STUDIA INFORMATICA 2006
Volume 27 Number 4 (69)

Piotr PECKA
Instytut Informatyki Teoretycznej i Stosowanej PAN

SYMULACJA KOLEJKI Z NEGATYWNYMI KLIENTAMI I WIELOMA SERWERAMI [1]

Streszczenie. Artykuł przedstawia model symulacyjny systemu MMCPP/GE/c-/LG z negatywnymi klientami, skończonym buforem i wieloma stanowiskami obsługi. Opisana została budowa symulatora, ze szczególnym uwzględnieniem rozwiązań koniecznych do wyznaczania charakterystyk stanu nieustalonego. Wyznaczone zostały przykładowe charakterystyki modelu, takie jak: rozkład czasu do pierwszego przepełnienia bufora i rozkład długości czasu, w którym bufor jest przepełniony. Badania przeprowadzono dla różnych parametryzacji modelu.

Słowa kluczowe: modele kolejkowe, MMCPP/GE/c/LG, kolejka z negatywnymi klientami, OMNeT++, symulacja stanów nieustalonych, symulator zdarzeń dyskretnych

SIMULATION OF A MULTI-SERVER QUEUE WITH NEGATIVE ARRIVALS

Summary. The article describes a simulation model of the MMCPP/GE/c/LG system with negativearrivals, limited buffer and many service points. The simulator's construction was described with particular emphasis on the solutions which are necessary to mark the characteristics of the transient state. Certain characteristics of the model were described, for example: distribution of time until the first over-loading of the buffer and distribution of the duration of time in which the buffer is overloaded. Studies were carried out for various model parameters.

Keywords: queueing models, MMCPP/GE/c/LG, queue with negative, arrivals, OMNeT++, simulation of transient states, discrete events simulator

[1] Praca naukowa finansowana ze środków Komitetu Badań Naukowych w latach 2004--2006 jako projekt badawczy nr 3 T11C 014 26

1. Wstęp

Systemy kolejkowe z negatywnymi klientami zostały zaproponowane przez Erola Gelenbe [3]. Do systemu takiego, oprócz strumienia normalnych (pozytywnych) klientów (zadań, zgłoszeń, pakietów, komórek), wpływa strumień tzw. negatywnych klientów. Działanie klientów negatywnych polega na usuwaniu klientów pozytywnych z kolejki, co daje duże możliwości modelowania różnorodnych zjawisk. Przykładowo, systemy kolejkowe z negatywnymi klientami zostały zaadaptowane do rozwiązywania problemów związanych z losowymi sieciami neuronowymi, wadliwymi elementami w liniach produkcyjnych, awariami serwerów, modelowaniem niezawodności itp. [5÷9].

Praca ta poświęcona jest symulacji kolejki z negatywnymi klientami, wieloma serwerami i skończoną poczekalnią (buforem) o długości L, czyli modelowi MMCPP/GE/c/LG, badanemu po raz pierwszy w [10]. W modelu tym zarówno strumień pozytywnych, jak i negatywnych klientów jest typu złożonego procesu Poissona z markowską modulacją (Markov-modulated compound Poisson Process, MMCPP), klienci zaś są obsługiwani przez *c* serwerów, z których każdy pracuje wg uogólnionego rozkładu wykładniczego (generalized exponential, GE). Oprócz intensywności strumieni wejściowych, również intensywności obsługi serwerów są modulowane przez łańcuch Markowa z ciągłym czasem. Ostatnia litera 'G' i w notacji MMCPP/GE/LG pochodzi od nazwiska Erola Gelenbe i oznacza właśnie system z negatywnymi klientami.

Zastosowanie modulacji markowskiej powoduje, że taki system może dobrze modelować kolejkowanie ruchu w sieciach komputerowych, gdzie kluczowe jest występowanie niekorzystnych zjawisk statystycznych w strumieniach przepływających danych. Chodzi tu o takie zjawiska jak, samopodobieństwo, dalekosiężne korelacje czy wybuchowość [11, 12]. Jak wiadomo, stosowanie markowskiej modulacji umożliwia dość dobre naśladowanie tego rodzaju zjawisk [13, 14].

Jeżeli chodzi o poprzednie wyniki dotyczące systemu MMCPP/GE/c/LG, to dostępne są jedynie dwie prace analityczne [10, 15], obie poświęcone podstawowym parametrom systemu w stanie ustalonym. Chociaż parametry stanu ustalonego umożliwiają ogólną orientację w wydajności systemu, to jednak często niezbędne jest poznanie charakterystyk stanu nieustalonego. Jak pokazano w [16], potrzeba ta nabiera szczególnego znaczenia w modelowaniu kolejkowania ruchu w sieciach komputerowych, właśnie ze względu na samopodobieństwo czy dalekosiężność. W takiej sytuacji system bardzo wolno osiąga stan ustalony i posługiwanie się parametrami stanu ustalonego bywa mylące. Wyprowadzenie wzorów dla podstawowych charakterystyk systemu MMCPP/GE/c/LG może być dość trudne z uwagi na dużą złożoność modelu (dwa strumienie wejściowe, modulacja, *c* serwerów).

W tej sytuacji pozostaje podejście oparte na badaniach symulacyjnych i takie zostało tutaj zastosowane. Do badania zostały wybrane dwa przykładowe parametry systemu w stanie nieustalonym, mianowicie: rozkład czasu do pierwszego przepełnienia bufora i rozkład długości czasu, w którym bufor jest przepełniony. Parametry te zostały wybrane nieprzypadkowo – obydwa pełnią szczególną rolę w ocenie efektywności kolejkowania ruchu w sieciach komputerowych. Pierwszy z nich, rozkład czasu do pierwszego przepełnienia bufora, jest szczególnie wrażliwy na samopodobieństwo w strumieniu wejściowym i daje nam informację, jak często bufor będzie się zapełniał (więcej na ten temat można znaleźć np. w [16, 17, 18]). Z kolei, rozkład długości czasu, w którym bufor jest przepełniony, daje obraz o statystycznej strukturze strat spowodowanych przepełnieniem bufora [19, 20]. W praktyce, ma to znaczenie np. przy dostrojaniu algorytmu FEC (Forward Error Correction, [21]), służącego unikaniu retransmisji w sieciach. W metodzie FEC do każdego bloku k wysyłanych pakietów dodaje się h pakietów nadmiarowych tak, aby utrata dowolnych h pakietów z bloku $k+h$ nie powodowała konieczności retransmisji (tzn. informacja może zostać odtworzona). Dla optymalnego doboru h istotne jest, jaka jest statystyczna struktura strat, tzn. rozkład liczby pakietów gubionych w okresie zapełnienia bufora.

