

Marcin Wawrzyniak*

**AN EVOLUTIONARY ALGORITHM FOR JOB SCHEDULING
IN MULTIMACHINE ENVIRONMENTS**

Introduction

The constant rise in competitiveness forces contemporary businesses to optimize use of their own resources and the appropriate process management is one of the ways of gaining an advantage over other companies. Task scheduling is a part of the operational research and despite its long history, it is still popular among researchers. This interest stems from the fact that task scheduling has great practical significance and from the fact that many of the considered problems do not have satisfactory solutions. It is worth mentioning that task scheduling modules (which support production management or supply chain management) are an element of all modern ERP systems. Despite the fact that the main aim of scheduling systems is to solve practical problems, most of the papers published nowadays focus on solving problems which are simplified models of systems existing in the real world. One of the most often used models of multimachine environment is the job-shop problem. In this model, the sequence of operations in jobs is forced and differs in each of them. For the job-shop model, benchmark problem sets were proposed which allow to compare the results obtained by individual authors.

There are two approaches of solving scheduling problems exact and approximate methods. The exact method, which guarantee obtaining optimal solutions, due to time and computational limitations may be used only when the number of operations is lower than a few hundreds. A good example illustrating the weakness of the exact method is the facts that benchmark problems consisting of 400 operations proposed by Taillard¹ in 1993 despite years of intensive research are still unsolved and the

* Author is preparing a PhD thesis in the Department of Information Systems under supervision of Professor Witold Abramowicz.

¹ E. Taillard, *Benchmarks for basic scheduling problems*, European Journal of Operational Research, 64/1993, s. 278-285.

difference between the upper and the lower bound reaches a few percent. Since in practice, the number of operations is usually higher than a few thousand, the application of exact methods for their optimization is impossible. Difficulties in solving practical problems with exact methods cause that nowadays most of the works lean towards the approximate methods such as taboo search, simulated annealing or evolutionary algorithms. The above-mentioned techniques do not guarantee obtaining an optimal solution but allow to get results just little worse from the optimal one in a short period of time.

Interest in evolutionary algorithms stems from the fact that this technique does not enforce any specific course of action. Since elements of system (such as: a problem representation, operators and their working conditions, selection method, the mechanism responsible for maintaining a population diversity) may be implemented in many various ways, evolutionary algorithms are flexible tools with great potential. Evolutionary algorithms are techniques based on the nature which allow to obtain very good results in searching extremes of functions. The basic elements of the evolutionary algorithms are: individuals (solutions of the problem), population (a set of individuals) and evolutionary operators - crossover (creating new solutions on the basis of two or more individuals) and mutation (creating new solutions on the basis of a small modification of the individual existing in the population). Individuals, being part of the population, take part in selection process, during which some of them are replaced with the new solutions created using evolutionary operators.

1. Related work

Many publications concerning the use of evolutionary algorithms for task scheduling describe new versions of operators and analyze their properties^{2, 3}. Applied operators have many parameters (such as probability of crossover and mutation or number and method of selecting operations performed by an operator) and, therefore, authors try to establish their optimal working conditions. A good example of such approach is the

² M. Watanabe, K. Ida, M. Gen, *A genetic algorithm with modified crossover operator and search area adaptation for the job-shop scheduling problem*, Computers & Industrial Engineering, 48/2005, s. 743-752.

³ F.T.S. Chan, S.H. Chung, P.L.Y. Chan, *An adaptive genetic algorithm with dominated genes for distributed scheduling problems*, Expert Systems with Applications, 29/ 2005, s. 364-371.

paper of Pongcharoen et. al. in which they tried to establish optimal values of crossover probability, mutation probability, population size and number of cycles⁴ and of Ponnambalam et. al. where optimal values of crossover probability, mutation probability and number of cycles were determined⁵. The calculation of optimal values of operator parameters is a complex problem because, in most cases, these values change during the simulation (the optimal values at the beginning of the simulation, when the diversity of the population is high, differ from the optimal values at the end of the simulation, when the diversity of the population is low). Taking this into account, some authors propose solutions in which values of operator parameters are established on-line using information about the current state of the population. Such a solution was presented by Kim, Gen and Yamazaki⁶ in their study on scheduling the resource-constrained projects where authors proposed to connect evolutionary algorithm with the fuzzy logic system, which was responsible for establishing working conditions.

