NOWE ROZWIĄZANIA W MODELOWANIU PRODUKCJI PARTIAMI Z ZASTOSOWANIEM CZASOWYCH KOLOROWANYCH SIECI PETRIEGO

Andrzej BOŻEK

Streszczenie: W pracy przedstawiono dwie nowe koncepcje dotyczące modelowania produkcji partiami z wykorzystaniem czasowych kolorowanych sieci Petriego. Pierwsza z nich polega na zbiorczej reprezentacji grup partii przetwarzanych w sposób regularny, to znaczy na jednej maszynie i w równych odstępach czasu, za pomocą pojedynczych tokenów. Druga koncepcja dotyczy sposobu generowania harmonogramów pseudoaktywnych przez wykorzystanie dwuetapowej symulacji modelu sieciowego. Obydwa rozwiązania są wdrażane w rozwijanym module harmonogramowania produkcji partiami, w celu poprawy wydajności i jakości jego działania.

Słowa kluczowe: produkcja partiami, czasowe kolorowane sieci Petriego, flexible job shop, CPN Tools.

1. Wprowadzenie

Prezentowana praca powiązana jest z projektem aplikacji harmonogramowania automatycznego dla fabryki śrub i dotyczy algorytmu modelowania systemu produkcyjnego, przeznaczonego dla procedury harmonogramującej. System produkcyjny przetwarza określoną liczbę zadań, z których każde składa się z predefiniowanego ciągu operacji. Liczba i kolejność operacji dla poszczególnych zadań mogą być w ogólnym przypadku odmienne. Dla każdego typu operacji istnieje zbiór alternatywnych maszyn właściwych do jej wykonania, przy czym ich parametry, takie jak szybkość przetwarzania czy dostępność, są zróżnicowane. Zgodnie z podaną charakterystyką, strukturę produkcyjną należy zakwalifikować jako elastyczny system gniazdowy (*flexible job shop*, FJS) [1]. Specyfikacja systemu zawiera dodatkowe elementy, najważniejszymi z nich są:

1. Czasy przezbrojeń: Każdej trójce (maszyna, operacja poprzedzająca, operacja kolejna) można przyporządkować dowolny czasu przezbrojenia.
2. Kalendarze dostępności maszyn: Dla każdej z maszyn można zdefiniować dowolny zbiór przedziałów czasowych, w których maszyna jest dostępna.
3. Przetwarzanie partiami: Operacje nie są ciągłymi przedziałami pracy maszyn, jak przyjmuje się w klasycznej definicji FJS, lecz dzielą się na partie produkcyjne. Gdy maszyna rozpocznie przetwarzanie partii produkcyjnych pewnej operacji, musi przetworzyć wszystkie partie tej operacji, zanim rozpocznie przetwarzanie partii innej operacji. Organizacja produkcji ma charakter równoległy, gdy tylko partia produkcyjna zostanie przetworzona na jednej maszynie, zostaje przekazana do maszyny kolejnej. Przetwarzanie na poziomie partii produkcyjnej nie może być przerwane, to znaczy jest niewywłaszczalne, ale zbiór partii składających się na operację nie musi być przetwarzany w sposób ciągły. Pomiędzy kolejnymi partiami

produkcyjnymi mogą powstawać przerwy czasowe, spowodowane różnicami w prędkościach pracy maszyn oraz okresami niedostępności maszyn.

Dla systemów wytwarzania typu gniazdowego, w tym FJS, w algorytmach harmonogramowania używa się zwykle reprezentacji w postaci grafu dysjunktywnego, bądź permutacji z powtórzeniami [1,2]. Jednakże, przy rozbudowanej specyfikacji systemu, wiele z jego elementów nie może zostać ujętych w tych reprezentacjach i wymaga oddzielnego modelowania. Bardziej uniwersalne możliwości daje między innymi formalizm sieci Petriego, pozwalający na łatwe modelowanie współbieżności, synchronizacji i współdzielenia zasobów, który jest stosowany zwłaszcza w odniesieniu do systemów produkcyjnych o złożonej specyfikacji lub niejednorodnej strukturze [3].

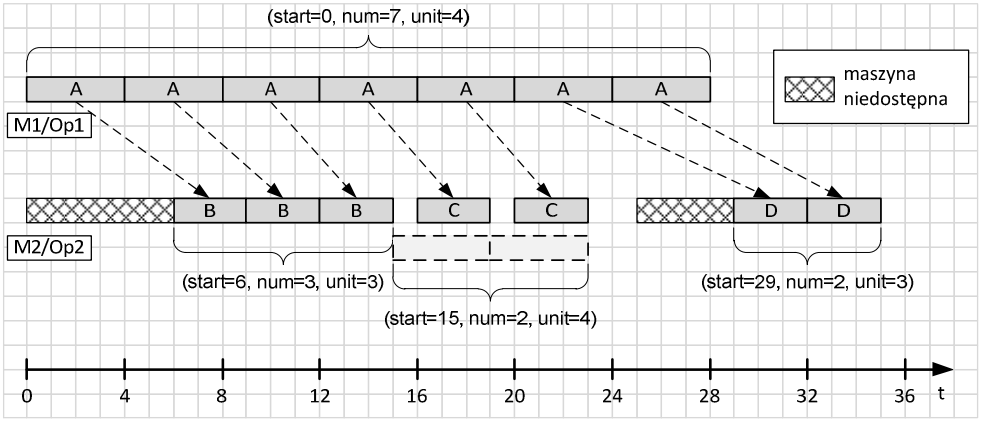
W rozważanym projekcie wykorzystywany jest formalizm hierarchicznych czasowych kolorowanych sieci Petriego [4]. Dotychczasowe rezultaty przedstawiono w pracach [5,6]. Rozwój algorytmu przebiega w dwufazowych etapach. W pierwszej fazie nowa koncepcja jest testowana symulacyjnie na podstawie modeli utworzonych w narzędziu CPN Tools [7]. W drugiej fazie przetestowane składniki dodawane są do właściwej aplikacji modułu harmonogramującego, która składa się z trzech zasadniczych elementów: maszyny symulacyjnej sieci Petriego, generatora tworzącego strukturę sieciową dla zadanej instancji problemu harmonogramowania oraz modułu optymalizacji stochastycznej.

