Збірник Наукових праць ОНАЗ ім. О.С. Попова, № 1, 2007. - с. 25-31.
УДК 621.39, 004.7

ЗАЙЦЕВ Д.А., ЛИТВИН Д.А.
ZAITSEV D.A., LITVIN D.A.

ОЦЕНКА ПРОИЗВОДИТЕЛЬНОСТИ СЕТЕЙ С КОММУТАЦИЕЙ МЕТОК В МОДЕЛИРУЮЩЕЙ СИСТЕМЕ NS

PERFORMANCE EVALUATION OF LABEL SWITCHING NETWORKS WITH SIMULATION SYSTEM NS

Аннотация. Представлена методика построения моделей MPLS магистралей в среде моделирующей системы NS на примере фрагмента Европейской магистрали Интернет. Для динамической оценки пропускной способности магистрали разработана процедура, периодически оценивающая производительность сети в процессе имитационного моделирования. Процедура может быть использована для обнаружения кратковременных перегрузок в магистралях. Подтверждена адекватность ранее представленных моделей, построенных в форме раскрашенных сетей Петри.

Summary. The technique of MPLS backbones' models construction into simulation system NS on the example of European Internet backbone's fragment was presented. For the dynamic performance evaluation of the backbone, a procedure was developed, that estimates periodically the network's performance during the simulation process. The procedure might be used for the revealing of short duration overloads into backbones. The adequacy of early presented models in the form of Colored Petri nets was acknowledged.

Технология коммутации меток MPLS [1] объединяет принципы коммутации пакетов и коммутации каналов в телекоммуникационных сетях. Метка представляет собой, по-существу, идентификатор некоторого виртуального канала и доставка пакета, дополненного меткой, не требует просмотра объёмных таблиц маршрутизации. В настоящее время коммутация меток является основной технологией для решения проблемы повышения производительности магистральных сетей, что обуславливает актуальность задач оценки производительности и качества обслуживания сетей MPLS.

Система NS (Network Simulator) [2] занимает центральное место среди специализированных систем имитационного моделирования телекоммуникационных сетей. NS создана как расширение объектно-ориентированных языков программирования OTcl, C++ и лидирует по количеству применений в реальных проектах.

В [3] представлена модель магистральной сети с классической IP-маршрутизацией, построенная в системе NS; исследования выполнены для фрагмента Европейской магистрали Интернет, изученного в [4] с помощью раскрашенных сетей Петри. Однако в [3] не рассмотрена коммутация меток. Кроме того, при построении модели использована простая последовательность директив NS, что привело к громоздкой модели; при оценке характеристик применён анализ полной сохранённой трассы имитации, которая также может быть достаточно большой при моделировании реальных магистралей.

Целью настоящей работы является построение компактных моделей магистральных MPLS сетей в системе NS и процедур оценки производительности сети непосредственно в процессе имитационного моделирования.

1. Описание топологии сети. В настоящей работе построена модель магистральной MPLS сети, описанной в [4]. На Рис. 1 представлена схема сети, полученная в аниматоре NAM. Сеть состоит из 6 терминальных подсетей T1-T6 (узлы 7-12 соответственно), территориально расположенных в различных странах [4], и магистральной сети, образованной 7-ю MPLS-маршрутизаторами LSR1-LSR7 (узлы 0-5 соответственно).

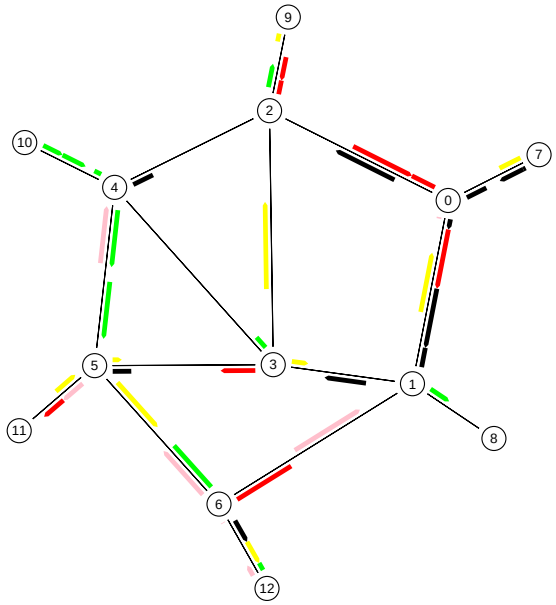

Рисунок 1 – Схема магистральной сети, полученная в аниматоре NAM

Следует отметить, что аниматор отображает потоки передаваемых (вдоль линий связи) пакетов и очереди; различные цвета использованы для отображения пакетов, сгенерированных различными терминальными сетями. Это позволяет оценить работу сети визуально и обнаружить узкие места.

