Modele kolejkowe w IT

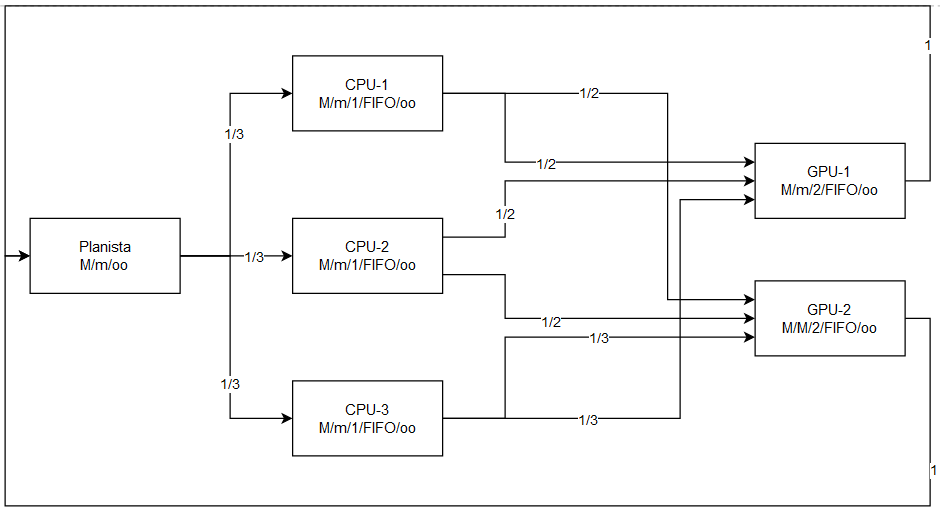
1. Wstęp

Celem jest zamodelowanie zamkniętej sieci kolejkowej BCMP2 dla systemu złożonego z planisty, modułów CPU oraz GPU w którym krąży klas zgłoszeń. Projekt zakłada utworzenie programu pozwalającego na zbadanie parametrów sieci określenie przez użytkownika.

Założeniami projektu jest wykorzystanie metody SUM oraz przyjęty brak możliwości zmiany przynależności do klasy przez zgłoszenie. Użytkownik musi znać parametry sieci, którą chce zamodelować takie jak: liczba zgłoszeń każdej klasy w systemie, ilość oraz typy systemów w sieci oraz prawdopodobieństwa przejść dla każdej klasy.

1. Opis zagadnienia

Badana w projekcie sieć kolejkowa zakładała się z Planisty będącego systemem typu , trzech CPU będących systemami oraz dwóch GPU będących typu gdzie zostało wybrane jako . W systemie krążyły 4 klasy zgłoszeń z różnym przetwarzaniem przez systemy.



Rysunek . Schemat sieci dla klasy zgłoszeń 1.

W celu znalezienia parametru lambda opisującego każdy system wykorzystana iteracyjną metodę SUM opisaną w instrukcji do ćwiczeń. Następnie parametry dla każdego systemu były wyznaczane zgodnie z wzorami im odpowiadającymi opisanymi w [1], [2].

Opisy klas zgłoszeń:

