General and Professional Education
4/2015 pp. 39-50
ISSN 2084-1469

SYMULATOR INTERPRETOWANEJ ROZMYTEJ SIECI PETRIEGO JAKO NARZĘDZIE DYDAKTYCZNE

THE FUZZY INTERPRETED PETRI NET EMULATOR AS EDUCATIONAL TOOL

Michał Markiewicz
Politechnika Rzeszowska im. Ignacego Łukasiewicza
Wydział Elektrotechniki i Informatyki
ul. Wincentego Pola 2
35-021 Rzeszów
e-mail: mmarkiewicz@kia.prz.edu.pl

Abstract: The paper presents Fuzzy Interpreted Petri Net Emulator as an educational tool. First, the usefulness of Petri nets is discussed. Then, the computer tools which are used to present a principle of the Petri nets operation are introduced. The next few sections of the article describe formal definition of Fuzzy Interpreted Petri Net emulator which is based on this net and example the use of the emulator. Finally, the educational values of the Petri nets are described. Arising from the article the additional conclusions are that emulator can be used for both educational and practical purposes and there exist many directions of further development for this tool.

Keywards: Petri nets, Fuzzy Interpreted Petri Net, Petri net emulator, educational tool.

Wprowadzenie

W 1962 roku Carl Adam Petri opublikował rozprawę doktorską pt. Kommunikation mit Automaten [13]. Opisał w niej graficzny język, który później znalazł szerokie zastosowanie w modelowaniu systemów, w których występują procesy współbieżne. Język ten nazwano od ich twórcy siecią Petriego (z ang. Petri Nets). W pierwotnej formie bądź też w zmodyfikowanej, przez ponad 50 lat sieci Petriego stały się użyteczne dla wielu zastosowań praktycznych w różnych dziedzinach nauki i przemysłu. Przykładowe obszary ich zastosowania są opisane w [2-6, 11] i obejmują między innymi projektowanie i programowanie sterowników przemysłowych, planowanie i sterowanie produkcją, wykrywanie i diagnostykę obiektów wykorzystywanych w procesach produkcyjnych oraz zwykłych obiektów technicznych takich jak np. most. Od czasu powstania sieci Petriego są nieustannie rozwijane. Z jednej strony systemy modelowane za ich pomocą są poddawane szczegółowej analizie, z drugiej powstało wiele modyfikacji klasycznego modelu zaproponowanego przez Carla A. Petriego.

Współcześnie można wyróżnić dwa główne kierunki badań, które są związane z sieciami Petriego [4]. Pierwszy, który jest obecnie najbardziej popularny, obejmuje sieci wysokiego poziomu. Dotyczy łączenia sieci Petriego z językami programowania wysokiego poziomu. Najczęściej używanymi sieciami w tej gałęzi są sieci kolorowane. Drugi nurt z kolei jest powiązany z sieciami niskiego poziomu, które znajdują zastosowanie w modelowaniu i weryfikacji systemów wykorzystywanych w sterownikach przemysłowych oraz w reprogramowalnych układach sprzętowych, używanych do sterowania i zastosowań diagnostycznych. Wraz z rozwojem wykorzystania i badań sieci Petriego częstą praktyką stało się tworzenie narzędzi komputerowych - symulatorów, które pozwalają na dokładną analizę oraz testy poprawności systemów budowanych w oparciu o sieci. Symulacje komputerowe umożliwiają zmniejszenie kosztów związanych z tworzeniem systemu.

Ułatwiają znalezienie w nim błędów na etapie projektowania oraz pozwalają również na poznanie jego właściwości.

Coraz częstsze wykorzystanie sieci Petriego w praktyce powoduje, że rośnie potrzeba coraz szerszego wprowadzania ich do edukacji. W związku z czym symulatory na bazie sieci Petriego mogą pełnić ważną funkcję dydaktyczną, pomagając zrozumieć działanie sieci. Z doświadczeń autora wynika, iż definicja oraz opis formalny sieci Petriego jaki zazwyczaj używa się do opisu jej działania, nie jest dla każdego zrozumiały. Natomiast pokazanie zasady jej działania w symulatorze może znacznie ułatwić oraz przyśpieszyć proces kształcenia.

Obecnie jest dostępnych kilka użytecznych i sprawdzonych rozwiązań pozwalających na symulowanie działania różnych rodzajów sieci Petriego. Jednak nie nadążają one za rozwojem i tworzeniem nowych wersji sieci, nawet jeśli umożliwiają modelowanie kilku różnych rodzajów sieci. Stąd istnieje realna potrzeba tworzenia nowych narzędzi do symulacji. Jedną z sieci, która nie posiada oficjalnej wersji symulatora jest rozmyta interpretowana sieć Petriego (z ang. Fuzzy Interpreted Petri Net, FIPN), zaproponowana przez L. Gniewka w [5, 6]. Sieć ta różni się tym od zwykłej sieci rozmytej, że może być wykorzystana nie tylko w diagnostyce, ale również do sterowania. Do tej pory w Katedrze Informatyki i Automatyki na Politechnice Rzeszowskiej istniała wersja testowa symulatora rozwijana w ramach prac dyplomowych, a czym narzędzie posiadało pewne ograniczenia związane z definicją (nie były zaimplementowane wszystkie warunki sieci oraz najnowsza definicja sieci) oraz funkcjonalnością. W ramach artykułu zmodyfikowano ten symulator i wydano pierwszą oficjalną wersję udostępnioną w internecie [14].

Celem niniejszej publikacji jest opisanie symulatora FIPN (SFIPN) oraz jego dużej przydatności dydaktycznej. W artykule dokonano zarysu problematyki związanej z symulatorami sieci Petriego oraz ich powiązania z edukacją. Opisane zostały różne symulatory sieci Petriego, definicja i ograniczenia FIPN oraz informacje o symulatorze, który powstał na bazie tej sieci. Omówiony został również system, dla którego przeprowadzono przykładową symulację oraz możliwości symulacyjne SFIPN. Wykonane testy pozwalają także na sprawdzenie użyteczności praktycznej SFIPN.

Wybrane symulatory sieci Petriego

Obecnie najbardziej znanym i popularnym symulatorem sieci Petriego jest CPN Tools [1, 10]. W początkowej fazie narzędzie było rozwijane w Aarhus University, obecnie na Eindhoven University of Technology. Umożliwia ono edycję, symulację i analizę kolorowanych sieci Petriego zwykłych oraz czasowych. Udostępnia interfejs graficzny, który pozwala na przeprowadzenie różnego rodzaju symulacji dla stworzonego systemu, takich jak zmienianie znakowania miejsc w trakcie działania symulacji sieci, badanie wydajności systemu oraz wyświetlanie komunikatów o błędach, dzięki czemu stają się łatwiejsze do wykrycia. Symulator CPN Tools może automatycznie określić różne właściwości sieci takie jak np. żywotność czy ograniczoność oraz wspiera hierarchiczność.

Kolejnym narzędziem do modelowania sieci Petriego jest Snoopy [8]. W przeciwieństwie do poprzedniego symulatora umożliwia nie tylko używanie kolorowanych sieci Petriego, ale także innych rodzajów sieci np. klasycznych, hybrydowych i stochastycznych. Również pozwala modelować sieci hierarchiczne. Chociaż Snoopy nie ma tak bogatej funkcjonalności jak CPN Tools, to żadne inne dostępne narzędzie do sieci Petriego nie udostępnia opcji budowania i analizy tak wielu rodzajów sieci Petriego. Dzięki czemu znajduje wiele obszarów zastosowań np. w biochemii [9].

