ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1992

Seria: AUTOMATYKA z. 109 Nr kol. 1175

Robert Wójcik

Politechnika Wrocławska

METODY DYNAMICZNEJ ALOKACJI ZASOBÓW W ZADANIACH STEROWANIA ESP

METHODS OF DYNAMIC RESOURCE ALLOCATION IN PROBLEMS OF FMS CONTROL

МЕТОДЫ ДИНАМИЧЕСКОЙ АЛЛОКАЦИИ РЕСУРСОВ В ЗАДАЧАХ УПРАВЛЕНИЯ ГПС

Streszczenie: W pracy rozważany jest problem syntezy procedur sterowania logicznego odpowiedzialnych za koordynację współbieżnie przebiegających procesów technologicznych współzawodniczących o dostęp do wspólnych zasobów ESP. Niektóre z możliwych realizacji konkurencyjnych żądań zasobowych mogą prowadzić do zablokowania systemu. Rozpatrzono problem koordynacji procesów w systemach sterowania scentralizowanego. Dla wybranych klas zadań opisano algorytmy alokacji zasobów gwarantujące bezblokadową realizację procesów. Zamieszczono również algorytmy oryginalne. Porównano efektywność omawianych metod.

Summary: In the paper a problem of logical control programs synthesis for concurrent processes competing with access to the FMS shared resources is considered. Some of the possible realizations of competitive resource requests may lead to the deadlock of the system. The deadlock handling problem in centralized systems is considered. For chosen problems the deadlock-free resource allocation algorithms are described. For some problems original resource allocation methods are presented. Efficiency of the considered methods is compared.

Резюме: В работе рассматривается вопрос синтеза алгоритмов логического управления координирующих выполнение параллельно протекающих технологических процессов, конкурирующих в доступе к совместным ресурсам ГПС. Некоторые из возможных реализаций рассматриваемых процессов могут привести к блокировке системы. Рассматривается вопрос координации процессов в системах централизованного управления. Для заданных классов проблем представлены алгоритмы распределения ресурсов, которые гарантируют безопасную реализацию процессов. Помещены оригинальные алгоритмы. Сравнена эффективность представляемых методов.

1. Wstęp

Wymagania na elastyczność i niezawodność systemów sterowania Elastycznych Systemów Produkcyjnych (ESP) najlepiej spełniają systemy sterowania asynchronicznego czasu rzeczywistego. Pozwalają one na niezależną i równoczesną realizację różnych operacji technologicznych oraz na szybką reakcję na awarie. Bardzo widoczna jest tendencja do decentralizacji systemu sterowania, który coraz częściej składa się z sieci przestrzennie rozproszonych, komunikujących się między sobą komputerów [1,2,3,4].

W pracy rozważany jest problem automatycznej syntezy procedur sterowania logicznego odpowiedzialnych za koordynację współbieżnie przebiegających procesów technologicznych współzawodniczących o dostęp do wspólnych zasobów ESP. Algorytm sterowania musi zapewnić przydział zasobów niezbędnych do wykonania operacji procesowych.

W przypadku istnienia ograniczeń zasobowych liczba dostępnych zasobów systemowych może być mniejsza niż żądania zasobowe zgłaszane przez procesy w danym stanie. Mogą pojawić się konkurencyjne żądania dotyczące tych samych zasobów, tzw. konflikty zasobowe. Oznacza to, że program sterujący przydziałami zasobowymi musi rozstrzygać na bieżąco (ze stanu na stan), które procesy uzyskają dostęp do zasobów, a które zostaną wstrzymane.

W pracy podjęto próbę klasyfikacji zadań alokacji zasobów w systemach współbieżnie

przebiegających procesów sekwencyjnych. Każdy proces wykonuje program, składający się z ciągu operacji. Współbieżność oznacza, że różne operacje w programach mogą być wykonywane równocześnie.

Dla wybranych zadań alokacji przedstawiono algorytmy alokacji zasobów gwarantujące bezblokadową realizację procesów. Rozpatrzono problem koordynacji procesów w systemach sterowania scentralizowanego. Dla pewnych klas zadań podano modyfikacje istniejących już rozwiązań zwiększające wykorzystanie zasobów systemu. Zamieszczono również własne oryginalne rozwiązania wybranych zadań alokacji. Porównano efektywność rozwiązań uzyskiwanych przez różne metody.

W rozdziale drugim przedstawiono sformułowanie problemu alokacji zasobów w ESP. Rozdział trzeci zawiera podstawowe definicje i twierdzenia, wykorzystywane do sformułowania warunków gwarantujących bezblokadową realizację zadań. W rozdziale czwartym przedstawiono klasyfikację systemów współbieżnych z procesami sekwencyjnymi. W rozdziale piątym opisano metody bezblokadowej koordynacji procesów. Rozdział szósty zawiera wnioski końcowe i rozważania dotyczące zastosowań algorytmów w zadaniach sterowania ESP .

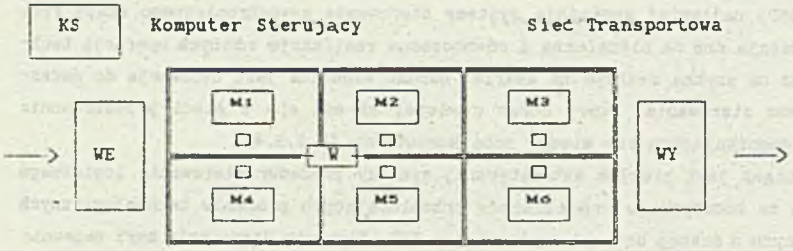
2. Sformułowanie problemu

Rozważamy systemy, w których asynchronicznie realizowane są współbieżne procesy sekwencyjne, korzystające ze wspólnego, skończonego zbioru zasobów. W przypadku ESP zasobami tymi są maszyny, środki transportu, bufory magazynowania palet itp.

Każdy proces realizuje program, tzn. ciąg kolejnych operacji, nazywany dalej marszrutą technologiczną i oznaczany przez MT. Struktura procesu opisana jest za pomocą digrafu operacji [5]. Wykonanie każdej operacji wymaga określonej liczby zasobów. Dla każdej marszruty istnieje zbiór zadań (procesów), które trzeba według niej zrealizować.

