Arkadiusz Antczak
Wydział Elektrotechniki i Elektroniki Politechniki Łódzkiej
dr hab. inż. Tadeusz Witkowski
Paweł Antczak
Wydział Inżynierii Produkcji Politechniki Warszawskiej

ANALIZA PROBLEMU HARMONOGRAMOWANIA WIELOASORTYMENTOWEJ PRODUKCJI MAŁOSERYJNEJ ZA POMOCĄ PROCEDURY HYBRYDOWEJ (HEURYSTYKI SB I GRASP)

W pracy przedstawiono procedurę optymalizacji do opracowania harmonogramów dla produkcji wieloasortymentowej produkcji małoseryjnej. Dla rozwiązania tej klasy zadania opracowano hybrydowy algorytm wykorzystujący heurystyki typu przesuwanego wąskiego gardła i GRASP (Greedy Randomized Adaptive Search Procedure).

ANALYSIS OF THE JOB SHOP SCHEDULING PROBLEM WITH APPLICATION HYBRID PROCEDURE (SHIFTING BOTTLENECK AND GRASP HEURISTICS)

The paper presents the application of optimization procedure for generation schedules to job shop scheduling problems (multi-assortment short-series production). For solved this class problem has been presented hybrid algorithm used Shifting Bottleneck and Greedy Randomized Adaptive Search Procedure.

1. WPROWADZENIE

Do optymalizacji harmonogramowania produkcji o dużych rozmiarach (duża liczba maszyn i wykonywanych operacji) najbardziej efektywnymi są obecnie algorytmy metaheurystyczne. Większość z nich wykorzystują lokalne przeszukiwanie i metody sztucznej inteligencji. Coraz częściej do optymalizacji tego typu problemów wykorzystuje się procedury hybrydowe obejmujące kombinację na ogół dwóch metod. W danej pracy przeanalizowano efektywność procedury skonstruowanej przy wykorzystaniu dwóch heurystyk typu przesuwanego wąskiego gardła SB oraz GRASP. Ta ostatnia w tej procedurze służy głównie do otrzymania dopuszczalnego rozwiązania początkowego. Efektywność harmonogramowania zależy od prawidłowego wykorzystania zasobów krytycznych. W zadaniach harmonogramowania zasobem krytycznym jest tak nazywana maszyna krytyczna t.j. maszyna która stanowi „wąskie gardło" w procesie tworzenia harmonogramu i najbardziej wpływa na jego jakość ocenianą na podstawie danego kryterium (w danym przypadku ocenianego na podstawie minimalnego czasu wykonania wszystkich zadań). Heurystykę GRASP przedstawiono w innej pracy tych materiałów.

2. SFORMUŁOWANIE ZADANIA

Rozpatrywane zadanie jest typowym przykładem zadania planowania małoseryjnej produkcji wieloasortymentowej. Ogólnie istota danego zadania zawiera się w następującym. Istnieje pewien zbiór typów części, które należy wytworzyć w ilości określonej przez produkcyjne zamówienia. Dla wytworzenia każdego typu części należy wykonać w określonym technologicznym porządku szereg operacji za pomocą ograniczonego zbioru maszyn. Dla każdej operacji określa się czas niezbędny na jej wykonanie. Każda operacja technologiczna może być wykonana na jednej maszynie spośród grupy technologicznie zamiennych maszyn. Przed rozpoczęciem wykonania operacji maszynę należy wstępnie przezbroić, jednak jeżeli wykonywane są operacje tego samego typu, to przezbrajanie maszyny nie jest wymagane (czas przezbrajania dla poszczególnych operacji jest różny). Należy wybrać dla każdej operacji maszynę i czas jej rozpoczęcia w taki sposób, aby zamówienia były wykonane w wymaganej wielkości, z określonymi ograniczeniami i harmonogram spełniał wybrane kryterium optymalności [1,2]. W danej pracy jako kryterium optymalności przyjęto kryterium Johnsona (maksymalny czas zakończenia operacji).