Innym powszechnie używanym symulatorem sieci Petriego jest Great SPN [7]. Narzędzie to pozwala modelować uogólnione hybrydowe oraz kolorowane rozszerzone sieci Petriego. Implementuje algorytm, który umożliwia efektywną analizę systemu pod względem wydajności oraz struktury. W przeciwieństwie do dwóch pierwszych omówionych symulatorów nie udostępnia hierarchiczności.

Również bardzo użytecznym narzędziem, znajdującym zastosowanie w praktyce jest Cell Illustator [12]. Pozwala modelować rozszerzone hybrydowe sieci Petriego do stosowania ścieżek biochemicznych. Program łączy w sobie sieci dyskretne oraz ciągłe, dzięki czemu możliwe jest korzystanie z tranzycji ciągłych, jak i stochastycznych. W narzędziu tym brakuje hierarchiczności, a dodatkowo jest dostępne do użytku komercyjnego jedynie za opłatą.

Spośród wyżej omówionych symulatorów każdy ma jakieś zalety. Można powiedzieć, że każde narzędzie jest przeznaczone dla

odbiorców o innych potrzebach i znajduje zastosowanie w różnych projektach praktycznych oraz naukowych. Jednak żadne z powyższych oraz dostępnych w internecie nie umożliwia modelowania FIPN.

Definicja FIPN i przyjęte ograniczenia sieci

FIPN, dla której stworzono symulator w ramach artykułu zgodnie z definicją zawartą w [5, 6] jest dwunastką:

$\text{FIPN} = (P,T,\Omega,\Psi,R,\Delta,K,W,\Gamma,\Theta,M_0,e)$,

gdzie:

$P = P' \cup P''$ - niepusty zbiór miejsc:

$P' = \{p'_1, p'_2, \ldots, p'_{a'}\}$ - zbiór miejsc związanych z modelowaniem działań lub procesów,

$P'' = \{p''_1, p''_2, \ldots, p''_{a''}\}$ - zbiór miejsc związanych z modelowaniem zasobów,

$T = \{t_1, t_2, \ldots, t_b\}$ - niepusty skończony zbiór tranzycji,

$\Omega = \{\omega_1, \omega_2, \ldots, \omega_{a'+a''}\}$ - niepusty skończony zbór stwierdzeń,

$\Psi = \{\psi_1, \psi_2, \ldots, \psi_b\}$ - niepusty, skończony zbiór warunków,

P, T, Ω, Ψ - są to zbiory rozłączne,

$R \subseteq (P \times T) \cup (T \times P)$ - relacja incydencji, w której dla każdej tranzycji $t_i \in T, i = 1, 2, \ldots, b,$ istnieje miejsce $p' \in P'$ takie, że $(p', t_i) \in R$ *lub* $(t_i, p') \in R$,

$\Delta : P \rightarrow \Omega$ - funkcja przypisująca każdemu miejscu stwierdzenie,

$K : P' \rightarrow 1$ i $P \rightarrow N \setminus \{1\}$ - funkcja przypisująca każdemu miejscu pojemność,

gdzie $N = \{1, 2, \ldots\}$,

$\Gamma : T \rightarrow \Psi$ - funkcja przypisująca każdej tranzycji warunek,

$\Theta : T \rightarrow [0,1]$ - funkcja określająca stopnie spełnienia warunków związanych z tranzycjami t,

$W : R \rightarrow N$ - funkcja wagowa spełniająca dwa warunki: $W(p,t) \le K(p)$ i $W(t,p) \le K(p)$

gdzie:

p oznacza p' lub p'',

$M_0 : P' \rightarrow \{0,1\}$ i $(P'') \rightarrow W_+$ - funkcja znakowania początkowego,

gdzie:

$M_0(p''_j) = z_j / K(p''_j), z_j \in N \cup \{0\},$

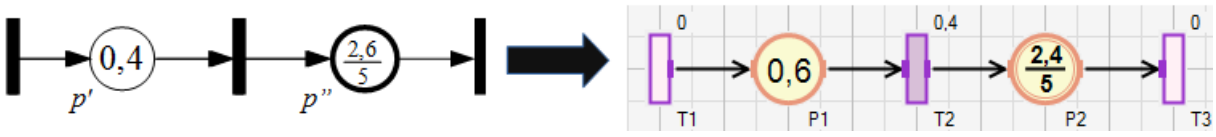
$z_j \le K(p''_j), j = 1, 2, \ldots, a''$

W_+ to zbiór nieujemnych liczb wymiernych,

e - zdarzenie synchronizujące działanie wszystkich tranzycji.

Sieć ta należy do klasy sieci uogólnionych, gdyż zostały do niej wprowadzone wagi, którymi etykietowane są łuki łączące miejsca z tranzycjami oraz tranzycje z miejscami. Miejsca w sieci mają skończoną pojemność. Wyróżnia się w niej dwa rodzaje miejsc: bis i prim. W miejscach typu bis możliwe jest umieszczanie więcej niż jednego znacznika dzięki wykorzystaniu pojemności miejsc, które pełnią rolę współczynników normalizujących. Sieć tę zalicza się do klasy rozmytych sieci Petriego. Jej działanie sprowadza się do przemieszczania znaczników, których wartości należą do przedziału [0, 1]. W sieci nie dochodzi do rozlewania jednego znacznika do wielu miejsc dzięki nałożonym ograniczeniom, mimo iż w jednym momencie może być aktywnych kilka tranzycji. Analogicznie do sieci interpretowanych, aktywnienie tranzycji może być związane z zewnętrznymi zdarzeniami procesowymi i jest synchronizowane przez zewnętrzny sygnał taktujący.

Postać graficzna FIPN to graf skierowany. Miejsca typu p' są rysowane za pomocą symbolu okręgu, natomiast p'' tak samo, lecz z wykorzystaniem pogrubionej linii. Tranzycje są przedstawiane poprzez pogrubioną kreskę (w symulatorze prostokąt). Z kolei elementy relacji incydencji R - łuki rysowane są za pomocą strzałek, które są etykietowane wagą. Przykład fragmentu sieci oraz jej odpowiednik w symulatorze znajdują się na rys. 1. Dodatkowo można zaobserwować, że w symulatorze przy każdej z tranzycji występuje liczba oznaczająca stopień spełnienia warunku oraz nazwa tranzycji.



