**Kotarski Sebastian, 212776  
Malarski Mateusz, 212878  
Zawisza Artur, 212675**Grupa śr. 13:15

**Teoria i Inżynieria Ruchu Teleinformatycznego**

**Badanie symulacyjne pracy przełącznika sieciowego**

Prowadzący:  
dr hab. inż. Krzysztof Juszczyszyn

Wrocław, 07.06.2017 r.

# Architektury przełączników sieciowych

Przełącznik sieciowy (ang. switch) jest urządzeniem pracującym w drugiej warstwie modelu ISO/OSI, którego zadaniem jest przekazywanie pakietów tej warstwy (ramek) między urządzeniami sieci lokalnej.

Przełączniki posiadają serię wejść oraz wyjść nazywanych portami, które to umożliwiają podłączenie innych urządzeń. Porty nie muszą być w obrębie switcha jednorodne  
- mogą występować zarówno porty jednokierunkowe – wejściowe oraz wyjściowe, jak i mieszane  
- wejściowo-wyjściowe, a każdy z nich może pracować na innej warstwie fizycznej – pojedyncze urządzenie może posiadać zarówno np. porty Ethernet, jak i te oparte na światłowodzie.

Jak wspomniane zostało wcześniej, główną rolą przełącznika jest przesyłanie pakietów między portami. W praktyce sprowadza się do czterech możliwych sytuacji pracy przełącznika – przełącznik może albo przesyłać pakiety w całości, albo dzielić je na tzw. Komórki (ang. cells) o stałym rozmiarze, zaś sama sieć może operować albo na pakietach o stałym rozmiarze, albo pakietach o rozmiarach różnych, o czego konsekwencjach mówi patent [1].

Cykl pracy dowolnego przełącznika składa się z czterech etapów:

1. Odbiór na porcie wejściowym,
2. Decyzja o przekierowaniu,
3. Przesłanie z poru wejściowego na port wyjściowy,
4. Transmisja w medium,

Etap pierwszy sprowadza się do odbioru pakietu przez port wejściowy, co oznacza odczytanie pakietu z medium transmisyjnego przez protokoły warstwy fizycznej.

W etapie drugim switch podejmuje decyzję odnośnie tego, czy odebrany pakiet ma być dalej przetwarzany, czy też pakiet ten należy odrzucić.

Etap trzeci polega na komunikacji między portem wejściowym, a odpowiednim portem wyjściowym, w efekcie czego przyjęty w etapie drugim pakiet zostaje przesłany z portu wejściowego na port wyjściowy.

Ostatni etap polega na wysłaniu pakietu w medium transmisyjne portu wyjściowego.

Fizyczne implementacje tego prostego schematu poskutkowały powstaniem dwóch rodzin rozwiązań różniących się realizacją trzeciego etapu cyklu [1]:

* przełączniki z pamięcią współdzieloną,
* przełączniki bazujące na przełącznicy krzyżowej.

Rozwiązania bazujące na pamięci współdzielonej charakteryzuje się obecnością dużego bloku pamięci, do którego dostęp współdzielą wszystkie porty przełącznika. Pakiety z portów wejściowych umieszczane są w pamięci, skąd odpowiednie porty wyjściowe je odczytują  
- pamięć stanowi zarówno bufor, jak i miejsce transmisji pakietów. Rozwiązanie przypomina pracę  
w środowisku wieloprocesowym, w którym procesy komunikują się za pośrednictwem pamięci współdzielonej i charakteryzuje się podobnymi problemami – występuje ciągła walka portów  
o dostęp do pamięci, co wraz ze wzrostem liczby i rozmiaru pakietów oraz liczby konkurujących portów skutkuje powstawaniem dużych opóźnień [1]. Tego typu przełączniki  
nie posiadają jednak pewnych wad drugiego podejścia, na co zwraca uwagę dokument [2].

Przełączniki bazujące na przełącznicy krzyżowej opierają się o strukturę, w której każdy port wejściowy może zostać połączony z każdym portem wyjściowym poprzez dynamiczne zestawianie fizycznych połączeń – porty wejściowe, jak i wyjściowe posiadają własne magistrale, które,  
w dowolnej chwili, mogą zostać ze sobą zwarte. Nazwa pochodzi od struktury kraty, którą tworzą magistrale portów wejściowych i wyjściowych, co przedstawione zostało na Rys. 1.



Rysunek Budowa przełącznicy krzyżowej (kratowej) [3]

Podstawowymi zaletami, w porównaniu do urządzeń z pamięcią współdzieloną, są brak rywalizacji o pamięć oraz znacznie większe prędkości transmisji między portami wejściowymi  
i wyjściowymi [1]. W tym podejściu pamięć jest rozproszona pod postacią kolejek pakietów/komórek, które to stanowią bufory przełącznika [1]. Jako, że kratownica może mieć jednocześnie zestawionych wiele niekonfliktowych połączeń (podobnie jak rozłączne obszary pamięci przełączników pierwszego rodzaju), ale prędkości transmisji są wyższe, tak więc opóźnienia dla tego rodzaju przełączników powinny być niższe niż w tych o współdzielonej pamięci [1].

Kolejki w danym przełączniku mogą być zlokalizowane w różnych miejscach, co skutkuje obecnością trzech podrodzajów przełączników kratownicowych [3]:

1. Przełączniki o kolejkowaniu wyjściowym (ang. output queueing),
2. Przełączniki o kolejkowaniu wejściowym (ang. input queueing),
3. Przełączniki o wirtualnym kolejkowaniu wyjściowym (ang. virtual output queueing, VOQ),

Dla kolejkowania wyjściowego każdy port wyjściowy switch’a posiada kolejkę pakietów oczekujących na transmisję do medium. Jest to bardzo prosta do implementacji opcja, jednak tego typu kolejkowanie wymusza natychmiastowe odrzucenie pakietu jeśli nie można zestawić połączenia portu wejściowego z wyjściowym oraz generuje problem N-krotnego przyśpieszenia (ang. N-speedup problem), dlatego też nie jest ono pożądane [3, 4, 5]. Problem N-krotnego przyśpieszenia sprowadza się do tego, że każdy port wyjściowy musi posiadać kolejkę zdolną pomieścić tyle pakietów  
o możliwym największym, ile jest wejść (N) oraz prędkość transmisji portu wyjściowego musi być  
N-razy większa od prędkości transmisji portu wejściowego [3, 5].

