule plans (evaluator, or fitness function in genetic algorithms).

As the definition of scheduling environment is clear, it is easy to develop an evaluation algorithm to express the "goodness" of scheduling in numerical terms. Figure 4 shows an algorithm that computes idle times to an arbitrary number of machines and jobs. The real benefit of the method is that it computes partial idle time data corresponding to individual machines and jobs.

The evaluator inputs the matrix of machining times (**M**) and the job and/or machining sequence permutation vectors (**r**, **p**). Both vectors are necessary for the proposed scheduling algorithm, not by the scheduling task itself. The number of jobs is indicated by n, the number of machines by m. The algorithm outputs a scalar value v indicating the sum of idle times, vector **d** stores sum of idle times preceded by the corresponding job.

The computation cost of evaluation is at level $O(nm)$. More details about the algorithm can be found in [5].

```
input M[m,n], r[m], p[n];
output v;
storage d[n], D[m,n];
function n_machine return D[m,n] or v
begin
  v:=0;
  for i:=1 to m do
    s[i]:=M[r[i],1];
    d[i]:=v;
    D[i,1]:=v;
    v:=v+M[r[i],1];
  end
  for j:=1 to n do D[1,j]:=0;
  for j:=2 to n do
    for i:=1 to m-1 do
      s[i]:=s[i]+M[r[i],p[j]];
      D[i+1,j]:=max(0,s[i]+d[i]-s[i+1]-d[i+1]);
      d[i+1]:=d[i+1]+D[i+1,j];
    end
    s[m]:=s[m]+M[r[i],p[j]];
  end
  v:=0;
  for i:=1 to m do v:=v+d[i];
  return v;
end
```

Figure 4 Determination of machining idle times for arbitrary number of machines and jobs

3.2. The learning module

Having the evaluation algorithm been examined, a question can be immediately addressed: how can it be used in real-life machine learning applications? There are two basic approaches under consideration: reinforcement learning and genetic algorithm.

One of the possible approaches is to use a reinforcement-learning (RL) based module, called Q-learning to maintain job precedence preferences, or in RL terms, action-state values.

```
input M[m,n], jobs, alpha, gamma;
output r[n];
storage Q[n][n], V[n];
function Q-update return r[n];
begin
  Q[i][j]=0 for all i=1..n, j=1..n;
  while annealing_cycle do
    r[n]=permutation(jobs);
    reward=n_machine(M[m,n],r[n]);
    for i:=1 to n-1 do
      update_Q_table(Q[r[i],r[i+1]], reward,alpha,gamma);
    end
    update_V(r[1],reward,alpha,gamma);
  end
end
```

Figure 5 Q-learning based flow-shop sequencer algorithm

RL methods evolved from dynamic programming and automata theory and model reality through a set of states, state-changes and values (preferences) assigned to both. RL also defines update rules over the state, state-change model, which are used for maintaining values with respect to the measured feedback provided by the environment of the learning model.

Whenever an RL method is applied to a certain problem, the property that can be used as states should be identified. Furthermore, the state-changes and the reward measurement should also be identified. Then the RL algorithm makes explorative and exploitative traverses in the state-space trying to find a path that is highly rewarded. The benefit of the algorithm is its capability of exploration, i.e. traversing through states that are not well-rewarded but may yield higher reward in the long run, bypassing local maxima this way. It is important to pay attention to exploitation and exploitation balancing problem [4]. Exploration is interpreted as an operation mode of the learning agent when it makes experiments and tries to discover its environment. On the other hand, exploitation is a mode when the agent has gathered enough knowledge and makes real decisions.

In the flow-shop scheduling exercise the model takes machining times, machining costs as input parameters, and job sequence as a variable parameter, and a certain job sequence is sought that minimizes idle time, in the long run.

To fit RL methods, it is reasonable to define states as job sequences, or more precisely job precedence relations. State-changes (or actions) are defined as changes in relations. An action step is performed by a permutation operator, which sets up a job sequence according to precedence preferences. At the beginning no preferences are given, so states are traversed randomly. As learning proceeds, preferences are updated, which, in turn, influences action selection policy converging to the found quasi-optimal job sequence. From this respect the learning algorithm is a directed search procedure.

Parameters of the algorithm are the same as those of the evaluation algorithm, except RL specific arguments: **Q** stores action-state values (decision preferences), alpha and gamma regarding to learning rate and discount rate [2].

The Q-learning based algorithm can be seen in Figure 5. It introduces a new element, **v**, which is a vector of preferences. Array **v** expresses the value of starting the job sequence with job_i, for all jobs. Update methods are formulated as the usual RL update rules:

$$Q_{n+1}(j_i, j_j) = Q_n(j_i, j_j) + \alpha(r + \gamma \max_k Q_n(j_i, j_k) - Q_n(j_i, j_j)) \quad (1)$$

$$V_{n+1}(j_i) = V_n(j_i) + \alpha(r + \gamma \max_k Q_n(j_i, j_k) - V_n(j_i)) \quad (2)$$

Evaluation algorithm shown in Figure 4 can be used as a fitness evaluator of any genetic algorithm-based method as well. In this case job sequences are regarded as "fenotypes", heuristic operators such as mutation and crossover are defined as specific job permutations.

4. Implementation

An RL-based scheduler has been developed in Java in order to validate theoretical results. The code is formulated to be modular to let non-RL modules be plugged in and allow comparisons between them.

Figure 6 shows the screenshot the Gantt chart of an example five-machine, nine-job scheduling task.

So far the scheduler is capable of learning using step-by-step iteration, and through a single annealing period. Annealing schedule[1] is set up manually. Algorithm in Figure 4 has been extensively tested on different job-machine setups. The accuracy of the learning procedure depends on the "speed" of the annealing schedule: the larger the annealing period is, the more accurate the solutions are. Given a time horizon three annealing schedules have been compared. The best result was achieved when concave annealing function was chosen, which let exploration until about 95% of the full time horizon and "chopped down" exploration turning immediately into exploitation phase.

The optimality criterion can be modified easily: multiple optimum criteria can also be applied provided that it can be mapped into a single scalar feedback.

5. Conclusions and future work

It has been established that RL-scheduler is able to find close-to-optimal solution, and RL combined with simulated annealing and balancing algorithms are also capable of finding quasi-optimal solutions when machining-times vary in time.

[1] The term annealing schedule regards the temperature-time function and has nothing in common with machining schedule.

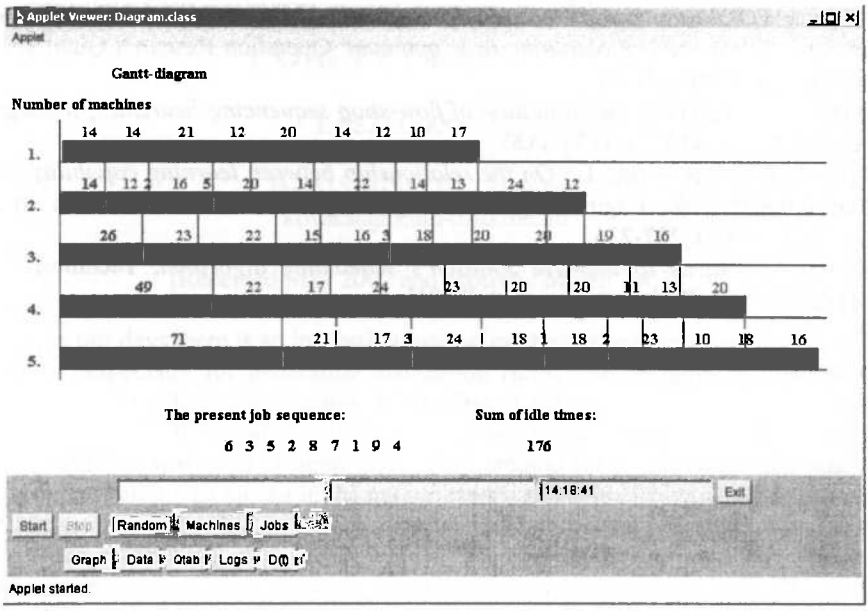


Figure 6 Gantt-chart of a 9-job, 5-machine task in a RL-based simulator, numbers above stripes indicate processing or idle times

As for the future plans, it is purposed to make a detailed comparison among the results of a RL-scheduler; a genetic algorithm-based scheduler and the Johnson algorithm. A new protocol is also under development which provides a real manufacturing environment-virtual manufacturing environment communication primitive, which can be used for on-line learning.

Acknowledgements: The author would like to express his gratitude to Prof. László Monostori, Prof. László Dudás, Prof. Ferenc Erdélyi and Prof. Tibor Tóth for their help and inspiration.

References

[1] TÓTH, T.: *Design and Planning Priciples, Models and Methods in Computer Integrated Manufacturing*, Publisher of the University of Miskolc, (1999), 252 p. (in Hungarian).

[2] MONOSTORI, L., MÁRKUS, A. VAN BUSSEL, H, WESTKAMPFER, E.: *Machine learning approaches to manufacturing*, Annals of the CIRP, (1996), pp.675-712.

[3] TETI, R., KUMARA, S.R.T.: *Intelligent computing methods for manufacturing systems*, Annals of the CIRP, (1997), pp.1-24.

[4] SUTTON, R., BARTO, A.: *Reinforcement Learning (An Introduction)*, The MIT Press, Cambridge, Massachusetts, (1998), 312 p.

[5] PALMER, D.S.: *Sequencing jobs through a multi-stage process in the minimum total time-a quick method of obtaining near optimum*, Operation Research Quarterly, Vol. 16, No. 3, (1965), 101-107.

[6] DANNENBRING, D.G.: *An evaluation of flow-shop sequencing heuristics*, Management Science 23(11), (1977), 1174-1182.

[7] STEFÁN, P., MONOSTORI, L.: *On the relationship between learning capability and the Boltzmann-formula*, Engineering of Intelligent Systems, Lecture Notes in AI 2070, Springer, (2001), 227-236.

[8] STEFÁN, P.: *Ideas to improve Johnson's scheduling algorithm*, Technical Report, MTA-SZTAKI, Budapest, Hungary, (2001).

Production Systems and Information Engineering, Miskolc, Vol. 1. (2003), pp. 91-106

ALGORITHMS FOR BUILDING CONCEPT SETS AND CONCEPT LATTICES

LÁSZLÓ KOVÁCS
Department of Information Technology, University of Miskolc
H-3515 MISKOLC, Hungary
kovacs@iit.uni-miskolc.hu

[Received May 2002 and accepted March 2003]

Abstract. In our days there is an increasing interest on the application of concept lattices for data mining, especially for generating association rules. The building of concept lattice consists of two, usually distinct phases. In the first phase the set of concepts is generated. The lattice is built in the second phase from the generated set. The paper gives an overview of the available methods and presents a proposed method for contexts of large size where the full context can not be stored in the main memory and some objects may be repeated in the context several times. The proposed algorithm for concept set generation is a fine-tuned version of the incremental concept set building method. At the end of the paper, the test results for comparing the new method with some known methods are given. The proposed method yields in a significantly better cost value than the other methods under the assumed conditions.

Keywords: formal concept analysis, concept lattice, algorithm, cost function

1. Introduction

Concept lattices are used in many application areas to represent conceptual hierarchies stored in a hidden form in the underlying data. The field of Formal Concept Analysis [1] introduced in the early 80ies has grown to a powerful theory for data analysis, information retrieval and knowledge discovery. In our days, there is an increasing interest on the application of concept lattices for data mining especially for generating association rules [8]. One of the main characteristics of this application area is the large amount of structured data to be analysed. Beside this area another important application field is the program analysis inside a compiler using concept lattices of very large size. A technical oriented application field of Formal Concept Analysis is the area of production planning where concept lattices are used to partition the products into disjoint groups yielding an optimal processing cost [6]. Since the cost of building a concept lattice is a super-linear function of the corresponding context size, the efficient computing of concept lattices is a very important issue investigated for several years [5].

The building of concept lattices consists of two usually distinct phases. In the first phase the set of concepts is generated. The lattice is built in the second phase from the generated set. We can find proposals in the literature for both variants, i.e. there are proposals addressing only one of the two phases and there are methods for combining these phases into a single algorithm. Based on the analysis of these methods, the cost for both steps is about the same order of magnitude and the asymptotic cost depends on mainly three parameters: the number of objects, the number of attributes and the number of concepts.

The cost is always larger than the product of these parameters. The concept-set generation algorithms have two main variants. The methods of the first group work in batch mode, assuming that every element of the context table is already present. The most widely known member of this group is the Ganter's next closure method. The other group of proposals uses an incremental building mode. In this case, the concept set is updated with new elements if the context is extended with a new object. The Godin's method belongs to this group. Regarding the phase for building the lattice, the proposed approaches are based on the considerations that the lattice should be built up in a top-down (or bottom-up) manner because in this case only the elements of the upper (or lower) neighbourhood are to be localised. The second usual optimisation step is to reduce the set of lattice elements tested during the localisation of the nearest upper or lower neighbour elements.

This paper addresses both of the problems, the generating of concept sets and the building of concept lattices. The proposal is intended to use for contexts of large size where the full context can not be stored in the main memory. According to our assumption, the access to context data is an expensive operation. Another basic feature of the investigated problem area is that the same incoming attribute set may occur several times in the different input objects, i.e. the objects may have the same set of attributes in the context.

2. Formal Concept Analysis

This section gives only a brief overview of the basic notations of the theory for *Formal Concept Analysis*. For a more detailed description, see [1].

A *K context* is a triple *K(G,M,I)* where *G* and *M* are sets and *I* is a relation between *G* and *M*. The *G* is called the set of *objects* and *M* is the set of *attributes*. The cross table *T* of a context *K(G,M,I)* is a matrix form description of the *relation I*:

$$t_{ij} = \begin{cases} 1, & \text{if } g_i I a_j \text{ and} \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $g_i \in G$, $a_j \in M$.

For every $A \subseteq G$, a *derivation operator* is defined:

$$A' = \{ a \in M \mid g\, I\, a \text{ for } \forall\, g \in A \} \tag{2}$$

and for every $B \subseteq M$

$$B' = \{ g \in G \mid g\, I\, a \text{ for } \forall\, a \in B \}. \tag{3}$$

The pair *C(A,B)* is a *concept* of the *K* context if

$$\begin{aligned} &- A \subseteq G \\ &- B \subseteq M \\ &- A' = B \\ &- B' = A \end{aligned} \tag{4}$$

are satisfied. In this case, the A is called the *extent* and B is the *intent* of the C concept. It can be shown that for any $A_i \subseteq G$, $i \in I$

$$(\cup_{i \in I} A_i)' = \cap_{i \in I} A'_I \tag{5}$$

and similarly for any $Bi \subseteq M$, $i \in I$

$$(\cup_{i \in I} B_i)' = \cap_{i \in I} B'_I \tag{6}$$

is satisfied.

Considering the Φ set of all concepts for the K context, an *ordering relation* can be introduced for the concept set in the following way:

$$C_1 \leq C_2 \tag{7}$$

if

$$A_1 \subseteq A_2$$

where C_1 and C_2 are arbitrary concepts. It can be shown that for every (C_1,C_2) pair of concepts, the following rules hold true:

$$C_1 \wedge C_2 \in \Phi \tag{8}$$

and

$$C_1 \vee C_2 \in \Phi.$$

Based on these features, $(\Phi, \leq)$ is a lattice, called *concept lattice*. According to the Basic Theorem of concept lattices, $(\Phi, \leq)$ is a complete lattice, i.e. the infinum and suprenum exist for every set of concepts. The following rules hold true for every family (A_i, B_i), $i \in I$ of concepts:

$$\begin{aligned} &\vee_{i \in I} (A_i, B_i) = (\cap_{i \in I} A_i, (\cup_{i \in I} B_i)''), \\ &\wedge_{i \in I} (A_i, B_i) = ((\cup_{i \in I} A_i)'', \cap_{i \in I} B_i) \end{aligned} \tag{9}$$

where A'' denotes the *closure* of the set A and it is defined as derivation of the derivated set:

$$A'' = (A')'. \tag{10}$$

Using these definitions and rules, some other important and interesting rules may be derived. Some of the derived rules are given in the following list:

$$\begin{aligned} &A_1 \subseteq A_2 \Rightarrow A_2' \subseteq A_1', \\ &A \subseteq (A')', \\ &A' \subseteq ((A')')' \end{aligned} \tag{11}$$

The structure of the concept lattice can be used not only to describe the concepts hidden in the underlying data system, but it shows the generalisation relation among the

objects and it can be used for clustering purposes, too. A good description on the related chapters of the lattice theory can be found among others in [2].

3. Algorithms for Generating the Concept Set

As for every concept the extent part is determined unambiguously by the intent part, the generation of the intent parts is investigated only. In most data mining applications the intent parts are enough to generate the rules. The rules define a relation, an implication among the attributes, i.e. on the intent parts. The actual support set for the rules is usually not important.

Among the sophisticated concept set generation algorithms the Ganter's next closure algorithm [1] is probably the most widely known method. It is widely accepted by experts, that this algorithm is not only the best known but the most effective one, too [4]. The concepts are generated according to an ordering relation. Based on the indexing of the elements, the lexicographical ordering between the concepts is defined in the following way:

$$A < B \quad \Leftrightarrow \exists\, a_i \in G: A <_i B \tag{12}$$

where

$$A <_i B \quad \Leftrightarrow a_i \in B \backslash A\,,\; A \cap \{a_1,\ldots,a_{i-1}\} = B \cap \{a_1,\ldots,a_{i-1}\} \tag{13}$$

This method calculates the extent part first, and the intent part is generated from the extent part. The key function element, the next extent routine, tests several extent variants until it finds an appropriate one. The total asymptotic cost of the algorithm is equal to

$$O(CN^2\sigma + CN^2M) \tag{14}$$

where

C : the number of concepts in the concept set, and
σ is a cost unit.

