FOUNDATIONS OF COMPUTING AND DECISION SCIENCES
Vol. 38 (2013) No. 4

DOI: 10.2478/fcds-2013-0015 ISSN 0867-6356
e-ISSN 2300-3405

GENETIC ALGORITHM MODIFICATION FOR PRODUCTION SCHEDULING

Tomasz BRZĘCZEK[*], Dariusz NOWAK[**]

Abstract. Scheduling of production soundly affects its capacity especially if system does complex production jobs. In the theoretical part of the article an overview of the scheduling methods proposed in the literature was presented. In this paper it was stated a variant of job shop problem, in which jobs can overlap in some machines and omit others. Authors designed and presented here genetic algorithm to optimize solution of such a problem. The algorithm finds jobs sequence priority and in accordance with it schedules operations and calculates their completion time. An adequate problem was met in an examined plant, where 20 production jobs consisted of 11 to 20 operations assigned to at most 15 machines. Such big parameter numbers are crucial for big formal models and their solution algorithms. The designed algorithm proved to deal with parameters scale, as it found the schedule with 23,8% shorter jobs completion time in comparison with FIFO heuristic, that has been used so far by the plant.

Keywords: production scheduling, the job shop problem with machines overlapping and omission, production scheduling heuristics, genetic algorithm

1. Introduction

According to D. Waters [14] „*capacity* is the maximum amount of the product that can be made in a given time”. In the literature the capacity term is used synonymously to a maximum output. The capacity is known to be very complex variable, as it is a function of the:

- Number and capacity of machines. Machines must be appropriate to the products demand and plant size, so that all the production jobs are performed and facilities utilization is high [14]. Newer machines usually have a larger capacity than older.

[*] Poznan University of Technology, tomasz.brzeczek@put.poznan.pl
[**] Poznan University of Technology graduate, dariuush@gmail.com

- Machines layout in the production hall. It should minimize transport costs during a production process.
- *Production schedule*. Timetable for jobs or even operations. It should meet time expectations for particular and all jobs completion.
- Number of work shifts – usually one, two, three or four shifts system is used. More shifts allow to raise capacity.
- Human factor. Commitment to work, motivation, skills and education are important factors of efficiency and capacity [1, 2].

E*ffective production capacity* specifies the actual output of the production system [10]. A company receives orders from clients. Order completion plan is a *production job*. *Production job* describes *technology operations*, needed machines types and time of completion [10]. *Production schedule* is a plan of all production jobs completion. Allocation of operations complicates, if there are more jobs than machines or when a few or more operations await the same machine. Such a situation occurs in small-scale production or customized and complex production. Scheduling problem is a dynamic programming issue, because operations scheduling sequence affects results.

Production scheduling has three main objectives. The first is to complete production in accordance with time-limit of customer orders. This is an important objective in the case of products with a long production cycle. The second objective is to minimize the time of order realization from submission of the contract by the customer till manufacturing the ordered products. It means less time taken by the production cycle. The third objective concerns the resources economy. The company usually wish to fully exploit the expensive equipment and personnel. Assuming that time equals money, the last objective was formulated here as minimization of all tasks completion time. Concerning only overall time minimization, company in case of jobs congestion have to stop one close to the completion to proceed with one that has many operations to go. At different stages of the scheduling one job can pass second and vice versa, so this is operation scheduling (so called the job shop problem with job priorities changing over time). If jobs order is unknown but the same for each workstation this is the job shop problem with jobs priorities constant over time. The problem with constant priorities has less solutions than the problem with changing priorities, if problem parameters are the same. However both of them are dynamic and NP-hard. In case of twenty jobs the number of solutions is 20! (i.e. $2{,}4 * 10^{18}$ solutions).

2. Production job scheduling methods

Review of all solutions is very time-consuming even for powerful computers. Therefore in the literature a number of heuristic methods were proposed for jobs scheduling problem [10]:

- FIFO principle (First in First out). The first job to be scheduled is the first order placed in the company, so jobs are ordered by order reception date. This principle is easy to use, if orders are big and come with time lag. It causes also the possibility of delays of the important or all jobs realization.

