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

Czesław Smutnicki

Politechnika Wrocławska
Instytut Cybernetyki Technicznej

O PEWNEJ KLASIE ALGORYTMÓW APROKSYMACYJNYCH DLA PROBLEMÓW SZEREGOWANIA

Streszczenie. W pracy rozważany jest jednomaszynowy problem szeregowania zadań z kryterium minimalizacji łącznego kosztu wykonywania zadań. Dla tego problemu zaproponowano podejście oparte na aproksymacji funkcji kosztu, prowadzące do konstrukcji nowych algorytmów aproksymacyjnych.

1. Wstęp

W literaturze przedstawiono wiele algorytmów aproksymacyjnych dla jednego z podstawowych problemów szeregowania, jakim jest jednomaszynowy problem szeregowania z kryterium mimimalizacji łącznego kosztu związanego z terminami zakończenia wykonywania zadań. Dla tego problemu znane są w literaturze wielomianowe algorytmy rozwiązywania przy założeniu szczególnych postaci funkcji kosztu, np. liniowe, afiniczne [22] lub afiniczne ze względu na parametry [8]. W ogólnym przypadku jest to problem NP-trudny. W pracy zaproponowano podejście, oparte na aproksymacji ogólnej funkcji kosztu funkcjami afinicznymi ze względu na parametry, prowadzące do konstrukcji algorytmów przybliżonych. W konsekwencji przedstawiono szereg nowych algorytmów aproksymacyjnych dla omawianego problemu, w tym również dla typowych funkcji kosztu. Zaproponowane podejście może być także rozszerzone na inne bardziej złożone problemy szeregowania. Praca była częściowo finansowana przez RP.I.02 "Teoria sterowania i optymalizacji układów dynamicznych i procesów dyskretnych".

Rozważany problem formułuje się następująco: Dany jest zbiór zadań $J=\{1,2,...,n\}$, które należy wykonać na jednej maszynie. Zadanie j jest wykonywane w czasie $p_j>0$ i ma koszt wykonywania równy $f_j(C_j)$, gdzie $f_j(t)$ jest funkcją niemalejącą dla $t\geq p_j$, zaś C_j jest terminem zakończenia wykonywania zadania j, $j\in J$. Zakłada się, że wykonywanie zadania na maszynie nie może być przerywane. Poszukuje się kolejności wykonywania zadań, która minimalizuje sumę kosztów związanych z zakończeniem wykonywania poszczególnych zadań.

Oznaczmy przez π dowolną permutację elementów zbioru J, zaś przez Π zbiór wszystkich takich permutacji. Dalej, przez $F(\pi)$ oznaczmy wartość kosztu dla permutacji π, tzn.

$$F(\pi) = \sum_{j=1}^{n} f_{\pi(j)}(C_{\pi(j)}), \quad C_{\pi(j)} = \sum_{i=1}^{j} p_{\pi(i)}, \ j=1,\dots,n. \qquad (1),(2)$$

Powyższy problem polega na wyznaczeniu permutacji $\pi^{*}\in\Pi$, dla której

$$F(\pi^{*}) = \min_{\pi\in\Pi} F(\pi) \qquad (3)$$

i jest oznaczany zgodnie z notacja Grahama przez $1||\Sigma f_j$. W przypadku liniowych funkcji kosztu zadań $f_j(t)=w_jt$ sformułowany problem staje się znanym problemem $1||\Sigma w_jC_j$, dla którego istnieje algorytm wielomianowy o złożoności $O(n\log n)$ polegający na uporządkowaniu zadań wg nierosnących wartości w_j/p_j [10], [22]. Z postaci kryterium (1) wynika, że algorytm ten rozwiazuje również problem (1)-(3) w przypadku $f_j(t)=w_jt+v_j$, $j\in J$. Jednakże już dla funkcji kosztu postaci $f_j(t)=w_j\max(0,t-d_j)$, gdzie d_j jest żadanym terminem zakończenia (due date), problem staje się silnie NP-trudny [6], [9]. Znane sa przy tym nieliczne przypadki szczególne, dla których udało się skonstruować algorytmy pseudowielomianowe, np. [6], [7]. W przypadku ogólnej funkcji kosztu, jedynym znanym przypadkiem rozwiazywalnym w wielomianowym czasie jest problem, w którym funkcje kosztu są postaci

