ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1990
Seria: AUTOMATYKA z.100 Nr kol.1082

Stanisław Zdrzałka
Politechnika Wrocławska

ALGORYTMY APROKSYMACYJNE DLA JEDNOMASZYNOWEGO ZAGADNIENIA SZEREGOWANIA Z JEDNAKOWYMI GRUPOWYMI CZASAMI PRZEZBROJEŃ *)

Streszczenie. W pracy przedstawiono wyniki analizy najgorszego przypadku algorytmów aproksymacyjnych dla jednomaszynowego zagadnienia szeregowania zadań z jednakowymi, grupowymi czasami przezbrojeń, w którym każde zadanie ma zadany czas gotowości do realizacji, czas wykonywania oraz czas dostarczenia, a problem polega na znalezieniu kolejności wykonywania minimalizującej czas, po którym wszystkie zadania są dostarczone. Podano również przegląd algorytmów aproksymacyjnych dla wersji zagadnienia bez przezbrojeń.

1. Wstęp

Dany jest zbiór zadań, utożsamiany dalej ze zbiorem indeksów $J=\{1,\dots,n\}$, które należy wykonać na pojedynczej maszynie; w danej chwili maszyna może wykonywać co najwyżej jedno zadanie. Zadanie j jest gotowe do realizacji w momencie r_j, potrzebuje p_j jednostek czasu na wykonanie, a czas potrzebny na jego dostarczenie, liczony od momentu zakończenia wykonywania przez maszynę, wynosi q_j; $r_j, q_j \geq 0$, $p_j > 0$. Zbiór zadań J jest rozbity na B podzbiorów (grup) $I_1,\dots,I_B$ i kiedy zadanie grupy I_b wykonywane jest bezpośrednio po zadaniu z grupy I_a, wymagane jest przezbrojenie maszyny, które zajmuje s_{ab} jednostek czasu; $s_{aa}=0$ dla każdego a. Rozważamy przypadek, kiedy czasy przezbrojeń są jednakowe, przyjmując, bez straty ogólności rozważań, że $s_{ab}=1$ dla $a \neq b$. Należy znaleźć uszeregowanie (kolejność wykonywania zadań), dla którego $\max_{j \in J}(C_j+q_j)$, to jest czas dostarczenia wszystkich zadań, przyjmuje wartość minimalną; przez C_j oznaczamy czas zakończenia wykonywania zadania j przez maszynę.

Zgodnie z notacją Grahama et al.[4], klasyczny, bez przezbrojeń, odpowiednik naszego problemu oznaczany jest przez $1|r_i,q_i|C_{max}$. Sformułowany problem jest silnie NP-trudny, ponieważ jego szczególny przypadek, problem $1|r_i,q_i|C_{max}$, należy do klasy problemów silnie NP-trudnych [7]. Do klasy problemów NP-trudnych należą również przypadki szczególne, gdy wszystkie r_j albo wszystkie q_j są jednakowe [2].

Rozpatrywany problem znany jest również pod następującą ekwiwalentną postacią. Niech K będzie stałą spełniającą warunek $K \geq \max_{j \in J} q_j$ i niech $d_j=K-q_j$ będzie pożądanym terminem zakończenia wykonywania zadania j. Łatwo

*) Praca była finansowana przez RP.I. 02 "Teoria sterowania i optymalizacji ciągłych układów dynamicznych i procesów dyskretnych".

zauważyć, że $\max_{j\in J}(C_j+q_j)=\max_{j\in J}(C_j-d_j)+K$, a zatem minimalizacja czasu, w którym wszystkie zadania są dostarczone, jest równoważna minimalizacji maksymalnej nieterminowości. Oba warianty problemu należą do podstawowych modeli teorii szeregowania, o wielu praktycznych i teoretycznych zastosowaniach; w tym drugim przypadku stosowane są do konstrukcji algorytmów dla bardziej złożonych zagadnień szeregowania.

Mimo tego, że dla problemu bez przezbrojeń istnieją wydajne algorytmy dokładne oparte na metodzie podziału i ograniczeń (patrz np. [3],[13] oraz przegląd literatury w [3]) oraz algorytm aproksymacyjny o oszacowaniu 4/3, bazujący na konstruktywnych własnościach samego problemu (patrz przegląd algorytmów w dalszej części niniejszej pracy), to jednak brak jest jak dotąd prób przeniesienia tych wyników na problem z przezbrojeniami. Badana była możliwość konstrukcji algorytmów dokładnych i aproksymacyjnych dla przypadku szczególnego, kiedy wszystkie czasy gotowości r_j, albo wszystkie q_j są jednakowe. W [8] sformułowano algorytm programowania dynamicznego dla tego przypadku, obejmujący również sytuację ogólną, kiedy czasy przezbrojeń zależą od grup. Algorytm ten jest wielomianowy ze względu na liczbę zadań i wykładniczy ze względu na ilość grup. Jednakże, poza przypadkami, w których B jest bardzo małe, posiada znaczenie wyłącznie teoretyczne. W [11] podano szereg własności ułatwiających konstrukcję algorytmów dla tego szczególnego problemu oraz przedstawiono serię algorytmów aproksymacyjnych o oszacowaniach błędu względnego max(1,n-1), 2 i 5/3; podejście tam zastosowane nie może być jednak rozszerzone na zagadnienie z różnymi czasami gotowości i różnymi czasami dostarczenia.

