POLITECHNIKA KRAKOWSKA
IM. TADEUSZA KOŚCIUSZKI
WYDZIAŁ FIZYKI MATEMATYKI I INFORMATYKI
KIERUNEK INFORMATYKA

Grzegorz Piguła

Symulator czasowych sieci Petriego

Time Petri net simulator

PRACA MAGISTERSKA
STUDIA NIESTACJONARNE

Ocena: ……..
Podpis promotora: ……
Promotor: Dr inż. Jerzy Raszka

Kraków 2015

Spis treści

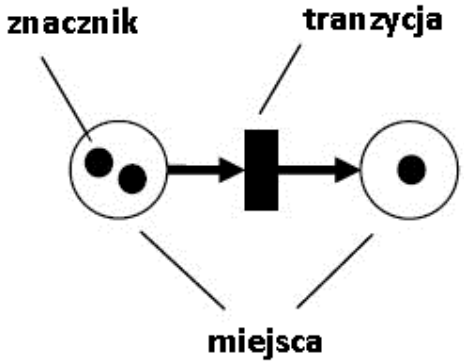
1. Wstęp

1.1. Wprowadzenie

Sieci Petriego wprowadzone przez Carla A. Petriego w latach 60-tych XX wieku stosowane były początkowo do graficznej i matematycznej analizy algorytmów a w szczególności do realizacji komunikacji z automatami. Obecnie narzędzie to wykorzystywane jest w różnych działach nauki do badania procesów przebiegających równolegle w złożonych systemach [1]. „*Charakteryzuje je intuicyjny graficzny język modelowania, wspierany przez zaawansowane metody formalnej analizy ich własności*[1].” Sieci Petriego dzięki możliwości modelowania zadań wykonywanych równocześnie mogą być zastosowane także do symulacji dyskretnych systemów współbieżnych.

Sieć Petriego, która w sposób adekwatny z punktu widzenia celu rozważań modeluje rzeczywiste procesy, umożliwia ich analizę, symulację i weryfikację. Sieć Petriego opiera się na grafie dwudzielnym składającym się z dwóch odrębnych zbiorów wierzchołków określanych, jako miejsca i tranzycje oraz zbioru krawędzi skierowanych, które muszą łączyć różne typy wierzchołków. Do modelowania stałej struktury procesów wykorzystywane są elementy takie jak: miejsca, tranzycje, krawędzie, natomiast do modelownia dynamiki sieci używane są znaczniki umieszczane w miejscach. Do oznaczenia tranzycji (przejść) wykorzystuje się kwadrat lub prostokąt, a do oznaczenia miejsc okręgi (Rys. 1).

Działanie sieci Petriego można opisać następująco. Obecność znaczników na wszystkich miejscach wejściowych konkretnej tranzycji, aktywuje tą tranzycje. Aktywność tranzycji a następnie wykonanie przejścia polega na zabraniu znaczników ze wszystkich miejsc wejściowych i dodaniu ich do wszystkich miejsc wyjściowych tej tranzycji[3]. Ilość zabranych i wstawianych znaczników zależna jest od wagi krawędzi (domyślnie waga jest równa 1).



Rys.1 Graficzna reprezentacja elementów sieci.

Najczęściej spotykane sieci Petriego:

- *uogólnione,*
- *znakowane,*
- *czasowe,*
- *kolorowane,*
- *kolorowane czasowe,*
- *hierarchiczne*

Ponadto istnieje jeszcze wiele różnych rozszerzeń sieci Petriego.

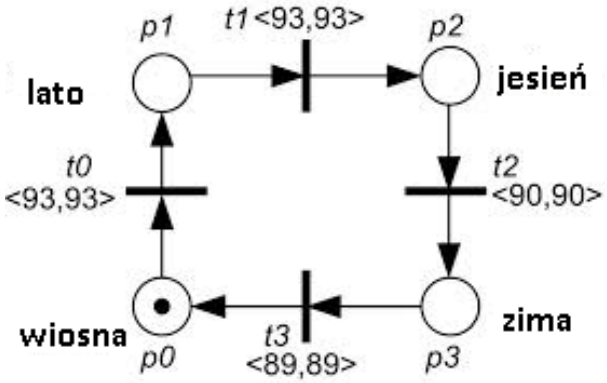
Sieć Petriego posiada właściwości:

- *strukturalne* – właściwości zależne od struktury sieci,
- *behawioralne* – niezależne od struktury, a od znakowania początkowego.

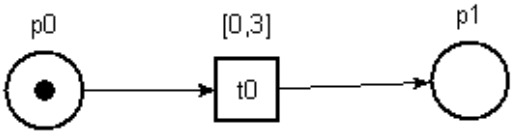
Właściwości behawioralne dzielimy na:

- *osiągalność* – sprawdzenie czy ze stanu początkowego M_0 możemy osiągnąć pożądany przez nas stan $M_{k,}$
- *ograniczoność* – sprawdzenie czy każde miejsce w sieci jest k-ograniczone, tzn., gdy ·w dowolnym miejscu liczba znaczników przy dowolnym stanie sieci nie przekracza liczby k; gdy k = 1 sieć jest bezpieczna,
- *zachowawczość* - sprawdzenie czy liczba znaczników w sieci jest stała,
- *żywotność* - sprawdzenie czy każda tranzycja w sieci może się wykonać,
- *odwracalność* – sprawdzenie czy dla każdego ze stanów osiągalnych możliwy jest powrót do stanu początkowego[4].

Czasowe sieci Petriego są najczęściej dzieli się na *proste sieci czasowe i przedziałowe sieci czasowe* [1]. W prostych sieciach czasowych i przedziałowych sieciach czasowych restrykcja czasowa jest związana z tranzycjami [1]. „*Proste sieci czasowe zdefiniowano, jako rozszerzenie sieci znakowanych, a sieci przedziałowe, jako rozszerzenie sieci uogólnionych. W obu przypadkach definicja została wzbogacona o funkcje opóźnień określoną na zbiorze przejść. Dla prostych sieci czasowych opóźnienie wykonania przejścia jest nieujemną liczbą wymierną określającą czas wykonania tego przejścia liczony od chwili, w której było ono uaktywnione. W przypadku sieci przedziałowych podaje się przedział liczb wymiernych określających minimalne i maksymalne opóźnienie wykonania przejścia, przy czym opóźnienie maksymalne nie musi być liczba skończoną*[1].”



Rys.2 Prosta sieć czasowa.



Rys.3 Przedziałowa sieć czasowa.

Realizację prostej sieci czasowej modelującej zmiany kalendarzowych pór roku przedstawia Rys.2. Każda pora roku ma swój czas trwania w dniach. Sieć zaczyna się od wiosny. Tranzycja przejścia z wiosny do lata będzie aktywna po 93 jednostkach czasu. tj. dniach. Po wykonaniu przejścia będziemy mieli lato które trwa również 93 dni i dopiero po 93 dniach, nastąpi zmiana pory na jesień. Jesień trwa 90 dni, po tym czasie, nastąpi zmiana na zimę. Następnie po 89 dniach trwania zimy, nastąpi zmiana pory roku na wiosnę, stan pór roku powróci do stanu początkowego.

1.2. Cel i zakres pracy

Celem niniejszej pracy jest przedstawienie czasowych sieci Petriego, jako narzędzia do komputerowej symulacji różnych rodzajów systemów. Zostaną również zaprezentowane przykładowe istniejące aplikacje do komputerowej symulacji sieci Petriego. Głównym celem jest wykonanie algorytmu realizującego czasową sieć Petriego, a także programu do prezentacji wykonanego algorytmu.

Na początku pracy zaprezentowano podstawy opisujące sieć Petriego oraz jej rozwinięcie, jakim jest czasowa sieć Petriego. Ma to na celu wprowadzenie czytelnika w zasadę działania sieci Petriego. W drugim rozdziale przedstawione są istniejące aplikacje, celem pokazania jak takie programy wyglądają i realizują symulację sieci Petriego. W następnym rozdziale przedstawiany jest projekt aplikacji do symulacji czasowej sieci Petriego. W czwartym

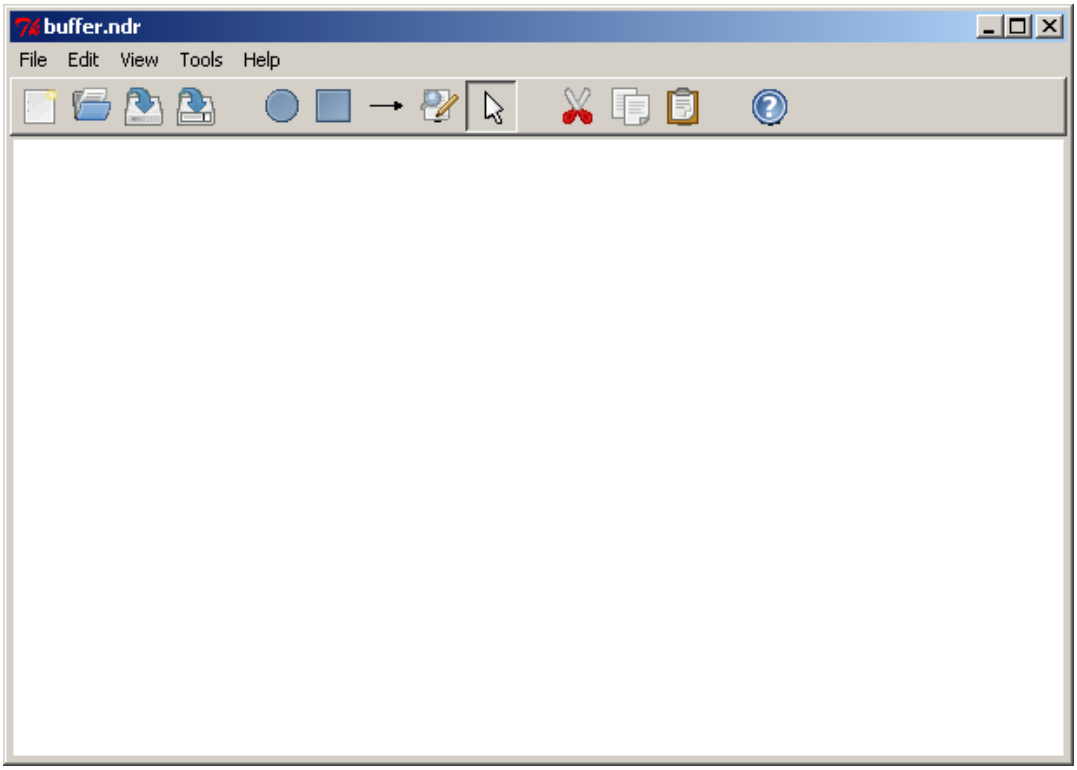
rozdziale prezentowana jest stworzona aplikacja, opisywany jest graficzny interfejs użytkownika, co ma na celu zapoznanie użytkownika z możliwościami aplikacji. Następnie opisywany jest algorytm realizujący czasową sieć Petriego i model struktury danych. W piątym rozdziale zostaną opisane testy sprawdzające poprawne funkcjonowanie interfejsu graficznego oraz poprawność przeprowadzanego procesu symulacji przykładowej czasowej sieci Petriego. Ostatni rozdział kończy się podsumowaniem niniejszej pracy.

2. Przegląd istniejących rozwiązań

2.1. Program ND (NetDraw)

Program ND jest narzędziem z zestawu narzędzi **TINA** (*TIme petri Net Analyzer*).

NetDraw jest graficznym edytorem sieci Petriego, czasowych sieci Petriego oraz automatów. Posiada graficzny oraz tekstowy opis sieci Petriego oraz automatów. Zawiera również *analizę osiągalności*, *analizę struktury*, *symulator krokowy* oraz *sprawdzenie poprawności sieci* [5].



