ZESZYTY NAUKOWE POLITECHNIKI ŚLASKIEJ 2006
Seria: AUTOMATYKA z. 143 Nr kol. 1726

Bartosz WÓJCIK
Studium Doktoranckie Automatyki i Robotyki, Politechnika Śląska

SZEREGOWANIE ZADAŃ METODĄ PROGRAMOWANIA W LOGICE Z OGRANICZENIAMI

Streszczenie. W pracy przedstawiono zastosowanie ograniczeń kolejności wykonywania operacji zadań na danej maszynie oraz zawężenie domen startów poszczególnych operacji na podstawie analizy danych wejściowych. Do badań wykorzystano zestaw danych H. Fishera i G. L. Thompsona (1963) znany pod nazwą MT10 lub FT10.

JOB-SHOP SHEDULING IN CONSTRAINT LOGIC PROGRAMMING (CLP) METHOD

Summary. The use of constraint sequences of job's tasks on the machine and start domain filtering on a base of data input analysis is presenting. H. Fisher's and G. L. Thompson's dataset known as MT10 or FT10 was used in the research.

1. Wprowadzenie

Szeregowanie jest problemem rozdziału n zadań pomiędzy m maszyn. Każde zadanie składa się z k operacji, które należy wykonać w odpowiedniej kolejności, na odpowiednich maszynach, wspólnych dla wszystkich zadań. Ciąg kolejno wykonywanych operacji dla danego zadania nosi nazwę marszruty tego zadania. Zakłada się przy tym, że:

- w dowolnej chwili dowolna maszyna może wykonywać tylko jedną operację;
- wykonywanie operacji nie może być przerwane;
- dozwolona jest przerwa pomiędzy kolejnymi operacjami danego zadania;
- każda maszyna jest albo w pełni dostępna, albo niedostępna;
- wszystkie maszyny są niezależne, tzn. że żadna maszyna nie może zostać zastąpiona inną;
- wszystkie zadania są niezależne, tzn. wykonanie zadania nie jest uzależnione od stanu wykonania pozostałych zadań;

Celem szeregowania jest wyznaczenie takiego ciągu operacji wszystkich zadań na wszystkich maszynach, by zostały one wykonane w możliwie najkrótszym czasie.

2. Sformułowanie problemu job-shop

- Zadania: $Z = \{ z_1, z_2, \ldots z_i, \ldots z_n \}$.
- Maszyny: $M = \{ m_1, m_2, \ldots m_m \}$.
- Ciągi operacji: $O = \{ o_1, o_2, \ldots o_i, \ldots o_n \}$.
- Ciąg operacji dla zadania i-tego: $o_i = \{ m_{i1}, \ldots m_{i2}, \ldots m_{im} \}$ gdzie m_{iv} są elementami zbioru M, o_i – zbiór uporządkowany w kolejności realizacji operacji dla zadania i-tego, m_{ik}, $m_{i(k+1)}$ – oznacza kolejność wykonywania operacji: operacja na maszynie $m_{i(k+1)}$ jest wykonywana dopiero po zakończeniu operacji na maszynie m_{ik}, m_{i1} – maszyna, na której jest wykonywana pierwsza operacja dla zadania i-tego.
- Każda operacja ma swój czas trwania: $\tau_{i1}, \tau_{i2}, \ldots \tau_{im} \}$.
- Na zbiorze o_i została w ten sposób zdefiniowana dwójkowa relacja kolejności operacji, a więc dla m_{iv} oraz m_{iw} przy $v \le w$, m_{iv} musi być wykonane przed m_{iw}. Relacje kolejności są określone dla operacji każdego zadania z osobna. Nie określa się relacji kolejności pomiędzy operacjami różnych zadań.