Rozwiązanie tego problemu stanowić ma kolejkowanie wejściowe, gdzie tym razem każde wejście posiada kolejkę. Pozwala to rozwiązać problem N-krotnego przyśpieszenia, ale wprowadza problem blokowania kolejki przez pakiet na czele (ang. head-of-line blocking, HOL bloking) [3, 4, 5].  
Dla kolejki FIFO (ang. First In First Out) może to skutkować zmniejszeniem przepustowości do 58,6% [3].

Problem blokowania ma rozwiązywać wirtualne kolejkowanie wyjść będące rozszerzeniem kolejkowania wejściowego. Polega ono na tym, że każde wejście posiada po kolejce na wyjście przełącznika i odebrawszy pakiet skierowany na konkretne wyjście umieszcza go w jego dedykowanej kolejce. W przypadku tego kolejkowania pozostaje tylko problem wyboru wejścia, z którego w danym momencie będą pakiety przekazywane [3].

Ostatnim elementem przełącznika jest jego algorytm szeregujący (ang. scheduler).  
Zadaniem algorytmu jest ustalenie, które porty są ze sobą, w danym takcie, zestawione.  
Dla przełączników sieciowych wykorzystuje się wiele różnych algorytmów np. WFQ, WF2Q,  
Deficit Round-Robin [3, 4, 5], a tworzenie nowych stanowi temat wielu prac naukowych.

Dokument [3] stwierdza, że obecnie standard szeregowania dla przełączników stanowi algorytm iSLIP, przy czym jest on szczególnie popularny przy wirtualnym kolejkowaniu wyjść. Zakłada on,  
że przełącznik bazuje na kratownicy i pomiędzy portami dokonywana jest transmisja komórek.  
Jedno okno czasowe, w którym następuje przesyłanie komórki nazywamy czasem komórkowym   
(ang. cell-time). Podczas jednego czasu komórkowego algorytm ustala połączenia między portami wejściowymi i wyjściowymi oraz dokonuje synchronicznej transmisji komórek uszeregowanych na ten konkretny takt.

Niech przełącznik ma NI wejść I1, I2, …, INI oraz NO wyjść O1, O2, …, ONO. W iSLIP każde wejście Ii ma listę cykliczną O1, O2, …, ONO ze wskaźnikiem ai domyślnie wskazującym na O1. Lista ta nazywa się harmonogramem karuzelowym (ang. Round Robin Schedule) dla wejścia Ii. Wyjście wskazywane przez ai ma najwyższy priorytet, a kolejne (mod NO) – odpowiednio niższe. Analogicznie każde wyjście Oj ma wskaźnik gj wskazujący wejście o najwyższym priorytecie.

Algorytm iSLIP składa się z trzech kroków:

1. Żądanie (ang. request): Każde niesparowane wejście wysyła żądania do każdego wyjścia, dla którego ma oczekującą komórkę,
2. Zezwolenie (ang. grant): Jeżeli niesparowane wyjście otrzyma jakieś żądania,  
   to przyznaje zezwolenie wejściu o najwyższym priorytecie według jego harmonogramu. Wyjście informuje każde z wejść, czy otrzymało zezwolenie, czy też nie. Wskaźnik  
   gj zostaje zwiększony (mod NI) do jednej pozycji za wejściem, które uzyskało zezwolenie wtedy i tylko wtedy, gdy zezwolenie zostanie zaakceptowane w kroku 3.
3. Akceptacja (ang. accept): Jeżeli wejście otrzyma zezwolenia, to akceptuje  
   te o najwyższym priorytecie na podstawie swojego harmonogramu. Wskaźnik ai jest zwiększany do pozycji o jeden dalszej niż zaakceptowana (mod NO).

Jako, że część zezwoleń może zostać niezaakceptowana iSLIP może wykonać do NO-krotnego powtórzenia sekwencji Żądanie-Zezwolenie-Akceptacja celem zwiększenia liczby dopasowań.