The evolutionary algorithm does not have to be the only element of the scheduling system, but it may be connected with other techniques of task scheduling. An example of such a connection was presented in the work of Tanev et. al who optimized the production system in a factory manufacturing plastic elements using an evolutionary algorithm cooperating with priority dispatching rules⁷. Tested rules included: FIFO (First Input First Output), TIS (Time In System), SPT (Shortest Processing Time), LPT (Longest Processing Time), DT (Due Time), ST (Start Time). In the proposed solution, the evolutionary algorithm was used to select a priority rule and the selected priority rule was employed to determine which order (from the list of the unscheduled orders) was to be executed next. According to Tanev et. al, such a hybrid procedure allows to use advantages of both techniques - the low computational cost

⁴ P. Pongcharoen, C. Hicks, P.M. Braiden, D.J. Stewardson, *Determining optimum genetic algorithm parameters for scheduling the manufacturing and assembly of complex products*, International Journal of Production Economics, 78/2002, s. 311-322.

⁵ S.G. Ponnambalam, B. Jawahar, S. Kumar, *Estimation of optimum genetic control parameters for job shop scheduling*, International Journal of Advanced Manufacturing Technology, 19/2002, s. 224-234.

⁶ K.W. Kim, M. Gen, G. Yamazaki, *Hybrid genetic algorithm with fuzzy logic for resource-constrained project scheduling*, Applied Soft Computing, 2-3F/2003, s. 174-188.

⁷ I. Tanev, T. Uozumi, Y. Morotome, *Hybrid evolutionary algorithm-based real-world flexible job shop scheduling problem: application service provider approach*, Applied Soft Computing, 5/2004, s. 87-100.

of priority dispatching rules and the ability of avoiding from local optima of evolutionary algorithms. Also Kumar and Srinivasan applied a similar approach when they compared the efficiency of seven dispatching rules with hybrid evolutionary algorithm⁸. Similarly, to Tanev et. al, the evolutionary algorithm was used to select a priority rule, and the selected priority rule was applied later on to determine which order (from the list of the unscheduled orders) was to be executed next. The proposed algorithm was tested on a practical job-shop problem with 80 jobs and 59 machines (the shortest job was performed on 2 machines and the longest job was performed on 37 machines). The obtained results were compared with the results obtained by each of dispatching rules used in hybrid algorithm. The schedules created using the proposed hybrid algorithm were 3% better than schedules created by using the best rule and 30% better than the schedules created by the worst one.

Since for many of the most frequently applied scheduling criteria of the job-shop problem optimal solution is always an element of a set of active schedules, therefore some of authors propose to limit the set of considered solutions to active schedules or to schedules without delays, which constitute part of a set of active schedules. This a kind of approach to solving scheduling problems is presented by Mattfeld and Bierwirth⁹. In their study they connected an evolutionary algorithm with the system allowing to create active schedules and schedules without delays. They also proposed to introduce a delay factor (describing maximal idleness of the machine), which allows to create parameterized active schedules. The schedules created using the delay factor approach to active schedules (when the value of the factor is big) or to schedules without delays (when the value of the factor is small). Such a procedure allows limiting the number of analyzed solutions by removing these schedules in which long periods of idleness take place. The presented algorithm was tested on 12 benchmark problems and four criteria of optimization: weighted mean flow time, weighted mean tardiness, maximum tardiness of jobs and weighted number of tardy jobs were used.

Mechanism responsible for maintaining diversity of the population constitute a crucial part of the evolutionary algorithm. It allows to

⁸ H. Kumar, G. Srinivasan, *A genetic algorithm for job shop scheduling – A case study*, Computers in Industry, 31/1996, s. 155-160.

⁹ D. Mattfeld, C. Bierwirth, *An efficient genetic algorithm for job shop scheduling with tardiness objectives*, European Journal of Operational Research, 155/2004, s. 616-630.

widen space of the analyzed schedules, and prevents a premature convergence on local minima. In the simplest solutions, a high diversity is maintained by the periodic replacement of some individuals belonging to the population with new randomly generated solutions. In more sophisticated systems, new individuals are created with the assistance of functions describing similarity between the schedules (these are so called distance functions). Functions measuring similarity may be based both on the data in individuals (genotype) and the data in schedules (phenotype). The simplest distance functions return a number of operations situated on the same position in the two compared chromosomes (if they are based on genotype) or return a number of operations situated on the same position in schedules (if they are based on phenotype). If the distance function is defined in such a way, its value may vary between 0% (none of the operations is in the same position) and 100% (all of the operations are in the same position). Sakawa and Kubota proposed a more sophisticated method of calculating a similarity between individuals¹⁰. In the presented case, in the first stage an operation is selected and, on the basis of its position in the schedule, the remaining operations performed on the same machine are divided into two sets (performed before and after selected dividing operation). In the second stage all operations, which in both compared schedules, belong to the same sets (such operations which, in both schedules, are before dividing operation or, in both schedules, are after dividing operation) are counted. This procedure is repeated for each operation in the schedule, and counted operations are summed to obtain a factor which defines a similarity of the schedules.