Rys.4 Główne okno programu ND

W głównym oknie programu (Rys. 4) jest edytor sieci, na którym można umieszczać elementy sieci takie jak: *tranzycje*, *miejsca*, *krawędzie*. Każdy element ma swoje właściwości. W tranzycji (Rys.5) możemy zmienić takie ustawienia jak:

- *nazwa,*
- *etykieta,*
- *przedział czasu odpalenia tranzycji.*

Rys.5 Okno właściwości tranzycji

Natomiast w elemencie miejsce (Rys.5) możemy zmienić ustawienia:

- *nazwa,*

- *etykieta,*

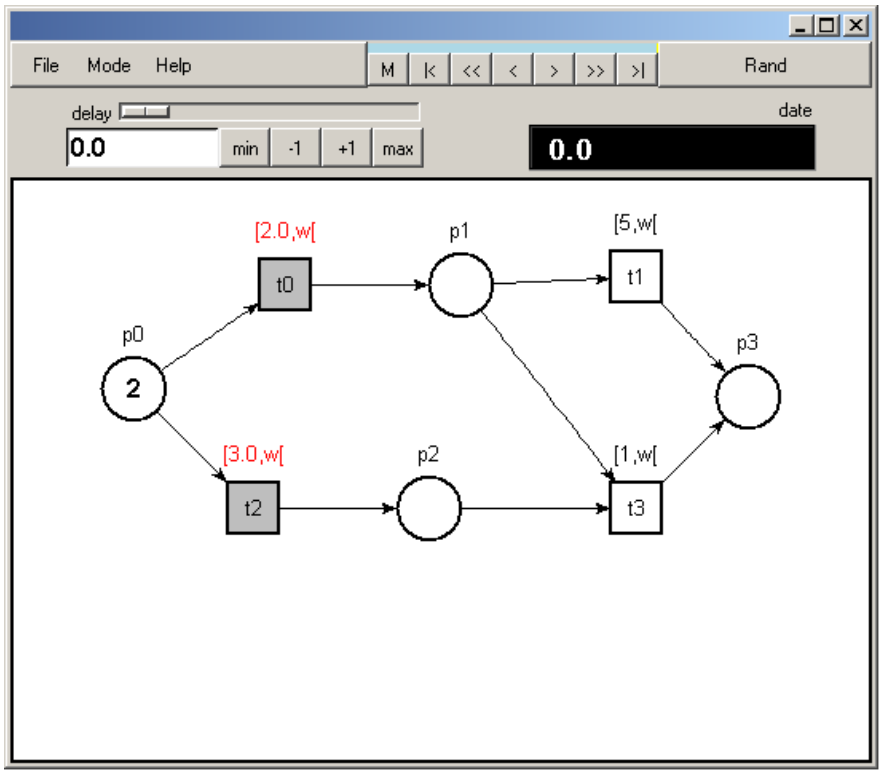
- *znakowanie początkowe (liczba znaczników).*

Rys.6 Okno właściwości miejsca

W ostatnim elemencie sieci w krawędzi, mamy możliwość zmiany wagi przepływu.

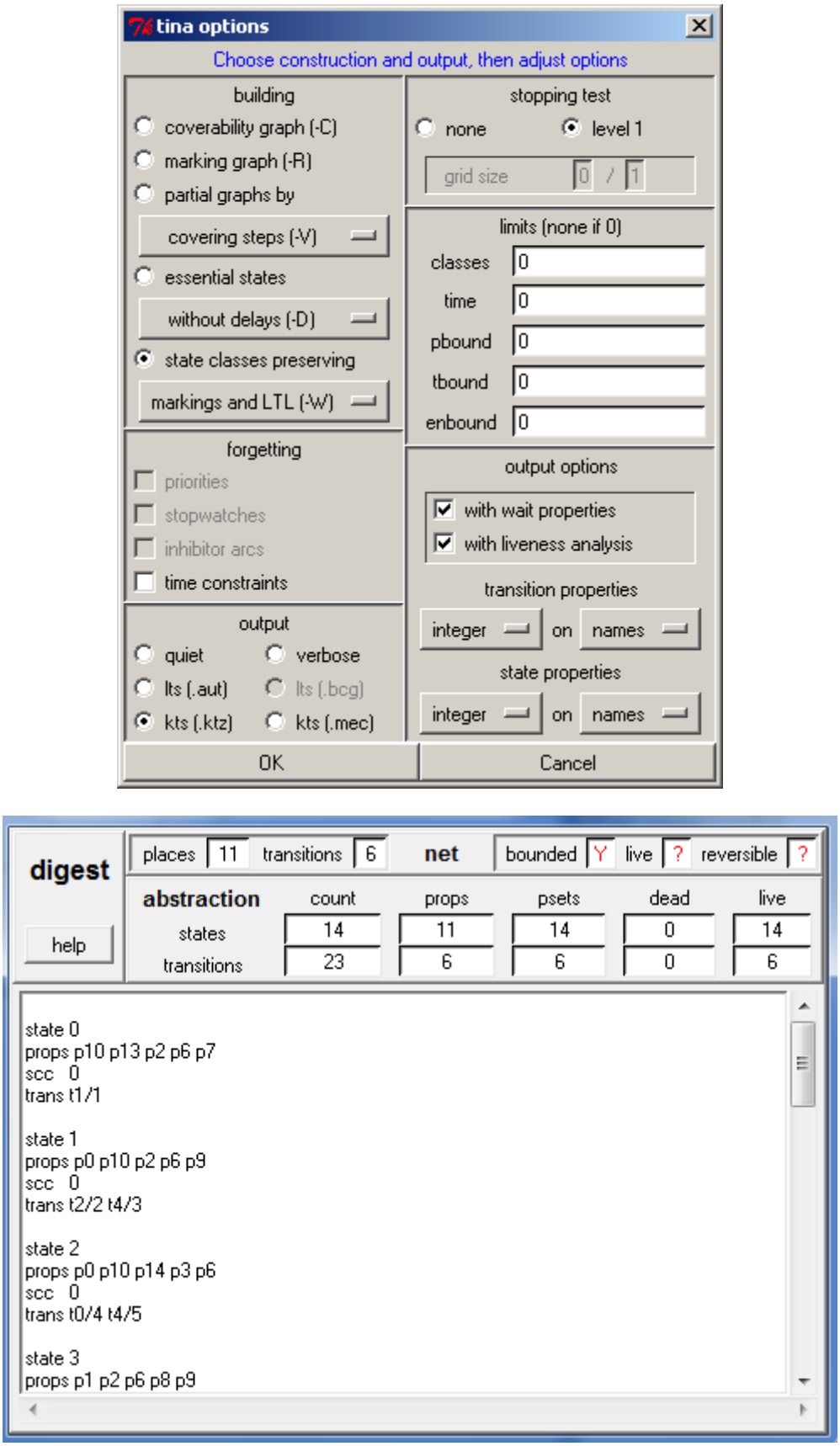
Rys.7 Okno właściwości krawędzi

Po wykonaniu w edytorze modelu sieci Petriego możemy przeprowadzić symulację sieci. W tym celu należy wejść opcje *Tools*, a następnie w „*stepper symulator*”. Po kliknięciu pokaże nam się okno symulacji (Rys. 8)

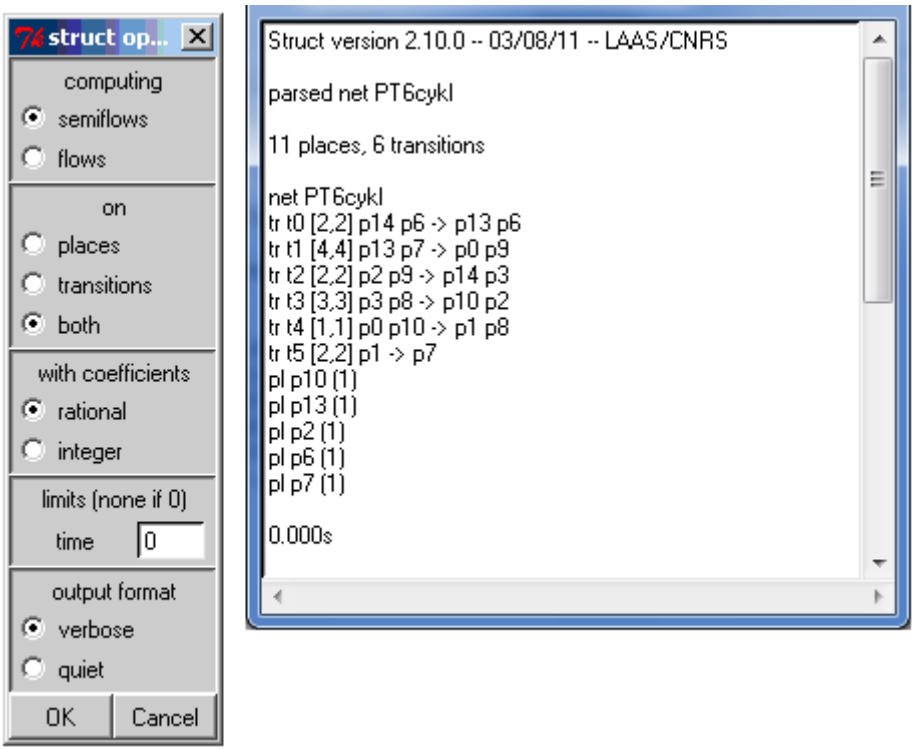


Rys.8 Okno symulacji

Symulację możemy przeprowadzić w sposób ręczny lub automatyczny. Przy ręcznej symulacji samodzielnie sterujemy czasem oraz odpaleniem wybranej aktywnej tranzycji.
Automatyczna symulacja samodzielnie steruje czasem oraz wybraniem aktywnej tranzycji. Przy automatycznej symulacji możemy zmienić opóźnienie w animacji uruchomień tranzycji, które pozwoli nam w sposób kontrolowany obserwować proces symulacji.
W obydwu przypadkach możemy prześledzić proces symulacji od początku do końca.
Istnieje też możliwość zapisu historii do pliku. Narzędzie do analizy osiągalności (Rys 9) pozwala na wyświetlenie statystyk sieci, sprawdzenie czy sieć jest żywa, ograniczona, odwracalna oraz na sprawdzenie stanów markowania, grafu osiągalności oraz grafu pokrycia (Rys 9). Ponadto narzędzie do analizy struktury sieci (Rys 10) generuje wyniki na podstawie przepływów w sieci. Określa również niezmienniki miejsc i tranzycji.



Rys.9. Okno statystyk sieci i przykład wyniku dla grafu pokrycia



Rys.10 Okno z ustawieniami generowania struktury sieci

2.2. Program PIPE (Platform Independent Petri net Editor)

Program PIPE jest *opensourcowym*, niezależnym systemowo narzędziem do tworzenia i analizy **GSPN** (*uogólnione stochastyczne sieci Petriego*) [6]. Dzięki łatwemu w użyciu graficznego interfejsu użytkownika można tworzyć, zapisywać, wczytywać i analizować sieci Petriego.

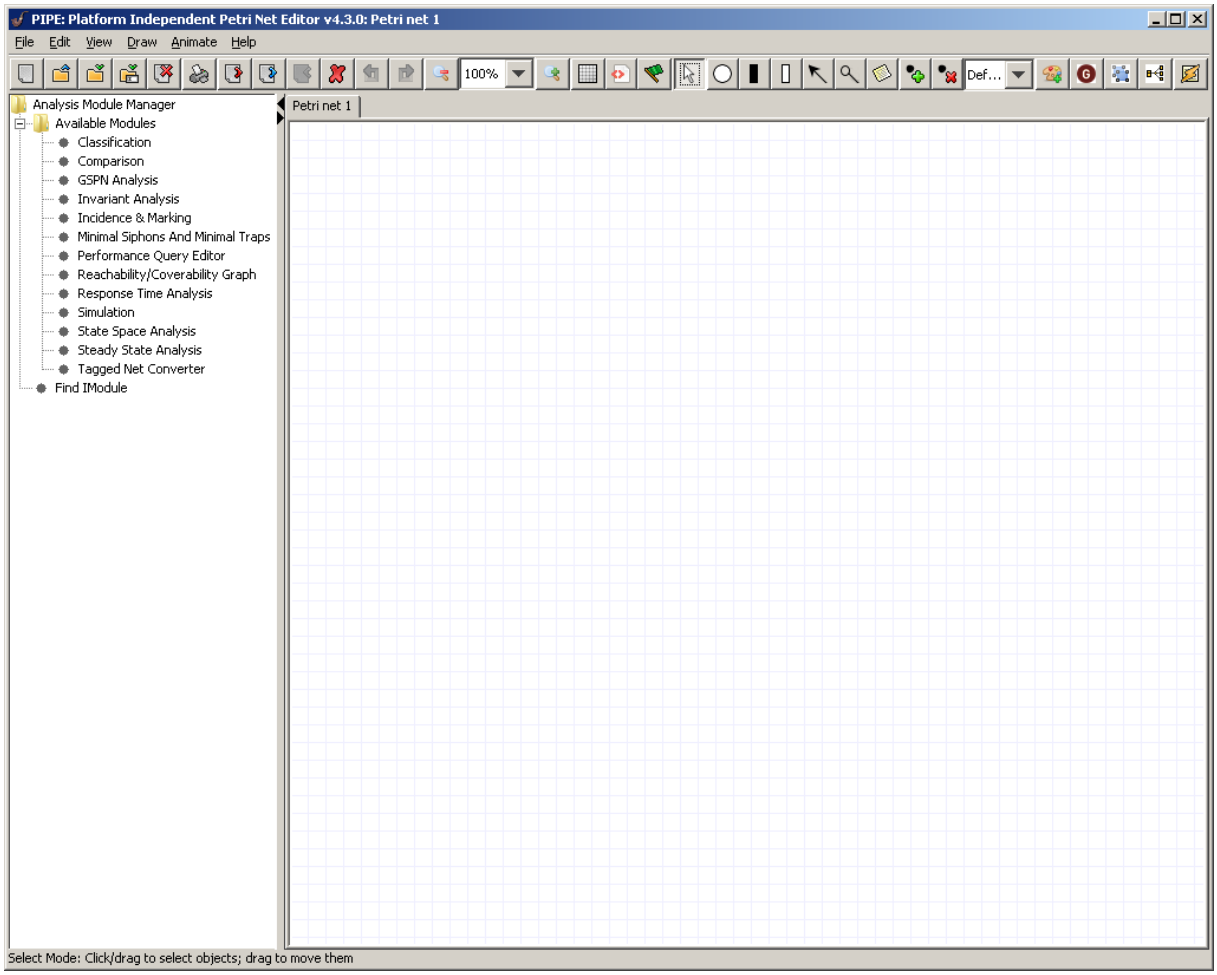
Aplikacja PIPE zawiera podstawowe moduły do analizy sieci Petriego takie jak:

- *classification* - bazując na połączeniach pomiędzy miejscami i tranzycjami klasyfikuje sieć Petriego,
- *comparison* – porównanie dwóch sieci Petriego,
- *GSPN analysis* – oblicza średnią liczbę znaczników w miejscu oraz wyznacza możliwe stany markowania sieci,
- *invariant analysis* – określanie niezmienników miejsc i tranzycji,
- *incidence & marking* – wyświetla macierze incydencji oraz znakowania,
- *minimal syphons and minimal traps* – wyznacza minimalną ilość miejsc, które nigdy nie zyskają znaczników oraz miejsc, które nigdy nie stracą wszystkich znaczników,
- *reachability / coverability graph* – prezentuje graf osiągalności i pokrycia dla możliwych sekwencji przejść podanej sieci Petriego,

- *simulation* – analiza symulacji, oblicza średnią liczbę znaczników przypadających na miejsce w trakcie całego czasu symulacji,

- *state space analysis* - analiza żywotności, zachowawczości oraz bezpieczeństwa sieci,

- *steady state analysis* - wyznacza możliwe stany markowania sieci [6].



Rys.11 Główne okno programu PIPE

W oknie edytora możemy wstawiać elementy sieci: miejsca, tranzycje, krawędzie, dodawać i odejmować znaczniki, wstawić notatkę.

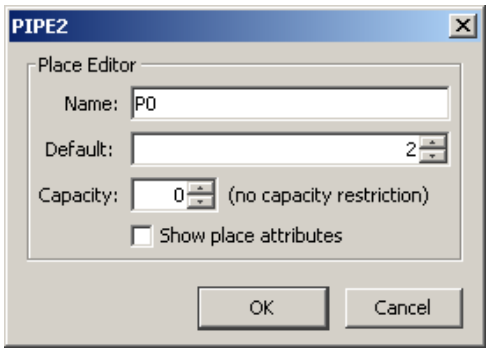
Każdy element posiada właściwości, które możemy zmienić.

Miejsce posiada opcje:

- *zmiany nazwy,*

- *zmiana liczby znaczników,*

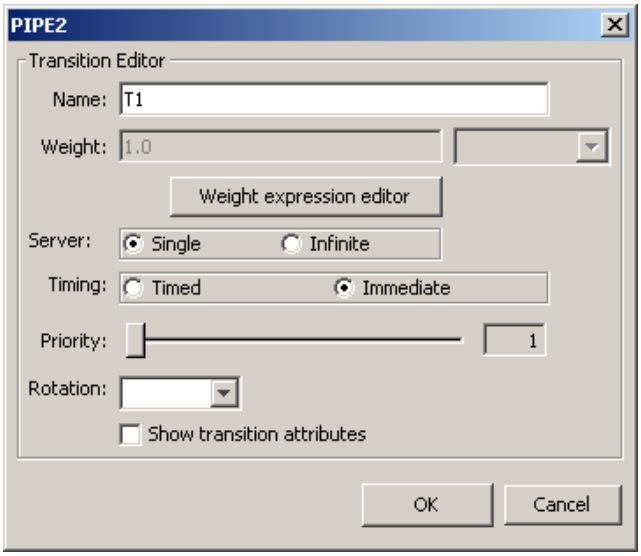
- *zmiana pojemności miejsca.*



Rys.12 Okno właściwości miejsca

W oknie właściwości tranzycji możemy ustawić takie rzeczy jak:

- *nazwa,*
- *typ tranzycji, czy jest to tranzycja czasowa czy nie,*
- *waga lub wartość czasu, w zależności od ustawienia typu tranzycji,*
- *priorytet,*
- *odpalenie tranzycji, za pobraniem znacznika lub dowolnie,*
- *kąt obrócenia tranzycji.*



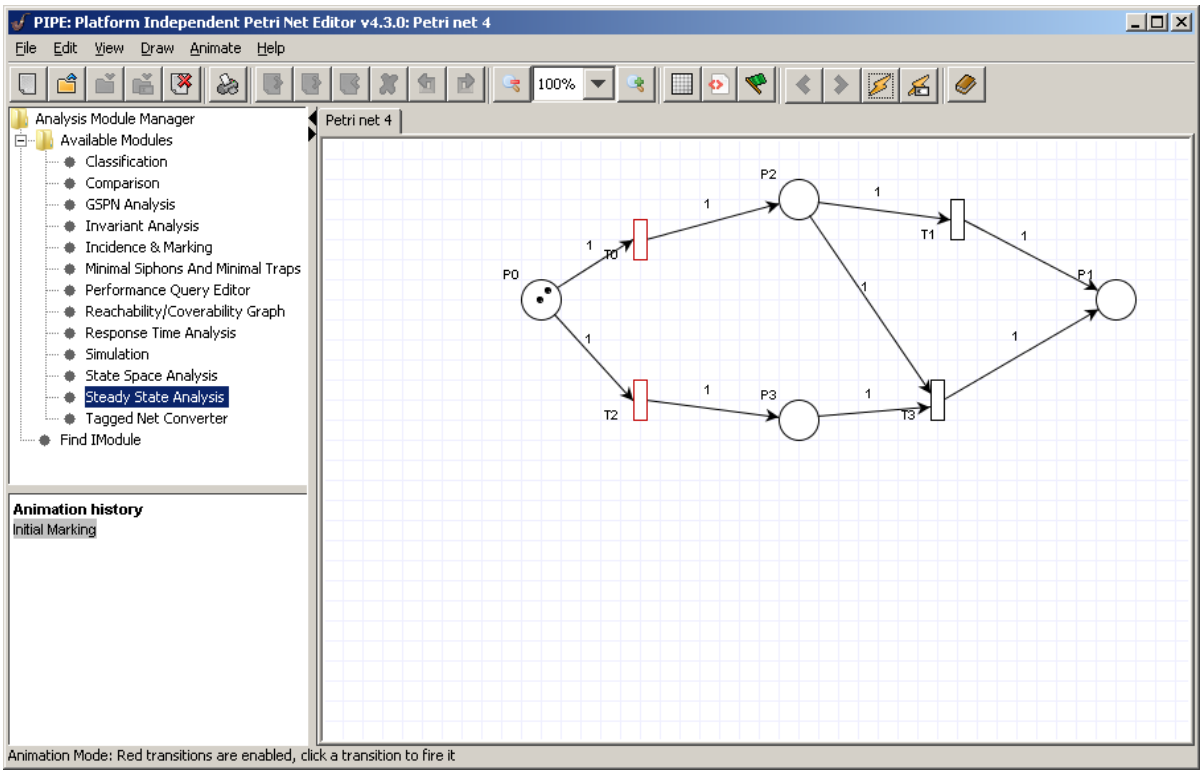
Rys.13 Okno właściwości tranzycji

We właściwościach krawędzi możemy zmienić tylko wagę krawędzi.

Gdy już stworzyliśmy sieć możemy przejść do trybu symulacji.

W trybie symulacji możemy ręcznie wskazać tranzycje do odpalenia lub odpalić ją losowo, istnieje też opcja odpalenia wybranej liczby tranzycji losowo, z opóźnieniami pomiędzy odpaleniami w ms.

W lewym dolnym okienku wyświetlana jest historia odpaleń tranzycji.



Rys.14 Okno z animacją symulacji

2.3. Program Symulator sieci Petriego ze znacznikami indywidualnymi

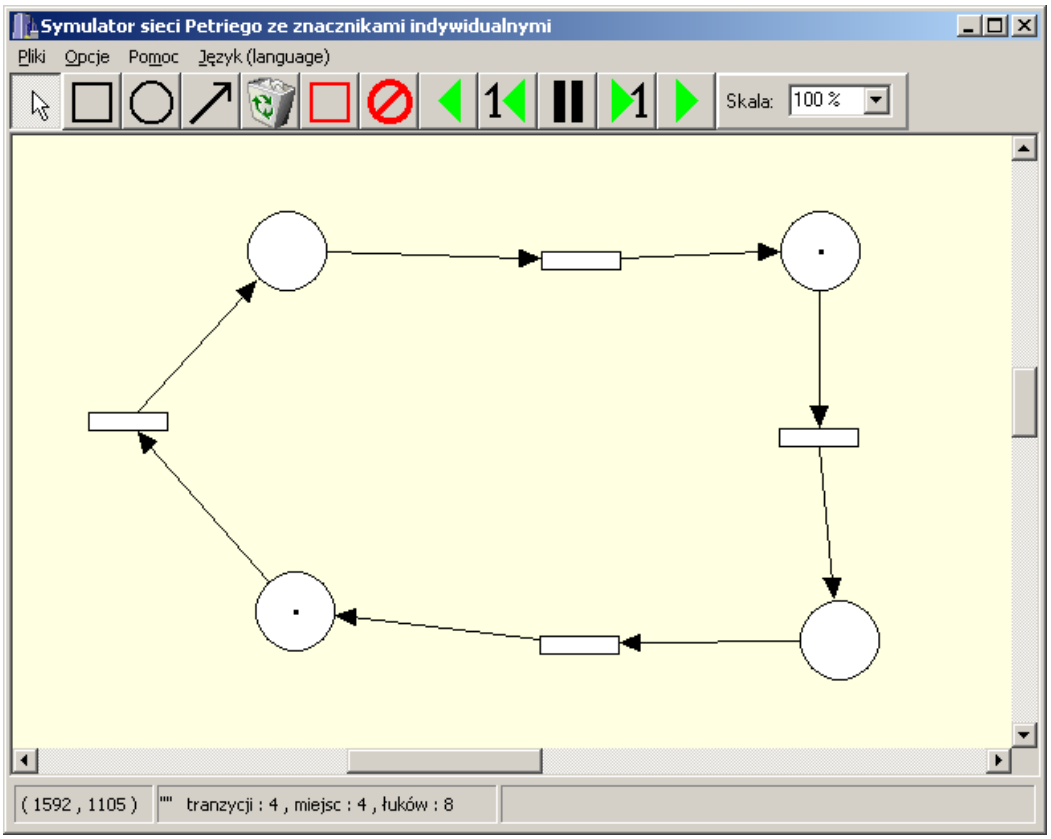
Symulator sieci Petriego ze znacznikami indywidualnymi jest narzędziem do tworzenia, analizy i symulacji sieci Petriego. Obsługiwane typy sieci to: PT sieci, sieci ze znacznikami indywidualnymi i stałymi oraz zmiennymi etykietami łuków.

Program posiada łatwy w użyciu graficzny interfejs, za którego pomocą możemy stworzyć sieć oraz przeprowadzić automatyczną symulację.

Siec możemy stworzyć z takich elementów jak: tranzycje, miejsca oraz krawędzie.

Dodany element możemy przenosić, usuwać oraz wyświetlić właściwości.

Tranzycji możemy nadać priorytet wysoki albo niski, możemy również zablokować tranzycje[9].