Głównym celem rozważań jest synteza algorytmu sterowania, gwarantującego wykonanie wszystkich zadań i zapewniającego maksymalne wykorzystanie dostępnych zasobów systemowych.

W celu ilustracji problemu rozpatrzmy zrobotyzowane gniazdo produkcyjne pokazane na Rys.1.



Rys.1 Zrobotyzowane gniazdo produkcyjne; WE (WY) - magazyn wejściowy (wyjściowy) składowania palet; M_i - numer maszyny; W - wózek; ═ - trasa toru jezdnego

Fig.1 Flexible Machining Cell (FMC); IN (OUT) - input (output) store; M_i - i-th machine; W - robot (car); ═ - transportation route

rozważanym gnieździe realizowane są dwie marszruty technologiczne MT_1 i MT_2, co oznacza, że wytwarzane są dwa typy produktów. Wózek W przewozi palety pomiędzy buforami

maszyn i magazynami. Przyjmuje się, że w danej chwili na wózku może znajdować się tylko jedna paleta. Przyjmuje się również, że maszyny wyposażone są w bufory (magazyny składowania międzyoperacyjnego) o pojemności jednostkowej.

Struktura marszrut technologicznych jest następująca:

$$MT_1 = \langle (D_1,W),(D_2,M_1),(D_3,W),(D_4,M_2),(D_5,W),(D_6,M_3),(D_7,W),(D_8,M_4),$$
$$(D_9,W),(D_{10},M_5),(D_{11},W),(D_{12},M_6),(D_{13},W),(D_{14},M_1),(D_{15},W)\rangle_1$$
$$MT_2 = \langle (D_1,W),(D_2,M_1),(D_3,W),(D_4,M_2),(D_5,W),(D_6,M_1),(D_7,W),(D_8,M_2),$$
$$(D_9,W),(D_{10},M_3)\rangle_2 \quad . \qquad /1/$$

gdzie:

(D_i,W) - oznacza operację transportową realizowaną przez wózek W,

(D_i,M_j) - oznacza operację technologiczną realizowaną przez maszynę M_j.

Należy zauważyć, że odpowiedni system sterowania musi zapewnić, aby paleta była ładowana na wózek tylko wtedy, gdy zostało zarezerwowane miejsce w buforze maszyny docelowej (lub magazynie wyjściowym). Niespełnienie tego warunku może prowadzić do stanu blokady systemu.

Przyjmując, że wymagania rezerwacji buforów maszyn są zawsze spełnione, wystarczy rozwiązywać zadanie alokacji zasobów w systemie zadanym w postaci (1) bez uwzględniania operacji transportowych. Struktura marszrut technologicznych odpowiadająca temu założeniu jest następująca:

$$MT_1 = \langle (D_1,M_1),(D_2,M_2),(D_3,M_3),(D_4,M_4),(D_5,M_5),(D_6,M_6),(D_7,M_1)\rangle_1$$
$$MT_2 = \langle (D_1,M_1),(D_2,M_4),(D_3,M_1),(D_4,M_2),(D_5,M_3)\rangle_2 \quad . \qquad /2/$$

Również i w tym przypadku niektóre z możliwych przydziałów zasobowych mogą prowadzić do blokady systemu. Ma ona miejsce, np. gdy w marszrucie MT_2, po wykonaniu operacji (D_2,M_4), wprowadzimy paletę P_1 do bufora maszyny M_4, a następnie wykonamy operację (D_1,M_1) w tej marszrucie. Przykładowe sformułowanie problemu alokacji zasobów jest następujące: podać kolejność wykonywania operacji technologicznych, gwarantującą bezblokadową realizację zadań.

W związku z powyższym rozwiązanie problemu dynamicznej alokacji zasobów sprowadza się do rozwiązania zadania szeregowania operacji z uwzględnieniem ograniczeń opisujących konflikty zasobowe. Okazuje się, że podanie ograniczeń eliminujących tylko te przydziały zasobowe, które prowadzą do tzw. „stanów niebezpiecznych" (tzn. stanów blokady procesów lub stanów prowadzących do blokady), jest problemem NP-trudnym [6]. Podanie takich ograniczeń jest równoznaczne z określeniem warunków koniecznych i wystarczających bezblokadowej realizacji procesów.

W przypadku ESP poszukujemy algorytmów bezblokadowego sterowania rozdziałem zasobów realizowalnych w czasie rzeczywistym, a więc o wielomianowej złożoności obliczeniowej.

W dalszych rozważaniach przyjmuje się, że każde zadanie alokacji można scharakteryzować za pomocą założeń podstawowych Z_p oraz trzech parametrów:

$$\Gamma \mid E \mid P \quad . \qquad /3/$$

gdzie:

Γ - parametry struktury operacji i zasobów,

E - ograniczenia na sposób realizacji zadań,

P - typ rozwiązania

Rozpatrywany problem dynamicznej alokacji zasobów będzie rozwiązywany przy uwzględnianiu następujących założeń podstawowych $Z_p = \{Z_1,\ldots,Z_6\}$:

Z_1. Procesy korzystają z zasobów nieprzywłaszczalnych (rozpoczęcie wykonywania operacji z wykorzystaniem zasobu nie może być przerwane ani zawieszone) i odnawialnych (po wykorzystaniu zasoby są zwracane do systemu - zerowy bilans zasobowy);

Z_2. Liczba jednostek każdego zasobu r jest stała i wynosi c(r);

Z_3. Jednostka zasobu może być przydzielona do realizacji tylko jednej operacji w danej chwili;

Z_4. Każdy proces polega na wykonaniu ciągu operacji należących do zbioru D i uporządkowanych przez relację częściowego porządku $Q \subset D \times D$ [7] ;

Z_5. Każda operacja $d \in D$ wymaga $i(d,r)$ jednostek zasobu $r \in \mathcal{R}(d)$; gdzie $\mathcal{R}(d) \subset R$, $\mathcal{R}(d) \neq \emptyset$, R - zbiór zasobów systemowych, $\mathcal{R}(d)$ - zbiór zasobów niezbędnych do wykonania operacji d, $0 \leq i(d,r) \leq c(r)$;

Z_6. Proces uwalnia zasoby przydzielone do wykonania operacji po uzyskaniu rezerwacji zasobów niezbędnych do wykonania następnej operacji.