Tabela 1
Kolejność wykonywania operacji

Z \ o	1	...	v	...	m
1	m_{11}	...	m_{1v}	...	m_{1m}
...	...	...	...	...	...
i	m_{i1}	...	m_{iv}	...	m_{im}
...	...	...	...	...	...
n	m_{n1}	...	m_{nv}	...	m_{nm}

Tabela 2
Czasy trwania operacji

Z \ o	1	...	v	...	m
1	τ_{11}	...	τ_{1v}	...	τ_{1m}
...	...	...	...	...	...
i	τ_{i1}	...	τ_{iv}	...	τ_{im}
...	...	...	...	...	...
n	τ_{n1}	...	τ_{nv}	...	τ_{nm}

Tabela 3
Czasy startu dla operacji zadań

Z \ o	1	...	v	...	m
1	s_{11}	...	s_{1v}	...	s_{1m}
...	...	...	...	...	...
i	s_{i1}	...	s_{iv}	...	s_{im}
...	...	...	...	...	...
n	s_{n1}	...	s_{nv}	...	s_{nm}

- Szeregowanie jest funkcją $S : Z \times O \to I \cup \{0\}$, określającą czas startu s_{iv} dla każdej operacji v każdego zadania i. Szeregowań takich może być bardzo dużo.
- Szeregowaniem optymalnym nazywa się takie szeregowanie, dla którego całkowity czas wykonania wszystkich zadań wynosi $T = \min(\max(s_{iv} + \tau_{iv}))$. Optymalne uszeregowanie umożliwia zakończenie zadania „job-shop" w możliwie najkrótszym czasie.

3. Dane wejściowe wg H. Fishera i G. L.Thompsona (1963)

Rozwiązanie zadania szeregowania z tak dobranymi danymi okazało się bardzo trudne. Minęło 20 lat zanim pojawiło się możliwe rozsądne rozwiązanie (w latach 1981-1982 wynosiło 935). Dopiero w Centrum Matematycznym w Amsterdamie w 1985 r. udało się uzyskać optymalne rozwiązanie 930 relaksacyjnymi metodami Lagrangiana. Zestaw danych składa się z 10 zadań, 10 operacji i 10 maszyn. Czasy trwania operacji były dobierane losowo z przedziału 1- 99 godzin. Maszyny były przydzielane odpowiednio do każdego z zadań. Pierwsze operacje zadań miały dolne

numery maszyn, a końcowe operacje zadań miały górne numery maszyn. Celem takiej marszruty było jak najlepsze odzwierciedlenie typowych procesów wytwarzania elementów na obrabiarkach.

Tabela 4

Dane wejściowe wg H. Fisher'a i G. L.Thompsona'a (1963)
gdzie: M – numer maszyny, T – czas trwania operacji

	Ope 1		Ope 2		Ope 3		Ope 4		Ope 5		Ope 6		Ope 7		Ope 8		Ope 9		Ope 10	
	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T
Zadanie 1	1	29	2	78	3	9	4	36	5	49	6	11	7	62	8	56	9	44	10	21
Zadanie 2	1	43	3	90	5	75	10	11	4	69	2	28	7	46	6	46	8	72	9	30
Zadanie 3	2	91	1	85	4	39	3	74	9	90	6	10	8	12	7	89	10	45	5	33
Zadanie 4	2	81	3	95	1	71	5	99	7	9	9	52	8	85	4	98	10	22	6	43
Zadanie 5	3	14	1	6	2	22	6	61	4	26	5	69	9	21	8	49	10	72	7	53
Zadanie 6	3	84	2	2	6	52	4	95	9	48	10	72	1	47	7	65	5	6	8	25
Zadanie 7	2	46	1	37	4	61	3	13	7	32	6	21	10	32	9	89	8	30	5	55
Zadanie 8	3	31	1	86	2	46	6	74	5	32	7	88	9	19	10	48	8	36	4	79
Zadanie 9	1	76	2	69	4	76	6	51	3	85	10	11	7	40	8	89	5	26	9	74
Zadanie 10	2	85	1	13	3	61	7	7	9	64	10	76	6	47	4	52	5	90	8	45

4. Budowa programu w języku CHIP

Oznaczenia zmiennych: $Z_{1,2,3}$ – czas rozpoczęcia operacji, $T_{1,2,3}$ – czas trwania operacji, gdzie: 1 – numer zadania; 2 – numer operacji; 3 – numer maszyny;

