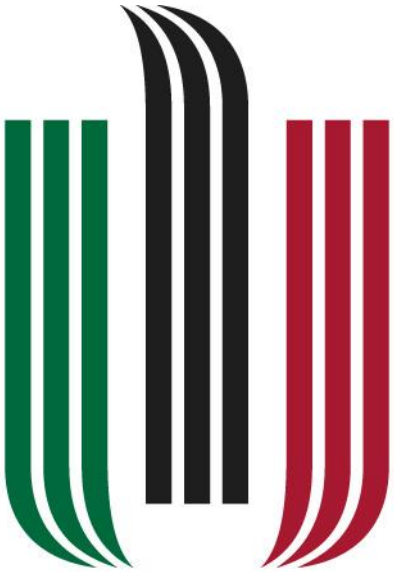
Inteligencja Obliczeniowa

Projekt I : Metaheurystyki



AGH

AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA
W KRAKOWIE

Projekt realizowali:

Michał Bubel
Katarzyna Czyż
Ewa Deoniziak
Justyna Zbiegień

Spis treści

Wstęp

Optymalizacja jest poszukiwaniem najlepszego rozwiązania spośród wszystkich możliwych, jednak w możliwym jak najkrótszym czasie, dlatego rozwiązanie optymalne niekoniecznie musi być rozwiązaniem najlepszym. Aby móc zacząć poszukiwania tego rozwiązania często wykorzystuje się ogólne algorytmy zwane metaheurystykami, które służą do rozwiązywania zadań obliczeniowych optymalizacyjnych. Do jej algorytmów możemy zaliczyć algorytmy ewolucyjne, systemy rozmyte czy sztuczne sieci neuronowe, a także metodę Wspinaczki, Symulowanego Wyżarzania czy Tabu Search, które będą obiektami badań w projekcie. Posłużą one do ułożenia 200 zadań w taki sposób, aby suma kwadratów odchyleń terminów pożądanych od planowanych była jak najniższa. Wszystkie kody, którymi się posłużono zostały napisane w języku *Visual Basic for Application*, a przed rozpoczęciem każdego algorytmu zadania są układane w kolejności rosnącej. Na koniec pracy porównano uzyskane wyniki z wynikiem uzyskanym przez użycie dodatku do programu Ms Excel – Solvera.

1. Metoda wspinaczki

Opis metody

Metoda wspinaczki, czyli Hill Climbing, jest jedną z podstawowych metod metaheurystyki. Metodę można również spotkać pod nazwą algorytmu największego wzrostu. Metoda ta nie buduje rozwiązania kawałek po kawałku, lecz daje wynik w całości – każdy punkt, który zostaje poddany analizie jest pełnym rozwiązaniem problemu, może być on lepszym bądź gorszym, które następnie jest lokalnie poprawiane, czyli w najbliższym sąsiedztwie. Jako zaletę metody wspinaczki można wymienić prostą jej implementacje, szybkie działanie oraz małe wymagania pamięciowe. Wadą jej jest duża zależność końcowego wyniku od początkowego punktu startu oraz uzyskiwanie takich rozwiązań, których nie da się poprawić dzięki prostej modyfikacji, ponieważ są one wynikiem (minimum) lokalnym. Algorytm natyka problemy również z określeniem kierunku poszukiwań, gdy na dużym obszarze wartość funkcji jest stała lub zmienia się nieznacznie.

Algorytm wspinaczki polega na rozpoczęciu poszukiwań z losowo wybranego punktu, a następnie przeanalizowaniu jego sąsiadów i wybraniu tego, który ma największą wartość. Wartość ta jest wyborem najlepszego rozwiązania jako początkowego do następnej iteracji, którą należy powtórzyć kilka razy (zalecane jest 2-4), aż znalezienie lepszego rozwiązania nie będzie możliwe.

Metoda Wspinaczki w kodzie

W metodzie Wspinaczki, jak wspomniano wyżej w opisie metody, najpierw wybiera się losową liczbę (zadanie) i sprawdza się jej sąsiedztwo. W zadaniu mamy do czynienia z zaledwie 200 zadaniami, dlatego założono, że sąsiedztwem losowego zadania jest każde inne zadanie wybrane z pozostałych. Dzięki temu założeniu kod może początkowo losować, nie jedno, lecz dwa zadania i sprawdzić ich zamianę. Program zapisuje stary wynik (sumę kwadratów różnic czasu wykonania zadania i jego terminu) i zamienia wylosowane zadania. Jeśli nowszy wynik okaże się być lepszy, program zostawia daną zamianę i od nowa losuje dwa zadania, jednak jeżeli wynik jest gorszy od starego, dane zadania zostają zamienione na swoje pierwotne pozycje a program ponawia cały

proces. Aby program nie utknął w optimum lokalnym, całość powtarza się w pięciu pętlach i dopiero program kończy swoje działanie. W programie zastosowano kolejno: 1.000, 5.000 oraz 10.000 iteracji (zamian) w każdej z pięciu pętli. Działanie to ma na celu sprawdzenie czy wynik metody Wspinaczki zależy od liczby jakie wykona.