$$f_j(t)=\int_{t-p_j}^{t} (\alpha_j\varphi(u)+\beta_j)du, \quad j\in J, \qquad (4)$$

gdzie $\varphi(u)$ jest funkcją monotonicznie niemalejącą (nierosnącą) w przedziale $[0,\Sigma_{j=1}^{n}p_j]$, [8], [18], [20]. W tym przypadku optymalną kolejność wykonywania otrzymuje się poprzez uporządkowanie zadań wg nierosnących wartości α_j (niemalejących wartości α_j) [8]. W tym kontekście problemy zawarte w pracach [19], [22] można traktować jako szczególne przypadki problemu $1||\Sigma f_j$ z funkcją kosztu postaci (4). Obszerny przegląd metod i podejść stosowanych do rozwiązywania problemów jednomaszynowych przedstawiono w pracy [4].

W dalszym ciagu pracy będzie używane oznaczenie $p(I)$ na sumę czasów trwania zadań $p(I)=\Sigma_{j\in I}\, p_j$ dla $I\subseteq J$ oraz oznaczenie P na sumę $p(J)$.

2. Algorytmy aproksymacyjne

Zdecydowana większość algorytmów aproksymacyjnych jest formułowana dla problemu z funkcjami kosztu zależnymi od spóźnienia zadań, tzn. $f_j(t)=\max(0,t-d_j)$ (oznaczonego $1||\Sigma T_j$), $f_j(t)=w_j\max(0,t-d_j)$ (oznaczonego $1||\Sigma w_jT_j$) lub z ogólnymi kwadratowymi lub wykładniczymi funkcjami kosztu. Problemy tego typu mają bezpośrednie praktyczne zastosowanie m.in. w elastycznych systemach produkcji [23].

Do konstrukcji algorytmów aproksymacyjnych stosowane sa głównie dwa następujace podejścia:

(a) wyznaczenie rozwiązania poprzez rozwiązanie pewnego problem zastępczego.

(b) poprawienie rozwiązania poprzez lokalne przeszukiwanie zbioru rozwiązań dopuszczalnych.

Realizacja podejścia (a) m.in. sa algorytmy oparte na statycznych lub dynamicznych regułach priorytetowych, np. [1], [2], algorytmy bazujace na prostych wielomianowych problemach szeregowania, np. [11], jak i algorytmy bazujace na budowie permutacji częściowych, np. [14]. Realizacja podejścia (b) są m.in. algorytmy lokalnego przeglądania zbioru rozwiązań dopuszczalnych (otoczeń) dla różnie zdefiniowanych otoczeń, kolejności i technik przeglądania zarówno dla przeglądania deterministycznego, jak i losowego, np. [5], [17]. Dodatkowo szereg podejść stosowanych do budowy algorytmów aproksymacyjnych dla innych zagadnień szeregowania może być zaimplementowanych w przypadku opisanego zagadnienia, np. [14], [17].