- The principle of the most important job priority. Jobs weights are assigned individually by a decision maker; production starts with higher priority jobs, and those with lower priority may wait and be performed later.
- The principle of the shortest job priority. The shortest job is made quickly and the execution of time-consuming job lasts longer,
- Principle of the completion date priority. Basis for preparation of the schedule is the due date of the job. A job with an earlier date of implementation has priority in the schedule. This policy, however, may cause unnecessary delays in the implementation of the subsequent jobs.

To illustrate the performance of particular heuristics, let's consider a hypothetical problem of two jobs scheduling. Job description includes: job number / name of product / list of following operation work machines (capital letters) and the execution time (h):

- 1 / Product I / *A*-10 *B*-5 *C*-10 *D*-10 *E*-5,
- 2 / Product II / *A*-5 *C*-10 *B*-5 *E*-10 *D*-5.

```
A:(1-Op1)0-10  (2-Op1)10-15
B:             (1-Op2)10-15                               (2-Op3)35-40
C:                          (1-Op3)15-25  (2-Op2)25-35
D:                                        (1-Op4)25-35                          (2-Op5)50-55
E:                                                        (1-Op5)35-40 (2-Op4)40-50
```

Figure 1. Production jobs schedule thanks to FIFO method

Figure 1 shows the production schedule set with use of FIFO method. The entry in the schedule includes: (job number-operation number), the starting and finishing time for operation (h). Therefore, the entry: "(1-Op1) 0-10" means that operation 1 of job 1 starts at 0 and ends at 10 at machine *A*. In accordance with the FIFO method job 1 goes first and job 2 awaits for the machine *A*, *C* and *E*. Job 1 is completed after 40 h, and both jobs are completed after 55 h.

Starting with job 2 is consistent with the principle of shorter job priority. The received schedule is shown in Figure 2. Job 2 ends after 35 h, both jobs end after 50 h. One and both jobs are made quicker than in FIFO method.

```
A:(2-Op1)0-5 (1-Op1)5-15
B:                      (2-Op3)15-20 (1-Op2)20-25
C:           (2-Op2)5-15                          (1-Op3)25-35
D:                                                  (2-Op5)30-35 (1-Op4)35-45
E:                                 (2-Op4)20-30                               (1-Op5)45-50.
```

Figure 2. Production jobs schedule with priority of the the shorter job 2

Jobs can be done even faster, allowing variable jobs order on various stages of the realisation. The price is that first job is completed later than in other methods schedules, As illustrated in Figure 3, jobs are completed after 45 h due to fact that on the machine *A* job 2 goes first and on the machine *B* job 1 goes first.

A:	(2-Op1) 0-5	(1-Op1) 5-15				
B:			(1-Op2) 15-20	(2-Op3) 20-25		
C:		(2-Op2) 5-15		(1-Op3) 20-30		
D:					(1-Op4) 30-40	(2-Op5) 40-45
E:					(2-Op4) 30-40	(1-Op5) 40-45.

Figure 3. Schedule with changing jobs order from: 2,1 to 1,2

In this example there are four time-effective operations sequences. Figure 4 shows the last effective allocation of operations. On the machine *A* job 2 awaits for job 1, and vice versa on the machine *C*.

A:	(1-Op1) 0-10	(2-Op1) 10-15				
B:		(1-Op2) 10-15		(2-Op3) 25-30		
C:			(2-Op2) 15-25	(1-Op3) 25-35		
D:					(1-Op4) 35-45	(2-Op5) 45-50
E:					(2-Op4) 30-40	(1-Op5) 45-50.