W danej pracy był rozpatrzony model dla szeregowego przebiegu procesu produkcyjnego. Przebieg tego typu charakteryzuje się następującymi właściwościami: jeżeli maszyna rozpoczęła wykonywać określoną operację produkcyjną z daną częścią określonego typu, to ona będzie wykonywać szeregowo podobne operacje dla wszystkich części tego typu aż do zakończenia ich obróbki. Tak więc, dla szeregowego przebiegu produkcji przezbrajanie maszyn jest przeprowadzane tylko jeden raz dla każdej partii wykonania tego samego typu operacji. Wszystkie operacje tego samego typu można rozpatrywać jak jedną operację, czas wykonania której równa się jest sumie czasów na ich wykonanie i czasowi niezbędnemu na wstępne przezbrojenie maszyny. Jak widać, przedstawione zadanie harmonogramowania wieloasortymentowej produkcji małoseryjnej dla szeregowego przebiegu produkcji różni się od klasycznego harmonogramowania (*job shop scheduling problem* – JSSP) tym, że dla każdej operacji należy wybrać maszynę , która będzie ja wykonywać spośród maszyn z grupy technologicznie zamiennych.

Zadanie może być sformułowane w następujący sposób. Określono zbiór maszyn M (moc zbioru M oznaczmy przez m), zbiór operacji O , elementami którego są poszczególne technologiczne operacje σ^i, $i = 1...n$, gdzie n – moc zbioru O .

Każdej operacji $\sigma^i \in O$ przyporządkowano podzbiór maszyn $M^i \in$ M, które mogą je wykonywać. Zbiór O jest zbiorem częściowo uporządkowanym, tj. określono kolejność wykonania C = $\{\sigma^i \prec \sigma^j\}$, który określa kolejność wykonania operacji $\sigma^i \prec \sigma^j$ (« $\sigma^i \prec \sigma^j$ » oznacza, że operacja σ^i powinna być wykonana przed rozpoczęciem wykonania operacji σ^j). Do rozpoczęcia wykonania operacji na maszynie należy przeprowadzić na niej operację przezbrajania. Oprócz tego, wprowadza się klasy operacji tego samego typu k^j, $j = 1..K$, gdzie K ilość klas tego samego typu. Sens tych klas jednego typu zawiera się w następującym: jeżeli σ^i i σ^j przynależy do tej samej klasy jednego rodzaju operacji i wykonywane są na jednej maszynie, przy czym po wykonaniu operacji σ^i ta maszyna nie wykonuje operacji do rozpoczęcia

wykonania operacji σ^j, to wstępne przezbrajanie maszyny dla wykonania operacji σj nie jest wymagane.
Oznaczmy przez $p(\sigma^i)$ czas, niezbędny do wykonania operacji σ^j, $t(\sigma^j)$ – czas niezbędny dla przezbrojenia maszyny przed wykonaniem operacji σ^i, $S(\sigma^i)$, $F(\sigma^i)$ – czas rozpoczęcia (zakończenia) wykonania operacji σ^i, m^i – maszyna, wybrana z M^i dla wykonania operacji σ^i.

Zadania polega na tym, aby wybrać dla każdej operacji $\sigma^i \in O$ maszynę ze zbioru M^i ($i = 1..n$) i po tym określić kolejność wykonania operacji na maszynach z M, w ten sposób, aby określony harmonogram minimalizował sumaryczny czas wykonania prac (kryterium Johnsona). Przy czym, jeżeli wszystkie wielkości $t(\sigma^i)$ są równe zero dla $\sigma^i \in O$, to można określić ograniczenia porządku i podzbiory M^i w taki sposób, aby otrzymać klasyczne sformułowanie zadania planowania dla problemu JSSP.