W niniejszej pracy przedstawione zostały dwie nowe koncepcje, które zostały pomyślnie zweryfikowane w fazie symulacyjnej i są implementowane w fazie wdrożeniowej. Nowe rozwiązania nie wprowadzają dodatkowych funkcjonalności, ale mają poprawić jakość działania modułu. Koncepcja grupowania partii produkcyjnych powoduje wydatne przyspieszenie pracy algorytmu. Koncepcja symulacji dwuetapowej umożliwia generowanie uszeregowań pseudoaktywnych, stanowiących korzystniejszą do przeszukiwania podprzestrzeń harmonogramów, niż podprzestrzenie uzyskiwane wcześniej.

2. Grupowanie partii produkcyjnych

W przypadku wymienionych wcześniej prac [5,6], obróbka każdej z partii produkcyjnych modelowana jest poprzez jeden krok symulacji sieci, to znaczy odpowiada jej wykonanie pojedynczej tranzycji. Konsekwencją takiego sposobu modelowania jest co najmniej liniowa złożoność czasowa algorytmu symulacyjnego względem liczby partii produkcyjnych, na jaką podzielono operacje. Wynika to z proporcjonalnej zależności pomiędzy liczbą partii produkcyjnych i liczbą kroków symulacji oraz z faktu, że koszt czasowy wykonania jednego kroku może wzrastać wraz liczbą partii produkcyjnych.

Opracowano nowy sposób modelowania przepływu partii produkcyjnych z wykorzystaniem czasowych kolorowanych sieci Petriego, który uniezależnia czas obliczeń od liczby partii produkcyjnych. Proponowana metoda opiera się na spostrzeżeniu, iż w ciągach partii produkcyjnych, przetwarzanych na poszczególnych maszynach, da się wydzielić następujące po sobie grupy partii przetwarzanych w sposób regularny, to znaczy opuszczających maszynę w równych odstępach czasu. Wszystkie partie produkcyjne wchodzące w skład jednej grupy można scharakteryzować zestawem kilku liczb i obliczenia realizować na poziomie grup, zamiast operować na pojedynczych partiach. Na rysunku 1 zamieszczono przykład wyjaśniający koncepcję grupowania partii produkcyjnych. Siedem partii produkcyjnych przetwarzanych jest w dwóch kolejnych operacjach, Op1 i Op2, najpierw przez maszynę M1, a następnie przez maszynę M2.



Rys. 1. Wydzielanie grup w ciągach partii produkcyjnych

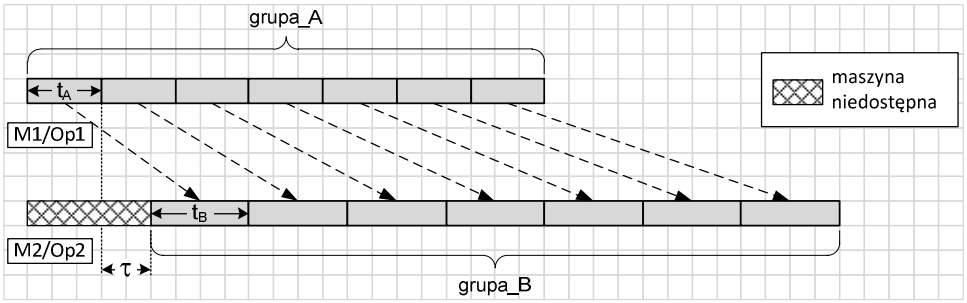
Czas przetwarzania jednej partii na maszynie M1 wynosi 4, a na maszynie M2 wynosi 3. Ponieważ wszystkie partie produkcyjne przetwarzane przez maszynę M1 tworzą regularny ciąg, zostały one przydzielone do jednej grupy (grupa A). W przypadku maszyny M2 partie utworzyły trzy grupy (B, C, D). Każda z grup zdefiniowana jest przez trójkę liczb:

- moment rozpoczęcia przetwarzania (*start*),
- liczba partii produkcyjnych tworzących grupę (*num*),
- czas przetwarzania pojedynczej partii produkcyjnej (*unit*).

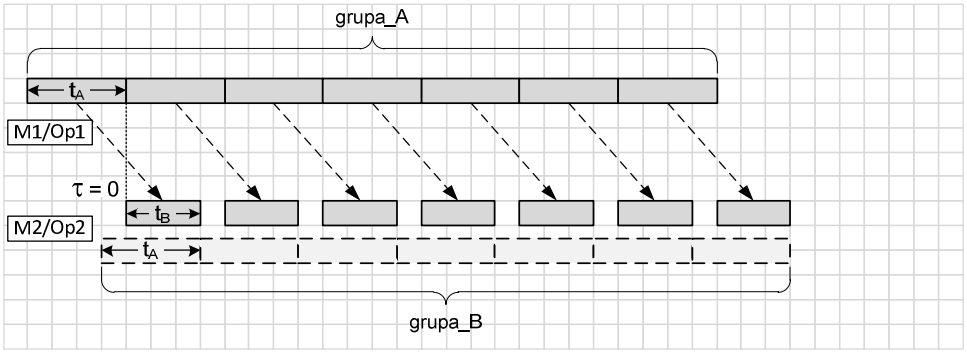
Dane definiujące grupę mają na celu przekazanie jednoznacznej informacji o momentach zakończenia przetwarzania kolejnych partii produkcyjnych, gdyż tylko te informacje są potrzebne do właściwego wyznaczenia czasów przetwarzania następnej maszyny. W przypadku, gdy praca maszyny odbywa się w sposób nieciągły, ze względu na relatywnie wolniejsze przetwarzanie maszyny poprzedzającej, jak w przypadku grupy C z rysunku 1, definicja parametrów grupy określana jest na podstawie grupy pomocniczej (rys. 1, prostokąty ograniczone przerywaną krawędzią). W grupie pomocniczej przetwarzanie partii kończy się w tych samych momentach co w grupie definiowanej, ale rozpoczyna się w tak wybranych chwilach, aby praca maszyny mogła być uznana jako ciągła. W ten sposób przypadek przetwarzania nieciągłego można wyrazić notacją spójną z przypadkiem pracy ciągłej, uzyskując w definicji grupy poprawne informacje o momentach zakończenia przetwarzania kolejnych partii produkcyjnych.