W pracy przedstawiamy próbę konstrukcji algorytmów aproksymacyjnych dla zagadnienia $1|r_i,q_i|C_{max}$ z jednakowymi grupowymi przezbrojeniami w oparciu o metodę analizy najgorszego przypadku, podobną do tej, która była stosowana w przypadku algorytmów dla zagadnienia bez przezbrojeń [9]. Podajemy ogólny algorytm, będący w pewnym sensie dalszym rozwinięciem uogólnionej reguły Jacksona (która okazała się być niezwykle silnym narzędziem dla przypadku bez przezbrojeń, patrz dalsza część pracy), i następnie badamy dokładność jego różnych wariantów i modyfikacji. W rezultacie przestawiono trzy algorytmy o oszacowaniach błędu względnego, odpowiednio, max(2,n-1), 3 i 8/3, oraz sformułowano prostą metodę ogólną, poddającą się analizie (najgorszego przypadku) i sugerującą dalsze modyfikacje mogące przynieść polepszenie dokładności. W przypadku gdy czasy gotowości r_j są jednakowe, oszacowanie 8/3 przechodzi w 5/3, a zatem otrzymany algorytm posiada tę samą dokładność z punktu widzenia analizy najgorszego przypadku, co algorytm podany w [12]; jest jednak od niego prostszy i posiada mniejszą złożoność obliczeniową.

2. Notacja

Niech π, permutacja zbioru J, określa kolejność wykonywania zadań; przez

$\pi(i)$ oznaczamy numer zadania na pozycji i w permutacji π. Dla dowolnego π oraz i, $2\le i\le n$, niech $s_{\pi(i)}=0$, jeżeli $\pi(i)$ oraz $\pi(i-1)$ należą do tej samej grupy, oraz $s_{\pi(i)}=1$ w przeciwnym przypadku; $s_{\pi(1)}=0$.

Dla danej kolejności wykonywania zadań π czas dostarczenia wszystkich zadań, oznaczany dalej przez $C_{max}(\pi)$, wynosi

$$C_{max}(\pi)=\max_{1\le i\le j\le n}(r_{\pi(i)} + \Sigma_{k=i}^{j}(p_{\pi(k)}+s_{\pi(k)}) + q_{\pi(j)}). \tag{1}$$

Rozpatrywany tu problem polega więc na znalezieniu permutacji π^*, w zbiorze wszystkich permutacji zbioru J, dla której funkcja C_{max} przyjmuje wartość minimalną; oznaczamy $C^*=C_{max}(\pi^*)$.

Oznaczmy dalej przez u_1 i u_2 wartości, odpowiednio i oraz j, dla których osiągane jest maksimum w wyrażeniu (1). Ciąg zadań $(\pi(u_1),\dots,\pi(u_2))$ nazywać będziemy *blokiem*; przez W oznaczać będziemy zbiór zadań bloku, $W=\{\pi(u_1),\dots,\pi(u_2)\}$, natomiast przez k, liczbę przezbrojeń w bloku permutacji π, z wyłączeniem przezbrojeń występujących bezpośrednio przed zadaniem $\pi(u_1)$ i za zadaniem $\pi(u_2)$.

Dla $V\subset J$, niech $z(V)$ będzie liczbą grup, których zadania należą do V. Dla danego π, maksymalny ciąg kolejnych zadań z tej samej grupy nazywamy *segmentem*. Ciąg segmentów $L_1,\dots,L_v$ permutacji π, taki, że $\pi=(L_1,\dots,L_v)$ (π jest połączeniem ciągów $L_1,\dots,L_v$) nazywamy *segmentową reprezentacją* π; v-1 jest liczbą przezbrojeń dla kolejności π. Należy zauważyć, że dla dowolnego π, $v\ge B$. Permutację, dla której v=B, nazywamy *uszeregowaniem grupowym*; w uszeregowaniu takim zadania każdej grupy wykonywane są kolejno po sobie, a liczba przezbrojeń osiąga swoją minimalną wartość B-1. Dla $V\subset J$, oznaczamy: $P(V)=\Sigma_{j\in V}p_j$, $r(V)=\min_{j\in V}r_j$, $q(V)=\min_{j\in V}q_j$.

Zgodnie z powyższymi oznaczeniami,

$$C_{max}(\pi) = r_{\pi(u_1)}+P(W)+k+q_{\pi(u_2)} \tag{2}$$

W dalszym ciągu będziemy często wykorzystywać następujące dolne ograniczenie na C^*. Dla dowolnego $V\subset J$,

$$C^* \ge r(V)+P(V)+z(V)-1+q(V). \tag{3}$$

Nierówność ta wynika bezpośrednio z (1).

