POZNAN UNIVERSITY OF TECHNOLOGY ACADEMIC JOURNALS
No 91 Electrical Engineering 2017

DOI 10.21008/j.1897-0737.2017.91.0018

Janusz KLEBAN*
Jarosław WARCZYŃSKI**

BADANIE ALGORYTMÓW STEROWANIA PAKIETOWYMI POLAMI KOMUTACYJNYMI

Praca dotyczy badania własności algorytmów sterowania przepływem komórek w wielosekcyjnych polach komutacyjnych, stanowiących jądro pakietowych węzłów komutacyjnych, którymi są routery klasy operatorskiej. Ze względu na możliwość występowania konfliktów w dostępie do zasobów pola, konieczne jest stosowanie algorytmów sterowania przepływem decydujących o tym, które komórki z portów wejściowych zostaną przesłane do portów wyjściowych. Własności algorytmów tego typu bada się na drodze symulacji, określając przede wszystkim opóźnienie komórek, długości kolejek oraz przepustowość pola komutacyjnego. W pracy przedstawiono budowę przestrajalnego symulatora pól komutacyjnych, pozwalającego na przeprowadzenie badań wspomnianych wyżej własności algorytmów sterowania polem.

SŁOWA KLUCZOWE: pole komutacyjne Closa, planowanie przepływu pakietów, komutacja pakietów, symulacja, algorytmy sterowania

1. WSTĘP

Dążenie do zbudowania *Internetu Przyszłości* (*Future Internet*) opartego na czterech filarach, do których zalicza się *Internet Społecznościowy* (*Internet by and for People*), *Internet Informacji i Wiedzy* (*Internet of Contents and Knowledge*), *Internet Rzeczy* (*Internet of Things*) i *Internet Usług (Internet of Services*) wymaga przygotowania infrastruktury sprzętowej zdolnej do przesyłania i przetwarzania bilionów pakietów w ciągu sekundy [11]. O ile optyczne sieci transportowe wykorzystujące zwielokrotnienie w długości fali WDM (*Wavelength Division Multiplexing*) doskonale radzą sobie z bardzo szybką transmisją danych, o tyle pakietowe węzły sieciowe (routery/przełączniki) stanowią wąskie gardło. Jest to spowodowane wieloma czynnikami; między innymi koniecznością przetwarzania nagłówków, a także zamianą sygnałów optycznych na elektryczne. Obecnie poszukuje się nowych rozwiązań przyspieszających obsługę pakietów, w szczególności dla sprzętu klasy operatorskiej, pracu-

* Politechnika Poznańska.

jącego w sieciach szkieletowych. Jednym z ważniejszych modułów routerów i przełączników sieciowych mających wpływ na ich wydajność jest pole komutacyjne, które dokonuje przekierowania pakietów z dowolnego wejścia do dowolnego wyjścia.

W routerach i przełącznikach średniej wielkości stosuje się pola typu *crossbar*, nazywane również jednosekcyjnymi polami komutacyjnymi. Składają się one z matrycy $N \times N$ punktów komutacyjnych, gdzie N jest to liczba wejść i wyjść pola. W przypadku zestawiania połączenia pomiędzy wejściem x i wyjściem y algorytm sterujący musi zmienić stan punktu komutacyjnego, położonego na przecięciu wiersza odpowiadającego wejściu i kolumny odpowiadającej wyjściu, z krzyżowego na równoległy. Sterowanie polem tego typu jest zatem bardzo proste, ale niestety struktura ta jest nieskalowana i nie jest stosowana przy dużej liczbie wejść i wyjść. Przykładowo już dla $N > 24$ można pokazać, że pole trzysekcyjne ma mniej punktów komutacyjnych.

W routerach klasy operatorskiej stosuje się pola wielosekcyjne, które mają budowę modułową i są skalowalne [1]. Takie pola wymagają jednak bardziej złożonego sterowania, które jest odpowiedzialne, z jednej strony, za wybór pakietów, które zostaną przesłane z wejść do wyjść pola, a z drugiej za znalezienie, dla każdego pakietu, drogi połączeniowej od wejścia, przez moduły wszystkich sekcji, aż do wyjścia. W sprzęcie tego typu zazwyczaj stosuje się pola Closa [2], np. routery CRS 3 3 firmy Cisco, T1600 firmy Juniper Networks, lub TX Matrix. Pola komutacyjne o dużej pojemności pracują w sposób synchroniczny i przesyłają, w szczelinach czasowych, pakiety o stałej długości zwane komórkami. Pakiety o zmiennej długości np., pakiety IP, napływające do węzła sieciowego są dzielone, w kartach liniowych, na segmenty o stałej długości, które następnie są pakowane do komórek i przesyłane przez pole komutacyjne do portów wyjściowych. Przed opuszczeniem węzła komutacyjnego pakiety są z powrotem składane w całość, odzyskując swoją pierwotną wielkość.