Wynik kodu i analiza

Program opierający się na powyższych założeniach otrzymał następujące wyniki:

Dla 1000 iteracji	Dla 5000 iteracji	Dla 10000 iteracji
Wynik: 4 818 303	**Wynik: 2 614 227**	**Wynik: 2 546 890**

Nr	Kolejność zadań			Nr	Kolejność zadań			Nr	Kolejność zadań			Nr	Kolejność zadań			Nr	Kolejność zadań		
	1 tyś.	5 tyś.	10. tyś		1 tyś.	5 tyś.	10 tyś		1 tyś.	5 tyś.	10 tyś		1 tyś.	5 tyś.	10 tyś.		1 tyś.	5 tyś.	10 tyś.
1	83	17	17	**41**	164	191	50	**81**	178	112	70	**121**	134	119	15	**161**	2	29	29
2	17	83	83	**42**	50	50	62	**82**	174	111	111	**122**	144	168	163	**162**	14	49	66
3	12	133	56	**43**	51	24	191	**83**	57	71	42	**123**	173	15	168	**163**	29	2	2
4	195	56	98	**44**	132	148	24	**84**	70	4	71	**124**	188	139	139	**164**	197	66	49
5	180	98	133	**45**	24	164	148	**85**	72	42	141	**125**	198	107	107	**165**	149	153	153
6	122	54	54	**46**	187	99	164	**86**	92	141	57	**126**	1	198	36	**166**	81	31	95
7	7	195	184	**47**	190	62	88	**87**	112	57	4	**127**	80	134	134	**167**	18	95	116
8	172	16	195	**48**	191	140	99	**88**	65	131	92	**128**	163	36	11	**168**	74	18	18
9	98	184	16	**49**	19	85	140	**89**	128	92	131	**129**	114	11	198	**169**	95	81	158
10	184	38	115	**50**	9	88	53	**90**	4	174	174	**130**	107	68	22	**170**	145	116	31
11	67	172	172	**51**	148	187	85	**91**	90	90	75	**131**	139	22	68	**171**	125	39	39
12	54	115	38	**52**	85	53	187	**92**	64	75	90	**132**	47	173	173	**172**	151	158	125
13	133	122	122	**53**	52	28	28	**93**	75	65	93	**133**	186	101	47	**173**	155	125	81
14	38	130	12	**54**	103	9	19	**94**	43	93	65	**134**	68	188	101	**174**	182	149	76
15	121	67	130	**55**	100	19	8	**95**	93	61	61	**135**	22	47	188	**175**	169	76	149
16	16	12	67	**56**	137	8	9	**96**	135	26	128	**136**	11	136	186	**176**	77	69	167
17	115	121	7	**57**	161	51	51	**97**	71	128	43	**137**	101	118	136	**177**	89	167	182
18	40	7	55	**58**	8	143	166	**98**	13	176	26	**138**	109	186	1	**178**	21	74	77
19	192	55	192	**59**	143	32	143	**99**	60	135	176	**139**	15	96	96	**179**	76	182	69
20	84	110	121	**60**	181	166	32	**100**	102	60	135	**140**	86	109	118	**180**	120	77	185
21	5	192	35	**61**	62	159	34	**101**	46	43	60	**141**	162	1	109	**181**	153	151	151
22	157	35	110	**62**	23	181	159	**102**	59	64	64	**142**	136	73	86	**182**	69	41	74
23	56	84	84	**63**	140	34	181	**103**	61	10	10	**143**	66	86	73	**183**	79	193	193
24	55	180	180	**64**	111	52	52	**104**	26	20	20	**144**	108	162	162	**184**	39	138	41
25	170	40	170	**65**	159	100	161	**105**	154	3	46	**145**	73	144	6	**185**	97	169	169
26	142	170	40	**66**	146	196	100	**106**	189	46	3	**146**	199	106	144	**186**	41	185	138
27	130	157	157	**67**	194	161	196	**107**	176	59	59	**147**	6	94	94	**187**	185	21	21
28	147	142	142	**68**	99	103	23	**108**	10	175	104	**148**	167	6	108	**188**	78	156	97

29	150	147	147	**69**	32	23	103	**109**	20	104	189	**149**	177	14	14	**189**	158	120	156
30	171	150	150	**70**	124	152	194	**110**	119	189	58	**150**	106	108	106	**190**	91	97	126
31	105	105	105	**71**	87	194	152	**111**	58	160	160	**151**	94	177	177	**191**	156	78	120
32	35	190	190	**72**	196	82	82	**112**	104	13	13	**152**	183	145	183	**192**	138	126	78
33	53	171	171	**73**	131	178	124	**113**	168	58	175	**153**	96	183	145	**193**	165	79	27
34	179	37	5	**74**	166	87	178	**114**	36	154	154	**154**	31	89	89	**194**	63	27	91
35	88	5	200	**75**	152	124	87	**115**	127	113	113	**155**	49	129	30	**195**	45	91	79
36	200	179	37	**76**	42	146	146	**116**	113	127	44	**156**	117	199	199	**196**	27	165	63
37	37	132	179	**77**	82	25	25	**117**	3	80	80	**157**	118	155	197	**197**	193	33	33
38	48	48	137	**78**	34	70	72	**118**	44	44	127	**158**	129	30	129	**198**	33	63	165
39	28	200	48	**79**	25	102	112	**119**	175	163	114	**159**	30	197	117	**199**	126	123	123
40	110	137	132	**80**	141	72	102	**120**	160	114	119	**160**	116	117	155	**200**	123	45	45

