ОЦЕНКА КАЧЕСТВА ПРОХОЖДЕНИЯ ТРАФИКА ДЛЯ АЛГОРИТМОВ ВЗВЕШЕННОГО «СПРАВЕДЛИВОГО» ОБСЛУЖИВАНИЯ ОЧЕРЕДЕЙ

THE TRAFFIC PASSAGE QUALITY ESTIMATION FOR ALGORITHMS OF THE WEIGHED FAIR SERVICE OF QUEUES

Механов Виктор Борисович / Mekhanov Victor B.,
проректор Пензенского государственного университета /
The prorector of the Penza state university,
mvb@pnzgu.ru

Аннотация

Описывается методика оценки качества прохождения трафика путем моделирования с помощью аппарата временных иерархических цветных сетей Петри механизма взвешенного «справедливого» обслуживания очередей (WFQ) в компьютерных сетях с поддержкой качества обслуживании (QoS).

Abstract

The traffic passage quality estimation technique with modeling using the apparatus of temporary hierarchical colored Petri nets weighted mechanism "fair" queuing (WFQ) in computer networks with Quality of Service (QoS) is described.

Ключевые слова: качество обслуживания, взвешенная «справедливая» очередь, сеть Петри.

Keywords: quality of service, weighed fair queue, Petri net.

В работах [1,2] предложены общие принципы моделирования компьютерных сетей с поддержкой QoS в системе CPN Tools и приведено описание сетей Петри, моделирующих алгоритмы SPQ (Strict Priority Queues - строго приоритетный алгоритм), WRR (Weighted Round Robin – взвешенный циклический алгоритм), DRR (Deficit Round Robin – дефицитный циклический алгоритм), применяемые в современных коммутаторах Ethernet второго уровня. Алгоритмы взвешенного «справедливого» обслуживания очередей (Weighted Fair Queuing - WFQ), рассматриваемые в настоящей статье, имеют высокую сложность и, как правило, применяются только в маршрутизаторах, подключаясь на их низкоскоростные интерфейсы, или коммутаторах третьего уровня.

Алгоритмы WFQ обеспечивают «справедливое» (честное) разделение полосы между существующими потоками трафика. Для этого доступная полоса делится на число потоков и каждый получает либо равную часть полосы, либо ее определенную долю, определяемую весом. При этом в случае недогрузки выделенной полосы какими либо потоками, ее свободная часть автоматически распределяется между остальными. В случае перегрузок ненагруженные высокоприоритетные потоки функционируют без изменений, а низкоприоритетные высоконагруженные – ограничиваются. Широкое распространение алгоритмов WFQ связано с тем, что они обеспечивают прогнозируемую задержку и могут работать вместе с протоколом резервирования сетевых ресурсов (*Resource ReSerVation Protocol* – RSVP), который используется в Internet при запросе определенного уровня качества сетевых услуг для мультимедийных приложений [3].

Существует множество алгоритмов WFQ, реализованных различными производителями оборудования и имеющих существенные отличия. В данной статье рассматривается интерпретация алгоритмов «справедливого» обслуживания, используе-

мых в оборудовании Cisco [4], поддерживающем несколько разновидностей WFQ:

- основанную на потоках (Flow-Based WFQ - FBWFQ);
- основанную на классах (Class-Based WFQ - CBWFQ);
- основанную на очередности с низкой задержкой (Low Latency Queuing - LLQ).

Различие между алгоритмами FBWFQ и CBWFQ состоит в правилах формирования (классификации) очередей и назначения им весовых коэффициентов, механизм же обеспечения «справедливости» сформированных очередей аналогичен.

В FBWFQ трафик разбивается на потоки по значениям IP адресов и/или портов источника и назначения с учетом маркировки в соответствующем поле заголовка (*IP Type of Service* - ToS). Если потоков больше, чем очередей, то в одну очередь помещается несколько потоков. Веса, присвоенные потокам, однозначно определяются значением приоритета, содержащегося в ToS, и обратно пропорциональны приоритету потока.