Wśród algorytmów najczęściej wymienianych dla zagadnienia $1||\Sigma T_j$ lub $1||\Sigma w_j T_j$ sa m.in. statyczne reguły SPT WSPT, EDD, MWSPT oraz dynamiczne reguły MDD, API. Reguła SPT (shortest processing time) polega na uporządkowaniu zadań wg nierosnacych wartości $1/p_j$ (niemalejących wartości p_j) i korzysta domyślnie z rozwiązania problemu $1||\Sigma C_j$. Reguła WSPT (weighted shortest processing time) polega na uporządkowaniu zadań wg nierosnących wartości w_j/p_j i korzysta domyślnie z rozwiązania problemu $1||\Sigma w_j(C_j-d_j)$. Reguły SPT i WSPT dostarczają rozwiązania optymalne, jeżeli wszystkie zadania sa spóźnione, tzn. $\max(d_j: j\in J) < \min(p_j: j\in J)$. Reguła EDD (earliest due date) polega na uporządkowaniu zadań wg niemalejących wartości d_j i korzysta domyślnie z rozwiązania problemu $1||T_{max}$. Podobnie jak SPT, reguła ta jest dedykowana dla problemu $1||\Sigma T_j$ i dostarcza rozwiazanie optymalne, jeżeli co najwyżej jedno zadanie w uporządkowaniu jest spóźnione, [1]. Reguła MWSPT (modified WSPT) polega na uporządkowaniu zadań wg nierosnacych wartości $(P-d_j)w_j/p_j$ [11]. Z kolei w regułach dynamicznych zakłada się, że w kolejnych krokach algorytmu wybiera się zadanie do uszeregowania kierując się dynamiczną wartością priorytetu. Zatem w każdym kroku algorytmu istnieje zbiór zadań uszeregowanych S (tworzący permutację częściową σ), zbiór zadań nieuszeregowanych J-S oraz zadanie wybrane k, które po uszeregowaniu da kolejną permutację częściową σk. Przykładowo, w regule MDD [2] (modified due date) jako kolejne do uszeregowania wybiera się zadanie k, dla którego wartość $\max(d_k, p(S)+p_k)$ jest najmniejsza. Oczywiście reguła ta jest dedykowana dla problemu $1||\Sigma T_j$. Reguła API (apparent priority index) jako kolejne do uszeregowania wybiera zadanie, dla którego wartość $(w_k/p_k)\exp(-Z\max(d_k-p(S)-p_k,0))$ jest największa, gdzie $Z=\mathrm{card}(J-S)K/p(J-S)$, zaś $K\in[0.5,2.0]$ jest pewnym parametrem [12]. W zakresie algorytmów bazujących na budowie permutacji częściowych możliwe jest stworzenie algorytmu analogicznego do przedstawionego w pracy [14]. W fazie wstępnej tego algorytmu ustala się listę zadań przez zastosowanie pewnej statycznej reguły priorytetowej. W fazie zasadniczej konstruuje się ciąg n permutacji częściowych poczynając od permutacji jednoelementowej i kończąc na permutacji n-elementowej. Kolejna permutacja częściowa jest tworzona na bazie poprzedniej i kolejnego zadania z listy poprzez wstawianie zadania na wszystkie możliwe pozycje w istniejącej permutacji częściowej. Permutacja otrzymana w wyniku fazy zasadniczej jest szukanym rozwiązaniem przybliżonym. Algorytm tego typu w zastosowaniu do problemu $1||\Sigma f_j$ nie był jeszcze badany eksperymentalnie. Podobnie stosunkowo mało jest wyników badań eksperymentalnych dla algorytmów poprawiających rozwiązanie początkowe metodą lokalnego przeszukiwania zbioru rozwiązań dopuszczalnych. Szczegółowe informacje dotyczące budowy otoczeń, sposobów i technik ich przeszukiwania można znaleźć m.in. w [5], [15], [16], [17], [24]. Szczególnie obiecująca przy tym wydaje się metoda, która została zaproponowana w pracy [17] pierwotnie dla problemu $F||C_{max}$. Inne algorytmy aproksymacyjne dla problemów $1||\Sigma w_jT_j$ przedstawiono także w pracach [2], [13], [24].

3. Aproksymacja funkcji kosztu

Ogólna idea budowy proponowanego algorytmu przybliżonego polega na aproksymacji funkcji kosztu $f_j(t)$ każdego zadania funkcją $g_j(t)$ spełniającą warunek (4). W konsekwencji zamiast problemu $1||\Sigma f_j$ rozwiązuje się problem $1||\Sigma g_j$, który dostarcza przybliżonej kolejności wykonywania zadań dla problemu $1||\Sigma f_j$. W wersji statycznej algorytmu przeprowadza się aproksymację wszystkich funkcji kosztu w przedziale $[0,P]$ (lub $[p_j,P]$), a następnie rozwiązuje problem $1||\Sigma g_j$. W wersji dynamicznej przeprowadza się aproksymację funkcji kosztu zadań ze zbioru $J-S$ w przedziale $[p(S),P]$ (lub $[p(S)+p_j,P]$), a następnie wybiera do uszeregowania zadanie o największej wartości α_j otrzymanej z aproksymacji. W każdym z wymienionych przypadków aproksymacja przeprowadzana może być w sensie jednej z typowych norm