Another element of the evolutionary algorithm, which is important for its efficiency, is the method of problem representation. The most important (apart from the quality of obtained schedules) features of representation taken into account include: possibility of an easy creation of feasible schedules and possibility of a constructing evolutionary operators. Another important issue is representation redundancy because when many different genotypes lead into the same phenotype (schedule), the increase in the computational complexity makes the scheduling problem harder to solve. The most frequent representation models are based on one of the following lists: operation, job or priority rules. A random key

¹⁰ M. Sakawa, R. Kubota, *Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy due date through genetic algorithms*, European Journal of Operational Research, 120/2000, s. 393-407.

representation is one of the most seldom used ones in scheduling. The one proposed by Bean¹¹ is universal and allows to model many other types of problems (e.g. a traveling salesman problem or quadratic assignment problem). In this type of representation, each operation (or job) has an assigned factor (a number from 0 to 1), and all the operations (or jobs) are placed into the schedule on the basis of factor values. Wang and Uzsoy applied the random key representation to minimize maximum lateness on a batch-processing machine¹². The system analyzed in their paper consists of one machine, whose work output allows simultaneous performance of a number of jobs. These jobs are treated as a group and the processing time of a given group is that of the job with the longest processing time in the batch. In the case of multimachine problems, the creation of the schedules only on the basis of factor values does not guarantee its feasibility, therefore in most cases schedules are created using different rules and factor values are used only when there are conflicts in the selected rule. Such an approach was applied by Goncalves et al. who used random key representation for modeling a job-shop problem¹³. In the presented solution, the schedules were created with Giffler and Thompson method and factors were employed only when the selected rule failed to provide unequivocal indication which operation should be put into schedule next.

Streeter and Smith¹⁴ described influence of the problem structure on its difficulty. They proved that problems in which number of jobs equals number of machines $J/M = 1$ (so called square problems) are difficult whilst problems in which $J/M \rightarrow 0$ or $J/M \rightarrow \infty$ are simple. Results obtained by Streeter and Smith explain why big benchmarks (like TA71-TA80 consisting of 100 jobs on 20 machines) can be easily solved whilst

¹¹ J. Bean, *Genetic algorithms and random keys for sequencing and optimization*, OR-SA Journal on Computing, 6/1994, s. 154-160.

¹² C.S. Wang, R. Uzsoy, *A genetic algorithm to minimize maximum lateness on a batch processing machine*, Computers & Operations Research, 29/2002, s. 1621-1640.

¹³ J.F. Goncalves, F.F. Magalhaes Mendes, M. Resende, *A hybrid genetic algorithm for the job shop scheduling problem*, European Journal of Operational Research, 167/ 2005, s. 77-95.

¹⁴ M.J. Streeter, S.F. Smith, *How the landscape of random job shop scheduling instances depends on the ratio of jobs to machines*, Journal of Artificial Intelligence Research, 26/2006, s. 247-287.

smaller problems (like TA21-TA30 consisting of 20 jobs on 20 machines) are still open.

The benchmark problem MT10 proposed by Muth and Thompson in 1963 is one of the most frequently used for testing new scheduling algorithms. It is an instance of 10 jobs and 10 machines and has a Cmax (completion time of last job) as its objective function. In spite of its small size (especially from the practical point of view) and was unsolved for 25 years. In 1989 Carlier and Pinson, applying the branch and bound method, proved that in the optimal solution the makespan equals 930¹⁵. An identical solution was obtained a year earlier (without the proof of its optimality) by Adams et. al. who used the shifting bottleneck method¹⁶. Yamada and Nakano as first found optimal solution using evolutionary algorithm in 1992¹⁷. Since the MT10 benchmark as a square problem is still considered difficult, it is widely used for testing new techniques of scheduling. Tsai and Lin proposed a hybrid algorithm (consisting of an evolutionary algorithm and a taboo search technique) for job-shop scheduling¹⁸ and tested it using, among others, the benchmark problem MT10. They examined the influence of simulation parameters (e. g. a method used for creating initial individuals, a type and a probability of crossover operator) on the obtained results. The hybrid algorithm was capable of finding optimal solutions for the best values of the above mentioned parameters and the average of the best solutions obtained in 30 simulations reached 948,8. Steinhöfel et. al. proposed a new definition of neighborhood and used it to create an algorithm based on the simulated annealing technique¹⁹. One of the benchmark problems used for testing the presented algorithm was MT10. During simulations (which in case of the MT10 problem were repeated five times), an optimal solution was