Dokładność algorytmu aproksymacyjnego A dla pewnego ustalonego problemu konkretnego oceniać będziemy za pomocą ilorazu C_A/C^*, gdzie C_A jest wartością funkcji celu otrzymaną przez algorytm A. Dokładność algorytmu A dla dowolnego problemu konkretnego określa parametr ρ_A,

$$\rho_A = \inf\{\zeta\ge 1:\ C_A/C^* \le \zeta \text{ dla każdego problemu konkretnego}\}.$$

Tak więc, dla każdego problemu konkretnego, $C_A/C^*\le\rho_A$, i ograniczenie to jest najlepsze z możliwych. W [1], ρ_A nazywane jest bezwzględnym oszacowaniem (błędu algorytmu). W niniejszej pracy, dla określenia górnego ograniczenia na ρ_A, używamy nazwy *oszacowanie błędu względnego*, natomiast

samo ρ_A nazywane jest *parametrem najgorszego przypadku*.

3. Algorytmy aproksymacyjne dla przypadku B=1

Przedstawimy najpierw przegląd algorytmów aproksymacyjnych dla problemu $1|r_i,q_i|C_{max}$ bez przezbrojeń (B=1), wyniki analizy najgorszego przypadku związane z tym problemem, i na tym tle sformułujemy w dalszym ciągu pracy algorytmy aproksymacyjne dla zagadnienia z przezbrojeniami (B>1).

Pierwszy algorytm aproksymacyjny sformułowany został w [10]; nosi on często nazwę Uogólnionej Reguły Jacksona (URJ). Mówi ona, że jeżeli w danej chwili maszyna jest wolna, to spośród zadań dostępnych do realizacji należy wybrać do wykonywania to, które ma najdłuższy czas dostarczenia. Bardziej sformalizowany opis tego algorytmu, przystosowany do dalszego wykorzystania wewnątrz innych procedur, ma postać następująca:

procedure URJ(S,t,α)
(Dane wejściowe: S - zbiór zadań, które należy ustawić po ostatnim zadaniu ciągu początkowego α, t - najwcześniejszy możliwy termin rozpoczęcia wykonywania zadań ze zbioru S, α - początkowy ciąg zadań ($S\cap\alpha=\emptyset$);
Dane wyjściowe: α - ciąg zadań będący połączeniem ciągu początkowego i ciągu otrzymanego przez uporządkowanie S, t - czas zakończenia wykonywania ostatniego zadania w α)

```
begin
  while S≠∅ do
    begin
      Q:={j∈S: r_j≤t};
      R:={j∈Q: q_j=max_{k∈Q} q_k};
      j:= dowolne zadanie ze zbioru R;
      α:=(α,j) (dołączenie j do α na ostatniej pozycji);
      S:=S\{j};
      t:=max(t+p_j,min_{k∈S} r_k);
    end
end
```

Analiza najgorszego przypadku algorytmu URJ oraz konstrukcja i analiza dalszych jego modyfikacji opierają się następujących spostrzeżeniach. Niech $(\pi(u_1),\ldots,\pi(u_2))$ będzie blokiem w permutacji π otrzymanej przez URJ (dla danych wejściowych: $J,\min_{j\in J} r_j,\emptyset$) o najmniejszym możliwym u_1. Łatwo można sprawdzić, że $r_{\pi(u_1)} \le r_{\pi(i)}$, $u_1\le i\le u_2$. Stąd, jeżeli $q_{\pi(u_2)}\le q_{\pi(i)}$ dla $u_1\le i<u_2$, to π jest uszeregowaniem optymalnym, ponieważ $C^* \ge r_{\pi(u_1)}+P(W)+q_{\pi(u_2)} = C_{max}(\pi)$. Jeżeli warunek ten nie jest spełniony, to znaczy istnieje zadanie $\pi(c)$, $u_1\le c<u_2$, takie że $q_{\pi(c)}< q_{\pi(u_2)}$ i $q_{\pi(i)}\ge q_{\pi(u_2)}$ dla $c+1\le i\le u_2$, to, [9],

$$C_{URJ}-C^{*} \le p_{\pi(c)}. \qquad (4)$$

Zadanie $\pi(c)$ nazywane jest dalej *kolizyjnym*.