Figure 4. Operations schedule with changing jobs order: from 1,2 to 2,1

The above simple example shows approximation character of heuristic solution and genetic algorithm as well. Researchers have also developed many formal models for operation scheduling. These are variants of standard problem: the job shop or the flow shop problem [see: 7]. First time effective solving method was so called Johnson's algorithm, but it works only for simple case concerning two workstations and *N* jobs going through them in series [13]. Some models are based on the dynamic programming methods [8], others proposed genetic algorithm as an appropriate solution method [3; 15]. Models are often extended by Boolean variables. Recently hybrid algorithms have been proposed [11]. Here we present not Boolean formulation of the job shop problem with job constant priorities over time and machines overlapping or omission. We assume that different jobs consist of various numbers of operations. We propose genetic algorithm and original simple algorithm for solution scheduling and aim function calculation [compare: 5]. Therefore we predict that the algorithm can deal with big number of machines, jobs and operations that we found in practice of the examined mechanical plant.

3. The model and algorithm design

According to D. E. Goldberg [6] genetic algorithms are review algorithms based on the natural principles of selection and evolution. It means mutation and crossover of new solutions with big representation of best known solutions, what can be used in many scientific fields [4, 9]. Second principle is random effect on new solutions creation. The job shop schedule problem stated in this paper was based on problem assumptions met in practice of a small mechanical plant work [compare: 7]:

1. There are N production jobs set $J = \{J_1, J_2, \dots J_n, \dots, J_N\}$ known at the beginning of scheduling. The solution is any jobs order in planning them into schedule (priorities of jobs are constant over time).
2. There are Z machines set $M = \{M_1, M_2, \dots M_z, \dots, M_Z\}$.

3. Job n-th consists of set of operations $O_n = \{O_{1nz}, O_{2nz}, \dots O_{pnz}, \dots, O_{Pnz}\}$, where p means order of operation in n-th job operations sequence. P_n is the number of all operations of n-th job. Index z denotes number of machine needed for operation pn. Jobs can overlap in some machines and do not go through all machines. This is not classic job shop problem [8].
4. Each machine can proceed only one operation at the time. Operations proceed without delays till appropriate machine is free.
5. Aim is to minimize completion time of all jobs.

A genetic algorithm bases on genetic terminology [12]. The algorithm was designed for production jobs scheduling so its elements received new sense (see Table 1).

Table 1. The designed genetic algorithm elements and functions in terms of production jobs scheduling

Algorithm	The designed production jobs scheduling algorithm
Individual - -solution	Any jobs schedule in production
Population	Given set of production jobs schedules
Genotype	Any permutation of jobs cardinal numbers
Gen	Job cardinal number
Allel	Order of the job cardinal number in the permutation
Chromosome	Subsequence of permutation of jobs cardinal numbers
Crossover	Linking two permutations into new two permutations
Mutation	Replacement of random two cardinal numbers of jobs permutation

Solution encoding is an important problem here. There are N jobs, that we numbered with natural numbers: 1, 2, …, N. Let $v = (v_1, v_2, \dots, v_s, \dots, v_N)$ be any permutation of those N jobs, where element v_s is s-th job in permutation that has s-th priority in scheduling. Algorithm codes solution permutation using only jobs numbers. For example solution is:

$$v = (v_1, v_2, v_3, v_4, \dots, v_N) = (J_4, J_3, J_1, J_{12}, \dots, J_5) = (4, 3, 1, 12, \dots, 5), \tag{1}$$

that means N jobs are scheduled and job 4 is scheduled first, job 3 goes second into the schedule, job 1 goes third, job 12 goes fourth and so on till the last scheduled job, that has number 5. Completion time function f (aim and fitness function in one) is following:

$$f(v) \to \min, \tag{2}$$

Any solution of the genetic algorithm should be assessed with value of *aim function f*. Knowing given by genetic algorithm solution v, *operations scheduling and aim function calculation algorithm* was proposed in this paper in following steps:

1. Define Z sets $t_1, \dots, t_Z$. These are times of machines availability. Starting time availability of every machine is given as a time interval $[0, b]$, where b is the maximum work time of every machine per day. After scheduling of all operations I_z requiring machine z, set t_z includes no more than $I_Z + 1$ ranges of free work time:

$$t_z = \{[a_{1z}, b_{1z}], [a_{2z}, b_{2z}], \dots, [a_{IZ+1,z}, b]\}, \tag{3}$$

where a_{iz} and b_{iz} are moments of beginning and ending of machine free time i-th period.

