

ОСОБЕННОСТИ ОПТИМАЛЬНОГО ВЫБОРА АЛГОРИТМОВ МЕЖМОДУЛЬНОГО ВЗАИМОДЕЙСТВИЯ КОМПОНЕНТ РАСПРЕДЕЛЕННЫХ ПРОГРАММНЫХ СИСТЕМ

Е.Н. Хабаров, О.Я. Кравец

В статье приводится описание и анализ эффективности работы алгоритмов активного управления и борьбы с перегрузками: RED, FRED, Blue и SFB и на основе имитационного моделирования производится сравнение с базовым алгоритмом Tail Drop

Ключевые слова: затор, перегрузка, взаимодействие

Введение

В процессе распределенного взаимодействия компонент программных систем распространенной ситуацией является случай (перегрузка), когда сумма пропускной способности входящих каналов от нескольких источников на транзитном модуле превосходит полосу пропускания исходящего канала по направлению к потребителю или потребителям. Для того, чтобы избежать потери информации, промежуточные модули используют механизм буферизации.

Разработано множество алгоритмов определения и уведомления о перегрузках (АУП) [1], все их можно разделить на два основных вида — с явным уведомлением о перегрузке и неявным. В случае явного уведомления по каналу связи отправляются специальные служебные пакеты, сигнализирующие о перегрузке, в то время как алгоритмы на основе неявного уведомления уменьшают скорость передачи на основе косвенных факторов, напр. процент потерь, отсутствие уведомления о получении, задержка и т. д. Алгоритмы активного управления и избежания перегрузками (ААУИП) в свою очередь используются для превентивного предотвращения заторов. Чаще всего эти алгоритмы измеряют размер буфера исходящего канала и уничтожают пакеты данных по достижении определенного порога до того, как буфер будет переполнен трафиком источника. Также как и в случае АУП могут использоваться специальные сигнальные пакеты, сигнализирующие источнику необходимость снижения скорости отправки данных. Примерами алгоритмов ААУИП можно назвать RED[1,2], FRED[3], WRED, Blue[4], SFB[5], CBT[6], CHOCe[7]. В этой статье будет произведен анализ несколь-

ких алгоритмов на основе данных имитационного моделирования в системе ns-2 по следующим параметрам: полнота загрузки канала связи (а), справедливость разделения ресурсов между различными потоками данных (б), сложность реализации алгоритма в плане вычислительной сложности и объема кода (в).

Random Early Detect (RED)

Алгоритм RED разработан для решения нескольких основных задач: минимизация потери пакетов и задержки в очереди (а), избежание эффекта «глобальной синхронизации» (б), поддержание предельной загрузки канала связи (в). Основная идея ААУИП RED — уничтожать пакет с целью уведомить источник трафика в процессе зарождения затора о необходимости снизить скорость отправки данных до того, как очередь будет переполнена и все последующие пакеты будут отброшены. С этой целью RED использует так называемое значение экспоненциально-взвешенного скользящего среднего (EWMA) длины очереди или среднюю длину очереди, которое используется для определения перегрузки и знаменатель вероятности маркировки (MPD), который используется для определения вероятности отбрасывания пакета. Когда средняя длина очереди превосходит заданный порог ($\min_{th}$), RED начинает случайным образом отбрасывать пакеты или отправлять ECN уведомления о перегрузке. При достижении порога ($\max_{th}$) весь трафик маркируется или отбрасывается. При этом вероятность отбрасывания/маркировки пакета в интервале $\min_{th} - \max_{th}$ растёт от 0 до $1/MPD*100\%$.

Хотя RED и является значительным улучшением по сравнению со стандартным поведением отбрасывания хвоста, этот ААУИП имеет несколько недостатков. Основная проблема RED заключается в том, что RED полагается только на значение длины очереди при определении факта перегрузки. Хотя постоянное присутствие данных в очереди и сигнализирует о наличии затора, длина очереди дает очень мало информации о его природе

Хабаров Евгений Николаевич - ВГТУ, аспирант, e-mail: enkhbarov@gmail.com

Кравец Олег Яковлевич - ВГТУ, д-р техн. наук, профессор, e-mail: csit@bk.ru

Для целей справедливого разделения ресурсов были разработаны различные алгоритмы управления очередями Fair Queue [8], Weighted Fair Queue [8], но их основной целью не является избежание перегрузок. Существуют также различные модификации базового алгоритма RED, напр. Weighted RED, который обеспечивает различные пороговые значения и MPD для пакетов с разными уровнями обслуживания.