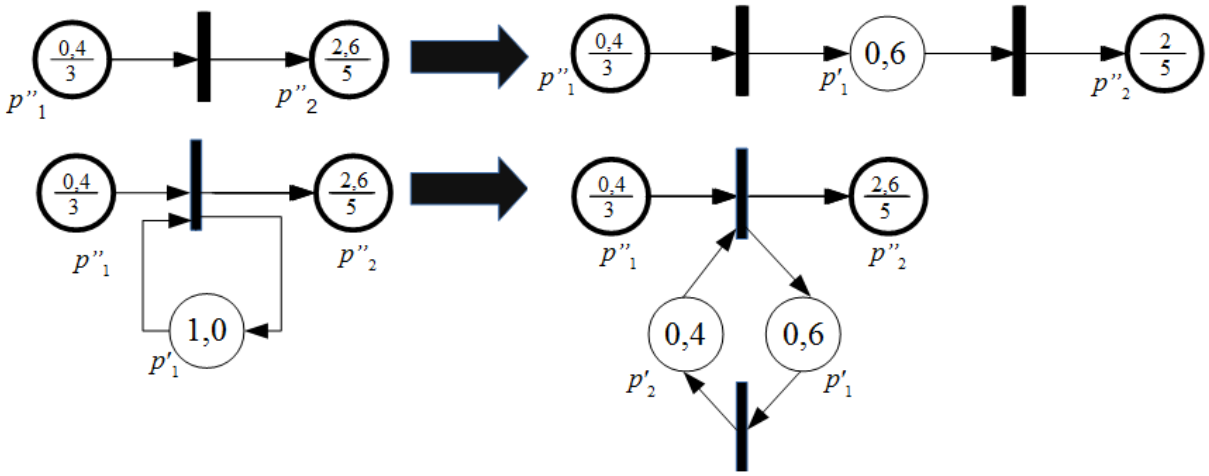
Rys. 1. Przykładowa FIPN oraz jej odpowiednik w symulatorze.

W postaci graficznej FIPN znakowanie miejsca $M(p)$ jest przedstawione poprzez umiejscowienie pewnej liczby w okręgu, który oznacza miejsce p. Dla miejsc typu p' może ona przyjmować wartość z przedziału domkniętego [0, 1]. Wynika to z tego, iż w miejscach tego typu może znajdować się maksymalnie jeden znacznik. Z kolei w miejscach typu p'' liczba reprezentująca znakowanie również może przyjmować wartość z przedziału [0, 1], ale jest poddana normalizacji. W formie graficznej będzie ona podana w postaci ułamka, aby ułatwić interpretację ile znaczników znajduje się w miejscu. Mianownik oznacza pojemność miejsca, natomiast licznik liczbę znaczników, które się w nim znajdują. W miejscach typu p' mianownik jest pomijany, gdyż zgodnie z definicją sieci jest on zawsze dla tych miejsc równy jeden.

Wartość $M(p)$ dla miejsc typu p' z przedziału (0, 1] oznacza obecność znacznika w miejscu p w pewnym stopniu, natomiast $M(p) = 0$ jest interpretowana jako brak znacznika w miejscu. Na rys. 1 widać, że w miejscu p' znajduje się znacznik w stopniu 0.4. Dla miejsca p'' wartość z przedziału (0,1] również oznacza obecność znacznika w miejscu, jednak jest inaczej interpretowana. Na rys. 1 wartość dla miejsca p'' wynosi $M(p'') = 2,6/5$ czyli 0,52. Oznacza to, że w tym miejscu występują dwa znaczniki w stopniu 1 oraz jeden w stopniu 0,6.

W celu utrzymania przejrzystości w FIPN zgodnie z [5, 6] ustalono, że w chwili początkowej znakowanie w miejscach typu p' jest równe 0 lub 1, natomiast w miejscach typu p'' jest wielokrotnością $1/K(p'')$ lub ma wartość 0. Sieć musi również spełniać dodatkowe warunki związane z tranzycjami i miejscami.

Dla każdej tranzycji $t \in T$ musi istnieć łuk prowadzący do lub wychodzący z miejsca typu p', oraz nie można zapętlać miejsc typu p'. Rys. 2 pokazuje jak można zastąpić pewne fragmenty sieci, aby spełniały one powyższe założenia. Symulator przy próbie zapętlania miejsc p' wyświetli błąd z opisem, której tranzycji dotyczy problem i nie umożliwi takiej czynności. W przypadku, gdy użytkownik spróbuje dodać wyłącznie miejsca p'' po obu stronach tranzycji, to analogicznie pojawi się informacja o błędzie.



Rys. 2. Przekształcenie fragmentów sieci, które nie spełniają założeń FIPN

Kolejną istotną kwestią dla FIPN jest otrzymywanie koncesji. Zgodnie z definicją [5, 6]:

Tranzycja $t \in T$ jest przygotowana do uaktywnienia (ma koncesję) dla znakowania M: $P \rightarrow [0, 1]$ od momentu, gdy stopień spełnienia warunku $\Theta(t) = \vartheta$, związanego z tą tranzycją jest większy od zera i są spełnione warunki:

$$\forall p \in {}^{\bullet}t, M(p) \geq \frac{W(p,t)}{K(p)}$$

i (1)

$$\forall p \in t^{\bullet}, M(p) \leq \frac{K(p) - W(t,p)}{K(p)}$$

do momentu gdy:

$$\exists p' \in {}^{\bullet}t, M(p') = 0$$

lub (2)

$$\exists p' \in t^{\bullet}, M(p') = 1$$

Zgodnie z tą definicją moment utraty aktywności przez tranzycję jest wyznaczany na

podstawie miejsc typu p', ponieważ założono, że każda tranzycja jest powiązana przynajmniej z jednym miejscem takiego typu. Symbol $t^\bullet$ oznacza miejsca wyjściowe tranzycji, natomiast $^\bullet t$ miejsca wejściowe.

W FIPN duże znaczenie ma również zmiana znakowania miejsc w zależności od ich typu. Zgodnie z definicją [5, 6]: jeżeli dla znakowania M tranzycja $t \in T$ jest przygotowana do uaktywnienia, stopień spełnienia warunku związanego z tą tranzycją $\Theta(t) = \vartheta$, gdzie $\vartheta \in [0,1]$ zmieni się o $\Delta\vartheta \geq 0$ i wystąpi zdarzenie e synchronizujące działanie wszystkich tranzycji, to nowe znakowanie sieci M' można wyznaczyć za pomocą następującej reguły:

$$M'(p) = \begin{cases} M(p) - \Delta\vartheta \cdot \dfrac{W(p,t)}{K(p)}, \text{ dla } p \in {}^\bullet t \setminus t^\bullet \\ M(p) + \Delta\vartheta \cdot \dfrac{W(t,p)}{K(p)}, \text{ dla } p \in t^\bullet \setminus {}^\bullet t \\ M(p) - \Delta\vartheta \cdot \dfrac{W(p,t)}{K(p)} + \Delta\vartheta \cdot \dfrac{W(t,p)}{K(p)}, \\ \qquad \text{dla } p \in t^\bullet \cap {}^\bullet t \\ M(p), \text{ dla } p \notin t^\bullet \cup {}^\bullet t \end{cases} \quad (3)$$

Przyrost $\Delta\vartheta < 0$ nie spowoduje zmian w znakowaniu sieci.

