SYMULACJA CYFROWA
RAPORT KOŃCOWY

Metoda planowania zdarzeń

Robert Jarecki

SK
nr albumu: 126192

1. Treść zadania

W sieci bezprzewodowej stacje nadawcze konkurują o dostęp do łącza. W losowych odstępach czasu $\mathbf{CGP}_k$ *k*-ta stacja nadawcza generuje pakiety gotowe do wysłania. Po uzyskaniu dostępu do łącza zgodnie z algorytmem **A**, *k*-ty terminal podejmuje próbę transmisji najstarszego pakietu ze swojego bufora. Czas transmisji wiadomości z *k*-tej stacji nadawczej do *k*-tej stacji odbiorczej wynosi $\mathbf{CTP}_k$. Jeśli transmisja pakietu zakończyła się sukcesem, stacja odbiorcza przesyła potwierdzenie ACK (ang. *Acknowledgment*) poprawnego odebrania wiadomości. Czas transmisji ACK wynosi **CTIZ**. Jeśli transmisja pakietu nie powiodła się, stacja odbiorcza nie przesyła ACK. Odbiór pakietu uznajemy za niepoprawny, jeśli w kanale transmisyjnym wystąpiła kolizja lub błąd. Przez kolizję rozumiemy nałożenie się jakiejkolwiek części jednego pakietu na inny pakiet (pochodzący z innego nadajnika). Dodatkowo każda transmisja pakietu może zakończyć się błędem TER. Brak wiadomości ACK po czasie ($\mathbf{CTP}_k$+ **CTIZ**) od wysłania pakietu jest dla stacji nadawczej sygnałem o konieczności retransmisji pakietu. Każdy pakiet może być retransmitowany maksymalnie **LR** razy. Dostęp do łącza w przypadku retransmisji opiera się na tych samych zasadach co transmisja pierwotna. Jeśli mimo **LR**-krotnej próby retransmisji pakietu nie udało się poprawnie odebrać, wówczas stacja nadawcza odrzuca pakiet i – jeśli jej bufor nie jest pusty – przystępuje do próby transmisji kolejnego pakietu.

Opracuj symulator sieci bezprzewodowej zgodnie z metodą **M**.

Za pomocą symulacji wyznacz:

- Wartość parametru **L**, która zapewni średnią pakietową stopę błędów (uśrednioną po ***K*** odbiornikach) nie większą niż 0.1, a następnie:

  - pakietową stopę błędów w każdym z odbiorników mierzoną jako iloraz liczby pakietów straconych do liczby przesłanych pakietów,

  - średnią liczbę retransmisji pakietów,

  - przepływność systemu mierzoną liczbą poprawnie odebranych pakietów w jednostce czasu, o średnie opóźnienie pakietu, tzn. czas jaki upływa między pojawieniem się pakietu w buforze, a jego poprawnym odebraniem,

  - średni czas oczekiwania, tzn. czas między pojawieniem się pakietu w buforze, a jego opuszczeniem

  - sporządź wykres zależności średniej liczby retransmisji pakietów od parametru P

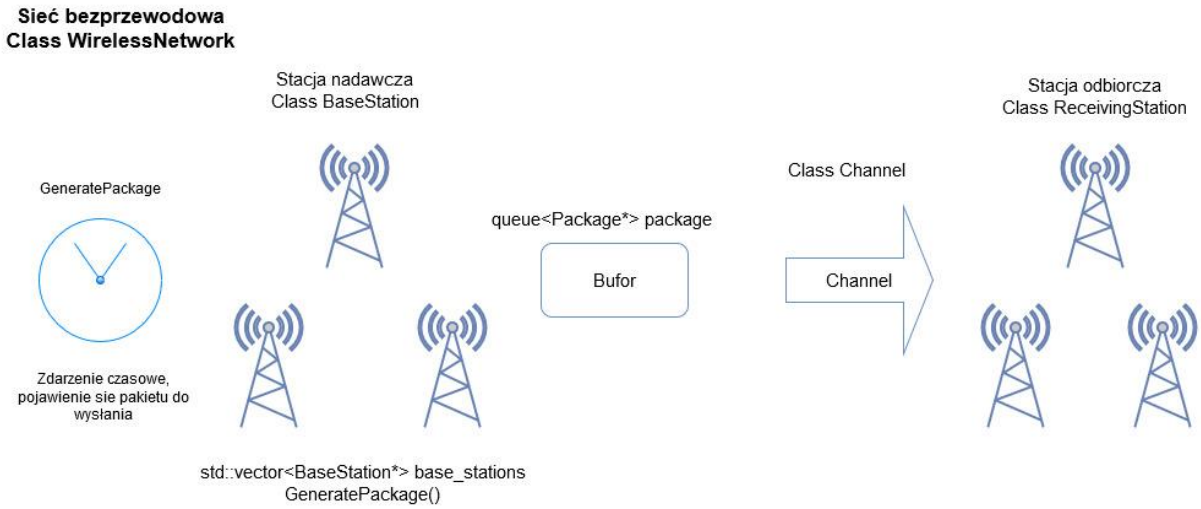
Sporządź wykres zależności przepływności systemu oraz średniej i maksymalnej pakietowej stopy błędów w zależności od wartości **L**.