В CBWFQ весь трафик разбивается на классы на основании либо так называемых групп QoS, соответствующих набору признаков из списка управления доступом, например, номеру входного интерфейса или номеру хоста или подсети, либо значений полей ToS. Для варианта групп QoS веса пропускной способности задаются администратором. Пакеты, не отнесенные ни к одной из групп, включаются в группу 0, которой выделяется оставшаяся после назначения весов полоса. В варианте классификации на основании значений ToS предусматриваются как назначенные администратором веса классов, так и веса по умолчанию,

Алгоритм LLQ представляет собой комбинацию строго приоритетного обслуживания для одного/двух потоков с алгоритмами FBWFQ и CBWFQ - для остальных.

В основе алгоритма WFQ лежит алгоритм FQ, в соответствии с которым все потоки трафика рассматриваются как равные между собой, т.е. имеющие одинаковый вес. При этом алгоритм FQ эмулирует работу некоторого идеального планировщика (Generalized Processor Sharing – GPS), используя принцип вычисления порядкового номера каждого пакета.

Каждый кадр при постановке в очередь получает специальную служебную метку, называемую его номером, который определяет относительный порядок обработки пакетов. Очередь считается активной, если она в данный момент содержит хотя бы один пакет соответствующего потока, в противном случае очередь будет пассивной. При постановке пакета в пассивную очередь его порядковый номер равен сумме порядкового номера последнего обслуженного планировщиком пакета и размера входящего пакета в байтах, а для случая активной очереди - сумме порядкового номера последнего находящегося в данной очереди пакета и размера входящего пакета в байтах.

В алгоритмах WFQ веса очередей учитываются при расчете номера умножением размера пакета на соответствующий весовой коэффициент, правило формирования которого для FBWFQ строго определено: 4096/(1+ IPP), где IPP - приоритет потока (трехбитное поле IP Precedence в ToS), а в остальных версиях алгоритма - задается администратором.

Учитывая особенности реализации версий алгоритмов WFQ, представляется целесообразным подробно рассмотреть сеть Петри, моделирующую алгоритм FQ, которая может служить базой для различных весовых модификаций. Для удобства проектирования, верификации, ана-

лиза функционирования модели сети, в настоящей статье реализуется подход к построению сети Петри, аналогичный известному в вычислительной технике принципу разделения цифрового устройства на операционные и управляющие блоки, взаимодействующие посредством управляющих и осведомительных сигналов [5].

Модель сети Петри коммутатора на верхнем уровне иерархии показана рис. 1 и включает в себя:

- подсети ***InPort*** и ***OutPort***, которые моделируют порты коммутатора, обеспечивающие доступ к линиям связи в полнодуплексном режиме;

- подсеть ***Switch***, моделирующую продвижение пакета с входного порта на определенный выходной порт в соответствии с таблицей коммутации, принцип ее функционирования приведен в [6];

- подсети ***Buf***, которые моделируют буфера пакетов, направленных в соответствующий выходной порт. Эти подсети собственно и реализуют механизмы QoS, поэтому подсеть Buf будет рассматриваться дальше.

Как было предложено в [5], при моделировании использовались два типа маркеров:

1) Информационные маркеры, продвижение которых имитирует обработку пакета в коммутаторе:

- множество цветов ***frm***, отражающих структуру передаваемого пакета и состоящих из цветов адресов получателя ***dst*** и отправителя ***srs***, управления приоритетом ***qos***, размера передаваемых данных ***szfrm***, и суммарной задержки ***delay***, используемой для подсчета задержки продвижения пакета по сети;

- множество цветов ***frame***, эмулирующих однонаправленный канал, который может быть либо занят передачей пакета ***(frm)***, либо свободен ***(avail)***, что позволяет выявлять и обрабатывать события, связанные с наличием или отсутствием пакета (в CPN Tools отсутствуют ингибиторные дуги).

2) Управляющие маркеры, цвет которых отображает состояние обработки потока пакетов в коммутаторе: размер очереди, количество переданных данных, признак очистки очереди, номер передаваемого пакета и др.