Z powyższych tabel można jasno odczytać, że wyniki się zmniejszają wraz ze wzrostem iteracji. Jednak jest mniejsza różnica w wyniku między 10 tyś. a 5 tyś. iteracji niż między 1 tyś. a 5 tyś. iteracji. Oznacza to, że liczba iteracji ma wpływ na wynik jednak maleje ona wykładniczo, a nie wprost proporcjonalnie. Łatwo także zauważyć, że układ zadań z 10 tyś. iteracji nie wiele się różni od układu zadań z 5 tyś. iteracji. Program wychwycił mniejszą liczbę opłacalnych zamian, co oznacza, że dla określonej liczby iteracji jest w stanie znaleźć najlepsze rozwiązanie, jednak wtedy czas oczekiwania na wynik może się okazać zbyt długi.

2. Metoda symulowanego wyżarzania

Opis metody

Symulowane wyżarzanie (SA) , jak sama nazwa wskazuje, ma wiele wspólnego ze zjawiskiem wyżarzania w metalurgii. Wyżarzanie jest to jedna z operacji obróbki cieplnej metali, która polega na nagrzaniu materiału do określonej temperatury, podtrzymaniu jej, a następnie powolnym jej zmniejszaniu. Celem wyżarzania jest przybliżenie stanu materiału do warunków równowagi, by np. uzyskać specyficzne jego cechy, takie jak twardość. Metal posiada na początku niewielkie mankamenty, które osłabiają jego całą strukturę. Dzięki ogrzewaniu, a następnie studzeniu pozbywamy się ich. Temperatura początkowa ogrzewania nie może być zbyt niska, a studzenie musi być kontrolowane i powolne, aby nie osiągnąć odwrotności zamierzonego efektu. Symulowane wyżarzanie jest jedną z metod przeszukiwania sąsiedztwa, w której zakładamy, że dla każdego elementu rozwiązań potrafimy wyznaczyć wartość funkcji oceniającej jakość rozwiązania. Celem poszukiwań jest znalezienie minimum funkcji. Każdy punkt z przeszukiwanego obszaru jest postrzegany jako stan pewnego fizycznego obiektu, a minimalizowana funkcja jest analogiczna do stanu energii wewnątrz tego obiektu.

Algorytm symulowanego wyżarzania działa iteracyjnie zbliżając się do rozwiązania optymalnego, podobnie jak algorytm wspinaczki. Jeżeli nowe rozwiązanie jest lepsze to zostaje ono zatwierdzone jako nowe rozwiązanie, tak jak w algorytmie wspinaczki. Różnica się pojawia w momencie gdy nowa propozycja jest gorsza od wcześniejszego - wybieramy rozwiązanie losując je. Zostaje wprowadzany parametr „temperatura”, który reguluje proces wyboru kolejnych rozwiązań i na początku jest on dużą wartością. Jeśli wartość parametru jest duża – istnieje duże prawdopodobieństwo wyboru nowego, lepszego rozwiązania, zaś jeśli parametr jest mały to

prawdopodobieństwo wyboru nowego, ale gorszego rozwiązania, jest bardzo małe. W miarę poszukiwań wartość temperatury jest stale obniżana wraz z kolejnymi iteracjami, aż do otrzymania zerowej wartości, dzięki czemu w kolejnych przeszukiwaniach szansa przejścia do nowego, ale gorszego rozwiązania maleje.

Metoda Symulowanego Wyżarzania w kodzie

Metoda ta zaczyna się analogicznie do metody wspinaczki – losowane są dwa zadania spośród wszystkich. Program zapisuje stary wynik (sumę kwadratów różnic czasu wykonania zadania i jego terminu) i zamienia wylosowane zadania. Jeśli nowy wynik jest lepszy program zostawia zamianę i rusza do kolejnej pętli, gdzie powtarza cały proces. Natomiast jeżeli nowy wynik okazał się być gorszy (w tym kontekście większy) program przechodzi do funkcji *if()*, gdzie wyznacza „temperaturę". „Temperatura" jest oznaczona wzorem:

$$e * \frac{Nowe\ rozwiązanie\ -\ Stare\ rozwiązanie}{100 * i}$$

gdzie e jest liczbą Eulera równą w przybliżeniu 2,71, a i oznacza numer aktualnie wykonywanej iteracji. Program losuje liczbę losową z przedziału 0 do 1 i porównuje ją z „temperaturą". Jeśli „temperatura" okazała się być większa od losowej liczby program zamienia z powrotem zadania na ich miejsca pierwotne. Jak widać według tego programu im wyższa iteracji, tym niższa jest „temperatura", a co za tym idzie wysokie prawdopodobieństwo na znalezienie takiej kombinacji zadań, w której na pozór „gorsza" zamiana może się okazać zamianą słuszną mającą udział w ogólnie niższym wyniku.