Protokół CSMA (ang. Carrier Sense Multiple Access) z wymuszaniem transmisji z prawdopodobieństwem p (ang. p-persistent) – w protokole tym czas jest podzielony jest na szczeliny o długości CSC. Po wygenerowaniu nowego pakietu, stacja nadawcza sprawdza zajętość kanału transmisyjnego. Jeśli kanał jest zajęty, to dalsze odpytywanie kanału odbywa się w odstępach co 0.5 ms. Gdy stacja wykryje, że kanał jest wolny, rozpoczyna transmisję w najbliższej szczelinie z prawdopodobieństwem PT. Z prawdopodobieństwem (1-PT) stacja wstrzymuje się z transmisją do następnej szczeliny, w której ponownie sprawdza status kanału. Jeśli następna szczelina okaże się również wolna, terminal rozpoczyna transmisję z prawdopodobieństwem PT lub wstrzymuje się z prawdopodobieństwem (1-PT). Ta procedura jest powtarzana tak długo, aż pakiet zostanie wysłany lub kanał stanie się zajęty. W tym ostatnim przypadku terminal nasłuchuje kanał w odstępach co 1 ms i gdy wykryje, że jest wolny, rozpoczyna opisaną wyżej procedurę od nowa.
W przypadku retransmisji, stacja nadawcza sprawdza stan kanału po losowym czasie CRP równym R*CTPk, gdzie R jest losową liczbą z przedziału od <0, (2r -1)>, a r jest numerem aktualnej retransmisji (przy każdej retransmisji czas ten jest losowany ponownie). Wówczas uruchamiana jest taka sama procedura jak w przypadku transmisji pierwotnej.

a) PT = 0.2
b) PT = 0.4
c) PT = 0.6
d) PT = 0.8

Metoda symulacji: metoda planowania zdarzeń
Protokół: A5b

a) schemat modelu symulacyjnego



Rys. 1. Schemat modelu symulacyjnego

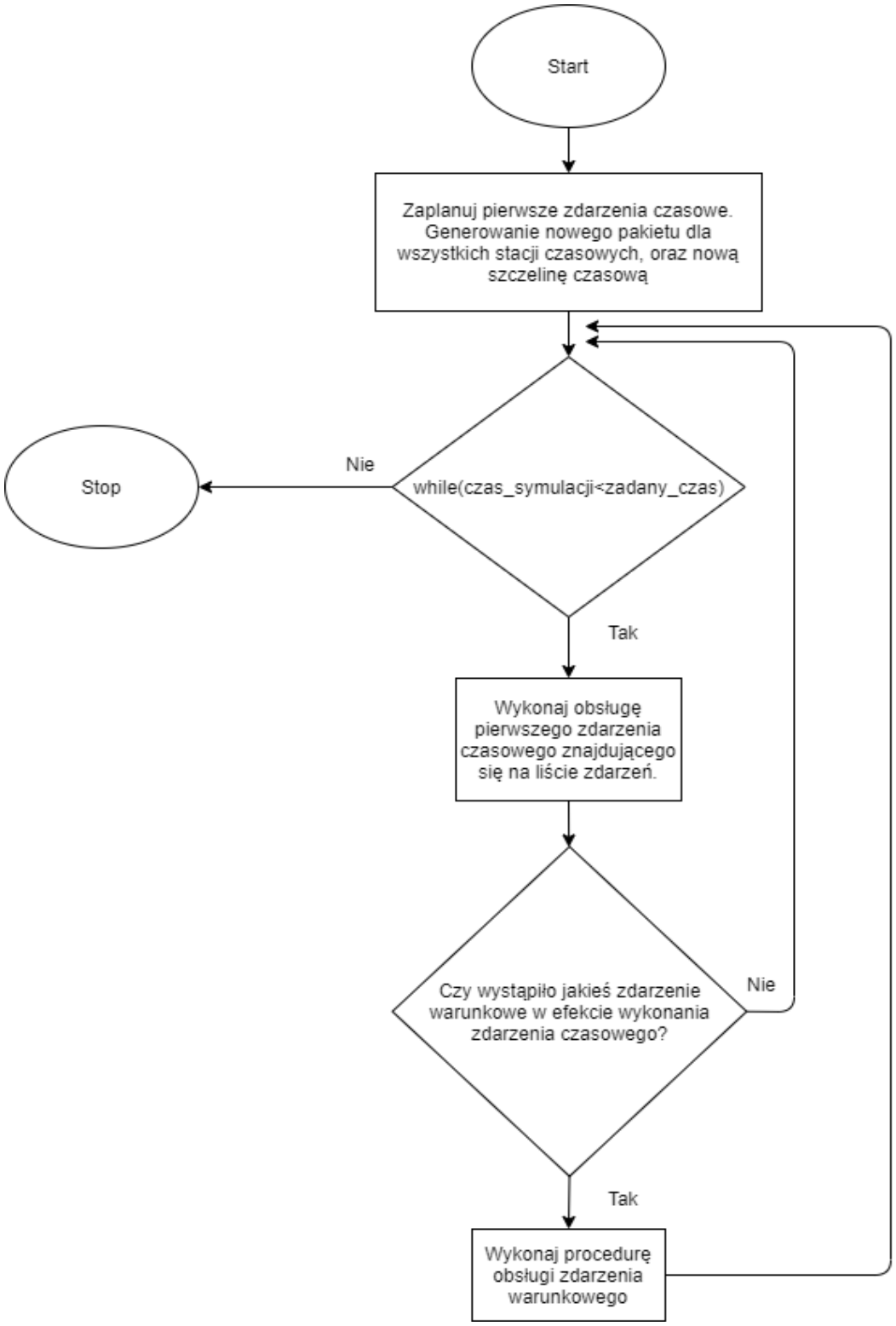
b) opis klas wchodzących w skład systemu i ich atrybutów

