|  |
| --- |
|  |
| Teoria i inżynieria ruchu teleinformatycznego |
| projekt |

1/1/2016

|  |
| --- |
| Marcin Borowiński 203393  Thibault Dehacq |

# Wstęp

Zadaniem projektu jest zamodelowanie ruchu teleinformatycznego w wybranym programie. Do tego celu wybraliśmy silnik do gier „Unreal engine” wyprodukowany przez firmę Epic Games. Pozwala on na blokową implementację algorytmów oraz wizualizację obiektów 3D.

Ruch teleinformatyczny może być zaimplementowany w różne sposoby w zależności od potrzeb systemów lub użytkowników. Administratorzy oraz architekci sieci mogą mieć wpływ na wiele czynników, takich jak sposób generowania ruchu w sieci, kontrola przyjęć lub harmonogramowanie ruchu.

Na początek zdefiniujmy czym jest ruch teleinformatyczny. Są to strumienie jednostek danych generowane w ramach procesów komunkacjyjnych obsługoiwane przez dane protokoły kumunikacyjne. Z kolei protokół komunikacyjny jest zbiorem reguł i kroków postępowania wykonywanych w sposób automatyczny przez urządzenia komunikacyjne. Protokoły zapewniają łączność i wymianę danych pomiędzy urządzeniami. Protokoły dzielimy na klasyczne, internetowe oraz stosowane w automatyce przemysłowej.

Podstawowym zadaniem sterowania ruchem w sieciach teleinformatycznych jest dostarczenie jakości usługi dla użytkownika w sieci oraz optymalne wykorzystanie dostępnych zasobów. Do podstawowych zadań związanych ze sterowaniem ruchem teleinformatycznym jest wyznaczanie tras, przeciwdziałania przeciążeniom sieci oraz sterowanie przepływem danych. Złożoność tych zadań rośnie wraz z urozmaiceniem grupy użytkowników w tej samej siecie teleinformatycznej.

W naszym przypadku będziemy mieć wpływ na sposób generowania ruchu, czyli odstępy między generowaniem kolejnych pakietów. Koncentrujemy się na rozkładzie Poissona, On-Off, wykładniczy oraz MMPP, które zostaną omówione pózniej. Dodatkowo zostaną zaimplementowane oraz analizowane takie kompomenty jak Admission Control, który jest procesem zatwierdzania w komunikacji pakietów. Innymu słowa sprawdza, czy podczas komunikacji pakietów są wystarczające środki. Scheduler, czyli algorytm kolejkowania, czyli w jaki sposób są wysyłane oraz odbierane pakiety. Profilowanie ruchu, czyli algorytmy które służą do kontrolowania ilości pakietów przechodzących przez dany węzeł w trakcie sekundy.