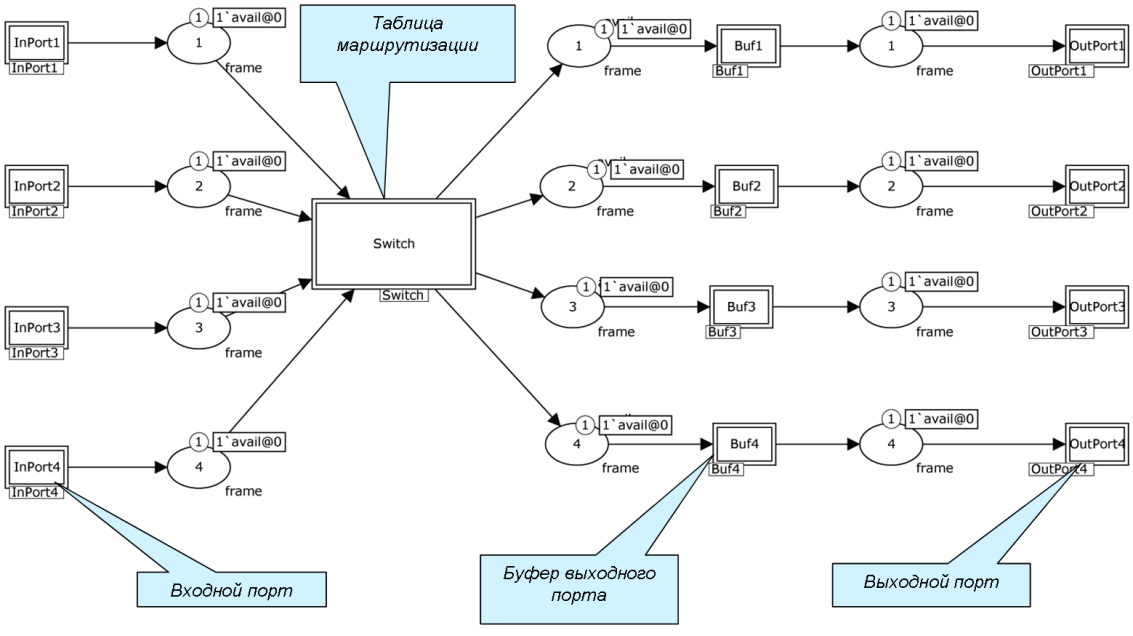

Рис. 1. Модель 4-х портового коммутатора

Сеть буфера выходного порта для алгоритмов «справедливого» обслуживания очередей показана на рис. 2 и включает в себя подсети:

- ***Classifier*** – подсеть классификатора трафика, который на основании значения признака требуемого уровня QoS помещает этот пакеты в очередь определенного класса.

- ***Queue*** – очередь типа FIFO, буферизирующая пакеты одного класса. В этой подсети также реализованы подсчет размера текущей очереди и отбрасывание поступивших пакетов при переполнении буфера.

- ***Scanner*** – подсеть, реализующая алгоритм «справедливого» выбора очереди для обслуживания.

Указанные подсети связаны между собой и внешним окружением через следующие контактные позиции:

- ***Buffer in*** и ***Buffer out***, через которые выполняется продвижение информационных маркеров (пакетов);

- ***OutPort Free***, которая отображает состояние выходного порта коммутатора, наличие маркера в ней разрешает старт выбора очереди на обслуживание пакета;

- ***Start selection***, наличие маркера в которой разрешает выборку очередного пакета из выбранной очереди;

- ***Size queue***, которая содержит значение текущего размера очереди;

- ***LN***, содержащая список номеров пакетов, стоящих в очереди;

- ***SN***, содержащая номер последнего обрабатываемого пакета.

Модель классификатора, как и в [2], выполняет анализ приоритетного поля ***qos*** в поступившем пакете и направление его в одну из очередей, но структура подсети, моделирующая очередь ***Queue***, имеет серьезные отличия от принятой в алгоритмах SPQ и RR.