Flow Random Early Detect (FRED)

Алгоритм FRED пытается решить проблемы RED, поддерживая информацию о состоянии активных потоков трафика с целью обеспечить различные уровни отбрасывания пакетов для соединений с различными требованиями к пропускной полосе. Очевидно, что этот ААУИП требует дополнительную память и процессорное время для хранения и обработки информации о потоках. FRED поддерживает информацию только о тех соединениях, пакеты которых находятся в буфере очереди, т.о. требования к дополнительной памяти в FRED напрямую зависят только от размера буфера очереди, а не от количества соединений. Другими важными особенностями FRED являются возможность наложения штрафных санкций на неадаптивные агрессивные потоки с помощью счетчика $\text{strike}(i)$, подсчитывающего количество неудачных попыток повлиять на пропускную полосу, занимаемую потоком, а i – id потока, (а), защита пульсирующих потоков данных от ошибочного отбрасывания пакетов путем приоритизации их на основе расчета отклонения длинны очереди потока $q(i)$ от значения avg_{cq} – средней длины очереди среди всех потоков (потоки с меньшим значением относительно avg_{cq} приоритизируются) (б).

Blue

Алгоритм Blue в отличие от алгоритмов на основе RED не использует значение q длины очереди. Вместо этого в Blue используется адаптивный параметр вероятности отбрасывания p_m , на основе которого принимается решение об отбрасывании пакета данных при поступлении пакета в очередь. Если в процессе работы происходит перегрузка, и пакеты начинают отбрасываться т.к. в буфере больше нет места для них (происходит процесс отбрасывания хвоста), Blue увеличивает параметр p_m , увеличивая тем самым частоту отбрасывания пакетов или частоту отправки явных уведомлений о перегрузке. Если очередь становится пустой, Blue уменьшает значение p_m . Такой метод позволяет этому ААУИП адаптироваться к различным сценариям перегрузки, обучаясь нужной частоте отбрасывания пакетов. Помимо p_m , Blue ис-

пользует параметры $d1$, $d2$ и freeze-time, где freeze-time определяет минимальный временной интервал между двумя изменениями значения p_m . Параметры $d1$ и $d2$ являются шагами увеличения и уменьшения p_m соответственно.

Stochastic Fair Blue (SFB)

SFB основан на алгоритме Blue и предназначен для решения проблем, присущих Blue. Как и базовый алгоритм RED, Blue не обеспечивает справедливое разделение пропускной полосы среди различных потоков данных, и в частности защиту импульсивных и короткоживущих потоков от агрессивных потоков данных. Как и Blue, SFB использует похожий механизм определения нужной вероятности отбрасывания пакета. Помимо этого в SFB используется структура данных размером $L \times M$, где L – количество уровней, а N – количество контейнеров на каждом уровне. Помимо этой структуры данных, SFB использует L -хэш функций по количеству уровней. Каждая хэш-функция L имеет своей целью определение соответствия потока данных контейнеру N на уровне L . Каждый контейнер содержит информацию о количестве пакетов в очереди, относящихся к этому контейнеру. Если количество пакетов в очереди превосходит некоторый порог (напр., размер контейнера – bin_size), SFB изменяет p_m для данного контейнера аналогично Blue. Следует заметить, что агрессивные потоки данных могут быстро свести вероятность отбрасывания пакетов данных, относящихся к своему контейнеру до 100%. Хотя к одному контейнеру могут относиться как агрессивные, так и адаптивные потоки, а адаптивные потоки могут разделять один-два контейнера с агрессивными потоками, это не является условием, что пакеты адаптивного потока будут отбрасываться с чрезмерной частотой.

Сравнение эффективности

Для сравнения эффективности ААУИП произведено имитационное моделирование в системе ns-2 [9] работы алгоритмов FRED, Blue, SFB и проведено их сравнение с результатами, полученными при моделировании работы стратегии «отбрасывания хвоста» и RED. Механизм ECN для этих целей был отключен, таким образом маркировка пакета означает его отбрасывание. В качестве топологии моделируемой сети использовалась схема, в которой N источников данных обмениваются с N получателями через два промежуточных узла связи. Канал связи между промежуточными узлами является разделяемым ресурсом, полоса пропускания которого не является достаточной для обмена данными без использования очереди.