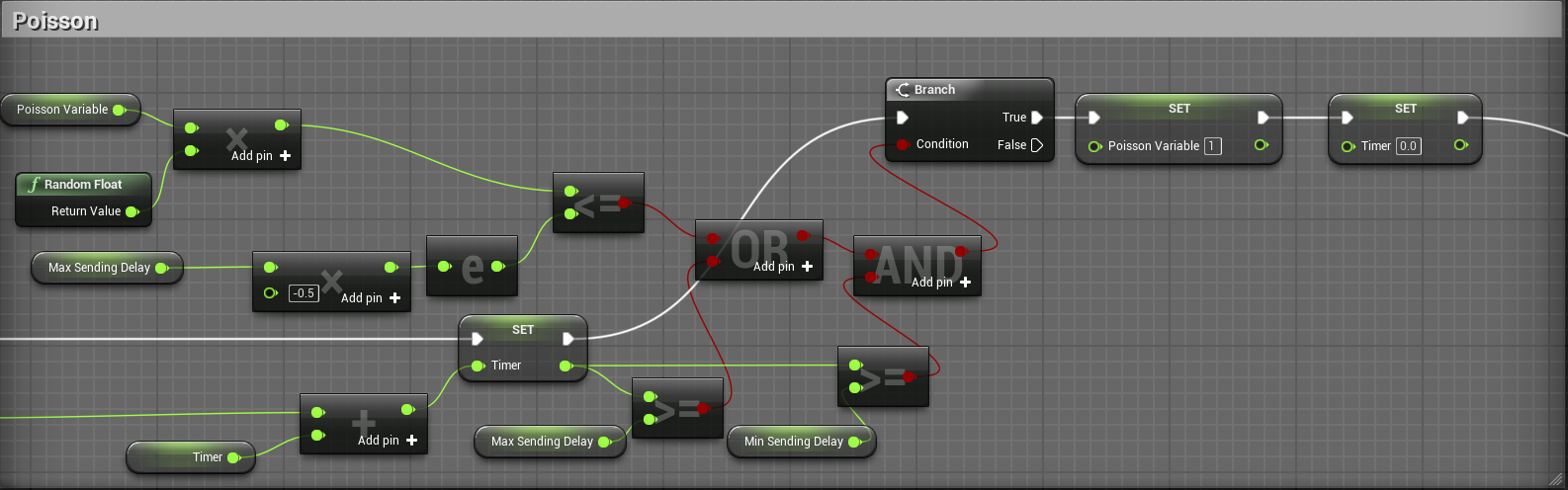
# Cel dokumentu

Dokument ma na celu wprowadzić osobę do problematyki ruchu teleinformaczycznego oraz opisanie i analizę zaimplementowanych algorytmów. Tego typu zadanie ma powinno wykazać, przynajmniej w teorii, jakie algorytmy się najlepiej sprawują w danej sytuacji.

# Przedstawienie problemu

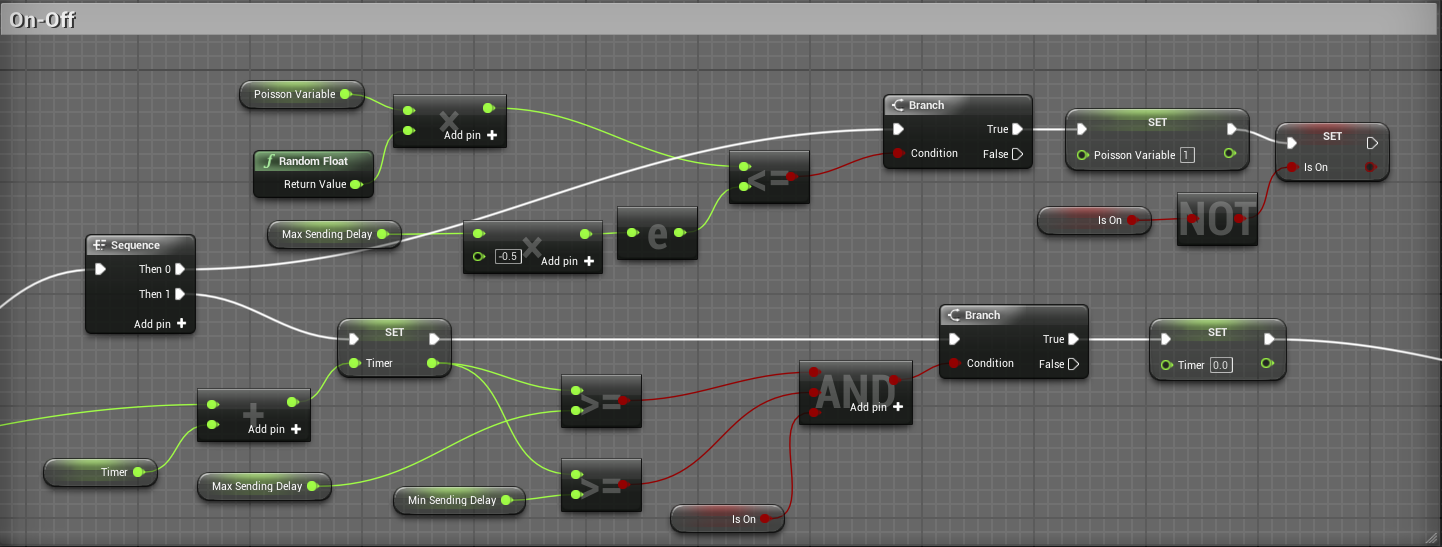
## Opis algorytmów

Pierwszym omawianym oraz implemntowanym algorytmem jest generowanie ruchu według rozkładu Poissona. Określa on prawdopodobieństow wysłania pakietu w danym momencie czasu. Na rozkład poissona można wpłwyać za pomocą parametru λ. Im jest on większy tym bardziej rozkład przypomina normlany.