Wśród różnych wariantów pól Closa proponowanych dla routerów/przełączników dość intensywnie bada się pole typu MSM (*Memory Space Memory)* – z buforami w pierwszej i trzeciej sekcji. Dla struktury tej zaproponowano różne algorytmy sterowania zwane również algorytmami planowania przepływu pakietów [4, 5, 9, 13]. Z punktu widzenia oceny wydajności pola, w którym stosuje się dany algorytm sterowania, dokonuje się oceny takich parametrów jak: średnie opóźnienie komórek na wyjściu systemu, również za każdą sekcją, maksymalne i średnie długości kolejek komórek oczekujących na przesłanie, a także przepustowość pola [1].

Ze względu na brak metod analitycznych pozwalających na ocenę parametrów jakościowych pól komutacyjnych korzysta się z symulacji komputerowej. Symulacyjne badanie modeli systemów telekomunikacyjnych nabiera coraz większego znaczenia ze względu na dużą moc obliczeniową obecnych komputerów. Można w ten sposób analizować dowolne zależności występujące

w sprzęcie rzeczywistym, oczywiście o ile zostanie prawidłowo przygotowany model symulacyjny. Algorytmy sterowania pakietowymi polami komutacyjnymi są najczęściej analizowane z wykorzystaniem programów napisanych przez autorów publikacji. W zależności od badanych problemów sieciowych czasami istnieje możliwość wykorzystywania gotowych środowisk jak np.: NS 3 [8] Riverbed Modeler (dawniej OPNET) [12], NetSim [7], OMNeT++ [10]. Niestety, ze względu na to, że symulatory tego typu są często ukierunkowane na badanie mechanizmów sieciowych, implementacja algorytmów sterowania polami komutacyjnymi jest w nich ograniczona.

W dalszej części artykułu przedstawiono możliwość wykorzystania do badania własności algorytmów sterowania polami komutacyjnymi pakietu matematycznego *Matlab*, z zastosowaniem biblioteki, *SimEvents*, przeznaczonej specjalnie do modelowania systemów zdarzeń dyskretnych. W szczególności pokazana zostanie konstrukcja symulatora pola Closa typu MSM wykonanego w tym środowisku modelowania.

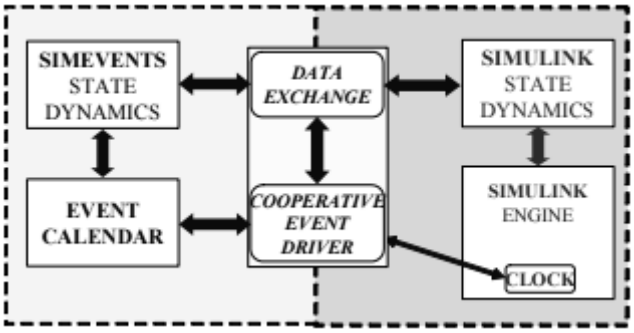
2. BIBLIOTEKA *SIMEVENTS*

Biblioteka *SimEvents* jest elementem opcjonalnym środowiska symulacyjnego *Simulink* pakietu matematycznego *Matlab* firmy *MathWorks*. Rozszerza ona funkcjonalność tego środowiska, umożliwiając modelowanie i symulację systemów zdarzeń dyskretnych, do badania których stosuje się metodologię systemów masowej obsługi lub teorii kolejek, i które generalnie nie są zależne od czasu, lecz raczej od występujących w systemie różnorodnych zdarzeń dyskretnych. Regułą działania takich systemów jest przepływ obiektów (nazywanych w bibliotece *entities*) przez sieci serwerów, bram, przełączników, świadczących określone usługi na rzecz obiektów. Można w ten sposób modelować m.in. funkcjonowanie sieci komputerowych z komutacją pakietów, zrządzanie procesami w systemach komputerowych, dyskretne systemy wytwórcze, marszruty technologiczne, systemy logistyczne, i wiele innych. Dodajmy, że oprogramowanie to uzupełnia silnik symulacji *Simulinka* o część sterowaną zdarzeniami, która doskonale współdziała z silnikiem symulacji taktowanym upływem czasu. Pozwala to tworzyć modele hybrydowe, łączące obie zasady symulacji. Architektura tej kooperatywnej funkcjonalności *Simulinka* została pokazana na rys. 1 [3, 6].