При построении модели структура IP-адреса не рассматривалась, использована простая схема адресов plain. Узлы сети описаны последовательностью tcl команд, представленной на Рис. 2.

```
set LSR_num 7
$ns node-config -MPLS ON
#Create routers
for {set i 0} {$i < $LSR_num} {incr i} {
        set LSR($i) [$ns node]
}
$ns node-config -MPLS OFF

#Create terminal networks
set T_num 6
for {set i 0} {$i < $T_num} {incr i} {
        set T($i) [$ns node]
}
```

Рисунок 2 – Описание узлов сети

Заметим, что при создании маршрутизаторов включена опция MPLS в конфигурации узла (`node-config`). Переменные `LSR_num` и `T_num` задают количество маршрутизаторов магистральной сети и количество терминальных сетей соответственно. Создано два массива узлов: маршрутизаторы – `LSR($i)` и терминальные сети – `T($i)` с помощью директивы `$ns node`.

Линии связи описаны также как в [3] простой последовательностью соответствующих директив в соответствии со схемой сети (Рис. 1). Заметим, что генерация топологии с помощью операторов цикла (как для узлов на Рис. 2) возможна только для регулярных структур. Приведём пример описания линии связи между терминальной сетью `T(5)` и маршрутизатором `LSR(6)`:

```
$ns duplex-link $T(5) $LSR(6) 1Mb 100ms DropTail
```

Линия полнодуплексная (`duplex-link`), указаны её конечные узлы, пропускная способность (`1Mb`), задержка обслуживания (`100ms`), тип очереди с потерей пакетов при переполнении (`DropTail`). Задержка линии моделирует время обслуживания в узле и выбрана в соответствии с процедурой масштабирования времён для конкретных MPLS-маршрутизаторов, описанной в [4].

В модели использованы опции симулятора, представленные на Рис. 3. Они описывают создание нового экземпляра симулятора при запуске модели, открытие файла для записи результатов динамических оценок пропускной способности, динамическую маршрутизацию, конфигурирование протоколов MPLS, в особенности, протокола динамического распределения меток LDP.

```
set ns [new Simulator]
set f0 [open mpls.tr w]
$ns rtproto DV
Classifier/Addr/MPLS set control_driven_ 1
Classifier/Addr/MPLS enable-on-demand
Classifier/Addr/MPLS enable-ordered-control
Agent/LDP set trace_ldp_ 1
Classifier/Addr/MPLS set trace_mpls_ 1
```

Рисунок 3 – Опции симулятора

2. Описание трафика сети. В соответствии с [4] каждая терминальная сеть генерирует потоки к каждой другой терминальной сети. В [4] взаимодействие клиент-сервер не рассматривалось, поэтому для генерации потокового трафика выбран агент UPD. На Рис. 4 представлены агенты источников и стоков трафика. Заметим, что в отличие от [3] использование массивов и циклов tcl позволило значительно сократить описания. Классы трафика (`class`) выбраны для отображения различными цветами пакетов различных терминальных сетей в NAM.

```
for {set i 0} {$i < $T_num} {incr i} {
for {set j 0} {$j < $T_num} {incr j} {
  if {$i != $j} {
     set ij [expr [expr $T_num * $i] + $j]
     #Create a UDP agent and attach it to node
     set udp($ij) [new Agent/UDP]
     $udp($ij) set class_ $i
     $ns attach-agent $T($i) $udp($ij)
     # Create a Exponential traffic source and attach it to udp
     set exp($ij) [new Application/Traffic/Exponential]
     $exp($ij) set packetSize_ 200
     $exp($ij) set birst_time_ 500ms
     $exp($ij) set idle_time_  500ms
     $exp($ij) set rate_ 200K
     $exp($ij) attach-agent $udp($ij)
     #Create a traffic sink and attach it to node
     set sink($ij) [new Agent/LossMonitor]
     $ns attach-agent $T($i) $sink($ij)
   }
}
}
```

Рисунок 4 – Описание источников и стоков трафика

Циклы (`for`) по переменным `i,j` задают выбор всех возможных пар терминальных сетей; условный оператор (`if`) исключает из рассмотрения потоки, направленные в ту же самую терминальную сеть; макрос `ij` служит для организации двумерных массивов агентов

`udp($ij)`, `exp($ij)`, `sink($ij)` в одномерных массивах языка tcl. На транспортном уровне создаются агенты UDP (`set udp($ij)`), которые затем присоединяются (`attach-agent`) к узлам терминальных сетей `T($i)`.