Параметры этого канал связи – 1 Мбит/с – полоса пропускания, 100 мсек – задержка. Всего было проведено два основных варианта эксперимента с различными параметрами, характеризующимися различным поведением трафика и количеством потоков данных. В качестве источника трафика для первого выбраны 5 TCP-потоков (обычное FTP-приложение) и одно UDP-приложение с постоянной скоростью передачи (CBR) 500000 бит/с, которое запускается по истечении 60 сек. после начала эксперимента, общая длительность которого 3 мин. Для TCP выбрано значение окна в 50 пакетов, а буфер очереди на разделяемом канале связи способен хранить до 100 пакетов по 1000 байт, т.е. размер буфера – 100Kb. Значения параметров RED $\max_{th}$ и $\min_{th}$ выбраны соответственно 30% и 70% (30 и 70 пакетов соответственно для простоты). Во втором эксперименте использовалось 2 потока данных TCP и 6 потоков UDP с постоянной скоростью передачи каждый – 200000 бит/с, что позволило создать затор в основном не адаптивными потоками UDP. На графиках сплошной линии соответствует протокол TCP, точечному пунктиру соответствует UDP, а штриховому – сумма пропускной способности TCP и UDP.

Основными изменяемыми параметрами эффективности в проеденных экспериментах являются длина очереди и используемая полоса пропускания. Пропускная способность отдельных потоков измерялась с целью выяснить справедливость разделения ресурсов различными ААУИП. Средняя длина очереди является также важным показателем использования ресурсов маршрутизатора.

Для RED параметры $w_q=0.002$ и $\max_q=0.02$, для FRED параметр $\min_q$, который зависит от размера буфера на транзитных устройствах, было выбрано значение 2, а источник TCP-трафика отправляет не более 3 пакетов: 2 из-за используемых ACK и 1 из-за увеличения размера окна.

Для Blue выбраны следующие параметры: $d1=0.02$, $d2=0.002$, $freeze_time=0.01s$.

Параметры SFB выбраны следующим образом: $d1=0.005$, $d2=0.001$, $freeze_time=0.001s$, $Boxtime=0.05 s$, $Hinterval=5$, $bin_size = 1,5/N$. N и L выбраны в соответствии с количеством потоков трафика.

На рис. 1-5 изображены основные результаты эксперимента. Все 4 алгоритма показали достаточно высокую общую загруженность канала связи в среднем на уровне, близком к полной загрузке, за исключением алгоритма SFB, что вызвано его характерной особенностью ли-

митирования пропускной способности агрессивных потоков. Результаты второго эксперимента показали, что алгоритмы RED и Blue не обеспечивают справедливого разделения полосы пропускания в условиях, когда поток UDP трафика занимает более 45% пропускной способности, более того на рис. 2 видно, что в случае, если пропускная полоса, требуемая для передачи UDP-потоков превышает доступную полосу канала связи, алгоритмы RED и Blue не способны предоставить потокам TCP сколько-либо достаточную пропускную полосу и их характеристики распределения пропускной полосы становятся сравнимыми с алгоритмом отбрасывания хвоста. Результаты эксперимента также позволили сделать вывод, что загруженность очереди напрямую характеризует использование пропускной полосы.

