ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 2000
Seria: AUTOMATYKA z. 129 Nr kol. 1474

Wiesław PIERZCHAŁA
Politechnika Krakowska

UNIKANIE ZASTOJÓW W SKALOWALNYM SYSTEMIE STEROWANIA WYTWARZANIEM

Streszczenie. Przedstawiono metodę zapobiegania zastojom w systemie sterowania wytwarzaniem, w którym stosuje się przetwarzany w czasie rzeczywistym oryginalny model A/E. Sformułowano regułę unikania zastojów oraz warunki dopuszczalności realizacji cykli funkcjonalnych elementów systemu wytwarzania. Stosowny algorytm nie pozwala rozpocząć cyklu, dla którego warunki te nie są spełnione.

DEADLOCK AVOIDANCE IN SCALABLE MANUFACTURING CONTROL SYSTEM

Summary. The paper presents the deadlock avoiding method in the manufacturing control system. In the system the original, processed in real time A/E model is used. The rule for deadlock avoiding has been formulated, together with the permissibility conditions for functional cycles of manufacturing system elements. The appropriate algorithm does not allow to start the cycle, for which the conditions are not fulfilled.

1. Wprowadzenie

Unikanie zastojów jest ważnym i trudnym problemem w symulacji procesów dyskretnych oraz w sterowaniu ich przebiegiem. Praca niniejsza dotyczy komputerowego sterowania systemem wytwarzania na podstawie przetwarzanego w czasie rzeczywistym jego specyficznego modelu posiadającego cechę *skalowalności* (dostosowywania rozmiarów) [1]. Model ten ze względu na jego strukturę nazwano *modelem A/E* (ang. Activities/Elements model).

Zasadniczymi składnikami modelu A/E są arbitralnie wyodrębnione *elementy* systemu modelowanego (maszyny, urządzenia, moduły) oraz *czynności elementarne* przez nie realizowane [2]. W skomplikowanej, zagnieżdżonej strukturze, przyporządkowanej każdej czynności elementarnej, przechowywane są informacje o elementach biorących w niej udział oraz wszystkie wymagane przez model wielkości, z których niektóre podlegają stosownym przekształceniom. Jeżeli w czynności uczestniczy kilka elementów, to każdy z nich musi być przygotowany do jej wykonywania. Ten warunek jest spełniony w modelu wtedy, gdy

wszystkie tzw. *S – wielkości*, we wspomnianej strukturze przypisanej rozpatrywanej czynności, posiadają wartości nieujemne.

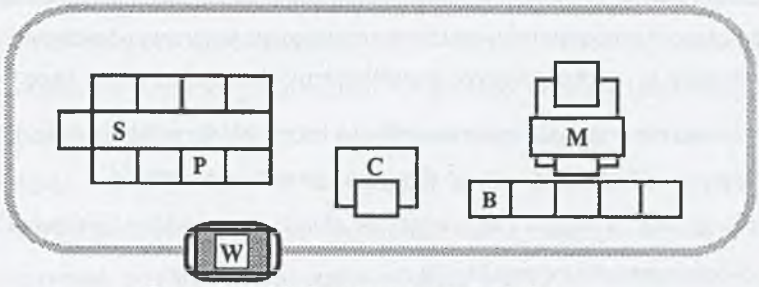
Znamienną cechą modelu A/E jest to, że w każdej chwili specyfikuje on wszystkie czynności możliwe do wykonania (jeśli takie istnieją). Często jednakże wśród czynności możliwych są i takie, które prowadzą do zastoju, a więc do zatrzymania systemu z powodu braku zasobów zdolnych do działania. Czynności takie należy wykryć i nie dopuścić do ich rozpoczęcia. Otrzymany po ich wyeliminowaniu podzbiór *czynności dopuszczalnych* powinien zawierać wyłącznie czynności "bezpieczne" dla systemu w tym znaczeniu, że wykonanie którejkolwiek z nich nigdy nie doprowadzi do jego zablokowania.