Artykuł składa się z następujących części. W rozdziale 2 przedstawiony jest dokładny opis modelu MMCPP/GE/c/LG i jego parametrów. W rozdziale 3 opisany został symulator systemu MMCPP/GE/c/LG zaimplementowany za pomocą OMNeT++ [2]. Szczególną uwagę poświęcono tutaj technikom koniecznym do wyznaczania charakterystyk stanu nieustalonego[2]. Wreszcie w rozdziale 4 zaprezentowano wyniki symulacji dla przykładowych parametryzacji modelu.

2. Opis i parametry modelu MMCPP/GE/c/LG

Model kolejkowy systemu MMCPP/GE/c/LG składa się z dwóch źródeł emitujących pakiety o zmiennej długości (długość pakietu to liczba bloków, z których się ten pakiet składa) do wspólnej kolejki. Pierwsze źródło emituje pakiety *pozytywne* (zwiększające kolejkę o odpowiednią liczbę bloków; drugie źródło emituje pakiety *negatywne*, które usuwają odpowiednią liczbę bloków z kolejki według zadanego algorytmu. Odstęp czasu między kolejnymi pakietami wysyłanymi do kolejki ma rozkład wykładniczy o zmiennym

[2] Ponieważ zwykle w badanich symulacyjnych systemów kolejkowych poprzestaje się na charakterystykach stanu ustalonego, środowiska do symulacji, takie jak OMNET++, nie są wyposażone w narzędzia odpowiednie do badania stanów nieustalonych. W związku z tym konstrukcja symulatora wymaga przezwyciężenia wielu trudności technicznych z tym związanych.

parametrze $\lambda(i)$ ($\kappa(i)$ dla pakietów negatywnych), gdzie i to stan modulatora, a λ to intensywność źródła (odwrotność średniego odstępu czasu między wysyłanymi pakietami). Długość pakietu opisana jest zmienną losową o rozkładzie geometrycznym o zmiennym parametrze $\theta(i)$ dla pakietów pozytywnych i $\rho(i)$ dla pakietów negatywnych. $\theta(i)$ (odpowiednio $\rho(i)$) to średnia długość pakietu wyrażona w blokach zależna od stanu modulatora. Liczba bloków, jaka może się pomieścić w kolejce, jest stała i wynosi L. Pakiet pozytywny umieszczany jest w kolejce, jeżeli liczba jego bloków nie przekracza liczby miejsc dostępnych w kolejce. Jeżeli warunek ten jest niespełniony, pakiet jest odrzucany. W przypadku pakietu negatywnego istnieją dwa algorytmy dostępu do kolejki:

- RCH (removal of customers from the head) – pakiet negatywny usuwa odpowiednią liczbę bloków najpierw z serwerów (omawiany model posiada C działających równolegle serwerów) a następnie z kolejki, jeżeli pozostaną tam wolne bloki. Ten algorytm wykorzystuje się przy modelowaniu sytuacji awaryjnych.
- RCE (removal of customers from the end) – istnieją dwa warianty tego algorytmu: pakiet negatywny usuwa z kolejki odpowiednią liczbę bloków. Jeżeli liczba bloków w pakiecie negatywnym jest większa niż liczba bloków w kolejce, wszystkie bloki z kolejki są usuwane. Bloki obsługiwane przez serwery pozostają nienaruszone. Drugi wariant usuwa bloki z kolejki, a następnie, jeżeli to konieczne, z serwerów.

W artykule omówiono pierwszy wariant algorytmu RCE. Bloki z kolejki pobierane są przez C równolegle działające serwery. Każdy z serwerów pobiera z kolejki liczbę bloków określoną tą samą zmienną losową $\varphi(i)$ o rozkładzie geometrycznym, której parametr φ (średnia liczba pobieranych bloków z kolejki) zależy od stanu modulatora. Bloki pobierane są przez serwery wg następującego algorytmu: załóżmy, że w kolejce znajduje się j bloków. Każdy serwer w sposób nieprzerwany pobiera z kolejki wylosowaną liczbę bloków, ale nie większą niż $j-C+1$, gdy $j>C$; jeden blok, gdy $j \le C$; gdy kolejka jest pusta bloki są niepobierane; następnie serwery przechodzą w stan obsługi na czas $\mu(i)$ określony zmienną losową o rozkładzie wykładniczym. Stan i jest stanem modulatora, który jest obiektem globalnym dla źródeł i serwerów. Modulator jest automatem stanów skończonych (N stanów) opisanym procesem Markowa z czasem ciągłym i dyskretną przestrzenią stanów. Parametry modulatora opisane są za pomocą macierzy tranzycji Q. Modulator będąc w stanie i pozostaje w nim przez czas określony zmienną losową o rozkładzie wykładniczym z parametrem q_i (wartość bezwzględna z elementu na przekątnej macierzy w i-tym wierszu); następnie przechodzi do stanu j z prawdopodobieństwem $q_{i,j}/q_i$.

$$Q = \begin{bmatrix} -q_1 & q_{1,2} & \cdots & q_{1,N} \\ q_{2,1} & -q_2 & \cdots & q_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ q_{N,1} & q_{N,2} & \cdots & -q_N \end{bmatrix}$$

Stan modulatora wyznacza odpowiedni parametr źródeł i serwerów. W czasie zmiany stanu modulatora parametry serwerów i źródeł zmieniane są w sposób natychmiastowy. Jeżeli, dla przykładu, źródło ma wysłać do kolejki następny pakiet po czasie t_1, a modulator zmieni swój stan po czasie $t_2<t_1$, czas wysłania następnego pakietu przedłuży się o t_3 (wylosowany czas wysłania następnego pakietu dla nowego stanu modulatora), czyli będzie równy t_2+t_3. Pakiet ten będzie miał inną długość zależną od parametru zmiennej losowej (rozkład geometryczny) dla nowego stanu modulatora.