$$l_\infty = \max_{a\le t\le b}\ \max_{1\le j\le n} |f_j(t) - g_j(t)| \tag{5}$$

$$l_2 = \int_a^b \sum_{j=1}^{n} (f_j(t) - g_j(t))^2 dt \tag{6}$$

gdzie

$$g_j(t) = \int_{t-p_j}^{t} (\alpha_j p(u) + \beta_j) du, \quad j\in J, \tag{7}$$

zaś $[a,b]$ jest przedziałem aproksymacji. Rozwiązanie problemu aproksymacji wymaga wyznaczenia $\alpha_j,\beta_j,j\in J$ oraz niemalejącej funkcji $\varphi(u)$ minimalizujących odpowiednią normę, i jest w ogólnym przypadku dość kłopotliwe. Wynika to z faktu, że nie istnieją metody efektywnego uzyskiwania aproksymacji jednostajnych (w sensie normy l_∞) z wyjątkiem kilku poszczególnych przypadków, zaś funkcja $f_j(t)$ jest zwykle nieróżniczkowalna, np. $f_j(t)=\max(0,t-d_j)$. Z tego też względu proponuje się rozważać pewne uproszczone problemy aproksymacji w jednej z następujących postaci:

(i) zakładając, że funkcja $\varphi(u)$ jest dana z dokładnością do pewnych (nieznanych) parametrów, należy wynaczyć $\alpha_j,\beta_j,j\in J$ oraz parametry tej funkcji,

(ii) zakładając, że funkcja $\varphi(u)$ jest znana, należy wyznaczyć $\alpha_j,\beta_j,j\in J$,

(iii) zakładając, że funkcja $\varphi(u)$ oraz $\beta_j,j\in J$ są znane, należy wyznaczyć $\alpha_j,j\in J$.

Pewnym ułatwieniem w rozwiązaniu ww. problemów jest fakt, że dla potrzeb algorytmu przybliżonego istotne jest jedynie wyznaczenie wartości współczynników $\alpha_j,j\in J$. Niekiedy możliwe jest uzyskanie rozwiązania problemu aproksymacji w przypadku ogólnym. Tak np. rozwiązując zadanie (ii) dla normy l_2, otrzymujemy z (7)

$$g_j(t)=\alpha_jG_j(t)+\beta_jp_j, \quad G_j(t)=\int_{t-p_j}^{t}\varphi(u)du, \quad j\in J. \qquad (8),(9)$$

Następnie z (6) oraz (8) dostajemy

$$l_2=\int_a^b\sum_{j=1}^n f_j^2(t)dt-2\int_a^b\sum_{j=1}^n f_j(t)[\alpha_jG_j(t)+\beta_jp_j]dt+\int_a^b\sum_{j=1}^n[\alpha_jG_j(t)+\beta_jp_j]^2dt$$

Wyznaczając pochodne $dl_2/d\alpha_j$, $dl_2/d\beta_j$ oraz przyrównując do zera otrzymujemy n układów równań postaci

$$\alpha_j\int_a^b G_j^2(t)dt + \beta_jp_j\int_a^b G_j(t)dt = \int_a^b f_j(t)G_j(t)dt,$$

$$\alpha_jp_j\int_a^b G_j(t)dt + \beta_jp_j^2(b-a) = p_j\int_a^b f_j(t)dt \ ,$$

które następnie dają następujące rozwiązania

$$\alpha_j=\frac{\int_a^b f_j(t)dt\int_a^b G_j(t)dt-(b-a)\int_a^b f_j(t)G_j(t)dt}{[\int_a^b G_j(t)dt]^2-(b-a)\int_a^b G_j^2(t)dt}, \quad j\in J.$$

Przyjmując następnie różne postacie funkcji $f_j(t)$, $j\in J$, $\varphi(u)$ oraz różne przedziały aproksymacji otrzymamy różne nowe reguły priorytetowe.

