Моделирование и анализ информационных систем. Т. 26, №1 (2019), с. 170–190
Modeling and Analysis of Information Systems. Vol. 26, No 1 (2019), pp. 170–190

©Степанов Е. П., 2019

DOI: 10.18255/1818-1015-2019-1-170-190

УДК 004.7

Анализ эффективности демультиплексирования транспортных потоков

Степанов Е. П.

Поступила в редакцию 10 января 2019
После доработки 12 февраля 2019
Принята к публикации 14 февраля 2019

Аннотация. Известно, что разделение отдельного транспортного потока на несколько независимых транспортных подпотоков может повысить скорость этого потока. Справедливость этого утверждения, верная для одного потока, не очевидна для *массового* случая, когда демультиплексированию (разделению на подпотоки) подвергают все транспортные потоки в сети одного ISP оператора. Возникает вопрос, какое влияние окажет массовое демультиплексирование транспортных потоков на пропускную способность сети ISP оператора. В статье этот вопрос рассмотрен для статического случая, когда каждый поток разделяется статически, т.е. перед его запуском, на одинаковое число подпотоков. Была предложена математическая модель, на основе которой построена имитационная модель с целью получения более точных оценок производительности сети с демультиплексированием потоков и без него. С помощью имитационной модели определены свойства сети, при которых применение демультиплексирования транспортных потоков оправдано. Корректность полученных результатов обосновывается при помощи эмуляции сети и нагрузки в ней на основе виртуализации стека протоколов при тех же входных данных. В статье рассмотрены разные политики маршрутизации, которые могут быть использованы при массовом демультиплексировании. Особое внимание уделяется алгоритмам, позволяющим строить маршруты с минимальными пересечениями, так как использование неоптимальных по длине, но непересекающихся маршрутов может повысить производительность сети. Маршруты, построенные при помощи этих алгоритмов, использовались как для анализа производительности сети на предложенной имитационной модели с демультиплексированными потоками, так и в случае балансировки недемультиплексированных потоков.

Ключевые слова: многопоточная маршрутизация, качество сервиса, MPTCP

Для цитирования: Степанов Е. П., "Анализ эффективности демультиплексирования транспортных потоков", *Моделирование и анализ информационных систем*, **26**:1 (2019), 170–190.

Об авторах:
Степанов Евгений Павлович, orcid.org/0000-0003-3649-0745, аспирант,
Московский государственный университет им. М.В. Ломоносова,
Ленинские горы, 1, стр. 52, г. Москва, 119991 Россия, e-mail: estepanov@lvk.cs.msu.su

Благодарности:
Работа выполнена при частичной поддержке РФФИ, грант № 18-07-01255.

Введение

В настоящее время продолжают активно исследоваться методы ускорения передачи данных на транспортном уровне (транспортных потоков). Добиться ускорения транспортных потоков можно разными способами. Например, при помощи настройки алгоритма управления перегрузкой, который регулирует интенсивность отправки пакетов в сеть. В случае неоднородности среды передачи данных (например, проводной и беспроводной сегменты) можно поделить маршрут потока на фрагменты с однородной средой передачи данных. Тогда на каждом из фрагментов можно использовать индивидуальный алгоритм управления перегрузкой, адаптированный под качество сервиса, достижимого в среде этого фрагмента. Так устроена версия транспортного протокола Split TCP [1]. В этой статье основное внимание уделено методу ускорения транспортных потоков, основанного на их демультиплексировании.

Идея *демультиплексирования потока* заключается в распределении пакетов одного и того же транспортного потока между несколькими транспортными *подпотоками*, у каждого из которых свой маршрут. Два и более подпотоков будем называть *родственными*, если они образованы пакетами одного и того же транспортного потока, который будем называть *родительским*. В настоящей статье техника демультиплексирования рассматривается применительно к сети Интернет-провайдеров (ISP), и исследования проводятся в предположении, что все родственные подпотоки проходят через сеть одного и того же Интернет-провайдера.

