GENERATOR MAX-PLUS ALGEBRAICZNYCH RÓWNAŃ STANU SYSTEMÓW PRODUKCYJNYCH

Jarosław STAŃCZYK

Streszczenie: Praca przedstawia modelowanie systemów produkcyjnych przy użyciu max-plus algebry, oraz opracowany przez autora publikacji program *Phoebe* - generator max-plus algebraicznych równań stanu, służący do automatyzacji modelowania takich systemów a co za tym idzie do wspomagania ich planowania i symulacji. Generator napisany został w języku Python. Generuje on kod, który może być bezpośrednio użyty w pakietach MathWorks Matlab lub GNU Octave bazując na opisie systemu produkcyjnego w języku YAML, przypominający język naturalny.

Słowa kluczowe: modelowanie systemów produkcyjnych, dyskretne systemy zdarzeniowe, max-plus algebra

1. Wstęp

Rozważmy klasę systemach produkcyjnych, którą można opisać za pomocą tzw. *dyskretnych systemów zdarzeniowych* (ang. *discrete event systems*, w skrócie DES). Są to systemy wykonane przez człowieka, które składają się ze skończonej liczby *zasobów* (np. maszyn produkcyjnych, buforów czy kanałów komunikacyjnych), które są dzielone poprzez skończoną liczbę *użytkowników* (np. zadań, typów produktów czy pakietów informacji) współpracujących (lub konkurujących) by osiągnąć cel (np. końcowy produkt, transmisję, obliczenia w rozproszonym systemie komputerowym). DES są systemami dynamicznymi, w których przejście z jednego stanu do innego jest wywołane poprzez *zdarzenie*, występujące w pewnej dyskretnej chwili czasowej. Zdarzeniu odpowiada rozpoczęcie lub zakończenie jakiegoś działania, rozważając procesy produkcyjne może to być rozpoczęcie lub zakończenie operacji technologicznej, zapełnienie bufora międzyoperacyjnego, etc. Wspólną cechą rozważanych systemów jest to, iż rozpoczęcie zdarzenia zależy od zakończenia szeregu innych zdarzeń w systemie. Tego rodzaju systemy nie dają się opisywać równaniami różniczkowymi czy różnicowymi. Dodatkowo, ze względu na realizacje określonej ilości wyrobów, charakteryzują się zachowaniem cyklicznym.

Wiele narzędzi i technik jest używanych w badaniach nad DES, przegląd można znaleźć np. w [3]. Najczęściej stosowaną techniką jest symulacja komputerowa. Zasadnicza niedogodność symulacji jest spowodowana tym, iż nie pozwala ona dostatecznie zrozumieć wpływu poszczególnych parametrów systemu na jego podstawowe własności, takie jak stabilność, odporność na zakłócenia czy efektywność funkcjonowania. Techniki analityczne dają pod tym względem o wiele lepsze wyniki, stad są one preferowane jako narzędzia do modelowania czy analizy DES. W niniejszej pracy rozważane są problemy modelowania pewnej klasy DES przy użyciu (max, +) algebry [1, 5]. Rozważana klasa DES jest ograniczona do systemów, których modele są (max, +) liniowe i stacjonarne, tzn. są to systemy deterministyczne, w których użytkownik nie ma możliwości wyboru zasobu. Przykładami takich systemów są między innymi: systemy produkcyjne [6] czy transportowe [10].

Praca zorganizowana jest następująco: rozdział 2 zawiera podstawy (max, +) algebry niezbędne do modelownia systemów produkcyjnych. W rozdziale 3 zaprezentowano przykładowy, prosty system produkcyjny i jego model oraz sformułowano rozważany problem. Program *Phoebe* i jego użycie przedstawiono w rozdziale 4. Na zakończenie, rozdział 5 podsumowuje niniejsza pracę, przedstawia wyniki oraz wskazuje kierunki dalszych badań.

Wszelkich eksperymentów obliczeniowych dokonano przy użyciu pakietu *Max–Plus Algebra Toolbox for Matlab*® [7]. Do generowania modeli systemów produkcyjnych wykorzystanych w przykładach użyto programu *Phoebe* dostępnego na stronie internetowej autora [8].

2. Max-plus algebra

Jako pierwszą publikację dotyczącą (max, +) algebry wskazuje się [4]. Standardową pozycją jest [1], a krótki przegląd metod i zastosowań można znaleźć w [5]. Pod pewnymi względami (max, +) algebra jest porównywalna z algebrą konwencjonalną. W algebrze (max, +) operatory dodawania (+) i multiplikacji (x) z konwencjonalnej algebry zostały zastąpione odpowiednio poprzez operatory maksymalizacji (*max*) i dodawania (+). Dzięki temu, uzyskuje się liniowy opis, w sensie (max, +) algebry, pewnych klas nieliniowych w konwencjonalnej algebrze systemów.

W ciągu ostatnich lat teoria dotycząca (max, +) algebry i jej zastosowań jest stale rozwijana, a jednym z kierunków jest modelowanie i planowanie produkcji [6, 9].