4. Ocena wyników

W celu oceny wyników przeprowadzono badania wybranej klasy reguł priorytetowych otrzymanych za pomocą opisanego podejścia. Ocenę jakości algorytmów przeprowadzano na drodze eksperymentalnej. Badano wpływ następujących elementów na jakość otrzymywanej reguły priorytetowej:

(a) norma (l_∞, l_2) przyjęta do aproksymacji funkcji kosztu,
(b) uproszczenia problemu aproksymacji (odpowiednio do p.(i)-(iii) z rozdz. 3),
(c) wybór funkcji $\varphi(u)$ w uproszczonym problemie aproksymacji (ii),
(d) wybór przedziału aproksymacji,
(e) typ reguły: statyczna/dynamiczna.

W celu możliwości porównania z innymi istniejącymi algorytmami przybliżonymi przyjęto funkcję kosztu postaci $f_j(t)=w_j\max(0,t-d_j)$, $j\in J$. Przykłady testowe do analizy eksperymentalnej generowano losowo wg schematu powszechnie przyjętego w literaturze [3], [18]. Przykłady te są charakteryzowane przez parametry odpowiednich rozkładów dla w_j, p_j i d_j. Badania przeprowadzono w oddzielnych testach odpowiednio do p. (a)-(e).

A. Przyjęto $\varphi(u)=u$ oraz przedział aproksymacji $[0,P]$ dla reguły statycznej. Zatem zgodnie z (8) mamy $g_j(t)=(\alpha_j p_j)t+(\beta_j p_j-\alpha_j p_j^2/2)$, co odpowiada aproksymacji funkcji ważonego spóźnienia $f_j(t)$ funkcją afiniczną lub liniową (w przypadku gdy $\beta_j=\alpha_j p_j/2$). Stosując odpowiednie normy otrzymano następujące wyniki

(A1) dla normy l_∞: $\alpha_j=\frac{w_j}{p_j}[1-(\frac{d_j}{P})]$,

(A2) dla normy l_2: $\alpha_j=\frac{w_j}{p_j}[1-3(\frac{d_j}{P})^2+2(\frac{d_j}{P})^3]$.

Zauważmy, że wynik (A1) jest identyczny z reguła MWSPT opisana w rozdz. 2, bowiem pomnożenie wszystkich α_j przez P nie ma wpływu na kolejność uporządkowania. Badania eksperymentalne na przykładach generowanych wg schematu z pracy [18] wykazały, że algorytm (A2) jest porównywalny lub nieznacznie lepszy od algorytmu (A1). Z kolei dla przykładów generowanych wg schematu z pracy [8] stwierdzono istnienie dużych podgrup przykładów, charakteryzujących się ustalonymi parametrami rozkładów, dla których algorytm (A1) był znacznie lepszy od (A2). Zatem można przypuszczać, że norma l_∞ jest korzystniejsza z punktu widzenia jakości otrzymywanego algorytmu. Jednakże w wielu praktycznych przypadkach rozwiązanie problemu aproksymacji jednostajnej może okazać się zdecydowanie trudniejsze niż rozwiązanie problemu aproksymacji średniokwadratowej.

B. Przyjęto $\varphi(u)=u$ oraz przedział aproksymacji $[0,P]$ dla reguły statycznej. Zatem $g_j(t)=(\alpha_j p_j)t+(\beta_j p_j-\alpha_j p_j^2/2)$. Pod uwagę wzięto reguły (A1), (A2) oraz następujące dwie reguły otrzymane odpowiednio przy dodatkowym warunku $f_j(0)=g_j(0), j\in J$ (patrz p.(iii)):

(B1) dla normy l_∞: $\alpha_j=\frac{w_j}{p_j}[1-\frac{2d_j}{P+d_j}]$.