Elementarnym zadaniem obliczeniowym dla algorytmu wykorzystującego prezentowaną koncepcję grupowania partii produkcyjnych jest wyznaczanie zbioru grup, które zostaną przypisane danej maszynie w związku z przetwarzaniem pojedynczej grupy z maszyny poprzedzającej. Zbiór taki może być jedno lub wieloelementowy, w przykładzie z rysunku 1 z jednej grupy partii produkcyjnych (A) powstał zbiór trzech grup (B, C, D). Pomijając wpływ kalendarza dostępności maszyny docelowej, wynik zależy od względnej relacji w szybkości pracy maszyn, pomiędzy którymi przekazywane są partie produkcyjne oraz od tego, czy maszyna odbierająca partie rozpoczyna przetwarzanie w najszybszym możliwym momencie, czy też z opóźnieniem spowodowanym wcześniejszą zajętością. Trzy możliwe przypadki zostały ukazane na rysunkach 2a,b,c. Symbolem τ oznaczono opóźnienie pomiędzy momentem, w którym pierwsza partia produkcyjna przybywa do maszyny M2, a rzeczywistą chwilą rozpoczęcia obróbki.

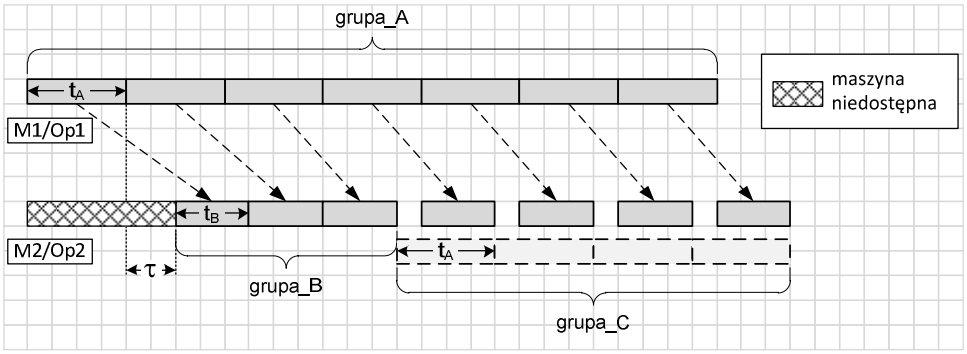
a) $t_A \le t_B,\ \tau \ge 0$



b) $t_A > t_B,\ \tau = 0$



c) $t_A > t_B,\ \tau > 0$



Rys. 2. Możliwe relacje między grupami partii produkcyjnych przetwarzanymi przez maszyny w następujących po sobie operacjach

Dla wszystkich przypadków z rysunku 2 przyjęto, że grupa partii produkcyjnych pierwszej operacji (*grupa_A*) zdefiniowana jest arbitralnie

$$\begin{gathered} grupa\_A = (start_A, num_A, unit_A), \\ start_A, num_A, unit_A \in \mathbf{Z}, \quad start_A \ge 0,\ num_A > 0,\ unit_A \equiv t_A \ge 0 \end{gathered} \tag{1}$$

gdzie: **Z** – zbiór liczb całkowitych.

Pierwsza z rozważanych możliwości (rys. 2a) uwzględnia sytuacje, w których czas przetwarzania partii produkcyjnej w operacji poprzedzającej (Op1) jest nie dłuży niż czas jej przetwarzania w kolejnej operacji (Op2). W takich przypadkach, niezależnie od opóźnienia początku przetwarzania na maszynie M2 ($\tau \geq 0$), partie produkcyjne przetwarzane są przez tę maszynę w sposób ciągły, tworząc jedną grupę

$$grupa\_B = (start_A + t_A + \tau, num_A, t_B). \tag{2}$$

Drugi wariant (rys. 2b) dotyczy przypadków, w których szybkość przetwarzania maszyny M2 jest większa niż maszyny M1, a opóźnienie rozpoczęcia przetwarzania na maszynie M2 nie występuje ($\tau = 0$). Wtedy partie produkcyjne w operacji Op2 są przetwarzane w sposób nieciągły, ale regularny, i można je reprezentować za pomocą jednej grupy

$$grupa\_B = (start_A + t_B, num_A, t_A). \tag{3}$$

Ostatni wariant (rys. 2c) zakłada taką samą relację między prędkościami pracy maszyn jak w przypadku poprzednim, ale występuje opóźnienie rozpoczęcia przetwarzania przez maszynę M2 ($\tau > 0$). W tej sytuacji partie produkcyjne operacji Op2 mogą utworzyć dwie grupy, pierwszą z przetwarzaniem ciągłym (*grupa_B*) i drugą z przetwarzaniem nieciągłym (*grupa_C*). Czas zakończenia *k*-tej partii produkcyjnej z grupy *grupa_A* oraz czas rozpoczęcia *k*-tej partii produkcyjnej z grupy *grupa_B* określają odpowiednio wyrażenia

$$TZA(k) = start_A + k \cdot t_A, \quad TRB(k) = start_A + t_A + \tau + (k-1)t_B\,. \tag{4}$$

Przetwarzanie w ramach operacji Op2 może odbywać się w sposób ciągły, jeżeli spełniona jest nierówność $TZA(k) \leq TRB(k)$. Największa wartość całkowita zmiennej *k*, spełniająca podaną nierówność wynosi

$$k_{\max} = \left\lfloor \frac{t_A + \tau - t_B}{t_A - t_B} \right\rfloor \tag{5}$$

i określa maksymalną liczbę partii produkcyjnych, składających się na grupę *grupa_B*. Ostatecznie zatem

$$grupa\_B = (start_A + t_A + \tau, num_B, t_B), \;\; num_B = \min(num_A, k_{max}). \tag{6}$$