2.1. Podstawy

Struktura algebraiczna zdefiniowana w zbiorze liczb rzeczywistych wraz z $-\infty$, tzn. $\mathbb{R}\varepsilon = \mathbb{R} \cup \{-\infty\}$, w której zdefiniowane są dwa działania:

- $\forall\, a, b \in \mathbb{R}\varepsilon$: $a \otimes b \equiv a + b$;
- $\forall\, a, b \in \mathbb{R}\varepsilon$: $a \oplus b \equiv max(a, b)$;

jest nazywana *max-plus algebrą* i oznaczana jako $\mathbb{R}_{max} = (\mathbb{R}\varepsilon, \otimes, \oplus)$. W publikacji przyjęto notację prezentowaną w [1], w której $\varepsilon = -\infty$ i $e = 0$. Notacja ε i e zamiast odpowiednio $-\infty$ i 0 jest używana dla odróżnienia specjalnego znaczenia tych wartości, jak również po to, aby uniknąć pomyłek wynikających z ich roli w algebrze konwencjonalnej. W niniejszej publikacji używany jest zapis ab zamiast $a \otimes b$ wszędzie tam, gdzie nie powoduje on dwuznaczności. Poniżej rozszerzono działania w (max, +) algebrze o operacje na macierzach. Dodawanie macierzy: niech dane będą dwie macierze $\mathbf{A}, \mathbf{B} \in \mathbb{R}\varepsilon^{m \times n}$. Macierz $(\mathbf{A} \oplus \mathbf{B}) \in \mathbb{R}\varepsilon^{m \times n}$ otrzymywana jest przez dodanie odpowiednich elementów, tzn.:

$$(\mathbf{A} \oplus \mathbf{B})_{ij} = (\mathbf{A})_{ij} \oplus (\mathbf{B})_{ij}, \quad i = 1, \ldots, m; j = 1, \ldots, n. \tag{1}$$

Mnożenie macierzy: niech $\mathbf{A} \in \mathbb{R}\varepsilon^{m \times p}$ i $\mathbf{B} \in \mathbb{R}\varepsilon^{p \times n}$, wtedy iloczyn $(\mathbf{A} \otimes \mathbf{B}) \in \mathbb{R}\varepsilon^{m \times n}$, gdzie $(\mathbf{A} \otimes \mathbf{B})_{ij}$ jest produktem i-tego wiersza $\mathbf{A}$ i j-tej kolumny $\mathbf{B}$, czyli:

$$(\mathbf{A} \otimes \mathbf{B})_{ij} = \bigoplus_{k=1}^{p} (\mathbf{A})_{ik} \otimes (\mathbf{B})_{kj} \equiv \max_{k} \left((\mathbf{A})_{ik} + (\mathbf{B})_{kj} \right), \quad i = 1, \ldots, m; j = 1, \ldots, n \tag{2}$$

gdzie: $\bigoplus_{j=1}^{m} a_j$ jest skróconym zapisem $a_1 \oplus \ldots \oplus a_m$.

2.2. Modelowanie DES

Jednym z najbardziej znanych równań systemów dynamicznych jest:

$$\mathbf{x}(t) = \mathbf{A}\mathbf{x}(t\text{-}1), \quad t = 1, 2, \ldots \tag{3}$$

gdzie wektor $\mathbf{x} \in \mathbb{R}^n$ reprezentuje *stan* rozważanego systemu, macierz $\mathbf{A} \in \mathbb{R}^{n \times n}$ jest *macierzą systemu*. Jeśli znane są warunki początkowe, tzn. $\mathbf{x}(0) = \mathbf{x}_0$, to zachowanie systemu jest w pełni zdeterminowane. Równanie (3) zapisane w notacji (max, +), tzn., gdy $\mathbf{x} \in \mathbb{R}\varepsilon^n$, $\mathbf{A} \in \mathbb{R}\varepsilon^{n \times n}$, przedstawia się następująco:

$$\forall k \in \mathbb{N}: \qquad \mathbf{x}(k) = \mathbf{A} \otimes \mathbf{x}(k\text{-}1). \tag{4}$$

Celowo w równaniu (4) zamiast t występuje k, gdyż nie oznacza ono czasu pojawienia się zdarzenia, lecz numer cyklu (iteracji), w którym zdarzenie ma miejsce. Stacjonarny model (max, +)-liniowy 1-go rzędu, wiążący wejścia i wyjścia systemu jest uogólnieniem (4):

$$\forall k \in \mathbb{N}: \qquad \mathbf{x}(k) = \mathbf{A}\mathbf{x}(k\text{-}1) \oplus \mathbf{B}\mathbf{u}(k), \tag{5}$$

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) \oplus \mathbf{D}\mathbf{u}(k). \tag{6}$$