Obiekt	Nazwa klasy implementującej obiekt	Opis	Atrybuty
Sieć bezprzewodowa	WirelessNetwork	Klasa gromadząca wszystkie pozostałe elementy systemu.	- double system_time_ zmienna opisująca ogólny czas systemowy - const int kNumberOfBaseStations stała opisująca ilość stacji nadawczych - vector<BaseStation*> base_stations_ wektor przechowujący wskaźniki do każdej stacji nadawczej - Channel * channel_ zmienna opisująca wskaźnik na kanał - ReceivingStation* receiving_station_ zmienna opisująca wskaźnik na stacje odbiorczą - const int kAmountOfRetransmission stała opisująca ilość dozwolonych retransmisji
Stacja nadawcza	BaseStation	Klasa ma za zadanie kontrolowanie przepływu gotowych pakietów do wysłania.	- int id_base_station_ zmienna opisująca numer stacji nadawczej -queue<Package*> package_ kolejka przechowująca gotowe pakiety do wysłania - bool start_transmiting_ zmienna określająca rozpoczęcie transmisji przez stację nadawczą
Stacja odbiorcza	ReceivingStation	Klasa ma za zadanie potwierdzanie	- int id_receiving_station_ zmienna opisująca numer stacji odbiorczej

		odebranych pakietów. Odpowiednio przesyła ACK lub nie.	- bool confirmation_ACK_ zmienna określająca czy wystawiono potwierdzenie ACK
Kanał	Channel	Klasa ma za zadanie wysyłać pakiety do stacji odbiorczej.	- bool channel_availability_ zmienna określa czy dany kanał jest wolny - bool collision_checking_ zmienna określa czy doszło do kolizji - double transmission_time_ zmienna opisująca czas transmisji - vector<Package*> current_package_ wektor przechowujący wskaźnik na aktualnie transmitowany pakiet
Pakiet	Package	Klasa określa pakiet danych, który jest później wysyłany przez kanał.	- int id_package_ zmienna, która określa numer pakietu - int number_current_retransmission_ zmienna określa numer aktualnej retransmisji - int id_source_transmiter_ zmienna opisująca id stacji nadawczej z której wysłany został pakiet - double average_waiting_time_ zmienna przechowująca średni czas oczekiwania pakietu

2. Opis przydzielonej metody symulacyjnej

a) schemat blokowy pętli głównej



Rys. 2. Schemat blokowy pętli głównej

b) lista zdarzeń czasowych i warunkowych

Zdarzenia czasowe:

Zdarzenie	Opis	Algorytm
Wygenerowanie pakietu	K-ta stacja generuje pakiety gotowe do wysłania w losowych odstępach czasu (CGPk).	1. Wygenerowany pakiet zostaje umieszczony na końcu kolejki 2. Zaplanuj kolejne zdarzenie czasowe wygenerowania pakietu
Sprawdzenie wiadomości ACK po określonym czasie	Po upływie czasu sprawdzane jest czy wiadomość ACK poprawnie dotarła do stacji nadawczej	1. Sprawdzanie dostarczenia wiadomości ACK 2. Brak wiadomości ACK powoduje zapisanie id stacji nadawczej do zdarzenia warunkowego wystąpienia błędu TER
Sprawdzanie zajętości kanału	Po wygenerowaniu pakietu sprawdzamy zajętość kanału.	1.Sprawdzanie zajętości kanału 2. Jeśli kanał jest wolny zaplanuj próbę wysłania pakietu w nowej szczelinie czasowej 3. Jeśli kanał jest zajęty zaplanuj sprawdzenie kanału za 0.5 ms
Nowa szczelina czasowa	Wygenerowanie nowej szczeliny czasowej	1. Wygeneruj nową szczelinę czasową 2. Zaplanuj nowe zdarzenie czasowe wygenerowania nowej szczeliny czasowej
Koniec transmisji pakietu	Następuje po upływie czasu przewidzianego na dotarcie pakietu do stacji odbiorczej.	1. Kanał stał się wolny 2. Jeżeli nie ma kolizji pakiet zostaje dodany do stacji odbiorczej 3. Zaplanuj zdarzenie czasowe zakończenia przesyłania wiadomości ACK 4. Jeżeli doszło do kolizji to zwróć pakiety do retransmisji
Zakończenie przesłania wiadomości ACK	Zakończenie przesłania wiadomości ACK przez kanał	1. Ustaw stan kanału na wolny 2. Zapisz wiadomość ACK do stacji nadawczej

Zdarzenia warunkowe:

Zdarzenie	Opis	Algorytm
Transmisja pakietu	Rozpoczęcie transmisji pakietu.	1. Próba uzyskania dostępu do łącza 2. Po uzyskaniu dostępu do łącza wyślij pakiet z określonym prawdopodobieństwem 3. Jeśli pakiet został wysłany to ustaw kanał na zajęty 4. Zaplanuj zdarzenie czasowe końca przesłania pakietu, oraz sprawdzenia czy wiadomość ACK dotarła poprawnie
Kolizja	Do kolizji dochodzi kiedy jakakolwiek część pakietu nakłada się na inny pakiet.	1. Transmisja jest niepoprawna 2. Dokonaj retransmisji pakietów, które biorą udział w kolizji 3. Gdy pakiet przekroczy maksymalną dostępną liczbę retransmisji usuwamy go 4. Zaplanowanie zdarzenia sprawdzania stanu kanału
Błąd TER	Pojawia się gdy wiadomość ACK nie dotrze poprawnie	1. Transmisja jest niepoprawna 2. Dokonaj retransmisji pakietu 3. Gdy pakiet przekroczy maksymalną dostępną liczbę retransmisji usuwamy go 4. Zaplanowanie zdarzenia sprawdzenia stanu kanału
Wysłanie wiadomości ACK	Potwierdzenie poprawnie dostarczonego pakietu do stacji odbiorczej sygnalizowane jest wystawieniem wiadomości ACK.	1. Jeżeli pakiet został poprawnie odebrany, wyślij wiadomość ACK 2. Zaplanowanie zdarzenia czasowego zakończenia wysyłania wiadomości ACK
Sprawdzanie zajętości szczeliny	Jeżeli prawdopodobieństwo PT jest rożne niż 0,4 sprawdzamy zajętość szczeliny.	1. Jeśli jakaś stacja nadawcza chce wysłać pakiet w danej szczelinie czasowej to wyślij pakiet z prawdopodobieństwem PT.