Regarding the efficiency of this algorithm and the objectives, some facts should be taken into consideration:

1. the disk IO cost may be very high if N is high;
2. the total cost is proportional to N^2, so it will be resulted in high costs for contexts with large number of objects, as it is assumed in our investigation.

One of the main characteristics of the Ganter's algorithm is that it accesses the context table several times during the generation of a concept. As the same context table element is accessed several times it is clear, this method assumes that

a: all parts of the context table are present at the concept set generation;
b: the context table can fit into the memory with low cost access operations.

Based on these assumptions, this method is called a batch method. A different kind of approach is presented by Godin [2]. His proposal is an incremental concept formation method, where the concept set is updated in an incremental manner, i.e. the set is updated

when the context table is extended by a new object instance. In this kind of method, every context table element is accessed only once, yielding a minimal IO cost. The building of the concept set in incremental mode is based on the following rule:

> *Every new concept intent after inserting a new object into the context, will be the result of intersecting the attribute set of the new object with some intent set already present in the concept set.*

Godin's method can be used for updating the concept set after insertion of a new object into the context. The algorithm consists of the following main steps. First, the concepts are partitioned into buckets based on the cardinality. Next, the buckets are processed in ascending cardinality order. Every intent in the current bucket is intersected with the intent set of the new object. If the result set is not present in the concept set, it will be added. The cost estimation for the algorithm can be given by

$$O(N\sigma + CNDM). \tag{15}$$

This formula assumes linear existence testing. Linear testing was implemented in the algorithm as testing can be reduced to the subset of the so called marked elements. The marking test can be performed only in linear mode. In the cost estimation formula D denotes the number of elements with a mark. This mark is assigned to the elements generated in the current phase. Comparing this cost function with the cost estimation of the next closure method, we can see that the incremental method will be more efficient if

1: the σ cost unit is high;

2: or N is high.

On the other hand, the cost of Godin's method is more sensitive to the C size of the concept set.

Beside these two basic concept set generation algorithms, there are some other proposals in the literature, mainly some kind of optimisation of the basic algorithms. From these papers, only some of the most recent ones will be presented here to demonstrate the computational efficiency of the most up-to-date variants.

In the paper of Hu [3], the concept set generation process is coupled with the calculation of the support value in order to discover association rules from the concept lattice. The concept set building part is based on the incremental method of Godin, thus resulting the same asymptotic calculation cost estimation value:

$$O(N\sigma + CNDM). \tag{16}$$

Another proposal is the Titanic algorithm, presented in [7]. This method uses the support values of the different attribute sets to determine the concept intents. It generates the candidate generator sets in increasing order of the size. A set is called a generator set if its closure is a concept intent and it is minimal, i.e. it does not contain any other generators for the same concept intent. The method processes first the one-attribute-long candidates and after then generates the candidate sets for the next level. At the next level, the length of the intents is increased by one

$$O(NM\sigma + aMCN + a^2C^2M). \quad (17)$$

The algorithm processes not only the concepts, but all of the candidates, thus in the cost estimation formula, a denotes how many times the number of candidates is larger than the number of concepts. This value is always greater than 1. The most costly part of the algorithm is the generation of candidate sets. In this phase, every pair at level l having the same values in the first (l-1) attributes will be processed to generate a new candidate set at level l.

The proposal of Lindig given in [4], is aimed at not only the generation of the concept set but on the building of the whole concept lattice. If we consider now only the concept set generation part of the algorithm, this method is related to the Ganter's method in many aspects. It assumes a lectical ordering among the concepts and the concepts are processed according to this ordering. The method also generates for every new concept the set of upper neighbour concepts to use this kind of information during the insertion into the concept lattice.

The neighbours of a concept are generated using the closure operation for the candidate neighbour attribute sets. At every call of the neighbour routine the full context table is scanned. The cost estimation of this algorithm is

$$O(Nc\sigma + CN^2M). \quad (18)$$

Thus the asymptotic complexity is the same as for the Ganter's method.

The aim of the investigation was to find an efficient algorithm that can be used for cases with large context size, so the proposals found in the literature were evaluated using the following criteria:

1. the disk IO should be minimal, every context table should be accessed only once and
2. the in-memory operations should be optimised to omit the redundant calculations.

Based upon these selection criteria, the incremental method is the best solution as it has only a linear disk IO cost and not all elements of the context table should be available at the beginning of the concept set building. To achieve a better performance, the objective was to improve the in-memory operations of the existing incremental methods. In the next sections of the paper a fine-tuned version of the incremental concept set building method is presented and the efficiency of the proposed method is also demonstrated with comparison tests. Based upon the test results, we can say that the incremental methods can outperform the batch method in practical applications. This result is in consonance with the results of Godin.

4. Fine tuned incremental method

According to the properties of the incremental methods, the context table is generated by adding single objects one by one, after each other. Let's denote the intent part of the concept set built up from the first k objects by

L_k

The L_{k+1} is constructed from L_k and a_k where a_k denotes the k-th object in the input list. The generation of L_k is based on the following considerations that can be proven very easily from the basic properties of the concept lattices, so we omit here the proofs.

Proposition 1.

$$\text{for every } a_k \in G : A(a_k) \in L_k \tag{19}$$

where $A(a)$ denotes the attribute set of object a.

Proposition 2.

$$\text{for every } A,B \in L_k \Rightarrow A \cap B \in L_k. \tag{20}$$

Proposition 3.

$$\text{for every } A \in L_{k+1}, A \neq A(a_k), A \notin L_k : \exists B \in L_k : A = A(a_{k+1}) \cap B. \tag{21}$$

Based on these propositions, in every iteration loop starting with L_k, the $A(a_k)$ can be added first to L and then the intersections of $A(a_k)$ with the elements already present in the L_k are generated and inserted into L_{k+1}. Since the number of possible pairs for an intersection is very large, the algorithm has a high cost in testing the intersections. A possibility of cost reduction is provided by the fact that not every pair generates a new concept intent. Most of the pairs yield in an existing intent value. The key point for fine tuning of the incremental algorithm in our proposal is based on the following simple considerations:

Proposition 4.

$$\text{for every } A,B,C \in L_k, A \cap B = 0, C \subseteq A: \; C \cap B = 0. \tag{22}$$

The meaning of this rule is for us the following: if A is disjoint with some B then all of its subsets can be pruned from testing.

Proposition 5.

$$\text{for every } A,B,C \in L_k, A \subseteq B \,, C \subseteq A: C \subseteq B \text{ and } C \cap B = C. \tag{23}$$

Thus, if an intent part is a subset of the tested element, then all its subsets can be eliminated.

Proposition 6.

$$\text{if } \exists B \in L_k, A(a_k)) = B \text{ then for } \forall C \in L_k: C \cap A(a_k) \in L_k. \tag{24}$$

Thus, if an intersect part is presented in the concept set, then the whole testing loop for $A(a_k)$ can be eliminated.

To implement the cost reduction elements into the concept set building algorithm, the following modifications of the basic incremental method were developed:

1. Before the testing loop for the new incoming object, it is tested whether it equals an already existing intent part. The test for existence checking is performed using a B-tree structure, resulting in a test cost of $O(log(C))$.
2. Before the intersects of the elements would be inserted into the concept set, the candidate elements (the results of the intersections) are stored in a hash table, so the sets generated repeatedly can be detected in a cost effective way.
3. To reduce the redundant intersection tests, the elements of the concept set are stored in a special pointer list where the elements containing a given attribute value are connected to each other. The intersection operation should be performed only for elements having at least one common attribute. Thus the intersection test for disjoint elements can be eliminated.
4. To eliminate the insertion testing for intents already present in the concept set, during the intersection phase, a special marker is used in the hash table. In most cases, the existence can be detected in the hash table building phase, before the insertion phase.

Based on these considerations, the structure of the algorithm is

```
L1  loop
      read_next(a)
      find_in_Btree(a)
C1    if not found() then
            update_pointer_chain(a)
            insert_set(a)
L2          foreach X (element of) L(k), X (intersect) A(a) not = 0 do
              Y = X (intersect) A(a)
              insert_hash(Y)
            endfor
L3          foreach X elements in hash do
              if it is not marked then
            insert_set(X)
              endif
            endfor
      endif
    until (no more input)
```

The estimated cost of the proposed algorithm is calculated in the following way.

read_next()	$O(\sigma)$
find_in_Btree	$O(Mlog(C))$
update_pointer_chain	$O(M)$
insert_hash	$O(M)$
intersect	$O(M)$
L1 loop	it is executed for every object, so the number of iterations is equal to N

L2 loop	it is executed for every intent set having common attribute with the new object, the number of iterations is equal to C' where $C' < C$
L3 loop	the insert operation is performed only if the intersection result is not marked, the number of iterations: $C'' << C$
C1 branching	the inner part is executed for objects with a new attribute set, $N' < N$

The total cost can be given by

$$O(N\{\sigma + MlogC)\} + N'\{M + MlogC + C'\{M\} + C''\{MlogC\}\}). \tag{25}$$

This expression can be transformed into a simpler form

$$O(N\sigma + NMlogC + N'MlogC + N'C'M + N'C''MlogC) \tag{26}$$

and pruning the non-dominant tags:

$$O(N\sigma + NMlogC + N'MlogC + N'C''MlogC). \tag{27}$$

One of the benefits of this algorithm is that it can reduce the computational cost if the new object has an attribute set contained in the context already. In applications this case may occur often, for example in processing questionnaires where several people may give the same answer. The C' value is the number of sets having intersection with the new object's attribute set. A rough estimation for C' can be given as follows:

Let's denote the length of the attribute set of the objects by K. For $K = 1$, the probability that it has no common part with a subset of M is equal to

$$2^{M-1} / 2^M = ½ \tag{28}$$

so

$$P_1 = ½. \tag{29}$$

In a similar way we get, that

$$P_i = 1/2^i \tag{30}$$

The number of subsets having length i is equal to

$$\binom{M}{i}. \tag{31}$$

So the probability that an arbitrary subset has no common part with an other subset is

$$P = \Sigma_i \binom{M}{i} P_i / 2^M = (\Sigma_i \binom{M}{i} 1/2^i) / 2^M = (3/4)^M \quad (32)$$

Although, the relative gain of using this intersection pointer list is lower for large M values, the absolute number of testing that can be omitted is large enough to use this kind of optimisation in the applications.

Regarding the C'' value, the gain here can be more dominant as the number of pruned sets is much higher. If c_k is the number of concepts after inserting the k-th object, then the following holds true:

$$1 = c_1 < c_2 < \ldots < c_N = |C|. \quad (33)$$

The total number of insertion testing without marking is

$$c_1 + c_2 + \quad + c_N. \quad (34)$$

So, the gain of the reduction is equal to the difference

$$c_1 + c_2 + \quad + c_N - |C|. \quad (35)$$

A rough estimation for c_k can be

$$O(k^2) \quad (36)$$

so this reduction step is very important.

5. Algorithms for Building Concept Lattices

Let's denote the set of concepts and the ordering relation on this set by $(\Phi, \leq)$. For any arbitrary concept C, the upper and lower neighbour can be defined in the following way. An $L \in \Phi$ is a lower neighbour of C if

$$L \leq C \text{ and } !\exists X \in \Phi : L \leq X \leq C. \quad (37)$$

The upper neighbour can be defined in a similar way. In a lattice an element may have several upper and lower neighbour elements. We denote the set of lower (upper) neighbours for C by $Low(C)$ and $Upp(C)$. For building the lattice $Low(C)$ and $Upp(C)$ must be known for every C.

The naive way to generate $Upp(C)$, $Low(C)$ is to test all of the concept pairs. The main structure of the algorithm for $Upp(C)$ consists of two nested loops to test every concept pair. If one of them is the ancestor of the other then it should be tested whether there is another element being between these two elements. So, this algorithm contains three nested loops and the cost of the execution can be estimated by the following formula:

$$O(C^2 C_u M) \quad (38)$$

where C_u is the number of upper neighbours. Comparing this estimation with the cost values for generating the concept set, we can see that this cost is of the same magnitude or sometimes higher than the cost of the first phase. This short evaluation shows the importance of an optimised lattice building method.

In the literature, we can find several approaches addressing this problem. Only the most recent ones are described here.

The proposal of Ky Hu [3] was published in 1999. The first phase of this method is based on Godin's incremental lattice generation method. The algorithm generates the concepts in increasing cardinality order of the intent part. According to this principle, the parents are generated first, and only after that come the children. This means, that during insertion of a new concept into the lattice only the parent neighbours, the ancestor part of the lattice should be tested. The lower neighbours will be determined during the insertion of the children elements. The concepts of smaller intent size are all tested to find the potential parents. For every potential parent, the set of its lower neighbours is tested whether they are parents of the new concept. If the candidate element is an upper neighbour, then all nodes marked as upper neighbour previously and being an upper neighbour of the tested element should be removed from the set of marked nodes. At the end, the marked elements will constitute the upper neighbour set.

The main optimisation elements presented in [3] are

- only a subset of concepts is tested during the search for potential parents,
- the test for pruning elements marked previously is reduced to a special subset of elements.

The cost estimation of the algorithm can be given by

$$O(CC_nC_aM) \tag{39}$$

where

C_a : number of ancestor nodes,
C_n : number of neighbour nodes.

Another current approach is the algorithm of Lindig presented in [4]. The proposed concept set and lattice building algorithm is related to the Ganter's method in many aspects. The concepts from the concept set are processed in total order to make sure all concepts that are inserted into the lattice are also considered for their neighbours. The lectical order used in Ganter's method is an appropriate ordering. Due to this processing order only the upper neighbour set is needed to be generated here, too. The test for upper neighbours is based on extending the extent part with a new element and performing a closure operation. The cost estimation for this algorithm can be given by

$$O(CN^2M), \tag{40}$$

i.e. it has the same asymptotic cost value as the next closure method has.

The method presented in [4] performs an element-wise update of the lattice according to a linear extension of the lattice order. The lattice extension is done in a top-down manner, starting from the top node and processing the rest of the nodes according to a total order which is a linear extension of the lattice order. At each step the current element is

connected to each of its immediate successors in the final lattice. During the building of the lattice a special subset of elements, the so called border elements play an important role. The border of a lattice Φ is defined as

$$Border(\Phi) = \{ C_i \in \Phi \mid \forall C_i' \in (\Phi \setminus 0), C_i \leq C_i \Rightarrow C_i' = C_i \} \quad (41)$$

where C_i and C_i' denote concepts in the lattice. During the insertion of a new X concept, the border will change. The border set always contains the new element, whereas all elements of the old border that are greater than X are dropped out. The cost estimation of this method is

$$O(CC'^2M) \quad (42)$$

where C' denotes the number of elements in the border region.

Based upon the proposals mentioned here, we can see that every optimisation approach is based on the following considerations:

- the lattice should be built up in a top-down (or bottom-up) manner so only the elements of the upper or lower neighbourhood are to be localised;
- the search for elements of $Upp(C)$ or $Low(C)$ are performed on only a subset of the whole lattice.

This kind of optimisation method requires more or less meta-data structures with significant administration cost. In the next section another approach for optimisation of concept lattice building is introduced, that is based on a simple insertion.

6. Efficiency Analysis of the simple lattice building algorithm

Let us consider now a simple lattice building algorithm which locates the elements of $Upp(X)$ and $Low(X)$ for an arbitrary X by using a simple top-down or bottom-up lattice scanning method. The search starts at the top (bottom) node traversing the concepts being an ancestor or descendant of X. The ancestor nodes having no child with this property are the elements of the neighbourhood. The search for neighbours can be defined in a recursive way:

```
search_upp(Y,X)
 c=0
 foreach C child of Y do
   if C > X then
     search_upp(C,X)
     c++
   endif
 endfor
 if c = 0 then
  Upp(X) = Upp(X) + Y
 endif
```

where X is the new concept's upper neighbour which we are searching for. Y is the tested lattice element.

During the tests with different input orders we became aware of another and more important factor for efficiency, namely the parent-child relationships among the elements. The rule is the following: in the search processes for upper or lower neighbour elements, the cost of lattice traversing depends on the number of nodes to be processed and not on the number of ancestors or descendants. The number of tested nodes is greater than the number of ancestors or descendants as there are a large number of nodes with negative test results. These nodes are located on the border of the ancestors' or descendants' sub-lattice. These elements are children of the ancestors (or parents of the descendants) but they themselves are not ancestors (descendants) of the new element. According to our test results, the cost for processing these border elements can be very high and it depends dominantly on the position and insertion ordering of the X nodes. The aim of our investigation was to find an optimal order of concept insertion yielding a low computational cost.

During the search for upper neighbour nodes, at every ancestor node, all of the lower neighbour elements are tested. A similar statement is true for the search for lower neighbouring nodes. The cost of insertion for an arbitrary C concept is proportional to the number of nodes to be tested in both directions:

$$Cost_C = \Sigma_{n \in SAC} NC_n + \Sigma_{n \in SDC} NP_n \tag{43}$$

where

NC_C : the number of lower neighbour nodes at C,
NP_C : the number of upper neighbour nodes at C,
SA_C : the set of ancestors of C,
SD_C : the set of descendants of C.

