Marcin RADOM[1,2], Piotr FORMANOWICZ[1]
[1]Instytut Informatyki, Politechnika Poznańska
[2]Instytut Chemii Bioorganicznej, Polska Akademia Nauk

ALGORYTMICZNE ASPEKTY ROZSZERZONYCH CZASOWYCH SIECI PETRIEGO*

Streszczenie. Sieci Petriego są popularnym narzędziem do modelowania złożonych systemów biologicznych. Systemy te zazwyczaj charakteryzują się wysoką liczbą powiązań między swoimi elementami składowymi. Bardzo istotnym elementem opisującym wiele aspektów takich systemów jest czas. Może to być zarówno czas do zainicjowania oraz trwania reakcji chemicznych, czy też czas życia substancji wchodzących w skład działającego układu. W niniejszym artykule proponowany jest nowy rodzaj sieci czasowej, umożliwiającej tworzenie modeli unifikujących wymienione dane czasowe. Przedstawione zostaną dwa przykładowe algorytmy wykorzystujące informacje związane z czasem, dzięki którym można uzyskiwać dodatkową wiedzę o modelu opartym na proponowanej sieci.

ALGORITHMIC ASPECTS OF EXTENDED TIME PETRI NETS

Summary. Petri nets are a popular tool for modeling complex biological systems. These systems are usually characterized by a high number of connections between their components. A very important element that describes many aspects of such systems is time. This can be both the time to initiate and the duration of chemical reactions, or the lifetime of the substances that make up a biological system. In this paper, a new kind of time net is proposed, thanks to which it is possible to create models unifying the above mentioned time data. Two algorithms using time-related information are presented, which can provide additional knowledge about the model based on the proposed net.

1. Wstęp

Sieci Petriego są jednym z wielu narzędzi używanych w modelowaniu złożonych systemów. Dotyczy to zarówno złożonych systemów przemysłowych, ale również nie mniej skomplikowanych systemów biologicznych. Zwłaszcza te ostatnie często charakteryzują się niezwykle wysokim stopniem skomplikowania z uwagi na liczbę elementów biochemicznych tworzących tego typu system, jak i bardzo gęstą siatkę powiązań pomiędzy nimi, reprezentującą wzajemne interakcje elementów systemu w komórkach, tkankach, itp. Analiza systemów biologicznych oparta na sieciach Petriego umożliwia zarówno uporządkowanie wiedzy o danym systemie, jak i wyciąganie nowych, cieka-

*Praca częściowo sfinansowana ze środków statutowych Politechniki Poznańskiej.

wych wniosków o jego działaniu [1]. Samo uzyskanie modeli ilościowych systemów biologicznych, na przykład czasowych, z modeli jakościowych nie jest kwestią trywialną, zagadnienie to jest rozważane między innymi w pracy [6].

Istnieje bardzo wiele rodzajów sieci Petriego, np. czasowe, stochastyczne lub ciągłe. Umożliwiają one wprowadzanie do modelu dodatkowych danych uzyskanych z innych badań nad danym systemem biologicznym. Analiza takich sieci może dostarczać ciekawych i nowych informacji o naturze modelowanego systemu [5, 3]. Należy zauważyć, że systemy biologiczne mogą być wyjątkowo skomplikowane. Poza dużą liczbą ich elementów składowych i ich wzajemnymi interakcjami, istotny w nich jest też czynnik czasu. Nowe związki biochemiczne muszą powstać i dojrzeć, a reakcje chemiczne, które odpowiadają za ich tworzenie zazwyczaj nie są natychmiastowe. Klasyczne sieci Petriego nie są w stanie modelować tego typu czynników jak czas czy prawdopodobieństwo zachodzenia reakcji chemicznych. Rozwiązaniem są różne rodzaje czasowych sieci Petriego. Jedna z pierwszych tego rodzaju sieci [4] zaoferowała tranzycje z przypisanym przedziałem czasu, określającym kiedy mogą one zostać uruchomione (ang. *Time Petri nets, TPN*). Kolejny rodzaj sieci czasowej definiuje ustalony czas trwania uruchomienia tranzycji (ang. Duration-time Petri nets, DPN). Jeszcze innym rodzajem sieci czasowych są takie, w których do miejsc przypisany jest przedział czasu określający, kiedy dokładnie tokeny z takiego miejsca mogą aktywować tranzycje [9]. Główny problem na jaki można natrafić przy tworzeniu modeli opartych na tego typu sieciach jest taki, że dane czasowe uzyskane z badań nad systemami biologicznymi mogą być niepełne, czasem wręcz sprzeczne oraz dostępne w wielu formach - zarówno jako zakresy czasu, czasy trwania reakcji, czy czasy życia elementów systemu. Takie bardzo konkretne sieci czasowe doskonale sprawdzają się w modelach przemysłowych, gdzie tak precyzyjne dane mogą być łatwiej i z większą precyzją uzyskane.

W niniejszej pracy proponowany jest nowy rodzaj sieci, zwany *rozszerzoną siecią czasową* (jej szerszy opis znajduje się w artykule [8]). Zawiera ona cechy trzech znanych sieci czasowych: sieci typu TPN z przedziałami czasu aktywacji tranzycji, sieci DPN (w zmodyfikowanej formie), które pozwalają określić czas trwania produkcji tokenów przez tranzycje oraz sieci czasowe, w których zakres czasu związany jest z miejscami. Użycie trzech przedziałów czasowych, dwóch dla tranzycji oraz jednego dla miejsc jest o tyle przydatne, że pozwala w modelu zawrzeć zarówno przybliżone, jak i dokładne wartości czasu. Wartości dokładne są możliwe do wprowadzenia do modelu w przypadku zrównania zakresów danego przedziału czasu. Z punktu widzenia modelowania systemu biologicznego tego typu sieć pozwoli określić zarówno czas do zainicjowania każdej reakcji, jak i czas jej trwania, a także umożliwia modelowanie czasu życia biochemicznych elementów systemu. Nie mniej istotny jest fakt, że modele zbudowane w oparciu o nowy rodzaj sieci mogą zawierać różne rodzaje danych czasowych równocześnie - mogą być to zakresy lub konkretne wartości. Nie jest wtedy potrzebna transformacja posiadanych danych do konkretnego formatu, jak miałoby to miejsca dla "czystych" modeli opartych tylko na sieciach TPN lub DPN.