		2. Jeśli nie wysłano pakietu wstrzymaj się z transmisją do następnej szczeliny i ponownie spróbuj wysłać pakiet 3. Jeśli kanał stał się zajęty zaplanuj zdarzenie czasowe nasłuchiwania kanału za 1 ms

3. Parametry wywołania programu

Lambda: 0,3
Liczba stacji nadawczych: 5
Liczba dozwolonych retransmisji: 6
Parametr P: 0,8
Liczba przeprowadzonych symulacji: 10
Resetowanie fazy początkowej: 50 000
Czas jednej symulacji: 300 000

4. Generatory

a) Opis zastosowanych generatorów liczb losowych z histogramami

- **Generator rozkładu równomiernego [0-1]**

Jest to główny generator użyty w programie. Reszta generatorów korzysta z tego generatora.

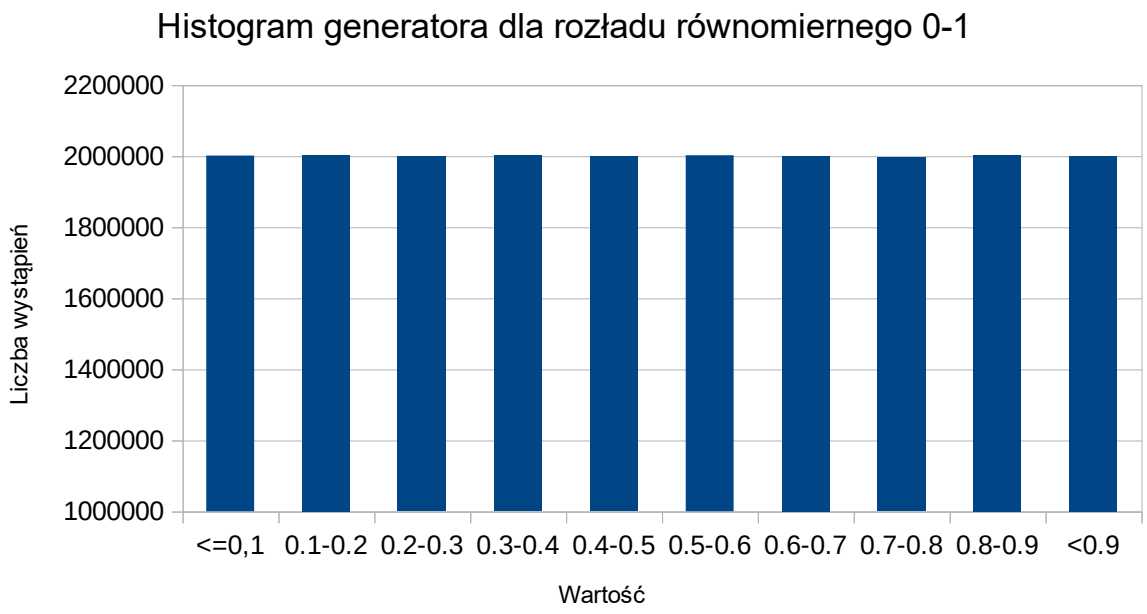
```
double WirelessNetwork::UniformGenerator2(int& seed)
{
    int h = seed / kQ;
    seed = kA * (seed - kQ * h) - kR * h;
    if (seed < 0)seed = seed + static_cast<int>(kM);

    return seed / kM;
}
```

Rys. 3. Implementacja generatora rozkładu równomiernego.

```
for (int i = 0; i < 20000000; i++)
{
    temp_chart = wireless_network->UniformGenerator2(seed_chart);
    tab1[int(temp_chart*10)]++;
}
for (int i = 0; i < 10; i++)
{
    ofstream save("uniform0-1.txt", ios_base::app);
    save <<tab1[i] << endl;
    save.close();
}
```

Rys. 4. Kod przedstawiający zapisywanie wyników.

Na powyższym obrazku widać implementację zapisu danych. W celu uproszenia zbierania wyników zdecydowałem się użyć tablicy 10 elementowej, a wynik wyjściowy z generatora pomnożyć przez 10 i rzutować na int. Dzięki użyciu tej operacji wynik 0,1 wynosi 1 i możemy zwiększyć wartość tablicy o indeksie 1.



Rys. 5. Histogram dla generatora o rozkładzie równomiernym.

- **Generator rozkładu równomiernego [1-10]**

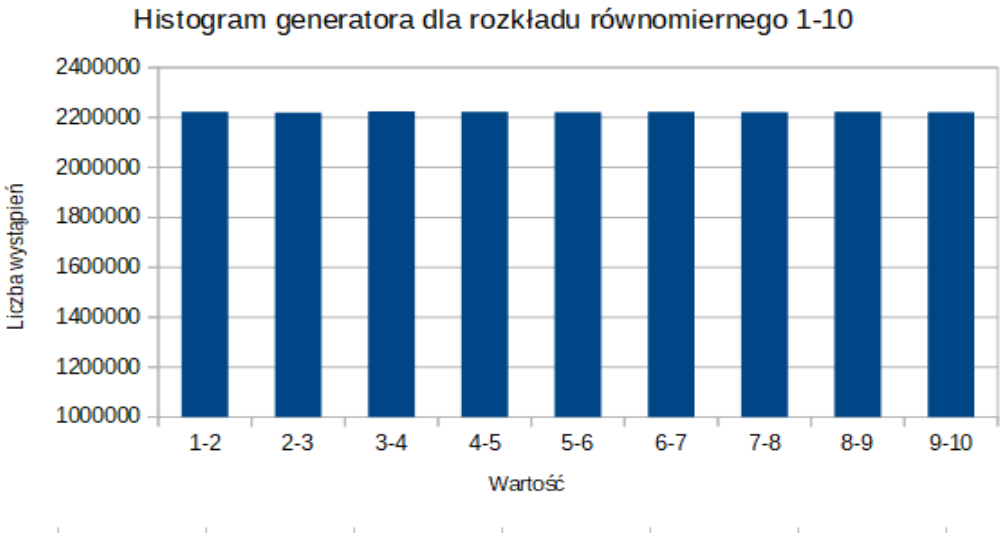
```
double WirelessNetwork::UniformGeneratorRange2(int maks, int min, int& seed)
{
	return UniformGenerator2(seed) * (maks - min) + min;
}
```

Rys. 6. Implementacja generatora rozkładu równomiernego generującego wartości z przedziału [min - max]

```
for (int i = 0; i < 20000000; i++)
{
	temp_chart = wireless_network->UniformGeneratorRange2(10,1,seed_chart);
	tab[int(temp_chart-1)]++;

}
for (int i = 0; i < 10; i++)
{
	ofstream save("uniform1-10.txt", ios_base::app);
	save << tab[i] << endl;
	save.close();
}
```

Rys. 7. Kod przedstawiający zapisywanie wyników.