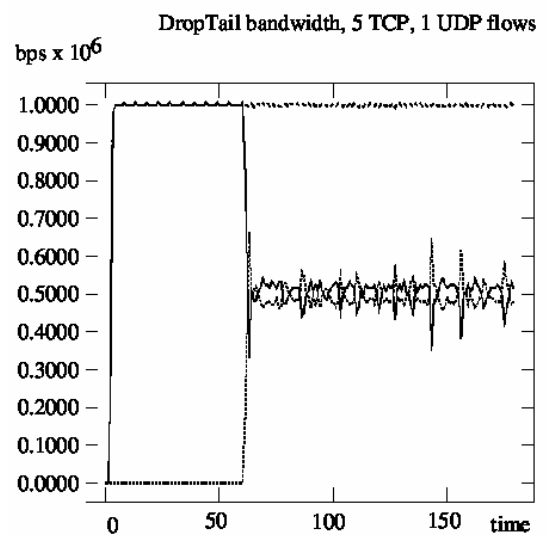


Рис. 1. График загрузки полосы пропускания при использовании алгоритма TailDrop

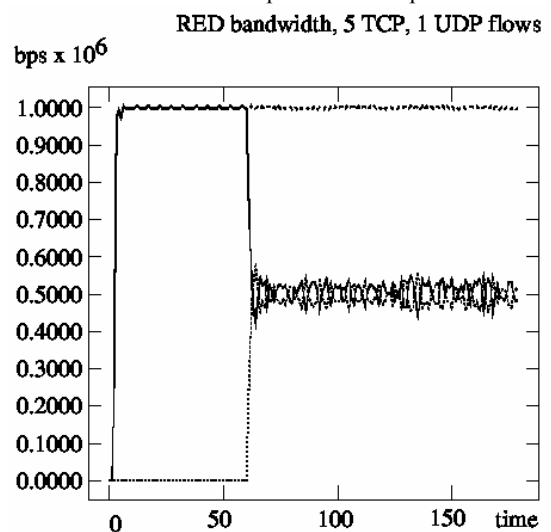


Рис. 2. График загрузки полосы пропускания при использовании алгоритма RED

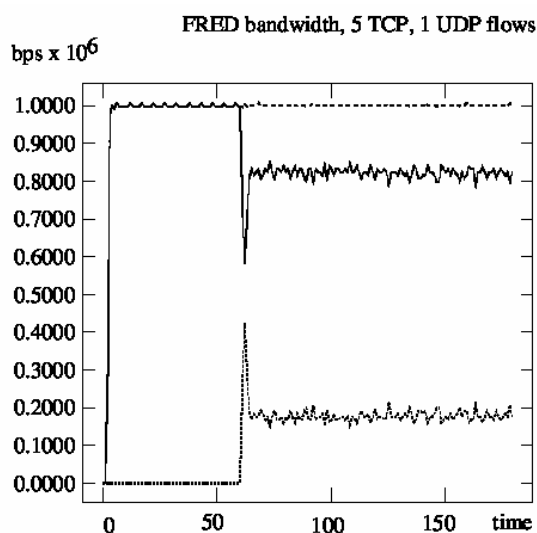


Рис. 3. График загрузки полосы пропускания при использовании алгоритма FRED

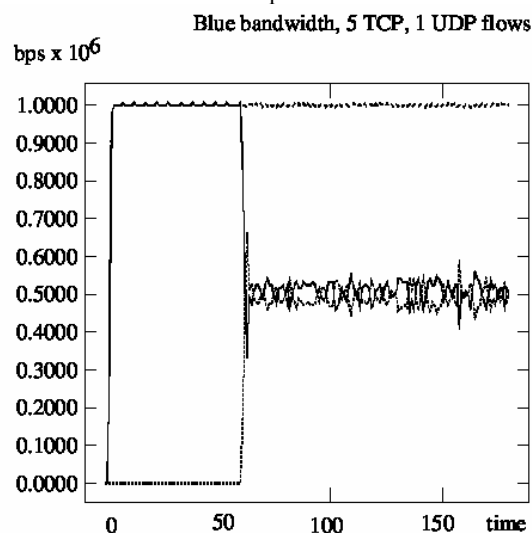


Рис. 4. График загрузки полосы пропускания при использовании алгоритма Blue

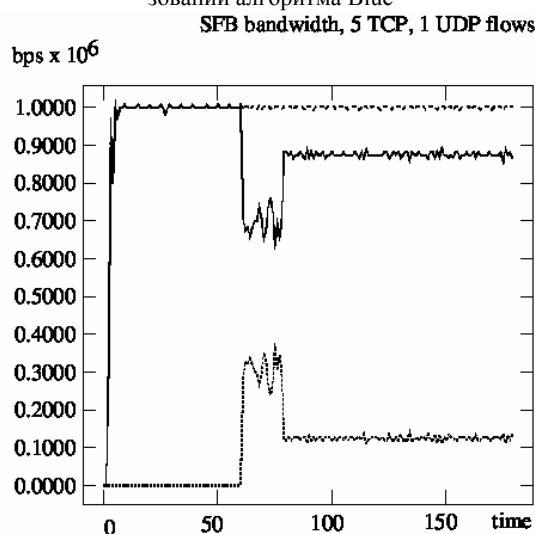


Рис. 5. График загрузки полосы пропускания при использовании алгоритма SFB

Как видно из табл. 1, алгоритм «отбрасывания хвоста» вызывает загруженность буфера

очереди до предельных значений и остается на нем. RED хотя и пытается устранить перегрузку, повлиять на потоки UDP он не может и как следствие загрузка очереди остается на высоком уровне, минимальное использование ресурсов буфера очереди показал алгоритм SFB. Важным преимуществом Blue является возможность управлять затором с участием не адаптивных потоков, используя минимальные ресурсы. Как видно из графика, Blue использует минимальные размеры очереди в процессе своей работы, однако при появлении неадаптивного потока UDP, эффективность Blue значительно снижается ввиду того, что алгоритм каждый раз при попытке стабилизировать очередь в условиях перегрузки начинает отбрасывать весь трафик, затем очередь снова достигает своих пределов и процесс повторяется вновь. Результат этого — всплески на графике длины очереди. Также нельзя не отметить сходство графиков загрузки очереди Blue и TailFrog в первые 60 с до появления неадаптивного потока UDP, TCP пытается занять максимальную пропускную полосу в связи с этим загрузка очереди остается на высоком уровне.