Model A/E powstaje w trybie samokreowania na podstawie bazy danych o składzie i strukturze systemu wytwarzania, która w skrócie będzie nazywana *CSDB* (ang. Completion and Structure Database) [2]. Istotne znaczenie ma fakt, że baza ta jest kreowana dla *typów elementów*, a nie dla elementów pojedynczych. Przez *typ elementów* rozumie się wszystkie elementy (maszyny, urządzenia, moduły) pełniące identyczne funkcje (identycznie „zachowujące się") w systemie wytwarzania. Dla każdego typu elementów tworzy się listę *wzorców czynności elementarnych* przez nie realizowanych. *Wzorzec* reprezentuje wszystkie czynności o identycznym charakterze, w których biorą udział elementy tych samych typów. Na przykład, wszystkie czynności obróbki wszystkich przedmiotów jednego typu na jednego typu obrabiarkach są reprezentowane przez jeden wzorzec. Po sporządzeniu list wzorców czynności elementarnych projektant definiuje porządek ich wykonywania, przez co określa sposób działania poszczególnych urządzeń. Polega to na tym, że każdemu ze zdefiniowanych wzorców przyporządkowuje się listę jego następników. W rezultacie baza CSDB opisuje wszystkie elementy systemu wytwarzania, ich funkcje, reguły działania i potencjalne możliwości współpracy. Celem niniejszej pracy jest podanie sposobu zapisu w bazie CSDB, będącej w pewnym sensie modelem uogólnionym, także informacji, które potem, we właściwym modelu, umożliwią algorytmiczne unikanie zastojów.

2. Zastój w dyskretnym systemie wytwarzania

Pojęcie *zastoju* wywodzi się z teorii komputerowych systemów operacyjnych [3]. W celu zilustrowania problemu zastojów w systemach wytwarzania przyjęto prosty przykład systemu, którego schemat przedstawiono na rys.1. W systemie tym wózek (W) dostarcza przedmioty (P) z magazynu (S) do stacji mocowania w uchwytach obróbkowych (C), a

następnie przewozi je do magazynu buforowego (B) przy stacji obróbkowej (M). Po obróbce i zdjęciu z uchwytów obróbkowych gotowe wyroby wózek zwraca do magazynu.



Rys.1. Przykład systemu wytwarzania
Fig.1. An example of manufacturing system

Skład systemu tworzą w tym przypadku:

- zbiór typów elementów

$$E = \{e_1 = \text{"W"}, e_2 = \text{"C"}, e_3 = \text{"B"}, e_4 = \text{"M"}, e_5 = \text{"S"}, e_6 = \text{"P"}\ , \tag{1}$$

- zbiór wzorców czynności

$$A = \left\{ \begin{matrix} a_1 = \text{"P; S » W"}, a_2 = \text{"P; W » C(do zam)"}, a_3 = \text{"P; C * (zam)"}, a_4 = \text{"P; C » W(po zam)"}, \\ a_5 = \text{"P; W » B"}, a_6 = \text{"P; M*"}, a_7 = \text{"P; B » W"}, a_8 = \text{"P; W » C(do odm)"}, \\ a_9 = \text{"P; C * (odm)"}, a_{10} = \text{"P; C » W(po odm)"}, a_{11} = \text{"P; W » S"} \end{matrix} \right\}, \tag{2}$$

gdzie czynności przemieszczania zapisane są według schematu *co;skąd»dokąd(komentarz)*, natomiast czynności obróbki i mocowania – według schematu *co;gdzie*(komentarz)*.

