A META-HEURISTICS FOR MANUFACTURING SYSTEMS OPTIMIZATION

Roman Stryczek

Summary

The paper presents a package of developed computer tools, suitable for generation of robust plans of operation that are efficient with respect to different criteria. The package is composed of user-friendly graphical interface based on a special class binary, timed, priority Petri nets, and of modified, high-performance genetic algorithm for sequence of contention tasks optimization. An illustrative simple example has been used to demonstrate this methodology.

Keywords: automated manufacturing systems, Petri nets, genetic algorithms

Metaheurystyka dla celów optymalizacji systemów wytwarzania

Streszczenie

Przedstawiono opracowany pakiet komputerowych narzędzi, zdolny do generowania efektywnych planów operacji, z uwzględnieniem różnych kryteriów. Pakiet składa się z przyjaznego interfejsu użytkownika bazującego na specjalnej klasie binarnych, czasowych, priorytetowych sieci Petriego oraz ze zmodyfikowanego, wydajnego algorytmu genetycznego dla ustalania optymalnej kolejności konkurujących zdarzeń. Zamieszczono prosty przykład demonstrujący opracowaną metodykę.

Słowa kluczowe: zautomatyzowane systemy produkcyjne, sieci Petriego, algorytmy genetyczne

1. Introduction

Presently, automated manufacturing systems create a complex combination of machines, industrial robots, conveyors, materials, measuring instruments, controllers, knowledge and information. These systems are typical examples of asynchronous concurrent systems. In the recent years both industry and academic world have been more and more interested in techniques used to model, analysis and control of complex production systems. In [1] is presented an overview of some important methodologies for discrete dynamic systems. Advance in information and communication technology have forced industrial activities aimed at usage of computers to manufacturing evaluation, planning and control.

Address: Roman STRYCZEK, Ph.D. Eng., University of Bielsko-Biała, Department of Manufacturing Technology and Automation, 43-309 Bielsko-Biała, 2 Willowa St., E-mail: r_stryczek@poczta.onet.pl

One from the main issues in the discrete event dynamic system planning is workflow scheduling problem. Optimal allocation of limited resources (operators, robots, tools, fixtures, pallets, buffers, …) to a tasks over time has been the prime subject of the research on scheduling problems. In general case, scheduling problem is seen as a very complex and NP-hard problem. One from approaches to solve this problem is usage of the Petri nets in conjunction with other methods. Usage of the Petri nets in modeling of the scheduling problems is not a new idea. During the last three decades, much researchers have been dealt simultaneously with the Petri nets and scheduling problems [2, 3]. The Petri nets have been used extensively in discrete manufacturing systems for specification of complex workflows.

Usage of simulation techniques can help to propose and evaluate modification as well as to improve the manufacturing performance. The simulation enables also quick adaptation of a manufacturing system to production of a new or modified products. The simulation enables generation and recording of reachability graph of the net. It is space of net's state. Basing on the reachability graph analysis of the Petri net model it is possible to develop efficient deadlock prevention policies, which can optimize usage of system resources [4]. One from the most extended Petri nets, called as *Extended neuro-fuzzy Petri net,* solves machine-loading problems in the FMS [5].

2. The class of Petri net for manufacturing systems modeling

Concept of the Petri nets has its origin in Carl Adam Petri doctor's thesis "*Kommunikation mit Automaten"* in 1962. Nowadays, the Petri nets constitute a powerful tool to modeling and analyzing of logical behavior in complex discrete systems of events. Petri nets are a graphical language. As result, the Petri nets constitute intuitive, easy to learn and user-friendly graphical interface. Binary Petri nets with inhibitor arcs can be used only to modeling of control commands using IF/THEN rules. These Petri nets do not include a notation on time, which is needed for scheduling problems. In timed Petri nets, the time is usually assigned to transitions. During the firing, suitable markers of the transition are restricted and not available for other transitions. Efficient usage of the resources (robots, workers) in a manufacturing system requires a real-time resource allocation policy, in compliance with resources assigned to a job. It creates a source of conflict and confusion (Fig. 1). The priority Petri nets were used to solve the scheduling problems [6]. In the priority Petri nets, a numbers (priorities) are assigned to the transitions. When more than one transition is enabled, only the one with the highest priority will be selected to fire.