Struktura niniejszej pracy jest następująca: w Rozdziale 2 przedstawiona zostanie koncepcja nowej czasowej sieci Petriego. W Rozdziale 3 zostaną przedstawione dwa algorytmy wykorzystujące dane czasowe opisujące elementy sieci. Ostatni rozdział zawiera podsumowanie możliwości, jakie daje nowy rodzaj sieci czasowej.

2. Rozszerzona czasowa sieć Petriego

W tej części rozważana jest koncepcja rozszerzonej czasowej sieci Petriego ($xTPN$), która łączy w sobie trzy oddzielnie występujące w literaturze sieci czasowe: TPN, DPN oraz sieci z tak zwanym oknem czasowym (por. [8]). Formalnie, rozszerzona czasowa sieć Petriego bez stanu określona jest Definicją 1.

Definicja 1. *Rozszerzona czasowa sieć Petriego bez stanu.*
Rozszerzona czasowa sieć Petriego bez stanu jest zbiorem $\mathcal{Z} = \{P, T, F, V, I, J\}$*, gdzie:*
P *oraz* T *to skończone, niepuste i rozłączne zbiory odpowiednio miejsc i tranzycji,*
$F \subseteq (P \times T) \cup (T \times P)$ *jest zbiorem łuków,*
$V : F \to \mathbb{N}$ *jest funkcją przypisującą wagi łukom sieci,*
$I : T \to \mathbb{Q}_0^+ \times \{\mathbb{Q}_0^+ \cup \{\infty\}\} \times \mathbb{Q}_0^+ \times \{\mathbb{Q}_0^+ \cup \{\infty\}\}$ *oraz dla każdego* $t \in T$*, gdy* $I(t) = (I_1(t), I_2(t), I_3(t), I_4(t))$ *spełnione są nierówności* $I_1(t) \leqslant I_2(t)$ *i* $I_3(t) \leqslant I_4(t)$*,*
$J : P \to \mathbb{Q}_0^+ \times \{\mathbb{Q}^+ \cup \{\infty\}\}$ *oraz dla każdego* $p \in P$*, gdy* $J(p) = (J_1(p), J_2(p))$ *spełniona jest nierówność* $J_1(p) < J_2(p)$*.*

Podzbiór elementów $N = \{P, T, F, V\}$ jest klasyczną siecią Petriego bez stanu. Funkcja I przypisuje tranzycjom dwie pary domkniętych przedziałów czasu, których granice określone są odpowiednio jako $[I_1(t), I_2(t)]$ oraz $[I_3(t), I_4(t)]$. Pierwszy z nich: $[I_1(t), I_2(t)]$ określa odpowiednio minimalny oraz maksymalny czas, po którym aktywna tranzycja musi zostać uruchomiona. Dalej wartości te będą zapisywane jako $I_1(t_i) = \alpha_{t_i}^L$ oraz $I_2(t_i) = \alpha_{t_i}^U$. Spełniają one nierówność $\alpha_{t_i}^L \leqslant \alpha_{t_i}^U$ dla każdej tranzycji $t_i \in T$. W literaturze poświęconej sieciom czasowym typu TPN oznaczane są one często jako $eft(t)$ (ang. *earliest firing time*, odpowiednik $\alpha_{t_i}^L$) oraz $lft(t)$ (ang. *latest firing time*, odpowiednik $\alpha_{t_i}^U$).

Drugi zakres $[I_3(t), I_3(t)]$ określa minimalny i maksymalny czas produkcji tokenu dla tranzycji, która po swojej aktywacji została uruchomiona. Zakresy będą dalej oznaczane jako $I_3(t_i) = \beta_{t_i}^L$ oraz $I_4(t_i) = \beta_{t_i}^U$ dla każdej tranzycji $t_i \in T$.

Jak widać połączone zostały tu elementy sieci TPN oraz, w nieco zmodyfikowanej wersji, DPN. Należy podkreślić dwie różnice w stosunku do sieci DPN. Po pierwsze, rozszerzona sieć czasowa nie wymusza konieczności uruchamiania tranzycji natychmiast po aktywacji, jak ma to miejsce w sieciach DPN. Uruchomienie tranzycji następuje po upływie pewnego czasu $\tau_a(t)$ takiego, że $0 \leqslant \alpha_t^L \leqslant \tau_a(t) \leqslant \alpha_t^U$. Po drugie, czas produkcji tokenów nie jest deterministyczny, lecz jest określony poprzez domknięty przedział określający minimalny i maksymalny czas produkcji. Po upływie czasu liczonego od momentu aktywacji i oznaczanego dalej jako $\tau_a(t)$, tranzycja pobiera tokeny ze swoich miejsc wejściowych (tak jak w sieci DPN), jednak wyprodukowanie tokenów w miejscach wyjściowych następuje dopiero po upływie pewnego czasu $\tau_b(t)$, liczonego od momentu rozpoczęcia produkcji, takiego, że $0 \leqslant \beta_t^L \leqslant \tau_b(t) \leqslant \beta_t^U$. Aby zredukować zachowanie rozszerzonej sieci czasowej w tym aspekcie do sieci DPN należałoby przyjąć $\beta_t^L = \beta_t^U$.