Opis SFIPN

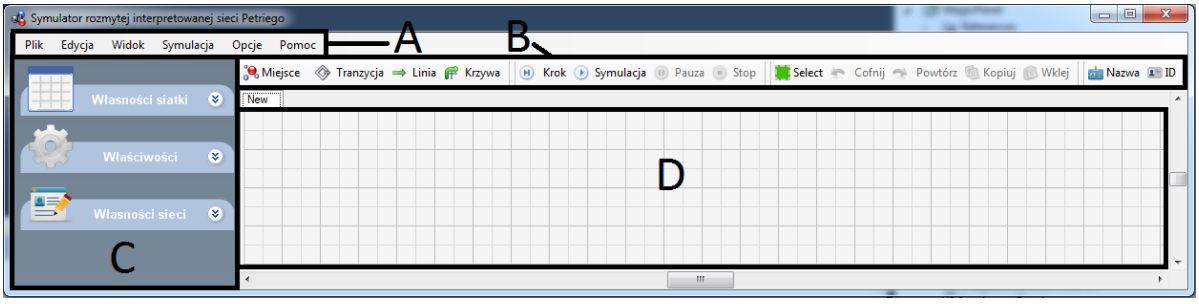
SFIPN został napisany w języku C# z wykorzystaniem platformy .NET. Widok narzędzia po uruchomieniu programu przedstawiono na rys. 3. Zgodnie z nim symulator składa się z czterech części:

1 -główne menu programu;

2 - pasek narzędzi do dodawania nowych elementów, cofania i powtarzania zmian, zaznaczania elementów, przeprowadzania symulacji oraz ustawiania sposobu wyświetlania miejsc (te opcje są dostępne również z poziomu głównego menu);

3 - sekcja, gdzie można ustawić własności siatki okna roboczego, właściwości poszczególnych miejsc, tranzycji, linii oraz wyświetlić podstawowe właściwości utworzonej sieci;

4 - obszar roboczy, gdzie można budować FIPN oraz symulować jej działanie.



Rys. 3. Główne okno SFIPN.

Główne menu programu składa się z następujących opcji:

- 'Plik': jest tu możliwość zapisania bieżących zmian do otwartego pliku ('Zapisz'), zapisania nowego pliku ('Zapisz jako'), otworzenia nowego pustego pliku ('Nowy'), otwarcia pliku z dysku ('Otwórz') oraz zamknięcia całej aplikacji ('Koniec');
- 'Edycja': w tej opcji dostępne są wszystkie funkcje paska narzędzi związane z dodawaniem elementów, ich zaznaczaniem, kopiowaniem i wklejaniem, zostały one omówione w tabeli 1;
- 'Widok': w tej opcji dostępne jest wyświetlanie poszczególnych macierzy, w oparciu o które działa symulator (***M*** - wektor znakowania miejsc, ***K*** - wektor pojemności miejsc, ***C*** - macierz incydencji, ***ΔΘ*** - wektor spełnienia warunków, ***U*** - wektor aktywności tranzycji);
- 'Symulacja' - dostępne są tutaj funkcje paska narzędzi związane z symulacją sieci, zostały one omówione w tabeli 1;
- 'Opcje' – w tej części menu dostępne są ustawienia globalne odnośnie podstawowych właściwości dla linii, miejsc i tranzycji, z których można zbudować sieć.

Tabela 1. Opcje paska narzędzi w SFIPN – sekcja B na rys. 3

Opcja	**Funkcja (wyłączenie funkcji odbywa się poprzez ponowne kliknięcie na opcję bądź użycie skrótu klawiszowego)**	**Skrót klawiszowy**
Miejsce	Dodawanie nowych miejsc.	Ctrl + P
Tranzycja	Dodawanie nowych tranzycji.	Ctrl + T
Linia	Dodawanie nowych linii bez zaokrągleń.	Ctrl + L
Krzywa	Dodawanie nowych linii, które przy posiadaniu więcej niż dwóch punktów (początkowego i końcowego) ulegają zaokrągleniu.	Ctrl + K
Krok	Umożliwia wykonanie symulacji krokowej dla zbudowanej sieci.	Ctrl + F
Symulacja	Umożliwia przeprowadzenie automatycznej symulacji po ustawieniu odpowiednich parametrów dla zbudowanej sieci.	Ctrl + W
Pauza	Wstrzymuje symulację.	Ctrl + G
Stop	Zatrzymuje symulację i pozwala powrócić do edycji i tworzenia nowych elementów w sieci.	Ctrl + H
Zaznacz	Umożliwia zaznaczenie elementów poprzez narzędzie prostokątnego zaznaczania.	Ctrl + R
Cofnij	Przycisk pozwalający cofnąć ostatnio dokonaną zmianę.	Ctrl + Z
Powtórz	Przycisk pozwalający na powtórzenie ostatnio wykonanej akcji.	Ctrl + Y
Kopiuj	Kopiowanie zaznaczonych elementów do bufora kopiującego.	Ctrl + C
Wklej	Przycisk umożliwiający wklejenie elementów skopiowanych do bufora kopiującego.	Ctrl + V
Nazwa	Wyświetlaj nazwy miejsc w obszarze roboczym (zamiast id).	Ctrl + I
ID	Wyświetlaj id miejsc w obszarze roboczym (zamiast nazw).	Ctrl + Shift + I

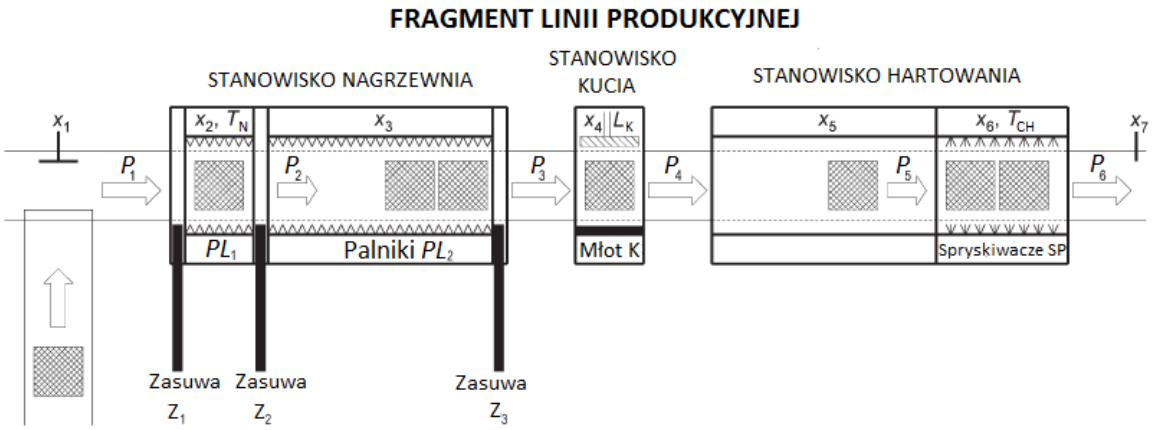
Zgodnie z tabelą 1, w celu utworzenia nowego miejsca należy kliknąć na opcje 'Miejsce' znajdujące się w sekcji B na Rys. 3, bądź użyć skrótu 'Ctrl + P'. Przy czym skróty klawiszowe programu działają, gdy użytkownik kliknął dowolnym klawiszem myszy na obszar roboczy (sekcja D na rys. 3). Po jednej z tych dwóch akcji kursor powinien zmienić swoją postać na krzyżyk i po kliknięciu w dowolne miejsce obszaru roboczego automatycznie zostanie utworzony nowy element sieci. Analogicznie można dodawać nowe tranzycje. Z kolei łuki są uwarunkowane od miejsc i tranzycji i należy je tworzyć poprzez dwa kolejne kliknięcia na miejsce i tranzycję lub na odwrót.