The following $PN = (P, T, E, I, D, R, M)$ structure is called as Binary Timed Priority Petri Net (BTPPN) with inhibitor arcs, where:

- P is a non-empty, finite set of places (conditions),
- T is a non-empty, finite set of transitions (events), disjoint with P, $P \cap T = 0$, $P \cup T \neq 0$,
- $E \subseteq (P\ 3\ T) \cup (T\ 3\ P)$ is the incidence relation,
- I is the set of inhibitor arcs,
- D: $T \rightarrow N_0$ is a duration function,
- R: $T \rightarrow [0,1]$ is the priority function,
- M_0: $P \rightarrow \{0,1\}$ is the function of initial marking,
- $N_0 = N \cup 0$, N is the set of natural number.

E is the set of $p \rightarrow t$ and $t \rightarrow p$ direct arcs. An inhibitor arcs connects a place to a transition. The inhibitor arc disables the transition when the place contains tokens and enables the transition when the place is empty. D defines the firing delay of each transition. M is in fact a unit vector with $|P|$ components, where the components which are equal to 1 (true) correspond to the places where the marker is located, 0 (false) otherwise. Vector M representing actual state of Petri net model.

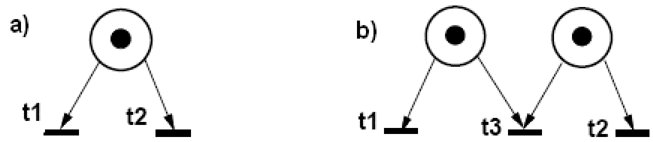

Fig. 1. The problem of resources allocation: a) conflict,
b) confusion (conflicts + concurrency)

3. Genetic algorithm to solve the scheduling problem

Presently, the computational intelligence methods [7], such as genetic algorithms [8], simulated annealing, ant colony [9], tabu search, or hybrid methods involving more than one solution approach, are used to solve complex optimization problems. Genetic algorithms (GA) are computing algorithms constructed in analogy with the process of evolution. Now, this method could be used to solve a variety of different problems, such as training of neural nets, extraction of image feature extraction, image feature recognition, process route sequencing [10] and also job scheduling [8]. Genetic algorithm presented in this paper is designed in the following way (Fig. 2 and 3):

1. Chromosome representation. A single chromosome represents exactly one solution. Solution of the problem corresponds to the priority function R. Therefore, each chromosome is made up of $|T|$ genes. Where $|T|$ is number of elements of the set T. Each gene represents priority of a single transition. The priorities take values from the range of [0,1].