Poszczególne elementy to: *wektor wejść* (wektor sterujący) $\mathbf{u} \in \mathbb{R}\varepsilon^r$, *macierz wejścia* $\mathbf{B} \in \mathbb{R}\varepsilon^{n \times r}$, *wektor wyjść* $\mathbf{y} \in \mathbb{R}\varepsilon^m$, *macierz wyjścia* $\mathbf{C} \in \mathbb{R}\varepsilon^{m \times n}$ i *macierz sprzężenia bezpośredniego* $\mathbf{D} \in \mathbb{R}\varepsilon^{m \times r}$. W ogólnym przypadku, dla systemu N-tego rzędu, tzn. takiego, w którym N poprzednich iteracji wpływa na bieżące zachowanie systemu, z uwikłanym $\mathbf{x}(k)$ po obu stronach równania stanu, model przedstawia układ równań (7) i (8):

$$\forall k \in \mathbb{N}: \qquad \mathbf{x}(k) = \bigoplus_{i=0}^{N} \mathbf{A}_i \mathbf{x}(k\text{-}i) \oplus \bigoplus_{i=0}^{N-1} \mathbf{B}_i \mathbf{u}(k\text{-}i), \tag{7}$$

$$\mathbf{y}(k) = \bigoplus_{i=0}^{N-1} \left(\mathbf{C}_i \mathbf{x}(k\text{-}i) \oplus \mathbf{D}_i \mathbf{u}(k\text{-}i)\right). \tag{8}$$

W dalszej części pracy człon **Du** w równaniu (6) i (8) jest pomijany a równanie wyjścia przybiera postać:

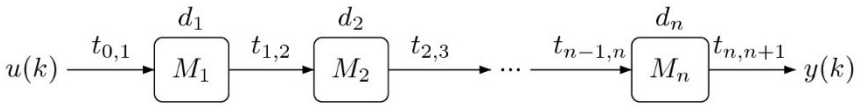
$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k). \tag{9}$$

3. Modelowanie systemów produkcyjnych

Modelowanie systemów produkcyjnych zostało przedstawione poniżej, bazując na sekwencyjnej linii produkcyjnej. Inne konfiguracje przedstawiono w [9].

3.1. Sekwencyjna linia produkcyjna

Rozważmy linię produkcyjną z szeregiem n stanowisk, przedstawioną na rys. 1. Z linią powiązany jest bufor wejściowy (magazyn), nazwijmy go u, z którego pobierane są detale.



Rys. 1. Sekwencyjna linia produkcyjna z szeregiem n stanowisk

Tego rodzaju linia ma jedno stanowisko wejściowe, tzn. takie, do którego trafiają detale z magazynu wejściowego, oraz jedno stanowisko wyjściowe, tzn. takie, z którego gotowy produkt (czy półprodukt) opuszcza linię. Produkt opuszczający linię trafia do magazynu wyjściowego (końcowego) y. Niech $u(k)$, $y(k)$ oraz $x_i(k)$ oznaczają odpowiednio czas, w którym k-ty detal jest dostępny na wejściu na linię produkcyjną (tzn. jest w magazynie wejściowym), wyjściu (tzn. jest w magazynie końcowym), oraz na stanowisku M_i. Jeśli materiał trafia do systemu w k-tej iteracji (tzn. w czasie $u(k)$), wówczas jest on dostępny na stanowisku M_1 w czasie $t = u(k) + t_{0,1}$, gdzie $t_{0,1}$ jest czasem transportu elementu z magazynu wejściowego do stanowiska M_1. Jednak, na M_1 można rozpocząć obrabiać ten element tylko wówczas, gdy zakończono już pracę nad poprzednim detalem, tzn. nad detalem z iteracji k-1. Zakładając, że czas operacji na stanowisku M_1 wynosi d_1, wówczas detal z iteracji k-1 opuści stanowisko M_1 w czasie $t = x_1(k\text{-}1) + d_1$. Zakładamy, że detal opuści stanowisko natychmiast po zakończeniu operacji, oraz, że detal zostanie poddany obróbce natychmiast, gdy tylko na danym stanowisku jest dostępny. Powyższe warunki można przełożyć na następujące równanie opisujące czas, w którym zacznie się obróbka k-tego detalu na stanowisku M_1:

$$\forall k \in \mathbb{N}: \; x_1(k) = \max\big(d_1 + x_1(k\text{-}1),\, t_{0,1} + u(k)\big), \tag{10}$$

(max, +) algebrze przedstawia się to następująco:

$$\forall k \in \mathbb{N}: \; x_1(k) = d_1\, x_1(k\text{-}1) \oplus t_{0,1}\, u(k), \tag{11}$$

Podobnie dla stanowiska M_i, k-ty detal zacznie być obrabiany na tym stanowisku, gdy:

- zakończy się obróbka detalu na stanowisku M_{i-1} i zostanie on dostarczony do M_i (czas transportu wynosi $t_{i-1,i}$),
- na M_i zakończy się obróbka k-1-go detalu.