Po utworzeniu miejsca, tranzycji oraz linii można na nie kliknąć i wyświetlić bądź zmodyfikować właściwości elementów. Dotyczą one wyglądu np. koloru (okno 'Właściwości' w sekcji C) lub działania sieci np. zmiana wagi dla linii, zmiana znakowania początkowego dla miejsc, natomiast dla tranzycji właściwości związanych z warunkiem i aktywnością tranzycji. Dodatkowo miejsca i tranzycje mają rozróżnione takie pola jak id elementu ('ItemID'), które nie podlega zmianie oraz 'Nazwa', która jest domyślnie wyświetlana na schemacie sieci i można ją wielokrotnie modyfikować.

Opis systemu sterowania modelowanego w SFIPN

W dalszej części został opisany fragment linii produkcyjnej kucia na gorąco zaprezentowanej w [5, 6] i poddanej modelowaniu w SFIPN. Celem procesu, który został pokazany na rys. 4 jest obróbka materiału kolejno w 3 etapach: podgrzewanie, kucie oraz hartowanie. Materiał w pierwszej kolejności podlega nagrzewaniu w komorze cieplnej przez określony czas, gdzie może znajdować się maksymalnie jeden element. Dalej trafia do komory oczekiwania na kucie, gdzie mogą być maksymalnie 4 elementy równocześnie. Jej zadaniem jest utrzymanie materiału w odpowiednio wysokiej temperaturze, aby posiadał jak najlepszą plastyczność. W kolejnym etapie materiał, który oczekuje

najdłużej trafia do stanowiska kucia. Następnie odkuwka jest przemieszczana do stacji hartowania, która składa się dwóch komór. W pierwszej odkuwki oczekują na hartowanie (maksymalnie cztery elementy). W drugiej odbywa się chłodzenie za pomocą spryskiwaczy (maksymalnie dwa).



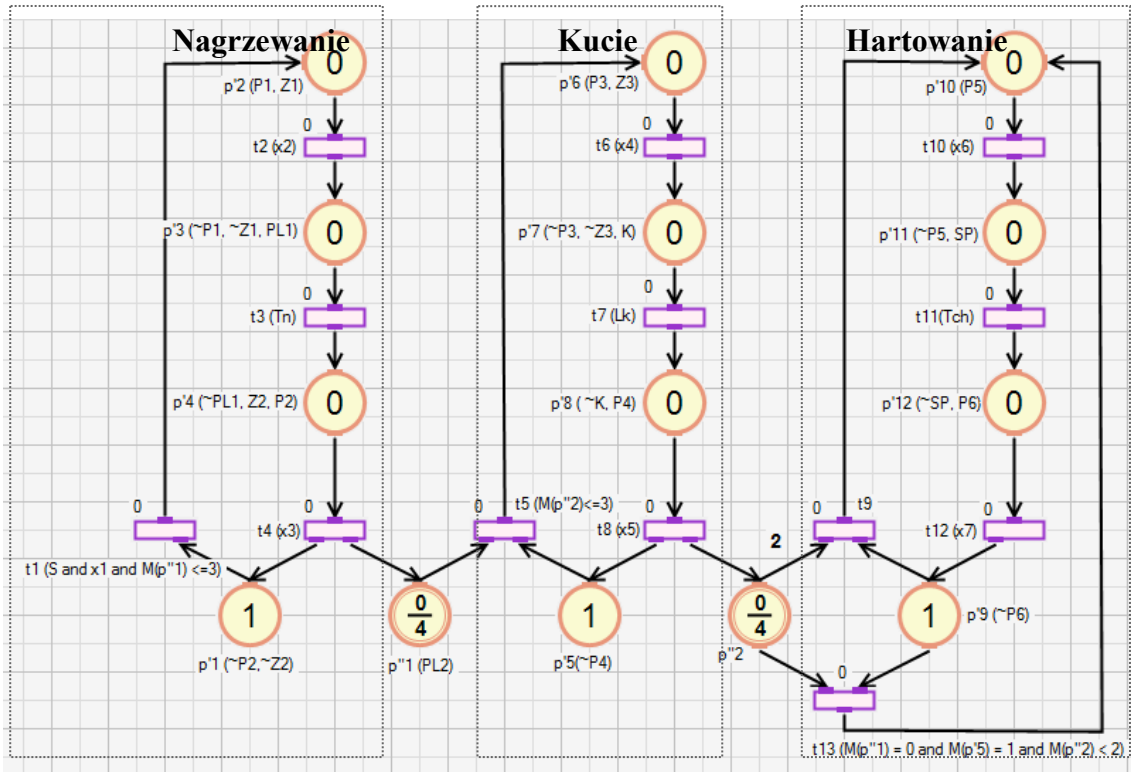
Rys. 4. Fragmentu linii produkcyjnej, dla którego dokonano symulacji (na podstawie[5, 6]).

Pozycje elementów poddanych obróbce są monitorowane przez czujniki x_i, i = 1, 2, …, 7. Czas podgrzewania oraz hartowania za pomocą gazu, są zależne od timerów T_N i T_{CH}. Sterowanie piecem gazowym PL_1 jest integralną częścią komory grzewczej i wymaga czujnika x_2 (analogicznie działa piec gazowy PL_2 i czujnik x_3). Kucie jest synchronizowane przez licznik uderzeń młota L_K. Podajniki są odpowiedzialne za poruszanie się elementów. Są oznaczone jako P_n, n = 1, 2, …, 6. Kiedy podajniki P_1, P_2, P_3 rozpoczną pracę, szybko otwierające się zasuwy Z_1, Z_2 oraz Z_3 są otwierane.

Sterowanie procesem kucia materiału można przedstawić za pomocą FIPN, czyli modelować za pomocą SFIPN. Schemat sieci w symulatorze znajduje się na rys. 5. W stanie początkowym znaczniki powinny być w miejscach p'_1, p'_5 oraz p'_9, co oznacza, że nie ma żadnych elementów na linii produkcyjnej. Zgodnie z rys. 5 sieć można podzielić na 3 części reprezentujące poszczególne etapy procesu produkcyjnego oraz dwa miejsca typu p'' (komory oczekiwania), które je łączą. Obecność znacznika w miejscach typu p'' oznacza, że w poszczególnych komorach znajduje się określona liczba elementów odpowiadająca liczbie znaczników.

W sieci znajdują się tranzycje t_1, t_2, t_4, t_6, t_8, t_{10}, t_{12}, których aktywność jest związana z czujnikami położenia elementów na linii produkcyjnej. Całkowite przelanie się znacznika przez każdą z tych tranzycji oznacza, że czujniki im odpowiadające zostały załączone (element znalazł się zasięgu danego czujnika). Z kolei przejście tokena przez tranzycję t_3 z miejsca p'_3 do p'_4, oznacza że materiał był nagrzewany przez czas T_N (warunek spełnienia tranzycji zależy od czasomierza, który jest podłączony do tranzycji). Analogicznie działają tranzycje t_7 i t_{11}, które są powiązane odpowiednio z kuciem (licznik uderzeń L_k) oraz hartowaniem (czas chłodzenia T_{CH}).

