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

Damian KRENCZYK, Bożena SKOŁUD
Politechnika Śląska

# SAMOSYNCHRONIZACJA SYSTEMÓW ROZPROSZONYCH: ZASTOSOWANIE LOKALNYCH REGUŁ WYBORU PRIORYTETU*

**Streszczenie.** W artykule przedstawiono problem konstrukcji reguł priorytetu w warunkach rozruchu systemu i jego samosynchronizacji. Przedstawiono metodę i algorytm konstrukcji reguł rozstrzygania konfliktów zasobowych w systemach ze sterowaniem rozproszonym. Przedstawione podejście zilustrowano przykładem, a poprawność zweryfikowano wykorzystując system Taylor for Windows.

# SELF SYNCHRONISATION OF DISTRIBUTED CONTROL: LOCAL DISPATCHING RULES ALLOCATION

**Summary.** The problem of dispatching rule construction in the phase of system synchronisation (starting phase) is dealt in the paper. A method and an algorithm of the rule construction in the distributed control systems are presented. Presented approach is illustrated by the example. The simulation confirms the correctness of the presented approach in the system Taylor for Windows.

## 1. Wstęp – logistyczne aspekty planowania produkcji

W ostatnich latach pojawiła się koncepcja zintegrowanego wytwarzania, której celem jest dążenie do integracji logistycznych funkcji przedsiębiorstwa, takich jak np. transport, magazynowanie, harmonogramowanie itd. Integracja obejmuje również problematykę zarządzania operacjami występującymi w dyskretnych procesach. Wspólną cechą procesów jest ich dyskretny charakter oraz związki określające następstwo operacji i związki określające współdziałanie różnych procesów (np. wzajemne wykluczanie się operacji produkcyjnych).

Obok determinowania wielkości serii, czasu pakowania i magazynowania międzyoperacyjnego tworzenie harmonogramu jest jednym z zadań logistyki [7]. W pracach [8,5] pokazano dynamiczny charakter harmonogramowania. Przeanalizowano ponadto konieczność integracji etapu harmonogramowania dynamicznego z etapem statycznym, utożsamianym z planowaniem produkcji przez zastosowanie odpowiednich reguł rozstrzygania konfliktów zasobowych. Złożoność zadań harmonogramowania wymusza potrzebę budowania cząstkowych modeli optymalizacyjnych dla określonych fragmentów

* Pracę wykonano w ramach projektu badawczego KBN nr 8 T11A 020 18.

produkcji. Utworzenie harmonogramu optymalnego w zadowalającym czasie jest na ogół niemożliwe ze względu na NP-trudny charakter tego problemu. Również harmonogramowanie cykliczne należy do tej klasy problemów [1].

Współcześnie obserwuje się dwie tendencje, pierwsza - wytwarzanie produktów w małych partiach, lecz zróżnicowanych asortymentowo; druga - wytwarzanie niewielkiej liczby produktów, lecz w zróżnicowanej wielkości partii. Systemy klasy MRP II, choć bywają stosowane dla obu wymienionych typów produkcji, to w drugim przypadku, czyli dla produkcji seryjnej ze stałym asortymentem są gorzej przystosowane do specyfiki przedsiębiorstwa. Spowodowało to, że w ciągu ostatnich lat wzrosło zainteresowanie implementacją w systemach wytwarzania metod sterowania rozproszonego [6,2]. W opracowaniach naukowych proponowane są liczne koncepcje systemów sterowania produkcją tego typu. Potrzeba taka wynika z faktu, iż w większości małych i średnich przedsiębiorstw przydział zadań do zasobów odbywa się na podstawie lokalnych informacji. Sterowanie rozproszone jest charakterystyczne między innymi dla koncepcji wytwarzania biologicznego i holonicznego, a te z kolei można odnieść do produkcji w przedsiębiorstwie wirtualnym, stanowiącym sieć niezależnych przedsiębiorstw: dostawców, klientów i konkurentów poprzez połączenia sieciowe. Systemy sterowania rozproszonego charakteryzują się decentralizacją podejmowanych decyzji. Istnieje potrzeba opracowania systemów produkcyjnych ze sterowaniem rozproszonym, które potrafiłyby radzić sobie z nagłymi zmianami planu produkcji, zróżnicowanymi potrzebami klientów i automatyczną rekonfiguracją w przypadku awarii. Sterowanie takim systemem nie wymaga centralnego komputera, zasoby wyposażone są w procesory, które określają kolejność wykonywania procesów.