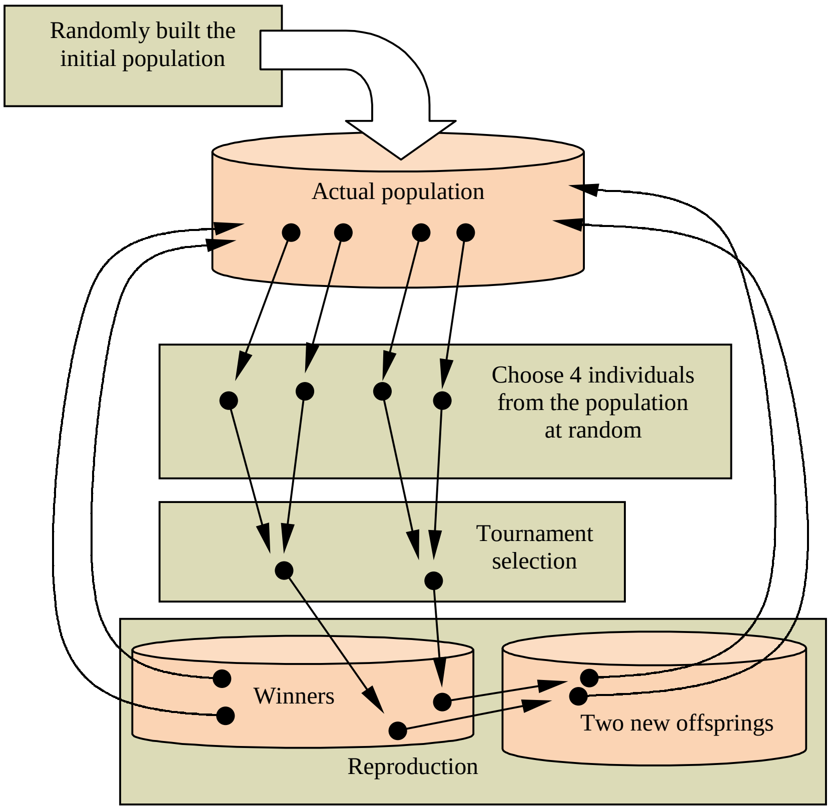

Fig. 2. Flowchart for genetic programming

2. Initializing population. All chromosomes of the first generation are randomly generated.

3. Function of fitness. The user can choose one of two fitness function:

- maximizing number of firing for choice transition,
- minimizing time to achievement of choice place.

4. Application example

An example of two-machine cell design (Fig. 4), that illustrates application of the proposed approach is shown below. The automated manufacturing system is composed of input conveyor, two CNC lathes, measuring station, industrial robot and output conveyor. The robot transfers a workpiece from the input conveyor to the machine tools, and from the machine tools to the measuring station or directly to the output conveyor. Automatic measuring station measures alternately workpieces machined on the first and the second lathe.