Wynik kodu i analiza

Program został puszczony dla kolejno 1.000, 5.000, 10.000 iteracji w celu porównania wyników i sprawdzenia czy wynik jest zależny od liczby wykonywanych iteracji. Wyniki te są następujące:

Dla 1000 iteracji	Dla 5000 iteracji	Dla 10000 iteracji
Wynik: 30 234 299	**Wynik: 4 604 827**	**Wynik: 2 909 527**

Nr	Kolejność zadań		
	1 tyś.	5 tyś.	10 tyś.
1	172	16	83
2	148	115	17
3	17	133	195
4	132	122	54
5	5	184	133

Nr	Kolejność zadań		
	1 tyś.	5 tyś.	10 tyś.
41	121	40	50
42	157	8	137
43	55	48	62
44	82	164	48
45	173	62	164

Nr	Kolejność zadań		
	1 tyś.	5 tyś.	10 tyś.
81	176	111	42
82	141	71	111
83	104	112	72
84	20	174	70
85	137	70	57

Nr	Kolejność zadań		
	1 tyś.	5 tyś.	10 tyś.
121	1	114	139
122	59	198	15
123	162	86	36
124	126	118	163
125	160	163	168

Nr	Kolejność zadań		
	1 tyś.	5 tyś.	10 tyś.
161	74	197	29
162	183	106	66
163	109	66	2
164	63	117	49
165	193	18	116

6	195	17	172	**46**	102	85	24	**86**	37	124	141	**126**	73	101	44	**166**	149	29	153
7	7	83	184	**47**	110	179	99	**87**	147	92	131	**127**	125	113	68	**167**	167	39	158
8	8	56	56	**48**	25	37	85	**88**	57	26	4	**128**	13	139	11	**168**	158	125	76
9	67	130	98	**49**	53	200	53	**89**	51	93	90	**129**	44	186	134	**169**	39	149	18
10	56	38	115	**50**	190	137	19	**90**	181	135	93	**130**	4	96	22	**170**	129	95	149
11	122	54	16	**51**	48	140	88	**91**	60	42	174	**131**	145	107	1	**171**	30	185	81
12	38	55	122	**52**	100	148	187	**92**	199	43	65	**132**	65	15	173	**172**	66	153	95
13	16	195	67	**53**	103	88	28	**93**	80	65	92	**133**	3	36	188	**173**	76	167	41
14	83	172	12	**54**	54	187	140	**94**	64	46	61	**134**	139	134	118	**174**	116	158	31
15	184	12	7	**55**	101	52	8	**95**	140	57	75	**135**	108	177	86	**175**	175	2	182
16	19	7	84	**56**	26	100	51	**96**	135	61	176	**136**	119	188	198	**176**	78	193	167
17	179	180	35	**57**	85	181	159	**97**	189	60	26	**137**	146	11	101	**177**	113	116	151
18	99	84	38	**58**	9	191	143	**98**	198	4	135	**138**	154	136	186	**178**	155	151	125
19	143	150	55	**59**	152	161	166	**99**	15	75	128	**139**	77	173	47	**179**	45	21	77
20	133	170	130	**60**	111	9	9	**100**	161	160	64	**140**	75	145	94	**180**	95	31	193
21	28	99	142	**61**	23	82	34	**101**	72	13	60	**141**	6	108	136	**181**	120	120	74
22	130	98	121	**62**	88	196	161	**102**	42	176	43	**142**	106	47	108	**182**	182	182	185
23	87	192	170	**63**	32	143	52	**103**	11	10	20	**143**	174	1	73	**183**	31	138	69
24	84	110	192	**64**	92	32	32	**104**	70	25	59	**144**	144	6	162	**184**	41	69	39
25	24	35	200	**65**	61	51	181	**105**	14	104	10	**145**	29	30	14	**185**	185	33	97
26	166	67	180	**66**	196	23	100	**106**	10	20	3	**146**	197	94	109	**186**	186	41	169
27	171	105	150	**67**	71	19	152	**107**	79	189	46	**147**	168	81	144	**187**	169	74	21
28	35	147	110	**68**	105	102	146	**108**	163	128	189	**148**	89	162	106	**188**	21	169	78
29	40	5	40	**69**	62	194	82	**109**	96	64	58	**149**	153	199	129	**189**	165	78	156
30	200	190	147	**70**	90	34	103	**110**	93	59	104	**150**	86	155	183	**190**	50	77	138
31	191	121	157	**71**	177	159	23	**111**	124	58	154	**151**	94	144	96	**191**	47	45	79
32	115	53	179	**72**	142	178	124	**112**	128	168	80	**152**	188	73	199	**192**	91	27	27
33	12	142	105	**73**	187	87	178	**113**	36	22	175	**153**	69	183	30	**193**	117	97	126
34	34	24	171	**74**	159	103	194	**114**	81	3	107	**154**	18	68	145	**194**	138	126	120
35	131	157	5	**75**	43	72	25	**115**	58	175	113	**155**	134	109	6	**195**	27	79	123
36	52	132	132	**76**	164	141	112	**116**	127	80	119	**156**	2	129	155	**196**	49	156	63
37	150	171	190	**77**	180	146	196	**117**	46	44	114	**157**	33	89	89	**197**	97	91	33
38	192	28	37	**78**	22	131	71	**118**	118	119	13	**158**	107	76	177	**198**	156	63	91
39	170	50	191	**79**	112	90	102	**119**	178	154	127	**159**	68	14	117	**199**	151	165	45
40	98	166	148	**80**	194	152	87	**120**	114	127	160	**160**	136	49	197	**200**	123	123	165

Z powyższych tabel wyniki, że Symulowane Wyżarzanie, tak jak Wspinaczka, zależy od liczby iteracji, jednak Symulowane Wyżarzanie wykazuje znaczenie większą zmienność w zależności od liczby iteracji. Dla tysiąca iteracji wynik wynosi ponad 30 mln, gdzie jednocześnie przy takiej samej liczbie iteracji dla Wspinaczki wynik ten wynosił niecałe 5 mln. Oznacza to, że Symulowane Wyżarzanie jest bardziej wrażliwe na liczbę iteracji jednak jedynie przy małej ich liczbie. Końcowy wynik dla 10 tyś. iteracji jest gorszy (wyższy) niż dla Wspinaczki.