W artykule przedstawiono problem rozruchu systemu ze sterowaniem rozproszonym. Podano warunki wystarczające przydziału reguł rozstrzygania konfliktów zasobowych, umożliwiających samosynchronizację systemu. Przedstawiony problem zilustrowano przykładem i zweryfikowano jego poprawność przez symulację.

## 2. Sformułowanie problemu

Rozpatrywany będzie system wieloasortymentowej produkcji rytmicznej z procesami współbieżnymi. System taki jest **spójny**, jeżeli pomiędzy dowolnymi dwoma zasobami istnieje możliwość przejścia wzdłuż istniejących marszrut lub ich fragmentów.

Założono takie sterowanie rozproszone, bazujące na lokalnych regułach rozstrzygania konfliktów zasobowych, które uwzględnia synchronizującą rolę wąskich gardeł. Reguły wyboru priorytetu spełniają funkcje negocjacyjne, działają lokalnie, określając kolejność dostępu procesów do poszczególnych zasobów. Każdy z procesów operujących na danym zasobie musi wystąpić przynajmniej raz w regule alokowanej na tym zasobie. Lokalna statyczna reguła priorytetu posiada następujący zapis :

$$R_i(p_{i1}, p_{i2}, \ldots, p_{io_i}), \qquad (1)$$

gdzie:

$R_i$ – reguła priorytetu alokowana na zasobie i-tym, i = (1,...,m), gdzie m – liczba zasobów w systemie,

$p_{i1}, p_{i2}, ..., p_{io_i}$ – numery procesów, które zostają przydzielane do zasobu,

$o_i$ – liczba operacji realizowanych w jednym cyklu i-tego zasobu.

Aby zapewnić założony ustalony przebieg procesów, relacje pomiędzy regułami rozstrzygania konfliktów przypisanych do zasobów uwzględniają **warunek bilansu systemu**. Jeżeli dla zadanych reguł rozstrzygania konfliktów zasobowych i skończonych pojemności magazynów, w czasie odpowiadającym stałej wielokrotności okresu systemu, liczba elementów wprowadzonych do systemu odpowiada liczbie elementów z systemu wypływających, to w systemie spełniony jest warunek bilansu [3,4].

Jeżeli w systemie spełniony jest warunek bilansu i jeżeli dowolnej parze $(M_i, M_k)_j$ zasobów sąsiadujących w systemie przydzielono pojemność magazynu międzyoperacyjnego, spełniającą zależność (2), to istnieje taki początkowy przydział procesów do zasobów, dla którego zagwarantowana jest bezblokadowa realizacja procesów [9].

$$Cs_{i,k} \geq n_{i,j} \cdot \chi_i, \tag{2}$$

gdzie:

$Cs_{i,k}$ – dostępna pojemność magazynu międzyoperacyjnego,

$n_{i,j}$ – krotność wystąpienia j-tego procesu w regule przydzielonej do i-tego zasobu,

$\chi_i$ – liczba powtórzeń reguły wyboru priorytetu przydzielonej do i-tego zasobu w okresie powtarzalności systemu.