3. ALGORYTM ROZWIĄZANIA ZADANIA

Ta procedura oparta jest na następującej idei. W zadaniach harmonogramowania zasobem krytycznym jest tak nazywana maszyna krytyczna t.j. maszyna która stanowi „wąskie gardło" w procesie tworzenia harmonogramu i najbardziej wpływa na jego jakość ocenianą na podstawie danego kryterium np. minimalnego czasu zakończenia wykonywania prac. Wykorzystując heurystykę SB kolejno określa się priorytet wykonania operacji na maszynach. Priorytet wyboru maszyn przy konstruowaniu harmonogramu określany jest na podstawie pojęcia maszyny krytycznej.
W danej pracy przedstawiono hybrydową procedurę dla rozwiązania zadania, która wykorzystuje heurystyki SB oraz GRASP [5].
Ogólnie procedurę przesuwanego wąskiego gardła [3] można opisać w następujący sposób. Niech będą skonstruowane już harmonogramy dla wszystkich maszyn z podzbioru. Wtedy dla wszystkich maszyn nie przynależących do zbioru Seq rozwiązywane jest zadanie skonstruowania harmonogramu dla jednej maszyny, przy wykorzystaniu algorytmu ze [4]. W tym przypadku uwzględniane są już skonstruowane harmonogramy dla maszyn przynależących do zbioru Seq. Maszyna dla której rozwiązanie zadania skonstruowania harmonogramu dla jednej maszyny jest rozwiązaniem najgorszym, określa się jako krytyczną i dołącza do zbioru Seq. Po wykonaniu tego etapu, uwzględniając ostatnio dołączony harmonogram, poprawiane (korygowane) są harmonogramy dla wszystkich maszyn przynależących do zbioru Seq, powtórnie rozwiązując dla każdej z maszyn zadanie konstruowania harmonogramu dla jednej maszyny z uwzględnieniem harmonogramów dla pozostałych maszyn ze zbioru Seq. Procedura ta jest powtarzana, dopóki wszystkie maszyny nie będą dołączone do zbioru Seq. Należy podkreślić, że procedura SB nie gwarantuje ostatecznie, otrzymanie dopuszczalnego rozwiązania, jednak, to można poprawić, dodając procedurę korygowania harmonogramu analizując maszynę, która spowodowała powstanie rozwiązania niedopuszczalnego.
Przy realizacji heurystyki SB należy w celu wykonania operacji realizować sposób wyboru maszyn spośród zbioru technologicznie zamiennych. W opracowanym

algorytmie wybór maszyn jest realizowany za pomocą procedury GRASP [5], która na każdej iteracji generuje określone rozwiązanie. Procedura GRASP kolejno dołącza do harmonogramu operacje w taki sposób, aby ograniczenia na kolejność wykonania operacji były zachowane, t.j. operacje dołączane są do harmonogramu, jeżeli wszystkie poprzedzające je operacje były już uwzględnione (dodane).

Dla każdej operacji, która może być dołączona, obliczane są wartości oceny częściowego harmonogramu po jej dodaniu do harmonogramu na wszystkie możliwe maszyny do jej wykonania. Wtedy, jeżeli wartość oceny lepszego wariantu harmonogramu po dodaniu operacji - A, a wartość oceny gorszego wariantu harmonogramu – B, to wariant dodania operacji do częściowego harmonogramu wybierany jest losowo spośród wariantów, wartość oceny których nie przewyższa wartości $A+\alpha^*(B-A)$, gdzie α – rzeczywisty parametr algorytmu, określony w przedziale [0,1].

Po generacji rozwiązania, wybór maszyn dla każdej operacji jest zapamiętywany i dalej rozpatrywane jest zadanie, w którym każdej operacji przypisana jest odpowiednia maszyna na której wykonywana jest ta operacja. Do tego zadania stosowana jest procedura SB i następnie jest generowane nowe rozwiązanie. Najlepsze znalezione rozwiązanie na wszystkich operacjach jest zwracane jako wynik działania algorytmu. Do wyboru danej maszyny z grupy technologicznie zamiennych maszyn można zastosować także np. absolutnie losowe generowanie rozwiązań, jednak jak pokazały eksperymenty takie podejście jest o wiele mniej efektywne w porównaniu z zastosowaniem do generowania rozwiązań procedury GRASP.

4. EKSPERYMENTY KOMPUTEROWE

Oprogramowanie dla przedstawionej wyżej procedury zrealizowano w języku C++. Eksperymenty komputerowe przeprowadzono dla danych przedstawionych w [1, 2]. Liczba operacji, dla których konstruowano harmonogram wynosiło 160.