Rys.15 Okno programu Symulator sieci Petriego ze znacznikami indywidualnymi

W oknie edycji miejsca możemy zmienić:

- *nazwę,*

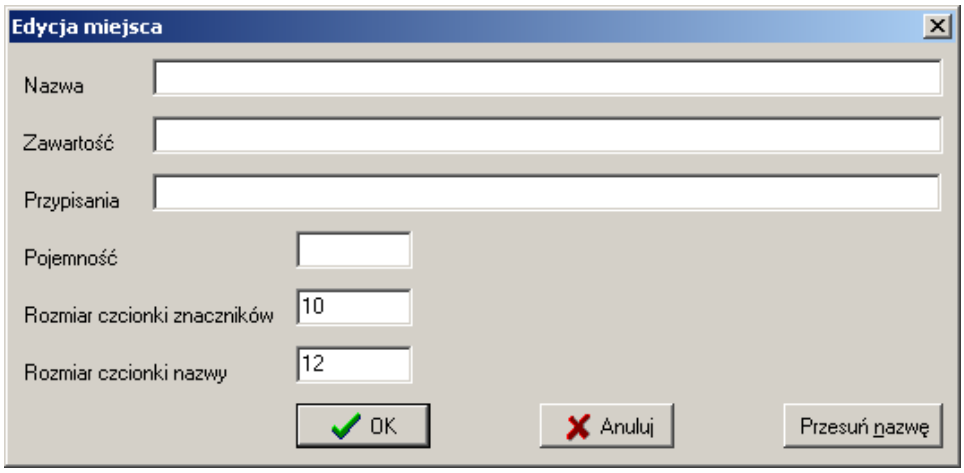
-zawartość,

- *przypisanie,*

- *pojemność,*

- *rozmiar czcionki znaczników,*

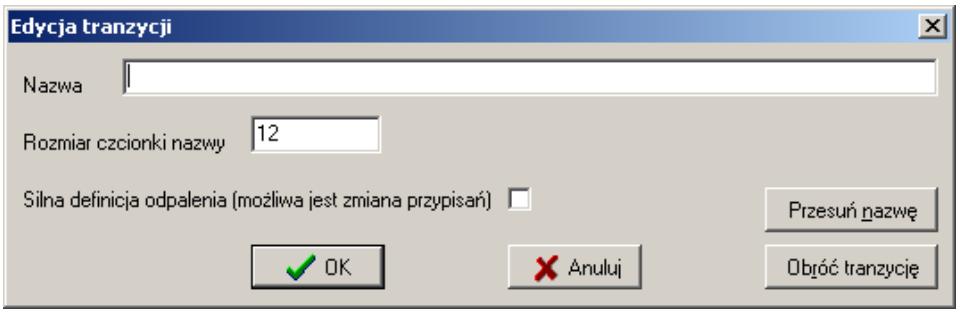
- *rozmiar czcionki, nazwy.*



Rys.16 Okno z właściwościami miejsca

We właściwościach tranzycji możemy zmienić:

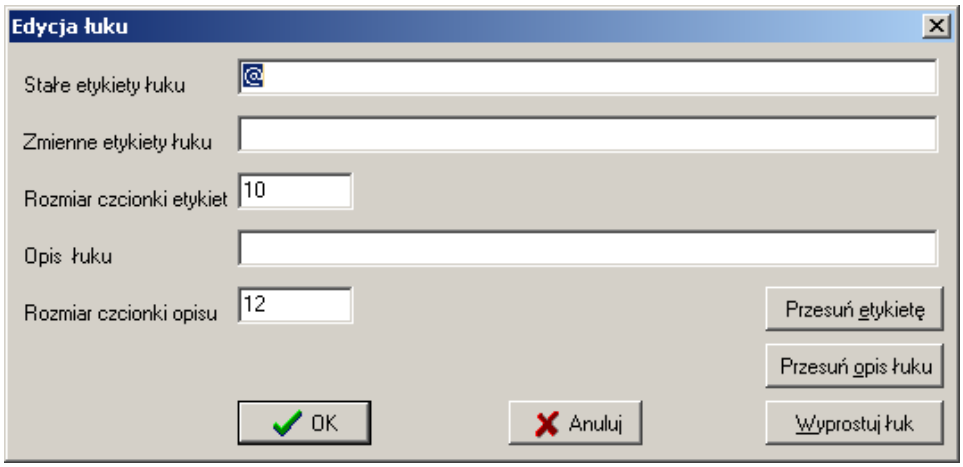
- *nazwę,*
- *rozmiar czcionki nazwy,*
- *priorytet odpalenia.*



Rys.17 Okno z właściwościami tranzycji

W oknie edycji łuku możemy zmienić:

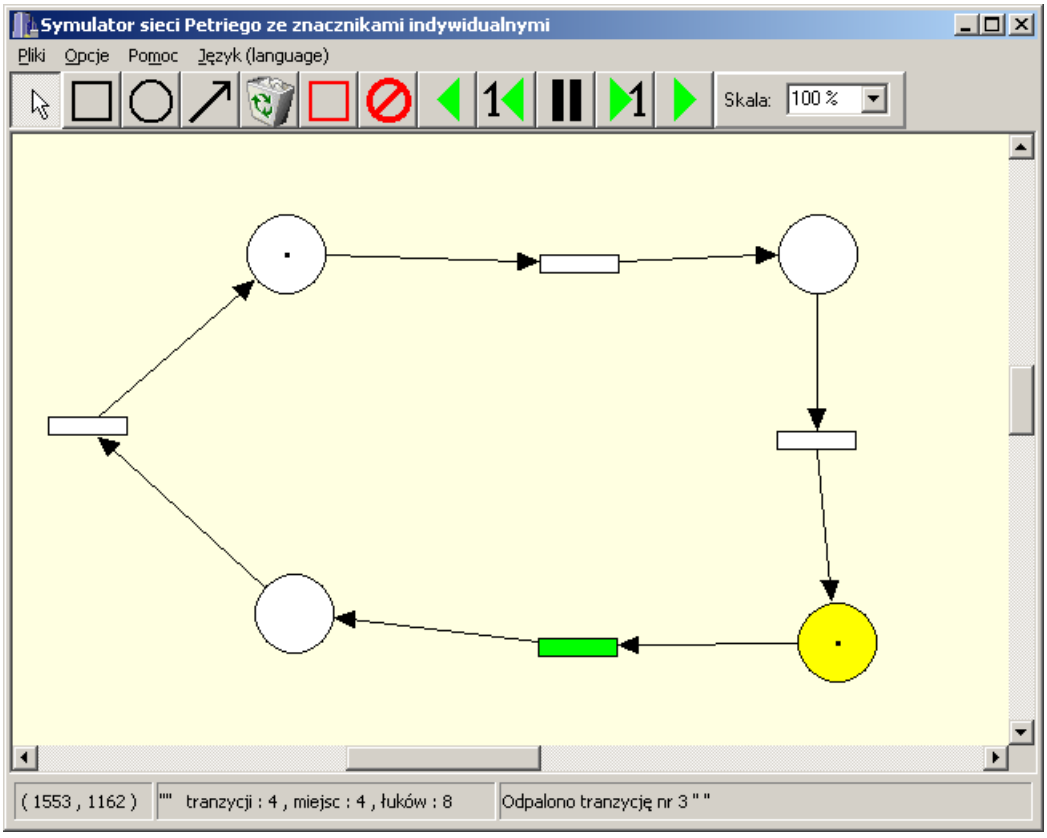
- *stałą etykietę łuku,*
- *zmienną etykietę łuku,*
- *rozmiar czcionki etykiet,*
- *opis łuku,*
- *rozmiar czcionki opisu.*



Rys.18 Okno z właściwościami łuku

Okno edycji jest również oknem symulacji. Symulację możemy włączyć w każdym momencie. Proces automatycznej symulacji możemy śledzić po jednym kroku lub uruchomić symulację ciągłą. Symulację możemy włączyć również w wstecz.

Jeden krok symulacji jest reprezentowany przez dwie akcje. Akcja pobrania znacznika z miejsca, które jest oznaczone kolorem żółtym do tranzycji oznaczonej kolorem zielony, drugą akcją jest przekazanie z tranzycji znacznika do miejsca oznaczonego kolorem niebieskim.



Rys.19 Okno programu przedstawia proces symulacji

W programie znajduje się zintegrowany analizator sieci **INA** (*Integrated Net Analyzer*). Jest on w stanie dostarczyć nam macierz incydencji, graf osiągalności, niezmienniki miejsc i tranzycji.

W oknie ustawień programu, są zakładki:

- *opisy* – za jej pomocą możemy dodać opis, który jest wyświetlany w oknie edycji,
- *edytor* - ustawienia rozmiarów elementów, rozmiar planszy edycji sieci,
-*symulator* – ustawienia czasu trwania jednej fazy animacji, przerwy między fazami, zabezpieczenie przed migotaniem oraz ilość faz animacji,
- *analizator* – głębokość analizy grafu osiągalności, czcionkę i rozmiar czcionki analizatora,
- *kolory* – kolory edytora, symulacji,
- *czcionki* – domyślne rozmiary i czcionki dla elementów sieci.

3. Projekt

3.1. Wymagania funkcjonalne

Program powinien zapewnić możliwość tworzenia, edycji, symulacji czasowych sieci Petriego oraz pokazania na wykresie przebiegu symulacji.

Użytkownik korzystający z aplikacji powinien, zatem móc:

- tworzyć nowe modele sieci,
- modyfikować istniejące modele,
- zapisywać oraz otwierać stworzone w aplikacji struktury sieci,
- symulować sieć,
- generować wykres Gantta przeprowadzonej symulacji.

3.2. Wymagania niefunkcjonalne

Aplikacja jest przenośna tzn. działa na różnych platformach systemowych i jedynym wymaganiem jest zainstalowane środowisko Java.

Interfejs aplikacji:

graficzny ułatwiający korzystanie z aplikacji,

obsługiwany jedną ręką z klawiatury lub za pomocą myszki

Aplikacja:

przeznaczona do używania przez pojedynczego użytkownika,

płynne działanie na większości urządzeń.

nie powinna wymagać szkolenia, użyte ikony powinny być intuicyjne.

Zasoby dyskowe ograniczone aplikacją i plikami danych i wyników poniżej 1 MB

3.3. Wybór technologii

Aplikacja jest tworzona w oparciu o technologię **Java**, wykorzystałem tą technologię ze względu na istnienie darmowego oprogramowania do tworzenia aplikacji, jakim jest **Netbeans**, pozwalające na tworzenie GUI za pomocą wbudowanych kontrolek w Swingu, a także na dostęp do różnych bibliotek rozszerzających możliwości tworzenia aplikacji.

Do wygenerowania wykresu Gantta została użyta darmowa biblioteka **JFreeChart**. Biblioteka jest dobrze udokumentowana oraz posiada liczne przykłady, co umożliwia wykorzystanie biblioteki do stworzenia wykresu Gantta z symulacji sieci Petriego.

Do zapisu struktury sieci do pliku wykorzystana została technologia **XML**. Technologia ta pozwala na bardzo łatwy zapis struktury sieci do pliku z wykorzystaniem

z wbudowanej specyfikacji **JAXB** (*The Java Architecture for XML Binding*). JAXB pozwala na zapis i odczyt klas języka Java do/z struktury xml.

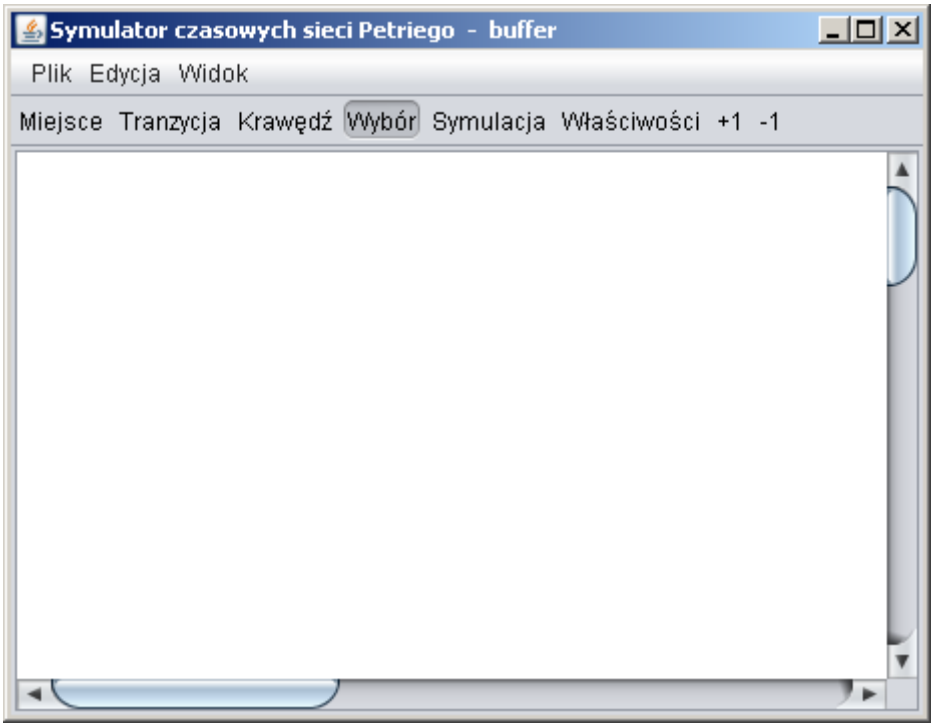
3.4. Diagram UML przypadków użycia

Rys.20 Diagram UML przypadków użycia

4. Implementacja

4.1. Graficzny interfejs użytkownika

Aplikacja posiada graficzny interfejs użytkownika ułatwiający tworzenie sieci. Za jego pomocą można wykonać takie rzeczy jak: stworzyć nowy model oraz otworzyć już zapisany, zapisać aktualnie edytowany, przeprowadzić symulacje oraz wygenerować wykres Gantta. Generowany jest również plik zawierający statystyczne parametry sieci.