The total cost for building the concept lattice is

$$Cost = \Sigma_{C \in \Phi} Cost_C \,. \tag{44}$$

After inserting a new element into a lattice, the *NC, NP, SA, SD* values may change, so the *NC, NP, SA, SD* parameters are a function of the discrete time value. Let's denote this time value by i which is a simple sequence number, thus we get

$$Cost = \Sigma^{\Phi}_{i=1} (\Sigma_{n \in SAii} NC_{n,i} + \Sigma_{n \in Sdii} NP_{n,i}) \tag{45}$$

where *SAni, SDni* denote the set of ancestors (descendants) of the concept n at the time point i. Similarly, the i index for *NC* and *NP* denotes the value at the time point i. At the end of the building process, the Φ lattice is built up completely. This resulting lattice does not depend on the insertion order of the concepts. So for every $C \in \Phi$, *NC, NP, SA, SD* have a given, fixed value. But for any $0 < i < |\Phi|$ point of time, these values may be unknown. Any of the functions may increase or decrease during the building phase.

Summarising these considerations, the rule of optimisation can be formulated as follows:

> ***The larger the number of elements below (above) an x element is, the longer the number of lower neighbour nodes (upper neighbour nodes) for x should remain on such a low value as is possible.***

This rule implies the following rule that can be implemented easier in the practice:

The elements with low ancestor (descendant) population should get a new upper (lower) neighbour first.

To provide a feasible method for this problem, a cost saving heuristic optimisation method based on the previous considerations is introduced.

The proposed heuristic method builds an approximate tree structure for the lattice. This can be considered as a spanning tree. This tree can be generated in an efficient way and the elements are inserted into the lattice in the order based on the hierarchy structure of this tree. The information to build this tree can be gathered during the concept set generation phase. The tree is processed in a top-down traversing and every processed element will be inserted into the lattice according to the order of traversing. The spanning tree is generated in the concept set building stage, during the intersection generation phase. The basic consideration behind the tree construction algorithm is the following. If

$$A = B \cap C$$

then B and C contain A, so B and C are candidate parents of A. The candidate parent concept with minimal length (the number of attributes not present in A is minimal) is selected as the parent element of A in the spanning tree.

7. Test Results

To compare the efficiencies of the different approaches, the different lattice building methods were implemented in a test system. The implementation programming language was the Java to create a platform independent solution. In the first phase the different lattice set building algorithms were tested. According to the test results the proposed fine-tuned method provides the best cost values among the tested ones. The next table summarises some results related to two different concept sets.

Table 1: Experimental test results for concept set generation

Method	Elapsed time	Size of the concept set
naive method	245	3865
Ganter method	14	
Godin method	6	
proposed method	3	
naive method	795	37344
Ganter method	170	
Godin method	83	
proposed method	42	

In the second phase the lattice building algorithms were tested. Taking the simple lattice building algorithm presented in the previous section, the proposed approximation tree ordering resulted in a better result than the other methods. It is an interesting

experience, that in the case of insertion ordering based on intent size the results are always worse than with random ordering. The next closure method that is also based on special ordering, is also worse than an average random ordering method. The following table shows the computational cost in elapsed time and in number of performed set operations.

Table 2: Experimental test results for lattice building

Method	Elapsed time	Number of operations
spanning tree order	10	7068255
lattice top-down traversing order	9	6093848
Keyun method's order	12	7136129
normal intersection order	13	8181065
random order	23	17020450
next closure order	40	38784364
increasing set size order	71	49308966
Valtchev method's order	98	56898773

In the tests, the size of the concept lattice is between 100 and 12000. All of the contexts were generated randomly.

We should mention that these results are based on the simple insertion algorithm. The methods mentioned in Table 2 are usually based on modified lattice building algorithms which include some kind of heuristic elements, too. In the next closure method, for example, the searching phase for the descendants is omitted as the current incoming element is always the smallest one without any descendants. Eliminating this step, the total cost can be significantly reduced. Thus, the result values in the table are related only to the insertion order, the original lattice building algorithms can provide better cost values. The aim of this investigation was only to analyse the effect of different insertion orders during the lattice building algorithm.

Acknowledgements

This work has been supported by the Hungarian Eötvös State Fellowship Grant No.: MÖB 595-1-2001.

REFERENCES

1. GANTER, B., WILLE, R.: *Formal Concept Analysis*: *Mathematical Foundations*, Springer Verlag, 1999.
2. GODIN, R., MISSAOUI, R., ALAOUI, H.: *Incremental concept formation algorithms based on Galois lattices*, Computational Intelligence, 11(2), 1995, pp. 246-267.
3. HU, K., LU, Y., SHI, C: *Incremental Discovering Association Rules*: *A Concept Lattice Approach*, Proceedings of PAKDD99, Beijing, 1999, pp. 109-113.
4. LINDIG, C.: *Fast Concept Analysis*, Proceedings of the 8th ICCS, Darmstadt, 2000.
5. NOURINE, L., RAYNAUD, O.: *A Fast Algorithm for Building Lattices*, Information Processing Letters, 71, 1999, pp. 197-210.
6. RADELECZKI, S., TÓTH, T.: *Fogalomhálók alkalmazása a csoporttechnológiában*, OTKA kutatási jelentés, Miskolc, Hungary, 2001.

7. STUMME, G. , TAOUIL, R., BASTIDE, Y., PASQUIER, N., LAKHAL, L.: *Fast Computation of Concept Lattices Using Data Mining Techniques,* 7th International Workshop on Knowlegde Representation meets Databases (KRDB 2000), Berlin, 2000.
8. ZAKI, M., OGIHARA, M. *Theoretical Foundations of Association Rules*, Proceedings of 3 rd SIGMOD'98 Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD'98), Seattle, Washington, USA, June 1998.
9. KOVACS, L.: *Efficiency Analysis of Building Concept Lattice*, Proceedings of 2nd ISHR on Computational Intelligence, Budapest, 2001.

Production Systems and Information Engineering, Miskolc, Vol. 1 (2003), pp. 107–124

FUZZY Q-LEARNING IN SVD REDUCED DYNAMIC STATE-SPACE

SZILVESZTER KOVÁCS*
Department of Information Technology, University of Miskolc,
Miskolc-Egyetemváros, Miskolc, H-3515, Hungary
szkovacs@iit.uni-miskolc.hu

Péter BARANYI*
Department of Telecommunication and Telematics, Technical University of Budapest,
Pázmány Péter sétány 1/d, B223, Budapest, H-1117, Hungary
baranyi@ttt.bme.hu

[Received May 2002 and accepted April 2003]

*Intelligent Integrated Systems Japanese Hungarian Laboratory,
Budapest University of Technology and Economics, Hungary

Abstract. Reinforcement Learning (RL) methods, surviving the control difficulties of the unknown environment, are gaining more and more popularity recently in the autonomous robotics community. One of the possible difficulties of the reinforcement learning applications in complex situations is the huge size of the state-value- or action-value-function representation [17]. The case of continuous environment (continuous valued) reinforcement learning could be even complicated, as the state-value- or action-value-functions are turning into continuous functions. In this paper we suggest a way for tackling these difficulties by the application of SVD (Singular Value Decomposition) methods [6], [19], [20].

Keywords: Reinforcement Learning, Fuzzy Q-Learning, Singular Value Decomposition

1. Introduction

Reinforcement learning methods are trial-and-error style learning methods adapting dynamic environment through incremental iteration. The principal ideas of reinforcement learning methods, the dynamical system state and the idea of "optimal return" or "value" function are inherited from optimal control and dynamic programming [7]. One common goal of the reinforcement learning strategies is to find an optimal policy by building the state-value- or action-value-function [17]. The state-value-function $V^{\pi}(s)$, is a function of the expected return (a function of the cumulative reinforcements), related to a given state $s \in S$ as a starting point, following a given policy π. Where the states of the learning agent are observable and the reinforcements (or rewards) are given by the environment. These rewards are the expression of the goal of the learning agent as a kind of evaluation follows the recent action (in spite of the instructive manner of error feedback based approximation techniques, like the gradient descent training). The policy is the description of the agent

behaviour, in the form of mapping between the agent states and the corresponding suitable actions. The action-value function $Q^{\pi}(s,a)$ is a function of the expected return, in case of taking action $a \in A_s$ in a given state s, and then following a given policy π. Having the action-value-function, the optimal (greedy) policy, which always takes the optimal (the greatest estimated value) action in every state, can be constructed as [17]:

$$\pi(s) = \arg\max_{a \in A_s} Q^{\pi}(s,a). \tag{1}$$

(Where the function arg is standing for the indexes of the set of possible actions.)

Namely for estimating the optimal policy, the action-value function $Q^{\pi}(s,a)$ is needed to be approximated. In discrete environment (discrete states and discrete actions) it means, that at least $\sum_{s \in S} \|A_s\|$ element must be handled. (Where $\|A_s\|$ is the cardinality of the set of possible actions in state s.) Having a complex task to adapt, both the number of possible states and the number of the possible actions could be an extremely high value.

1.1 Reinforcement Learning in Continuous Environment

To implement reinforcement learning in continuous environment (continuous valued states and actions), function approximation methods are widely used. Many of these methods are applying tailing or partitioning strategies to handle the continuous state and action spaces in the similar manner as it was done in the discrete case [17]. One of the difficulties of building an appropriate partition structure (the way of partitioning the continuous universe) is the anonymity of the action-value-function structure. Applying fine resolution in the partition leads to high number of states, while coarse partitions could yield imprecise or unadaptable system. Handling high number of states also leads to high computational costs, which could be also unacceptable in many real time applications

There are many methods in the literature for applying fuzzy techniques in reinforcement learning (e.g. for "Fuzzy Q-Learning" [1], [8], [9], [11], [12]). One of the main reasons of their application beyond the simplicity of expressing priory knowledge in the form of fuzzy rules is the universal approximation property [10], [22] of the fuzzy inference. It means that any kind of function can be approximated in an acceptable level, even if the analytic structure of the function is unknown. Despite of this useful property, the use of fuzzy inference could be strictly limited in time-consuming reinforcement learning by its complexity problems [13], because of the exponential complexity problem of fuzzy rule bases [5], [20]. Fuzzy logic inference systems are suffering from exponentially growing computational complexity in respect to their approximation property. This difficulty comes from two inevitable facts. The first is that the most adopted fuzzy inference techniques do not hold the universal approximation property, if the numbers of antecedent sets are limited, as stated by Tikk in [18]. Furthermore, their explicit functions are sparse in the approximation function space. This fact inspires to increase the density, the number of antecedents in pursuit of gaining a good approximation, which, however, may soon lead to a conflict with the computational capacity available for the implementation, since the increasing number of antecedents explodes the computational requirement. The latter is the second fact and stated by *Kóczy* et al. in [13]. The effect of this contradiction is gained by the lack of a mathematical framework capable of estimating the necessary minimal number

of antecedent sets. Therefore a heuristic setting of the number of antecedent sets is applied, which usually overestimates, in order to be on the safe side, the necessary number of antecedents resulting in an unnecessarily high computational cost. E.g. the structurally different Fuzzy Q-Learning method implementations introduced [8], [9], [11] and [12] are sharing the same concept of fixed, predefined fuzzy antecedent partitions, for state representation. One possible solution for this problem is suggested in [1]. By introducing "Adaptive State Partitions", an incremental fuzzy clustering of the observed state transitions. This method can lead to a better partition than the simple heuristic, by finding the best fitting one in respect to the minimal squared error, but still has the problem of limited approximation property inherited from the limited number of antecedent fuzzy sets.

Another promising solution, as a new topic in fuzzy theory, is the application of fuzzy rule base complexity reduction techniques.

1.2 Fuzzy rule base complexity reduction

The main goal of introducing fuzzy rule base complexity reduction techniques in reinforcement learning is enhancing the universal approximation property of the fuzzy inference by extending the number of antecedent sets while the computational complexity is kept relatively low. SVD based fuzzy approximation technique was initialized in 1997 by Yam [19], which directly finds a minimal rule-base from sampled values. Shortly after, this concept was introduced as SVD fuzzy rule base reduction and structure decomposition in [2], [20]. Its key idea is conducting SVD of the consequents and generating proper linear combinations of the original membership functions to form new ones for the reduced set. An extension of [21] to multi-dimensional cases may also be conducted in a similar fashion as the Higher Order SVD (HOSVD) reduction technique proposed in [5], [19], [20]. Further developments of SVD based fuzzy reduction are proposed in [3], [5] and its extension to the generalized inference forms are proposed in [14], [15], [16].

The key idea of using SVD in complexity reduction is that the singular values can be applied to decompose a given system and indicate the degree of significance of the decomposed parts. Reduction is conceptually obtained by the truncation of those parts, which have weak or no contribution at all to the output, according to the assigned singular values. This advantageous feature of SVD is used in this paper for enhancing the universal approximation property of the fuzzy inference by extending the number of antecedent sets while the computational complexity is kept relatively low. The complexity and its reduction is discussed in regard of the number of rules, which result simplicity in operating with the rules, in reinforcement learning methods.

On the other hand, as one of the natural problems of any complexity reduction technique, the adaptivity property of the reduced approximation algorithm becomes highly restricted. Since the crucial concept of the Fuzzy Q-learning is based on the adaptivity of the action-value function this paper is aimed propose to adopt an algorithm [6] capable of embedding new approximation points into the reduced approximation while the calculation cost is kept (where the calculation cost could be defined in the terms of the number of product operations done during the calculation).

2. Fuzzy Q-Learning

For introducing a possible way of application of SVD complexity reduction techniques in Fuzzy Reinforcement Learning, a simple direct (model free) reinforcement learning method, the Fuzzy Q-Learning, was chosen.

The goal of the Q-learning is to find the fixed-point solution Q of the Bellman Equation [7] through iteration. In discrete environment *Q-Learning* [23], the action-value-function is approximated by the following iteration:

$$Q_{i,u} \approx \tilde{Q}_{i,u}^{k+1} = \tilde{Q}_{i,u}^{k} + \Delta\tilde{Q}_{i,u}^{k+1} = \tilde{Q}_{i,u}^{k} + \alpha_{i,u}^{k} \cdot \left(g_{i,u,j} + \gamma \cdot \max_{v \in U} \tilde{Q}_{j,v}^{k+1} - \tilde{Q}_{i,u}^{k} \right), \quad \forall i \in I, \forall u \in U \tag{2}$$

where $\tilde{Q}_{i,u}^{k+1}$ is the $k+1$ iteration of the action-value taking the u^{th} action A_u in the i^{th} state S_i, S_j is the new (j^{th}) observed state, $g_{i,u,j}$ is the observed reward completing the $S_i \to S_j$ state-transition, γ is the discount factor and $\alpha_{i,u}^{k} \in [0,1]$ is the step size parameter (which can change during the iteration steps), I is the set of the discrete possible states and U is the set of the discrete possible actions.

For applying this iteration to continuous environment by adopting fuzzy inference (Fuzzy Q-Learning), there are many solutions exist in the literature [1], [8], [9], [11], [12]. Having only demonstrational purposes, in this paper one of the simplest one, the order-0 Takagi-Sugeno Fuzzy Inference based Fuzzy Q-Learning is studied (a slightly modified, simplified version of the Fuzzy Q-Learning introduced in [1] and [12]). This case, for characterising the value function $Q(s,a)$ in continuous state-action space, the order-0 Takagi-Sugeno Fuzzy Inference System approximation $\tilde{Q}(s,a)$ is adapted in the following manner:

$$\textbf{If } s \textbf{ is } S_i \textbf{ And } a \textbf{ is } A_u \textbf{ Then } \tilde{Q}(s,a) = Q_{i,u},\ i \in I, u \in U, \tag{3}$$

where S_i is the label of the i$^{\text{th}}$ membership function of the n dimensional state space, A_u is the label of the u^{th} membership function of the one dimensional action space, $Q_{i,u}$ is the singleton conclusion and $\tilde{Q}(s,a)$ is the approximated continuous state-action-value function. Having the approximated state-action-value function $\tilde{Q}(s,a)$, the optimal policy can be constructed by function (1).

Setting up the antecedent fuzzy partitions to be *Ruspini partitions*, the order-0 Takagi-Sugeno Fuzzy Inference forms the following approximation function:

$$\tilde{Q}(s,a) = \sum_{i_1,i_2,\ldots,i_N,u}^{I_1,I_2,\ldots,I_N,U} \prod_{n=1}^{N} \mu_{i_n,n}(s_n) \cdot \mu_u(a) \cdot q_{i_1 i_2 \ldots i_N u} \tag{4}$$

where $\tilde{Q}(s,a)$ is the approximated state-action-value function $\mu_{i_n,n}(s_n)$ is the membership value of the i_n^{th} state antecedent fuzzy set at the n^{th} dimension of the N dimensional state antecedent universe at the state observation s_n, $\mu_u(a)$ is the membership value of the u^{th} action antecedent fuzzy set of the one dimensional action antecedent universe at the action selection a and $q_{i_1 i_2 \ldots i_N u}$ is the value of the singleton conclusion of the $i_1, i_2, \ldots, i_N, u^{\text{-th}}$ fuzzy

rule. (A fuzzy partition is a *Ruspini partition* if the sum of the membership values of the member sets of the partition is equal to one for the entire universe of discourse: $\sum_{i}^{I} \mu_i(x) = 1$ for $\forall x \in X$, where $\mu_i(x)$ is the membership function of the i^{th} fuzzy set of the I element fuzzy partition on the universe of discourse X – see e.g. on Fig.1.a)

Applying the approximation formula of the Q-learning (2) for adjusting the singleton conclusions in (4), leads to the following function:

$$q^{k+1}_{i_1 i_2 \ldots i_N u} = q^{k}_{i_1 i_2 \ldots i_N u} + \prod_{n=1}^{N} \mu_{i_n,n}(s_n) \cdot \mu_u(a) \cdot \Delta\widetilde{Q}^{k+1}_{i,u} \tag{5}$$

$$q^{k+1}_{i_1 i_2 \ldots i_N u} = q^{k}_{i_1 i_2 \ldots i_N u} + \prod_{n=1}^{N} \mu_{i_n,n}(s_n) \cdot \mu_u(a) \cdot \alpha^{k}_{i,u} \cdot \left(g_{i,u,j} + \gamma \cdot \max_{v \in U} \widetilde{Q}^{k+1}_{j,v} - \widetilde{Q}^{k}_{i,u} \right)$$

where $q^{k+1}_{i_1 i_2 \ldots i_N u}$ is the $k+1$ iteration of the singleton conclusion of the $i_1 i_2 \ldots i_N u^{\text{th}}$ fuzzy rule taking action A_u in state S_i, S_j is the new observed state, $g_{i,u,j}$ is the observed reward completing the $S_i \rightarrow S_j$ state-transition, γ is the discount factor and $\alpha^{k}_{i,u} \in [0,1]$ is the step size parameter. The $\max_{v \in U} \widetilde{Q}^{k+1}_{j,v}$ and $\widetilde{Q}^{k}_{i,u}$ action-values can be approximated by equation (4).