Obiekty (*entities*) przepływające przez modelowaną sieć mogą przenosić dane, które są zawarte w atrybutach tych obiektów.

Do najważniejszych bloków funkcjonalnych biblioteki *SimEvents* należy zaliczyć [6]:

1. ***Generators**:* Bloki, które generują obiekty lub wywołania funkcji (tj. zdarzenia, które wywołują bloki funkcyjne *Simulinka*) o różnych charakterystykach losowych.
2. ***Queues**:* Bloki, w których obiekty mogą być czasowo przechowywane w oczekiwaniu na obsługę lub dostęp do zasobów.
3. ***Servers**:* Bloki modelujące różnorodne zasoby.
4. ***Routing**:* Bloki, które kierują ruchem obiektów pomiędzy zasobami i kolejkami.
5. ***Gates**:* Bloki, które sterują przepływem obiektów zatrzymując je lub umożliwiając ich dostęp do innych bloków.
6. ***Event Translation**:* Bloki umożliwiające komunikację pomiędzy biblioteką *SimEvents* i *Simulinkiem*, dzięki przekształcaniu zdarzeń na wywołania funkcji.
7. ***Attributes:*** Bloki, które przypisują dane do obiektów i je ewentualnie modyfikują. W oparciu o wartości tych danych można różnicować sposób przetwarzania obiektów w różnych blokach.



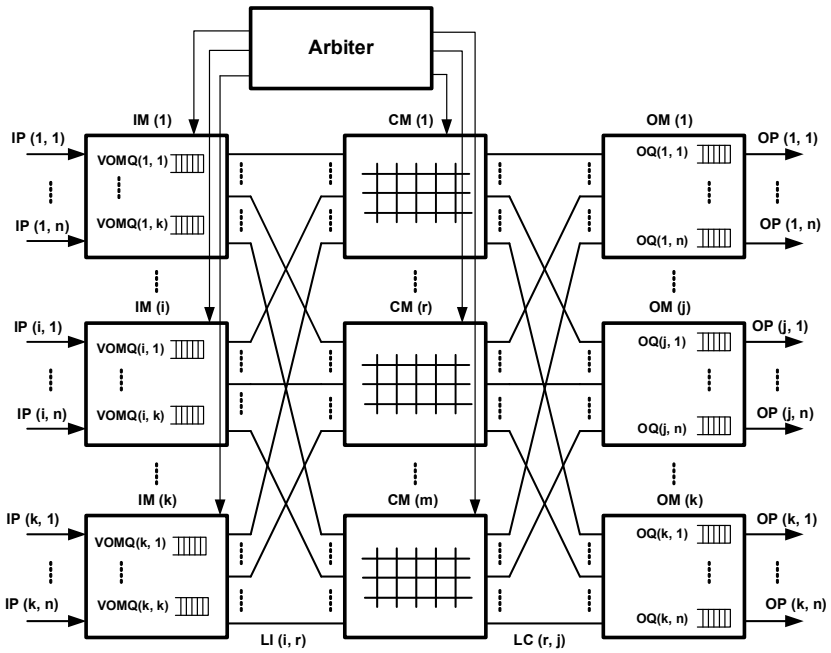
Rys. 1. Współdziałanie silników symulacji *SimEvents* i *Simulinka* [3]

Ważnym aspektem tej biblioteki są zagregowane w niej elementy statystyki, co ułatwia wyznaczenie istotnych statystycznych wskaźników badanych systemów, takich jak np. opóźnienie, przepustowość, średnie długości kolejek i wiele innych.

3. POLE CLOSA TYPU MSM

Trzysekcyjne pole komutacyjne Closa typu MSM o wymiarach $N \times N$, z kolejkami do modułów wyjściowych VOMQ (*Virtual Output Module Queues*) oraz z arbitrem centralnym przedstawiono na rys. 2. Pierwsza sekcja pola składa się z k komutatorów wejściowych IM (*Input Module*) z pamięciami współdzielonymi; porty wejściowe oznaczono jako IP (*Input Port*). Pamięci współdzielo-

ne są podzielone na k kolejek, z których każda przechowuje komórki do innego modułu wyjściowego. Taka organizacja buforów pozwala na uniknięcie zjawiska blokowania początku kolejki – HoL (*Head–Of–Line blocking*). Druga sekcja zawiera m komutatorów bez buforów, oznaczonych jako CM (*Central Module*). Z kolei w trzeciej sekcji znajduje się k komutatorów wyjściowych OM (*Output Module*). Porty wyjściowe modułów OM oznaczono jako OP (*Output Port*). Każdy port OP posiada bufor OQ (*Output Queue*).