Wartość funkcji kryterialnej równa 50242,2 min. jest optymalnym rozwiązaniem dla danych, dla których przeprowadzono eksperymenty komputerowe. Podczas eksperymentów dla wszystkich wartości parametrów α za pomocą przedstawionego algorytmu znaleziono optymalne rozwiązanie. W przypadku wartości parametru $\alpha = 1$, generowane są absolutnie losowe rozwiązania.

Poniższe tablice 1-7 przedstawiają rezultaty otrzymane za pomocą przedstawionej hybrydowej procedury (metoda wąskiego gardła i GRASP) dla rzeczywistego procesu produkcyjnego [1,2]. Parametr *Alpha* został wykorzystany w tym algorytmie do generowania początkowych rozwiązań dla każdej iteracji za pomocą procedury GRASP. Dla każdej wartości parametru *Alpha* [0...1], zostało przeprowadzonych 100 iteracji działania algorytmu, gdzie każda iteracja programu zwracała wartości *makespanu* $F(H)$ (minimalny czas obróbki wszystkich zadań produkcyjnych).

Tablica 1. Wartości F (H) dla alfa=0

	Makespan [min]		**Alpha = 0**		*dla 100 iteracji*
	1	2	3	4	5
1	57262,5	57609,0	52303,6	55168,9	56090,5
2	**50242,2**	55993,5	53112,0	54319,1	53821,6
3	54858,0	51519,5	54763,3	**50242,2**	53826,7
4	54069,2	52937,2	53038,4	53244,8	51343,3
5	51136,6	54022,1	**50242,2**	51207,2	54555,1
6	54099,5	50809,3	52552,1	53899,5	**50242,2**
7	57696,0	51729,0	52085,0	52389,6	55242,9
8	57707,7	53653,1	51610,6	51087,5	52931,1
9	56181,6	54129,6	54578,8	50244,4	52932,6
10	52992,2	50648,9	51224,3	52329,7	56572,9
11	52415,7	54913,5	51343,3	52219,7	53494,7
12	52022,8	53706,0	55568,0	52775,1	55327,6
13	53474,0	51703,3	55808,0	52219,7	**50242,2**
14	53525,3	54758,8	55021,1	54561,9	52742,2
15	56200,2	54916,3	53104,6	52009,0	**50242,2**
16	56597,7	53101,1	52219,7	**58542,1**	54949,4
17	50909,2	55064,4	53495,8	52576,8	57558,6
18	55276,5	51803,0	54950,1	**50242,2**	57538,9
19	50660,2	56509,8	56114,9	52764,9	55281,1
20	50380,4	51189,0	52341,0	54148,6	51016,8

Wartość *MIN*. = 50242,2, wartość *ŚREDNIA* = 53460,5, wartość *MAX*. = 58542,1
Ilość rozwiązań optymalnych = 7

Tablica 2. Wartości F(H) dla alfa =0,1

	Makespan [min]		**Alpha = 0,1**		*dla 100 iteracji*
	1	2	3	4	5
1	**50242,2**	52756,1	53682,2	50648,9	**50242,2**
2	54433,6	56176,8	53072,8	52741,2	54455,3
3	51502,0	55115,2	55240,5	**50242,2**	54321,5
4	54011,7	52111,5	53495,8	55030,0	52504,8
5	52665,4	51842,6	**50242,2**	51829,1	52994,5
6	55069,1	53089,0	53691,3	53513,5	57505,5
7	52441,1	50734,9	51095,3	51619,0	51343,3
8	53463,2	52646,7	55225,0	50635,0	53809,9
9	53496,4	51610,6	**50242,2**	51468,9	54409,6
10	51172,7	51503,5	**50242,2**	51482,3	54664,8
11	52019,3	**50242,2**	52009,0	53777,2	53238,9
12	50763,7	53256,8	52069,5	53781,2	57659,0
13	55033,3	58024,2	**50242,2**	54569,2	54727,6
14	69763,2	56176,8	52188,6	51610,6	51370,8
15	52834,0	55486,3	59491,1	**50242,2**	56724,9
16	51286,5	54268,3	56769,0	54372,5	59867,1
17	58055,3	56218,5	51519,5	60553,7	54827,6
18	53778,4	57744,8	53867,3	50648,9	52109,7
19	51861,3	52788,6	56331,8	50249,1	51351,6
20	**181706,0**	58002,1	59986,4	53884,2	53858,6