Jak widać wyniki testowane były dla przedziału [1-10] oraz ponownie użyto tablicy. Wynik z generatora -1 w indeksie tablicy służy tylko do poprawnego indeksowania w tablicy.



Rys 8. Wykres przedstawia histogram dla tego generatora.

- **Generator rozkładu wykładniczego**

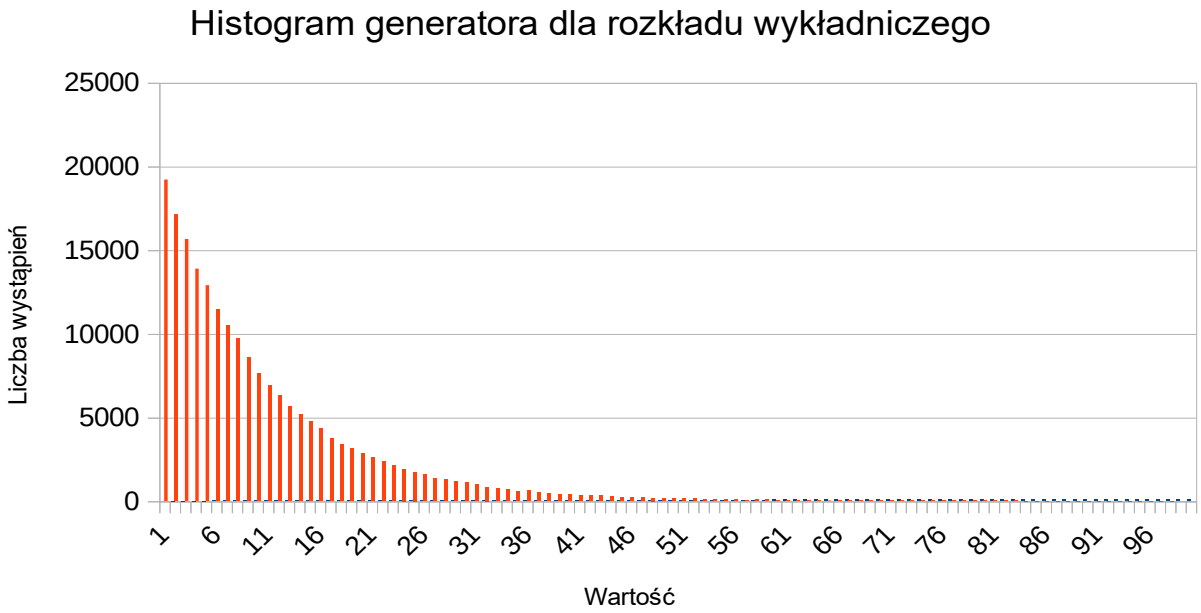
```
double WirelessNetwork::ExponentialGenerator2(double lambda, int& seed)
{
return -(1.0 / lambda) * log(UniformGenerator2(seed));
}
```

Rys. 9. Implementacja generatora rozkładu wykładniczego.

```
for (int i = 0; i < 200000; i++)
{
 temp_chart = wireless_network->ExponentialGenerator2(1,seed_chart);
 tab2[int(temp_chart * 10)]++;
}
for (int i = 0; i < 100; i++)
{
 ofstream save("exp.txt", ios_base::app);
 save << i << " " << tab2[i] << endl;
 save.close();
}
```

Rys. 10. Kod przedstawiający zapisywanie wyników.

Jak widać na załączonym rysunku wynik z generatora wywoływanego z parametrem lambda mnożony jest razy 10 i wpisywany jest do tablicy o rozmiarze 100. Dzięki mnożeniu wyniku razy 10 i rzutowaniu go na int dostajemy wartości w zakresie 0-99.



Rys. 11. Wykres przedstawia histogram dla generatora wykładniczego (parametr lambda=1).

b) Wyjaśnienie w jaki sposób została zapewniona niezależność sekwencji losowych w różnych symulacjach