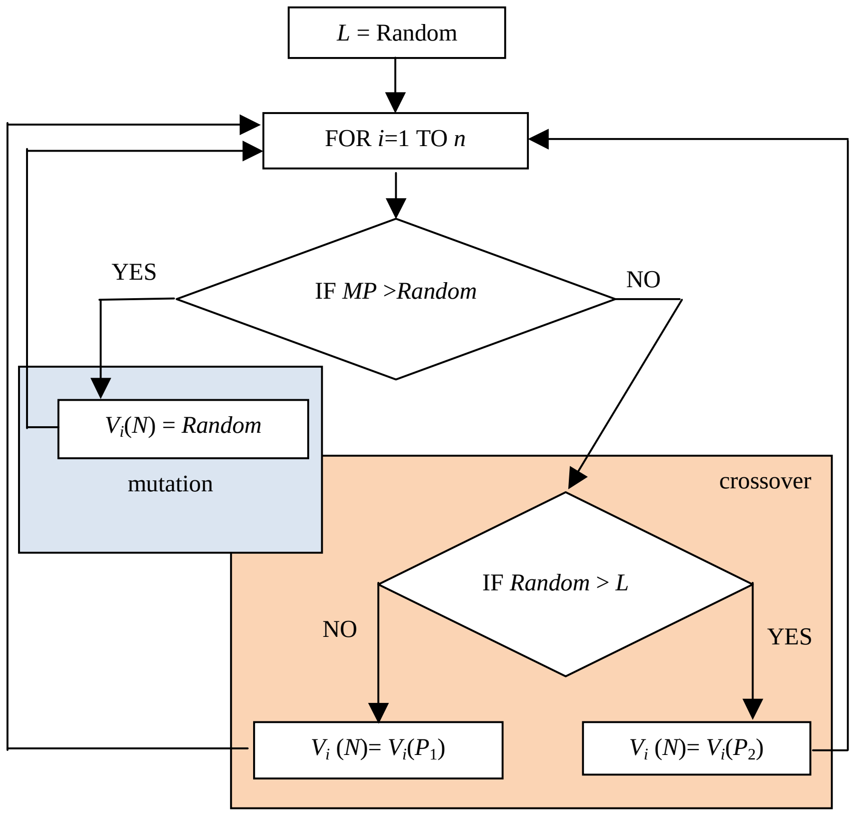

Fig. 3. Flowchart for reproduction operator, where: P_1 – first parent index, P_2 – second parent index, N – new offspring index, *Random* – randomly selected number in the range [0,1], V_i – value of i-th gene in the range [0,1], n – number of genes in chromosomes, *MP* – mutation probability, L – probability that the first parent will pass i-th gene

Below a list of places and transitions is presented (Table 1), model of simulation of the Petri net (Fig. 5) and results of genetic algorithms (Fig. 6). Figures 6 and 7 shows that for uncomplicated system can to gain a few various optimal solution. Table 2 contain only transitions connected to place *p*1. Additionally, freeze out the field for which reference transitions are members one of loops: *t*1-*t*11-*t*21 or *t*1-*t*11-*t*31 or *t*2-*t*12-*t*22 or *t*2-*t*12-*t*32 or *t*31-*t*41-*t*51-*t*32-*t*42-*t*52.

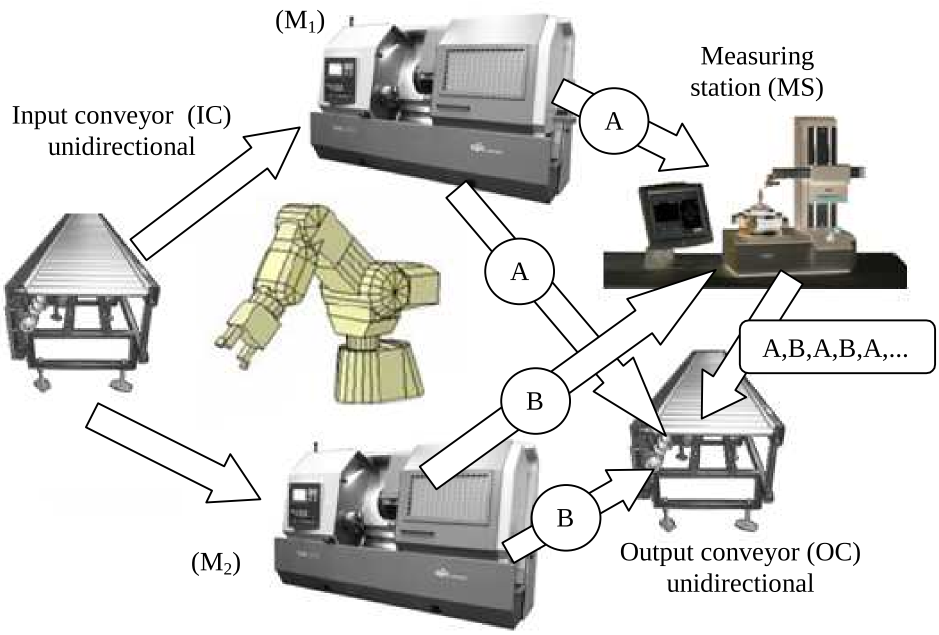

Fig. 4. Two-machine cell workflow schema

Table 1. List of the places and transitions

Places	Transitions
*p*1: Robot - idle	*t*1: Loading of workpiece on M1
*p*10: Workpiece waiting on IC	*t*2: Loading of workpiece on M2
*p*11: M1 - idle	*t*10: Movement of IC
*p*12: M2 - idle	*t*11: Machining process on M1
*p*21: M1 - workpiece before machining	*t*12: Machining process on M2
*p*22: M2 - workpiece before machining	*t*21: Transfer A from M1 to OC
*p*31: M1 - workpiece after machining	*t*22: Transfer B from M2 to OC
*p*32: M2 - workpiece after machining	*t*31: Transfer A from M1 to MS
*p*41: Required measurement A	*t*32: Transfer B from M2 to MS
*p*42: Required measurement B	*t*41: Measurement of A
*p*51: A before measurement	*t*42: Measurement of B
*p*52: B before measurement	*t*51: Transfer A from MS to OC
*p*61: A after measurement	*t*52: Transfer B from MS to OC
*p*62: B after measurement	*t*99: Movement of OC
*p*98: Workpiece on OC	
*p*99: OC without workpiece	