В топологии сети ISP всегда присутствует определенная избыточность, которая позволяет соединить любых двух абонентов несколькими маршрутами. Такая избыточность объективно необходима в сети для устойчивости к отказам сетевого оборудования и может быть использована для демультиплексирования пакетов одного и того же транспортного потока между несколькими маршрутами, соединяющими одну и ту же пару (отправитель, получатель).

Демультиплексировать транспортный поток можно несколькими способами. Например, для одной и той же пары (отправитель, получатель) можно заранее проложить несколько маршрутов и использовать балансировщик нагрузки (например, ECMP [2]) для распределения пакетов транспортного потока между построенными маршрутами.

Такой подход, во-первых, может вызвать нарушение порядка поступления пакетов к получателю. Из-за разной задержки пакетов на разных маршрутах ранее посланный пакет может дойти до получателя позже пакета, посланного позднее первого. Нарушение порядка доставки пакетов может быть воспринято алгоритмом управления перегрузкой на отправителе как признак потери пакетов, вследствие чего будет снижена скорость отправки пакетов в сеть. Как следствие, отправитель будет снижать скорость отправки пакетов в сеть и дублировать пакеты.

Во-вторых, в силу одного из основных принципов архитектуры Интернета – принципа “End-To-End” [3] не желательно вмешиваться в работу потока внутри сети, это следует делать на концах соединения. Поэтому балансировку пакетов между родственными подпотоками стоит делать на стороне отправителя. При этом интенсивность передачи пакетов каждого родственного подпотока должен регулировать свой экземпляр алгоритма управления перегрузкой. Благодаря этому автоматиче-

ски будет корректироваться количество пакетов в каждом родственном подпотоке пропорционально текущей пропускной способности каждого из их маршрутов.

В-третьих, балансировщики в своей работе опираются на эвристические предсказания. В случае неправильного предсказания потоки-“слоны” могут разделять один и тот же маршрут, в то время как на других маршрутах будет всего лишь несколько потоков-“мышей” с маленьким объемом передаваемых данных [4]. Как будет показано ниже, есть подходы, основанные на оперативной оценке реальных характеристик маршрутов (назовем такой подход демультиплексорами), что выгодно отличает его от сетевых балансировщиков.

Еще одним существенным недостатком методов демультиплексирования является то, что при условии массового демультиплексирования потоков увеличивается число транспортных потоков (подпотоков) в сети, вследствие чего растет конкуренция за сетевые ресурсы между ними. Здесь важно понимать, что маршруты подпотоков могут пересекаться как у родственных подпотоков, так и у подпотоков разных родительских потоков. Если алгоритм управления перегрузкой какого-то родственного подпотока вызовет временную перегрузку на каком-то участке сети, то она может оказать влияние на большее количество родительских потоков, чем в случае отсутствия демультиплексирования. Поэтому здесь мы исследуем границы применимости массового демультиплексирования потоков, т.е. определение тех условий, при которых массовое демультиплексирование будет давать преимущество.

Основной вклад этой работы в рассматриваемую проблематику состоит в следующем:

- Построена математическая модель, позволяющая оценить эффективность применения массового статического демультиплексирования в конкретной сети.
- С помощью этой модели (см. разделы 1.4. и 2.3.) определены свойства сети, при которых применение демультиплексирования транспортных потоков оправдано.

1. Массовое статическое демультиплексирование в сетях Интернет-провайдеров

1.1. Демультиплексирование потока: современное состояние

Сегодня есть много реализаций демультиплексирования транспортных потоков. Например, широкую популярность получил *Multi-Path TCP (MPTCP)* [5]. Важным достоинством MPTCP явилась его совместимость со стандартными интерфейсами TCP/IP стека как на сетевом, так и на прикладном уровнях [6]. По существу, MPTCP вводит дополнительный демультиплексирующий промежуточный слой, API которого полностью совместим со стандартным интерфейсом TCP сокета и чьи подпотоки существуют в сети как обычные TCP сессии. Этот промежуточный слой послужил основой для тестирования десятка различных идей по демультиплексированию потока [7].

Все подходы к демультиплексированию транспортных потоков можно разделить на статические и динамические. MPTCP является примером статического подхода.