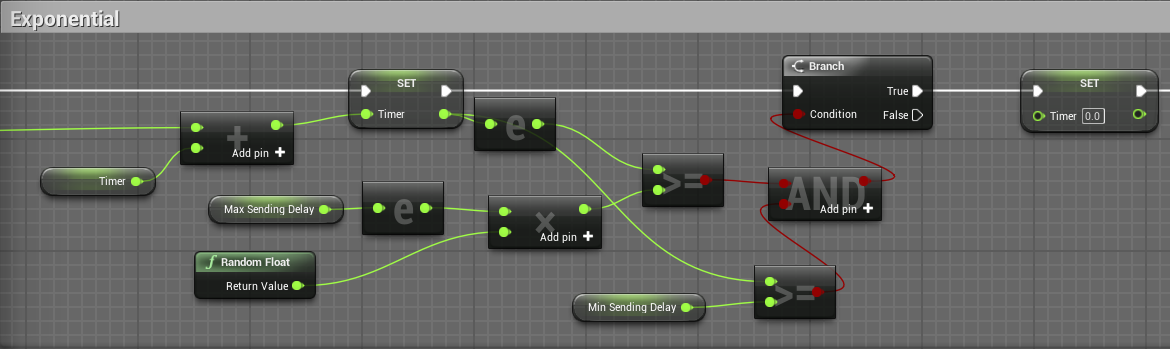
*Rysunek 3.1.1 przedstawiający blokową implementację alogrytmu generującego ruch według rozkładu Poissona.*

Drugim implementowanym algorytmem jest generowanie ruchu według rozkładu On-Off. W którym istnieją tylko dwa stany. On jest stanem w któtym pakiety są wysyłane natomiast stan Off jest stanem przeciwnym, żaden pakiet nie jest wysyłany.



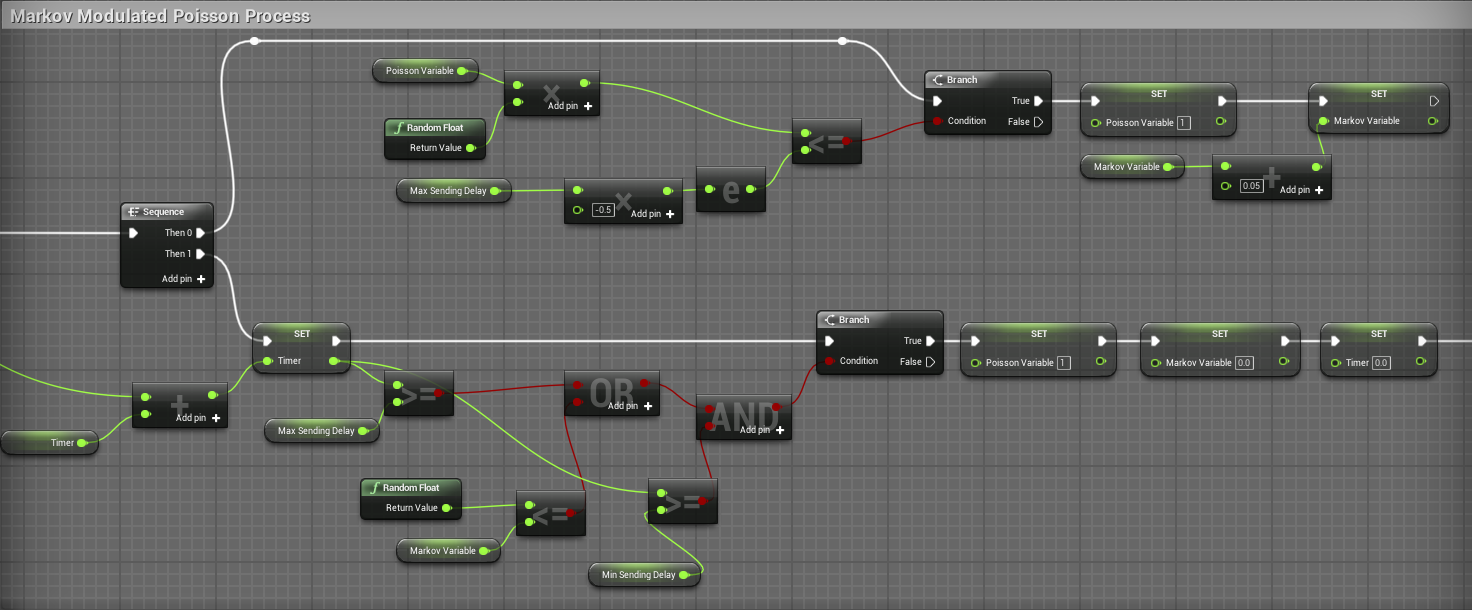
*Rysunek 3.1.2 Reprezentujący blokowa implementację algorytmu generującego ruch według rozkładu On-Off*