Wartość *MIN*. = 50242,2, wartość *ŚREDNIA* = 54909,8, wartość *MAX*. = 181706,0
Ilość rozwiązań optymalnych = 9

Tablica 3. Wartości F(H) dla alfa=0,2

	Makespan [min]		**Alpha = 0,2**		*dla 100 iteracji*
	1	2	3	4	5
1	50763,7	56755,8	50660,2	57689,1	52841,5
2	52559,1	55374,2	51866,3	53079,7	**50242,2**
3	52263,3	56275,8	52497,1	53358,2	53942,5
4	53820,4	54197,0	55911,6	55030,0	54147,7
5	51458,5	52777,6	52776,6	53592,1	52163,0
6	52580,3	51172,7	52654,2	57358,0	50972,3
7	53373,8	**50242,2**	52367,8	52690,5	51457,3
8	53085,3	56750,9	52572,4	53745,0	53182,6
9	52329,2	51926,7	55587,4	55210,1	55140,8
10	57268,2	59403,7	54205,9	53269,7	51610,6
11	56014,1	57206,5	53357,4	51324,4	55359,2
12	51548,2	53125,5	53386,7	**50242,2**	53536,2
13	52329,2	51306,2	53835,9	50935,8	56176,8
14	53158,3	53873,8	53228,0	52395,9	52931,1
15	51525,8	51610,6	53340,3	54798,3	53106,1
16	**50242,2**	**60108,3**	50648,9	50596,1	53340,3
17	51610,6	52541,1	50342,3	52608,6	53130,6
18	53363,4	52164,5	52663,0	52857,9	53839,1
19	53402,3	**50242,2**	53379,2	50957,7	54450,6
20	50763,7	52903,5	54617,9	53362,8	51935,7

Wartość *MIN.* = 50242,2, wartość *ŚREDNIA* = 53219,3, wartość *MAX.* = 60108,3
Ilość rozwiązań optymalnych = 5

Tablica 4. Wartości F(H) dla alfa=0,5

	Makespan [min]		**Alpha = 0,5**		*dla 100 iteracji*
	1	2	3	4	5
1	54432,8	56342,4	**66653,8**	50879,7	57628,2
2	54888,6	**50242,2**	57561,9	51786,6	58077,3
3	52117,1	52276,8	**50242,2**	52198,2	55652,0
4	50418,0	55790,4	57598,7	55169,9	62125,4
5	50509,2	54129,6	54960,9	51262,5	**50242,2**
6	51326,2	54726,6	55157,0	59134,9	52510,2
7	**50242,2**	53464,0	52174,1	53212,5	54466,4
8	54109,2	55653,4	51551,6	51997,3	57223,1
9	57727,1	53327,6	56456,8	57706,0	**50242,2**
10	51351,6	52434,3	50377,3	53571,4	**50242,2**
11	54352,4	52663,0	54858,0	52922,7	52104,1
12	52543,8	53426,9	54293,4	52769,4	**50242,2**
13	51409,2	53418,3	51933,7	55132,2	54104,4
14	52009,0	54187,1	52663,0	52933,3	54709,7
15	58639,1	52592,4	56440,0	54261,6	54019,9
16	51610,6	52576,2	51554,6	55930,9	51572,0
17	**50242,2**	52320,7	53851,3	55349,5	53915,9
18	52303,4	51320,3	54796,6	58105,9	58396,5
19	54970,3	55693,6	53925,1	54576,3	**50242,2**
20	52073,6	52894,7	51140,4	53728,4	54901,6

Wartość *MIN.* = 50242,2, wartość *ŚREDNIA* = 53801,9, wartość *MAX.* = 66653,8
Ilość rozwiązań optymalnych = 9