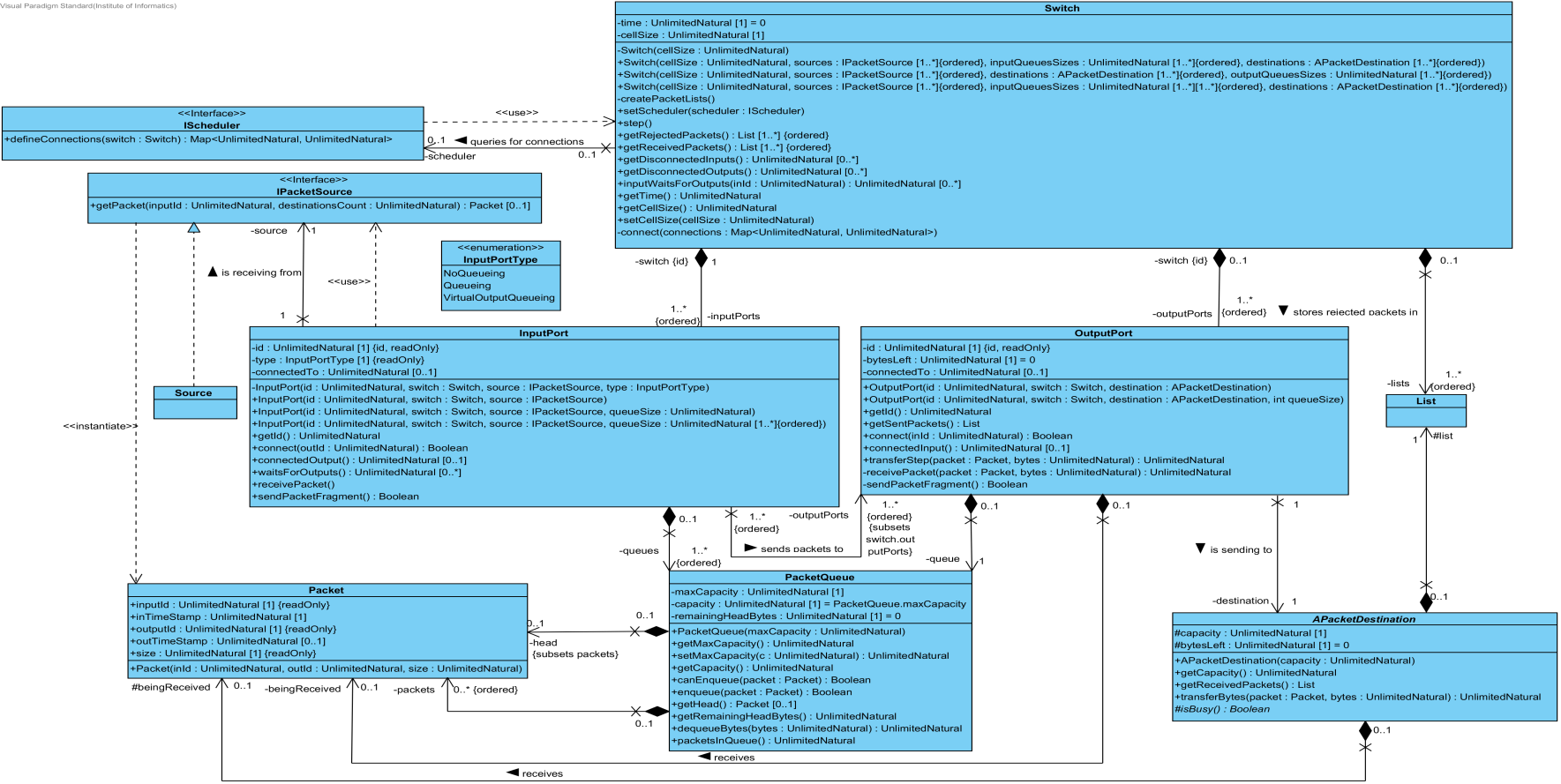
ISLIP może uzyskać 100% efektywności dla ruchu równomiernego oraz do uczciwego szeregowania bez zagłodzeń kolejek wejściowych dla ruchu nierównomiernego [3].

# Cel projektu

Celem niniejszego projektu jest stworzenie narzędzia do symulacji pracy przełącznika sieciowego, a następnie przeprowadzenie badań dotyczących efektywności pracy tego przełącznika  
w zależności od przyjętych parametrów symulacji.

Za modelowane przełączniki przyjęto te oparte o przełącznicę krzyżową w trzech wersjach kolejkowania pakietów. Na algorytm sterujący wybrany został algorytm iSLIP.

# Architektura symulatora

Rys. 2 przedstawia diagram klas dla przełącznika, rozumianego jako fizyczny sprzęt, stanowiącego serce symulatora.

Rysunek Diagram klas implementacji przełącznika - sprzętu

Jak zostało przedstawione na diagramie, implementację przełącznika tworzy raptem parę klas. Głównymi elementami tej implementacji są klasy: *Switch* – reprezentująca cały przełącznik,  
*InputPort* – port wejściowy przełącznika, *OutputPort* – port wyjściowy przełącznika, *Packet* – klasa pakietu, *IPacketSource* – interfejs urządzeń generujących pakiety, *APacketDestination* – interfejs urządzeń odbierających pakiety oraz *IScheduler* – interfejs algorytmu szeregowania.

Przełącznik tworzy przynajmniej para portów – jeden wejściowy oraz jeden wyjściowy.  
Każdy port, w zależności od typu przełącznika, może posiadać kolejki pakietów, zgodnie z definicją danego typu przełącznika. Do każdego wejścia podłączone jest urządzenie generujące pakiety,  
które co takt odpytywane jest o to, czy pojawił się już nowy pakiet. Analogicznie każde wyjście powiązane jest z jednym urządzeniem docelowym, które odbiera pakiety, aczkolwiek w danej chwili może ono być zajęte np. przeprowadzaniem obliczeń. Przełącznik w każdym takcie dokonuje odbierania pakietów na portach wejściowych, zestawiania połączeń na podstawie odpowiedzi algorytmu szeregującego, przesłania komórek pomiędzy połączonymi portami i przesłania, o ile jest  
w tym takcie możliwe, fragmentu pakietu do urządzenia docelowego.

Przełącznik utrzymuje dwa rodzaje list pakietów – listy pakietów, które zostały odrzucone oraz listy pakietów przesłanych od źródła do ujścia. Każda lista pakietów odrzuconych  
oraz przetworzonych reprezentuje port wyjściowy, do którego pakiet zmierzał.