Trzecią implementacją jest generowanie ruchu teleinformatycznego według rozkładu wykładniczego. Opisuje fenomeny bez pamięciowe innymi słowa, prawdopodobieństwo nie zależy od stanu poprzedniego. Według rozkładu im dłuższy czas mija, tym prawdopodobieństwo na przykład wysłania pakietu jest większe. Paramentrem tego rozkładu podobnie jak w rozkładzie Poissona jest λ. Wraz z jej wzrostem, spłaczamy wykres.



Rysunek 3.1.3 Reprezentujący blokową implementację algorytmu generującego ruch według rozkładu Wykładniczego.

Ostanim implementowanym algorytmem ruchu jest rozkład Markov Modulated Piosson Process. Jest to modyfikacja rozkładu Poissona. Każdy proces Poissona jest przełączany między ciągłym łańcuchem Markowa.



*Rysunek 3.1.4 Reprezentujący blokową implementację algorytmu generującego ruch według rozkładu MMPP*.

Algorytmy kolejkowania:

FIFO – czyli first in first out, polega na tym, że pierwszy pakiet który dochodzi do węzła jest pierwszym pakietem który z tego węzła zostanie wysłany.

LIFO – czyli last in first out, polega na tym, że ostatni pakiet który dochodzi do węzła jest pierwszym pakietem który zostanie z tego węzła wysłany.

WRR – czyli weighted round robin, polega na obsługiwaniu pakietów w sposób cykliczny, lecz algorytmy o większej wadze ( większy priorytet) są obsługiwane wcześniej tak, aby waga wszystkich pakietów była taka sama.

Kolejka priorytetowa –

## Wyniki symulacji i ich analiza

Analiza implementowanego programu polegała na badaniu średnich czasów wysyłania pakietów w zależności od rozkładu generowanego ruchu oraz algorytmów kolejkowania. Podczas badań następujące zmienne nie były modyfikowane:

-Liczba węzłów = 5

-Minimalna liczba odbiorców = 3

-Minimalne opóznienie wysłania = 5 [s]

-Liczba pakietów w węzle = 5

-Maxymalna liczba odbiorców = 3

-Maxymalne opóznienie wysłania = 5 [s]

-Leaky bucket size = 5

Otrzymane wyniki dla rozkładu Poissona i kolejkowania FIFO, LIFO i priority queue.

|  |  |  |  |
| --- | --- | --- | --- |
| Piosson | Fifo | Lifo | Priority |
|  | 1,197234 | 1,061471 | 3,021988 |
|  | 1,637968 | 1,880434 | 3,450442 |
|  | 1,647933 | 2,240472 | 4,075903 |
|  | 2,574592 | 2,515604 | 5,522051 |
|  | 1,097964 | 1,651258 | 3,328146 |
|  | 2,90669 | 3,282518 | 4,383612 |
|  | 1,794568 | 1,0611812 | 3,401916 |
|  | 3,122564 | 1,590284 | 6,965164 |
|  | 1,745207 | 1,341135 | 5,325416 |
|  | 1,613906 | 1,881645 | 6,654546 |
|  | 1,075496 | 5,246584 | 8,146514 |
|  | 2,689394 | 1,962145 | 3,265454 |
|  | 1,126218 | 5,451645 | 4,165465 |
|  | 1,829141 | 1,596452 | 4,265446 |
|  | 1,578523 | 6,210081 | 7,116464 |
|  | 2,086878 | 1,028013 | 9,165441 |
|  | 1,216676 | 3,133764 | 4,415465 |
|  | 1,925009 | 3,266399 | 3,146954 |
|  | 5,268195 | 2,294565 | 3,165468 |
|  | 1,368571 | 1,796452 | 3,246544 |
|  | 1,350284 | 3,305465 | 3,416456 |
|  | 4,966212 | 6,492654 | 3,465446 |
| AVG | 2,08269195 | 2,7404646 | 4,68683186 |