Parametr najgorszego przypadku dla URJ wynosi 2 [9]. Jak wynika z (4), jeżeli $p_j \le (1/2)P(J)$ dla każdego j, to $C_{URJ}/C^{*} \le 3/2$. Ponieważ istnieje co najwyżej jedno zadanie, które tej nierówności nie spełnia, widać więc, że wpływając na położenie tego zadania w uszeregowaniu w ten sposób, aby nie było ono zadaniem kolizyjnym, możemy ewentualnie otrzymać modyfikację URJ o oszacowaniu błędu względnego mniejszym niż 2. Jest to idea modyfikacji zaproponowanej przez Pottsa [9], nazywanej dalej algorytmem MURJ, dla której $\rho_{MURJ}=3/2$.

```
procedure MURJ
begin
  repeat
    π:=∅;
    URJ(J,min_{j∈J} r_j,π)
      if istnieje zadanie kolizyjne π(c), then r_π(c):=r_π(u_2);
  until nie istnieje zadanie kolizyjne lub wykonano n iteracji;
  wybierz uszeregowanie o najmniejszej wartości funkcji celu;
end
```

Algorytmy URJ i MURJ można łatwo przystosować do rozwiązywania zagadnienia z ograniczeniami kolejnościowymi. W tym celu wystarczy tak modyfikować wartości r_j i q_j, ażeby nie "zniekształcając" problemu, wymusić na URJ "automatyczne" spełnianie ograniczeń kolejnościowych. W przypadku URJ warunek ten spełnia modyfikacja: jeżeli zadanie i poprzedza j, to $r_j := \max(r_i, r_j)$ oraz $q_i := \max(q_i, q_j)$. Algorytm z tego typu modyfikacja oznaczamy przez URJ'. W przypadku MURJ należy zastosować URJ' w miejsce URJ oraz rozszerzyć modyfikowanie czasów gotowości, w instrukcji if, na te wszystkie zadania, obok $\pi(c)$, które muszą być poprzedzane przez $\pi(c)$.

Ostatnio Hall i Shmoys [6] przedstawili modyfikację algorytmu MURJ dla rozwiązywania $1|r_i,q_i,prec|C_{max}$, nazywaną dalej HS, dla której $\rho_{HS}=4/3$. Polega ona na stosowaniu MURJ' do oryginalnej i odwróconej wersji problemu, przy czym w przypadkach, gdy istnieją dwa długie zadania u i v takie, że $p_u \cdot p_v > (1/3)P(J)$, w miejsce n należy stosować 2n-3 iteracji procedury URJ'. Przez wersję odwróconą rozumiemy problem, w którym role r_j i q_j są zamienione, a ograniczenia kolejnościowe odwrócone.

```
procedure HS
begin
  begin
    if nie istnieją zadania u i v takie, że p_u·p_v>(1/3)P(J) then
      zastosuj MURJ' do oryginalnej i odwróconej wersji problemu;
    else
      if istnieje relacja kolejnościowa dla u i v then
```

zastosuj MURJ' z 2n-3 iteracjami do oryginalnej i odwróconej wersji problemu;

else dodaj ograniczenie "u poprzedza v" i zastosuj MURJ' z 2n-3 iteracjami do oryginalnej i odwróconej wersji problemu;

dodaj ograniczenie "v poprzedza u" i zastosuj MURJ' z 2n-3 iteracjami do oryginalnej i odwróconej wersji problemu;

end

wybierz uszeregowanie o najmniejszej wartości funkcji celu;

end

Hall i Shmoys [6] podali również dwa wielomianowe schematy aproksymacyjne dla $1|r_i,q_i|C_{max}$, doprowadzając w ten sposób dyskusję nad algorytmami aproksymacyjnymi dla tego zagadnienia niejako do końca; uzyskanie w pełni wielomianowego schematu aproksymacyjnego nie jest możliwe (przy założeniu P≠NP), ponieważ problem $1|r_i,q_i|C_{max}$ jest silnie NP-trudny.

Poniżej przedstawiamy zestawienie omówionych algorytmów.

Algorytm A	ρ_A	Złożoność oblicz.	Uwagi
URJ, URJ'	2	$O(n\log n)$	[9]
MURJ, MURJ'	3/2	$O(n^2\log n)$	[9]
HS	4/3	$O(n^2\log n)$	[6]
Wielom. sch. aproks. C	$1+\varepsilon$	$O(16^{(1/\varepsilon)}(n/\varepsilon)^{3+4/\varepsilon})$	[6]
Wielom. sch. aproks. D	$1+\varepsilon$	$O(n\log n+n(4/\varepsilon)^{8/\varepsilon^2+8/\varepsilon+2})$	[6]

4. Algorytmy aproksymacyjne dla przypadku B>1

Zajmować się będziemy wyłącznie zagadnieniem bez ograniczeń kolejnościowych. Rozpoczniemy od algorytmu, który jest rozszerzeniem Uogólnionej Reguły Jacksona. Niezwykle korzystne własności tej reguły usprawiedliwiają próbę jej wykorzystania również w zagadnieniu z przezbrojeniami.

