Wojciech MATUSIAK, Tadeusz WITKOWSKI
Politechnika Warszawska

ZASTOSOWANIE ALGORYTMU IMMUNOLOGICZNEGO DO OPTYMALIZACJI HARMONOGRAMU PRODUKCYJNEGO

Streszczenie. Artykuł przedstawia opracowany algorytm immunologiczny – AIS w celu rozwiązania problemu gniazdowego (JSP). Szczególną uwagę zwrócono na realizację niektórych operacji mutacji dodatkowej. Jej zastosowanie miało na celu poprawę efektywności algorytmu AIS. Przeprowadzono eksperyment komputerowy w celu określenia wpływu poszczególnych parametrów algorytmu AIS na jakość rozwiązania i porównano go z innymi rodzajami algorytmów tego typu. Opisano problem harmonogramowania produkcji za pomocą opracowanego algorytmu.

APPLICATION OF IMMUNE ALGORITHM FOR OPTIMIZATION OF PRODUCTION SCHEDULE

Summary. An article contains developed artificial immune system –AIS in order to solve the job scheduling problem (JSP). In particular the attention was paid to implementation of some additional mutation operations. Its application was aimed to improve efficiency of artificial immune system. The computer experiment was carried out to determine an impact of individual algorithm parameters on the quality of the solution, comparing AIS to other types of such an algorithm. The production scheduling problem was described with developed algorithm.

.

1. Wstęp

Konwencjonalne podejścia używane do rozwiązywania problemów harmonogramowania albo wymagają długiego czasu do osiągnięcia optymalnego rozwiązania albo zawodzą w przypadku gdy problem staje się zbyt złożony (NP-hard). Przestrzeń rozwiązań dla problemu gniazdowego (JSP) dla *N*-zadań na *M*-maszynach posiada górne ograniczenie $(N!)^M$ np. problem harmonogramowania złożony z 10 zadań i 10 maszyn (10X10) posiada $(10!)^{10}$= 3.9594e+065 rozwiązań. Całkowite wyszczególnienie wszystkich rozwiązań w celu odnalezienia rozwiązania optymalnego jest niepraktyczne. Stwarza to konieczność zwiększania efektywności technik harmonogramowania i opracowania nowych technik opartych na algorytmach heurystycznych. Obecnie do optymalizacji procesów produkcyjnych stosuje się szereg

algorytmów. Wśród nich bardzo szerokie zastosowanie znajdują metaheurystyki typu populacyjnego. Utworzone algorytmy takie jak: algorytmy genetyczne (Genetic Algorithms), Sieci neuronowe (Neural network), optymalizacja rojem cząsteczek (Particle Swarm Optimization), optymalizacja kolonią mrówek (Ant Colony optimization), przeszukiwanie tabu (Tabu serach), symulowane wyżarzanie (simulated annealing), czy też algorytmy oparte na systemie immunologicznym (Artificial Immune System (AIS)). Spośród tych algorytmów, algorytmy immunologiczne oparte na systemie immunologicznym stały się powszechnie używane ze względu na możliwość ich adaptacji do szerokiego spektrum zastosowań. Algorytmy AIS pozwalają na rozwiązywanie problemu gniazdowego (JSSP). Algorytmy te powstały na bazie inspiracji teoriami systemu immunologicznego organizmów żywych, obserwacji funkcji, zasad i modeli jego działania. Zdolność adaptacji oraz niezawodność systemów immunologicznych czynią AIS użytecznym także do rozwiązywania problemów harmonogramowania produkcji.

W tym artykule przedstawiono opracowany algorytm AIS dla problemu gniazdowego (JSP). Główną uwagę zwrócono na realizację niektórych operacji mutacji dodatkowej. Jej zastosowanie miało na celu poprawę efektywności algorytmu AIS. Przeprowadzono eksperymenty symulacyjne w celu określenia wpływu poszczególnych parametrów algorytmu na jakość rozwiązania. Przedstawiono porównanie algorytmu AIS z innymi rodzajami algorytmów tego typu. Opisano problem harmonogramowania produkcji rozwiązany za pomocą opracowanego algorytmu.