2. Set $v_{s=1}$. Algorithm starts with first job to be done in a given solution v. Read number of this job n.
3. Set number of n-th job operation $p = 1$ (algorithm starts with first operation) and read from data sets time of its completion t_{np} and needed machine number z.
4. Calculate moment of operation start a_{np} as in formula:

$$a_{np} = \min_i a_{iz} \tag{4}$$

subject to

$$a_{iz} + t_{np} \le b_{iz}, \tag{5}$$

$$a_{iz} \ge a_{np-1} + t_{np-1}. \tag{6}$$

Criterion chooses the earliest moment, when operation can start, from beginning moments of machine z free time periods. Constraint one ensures choice with such beginning moments, that free time periods are long enough to complete the operation. Constraint two ensures that earliest operations of this job were done (this condition concerns second and further operations).

5. Calculate actual machine z free time set as a difference of sets:

$$t_z' = t_z \,/\, [a_{np}, a_{np} + t_{np}]. \tag{7}$$

6. Go to step 3 and raise operation number p by 1 if $p < P_n$, where P_n is number of operations of n-th priority job. Otherwise go to step 7.
7. Go to step 2 and raise job priority number s by 1, if $s < N$, where N is total number of jobs. Otherwise go to step 8.
8. Calculate jobs completion time function value f for the solution v, using formula:

$$f(v) = \max_z \left(\wedge_z \max_{i \in I_z} (a_{iz} - 1) \right) \tag{8}$$

$$\underline{\vee}$$

$$f(v) = b, \quad b \notin t_z'. \tag{9}$$

Formula finds for each machine end of work moment, finding maximum value of free time period beginning moments reduced by a time unity. Then finds maximum for all Z workstations, unless the whole time b_z is spent.

4. The algorithm work

The genetic algorithm work flow chart is presented in Figure 5. Initial population is random, excluding first individual, that is FIFO solution of the problem (it has been used so far in the plant). Genotype of this individual consists of job numbers in the ascending order (1, 2, 3, 4, 5, …, N). Individuals number in population equals Nd_1, so depends on number of scheduled jobs N and algorithm parameter d_1, that controls population scale. Number of populations also depends on jobs number N and algorithm control parameter d_2. Small values of d_1, d_2 allow to economy time consumption through calculations.