W omawianym algorytmie zakłada się, że każda grupa I_a rozbita jest na m_a podzbiorów (podgrup) $I_{a1},\dots,I_{a,m_a}$. Zadania każdej z podgrup ustawiane są kolejno po sobie według procedury URJ. Z kolei podgrupy wybierane są według zasady: jeżeli w danej chwili maszyna jest wolna, to wybierz do realizacji podgrupę, do której należy zadanie o najdłuższym czasie dostarczenia spośród zadań dostępnych do wykonywnia. Algorytm ten zapisujemy w postaci następującej procedury.

procedure A(I_{ai}, i=1,...,m_a, a=1,...,B, π)

(Dane wejściowe: podgrupy I_{ai}, i=1,...,m_a, a=1,...,B; Dane wyjściowe: permutacja π)

begin

T:=J; τ:=$\min_{j\in J} r_j$; t:=τ-1; π:=∅; b:=0;

```
while T≠∅ do
  begin
    Q:={j∈T: r_j≤τ};
    R:={j∈Q: q_j=max_{k∈Q} q_k};
    (a,l):= indeksy podgrupy, do której należy zadanie ze zbioru R;
      if a≠b then t:=max(τ,t+1);
      else t:=τ;
    URJ(I_al,t,π);
    b:=a;
    T:=T\I_al;
    τ:=max(t,min_{j∈T} r_j);
  end
end
```

W algorytmie A, podział grup na podgrupy jest arbitralny. Rozpatrzmy dwie skrajne możliwości takiego podziału. Pierwsza, kiedy każde zadanie tworzy oddzielną podgrupę (algorytm A1), i druga, kiedy grupy nie są rozbite na podgrupy; $m_a=1$ dla $a=1,\dots,B$ (algorytm A2). Algorytm A1 zachowuje się tak jak URJ w odniesieniu do zagadnienia bez przezbrojeń - w doborze zadań ignoruje przezbrojenia. Drugie podejście, A2, jest jego przeciwstawieniem - zakłada minimalną liczbę przezbrojeń, B-1.

Twierdzenie 1. Dla algorytmów A1 i A2 zachodzi:

a) $\rho_{A1} = \max(2,n-1)$,

b) $\rho_{A2} = 3$.

Dowód. Niech π będzie permutacją otrzymaną przez algorytm A; dalej przyjmować będziemy, że pozycja u_1 pierwszego zadania bloku w π jest najmniejsza z możliwych. Na podstawie (3) mamy nierówność

$$C^* \geq r(W)+P(W)+z(W)-1+q(W). \tag{6}$$

a) Z opisu A1 wynika, że $\pi(u_1)$ jest albo pierwszym zadaniem w π, albo bezpośrednio przed $\pi(u_1)$ maszyna nie wykonuje żadnego zadania ani nie jest przezbrajana. Zatem $r_{\pi(u_1)}=r(W)$, i na podstawie (2) i (6) mamy,

$$C_{A1}-C^* \leq k-z(W)+1+q_{\pi(u_2)}-q(W). \tag{7}$$

Jeżeli $z(W)=1$, to $k=0$ i w konsekwencji $C_{A1}-C^* \leq q_{\pi(u_2)} \leq C^*$.

Załóżmy, że $z(W)\geq 2$. Oznacza to, że w skład bloku wchodzą zadania przynajmniej dwóch segmentów. Rozważmy dwa segmenty; segment zawierający zadanie $\pi(u_2)$ i segment bezpośrednio go poprzedzający. Jeżeli jeden z tych segmentów zawiera zadanie $\pi(j)$ należące do bloku i spełniające $q_{\pi(j)}<q_{\pi(u_2)}$, to zachodzi $C_{A1}/C^* \leq 2$. Rzeczywiście, niech $\pi(j)$ będzie pierwszym takim zadaniem idąc od $\pi(u_2)$ do $\pi(u_1)$, oraz niech τ będzie momentem jego rozpoczęcia. Zgodnie z algorytmem, w chwili τ zadanie $\pi(u_2)$ nie było dostępne, zatem $r_{\pi(u_2)}>\tau$. Niech $T=\{\pi(j),\dots,\pi(u_2)\}$. Mamy wtedy

$C_{A1} \le \tau+P(T)+1+q_{\pi(u_2)}$ i $C^* \ge \tau+q_{\pi(u_2)}$. Stąd $C_{A1}-C^* \le P(T)+1 \le C^*$.
Rozważmy teraz przypadek, gdy żaden z rozważanych segmentów nie zawiera zadania $\pi(j)$. Wynika stąd, na podstawie (2), że zachodzi $C^* \ge 1+q_{\pi(u_2)}$.
Łącząc tę nierówność z (7), oraz wykorzystując fakt, że $k \le n-1$, otrzymujemy $C_{A1}/C^* \le n-1$, co łącznie z ograniczeniami poprzednio rozważanych przypadków daje $C_{A1}/C^* \le \max(2, n-1)$.