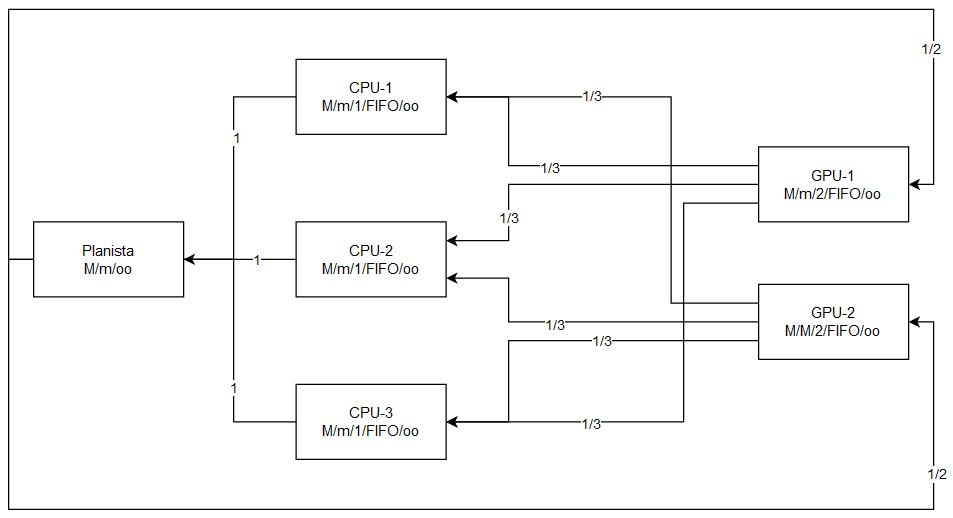
* Klasa 1: zgłoszenie musi zostać przetworzone najpierw przez CPU a następnie przez GPU.
* Klasa 2: zgłoszenie musi zostać przetworzone w kolejności GPU -> CPU.
* Klasa 3: zgłoszenie musi zostać przetworzone tylko przez CPU.
* Klasa 4: zgłoszenie musi zostać przetworzone tylko przez GPU.

Program udostępnia możliwość zdefiniowania różnej ilości klas.

Parametry dla systemów obliczane są z wykorzystaniem wzorów [1]:

Prawdopodobieństwa k zgłoszeń w systemie:

Oraz średni czas w systemie:



Rysunek . Schemat systemu dla zgłoszeń klasy 2

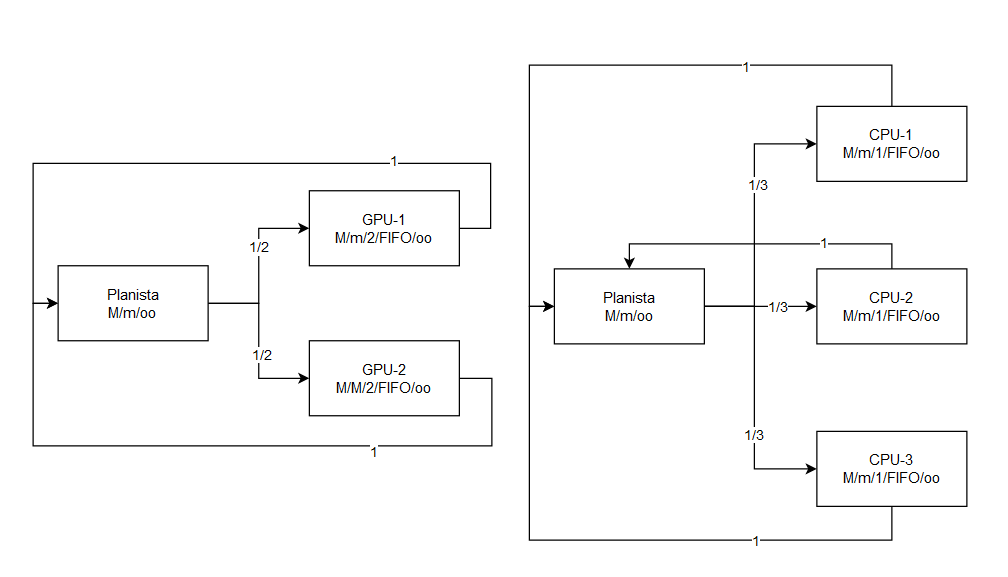
Parametry dla systemów obliczane są z wykorzystaniem wzorów [2]:

Prawdopodobieństwo k zgłoszeń w systemie:

Średnia długość kolejki:

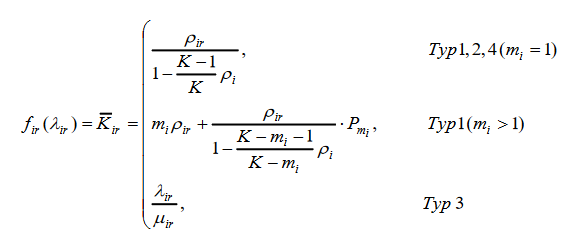
Średnia czas w kolejce:

Oraz średni czas odpowiedzi:



Rysunek . Schemat sieci dla klasy 3(prawo) oraz 4(lewo)

Gdzie wyznaczane jest z wykorzystaniem funkcji:



Macierze prawdopodobieństw przejść zostały stworzone na podstawie Rysunków 1,2 oraz 3.