Obecność znacznika w danym miejscu p' lub p'' oraz aktywność poszczególnych tranzycji pozwala na jednoznaczne określenie stanu w jakim znajduje się system. Obecność znacznika w danym miejscu oznacza załączenie/wyłączenie odpowiednich urządzeń. Jeśli np. $M(p''_1)$ = ¾, to w komorze oczekiwania na kucie znajdują się trzy elementy, natomiast jeśli $M(p'_7) = M(p'_8) = 0{,}5$ i tranzycja t_7 jest aktywna, to jest to równoznaczne z tym, że w komorze kucia znajduje się element i zostało wykonane 50% uderzeń młota w element. Przejście znacznika kolejno przez miejsca p'_1, p'_2, p'_3, p'_4, p''_1, p'_6, p'_7, p'_8, p'_5 i p''_2 (tu następuje równoczesne przelanie jednego tokenu do dwóch miejsc), p'_{10}, p'_{11}, p'_{12}, p'_9 oznacza przejście elementu przez cały fragment linii produkcyjnej.



Rys. 5. Schemat symulowanego przykładu.

W symulowanym systemie warto również zwrócić uwagę na tranzycje, które mogą być aktywowane przy spełnieniu dodatkowego warunku. Tranzycja t_1 jest aktywowana w reakcji na naciśnięcie przycisku startowego, załączenie czujnika x_1 oraz gdy w komorze oczekiwania na kucie jest maksymalnie 3 elementy. Przyznanie koncesji tranzycji t_5 jest ograniczona do warunku $M(p''_2) <= 3$, czyli jeśli jest miejsce w komorze oczekiwania na hartowanie. Natomiast do tranzycji t_{13} został przypisany warunek: $M(p''_1) = 0$ i $M(p'_5) = 1$ i $M(p''_2) < 2$, czyli jeśli w komorze oczekiwania na hartowanie jest mniej niż jeden element oraz żaden element nie czeka na kucie i w komorze kucia nie znajduje się żaden element, to wtedy element może być pojedynczo hartowany np. z przyczyny skończenia się materiału do produkcji. W przeciwnym przypadku odbywa się hartowanie dwóch elementów.

Symulacja fragmentu linii produkcyjnej w SFIPN

SFIPN bazuje na reprezentacji algebraicznej, czyli wykorzystuje operacje na macierzach. Dzięki niej możliwe jest obliczanie kolejnych znakowań sieci w wyniku uzyskania koncesji przez tranzycje. Wykorzystywane są cztery operacje macierzowe zdefiniowane następująco w [5, 6] dla macierzy $\boldsymbol{A} = \lfloor a_{ij} \rfloor_{n\times m}$ oraz $\boldsymbol{B} = \lfloor a_{ij} \rfloor_{n\times m}$:

$$\boldsymbol{D} = \lfloor d_{ij} \rfloor_{n\times m} = \boldsymbol{A} + \boldsymbol{B}, d_{ij} = a_{ij} + b_{ij} \text{ (dodawanie)} \quad (4)$$

$$\boldsymbol{E} = \lfloor e_{ij} \rfloor_{n\times m} = \boldsymbol{A} \wedge \boldsymbol{B}, e_{ij} = a_{ij} \wedge b_{ij}$$

(minimum z dwóch elementów) (5)

$$\boldsymbol{F} = \lfloor f_{ij} \rfloor_{n\times m} = \boldsymbol{A} \cdot \boldsymbol{B}, f_{ij} = a_{ij} \cdot b_{ij}$$

(mnożenie) (6)

$$\boldsymbol{G} = \lfloor g_{ij} \rfloor_{n\times m} = \frac{\boldsymbol{A}}{\boldsymbol{B}}, g_{ij} = \frac{a_{ij}}{b_{ij}} \text{ (dzielenie)} \quad (7)$$

dla $i = 1, 2, \ldots, n$ oraz $j = 1, 2, \ldots, m$.

Z kolei postać algebraiczna wygląda następująco [5, 6]:

$$\boldsymbol{M'} = \boldsymbol{M} + \frac{(\boldsymbol{U} \wedge \boldsymbol{\Delta\Theta}) \cdot \boldsymbol{C}}{\boldsymbol{K}} \quad (8)$$

gdzie:

$\boldsymbol{M}$ - wektor przechowujący bieżące znakowanie miejsc o wymiarze 1 x a (a to liczba miejsc),

$\boldsymbol{M'}$ - jest to nowe znakowanie w sieci o takim samym rozmiarze,

$\boldsymbol{U}$ - wektor zero-jedynkowy o wymiarze 1 x b (b to liczba wszystkich tranzycji), w

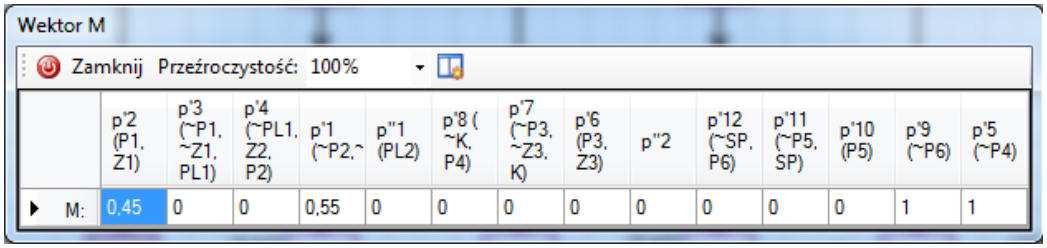
którym numer współrzędnej równej 1 odpowiada indeksowi tranzycji aktualnie przygotowanej w znakowaniu *M*,

ΔΘ - wektor o wymiarze 1 x *b*, w którym współrzędne $\Delta \vartheta_i$, i = 1, 2, … , b opisuje przyrost stopnia warunku związanego z tranzycją t_i,

K – wektor o wymiarze 1 x *a* przypisujący każdemu miejscu pojemność, $K(p) \in N$.

C – jest to macierz incydencji o wymiarze *b* x *a*, analogiczna do klasycznej sieci Petriego.

Symulator daje dwa sposoby sprawdzania wartości przechowywanych w poszczególnych macierzach. Pierwszy sposób polega na odczytaniu wartości bezpośrednio z grafu. Natomiast drugi to możliwość wyświetlenia poszczególnych macierzy poprzez opcję 'Widok' z głównego menu programu. Przykładowy wektor ***M*** z czasu trwania symulacji znajduje się na rys. 6.



	p'2 (P1, Z1)	p'3 (~P1, ~Z1, PL1)	p'4 (~PL1, Z2, P2)	p'1 (~P2,~	p''1 (PL2)	p'8 (~K, P4)	p'7 (~P3, ~Z3, K)	p'6 (P3, Z3)	p''2	p'12 (~SP, P6)	p'11 (~P5, SP)	p'10 (P5)	p'9 (~P6)	p'5 (~P4)
M:	0,45	0	0	0,55	0	0	0	0	0	0	0	0	1	1

Rys. 6. Wygląd przykładowego wektora ***M*** w SFIPN.