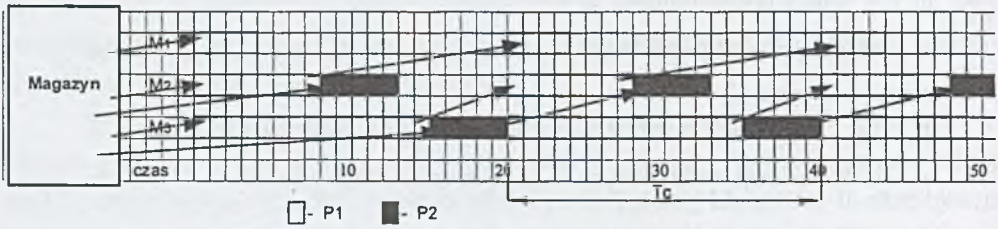
**Rozpatrywane będą następujące problemy:**

**Problem 1.** Jakie są wymagane wstępne zapasy magazynowe, by zapewnić synchronizację systemu zgodnie z przyjętymi regułami rozstrzygania konfliktów zasobowych?

**Problem 2.** Jaka konstrukcja rozruchowych reguł priorytetu (reguł rozruchu) umożliwia takie zapełnienie magazynów bez konieczności przesyłania informacji pomiędzy zasobami systemu?

## 3. Metoda konstrukcji reguł

Reguły rozstrzygania konfliktów (1) przydzielone do zasobów odnoszą się do stanu ustalonego systemu. W trakcie jednego cyklu systemu każda z operacji realizowanych w systemie zostanie wykonana, lecz ich kolejność w wielu przypadkach nie umożliwia zrealizowania całej marszruty procesu w jednym cyklu.




Rys.1. Przykład marszruty realizowanej w kilku okresach T
Fig.1. The example of the route executed in few cycles T

W niektórych przypadkach wykonanie całej marszruty odpowiada liczbie cykli systemu równej liczbie operacji w marszrucie (patrz rys.1). Dlatego też dla każdej pary maszyn sąsiadujących ze względu na j-ty proces $(M_i\ M_k)_j$ w magazynie międzyoperacyjnym powinna znajdować się liczba elementów odpowiadająca ilości realizowanej zgodnie z j-tym procesem w jednym cyklu. Takie wstępne zapełnienie magazynów gwarantuje, że każda z lokalnych reguł w systemie przydzielonych do zasobów może zostać wykonana niezależnie od jej konstrukcji, a to z kolei umożliwia dalsze ich realizowanie.

Założono, że sterowanie w systemie bazuje na regułach rozstrzygania konfliktów zasobowych, co wyklucza komunikowanie się zasobów miedzy sobą. Konieczne jest zatem określenie sposobu uruchamiania systemu (reguł rozruchu) tak, by doprowadzić do jego działania zgodnie z przyjętymi regułami, bez konieczności sterowania centralnego. Do zasobu przydzielona zostanie para reguł: reguła rozruchu, która wykonywana będzie jeden raz oraz reguła rozstrzygania konfliktów, wykonywana cyklicznie.

Konstruowanie reguł rozruchu przebiega zgodnie z przedstawioną procedurą:

1. Dane są lokalne statyczne reguły priorytetu $R_i(p_{i1}, p_{i2}, ..., p_{io_i})$, i=(1,...,m),przydzielone do zasobów systemu.
2. Na każdym z zasobów systemu uszereguj procesy $P_{p_{i1}}, P_{p_{i2}}, ..., P_{p_{io_i}}$ rosnąco według numerów $N_{iw}$, gdzie:
   $N_{iw}$ - numer kolejny operacji procesu $P_{p_{iw}}$ realizowanej na i-tym zasobie.
   Dla i-tego zasobu kolejne wystąpienie procesu oznaczone będzie $P_{p_{iw}}$, gdzie w=1,2,...,$o_i$.
3. Dla ustalonej kolejności wyznacz krotność występowania każdego z procesów w regule rozruchu i-tego zasobu $K_{i1}, K_{i2}, ..., K_{io_i}$ według zależności:

$$K_{iw} = (O_{iw} - N_{iw}) \cdot \chi_i, \tag{3}$$

gdzie:

$O_{iw}$ - liczba operacji procesu $P_{p_{iw}}$ realizowanych w systemie.