Tablica 5. Wartości F(H) dla alfa=0,6

	Makespan [min]		**Alpha = 0,6**		*dla 100 iteracji*
	1	2	3	4	5
1	53781,2	52339,7	51591,8	52251,2	138746,0
2	55101,3	57653,9	52570,6	50342,5	51866,3
3	54621,5	50739,2	50734,9	52136,5	54012,7
4	53335,3	50935,8	55675,0	53639,2	50648,9
5	50277,6	53358,2	51729,7	54855,1	53779,0
6	56793,5	63577,1	51273,4	54995,5	59860,3
7	**218446,0**	58523,7	54147,2	50313,1	58983,2
8	53017,9	52920,1	51273,4	55274,5	50874,0
9	53320,2	53019,3	51513,2	**50242,2**	53136,3
10	58375,3	53434,2	56362,4	52355,0	52899,8
11	52044,3	50648,9	55574,5	50277,6	57133,5
12	50380,6	51008,5	52666,2	54719,0	52530,5
13	53587,6	53668,1	50674,3	50323,3	51172,7
14	50449,0	61375,5	52848,2	53267,9	50935,8
15	53719,2	50734,9	55756,9	52411,7	51978,9
16	50952,7	55081,3	52575,5	57665,8	53492,0
17	152418,0	50734,9	**50242,2**	51798,8	50951,7
18	51511,6	54568,0	55997,9	54963,3	55259,1
19	**50242,2**	51610,6	50899,2	55254,1	58681,9
20	54880,0	52663,0	55237,4	56152,8	55672,0

Wartość *MIN.* = 50242,2, wartość *ŚREDNIA* = 56910,5, wartość *MAX.* = 218446,0
Ilość rozwiązań optymalnych = 3

Tablica 6. Wartości F(Ḣ) dla alfa =0,9

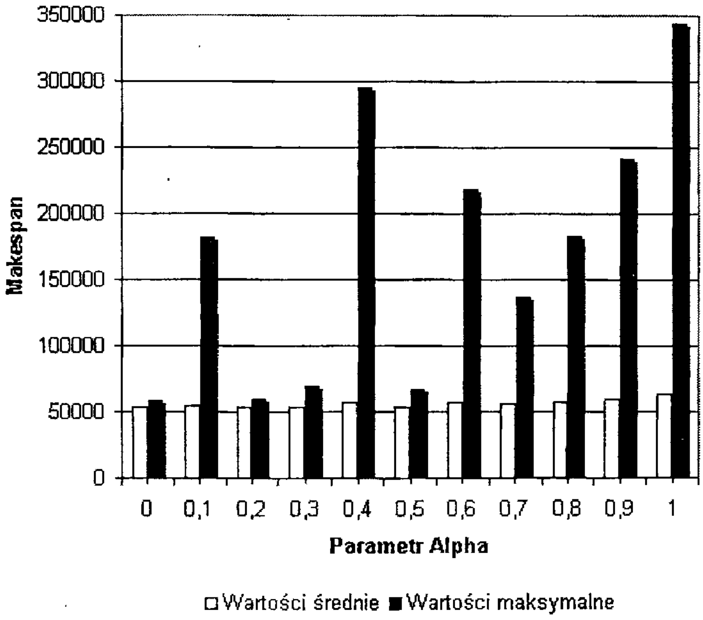
	Makespan [min]		**Alpha = 0,9**		*dla 100 iteracji*
	1	2	3	4	5
1	54256,0	58461,0	55035,7	50277,6	50899,2
2	53359,8	53401,5	56311,6	52692,8	56947,7
3	**50242,2**	53931,7	51492,9	52646,7	50707,1
4	54910,6	52427,5	58089,9	58444,7	53358,2
5	56726,7	51360,4	52465,2	56833,4	54585,8
6	62089,2	55826,3	59909,4	53159,0	55535,3
7	160003,0	51303,7	50845,3	51978,2	51146,2
8	**50242,2**	**50242,2**	51027,8	**241427,0**	50277,6
9	53602,6	51732,4	56275,2	55770,9	51330,1
10	53358,2	51550,9	58108,6	57684,3	56817,2
11	59850,5	54697,9	59669,4	60227,7	56332,4
12	54591,5	54816,0	56934,5	51320,7	54014,1
13	52544,0	60804,6	51212,2	52823,6	52009,0
14	55359,5	60143,4	107645,0	54893,2	63270,5
15	51807,4	50674,3	53667,4	53545,5	50684,5
16	52919,6	53781,8	55038,5	54273,2	58479,5
17	57054,7	52427,5	159501,0	53756,3	57124,9
18	55689,1	52645,9	56120,4	51003,0	162851,0
19	54407,2	53727,4	50674,3	51926,4	51312,5
20	52325,7	50734,9	54699,4	53015,4	59595,9