¹⁵ J. Carlier, E. Pinson, *An algorithm for solving the job-shop problem*, Management Science, 35/1989, s. 164-176.

¹⁶ J. Adams, E. Balas, D. Zawack, *The sifting bottleneck procedure for job shop scheduling problem*, Management Science, 34/1988, s. 391-401.

¹⁷ T. Yamada, R. Nakano, *A genetic algorithm applicable to large-scale job-shop problems*, Parallel Problem Solving from Nature, Brussels, Belgium 1992.

¹⁸ C.F. Tsai, F.C. Lin, *A new hybrid heuristic technique for solving job-shop scheduling problem*, IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Lviv, Ukraine 2003.

¹⁹ K. Steinhöfel, A. Albrecht, C.K. Wong, *An experimental analysis of local minima to improve neighborhood search*, Computers & Operations Research, 30/2003, s. 2157-2173.

obtained twice, a makespan of 937 - twice and a makespan of 949 - once. Park et. al. elaborated an evolutionary algorithm for job-shop scheduling²⁰ in which individuals were coded using a job list representation and schedules were created on the basis of jobs sequence on a list. The above mentioned researchers tested different methods of a creation of an initial population, different operators and different methods of selection. The employed procedure allowed to obtain optimal solutions during test using benchmark problem MT10. Wang and Zheng²¹ created a hybrid algorithm (consisting of an evolutionary algorithm with simulated annealing) for the job-shop scheduling. The authors emphasized that this combination allowed them to use advantages of both techniques: scatter search (in the case of the evolutionary algorithm) and local search (in the case of the simulated annealing). The results obtained using the hybrid system were better than those obtained by the employed techniques alone. During the simulations with the MT10 problem (which were repeated 20 times) optimal solutions were found and an average makespan was 2.5 % worse than the optimal one. Watanabe et al. proposed genetic algorithm with search area adaptation and tested it with using two benchmark problems MT10 and MT20. For the MT10 problem simulation were repeated 100 times and best found solution was 937²². Another publications in which the MT10 was one of the benchmarks used for testing of new scheduling algorithms are those of Zhang et al.²³, Huang and Liao²⁴ or Essafi et al.²⁵.

²⁰ B.J. Park, H.R. Choi, HS Kim, *A hybrid genetic algorithm for the job shop scheduling problems*, Computers & Industrial Engineering, 45/2003, s. 597-613.

²¹ L. Wang, D.Z. Zheng, *An effective hybrid optimization strategy for job-shop scheduling problems*, Computers & Operations Research, 28/2001, s. 585-596.

²² M. Watanabe, K. Idab, M. Gen, *A genetic algorithm with modified crossover operator and search area adaptation for the job-shop scheduling problem*, Computers & Industrial Engineering, 48/2005, s. 743-752.

²³ C.Y. Zhang, Y.Q. Li, P.G. Rao, Z.L. Guan, *A very fast TS/SA algorithm for the job shop scheduling problem*, Computers & Operations Research, 35/2008, s. 282-294.

²⁴ K.L. Huang, C.J. Liao, *Ant colony optimization combined with taboo search for the job shop scheduling problem*, Computers & Operations Research, 35/2008, s. 1030-1046.

²⁵ I. Essafia, Y. Matib, S. Dauzère-Pérès, *A genetic local search algorithm for minimizing total weighted tardiness in the job-shop scheduling problem*, Computers & Operations Research 35/2008 s. 2599-2616.

In papers published nowadays in many cases evolutionary algorithms are supported by some other optimization techniques such as priority rules, local search or Giffler and Thompson procedure. The application of the above mentioned techniques usually allows to obtain better results but at the same time it limits the flexibility of the evolutionary algorithm. The main disadvantage of the hybrid algorithms is that the techniques supporting the evolutionary algorithm often use specific features of the analyzed problems and do not work efficiently when the type of the problem (or objective function) is changed. Taking into account disadvantages of hybrid algorithms, it is worth to pay attention to such solutions where the evolutionary algorithm is the only element. In the further part of this study such an algorithm will be presented.