W artykule przeprowadzono symulację modelu w stanie nieustalonym w celu znalezienia rozkładu czasu pierwszego przepełnienia bufora oraz długości czasu, w którym bufor jest przepełniony dla różnych wartości obciążenia systemu σ. Wartość tę można wyznaczyć wg następujących wzorów:

$$\sigma \approx \frac{\overline{\lambda}}{\overline{\kappa} + C\overline{\alpha}} \ ,$$

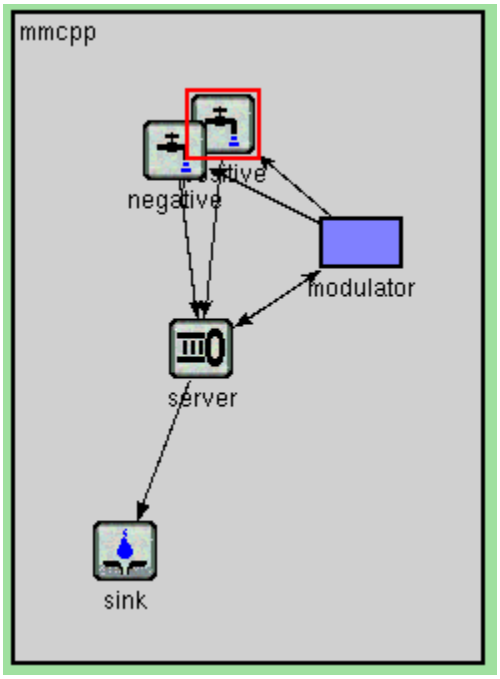
$$\overline{\alpha} = \sum_{i=1}^{N} \frac{r_i \mu_i}{1-\varphi_i} \ , \overline{\lambda} = \sum_{i=1}^{N} \frac{r_i \lambda_i}{1-\theta_i} \ , \overline{\kappa} = \sum_{i=1}^{N} \frac{r_i \kappa_i}{1-\rho_i}$$

Wektor r_i to stacjonarny rozkład łańcucha modulującego, można go uzyskać po przekształceniu Q do macierzy stochastycznej. Wektor własny dla wartości własnej równej jeden macierzy Q będzie wektorem r_i.

3. Model symulacyjny

Do symulacji wykorzystano obiektowy symulator OMNeT++ [2]. W skład symulatora wchodzi specjalny język NED, w którym opisuje się parametry i połączenia symulowanych obiektów. Z każdym symulowanym obiektem związana jest klasa napisana w języku C++. Szczegóły działania obiektu opisane są w redefiniowanej funkcji wirtualnej *activity*(). Obiekty symulacji komunikują się, wymieniając między sobą wiadomości. W przypadku symulowania sieci, wiadomości to pakiety, a dodatkowe informacje zapisane w polach wiadomości to np. długość pakietu, priorytet, rodzaj itd. Oprócz tego typu wiadomości istnieją jeszcze wiadomości sterujące, informujące o zdarzeniach czasowych i natychmiastowych. Przykładem zdarzenia czasowego jest wysłanie pakietu do kolejki po czasie określonym przez generator losowy – obiekt zgłaszający tego typu zdarzenie (przez wysłanie wiadomości

sterującej do *schedulera* symulatora) po odpowiednim czasie (wyznaczonym przez generator losowy) dostaje wiadomość (wiadomości sterująca od *schedulera* symulatora), że zadanie należy wysłać do kolejki. *Scheduler* to wewnętrzna kolejka w jądrze symulatora w której szeregowane są wiadomości sterujące wg czasu. Przykład zdarzenia natychmiastowego to wysłanie wiadomości sterującej do obiektów o pewnym zdarzeniu, np. przepełnieniu bufora.



Rys. 1. Model symulacyjny MMCPP/GE/c/LG
Fig. 1. Simulation of MMCPP/GE/c/LG in OMNeT++

Komunikacja między obiektami odbywa się poprzez nie blokującą funkcję *send*(), za pomocą której wysyła się wiadomość do określonego obiektu poprzez port (zdefiniowany w języku NED). Obiekt docelowy odbiera wiadomość za pomocą blokującej funkcji *recive*(). Ponadto, wiadomość można wysłać do samego siebie (wiadomość sterująca) o określonym czasie za pomocą funkcji *scheduleAt*() – w ten sposób można symulować zdarzenia czasowe, takie jak np. koniec obsługi na danym stanowisku. Jest to jedna z technik symulacji, jaką oferuje symulator: każdy obiekt działa jako niezależny wątek komunikujący się z innymi obiektami (działającymi niezależnie) za pomocą komunikatów. Inną techniką jest symulacja z obsługą zdarzeń (ang. message handler). Ze względu na równoległy charakter symulowanych obiektów metoda pierwsza jest zdaniem autora bardziej naturalna i łatwiejsza w implementacji. Ponadto, symulator dostarcza wygodnego interfejsu graficznego dla użytkownika z możliwościami animacji symulowanych zdarzeń i wizualizacji wyników. Na rys. 1 pokazano model symulacyjny systemu MMCPP/GE/c/LG