Rys.21 Główne okno aplikacji

Główne okno programu (Rys. 21) zawiera pasek menu, pasek narzędzi i pole rysowania z paskami przewijania. Na pasku tytułu aplikacji znajduje się nazwa aktualnie edytowanego pliku.

Menu składa się z trzech podmenu: *Plik, Edycja, Widok*.

Podmenu *Plik* posiada takie pola jak:

- *Nowy* – usuwa istniejący model sieci i pozwala na stworzenie nowego,
- *Otwórz* – wczytuje model sieci z pliku,
- *Zapisz* – zapisuje model sieci do pliku,
- *Koniec* – zamyka aplikacje.

Podmenu *Edycja* zawiera pola, które występują na pasku narzędzi, z wyświetlonymi skrótami klawiszowymi.

Podmenu *Widok* posiada następującą pola:

- *Pokaz ID* – włącza lub wyłącza wyświetlanie ID tranzycji i miejsc.

Pasek narzędzi zawiera następujące pozycję:

- *Miejsce* – włącza tryb dodawania miejsc,
- *Tranzycja* – włącza tryb dodawania tranzycji,
- *Krawędź* – włącza tryb dodawania krawędzi,
- *Wybór* – włącz tryb kursora,
- *Symulacja* – włącza okno symulacji,
- *Właściwości* – włącza tryb właściwości wybranego elementu sieci,
- *+1* – włącza tryb dodawania jednej jednostki znacznika lub czasu w zależności od wybranego elementu,
- *-1* – włącza tryb odejmowania jednej jednostki znacznika lub czasu w zależności od wybranego elementu.

Po włączeniu raz trybu jest on aktywny, aż do wyłączenia go poprzez klawisz skrótu, kliknięcie jeszcze raz w pozycję używanego trybu, przejście w innym tryb lub poprzez klawisz wyjścia *ESC*.

Dodawanie miejsc do pola edycji.

Aby dodać miejsce do sieci należy być w trybie dodawania miejsc, po włączeniu trybu przycisk odpowiadający za tryb będzie zaciemniony możemy dodawać miejsca poprzez kliknięcie lewym klawiszem myszy w polu edycji. Po kliknięciu pojaw się nam miejsce wyświetlane, jako okrąg z wpisaną w środku ilością znaczników.

Dodawanie tranzycji do pola edycji.

Dodawanie tranzycji wygląda analogicznie do dodawania miejsc, z tą różnicą, że tranzycja jest wyświetlana, jako kwadrat z opisanym na zewnątrz czasem opóźnienia.

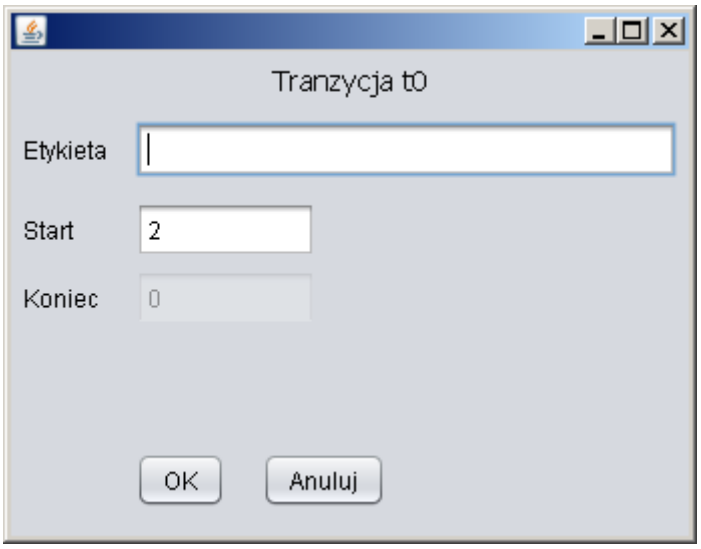
Dodawanie krawędzi do pola edycji.

Dodawanie krawędzi jest podobne do dodawania miejsc i tranzycji, z tym, że w polu edycji musimy najpierw kliknąć w obszar tranzycji lub miejsca, a następnie w obszar innego elementu niż kliknęliśmy za pierwszym razem. Pojawi nam się krawędź łącząca dwa różne elementy.

Wyświetlenie właściwości obiektu.

Aby wyświetlić właściwości elementu mamy dwa sposoby, albo wejść w tryb właściwości i kliknąć lewym klawiszem myszy w obszarze obiektu albo kliknąć prawym przyciskiem myszy na interesującym nas obiekcie i wyświetli się menu kontekstowe (Rys.22) i wejść w opcję właściwości.

Rys.22 Menu kontekstowe miejsca

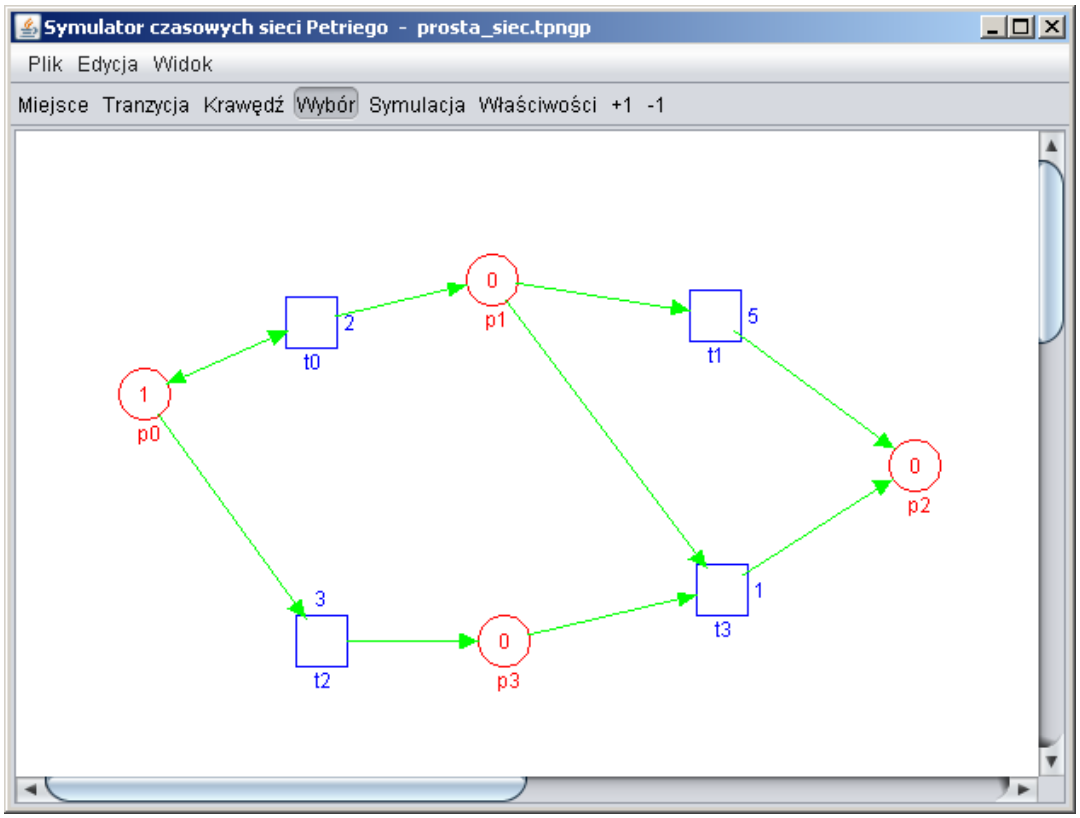


Rys.23 Okno z właściwościami tranzycji

Dodanie lub odjęcie znacznika w miejscu lub czasu tranzycji.

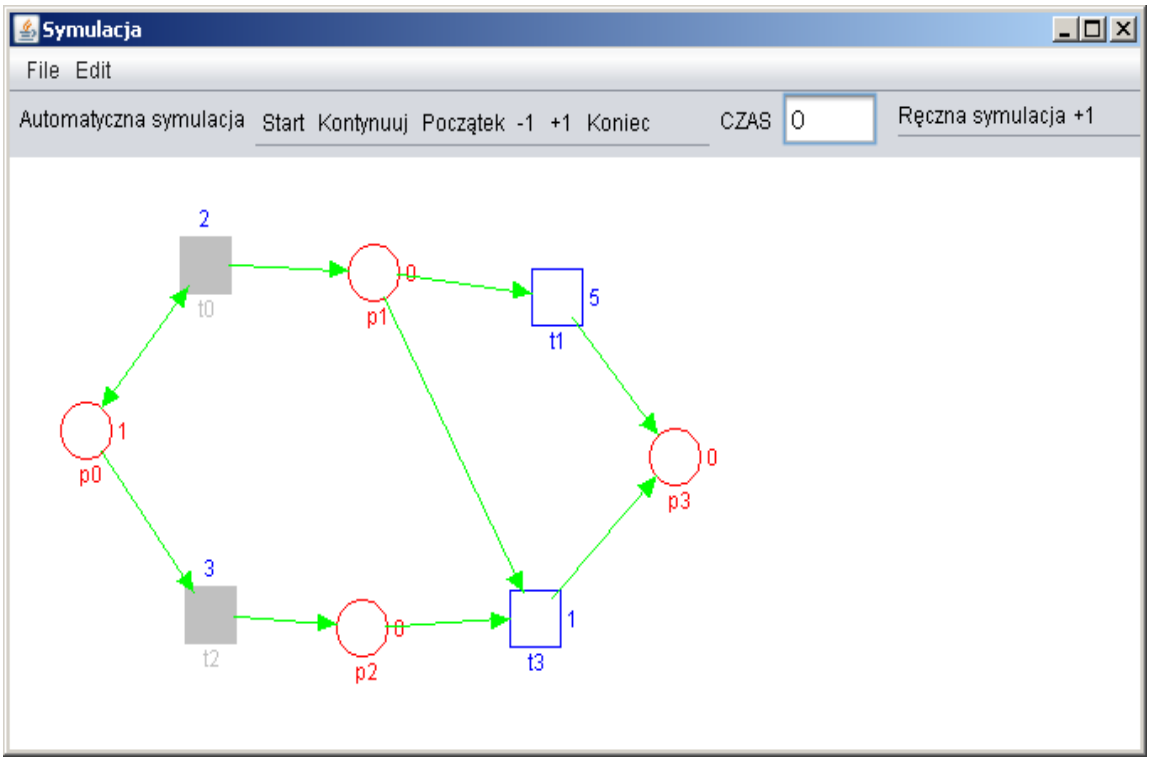
Aby wykonać powyższą operację należy tak jak w przypadku wyświetlenia właściwości wejść w odpowiedni tryb i kliknąć na obiekt, lub poprzez menu kontekstowe obiektu kliknąć w odpowiednią opcję.

Obiekty posiadają własne menu kontekstowe, za którego pomocą możemy dodać lub odjąć znacznik lub czas, usunąć dany obiekt lub wejść we właściwości.



Rys.24 Okno programu z narysowaną siecią

Po stworzeniu odpowiedniej sieci możemy przejść do okna symulacji.



Rys.25 Okno w stanie początkowym symulacji

Okno symulacji (Rys. 25) zawiera pasek menu, pasek narzędzi oraz pole symulacji ze stworzoną wcześniej strukturą sieci.

Menu składa się z dwóch podmenu *File* i *Edit*.

Podmenu *File* zawiera pole:

- *Wykres Gantta* – otwiera okno z wygenerowanym wykresem Gantta naszej symulacji.

Podmenu *Edit* zawiera pola ze zmianą czasu w ms w przerwach między uruchomieniami tranzycji.

Pasek narzędzi dzieli się na trzy części przeznaczone do symulacji automatycznej i symulacji ręcznej oraz do historii przeprowadzonej symulacji.

Menu do symulacji automatycznej zawiera pozycje takie jak:

- *Start* – uruchamia automatyczną symulacje,
- *Kontynuuj* – kontynuuje zatrzymaną symulację, po uruchomieniu symulacji pole zmienia nazwę na *Pause*, jeżeli pole nosi taką nazwę to symulację można zatrzymać, po zatrzymaniu pole z powrotem powróci do nazwy *Kontynuuj*.