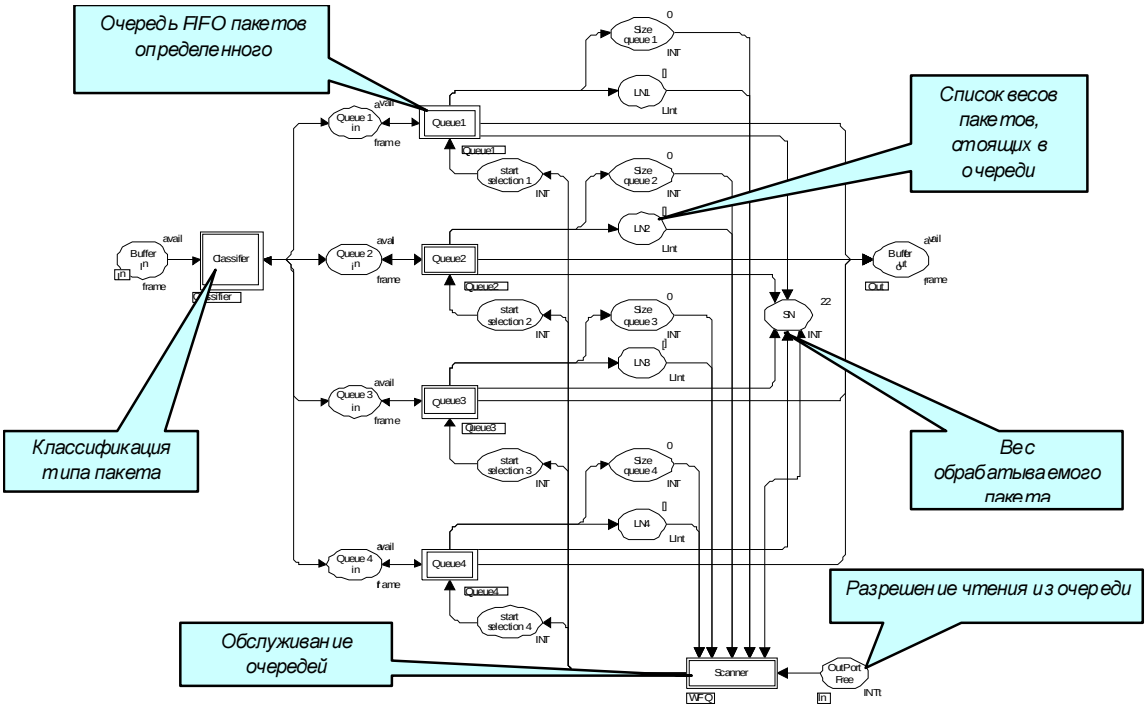

Рис. 2. Модель буфера выходного порта

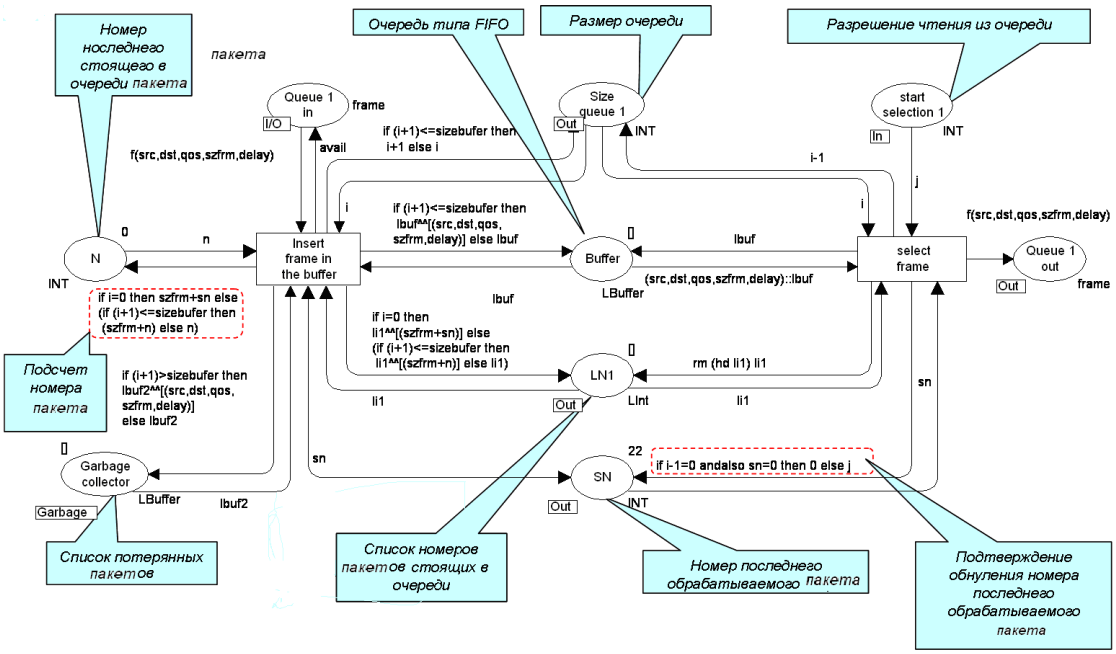

Рис. 3. Подсеть очереди

Каждая подсеть очереди ***Queue*** (рис. 3) содержит следующие позиции:

- ***Queue in*** и ***Queue out*** - входной и выходной порты очереди;

- ***Buffer*** - список пакетов, ожидающих обработки;

- ***Garbage collector*** - список потерянных пакетов;

- ***N*** - номер пакета, стоящего в очереди последним;

- ***Size queue, LN, SN*** – аналогично описанным выше (позиция ***SN*** является общей для всех очередей).

При срабатывании перехода ***Insert frame in the buffer*** выполняется занесение пакета в очередь, пересчет заполнения очереди и, в случае отсутствия в очереди места, сброс маркера пакета в ***Garbag collector***. Чтение из очереди выполняется срабатыванием перехода ***Select frame*** по приходу в позицию ***Start selection*** разрешающего маркера из подсети ***Scan***. Все эти операции аналогичны описанным в [1,2]. При этом для описания очереди используется функция языка CPN ML «список», позволяющая содержать в позиции ***Buffer*** только один маркер, который является списком пакетов. В начальной маркировке список пустой.

При постановке пакета в очередь выполняется формирование его номера с использованием позиций ***N, LN*** и ***SN***.

При срабатывании перехода ***Insert frame in the buffer*** одновременно с записью информационного маркера в список позиции ***Buffer*** выполняется вычисление номера принимаемого в очередь пакета по правилу:

if i=0 then (szfrm+sn) else (if (i+1)<=sizebufer then (szfrm+n) else n,

где ***i*** – текущий размер очереди (значение цвета маркера позиции ***Size queue***);