(B2) dla normy l_2: $\alpha_j=\frac{w_j}{p_j}[1-1.5(\frac{d_j}{P})+0.5(\frac{d_j}{P})^3]$.

Warunek ten implikuje, że $\beta_j=\alpha_j p_j/2$, zaś funkcje $f_j(t)$ są aproksymowane funkcjami liniowymi $g_j(t)=\alpha_j p_j t$. Dla przykładów generowanych wg [18] nie stwierdzono wyraźnych różnic między (B1) a (A1) oraz między (B2) a (A2). Dla przykładów generowanych wg [8] algorytmy (B1) i (B2) były zdecydowanie lepsze niż (A1) i (A2), chociaż i w tym przypadku obserwowano pewne nieliczne podgrupy przykładów, dla których zależność ta była odwrotna.

C. Przyjęto $\varphi(u)=u^2$ oraz przedział aproksymacji $[0,P]$ odpowiadający regule statycznej. Zatem z (8) mamy $g_j(t)=(\alpha_j p_j)t^2-(\alpha_j p_j^2)t+(\alpha_j p_j^3/3+\beta_j p_j)$, co odpowiada aproksymacji funkcji ważonego spóźnienia $f_j(t)$ funkcją kwadratową. Przyjmując normę l_2 otrzymano następujący rezultat

(C1) $$\alpha_j=\frac{w_j}{p_j}\frac{135}{P}\frac{1+x^4-y-2x^3y-2x^2+3x^2y}{1-1.875y-0.9375y^2},$$

gdzie $x=d_j/P$, $y=p_j/P$. Rezultat ten jest ważny również (z dokładnością do przemnożenia przez stałą) dla przypadku $\varphi(u)=bu^2$, $b>0$. Następnie dokonano porównania eksperymentalnego reguł (A2) i (C1). Otrzymane wyniki wskazują, że aproksymacja funkcji ważonego spóźnienia $f_j(t)$ funkcją kwadratową daje porównywalne lub gorsze wyniki, w stosunku do aproksymacji funkcją afiniczniczną. Podobne wyniki otrzymano przy aproksymacji funkcją wykładniczą $\varphi(u)=e^u$. Można stąd wyciągnąć wniosek, że nie zawsze zwiększanie złożoności reguły prowadzi do poprawy jej jakości. Kolejny wniosek sugeruje, iż należałoby w tym przypadku wybrać inaczej funkcję $\varphi(u)$, np. tak, by uzyskać funkcję $g_j(t)$ jako nieliniową lub odcinkowo-liniową funkcję spóźnienia.

D. Przyjęto $\varphi(u)=u$ oraz przedział aproksymacji $[p_j,P]$ dla reguły statycznej. Zatem podobnie jak w A mamy $g_j(t)=(\alpha_j p_j)t+(\beta_j p_j-\alpha_j p_j^2/2)$. Otrzymane wyniki są w pełni analogiczne:

$$\text{(D1) dla normy } l_\infty:\ \alpha_j = \begin{cases} \dfrac{w_j}{p_j}\left[1-\left(\dfrac{d_j-p_j}{P-p_j}\right)\right] & \text{jeśli } d_j \ge p_j, \\ \dfrac{w_j}{p_j} & \text{przeciwnie,} \end{cases}$$

$$\text{(D2) dla normy } l_2:\ \alpha_j = \begin{cases} \dfrac{w_j}{p_j}\left[1-3\left(\dfrac{d_j-p_j}{P-p_j}\right)^2+2\left(\dfrac{d_j-p_j}{P-p_j}\right)^3\right] & \text{jeśli } d_j \ge p_j, \\ \dfrac{w_j}{p_j} & \text{przeciwnie.} \end{cases}$$

Porównanie eksperymentalne wskazuje na nieznaczna przewagę reguły (D2) nad (A2) oraz na nieznaczącą różnicę w przypadku reguł (A1) i (D1).