3. Dynamic Partition Allocation

The next problematic question of the Fuzzy Reinforcement Learning, as it was introduced in Section 1, is the proper way of building the fuzzy partitions. The methods sharing the concept of fixed, predefined fuzzy partitions, like [8], [9], [11] and [12] are facing the following question: More detailed partitions are yielding exponentially growing state spaces (rule base sizes), elongating the adaptation time, and dramatically increasing the computational resource demand, while less detailed partitions (containing only a few member fuzzy sets) could cause high approximation error, or unadaptable situation. One possible solution for this problem is suggested in [1]. By introducing "Adaptive State Partitions", an incremental fuzzy clustering of the observed state transitions. This method can lead to a better partition than the simple heuristic, by finding the best fitting one in respect to the minimal squared error, but still has the problem of limited approximation property inherited from the limited number of antecedent fuzzy sets.

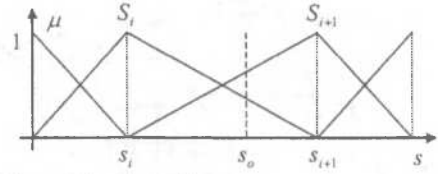
In this paper another dynamic partition allocation method is suggested, which is instead of adjusting the sets of the fuzzy partition, simply increase the number of the fuzzy sets by inserting new sets in the required positions. The main idea is very simple (see Fig.1. for an example). Initially a minimal sized (e.g. 2-3 sets only) Ruspini partition built up triangular shaped fuzzy sets on all the antecedent universes (see Fig.1.a). In the case when the action-value function update (5) is high (e.g. greater than a preset limit ε_Q: $\Delta\widetilde{Q} > \varepsilon_Q$), and the partition is not too dense already at that point (e.g. the distance of the cores of the surrounding fuzzy sets (d_s) is greater than a preset limit ε_s: $s_{i+1} - s_i = d_s > \varepsilon_s$), and the actual state-action point (s_o) is far from the existing partition members (e.g. the actual state-action point is closer to the middle than one of the surrounding fuzzy sets cores:

$\left|s_o - \frac{s_i+s_{i+1}}{2}\right| < \frac{d_s}{4}$) – see e.g. on Fig.1.b, then a new fuzzy state is inserted among the existing partition to increase the resolution (e.g. $s_{k+1}=s_k$, $\forall k > i$, $s_{i+1}=\frac{s_i+s_{i+2}}{2}$) – see e.g. on Fig.1.d.

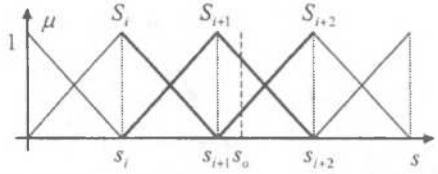
If the update value is relatively low ($\Delta\tilde{Q} \le \varepsilon_Q$, see e.g. Fig.2.), or the actual state-action point is close to the existing partition members ($\left|s_o - \frac{s_i+s_{i+1}}{2}\right| \ge \frac{d_s}{4}$, see e.g. Fig.3.), than the partition is staying unchanged. The state insertion is done in every state dimensions separately (in multidimensional case it means an insertion of a hyperplane), by interpolating the inserted values from the neighbouring ones (see Fig.1.e and Fig.4. as a two dimensional example). Having the new state plane inserted in every required dimension, the value update is done regarding to the Fuzzy Q-Learning method as it was introduced in Section 2, by the equation (5). (See e.g. on Fig.1.c, Fig.1.d, or Fig.4.d.)

The proposed dynamic partition allocation method has the property of local step-by-step refinement in a manner very similar to the binary search. It can locate the radical positions of the value action function with the precision of $d_s^{i+k} = \frac{d_s^i}{2^k}$ in k steps (where d_s^i is the starting precision).

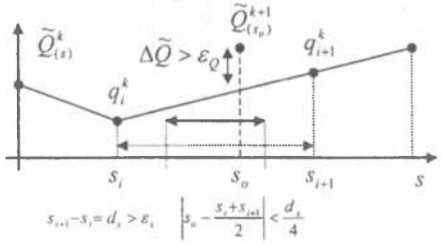
The main problem of the proposed simple dynamic partition allocation method is the non-decreasing adaptation manner of the antecedent fuzzy partitions. In some situation, it could mean rapidly increasing partition sizes in the sense of the number of the component fuzzy sets. Moreover, these cases also lead rapidly growing, or at least non-decreasing computational resource demand.



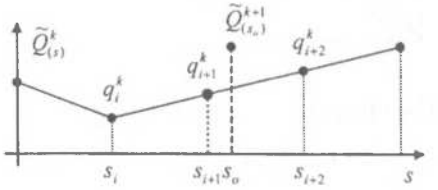
a, The original partition



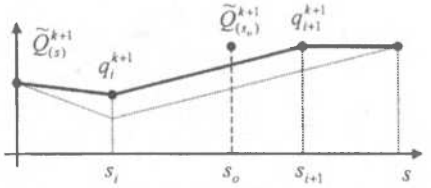
d, The modified partition (S_{i+1} is inserted)



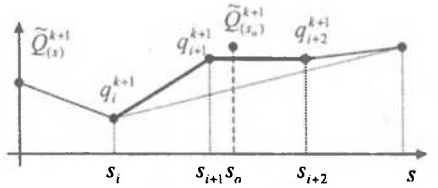
b, Next approximation of Q at s_o: $\tilde{Q}^{k+1}_{(s_o)}$



e, The inserted q^k_{+i} values are interpolated

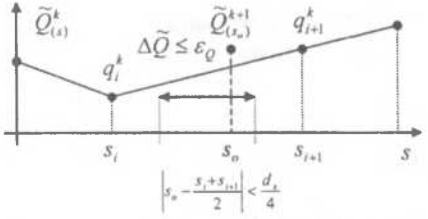


c, Next approximation, without partition modification



f, Next approximation, with partition modification

Fig. 1. The proposed dynamic partition allocation method.



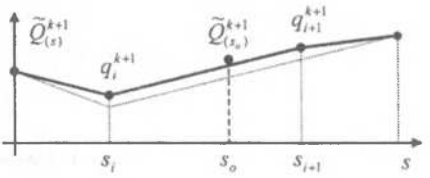
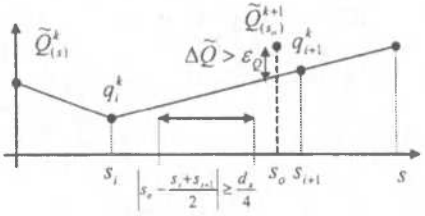


Fig. 2. The action-value function update is relatively low.



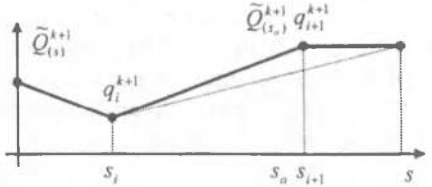
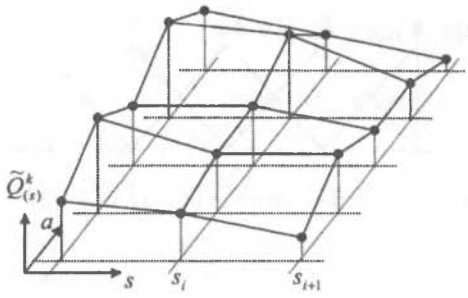
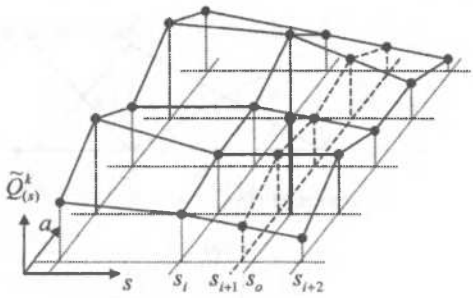


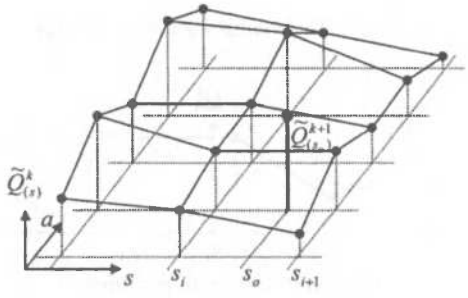
Fig. 3. The actual state-action point is close to the existing partition members.



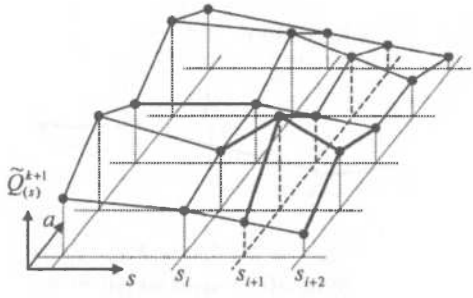
a, The original approximation of $\tilde{Q}^k_{(s)}$



c, State insertion s_{i+1}, by interpolating the inserted action values $a^k_{i+1,j}$, $\forall j \in [1,J]$, from the neighbouring $a^k_{i,j}$ and $a^k_{i+2,j}$ ones.



b, Next approximation of Q at s_2: $\tilde{Q}^{k+1}_{(s_0)}$



d, Next approximation, with state insertion s_{i+1}, and value update regarding to (7)

Fig. 4. The proposed dynamic partition allocation in two-dimensional (single state and action) antecedent case.

4. SVD based Complexity Reduction

For retaining the benefits of the dynamic partition allocation and maintaining the overall computational resource demand low, in this paper, the adoption of Higher Order SVD [5] based fuzzy rule base complexity reduction techniques and its fast adaptation method is suggested. The application of the fast adaptation method [6] gives a simple way for increasing the rule density of a rule base stored in a compressed form directly. Providing an economic sized structure for handling continuously increasing and varying rule bases, which is so typical in reinforcement learning.

4.1. SVD Based Fuzzy rule base complexity reduction

The essential idea of using SVD in complexity reduction is that the singular values can be applied to decompose a given system and indicate the degree of significance of the decomposed parts. Reduction is conceptually obtained by the truncation of those parts, which have weak or no contribution at all to the output, according to the assigned singular values. This advantageous feature of SVD is used in this paper for enhancing the universal approximation property of the fuzzy inference by extending the number of antecedent sets while the computational complexity is kept relatively low. The complexity and its reduction is discussed in regard of the number of rules, which result simplicity in operating with the rules, in reinforcement learning methods.

Definitions:

N-mode matrix of a given tensor A**:** Assume an N-th order tensor $A \in \Re^{I_1 \times I_2 \times \cdots}$ The n-mode matrix $A_{(n)} \in \Re^{I_n \times J}$, $J = \prod_k I_l$ contains all the vectors in the n-th dimension of the tensor A. The ordering of the vectors is arbitrary, this ordering shall, however, be consistently used later on. $(\mathbf{A}_{(n)})_j$ is called an j-th n-mode vector. Note that any matrix of which the columns are given by n-mode vectors $(\mathbf{A}_{(n)})_j$ can evidently be restored to be the tensor A. (See a three dimensional example on Fig.5.)

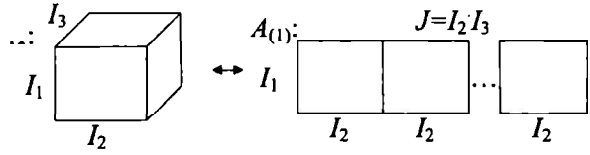


Fig. 5. N-mode matrix of a tensor (three dimensional example).

N-mode matrix-tensor product: The n-mode product of a tensor $A \in \Re^{I_1 \times I_2 \times \ldots \times I_N}$ by a matrix $\mathbf{U} \in \Re^{J \times I_n}$, denoted by $A \times_n \mathbf{U}$ is an $(I_1 \times I_2 \times \ldots \times I_{n-1} \times J \times I_{n+1} \times \ldots \times I_N)$-tensor of which the entries are given by $A \times_n \mathbf{U} = B$, where $B_{(n)} = \mathbf{U} \cdot A_{(n)}$. Let $A \bigotimes_{n=1}^{N} \mathbf{U}_n$ stand for $A \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \ldots \times_N \mathbf{U}_N$.

N-th Order SVD or Higher Order SVD (HOSVD):

Every tensor $A \in \Re^{I_1 \times I_2 \times \ldots \times I_N}$ can be written as the product $A = S \bigotimes_{n=1}^{N} \mathbf{U}_n$, in which $\mathbf{U}_n = [\mathbf{u}_{1,n} \quad \mathbf{u}_{2,n} \quad \mathbf{u}_{I_N,n}]$ is a unitary $(I_N \times I_N)$-matrix called n-mode singular matrix. Tensor $S \in \Re^{I_1 \times I_2 \times \ldots \times I_N}$ of which the subtensors $S_{i_n=\alpha}$ have the properties of all-orthogonality (two subtensors $S_{i_n=\alpha}$ and $S_{i_n=\beta}$ are orthogonal (their scalar product equals 0) for all

possible values of n,α and β: $\langle S_{i_n=\alpha}, S_{i_n=\beta}\rangle = 0$ when $\alpha \neq \beta$ (where $\langle A,B\rangle = \sum_{i_1}\sum_{i_2}\dots\sum_{i_N} a_{i_1 i_2 \dots i_N} b_{i_1 i_2 \dots}$ is the scalar product of two tensors $A,B \in \Re^{I_1 \times I_2 \times \dots \times I_N}$)) and ordering: $\|S_{i_n=1}\| \geq \|S_{i_n=2}\| \geq \dots \geq \|S_{i_n=I_n}\| \geq 0$ for all possible values of n (where $\|A\| = \sqrt{\langle A,A\rangle}$ is the Frobenius-norm of a tensor A). (See a three dimensional example on Fig.6.) See detailed discussion and notation of matrix SVD and Higher Order SVD (HOSVD) in [5].

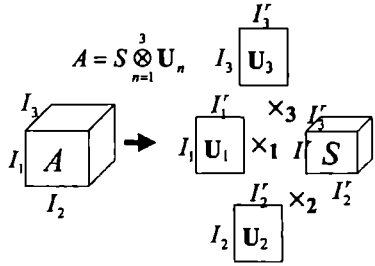


Fig. 6. N-th Order SVD or Higher Order SVD (three dimensional example).

Exact / non-exact reduction

Assume an N-th order tensor $A \in \Re^{I_1 \times I_2 \times \dots \times I_N}$. **Exact** reduced form $A = A^r \overset{N}{\underset{n=1}{\otimes}} \mathbf{U}_n$, where "$r$" denotes "reduced", is defined by the tensor $A^r \in \Re^{I_1^r \times I_2^r \times \dots}$ and basis matrices $\mathbf{U}_n \in \Re^{I_n \times I_n^r}$, $I_n^r \leq I_n$, $n = 1,2,\dots,N$ which are the result of HOSVD, where only zero singular values and the corresponding singular vectors are discarded. **Non-exact** reduced form $\hat{A} = A^r \overset{N}{\underset{n=1}{\otimes}} \mathbf{U}_n$, is obtained if not only zero singular values and the corresponding singular vectors are discarded.

SVD Based Fuzzy Rule Base Complexity Reduction

The explicit formula of the order-0 Takagi-Sugeno Fuzzy Inference method: (e.g. (4)) Assume an N-variable fuzzy rule base given by: antecedent fuzzy sets $\mu_{i_n,n}(x_n)$ defined on input universe X_n and all combination of the antecedents corresponds to one consequent fuzzy set defined on output universe Y These relations are expressed by rules in the form of

If $\mu_{i_1,1}(x_1)$ **And** $\mu_{i_2,2}(x_2)$ **And ... And** $\mu_{i_N,N}(x_N)$ **Then** $y = \beta_{i_1 i_2 \dots}$. (6)

Singleton consequent fuzzy sets $\beta_{i_1 i_2 \ldots}$ are defined by their location $b_{i_1 i_2 \ldots i_N}$ Setting up the antecedent fuzzy partitions to be *Ruspini partitions*, the explicit formula of the inference technique is (for more detailed explanation see [20]):

$$f(x_1, x_2, \ldots, x_N) = \sum_{i_1, i_2, \ldots, i_N}^{I_1, I_2, \ldots, I_N} \prod_{n=1}^{N} \mu_{i_n, n}(x_n) b_{i_1 i_2 \ldots} \tag{7}$$

SVD Based Fuzzy Rule Base Complexity Reduction: The formula of the order-0 Takagi-Sugeno Fuzzy Inference method (7) can be equivalently written in tensor product form as: $f(x_1, x_2, \ldots, x_N) = B \overset{N}{\underset{n=1}{\otimes}} \mathbf{m}_n$, where the tensor $B \in \Re^{I_1 \times I_2 \cdots}$ and the vector $\mathbf{m}_n$ respectively contain elements $b_{i_1 i_2 \ldots i_N}$ and $\mu_{i_n, n}(x_n)$. This reduction can be conceptually obtained by reducing the size of the tensor B via Higher Order SVD (HOSVD). See more detailed description in [5], [19], [20]. The SVD based fuzzy rule base reduction transforms the structure of equation (7) to the form of:

$$f(x_1, x_2, \ldots, x_N) = \sum_{i_1, i_2, \ldots, i_N}^{I_1^r, I_2^r, \ldots, I_3^r} \prod_{n=1}^{N} \mu_{i_n, n}^r(x_n) b_{i_1}^r \tag{8}$$

where $\forall n : J_n^r \leq J_n$ is obtained as the main essence of the reduction.