Во всех известных реализациях MPTCP среди всех родственных подпотоков выделяется стартовый, который начинает передачу данных первым. Порождение остальных родственных подпотоков происходит только после того, как получатель заявил о поддержке работы по протоколу MPTCP. В статьях [6,8] было выявлено несколько случаев, когда применение MPTCP оказалось полезным. В этих работах для каждого рассмотренного случая предложена стратегия выделения маршрутов для родственных подпотоков. В работе [6] предложена стратегия *fullmesh*, ориентированная на случай, когда у устройства отправителя есть несколько сетевых интерфейсов, например, если устройство оборудовано как проводным интерфейсом (Ethernet), так и беспроводным (WiFi) или GSM радиоинтерфейсом. Эта стратегия предполагает установление маршрута для каждой пары интерфейсов отправителя и получателя. Другая стратегия допускает существование нескольких маршрутов между одной и той же парой интерфейсов при условии, что у этих маршрутов нет общих “узких” мест. Именно такой случай рассмотрен в работе *Binder* [8]. Примером динамического подхода к демультиплексированию транспортных потоков является протокол *Flow DeMultiplexing Protocol (FDMP)* [9]. Этот протокол выделяет новые маршруты для родственных подпотоков только в случае снижения скорости родительского потока ниже заранее заданного предела. Как только этот предел достигнут, протокол старается закрыть некоторые родственные подпотоки и освободить маршруты. Далее в статье будем рассматривать только MPTCP со статическим количеством используемых подпотоков.

Пакеты родственных подпотоков в MPTCP могут различаться значениями следующих полей заголовка: IP-адреса источника, IP-адреса назначения, порта источника и порта назначения. Так, IP-адреса источника могут различаться, если родственные подпотоки маршрутизируются через разные сетевые интерфейсы, например, WiFi и Ethernet.

Реализация идеи демультиплексирования транспортного потока требует решения ряда сложных проблем, связанных с управлением перегрузкой. Прежде всего это проблема справедливого распределения сетевых ресурсов. Родительский поток благодаря использованию нескольких родственных подпотоков может получить больше сетевых ресурсов по сравнению с недемультиплексированным потоком. Независимое демультиплексирование нескольких разных родительских потоков может привести к тому, что один родительский поток получит сетевых ресурсов больше других родительских потоков. Такой случай распределения ресурса будем считать *несправедливым* [10]. В ряде работ предлагается поддерживать справедливое распределение, установив дополнительные ограничения на работу алгоритмов управления перегрузкой в MPTCP подпотоках. Например, существуют реализации известных алгоритмов управления перегрузкой, которые были адаптированы под MPTCP таким образом, чтобы не нарушать справедливое распределение ресурсов. Такие работы проводились для реализаций алгоритмов New Reno [11,12], Cubic [13], Compound [14], Fast [15] и многих других. Кроме того, ограничения на работу алгоритмов управления перегрузкой можно вводить не для всех родственных подпотоков, а только внутри множеств родственных подпотоков, которые обладают общим «узким» местом. Такие множества можно динамически составлять на основе анализа характеристик родственных подпотоков, как это делает Алгоритм Dynamic Window Coupling [16].

Итак, методы демультиплексирования уже можно активно применять в транспортных сетях. Однако большое количество способов выделения маршрутов говорит о том, что необходима тонкая настройка демультиплексоров в зависимости от параметров сети, где они будут использоваться. Для этого нужно уметь оценивать эффективность методов демультиплексирования в условиях конкретной сети. Об оценке эффективности методов демультиплексирования пойдет речь в следующих подразделах.

1.2. Модель функционирования сети

Для оценки методов демультиплексирования построим математическую модель функционирования сети. Представим сеть в виде неориентированного связного графа $G = (V, E)$, где V – пронумерованный набор вершин графа, а E – множество ребер, представленных парами (v_i, v_j) на V. E опишем в виде матрицы смежности $\Omega \in \mathbb{R}^{n\times n}$ с весами $\omega_{ij} = \omega(v_i, v_j)$. Если канала от v_i к v_j не существует, то $\omega_{ij} \equiv \omega_{ji} \equiv 0$. В опорной сети ISP оператора все каналы связи обычно обладают примерно одинаковой пропускной способностью, поэтому будем предполагать, что вес всех ребер в нашей модели одинаков и равен единице. Так как граф неориентированный, то $\omega_{ij} \equiv \omega_{ji}$ и матрица Ω является симметричной. Поскольку в силу сделанного выше предположения пропускные способности всех каналов считаются одинаковыми, то обозначим её значение как $\psi_{ch}(\psi_{ch} \in \mathbb{R})$.