Chcemy skonstruować algorytm bezblokadowego sterowania procesami współbieżnymi, dla których spełnione są założenia Z_p i które dodatkowo opisane są parametrami /3/.

3. Oznaczenia i definicje

Niech $R = \{R_1,\ldots,R_i,\ldots,R_m\}$ - zbiór zasobów, m - liczba zasobów systemu. Wektor $c = (c_1,\ldots,c_i,\ldots,c_m)$, gdzie c_i oznacza liczbę jednostek zasobu R_i. Struktura zadań, realizowanych w oparciu o zasoby R, utworzona przez zbiór operacji $D = \{D_k \mid k=1,\ldots,L\}$, jest opisywana digrafem operacji [5] :

$$V = (A, Q) \quad , \qquad /4/$$

$A \subset D \times H$; $Q \subset A \times A$; $H = 2^R$ - zbiór wszystkich podzbiorów zbioru R .

Według cech strukturalnych wyróżnia się następujące typy procesów:

- procesy liniowe (PL), dla których relacja Q opisująca strukturę zadań jest relacją liniowego porządku [7], np. procesy zautomatyzowanego transportu międzystanowiskowego;
- procesy grupowe (PG), dla których relacja Q jest relacją częściowego porządku, np. procesy montażu mechanicznego lub demontażu form odlewniczych.

W dalszej części przyjmuje się, że struktura procesów realizowanych w systemie ma następującą postać:

$$PS = \{MT_i \mid i=1,\ldots,v\} \quad , \qquad /5/$$

gdzie:

$MT_i = \langle A_1,\ldots,A_k,\ldots A_{l_i}\rangle_i$ - sekwencja operacji realizowana w i-tym programie,

$A_k = (D_k, \mathcal{R}_k)$ - k-ty element programu , $k=1,\ldots,L$, $L = \sum_{i=1}^{v} l_i$,

D_k - numer operacji,

$\mathcal{R}_k = \{R_1^k, \ldots, R_u^k\}$ - zbiór zasobów wykorzystywanych do realizacji operacji D_k;

R_u^k - numer zasobu (k - numer operacji, u - indeks),

$i(D_k,r)$ - liczba jednostek zasobu $r \in \mathcal{R}_k$ wykorzystywana do realizacji operacji D_k, $i(D_k,r) \neq 0$.

$D = \{ D_k \mid k=1,\ldots,L\}$ - zbiór operacji.

$B(i)$ - liczba zleceń (procesów), które należy zrealizować według marszruty MT_i.

$P = \bigcup_{i=1}^{v} B_i = \{ P_j \mid j=1,\ldots,n\}$ - zbiór procesów realizowanych w systemie.

Z kolei wszystkie możliwe sekwencje przydziałów zasobowych opisuje graf poprzedzania zasobów:

$$G = (R, <) \quad , \qquad /6/$$

gdzie:

R - zbiór zasobów systemowych,

$$< \subset R \times R \, , \qquad (\forall R_i, R_j \in R)(\, (R_i, R_j) \in < \iff (\exists i=\overline{1,v})(\exists l=\overline{1, |MT_i|-1})$$
$$(R_i \in crd^2 crd^l MT_i \;\&\; R_j \in crd^2 crd^{l+1} MT_i \;\&\; R_i \neq R_j))$$

$crd^i S \overset{def}{=\!=} s_i \, , \quad S = (s_1,\ldots,s_i,\ldots,s_u)$.

Graf G jest odpowiednikiem grafu zadań zasobowych [8], w którym węzły odpowiadające procesom zastąpiono łukami żądań zasobowych.

Stan systemu opisuje wektor:

$$ST = (A, f) \quad , \qquad /7/$$

gdzie:

$A = (a_1,\ldots,a_i,\ldots,a_n)$ - macierz alokacji zasobowych;

$a_i = (a_{i1},\ldots,a_{ik},\ldots,a_{im})^T$ - wektor zasobów przydzielonych procesowi P_i, a_{ik} - liczba jednostek zasobu R_k przydzielona procesowi P_i, $k=1,\ldots m$;

$f = (f_1,\ldots,f_k,\ldots,f_m)$ - wektor dostępnych zasobów $f \leq c$,

f_k - liczba dostępnych jednostek zasobu R_k.

W rozważanej klasie systemów zbiór wszystkich stanów ZST opisanych /7/ nazywa się przestrzenią stanów. Pojęcie stanu systemu wykorzystamy do opisu wybranych metod alokacji zasobów.

4. Klasyfikacja zadań rozstrzygania konfliktów zasobowych

W literaturze znanych jest wiele podejść do rozwiązania problemu konfliktów zasobowych w systemach z procesami potokowymi [8,9,10,11]. Rozpatrzymy tylko te zadania, dla których spełnione są założenia Z_p. Klasyfikacji zadań dokonamy w oparciu o parametry /3/.

W przypadku parametru Γ wyróżniamy dwie kategorie parametrów $\Gamma = (\Gamma_o, \Gamma_z)$.

Wektor Γ_o opisuje parametry struktury operacji,

$$\Gamma_o = (\gamma_{zp}, \gamma_{uo}, \gamma_{zo}, \gamma_{jo}) \quad . \qquad /8/$$

Przykładowo, parametr γ_{zp} określa, czy procesy realizują ten sam program ($\gamma_{zp} = 0$), czy różne programy ($\gamma_{zp} = 1$). Pozostałe parametry opisują następujące elementy:

γ_{uo} - sposób uporządkowania operacji procesowych (porządek liniowy lub częściowy);

γ_{zo} - liczbę zasobów przypadającą na operację (jeden lub wiele zasobów);

γ_{jo} - liczbę jednostek zasobowych przypadającą na operację (jedna lub więcej) .