Na wstępie musimy każdej operacji przypisać domenę (przedział przyjmowanych wartości), aby wyznaczyć początkową, górną wartość domen, sumujemy czasy trwania wszystkich operacji i otrzymujemy wynik 5109, więc domena startów operacji to 0..5109. Ograniczenia kolejności danego zadania zapewnia nam predykat *precedens*, poniżej zastosowano predykat dla zadania pierwszego i analogicznie dla pozostałych dziewięciu zadań. Operacja $Z_{1,1,1}<Z_{1,2,2}$, $Z_{1,2,2}<Z_{1,3,3}$, operacja $Z_{1,1,1}$ jest wykonywana przed $Z_{1,2,2}$ i są to operacje tego samego zadania.

```
precedence([1],[Z111,Z122,Z133,Z144,Z155,Z166,Z177,Z188,Z199,Z1aa,E1],
                [P111,P122,P133,P144,P155,P166,P177,P188,P199,P1aa,0],
                [[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1]],
        [[1,2],[2,3],[3,4],[4,5],[5,6],[6,7],[7,8],[8,9],[9,10],[10,11]]),
```

Kolejnym ograniczeniem jest to, aby operacje poszczególnych zadań wykonywanych na tej samej maszynie nie nakładały się wzajemnie na siebie. Do tego ograniczenia wykorzystujemy predykat *cumulative*. Poniżej pokazano jego zastosowanie dla maszyny pierwszej, analogicznie dla pozostałych dziewięciu maszyn.

```
cumulative([Z111,Z211,Z911,Z321,Z521,Z721,Z821,Za21,Z431,Z671],
           [P111,P211,P911,P321,P521,P721,P821,Pa21,P431,P671],
           [   1,   1,   1,   1,   1,   1,   1,   1,   1,   1],
           unused,unused,1,unused,unused),
```

Różnica pomiędzy *precedence* a *cumulative*: predykat *precedence* dopuszcza opóźnienia pomiędzy operacjami, natomiast nie dopuszcza zmiany ustalonych kolejności operacji, zabezpiecza również przed nakładaniem się operacji. Predykat

cumulative dopuszcza opóźnienie i zmianę kolejności wykonywanych operacji oraz zabezpiecza przed nakładaniem się operacji.

Następnie musimy zoptymalizować nasze rozwiązanie, korzystając z predykatu min_max, a następnie ukonkretnić zmienne, korzystając z predykatu labeling.

```
min_max(labeling([Z111,Z122,.....Zaa8],0,input_order,indomain), Koniec)
```

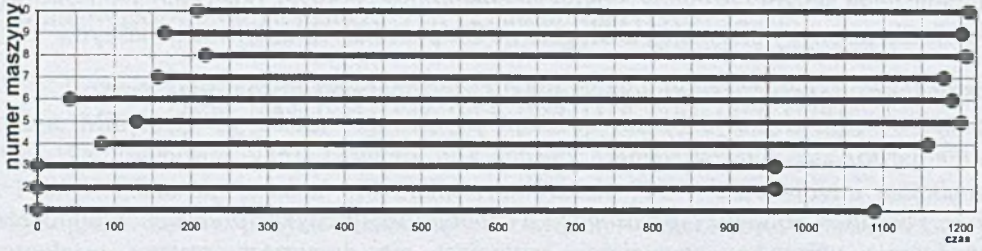
Na podstawie tak zbudowanego programu otrzymujemy pierwsze rozwiązanie po 10 s wynoszące 1232, które nie jest rozwiązaniem optymalnym. Po 10 h nadal brak rozwiązania optymalnego. Test został przeprowadzony na jednostce obliczeniowej z procesorem 1,6 GHz.