Na rys. 7 znajduje się przykładowy screen trwającej symulacji. Dla tego stanu systemu w SFIPN wektor ***M*** wygląda następująco (miejsca są w kolejności od p'_1 do p'_{12} oraz p''_1, p''_2):

$$\boldsymbol{M} = [0{,}55 \quad 0{,}45 \quad 0 \quad 0 \quad 0{,}55 \quad 0{,}45 \quad 0 \quad 0 \quad 0{,}55 \quad 0{,}45 \quad 0 \quad 0 \quad 0{,}55/4 \quad 1{,}1/4]$$

Wektor ***K*** przyjmuje postać (kolejność miejsc ta sama):

$$\boldsymbol{K} = [1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 4 \quad 4]$$

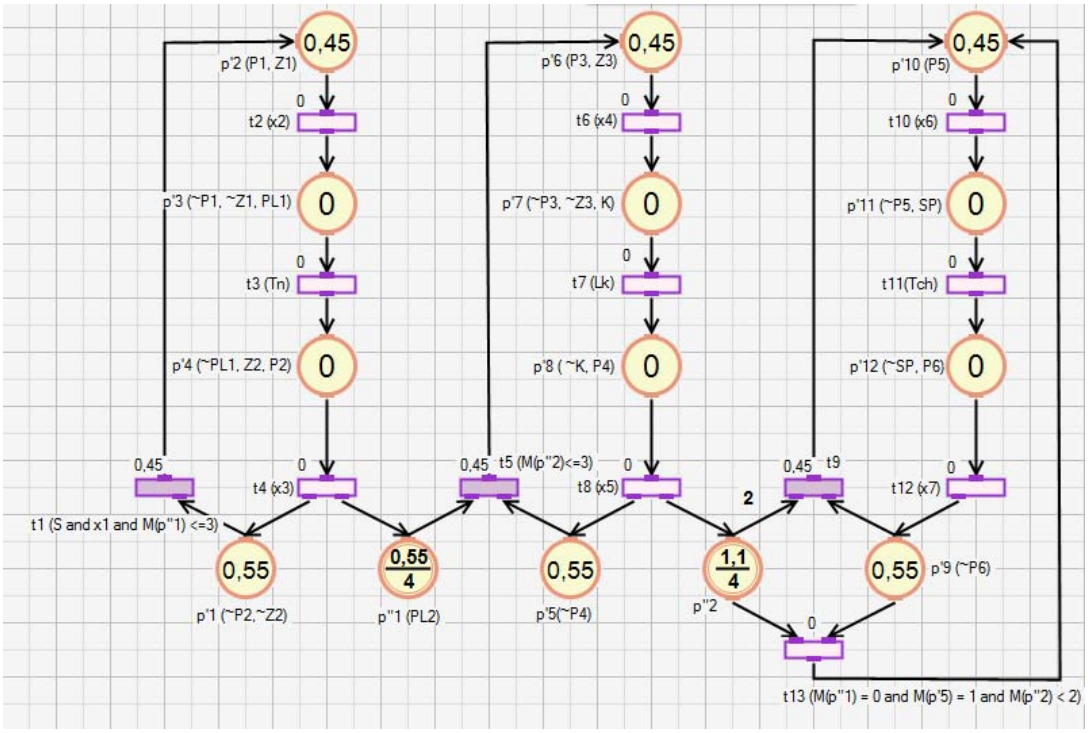
Wektory ***U*** i $\Delta\boldsymbol{\Theta}$ (tranzycje są ułożone od t_1 do t_{13}):

$$\boldsymbol{U} = [1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0]$$

$$\boldsymbol{\Delta\Theta} = [0{,}45 \quad 0 \quad 0 \quad 0 \quad 0{,}45 \quad 0 \quad 0 \quad 0 \quad 0{,}45 \quad 0 \quad 0 \quad 0 \quad 0]$$

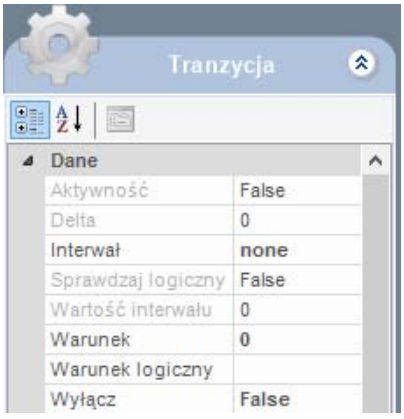
Macierz ***C*** ma następującą postać:

$$\boldsymbol{C} = \begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \\ t_5 \\ t_6 \\ t_7 \\ t_8 \\ t_9 \\ t_{10} \\ t_{11} \\ t_{12} \\ t_{13} \end{array} \begin{array}{c} \begin{array}{cccccccccccccc} p'_1 & p'_2 & p'_3 & p'_4 & p'_5 & p'_6 & p'_7 & p'_8 & p'_9 & p'_{10} & p'_{11} & p'_{12} & p''_1 & p''_2 \end{array} \\ \left[\begin{array}{cccccccccccccc} -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & -1 \end{array}\right] \end{array}$$



Rys. 7. Schemat fragmentu linii produkcyjnej w trakcie symulacji w SFIPN

Istotnym dla działania SFIPN jest możliwość dokonania różnego typu symulacji. Pierwszy sposób to symulacja krokowa, która daje możliwość ręcznego ustawiania aktywności tranzycji oraz stopnia ich spełnienia. W tym celu należy podać wartość liczbową w opcji 'Warunek' w oknie właściwości, które znajduje się na rys. 8. Okno właściwości wyświetla się po kliknięciu na tranzycję. Może się zdarzyć, że ten sposób symulacji będzie pracochłonny, ale zarazem niezbędny w przypadku wykonywania konkretnego testu w określonym miejscu modelowanego systemu. Drugi sposób to symulacja automatyczna, gdzie w podstawowej wersji są uaktywniane kolejne tranzycje w zależności od posiadania miejsc wejściowych, w których znajduje się odpowiednia liczba znaczników. W tego typu symulacji jest możliwość ustawienia czasu, co jaki zmienia się aktywność tranzycji oraz różnica o jaką zmienia się stopień spełnienia tranzycji. W automatycznym trybie symulacji możliwe jest również wybranie losowego przyznawania koncesji dla tranzycji. Może to być przydatne do zwiększenia zakresu testów modelowanego systemu.



Rys. 8. Widok opcji symulacyjnych tranzycji.

W symulowanym przykładzie istotne miejsce zajmują warunki do uaktywnienia tranzycji w zależności od znakowania innych miejsc sieci. SFIPN umożliwia wpisywanie do każdej tranzycji warunku związanego z działaniem na liczbach całkowitych, zmiennoprzecinkowych oraz znakowaniem miejsc w sieci. Można skorzystać z funkcji logicznych and (*AND*(...)), or (*OR*(...)), funkcji sprawdzających czy dana liczba lub wynik działania jest większy (*IF*>(...)), równy (*IF*=(...)), mniejszy (*IF*<(...)), większy lub równy (*IF*>=(...)), mniejszy lub równy (*IF*<=(...)) od innego. Dostępne są funkcje dodawania, odejmowania, mnożenia i dzielenia (*ADD*(...), *SUB*(...), *MUL*(...), *DIV*(...)).Opis funkcji i przykłady ich użycia znajdują się w tabeli 2. W trakcie trwania symulacji może zajść potrzeba, aby symulować brak zewnętrznego zdarzenia synchronizującego aktywność tranzycji albo zatrzymać daną tranzycję z powodu np. awarii czujnika. Wtedy wystarczy wybrać opcję "Wyłącz", którą widać na Rys. 8. Możliwe jest również ustawienie prędkości przepływu znacznika przez wybraną tranzycję niezależnie od głównych ustawień symulacyjnych ("Interwał" na rys. 8).

Tabela 2. Przykładowe warunki logiczne możliwe w programie.