Część przeznaczona do historii symulacji zawiera pozycje:

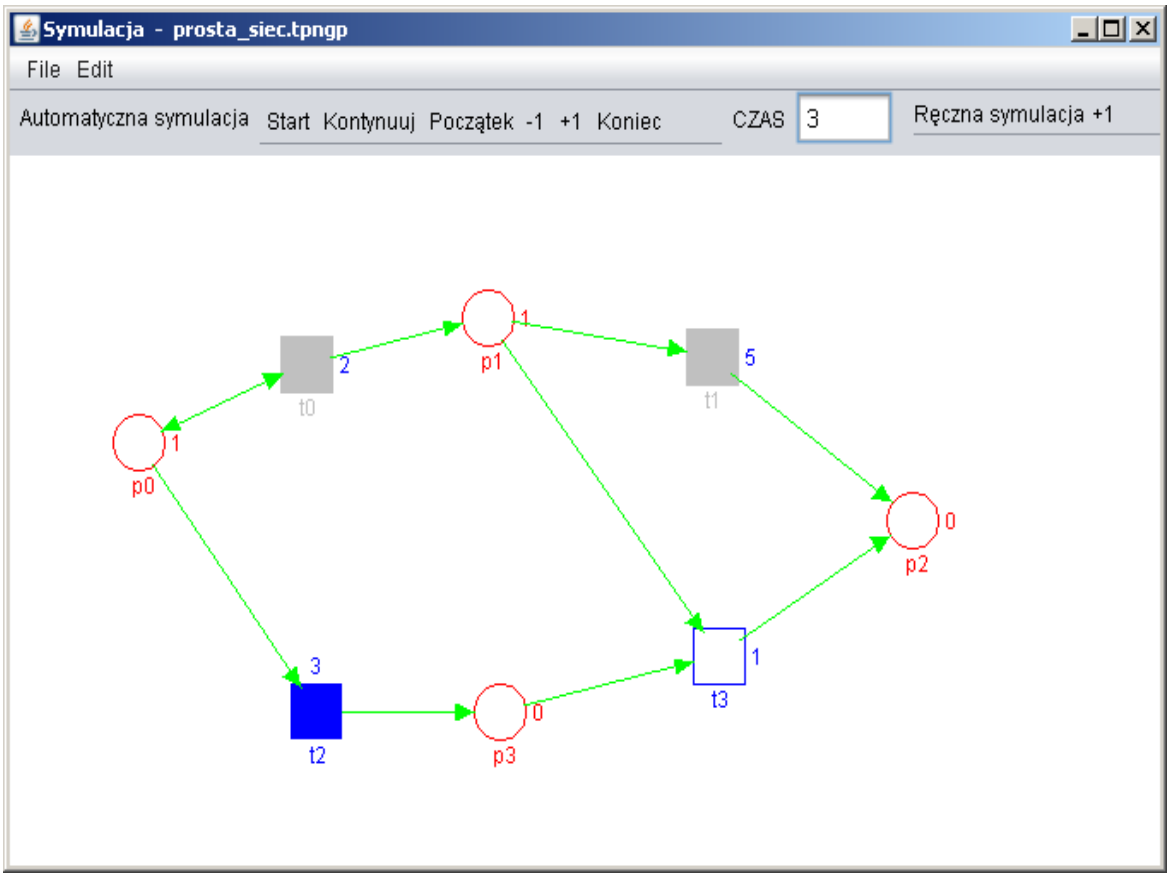
- *Początek* – przenosi stan sieci do stanu początkowego,
- - *1*- cofa stan o jeden krok do tyłu,
- *+1* – przenosi stan o jeden krok do przodu,
- *Koniec* – przenosi stan sieci do stanu końcowego symulacji.

Część przeznaczona do symulacji ręcznej zawiera tylko jedno pole:

- *+1* – zwiększa jednostkę czasu o 1.

Na pasku narzędzi znajduję się jeszcze pole czasu, które wyświetla aktualny czas symulacji (Rys 26).

Aktywne tranzycje mają kolor szary, natomiast tranzycje wykonywalne mają kolor niebieski.



Rys.26 Okno w trakcie przebiegu symulacji

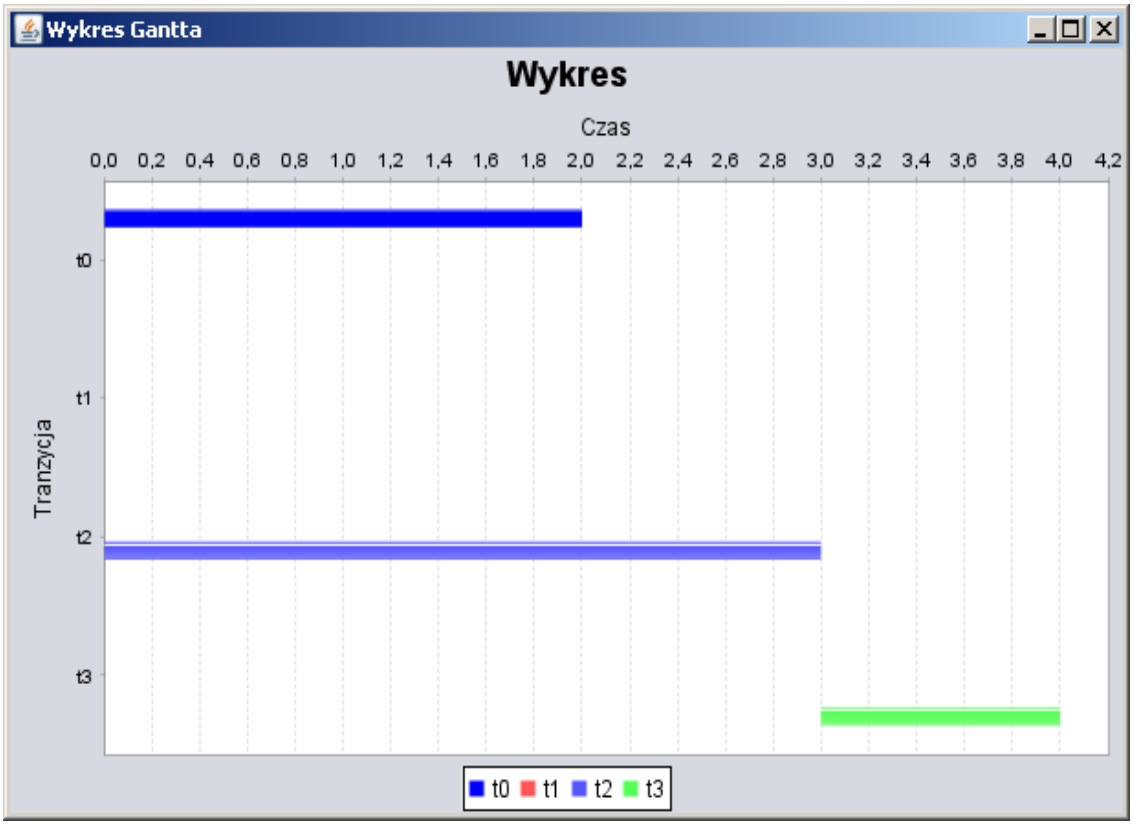
Wykres Gantta zostanie przedstawiony w poświęconym mu rozdziale 4.3.

4.2. Algorytm realizujący symulacje czasowej sieci Petriego

Schemat działania automatycznej symulacji czasowej sieci Petriego. Przedstawione są schematy działania głównej procedury oraz procedur wykonywanych podczas działania algorytmu.

4.3. Wykres Gantta

Wykres Gantta jest generowany przy użyciu biblioteki *JFreeChart* dla Java. Wykorzystano i odpowiednio zmodyfikowano przykład *GanttDemo2*, aby odpowiednio generował wykres Gantta dla wykonanej symulacji.



Rys.27 Okno z wygenerowanym wykresem Gantta

Wykres przedstawia czas palenia tylko wykonanych tranzycji, tranzycje, które były aktywne, ale nie były wykonane nie są umieszczone na wykresie.

W oknie wykresu mamy możliwość wyświetlenia menu kontekstowego z opcjami:

- *Właściwości* – wyświetla okno z właściwościami wykresu,
- *Kopiuj* – wykonuje PrintScreen zawartości okna z wykresem,
- *Save as* – pozwala wyeksportować wykres do pliku: PNG, SVG, PDF,
- *Drukuj* – drukuje wykres,
- *Powiększ* – zmniejsza zakres czasu wyświetlanego wykresu,
- *Pomniejsz* – zwiększa zakres czasy wyświetlanego wykresu,
- *Automatyczny zakres* – dobiera zakres czasu, aby wyświetlić cały wykres.

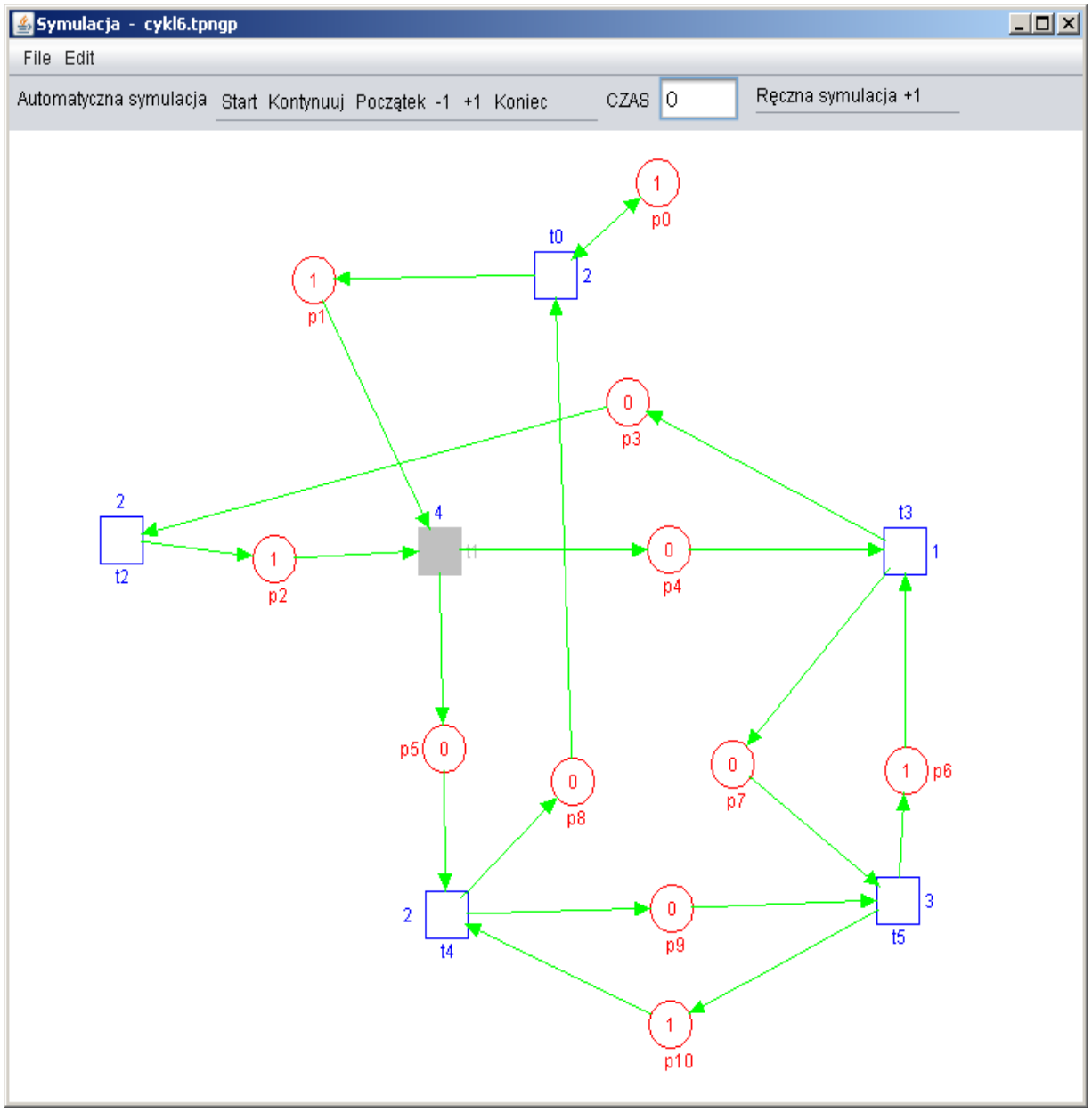
Zakres wykresu możemy również zmienić za pomocą myszki, przytrzymując i przeciągając myszkę w prawo powiększamy wykres, a w lewo pomniejszamy.