The reduced form (8) of (7) is obtained via HOSVD capable of decomposing B into $B = B^r \overset{N}{\underset{n=1}{\otimes}} \mathbf{U}_n$ Having $B^r \in \Re^{I_1^r \times I_2^r \cdots}$ and its singular vectors the reduced form is determined as: $f(x_1, x_2, \ldots, x_N) = B^r \overset{N}{\underset{n=1}{\otimes}} \mathbf{m}_n^r$, where $\mathbf{m}_n^r = \mathbf{m}_n \mathbf{U}_n$.

4.2. Adaptation of SVD based Approximation

According to the previous sections the crucial concept of the reinforcement learning is based on the adaptivity of the action-value function. It was also concluded in the previous sections that the approximation accuracy of the action-value function is highly restricted by its computational complexity. For instance, the increase of the density of the approximation grid on Fig. 4 improves the approximation accuracy. Each learning step may insert a new gridline into dimension *S*. However, this may lead to a high complexity soon, since adding a grid-line exponentially increases the number of the approximation grid. Therefore, it is highly desired to reduce the complexity of the action-value function. However, one should note that a natural problem of typical complexity reduction is that it decreases the adaptivity property with the complexity in general. This disadvantage is also true for SVD based reduction technique discussed in the previous section. This implies that executing the SVD based reduction on the action-value function would destroy the effectiveness of the whole learning concept. Therefore, this paper proposes to utilize a "fast adaptation" technique, introduced in [6], capable of keeping the action-value function in SVD based complexity reduced form, but also capable of adapting the function without considerable computational effort. This method let us directly adapt the complexity compressed action-function over any specified point of the learning space and add new approximation grid-

lines, see Fig. 4. The key idea is that the fast adaptation technique transforms the given new grid-lines and corresponding values into the complexity reduced space of the action-function where the adaptation can immediately be done. The ability of embedding new approximation points provides the practical applicability of the proposed dynamic partition allocation method discussed in the previous section. Therefore, the application of the fast adaptation method in the proposed reinforcement learning structure is twofold. On one side, it helps the dynamic partition allocation by increasing the grid density. On the other side, by the replacement of the previously fetched and modified values serves the adaptation of the approximated action-value function.

Let the goal of the adaptation technique be specified in the followings: The goal is to insert a set of new rules included in A into the existing rule base B. Assume that the rule base B is already reduced into B^r The new rules contained in A should directly be inserted into B^r Directly means that without decompressing B^r to B. Assume that the size of B^r is fixed, it must not be increased with the adaptation. As a matter of fact, there may be such rules in A which require the increase of the size of B^r The fast adaptation technique discards these rules and inserts only those ones collected in A' which do not increase the size of B. In order to insert as much rules as possible the fast adaptation technique has a further option. Subject to a given threshold ∇, it is capable of modifying the discarded rules in order to insert them to B^r If the rule bases are represented by tensors as discussed in the previous section then the adaptation can be defined as: only those sub-tensors A' of A are embedded into B^r, which are linearly dependent from B^r [6]. An important advantage of the fast adaptation is that no SVD is needed during inserting the new rules.

N mode fast adaptation [6]:

"N mode" means in the present case that the rules, to be inserting, have new antecedents on dimension N. Namely, this means that the number the approximation grid-lines under the function, see Fig. 4, is increased in dimension N.

Assume a reduced rule base defined by tensor $B^r \in \Re^{J_1^r \times J_2^r \times \cdots}$ and its corresponding matrices $\mathbf{Z}_n \in \Re^{J_n \times J_n^r}$ resulted from B by HOSVD as:

$$B = B^r \overset{N}{\underset{n=1}{\otimes}} \mathbf{Z}_n \tag{9}$$

Furthermore, let $A \in \Re^{J_1 \times J_2 \times \cdots}$ be given, that has the same size as B except in the n-th dimension where I may differ from J_n. Let us have a brief digression here and explain A and B on Fig. 4. Tensor B, which is a matrix in the case of Fig. 4, consists of the values of the function over the grid-points defined by the orthogonal grid-lines located at values s. Tensor A, which is also a matrix in the present case, contains the values over the grid-points and the new grid-lines located on dimension N, that is S on Fig. 4. We can observe that the size of A is equivalent to the size of B except on that dimensions where the new gridlines are defined. If B is compressed to B^r then we do not have this matrix point-wise equivalency to the rectangular grid. In this case the inserting of the new grid-lines and their corresponding new approximation points is not trivial.

The localized error interval of the adaptation is defined by ∇ Localised means that ∇ is a tensor whose elements are intervals and assigned to the grid-points like in the case of A and B. It defines the acceptable varying of the function over the grid-points. The goal is to

determine the reduced form E' of extended rule base E, defined by tensor $E'=[B \quad A']_n$. In the case of Fig. 4 E is a matrix and contains the values of the function over all the new and the original grid-lines. E' contains the selected n mode sub-tensors of E according to the given interval ∇ In the case of Fig. 4 E' contains the values over all the original grid-points and over those grid-lines, which are accepted to be inserted. Only those grid-lines are accepted which do not increase the size of B', or whose modified values are still in the intervals of ∇ Thus

$$\hat{E}'=\left(B' \overset{N}{\underset{k=1}{\otimes}} \mathbf{Z}_k\right)\times_n \mathbf{U}, \tag{10}$$

and $A\in\Re^{J_1\times J_2\times..J_{n-1}\times J'\times J_{n+1}\times..\times J_N}$ contains the selected n mode sub-tensors of A and let the corresponding sub-tensors $T'_{\min/\max}$ be selected from the corresponding $T_{\min/\max}$ which define the maximal and minimal values of the elements of ∇ For brevity let $\nabla'=[T'_{\min} \quad T'_{\max}]$. $\mathbf{U}=[\mathbf{Z}_n \quad \mathbf{V}]\in\Re^{(J_n+I')\times J'_n}$, $I'\le I$, where $\mathbf{V}$ is determined to satisfy (10) subject to $\hat{E}'-E'\in_t \nabla'$. $\in_t$ means that the elements of tensor $\hat{E}'-E'$ is in the interval defined by the corresponding elements of ∇' (the bound of the intervals are defined by the corresponding elements of $T_{\min}$ and $T_{\max}$).

Inserting new gridlines on all dimensions is done in the same way. This means that the desnity of the hyper rectangular approximation grid can be incerased by the above algorithm even in case when the values assigned to grid are compressed into a reduced form where there is no structure which can be localised according to the grid-points. The more detailed description of the fast adaptation algorithm is given in [4] and [6].

5. Practical use of the Proposed Reinforcement Technique

For introducing the proposed application way of SVD based fuzzy rule based approximation techniques in reinforcement learning, a simple application example, where the state-transition function characterised by the following equation, was chosen:

$$s^{k+1}=2\cdot\left(s^k+a^k\right), \tag{11}$$

where $s\in S=[-1,1]$ is the one dimensional state and $a\in A=[-0.2,0.2]$ is the action. The reward is calculated in the following manner: $r=1$ *iff* $s\in[-0.1,0.1]$ *else* $r=0$

The first experiment is related to the efficiency of the proposed dynamic partition allocation method (see results on Fig.7). Fig.7.b is introducing the two basic problems of fixed partition: The lack of universal approximation property in case of rough partition and the difficulties of the adaptation.

The second experiment is related to the efficiency of the proposed SVD based complexity reduction and approximation adaptation (fast adaptation method). Fig.8. introduces three stages of a 20000 step iteration. On Fig.8.a the iteration process turns the action-value rule base to reduced form at the iteration step 1000, by applying the SVD Based Fuzzy Rule Base Complexity Reduction (introduced in Section 4.1.) From this step the iteration is continuing up to 20000 iterations using the fast adaptation method (introduced in Section

4.2.). Fig.8.b is the same experiment as Fig.8.a, except the turning the reduction is done earlier at the step 5000.

6. Conclusions

One of the possible difficulties of the reinforcement learning applications in complex situations is the huge size of the state-value- or action-value-function representation [17]. The case of continuous environment reinforcement learning could be even complicated, in case of applying dense partitions to describe the continuous universes, to achieve precise approximation of the basically unknown state-value- or action-value-function. The fine resolution of the partitions leads to high number of states, and handling high number of states usually leads to high computational costs, which could be unacceptable not only in many real time applications, but in case of any real (limited) computational resource. As a simple solution of these problems, in this paper the adoption of Higher Order SVD [5] based fuzzy rule base complexity reduction techniques and its fast adaptation method [6] is suggested. The application of the fast adaptation method [6] gives a simple way for increasing the rule density of a rule base stored in a compressed form directly. To fully exploit this feature, a dynamic partition allocation method is also suggested. Based on the application examples, the main conclusion of this paper is the reducibility of action-value function. It seems that in many cases the representation of the action-value function is considerably reducible. Moreover due to the fast adaptation method this reduction can be performed in an early stage of the adaptation and the iteration steps can be continued on an economic sized action-value function representation.

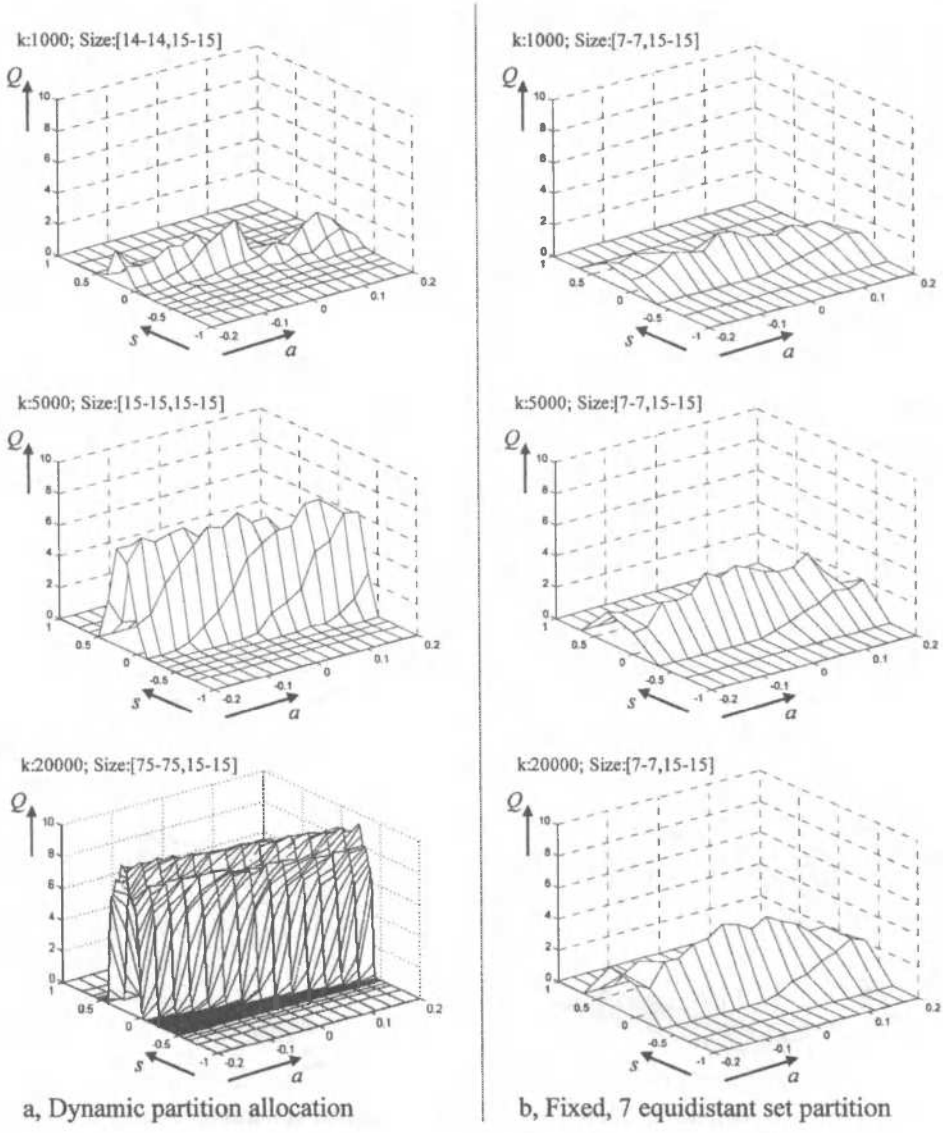
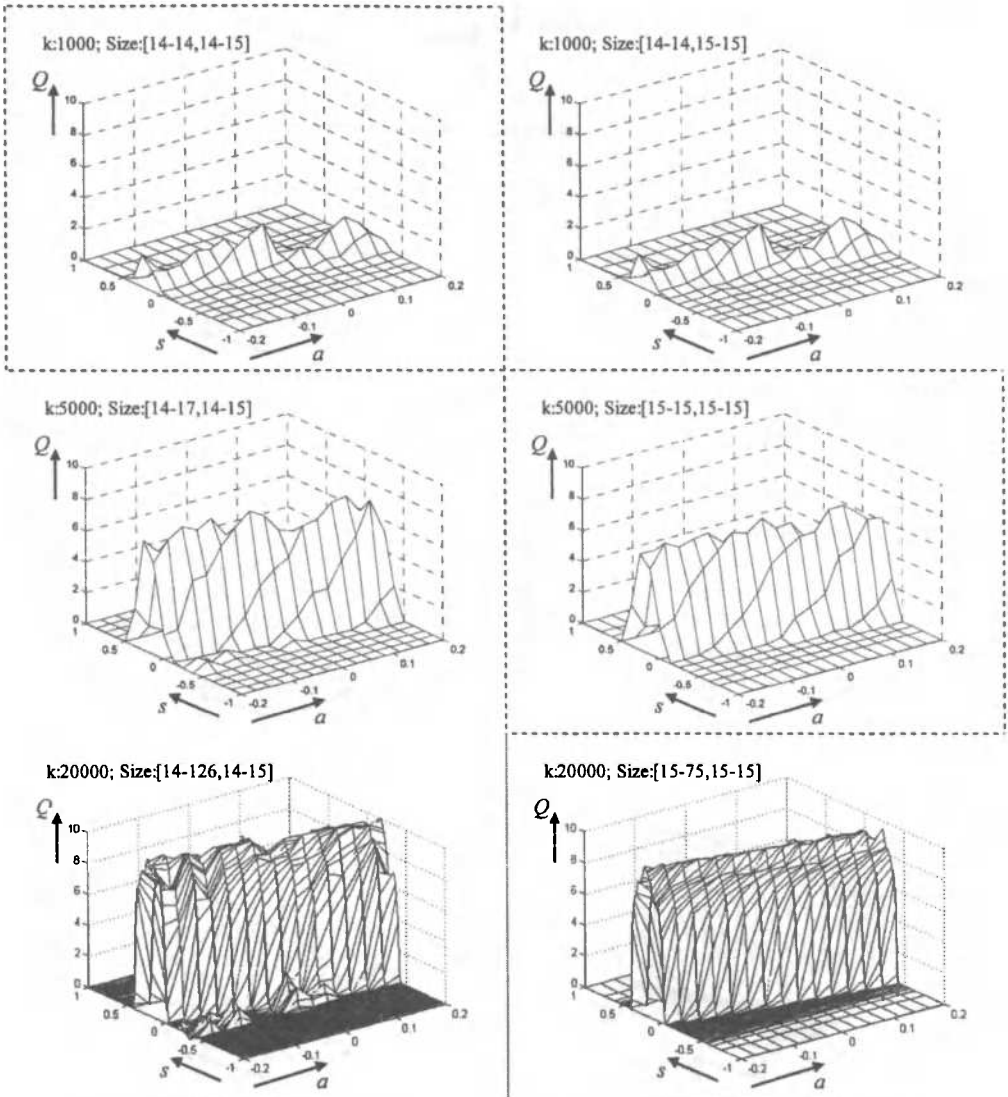


Fig. 7. The first experiment, the lack of universal approximation property in case of rough predefined fixed partition (difficulties in adaptation).



a, Reduction at iteration step 1000

b, Reduction at iteration step 5000

Fig. 8. The effect of SVD based complexity reduction and approximation adaptation, where k is the iteration number and Size is the size of the reduced (B' as it is stored) and the extended (B as its used) action-value rule base (e.g. Size:[14-126,14-15] means, that the original 126x15 sized action value rule base is stored and adapted in a 14x14 reduced format).

ACKNOWLEDGEMENTS

This research was partly supported by the Hungarian National Scientific Research Fund grant no. F 029904. Szilveszter Kovács is supported by the György Békésy Postdoctoral Scholarship. Péter Baranyi is supported by the Zoltán Magyary Postdoctoral Scholarship.