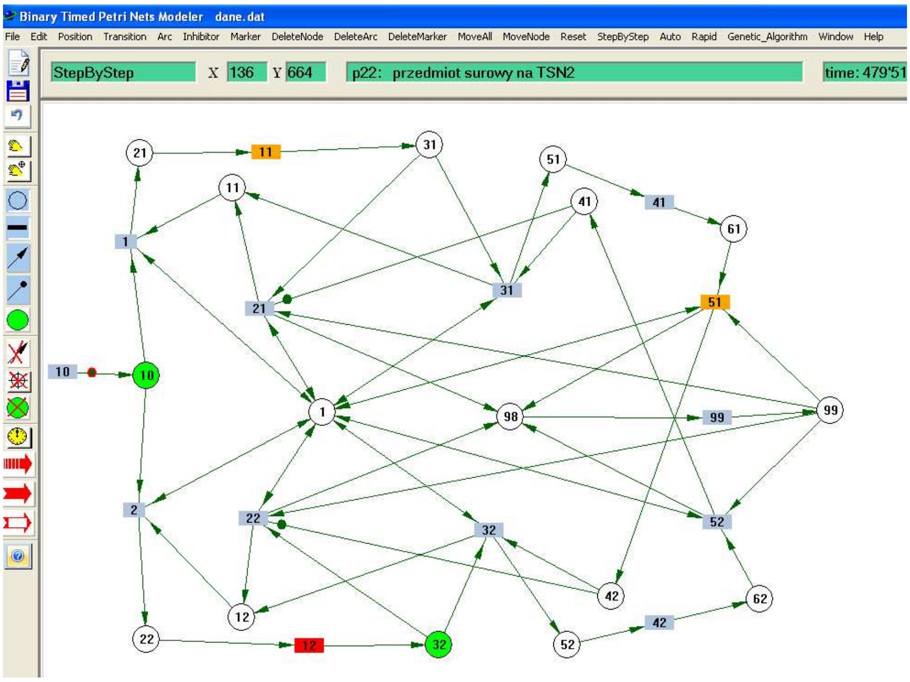

Fig. 5. Model of simulation of the priority Petri net

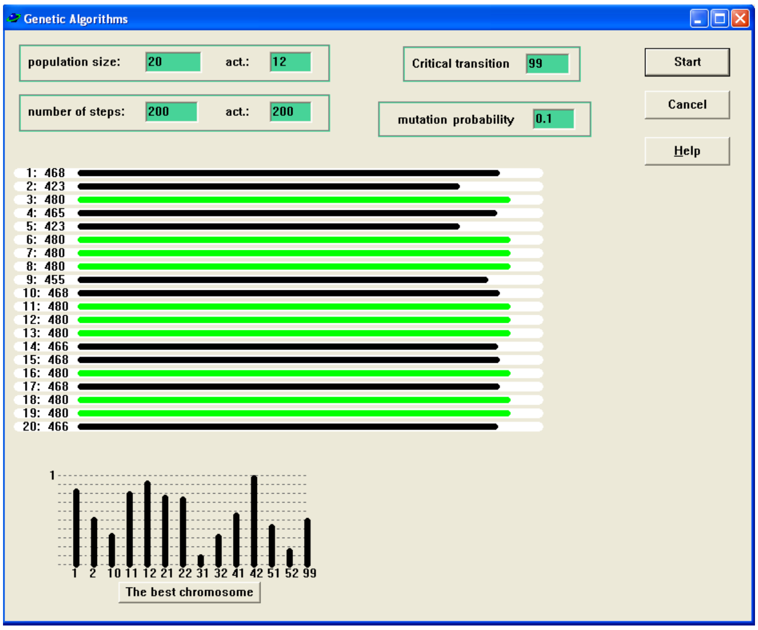

Fig. 6. Result of genetic algorithm

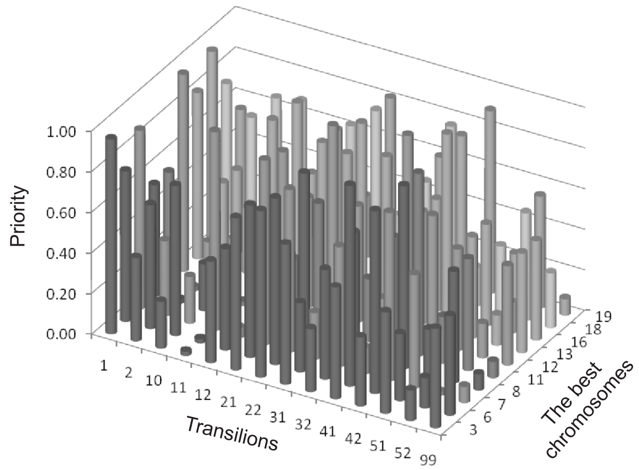

Fig. 7. The set of the best chromosomes

Basing on the Table 2, one can establish relationship of significant priorities:

VP={*t*1→*t*2, *t*1→*t*22, *t*1→*t*32, *t*1→*t*51, *t*1→*t*52, *t*2→*t*31, *t*2→*t*52, *t*21→*t*2, *t*21→*t*22, *t*21→*t*32, *t*21→*t*51, *t*21→*t*52, *t*22→*t*31, *t*22→*t*51, *t*22→*t*52}.

Having at disposal the VP, one can simplify the Petri net to the (*P*, *T*, *E*, *I*, *D*, M_0) class, very easy to practical implementations. It is necessary to remove priority function and add a few inhibitor arcs as in the Figure 8, accordingly to the VP.

Table 2. Binary matrix of vital priority (VP) for the best chromosome from Figure 6

	*t*1	*t*2	*t*21	*t*22	*t*31	*t*32	*t*51	*t*52
*t*1	x	**1**	1	**1**	1	**1**	**1**	**1**
*t*2	0	x	0	0	**1**	1	0	**1**
*t*21	0	**1**	x	**1**	1	**1**	**1**	**1**
*t*22	0	1	0	x	**1**	1	**1**	**1**
*t*31	0	0	0	0	x	0	0	0
*t*32	0	0	0	0	1	x	0	1
*t*51	0	0	0	0	1	1	x	1
*t*52	0	0	0	0	1	0	0	x