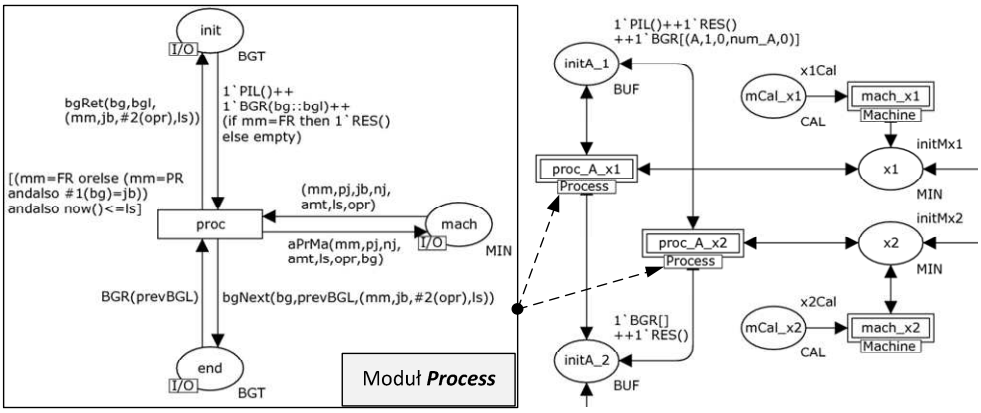
Grupa z partiami przetwarzanymi w sposób nieciągły (*grupa_C*) powstanie przy spełnieniu relacji $num_A > k_{max}$ i będzie zdefiniowana jako

$$grupa\_C = (start_A + k \cdot t_A + t_B, num_A - k_{max}, t_A). \tag{7}$$

Pełna procedura wyznaczania grup partii produkcyjnych wymaga dodatkowo uwzględnienia kalendarza dostępności maszyny M2. Jeżeli długość okresu dostępności, w którym ma być wykonane przetwarzanie grup wyznaczonych z relacji (2)-(7) nie będzie wystarczająca,

liczba partii produkcyjnych zostanie zmniejszona do takiej, aby przetwarzane partie zmieściły się w całości w przedziale dostępności maszyny. W takim przypadku grupa z maszyny M1 (*grupa_A* z rys. 2a,b,c) nie zostanie zużyta w całości, lecz zredukowana od lewej strony osi czasu o taką liczbę partii produkcyjnych, jaka może zostać przetworzona przez maszynę M2 w danym okresie dostępności. Zredukowana grupa będzie ponownie uwzględniona w kolejnym okresie dostępności maszyny M2. Proces ten może powtarzać się wielokrotnie (przypadek z jednokrotną redukcją reprezentuje przykład z rys. 1). Proste obliczenia, którymi trzeba uzupełnić zależności (2)-(7), aby w opisany sposób rozszerzyć procedurę, nie będą tutaj omawiane.

Stosując przedstawioną koncepcję grupowania partii produkcyjnych, opracowano reguły transformacji struktury systemu produkcyjnego typu FJS z przetwarzaniem partiami do modelu w postaci czasowej kolorowanej sieci Petriego. Podobnie jak w pracach [5,6], zastosowano sieć z dwupoziomową strukturą hierarchiczną. Poziom nadrzędny reprezentuje ogólną strukturę systemu produkcyjnego, na poziomie podrzędnym zdefiniowane są dwa moduły *Process* oraz *Machine*. Na rysunku 3 zamieszczono fragment struktury nadrzędnej przykładowego modelu sieciowego (po prawej stronie) oraz wnętrze modułu *Process*.



Rys. 3. Wnętrze modułu *Process* i fragment struktury nadrzędnej sieciowego modelu systemu produkcyjnego

Istotna zmiana w sposobie modelowania, w porównaniu z pracami [5,6], wynikająca z zastosowania koncepcji grupowania partii produkcyjnych, dotyczy wewnętrznej struktury modułu *Process*. Znakowanie miejsca *mach* przechowuje zbiór informacji o maszynie [6], w tym wartość określającą jeden z czterech możliwych stanów jej pracy:

- *FR* (*Free*) – maszyna wolna, oczekuje na przyjęcie kolejnej operacji,
- *ST* (*Setup*) – maszyna oczekuje na przezbrojenie do nowej operacji,
- *PR* (*Process*) – maszyna jest gotowa do rozpoczęcia przetwarzania grupy partii produkcyjnych, pierwszej lub kolejnej w ramach operacji,
- *WT* (*Wait*) – ostatnia czynność przezbrojenia lub przetwarzania nie zmieściła się w bieżącym przedziale dostępności maszyny, maszyna oczekuje na kolejny przedział dostępności.

Grupy partii produkcyjnych, przechodzące przez kolejne operacje, modelowane są znakowaniami miejsc *init* oraz *end*. Znakowania te reprezentują odpowiednio zbiór grup

partii produkcyjnych oczekujących na przetworzenie w operacji modelowanej przez daną podstronę *Process* oraz zbiór grup partii powstałych po przetworzeniu i oczekujących na kolejną operację. Typ znakowania tych miejsc, nazwany *BGT* (*Batch Group Type*) zdefiniowany jest przez następujące deklaracje