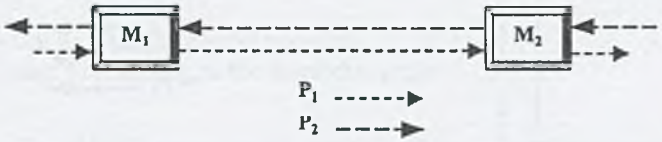
$K_{iw}$ jest to iloczyn ilości operacji pozostałych do wykonania w procesie $P_{p_{iw}}$ po zejściu z i-tego zasobu i liczby powtórzeń reguły priorytetu przydzielonej do i-tego zasobu w okresie powtarzalności systemu.

Zapis lokalnej reguły rozstrzygania konfliktów zasobowych z uwzględnieniem fazy rozruchu przyjmuje postać:

$$R_i\{(K_{i1}\cdot p_{i1}, K_{i2}\cdot p_{i2},\ldots,K_{io_i}\cdot p_{io_i});(p_{i1}, p_{i2},\ldots,p_{io_i})\}. \quad (4)$$

Pierwszy człon w zapisie jest regułą rozruchu, która jest wykonywana jeden raz, drugi człon to reguła dla stanu ustalonego, wykonywana cyklicznie.

Dla zilustrowania metody przydziału reguł do zasobów rozważmy system procesów współbieżnych, przedstawiony na rysunku 2.




Rys.2. Dwuzasobowy system wytwórczy
Fig.2. Two-resources manufacturing system

Dla stanu ustalonego do zasobów przydzielono reguły $R_1(1,1,2)$ i $R_2(1,2,1)$, których liczba powtórzeń w okresie powtarzalności systemu wynosi 1. W regule rozruchu przydzielonej do zasobu $M_1$ kolejność procesów według kroku 2 procedury jest następująca: $P_1$ i $P_2$. Zgodnie z zależnością (3) w regule $R_1$ proces $P_1$ występuje $K_{11}=(2-1)\cdot 1=1$ raz, gdyż w systemie realizowane są dwie operacje tego procesu, a na zasobie $M_1$ realizowana jest pierwsza operacja tego procesu. Proces $P_2$ występuje $K_{12}=(2-2)\cdot 1=0$ razy. Na zasobie $M_2$ kolejność procesów jest następująca: $P_2$, $P_1$. Krotności procesów $P_1$ i $P_2$ w regule $R_2$ wynoszą odpowiednio $K_{21}=(2-2)\cdot 1=0$, $K_{22}=(2-1)\cdot 1=1$. Wyznaczone w ten sposób reguły rozstrzygania konfliktów zasobowych z uwzględnieniem fazy rozruchu dla rozpatrywanego przypadku przyjmują postać:

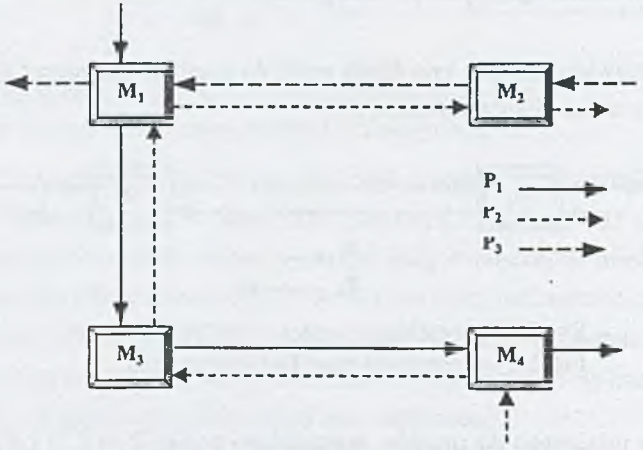
$R_1\{(1, 1);(1,1,2)\}$, $R_2\{(2);(1,2,1)\}$.