Funkcje i reguły działania elementów e_i opisuje tzw. *e – struktura* ω_i:

$$\omega_i = \left(e_i, A_i, A_i^{beg}, \upsilon_i, O_i\right), \tag{3}$$

gdzie:

- $e_i \in E$ jest nazwą typu elementów;
- $A_i \subseteq A$ jest podzbiorem wzorców czynności, w których uczestniczą elementy typu e_i, np. $A_1=\{a_1, a_2, a_4, a_5, a_7, a_8, a_{10}, a_{11}\}$ (aby zaoszczędzić miejsca mnemoniczne nazwy czynności zastąpiono ich oznaczeniami);
- $A_i^{beg} \subseteq A_i$ jest podzbiorem wzorców reprezentujących czynności, od których element typu e_i może rozpocząć swoje działanie w systemie wytwarzania, np. $A_1^{beg}=\{a_1, a_4, a_7, a_{10}\}$; są to czynności rozpoczynające cykle elementarne elementu, którego działanie jest cykliczne (np. wózka, stacji obróbkowej), lub czynności, w których element uczestniczy po raz pierwszy, w przypadku elementów nie wykazujących cykli działania (np. przedmiotów obrabianych);
- υ_i: $A_i \rightarrow \mathbb{N}$ jest funkcją o wartościach $\upsilon_i(a_k) = \upsilon_{ik}$ określających liczbę elementów typu e_i biorących udział w czynności reprezentowanej przez wzorzec a_k (w przykładzie przyjmuje się, że zawsze $\upsilon_{ik}=1$);

$O_i \subseteq A_i \times A_i$ jest relacją określającą kolejność wykonywania czynności elementarnych; jeżeli dla pewnych $a_k, a_l \in A_i$ para $(a_k, a_l) \in O_i$, oznacza to, że element typu e_i po zakończeniu czynności reprezentowanej przez wzorzec a_k (*poprzednik*) jest gotowy do rozpoczęcia czynności reprezentowanej przez wzorzec a_l (*następnik*).

Zbiór e – *struktur* ω_i reprezentujących wszystkie typy elementów tworzy bazę CSDB.

Kluczową rolę w powstawaniu zastojów, a także, jak się później okaże, w ich unikaniu, odgrywa pojęcie *cyklu funkcjonalnego* elementu systemu wytwarzania.

Jeżeli $a_{(1)}, a_{(2)}, ..., a_{(p)}, ..., a_{(q)}$ należą do zbioru A_i, to *cyklem funkcjonalnym elementu typu* $e_i \in E$ będzie nazywany ciąg $(a_{(1)}, a_{(2)}, ..., a_{(p)}, ..., a_{(q)})$, o ile:

1° $a_{(1)} \in A_i^{beg}$;

2° $(a_{(p)}, a_{(p+1)}) \in O_i$, dla $p = 1, 2, ..., q-1$;