K1:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 1 | 1 |
| 1/3 | 0 | 0 | 0 | 0 | 0 |
| 1/3 | 0 | 0 | 0 | 0 | 0 |
| 1/3 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1/2 | 1/2 | 1/2 | 0 | 0 |
| 0 | 1/2 | 1/2 | 1/2 | 0 | 0 |

K2:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1/3 | 1/3 |
| 0 | 0 | 0 | 0 | 1/3 | 1/3 |
| 0 | 0 | 0 | 0 | 1/3 | 1/3 |
| 1/2 | 0 | 0 | 0 | 0 | 0 |
| 1/2 | 0 | 0 | 0 | 0 | 0 |

K3:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 1 | 1 | 1 | 0 | 0 |
| 1/3 | 0 | 0 | 0 | 0 | 0 |
| 1/3 | 0 | 0 | 0 | 0 | 0 |
| 1/3 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |

K4:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1/2 | 0 | 0 | 0 | 0 | 0 |
| 1/2 | 0 | 0 | 0 | 0 | 0 |

Macierz stosunku wizyt jest obliczana z wykorzystaniem funkcji opartej na wzorze:

Obraz zawierający tekst, zegarek

Opis wygenerowany automatycznie

Przy założeniu, że .

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 1/3 | 1/3 | 1/3 | 1/2 | 1/2 |
| 1 | 1/3 | 1/3 | 1/3 | 1/2 | 1/2 |
| 1 | 1/3 | 1/3 | 1/3 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1/2 | 1/2 |

1. **Dokumentacja**

Wymagane środowisko: Python 3.6,  
Wymagane biblioteki: numpy, math, matplotlib, sympy

Aplikacja zawiera w sobie 2 programy, jeden pozwalający na obliczenia dla systemu zamkniętego w przypadku, kiedy posiadamy tylko jedną klasę zgłoszeń oraz drugi oferujący możliwość zbadania sieci z wieloma zgłoszeniami. Oba programy są odpowiednio podpisane.

Aby uruchomić program użytkownik musi wypełnić macierze prawdopodobieństwa przejść, których może być dowolna ilość. Następnie należy uruchomić funkcję get\_e() z parametrami:

Program następnie wyświetli otrzymaną macierz stosunku wizyt.

Następnie użytkownik musi wypełnić parametry inicjalizacyjne programu:

K – wektor zgłoszeń k-tej klasy w sieci  
class\_num – ilość klas zgłoszeń w sieci  
sys\_num – ilość systemów w sieci  
mu – macierz   
mi - , ilość serwerów w i-tym systemie  
typ – typ systemu, jeśli i-ty system należy to typu 3 należy wpisać wartość 2, w przeciwnym razie należy wpisać wartość 1.

Następnie należy uruchamiać po kolei fragmenty kodu. Pozwoli nam to uzyskać parametry takie jak:

* Stabilność sieci – jeśli któryś z systemów wykazuje niestabilność musimy zmienić parametry, ponieważ pozostałe parametry będą niemiarodajne.
* Średnia liczba zgłoszeń w poszczególnych systemach zostanie wyświetlona na wykresie z podziałem na system oraz klasę zgłoszeń.
* Macierz
* Średnia długość kolejki dla każdego systemu
* Wykresy prawdopodobieństw dla każdego systemu
* Macierz średnich czasów odpowiedzi

1. **Eksperymenty**

Dla opisanego w zagadnieniach modelu sieci przyjęto dodatkowo założenia:

* Liczba zgłoszeń w systemie: K1=2, K2= 3, K3=2, K4=5.
* Macierz :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | System 1 | System 2 | System 3 | System 4 | System 5 | System 6 |
| Klasa z. 1 | 120 | 80 | 80 | 80 | 50 | 50 |
| Klasa z. 2 | 120 | 70 | 70 | 70 | 60 | 60 |
| Klasa z. 3 | 120 | 50 | 50 | 50 | - | - |
| Klasa z. 4 | 120 | - | - | - | 30 | 30 |

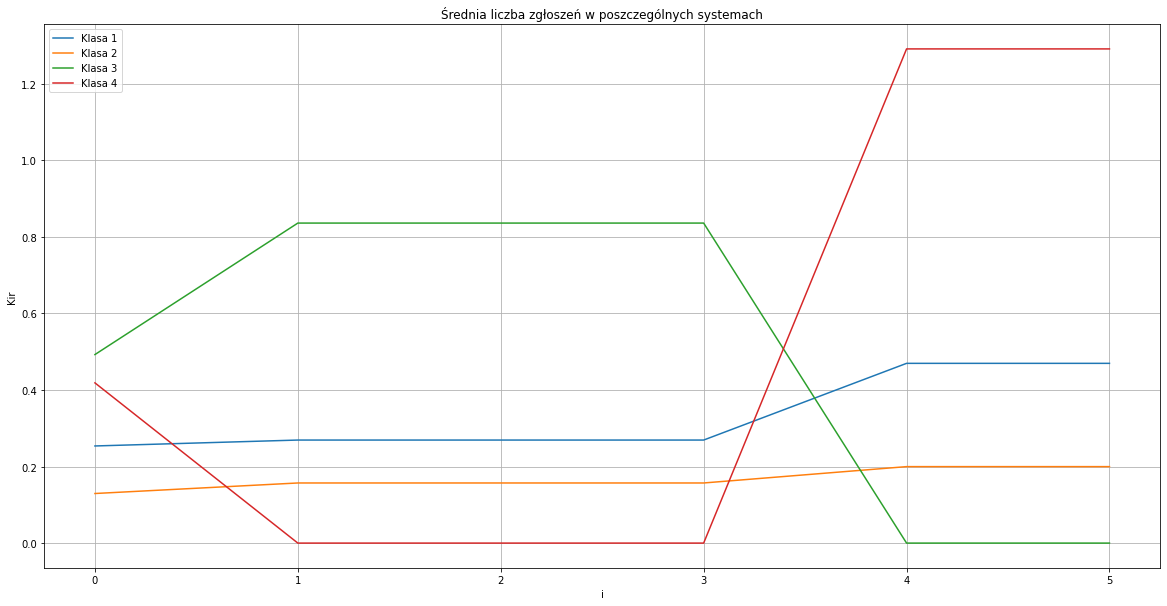
Następnie po uruchomieniu programu, otrzymaliśmy wyniki:

Stabilności systemów:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Średnią liczbę zgłoszeń każdej klasy w poszczególnych systemach:



Wartości :

Obraz zawierający tekst

Opis wygenerowany automatycznie

Średnie długości kolejki:

Obraz zawierający tekst

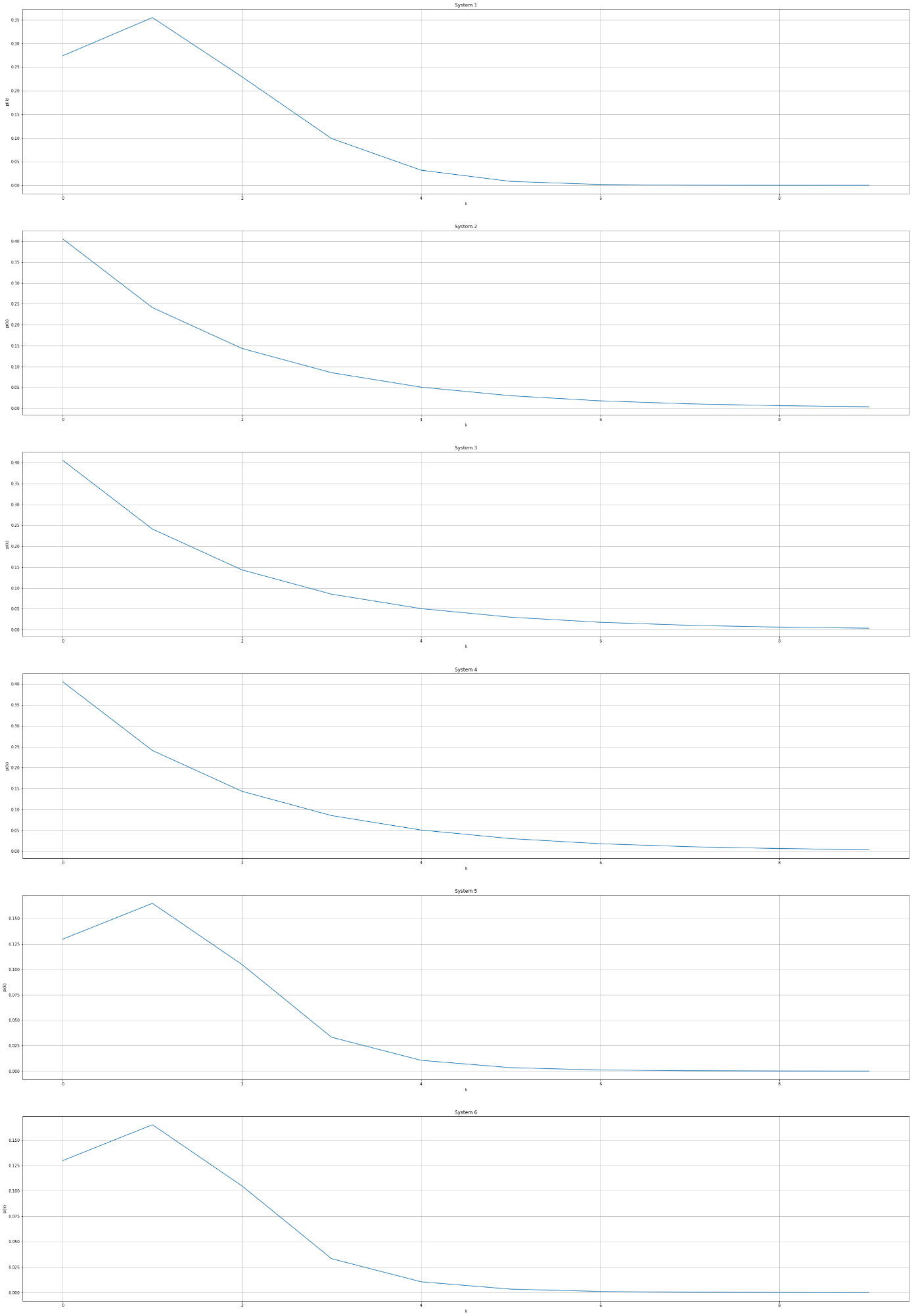
Opis wygenerowany automatycznie

Średnie czasy odpowiedzi:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Prawdopodobieństwa :



1. **Podsumowanie**

Najcięższym punktem projektu jest dobranie parametrów tak aby system był stabilny. Podczas obliczeń mogą wystąpić 2 problemy, pierwszy w którym brak zbieżności wyników może uniemożliwić obliczanie parametrów systemów w sieci oraz drugi – system nawet po obliczeniu lambdy jest niestabilny. Jeśli obliczenia przeciągają się, oznacz to, że najprawdopodobniej rozwiązanie dla badanego systemu nie istnieje, co czyni go niestabilnym.

1. Literatura:

[1]. B. FILIPOWICZ and J. KWIECIEŃ, Queueing systems and networks. Models and applications, BULLETIN OF THE POLISH ACADEMY OF SCIENCES TECHNICAL SCIENCES Vol. 56, No. 4, 2008

[2]. Bolch, S. Greiner, H. de Meer, and K.S. Trivedi, Queueing Networks and Markov Chains. Modelling and Performance Evaluation with Computer Science Applications, John Wiley&Sons, Inc., London, 1998