Для генерации трафика использованы источники (`set exp($ij)`) со случайным экспоненциально распределённым временем и фиксированной длиной пакетов, равной 200 байтов. Генератор `exp($ij)` включается на случайный экспоненциально распределённый интервал времени со средним `burst_time_` и выключается на случайный экспоненциально распределённый интервал времени со средним `idle_time_`; во включенном состоянии он генерирует пакеты с постоянной битовой скоростью `rate_`. Генераторы трафика присоединяются к соответствующим агентам UDP (`attach-agent`). Для поглощения трафика на узлах назначения создаются стоки `sink($ij)`, которые затем присоединяются (`attach-agent`) к соответствующим узлам.

Аналогичным образом в двойном цикле выполнено соединение источников и стоков (Рис. 5), задающее направление (адресацию) трафика в сети.

```
for {set i 0} {$i < $T_num} {incr i} {
for {set j 0} {$j < $T_num} {incr j} {
  if {$i != $j} {
    set ij [expr [expr $T_num * $i] + $j]
    set ji [expr [expr $T_num * $j] + $i]

    $ns connect $udp($ij) $sink($ji)
  }
}
}
```

Рисунок 5 – Соединение источников и стоков трафика (адресация)

Дополнительный макрос `ji` создан для обеспечения регулярного использования индексов двумерного массива: первый индекс везде задаёт собственную терминальную сеть; в генераторах второй индекс задаёт сеть назначения, в стоках – сеть, отправляющую трафик.

3. Оценка характеристик модели. В [3] использован простейший способ оценки характеристик сети с помощью анализа полной трассы симулятора; представлены скрипты ОС Unix, вычисляющие пропускную способность сети и время доставки пакета. В настоящей работе для получения оценок производительности (пропускной способности) сети разработана специальная tcl процедура `evalPerformance`, представленная на Рис. 6. Особенностью процедуры является обеспечение динамических оценок. Процедура запускается каждые 0.1 с. и, используя счётчики байтов `bytes_` агентов `sink($i)` класса `LossMonitor`, оценивает трафик, полученный каждой из терминальных сетей и общий трафик.

```
proc evalPerformance {} {
        global T_num sink f0
        set ns [Simulator instance]
        #Set the time slice
        set time 0.1
        set total 0.0
        #Get the current time
        set now [$ns now]
        #Calculate number bytes received by the traffic sinks
        for {set i 0} {$i < $T_num} {incr i} {
          set sum 0.0
          for {set j 0} {$j < $T_num} {incr j} {
            if {$i != $j} {
                    set ij [expr [expr $T_num * $i] + $j]
                    set bb [$sink($ij) set bytes_]
                    set sum [expr $sum + $bb]
                    #Reset the bytes_ values on the traffic sink
                    $sink($ij) set bytes_ 0
            }
          }
          #Calculate the bandwidth (in MBit/s) and write it to the file
          set bw($i) [expr $sum/$time*8/1000000]
          set bytes [expr $total + $sum($i)]
        }
        #Calculate the total bandwidth (in MBit/s) and write it to the file
        puts bwt [expr $total/$time*8/1000000]
        puts $f0 "$now $bw(0) $bw(1) $bw(2) $bw(3) $bw(4) $bw(5) $bwt"
        #Re-schedule the procedure
        $ns at [expr $now+$time] "evalPerformance"
}
```

Рисунок 6 – Процедура оценки производительности сети

Переменная `time` устанавливает интервал перевычислений; переменная `total` накапливает общее количество байт, переданных в сети; переменная `sum` накапливает количество байт, полученных текущей терминальной сетью `T($i)` от всех остальных терминальных сетей `T($j)`, `$j≠$i`, используя счётчики байтов `bytes_` стоков `sink($ij)`; в переменной `now` сохраняется текущее модельное время. Производительность каждой терминальной сети `T($i)` вычисляется в переменных `bw($i)`, затем вычисляется суммарная производительность в переменной `bwt`. Процедура завершается добавлением в расписание симулятора нового будущего события, состоящего в запуске процедуры `evalPerformance` через интервал времени `$time` (`$now+$time`). Полученные оценки выводятся в файл в формате строки, удобном для дальнейшего построения диаграмм и графиков. Фрагмент полученного файла представлен на Рис. 7.

```
5.0999 0.2800 0.4400 0.8000 0.7600 0.8800 0.5999 3.7599
5.1999 0.4000 0.7600 1.0000 0.5200 0.8000 0.6800 4.1600
5.2999 0.4799 0.5999 1.0000 0.4400 0.7600 0.4400 3.7200
5.3999 0.5999 0.5999 0.9599 0.4000 0.7600 0.8000 4.1200
```