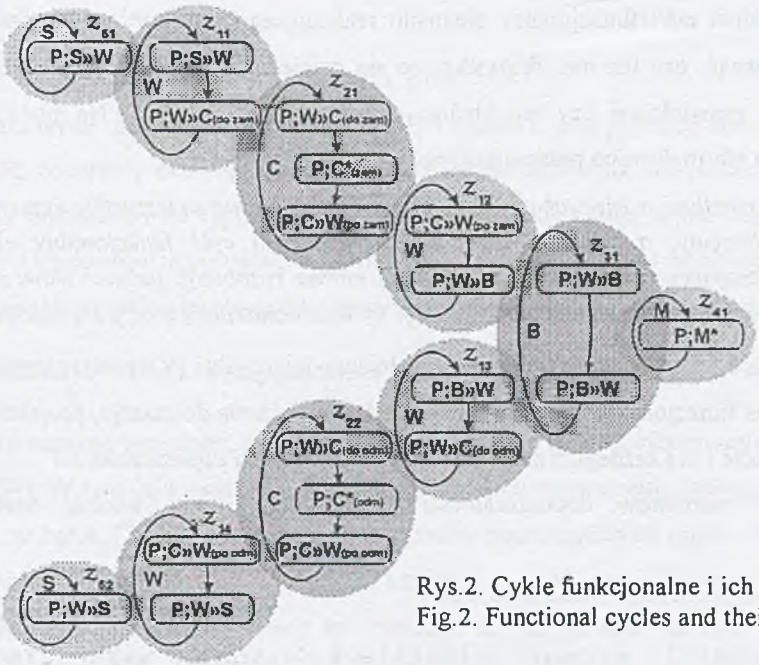
3° $(a_{(q)}, a_{(1)}) \in O_i$

Z przedstawionej definicji wynika, że cyklem funkcjonalnym elementu typu e_i jest każda taka sekwencja czynności wykonywanych w dopuszczalnym porządku, która przywraca temu elementowi stan początkowy, oznaczający jego zdolność do udziału w czynnościach reprezentowanych przez wzorce ze zbioru A_i^{beg}. Numeruje się cykle funkcjonalne liczbami naturalnymi ze zbioru $P = \{1, 2, ...\}$; t – ty cykl elementu typu e_i będzie oznaczany z_{it}.

Do zastoju w systemie dochodzi wtedy, gdy co najmniej jeden cykl funkcjonalny któregoś z elementów nie może się zakończyć. Ten związek wskazuje na istotę mechanizmu powstawania zastojów, a ponadto stanowi podstawę dla proponowanej tutaj metody ich unikania. Realizowany proces produkcyjny wymaga bowiem właściwego współdziałania odpowiednich elementów systemu, co oznacza częściowe powiązanie ich cykli funkcjonalnych poprzez wspólne wykonywanie pewnych czynności (rys.2).

Analizując istniejące powiązania cykli funkcjonalnych można zauważyć potencjalne możliwości wystąpienia zastoju. Na początku marszruty technologicznej przedmiotów typu P, w swych pierwszych cyklach funkcjonalnych z_{51} i z_{11} elementy S oraz W wykonują wspólnie czynność P;S»W. W obu cyklach jest to czynność, która je rozpoczyna. Jeżeli zatem elementy biorące w niej udział, tzn. P, S i W są przygotowane, czynność tę można by rozpocząć. Dla magazynu S jest to czynność kończąca jego cykl z_{51}. Kolejną czynnością w cyklu z_{11} wózka W jest czynność P;W»C(do zam), poprzez którą następuje jego powiązanie z cyklem z_{21} stacji mocowania C. Czynność ta rozpoczyna cykl z_{21} elementu C, co oznacza, że w przypadku gotowości wszystkich uczestniczących w niej elementów może

być wykonana. Jednocześnie jest ona ostatnią czynnością elementu W. Aby zatem cykl ten mógł się zakończyć, niezbędna jest jej gotowość do wykonania elementu C, który do tej pory nie uczestniczył w cyklu wózka. Idąc dalej można zauważyć, że zakończenie cyklu z_{21} stacji C wiąże się z równoczesnym rozpoczęciem cyklu z_{12} wózka W. Warto tutaj zwrócić uwagę na aktywną rolę wózka (w szerszym rozumieniu - wszelkich urządzeń przemieszczających, transportujących, manipulujących) w tym współdziałaniu. Stacja mocowania C oczekuje bowiem na obsługę i dopóki wózek W nie zabierze z niej przedmiotu P, pozostaje ona zablokowana. Gdyby zaś wózek pobrał kolejny półfabrykat z magazynu w celu dostarczenia go do stacji mocowania, powstaje zastój, ponieważ zajęty wózek nie może obsłużyć stacji, a ta z kolei nie może przyjąć kolejnego przedmiotu, uniemożliwiając opróżnienie wózka. Można stąd wysnuć wniosek, że problem unikania zastojów sprowadza się do dbania o to, aby każde używane urządzenie transportowe mogło być możliwie szybko zwolnione.