**4. Przykład ilustrujący**

Dany jest system produkcyjny, składający się z czterech zasobów: $M_1$, $M_2$, $M_3$, $M_4$. Na realizację w systemie oczekują procesy $P_1$, $P_2$, $P_3$. Marszruta procesu $P_1$ przebiega kolejno przez zasoby $M_1$, $M_3$, $M_4$, procesu $P_2$ przez $M_4$, $M_3$, $M_1$, $M_2$, procesu $P_3$ przez $M_2$, $M_1$, co ilustruje rys.3. Czasy realizacji procesów na zasobach zapisano w tablicach procesów:

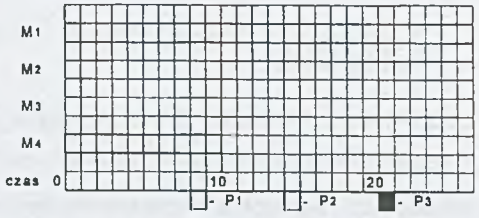
$$M_{P_1}=\begin{bmatrix}1 & 2 & 3\\ 1 & 3 & 4\\ 4 & 2 & 3\end{bmatrix}, M_{P_2}=\begin{bmatrix}1 & 2 & 3 & 4\\ 4 & 3 & 1 & 2\\ 5 & 6 & 5 & 4\end{bmatrix}, M_{P_3}=\begin{bmatrix}1 & 2\\ 2 & 1\\ 4 & 3\end{bmatrix}.$$

Pierwszy wiersz tablicy zawiera numer operacji, drugi wiersz przedstawia marszrutę procesu, a trzeci czasy jednostkowe poszczególnych operacji. Przyjęto początkową alokację statycznych reguł rozstrzygania konfliktów zasobowych gwarantującą bezblokadową pracę systemu [9]: $R_1(3,2,2,1)$, $R_2(2,2,3)$, $R_3(1,2,2)$, $R_4(2,2,1)$, których liczba powtórzeń w okresie powtarzalności systemu wynosi 1.




Rys.3. System współbieżnych procesów produkcyjnych
Fig.3. System of concurrent processes (production rates)

Rozpoczęcie pracy systemu przy założonych regułach statycznych, lecz bez uwzględnienia fazy rozruchu, spowoduje wystąpienie cyklu wzajemnych oczekiwań, co jest równoznaczne z powstaniem blokady. Zasób $M_4$ wykona zgodnie z regułą dwa razy proces $P_2$ i wyśle elementy tego procesu do magazynu międzyoperacyjnego. Od tego momentu zasób $M_4$ oczekuje na proces $P_1$. Zasób $M_3$ oczekuje na proces $P_1$. Zaś zasoby $M_1$ i $M_2$ oczekują odpowiednio na proces $P_3$ i $P_2$.




Rys.4. Wykres Gantta. Powstanie blokady w systemie
Fig.4. Gantt's chart. Deadlock appearance in the system

Aby uniknąć blokad, dla przyjętych lokalnych statycznych reguł wyboru priorytetu skonstruowano reguły z uwzględnieniem fazy rozruchu według procedury z rozdziału 3:

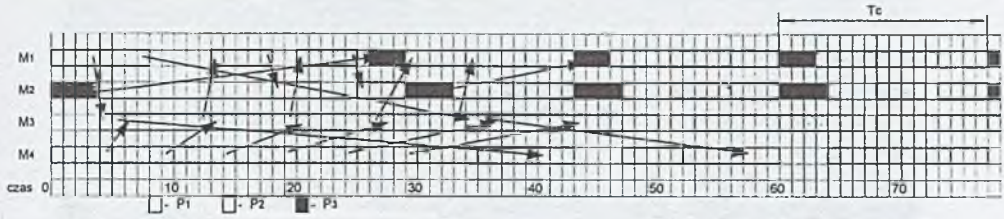
$R_1\{(1,1,2,2);(3,2,2,1)\}$,

$R_2\{(3);(2,2,3)\}$,

$R_3\{(1,2,2,2,2);(1,2,2)\}$,

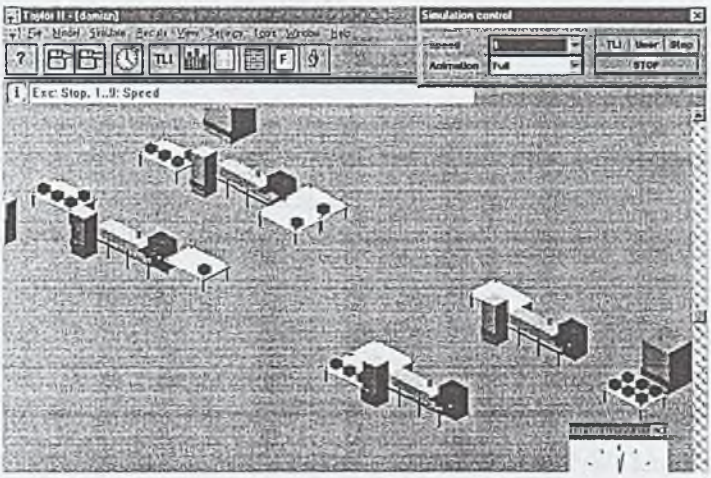
$R_4\{(2,2,2,2,2,2);(2,2,1)\}$.

Praca systemu, do którego alokowano reguły z uwzględnieniem fazy rozruchu, pokazana jest na rysunku 5. Wstępne zapełnienie magazynów międzyoperacyjnych przez wykonanie reguły rozruchu umożliwia pracę bez blokad. Cykl systemu ustala się zgodnie z pracą zasobu krytycznego i wynosi 17 jednostek czasu.




Rys.5. Wykres Gantta. Działanie reguł rozruchu
Fig.5. Gantt's chart. Functioning of the dispatching rules

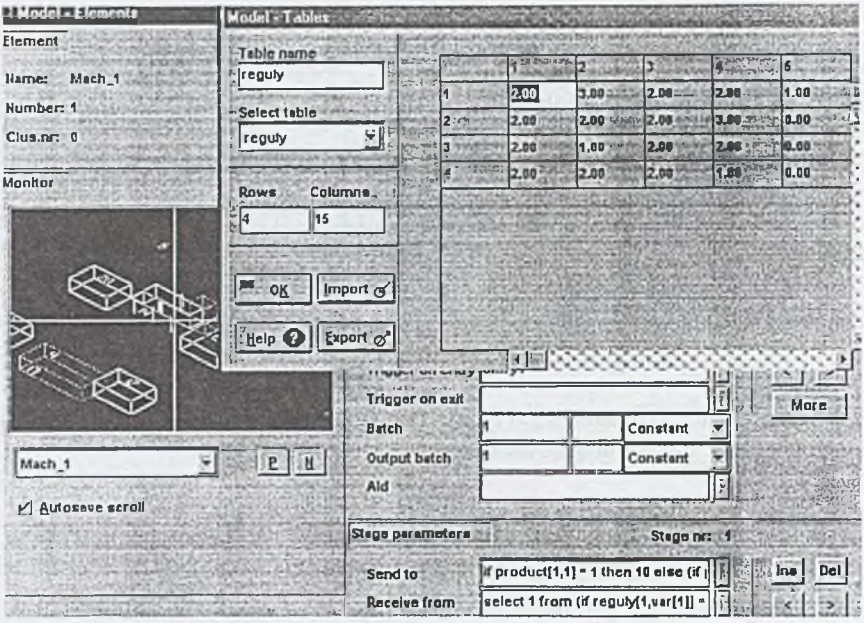
W celu weryfikacji poprawności przedstawionego podejścia przedstawiony system produkcyjny zamodelowano w programie symulacyjnym Taylor for Windows [10]. Procedury sterowania rozproszonego wprowadzono w taki sposób, aby sterowanie odbywało się na podstawie podanych w postaci tablic reguł rozstrzygania konfliktów zasobowych. Model systemu przedstawiono na rysunku 6.