Funkcja J przypisuje do każdego miejsca p przedział domknięty $[J_1(p), J_2(p)]$, określający czas przebywania indywidualnego tokenu w danym miejscu. Zakresy dalej będą oznaczane jako $J_1(p_j) = \gamma_{p_j}^L$ oraz $J_2(p_j) = \gamma_{p_j}^U$. Wartość γ_p^L określa czas, który musi upłynąć osobno dla każdego istniejącego w danym miejscu tokenu, aby był on zdolny do aktywowania tranzycji wyjściowych tego miejsca. Całkowity dopuszczalny czas życia tokenu w danym miejscu p trwa od 0 do γ_p^U. Czas w rozważanej sieci płynie tak

samo dla każdego jej elementu. Gdy upływ czasu dla tokenu spowoduje przekroczenie γ_p^U, a token do tego momentu nie został użyty do produkcji, jest on natychmiast usuwany z miejsca p. Jest to aspekt, który nie występuje w klasycznych sieciach Petriego, w których token jest usuwany tylko wtedy, gdy pobiera go uruchamiana tranzycja.

Z uwagi na obszerność wymaganych definicji określających stany sieci, rozszerzona sieć czasowa ze stanem zostanie tutaj opisana w sposób uproszczony. W ogólności pełny stan sieci $\mathcal{Z}$ składa się z tak zwanego p-stanu oraz t-stanu. Pierwszy określa przypisanie tokenów do każdego miejsca sieci. Drugi z nich (t-stan) jest opisany parą wartości (u_{t_i}, w_{t_i}), które muszą mieścić się w zakresach opisanych funkcją I. Pierwsza z wartości to czas jaki upłynął od momentu aktywacji tranzycji, druga to czas jaki upłynął od momentu, kiedy tranzycja rozpoczęła fazę produkcji tokenów. Oba opisane tu pokrótce stany składowe tworzą pełen stan sieci w danym momencie czasu.

Definicja 1 określa, że do każdego miejsca przypisane są wartości $J_1(p_j) = \gamma_{p_j}^L$ oraz $J_2(p_j) = \gamma_{p_j}^U$. Druga z nich to maksymalny czas życia tokenów w miejscu. Zakładamy, że każdy nowo utworzony token zaczyna z czasem równym zero. Oznacza to, że stan każdego miejsca określa pewien specyficzny multizbiór K, którego elementy są liczbami κ_x takimi, że $0 \leqslant \kappa_x \leqslant \gamma_{p_j}^U$, gdzie x to indeks konkretnego tokenu. Każda z tych liczb należących do K określa więc aktualny czas życia tokenu, od momentu jego powstania w danym miejscu. Multizbiór tego typu opisuje Definicja 2.

Definicja 2. *Multizbiór K.*
Niech ${}_b^a K$*, gdzie* $a > 0$ *oraz* $0 \leqslant b \leqslant a$*, będzie multizbiorem nieujemnych liczb wymiernych lub zawiera on tylko symbol* $\#$*. Elementy takiego multizbioru to wartości z przedziału domkniętego* $[b, a]$*, co zapisujemy jako:* $\kappa_x \in {}_b^a K$ *jeżeli* $\kappa_x \in \mathbb{Q}_0^+ \wedge 0 \leqslant b \leqslant \kappa_x \leqslant a$*.*

Każde miejsce rozszerzonej sieci czasowej jest w każdym momencie czasu opisane multizbiorem ${}_b^a K_{p_j}$, $a = \gamma_{p_j}^U$, $b = 0$. Wartość a to maksymalny czas życia tokenu w miejscu p_j, natomiast b określa czas przypisywany tokenowi, który zostaje wyprodukowany w p_j. Należy zaznaczyć, że wartość b z definicji 2 niekoniecznie musi być zerem, oraz do określenia pewnych cech sieci niektóre multizbiory K mogą zawierać zamiast liczb określających tokeny, specjalny symbol $\#$. Odpowiednie definicje wykorzystujące te właściwości wykraczają jednak poza zakres niniejszego artykułu.

W celu ustalenia, czy dana tranzycja jest aktywna, należy dokładniej przyjrzeć się elementom multizbioru K. Załóżmy dla przykładu, że miejsce p_j jest jedynym miejscem wejściowym tranzycji t_i z którą łączy je łuk o wadze $V(p_j, t_i) = 3$, $\gamma_{p_j}^L = 2$, ${}_0^9 K_{p_j} = \{0, 0, 1, 2, 3, 3\}$. Tranzycja t_i, której jedynym miejscem wejściowym jest p_j, jest aktywna, ponieważ istnieje pewien podzbiór multizbioru ${}_0^9 K_{p_j}$, zwany podzbiorem aktywującym. Aby t_i była aktywna, podzbiór ten musi posiadać przynajmniej $V(p_j, t_i)$ wartości z ${}_0^9 K_{p_j}$, z których każda być nie mniejsza niż $\gamma_{p_j}^L = 2$. Podzbiór aktywujący oznaczamy jako ${}_2^9 K_{p_j}^{akt} = \{2, 3, 3\}$ oraz $|{}_2^9 K_{p_j}^{akt}| \geqslant 3$. Prostszymi słowy tranzycja t_i jest aktywna, jeżeli w multizbiorze ${}_0^9 K_{p_j}$ istnieje w danej chwili podzbiór przynajmniej 3 tokenów (waga łuku) o czasie życia większym lub równym $\gamma_{p_j}^L = 2$.