Okno właściwości wykresu posiada opcje:

- *Title* – pozwala zmienić tekst wykresu, czcionkę oraz kolor tekstu, a także wyłączyć wyświetlanie tytułu,
- *Wykres* – pozwala zmienić etykiety, czcionkę, kolor tekstu oraz podziałkę osi, a także wygląd wykresu, kolor tła, grubość konturu wykresu, pozycję wykresu.
- *Inne* – pozwala włączyć opcję wygładzania krawędzi.

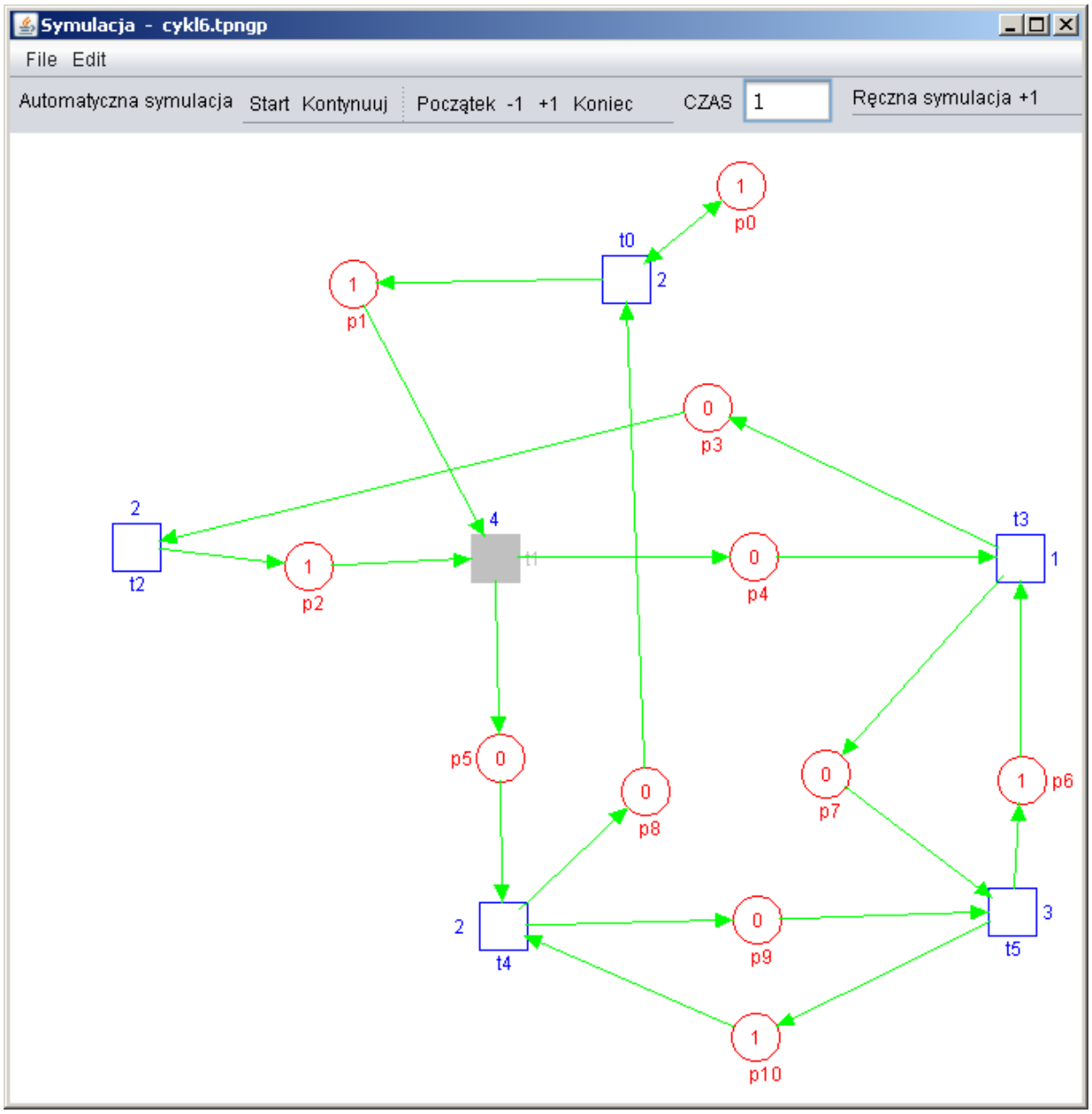
5. Testy

W celu przetestowania poprawności działania aplikacji, wykonana zostanie symulacja przykładowej sieci wraz wygenerowaniem wykresu Gantta. W celu dokładnego prześledzenia zostanie przeprowadzona symulacja ręczna.



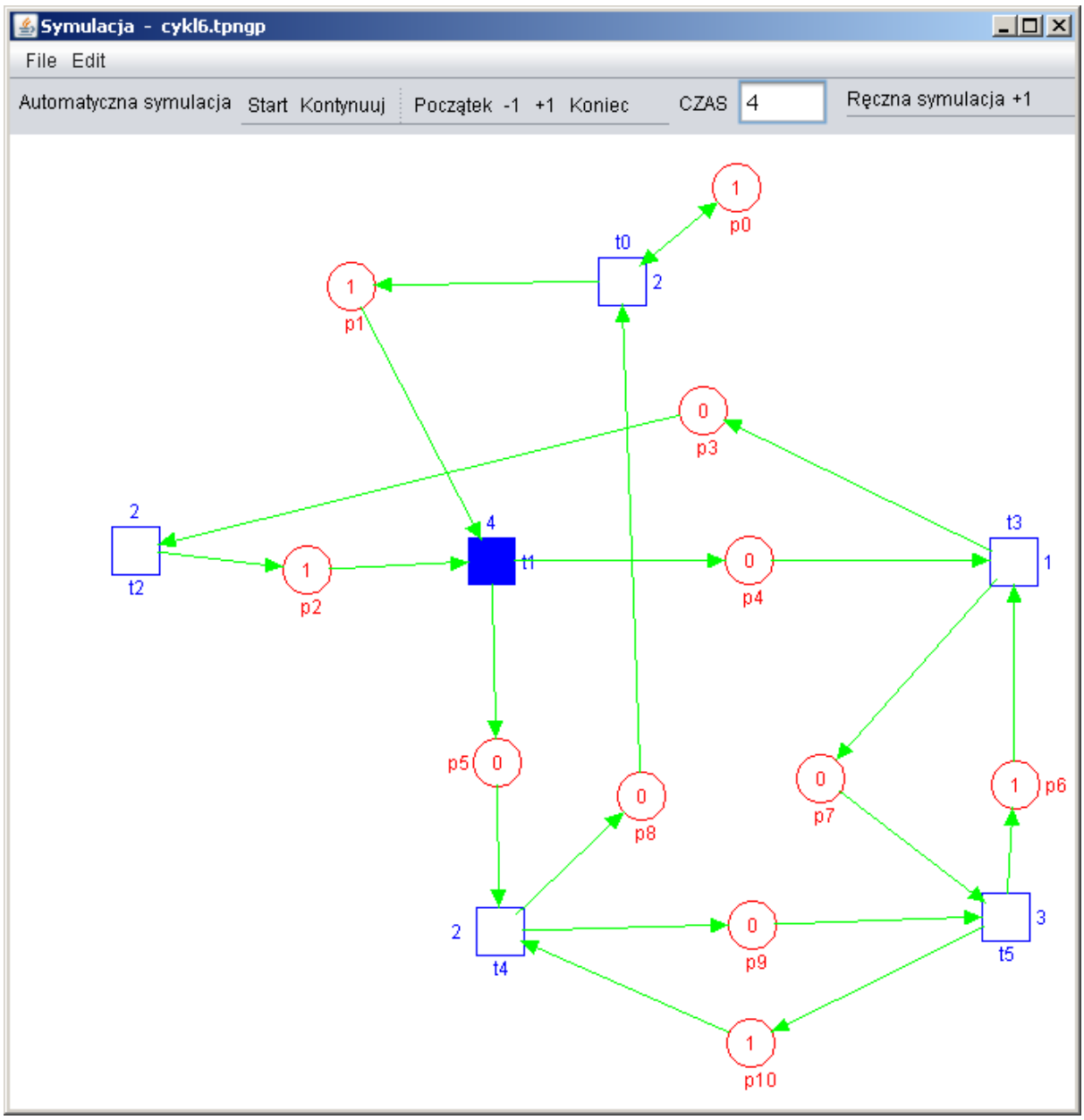
Rys. 29 Okno symulacji, stan początkowy sieci, aktywna jedna tranzycja.

Rysunek przedstawia sieć po przejściu do okna symulacji. Czas symulacji jest równy *0*. Aktywna jest tylko jedna tranzycja *t1*, z czasem opóźnienia *4*. Tranzycja *t1* jest aktywna, ponieważ wszystkie miejsca wejściowe posiadają znaczniki, natomiast żadna z pozostałych tranzycji nie posiada wystarczającej ilości znaczników na wejściach.



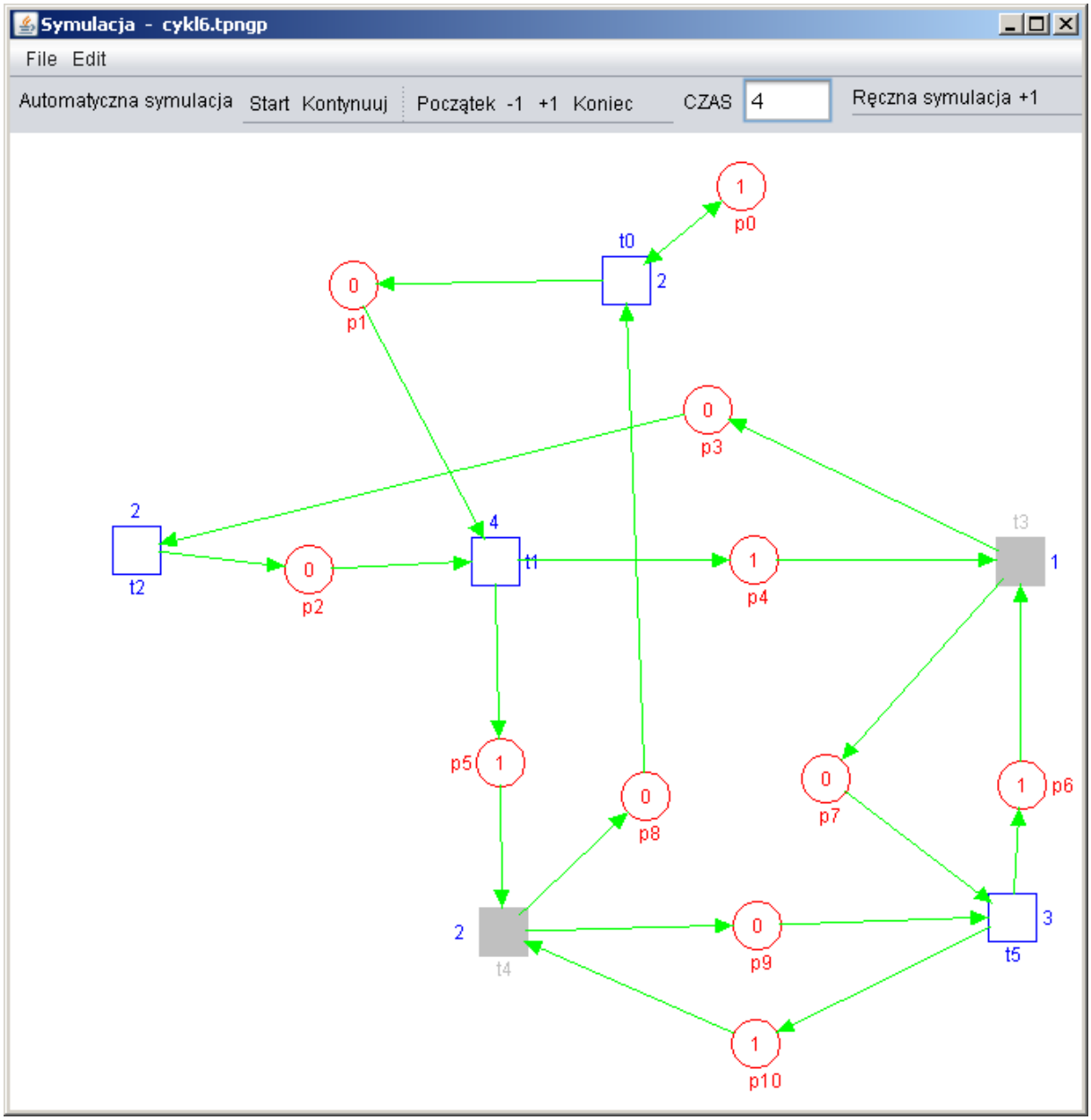
Rys.30 Okno symulacji po wykonaniu zmiany czasu, aktywna jedna tranzycja.