5. Zastosowanie dodatkowych ograniczeń na w/w przykładzie

Pierwsze możliwe rozwiązanie FT10 z podstawowymi ograniczeniami uzyskujemy po czasie 10 s i jest ono równe 1232. Przyjmuję więc jako górną granicę domen wartość 1232. Na każde zadanie nałożone są ograniczenia kolejności dla operacji (marszruta zadania) wynikające z procesu technologicznego. Początkowe domeny wszystkich operacji możemy zredukować z (0..5109) na (0..1232), po wykonaniu wcześniejszych obliczeń. Dalszy etap zawężania domen poszczególnych operacji opisany jest na poniższym przykładzie. Operacja trzecia zadania pierwszego na maszynie trzeciej ma domenę $Z_{1,3,3}$ = (107..944). Zawężenie tej domeny jest uzasadnione poniższej:

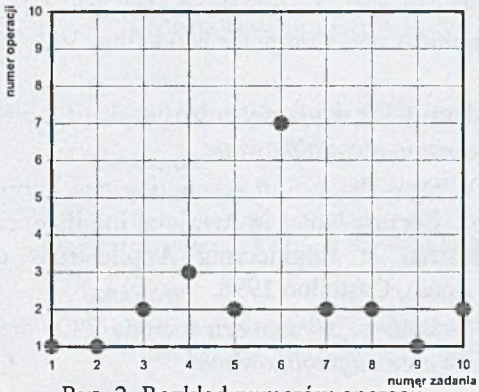
- Pełna domena startu operacji $Z_{1,3,3}$ :: 0..1232.
- Dolna granica domeny startu operacji $Z_{1,3,3}$ to suma czasów trwania poprzednich operacji tego zadania, $\tau_{1,1} + \tau_{1,2} = 107$.
- Górna granica domeny startu operacji $Z_{1,3,3}$ to różnica pomiędzy maksymalną wartością domeny a sumą czasów trwania operacji które należy jeszcze wykonać w tym zadaniu, $1232 - \sum_{i=3}^{10} \tau_{1i} = 944$.
- Wartości domen dla pozostałych operacji wyznaczamy w podobny sposób.

Na tej podstawie wyznaczamy przedziały pracy poszczególnych maszyn. Początek dla maszyny numer 1 jest równy $\min(Z_{i,j,1},\ldots Z_{10,k,1})$, a koniec $\max(Z_{i,j,1},\ldots Z_{10,k,1})$, podobnie dla pozostałych maszyn. Na rysunku 1 widoczne są przedziały pracy poszczególnych maszyn. Maszyny od 4 do 10 nie rozpoczynają pracy w chwili 0 tylko z pewnym opóźnieniem, wynikającym z ograniczeń kolejności danego zadania.

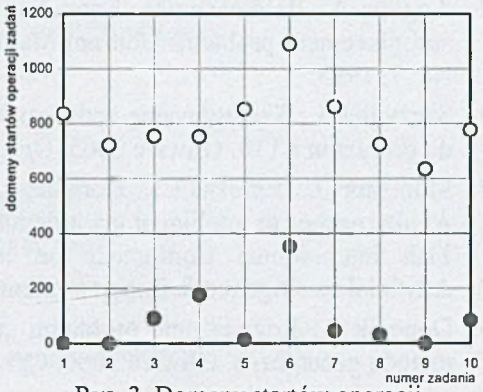


Rys. 1. Przedziały pracy poszczególnych maszyn w czasie

Na podstawie danych wejściowych możemy wyznaczyć dla każdej z maszyn rozkład wykonywanych numerów operacji wszystkich zadań. Rysunek 2 przedstawia rozkład dla maszyny 1 oraz kolejne zadania wraz z odpowiednim numerem operacji danego zadania wykonywanej na maszynie 1. Znając rozkład numerów operacji zadań na maszynie 1, możemy wyznaczyć graficzny rozkład domen startów operacji zadań na maszynie 1, co przedstawia rysunek 3. Znacznik pełny - dolna granica domeny startu operacji, znacznik pusty - górna granica domeny startu operacji, są to domeny operacji na maszynie 1. Podobnie wykonujemy rozkłady dla pozostałych maszyn.