3. Metoda przeszukiwania Tabu Search

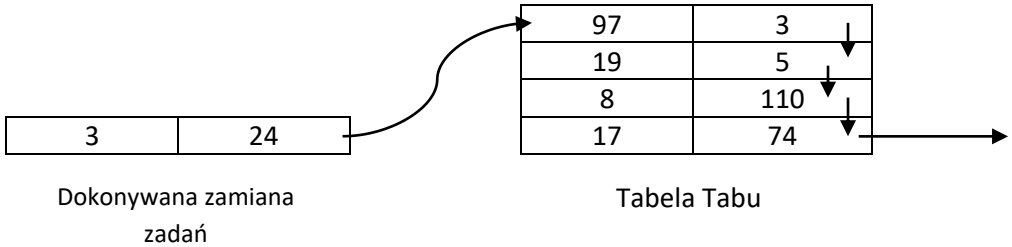
Opis metody

Tabu Search jest metaheurystyką, w której stosuje się nakładanie ograniczeń i wykorzystuje się je do znalezienia najlepszego rozwiązania. Tabu Search można określić jako metodę dynamicznej zmiany sąsiedztwa danego rozwiązania, które może się zmieniać w zależności od uzyskanych informacji na temat rozwiązywanego problemu. TS przypomina algorytm wspinaczki, a nawet niektórzy twierdzą, że TS jest jednym z wariantów wspinaczki, jednak podstawowa różnica między nimi polega na tym, że w TS mamy możliwość „odwiedzenia" sąsiadów, którzy posiadają gorsze wartości od aktualnie rozpatrywanych, więc możemy wyjść z maksimum lokalnego i dalej poszukiwać rozwiązań. W algorytmie wspinaczki nie można by tego zastosować, gdyż wpadlibyśmy w pętle – wychodząc z optimum lokalnego, a następnie od razu do niego wchodząc, a w TS lista tabu nam to uniemożliwia.

Algorytm TS polega na przejrzeniu przestrzeni rozwiązań, a następnie przechodzeniu do tego sąsiada, który posiada najwyższą wartość funkcji, o ile nie znajduje się on na liście punktów zabronionych, czyli tabu. Na liście tej są przechowywane *k* ostatnio odwiedzane punkty, które nie mogą zostać wykorzystane przez określoną liczbę iteracji. Zakończenie poszukiwań następuje po przekroczeniu limitu czasu, bądź wykonaniu określonych liczby iteracji.

Metoda TS w kodzie

Program bazuje na przejściu przez dwie zagnieżdżone pętle *for()*, w których wybiera po kolei zadanie i sprawdza jego zamianę z każdym pozostałym zadaniem. Dokonuje się to w analogiczny sposób jak w powyższych programach. Program wybiera dwa zadania, zamienia je miejscami i porównuje zapisany wcześniej stary wynik (sumę kwadratów różnic czasu wykonania zadania i jego terminu) z nowopowstałym wynikiem. Jeśli jest on lepszy zapisuje nowy wynik jako stary i zapamiętuje kombinację zdań, jednak zamienia jest z powrotem na pierwotne miejsca. Program bierze kolejne zadanie i znowu sprawdza w ten sposób zamianę. Gdy program sprawdzi wszystkie kombinacje z pierwszym zadaniem (czyli 199 kombinacji, ponieważ eliminuje się kombinację zamiany zadania z samym sobą), wybiera zapisaną „najlepszą" kombinację z najniższym wynikiem i dokonuje permanentnej zamiany. Zamiana ta jest dodana pierwszego wiersza tablicy Tabu, czyli tablicy zakazanych zamian. Następnie program zwiększa zadanie o 1 i znowu powtarza algorytm, jednak od teraz sprawdza najpierw czy najbardziej optymalna zamiana nie znajduje się w tablicy Tabu, jeśli tak to nie wykonuj daje zamiany tylko następną z kolei najlepszą zamianę. Tablica tabu posiada dwie kolumny (ponieważ zamienia się ze sobą dwa zadania – jedno zadanie jest umieszcza w jednej kolumnie) i tyle wierszy, na ile ma być blokowana dana zamiana. Przy każdej nowej zamianie zawartość wierszy (czyli zamiany) przesuwają się o jeden wiersz w dół, aż w końcu z niej „wypadną" i zostaną wykreślone z listy Tabu, np.:



Według powyższego przykładu w aktualnej iteracji dokonuje się zamiany zadania 3 z zdaniem 24. Ta zamiana wędruje do pierwszego wiersza tabeli, a pozostałe wiersze przesuwają się o jeden w dół, oprócz wiersza ostatniego, w którym znajduje się zamiana zadania 17 z zadaniem 24. Ta zamian zostaje usunięta z listy Tabu, a co za tym idzie, została ona umożliwiona przy następnym wyborze zamiany.