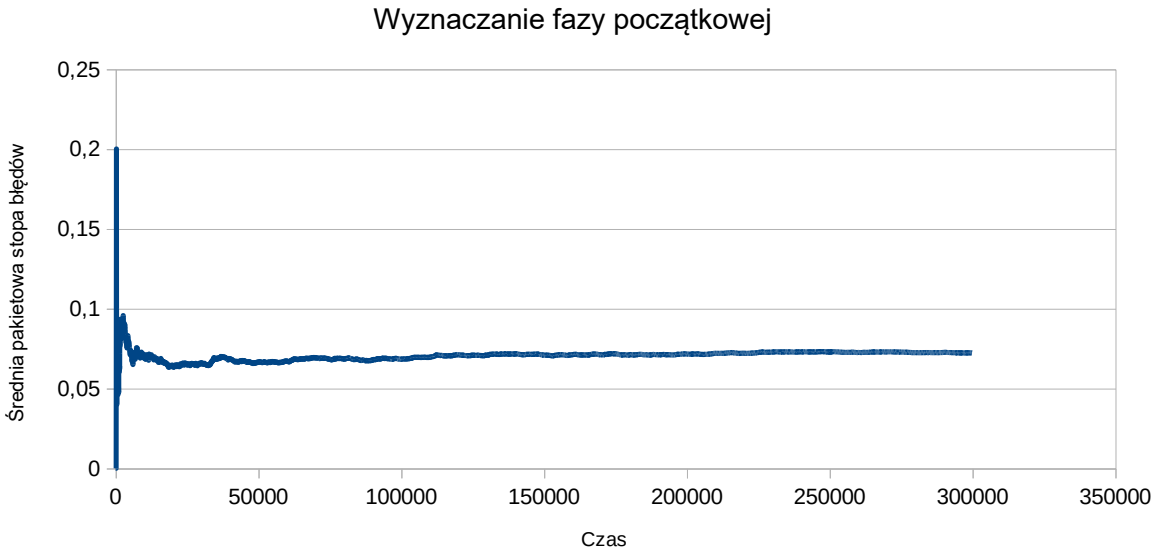
Niezależność sekwencji losowych zapewniono dzięki wygenerowaniu ilości seedów wystarczających do przeprowadzenia 10 symulacji. Ziarna generowano w pętli używając startowego seeda o wartości 5 i używając rozkładu równomiernego w przedziale od 0 do 1 który podmieniał wartość seeda. Seed zapisywany był co 100000 wywołań generatora. Wygenerowano 110 seedów (50 - do generowania pakietów dla 10 symulacji, 50 - do czasu przesyłania dla 10 symulacji, 10 - do decyzji czy pakiet poprawnie został przesłany przez kanał dla 10 symulacji). Seedy przechowywane są w wektorze o nazwie „seeds”. W wektorze, przechowywane są wszystkie seedy potrzebne dla przeprowadzenia określonej liczby symulacji. W programie można ustawić liczbę wykonywanych symulacji.

5. Opis zastosowanej metody testowania i weryfikacji poprawności działania programu

Metoda zastosowana do sprawdzenia poprawności programu opiera się na trybie debug (który służy do szczegółowego wyświetlania tego co dzieje się w programie), oraz zastosowania trybu krokowego. Po każdym przejściu pętli w trybie krokowym symulacja zostaje zatrzymana, aby użytkownik mógł zobaczyć co stało się w danym momencie programu. W chwili zatrzymania symulacji na ekranie widać jakie zdarzenia czasowe są zaplanowane, oraz co zostało już wykonane w danym punkcie (zdarzenia czasowe i warunkowe).

6. Wyniki symulacji

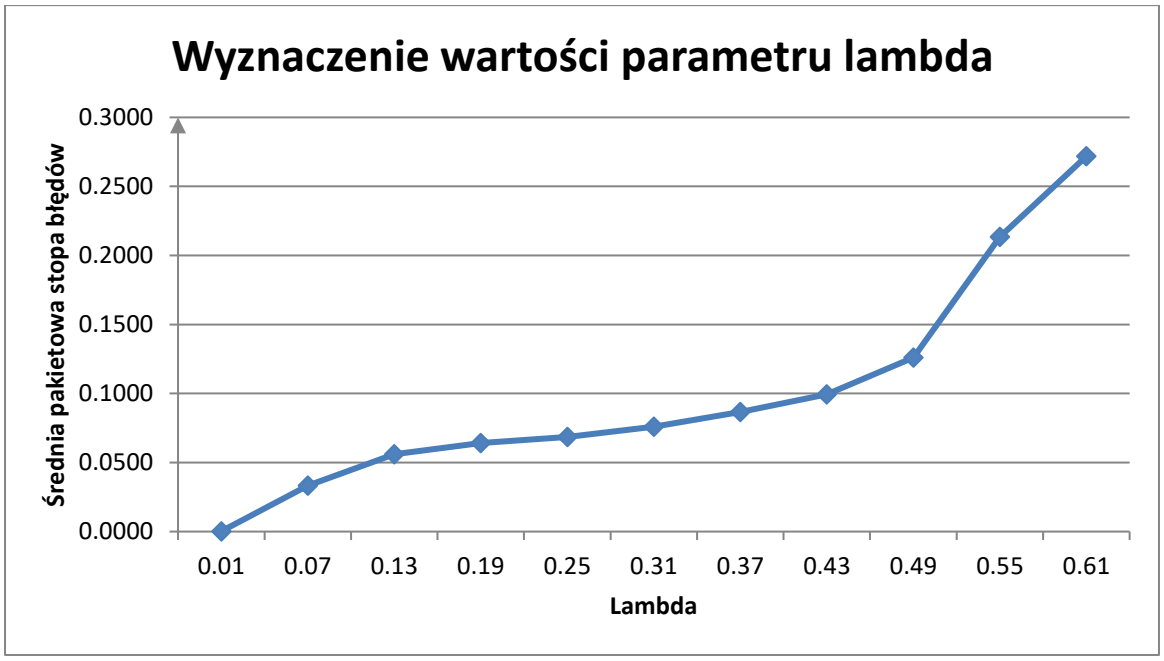
a) Wyznaczenie długości fazy początkowej



Rys. 12. Wyznaczanie fazy początkowej

Wykres wyznaczenia długości fazy początkowej powstał z zapisywania wartości średniej pakietowej stopy błędów zapisywanych w określonym czasie. Dane te są zapisywane w pliku „IinitialPhase.txt”. Z wykresu można wywnioskować, że faza początkowa kończy się w czasie 50 tys. ms.