Pokażemy teraz, że ograniczenie to jest najlepsze z możliwych. Niech p_{aj} i q_{aj} będą, odpowiednio, czasem wykonywania i czasem dostarczenia zadania j-tego w grupie a. Zakładamy, że: B=2, n=2M, oraz że dla j=1,...,n dane poszczególnych zadań są jak następuje:

a	r_{aj}	p_{aj}	q_{aj}
1	0	2ε	$(2j-1)\varepsilon$
2	0	2ε	$(2j-2)\varepsilon$

gdzie $0<2M\varepsilon<1$. Można łatwo sprawdzić, że algorytm A1 generuje permutację
((1,M),(2,M),(1,M-1),(2,M-1),...,(1,1),(2,1)),
gdzie para (a,j) oznacza j-te zadanie grupy a, dla której $C_{A1}=n-1+2n\varepsilon$.
Uszeregowanie
((1,M),...(1,1),(2,M),...,(2,1))
daje wartość funkcji celu $1+2n\varepsilon \ge C^*$. Zatem $C_{A1}/C^* \ge (n-1+2n\varepsilon)/(1+2n\varepsilon)$, gdzie prawa strona nierówności może być dowolnie bliska n-1 dla $\varepsilon \to 0$.
b) Na podstawie (6),(7) oraz faktu, że $k=z(W)-1$ mamy,
$C_{A2}-C^* \le r_{\pi(u_1)}-r(W)+q_{\pi(u_2)}-q(W) \le 2C^*$.

Następujący przykład pokazuje, że ograniczenie to jest najlepsze z możliwych. Niech będzie B grup, a w każdej grupie 3 zadania o następujących danych, jednakowych dla a=1,...,B.

j	r_{aj}	p_{aj}	q_{aj}
1	0	1	BA+B-1
2	0	A	0
3	BA+B-1	1	0

Algorytm A2 daje uszeregowanie
((1,1),(1,2),(1,3), ... ,(B,1),(B,2),(B,3))
z wartością funkcji celu $C_{A2}=3AB+5B-2A-5$. Natomiast dla permutacji
((1,1),...,(B,1),(1,2),...,(B,2),(1,3),...,(B,3))
wartość funkcji celu wynosi $AB+5B-3 \ge C^*$. Zatem $C_{A2}/C^* \ge (3AB+5B-2A-5)/(AB+5B-3)$, przy czym prawa strona tej nierówności dąży do 3 dla A i B dążących do nieskończoności. □

Rozpatrzymy teraz modyfikację algorytmu A, w której procedura A powtarzana jest w odniesieniu do zmieniającego się, z iteracji na iterację, rozbicia grup na podgrupy. Zakłada się przy tym, że każda grupa I_a składa

się z co najwyżej dwóch podgrup I_{a1} i I_{a2}, przy czym początkowo $I_{a1}:=I_a$ i $I_{a2}:=\emptyset$. W każdej kolejnej iteracji algorytmu A, zadanie o najmniejszym q_j w bloku , mniejszym od $\pi(u_2)$, przenoszone jest z odpowiadającej mu podgrupy I_{a1} do I_{a2} o tym samym indeksie a.

Ponadto, przyjmuje się w niej, że czasy gotowości zadań są zmodyfikowane następująco: $r_j:=r_{max}, j\in J$, gdzie $r_{max}=\max_{j\in J} r_j$. Ponieważ, dla potrzeb algorytmu, czasy gotowości są jednakowe, procedura A ustawia zadania w podgrupach według nierosnących q_j, zaś podgrupy według nierosnących $\max_{j\in I_{ai}} q_j$. Procedura kończy się, jeżeli brak jest takiego zadania lub blok w kolejno utworzonej permutacji zawiera niepustą podgrupę I_{a2}.

```
procedure A3
begin
  for a:=1 to B do I_a1:=I_a; I_a2:=∅;
  for j:=1 to n do r_j:=r_max;
    repeat
      A(I_a1,I_a2,a=1,...,B,π);
      j:=zadanie ze zbioru W spełniające q_j=q(W);
      a:=grupa, do której należy j;
      I_a1:=I_a1\{j}; I_a2:=I_a2∪{j};
    until q(W)≥q_π(u_2) or k>z(W)
    wybierz permutację π o najmniejszej wartości funkcji celu;
end
```

<u>Twierdzenie 2</u>. Dla każdego problemu konkretnego, $3C_{A3} \le 5C^* + 3r_{max}$.

<u>Dowód</u>. Niech π będzie permutacją, dla której $k>z(W)$. Ponieważ $C_{max}(\pi)$ jest dla ustalonego π niemalejącą funkcją dowolnego ze swoich parametrów r_j, p_j, q_j, $j\in J$, przy ustalonych pozostałych, stąd

$$C_{A3} \le r_{max}+P(W)+k+q_{\pi(u)} \le r_{max}+P(J)+2B-1+q_{\pi(u)} \tag{8}$$

gdzie $u\equiv u_2$; zauważmy, że dla jednakowych r_j zachodzi zawsze $u_1=1$. Druga z powyższych nierówności jest konsekwencją faktu, że $k\le 2z(W)-1$ oraz $z(W)\le B$.