2. Problem gniazdowy

JSP składa się z $N(j = 1, 2, 3...N)$ zadań do wykonania na $M(i = 1, 2, 3...M)$ maszynach, z zastrzeżeniem następujących ograniczeń: każde z zadań składa się z operacji $Oj = o1j, ..., omj$. Każda z operacji musi być wykonywana na określonej maszynie. Pojedyncze zadanie nie może być wykonywane jednocześnie przez wiele maszyn. Każda maszyna może wykonywać tylko jedną operację w danym czasie. Każda operacja ma czas wykonania p_{ij} zaczynający się w chwili r_{ij}. Dla poszczególnych zadań istnieje określona ścieżka technologiczna. Dla każdej operacji wykonywanej na maszynie Mj istnieje poprzednik PMj i następca SMj. W tym artykule problemem harmonogramowania produkcji typu gniazdowego jest znalezienie optymalnego harmonogramu który spełnia kryterium optymalności jakim jest minimalizacja czasu wykonywania procesu.

3. Opracowany algorytm immunologiczny

Podczas implementacji modułu do harmonogramowania zadań produkcyjnych rozważano użycie algorytmu genetycznego. Wybrano algorytm immunologiczny ze względu na mechanizm sterujący ewolucją struktury populacji. W algorytmach genetycznych zróżnicowanie wynika z konkurencji pomiędzy różnymi, krzyżującymi się osobnikami – jest wymuszane z poziomu otoczenia. W algorytmach immunologicznych mutacje nakładają się na siebie, presja istnieje głównie ze strony populacji, co zwiększa zakres poszukiwania przez większe zróżnicowanie osobników.

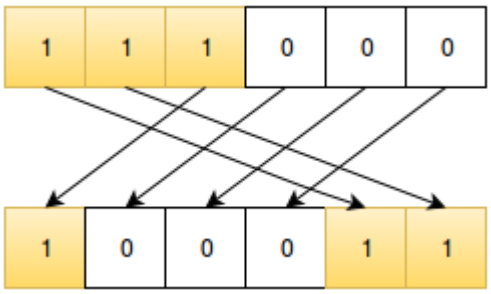
Ogólny schemat algorytmu obejmuje etapy:

1. Pobranie danych konfiguracyjnych
2. Tworzenie populacji początkowej
3. Mutacja główna
4. Mutacja dodatkowa
5. Koniec

Pierwszy etap polega na pobraniu danych sterujących do modułu głównego algorytmu. Parametry te dostosowywane są w procesie testowania na określonych, znanych problemach. Moduł do testowania dostępny jest jedynie dla administratora systemu.
Stosowane parametry algorytmu:
loop – ilość iteracji;
populacja – ilość osobników;
paramr1 – typ mutacji głównej;
proby – ilość prób (nowopowstałych podczas mutacji, konkurujących osobników)
paramw1 – iteracja aktywująca mutację dodatkową 1,
paramw2 – ilość potomnych, rywalizujących ze sobą osobników dla mutacji dodatkowej 1,
paramw3 – typ mutacji dodatkowej 1,
paramwb – rodzaj mutacji dodatkowej 1,
parame1 – iteracja aktywująca mutację dodatkową 2,
parame2 – ilość potomnych, rywalizujących ze sobą osobników dla mutacji dodatkowej 2,
parame3 – typ mutacji dodatkowej 2,
parameb – rodzaj mutacji dodatkowej 2,

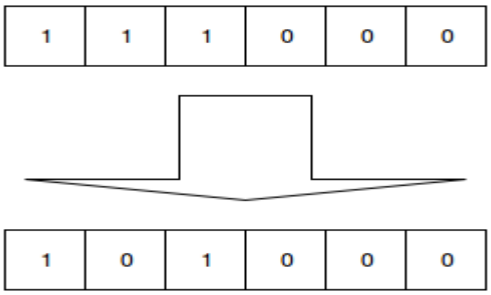
Opis stosowanych rodzajów mutacji:
0 – pusta,
1 – użyteczna,

Opis stosowanych typów mutacji:
0 - zmiana kolejności w genie, przykładowo dla wartości 2 w lewo:



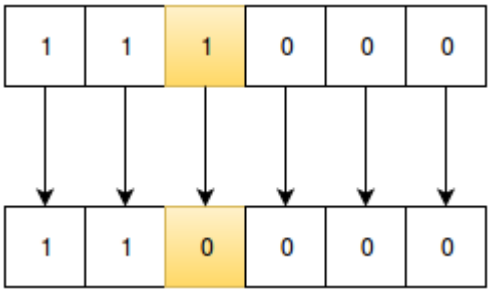
Rys. 1a. Mutacja, zmiana kolejności

1 – w pełni nowy gen, następuje utworzenie nowego osobnika o losowym układzie genu:



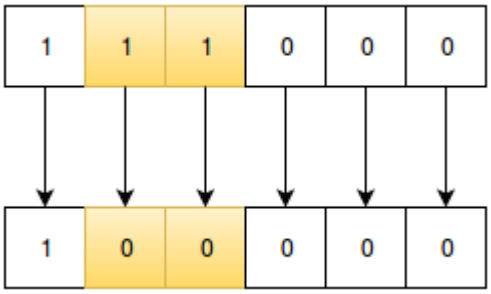
Rys. 1b. Mutacja, nowy gen

2 – mutacja 1-punktowa, następuje zamiana jednego elementu genu:



Rys. 1c. Mutacja, 1-pkt

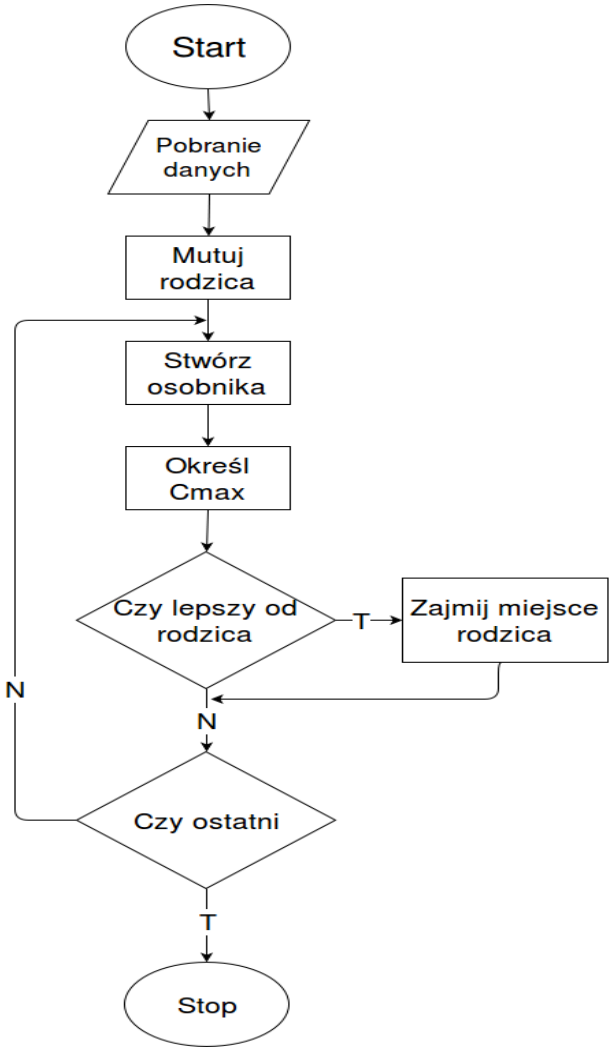
3 – mutacja *n*-punktowa o zmiennym *n,* następuje zmiana n elementów genu, dla *n*=2:



Rys. 1d. Mutacja, n-pkt

Drugim etapem jest tworzenie populacji pierwotnej.
Populacja pierwotna w zaimplementowanym algorytmie jest najmniej wydajnym, uporządkowanym genem. Dla problemu FT6 gen ten ma postać: [1,1,1,1,1,1,2,2,2,2,2,2,3,3,3,3,3,3,4,4,4,4,4,4,5,5,5,5,5,5,6,6,6,6,6,6]

Trzeci etap polega na wywołaniu mutacji głównej zgodnie z założonymi parametrami. Mutacja główna została określona jako domyślnie użyteczna. Użytkownik posiada jedynie możliwość wyboru jej typu oraz ilości nowopowstałych konkurujących osobników.



Rys. 2. Etap mutacji głównej algorytmu immunologicznego

Czwarty etap polega na wykonaniu mutacji dodatkowych zgodnie z wprowadzonymi parametrami. Mutacje dodatkowe zostały dodane po zablokowaniu możliwości wyboru rodzaju mutacji głównej, dla niskiej ilości osobników oraz dużej ilości iteracji pozwalają bardziej zróżnicować populację. W mutacje dodatkowe został również wbudowany system wstępnej analizy wyników, oraz gromadzenia wyników testów.