Rys.2. Cykle funkcjonalne i ich powiązania
Fig.2. Functional cycles and their connections

Analizowany przykład pozwala pokazać kolejną konsekwencję wprowadzenia zbyt dużej liczby przedmiotów obrabianych do systemu wytwarzania. Otóż, warto zwrócić uwagę, że kolejny przedmiot po zamocowaniu jest przewożony do bufora B (cykl z_{12} rozpoczynający się czynnością P;C»W(po zam)). Jeżeli więc wszystkie miejsca w buforze są zajęte, wózek nie powinien tego cyklu rozpoczynać. Jednak zastój wtedy i tak wystąpi, ponieważ stacja mocowania pozostanie zablokowana, uniemożliwiając opróżnienie miejsca w buforze poprzez odwiezienie obrobionego przedmiotu do odmocowania (cykl z_{13}). Reasumując, cykl z_{11}

wózka W może być rozpoczęty (czynność P;S»W) jedynie wówczas, gdy zarówno stacja mocowania C jest wolna, co jest równoznaczne z jej zdolnością do wykonania czynności P;W»C(do zam) rozpoczynającej cykl z_{21}, jak i co najmniej jedno miejsce w buforze B jest puste, co z kolei jest równoznaczne ze zdolnością bufora do udziału w czynności P;W»B, rozpoczynającej jego cykl z_{31}.

Na koniec należy zauważyć, że pobranie obrobionego przedmiotu na wózek, tzn. rozpoczęcie cyklu z_{13} (czynność P;B»W) może nastąpić jedynie wtedy, gdy stacja mocowania jest wolna, bo w przeciwnym razie cykl ten nie mógłby się zakończyć.

3. Reguła i warunki unikania zastojów

Pokazane sytuacje, w których powstaje zastój w przykładowym systemie wytwarzania, prowadzą do wniosku, że zapobiec zastojowi można jedynie w momencie rozstrzygania o tym, czy pewien cykl funkcjonalny elementu realizującego transport wewnątrzsystemowy należy rozpocząć, czy też nie. Wszystko, co się dzieje potem, jest jedynie nieuchronnym następstwem prawidłowej czy też błędnej decyzji w tym zakresie. Na podstawie tego spostrzeżenia sformułowano następującą regułę unikania zastojów:

Czynność możliwą, a więc taką, do wykonania której gotowe są wszystkie uczestniczące w niej elementy, a ponadto taką, która rozpoczyna cykl funkcjonalny elementu przemieszczającego wytwarzane przedmioty, można rozpocząć jedynie wówczas, gdy wszystkie cykle realizowane w konsekwencji jej wykonania będą mogły się zakończyć.

Pozostaje kwestią algorytmizacja wypełniania tej reguły. Ponieważ zazwyczaj tylko niektóre cykle funkcjonalne wnoszą zagrożenie doprowadzenia do zastoju, projektant modelu musi je wskazać i dla każdego z nich podać stosowne *warunki dopuszczalności*

Zestaw warunków dopuszczalności będzie definiowany według następującego schematu:

```
cykl funkcjonalny uwarunkowany
   lista 1:
      warunek 1: element warunkujący <czynność warunkująca>
      warunek 2: element warunkujący <czynność warunkująca>
                 ...
   lista 2:
      warunek 1: element warunkujący <czynność warunkująca>
      warunek 2: element warunkujący <czynność warunkująca>
                 ...
   ...
```

Warunek jest spełniony wtedy, gdy *element warunkujący* jest gotowy do udziału w *czynności warunkującej*. Pomiędzy warunkami z listy zachodzi relacja *alternatywy*, natomiast

pomiędzy listami – relacja *koniunkcji*. Oznacza to, że czynność rozpoczynająca cykl uwarunkowany jest dopuszczalna wtedy, gdy co najmniej jeden warunek z każdej listy jest spełniony.