Rys. 2. Pole Closa typu MSM z arbitrem centralnym

Na rys. 2 przyjęto następujące oznaczenia: IM(i) – i–ty komutator pierwszej sekcji, gdzie $1 \le i \le k$; CM(r) – r–ty komutator drugiej sekcji, gdzie $1 \le r \le m$; OM(j) – j–ty komutator trzeciej sekcji, gdzie $1 \le j \le k$; n – liczba we/wy w każdym IM/OM; k – liczba komutatorów IM/OM; m – liczba komutatorów CM; IP(i, h) – h–ty port wejściowy w IM(i), $1 \le h \le n$; OP(j, h) – h–ty port wyjściowy w OM(j), $1 \le h \le n$; OQ(j, h) kolejka wyjściowa w OM(j); VOMQ(i, j) – wirtualna kolejka wyjściowa w IM(i), przechowuje komórki przeznaczone dla OM(j); LI(i, r) – łącze międzysekcyjne między komutatorem IM(i) oraz CM(r); LC(r, j) – łącze międzysekcyjne między komutatorem CM(r) oraz OM(j).

Arbiter centralny przedstawiony na rys. 2 realizuje algorytm sterowania polem i jest odpowiedzialny za wybór komórek do przesłania z wejść do wyjść oraz za rozwiązywanie konfliktów w dostępie do zasobów wewnętrznych pola.

4. SYMULATOR POLA KOMUTACYJNEGO

Poniżej przedstawiono modele podstawowych elementów symulatora pakietowego pola komutacyjnego Closa typu MSM. Symulator modeluje pole pracujące synchronicznie w tzw. szczelinach czasowych.

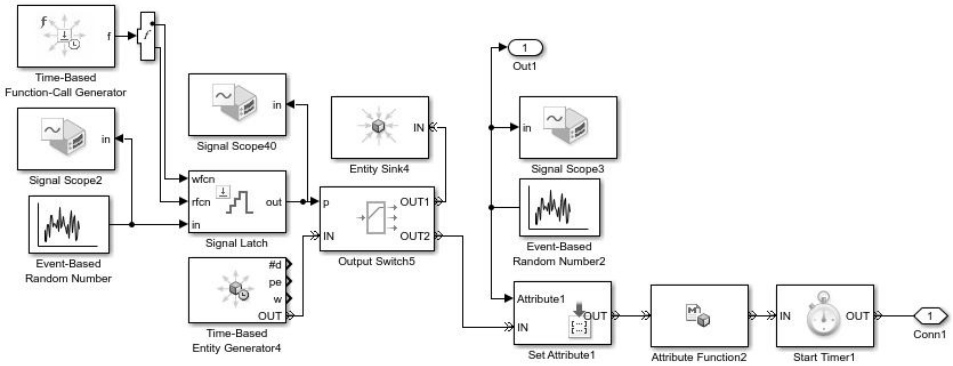
4.1. Port wejściowy

Podstawowym celem modelu portu jest generowanie procesu przybywania komórek w dyskretnych chwilach czasu Δt. Powszechnie uważa się, że najlepiej do modelowania takiego ruchu nadaje się model Bernoulliego, który generuje strumień bezpamięciowy. W szczelinie czasu Δt jedno zdarzenie pojawia się z prawdopodobieństwem p, a wystąpienie kilku zdarzeń na raz jest niemożliwe. Z wykorzystaniem tego modelu. można określić prawdopodobieństwo skierowania do węzła komutacyjnego, w czasie jednej szczeliny czasowej, n komorek: $\lambda = n \cdot p$.

Zatem, na wejście portu mają przybywać komórki z rozkładem Bernoulliego o zadanym prawdopodobieństwie p. Zakłada się, że na wejście portu w jednej szczelinie czasowej może przybyć co najwyżej jedna komórka (jest to założenie tzw. ruchu dopuszczalnego), co oznacza, że np. dla $p = 0,7$ należy oczekiwać przybycia średnio 7 komórek w trakcie każdych 10 szczelin czasowych. Zakładamy, że komórki, modelowane jako obiekty, będą miały, podobnie jak pakiety, jako atrybuty: adres portu docelowego i zależny od niego adres wyjściowego modułu docelowego.