Obiekt *positive* reprezentuje źródło pakietów pozytywnych, obiekt *negative* źródło pakietów negatywnych. Obiekty wysyłają pakiety pozytywne i negatywne do kolejki obiektu

server. Długość wysyłanych pakietów i intensywność wysyłania ich do kolejki jest określona przez generatory pseudolosowe o rozkładach wykładniczych i geometrycznych (dokładną charakterystykę źródeł omówiono w poprzednim rozdziale). Obiekt *server* składa się z kolejki i *C* równolegle działających serwerów. Serwery pobierają odpowiednią liczbę bloków z bufora (określoną przez parametr zmiennej losowej), symulują czas obsługi, następnie wysyłają pakiet do obiektu *sink*, gdzie jest on niszczony. Obiekt *modulator* jest obiektem globalnym, zmienia swój stan od 1 do *N*, jego parametry opisane są macierzą *Q*. W momencie zmiany stanu modulator wysyła wiadomość sterującą do źródeł i serwera z informacją o nowym stanie. Obiekty te muszą usunąć ze *schedulera* informację o czasie wysłania (odpowiednio: czasie pobrania dla serwera) następnego pakietu. Poniżej przedstawiono fragment funkcji *activity*() dla źródła pozytywnego, które otrzymało wiadomość od modulatora o zmianie stanu:

```
msg=receive(); //odebranie wiadomości
if(msg->kind()==MOD_MSG) //  czy wiadomość  z modulatora ?
  {
   //tak
   // odczytujemy nowy stan modulatora:
   state=msg->length();

   // usunięcie z kolejki schedulera  zdarzenia  czasowego  dla
   // poprzedniej wartości modulatora:
  cMessage * lst =cancelEvent(last_packet_msg);
  delete lst;

   // utworzenie nowego pakietu:
   send_packet_msg=new cMessage("send_packet");
   send_packet_msg->setKind(0); // argument 0  oznacza pakiet pozytywny

   // zachowanie adresu  wiadomości, w celu możliwości usunięcia jej
   // ze schedulera:
   last_packet_msg=send_packet_msg;

   // wysłanie wiadomości po czasie exponential(1/Lambda(stan)):
   scheduleAt( simTime()+ (double)exponential(1/Lambda[state]), send_packet_msg

   delete msg;
  }
```

Usunięcie wiadomości ze *schedulera* odbywa się za pomocą funkcji *cancelEvent*(*cMessage * last_packet_msg*). Funkcja ta przegląda kolejkę wiadomości i usuwa wiadomość o adresie *last_packet_msg*. Ze względu na to, że adres ten trzeba podać jako parametr funkcji *cancelEvent*(), programista sam musi zadbać, aby adres ostatniej wiadomości został przechowany w jakieś zmiennej.

Parametry symulacji zapisuje się w pliku *omnetpp.ini* (wczytywany automatycznie przez symulator), ponadto wykorzystano dodatkowe pliki z rozszerzeniem *.dat*:

- *omnetpp.ini*: w zmiennej *mmcpp.server.buffsize* podaje się długość kolejki, w *mmcpp.-modulator.init_state* stan początkowy modulatora, a w *mmcpp.server.C* liczbę serwerów*;*

- *arrival_pos.dat*: w pierwszej linii podaje się liczbę stanów modulatora, w drugiej wartości $\lambda(i)$ a w trzeciej $\theta(i)$*;*
- *arrival_neg.dat*: w pierwszej linii podaje się liczbę stanów modulatora, w drugiej wartości $\kappa(i)$*, a w* trzeciej $\rho(i)$*;*
- *server.dat*: w pierwszej linii podaje się liczbę stanów modulatora, w drugiej wartości $\mu(i)$, *a w* trzeciej $\varphi(i)$*;*
- *qmatrix.dat*: w pierwszej linii podaje się liczbę stanów modulatora (wymiar macierzy modulatora), a następnie wszystkie elementy macierzy (druga linia to pierwszy wiersz macierzy);
- *r.dat*: w pierwszej linii podaje się liczbę stanów modulatora, w drugiej wartości wektora stacjonarnego r_i dla łańcucha modulującego.

3.1. Symulacja stanów nieustalonych w OMNeT ++

W symulatorze OMNeT++ nie ma możliwości wyzerowania zegara symulatora bez przerwania symulacji, co utrudnia symulację stanów nieustalonych. Przedmiotem badań było znalezienie czasu pierwszego przepełnienia bufora oraz czasu trwania stanu, w którym był on przepełniony. Problem ten rozwiązano następująco: Obiekt *serwer* który kontroluje kolejkę, w momencie jej przepełnienia, zapisuje bieżący czas symulacji, odejmuje go od czasu, w którym nastąpiło poprzednie zapełnienie bufora, zeruje swoją kolejkę (jeżeli warunkiem początkowym badanego stanu nieustalonego jest pusta kolejka); wysyła informację do modulatora (zdarzenie natychmiastowe) o przepełnieniu, który ustawia się na stan jeden (w zależności od warunku początkowego) i wysyła (modulator) informację o nowym stanie do serwera i źródeł, które rozpoczynają pracę od nowego stanu, usuwając za pomocą funkcji *cancelEvent*() poprzednie wiadomości. Średnia arytmetyczna różnicy czasu poprzedniego i aktualnego przepełnienia bufora jest średnim czasem pierwszego przepełnienia bufora. Analogicznie wyznacza się czas trwania stanu, w którym bufor był przepełniony.

4. Wyniki symulacji

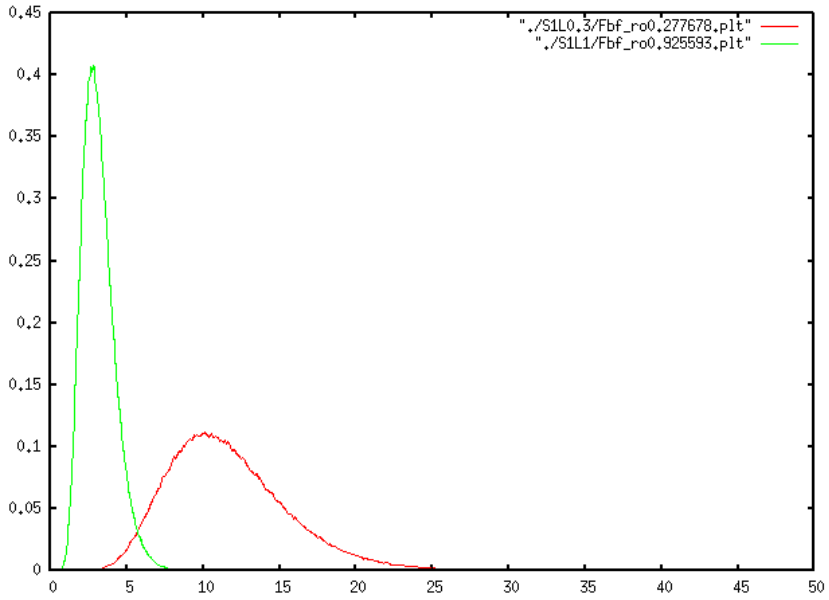
Symulację przeprowadzono wstępnie dla następujących parametrów:

- λ=(10, 20, 30), θ=(0.1, 0.2, 0.3);
- κ=(6, 9, 5), ρ=(0.25, 0.1, 0.2);
- μ=(2, 0.5, 1.5), φ=(0.5, 0.6, 0.289);

- $Q = \begin{pmatrix} -2 & 1 & 1 \\ 1 & -7 & 6 \\ 3 & 2 & -5 \end{pmatrix}$, r=(0.5359, 0.163, 0.302);
- N=3, L=250, C=5.

Rysunek 2 przedstawia rozkład prawdopodobieństwa pierwszego przepełnienia bufora dla obciążenia σ=0.28 i 0.93. Mniejsze obciążenie (σ=0.28) uzyskano zmniejszając parametr λ ($\lambda = 0.3\lambda$). Uzyskano następujące wyniki:

- dla $\sigma = 0.28$: $t_{min.} = 1.61$, $t_{max} = 48.49$, $t_{śr} = 11.71$, $S_n = 3.98$;
- dla $\sigma = 0.93$: $t_{min.} = 0.4$, $t_{max} = 12.97$, $t_{śr} = 3.16$, $S_n = 1.08$.



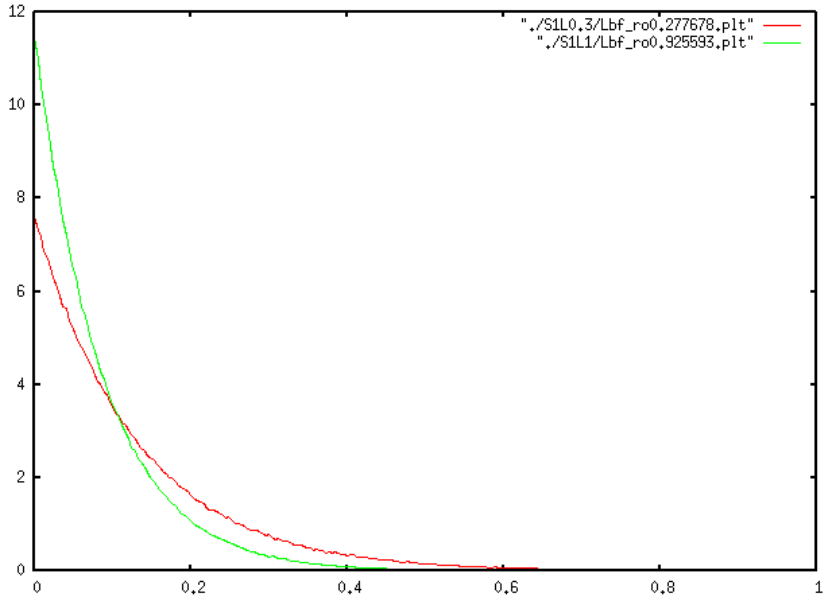
Rys. 2. Rozkład prawdopodobieństwa pierwszego wypełnienia bufora dla $\sigma = 0.28$ *i* $\sigma = 0.93$
Fig. 2. Distribution of probability of the buffer's first filling for $\sigma = 0.28$ and $\sigma = 0.93$

Dla mniejszego obciążenia średni czas przepełnienia bufora oraz zakres jest prawie 3 razy dłuższy niż dla większego obciążenia, wynika to z tego, że źródło pakietów pozytywnych ma intensywność 3 razy mniejszą.

Na rysunku 3 przedstawiono rozkład prawdopodobieństwa długości czasu, w którym bufor jest przepełniony dla tych samych parametrów wejściowych. Wyniki symulacji:

- dla σ=0.277678: $t_{min.}$= 1.3e-7, t_{max}=1.92, $t_{śr}$=0.12, S_n= 0.13;
- dla σ=0.925593: $t_{min.}$= 1.9e-7, t_{max}=1.41, $t_{śr}$=0.08, S_n=0.08.

Średni czas trwania stanu, w którym bufor jest wypełniony, jest znacznie krótszy od średniego czasu jego wypełnienia. Interesujący jest fakt, że dla mniejszego obciążenia jest on dłuższy niż dla większego, chodź mogłoby się wydawać, że powinien być taki sam, ponieważ źródło pozytywne nie powinno mieć wpływu na czas trwania tego stanu.



Rys. 3. Rozkład prawdopodobieństwa długości czasu, w którym bufor jest przepełniony dla $\sigma = 0.28$ *i* $\sigma = 0.93$.

Fig. 3. Distribution of probability in reference to the duration of time in which the buffer is overloaded for $\sigma = 0.28$ i $\sigma = 0.93$

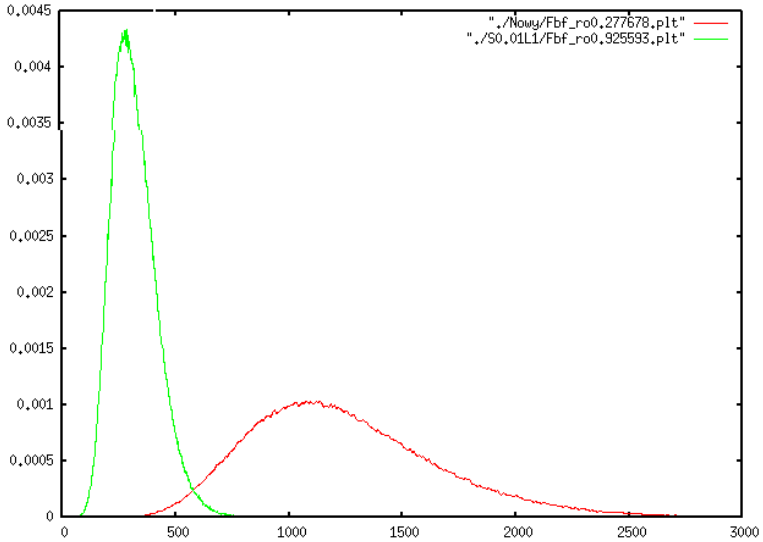
W tym rozdziale przedstawiono wyniki symulacji dla wartości parametrów: λ, κ *i* μ stukrotnie mniejszych:

- λ=(0.01, 0.02, 0.03);
- κ=(0.06, 0.0 9,0.05);
- μ=(0.02, 0.005, 0.015).