Wartość *MIN.* = 50242,2, wartość *ŚREDNIA* = 59897,3, wartość *MAX.* = 241427,0
Ilość rozwiązań optymalnych = 3

Tablica 7. Wartości F(H) dla alfa=1,0

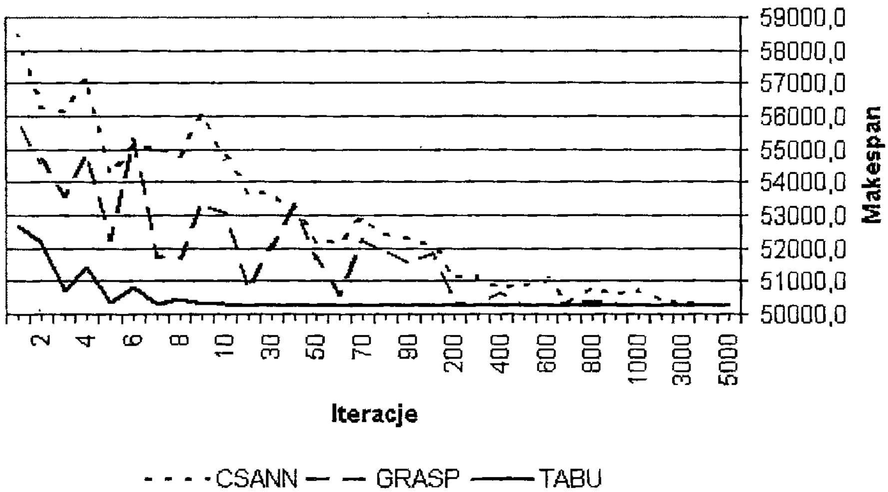
	Makespan [min]		Alpha = 1		*dla 100 iteracji*
	1	2	3	4	5
1	56738,1	**50242,2**	50739,0	53407,5	50558,7
2	53348,7	53794,7	62679,7	55388,0	55762,7
3	55887,9	51318,0	50935,8	55893,0	55538,0
4	57654,2	51697,2	52714,3	53829,6	54922,6
5	52476,8	50281,7	53946,9	61460,8	53587,1
6	50766,7	51578,7	53861,2	59611,1	55746,3
7	53897,2	55279,7	57116,7	54760,6	52972,3
8	58060,5	53404,4	55699,3	56374,1	60425,7
9	54512,1	52795,0	58885,9	**50242,2**	58191,7
10	59449,2	55809,0	56098,0	54021,1	52787,0
11	64218,9	63069,2	65077,9	52870,3	60661,0
12	54143,6	54252,4	57140,3	50865,1	52339,7
13	**344245,0**	53477,9	55366,7	**50242,2**	59740,1
14	56605,4	59034,9	150007,0	211189,0	60227,9
15	56398,0	50612,7	52994,1	53950,5	57370,2
16	269321,0	55129,3	56266,1	51367,9	55847,8
17	57472,2	66655,0	52398,5	55597,0	57578,8
18	53662,8	55894,7	54913,7	71972,7	52441,1
19	58533,3	59694,6	59514,6	60266,1	52646,7
20	55481,0	56413,3	54382,6	60401,9	58121,0

Wartość *MIN.* = 50242,2, wartość *ŚREDNIA* = 63152,2, wartość *MAX.* = 344245,0
Ilość rozwiązań optymalnych = 3



Rys. 1. Średnie i maksymalne wartości F(H) dla alfa=[0,1]

Na podstawie danych eksperymentu, został zbudowany wykres przedstawiający otrzymane rezultaty w zależności od parametru *Alpha* (*Makespan/Parametr Alpha*) oraz przedstawiona została zależność ilości znalezionych optymalnych rozwiązań od parametru *Alpha* (rys.1). Wartości F(H) w zależności od liczby iteracji dla wybranych metod przedstawiono na rys. 2.