3. Podstawowe algorytmy dla rozszerzonej sieci czasowej

Wartości określające przedziały czasu, które opisują tranzycje oraz miejsca sieci, umożliwiają stworzenie wielu przydatnych algorytmów weryfikujących wzajemne relacje tych wartości. Może to pomagać w tworzeniu lepszych i bardziej precyzyjnych modeli opartych na nowej sieci. W tej części przedstawione zostaną dwa podstawowe algorytmy wykorzystujące wartości czasu przypisane do elementów sieci. Należy podkreślić, że są to podstawowe wersje algorytmów, co będzie widoczne w opisanych dla nich ograniczeniach i założeniach wstępnych. Rozszerzone wersje działające dla dowolnej struktury proponowanej sieci czasowej są przedmiotem dalszych prac. Tym niemniej przedstawione tutaj propozycje pokazują znaczny potencjał analityczny nowego rodzaju sieci.

Należy zauważyć, że algorytmy te działają na liczbach naturalnych, podczas gdy w Definicji 1 używane są liczby wymierne nieujemne. Można przyjąć bez utraty dokładności modelu, że granice przedziałów przypisane do tranzycji są liczbami naturalnymi [7]. Jest to możliwe, ponieważ każdy zbiór liczb wymiernych nieujemnych przypisanych do elementów sieci można przekształcić na liczby naturalne przez pomnożenie ich przez najmniejszą wspólną wielokrotność mianowników wszystkich wartości wymiernych. Na przykład, jeśli dla miejsca p istnieją liczby $\alpha_p^L = 0.75$ i $\alpha_p^U = 2.5$, można je pomnożyć przez 4 będące najmniejszą wspólną wielokrotnością dwóch mianowników liczb $0.75 = 3/4$ i $2.5 = 5/2$, czyli $lcm(4, 2) = 4$. Odpowiednimi liczbami naturalnymi opisującymi przedział są wtedy $\alpha_p^L = 3$ i $\alpha_p^U = 10$). W takiej postaci algorytmy są prostsze do opisania, a wyniki da się skalować do wartości oryginalnych.

3.1. Określenie maksymalnej liczby tokenów dla miejsc

W sieci xTPN możliwe jest określenie przybliżonej maksymalnej liczby tokenów, które jednocześnie mogą przebywać w tym samym miejscu. Co więcej, każde miejsce z przedziałem czasu, którego górny zakres nie jest określony w nieskończoności, jest niejako z definicji k-ograniczone. Tak zwana k-ograniczoność dla miejsca w sieci Petriego (gdzie k to maksymalna liczba tokenów w miejscu) jest ważną własnością, mającą liczne zastosowania analityczne [2]. W kontekście systemów biologicznych można wyobrazić sobie wiele sytuacji, zwłaszcza na etapie tworzenia modelu, w których znajomość choćby przybliżonej górnego ograniczenia dla liczby tokenów będzie przydatna. Z dostępnej wiedzy o systemie może wynikać, że niektóre jego elementy są zazwyczaj dostępne w nadmiarze. Model takiego systemu powinien to uwzględnić, co można zbadać poprzez określenie k-ograniczoności. Odwrotnie, niektóre elementy systemu mogą być bardzo rzadkie, dlatego wysoka granica k-ograniczoności może wskazywać na problemy w konstrukcji modelu systemu. Co istotne, przedstawiony w niniejszym podrozdziale algorytm umożliwia zbadanie tej właśności z pewnym przybliżeniem, jednak bez potrzeby analizy przestrzeni stanów.

Opisana tutaj jest nieco uproszczona wersja algorytmu, która nie bierze pod uwagę tranzycji wyjściowych miejsca, dla którego oblicza maksymalną możliwą liczbę tokenów. Algorytm może więc zwracać wyższe maksymalne wartości niż faktycznie wynikające z całościowej analizy przestrzeni stanów takiej sieci. Należy również zauważyć, że algorytm zakłada zarówno maksymalnie szybki czas aktywacji i uruchomienia tranzycji wejściowych badanego miejsca, a także to, że każda tranzycja wejściowa jest zawsze aktywna. Może to zawyżyć na wyniku, wiadomo jednak, że rzeczywista liczba

tokenów w miejscu w dowolnym stanie sieci nigdy nie przekroczy wartości określonej przez proponowany algorytm. Na wstępie należy określić funkcję pomocniczą $fast(t)$.

Definicja 3. *Funkcja czasowa* $fast(t)$.
Dla każdej tranzycji $t_i \in T$ *rozszerzonej sieci czasowej* $\mathcal{Z} = \{P, T, F, V, I, J\}$ *dana jest funkcja* $fast(t_i) = \alpha^L_{t_i} + \beta^L_{t_i}$. *Funkcja ta określa najkrótszy możliwy czas, który musi upłynąć, aby aktywna tranzycja zakończyła fazę produkcji.*

Należy podkreślić, że algorytm przyjmuje deterministyczny, najszybszy możliwy czas do wyprodukowania tokenów przez tranzycję dany przez $fast(t_i)$.
Dane wejściowe dla Algorytmu 1 są następujące:

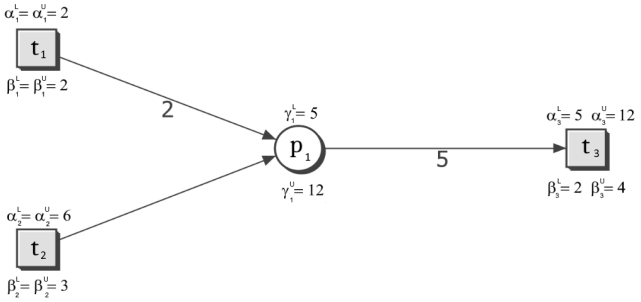
- Zbiór wszystkich tranzycji wejściowych miejsca p, oznaczony jako ${}^\bullet p$.
- Wartość γ^U_p określająca maksymalny czas życia tokenu w miejscu p.
- Wagi łuków $V(t_i, p)$, gdzie $t_i \in {}^\bullet p$.