Symulator portu wejściowego został przedstawiony na rys. 3. Generuje on obiekty z natężeniem λ, dzięki wykorzystaniu bloku *Event_Based Random Number*, który losuje dwie liczby 2 i 1, zgodnie z zadanym prawdopodobieństwem, odpowiednio p oraz $(1–p)$. Liczby te sterują, w danej szczelinie czasowej, otwarciem jednego z dwóch wyjść bloku *Output Switch5*, który przenosi obiekty ze swojego wejścia, gdzie napływają one w każdej szczelinie czasowej z generatora *Time–based Entity Generator4*. Wyjście 1 prowadzi do bloku *Entity Sink4*, gdzie obiekty są pochłaniane i nie biorą udziału w dalszej symulacji. W tej sytuacji, obiekty, na wyjściu 2 pojawiają się z natężeniem $\lambda = n\,p$ i reprezentują komórki pojawiające się na wejściu pola komutacyjnego (*Conn1*). Pozostałe bloki odpowiadają za przypisanie komórkom adresów portów przeznaczenia (mogą być obserwowane przez wyjście *Out1*), co odbywa się zgodnie z zadanym rozkładem prawdopodobieństwa. Wykorzystuje się tutaj rozkłady będące wzorcami typowych ruchów w sieciach, takich jak m.in. ruch równomierny, diagonalny, Changa. W ramach konstruowanego symulatora rodzaj dystrybucji komórek do wyjść będzie można swobodnie ustalić. Ponadto, każda komórka otrzymuje jeszcze stempel czasowy, który pozwoli ustalić jej czas

przejścia przez pole, gdy dotrze ona do portu wyjściowego pola. W ten sposób możliwe będzie badanie czasu opóźnienia komórek w zależności od zastosowanego algorytmu sterowania polem.



Rys. 3. Model portu wejściowego w symulatorze pola komutacyjnego

4.2. Moduł wejściowy IM

W module wejściowym IM odbywa się sortowanie komórek przybywających do poszczególnych portów wejściowych i kierowanie ich do buforów powiązanych z wyjściowymi portami przeznaczenia, tj. do właściwych kolejek VOMQ. W modelu modułu wejściowego sortowanie to odbywa się na podstawie atrybutu obiektu modelującego komórkę, jakim jest adres portu przeznaczenia, z którego wyznaczany jest numer kolejki VOMQ. Model układu sortującego pokazany został na rys. 4.

4.3. Arbiter centralny

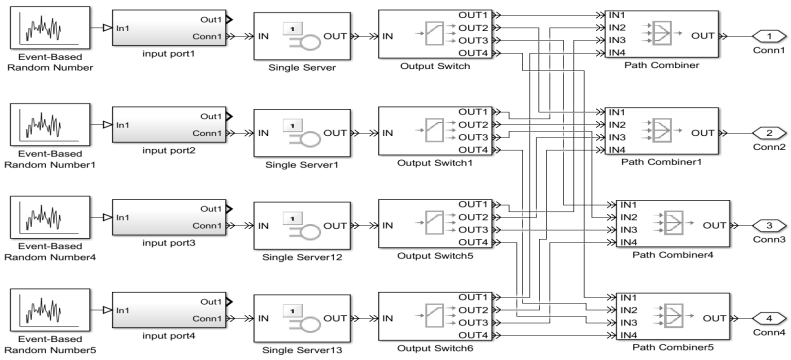
Rolą arbitra centralnego jest parowanie modułów wejściowych i wyjściowych według zadanych kryteriów. Algorytm wyznaczania tego dopasowania jest faktycznie algorytmem sterowania polem komutacyjnym.

W opisywanym symulatorze algorytm ten jest zrealizowany w postaci bloku funkcyjnego, którego funkcja może być ustalana przez użytkownika zgodnie z algorytmem podlegającym badaniu. Zmiana algorytmu nie wpływa na postać modeli poszczególnych sekcji pola komutacyjnego.

Wspomniane dopasowanie modułów oznacza, że ustalane są kolejki VOMQ, w poszczególnych modułach, z których w najbliższej szczelinie czasowej nastąpi przesłanie po n komórek do sparowanych modułów wyjściowych.

Przykładowy algorytm, użyty w prezentacji działania symulatora zakłada, że do kolejnych modułów wyjściowych przypisywane są te kolejki VOMQ

z poszczególnych modułów wejściowych, w których zgromadzonych jest najwięcej komórek, czyli wg. malejącej maksymalnej liczby zgromadzonych komórek– można go nazwać *Maximal OM Queue Length*.