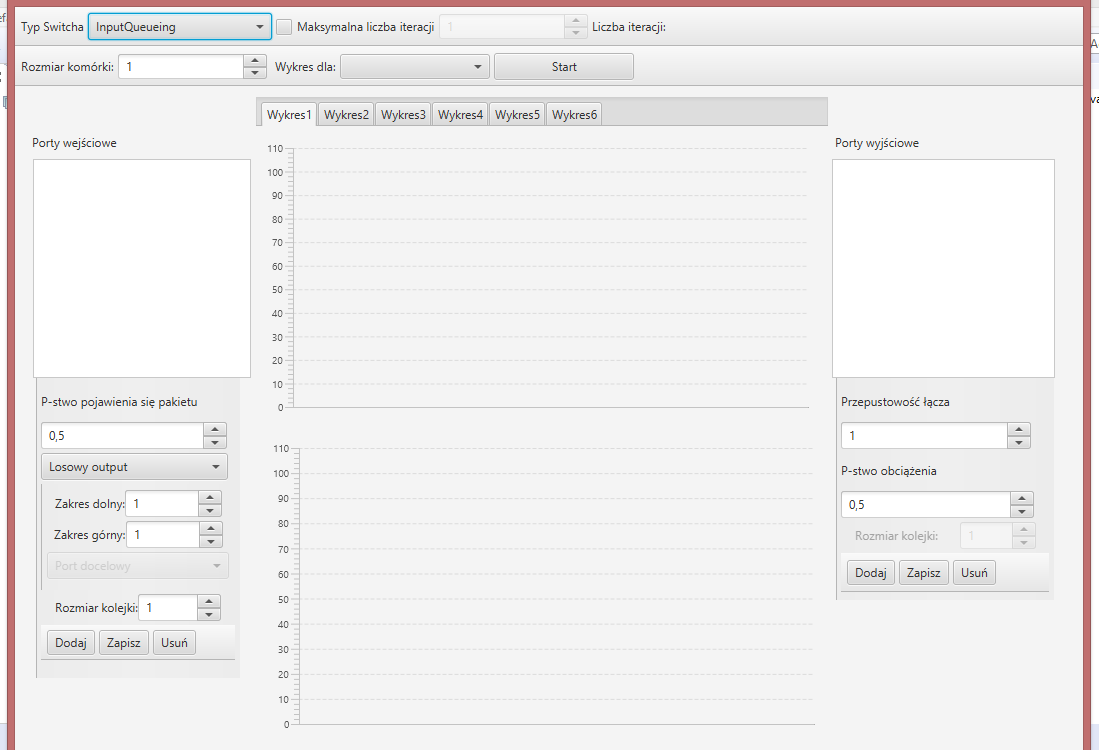
Każdy pakiet zapisane ma parametry – identyfikator portu wejściowego oraz wyjściowego, stemple czasowe odebrania pakietu oraz jego całkowitego przesłania do celu oraz rozmiar.  
Takie dane umożliwiają wyznaczenie kilku statystyk odnośnie pracy przełącznika.

Należy zwrócić uwagę, że implementacja jest bardzo elastyczna i w łatwy sposób umożliwia rozszerzanie symulatora o nowe rodzaje źródeł pakietów, ich ujść oraz algorytmów szeregowania.

# Symulator – interfejs użytkownika

Rys. 3 przedstawia wygląd powstałej aplikacji.

Interfejs składa się z czterech głównych sekcji:

* Górnego paska nawigacyjnego umożliwiającego ustawienie takich parametrów symulacji jak typ przełącznika, maksymalną liczbę iteracji oraz rozmiar komórki przesyłanej  
  w przełączniku. Dodatkowo na pasku wyświetlana jest aktualna liczba iteracji, umiejscowiony jest przycisk uruchamiający symulację oraz lista pozwalająca wybrać wyjście przełącznika dla którego wyświetlane są wykresy (o ile danych wykresów to dotyczy),
* Lewy panel pozwala dodawać, edytować i usuwać porty wejściowe przełącznika.  
  Istniejące porty wyświetlane są na liście. Każdy port może mieć ustalone takie parametry jak prawdopodobieństwo pojawienia się pakietu, wybór tego, czy port docelowy jest losowany, czy też jeden wybrany, ustalenie minimalnego i maksymalnego rozmiaru pakietu, a w przypadku przełączników z kolejkowaniem wejściowym oraz VOQ – ustalenie rozmiaru kolejki/kolejek,
* Prawy panel umożliwia utworzenie, edycję i usuwanie portów wyjściowych przełącznika.   
  Wyświetlana jest tu lista obecnie utworzonych portów wyjściowych. Parametrami portów są – przepustowość łącza, prawdopodobieństwo tego, że port będzie w danej chwili obciążony oraz rozmiar kolejki wyjściowej, jeżeli przełącznik obsługuje kolejkowanie wyjściowe,

Rysunek Interfejs symulatora

* Środkowa sekcja aplikacji zawiera zakładki z wykresami. Każda zakładka zawiera dwa wykresy – jeden prezentujący zmiany w perspektywie czasu i drugi – dane zagregowane.  
  Pierwsze trzy zakładki zawierają wykresy procentu odrzuceń pakietów ze względu na: rozmiar, port wejściowy oraz port wyjściowy. Kolejne trzy zakładki zawierają wykresy średniego opóźnienia pakietów (średniego czasu spędzanego przez pakiet  
  w przełączniku) ze względu na: rozmiar, port wejściowy oraz port wyjściowy.

# Badania

## Przełącznik z kolejkowaniem wejściowym

### Wzmożony ruch sieciowy

Pierwszym badaniem przeprowadzonym na tym typie przełącznika było sprawdzenie jego efektywności przy wzmożonym ruchu sieciowym składającym się z dużej liczby małych pakietów  
tj. pakietów rozmiarami porównywalnych z rozmiarem komórki przełącznika. Badanie polegało  
na czterokrotnym powtórzeniu tego samego doświadczenia. Przełącznik wyposażony został w cztery porty wejściowe oraz cztery porty wyjściowe. Prawdopodobieństwa pojawienia się pakietów były równe 75%. Za rozmiar komórki przyjęto 10, a pakiety losowano z przedziału <5, 10>. Rozmiar kolejek ustalono na 50. Każde wyjście wyposażono w łącze o przepustowości 100, a prawdopodobieństwo,  
że w danej chwili urządzenie wyjściowe jest zajęte wyniosło 25%. Każda symulacja trwała 200 iteracji.

Wyniki zebrano w tabelach 1-5.