Wektor Γ_z opisuje parametry struktury zasobów,

$$\Gamma_z = (\upsilon_{tz}, \upsilon_{pz}) \quad . \qquad /9/$$

Parametr υ_{tz} określa, czy w systemie istnieje jeden rodzaj zasobu (υ_{tz}= 0), czy wiele rodzajów zasobów (υ_{tz}= 1). Parametr υ_{pz} określa, czy pojemności zasobów są większe od 1 (υ_{pz}= 0), czy też istnieje zasób o pojemności równej jeden (υ_{pz}= 1).

Wektor E opisuje ograniczenia determinujące sposób realizacji zadań.

$$E = (e_{pz}, e_{zz}, e_{kz}, e_{gz}, e_{ro}) \quad . \qquad /10/$$

Parametr e_{pz} opisuje strukturę żądań zasobowych, parametr e_{zz} sposób zwalniania zasobów, e_{kz} sposób korzystania z zasobu, e_{gz} globalne żądania zasobowe, i e_{ro} sposób realizacji operacji. Jeżeli e_{pz}= 0, to operacje pobierania zasobów poprzedzają operacje zwalniania zasobów, a jeżeli e_{pz}= 1, to żądania i zwolnienia zasobowe są przemienne. Jeżeli e_{zz}= 0, to wszystkie zasoby przydzielone do wykonania operacji są zwalniane po jej zakończeniu, a jeżeli e_{zz}= 1, to jedynie część zasobów jest zwalniana po zakończeniu operacji. Jeżeli e_{kz}= 0, to w danym programie proces korzysta z zasobu tylko raz, a jeżeli e_{kz}= 1, to proces może korzystać z zasobu wielokrotnie. Jeżeli e_{gz}= 0, to od ustalonego etapu realizacji programów globalne żądania zasobowe procesów są jednakowe, a jeżeli e_{gz}= 1, to żądania etapowe procesów są dowolne. Jeżeli e_{ro}= 0, to każda operacja zaczyna się, jeżeli wszystkie operacje ją poprzedzające są zakończone, a jeżeli e_{ro}= 1, to operacja może zacząć się, gdy część operacji ją poprzedzających jest zrealizowana.

Parametr P określa typ podanych warunków zabezpieczania przed blokadami. Jeżeli p = 0, to podano warunki konieczne i wystarczające (rozwiązanie optymalne), a jeżeli p = 1, to podano warunki wystarczające (rozwiązanie suboptymalne).

Wartości parametrów są tak dobrane, że jedynka oznacza ogólniejsze sformułowanie problemu. Przedstawiona klasyfikacja umożliwi określenie większości spotykanych zadań alokacji zasobów w systemach współbieżnych procesów sekwencyjnych.

5. Metody alokacji zasobów

Znane w literaturze metody bezblokadowej koordynacji procesów współbieżnych dzieli się na trzy grupy metod [8]: wykrywania i likwidacji blokad, zapobiegania blokadom oraz metody unikania blokad. Działanie tych metod zilustrowano dla zadań typu PS /5/.

Metody wykrywania i likwidacji blokad polegają na okresowym sprawdzaniu, czy w systemie nie pojawiła się grupa zablokowanych procesów. W metodach tych konstruuje się graf zadań procesowych G_p, który opisuje żądania zasobowe zgłaszane przez procesy. Wierzchołkami grafu są elementy zbioru P. Łuk (P_i, P_j) oznacza, że proces P_i oczekuje na zwolnienie zasobów przydzielonych procesowi P_j. W celu wykrycia blokady sprawdza się, czy w grafie G_p nie pojawiła się petla. W przypadku wykrycia takiego stanu wykonywany jest odpowiedni algorytm likwidacji. Polega on na odłączaniu zasobów od procesów i badaniu po każdym odłączeniu, czy doprowadziło to do zlikwidowania stanu blokady.

Metody zapobiegania blokadom oparte są na negacji jednego lub kilku warunków koniecznych występowania blokad [8]:

- wzajemnego wykluczania: w danej chwili zasób może być przydzielony tylko jednemu procesowi;
- szeregowania bez wywłaszczania: zasoby nie mogą być odebrane procesowi, który z nich korzysta;

- częściowego przydziału: proces przetrzymuje przydzielone mu zasoby, oczekując na żądane zasoby;
- wzajemnego oczekiwania: istnieje łańcuch procesów, z których każdy przetrzymuje zasoby żądane przez poprzednika w łańcuchu.

W rozważanej klasie systemów nie można zanegować warunku wzajemnego wykluczania, gdyż zasób może być przydzielony tylko do jednego procesu. Również negacja warunku szeregowania bez wywłaszczania nie zawsze może być zrealizowana w praktyce, gdyż wymaga powtarzania operacji przerwanych na skutek odebrania im zasobów.

Negacja warunku częściowego przydziału prowadzi do metod alokacji zasobów opartych na zasadzie „wszystko albo nic". W metodach tych procesowi przydziela się wszystkie zasoby, z których będzie korzystał realizując zadany program. Prowadzi to do prostej strategii sterowania, ale zmniejsza stopień wykorzystania zasobów. Złożoność obliczeniowa algorytmu sterowania przydziałami zasobowymi wynosi O(n), gdzie n = max $|MT_i|$ ($|MT|$ - oznacza długość marszruty).

Metody oparte na negacji warunku wzajemnego oczekiwania pozwalają tylko na takie alokacje zasobów, dla których numery priorytetowe zasobów tworzą ciąg monotoniczny. W przypadku uporządkowanego ciągu zasobów $R_1 < R_2 < \ldots < R_k$ proces, który wykorzystuje zasób R_i, może uzyskać dostęp do zasobu R_j takiego, że $R_i < R_j$. W metodzie tej przewiduje się możliwość zwalniania części zasobów, jeżeli nie można zrealizować przydziałów zgodnie z regułą monotoniczności [8]. Metoda ta może być stosowana w zadaniach alokacji zasobów określonych następująco: $\Gamma_o = (\gamma_{zp}=1, \gamma_{uo}=1, \gamma_{zo}=1, \gamma_{jo}=1)$, $\Gamma_z = (\upsilon_{tz}=1, \upsilon_{pz}=1)$, $E = (e_{pz}=1, e_{zz}=1, e_{kz}=1, e_{gz}=1, e_{ro}=0)$ i pozwala uzyskiwać rozwiązania suboptymalne. Złożoność obliczeniowa algorytmu alokacji zasobów wynosi O(m), gdzie m - liczba zasobów.