Rys. 4. Układ sortowania komórek w modułach wejściowych IM

W dopasowaniu tym muszą być pomijane kolejki VOMQ do modułów już sparowanych, gdyż nie można z jednego modułu wejściowego IM wysłać komórek do różnych modułów wyjściowych OM w tej samej szczelinie czasowej.

Moduł wyjściowy, od którego rozpoczyna się dopasowywanie, jest cyklicznie zmieniany w trybie *Round Robin*. Każdy moduł IM realizuje przesyłanie komórek do modułów OM zgodnie z określonymi wzorami połączeń w modułach CM, co oznacza, że jest to realizowane sprzętowo i algorytm sterowania polem nie wyznacza dróg połączeniowych.

Algorytm sterowania polem:

Inicjalizacja: Ustaw wartość wskaźnika *g,* modułu wyjściowego OM, od którego rozpocznie się dopasowywanie modułów wejściowych IM, na wartość *g = 1*.

Krok 1: Nadaj wskaźnikowi *f* bieżącego modułu wyjściowego wartość *f = g*.

Krok 2: W każdym module IM, przygotuj listę stanów kolejek VOMQ i prześlij ją do arbitra centralnego.

Krok 3: Na podstawie otrzymanych list, arbiter centralny szuka, dla modułu wyjściowego OM(*f*), takiego modułu wejściowego IM*(l)* na którego liście, na pozycji *f,* jest liczba maksymalna (większa od 0) spośród wszystkich dostępnych list. W ten sposób kolejka VOMQ*(l, f)*, zostaje przypisana, w danej szczelinie czasowej, do modułu wyjściowego OM*(f)*.

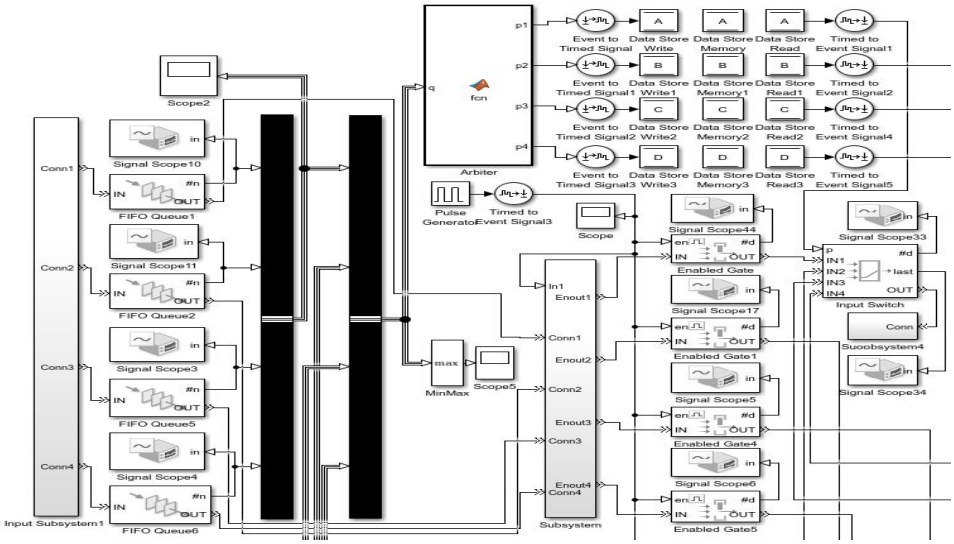
Krok 4: Usuń listę *l* ze zbioru list przesłanych do arbitra centralnego i przesuń wskaźnik bieżącego modułu wyjściowego *f* na następny moduł w cyklu.

Krok 5: Jeśli przypisanie zostało powtórzone *k* (liczba modułów OM) razy przesuń wskaźnik *g* na następny moduł OM w cyklu i przejdź do kroku 5, w przeciwnym przypadku wróć do kroku 2.

Krok 6: Każdy wybrany przez arbitra centralnego moduł IM(*l*) wyczytuje komórki z wybranej w kroku 2 kolejki VOMQ*(l, f)* oraz przesyła je do wyjść modułu IM*(l)*.

Krok 7: W następnej szczelinie czasowej następuje przesłanie *n* komórek do powiązanego komutatora OM*(f),* korzystając ze wszystkich wyjść komutatora IM*(l)*.