2. Evolutionary algorithm

2.1. Problem representation and schedules creation

Table 1 presents an example of a job-shop problem which the author intended to use for the description of the proposed algorithm.

Table 1. An example of a job shop problem

	O1	O2	O3
J1	M1 (T11)	M2 (T12)	M3 (T13)
J2	M2 (T21)	M1 (T22)	M3 (T23)
J3	M3 (T31)	M1 (T32)	M2 (T33)

The examined problem consists of 3 jobs (J1, J2 and J3) and each job, intern, 3 operations (O1, O2 and O3). Specific executions times (T11, T12... T32, T33) and one of 3 machines (M1, M2 and M3) on which this operations are performed are assigned to each operation.

Presented algorithm utilize random key representation in which jobs are encoded as a list of operations with a factor (being a number ranging form x to y) assigned to each of them. Initial population consists of organisms with randomly generated factors and later on, during simulation process values of factors are modified by evolutionary operators. Creation of the schedule begins with the calculation of the aggregated factors, which are sums of factors of the current operation and all operations that occurred in the job before current operation. Table 2 presents a list of operations with factor values and aggregated factors values created for problem from table 1.

Table 2. An example list of operations with factor values and aggregated factors values

job	J1	J1	J1	J2	J2	J2	J3	J3	J3
operation	O1	O2	O3	O1	O2	O3	O1	O2	O3
machine	M1	M2	M3	M2	M1	M3	M3	M1	M2
factor (an example)	34	23	63	84	36	79	11	84	35
aggregated factor (an example)	34	57	120	84	120	199	11	95	130

In the next stage, operations are assigned to machines and their sequence is established on the basis of aggregated factor values. Table 3 presents the sequence of operations on machines created according to the described rules for an example job-shop problem from table 1 and the factor values from table 2.

Table 3. Sequence of operations for a job-shop problem from Table 1 and values of aggregated factors from Table 2

M1	J1O1 (T11)-34	J3O2 (T32)-95	J2O2 (T22)-120
M2	J1O2 (T12)-57	J2O1 (T21)-84	J3O3 (T33)-130
M3	J3O1 (T31)-11	J1O3 (T13)-120	J2O3 (T23)-199

The schedule in Table 4 is created on the basis of the sequence of operations on machines and information about relationship between operations and operation execution times.

Table 4. The schedule obtained from the sequence of operations from Table 3

M1	J1O1 (T11)-34	J3O2 (T32)-95		J2O2 (T22)-120	
M2		J1O2 (T12)-57	J2O1 (T21)-84	J3O3 (T33)-130	
M3	J3O1 (T31)-11		J1O3 (T13)-120		J2O3 (T23)-199

The introduction of aggregated factors reflects the positions of operations in jobs and is a development of the random key representation proposed by Bean. Since the sequence of all operations in the proposed algorithm is established only on the basis of the aggregated factor values, therefore the position of the operation in schedule depends on the position of the operation in a given job. This means that operations blocked by lower number of operations appearing before it in a job are scheduled before operations blocked by higher number of operations appearing before it in a job (aggregated factor of operations blocked by lower number

of operations appearing before it in a job are usually lower than aggregated factor of operations blocked by bigger number of operations appearing before it in a job). The application of aggregated factors shifts simulations towards more promising areas of the analyzed space of solutions. It is worth mentioning that the proposed way of representation allows to create schedules with assistance of other techniques. It is possible that values of aggregated factors may solve conflicts during the creation of active schedules using the Giffler and Thompson method or during the creation of schedules without delays.

2.2 Operators

Random key representation allows on implementation of different version of crossover and mutation operators. In this paper an crossover operator exchanging factors between individual is examined. At the beginning of the proposed procedure two organisms are randomly selected from the population. In the next stage, some operations from the list of all operations are selected and the factors assigned to these operations are exchanged. New individuals are created in such a way that for the first child factors of the unselected operations are copied from the first parent and factors of selected operations are copied from the second parent. Similarly, when the second child is created, factors of the unselected operations are copied from the second parent and factors of the selected operations are copied from the first parent. In the final stage, the aggregated factors are calculated and schedules are created. An operator exchanging factors, create two new solutions.