Рисунок 7 – Фрагмент файла вычисленных оценок производительности сети

Первая колонка равна текущему модельному времени (`$now`); далее следуют оценки производительности по терминальным сетям в порядке возрастания их номеров; в последний столбец выводится оценка общей производительности. Заметим, что оценки в каждой строке получены для интервала времени `[$now-$time, $now]`.

4. Анализ результатов моделирования. Выполнено имитационное моделирование процессов в указанном фрагменте Европейской магистрали Интернент (Рис. 1). Оценивались

динамика изменения производительности магистрали (по терминальным сетям и общей) с помощью процедуры `evalPerformance`, приведенной на Рис. 6. Полученный график представлен на Рис. 8.

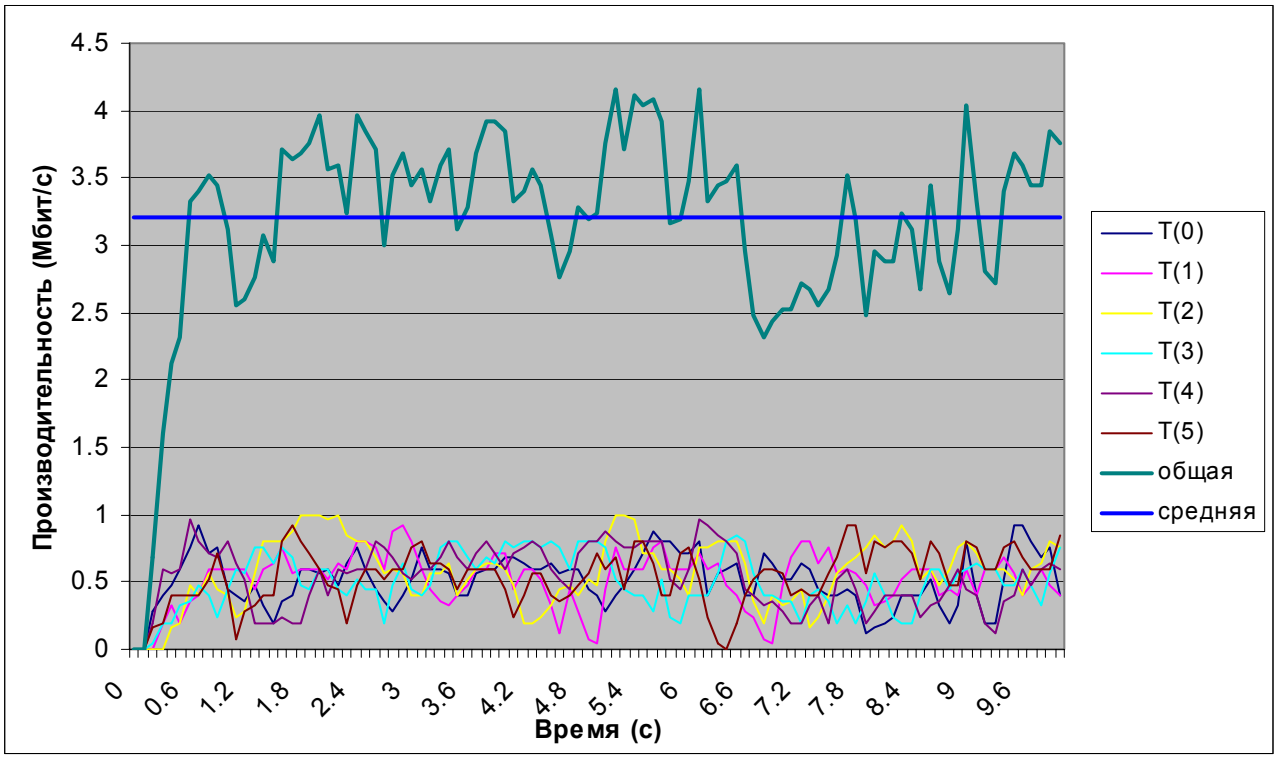

Рисунок 8 – Динамика изменения производительности сети во времени

Так как интенсивность трафика подвержена колебаниям в соответствии экспоненциальным распределением времени использованных генераторов, суммарная производительность изменяется в границах от 2,3 до 4,2 Мбит/с; на графике прямой линией представлена средняя производительность сети равная 3,3 Мбит/с. Указанная процедура может быть использована для определения временных перегрузок в магистральных сетях при моделировании всплесков трафика.