4. Eksperyment komputerowy

Przeprowadzono eksperyment mający na celu dobór parametrów algorytmu do rozwiązywanego problemu. Wszystkie oznaczenia użyte w tabelach zostały opisane w punkcie 3. Jedynym uwzględnianym kryterium jest wynik eksperymentu. Eksperyment został przeprowadzony na problemach FT20 oraz La02. Po dobraniu parametrów wykonano próbę dla problemów umieszczonych w tabeli 6. Dla każdego wariantu eksperymentu wykonano 5 prób, w tabelach zawarto najlepszy oraz średni wynik.

Eksperyment 1 – badanie wpływu mutacji głównej na wynik.
Parametry stałe: paramw3 (1), paramwb(1), populacja(200), parameb(1), proby(20), paramw2(1), parame3(2), parame2(1), parame1(1000), loop(100), paramw1(1000)
Parametr zmieniany: paramr1

Tabela 1

Eksperyment 1

paramr1:		0	1	2	3
Nazwa	Optymalne	E 1.1	E 1.2	E 1.3	E 1.4
FT20	1165	Min:1262, Avg:1274.4	Min:1452, Avg: 1478.4	Min:1289, avg: 1302.8	Min: 1309, Avg: 1329.8
La02	655	Min: 655, Avg: 664.8	Min: 714, Avg:721.4	Min:666, Avg:668.0	Min: 658, Avg: 675.0

Eksperyment 2 – badanie wpływu wielkości populacji na wynik.
Parametry stałe: paramw3 (1), paramwb(1), parameb(1), proby(20), paramw2(1), parame3(2), parame2(1), parame1(1000), loop(100), paramr1 (2), paramw1(1000)
Parametr zmieniany: populacja

Tabela 2

Eksperyment 2

Populacja:		100	150	200	300	400
Nazwa	Optymalne	E 2.1	E 2.2	E 2.3	E 2.4	E 2.5
FT20	1165	Min: 1301, Avg: 1306.0	Min: 1258, Avg: 1279	Min:1262, Avg:1274.4	Min: 1246, Avg: 1268.0	Min: 1248, Avg: 1268.0
La02	655	Min: 660, Avg: 666.2	Min: 666, Avg: 668.6	Min: 655, Avg: 664.8	Min: 658, Avg: 661.0	Min: 655, Avg: 658.6

Eksperyment 3 – badanie wpływu ilości iteracji na wynik.
Parametry stałe: paramw3 (1), paramwb(1), parameb(1), proby(20), paramw2(1), parame3(2), parame2(1), parame1(1000), paramr1 (2), populacja(200), paramw1(1000)
Parametr zmieniany: loop

Tabela 3

Eksperyment 3

loop		50	75	100	150	200
Nazwa	Optymalne	E 3.1	E 3.2	E 3.3	E 3.4	E 3.5
FT20	1165	Min: 1297, Avg: 1315.8	Min: 1280, Avg: 1290.0	Min:1262, Avg:1274.4	Min: 1228 Avg: 1253.6	Min: 1256 Avg: 1263
La02	655	Min: 656, Avg: 665.4	Min: 658, Avg: 666.2	Min: 655, Avg: 664.8	Min: 655, Avg: 662.0	Min: 655, Avg: 657

Eksperyment 4 – badanie wpływu ilości osobników powstających podczas mutacji na wynik.

Parametry stałe: paramw3 (1), paramwb(1), parameb(1), paramw2(1), parame3(2), parame2(1), parame1(1000), paramr1 (2), populacja(200), loop(100), paramw1(1000)

Parametr zmieniany: proby

Tabela 4

Eksperyment 4

proby		10	15	20	30	40
Nazwa	Optymalne	E 4.1	E 4.2	E 4.3	E 4.4	E 4.5
FT20	1165	Min: 1294 Avg: 1312.8	Min: 1277 Avg: 1299	Min:1262, Avg:1274.4	Min: 1247 Avg: 1265.4	Min: 1236 Avg: 1257.2
La02	655	Min: 655 Avg: 664	Min: 657 Avg: 663.2	Min: 655, Avg: 664.8	Min: 659 Avg: 664.4	Min: 655 Avg: 664.4

Eksperyment 5 – badanie wpływu mutacji dodatkowych na wynik.