Tabela Wyniki pierwszego powtórzenia eksperymentu

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| I | WEJŚCIA | | | | WYJŚCIA | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| Odrzucone pakiety [%] | 26,5 | 26,5 | 27,0 | 34,0 | 29,5 | 32,0 | 21,5 | 30,5 |
| Średnie opóźnienie | 11,5 | 11,5 | 11,0 | 12,5 | 11,4 | 11,8 | 11,3 | 11,4 |

Tabela Wyniki drugiego powtórzenia eksperymentu

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| II | WEJŚCIA | | | | WYJŚCIA | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| Odrzucone pakiety [%] | 31,0 | 28,0 | 32,0 | 32,0 | 30,0 | 29,5 | 32,5 | 32,0 |
| Średnie opóźnienie | 12,7 | 12,3 | 12,3 | 12,7 | 12,6 | 12,5 | 13,0 | 12,3 |

Tabela Wyniki trzeciego powtórzenia eksperymentu

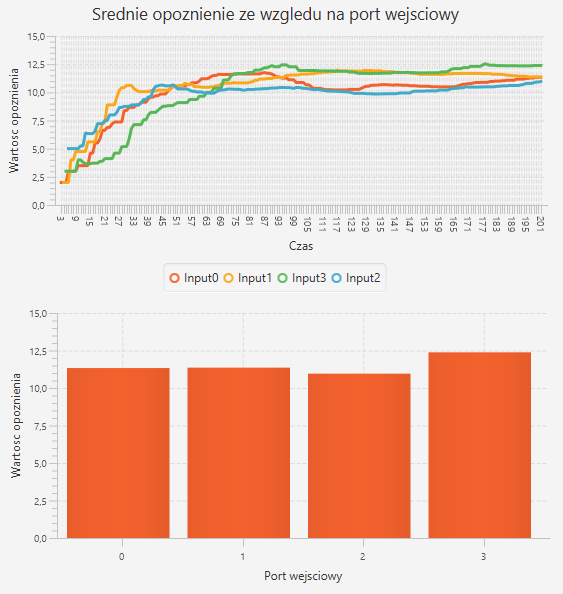
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| III | WEJŚCIA | | | | WYJŚCIA | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| Odrzucone pakiety [%] | 17,7 | 21,0 | 30,0 | 19,0 | 25,0 | 21,0 | 21,0 | 21,2 |
| Średnie opóźnienie | 9,9 | 10,5 | 11,5 | 9,5 | 10,2 | 10,4 | 10,1 | 10,6 |

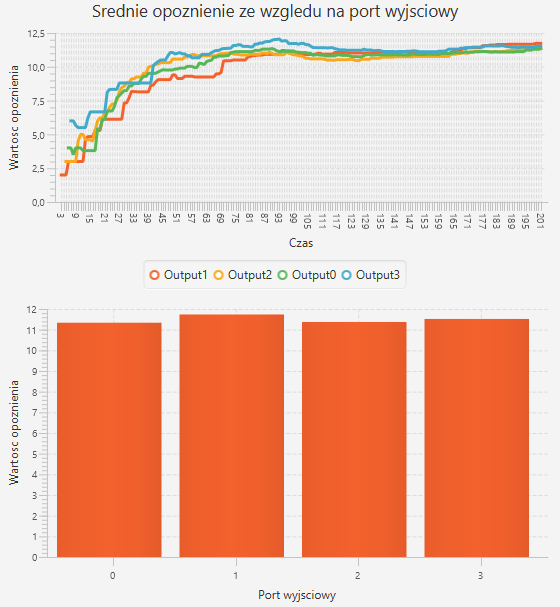
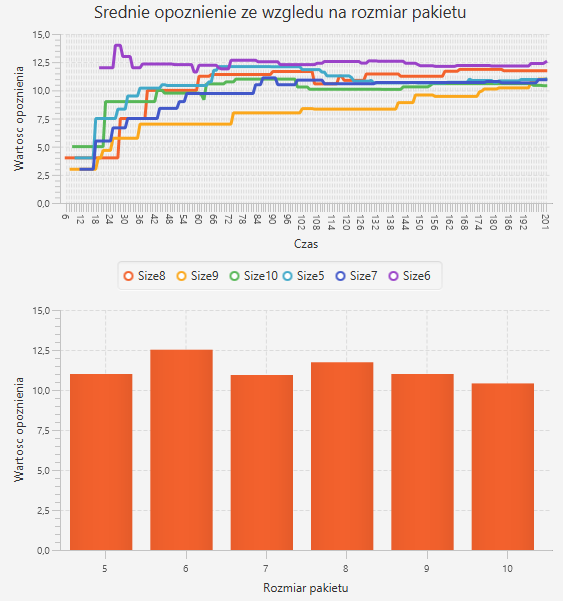
Tabela Wyniki czwartego powtórzenia eksperymentu

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| IV | WEJŚCIA | | | | WYJŚCIA | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| Odrzucone pakiety [%] | 26,0 | 18,0 | 21,5 | 31,0 | 26,0 | 22,3 | 25,5 | 23,5 |
| Średnie opóźnienie | 10,2 | 10,1 | 10,0 | 12,5 | 10,3 | 10,8 | 10,4 | 10,8 |

Tabela Średnie wyniki eksperymentu

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ŚREDNIA | WEJŚCIA | | | | WYJŚCIA | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| Odrzucone pakiety [%] | 25,3 | 23,4 | 27,6 | 29,0 | 27,6 | 26,2 | 25,1 | 26,8 |
| Średnie opóźnienie | 11,1 | 11,1 | 11,2 | 11,8 | 11,1 | 11,4 | 11,2 | 11,3 |

Poniższe rysunki zawierają wyniki symulacji uzyskane dla jednego z powtórzeń badania.