W (max, +) algebrze wyraża się to następująco:

$$\forall k \in \mathbb{N}: \; x_i(k) = t_{i-1,i}\, d_{i-1}\, x_{i-1}(k) \oplus d_i\, x_i(k\text{-}1), \tag{12}$$

Scalając powyższe równania w zapis macierzowy:

$$\forall k \in \mathbb{N}: \; \mathbf{x}(k) = \mathbf{A}_0\mathbf{x}(k) \oplus \mathbf{A}_1\mathbf{x}(k\text{-}1) \oplus \mathbf{B}_0\mathbf{u}(k), \tag{13}$$

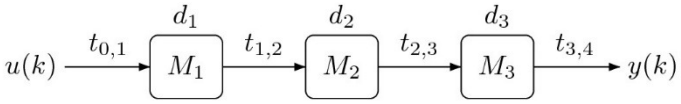
gdzie:

$$\mathbf{x}(k) = \begin{bmatrix} x_1(k) \\ x_2(k) \\ \vdots \\ x_n(k) \end{bmatrix}, \mathbf{A}_0 = \begin{bmatrix} \varepsilon & \varepsilon & \cdots & \varepsilon \\ t_{1,2}d_1 & \varepsilon & t_{1,2}d_1 & \varepsilon \\ & \ddots & & \vdots \\ \varepsilon & \varepsilon & t_{n-1,n}d_{n-1} & \varepsilon \end{bmatrix}, \mathbf{A}_1 = \begin{bmatrix} d_1 & \varepsilon & \cdots & \varepsilon \\ \varepsilon & d_2 & \ddots & \varepsilon \\ \vdots & & \ddots & \vdots \\ \varepsilon & \varepsilon & \cdots & d_n \end{bmatrix}, \mathbf{B}_0 = \begin{bmatrix} t_{0,1} \\ \varepsilon \\ \vdots \\ \varepsilon \end{bmatrix}, \mathbf{u}(k) = [u(k)].$$

Wyjście linii produkcyjnej opisane jest równaniem (9), tzn. $\mathbf{y}(k) = [y(k)]$, a macierz $\mathbf{C} = [\varepsilon \ \ldots \ \varepsilon \ t_{n,n+1}d_n]$.

3.2. Przykład

Rozważmy linię produkcyjną z szeregiem 3 stanowisk M_1, M_2 i M_3, przedstawioną na rys. 2. Czasy operacji dla poszczególnych stanowisk wynoszą odpowiednio: $d_1 = 3$, $d_2 = 2$ i $d_3 = 6$ jednostek czasu. Czasy transportu są następujące: $t_{0;1} = 1$, $t_{1;2} = 2$, $t_{2;3} = 0$, $t_{3;4} = 0$.



Rys. 2. Przykładowa sekwencyjna linia produkcyjna z szeregiem 3 stanowisk

Model linii produkcyjnej z rys. 2 opisany jest równaniami (13) i (9), gdzie:

$$\mathbf{x}(k) = \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix}, \ \mathbf{A}_0 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ t_{1,2}d_1 & \varepsilon & \varepsilon \\ \varepsilon & t_{2,3}d_2 & \varepsilon \end{bmatrix}, \ \mathbf{A}_1 = \begin{bmatrix} d_1 & \varepsilon & \varepsilon \\ \varepsilon & d_2 & \varepsilon \\ \varepsilon & \varepsilon & d_3 \end{bmatrix}, \ \mathbf{B}_0 = \begin{bmatrix} t_{0,1} \\ \varepsilon \\ \varepsilon \end{bmatrix},$$

$$\mathbf{u}(k) = \begin{bmatrix} u(k) \end{bmatrix}, \mathbf{y}(k) = \begin{bmatrix} y(k) \end{bmatrix}, \mathbf{C} = \begin{bmatrix} \varepsilon & \varepsilon & t_{3,4}d_3 \end{bmatrix}.$$

Po podstawieniu odpowiednich wartości liczbowych:

$$\mathbf{A}_0 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ 5 & \varepsilon & \varepsilon \\ \varepsilon & 2 & \varepsilon \end{bmatrix}, \mathbf{A}_1 = \begin{bmatrix} 3 & \varepsilon & \varepsilon \\ \varepsilon & 2 & \varepsilon \\ \varepsilon & \varepsilon & 6 \end{bmatrix}, \mathbf{B}_0 = \begin{bmatrix} 1 \\ \varepsilon \\ \varepsilon \end{bmatrix}, \mathbf{C} = \begin{bmatrix} \varepsilon & \varepsilon & 6 \end{bmatrix}.$$

4. Generator *Phoebe*

Generator *Phoebe* napisano w języku Python. Aby wygenerować model systemu produkcyjnego, tzn. kod, który może być użyty w pakietach obliczeniowych MathWorks Matlab lub GNU Octave, należy przygotować opis takiego systemu. Jako języka opisu rozważano kilka możliwości np. XML, JSON ostatecznie wybrano język YAML [2]. Po zaadoptowaniu YAML do opisu systemów produkcyjnych, opis wykorzystywany przez *Phoebe* podzielony jest na następujące sekcje:

- *input* – definiuje wejścia systemu produkcyjnego;

– *output* – definiuje wyjścia oraz
– *prod-unit* – definiuje stanowiska produkcyjne.