Szczupłość miejsca nie pozwala na pokazanie całego symulatora, natomiast na rys. 5. przedstawiono w zarysie jedną jego warstwę, tzn. jeden moduł wejściowy, gdzie widać 4 kolejki VOMQ, arbiter centralny oraz moduł wyjściowy OM, w którym znajdują się kolejki wyjściowe, gdzie komórki oczekują na przekazanie do portów wyjściowych



Rys. 5. Pojedyncza warstwa symulatora z arbitrem centralnym

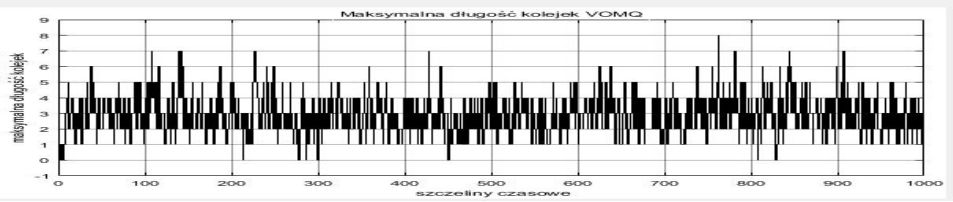
5. WYNIKI SYMULACJI

Poniżej przedstawiono przykładowe wyniki symulacji, wykonanej dla algorytmu *Maximal OM Queue Length.* Badaniu podlegały maksymalne długości kolejek wirtualnych VOMQ w modułach wejściowych oraz średnie opóźnienie komórek, obserwowane w portach wyjściowych modułów wyjściowych OM.

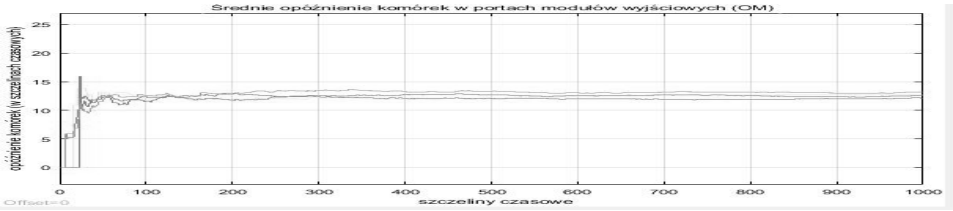
Eksperymenty przeprowadzone zostały dla pola komutacyjnego Closa typu MSM o wymiarach 16 × 16, czyli 4 moduły IM oraz OM. Przyjęto równomierny rozkład adresów docelowych komórek.

Wykresy otrzymane w wyniku symulacji, przedstawione na rys. 6–9, pokazują, iż maksymalne długości kolejek zależą od obciążenia pola, tzn. od natężenia przybywania komórek do portów wejściowych: Dla prawdopodobieństwa $p = 0{,}5$ w rozkładzie Bernoulliego maksymalne obserwowane długości kolejek nie przekraczają 8 komórek, a dla $p = 0{,}7$ szczytowe wartości dochodzą do 13 komórek.

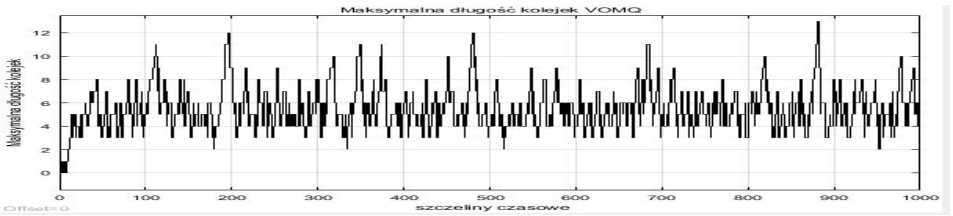
Opóźnienia komórek w portach wyjściowych również zależą od obciążenia pola i odpowiednio dla $p = 0{,}5$ nie przekraczają 15 szczelin czasowych a dla $p = 0{,}7$ nie przekraczają 20 szczelin czasowych, przy czym maksymalne wartości obserwuje się tutaj na początku procesu symulacji a późniejsze opóźnienia są w otoczeniu 15 szczelin czasowych. Dodajmy, że maksymalne długości kolejek w modułach wyjściowych OM (nie pokazane na wykresach) nie przekraczały w obu przypadkach 2 komórek.