Algorytm 1 Wyznaczanie maksymalnej liczby tokenów dla miejsca

1: $currentTime = 0$, $result = 0$, $K_p \leftarrow \emptyset$
2: **for** $(i = 1$ **to** $lcm(fast(t_i) \ : \ t_i \in {}^\bullet p) + \gamma^U_p)$ **do**
3: $\quad currentTime = currentTime + 1$
4: $\quad updateTokensTimeInPlace(p, +1)$
5: $\quad$ **for** $(\forall t_i \in {}^\bullet p \ : \ currentTime \ MOD \ fast(t_i) = 0)$ **do**
6: $\quad\quad produceTokensInPlace(p, +V(t_i, p))$
7: $\quad$ **end for**
8: $\quad$ **if** $(|K_p| > result)$ **then**
9: $\quad\quad result = |K_p|$
10: $\quad$ **end if**
11: **end for**

Najistotniejszym parametrem jest liczba kroków w głównej pętli algorytmu (innymi słowy ile kolejnych kroków/stanów Algorytm 1 musi przeanalizować). Bazując na przykładzie sieci z Rysunku 1 należy odpowiedzieć na pytanie, po ilu krokach możliwe jest uzyskanie największej zgromadzonej jednocześnie liczby tokenów w miejscu p. Jak było już wyjaśnione przy opisie założeń dla algorytmu, tranzycja wyjściowa t_3 z Rysunku 1 będzie ignorowana.

Dla każdej tranzycji zakłada się deterministyczny czas od aktywacji do produkcji równy $fast(t_i)$. Należy zauważyć, że w pewnych momentach czasu wszystkie tranzycje wejściowe danego miejsca uruchomić się mogą równocześnie. Ten moment to najmniejsza wspólna wielokrotność czasów określonych przez $fast(t_i)$ dla wszystkich $t_i \in {}^\bullet p$, określona jako $lcm(fast(t_i) \ : \ t_i \in {}^\bullet p)$. Po upływie od tego momentu czasu γ^U_p wszystkie wyprodukowane wtedy tokeny osiągną swój maksymalny dopuszczalny czas życia w analizowanym miejscu. Suma z linii 2 stanowi maksymalną liczbę kroków algorytmu. Najpóźniej w tym momencie miejsce posiadać może równocześnie maksymalną możliwą przy wcześniejszych założeniach liczbę tokenów. Tranzycje wyjściowe dla p są one ignorowane, w związku z czym nie zabierają tokenów. Tokeny znikają z miejsca tylko na skutek starzenia się. W linii 4 jest odwołanie do funkcji, której zadaniem jest zwiększenie wartości czasowych wszystkich tokenów istniejących w danym momencie w miejscu



τ:	Tokeny:	#	τ:	Tokeny:	#
1	-	0	● 25	1,1,5,5,7,9,9	7
2	-	0	● 26	2,2,6,6,8,10,10	7
3	-	0	● 27	0,3,3,7,7,9,11,11	8
4	0,0	2	● 28	0,0,1,4,4,8,8,10,12,12	10
5	1,1	2	● 29	1,1,2,5,5,9,9,11	8
6	2,2	2	● 30	2,2,3,6,6,10,10,12	8
7	3,3	2	31	3,3,4,7,7,11,11	7
8	0,0,4,4	4	● 32	0,0,4,4,5,8,8,12,12	9
9	0,1,1,5,5	5	● 33	1,1,5,5,6,9,9	7
10	1,2,2,6,6	5	● 34	2,2,6,6,7,10,10	7
11	2,3,3,7,7	5	● 35	3,3,7,7,8,11,11	7
12	0,0,3,4,4,8,8	7	● 36	0,0,0,4,4,8,8,9,12,12	10
13	1,1,4,5,5,9,9	7	● 37	1,1,1,5,5,9,9,10	8
● 14	2,2,5,6,6,10,10	7	● 38	2,2,2,6,6,10,10,11	8
● 15	3,3,6,7,7,11,11	7	● 39	3,3,3,7,7,11,11,12	8
● 16	0,0,4,4,7,8,8,12,12	9	40	0,0,4,4,4,8,8,12,12	9
● 17	1,1,5,5,8,9,9	7	● 41	1,1,5,5,5,9,9	7
● 18	0,2,2,6,6,9,10,10	8	● 42	2,2,6,6,6,10,10	7
● 19	1,3,3,7,7,10,11,11	8	● 43	3,3,7,7,7,11,11	7
● 20	0,0,2,4,4,8,8,11,12,12	10	● 44	0,0,4,4,8,8,8,12,12	9
● 21	1,1,3,5,5,9,9,12	8	● 45	0,1,1,5,5,9,9,9	8
22	2,2,4,6,6,10,10	7	● 46	1,2,2,6,6,10,10,10	8
● 23	3,3,5,7,7,11,11	7	● 47	2,3,3,7,7,11,11,11	8
● 24	0,0,4,4,6,8,8,12,12	9	● 48	0,0,3,4,4,8,8,12,12,12	10

Rys. 1. Przykład sieci xTPN dla problemu maksymalnej liczby tokenów z rozpisanymi wszystkimi multizbiorami K dla zadanego przedziału czasu od 1 do 48. Kolumna τ oznacza konkretną chwilę czasu, druga kolumna zawiera liczby multizbioru K reprezentującego tokeny, ostatnia kolumna oznacza liczbę tokenów. Komórki w pierwszej kolumnie z szarym tłem oznaczają czas, w którym powstały nowe tokeny. Czarne koło oznacza istnienie podzbioru aktywującego dla tranzycji t_3 w miejscu p_1 w chwili τ.

p oraz usunięcie tych, które po zwiększeniu swojej wartości czasowej przekraczają wartość graniczną dla tego miejsca, tj. γ_p^U. Wewnętrzna pętla algorytmu w liniach 5-7 działa tylko dla tych tranzycji, które w danym czasie ($currentTime$) produkują tokeny. Są to te kroki głównej pętli określone przez $currentTime$ dla których reszta z dzielenia ich przez $fast(t_i)$ jest równa zeru. Tylko w tych momentach czasu, przy przyjętych założeniach, t_i kończy produkcję. Funkcja w linii 6 liczy sumę wyprodukowanych w danej chwili tokenów w miejscu p. Dodawane jest do aktualnej sumy dokładnie tyle tokenów, ile wynosi waga łuku $V(t_i, p)$.