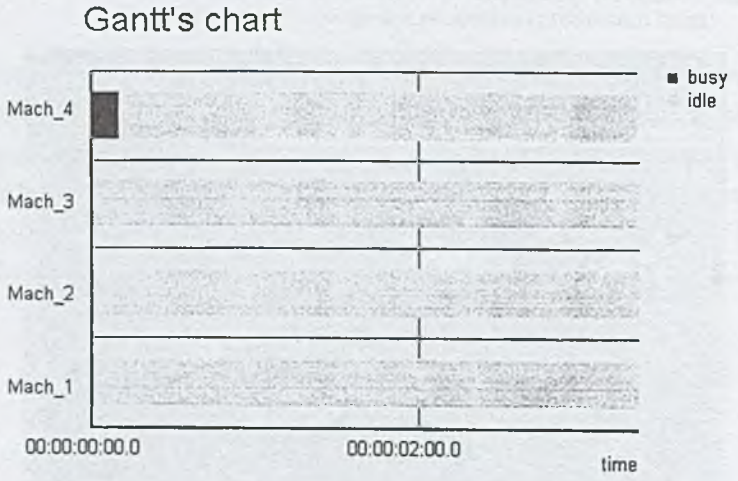

Rys.6. System współbieżnych procesów produkcyjnych (Taylor for Windows)
Fig. 6. Model of the system of concurrent processes (Taylor for Windows)

Podobnie jak w poprzednim przypadku przeprowadzono dwa eksperymenty.

W pierwszym przypadku do zamodelowanego systemu wpisano reguły rozstrzygania konfliktów zasobowych bez uwzględnienia fazy rozruchu (rys.7.). Dla tych warunków przeprowadzono symulację pracy systemu. Wynik symulacji pokazuje rysunek 8.




Rys.7. Zapis reguł
Fig.7. The rule definition

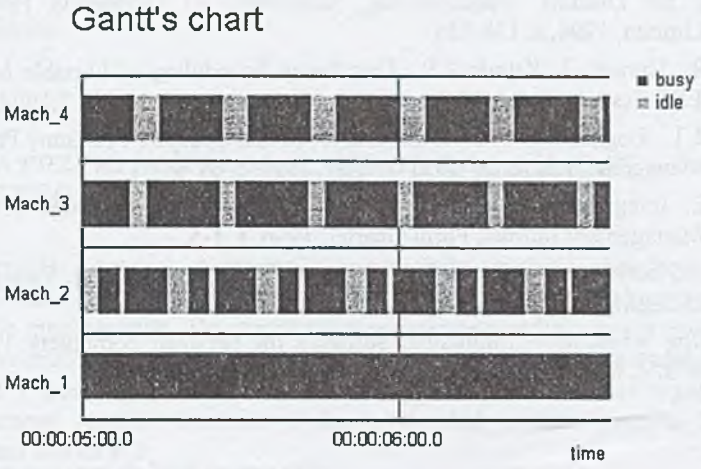



Rys.8. Wykres Gantta. Powstanie blokady w systemie (por. rys. 4)
Fig. 8. Gantt's chart. Deadlock of the system (add.fig.4 )

Na podstawie przeprowadzonej symulacji można stwierdzić, że w przypadku alokacji reguł bez uwzględnienia fazy rozruchu, po wykonaniu dwóch elementów procesu $P_2$ na zasobie $M_4$ w systemie powstaje stan blokady ( analogicznie do sytuacji pokazanej na rysunku 4).

W drugim przypadku przedstawiono pracę systemu w przypadku wpisania w programie symulacyjnym reguł z uwzględnieniem fazy rozruchu systemu (rysunek 9). System

synchronizuje się do cyklu pracy zgodnego z cyklem pracy zasobu krytycznego, który w tym przypadku wynosi 17 jednostek czasu (porównaj z rys.5).