E. Przyjeto $\varphi(u)=u$ oraz przedział aproksymacji $[p(S),P]$ dla reguły dynamicznej. Prowadzi to do otrzymania dwóch reguł dynamicznych, odpowiednio do zastosowanych norm aproksymacji

$$\text{(E1) dla normy } l_\infty:\ \alpha_j = \begin{cases} \dfrac{w_j}{p_j}\left[1-\left(\dfrac{d_j-p(S)}{p(J-S)}\right)\right] & \text{jeśli } d_j \ge p(S), \\ \dfrac{w_j}{p_j} & \text{przeciwnie,} \end{cases}$$

$$\text{(E2) dla normy } l_2:\ \alpha_j = \begin{cases} \dfrac{w_j}{p_j}\left[1-3\left(\dfrac{d_j-p(S)}{p(J-S)}\right)^2+2\left(\dfrac{d_j-p(S)}{p(J-S)}\right)^3\right] & \text{jeśli } d_j \ge p(S), \\ \dfrac{w_j}{p_j} & \text{przeciwnie.} \end{cases}$$

Badania eksperymentalne wykazały, że wersje dynamiczne reguł priorytetowych zachowuja sie znacznie lepiej od odpowiednich wersji statycznych. W szczególności reguła (E1) jest lepsza od reguły MWSPT. Podobne wnioski wyciagnięto badajac dynamiczne wersje reguł B1 i B2.

We wszystkich testach przeprowadzano również porównanie badanych algorytmów z dynamiczna reguła API. W odniesieniu do przykładów testowych generowanych wg schematu z pracy [18] reguła API była znacznie gorsza od najlepszej z badanych reguł (tzn. reguły E1 lub dynamicznej wersji reguły B1). W odniesieniu do przykładów generowanych wg schematu z pracy [8] trudno jest dokonać oceny globalnej, bowiem istniały klasy przykładów, charakteryzujace sie ustalonymi parametrami rozkładów, dla których reguła API była lepsza, jak i klasy, dla których API była gorsza.

5. Wnioski końcowe

Szereg nowo zaproponowanych reguł priorytetowych generuje rozwiązania, których jakość zależy m.in. od parametrów rozkładów (średniej, wariancji, korelacji wzajemnej) wartości $w_j, p_j, d_j, j \in J$. Zatem, wykonując odpowiednio obszerne badania eksperymentalne, można dla każdej ustalonej wartości parametrów rozkładu wskazać regułę (spośród wymienionych), która "potencjalnie" wygeneruje rozwiązanie najlepsze. Stąd wynika koncepcja algorytmu przybliżonego, który po statystycznym zanalizowaniu danych problemu dobiera odpowiednią regułę do jego rozwiązania.

Niezależnie od powyższego, wydaje się celowe przeprowadzenie szczegółowych badań w celu rozwiązania odpowiednich problemów aproksymacji zarówno w przypadku ogólnym, jak i w przypadku typowych funkcji kosztu. Z przeprowadzonych eksperymentów obliczeniowych wynika, że należy się skoncentrować na normie l_∞, regułach dynamicznych oraz na odpowiednim doborze funkcji $\varphi(u)$.

LITERATURA

[1] Baker K.R.: Introduction to Sequencing and Scheduling, Wiley, New York, 1974.

[2] Baker K.R.: A dynamic priority rule for scheduling against due dates, TIMES/ORSA Conference, 1981, Houston.

[3] Baker K.R., Martin J.B.: An experimental comparison of solution algorithms for the single-machine tardiness problem, Naval Research Logistic Quarterly 21, 1974, 187-199.

[4] Gupta S.K., Kyparysis J.: Single Machine Scheduling Research, OMEGA International Journal of Mamagement Science 15(3), 1987, 207-227.

[5] Krone M.J., Steiglitz K.: Heuristic programming solution of a flow-shop problem, Operations Research 22, 1974, 629-638.

[6] Lawler E.L.: A pseudopolynomial algorithm for sequencing jobs to minimize total tardiness, Annals of Discrete Mathematics 1, 1977, 331-342.