### 1_REJECT_OUT.png1_REJECT_IN.png

### 1_REJECT_SIZE.pngWnioski:

Dla przedstawionego przypadku przełącznik zachowuje się dosyć dobrze – odrzuca około 23 – 30 procent pakietów i wprowadza opóźnienie nieco mniej niż 12 cykli pracy. Uzyskane wyniki mogą mieć dwie przyczyny – pierwszą z nich mogą być rozmiary kolejek, które w naszym przypadku wyniosły jedynie 5 pakietów największego możliwego rozmiaru, zaś drugim powodem może być dosyć duże obciążenie urządzeń wyjściowych, bo wynoszące 25%. Oznacza to, że dane wyjście nie może transmitować średnio przez jedną czwartą czasu, a przez to wzrasta odpowiednio liczba pakietów odrzuconych oraz całkowite opóźnienie jakie switch wprowadza. Niemniej jednak wynik należy uznać za dobry, gdyż obciążenie sieci jest wysokie, a przełącznik – zalewany dużą ilością małych pakietów.

### Problem blokowania kolejek (head-of-line problem)

Dla tej architektury przełącznika notorycznie wskazywanym problemem jest blokowanie kolejek wejściowych przez duże pakiety stojące na ich czele. Niniejsze badanie miało na celu stwierdzenie,   
czy symulator pozwala na zaobserwowanie tego efektu.

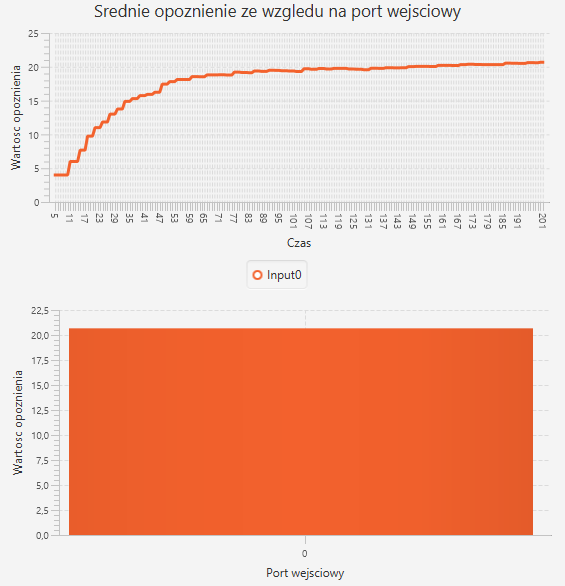
W obecnej postaci symulator nie umożliwia wygenerowania interesującego nas typu ruchu tzn. ruchu, w którym jedno wejście generuje duże pakiety wysyłane na pewne wyjścia, oraz małe pakiety wysyłane na inne wyjścia. Taki rodzaj ruchu, w przypadku występowania blokowania, spowodowałby powstanie wyraźnych opóźnień pakietów na wyjściach, do których skierowane były by małe pakiety.  
Oczywiście symulator jest rozszerzalny i można by przygotować odpowiednie źródło pakietów,  
ale w obecnej postaci można podejść do problemu inaczej. Badanie polegało na wysyłaniu pakietów  
z jednego portu wejściowego na dwa porty wyjściowe i sprawdzeniu, czy wraz z powiększaniem rozmiaru generowanych pakietów rośnie liczba odrzuceń i opóźnienie. Oczywiście wraz   
ze zwiększeniem rozmiaru pakietów należało proporcjonalnie zwiększać rozmiar kolejki, aby nie miała ona znacznego wpływu na liczbę odrzuceń – kolejka o stałym rozmiarze mieściła by coraz mniej pakietów, co zwiększyło by liczbę odrzuceń.

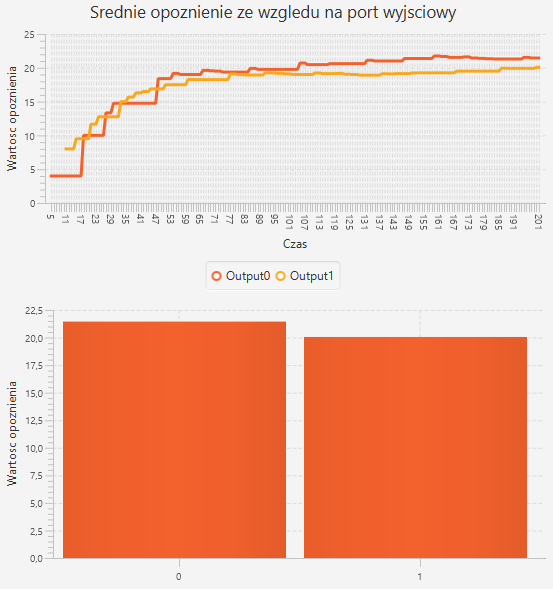
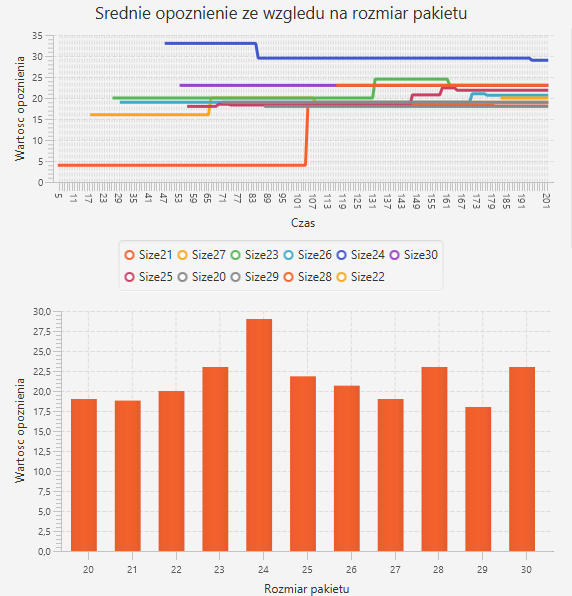
Tak samo jak w poprzednim doświadczeniu przyjęto: rozmiar komórki równy 10, prawdopodobieństwo pojawienia się pakietu na wejściu równe 75% oraz liczbę iteracji równą 200. Przepustowość łącz wyjściowych – 1000, a prawdopodobieństwo obciążenia – 1%.   
Rozmiary losowanych pakietów należały do przedziału <20, 30>, a rozmiar kolejki na 150, przy czym w kolejnych próbach końce przedziału były przesuwane o pewną stała wartość, a rozmiar kolejki   
– skalowany proporcjonalnie.