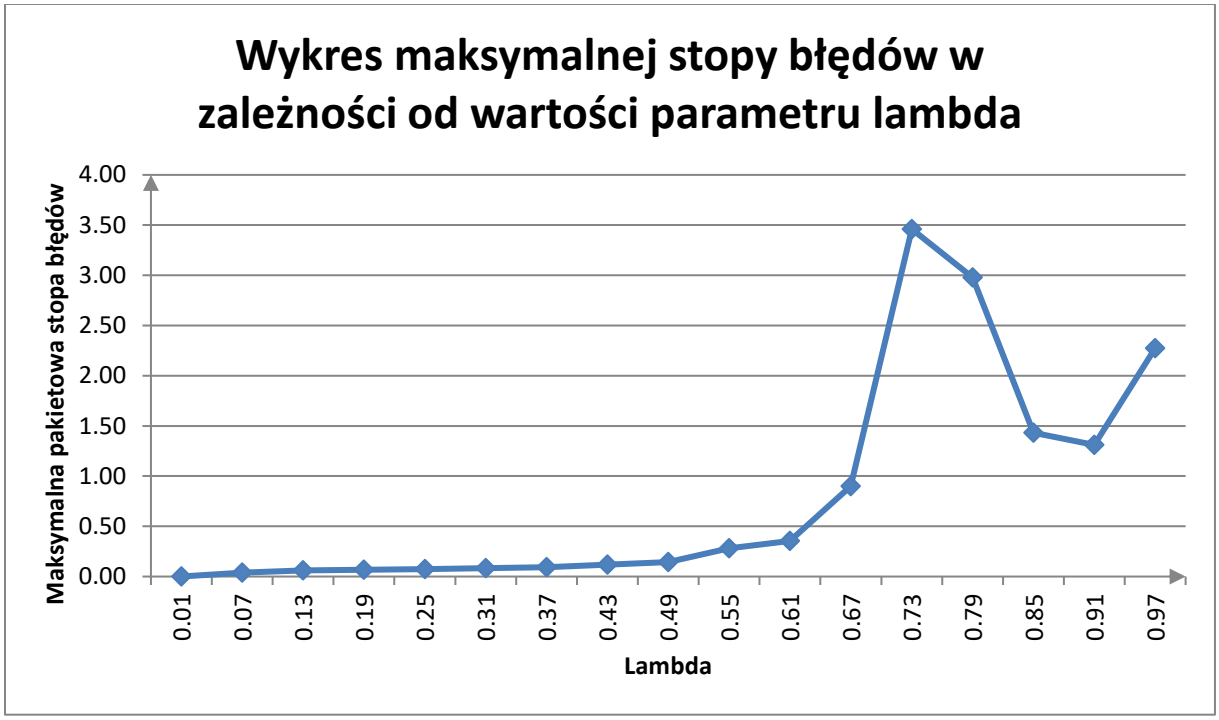
b) Wyznaczenie wartości parametru lambda



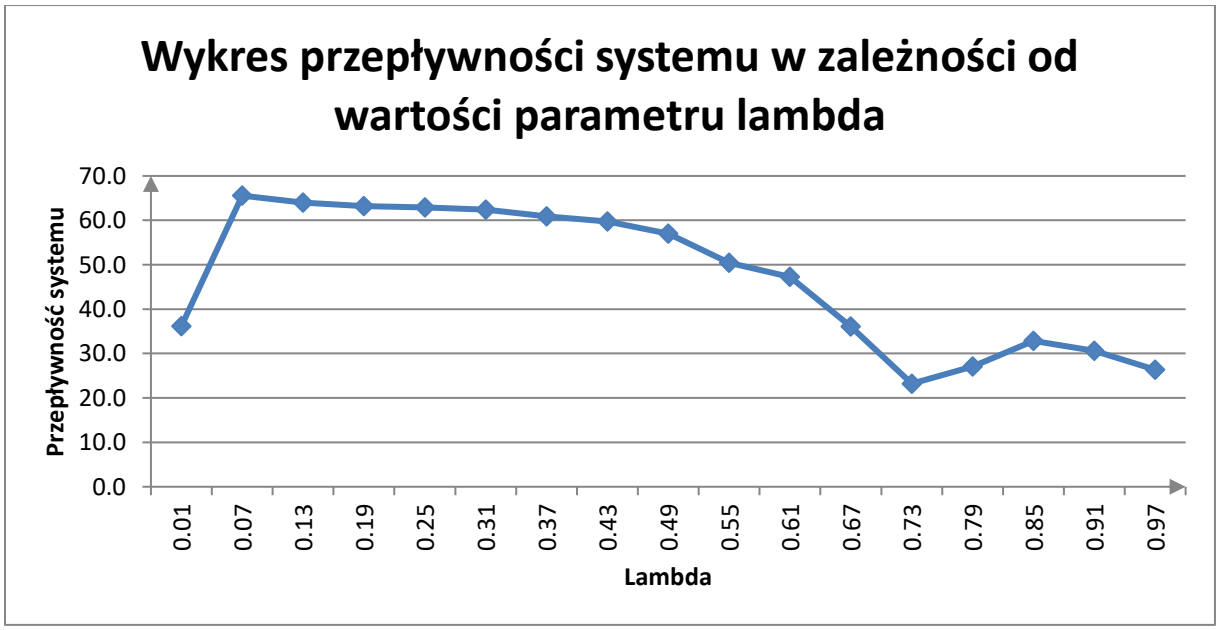
Rys. 13. Wyznaczanie wartości parametru lambda

Do zasymulowania programu wybrałem lambdę o wartości 0,3. Dla tej wartości parametru średnia pakietowa stopa błędów nie przekraczała założonego w poleceniu 0,1.

c) Wykres maksymalnej pakietowej stopy błędów i przepływności w zależności od wartości parametru lambda