|  |  |  |  |
| --- | --- | --- | --- |
| On-Off | Fifo | Lifo | Priority |
|  | 10,04965 | 10,000181 |  |
|  | 10,054795 | 10,000181 |  |
|  | 11,946542 | 10,000181 |  |
|  | 17,465446 | 10,000181 |  |
|  | 10,165445 | 11,900373 |  |
|  | 10,065445 | 10,008787 |  |
|  | 10,000654 | 10,008787 |  |
|  | 9,9999416 | 10,008787 |  |
|  | 9,9999859 | 11,076088 |  |
|  | 30,827166 | 10,162033 |  |
|  | 10,000031 | 10,162033 |  |
|  | 10,000031 | 21,094698 |  |
|  | 9,999996 | 9,999973 |  |
|  | 9,999966 | 22,333279 |  |
|  | 60,217674 | 9,999962 |  |
|  | 9,999996 | 9,999962 |  |
|  | 9,999966 | 9,999962 |  |
|  | 10,000031 | 10,00034 |  |
|  | 10,000031 | 10,000054 |  |
|  | 10,000038 | 62,375015 |  |
| AVG | 14,0396415 | 13,9565429 |  |

Otrzymane wyniki dla rozkładu On-Off i kolejkowania FIFO, LIFO i priority queue.

|  |  |  |  |
| --- | --- | --- | --- |
| Exponential | Fifo | Lifo | Priority |
|  | 4,022345 | 3,377654 |  |
|  | 4,334564 | 3,64528 |  |
|  | 5,864146 | 5,365449 |  |
|  | 6,046876 | 6,214654 |  |
|  | 6,754132 | 3,914465 |  |
|  | 5,511354 | 4,145654 |  |
|  | 5,165446 | 2,941064 |  |
|  | 4,823445 | 5,314654 |  |
|  | 4,776545 | 1,214564 |  |
|  | 6,795431 | 4,564984 |  |
|  | 3,4564214 | 4,486541 |  |
|  | 6,125987 | 6,066465 |  |
|  | 4,524984 | 2,08269195 |  |
|  | 5,1846546 | 2,798462 |  |
|  | 6,3054654 | 5,744564 |  |
|  | 5,846545 | 4,785462 |  |
|  | 5,0546545 | 4,245654 |  |
|  | 4,452654 | 6,104654 |  |
|  | 6,624684 | 5,846548 |  |
|  | 3,056565 | 4,56546 |  |
| AVG | 5,23634495 | 4,3712462 |  |

Otrzymane wyniki dla rozkładu Wykładniczego i kolejkowania FIFO, LIFO i priority queue.

Otrzymane wyniki dla rozkładu MMPP i kolejkowania FIFO, LIFO i priority queue.

|  |  |  |  |
| --- | --- | --- | --- |
| MMPP | Fifo | Lifo | Priority |
|  | 1,009026 | 1,02481 |  |
|  | 1,028465 | 1,055714 |  |
|  | 1,334564 | 1,114564 |  |
|  | 1,863375 | 1,984564 |  |
|  | 1,126544 | 1,000896 |  |
|  | 1,064435 | 1,132896 |  |
|  | 2,161108 | 2,684654 |  |
|  | 1,428645 | 1,007544 |  |
|  | 1,110044 | 1,112837 |  |
|  | 2,113308 | 1,016809 |  |
|  | 1,184652 | 1,249451 |  |
|  | 1,846522 | 2,498556 |  |
|  | 1,097465 | 1,112484 |  |
|  | 6,345546 | 1,034955 |  |
|  | 1,052121 | 1,062254 |  |
|  | 2,44584 | 1,02455 |  |
|  | 1,034565 | 3,984621 |  |
|  | 1,154565 | 1,334564 |  |
|  | 4,269458 | 1,721654 |  |
|  | 2,176242 | 1,999984 |  |
|  | 1,011754 | 1,099992 |  |
|  | 1,174555 | 1,134565 |  |
| AVG | 1,77421814 | 1,47240536 |  |

Analiza otrzymanych wyników pozwala na wyciągnięcie następujących wniosków. Zazwayczaj najszybszym algorytmem kolejkowania jest LIFO, gdyż w 3 przypadkach na 4 ma on najkrótsze czasy wysyłania pakietów, ale różnica z FIFO jest przeważnie mała. Najgorszym algorytmem kolejkowania wychodzi na kolejkę priorytetową, gdyż czas wysyłania pakietów różni się dość dużymi wartościami.

Porównując rozkłady generowania ruchu, najlepszym z nich jest MMPP, następnie Poisson oraz Wykładniczy a na końcu Algorytm On-Off, których czasy wysyłania pakietów są wyższe niż dla innych algorytmów.

## 3.3 Podsumowanie