Wracając do przykładu można stwierdzić, że w celu uniknięcia zastojów w systemie wytwarzania o takiej strukturze należy stosować dwa zestawy warunków dopuszczalności cykli funkcjonalnych wózka:

```
cykl z11:
   P;S»W
   P;W»C(do zam)
      lista 1:
         warunek 1: C <P;W»C(do zam)>
      lista 2:
         warunek 1: B <P;W»B>
cykl z13:
   P;B»W
   P;W»C(do odm)
      lista 1:
         warunek 1: C <P;W»C(do odm)>
```

Warunki te umieszczono w bazie CSDB i stanowią one jej integralną część. Należy podkreślić, że zostały one sformułowane i zapisane, podobnie jak wszystkie składniki bazy CSDB, w odniesieniu do typów elementów systemu wytwarzania, a nie konkretnych jego urządzeń. Uzyskano w ten sposób swego rodzaju uogólnienie metody zapobiegania zastojom, uniezależniając ją od liczby elementów poszczególnych typów w rzeczywistym systemie.

4. Multiplikacja warunków dopuszczalności

Wspomniano wcześniej, że model A/E powstaje w trybie samokreowania na podstawie bazy CSDB. W tym celu należy dla konkretnego systemu wytwarzania, którego struktura jest zapisana w bazie CSDB, podać nazwy elementów poszczególnych typów. W możliwości przypisywania różnej liczby elementów każdemu typowi przejawia się zresztą cecha *skalowalności* modelu A/E. Niech na przykład to przypisanie dla pewnego systemu rzeczywistego wygląda w sposób następujący: dwa wózki W1 i W2 typu W; dwa stanowiska mocowania C1 i C2 typu C; bufor B typu B (zbieżność nazw jest niewykluczona); trzy stacje obróbkowe ASO1, ASO2 i ASO3 typu M; magazyn S typu S; trzy rodzaje przedmiotów obrabianych PDM1, PDM2 i PDM3 typu P. Kompletny model A/E systemu złożonego z wymienionych 12 elementów jest kreowany algorytmicznie na podstawie bazy CSDB. Liczy on 95 czynności wywodzących się z 11 wzorców (wzór (2)). Na przykład, z wzorca P;S»W rozpoczynającego uwarunkowany cykl funkcjonalny z_{11} wózka typu W wywodzi się sześć

czynności rzeczywistych: PDM1;S»W1, PDM1;S»W2, PDM2;S»W1, PDM2;S»W2, PDM3;S»W1, PDM3;S»W2.

Algorytmicznie multiplikowane są także warunki unikania zastojów, które teraz będą przypisane rzeczywistym cyklom elementów. Oto przykład zestawu warunków dla pewnego cyklu wózka W2, wywodzącego się z podanego wcześniej zestawu dla cyklu funkcjonalnego z_{11} (łatwo stwierdzić, że podobnych zestawów powstanie łącznie 12):