Pozostałe parametry zostały niezmienione. Przy takich warunkach pracy obciążenie systemu się nie zmieni, natomiast modulator będzie 100 razy częściej zmieniał swój stan.

Rysunek 4 przedstawia rozkład prawdopodobieństwa pierwszego przepełnienia bufora dla obciążenia σ=0.28 i 0.93. Uzyskano następujące wyniki:

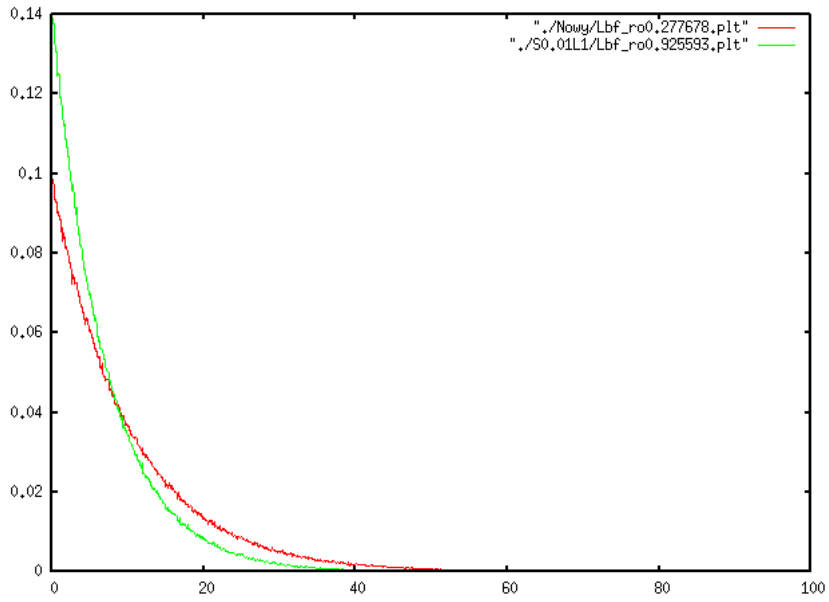
- dla σ=0.28*:* $t_{min.}$=112.76, t_{max}=5131.85, $t_{śr}$=1256.31, S_n= 430.67*;*
- dla σ=0. 93*:* $t_{min.}$= 34.09, t_{max}=1130.67, $t_{śr}$=320.76, S_n=100.37.



Rys. 4. Rozkład prawdopodobieństwa pierwszego wypełnienia bufora dla $\sigma = 0.28$ *i* $\sigma = 0.93$
Fig. 4. Distribution of probability of the buffer's first filling for $\sigma = 0.28$ and $\sigma = 0.93$

Jak widać na wykresie, czas pierwszego zapełnienia wzrósł 100-krotnie. Można wytłumaczyć to tym że odstępy czasów między emitowanymi pakietami dla źródeł i czasy obsługi dla serwerów są stale wydłużane przez ciągłe zmiany modulatora. Podobnie czas stanu, w którym bufor jest wypełniony (rys. 5), wzrósł stukrotnie:

- dla σ=0.28*:* $t_{min.}$=5.39e-06, t_{max}=145.77, $t_{śr}$=10.15, S_n= 10.16*;*
- dla σ=0. 93*:* $t_{min.}$= 2.89e-06, t_{max}=118.29, $t_{śr}$=7.05, S_n=7.05.



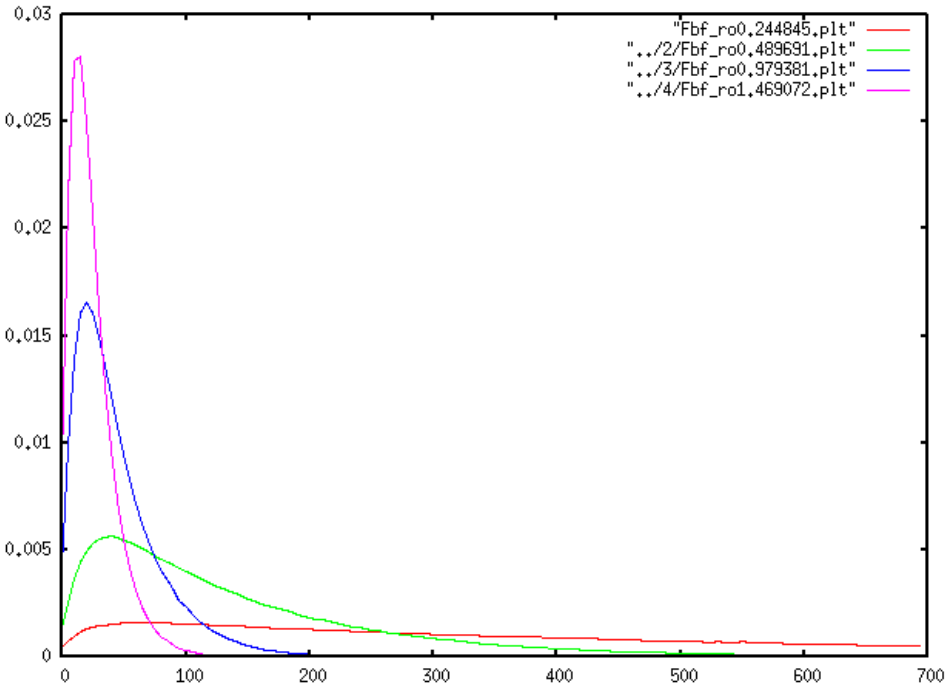
Rys. 5. Rozkład prawdopodobieństwa długości czasu, w którym bufor jest przepełniony dla $\sigma = 0.28$ *i* $\sigma = 0.93$
Fig. 5. Distribution of probability in reference to the duration of time in which the buffer is overloaded for $\sigma = 0.28$ *i* $\sigma = 0.93$

Odchylenie standardowe dla rozkładu prawdopodobieństwa czasu trwania stanu w którym bufor jest przepełniony, jest takie same jak średni czas trwania tego stanu.