AND(IF=(M(P1),M(P2)),IF>(M(P3), 0))	Uaktywnianie tranzycji jeśli bieżące znakowanie dla miejsc p_1 i p_2 są równe i jeżeli dla miejsca p_3 jest większe od zera
OR(IF<(M(P4),DIV(M(P1),2)),IF>=(M(P5) , M(P1))	Uaktywnienie tranzycji jeśli znakowanie dla p_4 jest mniejsze od dzielenia znakowania dla p_1 przez 2 lub jeżeli znakowanie dla p_5 jest większe lub równe od znakowania dla p_1

Wartość dydaktyczna SFIPN

Obecnie można zauważyć coraz większe wykorzystanie w praktyce różnego rodzaju sieci Petriego. Dowodzą tego choćby realizacje: [2, 3, 6, 9, 11]. Główną zaletą FIPN jest to, że można dzięki niej równocześnie diagnozować współbieżne procesy oraz nimi sterować. W związku z tym, istnieje realna potrzeba wprowadzenia tej sieci do edukacji. Zaprezentowana definicja formalna FIPN może nie być dla każdego zrozumiała przy pierwszym kontakcie. Dodatkowo pokazanie działania sieci z wykorzystaniem kartki lub za pomocą prezentacji multimedialnej wymaga dużego nakładu pracy. Dlatego też bardzo pomocne może być wykorzystanie symulatora opisanego w ramach artykułu w procesie dydaktycznym. Za pomocą kilku kliknięć możliwe staje się stworzenie różnych modeli sieci, zmiana właściwości poszczególnych elementów, dodanie lub usunięcie elementów. Dodatkowo za pomocą symulatora można w bardzo krótkim czasie pokazać przepływ znaczników w sieci, wykryć konflikty w sieci oraz miejsca zagrożone rozlaniem się żetonu. Dzięki tym wszystkim możliwościom, które daje symulator, można powiedzieć, że jest on niezbędny w procesie dydaktycznym. Z pewnością nie wykluczy on teorii związanej z FIPN, ale może przyśpieszyć zrozumienie pojęć, które będą w ramach niej użyte, przez co stanie się wartościowym uzupełnieniem.
Dodatkową zaletą symulatora jest to, że może być uruchomiony przez ucznia samodzielnie w domu. Dzięki temu narzędzie pozwala wspierać e-learning. Tym bardziej, jeśli w ramach zajęć praktycznych powiązanych z sieciami Petriego będą uruchamiane rzeczywiste systemy.

Podsumowanie

Głównym celem artykułu jest pokazanie rosnącej potrzeby wprowadzenia coraz szerszego zakresu wiedzy na temat sieci Petriego do edukacji oraz konieczności rozwijania narzędzi, które mogą ułatwić i przyśpieszyć ten proces. Dzięki symulatorom możliwe jest łatwiejsze zrozumienie definicji sieci Petriego, ich ograniczeń oraz zasad przepływu znacznika w sieci. W ramach artykułu wykorzystano symulator będący własnością Katedry Informatyki i Automatyki na Wydziale Elektrotechniki i Informatyki Politechniki Rzeszowskiej. Wprowadzono do niego następujące modyfikacje:
a) dostosowano do najnowszej definicji i ograniczeń związanych z FIPN;
b) rozszerzono funkcjonalność związaną z uaktywnianiem tranzycji na warunek logiczny,

wyłączanie tranzycji, zwiększono zakres modyfikowanych interwałów tranzycji;
c) dodano funkcjonalności: kopiuj, wklej element, cofnij, powtórz, zaznacz, zaznacz wszystko, wyświetl id i nazwy miejsc;
d) zmodyfikowano zaznaczanie elementów, umożliwiające ich zbiorowe kopiowanie, wklejanie, przemieszczanie oraz usuwanie;
e) zwiększono efektywność dodawania, budowania, zapisywania sieci oraz dodano skróty klawiszowe, które ułatwiają obsługę symulatora.
W ramach artykułu również przeprowadzono symulacje wybranego przykładu, który potwierdził użyteczność symulatora do celów dydaktycznych. W wyniku przeprowadzonej symulacji udowodniono również, że symulator rozmytej interpretowanej sieci Petriego pokazany w artykule może być również wykorzystywany w praktyce.
Pomimo iż symulator posiada wiele zalet, to możliwe jest jego dalsze ulepszanie. Przykładowe kierunki rozwoju tego narzędzia to: wprowadzenie hierarchiczności do modelowanej sieci Petriego oraz do symulatora; dodanie funkcjonalności automatycznego generowania kodu używanego przez sterowniki przemysłowe np. w języku ST na podstawie utworzonego schematu sieci w symulatorze; modyfikacje zwiększające atrakcyjność i użyteczność interfejsu graficznego dla zastosowań edukacyjnych i praktycznych. Tej tematyce badań zostaną poświęcone dalsze prace.

Bibliografia

1. AIS group: *CPN Tools 4.0, http://cpntools.org/*, Eindhoven University of Technology, The Netherlands (dostęp 01.07.2015).
2. Chen, T.M., Sanchez-Aarnoutse, J.C., Buford, J., Petri net modeling of cyber-physical attacks on smart grid, *Smart Grid, IEEE Transactions on*, *2*(4), 2011, s. 741-749.
3. Chiang, W., Liu, K.F., Lee, J., Bridge damage assessment through fuzzy Petri net based expert system, *Journal of computing in civil engineering*, 14(2), 2000, s. 141-149.
4. David R., Alla, H., Discrete, Continuous, and Hybrid Petri Nets*,* Berlin, Germany: Springer-Verlag, 2005.
5. Gniewek, L., Modelowanie i synteza układów sterowania z wykorzystaniem rozmytej interpretowanej sieci Petriego, Oficyna Wydawnicza Politechniki Rzeszowskiej, Rzeszów 2012.
6. Gniewek, L., Sequential control algorithm in the form of fuzzy interpreted Petri net, Systems, Man, and Cybernetics: Systems, *IEEE Transactions on*, 43(2), 2013, s. 451-459.
7. GreatSPN, http://www.di.unito.it/~greatspn/index.html*,* Dipartimento di Informatica, Università di Torino (dostęp 01.07.2015).
8. Heiner, M., Herajy, M., Liu, F., Rohr, C., Schwarick, M., Snoopy–a unifying Petri net tool, Application and Theory of Petri Nets*,* Springer Berlin Heidelberg, 2012, s. 398-407.
9. Herajy, M., Schwarick, M., A hybrid Petri net model of the eukaryotic cell cycle, In: *Proc. 3th International Workshop on Biological Processes and Petri Nets,* June 2012, s. 29-43.
10. Jensen, K., Kristensen, L. M., Wells, L., Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems, *International Journal on Software Tools for Technology Transfer*, *9*(3-4), 2007, s. 213-254.
11. Kyriakarakos, G., Dounis, A. I., Arvanitis, K. G., Papadakis, G., A fuzzy cognitive maps–petri nets energy management system for autonomous polygeneration microgrids, *Applied Soft Computing*,*12*(12), 2012, s. 3785-3797.
12. Nagasaki, M., Saito, A., Jeong, E., Li, C, Kojima, K., Ikeda, E., Miyano, S., Cell Illustrator 4.0: A computational platform for systems biology, In Silico Biol, 2010.
13. Petri, C.A., Kommunikation mit Automaten. Schriften des Institutes fur Intrumentelle Mathematik, Bonn, 1962.
14. SFIPN, http://mmarkiewicz.sd.prz.edu.pl/pl/67/art8336.html (dostęp 01.07.2015).