W przykładzie z Rysunku 1 chwile czasu τ, w których następuje produkcja tokenów to $4, 8, 9, 12, 16, 18, 20$ itd., czyli wielokrotności $fast(t_1) = 4$ oraz $fast(t_2) = 9$. W przykładzie najmniejsza wspólna wielokrotność ma wartość: $lcm(4, 9) = 36$. Rysunek 1 zawiera wypisane wszystkie wartości multizbioru ${}_0^{12}K_p$ (reprezentujące tokeny) w czasie od 0 do 48. Maksymalna liczna tokenów istniejących równocześnie w p nigdy nie przekracza 10. W omawianym przykładzie jest ona osiągana w czasie τ równym 20, 28, 36 oraz 48.

Należy ponownie podkreślić, że Algorytm 1 nie uwzględnia tranzycji pobierających tokeny z miejsca p, a ich uwzględnienie rozbudowałoby schemat postępo-

wania. Należałoby przyjąć dla takich tranzycji pewne założenia, które niekoniecznie musiałyby być zawsze prawdziwe. Uwzględniać musiałyby one średni (lub minimalny/maksymalny) czas produkcji, a dodatkowo takie tranzycje mogłyby dodatkowo zależeć od tokenów w zupełnie innych miejscach. Podsumowując, bez dokładnej analizy stanów danej sieci, nie da się inaczej niż w sposób przybliżony określić górnej granicy liczby tokenów w każdym miejscu. Mimo tego znajomość wartości przybliżonej stanowić może istotną informację dla dalszych analiz rozważanej sieci.

Złożoność czasowa algorytmu to $\mathcal{O}(n^2)$, przy czym należy zauważyć, że najwięcej powtórzeń niemal zawsze będzie mieć pętla zewnętrzna (liczba głównych kroków, w których analizowane jest miejsce i jego tokeny). Wewnętrzna pętla działa tylko dla tranzycji wejściowych i tylko w tych krokach, w których według przyjętych założeń mają one wyprodukować tokeny. Rozwinięcie algorytmu o analizę tranzycji wyjściowych wprowadziłoby dodatkowe pętle wewnętrzne, jednak nie byłyby one bardziej zagnieżdżone niż już istniejąca, stąd złożoność takiego algorytmu pozostałaby $O(n^2)$.

3.2. Określenie warunków żywotności tranzycji

Jedną z cech charakterystycznych proponowanej sieci jest istnienie związku pomiędzy wartościami czasowymi określonymi, np. przez funkcję $fast(t_i)$ dla tranzycji $t_i \in {}^\bullet p$, wartościami γ_p^L oraz γ_p^U miejsca p oraz tym czy t_i, dla której $p \in {}^\bullet t_i$ będzie miała szansę na uruchomienie. Notacja ${}^\bullet t_i$ oznacza zbiór wszystkich miejsc wejściowych dla tranzycji t_i. Tranzycja t_3 z Rysunku 1 jest tranzycją wyjściową, jednak działa ona na tych samych zasadach co inne tranzycje sieci. Różnica jest tylko taka, że po zakończeniu fazy produkcji nie tworzy ona nowych tokenów. Tranzycja t_3 w pewnym momencie może się aktywować, co wymaga pięciu tokenów w miejscu p_1 o odpowiednio długim czasie życia (jest to tzw. *okno uruchomienia*). Okno uruchomienia dla t_3 to czas, w którym w miejscu p_1 musi zaistnieć podzbiór aktywujący, albo prostszymi słowy w multizbiorze ${}^{12}_{0}K_{p_1}$ musi być przynajmniej 5 elementów, o wartościach równych przynajmniej $\gamma_{p_1}^L = 5$.

W tabeli pod Rysunkiem 1 widać, że w chwili $\tau = 14$ jest już 5 takich tokenów (o wartościach $5, 6, 6, 10, 10$). Pomimo zmieniania się wartości czasowych tokenów, łącznie do chwili $\tau = 21$ istnieje podzbiór aktywujący. Okno uruchomienia t_3 od momentu aktywacji jest mniejsze niż maksymalny czas aktywacji $\alpha_{t_3}^U = 12$, trwa ono 7 jednostek czasu. W chwili $\tau = 22$ tranzycja t_3 przestanie być aktywna, jeżeli nie uruchomi się do chwili $\tau = 21$ włącznie. Precyzyjniej, jej pierwsze "okno uruchomienia" trwa od 14 do 21 jednostki czasu, ponieważ w tym przedziale miejsce p_1 posiada przynajmniej 5 tokenów starszych niż czas $\gamma_{p_1}^L$. W chwili $\tau = 22$ miejsce p_1 posiada co prawda 7 tokenów, jednak tylko 4 z nich mają czas większy lub równy γ_p^L. Nie istnieje więc podzbiór aktywujący, co spowoduje dezaktywację t_3. Należy zaznaczyć, że jeśli t_3 do tego czasu się uruchomi i rozpocznie fazę produkcji tokenów, warunki aktywacji w rozszerzonej sieci czasowej przestają jej dotyczyć do momentu zakończenia przez nią produkcji.