Parametry stałe: paramw3 (1), paramwb(1), parameb(1), parame3(2), paramr1 (2), populacja(200), loop(100), proby(20)

Parametry zmieniane: paramw1, paramw2, parame1, parame2

Tabela 5

Parametry dla eksperymentu 5

L. p.	paramw1	paramw2	parame2	parame1	FT20 (min)	FT20 (avg)	La02 (min)	La02 (avg)
5.1	10	10	10	1000	1232	1270.6	655	664.2
5.2	1000	10	10	1000	1276	1283.8	663	665.2
5.3	10	10	10	10	1258	1275.6	666	667.4
5.4	1000	10	10	10	1263	1282.6	658	663.4
5.5	10	1	10	1000	1269	1284.8	660	663.8
5.6	1000	1	10	1000	1264	1281.8	663	666.2
5.7	10	1	10	10	1251	1276	657	664
5.8	1000	1	10	10	1238	1280.2	661	664.2
5.9	10	10	1	1000	1265	1279.8	655	660
5.10	1000	10	1	1000	1253	1273.6	663	666.6
5.11	10	10	1	10	1267	1278.8	655	664.8
5.12	1000	10	1	10	1252	1272.2	656	661.4
5.13	10	1	1	1000	1270	1279.2	656	664
5.14	1000	1	1	1000	1281	1288.6	655	662
5.15	10	1	1	10	1257	1277.6	661	665.4
5.16	1000	1	1	10	1260	1278.8	658	664.8

W tabeli 6 przedstawiono porównanie wartości Cmax dla wybranych typów problemów testowych: FT06, FT10, LA01, LA02, LA03, LA04, LA05, LA06, ABZ5 oraz ABZ6. A1) Atay Y., Kodaz H.; A2) Bondal A.A. (2008) –AIS (MSc); A3) Bondal A.A. (2008) – GA (MSc); A4) Murugesan R. (2012) CSA – Clonal Selection Algorithm; A5) Murugesan R. (2012) PSMCSA – Positive Selection based Modified Clonal Selection Algorithm; A6) badany algorytm.

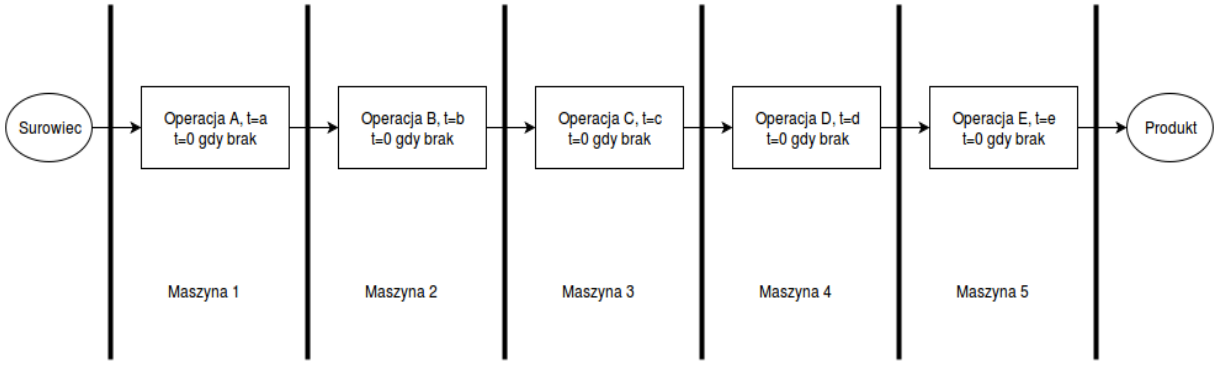
Tabela 6

Porównanie wartości makespan Cmax dla wybranych problemów testowych

	Cmax value									
	FT06	FT10	LA01	LA02	LA03	LA04	LA05	LA16	ABZ5	ABZ6
Opt	55	930	666	655	597	590	593	945	1234	943
A1	55	1034	666	655	597	590	593	1024	1270	943
A2	55	1208	702	708	672	644	593	1124	1434	1084
A3	55	1099	666	716	638	619	593	1033	1339	1043
A4	55	1208	702	708	672	644	593	1124	1434	1084
A5	55	940	666	655	597	590	593	960	1249	967
A6	55	956	666	655	603	590	593	959	1242	948

5. Zastosowanie algorytmu do optymalizacji wybranego harmonogramu produkcji

Przedsiębiorstwo produkcyjne zajmujące się przetwarzaniem bali drewnianych na belki o zadanych parametrach posiada następujący proces produkcyjny, zidentyfikowany jako problem gniazdowy.