Далее исследовалась зависимость общей производительности от интенсивности источников трафика во времени; поверхность производительности сети представлена на Рис. 9.

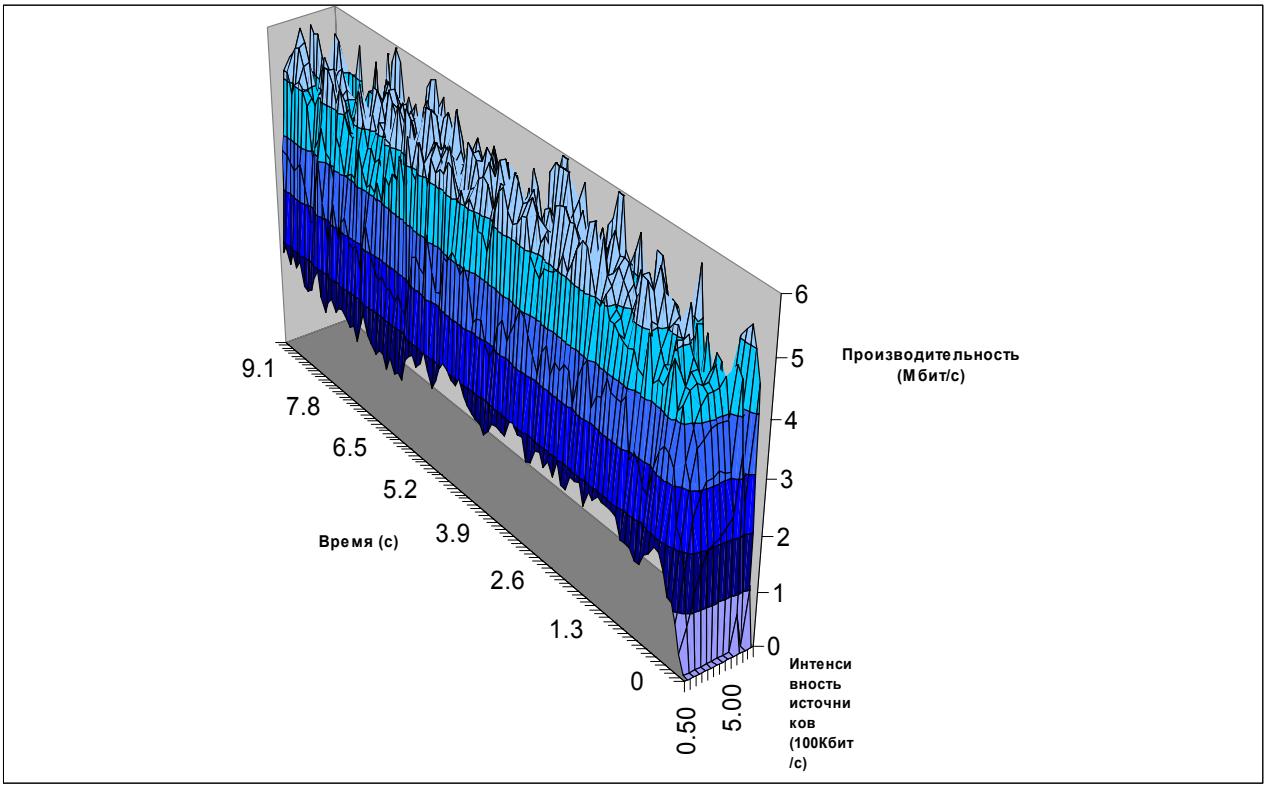

Рисунок 9 – Поверхность производительности сети

В среднем производительность растёт до максимальной производительности сети и стабилизируется, затем наблюдается снижение производительности ввиду потери пакетов в очередях. Рост очередей пакетов и потери пакетов могут быть оценены визуально в аниматоре NAM; пример образа экрана с отображением очередей и командные файлы оценки числа потерянных пакетов приведены в [3]. Указанную поверхность удобно оценивать в различных ракурсах для поиска аномалий производительности, вызванных неравномерностью трафика; цветное представление градации производительности повышает наглядность изображения.