Среди множества всех вершин V выделим подмножество вершин P, которые будем называть полюсами. На полюсах расположены отправители и получатели транспортных потоков. Для упрощения математической модели введем следующее ограничение для полюсов: каждая вершина, не являющаяся полюсом, может быть соединена не более чем с одним полюсом:

$$\forall v_i \in V \setminus P\colon (\exists! v_j \in P\colon \omega_{ij} > 0) \vee (\forall v_j \in P\colon \omega_{ij} = 0).$$

Таким образом, даже если в сети ISP к узлу агрегации трафика абонентов подключены сотни-тысячи пользователей, то в нашей математической модели они будут представлены одним полюсом, на котором могут терминироваться сотни-тысячи соединений соответственно. Отсюда вытекает ограничение на размер подмножества полюсов: $2 \le |P| \le \frac{|V|}{2}$. В математической модели сложно учесть все возможные варианты размещения полюсов. Поэтому оценивать методы демультиплексирования мы будем, варьируя соотношения полюсов к остальным вершинам $\frac{|P|}{|V\setminus P|}$ графа сети.

Пусть $F = f_1, ..., f_m$ – множество всех родительских потоков в сети, где $\forall i\colon f_i = ((v_s^i, v_r^i), \mu_i), v_s^i, v_r^i \in P, \mu_i \in \mathbb{R}, (v_s^i, v_r^i)$ задает пару отправителя и получателя, а μ_i – требование к минимальной пропускной способности, необходимой для потока. Поскольку наибольший интерес представляет случай максимальной нагрузки в сети, поэтому сделаем предположение, что все потоки в сети стартуют и заканчиваются одновременно. Пусть также задан коэффициент демультиплексирования $k \in \mathbb{N}$, который определяет число родственных подпотоков у каждого родительского потока. Отсутствие демультиплексирования соответствует случаю $k = 1$.

Пусть также задана функция выбора маршрута R. Аргументом функции R являются граф сети G, родительский поток f_i и коэффициент демультиплексирования

k. Тогда значением функции R является множество $\{\pi_{ij}\}$ из не более чем k маршрутов от отправителя до получателя для i-го потока. Под маршрутом понимается простоя цепь в графе сети, начинающаяся с полюса-отправителя и заканчивающаяся полюсом-получателем. При этом все маршруты, получаемые функцией R, не должны иметь пересечений (под пересечением маршрутов понимается пересечение по дугам графа). Если всевозможных непересекающихся маршрутов между отправителем и получателем только l, то функция R вернет множество из этих l маршрутов. Если всевозможных непересекающихся маршрутов больше k, то результатом будут случайно выбранные k маршрутов:

$$\pi_{ij} = \{v_1^j, v_2^j, ..., \quad v_{n_j}^j\}, \quad f_i \in F, \quad j = \overline{1,l}, \quad l < k, \quad v_1^j = v_s^i, \quad v_{n_j}^j = v_r^i,$$

$$\forall c \in \overline{1, n_j - 1} : \omega(v_c^j, v_{c+1}^j) = 1;$$

$$\forall i, j \in \overline{1, |F|} : \forall \pi_{iu} \in R(G, f_i, k) \wedge \pi_{jv} \in R(G, f_j, k) : \pi_{iu} \cap \pi_{jv} = \emptyset \vee i \equiv j \wedge v \equiv u.$$

Таким образом, набор $\langle V, \Omega, P, \psi_{ch}, F, R, k\rangle$ определяет набор входных параметров математической модели.