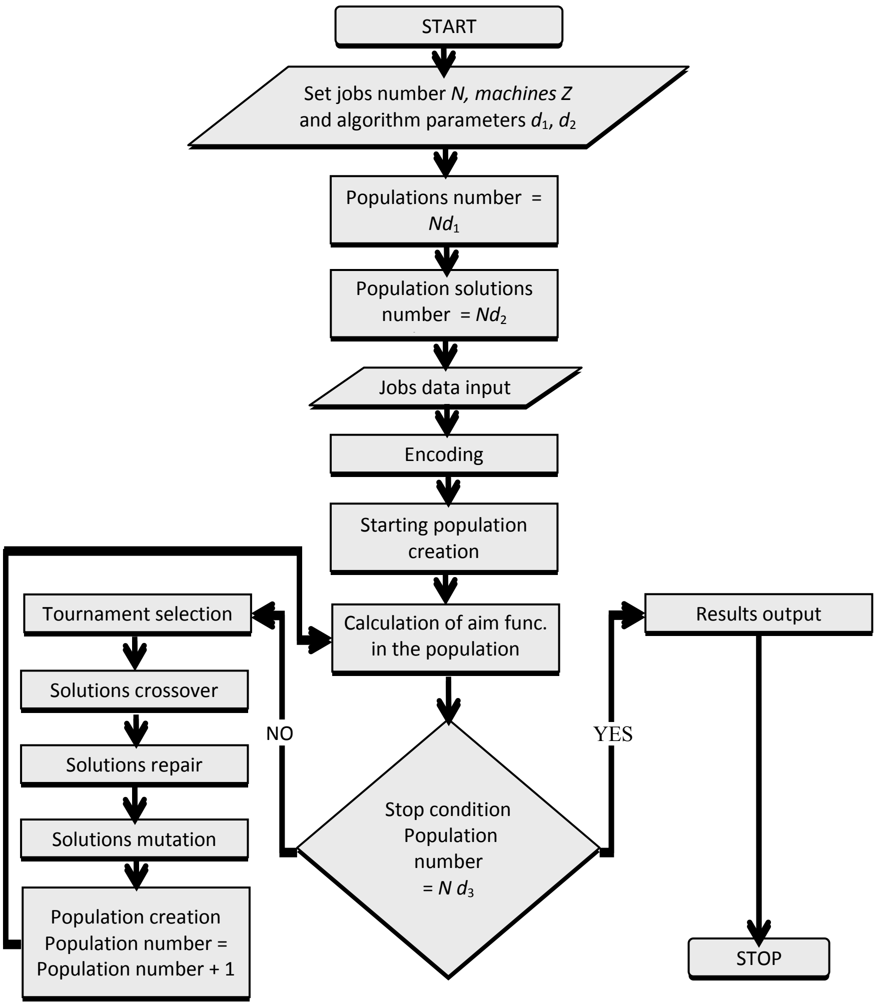

START
Set jobs number N, machines Z and algorithm parameters d1, d2
Populations number = Nd1
Population solutions number = Nd2
Jobs data input
Encoding
Starting population creation
Calculation of aim func. in the population
Stop condition Population number = N d3
NO
YES
Results output
STOP
Tournament selection
Solutions crossover
Solutions repair
Solutions mutation
Population creation Population number = Population number + 1

Figure 5. Flow chart of genetic algorithm for jobs scheduling

After creation of each population of solutions, the algorithm evaluates it individual solutions using aim (fitness) function algorithm written in the previous section. Algorithm proceeds with generating of new population of solutions, which should give shorter completion time, until stop condition is met. Stop condition allows to terminate the algorithm after formation of Nd_3 number of populations, where d_3 is control parameter. "Parent" solutions are selected with method called a tournament. The first stage of the method figures on drawing the eight solutions to the tournament population. Comparison in pairs is made until the best solution is selected from the tournament group. The solution will be involved in reproduction. This procedure applies until ends formation of whole population of "parent" solutions.

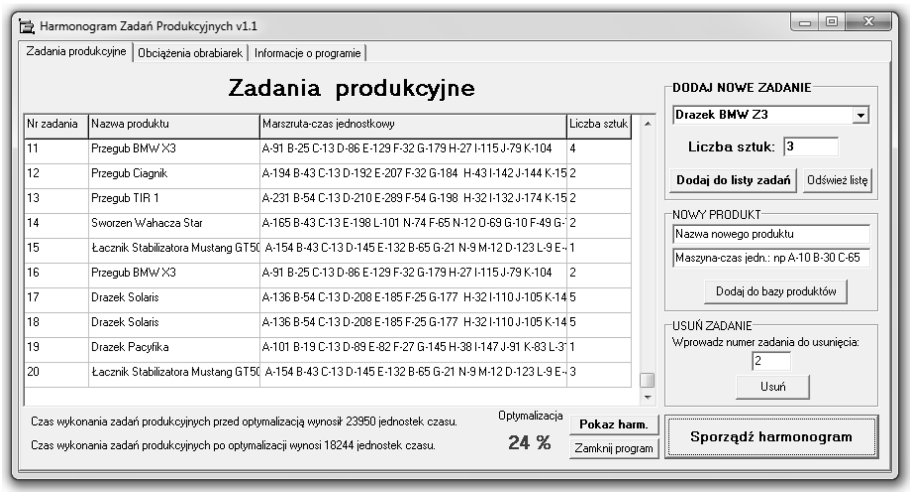
Than *crossover* and *mutation* operations on "parent" solutions lead to new solutions. Crossover operation includes repair procedure in case that in new solution one job repeats. *Mutation* meant replacement of two random jobs in the solution. Number of mutations in a population raises progressively with number of production jobs, which determines complexity of task. In task with more than 40 jobs 10 percent of population solutions go through mutation, but in small task with less than 5 jobs mutation concerns less than 1 percent of solutions. Such a progressive mutation provides diversity of solutions in small and in big populations.