REFERENCES

1. APPL, M.: *Model-based Reinforcement Learning in Continuous Environments.* Ph.D. thesis, Technical University of München, München, Germany, dissertation.de, Verlag im Internet (2000)
2. BARANYI, P., YAM, Y.: *Singular Value-Based Approximation with Non-Singleton Fuzzy Rule Base.* 7th Int. Fuzzy Systems Association World Congress (IFSA'97) Prague (1997) pp 127-132.
3. BARANYI, P., YAM, Y. VÁRLAKI, P., MICHELBERGER, P.: *Singular Value Decomposition of Linguistically Defined Relations.* Int. Jour. Fuzzy Systems, Vol. 2, No. 2, June (2000) pp 108-116.
4. BARANYI, P., VÁRKONYI-KÓCZY, A.R.: *Adaptation of SVD Based Fuzzy Reduction via Minimal Expansion.* IEEE Trans. on Instrumentation and Measurement, Vol. 51, No. 2 (2002) pp 222-226. (ISSN 0018-9456)
5. BARANYI, P., VÁRKONYI-KÓCZY, A.R., YAM, Y., PATTON, R.J., MICHELBERGER, P., SUGIYAMA, M.: *SVD Based Reduction to TS Fuzzy Models.* IEEE Transaction on Industrial Electronics, Vol. 49, No. 2, 2002, pp 433-443.
6. BARANYI, P., VÁRKONYI-KÓCZY, A.R., YAM, Y., VÁRLAKI, P., MICHELBERGER, P.: *An Adaption Technique to SVD Reduced Rule Bases.* IFSA 2001, Vancouver (2001) pp 2488-2493.
7. BELLMAN, R. E.: *Dynamic Programming.* Princeton University Press, Princeton, NJ (1957)
8. BERENJI, H.R.: *Fuzzy Q-Learning for Generalization of Reinforcement Learning.* Proc. of the 5th IEEE International Conference on Fuzzy Systems (1996) pp 2208-2214.
9. BONARINI, A.: *Delayed Reinforcement, Fuzzy Q-Learning and Fuzzy Logic Controllers.* In Herrera, F., Verdegay, J. L. (Eds.) Genetic Algorithms and Soft Computing, (Studies in Fuzziness, 8), Physica-Verlag, Berlin, D, (1996) pp 447-466.
10. CASTRO, J.L.: *Fuzzy Logic Controllers are Universal Approximators.* IEEE Transaction on SMC, Vol.25, 4 (1995)
11. GLORENNEC, P.Y., JOUFFE, L.: *Fuzzy Q-Learning.* Proc. of the 6th IEEE International Conference on Fuzzy Systems (1997) pp 659-662.
12. HORIUCHI, T., FUJINO, A., KATAI, O., SAWARAGI, T.: *Fuzzy Interpolation-Based Q-learning with Continuous States and Actions.* Proc. of the 5th IEEE International Conference on Fuzzy Systems, Vol.1 (1996) pp 594-600.
13. KÓCZY, L.T., HIROTA, K.: *Size Reduction by Interpolation in Fuzzy Rule Bases.* IEEE Trans. SMC, vol. 27 (1997) pp 14-25.

14. RUDAS, I.J.: *Towards the generalization of t-operators: a distance-based approach.* Journal of Information and Organizational Sciences. Vol.23. No.2. (1999) pp 149-166.
15. RUDAS, I.J. KAYNAK, M.O.: *Entropy-Based Operations on Fuzzy Sets.* IEEE Transactions on Fuzzy Systems, vol.6, no. 1, (1998) pp 33-40.
16. RUDAS, I.J., KAYNAK, M.O.: *Minimum and maximum fuzziness generalized operators.* Fuzzy Sets and Systems (1998) pp 83-94.
17. SUTTON, R. S., BARTO, A. G.: *Reinforcement Learning: An Introduction,* MIT Press, Cambridge (1998)
18. TIKK, D.: *On nowhere denseness of certain fuzzy controllers containing prerestricted number of rules.* Tatra Mountains Mathematical Publications vol. 16. (1999) pp 369-377.
19. YAM, Y.: *Fuzzy approximation via grid point sampling and singular value decomposition.* IEEE Trans. SMC, Vol. 27 (1997) pp 933-951.
20. YAM, Y., BARANYI, P., YANG, C. T.: *Reduction of Fuzzy Rule Base Via Singular Value Decomposition.* IEEE Transaction on Fuzzy Systems. Vol.: 7, No. 2 (1999) pp 120-131.
21. YEN, J., WANG, L.: *Simplifying Fuzzy Rule-based Models Using Orthogonal Transformation Methods.* IEEE Trans. SMC, Vol 29: Part B, No. 1 (1999) pp 13-24.
22. WANG, L.X.: *Fuzzy Systems are Universal Approximators.* Proceedings of the First IEEE Conference on Fuzzy Systems, San Diego (1992) pp 1163-1169.
23. WATKINS, C. J. C. H.: *Learning from Delayed Rewards.* Ph.D. thesis, Cambridge University, Cambridge, England (1989).

Production Systems and Information Engineering, Miskolc, Vol. 1. (2003), pp. 125-133

SYMBOLIC ALGEBRAIC COMPUTATION AS A WAY FOR MODELLING KINEMATICAL TASKS THEMSELVES

LÁSZLÓ DUDÁS*
Department of Information Engineering, University of Miskolc
H-3515 MISKOLC, Hungary
iitdl@gold.uni-miskolc.hu

[Received February and accepted April 2003]

* Production Information Engineering Research Team (PIERT) of the Hungarian Academy of Sciences; Department of Information Engineering of the University of Miskolc.

Abstract The paper introduces the symbolic algebraic computational capability of Surface Constructor, a software tool for gear investigation. Use of symbolic expressions in the kinematical model gives a higher freedom in constructing and modifying the model. Modelling in this manner we can model the kinematical model itself and can make a kinematical modelling shell. The advantages of applying symbolic algebraic computation are discussed and an example on determining the working surfaces of a hypoid pair and analysing the pattern of connection is presented.

Keywords: Symbolic algebraic computation, Kinematical modelling, Design of hypoid gear

1. Introduction

Modelling complex kinematical systems such as multi degrees of freedom robots or multi axes machine tools needs a freedom in definition of the relative motions and positions of parts. This task becomes especially interesting at simulating work of hobbing or teething machines. It would be the best to have a general modelling tool that gives the maximal possibilities to define the relative kinematical relations and the applied tool-surfaces, too. The well-known CAD and CAM applications apply numerical representations of required kinematical transformation matrices and spline surfaces. This method gives fairly good resolution at modelling fixed architectures with simple translations and rotations. However, in more complex situations there is a need for applying algebraic expressions as transformation matrix elements or in the parametric surface vector. This approach is valid when the task is the modelling of contacting of mating hypoid gear surfaces or producing the machined surface cut by a conical hob. To covering all these situations the best method is to apply symbolic representation and symbolic algebraic computation. Symbolic computation is known as one of the areas of applied artificial intelligence. Use of symbolic expressions in the model gives a higher freedom in constructing and modifying the model. Modelling in this manner we can model the kinematical model itself and can make a kinematical modelling shell. The paper presents a developed design tool that fulfils the above theory and an example for modelling hypoid gear surfaces generated by the same intermediary generating cone.

2. Symbolic algebraic computation in the Surface Constructor software

One of the requirements of today gear-connection investigator software is a powerful geometric modeller, which can give the engineer the maximum freedom to build the kinematical model of the analysed cutting machine or the gearing. The Surface Constructor (SC) software developed by the author has these characteristics. SC applies the 'Reaching Model', a hybrid theory that can produce the enveloped surface and can detect all the types of local undercuts and global cuts and applies symbolic algebraic representation and computation to give the maximum freedom in kinematical modelling [1]. The use of computers for symbolic algebraic computation is not as well known as using for numerical computations.

What is symbolic computation? Using symbolic computation we can enter not only numbers but algebraic expressions, too. The computer manipulates the expressions, simulates hand algebra, uses algebraic identities and so on and gives the result: another expression or group of expressions. We may read articles or books discussing general symbolic software, for example REDUCE, MACSYMA, CAMAL, LAM, FORMAC, SYMBOL, and so on [2], and articles about special software application including symbolic computation module [3]. SC is the member of the last mentioned group.

The set of algebraic manipulations performed by a general symbolic software is big: basic operations: +,-,*,/; manipulation of parentheses; cancelling common factors; factorisation; manipulating functions and expressions, variables and simple numbers; performing sequence of assignments; the use of trigonometric and logarithmic identities; derivation and integration of expressions; simplifying and so on.

SC can perform a suitable selected subset of above mentioned manipulations only: basic operations, manipulation of parentheses, manipulation of SIN, COS, TAN, SQRT functions, manipulation of expressions, variables and numeric values, performing sequence of assignments, simplifying expressions and using trigonometric identities. The symbolic module in SC can compute multiplied matrices and vectors, can generate the inverse matrix in symbolic form because these manipulations and the use of trigonometric identities are very frequent in problems resolved by SC.

The symbolic computations are needed in the following circumstances:

- to generate the transformation matrix and inverse transformation matrix of two neighbouring systems of co-ordinates,
- to produce the resultant matrix from one system of co-ordinates to another using sequence of matrix multiplication,
- to produce the generating surface using matrix-vector multiplication (given the matrix of the generating motion and vector-scalar function of space curve),
- to produce the resultant matrix from the selected system of co-ordinates to the world system of co-ordinates (to give an animated, moving picture of the modelled problem).

3. An example of realisation of a symbolic operation

For example let us examine one of the most frequently used symbolic operation. We have SIN and COS function for sum of angles and we want to factorise it. We have to use one of the well-known trigonometric identities, for example SIN(A+B) SIN(A)*COS(B)+SIN(B)*COS(A). In a more complicated situation A or B may be a complex symbolic expression involving terms and factors of constants, variables and functions. So we have to give a general and exact resolution. For this purpose it is excellent to give the rules of transformation using a sequence of syntax diagrams. In order to spare memory and running time a one byte coding had been applied. A constant, a variable, a function or an expression may substitute for a one byte code. In this manner we can get a very quick program: it can produce a 20 line matrix expression in an imperceptible amount of time. The meanings of the symbols +,-,* and / are obvious. Hereafter S will be the code for SIN, C for COS.

Unfortunately we can only give here a very short overview of realisation because of page limit of the paper.

The steps of the used process are:

1. to find an occurrence of a function of angle-sum in the given symbolic expression, if there is,
2. to produce the transformed function of sum with its environment,
3. to substitute the function of sum and its environment for transformed function of sum with its environment,
4. to repeat the above steps.

Fig.1. on the next page shows a few syntax diagrams related to the earlier discussion.

4. Demonstration of symbolic computation in SC

In the next the fist example will demonstrate the symbolic data handling, manipulation and calculation in the SC program. Then, in the second example, a fairly complex geometrical problem will be resolved by the SC: determining the two connecting surfaces of a hypoid gears using an intermediary generating cone surface. By this example we may get some impression about modelling and visualisation power of SC. Among features not discussed here there are visualisation of path of moving, space of relative speed and acceleration, curvature parameters, animation of the enveloping process, visualisation of R-Φ functions that are special features of SC. The software is capable to determine undercuts and calculate the unknown enveloped surface using local or global calculation.

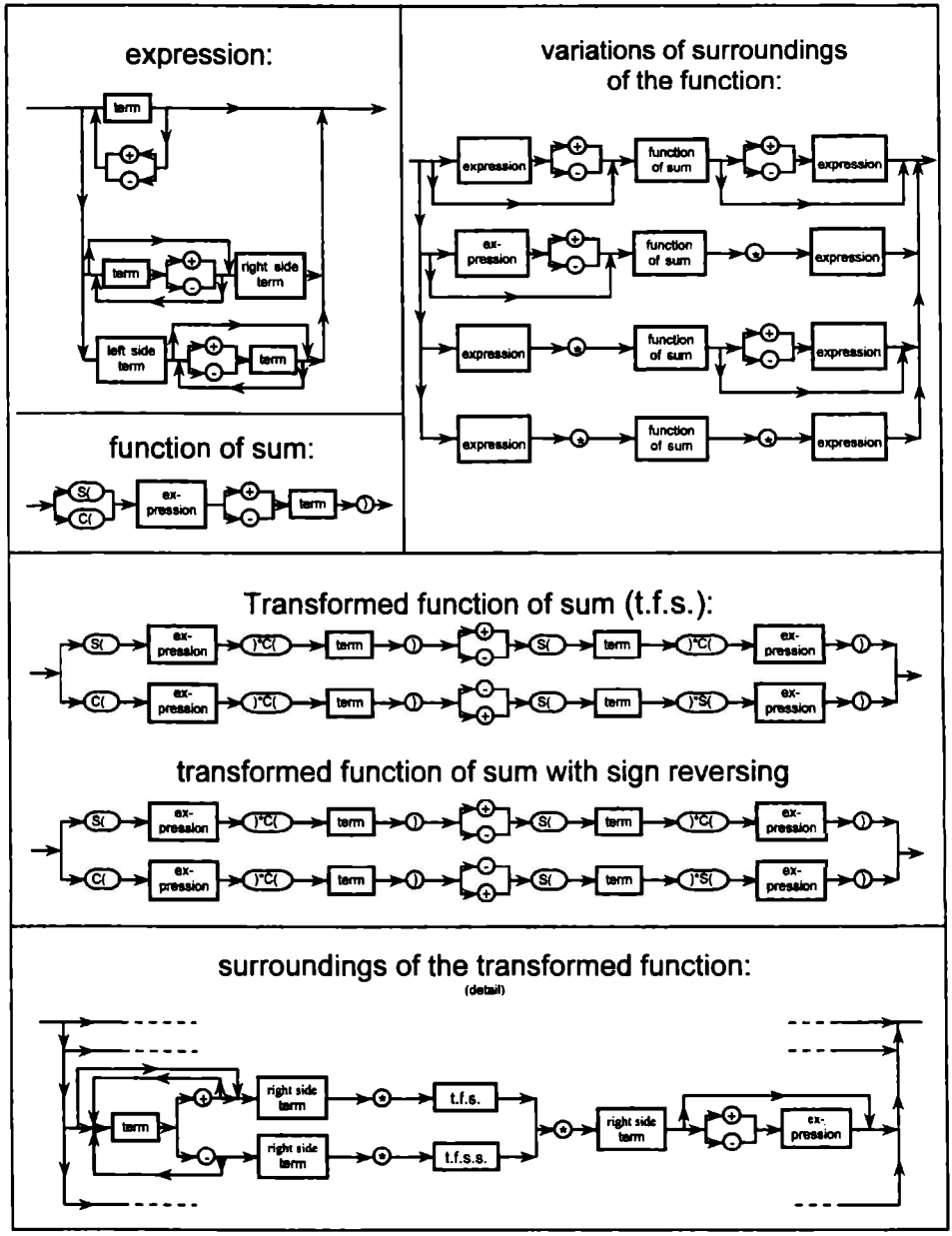


Fig. 1. Syntax diagrams

The symbolic computation starts with entering the generating curves or surfaces by parametric expressions. The input panel for the generating surface is shown in Fig. 2.

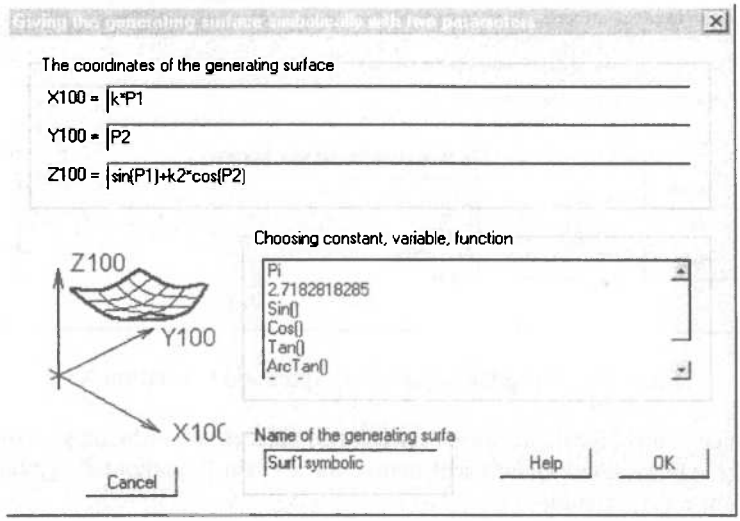


Fig.2. Entering a two-parametric symbolic generating surface

The kinematical relations can be defined by a chain of co-ordinate systems. The relation between two adjacent systems can be entered on a panel like shown in Fig.3.

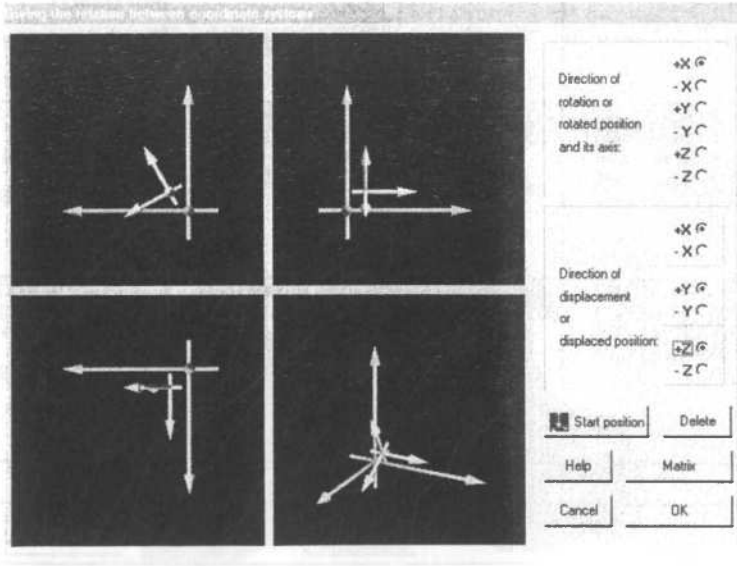


Fig.3. Selecting the directions of translation and rotation

The symbolic expression for position or time-parameter dependent motion have to be assigned in every selected direction using a window similar to the next shown in Fig.4.