Number of operations exchanged by operator is a parameter called width of the operator and is determined randomly. The selection of operations performed by operator is an important issue. Operations which show up on the list next to each other belong to the same job with the exception of the first and the last in a given job (table 2). If the crossover operator selects all operations randomly, they belong to a large number of jobs, but if the selected operations appear on the list next to each other, a number of jobs performed by a given operator is lower. Bearing this in mind, two crossover operators can be proposed: exchanging factors of randomly selected operations (when all operations are selected randomly), exchanging factors of scopes of operations (when all selected operations appear one the list next to each other and first operation in the group is selected randomly).

Picture 1 shows the working mechanism of the proposed operators.

Picture 1. The operator of crossover exchanging randomly selected operations. Bold type and arrows are used to indicate the performed operations. In the case of operator exchanging scopes of operations, all performed operations occur next to each other on the list. The width of the operator in the example equals 33%.

child 1

Operation	J1O1	J1O2	J1O3	J2O1	J2O2	J2O3	J3O1	J3O2	J3O3
Factors	34	78	63	64	36	59	11	84	35

parent 1

Operation	J1O1	J1O2	J1O3	J2O1	J2O2	J2O3	J3O1	J3O2	J3O3
Factors	34	23	63	84	36	79	11	84	35



parent 2

Operation	J1O1	J1O2	J1O3	J2O1	J2O2	J2O3	J3O1	J3O2	J3O3
Factors	12	78	43	64	21	59	17	42	31

child 2

Operation	J1O1	J1O2	J1O3	J2O1	J2O2	J2O3	J3O1	J3O2	J3O3
Factors	12	23	43	84	21	79	17	42	31

2.3 Selection

Makespan is the most important criterion influencing on the replacement of one of the individuals existing in the population with the new one. If this criterion was to be the only one, the population would consist of individuals coding similar solutions in a very short time. Bearing it in mind, it appears necessary to extend selection procedure in such a way which would guarantee a diversity of the population. In this study it is achieved by the introduction of function measuring the similarity of solutions. This distance function is defined as an average module of differences in a position of operations on machines.

$$D_{ab} = \frac{\sum_{m=1}^M \sum_{n=1}^{Nm} |P_{amn} - P_{bmn}|}{\sum_{m=1}^M Nm} \quad (1)$$

where:

D_{ab} – distance between schedules a and b

M – number of jobs

m – job number

N_m – number of operations in a job m

n – operation number

P_{am_n} – position of the n th operation in the m th job in schedule a

P_{bm_n} – position of the n th operation in the m th job in schedule b

In the proposed algorithm, a new solution replaces the one existing in a given population if:

- it is better than the best one, in which case replaces the worst one
- it is better than worse of the parents and the distance between the new solution and the better parent is higher than the assumed minimal distance, in which case replaces worse of the parents
- it is better than the worst in the population and at the same time the lowest from distances between new solution and solutions present in the population is higher than the assumed minimal distance, in which case it replaces the worst in the population.

If these two criteria – makespan and minimal distance are taken into account it is possible to reduce gradually the average makespan and, simultaneously, maintain population diversity. Moreover it should be emphasize, that the value of the minimal distance is a parameter that has an essential influence on the obtained results.

2.4. Simulation results

Simulations which were performed using the benchmark problem MT10 with C_{max} as objective function were divided into two parts and in both stages, the number of individuals in population equaled 500. In the first stage, the influence of operator type and the value of minimal distance on the obtained results were tested. The tested operators included those described in a previous chapter and minimal distance took 9 values from 0,1 to 0,9. Each time a new solution was created with using the evolutionary algorithm, the width of the operator was established randomly assuming values ranging from 1% to 50%. For each of the analyzed operators and for each of the analyzed minimal distances, simulations were repeated 100 times and in each repetition 5 000 000 solutions were

created. Altogether, in the first stage 9 000 000 000 solutions were created (2 operators * 9 values of a minimal distance * 100 repetitions * 5 000 000 solutions in each repetition). The results obtained in the first stage are presented on tables 5 and 6 (top rows of the table), which show the influence of the operator type, the minimal distance value and the number of generated solutions on average from the best makespans. For in the first stage the best results were obtained for the operator exchanging scopes of factors, so in the second stage further experiments using the above mentioned operator were performed. These simulations were repeated 100 times and in each repetition 15 000 000 solutions were created. Altogether, in the second stage 18 000 000 000 solutions were created (1 operator * 9 values of minimal distance * 100 repetitions * 15 000 000 solutions in each repetition). The results obtained in the second stage are presented on table 6 (bottom rows of the table)