Dodatkowo możliwe jest dodanie sekcji *values* definiującej poszczególne wartości liczbowe. Dla każdego z elementów definiowane są czasy operacji (*op-time*), połączenia z innymi elementami (*connect*), czasy transportu (*tr-time*) i ewentualne bufory międzyoperacyjne (*buffers*) - domyślnie nieograniczone. Szczegóły przedstawiono poniżej.

4.1. Modelowanie sekwencyjnej linii produkcyjnej

Powróćmy do systemu produkcyjnego z rozdziału 3.2. Opis tego systemu w języku YAML przedstawia rys. 3.

```
---
input:
- u: {connect: M_1, tr-time: 't_{0,1}'
prod-unit:
- M_1: {op-time: d_1, connect: M_2, tr-time: 't_{1,2}'}
- M_2: {op-time: d_2, connect: M_3, tr-time: 't_{2,3}'}
- M_3: {op-time: d_3, connect: y, tr-time: 't_{3,4}'}
output:
- y: {}
values: {
't_{0,1}': 1, 't_{1,2}': 2, 't_{2,3}': 0, 't_{3,4}': 1,
d_1: 3, d_2: 2, d_3: 6
}
```

Rys. 3. Opis systemu produkcyjnego z rozdziału 3.2 w języku YAML

W opisie systemu zawarto:
Sekcja *input*: system posiada jedno wejście, nazwane *u* połączone z M_1, gdzie czas transportu z *u* do M_1, tzn. *tr-time* wynosi $t_{0,1.}$
Sekcja *prod-unit*: definiuje 3 maszyny – M_1, M_2 i M_3. Na M_1 realizowane są operacje o czasie trwania *op-time* równym d_1. Detale z M_1 po zakończonej operacji trafiają do M_2, czas transportu *tr-time* wynosi $t_{1,2}$. W podobny sposób zdefiniowano M_2. M_3 – operacje na tym zasobie trwają d_3 jednostek czasu. Następnie detal transportowany jest do *y* (czyli na wyjście) i trafia tam po czasie $t_{3,4}$.
Sekcja *output* definiuje wyjścia systemu, w tym przypadku jedno wyjście *y*.
W sekcji *values* określono wartości liczbowe poszczególnych czasów transportu i operacji.

Wynik działania *Phoebe* dla powyższych danych wejściowych przedstawia rys. 4. Można go bezpośrednio zapisać w pliku .m, tak aby mógł być przetwarzany przez wybrany pakiet obliczeniowy. Aby używać wygenerowanego kodu należy zainstalować *Max–Plus Algebra Toolbox for Matlab®* [7].

Po uruchomieniu kodu z rys. 4. w pakiecie obliczeniowym MathWorks Matlab lub GNU Octave uzyskuje się zachowanie systemu produkcyjnego w poszczególnych iteracjach, tzn. czasy startu poszczególnych operacji. Następnie, w łatwy sposób otrzymać można różnorakie ilościowe wskaźniki efektywności, będące pochodną realizacji procesów w czasie.

Zachowanie modelowanej linii produkcyjnej, dla poszczególnych *k*, przyjmując warunki początkowe $\mathbf{x}(0) = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \end{bmatrix}^T$, oraz $\forall k \in \mathbb{N}$: $\mathbf{u}(k) = 0$, przedstawia tabela 1.

```
clear
disp('x(k) = A0x(k) + A1x(k-1) + B0u(k)');
disp('y(k) =  Cx(k)');
disp('u(k) = [ u(k); ]');
disp('x(k) = [ x_1(k); x_2(k); x_3(k); ]');
disp('y(k) = [ y(k); ]');

U  = mp_ones(1, 1)
X0 = mp_zeros(3, 1)

t01 = 1; t12 = 2; t23 = 0; t34 = 1; d1 = 3; d2 = 2; d3 = 6;

A0 = mp_zeros(3, 3);
   A0(2, 1) = mp_multi(d1, t12); A0(3, 2) = mp_multi(d2, t23);
   A0
A1 = mp_zeros(3, 3);
   A1(1, 1) = d1; A1(2, 2) = d2; A1(3, 3) = d3;
   A1
B0 = mp_zeros(3, 1);
   B0(1, 1) = t01;
   B0
C = mp_zeros(1, 3);
   C(1, 3) = mp_multi(d3, t34);
   C
As = mp_star(A0)
A = mp_multi(As, A1)
B = mp_multi(As, B0)

k = 12;
X(:, 1) = mp_add(mp_multi(A, X0), mp_multi(B, U));
Y(1) = mp_multi(C, X(:, 1));
for i = 2:k
    X(:, i) = mp_add(mp_multi(A, X(:, i - 1)), mp_multi(B, U));
    Y(i) = mp_multi(C, X(:, i));
end
```