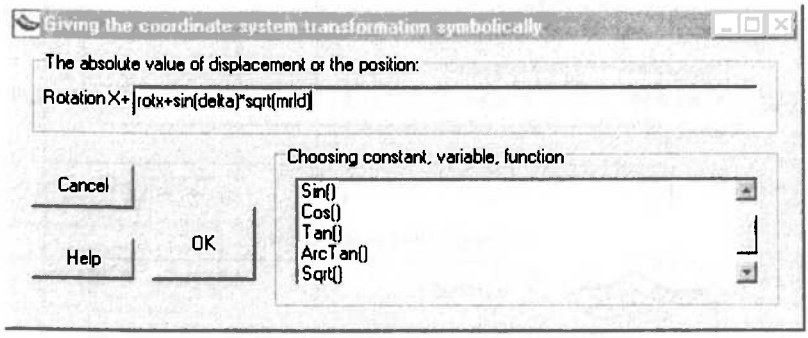


Fig.4. Assigning the parametric expression to rotation X+

The kinematical matrix for the relation between two adjacent co-ordinate systems generated automatically. The resulted kinematical matrix for a chain is computed symbolically in a snap. Fig.5. shows an example.

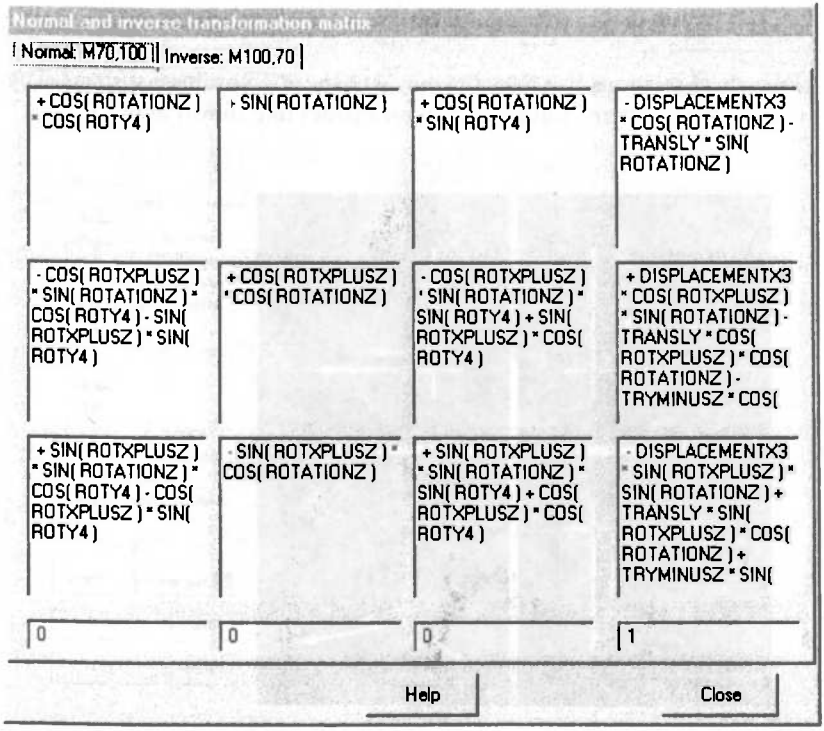


Fig.5. Result of a kinematical matrix concatenation

After building up the kinematical model symbolically the value entering follows. This is important for numerical calculations, to determine the second enveloped surface, or visualise the model. A comfortable panel helps this task, see below in Fig.6.

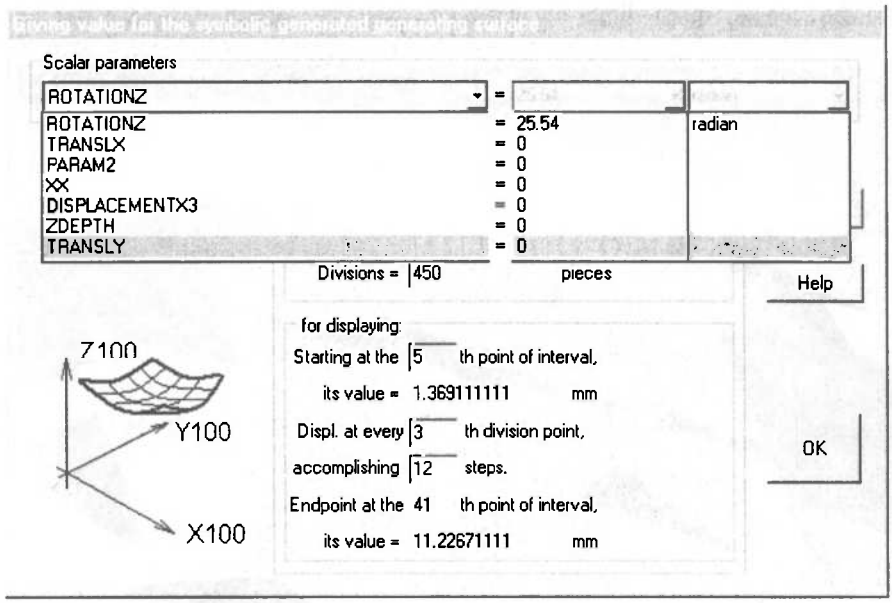


Fig.6. Assigning numerical values to parameters

In the next the result of modelling a hypoid gearing will be presented. The generated connecting surface parts are drawn in Fig.7.

Fig.7. Motion of the hypoid gears can be animated

The generation of the two hypoid gears was made using an intermediary generating cone. This cone envelops the hypoid gears. The cone contacts to one of the hypoid gears in a curve as shown in Fig.8. The two contacting lines intersect each other on the cone. This intersection point will give the centre of the point-like connection of the two hypoid gears. This can be seen as a quasi-elliptical pattern.

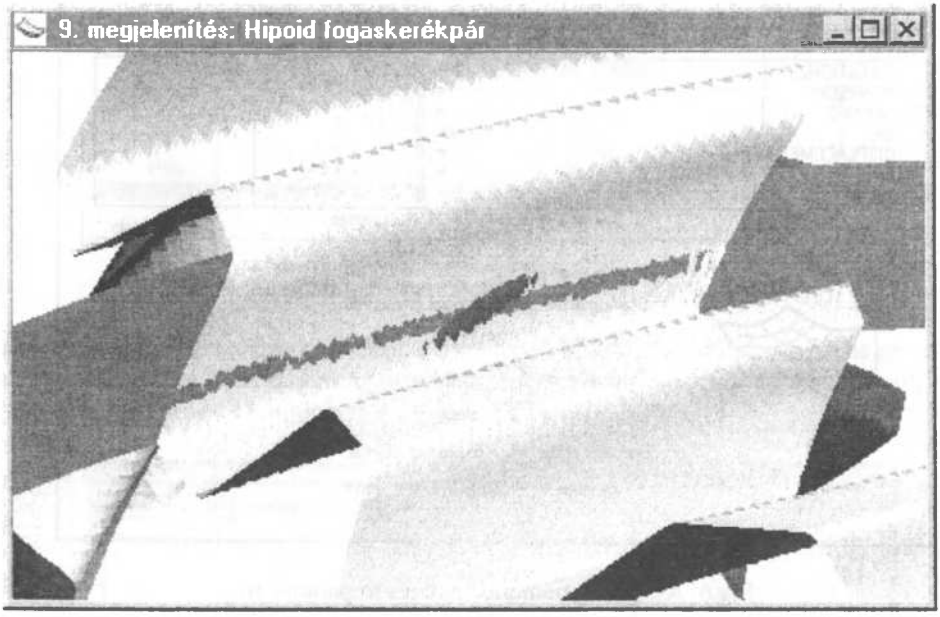


Fig.8. Contacting curves and patterns can be analysed

5. Conclusion

The paper gave a short overview of symbolic computational ability of Surface Constructor, a kinematical modelling software. Using SC, the designer can generate the conjugate surface of a given surface. In this process the relative motion has to be entered and represented by transformation matrices. Both the given surface and the relative motion are used in symbolic algebraic form. The main calculations concerning on kinematical relations are performed automatically in symbolic way. Having substituted concrete parameter values the user can get the kinematical model of a real problem, i.e. contacting gear surfaces or manufacturing worm using a grinding wheel. The task can be analysed changing values of parameters, or the model can be changed itself entering new symbolic expressions for the generating surface and/or the kinematical relations.

Thanks to symbolic algebraic representation and computation it is possible to model the kinematical modelling process itself.

Acknowledgements

The investigation summarised in the paper has been continued within the framework of Production Information Engineering Research Team (PIERT) established at the Department of Information Engineering in 1999 and supported by Hungarian Academy of Sciences. The financial support of the research is gratefully acknowledged.

REFERENCES

1. DUDÁS, L.: *A Consistent Model for Generating Conjugate Surfaces and Determining All the Types of Local Undercuts and Global Cut,* Proceedings of UMTIK'96 International Machine Design and Production Conference, Ankara, Sept.11-13. 1996, pp. 467-476.
2. RAJNA, G.: *REDUCE Software for Algebraic Computation*, Springer-Verlag, New York • Berlin • Heidelberg, 1987.
3. TOWNSEND, M. A. – GUPTA, S.: *Automated Modelling and Rapid Solution of Robot Dynamics Using the Symbolic Polynomial Technique*, Journal of Mechanisms, Transmissions and Automation in Design, December 1989, Vol.11., pp.537-544.

Production Systems and Information Engineering. Miskolc, Vol. 1. (2003), pp. 135-144

PARAMETERS OF THE QUALIFYING SYSTEM FOR LOGISTICAL ACTIVITIES

ORSOLYA SÁRA CSÉPES
Institute of Management Science, University of Miskolc
szvorsi@mail.uni-miskolc.hu

[Received April 2002 and accepted February 2003]

Abstract. This paper introduces a qualifying parameter system for logistical service companies. The system is based on the chain of logistical activities. By means of this chain of activities and tailor-made parameter system companies can create a decision supporting system. This system cannot substitute fot the well-educated specialists, but it could help them to accomplish their tasks. Thereby the decision making process can be easier, shorter than earlier and documented at the operative and the strategic field too.

Keywords: Quality assurance, qualifying parameters, logistic strategy

1. Introduction

In Hungary the use of quality insurance systems have come to the front recently, which is partly natural, and partly the consequence of artificially induced demand. The artificial demand was induced by the multinational companies by requiring the certificate of the standard of ISO 9000 family from their suppliers. Those companies, which got this certificate in the 90s, take the possession of this evident but most of them consider it unnecessary till today, forgetting what kind of refinement possibilities it includes.
The recession, which is characteristic all over the world, can be felt in the circle of multinational, companies more and more. It has different effects to various sectors but it has the most striking effects on the IT industry. The recession forces all the large companies to try to reduce their costs. It is an extremely difficult task to define the level of costs which still allows the proper quality service of customers to be achieved but does not lead to unwanted costs.

Simplifying the content of the definition of logistics according to the current practice it can be regarded as information, material and energy flow between two consecutive statements. There are approaches in the literature of course, which are more vague and comprehensive.

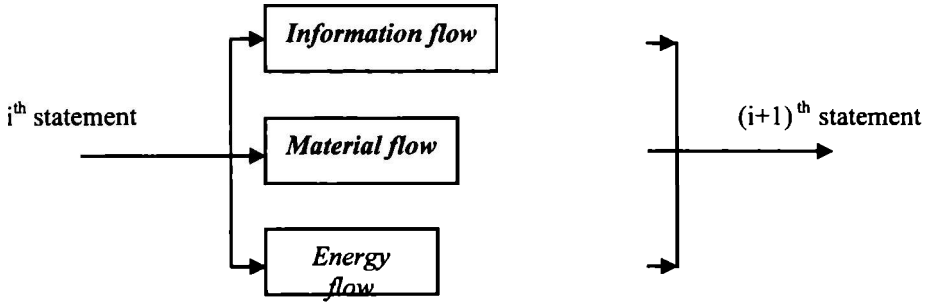


Fig. 1: The interpretation of logistics

During my research I have been focusing on only the information and material flow which is connected to logistics. The reason for this is that the study of the energy flow linking with logistics can be the basis of a new research interrest because it is complicated and large-sized.

2. Matrix system of the logistical activities

In my PhD study I examined the connections and relations between logistics and quality insurance. First of all we have to define the quality insurance parameters, which can be used in the field of logistics to make any survey of the situation, any corrections or any measurements of results. To define these we have to apply some restrictions. In first approach to work out the system it is necessary to know what kind of logistical activities the company apply. The type of applied logistical activities depends on whether it is a manufacturer or a service company. The classification of logistical activities done by companies can be seen on Fig. 2. Logistical service companies usually carry out activities only outside and inside the company or maybe between workshops, where workshops can be called rather plants.

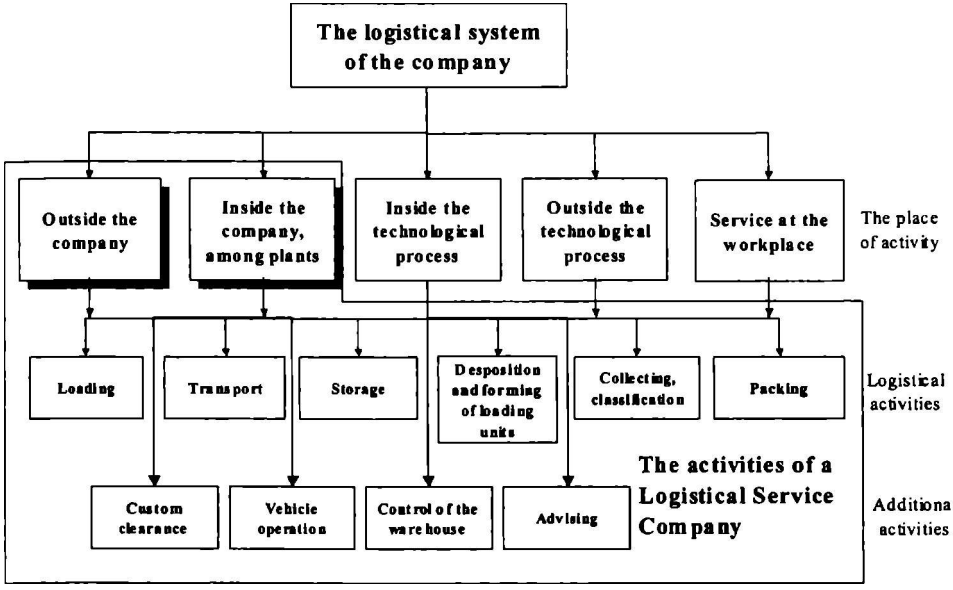


Fig. 2: The classification of logistical activities

In the course of my work I have worked out a method by means of which the changes of the quality features can be followed by attention and can be made numerical during the logistical processes. As Fig. 3 shows the starting point of the examination is the product. Theoretically the processes can be connected with the product indeed but in practice it is more complicated. The cause of this is that the company deals with several types of products and the number of customers is also significant in an ideal case.

That is why it is worth transforming the system in such a way that the processes are connected to the order of the customer (to the customer's number of order). As a result of this the logistical process is linked with the customer's number of order, which can be related to several ones from the wide range of products manufactured or brought in by the company. (Fig. 4)

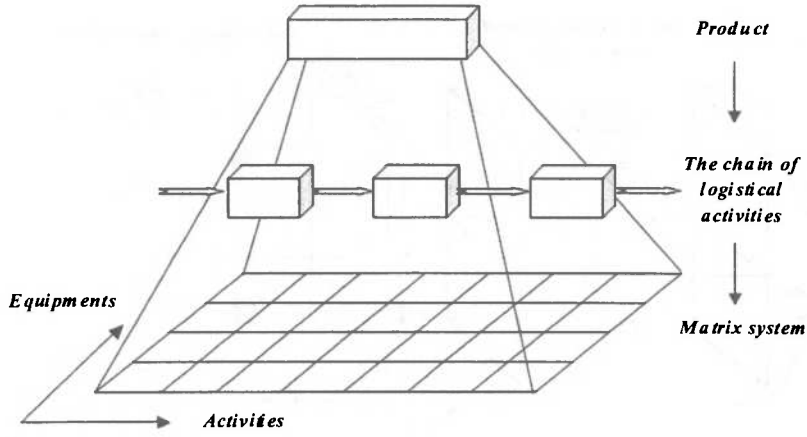


Fig. 3: From the product to the matrix system [2]

This type of approach does not modify the principles and results I mentioned earlier, only make them more exact. The connection between the order numbers and the products is shown by a quantitative matrix, which contains how many customers want to buy from the studied product in course of certain orders.