Table 5. Averages from best makespans obtained with using an operator of crossover exchanging randomly selected operations – simulations repeated 100 times

Number of solutions (in millions)	Minimal distance value								
	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9
0,0005	1293,0	1290,0	1288,9	1294,6	1290,2	1296,8	1291,9	1291,5	1291,9
1	968,3	961,5	962,2	967,8	972,1	976,2	981,3	989,8	997,8
2	967,2	957,3	955,9	960,8	966,1	970,1	973,6	980,3	986,9
3	966,9	955,1	953,5	957,1	962,3	966,6	970,4	975,4	980,2
4	966,5	953,7	951,7	955,4	959,6	964,2	967,7	972,4	977,2
5	966,2	952,7	951,1	953,9	957,7	962,0	965,6	969,7	974,7

It appears that the value of minimal distance is a crucial element of the evolutionary algorithm and has a significant influence on the quality of the obtained results. In an early phase of the simulation, the lower the value of a minimal distance, the lower the probability of a conflict during replacement solutions existing in population by the new ones and thus an average makespan is reduced faster. Simultaneously, when the value of minimal distance is small, the population loses its diversity fast and the creation of new valuable solutions is harder. When the values of the minimal distance are higher during the early phase of simulations an average makespan is reduced slower but because of the fact, that the population maintains its diversity in a longer period of time, it is possible to obtain better solutions (table 5 and 6).

Table 6. Averages from best makespans obtained with using an operator of crossover exchanging scopes of operations – simulations repeated 100 times

Number of solutions (in millions)	Minimal distance value								
	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9
0,0005	1298,3	1292,8	1293,7	1295,3	1291,0	1290,6	1292,2	1290,4	1291,0
1	977,6	980,3	982,6	989,1	998,9	1007,9	1015,5	1026,1	1035,4
2	972,0	970,7	970,5	976,1	982,3	987,3	993,7	1003,2	1010,9
3	970,8	966,4	964,9	970,3	975,4	978,6	983,2	991,5	998,1
4	967,1	958,0	953,9	955,2	957,7	959,6	961,2	966,0	969,3
5	965,6	953,4	945,8	944,1	944,5	945,6	946,7	948,7	948,3
10	965,1	952,2	944,3	942,0	941,9	942,7	943,3	945,5	945,5
15	964,8	951,7	943,6	940,8	940,2	941,3	941,6	943,9	944,0
20	964,8	951,5	943,1	940,2	939,4	940,4	941,2	943,2	943,3

Table 7. Solutions obtained with using an operator of crossover exchanging scopes of operations after generating 20 000 000 individuals - simulations repeated 100 times (md minimal distance value)

Md	Makespans															
	930	935	937	938	939	940	941	942	943	944	945	946	947	948	949	>950
0,1	0	1	3	1	0	0	0	0	0	0	3	0	1	0	0	91
0,2	2	13	2	2	0	0	3	1	3	0	3	1	1	0	2	67
0,3	9	19	12	2	0	0	4	4	1	3	1	2	1	3	2	37
0,4	10	17	23	7	0	0	5	4	7	2	4	0	1	0	2	18
0,5	7	25	19	5	1	1	6	3	11	3	7	0	2	0	7	3
0,6	3	12	27	14	0	0	8	4	9	6	2	0	0	1	5	9
0,7	2	11	17	13	4	0	7	3	18	2	3	0	0	0	17	3
0,8	3	5	14	2	6	1	3	5	18	2	6	1	2	2	25	5
0,9	0	4	12	7	6	3	8	2	15	1	10	0	3	2	23	4

For the best values of the minimal distance (0,4 when the highest number of optimal solutions is obtained and 0,5 when the average from the best solutions is the lowest) 57 % of the obtained solutions is optimal or worse than the optimal by less than 1 % (table 7).

Conclusion

The random keys representation allows many different implementations of crossover operators. The best results of the two tested operators were obtained for the operator exchanging scopes of operations. It was proved that the introduction of the mechanism maintaining the population diversity allows to obtain lower makespans. During the simulations with the well-known benchmark problem MT10 with Cmax as the objective functions, the optimal solutions were found. It is worth mentioning that the

proposed evolutionary algorithm does not use any properties of the objective functions (does not use any priority rules, does not create only active schedules).