W metodach unikania blokad analizuje się aktualny stan systemu i na tej podstawie dokonuje się przydziałów zasobowych. Stosuje się je w systemach, w których warunki konieczne istnienia blokad nie mogą być zanegowane. Metody te wykorzystują różne odmiany algorytmu bankiera [8], w którym testuje się, czy dany stan rozdziału zasobów jest stanem bezpiecznym, tzn. czy gwarantuje zakończenie realizacji wszystkich procesów. Złożoność obliczeniowa odpowiedniego algorytmu alokacji zasobów zależy od złożoności zastosowanego testu bezpieczeństwa stanów.

Inną grupę metod opartą na koncepcji unikania blokad stanowią metody polegające na dekompozycji każdej marszruty MT_i na parami rozłączne strefy zasobów powtarzalnych SR i niepowtarzalnych UR [9,11]. Dla systemów o jednostkowych pojemnościach zasobowych metoda stref oparta jest na następujących regułach alokacji zasobowych:

- zasób znajdujący się w strefie powtarzalnej może być zajęty tylko wtedy, gdy w tej strefie wszystkie zasoby są wolne i w sąsiedniej strefie niepowtarzalnej istnieje wolny zasób;
- zasób w strefie niepowtarzalnej może być zajęty, jeżeli tylko jest dostępny.

Metoda ta może być stosowana w zadaniach alokacji zasobów określonych następująco: $\Gamma_o = (\gamma_{zp}=1, \gamma_{uo}=1, \gamma_{zo}=1, \gamma_{jo}=0)$, $\Gamma_z = (\upsilon_{tz}=1, \upsilon_{pz}=1)$, $E = (e_{pz}=1, e_{zz}=1, e_{kz}=1, e_{gz}=1, e_{ro}=0)$ i pozwala uzyskiwać rozwiązania suboptymalne.
Złożoność obliczeniowa algorytmu alokacji zasobów wynika ze sposobu podziału na strefy i wynosi $O(n^2)$, gdzie n równa się sumie długości marszrut.

Kolejna grupa metod unikania blokad oparta jest na rozpatrywaniu struktury grafu po-

przedzania zasobów G i dynamicznego grafu żądań zasobowych G_d [9]. Graf G_d, opisuje aktualny stan rozdziału zasobów. Wierzchołkami grafu są elementy zbioru $\mathcal{R} \subset R$. Każdy łuk $(\mathcal{R}_i, \mathcal{R}_j)$ określa żądanie zasobowe $\mathcal{R}_j$ pierwszego procesu, któremu przydzielono zbiór zasobów $\mathcal{R}_i$.

W grafie G znajduje się pętlę o minimalnej długości L, a następnie ogranicza się liczbę procesów U realizowanych w systemie tak, aby zawsze zachodził warunek U < L. W ten sposób zapewnia się negację warunku wzajemnego oczekiwania w każdym stanie, a więc i bezpieczność. Omawiane metody mogą być stosowane dla tej samej klasy zadań co metody oparte na negacji warunku wzajemnego czekania. Złożoność obliczeniowa metody wynosi $O(n^2 \log_2 n)$, gdzie $n = \max(w,k)$, w - liczba węzłów, k - liczba kolumn w grafie G i wynika ze złożoności algorytmu znajdowania najkrótszej ścieżki w grafie G [13].

Metoda oparta na wykorzystaniu grafu G_d stosowana jest w systemach $\Gamma_o = (\gamma_{zp}=1, \gamma_{uo}=1, \gamma_{zo}=0, \gamma_{jo}=0)$, $\Gamma_z = (\upsilon_{tz}=1, \upsilon_{pz}=0)$, $E = (e_{pz}=1, e_{zz}=0, e_{kz}=1, e_{gz}=1, e_{ro}=0)$. Są to systemy, w których każda operacja wymaga tylko jednego zasobu R_i, spełniającego warunek $c(R_i) \geq 2$. Metoda ta polega na niedopuszczaniu do powstania pętli w grafie G_d, w której wszystkie zasoby tworzące pętlę nie posiadają ani jednej wolnej jednostki.

Metoda oparta na konstrukcji grafu G_d pozwala uzyskać rozwiązania optymalne, tzn. takie, które odrzucają stany prowadzące do blokady i tylko te stany.

W przypadku istnienia zasobów powtarzalnych SR (tzn. zasobów wykorzystywanych przez kilka operacji) i zasobów niepowtarzalnych UR (tzn. zasobów wykorzystywanych tylko przez jedną z operacji) może być ona stosowana w zadaniach, w których $(\forall r \in SR)(\ c(r) \geq 2\ )$ i $(\forall r \in UR)\ (\ c(r) \geq 1\ )$, a więc $\Gamma_z = (\upsilon_{tz}=1, \upsilon_{pz}=1)$. Złożoność obliczeniowa metody wynosi $O(m^2)$, gdzie m jest liczbą zasobów.

Jako ostatnią omówimy oryginalną metodę, wykorzystującą koncepcję stref i pętli. Metoda może być stosowana dla tych samych klas zadań alokacji zasobów co metoda bankiera. Poniżej opisano wersję metody dla systemów, w których każda operacja wymaga tylko jednego zasobu.