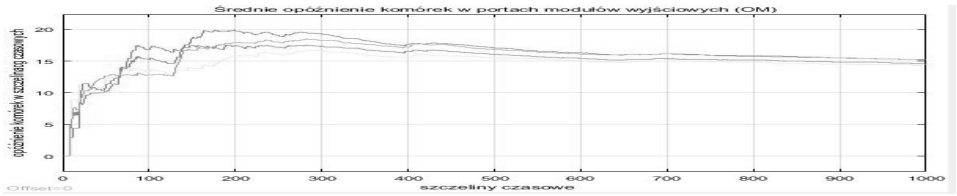
Rys. 6. Maksymalna długość kolejek VOMQ dla $p = 0{,}5$



Rys. 7. Średnie opóźnienie komórek w portach modułów wyjściowych dla $p = 0{,}5$



Rys.8. Maksymalna długość kolejek VOMQ dla $p = 0{,}7$



Rys. 9. Średnie opóźnienie komórek w portach modułów wyjściowych dla $p = 0{,}7$

6. PODSUMOWANIE

W ramach pracy przedstawiono konstrukcję przestrajalnego symulatora pól komutacyjnych Closa typu MSM. Pozwala on na badanie algorytmów sterowania polem w kontekście takich ważnych jego cech jak stabilność, mierzona rozrostem długości kolejek VOMQ w modułach wejściowych, czas opóźnienia komórek w portach wyjściowych pola oraz przepustowość pola. Załączone zostały wyniki przykładowej symulacji dla omówionego w pracy algorytmu *Maximal OM Queue Length*.

Symulator został wykonany w środowisku *Simulink* z wykorzystaniem biblioteki *SimEvents*, która okazała się wygodnym narzędziem programistycznym, w konstrukcji stosunkowo skomplikowanych modeli zdarzeń dyskretnych. Możliwość enkapsulacji modeli w podsystemy sprzyja skalowalności i pozwoli na budowę modelu pola o większych rozmiarach.

Projekt został sfinansowany z dotacji Ministerstwa Nauki i Szkolnictwa Wyższego na rok 2016, granty 08/82/DSPB/8214 i 04/45DSPB/0148.

LITERATURA

[1] Chao H.J., Liu B., High Performance Switches and Routers, Wiley–Interscience, New Jersey, 2007.
[2] Clos C., A Study of Non–Blocking Switching Networks, Bell Sys. Tech. Jour., 1953, pp. 406–424.
[3] Clune M.I., Mosterman P.J., Cassandras C.G, Discrete Event and Hybrid System Simulation with SimEvents, Proc. of 2nd IFAC Conference on Analysis and Design of Hybrid Systems, pp. 136–141, June 2006.
[4] Kleban J., Warczyński J.: Stabilność trzysekcyjnego pola Closa typu MSM z algorytmem MMLM, Przegląd Telekomunikacyjny i Wiadomości Telekomunikacyjne, nr 8–9, 2015, str. 754–761.
[5] Kleban J., Warczyński J., Badanie stabilności algorytmu sterownia polem Closa typu MSM, Poznan University of Technology Academic Journals: Electrical Engineering, no 87, pp. 353 – 364, 2016.
[6] MathWorks, SimEvents Documentation 2015b, The MathWorks, Inc.
[7] NetSim, http://tetcos.com.

[8] NS–3 Network Simulator, http://www.nsnam. org/.
[9] Oki E., Jing Z., Rojas–Cessa R., Chao H.J., Concurrent Round–Robin–Based Dispatching Schemes for Clos–Network Switches, IEEE/ACM Trans. on Networking, vol. 10, no.6, 2002, pp. 830–844.
[10] OMNeT++, https://omnetpp.org/.
[11] Papadimitriou D. (editor), Future Internet, The Cross–ETP Vision Document. http://www.future–internet.eu/fileadmin/documents/reports/Cross–ETPs_FI_Vi–sion Document_v1_0.pdf.
[12] Riverbed Modeler, https/www.riverbed.com/pl/products/steelcentral/steelcentral –riverbed–modeler.html.
[13] Xia Y., Chao H.J., Module–Level Matching Algorithms for MSM Clos–Network Switches, Proc. IEEE 13th International Conference on High Performance Switching Routing and 2012, pp. 36–43.

INVESTIGATION OF CONTROL ALGORITHMS FOR PACKET SWITCHING NETWORKS

In this paper, the research on packet dispatching schemes for multistage switching networks is discussed. These kinds of networks are used in high performance packet switching nodes, such as high–end routers. While a packet is being routed in a switching network it can face a contention problem resulting from two or more cells competing for a single resource. To avoid packet contention, it is necessary to use packet dispatching algorithms. These control algorithms decide which cells at input buffers will be transferred to outputs. The performance parameters, such as: average packet delay, queue length and throughput, of the switching network under the particular control algorithm, are investigated using computer simulation. The paper presents a tunable simulator of the MSM (Memory–Space–Memory) Clos–network switch. This simulator can be used for investigation of performance parameters under any implemented control algorithm.

(Received: 04. 02. 2017, revised: 27. 02. 2017)