szfrm – длина обрабатываемого пакета;

n – номер предыдущего пакета в очереди (значение цвета маркера позиции N);

sizebufer – максимально допустимый размер очереди.

Вычисленный номер фиксируется в значении цвета маркера позиции ***N*** и записывается в конец списка, хранящегося в позиции ***LN***.

Выборка маркера из очереди в позиции ***Buffer*** и передача его в вы-

ходной порт выполняется при срабатывании перехода ***Select frame*** при наличии разрешающего маркера в позиции ***Start selection***, имеющего цвет ***j***. При этом корректируется значение размера очереди в позиции ***Size queue***, а так же выполняется извлечение первого элемента из списка, хранящегося в позиции ***LN***, и изменение цвета маркера в позиции ***SN*** по правилу:

if i-1=0 andalso sn=0 then 0 else j ,

где ***i*** – текущий размер очереди;

sn – номер обработанного пакета (значение цвета маркера позиции ***SN***;

j – номер извлеченного из очереди пакета, переданный из подсети сканирования маркером ***Start selection***.

Данное правило обеспечивает подтверждение обнуления значения цвета маркера в позиции ***SN***, иначе в позицию ***SN*** будет записываться номер выбранного из очереди пакета, даже если все очереди станут пустыми (необходимость этой операции будет пояснена ниже).

Подсеть ***Scanner***, реализующая алгоритм «справедливого» выбора, показана на рис. 4. Она содержит один переход, разрешение на срабатывание которого дает маркер в позиции ***Outport Free***. Этот маркер вырабатывается после освобождения выходного порта коммутатора очередным пакетом, при его формировании учитываются задержка сериализации, преамбула и межпакетный интервал.