5. The algorithm parametrization and results

The designed algorithm was implemented by the own application, called Production Jobs Scheduler v1.1 (Harmonogram Zadań Produkcyjnych v1.1), programmed in Delphi language. The application accepts from 2 to 50 production jobs for scheduling. In the end application outputs schedule in the form presented in Figures 1-4. In Figure 6 screen saver of application interface is presented.

A mechanical plant of steering systems was examined, in particular its production plan in the 19-th of January 2012. Parameters received from this plan and technology documentation were:

1. 20 production jobs listed in the production plan. Each job was descripted with following technology operations, time of their realization and appropriate machines. Jobs examined in the plant had from eleven to twenty operations.
2. The plant has 15 machines coded with letters A to O. One of them was matched to each operation in the production job description. In the application we coded machines with Latin alphabet letters starting from A, as it is easier to code and read data especially when there are two other numerical indexes. In this case index $z = \{A, B, \ldots, O\}$ but nothing more changes in the aim function algorithm.
3. Operations lead time was received from the company technology documentation. Saving business secret of the company, time was recalculated to hypothetical unit. Lead time of job operations was from 13 to 289 hypothetical unit.
4. Preparing time of operations was skipped, as it was not presented in the plant technology documentation.
5. Transportation time was constant and included in operation time. It was assumed 1 hypothetical unit for each change of machine. It was also not presented in the plant technology documentation.
6. Optimization result will be compared with jobs completion time achieved in the plant using FIFO method.
7. Algorithm control parameters d_1, d_2, d_3 were set to appropriate values: 5, 14, 5.



Zadania produkcyjne

Nr zadania	Nazwa produktu	Marszruta-czas jednostkowy	Liczba sztuk
11	Przegub BMW X3	A-91 B-25 C-13 D-86 E-129 F-32 G-179 H-27 I-115 J-79 K-104	4
12	Przegub Ciagnik	A-194 B-43 C-13 D-192 E-207 F-32 G-184 H-43 I-142 J-144 K-15	2
13	Przegub TIR 1	A-231 B-54 C-13 D-210 E-289 F-54 G-198 H-32 I-132 J-174 K-15	2
14	Sworzen Wahacza Star	A-165 B-43 C-13 E-198 L-101 N-74 F-65 N-12 O-69 G-10 F-49 G-	2
15	Łacznik Stabilizatora Mustang GT5	A-154 B-43 C-13 D-145 E-132 B-65 G-21 N-9 M-12 D-123 L-9 E-	1
16	Przegub BMW X3	A-91 B-25 C-13 D-86 E-129 F-32 G-179 H-27 I-115 J-79 K-104	2
17	Drazek Solaris	A-136 B-54 C-13 D-208 E-185 F-25 G-177 H-32 I-110 J-105 K-14	5
18	Drazek Solaris	A-136 B-54 C-13 D-208 E-185 F-25 G-177 H-32 I-110 J-105 K-14	5
19	Drazek Pacyfika	A-101 B-19 C-13 D-89 E-82 F-27 G-145 H-38 I-147 J-91 K-83 L-3	1
20	Łacznik Stabilizatora Mustang GT5	A-154 B-43 C-13 D-145 E-132 B-65 G-21 N-9 M-12 D-123 L-9 E-	3

Czas wykonania zadań produkcyjnych przed optymalizacją wynosił 23950 jednostek czasu.