Wyniki kodu i analiza

W programie sprawdzono wyniki dla trzech różnych długości iteracji – 2, 5, 10 z długością tablicy Tabu równą 5 oraz dla trzech różnych długości tablicy Tabu – 5, 20, 500 dla 1 iteracji. Dokonano tyle zmian w celu sprawdzenia od czego zależy dany wynik w tej metodzie.

Wyniki w zależności od liczby iteracji:

Dla 2 iteracji	Dla 5 iteracji	Dla 10 iteracji
Wynik: 2 636 007	**Wynik: 2 537 330**	**Wynik: 2 537 330**

Nr	Kolejność zadań		
	2	5	10
1	83	17	17
2	17	83	83
3	56	133	133
4	98	56	56
5	195	98	98
6	184	54	54
7	16	195	195
8	172	184	184
9	133	16	16
10	54	115	115
11	115	172	172
12	122	38	38
13	12	122	122
14	38	12	12
15	7	130	130
16	130	7	7
17	67	67	67
18	192	55	55
19	110	192	192
20	121	121	121

Nr	Kolejność zadań		
	2	5	10
41	171	50	50
42	137	191	191
43	53	62	62
44	164	24	24
45	191	148	148
46	62	164	164
47	148	88	88
48	88	99	99
49	99	53	53
50	8	140	140
51	9	85	85
52	19	187	187
53	140	28	28
54	85	19	19
55	28	8	8
56	166	9	9
57	187	51	51
58	51	166	166
59	32	143	143
60	159	32	32

Nr	Kolejność zadań		
	2	5	10
81	70	102	102
82	146	42	42
83	71	111	111
84	102	71	71
85	57	141	141
86	141	57	57
87	4	4	4
88	92	92	92
89	131	131	131
90	75	75	75
91	90	90	90
92	174	174	174
93	93	93	93
94	61	65	65
95	43	61	61
96	65	128	128
97	128	43	43
98	26	26	26
99	60	176	176
100	135	60	60

Nr	Kolejność zadań		
	2	5	10
121	15	15	15
122	163	163	163
123	36	168	168
124	139	107	107
125	168	36	36
126	134	139	139
127	107	134	134
128	68	11	11
129	11	198	198
130	198	68	68
131	22	22	22
132	173	173	173
133	101	101	101
134	47	47	47
135	188	188	188
136	186	186	186
137	136	136	136
138	118	1	1
139	1	96	96
140	96	118	118

Nr	Kolejność zadań		
	2	5	10
161	29	29	29
162	66	66	66
163	49	49	49
164	2	2	2
165	116	153	153
166	153	95	95
167	95	116	116
168	18	18	18
169	158	158	158
170	31	31	31
171	81	81	81
172	76	39	39
173	39	76	76
174	125	125	125
175	149	149	149
176	182	182	182
177	77	77	77
178	167	167	167
179	69	69	69
180	151	74	74

21	55	35	35	**61**	34	34	34	**101**	176	135	135	**141**	109	109	109	**181**	185	151	151
22	180	110	110	**62**	181	159	159	**102**	10	64	64	**142**	86	86	86	**182**	74	185	185
23	170	84	84	**63**	100	181	181	**103**	3	10	10	**143**	73	73	73	**183**	193	193	193
24	84	180	180	**64**	52	52	52	**104**	20	20	20	**144**	162	162	162	**184**	41	41	41
25	40	170	170	**65**	143	100	100	**105**	46	46	46	**145**	6	144	144	**185**	169	169	169
26	35	40	40	**66**	196	196	196	**106**	64	3	3	**146**	144	6	6	**186**	138	138	138
27	157	142	142	**67**	152	161	161	**107**	59	59	59	**147**	94	94	94	**187**	21	21	21
28	147	157	157	**68**	194	23	23	**108**	104	104	104	**148**	108	108	108	**188**	156	97	97
29	105	147	147	**69**	103	103	103	**109**	189	189	189	**149**	106	106	106	**189**	97	156	156
30	150	150	150	**70**	161	194	194	**110**	160	58	58	**150**	177	14	14	**190**	78	78	78
31	200	105	105	**71**	23	152	152	**111**	58	160	160	**151**	183	177	177	**191**	120	120	120
32	142	171	171	**72**	82	82	82	**112**	113	13	13	**152**	14	183	183	**192**	126	126	126
33	190	190	190	**73**	124	124	124	**113**	13	175	175	**153**	145	145	145	**193**	79	79	79
34	179	5	5	**74**	178	178	178	**114**	175	154	154	**154**	89	89	89	**194**	27	27	27
35	132	200	200	**75**	25	146	146	**115**	154	113	113	**155**	30	30	30	**195**	63	33	33
36	37	179	179	**76**	112	87	87	**116**	44	80	80	**156**	199	199	199	**196**	33	63	63
37	50	37	37	**77**	87	25	25	**117**	80	44	44	**157**	197	197	197	**197**	91	91	91
38	48	48	48	**78**	72	72	72	**118**	114	114	114	**158**	129	129	129	**198**	165	165	165
39	24	137	137	**79**	42	112	112	**119**	127	127	127	**159**	117	117	117	**199**	45	45	45
40	5	132	132	**80**	111	70	70	**120**	119	119	119	**160**	155	155	155	**200**	123	123	123

Z powyższych tabel wynika, że dla 5 oraz 10 iteracji wynik jest ten sam. Nawet układ zadań jest identyczny. Oznacza to, że wraz ze wzrostem iteracji nie polepsza się wynik. Metoda Tabu Search jest mało wrażliwa na liczbę iteracji i dochodzi do pewnego momentu, w którym nie może znaleźć żadnego lepszego rozwiązania.