Z nierówności $k>z(W)$ wynika, że istnieje przynajmniej jedna podgrupa I_{a2}, której zadania należą do bloku w π. Niech l będzie zadaniem podgrupy I_{a2}, należącym do bloku, i posiadającym największe q_j w tej podgrupie. Z opisu algorytmu wynika, że

$$q_l \ge q_{\pi(u)}. \tag{9}$$

Oznaczmy dalej przez π' permutację, otrzymaną w iteracji, w której zadanie l przeniesione zostało z I'_{a1} do I'_{a2}; w dalszym ciągu, dla π' używać będziemy tych samych oznaczeń co dla π, z tym że opatrzonych symbolem " ' ". Zauważmy, że zgodnie z opisem algorytmu, $k'=z(W')-1$, stąd, w oparciu o te same argumenty, co dla nierówności (8), mamy

$$C_{A3} \le r_{max}+P(W')+z(W')-1+q_{\pi'(u')}. \tag{10}$$

Dobierając odpowiednio zbiór V w nierówności (3) oraz zauważając, że $q(W')=q_1$, otrzymamy następujące dolne ograniczenia na C^*.

$$C^* \ge P(J)+B-1, \tag{11}$$

$$C^* \ge P(W')+z(W')-1+q_1, \tag{12}$$

$$C^* \ge z(W')-1+q_{\pi'(u')}; \tag{13}$$

w przypadku ostatniej nierówności zbiór V składa się z pierwszych zadań podgrup występujących w bloku $(\pi'(1),\dots,\pi'(u'))$ oraz z zadania $\pi'(u')$; zgodnie z opisem algorytmu zadanie $\pi'(u')$ posiada najmniejsze q_j w zbiorze V oraz $z(V)=z(W')$.

Wykorzystując pary nierówności ((9),(11)), ((10),(12)) i ((10),(13)), otrzymujemy

$$C_{A3}-C^* \le r_{max}+B+q_{\pi(u)},$$

$$C_{A3}-C^* \le r_{max}+q_{\pi'(u')}-q_1,$$

$$C_{A3}-C^* \le r_{max}+P(W').$$

Stąd, uwzględniając (9), mamy

$$3C_{A3}-3C^* \le 3r_{max}+P(W')+B-1+1+q_{\pi'(u')}+q_{\pi(u)}-q_1 \le 3r_{max}+2C^*.$$

Rozważmy teraz przypadek, gdy A3 zatrzymuje się z powodu zajścia warunków: $q(W)\ge q_{\pi(u_2)}$ i $k=z(W)$; przypominamy, że $u_2=u$. Niech π będzie permutacją, dla której są one spełnione. Mamy wtedy $C_{A3} \le r_{max}+P(W)+z(W)-1+q_{\pi(u)}$ oraz $C^* \ge P(W)+z(W)-1+q_{\pi(u)}$, i w konsekwencji $C_{A3}-C^* \le r_{max}$. □

Dla najbardziej niekorzystnych wartości r_j oszacowanie z Twierdzenia 2 przyjmuje postać $C_{A3}/C^* \le 8/3$. Pokażemy teraz, że istnieje ciąg problemów konkretnych, dla którego C_{A3}/C^* jest zbieżny do liczby nie mniejszej niż 5/2.

Rozważmy problem, w którym $B=M+2$, $M\ge 1$, każda grupa I_a posiada dwa zadania $(a,1)$ i $(a,2)$, a pozostałe dane są jak niżej. Zakłada się, że $0<\varepsilon<1$.

a	1	2	...	M	M+1	M+2
r_{a1}	0	0		0	0	0
p_{a1}	ε	ε		ε	ε	ε
q_{a1}	$M+M\varepsilon$	$M+M\varepsilon$		$M+M\varepsilon$	$2M+2M\varepsilon$	$2M+2M\varepsilon+\varepsilon$
r_{a2}	0	0		0	0	2M
p_{a2}	ε	ε		ε	$M-1+(M-1)\varepsilon$	ε
q_{a2}	0	0		0	ε	0

Algorytm A3 generuje permutację
((M+2,1),(M+2,2),(M+1,1),(M+1,2),(1,1),(2,1),...(M,1),(M,2),(M-1,2),...
...,(1,2)),

dla której $C_{A3}=5M+3M\varepsilon+\varepsilon$. Natomiast permutacja
((M+2),1),(M+1,1),(1,1),(1,2),(2,1),(2,2),...,(M,1),(M,2),(M+1,2),(M+2,2))
daje wartość funkcji celu $C'=2M+2+3M\varepsilon+2\varepsilon$. Mamy więc

$$C_{A3}/C^{*} \geq C_{A3}/C' = (5M+3M\varepsilon+\varepsilon)/(2M+2+3M\varepsilon+2\varepsilon),$$

gdzie prawa strona może być dowolnie blisko 5/2 dla $M\to\infty$ i $\varepsilon\to 0$.