Czas wykonania zadań produkcyjnych po optymalizacji wynosi 18244 jednostek czasu.

Optymalizacja 24 %

DODAJ NOWE ZADANIE

Drazek BMW Z3

Liczba sztuk: 3

NOWY PRODUKT

Nazwa nowego produktu

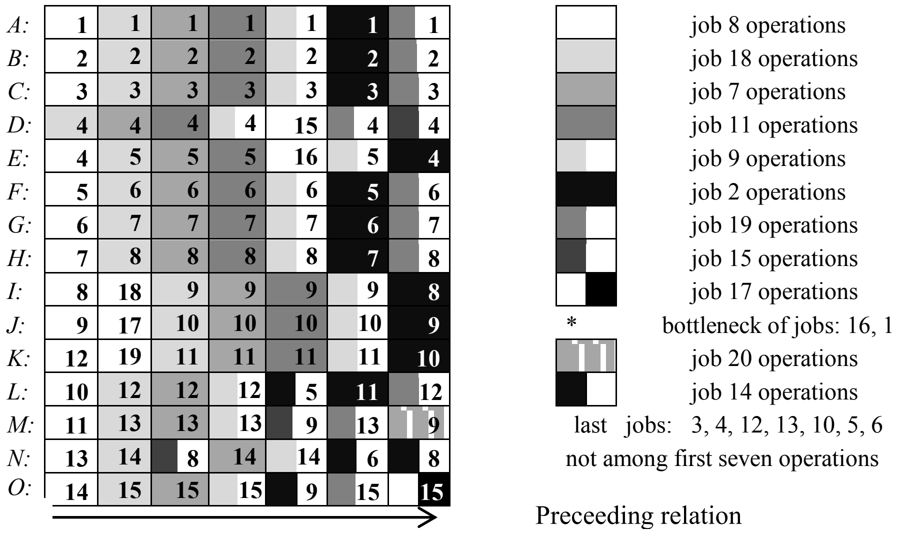
Maszyna-czas jedn.: np A-10 B-30 C-65

USUŃ ZADANIE

Wprowadz numer zadania do usunięcia: 2

Figure 6. Interface of the designed algorithm application

The application found the shortest completion time at 18244 hypothetical unit. This completion time is achieved for following priority of jobs (first one has the highest priority): 8, 18, 7, 11, 9, 2, 19, 15, 17, 16, 1, 20, 14, 3, 4, 12, 13, 10, 5, 6. Received schedule has 15 rows of machines and width of 9 pages (after omission of workstations downtime). Therefore we put Figure 7 that contains only beginning 7 operations (columns) on each machine.



A:	1	1	1	1	1	1	1
B:	2	2	2	2	2	2	2
C:	3	3	3	3	3	3	3
D:	4	4	4	4	15	4	4
E:	4	5	5	5	16	5	4
F:	5	6	6	6	6	5	6
G:	6	7	7	7	7	6	7
H:	7	8	8	8	8	7	8
I:	8	18	9	9	9	9	8
J:	9	17	10	10	10	10	9
K:	12	19	11	11	11	11	10
L:	10	12	12	12	5	11	12
M:	11	13	13	13	9	13	9
N:	13	14	8	14	14	6	8
O:	14	15	15	15	9	15	15

Figure 7. The beginning 7 operations numbers and jobs for each machine
(preceeding relation preserved only for a given machine, downtimes were omitted)

The schedule in Figure 7 informs that machine *A* should start work with first operations of jobs 8, 18, 7, 11, 9, 2 and 19 and machines *B*, *C* continue this flow of subsequent operations, but other machines are omitted by some jobs. So far the company has used FIFO method. FIFO schedule planned completion of the same jobs in time of 23950 hypothetical unit. FIFO schedule has job sequence 1, 2 … 20. The designed algorithm and application allowed to decrease time of jobs completion by 23,8% in comparison to FIFO method. Lead time of jobs in the found sequence is following: 1590, 2290, 1815, 880, 1624, 1887, 1593, 1104, 2290, 880, 1456, 1104, 1178, 2122, 1481, 1345, 1539, 1593, 1624 in hypothetical units. It was noticed, that the found order of jobs is not affected by their lead times neither numbers of operations included in jobs. Sorting jobs in ascending order of it operations numbers gives a result: 20, 20 20, 11, 20, 20, 20, 16, 20, 11, 20, 16, 14, 19, 20, 11, 11, 20, 20.