Rys. 3. Składowe analizowanego procesu produkcyjnego

Poszczególne składowe procesu są następujące: operacja A – manipulacja surowca, może zachodzić jedynie na jednej maszynie, operacja polega na wybraniu, dostarczeniu oraz przygotowaniu do obróbki surowca; operacja B – przetarcie, może zachodzić jedynie na jednej maszynie, polega na wycięciu z bali drewna belek

o zadanych wymiarach. Operacja poprzedzona jest ustawieniem odpowiedniego sprzęgu pił. Po wykonaniu operacji B piły wracają do pozycji startowej; operacja C – manualne struganie, może zachodzić jedynie na jednym stanowisku, polega na manualnej obróbce wstępnie przygotowanego surowca zgodnie z założeniami klienta; operacja D – suszenie, może zachodzić jedynie na jednym stanowisku, czas trwania uzależniona jest od wymaganej końcowej wilgotności oraz wymiarów belki; operacja E – impregnacja produktu, może zachodzić jedynie na jednym stanowisku.

Każda z tych operacji może, lecz nie musi zachodzić w procesie produkcji. W przypadku gdy dana operacja nie jest konieczna, jej czas trwania jest zerowany. Harmonogramowanie zadań wywoływane jest manualnie, po akceptacji parametrów produkcyjnych dla każdego z oczekujących zamówień produkcyjnych.

6. Wnioski

Przeprowadzone liczne eksperymenty symulacyjne pozwoliły na określenie najbardziej efektywnych parametrów algorytmu immunologicznego. Po zakończeniu konfiguracji algorytmu, zaimplementowano go, w istniejącym w przedsiębiorstwie, systemie informatycznym. Pozwoliło to na zmniejszenie całkowitego czasu produkcji o kilkanaście procent. Jednak uzyskany czas realizacji rozwiązania jest znaczny, co jest konsekwencją wybranego środowiska programistycznego.

LITERATURA

1. Akhshabi M., Akhshabi M., Khalatbari J.: Solving flexible job-shop scheduling problem using clonal selection algorithm, Indian Journal of Science and Technology, Vol. 4 No. 10(Oct2011), p. 1248–1251
2. Atay Y., Kodaz H.: Optimization of job shop scheduling problems using modifie clonal selection algorithm, Turkish Journal of Electrical Engineering & Computer Sciences, N 22, 2014, p. 1528–1539.
3. Bondal A.A.: Artificial immune systems applied to job shop scheduling, MSc, Ohio University, Athens, OH, USA, 2008.
4. Lau H., Qiu X.: A n Artificial Immune Systems (AIS)-based Unified Framework for General Job Shop Scheduling, Preprints of the 19th World Congress The International Federation of Automatic Control Cape Town, South Africa. August 24-29, 2014, p. 6186–6191.
5. Mahapatra K.: Job Shop Scheduling Using Artificial Immune System, A Thesis National Institute Technology Rourkela, 2012.
6. Murugeasan R., Sivasakthi Balan K.: Positive Selection Based Modified Clonal Selection Algorithm for Solving Job Shop Scheduling Problem, Applied mathematical Sciences, vol. 6, 2012, N 46, p. 2255–2271.
7. Ong Z., Tay J., Kwoh C.: Applying the Clonal Selection Principle to Find Flexible Job-Shop Schedules, Volume 3627 of the series LNCS p. 442-455.
8. Ren Q.,, Wang Y., A New Immune Genetic Algorithm for Job Shop scheduling Problem Journal of Computational Information Systems 9:3 (2013), p.1011–1018.

9. Skołud B., Wosik I.: Algorytmy immunologiczne w szeregowaniu zadań produkcyjnych, Zarządzanie Przedsiębiorstwem, 2008, N1, str.47–56.
10. Wierzchoń, S.: Sztuczne systemy immunologiczne, Wydawnictwo EXIT, Warszawa 2001.
11. Witkowski, Antczak A., Antczak P.: Random and Evolution Algorithms of Tasks Scheduling and the Production Scheduling. Proceedings of the IEEE International Conference on Fuzzy Systems – FUZZ-IEEE 2004, Budapeszt 2004, vol. 2, p. 727–732.