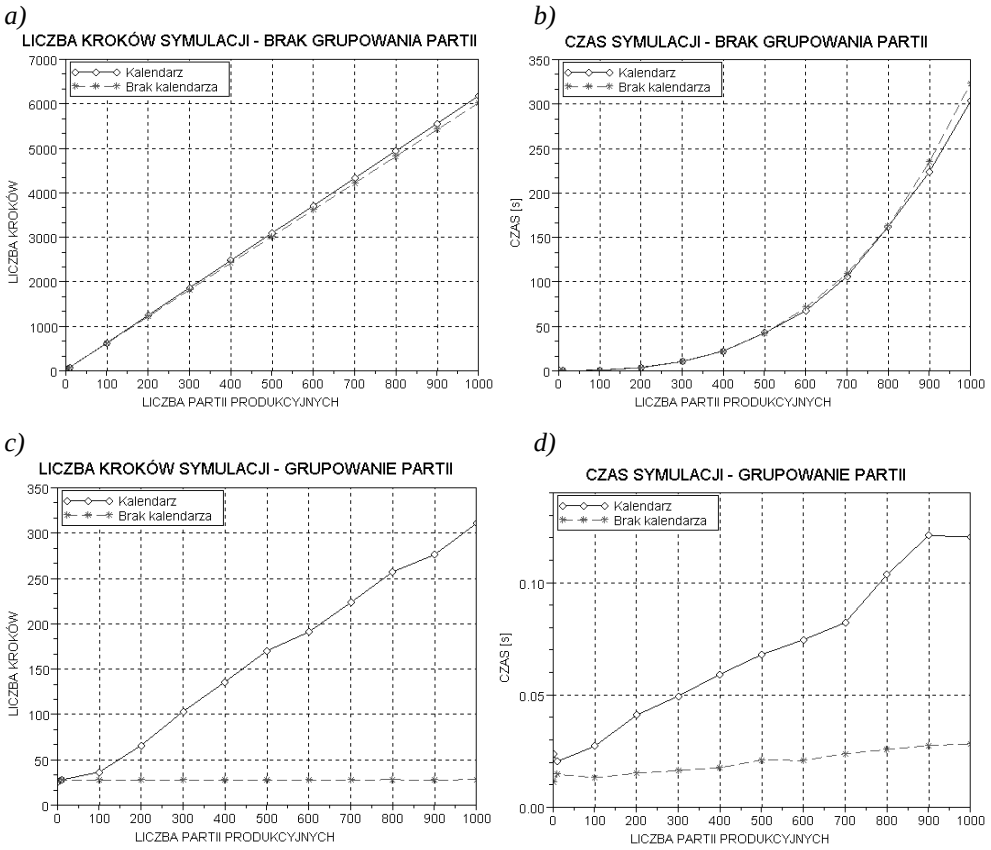
```
colset BG = product JOB * INT * INT * INT * INT;
(* BatchGroup: (job, size, start, num, unit) *)

colset BG_L = list BG;

colset BGT = union BGR:BG_L + PIL:UNIT + RES:UNIT timed;
```

Typ *BGT* jest trójelementową unią, co oznacza, że tokeny tego typu mogą wystąpić w trzech wariantach *BGR*, *PIL* i *RES*, posiadających typy podrzędne odpowiednio *BG_L*, *UNIT* i *UNIT*. Token wariantu *BGR* jest listą elementów typu *BG*, z których każdy element definiuje jedną grupę partii przetwarzania. Definicja grupy (typ *BG*) stanowi krotkę pięcioelementową, w której pierwsze pole (*job*) określa zadanie do którego należy grupa, drugie pole (*size*) definiuje rozmiar pojedynczej partii przetwarzania, to jest liczbę przetwarzanych elementów, które się na nią składają, trzy kolejne pola (*start*, *num*, *unit*) są równoważne parametrom definiującym grupę, wprowadzonym na początku rozdziału. Tokeny wariantu *PIL* (*Pilot*) wprowadzane są do miejsc *end* wraz z dołączaniem do list *BGR* kolejnych grup partii produkcyjnych. Pieczątki czasowe tokenów *PIL* ustawiane są na taką wartość, aby grupy nie mogły być pobierane do kolejnych operacji przed przetworzeniem pierwszej partii z grupy operacji poprzedzającej, to znaczy, wraz z uzupełnieniem listy o grupę (*start*, *num*, *unit*) dodawany jest token *PIL* z pieczątką czasową równą *start* + *unit*. W ramach znakowania początkowego każdemu miejscu posiadającemu typ *BGT* przydzielany jest jeden token wariantu *RES* (*Reservation*). Token ten pobierany jest bez zwracania przy pierwszym wykonaniu tranzycji *proc*, modelującej daną operację, a pobranie tego tokenu jest warunkiem przejścia maszyny przypisanej do operacji ze stanu *FR* do stanu *ST*. Mechanizm taki zapewnia przetworzenie każdej operacji, to jest wszystkich jej partii, w całości przez jedną z maszyn dla niej przeznaczonych. Przekształcenia grup partii produkcyjnych przesyłanych do kolejnych operacji realizują funkcje *bgNext* oraz *bgRet*.

Wykonano badania symulacyjne, pozwalające zebrać dane porównawcze dotyczące zależności liczby kroków i długości czasów symulacji od średniej liczby partii produkcyjnych przypadających na jedną operację dla podstawowego i wykorzystującego grupowanie partii sposobu modelowania. Uzyskane wyniki przedstawiono na rysunkach 4a,b,c,d. Wszystkie wykresy dotyczą tej samej struktury systemu produkcyjnego, złożonej z pięciu maszyn i dwóch zadań po trzy operacje każda, czas obróbki poszczególnych partii produkcyjnych pozostawał jednakowy, niezależnie od liczby partii. Rejestrowano liczbę kroków i czas wykonania, przypadające średnio na jedną symulację, wykonując po dziesięć symulacji z przydzieleniem dla każdej operacji kolejno 1, 10, 100, 200, …, 900, 1000 partii produkcyjnych. Zestawienie uwzględnia symulacje wykonane z użyciem modelu bez grupowania i z grupowaniem partii produkcyjnych oraz dwa warianty: z kalendarzami dostępności maszyn i z maszynami dostępnymi bez ograniczeń. W wariancie z kalendarzami dostępności zdefiniowano dla każdej z maszyn dwadzieścia przedziałów dostępności, rozmieszczonych statystycznie równomiernie, ale z losowo wybranymi czasami rozpoczęcia i zakończenia, przy czym długość uszeregowania sięgała końcowych przedziałów dostępności dopiero przy największej liczbie partii produkcyjnych, równej 1000.