Таблица 1. Средняя длина очереди

Алгоритм/ Эксперимент	TailDrop	Red	Fred	Blue	SFB
1, TCP	51.257	22.860	28.726	35.520	10.335
1, UDP	42.192	19.383	6.425	18.175	2.225
1, Сумма	93.449	42.244	35.151	53.695	12.560
2, TCP	5.972	6.050	10.022	5.330	2.397
2, UDP	94.908	81.900	28.683	36.708	5.742
2, Сумма	100.880	87.950	38.706	42.038	8.138
3, TCP	-	-	-	-	1.240
3, UDP	-	-	-	-	1.850
3, Сумма	-	-	-	-	3.090

В процессе моделирования SFB были подтверждена эффективность этого алгоритма, для единичных случаев агрессивных потоков этот ААУИП идентифицирует их и применяет к ним штрафные санкции, однако полоса пропускания для этих потоков значительно занижена, такая ситуация может быть исправлена путем более точной подстройки параметра VoxTime.

Заключение

В статье приведено описание и последующий анализ ААУИП RED, FRED, Blue, SFB на основе имитационного моделирования, были представлены результаты моделирования и сравнение результатов. В результате установлено, что базовый алгоритм RED имеет ограни-

ценную применимость и его использование в условиях распределённого взаимодействия не всегда оправдано, в то же время такие алгоритмы, как Blue и в особенности SFB показали, что эффективные методы управления заторами возможны, однако требуют аккуратно подходить к выбору параметров алгоритма, их исследование, модификация и оптимизация - тема для отдельного исследования.

Литература

1. Floyd S., Jacobson V. Random early detection gateways for congestion avoidance// IEEE/ACM Transactions on Networking, vol. 1, Aug, 1993, p. 397-413.
2. Balakrishnan H., Padmanabhan V., Seshan S., Katz R. Effectiveness of loss labeling in improving TCP performance in wired/wireless networks/ In Proc. of ICNP'2002: The 10th IEEE Int. Conf. on Network Protocols, Paris, France, November 2002.
3. Lin D., Morris R. Dynamics of Random Early Detection/ SIGCOMM '97: Proc. of the ACM SIGCOMM '97 conf. on Applications, technologies, architectures, and proto-

cols for computer communication, 1997, p. 127-137.

4. Feng W.-C., Kang S.G., Kandlur D.D., Saha D. BLUE: A New Class of Active Queue Management Algorithms// IEEE/ACM Trans. on Networking (TON) archive Vol. 10 Iss. 4, August 2002, p. 158-167.

5. Feng W.-C., Kandlur D.D., Saha D., Shin K.G. Stochastic Fair Blue: A Queue Management Algorithm for Enforcing Fairness/ INFOCOM 2001. Twentieth Annual Joint Conf. of the IEEE Computer and Communications Societies. Proc. vol.3 IEEE 2001, p.1520-1529.

6. Parris M., Jeffay K., Smith F.D. Lightweight active router-queue management for multimedia networking/ Multimedia Computing and Networking, SPIE Proc. Series, vol. 3020, January 1999.

7. Pan R., Prabhakar B., Psounis K. CHOKe: A stateless active queue management scheme for approximating fair bandwidth allocation/ Proc. of IEEE INFOCOM 2000, April 2000, p. 942-951.

8. Yavatkar R., Bhagwat N. Improving End-to-End Performance of TCP over Mobile Internetworks/ Proc. of Workshop on Mobile Computing Systems and Applications, Dec., 1994. p. 146-152.

9. Система имитационного моделирования "ns2: network simulator". - <http://www.isi.edu/nsnam/ns>.

Воронежский государственный технический университет

FEATURES OF AN OPTIMUM CHOICE OF ALGORITHMS OF INTERMODULAR INTERACTION A COMPONENT OF THE DISTRIBUTED PROGRAM SYSTEMS

E.N. Khabarov, O.Ja. Kravets

The article describes and analyzes the efficiency of active queue management algorithms for control of overloads such as RED, FRED, Blue and SFB, and based on simulation it is compared with the base Tail Drop

Key words: congestion, overload, interconnection