Rys. 2. Rozkład numerów operacji zadań na maszynie 1



Rys. 3. Domeny startów operacji na maszynie 1

Ciąg operacji poszczególnych zadań wykonywanych na danej maszynie to marszruta maszyny. Możemy na podstawie rozkładu numerów operacji zadań utworzyć kilka ograniczeń kolejności operacji maszyny. Na rysunku 2 widać, że zadanie 6. ma 7. operację na maszynie 1.j, a zadanie 2. ma 1. operację na tej maszynie. Różnica jest równa 4 operacje, na tej podstawie wnioskujemy, że $Z_{2,1,1} < Z_{6,7,1}$. Postępowanie dla pozostałych operacji na tej maszynie jest podobne. Należy utworzyć wszystkie możliwe ograniczenia kolejności operacji dla pozostałych maszyn.

Wprowadzamy więc do programu kolejne ograniczenia, wykorzystując predykat *precedens*. Operacja zadania 2. jest wykonywana przed operacją zadania 6. (wszystkie pary uwzględnione są w ostatniej linijce predykatu). Podobnie postępujemy w tworzeniu dwójkowych par dla kolejności pozostałych maszyn.

```
precedence([1],[Z111,Z211,Z321,Z431,Z521,Z671,Z721,Z821,Z911,Za21],
                [P111,P211,P321,P431,P521,P671,P721,P821,P911,Pa21],
                [[1],[1],[1],[1],[1],[1],[1],[1],[1],[1]],
                [[2,6],[1,7],[7,4],[5,6],[8,6],[1,3]])
```

Program z tak dobranymi ograniczeniami kolejności operacji maszyn oraz z tak zawężonymi domenami startów operacji zadań znacznie szybciej wyznacza rozwiązanie optymalne problemu szeregowania job-shop. Po 43 minutach znajduje rozwiązanie optymalne. Test został przeprowadzony na tej samej jednostce obliczeniowej z procesorem 1,6 GHz.

Podziękowania

Autor pragnie wyrazić wdzięczność Panu Prof. Antoniemu Niederlińskiemu za pomoc i zaangażowanie.

LITERATURA

1. Mutch J. F., Thompson G. L.: Industrial Scheduling. Prentice-Hall Inc., Englewood Clifs, New Jersey 1963.
2. Aggoun A., Beldiceanu N.: Extending Chip in order to solve complex scheduling and placement problems. Journal Mathematical and Computer Modeling, Vol. 17, No. 7, 1993.
3. Szczygieł T.: Szeregowanie zadań metodami CLP, wykorzystując język ECLiPSe do problemu FT10. Gliwice 2005. *Opracowanie niepublikowane.*
4. Monostori L., Egresits Cs., Hornyák J., Viharos Zs. J: Soft computing and hybrid AI approaches to intelligent manufacturing. Lecture Notes in Artificial Intelligence, 11th International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems, Spain, Castellon 1998.
5. Donocik S.: Rozwiązanie problemu „m" zadań – „n" maszyn metodą CLP oraz metodą genetyczną. Gliwice 2000. *Opracowanie niepublikowane.*

Recenzent: Prof. dr hab. inż. Zbigniew Banaszak

Abstract

A job shop is a problem in which a set of *n jobs* $Z = \{z_1, \ldots, z_n\}$ has to be performed on a set of *m* machines $M = \{m_1, \ldots, m_m\}$. Each job z_i is composed of a set of tasks $o_i = \{m_{i1}, m_{ij}, \ldots, m_{im}\}$, where *i* is the index of the job, and *j* is the index of the step task in the overall job. In this context, tasks are regarded as scheduling entity. The job shop scheduling problem involves the synchronization of the completion of *n* jobs on *m* resources, known as the *NP-hard* combinatorial optimization problem. The first action is to use constraint sequences of job's tasks on the machine base of data input analysis. The second actions is start domains filtering, the lower limit of start task domain S_{iv} is equal to the sum of previous tasks time duration $\sum_{v=0} \tau_{iv}$, the upper limit of start task domain S_{iv} is equal to the difference between maximum value of the domain and the sum of time duration of the tasks that remain to do in the job $t_{\max} - \sum_{v=k} \tau_{iv}$. The use of constraint sequences of job's tasks on the machine base of data input analysis and start domains filtering significantly improve finding an optimal solution.