Załóżmy, że zbiór zasobów R podzielono na podzbiór zasobów powtarzalnych SR i podzbiór zasobów niepowtarzalnych UR , a każdą z marszrut MT_i podzielono na strefy powtarzalne i niepowtarzalne (podciągi marszrut złożone z zasobów należących do odpowiedniej klasy). Aby podać warunki dostateczne unikania blokad, wprowadźmy poniższe oznaczenia. Niech

K_N^i - oznacza zbiór zasobów należących do i-tej strefy niepowtarzalnej;

$K = \{K_1, K_2, \ldots, K_l\}$ - zbiór klas zasobowych, gdzie $K_1 = SR$, $K_2 = K_N^1, \ldots, K_l = K_N^r$;

$D(G_1, K_1)$ - długość minimalnej pętli (liczba zasobów tworzących pętlę) w podgrafie $G_1 \subset G$ (G - graf poprzedzania zasobów), składającym się z elementów klasy K_1; gdy graf G_1 jest acykliczny to $D(G_1, K_1) = \|K_1\| + 1$, gdzie $\|K_1\|$ - moc zbioru K_1 ;

$D(G_i, K_i) = \|K_i\| + 1$, $i=2,\ldots,l$, $G_i \subset G$ składa się z elementów klasy K_i ;

DN(G) - długość minimalnej pętli w grafie G, zawierającej co najmniej jeden zasób UR ;

L(ST) - liczba procesów korzystających z zasobów systemu w stanie ST, określonym /7/;

$L(ST, K_i)$ - liczba procesów korzystających z zasobów klasy K_i $(i=1,2,\ldots,l)$ w stanie ST.

Twierdzenie. Jeżeli stan ST spełnia poniższe warunki, to istnieje sekwencja alokacji zasobowych prowadząca do zakończenia wszystkich procesów realizowanych

w systemie w tym stanie (ST jest stanem bezpiecznym).

(i) $L(ST) \leq DN(G)-1$

AND /11/

(ii) $(\forall i=1,\ldots,l)(\ L(ST,K_i) \leq D(G_i,K_i)-1\ )$.

Dowód.

Zauważmy, że każdy stan spełniający warunki /11/ nie jest stanem blokady, co wynika z faktu, że warunek wzajemnego czekania nie jest spełniony. Niech $ZST_1 \subset ZST$ oznacza zbiór wszystkich stanów w przestrzeni stanów ZST, które spełniają warunek /11/. Aby pokazać, że stan $ST_n \in ZST_1$ jest bezpieczny, należy skonstruować ciąg stanów $ST_1,\ldots,ST_k$ prowadzący do stanu początkowego $ST_o=ST_k$. Ponieważ liczba procesów, które mają być zrealizowane w systemie jest skończona $P = \{P_i | i=1,\ldots,n\}$, wystarczy pokazać, że w każdym stanie $ST_n \in ZST_1$ można wykonać alokacje zasobowe prowadzące do stanu $ST_{n+1} \in ZST_1$.

Rozważmy stan $ST_n \in ZST_1$. Mogą zachodzić dwa przypadki:

1. Istnieje proces P_i , dla którego żądany zasób jest dostępny (wolny) i po zrealizowaniu przydziału system osiąga stan $ST_{n+1} \in ZST_1$;
2. Nie istnieje proces, dla którego zachodzi przypadek 1.

Zakładamy, że system osiągnął stan $ST_n \in ZST_1$, w którym realizacja każdego przydziału prowadzi do stanu $ST_{n+1} \notin ZST_1$. Na mocy założeń twierdzenia stan ST_n nie jest stanem blokady, więc istnieje niepusty zbiór procesów PW, dla których żądane zasoby są dostępne. Niech RW będzie zbiorem zasobów wykorzystywanych przez procesy należące do PW. Zauważmy, że każdy proces $P_i \in PW$ korzysta z zasobu $R_i \in UR$ i żąda zasobu $R_n \in SR$, który jest dostępny. Tworzymy drogę X w grafie zorientowanym G_d, której początkiem jest wierzchołek $R_i \in K_1$ [7]. Będąc w węźle R_i grafu G_d losowo wybieramy incydentny z nim węzeł R_k, który jest kolejnym elementem drogi X. Ponieważ w grafie G_d nie ma żadnego cyklu blokadowego, więc po co najwyżej m krokach (m - liczba zasobów) na liście elementów drogi X pojawi się pierwszy element $R_{i+k} \in RW$. Rozpatrzmy drogę $X = \langle R_i,\ldots,R_{i+k-1},R_{i+k}\rangle$ oraz ciąg procesów $Y = \langle P_i,\ldots,P_{i+k-1},P_{i+k}\rangle$ wykorzystujących zasoby tworzące drogę X. Niech R_j będzie zasobem żądanym przez proces P_{i+k}. Ponieważ R_i, $R_j \in K_1$ i zasób R_j jest dostępny, więc po zrealizowaniu przydziałów zasobowych dla procesów P_{i+k}, $P_{i+k-1},\ldots,P_i$ otrzymamy stan ST_{n+1} taki, że $L(ST_n) = L(ST_{n+1})$ i $L(ST_n,K_t) = L(ST_{n+1},K_t)$ $(t=1,\ldots,l)$, co oznacza, że stan $ST_{n+1} \in ZST_1$.

Pokazaliśmy więc, że dla każdego stanu $ST_n \in ZST_1$ można zrealizować ciąg alokacji zasobowych prowadzący do stanu $ST_{n+1} \in ZST_1$. Przy założeniu skończonej liczby procesów i skończonej liczby operacji w marszrutach jest to równoważne z bezpiecznością stanu ST_n. Na mocy zasady indukcji wynika stąd, że każdy stan $ST \in ZST_1$ jest stanem bezpiecznym, gdyż istnieje dla niego ciąg alokacji zasobowych prowadzący do stanu ST_o, co dowodzi tezy twierdzenia. ■

Dla ilustracji rozpatrzmy ponownie system opisany w przykładzie /2/.

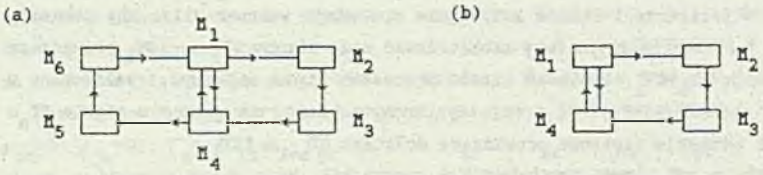
$MT_1 = \langle (D_1,M_1),(D_2,M_2),(D_3,M_3),(D_4,M_4),(D_5,M_5),(D_6,M_6),(D_7,M_1)\rangle_1$,

$MT_2 = \langle (D_1,M_1),(D_2,M_4),(D_3,M_1),(D_4,M_2),(D_5,M_3)\rangle_2$, $c(M_i)=1$, $i=1,\ldots 6$.