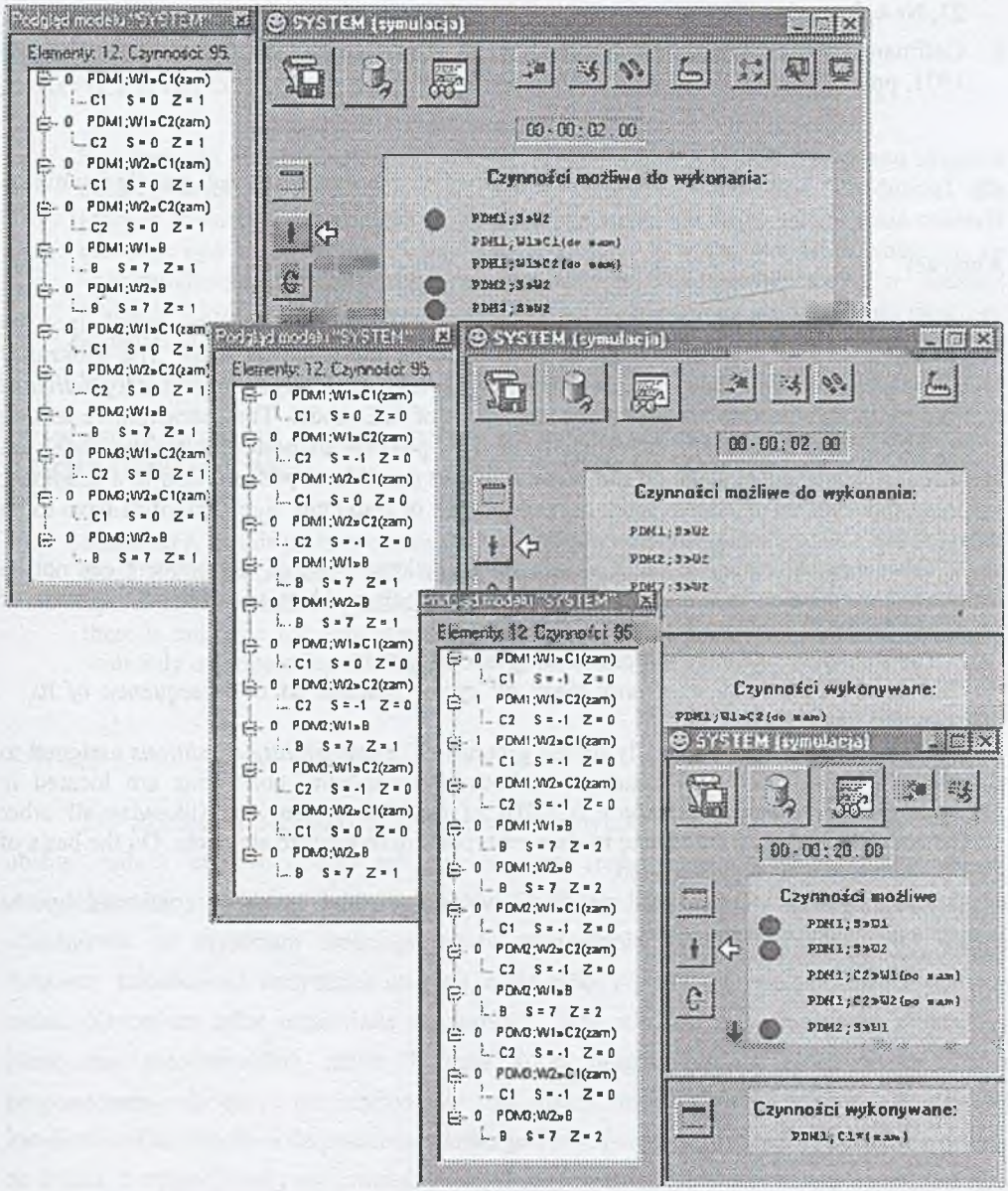
```
cykl:    PDM2;S»W2
         PDM2;W2»C1(do zam)
            lista 1:
               warunek 1: C1 <PDM2;W2»C1(do zam)>
            lista 2:
               warunek 1: B <PDM2;W2»B>
```

5. Przykład unikania zastojów

Algorytm unikania zastojów na podstawie warunków dopuszczalności cykli jest złożony i dość obszerny, toteż nie będzie tutaj przytaczany. W zamian na rys. 3 przedstawiono okna oprogramowania, które go realizuje. Wybrano pewne charakterystyczne momenty początkowej fazy symulowanego procesu wytwarzania, umożliwiające omówienie istoty rzeczy. W pomocniczych oknach podglądowych po lewej stronie pokazane są wszystkie czynności warunkujące cykle wywodzące się z cyklu funkcjonalnego z_{11} wraz z właściwymi elementami warunkującymi oraz tzw. *S – wielkościami* i *Z – wielkościami*. Jeżeli S - wielkość przyjmuje wartość ujemną, oznacza to brak elementów zdolnych do wykonywania danej czynności. Z – wielkość mówi z kolei o tym, ile elementów zostało zawłaszczonych (zarezerwowanych) przez cykl uwarunkowany, który się rozpoczął.