Rys. 14. Wyznaczanie maksymalnej stopy błędów

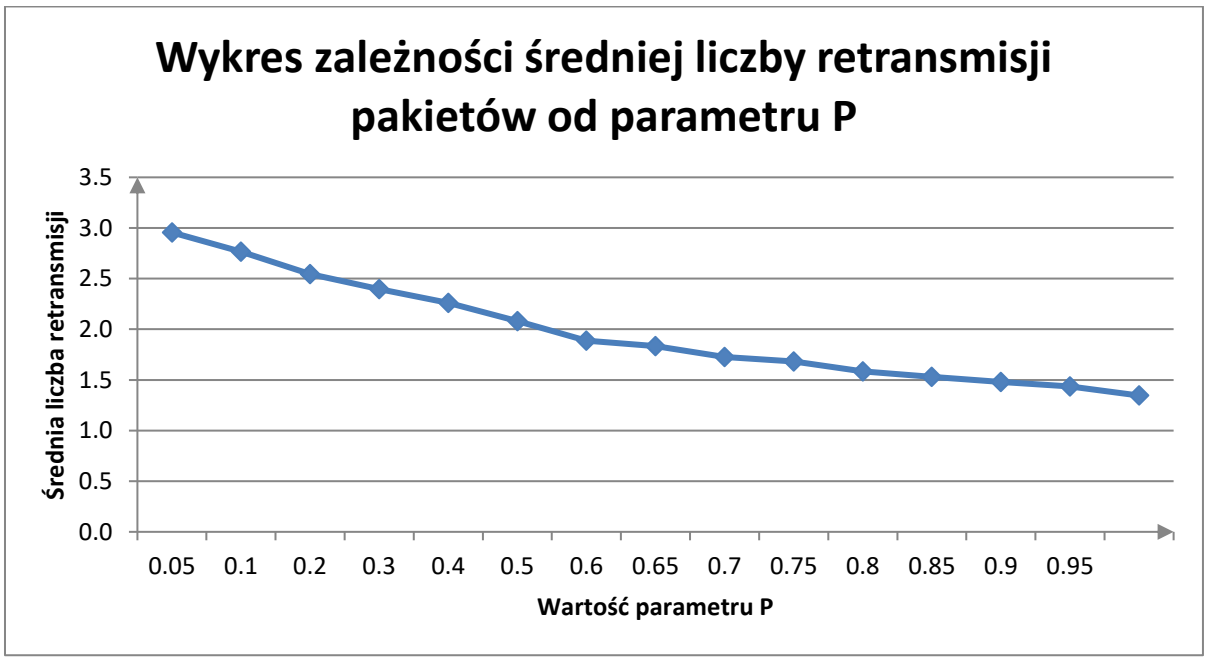


Rys. 15. Wyznaczanie przepływności systemu

d) Tabelka z wynikami symulacji dla każdego przebiegu symulacyjnego wraz z uśrednionymi wynikami po wszystkich przebiegach

	Średnia pakietowa stopa błędów	Maksymalna pakietowa stopa błędów	Średnia liczba retransmisji	Średni czas oczekiwania [ms]	Średnie opóźnienie [ms]	Przepływność [pakiet/s]
1	0,0733046	0,076494	1,58364	166265	166226	62,79
2	0,0731303	0,0757542	1,599	141792	141753	74,9867
3	0,0736384	0,0785926	1,58969	142361	142323	75,22
4	0,0735634	0,081642	1,59861	142047	142018	75,1133
5	0,0786702	0,0850332	1,58653	142014	141976	74,62
6	0,0772870	0,0812745	1,6089	141514	141481	74,15
7	0,0723577	0,0748376	1,60371	142561	142545	75,03
8	0,0721796	0,0768185	1,60562	141676	141658	74,8933
9	0,0755412	0,079602	1,60734	141146	141107	74,73
10	0,0730792	0,0796178	1,59808	141465	141429	75,04
Średnia:						
1.	0,07427516	0,07896664	1,598112	144284,1	144251,6	73,65
Odchylenie standardowe:						
2.	0,002176988	0,00313318	0,008849	7734,96	7732,86	3,83074
Przedział ufności:						
3.	0,001349286	0,00194193	0,005484	4794,09	4792,79	2,37427

e) Wykres zależności średniej liczby retransmisji pakietów od parametru P dla wyznaczonej wartości parametru lambda



Rys. 16. Wykres zależności średniej liczby retransmisji pakietów od parametru P

7. Wnioski

- W programie zmieniono ilość stacji nadawczych na 5, oraz maksymalną liczbę retransmisji na 6. Dane te zostały dobrane eksperymentalnie tzn. uruchamiano symulację dla różnych parametrów i sprawdzano wyniki. Ostatecznie postanowiłem użyć powyższych parametrów oraz lambdy 0.3, ponieważ parametry te były najbliższe założeniom projektowym podanym w treści polecenia tj. średnia pakietowa stopa błędów nie większa niż 0,1.

- Dla wykresu maksymalnej stopy błędów w zależności od wartości parametru lambda możemy stwierdzić, że dla pewnej wartości parametru lambda maksymalna pakietowa stopa błędów osiąga maksymalną wartość której nie da się przekroczyć wraz ze zwiększającą się lambdą. Parametr lambda odpowiada za czas generowania pakietu. Im większy parametr lambda tym szybciej pojawi się nowy pakiet. W związku z występowaniem bufora w którym pakiety są przechowywane i czekają na przesłanie biorąc pod uwagę czas transmisji, występowanie kolizji oraz retransmisji. Maksymalną pakietową stopę błędów można podnieść parametrem lambda tylko do określonej wartości.

- Dla wykresu przepływności systemu w zależności od parametru lambda można stwierdzić, że zbyt mała lambda powoduje zmniejszenie wartości tego parametru i jest to spowodowane tym, że kanał jest pusty ponieważ brakuje pakietów które można by przesłać. Natomiast przy zbyt dużej wartości lambda przepływność systemu maleje ponieważ występują kolizje w kanale. Jednak niemożliwe jest aby dobierając zbyt duży parametr lambda obniżyć przepływność systemu poniżej pewnego poziomu.

- Z wykresu zależności średniej liczby retransmisji pakietów od parametru P możemy stwierdzić, że średnia liczba retransmisji rośnie, kiedy maleje wartość P. Parametr P określa to czy pakiet zostanie odebrany poprawnie czy nie. Wykres więc jest poprawny, a dzięki zastosowaniu poprawnych generatorów jest on w przybliżeniu liniowy.