Rys. 4. Kod modelowanego systemu produkcyjnego wygenerowany przez *Phoebe*

Tab. 1: Wektor stanu oraz wyjście dla poszczególnych k dla przykładu 4.1

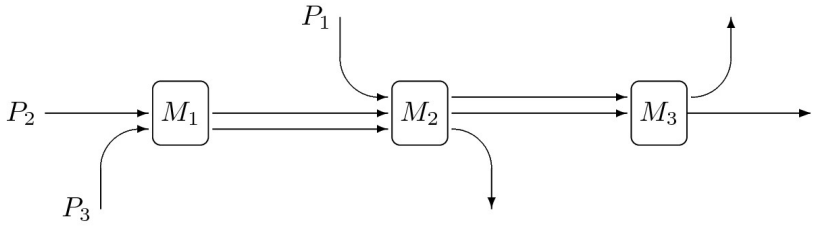
k	1	2	3	4	5	6	7	8	9	10	11	12
	1	4	7	10	13	16	19	22	25	28	31	34
$\mathbf{x}(k)$	6	9	12	15	18	21	24	27	30	33	36	39
	8	14	20	26	32	38	44	50	56	62	68	74
$\mathbf{y}(k)$	14	20	26	32	38	44	50	56	62	68	74	80

4.2. Modelowanie systemu produkcji wieloasortymentowej

Używając *Phoebe* możliwe jest modelowanie systemów produkcji wieloasortymentowej, jak np. przykładu zaczerpniętego z [1].

Rozważmy system produkcyjny składający się z 3 maszyn (M_1, M_2 i M_3). W systemie tym wytwarzane są trzy różne typy produktów (P_1, P_2 i P_3). Marszruty detali, dla poszczególnych typów produktów przedstawia rys. 5.

Detale typu P_1 z magazynu wejściowego trafiają na maszynę M_2 gdzie poddawane są obróbce, a następnie na maszynę M_3. Po zakończeniu obróbki na maszynie M_3 opuszczają system trafiając do magazynu wyjściowego. Detale typu P_2 dostarczane są na maszynę M_1 następnie na M_2 i ostatecznie na M_3. Detale typu P_3 przechodzą przez M_1 i M_2. Detale w systemie poruszają się na paletach. Zakłada się, że czasy transportu detali pomiędzy maszynami są pomijalne, oraz że brak jest czasów przełączeń maszyn, pomiędzy detalami różnych typów. Sekwencje poszczególnych detali na poszczególnych maszynach są znane. Na M_1 jest to (P_2, P_3), tzn. na M_1 przetwarzane są detale typu P_2 następnie typu P_3, następnie P_2, itd. Na M_2 sekwencja jest (P_1, P_2, P_3), i (P_1, P_2) na maszynie M_3. Sekwencje te nazwijmy *lokalnymi regułami dostępu* i oznaczmy jako σ_1 dla reguły dostępu do M_1, σ_2 dla M_2 i σ_3 dla M_3.



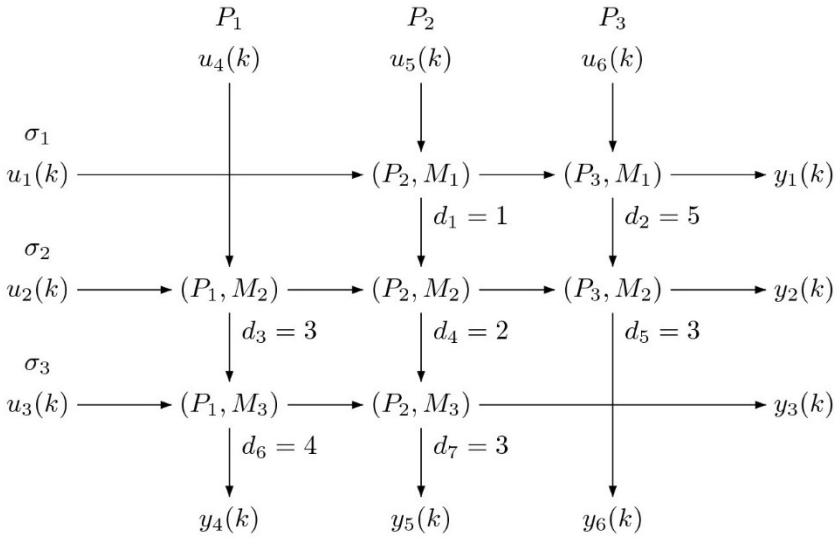
Rys. 5. Marszruty poszczególnych typów produktów

Informacje dotyczące poszczególnych reguł dostępu jak i czasów obróbki przedstawia rys. 5. Operacje na poszczególnych maszynach dla poszczególnych typów detali reprezentowane są poprzez pary typu (P_i, M_j), w tym przypadku oznaczającą obróbkę detalu typu P_i na maszynie M_j. Z każdą operacją związany jest czas operacji d_k, np. operacja 4 oznaczona jako (P_2, M_2), czyli obróbka P_2 na M_2 trwająca $d_4 = 2$ jednostki czasu.