System ten można opisać równoważnie:

$MT_1 = \underbrace{\langle M_1M_2M_3M_4\rangle}_{\text{klasa 1}}\ \underbrace{\langle M_5M_6\rangle}_{\text{klasa 2}}\ \underbrace{\langle M_1\rangle}_{\text{klasa 1}}$, $MT_2 = \underbrace{\langle M_1M_4M_1M_2M_3\rangle}_{\text{klasa 1}}$.

Niech P_1, P_2, P_3 procesy realizowane według marszruty MT_1, a P_4, P_5 procesy realizowane według marszruty MT_2. Zbiór procesów $P = \{ P_1,P_2,P_3,P_4,P_5 \}$. Zachodzą związki: $K_1 = SR = \{ M_1,M_2,M_3,M_4 \}$, $K_2 = K_N^1 = \{ M_5,M_6 \}$, $K = \{ K_1, K_2 \}$. Graf poprzedzania zasobów G przedstawiono na Rys.2(a), natomiast podgraf $G_1 \subset G$ ograniczony do zasobów klasy K_1 na Rys.2(b).



Rys.2.(a) Graf G systemu PS /2/; (b) podgraf G_1 grafu G ograniczony do zasobów klasy K_1

Fig.2.(a) Graph G of the system PS /2/; (b) subgraph G_1 of the graph G consisting of the resources of the class K_1

Zauważmy, że $D(G_1,K_1) = 2$, $D(G_2,K_2) = 2 + 1 = 3$, $DN(G) = 4$ (co odpowiada pętli $M_1M_4M_5M_6M_1$) . Zatem warunki twierdzenia przyjmują postać:

$$L(ST) \le 3 \quad \text{AND} \quad L(ST,K_1) \le 1 \quad \text{AND} \quad L(ST,K_2) \le 2 \ . \qquad /12/$$

Rozpatrzmy sposób wykonywania alokacji zasobowych. Przyjmijmy, że żądania zasobowe realizujemy w kolejności $P_1,\ldots,P_4$. Wprowadźmy oznaczenia:

(P_i,M_i) - do procesu P_i przydzielono zasób M_i;

$(P_i,+)$ - do procesu P_i nie przydzielono żądanego zasobu (po wykonaniu przydziału nie są spełnione warunki /12/);

$(P_i,-)$ - do procesu P_i nie przydzielono zasobu (zasób nie jest dostępny).

Sekwencja alokacji zasobowych ma postać :

1. $(P_1,M_1),(P_2,-),(P_3,-),(P_4,-),(P_5,-)$;
2. $(P_1,M_2),(P_2,+),(P_3,+),(P_4,+),(P_5,+)$;
3. $(P_1,M_3),(P_2,+),(P_3,+),(P_4,+),(P_5,+)$;
4. $(P_1,M_4),(P_2,+),(P_3,+),(P_4,+),(P_5,+)$;
5. $(P_1,M_5),(P_2,M_1),(P_3,-),(P_4,-),(P_5,-)$;
6. $(P_1,M_6),(P_2,M_2),(P_3,+),(P_4,+),(P_5,+)$;
7. $(P_1,+),(P_2,M_3),(P_3,+),(P_4,+),(P_5,+)$;
8. $(P_1,+),(P_2,M_4),(P_3,+),(P_4,+),(P_5,+)$;
9. $(P_1,+),(P_2,M_5),(P_3,M_1),(P_4,-),(P_5,-)$;
10. $(P_1,-),(P_2,-),(P_3,M_2),(P_4,+),(P_5,+)$;
11. $(P_1,+),(P_2,-),(P_3,M_3),(P_4,+),(P_5,+)$;
12. $(P_1,+),(P_2,-),(P_3,M_4),(P_4,+),(P_5,+)$;
13. $(P_1,+),(P_2,-),(P_3,-),(P_4,+),(P_5,+)$.

Interpretacja pierwszego wiersza (tzn. zawierającego dwie sekwencje) jest następująca: proces P_1 korzysta z zasobu M_1, procesy P_2,P_3,P_4,P_5 nie uzyskały żądanych zasobów, gdyż nie są one dostępne (wszystkie żądają M_1). W drugiej sekwencji proces P_1 uzyskał dostęp do zasobu M_2, natomiast procesy P_2,P_3,P_4,P_5 nie uzyskały dostępu do zasobu M_1, gdyż prowadzi to do stanu nie spełniającego warunków bezpieczności ($L(ST,K_1) \le 1$).

Zrealizowane alokacje zasobowe doprowadziły do stanu ST, w którym wykonanie dowolnego przydziału zasobowego prowadzi do stanu, w którym nie są spełnione założenia twierdzenia. Należy wyznaczyć drogę X w grafie G_d i ciąg procesów Y w sposób podany w dowodzie twierdzenia. Procesem wykorzystującym zasób powtarzalny M_4 jest proces P_3. W grafie G_d, odpowiadającym stanowi ST, jedyną drogą o początku w M_4 jest $X=\langle M_4,M_5,M_6\rangle$. Ciąg procesów

odpowiadający tej drodze Y= $\langle P_3P_2P_1\rangle$. Po zrealizowaniu przydziałów dla procesów P_1,P_2,P_3 otrzymamy stan $ST' \in ZST_1$. Sekwencja alokacji ma postać: $(P_1,M_1),(P_2,M_6),(P_3,M_5),(P_4,-)$, $(P_5,-)$. Proces P_1 zostaje zakończony. Zmieniając kolejność przydziałów mamy:

15. $(P_4,M_1),(P_2,-),(P_3,-),(P_5,-)$; 16. $(P_4,M_4),(P_2,+),(P_3,-),(P_5,+)$;
17. $(P_4,M_1),(P_2,-),(P_3,-),(P_5,-)$; 18. $(P_4,M_2),(P_2,+),(P_3,-),(P_5,+)$;
19. $(P_4,M_3),(P_2,+),(P_3,-),(P_5,+)$; 20. $(P_2,M_1),(P_3,M_6),(P_5,-)$;
21. $(P_3,M_1),(P_5,-)$; 22. (P_5,M_1); 23. (P_5,M_4);
24. (P_5,M_1); 25. (P_5,M_2); 26. (P_5,M_3).