Эффективность применения методов демультиплексирования MPTCP с коэффициентом k (обозначим как $MPTCP(k)$) в математической модели будем оценивать отношением производительности сети с применением демультиплексирования T_D к производительности сети без применения демультиплексирования T_C:

$$MPTCP(k) = \frac{T_D}{T_C}.$$

Для определения понятия производительности сети введем дополнительные обозначения. Степенью η_i вершины $v_i \in V$ назовем количество трансцендентных ей ребер, т.е.

$$\eta_i = \sum_j \omega_{ij}. \tag{1}$$

Тогда максимальная степень вершин: $\eta_{max} = \max_i \eta_i$.

Максимально возможная длина маршрута в графе G в общем случае может быть V (на функцию R мы пока не накладываем каких-либо ограничений на выбор маршрутов, например выбор только оптимальных маршрутов, поэтому мы не можем оценить эту длину через диаметр графа). Количество всех маршрутов от вершины v_i до вершины v_j можем оценить методом индукции. В исходной вершине v_i у нас может быть η_{max} вариантов для второй вершины в маршруте. В m-й вершине каждого варианта маршрута может быть $\eta_{max} - 1 \leq \eta_{max}$ вариантов для выбора $(m+1)$-ой вершины в маршруте. В наихудшем случае надо выбрать $|V| - 2$ вершин из маршрута (так как первая и последняя вершины нам уже известны). Тогда количество всевозможных маршрутов от вершины v_i до вершины v_j (множество таких маршрутов обозначим как $ROUTES_{ALL(i,j)}$) можно оценить как

$$|ROUTES_{ALL(i,j)}| \leq \eta_{max}^{|V|-2}.$$

Количество всевозможных маршрутов в графе между полюсами (множество всевозможных маршрутов между полюсами обозначим как $ROUTES_{ALL}$) можно оценить как

$$|ROUTES_{ALL}| \leq C_{|P|}^2 * max_{v_i, v_j \in P} |ROUTES_{ALL(i,j)}| \leq C_{|P|}^2 * \eta_{max}^{|V|-2}.$$

Обычно под скоростью подпотока понимают минимум из скоростей подпотока на каналах маршрута этого подпотока (полученного при помощи функции R):

$$\lambda_{sf} = min_{\langle v_i, v_j \rangle \in \pi_{sf}} \lambda_{sf;(v_i,v_j)}, \quad \lambda_{sf;(v_i,v_j)} \in \mathbb{R}^+.$$

Однако в силу ранее сделанного предположения о том, что все маршруты, получаемые функцией R, не должны иметь пересечений, получаем

$$min_{\langle v_i, v_j \rangle \in \pi_{sf}} \lambda_{sf;(v_i,v_j)} = \psi_{ch}, \quad \forall i \colon \mu_i > \psi_{ch}.$$

Под скоростью родительского потока понимается сумма скоростей родственных подпотоков

$$\lambda_{f_i} = \lambda_{sf_1^i} + ... + \lambda_{sf_k^i} = k * \psi_{ch}.$$

В случае $k = 1$ подпоток совпадает с родительским потоком. С использованием введенного выше производительность сети для любого k определим как:

$$T_D = T_C = \sum_{f_i \in F} \lambda_{f_i}.$$

В сети в случае пересечения маршрутов пропускная способность общего для этих маршрутов канала обычно распределяется не стихийно, а в соответствии с определенным понятием справедливости. В математической модели будем использовать широко применяемое толкование понятия справедливости – *max-min справедливость*. Напомним, что множество скоростей всех подпотоков в сети $\Lambda = \{\lambda_{sf}\}$ является max-min справедливым для заданной сети, если скорость ни одного из потоков не может быть увеличена без ущемления скорости потоков с меньшей скоростью:

$$\forall \Lambda' \colon \lambda'_{sf} > \lambda_{sf} (\lambda'_{sf} \in \Lambda') \implies \exists sf'' \colon \lambda_{sf''} < \lambda_{sf} \wedge \lambda'_{sf''} < \lambda_{sf''}.$$