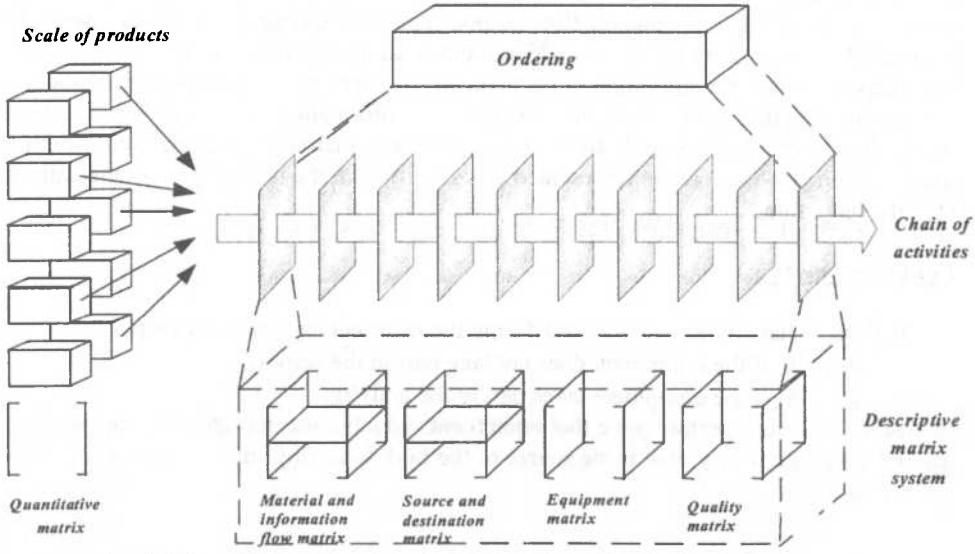


Fig. 4: The connection between logistical processes and the matrix system

The elements of the matrix system, which describes the processes determined by the order, are the following:

Material and information flow matrix and the source-destination matrix

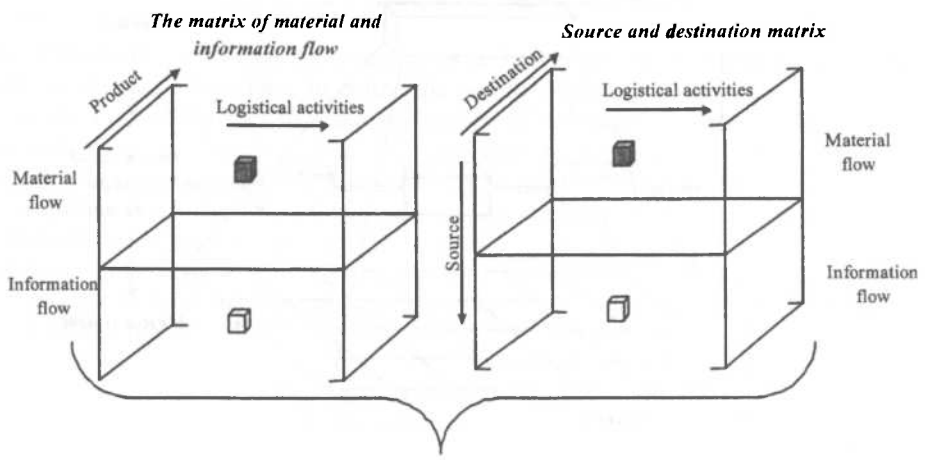


Fig. 5: The material and information flow matrix, the source-destination matrix

For a the simplier and clearer applying the matrixes are divided. The material and information flow matrix contains the elements of 1 and 0, the value depends on whether there is material and information flow or not. The source-destination matrix also should have 1 and 0 values depending on whether there is an information rise or utilization. Filling the matrixes it can be determined unequivocally if there is material or information flow during the activities and where the material and information enter and exit the system, where they appear and are used. By means of these two matrixes a material and information map is outlined which can be represented graphically with the help of graphs specialized on activity and product.

Equipment matrix

It shows what equipment are used during the different logistical activities.

$e_{i,j,k} = 0$, if the equipment does not take part in the activity,

$e_{i,j,k} = 1$, if the equipment takes part in the activity.

In case of incorrect performance that equipment, which does not operate properly, can be filtered out immediately, that is the source of the fault in quality can be stopped.

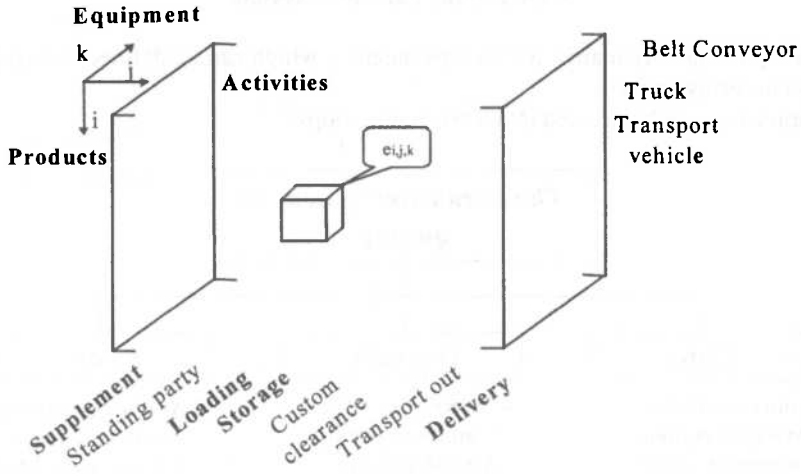


Fig. 6: The equipment matrix [2]

Quality matrix

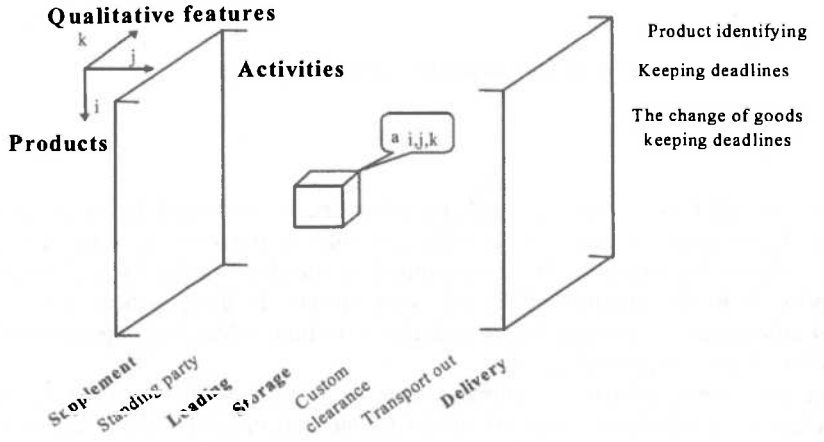


Fig. 7: Quality matrix [2]

3. Qualifying paramter system

To fill up the quality matrix we need parameters, which can be defined, interpreted and calculated in reality as well.
These parameters can be divided into three main groups:

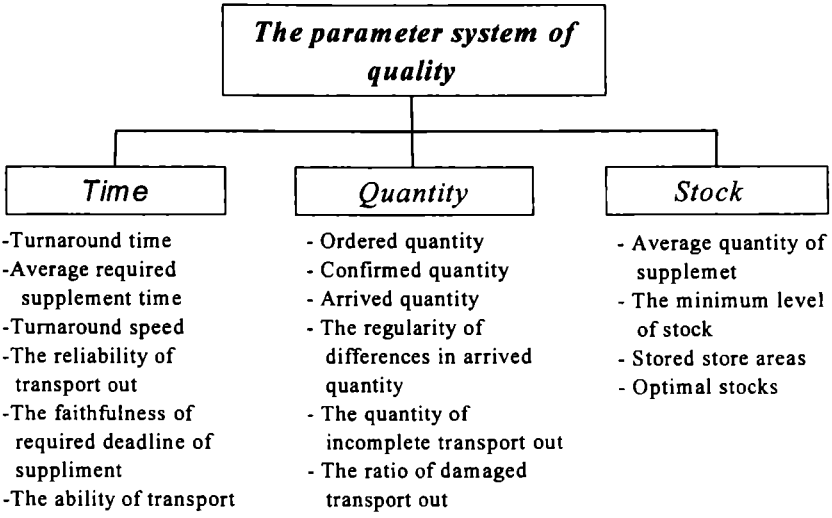


Fig. 8: The parameter system of quality

Time

The turnaround time of the logistical processes can be measured by parameters. We have to make comparison points to be able not only to measure the time but also to determine whether the measured turnaround time and speed is suitable for the company. It may be possible by the creation of the milestone system. In the logistical processes the individual milestones are situated before and after activities, which are important from the point of view of the company. (Fig. 9.)

During the investigations the optimal turnaround time of the individual logistical processes has to be determined with which the turnaround time of real processes may be compared later. The group of „time" parameters contains not only parameters with hour, minute dimension but all the ratios, which are related to the time of logistical processes, too.

Quantity

As in the case of time the quantitative parameters of processes and products, which take part in the processes, also belong here.

Beside the quantity of the transported product the quantitative features related to products are the followings:

The number and quantity of incomplete transports,
The number and quantity of surplus transports,
The number and quantity of damaged transports.

The ratios related to quantitative parameters also belong here.

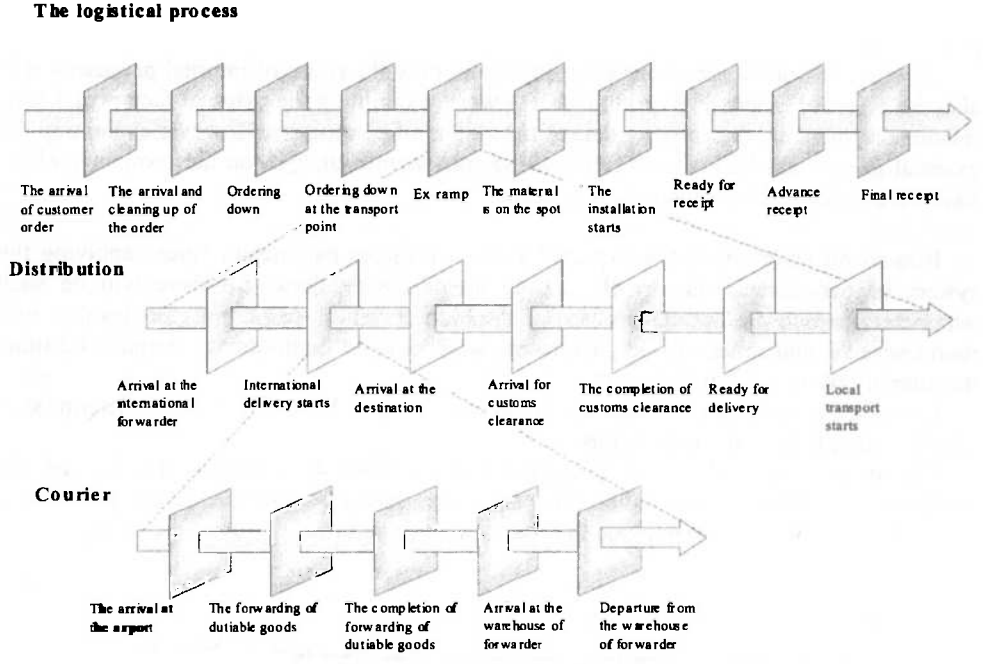


Fig. 9: The system of milestones

Stocks

Although the stocks are not direct features of the processes, they determine the operation of the logistical system basically. In case of a company, which has incorrect stock management the financial resources drain away or accumulate unnecessarily almost, unnoticed. In the event of high level of supplies because of the accumulated stocks the situation of the company from the point of view of liquidity worsens. But in the case of low level of supplies there is a constant force of order. As a result of this, the company is unable to take advantage of the larger quantities provided by the suppliers.

The parameters of stored, ordered, sorted out stocks and the demand for the territory of warehouse, which is necessary for storage, belongs to this group of parameters. This leads us to the field of costs but this topic is needless to examine in general because it is specified on the company.

We need to be aware of the demands and purchasing habits of the company's customers, and the turnaround time and operation of the processes of the company and the suppliers to be able to determine the optimal level of stocks.

In case of small companies where the number of customers and suppliers is also low the processes and the related critical length of route can be determined simply. In the event of a large company planning and examination can be solved only with the help of informatics. Where the system of parameters worked out by me can be one part of the system of data processing.

However the operation of the company is not only the result of internal processes, it is also the effect of external environment. It is worth examining the external factors and their relations influencing the entering data of the system of parameters. However examining the external factors can lead to the effects of macroeconomic changes on the company, which have been examined for centuries by experts.

It is worth determining the expected values of all the parameters before applying the system of parameters, also by the setting up of the database. So there will be such parameters, which are optimal in case of the highest value – e.g.: transport loyalty, turn round speed – and there will be parameters, which should be minimal – turnaround time, the quantity of incomplete transports.

It is not enough to know what we would like, we also have to be able to determine in which cases are needed interventions.

The method applied in the Statistical Process Control is suitable for solving the problem. According to this method different limits should be allocated to the parameters. Such limits are the critical limit and the limit of intervention.

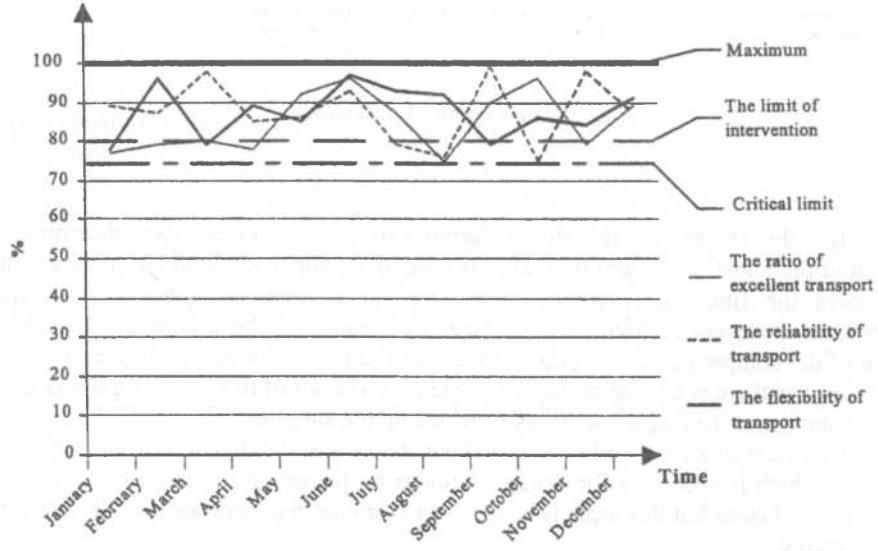


Fig. 10: The changes of the parameters in the function of time

Of course in the same diagram either only those features can appear which should be maximized or only those, which should be minimized. The critical limit means that if the feature decreases under this limit the satisfaction of the customer sustains a lasting loss. When the reliability of transport is not satisfactory the customer looks for an other supplier. None of the companies can afford such kind of loss of market in the sharp competition of today. To avoid such situations we should introduce the limit of intervention, which provides the opportunity of correcting negative trends.

Applying the limits in this form means only to keep up the quality, which is completely insufficient for a company, which is in a leader position on the market. The principles of constant refinement can be realized with the help of the system if we review the values of critical limit and the limit of intervention continuously and accept only those performances, which are better and better. Applying the system of parameters makes sense and is of importance only if we are really curious to know the state and quality of the logistical processes. Of course the matrixes and parameters are not able to refine the quality of internal and external logistical activities on their own. They can only help to outline the present situation. By means of this the critical points can be determined and also the necessary decisions can be made. It is the responsibility of the quality management whether it applies it or not.

After completing the analysis the quality management can determine the necessary steps to bring the activities of the company to perfection and to reduce costs. Another advantage of applying matrixes is that if the company contacts a new product then it is not necessary to create a new a matrix, it is enough only to complete the actual one.

The advantage of the measurement of time

The turnaround time of the logistical processes of the company can be shown by orders,
The place and the reason of the differences compared to the milestones and the person who is responsible for it can be determined unambiguously,
Expedient arrangements of correction can be done,
The deadlines of transports can be kept more accurately, the customers are more satisfied,
The logistical processes can be traced up-to-date,
We can plan more accurately how to place the orders in time,
It is possible to prepare for the auditing of quality insurance with the help of numerical quality features,
The possibilities of refinement can be shown and identified.

The results of monitoring stocks and quantitative parameters

The logistical system is completely clear-cut,
The qualification of suppliers is more founded,
The data of corrections related to quantity can be allocated to the auditing of quality insurance,
- A more rational stock management can be developed,

The surplus costs arising because of inadequate warehouse management can be shown unambiguously,
The optimal level of stock can be set by products,
Reorders can be scheduled more accurately,
The order units can be optimalized,
The quantity of the materials which should be sorted out can be reduced,
The surplus costs arising because of qualitative problems can be shown,
The consequences of qualitative unsuitability can be reduced (penalty),
The processes can be optimal zed expediently,
The arising costs can be planned in time,
The projects can be scheduled more tightly.

The advantage of creating the system of parameters is that the company does not have to develop a new information system; it only has to make use of the data of the actual system. In this way the cost of introducing is insignificant compared to the advantages we can profit by it.

4. Conclusions

Reading the article one may miss concrete parameters. The reason for the lack of them is that during my researches I was given the opportunity by the Siemens Company to realize my ideas in practice. According to the agreement with the company no data can be published in connection with its processes, thus the specified parameters of the company also cannot be.

At present the creation of the database is in progress, the survey of the logistical processes and the collection of documents and information sources are over. After finishing the stage of data collection, later on the estimation, then the setting of the critical limit and the limit of intervention follow. And after the trial operation comes the real operation which hopefully will be able to support the decisions of the leaders of the Siemens Company efficiently and to increase the satisfaction of the company's customers.

One may not forget, that this is only a system, which supports decisions; the leaders and experts should make the decisions.

REFERENCES

1. HALÁSZNÉ SIPOSS ERZSÉBET: *Logisztika, szolgáltatások versenyképesség,* Logisztikai Fejlesztési Központ, Magyar Világ Kiadó, 1998, pp. 73-75.
2. CSÉPES, O. S.: *The assessment of variations of logistic system at service companies from the point of view of quality assurance, 2nd* PhD Conference, Engineering Section, Miskolc, 1999, pp. 267-275.
3. KING, B.: *How valid are our QA assumptions: an examination of underpinning Axioms*, Accreditation and Quality Assurance, Vol. 4, Issue 8, 1999, pp. 326-335, ISSN 1432-0517.