Na rys. 6 pokazano rozkład prawdopodobieństwa czasu pierwszego przepełnienia bufora dla wartości obciążenia systemu σ= 0.24, 0.49, 0.98, 1.47. Przyjęto następujące parametry wejściowe:

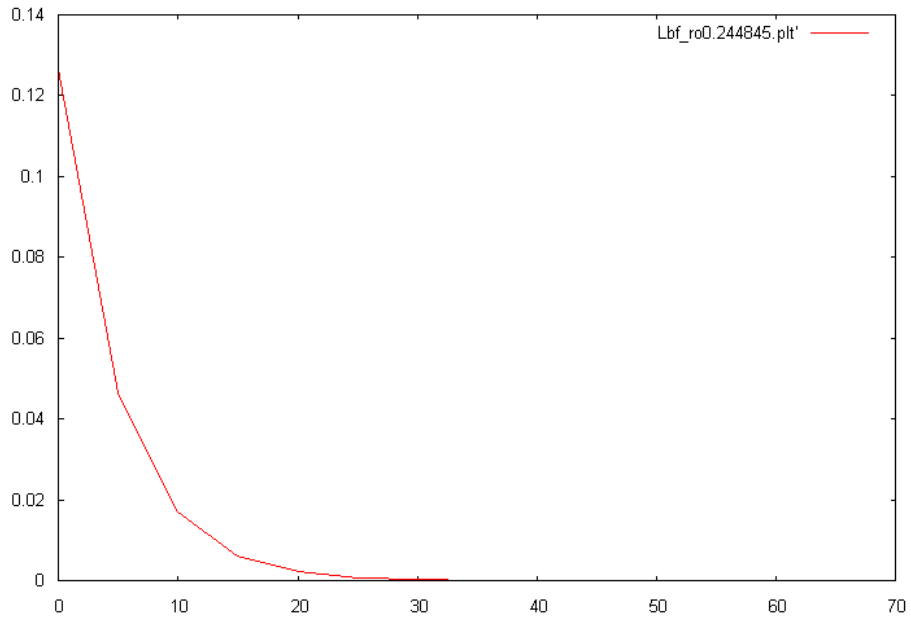
- λ = (0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05), θ=(0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01), σ = 0.24;
- λ=(0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1), θ=(0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01), σ = 0.49;
- λ = (0.2, 0.2, 0.2, 0.2, 0.2, 0.2, 0.2, 0.2), θ=(0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01), σ = 0.98;
- λ=(0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3), θ=(0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01), σ = 1.47;
- κ = (0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1), ρ=(0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01);
- μ = (0.,1 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1), φ(0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05);
- $Q = \begin{pmatrix} -0.059 & 0.06 & 3.48e-06 & 1.6e-10 & 6.13e-15 & 2.23e-19 & 7.34e-24 & 1.15e-28 \\ 0.03 & -0.06 & 0.02 & 1.6e-06 & 8.7e-11 & 3.9e-15 & 1.53e-19 & 2.28e-24 \\ 2.9e-06 & 0.035 & -0.059 & 0.024 & 1.89e-06 & 1.11e-10 & 5.5e-15 & 1.212e-19 \\ 1.79e-10 & 3.188e-06 & 0.03 & -0.059 & 0.03 & 2.3e-06 & 1.5e-10 & 4.3e-15 \\ 7.4e-15 & 1.7e-10 & 2.63e-16 & 0.02 & -0.06 & 0.03 & 2.9e-06 & 1.09e-10 \\ 6.58e-15 & 1.3e-10 & 2.13e-06 & 0.025 & 0.005 & -0.06 & 0.03 & 1.8e-06 \\ 7e-24 & 2.511e-19 & 6.2e-15 & 1.3e-10 & 2.43e-06 & 0.032 & -0.059 & 0.02 \\ 2.4e-28 & 1.02e-23 & 3.05e-19 & 8.28e-15 & 1.97e-10 & 3.9ee-06 & 0.059 & -0.059 \end{pmatrix}$;
- r = (0.12, 0.20, 0.14, 0.1, 0.1, 0.123, 0.12, 0.05);
- N = 8, L = 500, C = 5.

Dla dużych wartości σ kolejka się szybko zapełnia, ale i szybko opróżnia, dla małych wartości σ zarówno czas zapełniania, jak i opróżniania bufora jest długi. Rysunek 2 przedstawia rozkład prawdopodobieństwa długości czasu, w którym bufor jest przepełniony. Ponieważ symulację przeprowadzono dla różnych wartości $\lambda(i)$ (intensywność źródła pakietów pozytywnych nie wpływa na długość czasu, w którym bufor jest zapełniony), rozkład prawdopodobieństwa na rys. 7 jest taki sam dla wszystkich wartości σ.



Rys. 6. Rozkład prawdopodobieństwa pierwszego wypełnienia bufora dla $\sigma = 0.24, 0.49, 0.98, 1.47$

Fig. 6. Distribution of probability in reference to the duration of time in which the buffer is overloaded for $\sigma = 0.24, 0.49, 0.98, 1.47$



Rys. 7. Rozkład prawdopodobieństwa długości czasu, w którym bufor jest przepełniony

Fig. 7. Distribution of probability in reference to the duration of time in which the buffer is overloaded

5. Podsumowanie

W artykule zaprezentowano model symulacyjny systemu MMCPP/GE/c/LG z negatywnymi klientami, skończonym buforem i wieloma stanowiskami obsługi. Wyznaczone zostały przykładowe charakterystyki modelu, takie jak: rozkład czasu do pierwszego przepełnienia bufora i rozkład długości czasu, w którym bufor jest przepełniony. Badania przeprowadzono dla różnych parametryzacji modelu, wykorzystując bibliotekę do symulacji OMNET++. Pokazano, w jaki sposób symulować stany nieustalone za pomocą wspomnianej biblioteki, która nie jest w pełni przystosowana do tego typu analizy.