Po zwiększeniu czasu o *1* stan wszystkich elementów jest taki sam, nie zmieniło się nic. Czekamy na możliwość odpalenia tranzycji.



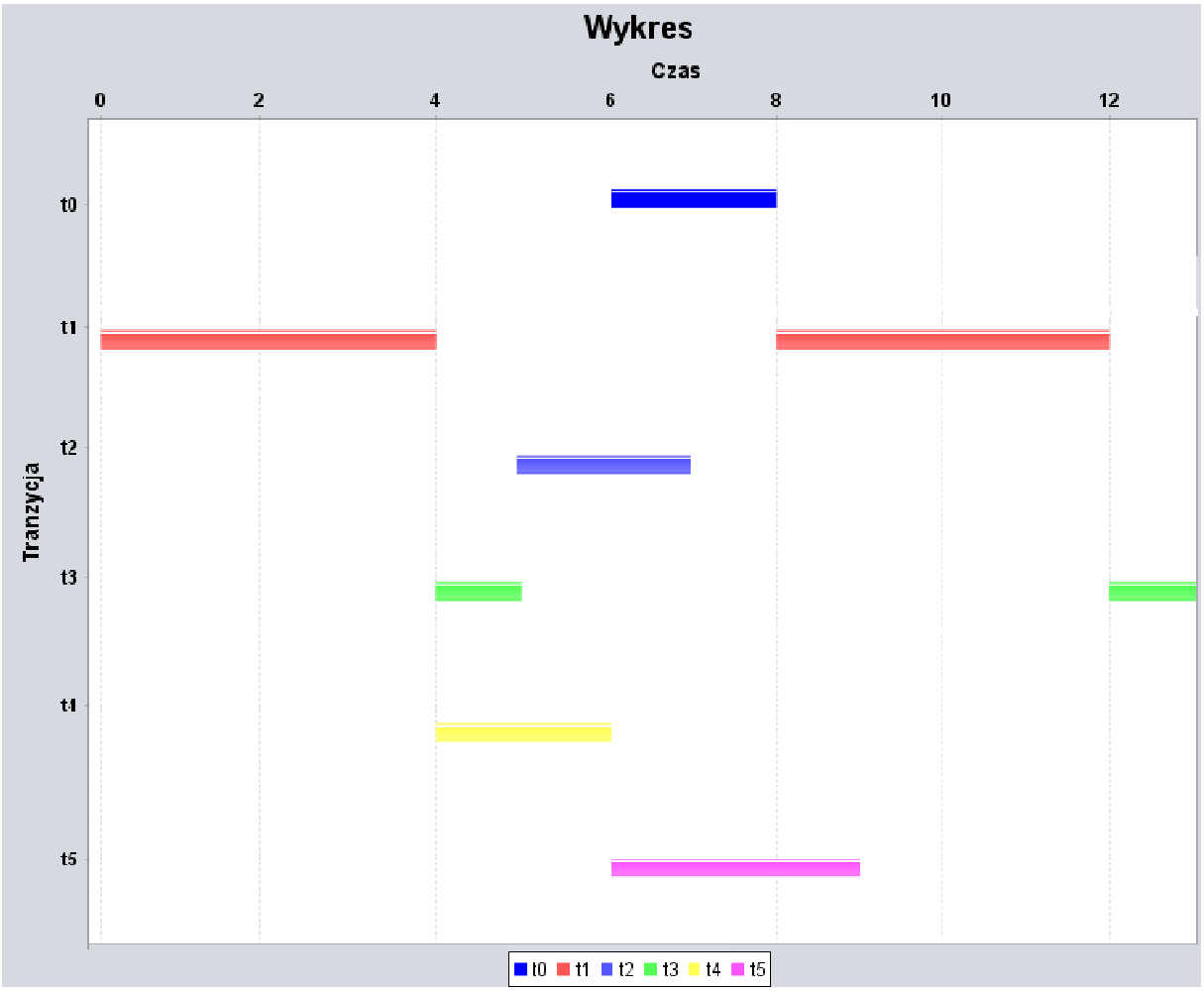
Rys.31 Okno symulacji z widoczną jedną wykonywalną tranzycją

Po przeskoczeniu do czasu równego *4*, tranzycja jest możliwa do odpalenia. Nie ma możliwości zwiększenia czasu, sieć została zablokowana, musi zostać odpalona tranzycja *t1*, odpalenie odblokuje sieć.



Rys.32 Okno symulacji, po wykonaniu tranzycji.

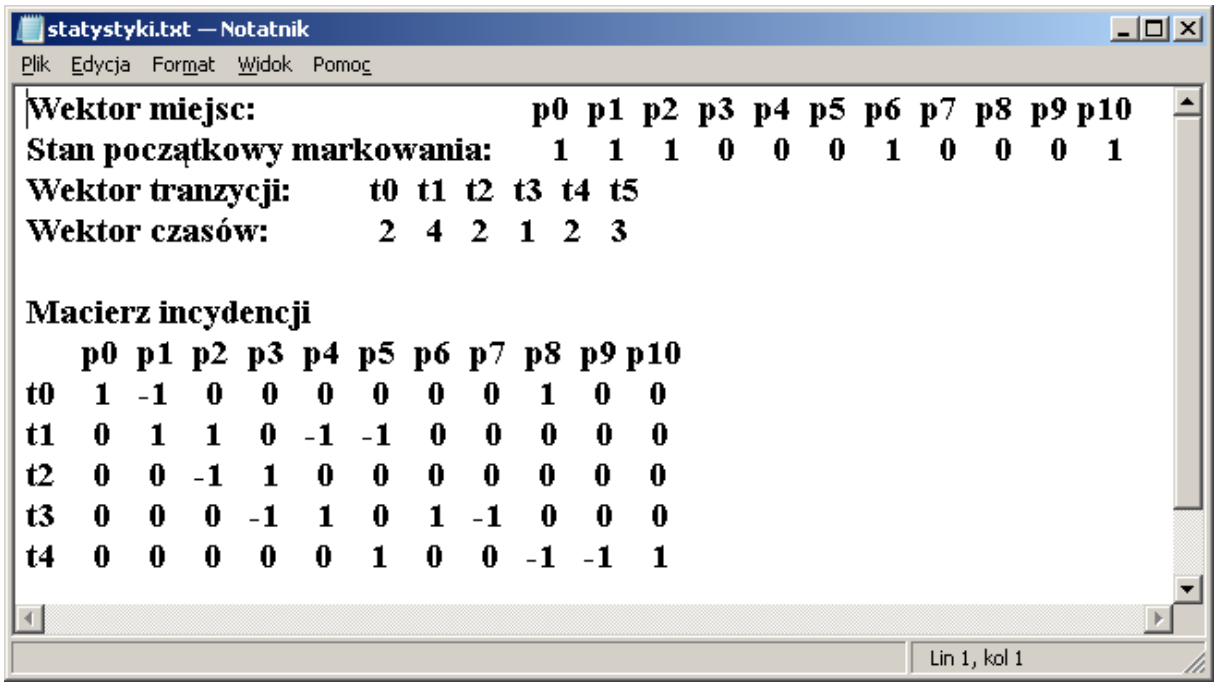
Po odpaleniu tranzycja *t1* stała się nieaktywna z miejsc wejściowych zostały pobrane znaczniki, a pojawiły się w miejscach wyjściowych. Tranzycje *t3* oraz *t4* stały się w tym momencie aktywne, ponieważ we wszystkich miejscach wejściowych znajdują się znaczniki. Ten krótki fragment symulacji obrazuje poprawne działanie procesu symulacji czasowej sieci Petriego.



Rys.33 Wykres Gantta

Na Rys. 33 przedstawiono wygenerowany wykres Gantta dla przykładowej sieci. Przykładowa sieć jest siecią cykliczną, czas trwania tranzycji *t1* od początku aktywności, aż do momentu odpalenia wynosi *4* jednostki czasu. Po wykonaniu tranzycji *t1* w *4* jednostce czasu symulacji, aktywne stały się tranzycje *t3* oraz *t4*. Tranzycja *t3* trwała *1* jednostkę czasu, a tranzycja *t4 2* jednostki czasu. Odpalenie tranzycji *t3* aktywowało tranzycje *t2*, której czas trwania wynosi *2* jednostki czasu. Odpalenie tranzycji t*4* aktywowało natomiast tranzycje *t0* oraz *t5*, których czas trwania *t0* wynosi *2* jednostki czasu, a *t5 3* jednostki czasu. Następnie wykonanie tranzycji *t0* aktywuje tranzycje *t1*, w tym momencie cykl został rozpoczęty na nowo.

Aplikacja wygenerowała poprawny plik z danymi symulowanej czasowej sieci Petriego. Przedstawia on wektor miejsc, stan początkowy markowania, wektor tranzycji, wektor czasów oraz macierz incydencji.



Rys. 34 Plik wygenerowany na podstawie edytowanej sieci

6. Podsumowanie

W stworzonej aplikacji udało się zaimplementować algorytm poprawnie realizujący działanie czasowych sieci Petriego. Aplikacja dostarcza graficzny interfejs użytkownika, który pozwala na szybkie i łatwe tworzenie oraz edycje sieci Petriego. Utworzoną sieć można zapisać do pliku, w formacie XML. Zapisane pliki mają rozszerzenie **tpngp**.

Aplikacja posiada możliwość przeprowadzenia symulacji działania czasowej sieci Petriego. Występują dwa tryby przeprowadzenia symulacji: tryb ręczny oraz automatyczny. W trybie ręcznym użytkownik sam zwiększa czas symulacji oraz uruchamia wykonywalne tranzycje. Natomiast w trybie automatycznym za cały proces symulacji odpowiada aplikacja, która sama wykonuje wszystkie operacje. W trybie automatycznym mamy możliwość ręcznego zatrzymania i wznowienia procesu symulacji. Symulacja w trybie automatycznym przedstawiona jest w postaci animacji, w której mamy możliwość zmiany czasu wykonywania.

Poza symulacją, aplikacja posiada w odróżnieniu od przeanalizowanych aplikacji przedstawienia przebiegu symulacji w postaci wykresu Gantta. Za wygenerowanie wykresu odpowiada zewnętrzna biblioteka JFreeChart. Wykres przedstawia czas palenia tranzycji w trakcie symulacji. Aplikacja generuje również plik z właściwościami stworzonej sieci. Plik zawiera takie elementy jak stan markowanie początkowego, czasy poszczególnych tranzycji oraz macierz incydencji.

Graficzny interfejs użytkownika jest przygotowany do realizacji przedziałowych sieci Petriego, także aplikacje można w łatwy sposób rozszerzyć o realizacje przedziałowych sieci Petriego. Ponieważ aplikacja nie posiada możliwości analizy sieci Petriego może być w przyszłości rozszerzona o odpowiednie narzędzia. Istnieje ponadto jeszcze wiele możliwości rozszerzenia aplikacji np. dotyczące innych klas sieci Petriego.

Aplikacja właściwie i poprawnie ilustruje działanie czasowych Sieci Petriego, co zostało zaprezentowane w rozdziale Testy. Udanie został zrealizowany również główny cel, czyli wykonanie algorytmu realizującego czasową sieć Petriego.

Testy udowodniły poprawne działanie programu oraz algorytmu, animacja symulacji obrazuje rzeczywiste działanie algorytmu. Prezentacja wyników działa zgodnie z oczekiwaniami.

Bibliografia

1. Szpyrka Marcin, Sieci Petriego w modelowaniu i analizie systemów współbieżnych – Wydawnictwo Naukowo-Techniczne W-wa 2008
2. Kulikowski Juliusz Lech., Zarys teorii grafów: zastosowania w technice. PWN, Warszawa 1986
3. http://sirius.cs.put.poznan.pl/~inf89721/MiAPB/Nowe/4%20-%20Wprowadzenie%20do%20Sieci%20Petriego.pdf
4. Ulasiewicz Jędrzej, http://jedrzej.ulasiewicz.staff.iiar.pwr.wroc.pl/ProgramowanieWspolbiezne/wyklad/Sieci-Petriego15.pdf
5. http://projects.laas.fr/tina//
6. http://pipe2.sourceforge.net/index.html
8. Witas Paweł, https://www.academia.edu/3994494/Symulator_sieci_Petriego_ze_znacznikami_indywidualnymi

Lista rysunków

Załączniki

- płyta z aplikacją, projektem i pracą w wersji PDF