Rys.9. Wykres Gantta. Działanie reguł rozruchu (por.rys.5)
Fig.9. Gantt's chart representation of the priority rule functioning (add. fig.5 )

**5. Podsumowanie i kierunek dalszych badań**

W artykule przedstawiono sposób synchronizacji systemu ze sterowaniem rozproszonym polegający na przydziale do każdego z zasobów systemu pary reguł: reguły rozruchu wykonywanej jeden raz i reguły statycznej realizowanej cyklicznie. Przedstawiono metodę konstruowania reguły rozruchu w zależności od przyjętej reguły stałej. Podejście zilustrowano przykładem, a jego poprawność zweryfikowano symulacyjnie, korzystając z pakietu Taylor for Windows. Problem samosynchronizacji systemu w sytuacji wystąpienia niepożądanych, wcześniej nie przewidywanych zakłóceń, przewidziano jako kierunek dalszych badań.

LITERATURA

1. Camus H., Ohl W., Korbaa O., Gentina FJ-C.: Cyclic Schedules in Flexible Manufacturing Systems with Flexibilities in Operating Sequences. Proceedings of the 17th International Conference on Application and Theory of Petrii Nets, Manufacturing and Petrii Nets, Osaka , Japan, 1996, s.97-116.
2. Cyklis J., Zając J.: Koncepcja rozproszonego systemu sterowania dyskretnymi systemami wytwarzania, planowanie przebiegu zleceń w systemach produkcji rytmicznej. Automation'98, Warszawa, 1998, s. 47-52.
3. Kłos S., Majdzik P., Banaszak Z.: Algebraic Verification of the Concurrent Systems Operation of Sequential Processes. Proceedings of the 8th European Simulation Symp. & Exhibition, vol. 2, Genoa, Italy,1996, s. 245-251.

4. Kłos S., Skołud B., Gattner D.: Terminowość realizacji zleceń w systemie współbieżnych procesów produkcyjnych. II KK Komputerowo zintegrowane zarządzanie, Zakopane, 1998, s. 161-168.
5. Lazaro J., Maseda J., Diaz F., Sturesson H., Escalada G.,:INTESIMPO Simple Dynamic Scheduling for Discrete Manufacturing, Scheduling of Production Processes. Ellis Horwood Limited, 1994, s. 130-138.
6. Perkins J.R., Humes C., Kumar P.R.: Distributed Scheduling of Flexible Manufacturing Systems. IEEE Transactions on Robotics and Automation, vol.10, no. 2,1994, s. 133-141.
7. Puttman M.T.: Logistics in Process Industries: Is It a Specyfic Problem? Production and Inventory Management Journal, Third Quarter, 1991, s. 61-66.
8. Selegna G.: Integrated the Planning and Scheduling in a Job Shop. Production and Inventory Management Journal, Forth Quarter, 1996, s. 1-6.
9. Skołud B.: Planowanie wieloasortymentowej produkcji rytmicznej. Zeszyty Naukowe Politechniki Śląskiej, seria Mechanika, z.136, Gliwice, 2000.
10. Taylor II for Windows – Simulation software for personal computers 1996 by F&H Simulations B.V. Maliebaan 88, 3518 CX Utrecht, The Nedherlands.

Recenzent: Prof. dr hab. inż: M.Zaborowski

**Abstract**

Nowadays, two tendencies are observed in industry: the first one manufacturing products in small bathes, but varying in assortment and the second one - manufacturing of small variety but in different sizes of bathes. Although MRP II systems are used for both types of manufacturing, but in the second case they are worse adopted for the characteristics of an industrial plant. It causes that during last years, the interest for implementing the methods of distributed control has been increased.

The distributed system using the dispatching rule should be synchronised in the beginning of its functioning to assure the cyclic behaviour. The procedure of the creating the dispatching rules consists of three steps: dispatching rule construction, the scheduling of the processes on each resource and the starting rule definition.

The illustrative example is presented and the simulation adaptation in the system Taylor for Windows confirms the correctness of the presented approach. As a further research the problem of synchronisation of the system in case of disturbance is foreseen.