Rys.2. Wartości F(H) w zależności od liczby iteracji dla wybranych metod (sieć neuronowa CSANN, metoda Tabu search i metaheurystyka GRASP)

Podczas eksperymentu zbadano również wartości F(H) w zależności od parametru *Alpha* dla większej ilości iteracji (100...25000), a następnie określono zależność otrzymanych rezultatów na poszczególnych iteracjach od parametru *Alpha*.
Podczas eksperymentu dla każdego parametru *Alpha* algorytm znalazł optymalne rozwiązanie. Choć dla większej ilości iteracji (100...25000) dla Alpha = 0,9 oraz *Alpha* = 1 nie znaleziono optymalnych rozwiązań, podkreślić należy to, że dana wartość F(H) obliczona została dla konkretnej wartości iteracji (100-ej, 200-ej, ..., 25000-cznej) a nie dla przedziału od 1 do 25000 iteracji. Warto też zauważyć, że dla *Alpha* = 1 generowane są absolutnie losowe rozwiązania.
Jak widać z przeprowadzonego eksperymentu dla metody przesuwanego wąskiego gardła (SB), najlepsza wartość F(H) wynosi 50242,2 i jest ona wartością optymalną przy szeregowym przepływie produkcji. Wartość tę uzyskano już przy stosunkowo małej liczbie iteracji (w zakresie od 1 do 100 otrzymano optymalne rozwiązania). Za pomocą algorytmu SB przy szeregowym przepływie produkcji uzyskano lepsze wyniki niż dla algorytmu genetycznego AGHAR [6].

5. PODSUMOWANIE

Otrzymane wyniki z przeprowadzonej analizy eksperymentalnej pokazują, że techniki metaheurystyczne poprawiają rozwiązania problemów harmonogramowania. Za pomocą algorytmu SB (przesuwanego wąskiego gardła) otrzymano dobre rezultaty, co potwierdza fakt, że metoda ta jest skuteczną metodą stosowaną w projektowaniu harmonogramów.

W dalszym ciągu jednak trudno określić, która z metod jest generalnie najlepsza, ponieważ w zależności od specyfiki rozwiązywanego problemu krytycznym wskaźnikiem efektywności algorytmu może być czas uzyskiwania wyniku, jakość wyniku, czy też stopień złożoności realizacji algorytmu. Jednak liczne eksperymenty badawcze potwierdzają efektywność metod metaheurystycznych w rozwiązywaniu bardzo trudnych z algorytmicznego punktu widzenia problemów. Dlatego też panuje przekonanie, że praktyczne zastosowanie tych technik jest obecnie jedną z najbardziej skutecznych metod optymalizacji harmonogramowania w złożonych rzeczywistych procesach produkcyjnych.

Dla przedstawionego zadania harmonogramowania wieloasortymentowej produkcji małoseryjnej opracowano algorytm wykorzystujący procedurę przesuwanego wąskiego gardła. Eksperymenty komputerowe potwierdziły wysoką efektywność zastosowania przy generacji rozwiązań procedury GRASP w porównaniu z przypadkiem gdy zamiast niej wykorzystano generowanie losowe.

BIBLIOGRAFIA

[1] *Witkowski T.*, Szeregowanie zadań i harmonogramowanie produkcji metodą optymalizacji Szeza, Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa 1998

[2] *Witkowski T.*, Decyzje w zarządzaniu przedsiębiorstwem, WNT, Warszawa 2000

[3] *Adams J., Balas E., Zawack D.* The Shifting Bottleneck Procedure for Job Shop Scheduling, Manag. Sci.– 1988. – vol.34, No.3, Focussed Issue on heuristics., pp. 391-401.

[4] *Carlier J.* The one machine sequencing problem, European J. Oper. Res., 11 1982, pp. 42-47.

[5] *Binato S., Hery W.J., Loewenstern D., Resende M.G.C.* A GRASP for Job Shop Scheduling, Essays and surveys on metaheuristics, Kluwer Academic Publishers, New York 2001, pp..59–79.

[6] *Witkowski T., Antczak P., Antczak P.* , Narzędzia sztucznej inteligencji do harmonogramowania produkcji, w: Przedsiębiorstwo i region w Zjednoczonej Europie, Prace Naukowe KN O i Z PAN, AŚ, Kielce 2004