Na rys. 3 w części górnej pokazano sytuację, gdy została wykonana czynność PDM1;S»W1, rozpoczynająca dwa możliwe cykle wózka W1. Pierwszy kończyłaby czynność PDM1;W1»C1(do zam), drugi natomiast czynność PDM1;W1»C2(do zam). Dlatego obie te czynności pojawiły się w oknie czynności możliwych do wykonania. Wózek W1 biorąc z magazynu przedmiot PDM1 "nie wiedział", dokąd go powiezie. Stąd, we wszystkich czynnościach warunkujących dokonał zawłaszczenia odpowiednich elementów (wszędzie Z = 1), rezerwując je sobie na swoje potrzeby. Wskutek tego pozostałe czynności możliwe uzyskują znacznik niedopuszczalności (szare kółko), ponieważ stosowne warunki nie są spełnione (S - Z < 0 dla elementów C1 i C2). Po rozpoczęciu czynności PDM1;W1»C2(do zam) (rys. 3 część środkowa) element C2 staje się niezdolny do działania (S = -1), natomiast wcześniejsze zawłaszczenia zostają zdjęte (Z = 0 dla elementów

C1 i C2). W rezultacie czynności PDM1;S»W2, PDM2;S»W2 oraz PDM1;S»W2 stają się czynnościami dopuszczalnymi. W części dolnej rysunku pokazano sytuację, kiedy stacje mocowania C1 i C2 są zajęte (S = -1, Z = 0), natomiast dwa miejsca w ośmiomiejscowym buforze B są zawłaszczone (S = 7, Z = 2), bo będą tam dostarczone zamocowane przedmioty PDM1. Dlatego pobranie kolejnych przedmiotów z magazynu jest niedopuszczalne.



Rys.3. Przykład unikania zastojów
Fig.3. An example of deadlock avoidance

LITERATURA

1. Pierzchała W.: Otwarty model przepływu produkcji w zautomatyzowanym systemie wytwarzania. Raport końcowy z realizacji projektu badawczego nr 1010/T07/98/15, ITMiAP, Politechnika Krakowska, Kraków 2000.
2. Pierzchała W.: Coloured object-oriented model of manufacturing system. Postępy Technologii Maszyn i Urządzeń, Oficyna Wydawnicza Politechniki Rzeszowskiej, vol. 23, Nr 4, Rzeszów 1999.
3. Coffman E. G., Elphick M. J., Shoshani A.: System Deadlocks. Computing Surveys, 3, 1971, pp. 67-78.

Recenzent: Dr hab.inż. Cz.Smutnicki

Abstract

The paper concerns the computer control of manufacturing system, based upon its specific model. The original *A/E model*, processed in real time, is used. The arbitrarily separated *elements* of modeled system (machines, devices, modules) and elementary *activities* realized by these elements are essential components of A/E model. The characteristic feature of the model is that in every moment all activities possible to perform (if any exists) are specified. It is a frequent situation that some activities possible to perform lead to a deadlock, i.e. to an interruption of system work because of lack of resources. Such activities have to be detected and their realization is not permissible.

The deadlock is reached when at least one functional cycle of any element can not be finished. This relation indicates the essence of deadlock problem and also is the basis of proposed deadlock avoiding method.

The following rule of deadlock avoiding is formulated:

The activity may be started only when all cycles realized as a consequence of its execution can be finished.

This rule is verified algorithmically on the grounds of *permissibility conditions* assigned to functional cycles which may cause a deadlock. Permissibility conditions are located in *Completion and Structure Database* (CSDB) of modeled system and (likewise all other CSDB components) they are related to element types, not to specific elements. On the basis of CSDB the model A/E is self-created.

The efficiency of described deadlock avoiding method has been confirmed by the example presented in the paper.