Так как маршруты функции R не имеют пересечений, то никакие из двух подпотоков не разделяют один и тот же канал, поэтому распределение $\Lambda = \{\lambda_{sf_i} = \psi_{ch}\}$ удовлетворяет понятию max-min справедливости. Производительность сети можно оценить в этом случае как:

$$T_D = \sum_{f_i \in F} \lambda_{f_i} = min(|F| * k, C^2_{|P|} * \eta_{max}^{|V|-2}) * \psi_{ch};$$

$$T_C = \sum_{f_i \in F} \lambda_{f_i} = min(|F|, C^2_{|P|} * \eta_{max}^{|V|-2}) * \psi_{ch}.$$

Минимум необходимо брать, так как максимальное количество маршрутов может оказаться меньше, чем количество родительских потоков или родственных подпотоков.

Таким образом, становится возможным оценить эффективность методов демультиплексирования $MPTCP(k)$ как отношение производительности сети с демультиплексированием к производительности сети без демультиплексирования:

$$MPTCP(k) = \frac{T_D}{T_C} = \frac{min(|F| * k, C^2_{|P|} * \eta_{max}^{|V|-2}) * \psi_{ch}}{min(|F|, C^2_{|P|} * \eta_{max}^{|V|-2}) * \psi_{ch}}.$$

Однако надо понимать, что получившаяся оценка является достаточно грубой. Грубость этой оценки следует из того, что оценка количества всех маршрутов дана для наихудшего случая, а также из-за предположения, что маршруты не пересекаются. В силу общности математической модели она не учитывает особенности каждого конкретного маршрута и топологию графа сети. Поэтому далее на основе математической модели будет предложена имитационная модель, которая будет учитывать то, от чего мы абстрагировались в математической модели, например политику маршрутизации и топологию сети.

Даже для грубой полученной оценки эффективности методов демультиплексирования можно сделать несколько выводов об области применения демультиплексирования. Применение демультиплексирования будет неоправданным, если количество потоков превышает максимальное число маршрутов между полюсами, тем самым задействуя все имеющиеся ресурсы сети, и при этом потоки максимально используют эти ресурсы:

$$(|F| > C^2_{|P|} * \eta^{|V|-2}_{max}) \wedge (\forall f_i \colon \mu_i > \psi_{ch}).$$

При соблюдении описанных в математической модели ограничений и при условии

$$(|F| * k \ll C^2_{|P|} * \eta^{|V|-2}_{max}) \wedge (\forall f_i \colon \mu_i > k * \psi_{ch}),$$

эффективность демультиплексирования будет равняться $\frac{T_D}{T_C} = k$, что является весомым аргументом в пользу демультиплексирования.

1.3. Оценка эффективности массового статического демультиплексирования

Для того чтобы получить более точную оценку эффективности применения метода массового демультиплексирования, построим на основе математической модели имитационную модель. Вначале снимем ограничение на непересекаемость маршрутов, получаемых функцией R. Функция R будет задаваться алгоритмом выбора маршрута, который может основываться на политике маршрутизации в сети (кратчайший маршрут в OSPF), а также на методах балансировки в сети (один из кратчайших маршрутов равной стоимости в ECMP). Так как маршруты теперь могут пересекаться, мы не можем считать, что скорость любого подпотока одинакова и равна ψ_{ch}. Однако на распределение пропускной способности согласно нашей математической модели наложено условие max-min справедливости. Благодаря этому условию мы можем рассчитать распределение пропускной способности согласно алгоритму постепенного заполнения [17]. Таким образом, для расчета производительности сети в имитационной модели необходимо проделать следующие действия:

1. Рассчитать маршруты для каждого потока в сети в случае применения демультиплексирования (обозначим как $ROUTES_D$) и без него (обозначим как $ROUTES_C$):

$$ROUTES_D = \cup_{f_i \in F} R(G, f_i, k);$$

$$ROUTES_C = \cup_{f_i \in F} R(G, f_i, 1).$$

Также обозначим множество всех подпотоков всех потоков как SF_D ($|SF_D| = k * |F|$) и SF_C ($SF_C \equiv F$).