W ten sposób wszystkie procesy zostają zakończone. Maksymalna liczba równocześnie realizowanych procesów wynosi 3. Zauważmy, że w rozpatrywanym przypadku metoda bankiera [8] pozwala na realizację jednego procesu, a metoda stref [9,11] dwóch procesów.

Złożoność obliczeniowa metody strefowo-pętlowej wynosi $O(n^2)$, gdzie n = max(w,k) (w - liczba wierzchołków w grafie G, k - liczba krawędzi) i wynika ze złożoności algorytmu znajdowania najkrótszej ścieżki w grafie zorientowanym [13].

Przedstawiona metoda może być stosowana dla tej samej klasy systemów procesów sekwencyjnych co metoda bankiera. Jest to metoda heurystyczna, która w przypadku zbioru marszrut /2/ pozwala na równoczesną realizację większej liczby procesów niż pozostałe metody. W oparciu o tę metodę można konstruować algorytmy rozdziału zasobów dla szerszej klasy zadań niż w metodzie stref, czy metodzie grafu żądań zasobowych, a przy tym mogące dawać rozwiązania efektywniejsze.

6. Wnioski końcowe

Metoda stref zasobowych i metoda strefowo-pętlowa są metodami heurystycznymi. Efektywność algorytmów wykorzystujących te metody zależy od struktury programów. Metoda stref może być wykorzystana w systemach, w których do wykonania operacji niezbędnych jest wiele zasobów, ale każdy zasób w ilości jednostkowej. Efektywność metody rośnie wraz ze wzrostem liczby stref niepowtarzalnych. Metoda strefowo-pętlowa może być stosowana w również w systemach, w których do wykonania operacji wykorzystuje się więcej niż jedną jednostkę zasobu. Efektywność metody rośnie, gdy średnia długość pętli istniejących w grafie G wzrasta.

Przedstawione algorytmy znajdują zastosowanie w zadaniach sterowania współbieżnymi procesami w ESP, sieciach transmisji danych oraz komputerach wieloprogramowych. Mogą one być implementowane w procedurach automatycznej syntezy programów sterowania adaptacyjnego [5], które umożliwiają systemowi funkcjonowanie nawet w przypadku uszkodzeń jego komponentów składowych. Przykład podejścia do zagadnienia budowy systemów sterowania adaptacyjnego ESP w oparciu o sieci Petri'ego można znaleźć w literaturze [9,14].

LITERATURA

[1] Ranky P.G.: Computer Integrated Manufacturing. Prentice Hall Inc., U.K. Ltd. 1986.

[2] Automatyzacja dyskretnych procesów przemysłowych (red. H. Kowalowski). WNT, Warszawa, 1984.

[3] Browne J.: Classification of flexible manufacturing systems. The FMS Magazine, April, pp.114-117, 1984.

[4] Bedworth D.D., Henderson M.R., Wolfe P.M.: Computer integrated design and manufacturing. McGraw-Hill, Inc. 1991.

[5] Banaszak Z., Jampolski L.: Komputerowo wspomagane modelowanie elastycznych systemów produkcyjnych. WNT, Warszawa 1991.

[6] Roszkowska E.: Zastosowanie sieci Petri'ego do modelowania i oceny efektywności ESP. Praca doktorska. Instytut Cybernetyki Technicznej Politechniki Wrocławskiej, Wrocław 1991.

[7] Lipski W., Marek W.: Analiza kombinatoryczna. PWN, Warszawa 1986.

[8] Peterson J.L, Silberschatz A.: Operating system concepts. Addison-Wesley Publ. Comp. , Amsterdam 1983.

[9] Modelling and control of FMS: Petri net approach (red. Z. Banaszak). Wrocław Technical University Press, Wrocław 1991.

[10] Gerla M, and Kleinrock L.: Flow control, a comparative study. IEEE Trans. Commun., vol. COM-28, pp.553-574. Apr.1980.

[11] Banaszak Z., Krogh B.: Deadlock avoidance in flexible manufacturing systems with concurrently competing process flows. IEEE Trans. on Robotic and Automation, No.6, pp.724-734, 1990.

[12] Ben-Ari M.: Podstawy programowania współbieżnego. WNT, Warszawa 1989.

[13] Lipski W.: Kombinatoryka dla programistów. WNT, Warszawa 1985.

[14] Banaszak Z.: Coordination of concurrent processes: automatic program synthesis. In: Cybernetics and Systems '86 (red. R. Trappl). Dordrecht, D. Reidel Publ. Co., pp.708-715, 1986.

Recenzent: Prof.dr h.inż. Franciszek Marecki
Wpłynęło do Redakcji do 30.04.1992 r.

Abstract: In the paper a problem of logical control programs synthesis for concurrent processes competing with access to the FMS shared resources is considered. It is assumed that each process follows operations described by production route MT. Each operation requires a set of system resources. There is a batch of processes for each route. When resources are limited there may be competitive processes requests concerning shared system resources (so-called resource conflicts). Some of the possible realizations of competitive resource requests may lead to the deadlock of the system. In order to avoid deadlock one has to solve a problem of task scheduling under resource constraints. Since the problem is NP-complete we are looking for heuristic, effective solutions that can be implemented in real-time control programs. The deadlock handling problem in centralized systems is considered. For chosen problems the deadlock-free resource allocation algorithms are described. For some problems original resource allocation methods are presented. Efficiency of the considered methods is compared. Presented algorithms can be used for automatic synthesis of real-time control procedures for FMSs, transmition networks and multitasking computers.