Dla takich systemów, przed przygotowaniem opisu dla *Phobe* należy przyjrzeć się, co dzieje się w systemie w jednym cyklu procesu produkcyjnego. I tak zdefiniowano:

- $u_i(k)$ – chwila czasowa, w jakiej maszyna M_i jest dostępna do realizacji operacji, w k-tym cyklu produkcji, dla i = 1, 2 i 3, (tzn. w jakim reguła dostępu σ_i udostępnia maszynę);
- $u_j(k)$ – chwila czasowa, w jakiej detal P_{j-3} jest dostępny w magazynie wejściowych, w k-tym cyklu dla j = 4, 5, 6;
- $x_i(k)$ – chwila czasowa, w której i-ta operacja startuje w k-tym cyklu, dla i = 1, 2, …, 7;
- $y_i(k)$ – chwila czasowa, w której zakończyła się lokalna reguła dostępu dla k-tego cyklu produkcji, i = 1, 2 i 3;
- $y_j(k)$ – chwila czasowa, w której zakończono obróbkę detalu typu P_{j-3}, dla k-tego cyklu produkcyjnego, j = 4, 5, 6.

Bazując na powyższych założeniach, przedstawionych graficznie na rys. 6, wygenerowano opis systemu produkcyjnego w języku YAML – rys. 7.



Rys. 6. Sekwencje oraz czasy poszczególnych operacji

```
---
input:
- u_1: {connect: X_1}
- u_2: {connect: X_3}
- u_3: {connect: X_6}
- u_4: {connect: X_3}
- u_5: {connect: X_1}
- u_6: {connect: X_2}
prod-unit:
- X_1: {op-time: d_1, connect: {X_2, X_4}}
- X_2: {op-time: d_2, connect: {y_1, X_5}}
- X_3: {op-time: d_3, connect: {X_4, X_6}}
- X_4: {op-time: d_4, connect: {X_5, X_7}}
- X_5: {op-time: d_5, connect: {y_2, y_6}}
- X_6: {op-time: d_6, connect: {X_7, y_4}}
- X_7: {op-time: d_7, connect: {y_3, y_5}}
output:
- y_1: {connect: u_1}
- y_2: {connect: u_2}
- y_3: {connect: u_3}
- y_4: {}
- y_5: {}
- y_6: {}
values: {d_1: 1, d_2: 5, d_3: 3, d_4: 2, d_5: 3, d_6: 4, d_7: 3}
```

Rys. 7. Kod YAML dla systemu z przykładu 4.2

W porównaniu do opisu z poprzedniego przykładu widać w nim kilka różnic. Przede wszystkim, możliwość tworzenia pętli zwrotnych oraz rozwidlania przepływu. Dodatkowo w opisie nie zdefiniowano czasów transportu, oznacza to, że są one pomijalne.

Dla powyższych danych *Phebe* generuje kod modelu systemu przedstawiony na rys. 8. Realizację procesów po uruchomieniu tego kodu przedstawia tab. 2. Widać w niej, że np. detal typu P_3 w pierwszym takcie (tzn. dla $k = 1$) uzyska dostęp do M_1 w czasie $x_2 = 1$, następnie dostęp do M_2 uzyska w czasie $x_5 = 6$, system opuści w czasie $y_6 = 9$.

```
clear
disp('x(k)=A0x(k) + A1x(k-1) + B0u(k)');
disp('y(k)= Cx(k)');
disp('u(k)=[u_1(k);u_2(k);u_3(k);u_4(k);u_5(k);u_6(k);]');
disp('x(k)=[x_1(k);x_2(k);x_3(k);x_4(k);x_5(k);x_6(k);x_7(k);]');
disp('y(k)=[y_1(k);y_2(k);y_3(k);y_4(k);y_5(k);y_6(k);]');

U  = mp_ones(6, 1)
X0 = mp_zeros(7, 1)

d1 = 1; d2 = 5; d3 = 3; d4 = 2; d5 = 3; d6 = 4; d7 = 3;

A0 = mp_zeros(7, 7);
   A0(2, 1) = d1; A0(4, 1) = d1; A0(4, 3) = d3; A0(5, 2) = d2;
   A0(5, 4) = d4; A0(6, 3) = d3; A0(7, 4) = d4; A0(7, 6) = d6;
   A0
A1 = mp_zeros(7, 7);
   A1(1, 1) = d1; A1(1, 2) = d2; A1(2, 2) = d2; A1(3, 3) = d3;
   A1(3, 5) = d5; A1(4, 4) = d4; A1(5, 5) = d5; A1(6, 6) = d6;
   A1(6, 7) = d7; A1(7, 7) = d7;
   A1
B0 = mp_zeros(7, 6);
   B0(1, 1) = 0; B0(1, 5) = 0; B0(2, 6) = 0; B0(3, 2) = 0;
   B0(3, 4) = 0; B0(6, 3) = 0;
   B0
C = mp_zeros(6, 7);
   C(1, 2) = d2; C(2, 5) = d5; C(3, 7) = d7; C(4, 6) = d6;
   C(5, 7) = d7; C(6, 5) = d5;
   C
As = mp_star(A0)
A = mp_multi(As, A1)
B = mp_multi(As, B0)

k = 12;
X(:, 1) = mp_add(mp_multi(A, X0), mp_multi(B, U));
Y(1) = mp_multi(C, X(:, 1));
for i = 2:k
    X(:, i) = mp_add(mp_multi(A, X(:, i - 1)), mp_multi(B, U));
    Y(i) = mp_multi(C, X(:, i));
end
```