Rys. 4. Wyniki symulacji modeli sieciowych systemów produkcyjnych: a), b) brak grupowania partii produkcyjnych; c), d) grupowanie partii produkcyjnych

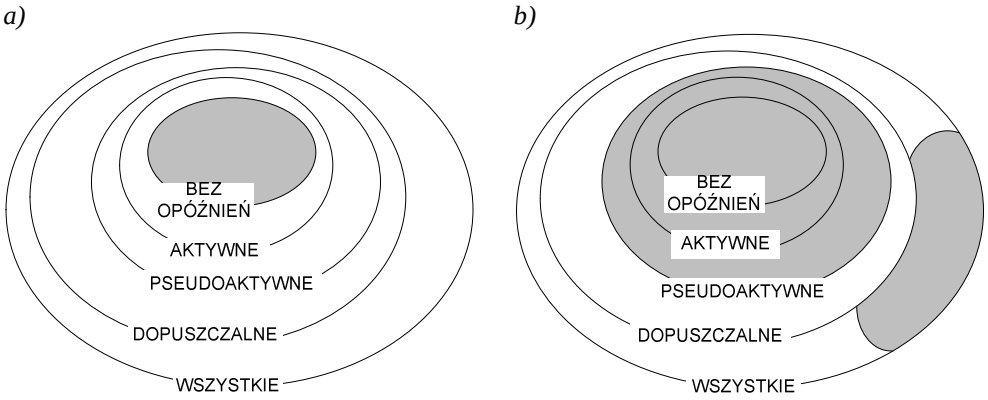
W modelu bez grupowania partii produkcyjnych, liczba kroków okazała się w bardzo dobrym przybliżeniu proporcjonalna do liczby partii (rys. 4a), a współczynnikiem proporcjonalności jest sumaryczna liczba operacji. Wynika to z faktu, że liczba kroków symulacji związanych ze zdarzeniami innymi niż przetwarzanie partii produkcyjnych, na przykład z obsługą kalendarzy dostępności czy przezbrojeniami, jest zaniedbywalna. W rozwiązaniu wykorzystującym grupowanie partii produkcyjnych (rys. 4c), gdy maszyny są dostępne bez ograniczeń czasowych, symulacja wymaga stałej, niewielkiej liczby kroków (równej około 30), ponieważ grupy partii przetwarzane są ten sam sposób, niezależnie od liczebności. Ograniczenie dostępności maszyn zwiększa liczbę kroków, gdyż grupy partii produkcyjnych niemieszczące się w jednym przedziale dostępności ulegają podziałowi, tworząc coraz liczniejsze zbiory grup przetwarzanych w kolejnych operacjach. W realistycznej konfiguracji systemu produkcyjnego należy oczekiwać liczby kilkudziesięciu do kilkuset partii produkcyjnych przypadających na przedział dostępności maszyny. Ta liczba jest szacunkowym współczynnikiem względnego zmniejszenia liczby kroków symulacji przy przejściu z modelu bez grupowania partii do modelu, w którym zastosowano grupowanie. Funkcje reprezentujące czas symulacji (rys. 4b,d) rosną gwałtowniej niż liczby kroków, co wiąże się z algorytmem symulacji modelu, zastosowanym w CPN Tools. W

algorytmie utworzonym dla aplikacji harmonogramującej zapewniono stały czas wykonań tranzycji, niezależnie od rozmiaru znakowania miejsc połączonych z tranzycjami, dlatego w rozwiązaniu docelowym należy oczekiwać złożoności czasowej odpowiadającej raczej wykresom z rysunków 4a,c niż 4b,d.

3. Generowanie harmonogramów pseudoaktywnych

Stosując formalizm czasowych sieci Petriego do modelowania systemów produkcyjnych z ograniczeniami kolejnościowymi, można w prosty sposób zaimplementować dwie reguły przydziału maszyn do operacji [8]. W pierwszym przypadku przetwarzanie rozpoczyna się, gdy tylko maszyna jest wolna (przydział zachłanny), co prowadzi do powstawania uszeregowań bez opóźnień (*nondelay schedules*) [1]. W drugim przypadku konkretne maszyny są arbitralnie, to znaczy losowo lub poprzez zmienne decyzyjne sterujące wykonywaniem modelu, przydzielane do operacji i arbitralnie określana jest kolejność przetwarzania operacji przez maszyny. Pozwala to uzyskać wszystkie harmonogramy, w których niemożliwe będą lokalne przesunięcia operacji w lewo, to jest harmonogramy pseudoaktywne (*semi-active schedules*) [1] wraz z podzbiorem harmonogramów niedopuszczalnych, naruszających ograniczenia kolejnościowych systemu.

Na rysunkach 5a,b przedstawiono zbiory uszeregowań uzyskiwanych dla dwóch wymienionych wariantów modelowania, wykorzystując znaną hierarchię klas harmonogramów, reprezentowaną za pomocą diagramu Venna [1].



Rys. 5. Podzbiory uzyskiwanych uszeregowań: a) dla modelu sieciowego z zachłannym przydziałem maszyn, b) dla modelu sieciowego z arbitralnym przydziałem maszyn