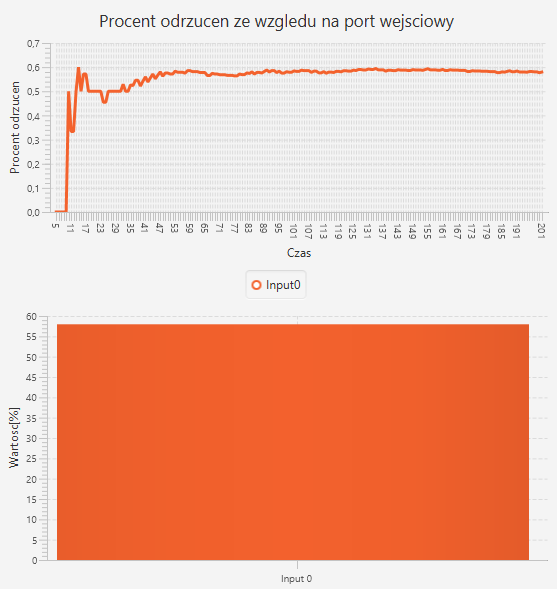
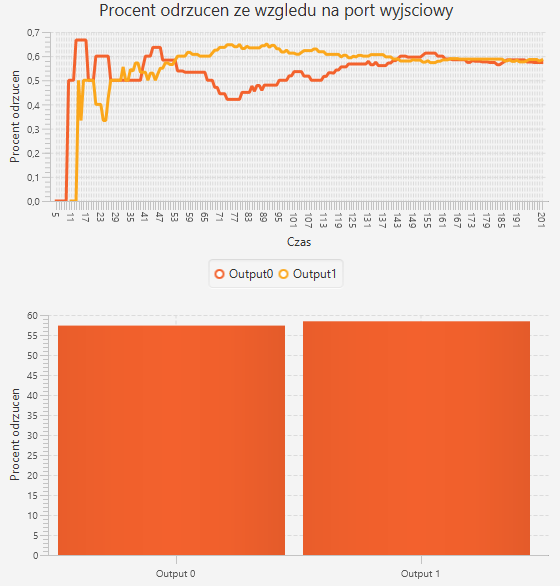
Doświadczenie przeprowadzono dla przedziałów <20, 30>, <40, 50>, <90, 100> oraz <190, 200>.  
Wyniki zebrano w tabeli 6.

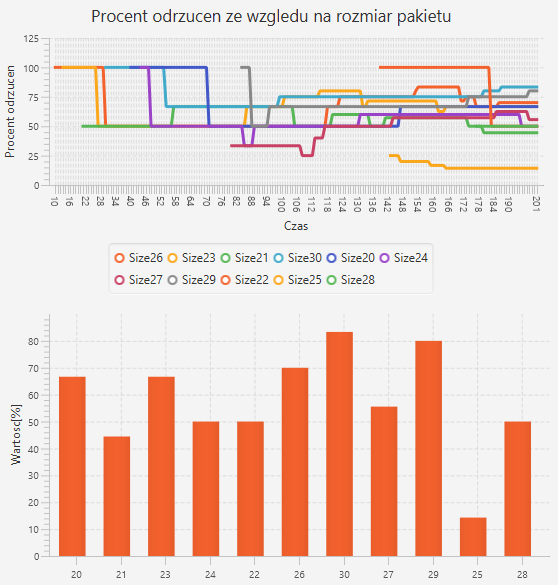
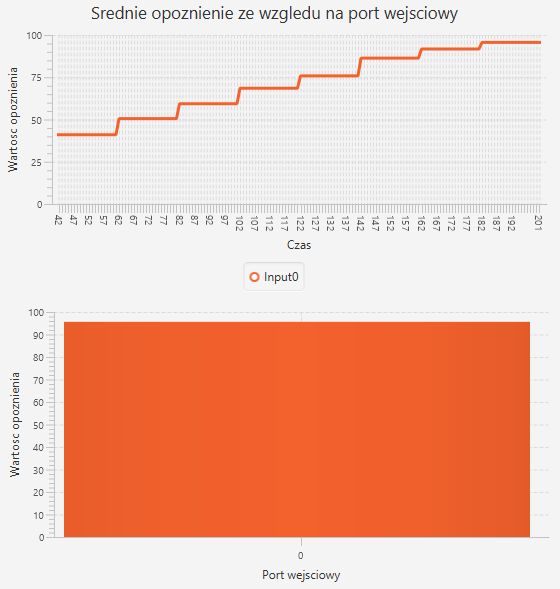
Tabela Wyniki badania efektu HOL

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | Przedział rozmiaru pakietów | | | |
| <20, 30> | <40, 50> | <90, 100> | <190, 200> |
| Odrzucone pakiety [%] | Wejście | 58,0 | 73,0 | 87,0 | 94,0 |
| Wyjście 1 | 57,0 | 76,0 | 84,0 | 93,0 |
| Wyjście 2 | 58,0 | 70,0 | 91,0 | 96,0 |
| Średnie opóźnienie | Wejście | 20,6 | 30,5 | 56,0 | 96,0 |
| Wyjście 1 | 21,5 | 30,5 | 51,0 | 93,0 |
| Wyjście 2 | 20,0 | 30,9 | 64,0 | 101,0 |