Mozna sformułować ogólniejszy problem dotyczący tego, czy na podstawie znanych wartości czasowych innych miejsc i tranzycji, pewna tranzycja t ma szanse na uruchomienie w pewmym określonym czasie. Algorytm rozwiązujący tego typu problem może między innymi pozwalać na określenie, czy przypisane do tranzycji wartości czasowe są nieodpowiednie w tym sensie, że razem z wartościami czasowymi jej miejsc wejściowych spowodują, że np. w ekstremalnym przypadku tranzycja ta będzie cały czas nieaktywna.

Problem ten można rozważać dla wielu przypadków różniących się stopniem skomplikowania układu miejsc i tranzycji. Jedną z prostszych propozycji jego rozwiązywania jest Algorytm 2. Zakłada się tutaj prostszy przypadek, w którym badana tranzycja ma tylko jedno miejsce wejściowe. Dodatkowo nie są tutaj uwzględniane konflikty tranzycji konkurujących o tokeny z tego samego miejsca. Bez tego założenia rozważania wkraczałyby już w obszar analizy przestrzeni stanów, co wykracza poza zakres niniejszego artykułu.
Dane wejściowe dla Algorytmu 2 są następujące:

- Określona tranzycja t_x taka, że istnieje miejsce wejściowe $p \in {}^\bullet t_x$.
- Wartości γ_p^L i γ_p^U przypisane do miejsca p.
- Zbiór wszystkich tranzycji wejściowych miejsca p, oznaczony jako ${}^\bullet p$.
- Waga łuku $V(p, t_x)$.

Algorytm 2 Weryfikacja warunków uruchomienia tranzycji

1: $counter = 0, last\_max = 0; MULTISETS[] = \emptyset, VECTOR[] = \emptyset$
2: *early_firing_possible* = *late_firing_possible* = **false**
3: **for** $(time = 1$ **to** $(lcm(fast(t_i) \ : \ t_i \in {}^\bullet p) + \gamma_p^U))$ **do**
4: $\quad MULTISETS[time] \leftarrow generateMultisetforPlace(time)$
5: $\quad$ **if** $(activationSubsetExist(MULTISETS[time], t_x))$ **then**
6: $\quad\quad VECTOR[time] = 1$
7: $\quad$ **else**
8: $\quad\quad VECTOR[time] = 0$
9: $\quad$ **end if**
10: **end for**
11: **for** $(time = 1$ **to** $(lcm(fast(t_i) \ : \ t_i \in {}^\bullet p) + \gamma_p^U))$ **do**
12: $\quad$ **if** $(VECTOR[time] = 1)$ **then**
13: $\quad\quad counter = counter + 1$
14: $\quad\quad$ **if** $(counter > last\_max)$ **then**
15: $\quad\quad\quad last\_max = counter$
16: $\quad\quad$ **end if**
17: $\quad$ **else**
18: $\quad\quad counter = 0$
19: $\quad$ **end if**
20: **end for**
21: **if** $(last\_max \geqslant \alpha_{t_x}^L)$ **then**
22: $\quad early\_firing\_possible =$ **true**
23: **end if**
24: **if** $(last\_max \geqslant \alpha_{t_x}^U)$ **then**
25: $\quad latest\_firing\_possible =$ **true**
26: **end if**

Maksymalna liczba kroków (tj. jednostek czasu), która podlega sprawdzeniu przez Algorytm 2 określona jest w dwóch niezależnych pętlach działających w liniach 3-10 oraz 11-20. W obu przypadkach liczba kroków jest równa wielokrotności wartości funkcji $fast(t_i)$ dla wszystkich tranzycji wejściowych $t_i \in {}^\bullet p$ oraz maksymalnego

dopuszczalnego czasu życia tokenu (γ_p^U), tak jak w Algorytmie 1. Pierwsza pętla ma dwa zadania. Po pierwsze dla każdego czasu (iteracji) $time$, w liście $MULTISETS[]$ na pozycji $time$ tworzony jest aktualny multizbiór K reprezentujący tokeny miejsca p w chwili $time$. Odpowiada za to funkcja $generateMultisetforPlace(time)$. Innymi słowy pętla ta przeprowadza prostą symulację tworzenia nowych tokenów od chwili $time = 1$, w której to zakłada się, że miejsce p nie posiada jeszcze tokenów, o ile nie utworzy ich wspomniana funkcja. Listę $MULTISET[]$ w pełnej postaci można sobie wyobrazić jako listę wierszy z kolumn tabeli z Rysunku 1 opisanych jako "Tokeny". Drugim zadaniem pętli jest sprawdzanie w każdym kroku (linie 5-9), czy w chwili $time$ istnieje podzbiór aktywujący dla t_x. Jeżeli istnieje, do listy $VECTOR[]$ na pozycji $time$ wpisywane jest 1, zero w przeciwnym wypadku.

Druga pętla Algorytmu 2 znajduje się w liniach 11-20. Jej zadaniem jest wyliczenie maksymalnej liczby kroków (przechowywanej w $last\_max$), w których bez żadnej przerwy istnieje podzbiór aktywujący dla t_x. Za każdym razem gdy na liście $VECTOR[time]$ pojawi się zero, co oznacza brak takiego podzbioru w chwili $time$, suma pomocnicza $counter$ jest resetowana (linia 20).