5. Uwagi końcowe

Przedstawiony algorytm aproksymacyjny A3 dla problemu $1|r_i,q_i|C_{max}$ z jednakowymi, grupowymi przezbrojeniami należy traktować jako pierwsze podejście do konstrukcji dobrego algorytmu. Podstawowym celem tej pracy było otrzymanie procedury, która łatwo poddaje się analizie i równocześnie posiada oszacowanie błędu względnego mniejsze od 3; procedura taka może wskazać ewentualny kierunek dalszych poszukiwań. Chociaż wiele własności Uogólnionej Reguły Jacksona, która okazała się być tak owocna dla zagadnienia bez przezbrojeń, można było przenieść na ogólny algorytm A (materiał dotyczący tego nie jest zawarty w tym opracowaniu), to jednak nie okazały się one tak efektywne, jak w tamtym przypadku. Procedura typu A3, uwzględniająca oryginalne (różne) czasy gotowości, posiada oszacowanie nie mniejsze niż 5/2; ażeby to zobaczyć, wystarczy zastosować A3, bez modyfikacji r_j, do przedstawionego wyżej problemu konkretnego. Przykład algorytmu A3 wskazuje na to, że kluczem do konstrukcji dobrego algorytmu będzie raczej rozbicie grup na odpowiednie podgrupy, niż jakość uszeregowania zadań w grupach czy jakość uszeregowania podgrup.

Otrzymany algorytm daje dobre rezultaty dla zagadnień z małymi czasami gotowości. W przypadku zagadnienia $1|q_i|C_{max}$, posiada oszacowanie 5/3, to samo co algorytm zaproponowany w [12] specjalnie dla tego przypadku; jest jednak od niego prostszy i znacznie łatwiej analizowalny.

LITERATURA

[1] Błażewicz J.: Złożoność obliczeniowa problemów kombinatorycznych. WNT, Warszawa 1988.

[2] Bruno J.,Downey P.: Complexity of Task Sequencing with Deadlines, Set-Up Times and Changeover Costs, SIAM J. Comput. 7(1978), 393-404.

[3] Grabowski J., Nowicki E., Zdrzałka S.: A block approach for single-machine scheduling with release dates and due dates, European J. Oper. Res. 28(1986), 278-285.

[4] Graham R.L. et al.: Optimization and approximation in deterministic sequencing and scheduling: A Survey, Annals of Discrete Mathematics 5(1979), 287-326.

[5] Gupta J.N.D., Darrow W.P.: The two-machine sequence dependent flowshop scheduling problem, European J. Oper. Res. 24(1986), 439-446.

[6] Hall L.A., Shmoys D.B.: Jackson's Rule for Single-Machine Scheduling: Making a Good Heuristic Better, Raport, The Massachusetts Institute of Technology, December 1989.

[7] Lenstra J.K., Rinnooy Kan, A.H.G., Brucker P.: Complexity of machine scheduling problems, Annals of Discrete Mathematics 1(1977), 343-362.

[8] Monma C.L., Potts C.N.: On the complexity of scheduling with batch set-up times, Research Report, University of Southampton, 1987.

[9] Potts C.N.: Analysis of a heuristic for one machine with release dates and delivery times, Oper. Res. 28(1980), 1436-1441.

[10] Schrage L.: Obtaining optimal solutions to resource constrained network scheduling problems. Raport, 1971.

[11] Zdrzałka S.: Problemy szeregowania zadań z przezbrojeniami maszyn, Zeszyty Naukowe Politechniki Śląskiej 970(1988), Automatyka z.94, 373-391.

[12] Zdrzałka S.: Approximation algorithms for single machine sequencing with delivery times and unit batch set-up times, Instytut Cybernetyki Technicznej, Politechnika Wrocławska, Raport Nr 86/88, 1988.(praca przyjęta do druku w European J. Oper. Res., 1989).

[13] Zdrzałka S., Grabowski J.: An algorithm for single machine sequencing with release dates to minimize maximum cost, Discrete Appl. Math. 23 (1989), 73-89.

Recenzent: Doc.dr h.inż.J.Kałuski

Wpłynęło do Redakcji do 1990-04-30.

APPROXIMATION ALGORITHMS FOR SINGLE-MACHINE SEQUENCING PROBLEM WITH UNIFIED BATCH SET-UP TIMES

S u m m a r y

The paper presents results of the worst-case analysis of approximation algorithms for single machine sequencing problem with batch set-up times, in which each job has a release data, a processing time and a delivery time; the problem is to find a sequence of jobs minimizing the delivery time of all the jobs. A review of the approximation algorithms for the problem without set-ups is also given.

АППРОКСИМАЦИОННЫЕ АЛГОРИТМЫ ДЛЯ ОДНОМАШИННОЙ ПРОБЛЕМЫ УПОРЯДОЧЕНИЯ С ОДИНАКОВЫМ ГРУППОВЫМ ВРЕМЕНЕМ

Р е з ю м е

В работе представлены результаты анализа наихудшего случая аппроксимационных алгоритмов для одномашинной проблемы упорядочения задач с одинаковым групповым временем перевооружения, в котором каждая задача имеет определенное время готовности к реализации, время реализации, время доставки и вопрос в том, чтобы найти минимализирующую время очерёдность выполнения, после чего все задания доставляются. Представлен также обзор аппроксимационных алгоритмов для проблемы без перевооружения.