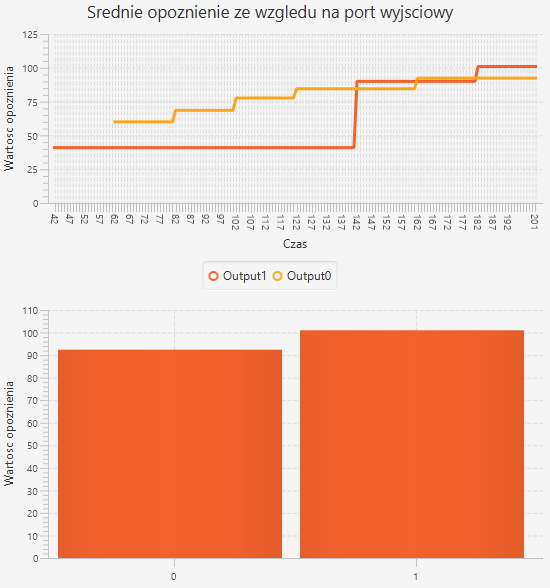
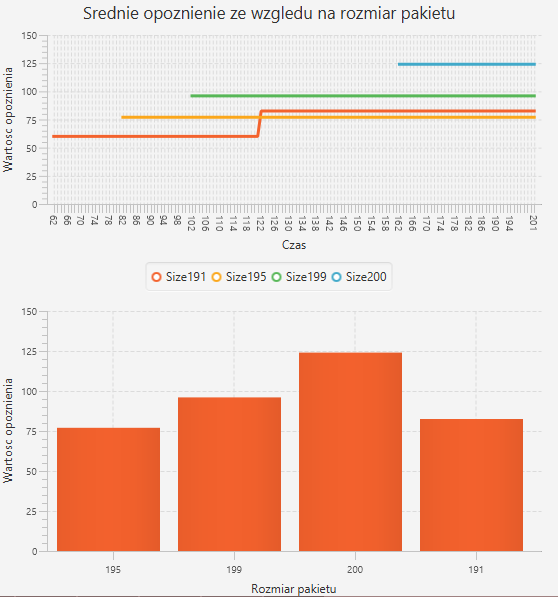
Poniższe rysunki zawierają zrzuty ekranu symulatora po zakończeniu symulacji <20, 30> oraz <190, 200>.

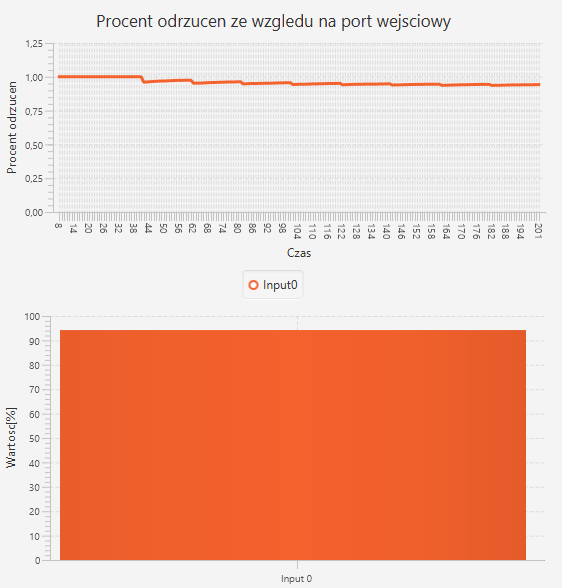
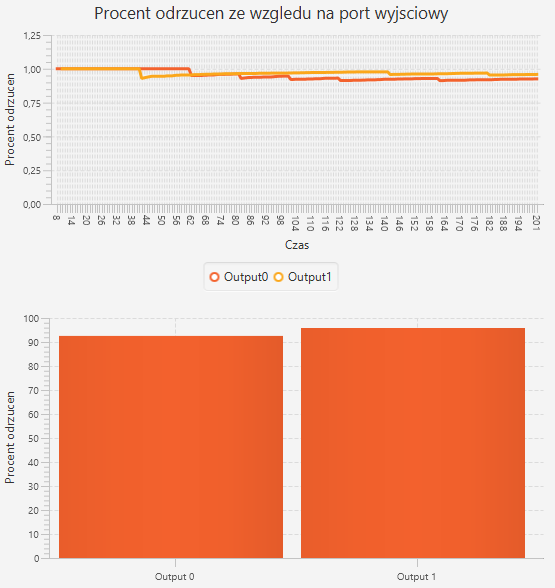






Rysunek Wyniki doświadczenia dla przedziału losowania <20, 30>





### 4_DELAY_SIZE.pngWnioski:

Rysunek Wyniki doświadczenia dla przedziału losowania <190, 200>

Zebrane wyniki symulacji w pełni potwierdzają występowanie problemu blokowania kolejek.

Zarówno opóźnienia występujące na switch’u, jak i liczba odrzucanych pakietów rośnie, wraz   
z rozmiarem pakietów, które do niego przychodzą pomimo proporcjonalnego zwiększania kolejki.  
Oczywiście na wyniki wpływ mają także inne parametry przełącznika np. rozmiar komórki, ale warto zwrócić uwagę na fakt, że zmiana rozmiaru pakietów poskutkowała bardzo silną zmianą w liczbie odrzuceń i opóźnieniu.

# Bibliografia

[1] Opalka Z., Aggarwal V., Kong T., Firth C., Costantino C., “*System architecture for and method of processing packets and/or cells in common switch*”, Malboro, 2001

[2] Trevitt S., Grant R., Book D., “*Low latency shared memory switch architecture*”, Broomfield, 2000

[3] Wang Q., Gopalakrishnan S., “*Adapting a Main-Stream Internet Switch Architecture for Multihop Real-Time Industrial Networks*”, IEEE, 2010

[4] Chao H., “*Saturn: A**Terabit Packet Switch Using Dual Round-Robin*”, IEEE, New York, 2000

[5] Stephens Z., Zhang H., “Implementing Distributed Packet Fair Queueing in a Scalable Switch Architecture”, IEEE, 1998