References

- Adams J., Balas E., Zawack D., *The sifting bottleneck procedure for job shop scheduling problem*, Management Science, 34/1988, s. 391-401.
- Bean J., *Genetic algorithms and random keys for sequencing and optimization*, ORSA Journal on Computing, 6/1994, s. 154-160.
- Carlier J., Pinson E., *An algorithm for solving the job-shop problem*, Management Science, 35/1989, s. 164-176.
- Chan F.T.S., Chung S.H., Chan P.L.Y., *An adaptive genetic algorithm with dominated genes for distributed scheduling problems*, Expert Systems with Applications, 29/2005, s. 364-371.
- Essafia I., Matib Y., Dauzère-Pérès S., *A genetic local search algorithm for minimizing total weighted tardiness in the job-shop scheduling problem*, Computers & Operations Research, 35/2008, s. 2599-2616.
- Goncalves J.F., Magalhaes Mendes F.F., Resende M., *A hybrid genetic algorithm for the job shop scheduling problem*, European Journal of Operational Research, 167/2005, s. 77-95.
- Huang K.L., Liao C.J., *Ant colony optimization combined with taboo search for the job shop scheduling problem*, Computers & Operations Research, 35/2008, s. 1030-1046.
- Kim K.W., Gen M., Yamazaki G., *Hybrid genetic algorithm with fuzzy logic for resource-constrained project scheduling*, Applied Soft Computing, 2-3F/2003 s. 174-188.
- Kumar H., Srinivasan G., *A genetic algorithm for job shop scheduling – A case study*, Computers in Industry, 31/1996, s. 155-160.
- Mattfeld D., Bierwirth C., *An efficient genetic algorithm for job shop scheduling with tardiness objectives*, European Journal of Operational Research, 155/2004, s. 616-630.
- Park B.J., Choi H.R., Kim H.S., *A hybrid genetic algorithm for the job shop scheduling problems*, Computers & Industrial Engineering, 45/2003, s. 597-613.
- Pongcharoen P., Hicks C., Braiden P.M. Stewardson, D.J., *Determining optimum genetic algorithm parameters for scheduling the manufacturing and assembly of complex products*, International Journal of Production Economics, 78/2002, s. 311-322.
- Ponnambalam S.G., Jawahar B., Kumar S., *Estimation of optimum genetic control parameters for job shop scheduling*, International Journal of Advanced Manufacturing Technology, 19/2002, s. 224-234.

- Sakawa M., Kubota R., *Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy due date through genetic algorithms*, European Journal of Operational Research, 120/2000, s. 393-407.
- Steinhöfel K., Albrecht A., Wong C.K., *An experimental analysis of local minima to improve neighborhood search*, Computers & Operations Research, 30/2003, s. 2157-2173.
- Streeter M.J., Smith S.F., *How the landscape of random job shop scheduling instances depends on the ratio of jobs to machines*, Journal of Artificial Intelligence Research, 26/2006, s. 247-287.
- Taillard E., *Benchmarks for basic scheduling problems*, European Journal of Operational Research, 64/1993, s. 278-285.
- Tanev I., Uozumi T., Morotome Y., *Hybrid evolutionary algorithm-based real-world flexible job shop scheduling problem: application service provider approach*, Applied Soft Computing, 5/2004, s. 87-100.
- Tsai C.F., Lin F.C., *A new hybrid heuristic technique for solving job-shop scheduling problem*, IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Lviv, Ukraine 2003.
- Wang C.S., Uzsoy R., *A genetic algorithm to minimize maximum lateness on a batch processing machine*, Computers & Operations Research, 29/2002, s. 1621-1640.
- Wang L., Zheng D.Z., *An effective hybrid optimization strategy for job-shop scheduling problems*, Computers & Operations Research, 28/2001, s. 585-596.
- Watanabe M., Ida K., Gen M., *A genetic algorithm with modified crossover operator and search area adaptation for the job-shop scheduling problem*, Computers & Industrial Engineering, 48/2005, s. 743-752.
- Yamada T., Nakano R., *A genetic algorithm applicable to large-scale job-shop problems*, Parallel Problem Solving from Nature, Brussels, Belgium 1992.
- Zhang C.Y., Li Y.Q., Rao P.G., Guan Z.L., *A very fast TS/SA algorithm for the job shop scheduling problem*, Computers & Operations Research, 35/2008, s. 282-294.