Ostatnie dwie instrukcje warunkowe algorytmu w liniach 21 oraz 24 sprawdzają, czy wyliczona suma $last\_max$ jest większa odpowiednio od $\alpha_{t_x}^L$ oraz od $\alpha_{t_x}^U$. W pierwszym przypadku oznacza to, że tranzycja t_x ma w ogóle szanse się uruchomić. W przypadku gdy $last\_max \geqslant \alpha_{t_x}^U$ wiemy, że podzbiór aktywacyjny w badanym przez Algorytm 2 zakresie czasu istniał przez cały czas, kiedy t_x może pozostawać aktywna.

Na Rysunku 1 oznaczono wiersze, w których istnieje podzbiór aktywujący. Widać, że w analizowanym czasie istniał on bez przerwy czterokrotnie, w wierszach $14 - 21$, $23 - 30$, $32 - 39$ oraz $41 - 48$. Za każdym razem istniał on 8 jednostek czasu, co oznacza, że od momentu aktywacji tranzycja t_3 musi rozpocząć produkcję najdalej w tym momencie (tj. do ósmej jednostki czasu włącznie od momentu aktywacji), pomimo tego, że jej teoretyczny maksymalny czas do uruchomienia to $\alpha_{t_x}^U = 12$.

Co do złożoności czasowej Algorytmu 2, należy najpierw zauważyć, że formalnie istnieją dwie pętle (linie 3-9 oraz 11-20), które jednak są rozdzielone i działają jedna po drugiej. Jednakże w linii 5 znajduje się odwołanie do funkcji, która sprawdza czy istnieje podzbiór aktywujący. Jego wyznaczenie musi obejmować sprawdzenie wszystkich miejsc wejściowych danej tranzycji, a dla każdego miejsca analizę multizbioru K opisującego tokeny miejsca. Funkcja ta ma więc złożoność $\mathcal{O}(p \cdot k)$, gdzie p to liczba miejsc wejściowych, a k to liczba tokenów w każdym miejscu. Podsumowując, złożoność całego Algorytmu 2 wynosi $\mathcal{O}(n \cdot p \cdot k)$, gdzie n to liczba kroków analizowanych przez algorytm (linia 3).

4. Podsumowanie

Rozważana rozszerzona sieć czasowa posiada wiele ciekawych właściwości. Pełny matematyczny opis przedstawionej sieci planowany jest w przyszłych publikacjach. Można tutaj wymienić definicje dozwolonych operacji na multizbiorach K, takie jak wpływ upływu czasu, dodawania oraz usuwania tokenów, porównywanie wartości czasowych na potrzeby określania zbioru aktywnych w danej chwili tranzycji oraz wiele innych. Zaproponowane w niniejszym artykule algorytmy nie są skomplikowane, jednak pokazują możliwości weryfikacji modeli opartych na nowego rodzaju sieci czaso-

wej. Pierwszy algorytm podaje przybliżoną, maksymalną liczbę tokenów możliwą dla każdego miejsca sieci. Oznacza to, że sieć taka jest ograniczona, o ile określony jest maksymalny czas życia tokenu dla każdego miejsca. Kolejny proponowany algorytm jest w stanie weryfikować, w oparciu o dane czasowe zawarte w elementach sieci, czy tranzycje są w stanie się uruchomić. Jest to także podstawowa wersja, która może być ulepszana, aby wskazywać problemy z zagłodzeniem tranzycji dla dowolnej struktury zbudowanej w oparciu o sieć czasową. Inne algorytmy dla rozważanej sieci mogą na przykład określać przybliżoną liczbę produkowanych i konsumowanych tokenów w pewnych określonych przedziałach czasu.

Rozszerzona sieć czasowa xTPN daje bardzo duże możliwości tworzenia modeli na niej opartych, w sytuacji, gdy dane czasowe o procesie biologicznym są niepełne lub bardzo przybliżone. W przypadku bardzo precyzyjnych danych możliwe jest takie dostosowanie wartości czasowych opisujących np. tranzycje, aby czas działania był deterministycznie określony. Co istotne, za pomocą proponowanej sieci można tworzyć modele, które będą miały różne poziomy precyzji w ramach jednej sieci, zależnie od dostępności danych opisujących badany proces lub system biologiczny.

LITERATURA

1. Formanowicz D., Radom M., Rybarczyk A., Formanowicz P.: The role of Fenton reaction in ROS-induced toxicity underlying atherosclerosis - modeled and analyzed using a Petri net-based approach, Biosystems, 165, 2018, p. 71–87.
2. Heiner, M., Gilbert, D., Donaldson, R.: Petri nets for systems and synthetic biology, Lecture Notes in Computer Science, 5016, 2008, pp. 215–264.
3. Heiner M., Lehrack S., Gilbert D., Marwan W.: Extended Stochastic Petri Nets for Model-Based Design of Wetlab Experiments, Lecture Notes in Computer Science, 5750, 2009, p. 138–163.
4. Merlin P.M.: A Study of the Recoverability of Computing Systems, Ph.D. Thesis in University of California, Irvine, 1974.
5. Olszak J., Radom M., Formanowicz P.: Some aspects of modeling and analysis of complex biological systems using time Petri nets, Bulletin of the Polish Academy of Sciences. Technical Sciences, 66(1), 2018, p. 67–78.
6. Popova-Zeugmann L., Heiner M., Koch I.:Modelling and Analysis of Biochemical Networks with Time Petri Nets, Fundamenta Informaticae, 67(1), 2005, p. 149–162.
7. Popova-Zeugmann L.: Time and Petri Nets, Springer, Berlin, 2013.
8. Radom M., Formanowicz P.: A proposal for a new type of Petri net with time parameters, 2022, *submitted for publication*.
9. Wegener J.P., Popova-Zeugmann L.: Petri Nets with Time Windows: A Comparison to Classical Petri Nets, Fundamenta Informaticae, 93(1), 2009, p. 337–352.