6. Summary

Job or operation scheduling concerns especially small series production of complex products. The paper discussed the job shop problem with jobs constant priorities over time, but with new assumptions that jobs can overlap in some machines and do not go through all machines. The designed algorithm consists of genetic algorithm for jobs sequence finding and nested algorithm of operations sequence scheduling and completion time calculation [compare: 5]. Full review of solutions is very time-consuming. Big formal models has constraints in optimization parameters number and calculation time. The designed algorithm dealt with big number of 20 production jobs, each consisting of 11-20 operations using at most 15 machines. The algorithm found the solution that allowed to decline all jobs completion time by 23,8% in comparison with FIFO method. The solution is not related to known heuristics.

References

[1] Barney J.B., Clark D.N., *Resource-Based Theory: Creating and Sustaining Competitive Advantage*, Oxford University Press, New York 2007, 18.

[2] Beksiak J., *Economics (Ekonomia),* PWN, Warsaw 2001, 112.

[3] Błażewicz, J., Domschke, W., Pesch, E., The job shop scheduling problem: Conventional and new solution techniques. *European Journal of Operational Research*, 93, 1996, 1-30.

[4] Eiben A.E., Smith J.E., *Introduction to Evolutionary Computing, Natural Computing Series*, Springer, Berlin 2008.

[5] Gao J., Gen M., Sun L. A hybrid of genetic algorithm and bottleneck shifting for flexible job shop scheduling problem, GECCO '06: *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, ACM 2006, 1158.

[6] Goldberg D.E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Dorling Kindersley Pvt Ltd, New Delhi 2008, 17.

[7] Grimes D., Hebrard E., Model and strategies for variants of the job shop scheduling problem, *CP'11: Proceedings of the 17th international conference on Principles and practice of constraint programming*, Springer-Verlag, Berlin 2011.

[8] Gromicho J.A.S., van Hoorn J.J., Saldanha da Gamma F., Timmer G.T., Solving the job shop scheduling problem optimally by dynamic programming, *Computers & Operations Research*; 39, 12, 2012, 2968-2977, DOI: 10.1016/j.cor.2012.02.024

[9] Manolas D.A., Gialamas T., Tsahalis D.T. A genetic algorithm for the simultaneous optimization of the sensor and actuator positions for an active noise and/or vibration control system, *Inter-Noise*, 96, 1996, 1187-1191.

[10] Pająk E., *Production Management (Zarządzanie produkcją)*, PWN, Warsaw 2006, 28, 202-232.

[11] Qing-dao-er-ji R., Yuping W., A new hybrid genetic algorithm for job shop scheduling problem, *Computers & Operations Research*, 39, 10, 2011 DOI: 10.1016/j.cor.2011.12.005.

[12] Rutkowski L., *Methods and Techniques of Artifical Intelligence (Metody i techniki sztucznej inteligencji)*, PWN Warsaw 2005, 233.

[13] Sikora W., *Operations Research (Badania operacyjne)*, PWE, Warsaw 2008, 212.

[14] Waters D., *Operations Management. Producing Goods and Services*, Pearson Education Limited, Essex 2002, 220.

[15] Zhenyuan J., Xiaohong L., Jiangyuan Y., Defeng J., Research on job-shop scheduling problem based on genetic algorithm, *International Journal of Production Research* 49, 12, 2011 DOI: 10.1080/00207543.2010.481293.

Presented at the XII Conference: Systems and Operational Research – BOS 2012, 17-19 September 2012, Warsaw, Poland