LITERATURA

1. Czachórski T.: Modele kolejkowe w ocenie efektywności sieci i systemów komputerowych. Pracownia Komputerowa Jacka Skalmierskiego, Gliwice 1999.
2. Varga A.: The OMNeT++ Discrete Event Simulation System. Proceedings of the European Simulation Multiconference, Praga 2001.
3. Gelenbe E.: Random neural networks with positive and negative signals and product form solution. Neural Comput., 1(4), 1989, s. 502÷510.
4. Gelenbe E., Glynn P., Sigman K.: Queues with negative arrivals. J. Appl. Probab. 28, 1991, s. 656÷663.
5. Gelenbe E.: Queueing networks with negative and positive customers. J. Appl. Probab. 26, 1991, s. 623÷643.
6. Fourneau J. M., Gelenbe E., Suros R.: G-networks with multiple classes of positive and negative customers. Theor. Comput. Sci. 155, 1996 s. 141÷156.
7. Fourneau J. M., Hernandez M.: Modelling defective parts in a flow system using G networks. Second International Workshop on Performability Modelling of Computer and Communication Systems, Le Mont Saint-Michel, June 1993.
8. Harrison P. G., Patel N. M., Pitel E.: Reliability modelling using G-queues. Eur. J. Oper. Res. 126, 2000, s. 273÷287.
9. Gelenbe E., Glynn P, Sigman K.: Queues with negative arrivals. J. Appl. Probab., 28, 1993, s. 656÷663.
10. Chakka R., Harrison P. G.: A Markov modulated multi-server queue with negative customers – The MMCPP/GE/c/L G-queue. Acta Informatica 37, 2001.
11. Leland W., Taqqu M., Willinger W.: On the self-similar nature of ethernet traffic (extended version). IEEE/ACMTransactions on Networking 2(1), 1994, s. 1÷15.

12. Crovella M., Bestavros A.: Self-similarity in World Wide Web traffic: Evidence and possible causes. IEEE/ACM Transactions on Networking 5(6), 1997, s. 835÷846.
13. Yoshihara T., Kasahara S. and Takahashi Y.: Practical time-scale fitting of self-similar traffic with Markov-modulated Poisson process. Telecommunication Systems 17(1/2), 2001, s. 185÷211.
14. Salvador P., Valadas R. and Pacheco A.: Multiscale Fitting Procedure Using Markov Modulated Poisson Processes. Telecommunication Systems 23, 2003, s. 123÷148.
15. Harrison P. G. The MMCPP/GE/cG-Queue: sojourn time distribution. Queueing Systems, Vol: 41, 2002, s. 271÷298.
16. Schwefel H. P., Lipsky L., Jobmann M.: On the necessity of transient performance analysis in telecommunication systems. Teletraffic Engineering in the Internet Era, Elsevier, Amsterdam, 2001.
17. Asmussen S., Jobmann M and Schwefel H. P.: Exact Buffer Overflow Calculations for Queues via Martingales. Queueing Systems 42(1),2002, s. 63÷90.
18. Chydziński A., Winiarczyk R.: Distribution of the First Buffer Overflow Time in a Deterministic Service Time Queue. Proc. Of 10-th IEEE Symposium on Computers and Communications, La Manga, Spain, 2005.
19. Chydzinski A.: On the Distribution of Consecutive Losses in a Finite Capacity Queue. WSEAS Transactions on Circuits and Systems. Issue 3, Vol. 4, 2005, s. 117÷124.
20. Boer de P. T., Nicola V. F., Ommeren J. C. W.: The Remaining Service Time Upon Reaching a High Level in M/G/1 Queues. Queueing Systems, 39, 2001, s. 55÷78.
21. Carle, G., Biersack, E.: Survey of error recovery techniques for IP-based audiovisual multicast applications. IEEE Network 11(6), 1997, s. 24÷36.

Recenzent: Dr inż Andrzej Chydziński

Wpłynęło do Redakcji 27 października 2006 r.

Abstract

Queuing systems with negative customers were proposed by Erol Gelenbe. Apart from a stream of normal(positive) customers (tasks, applications, packets, cells) a stream of so-called negative customers flows into this system. The operation of negative customers consists in removing positive customers from the queue which creates great opportunities of modelling various phenomena. For example queuing systems with negative customers have

been adapted to solving problems connected with random neural networks, faulty elements in production lines, server breakdowns, modelling reliability etc.

This paper is concerned with a queue simulation with negative customers, multiple servers and a finished waiting room (buffer) of the L length, that is a MMCPP/GE/c/LG model.

Thanks to using a Markov modulation in the model this system can model well the queuing of the movement in computer networks, where the occurrence of disadvantageous statistical phenomena in the streams of flowing data is crucial. This is to mean such phenomena as self similarity, far-reaching correlations or burstiness.

Two examples of system parameters in a transient state were selected for examination, that is: time distribution until the first overfilling of the buffer and distribution of the length of time in which the buffer is overfilled.

The paper consists of the following parts: in the second chapter there is a detailed description of the MMCPP/GE/c/LG model and its parameters, in the third chapter the MMCPP/GE/c/LG system simulator, implemented in OMNET++, was described. In the third charter particular attention was given to the techniques necessary to mark the characteristics of a transient state. Finally, in the fourth chapter the results of simulation for examples of model parametrizing were presented.

Adres

Piotr PECKA: Instytut Informatyki Teoretycznej i Stosowanej PAN, ul.Bałtycka 5,
44-100 Gliwice, Polska piotr@iitis.gliwice.pl