Wyniki w zależności od długości tablicy Tabu:

Długość tabeli tabu = 5	Długość tabeli tabu = 20	Długość tabeli tabu = 500
Wynik: 6 463 961	**Wynik: 6 463 961**	**Wynik: 6 463 961**

Nr	Kolejność zadań			Nr	Kolejność zadań			Nr	Kolejność zadań			Nr	Kolejność zadań			Nr	Kolejność zadań		
	5	20	500		5	20	500		5	20	500		5	20	500		5	20	500
1	195	195	195	**41**	159	159	159	**81**	141	141	141	**121**	198	198	198	**161**	29	29	29
2	172	172	172	**42**	148	148	148	**82**	111	111	111	**122**	134	134	134	**162**	155	155	155
3	16	16	16	**43**	19	19	19	**83**	196	196	196	**123**	11	11	11	**163**	66	66	66
4	184	184	184	**44**	23	23	23	**84**	4	4	4	**124**	58	58	58	**164**	116	116	116
5	115	115	115	**45**	28	28	28	**85**	174	174	174	**125**	22	22	22	**165**	2	2	2
6	192	192	192	**46**	152	152	152	**86**	46	46	46	**126**	168	168	168	**166**	18	18	18
7	67	67	67	**47**	161	161	161	**87**	100	100	100	**127**	101	101	101	**167**	31	31	31

8	130	130	130	**48**	98	98	98	**88**	10	10	10	**128**	47	47	47	**168**	95	95	95
9	122	122	122	**49**	62	62	62	**89**	102	102	102	**129**	139	139	139	**169**	158	158	158
10	38	38	38	**50**	105	105	105	**90**	112	112	112	**130**	113	113	113	**170**	149	149	149
11	170	170	170	**51**	99	99	99	**91**	103	103	103	**131**	186	186	186	**171**	153	153	153
12	83	83	83	**52**	110	110	110	**92**	90	90	90	**132**	15	15	15	**172**	39	39	39
13	133	133	133	**53**	194	194	194	**93**	61	61	61	**133**	188	188	188	**173**	41	41	41
14	179	179	179	**54**	191	191	191	**94**	57	57	57	**134**	1	1	1	**174**	167	167	167
15	190	190	190	**55**	181	181	181	**95**	160	160	160	**135**	36	36	36	**175**	125	125	125
16	180	180	180	**56**	34	34	34	**96**	128	128	128	**136**	68	68	68	**176**	81	81	81
17	84	84	84	**57**	32	32	32	**97**	65	65	65	**137**	173	173	173	**177**	151	151	151
18	54	54	54	**58**	140	140	140	**98**	26	26	26	**138**	144	144	144	**178**	77	77	77
19	147	147	147	**59**	143	143	143	**99**	176	176	176	**139**	109	109	109	**179**	182	182	182
20	137	137	137	**60**	5	5	5	**100**	64	64	64	**140**	162	162	162	**180**	76	76	76
21	35	35	35	**61**	52	52	52	**101**	135	135	135	**141**	136	136	136	**181**	69	69	69
22	171	171	171	**62**	85	85	85	**102**	104	104	104	**142**	6	6	6	**182**	138	138	138
23	150	150	150	**63**	70	70	70	**103**	60	60	60	**143**	96	96	96	**183**	185	185	185
24	121	121	121	**64**	25	25	25	**104**	75	75	75	**144**	73	73	73	**184**	74	74	74
25	132	132	132	**65**	50	50	50	**105**	119	119	119	**145**	106	106	106	**185**	21	21	21
26	164	164	164	**66**	82	82	82	**106**	59	59	59	**146**	86	86	86	**186**	193	193	193
27	40	40	40	**67**	178	178	178	**107**	131	131	131	**147**	118	118	118	**187**	97	97	97
28	56	56	56	**68**	146	146	146	**108**	154	154	154	**148**	108	108	108	**188**	169	169	169
29	55	55	55	**69**	48	48	48	**109**	163	163	163	**149**	177	177	177	**189**	120	120	120
30	157	157	157	**70**	88	88	88	**110**	175	175	175	**150**	199	199	199	**190**	156	156	156
31	53	53	53	**71**	124	124	124	**111**	3	3	3	**151**	197	197	197	**191**	78	78	78
32	8	8	8	**72**	51	51	51	**112**	189	189	189	**152**	183	183	183	**192**	79	79	79
33	166	166	166	**73**	42	42	42	**113**	71	71	71	**153**	14	14	14	**193**	126	126	126
34	17	17	17	**74**	72	72	72	**114**	20	20	20	**154**	89	89	89	**194**	27	27	27
35	187	187	187	**75**	24	24	24	**115**	80	80	80	**155**	117	117	117	**195**	63	63	63
36	142	142	142	**76**	92	92	92	**116**	107	107	107	**156**	145	145	145	**196**	91	91	91
37	7	7	7	**77**	43	43	43	**117**	114	114	114	**157**	94	94	94	**197**	33	33	33
38	200	200	200	**78**	87	87	87	**118**	44	44	44	**158**	129	129	129	**198**	165	165	165
39	37	37	37	**79**	93	93	93	**119**	13	13	13	**159**	49	49	49	**199**	45	45	45
40	12	12	12	**80**	9	9	9	**120**	127	127	127	**160**	30	30	30	**200**	123	123	123