Если выходной порт коммутатора свободен, то условием срабатывания перехода ***Scanner*** является отличие размера хотя бы одной очереди от нуля. При срабатывании перехода для выдачи маркера разрешения выборки пакета определяется очередь, номер головного (первого) пакета, которой имеет наименьшее значение. Это выполняется путем сравнения первых записей списков позиций ***LN*** каждой из очередей между собой. При сравнении используется пользовательская функция ***WF***, которая определяется средствами CPN ML следующим образом:

fun WF(li:LInt)= if (length li=0) then 1073700000 else hd li,

где ***li*** – список цвета ***LInt***, являющийся входным параметром функции;

hd li –головной элемент списка.

Функция работает следующим образом: если длина списка, подаваемого на вход функции, равна 0, то функция возвращает «очень большую» константу 1073700000, иначе - значение головного элемента из списка ***li***. Этот искусственный прием позволяет избежать некорректной работы сети при пустом списке и переполнение разрядной сетки при моделировании в CPN Tools длительных реализаций трафика.

Рассмотрим для примера условие выдачи разрешения выборки пакета из второй очереди:

if (length li2)>0 andalso (WF(li2))<(WF(li1)) andalso (WF(li2))<(WF(li3)) andalso (WF(li2))<(WF(li4)) then 1`WF(li 2) else empty

Данное условие означает, что если вторая очередь не пустая и номер ее первого элемента является наименьшим для всех очередей, то в качестве значения маркера, разрешающего выборку пакета из этой очереди, указывается номер первого пакета второй очереди, иначе маркер, разрешающий выборку, не создается.

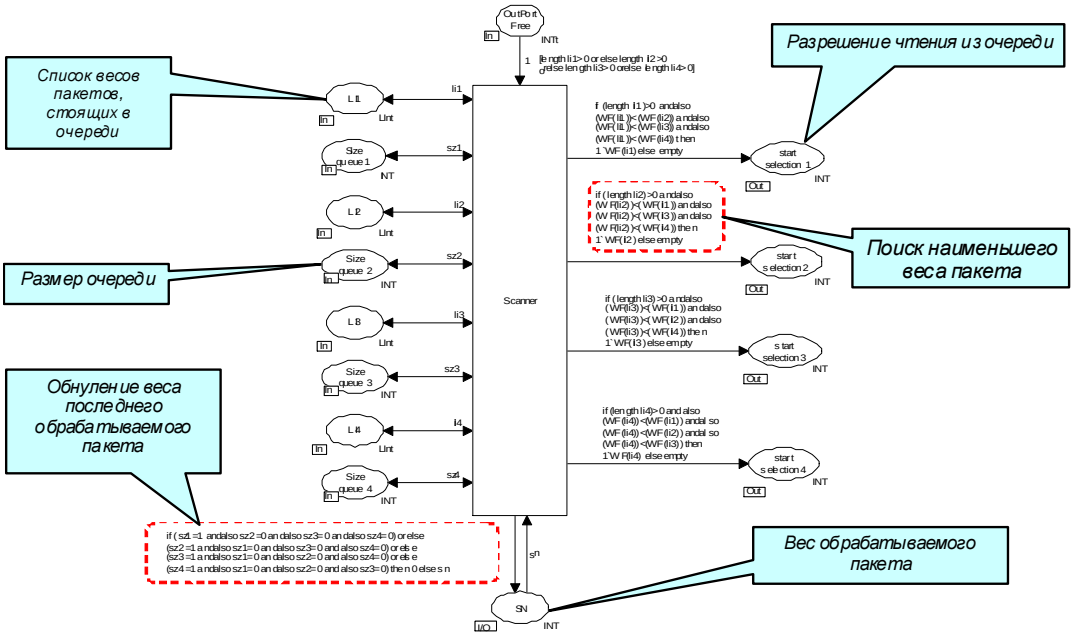

Рис. 4. Подсеть обслуживания очередей

Так как значение цвета маркера в позиции ***SN*** постоянно увеличивается, то может настать момент, когда оно превысит максимальное значение, определяемое длиной разрядной сетки инструментальной ЭВМ. Для предотвращения переполнения при срабатывании перехода ***Scanner*** вычисляется выражение, определяющее момент, когда все очереди станут пустыми, в этом случае сбрасывается в нуль значение цвета маркера в позиции ***SN***, т.е. нумерация пакетов начнется сначала.