[7] Lawler E.L., Moore J.M.: A functional equation and its application to resource allocation and sequencing problems, Management Science 16, 1969, 77-84.

[8] Lawler E.L., Sivazlian B.D.: Minimization of Time-Varying Costs in Single-Machine Scheduling, Operations Research 26(4), 1978, 563-569.

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

[10] McNaughton R.: Scheduling with deadlines and loss functions, Management Science 6, 1959, 1-12.

[11] Montagne E.: Sequencing with delay costs. Arizona State University Industrial Engineering Research Bulletin 5, 1969, 20-31.

[12] Morton T.F., Rachamadugu R.M.V.: Mopic heuristics for single machine weighted tardiness problem, Technical Report CMU-RI-TR-83-9, Carnegie-Mellon University, 1982.

[13] Nakamura N., Yoshida T., Hitomi K.: Group production scheduling for

minimum total tardiness. Part I. AIIE Trans. 10, 1978, 152-162.

[14] Nawaz M., Enscore E.E.Jr., Ham I.: "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem", OMEGA International Journal of Management Science 11(7), 1983, 91-95.

[15] Nowicki E., Smutnicki C., Zdrzałka S.: Algorytmy aproksymacyjne w wybranych zagadnieniach kolejnościowych przy kryterium minimalizacji sumy kar, Archiwum Automatyki i Telemechaniki (w druku), 1989.

[16] Nowicki E., Smutnicki C.: Analiza najgorszego przypadku algorytmów aproksymacyjnych dla problemu przepływowego, referat na VII KKA DPP, Kozubnik 1990.

[17] Osman I.H., Potts C.N.: Simulating Annealing for Permutation Flow-Shop Scheduling, Preprint OR17, Faculty of Mathematical Studies, University of Southampton, 1989.

[18] Rinnooy Kan A.H.G., Lageweg B.J., Lenstra J.K.: Minimizing total cost in one-machine scheduling, Operations Research 26, 1975, 908-927.

[19] Rothkopf M.: Scheduling independent tasks on parallel procesors, Management Science 12, 1966, 707-713.

[20] Rothkopf M.H., Smith S.A.: There are no undiscovered priority index sequencing rules for minimizing total delay costs, Operations Research 32, 1984, 451-456.

[21] Schild A., Fredman U.: On scheduling tasks with associated linear loss functions, Management Science 7, 1961, 280-285.

[22] Smith W.E.: Various optimizers for single-stage production, Naval Research Logistics Qartterly, 1956, 59-66.

[23] Smith M.L., Ramesh R., Dudek R.R., Blair E.L.: Characteristics of U.S. Flexible Manufacturing Systems: a survey, in Proceeding of the second ORSA/TIMS conference on flexible manufacturing systems: Operations research models and applications, Amsterdam, Elsievier Science, 1986, 477-486.

[24] Wala K.: Metody lokalnej optymalizacji z zagadnieniu harmonogramowania, Zeszyty Naukowe Pol. Śl., Seria: Automatyka 95, 1988, 179-190.

[25] Wilkerson I.J., Irwin J.D.: An improved algorithm for scheduling independent tasks, AIIE Trans. 3, 1971, 239-245.

Recenzent: Doc.dr h.inż.J.Klamka

Wpłynęło do Redakcji do 1990-04-30.

ON CERTAIN CLASS OF APPROXIMATION ALGORITHMS FOR SCHEDULING PROBLEMS

S u m m a r y

The paper deals with one-machine scheduling problem with minimum total job processing cost criterion. An approach based on cost function approximamtion is proposed. The approach yields new approximation algorithms for the problem.

ОБ ОДНОМ КЛАССЕ АППРОКСИМАТИВНЫХ АЛГОРИТМОВ ДЛЯ ПРОБЛЕМЫ РАСПИСАНИЯ ЗАДАЧ

Р е з ю м е

В статье представлена одномашинная проблема расписания задач с использованием критерия минимизации общих расходов выполнения задач. Для такой проблемы предложен подход, использующий аппроксимацию расходов, позволяющей конструировать новые алгоритмы аппроксимации.