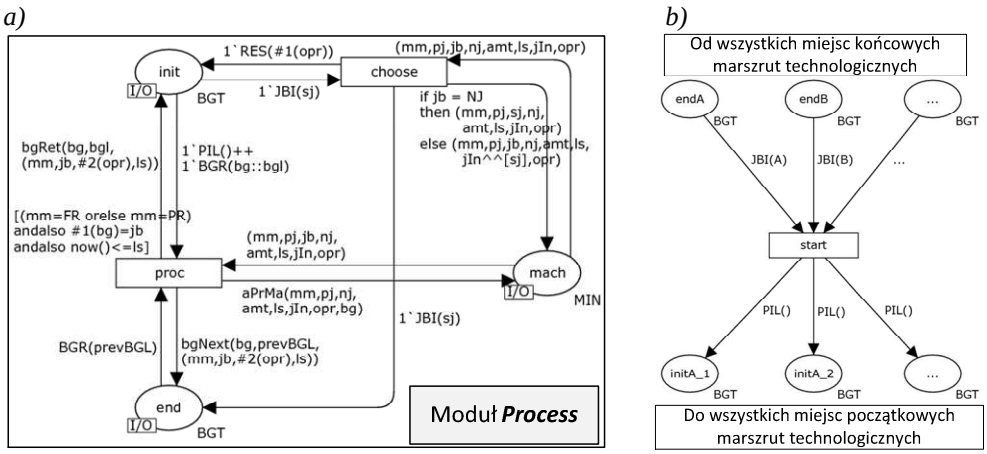
Oba zbiory nie są najlepsze do przeszukiwań w procesie harmonogramowania. Zbiór uszeregowań bez opóźnień nie musi zawierać harmonogramów optymalnych w odniesieniu do standardowych funkcji kryterialnych. W przypadku zbioru z rysunku 5b, podzbiór uszeregowań niedopuszczalnych może istotnie zdominować rozmiarem podzbiór uszeregowań pseudoaktywnych, czyniąc przeszukiwania bardzo nieefektywnymi, co potwierdzają badania symulacyjne przedstawione w pracy [8].

Przyczyną problemów z uzyskaniem przestrzeni harmonogramów o własnościach dogodnych dla optymalizacji, patrząc z perspektywy wariantu generującego uszeregowania bez opóźnień, jest posiadanie przez tokeny przechodzące przez miejsca *init*/*end* podwójnej

roli: modelowania przetwarzanego materiału oraz rezerwacji maszyny. Maszyny są rezerwowane dla danej operacji równolegle z rozpoczęciem jej przetwarzania, co nie pozwala na rezerwacje z wyprzedzeniem, które prowadziłyby do uszeregowań pseudoaktywnych innych niż bez opóźnień.

Proponuje się nowe rozwiązanie, polegające na zastosowaniu dwóch rodzajów tokenów, przechodzących przez miejsca *init*/*end*, oraz dwóch etapów symulacji, jednego dla rezerwacji maszyn i ustalenia kolejności wykonywania operacji przez maszyny, drugiego do modelowania przepływu przetwarzanego materiału. Model weryfikujący metodę zaimplementowano z wykorzystaniem CPN Tools, jako rozszerzenie modelu omówionego w poprzednim rozdziale:

1. W deklaracji typu *BGT* dodano wariant *JBI* (*JoB Information*) dla tokenów uczestniczących w akcjach rezerwacji maszyn oraz zmodyfikowano wariant *RES*, tak aby przechowywał informację o maszynie zarezerwowanej dla operacji.
2. Definicję typu tokenów reprezentujących stan maszyny (typ *MIN*) uzupełniono o listę przechowującą oznaczenia kolejnych zadań, do których przypisano maszynę.
3. W module *Process* dodano tranzycję *choose*, odpowiedzialną za akcje rezerwacji maszyn (rys 6a).
4. W strukturze nadrzędnej modelu dodano tranzycję *start* (rys. 6b).
5. Usunięto znaczniki *PIL* z początkowych miejsc *init* wszystkich zadań i wstawiono do nich znaczniki *JBI* z odpowiednimi symbolami zadań: *JBI(A)*, *JBI(B)* itd.



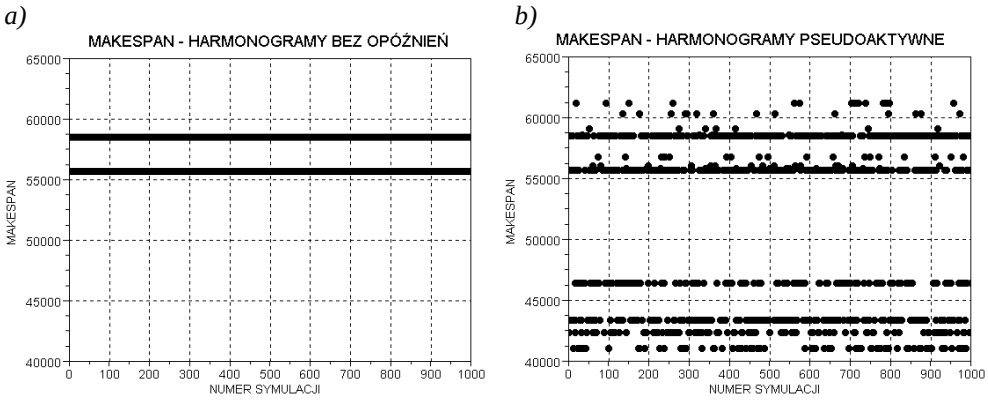
Rys. 6. Model sieciowy systemu produkcyjnego z symulacją dwuetapową:
a) wnętrze modułu *Process*, b) sposób przyłączenia tranzycji *start*

W pierwszym etapie symulacji modelu odbywa się przypisanie maszyn do operacji oraz ustalenie kolejności wykonywania operacji przez każdą z maszyn. Wiąże się to z wykonywaniem tranzycji *choose*. Brak znaczników *PIL* w miejscach *init* rozpoczynających marszruty blokuje możliwość wykonań tranzycji *proc* w tym etapie. Podczas wykonania, tranzycja *choose* pobiera z miejsca *init* token *JBI* i oddaje go do kolejnego miejsca marszruty technologicznej (*end*). Równocześnie do miejsca *init* wprowadzany jest token *RES(m)*, gdzie *m* jest symbolem maszyny połączonej z wykonywaną tranzycją *choose*, a lista zadań obecna w tokenie znakującym miejsce *mach* uzupełniana jest symbolem zadania pozyskanym z tokenu *JBI*. Sposób przekazywania znaczników *JBI* powoduje, że w każdym

kroku pierwszego etapu symulacji wykonana może być dowolna tranzycja *choose* modelująca operację, której wszystkie operacje poprzedzające w ciągu technologicznym zostały już wykonane. Czyni to rozważany etap symulacji równoważnym co do efektów procedurze wyznaczania ścieżek acyklicznych w grafie dysjunktywnym lub tworzeniu listy permutacyjnej z powtórzeniami [1,2]. W całym tym etapie zegar globalny modelu utrzymuje wartość 0, gdyż do pieczątek czasowych tokenów nie są doliczane żadne opóźnienia, znaczenie mają bowiem tylko relacje kolejnościowe, a nie czasowe.