Из рассмотренных моделей алгоритма FQ легко могут быть получены модели взвешенных алгоритмов «справедливого» обслуживания очередей, для чего при срабатывании перехода ***Insert frame in the buffer*** вычисление номера принимаемого в очередь пакета должно выполняться по следующему правилу:

if i=0 then (szfrm* Km+sn) else (if (i+1)<=sizebufer then (szfrm* Km +n) else n,

где ***Km*** – весовой коэффициент очереди, значение которого определяется реализуемым механизмом формирования весов. Вычисление коэффициентов целесообразно оформить в виде пользовательской функции.

В качестве примера ниже приведены результаты исследования джиттера трафика в сети Ethernet с равновесным справедливым алгоритмом обслуживания очередей. С помощью специальной сети Петри моделировался входной трафик, который представляет собой пуассоновский поток пакетов, показатели качества которых с равной вероятностью принимают значения от 0 до 7, длительность пакетов равномерно распределена в интервале от 80 до 1500 байт. Трафик разделялся на четыре потока (в поток 1 объединялись пакеты с приоритетами 7/6, поток 2 – 5/4, поток 3 – 3/2, а поток 4 – 1/0). Джиттер определялся как время ожидания пакета в очереди.

На рис. 5 представлен профиль трафика по параметру «джиттер», который измеряется в тактовых интервалах (TI) Ethernet для случая загрузки выходного порта равной 80%. По профилю вычислялись оценки функции распределения джиттера для каждого потока (рис. 6).