Rys. 8. Wynik działania *Phoebe* dla danych wejściowych z rys. 7

Tab. 2: Realizacja procesów z przykładu 4.2

k	1	2	3	4	5	6	7	8	9	10	11	12
	0	6	12	18	24	30	36	42	48	54	60	66
	1	7	13	19	25	31	37	43	49	55	61	67
	0	9	17	25	33	41	49	57	65	73	81	89
$\mathbf{x}(k)$	3	12	20	28	36	44	52	60	68	76	84	92
	6	14	22	30	38	46	54	62	70	78	86	94
	3	12	20	28	36	44	52	60	68	76	84	92
	7	16	24	32	40	48	56	64	72	80	88	96
	6	12	18	24	30	36	42	48	54	60	66	72
	9	17	25	33	41	49	57	65	73	81	89	97
	10	19	27	35	43	51	59	67	75	83	91	99
$\mathbf{y}(k)$	7	16	24	32	40	48	56	64	72	80	88	96
	10	19	27	35	43	51	59	67	75	83	91	99
	9	17	25	33	41	49	57	65	73	81	89	97

5. Wnioski

Modelowanie systemów produkcyjnych, ich analiza czy planowanie implikuje wiele skomplikowanych problemów analitycznych. W niniejszym artykule przedstawiono program *Phoebe*, użyteczny do automatycznego modelowania systemów produkcyjnych przy użyciu (max, +) algebry. Algebra (max, +) jest wygodnym narzędziem
do modelowania deterministycznych DES, w których czasy zdarzeń są stałe, stałe są również sekwencje operacji. Jest to również podstawowa wada tego narzędzia, niemniej posiada ona przewagę w postaci jednoznacznego modelu analitycznego, co pozwala na zrozumienie pewnych zjawisk, czy optymalizacje parametrów badanych systemów.

Prezentowany materiał jest wstępem do dalszych badan, przede wszystkim nad optymalna czy suboptymalna synteza poszczególnych systemów jednoasortymentowych, w system produkcji wieloasortymentowej.

Literatura

1. Baccelli F. i inni: Synchronisation and Linearity. An Algebra for Discrete Event Systems. Wiley, 1992. url: https://www.rocq.inria.fr/metalau/cohen/SED/book-online.html
2. Ben-Kiki O, Evans C., dot Net I. *YAML Ain't Markup Language*. Version 1.2. 2009. url: http://www.yaml.org/spec/1.2/spec.html
3. Cassandras C. G., Lafortune S.: Introduction to Discrete Event Systems. Springer, 2007.
4. Cuninghame–Green R.: Minimax Algebra. Lecture Notes in Economics and Mathematical Systems, vol. 166. Springer-Verlag, 1979.
5. Heidergott B., Olsder G.J., van der Woude J.: Max Plus at Work: Modeling and Analysis of Synchronized Systems. A Course on Max-Plus Algebra and Its Applications. Princeton University Press, 2005.
6. Seleim A., El Maraghy H.: Max-Plus Modeling of Manufacturing Flow Lines. 47CIRP Conference on Manufacturing Systems (CMS2014). Vol. 17, 2014, 302–307.
7. Stańczyk J.: Max–Plus Algebra Toolbox for Matlab. Ver.1.7, 2016. url: http://gen.up.wroc.pl/stanczyk/mpa/
8. Stańczyk J.: Phoebe. Generator of max-plus algebraic state space description. Ver.0.6, 2018. url: http://gen.up.wroc.pl/stanczyk/
9. Stańczyk J.: Zastosowanie max-plus algebry w modelowaniu i analizie efektywności systemów produkcyjnych. Zarzadzanie Przedsiębiorstwem, Vol.3, 2017, 46–53. url: http://www.zp.ptzp.org.pl/wp-content/uploads/2017/10/17_3_7.pdf.

10. Stańczyk J., Mayer E., Raisch J.: Modelling and Performance Evaluation of DES – a Max-Plus Algebra Toolbox for Matlab". Int. Conf. Informatics in Control, Automation and Robotics, ICINCO'04. Vol. 3. 2004, 270–275.

Dr inż. Jarosław STAŃCZYK
Katedra Genetyki
Uniwersytet Przyrodniczy we Wrocławiu
51-631 Wrocław, ul. Kożuchowska 7
tel.: (0-71) 774 01 58
e – mail: jaroslaw.stanczyk@upwr.edu.pl