Далее выполнены сравнительные оценки IP [3] и MPLS сетей при различных нагрузках сети, в том числе и пиковых; результаты представлены на Рис. 10.

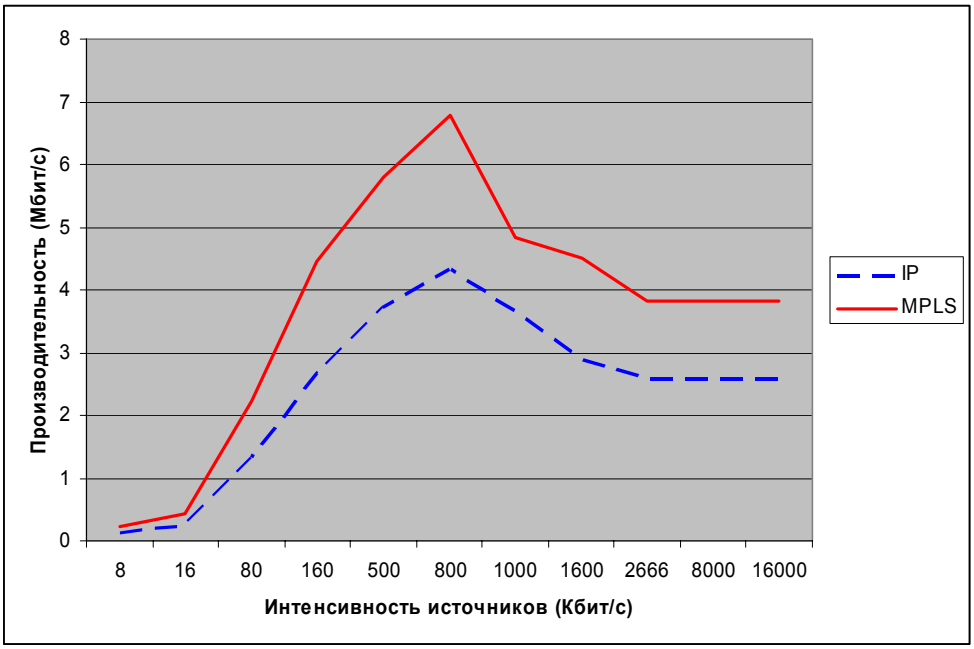

Рисунок 10 – Сравнение производительности IP и MPLS сетей

Исследования проводились для каналов с пропускной способностью 2Мбит/с и интенсивность источников трафика с постоянной битовой скоростью от 8 до 16000 Кбит/с. При интенсивности источников около 800 Кбит/с достигается максимальная производительность; при этом MPLS магистраль обеспечивает производительность в 1,6 раз больше, чем IP магистраль. Это косвенно подтверждает адекватность ранее построенных моделей в форме раскрашенных сетей Петри [4] (с погрешностью около 6%). Дальнейший рост интенсивности трафика приводит к снижению производительности из-за роста очередей и потери пакетов; указанная аномалия в [4] не наблюдалась, так как процессы потери пакетов и повторной передачи не моделировались.

Таким образом, в настоящей работе представлена методика построения моделей MPLS магистралей в среде моделирующей системы NS и динамической оценки пропускной способности магистралей с помощью специально разработанной процедуры, периодически оценивающей производительность сети в процессе имитационного моделирования, для обнаружения кратковременных перегрузок в магистралях. Полученные численные результаты хорошо согласуются с известными оценками производительности MPLS сетей.

Литература

1. FRC 3031: Multiprotocol Label Switching Architecture. E. Rosen, A. Viswanathan, R. Callon. January 2001, 61p.
2. The ns Manual / Ed.: Kevin Fall, Kannan Varadhan. – The VINT Project, 2006. – 414 p.
3. Зайцев Д.А., Шинкарчук Т.Н. Моделирование телекоммуникационных сетей в системе NS // Труды Одесской национальной академии связи им. А.С. Попова. - 2006, №2. – с. 35-43.
4. Зайцев Д.А., Сакун А.Л. Исследование эффективности технологии MPLS с помощью раскрашенных сетей Петри // Зв'язок. – 2006. – Т. 65, №5. – С. 49-55.