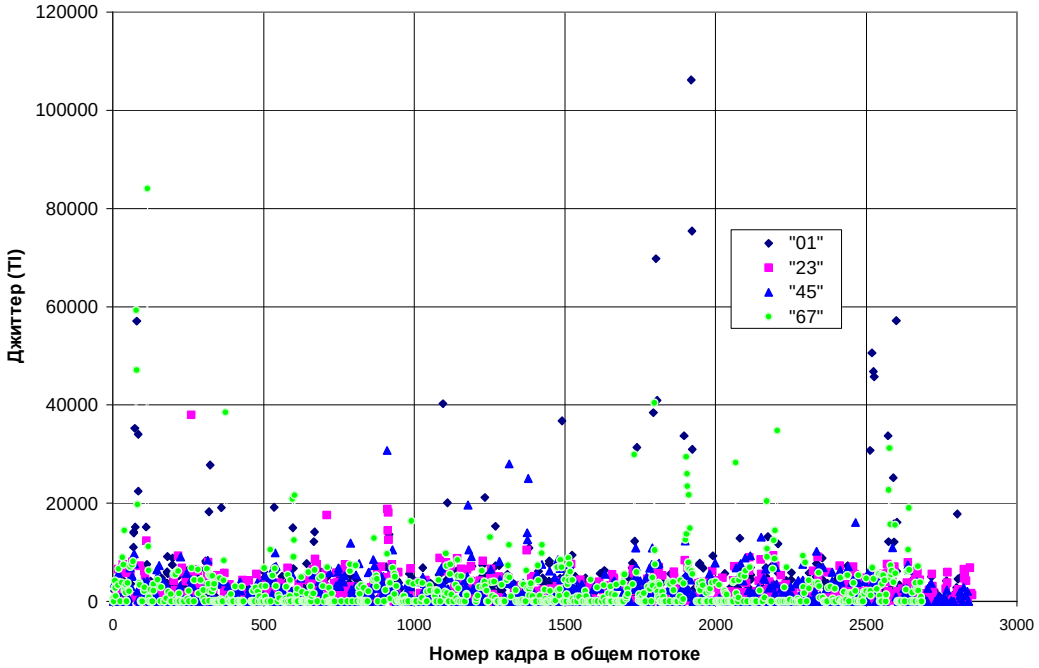

Рис. 5. Профиль джиттера

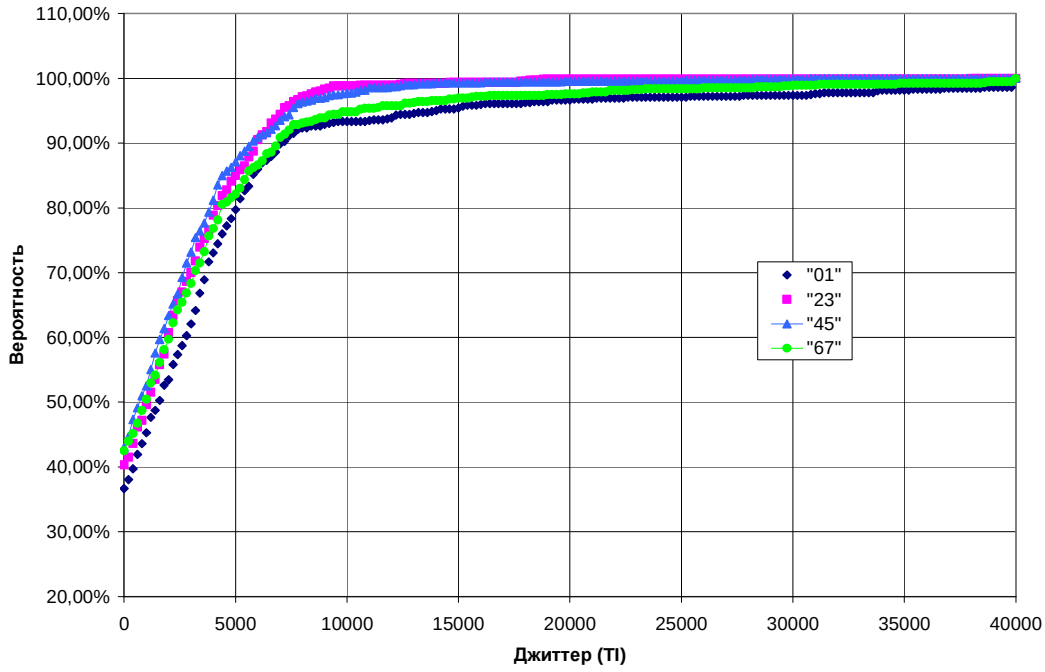

Рис. 6. Функция распределения джиттера

Таким образом, разработанный подход позволяет оценить вероятностно-временные характеристики трафика при алгоритмах взвешенного «справедливого» обслуживания очередей, наиболее распространенных в IP сетях. Эти характеристики, в свою очередь, можно использовать в качестве компонентов для исследования эффективности службы QoS в корпоративных сетях.

Литература

1. Механов В. Б. Моделирование сетями Петри алгоритмов обслуживания очередей // Материалы международного симпозиума «Новые информационные технологии и менеджмент качества (NIT&MQ’2011)» «ФГУ ГНИИ ИТТ “Информика». - М.: ООО «Арт-Флеш», 2011. –С. 8–11.

2. Механов В. Б. Моделирование алгоритмов обслуживания очередей в сетях с поддержкой QOS // Информатизация образования и науки, 2011, № 4(12) – с. 29 – 38

3. Олифер В. Г., Олифер Н. А. Компьютерные сети. Принципы, технологии, протоколы. 4-е изд. – СПБ: Изд-во «Питер», 2010, - С. 943.

4. Вегешна Ш. Качество обслуживания в сетях IP. - М.: Изд-во «Вильямс», 2003. - С. 368.

5. Механов В. Б. Применение сетей Петри для моделирования телекоммуникаций с поддержкой качества обслуживания. // Труды XVII Всероссийской научно-методической конференции «Телематика'2010», Том 2. СПб: СПбГУ ИТМО, 2010. – С. 283–285.

6. Zaitsev D. A. Switched LAN simulation by colored Petri nets // Mathematics and Computers in Simulation, vol. 65, № 3, 2004. - P. 245–249.