1 if t_{52} must prior to t_{22}, 0 otherwise

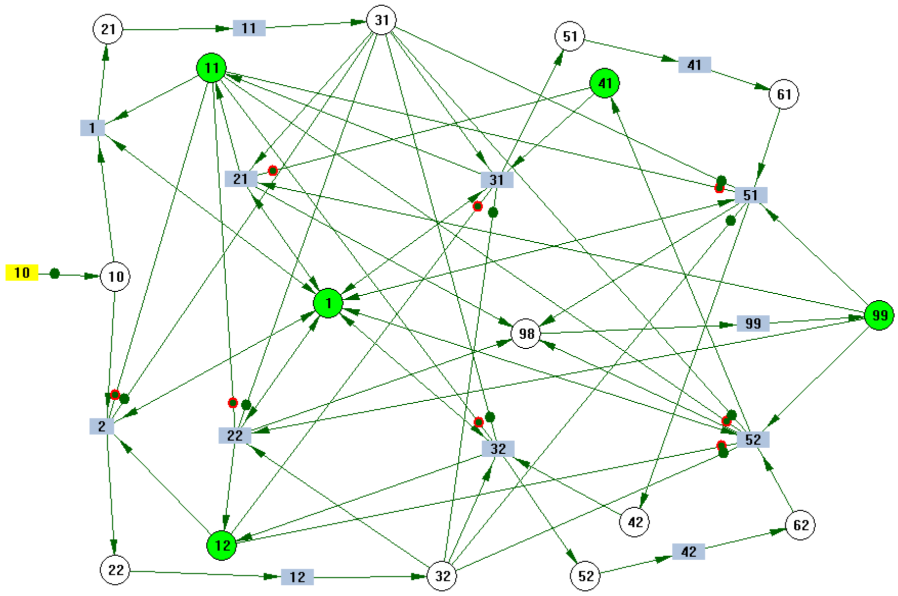

Fig. 8. Complete model

5. Conclusions

The presented paper shows that it is possible to make optimal workflow schedule with aid of an appropriate tools. The developed tool, as a modeler of the Binary Timed Petri Net, offers a graphical, user-friendly interface to plot suitable net and auxiliary graphical elements. This free software package in its standard version also contains truly effective function of genetic algorithm to optimization of scheduling problems. The main differences between the approach proposed here and these already existing are in the genetic algorithm, particularly in the reproduction. Results of the performed experiments show that this developed methodology significantly reduces complexity and time of computations. The approach presented here can be used for a similar purpose, such as assembly and disassembly planning or workplace set-up planning.

References

[1] L. R. BEN-NAUM, R. BOEL, L. BONGAERTSs, B. De SCHUTTER, Y. PENG, P. VALCKENAERS, J. VANDEWALLE, V. WERTZ: Methodologies for discrete event dynamic systems: A survey. *Journal A*, **36**(1995)4, 3-14.

[2] W.M.P. VAN DER AALST: Petri net based scheduling. Computing Science Reports, no 95. Eindhoven University of Technology, Eindhoven 1995.

[3] G. TUNCEL, G.M. BAYHAN: Applications of Petriego nets in production scheduling: A review. *J. Advanced Manufacturing Technology*, **34**(2007)7-8, 762-773.

[4] M. UZAM: The use of the Petriego net reduction approach for an optimal deadlock prevention policy for FMS. *J. of Advanced Manufacturing Technology*, **23**(2004) 3-4, 204-219.

[5] R.R. KUMAR, A. KUMAR SINGH, M.K. TIWARI: A fuzzy based algorithm to solve the machine-loading problems of a FMS and its neuro fuzzy Petriego net model. *J. of Advanced Manufacturing Technology*, **23**(2004)5-6, 318-341.

[6] K. RAVI, O.V. KRISHNAIAH CHETTY: Priority nets for scheduling flexible manufacturing systems. *J. of Manufacturing Systems*, **12**(1993)3, 326-340.

[7] R. STRYCZEK: Computational intelligence in computer aided process planning – a review. *Advances in Manufacturing Science and Technology*, **31**(2007)4, 77-92.

[8] L, LI, J.Y.H. FUH, Y.F. ZHANG, A.Y.C. NEE: Application of genetic algorithm to computer-aided process planning in distributed manufacturing environments. *Robotics and Computer-Integrated Manufacturing*, **21**(2005), 568-578.

[9] W.B. ZHANG, L.Z. HUA, Z.G. YU: Optimization of Process Route by Genetic Algorithms. *Robotics and Computer-Integrated Manufacturing*, **22**(2006), 180-188.

[10] T. HSU, O. KORBAA, R.DUPAS, G. GONCALVES: Genetic algorithm for F.M.S. cyclic scheduling. Proc. 4th Conf. Francophone de Modélisation et Simulation MOSIM'03, Toulouse 2003.

Received in June 2009