Zakończenie pierwszego etapu następuje wraz z przybyciem tokenów *JBI* do miejsc końcowych marszrut technologicznych wszystkich zadań. Wtedy wykonana zostaje tranzycja *start*, która wprowadza tokeny *PIL* do miejsc początkowych marszrut technologicznych. Rozpoczyna się drugi etap symulacji, w którym uczestniczą tranzycje *proc* z modułu *Process*. Etap ten jest niemalże identyczny z procesem symulacji produkcji z grupowaniem partii, omówionym w rozdziale 2, z tą różnicą, że przydziały maszyn do operacji oraz kolejności przetwarzania operacji przez maszyny są arbitralnie ustalone na podstawie rezultatów poprzedniego etapu.

Przedstawiona dwuetapowa metoda symulacji pozwala generować dowolne uszeregowania spełniające ograniczenia kolejnościowe systemu (etap pierwszy) z operacjami rozpoczynanymi możliwie najszybciej (etap drugi), efektywnie będzie to więc pełny zbiór harmonogramów pseudoaktywnych. W ramach dalszych prac planowana jest implementacja dodatkowej funkcjonalności, która umożliwi generowanie uszeregowań aktywnych. Przykładowe rezultaty, uzyskane z zastosowaniem zaproponowanej metody, ukazane są na rysunkach 7a,b.



Rys. 7. Długości uszeregowania uzyskiwane dla dwóch metod symulacyjnych:
a) symulacja jednoetapowa (harmonogramy bez opóźnień),
b) symulacja dwuetapowa (harmonogramy pseudoaktywne)

Zamodelowana została ta sama struktura systemu produkcyjnego, co w rozdziale poprzednim (5 maszyn, 2 zadania, 6 operacji). Dokonano po 1000 symulacji, dla dwóch wariantów modelu: z przetwarzaniem jednoetapowym i zachłannym przydziałem maszyn (harmonogramy bez opóźnień) oraz z przetwarzaniem dwuetapowym. Dla każdej z symulacji rejestrowano otrzymywane długości uszeregowania. Odnotowano 2 możliwe długości uszeregowania wśród harmonogramów bez opóźnień (rys. 7a) i 17 różnych długości uszeregowania wśród harmonogramów pseudoaktywnych (rys. 7b).

4. Podsumowanie

Pierwsza z przedstawionych propozycji, grupowanie partii produkcyjnych, zmniejsza złożoność obliczeniową algorytmu symulacji modelu, dzięki zastąpieniu sekwencyjnego przetwarzania każdej partii z osobna poprzez obliczenia o charakterze skumulowanym. Koncepcja ta nie wymaga, aby system produkcyjny był reprezentowany przez sieć Petriego. Można ją zastosować w realizacji różnych algorytmów modelowania i harmonogramowania zadań, w których występuje przetwarzanie partiami, z produkcją o organizacji równoległej, szeregowo-równoległej lub szeregowej.

Druga propozycja wiąże się ściśle z użytym formalizmem modelowania. Stosując czasowe sieci Petriego, uzyskanie zbioru uszeregowań pseudoaktywnych nie jest bezpośrednio możliwe w modelu wykonywanym jednoetapowo z jednym typem tokenów, reprezentujących przetwarzany materiał. Przedstawiona koncepcja zakłada symulację dwuetapową i dwa rodzaje tokenów dla rozdzielenia czynności wyznaczenia dopuszczalnego przydziału maszyn i kolejności operacji oraz określenia czasów przetwarzania poszczególnych składników produkcyjnych. Pierwszy z etapów można również zrealizować w oparciu o procedurę zewnętrzną względem modelu sieciowego, na przykład poprzez utworzenie listy permutacji z powtórzeniami, jednakże zaproponowana metoda pozwala na reprezentację całego algorytmu w ramach spójnego formalizmu oraz łatwe wprowadzanie ewentualnych modyfikacji i rozszerzeń. Metoda może okazać się przydatna w modelowaniu dowolnych systemów produkcyjnych typu gniazdowego.

Literatura

1. Pinedo M. L.: Scheduling. Theory, Algorithms, and Systems, Springer 2008.
2. Bierwith, C.: A Genaralised Permutation Approach to Job Shop Scheduling with Genetic Algorithms. OR Spectrum, vol. 17, 2-3, 1995, 89-92.
3. Tuncel G., Bayhan G. M.: Applications of Petri nets in production scheduling: a review, Int. J. Adv. Manuf. Technol., Springer-Verlag, vol. 34, 7-8, 2007, 762-773.
4. Jensen K., Kristensen L. M.: Coloured Petri Nets. Modeling and Validation of Concurrent Systems, Springer 2009.
5. Bożek A., Żabiński T.: Kolorowane czasowe sieci Petriego jako narzędzie symulacji off-line dla inteligentnych systemów produkcyjnych, Przegląd Elektrotechniczny 9/2010.
6. Bożek A.: Zastosowanie kolorowanych sieci Petriego w nadążnym harmonogramowaniu produkcji, XVII Konferencja „Systemy Czasu Rzeczywistego”, Gdańsk 27-29 września 2010.
7. CPN Tools Home Page, http://cpntools.org/.
8. Zhang H., Gu M., Song X.: Modeling and Analysis of Real-life Job Shop Scheduling Problems by Petri Nets, 41st Annual Simulation Symposium, 2008.

Mgr inż. Andrzej BOŻEK
Katedra Informatyki i Automatyki
Politechnika Rzeszowska
35-959 Rzeszów, ul. W. Pola 2
tel.: (17) 865 17 66
e-mail: abozek@prz-rzeszow.pl