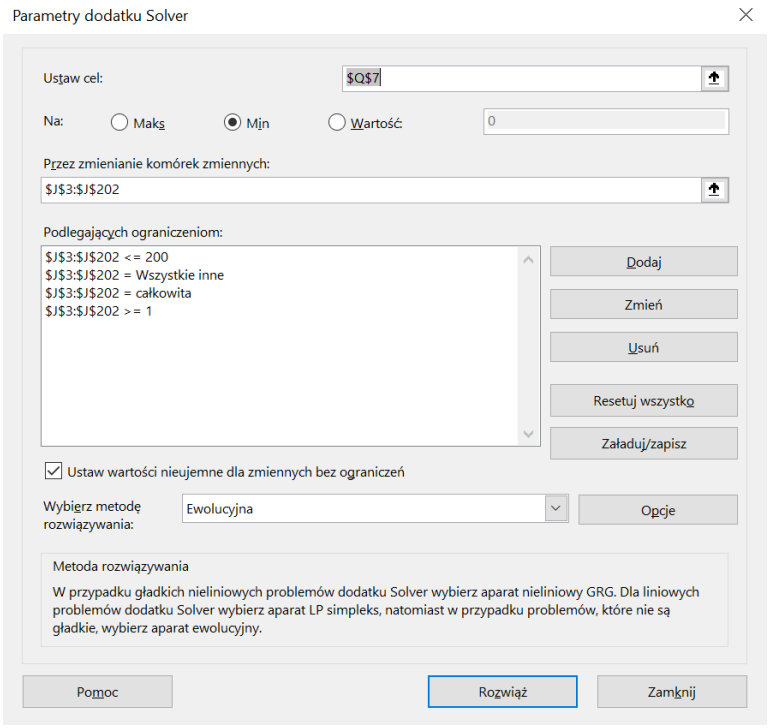
Z powyższych tabel wynika, że wynik Tabu Search kompletnie nie zależy od długości tablicy Tabu – wynik jest ten sam oraz układ zadań.

4. Solver

Dla porównaniu wyników użyto dodatku do programu Ms Excel – Solver. Umożliwia on przeprowadzanie analiz warunkowych. Za pomocą dodatku Solver można znaleźć optymalną (maksymalną lub minimalną) wartość formuły w jednej komórce — zwanej komórką celu — podlegającej ograniczeniom, czyli limitom. Dodatek Solver pracuje z grupą komórek, zwanych zmiennymi decyzyjnymi lub po prostu komórkami zmiennych, które służą do obliczania formuły w komórkach celu i komórkach ograniczeń. Dodatek Solver dostosowuje wartości w komórkach

zmiennych decyzyjnych tak, aby spełnić limity obejmujące komórki ograniczeń i uzyskać pożądany wynik w komórce celu.

Jako komórkę celu oznaczono komórkę z sumą kwadratów różnic czasu wykonania zadania i jego terminu, a jako komórki ze zmiennymi decyzyjnymi zostały oznaczone komórki z numerami zadań. Na poniższym obrazie widoczne są użyte ograniczenia:



Wyniki Solvera przedstawia poniższa tabela:

Solver
2 742 906

Wnioski

Analizując powyższe zestawienia, można zauważyć, że w metodzie wspinaczkowej oraz symulowanego wyżarzania znaczący wpływ na wynik ma liczba iteracji. Obie metody wykazują zależność wykładniczą tej wartości do wyniku końcowego, jednak metoda symulowanego wyżarzania charakteryzuje się pod tym względem większą zmiennością. W metodzie Tabu Search liczba iteracji posiada mniej zauważalny wpływ – po wykonaniu kilku iteracji algorytm znalazł optymalny wynik, który potem nie ulegał zmianie. Co więcej, w rozwiązywanym problemie, długość listy Tabu nie miała żadnego znaczenia na wynik końcowy.

Porównując uzyskane wyniki do wartości otrzymanej dzięki dodatkowi Solver, metoda symulowanego wyżarzania przy ustalonych liczbach iteracji nie pozwoliła na uzyskanie lepszego wyniku. Metoda wspinaczkowa pozwala na uzyskanie zadawalającego wyniku – lepszego niż „zaproponowanego" przez dodatek Solver, jednak tylko w przypadku wykonania dużej liczby iteracji. Najefektywniejszą metaheurystyką okazała się metoda Tabu Search, dzięki której uzyskano wynik lepszy od dodatku Solver już po dwóch iteracjach i najlepszy wynik ze wszystkich metod po

wykonaniu pięciu iteracji. Poniższa tabela jest zestawieniem wyników, potwierdzająca powyższe konkluzje:

Solver	Wspinaczka	Wyżarzanie	Tabu Search
2 742 906	**2 546 890**	**2 909 527**	**2 537 330**