2. При помощи алгоритма постепенного заполнения [17] рассчитать распределение пропускной способности для значений входных параметров $\langle G, \Omega, P, SF_C, ROUTES_C \rangle$ и $\langle G, \Omega, P, SF_D, ROUTES_D \rangle$. Из распределения пропускной способности получим скорость λ_{sf} для каждого подпотока из SF_C и SF_D.

3. Далее согласно введенному ранее определению производительности сети, рассчитать T_C и T_D как сумму соответствующих скоростей подпотоков.

Для оценки эффективности метода статического демультиплексирования имитационная модель исследовалась на ряде входных данных, описанных ниже. Сравнение и анализ полученных результатов приведены в конце этого раздела. В следующем разделе обосновывается корректность полученных результатов при помощи эмуляции сети и нагрузки в ней на основе виртуализации стека протоколов при тех же входных данных.

Вначале опишем набор входных параметров для имитационной модели и множество их значений, для которого мы провели имитационные эксперименты. В качестве первого входного параметра выступает топология сети. По конкретной топологии можно построить множество вершин V и матрицу смежности Ω из математической модели. Пропускная способность каналов была положена равной 1 Гбит/с (для упрощения эмуляции сети). Варианты топологии сети для моделирования брались из библиотеки Topology Zoo [18]. Поскольку эта библиотека не предоставляет информацию о распределении потоков трафика, это распределение было синтезировано следующим образом. Для каждого запуска имитационной модели выбиралось случайное подмножество размера из узлов рассматриваемой топологии сети, к которым подключались дополнительные узлы – *полюса*, имитирующие отправителей и получателей потоков. В отличие от математической модели, полюса не входят в топологию сети, а являются дополнительными вершинами. Поэтому чтобы охватить разные шаблоны нагрузки, варьировалась доля полюсов к узлам топологии сети $\frac{|P|}{|V|}$, а также распределение полюсов по топологии. Еще одним параметром является $|F|$ – количество родительских потоков. Потоки распределялись равномерно по всем полюсам в сети. Требования потоков к пропускной способности возьмем больше, чем $k * 1$ Гбит/с, чтобы максимально задействовать ресурсы сети.

Имитационные эксперименты проводились в предположении, что каждый узел в сети поддерживает метод балансировки ECMP и распределяет входящие потоки между кратчайшими маршрутами до соответствующего получателя. Чтобы сымитировать такую политику маршрутизации, от каждого узла сети были построены кратчайшие маршруты ко всем другим узлам. Тогда для каждого узла можно определить множество альтернативных *следующих скачков* для балансировки, образованное соседями, через которые проходят маршруты до заданного получателя с минимальной стоимостью. В таком случае маршрут каждого потока определяется итерационно, начиная с полюса-отправителя. Для любого узла в маршруте следующий узел выбирается случайным образом из множества альтернативных следующих скачков для соответствующего получателя. Описанный выше метод построения

маршрута назовем *Случайным Следующим Скачком* (ССС) или Random Next Hop (RNH).

Итак, входными параметрами имитационной модели являются:

- Топология (из Topology Zoo) – соответствует V, Ω из математической модели;
- Доля полюсов к узлам топологии (10%, 20%, ..., 100%) – соответствует P из математической модели;
- Пропускная способность каналов (1 Гбит/с) – соответствует ψ_{ch} из математической модели;
- Количество родительских потоков (512, 1024, 2048, 4096) – соответствует $|F|$ из математической модели;
- Размещение отправителя и получателя для каждого потока – определяет F из математической модели;
- Политика маршрутизации (ССС) – соответствует R из математической модели;
- Количество родственных подпотоков у родительского (1, 2, 3) – соответствует k из математической модели.

Для сравнения производительности сети в случае массового статического демультиплексирования потоков с двумя и тремя подпотоками и в случае отсутствия демультиплексирования был проведен анализ скорости потоков для полумиллиона разных значений параметров имитационной модели. Влияние демультиплексирования на производительность сети в случае двух и трех подпотоков отображено на рисунках 1a и 1b соответственно. Каждая метка на графиках представляет подмножество значений входных параметров, для которых совпадает число узлов в топологии и значение утилизации сети. Под утилизацией сети понимается среднее значение утилизации всех каналов. Утилизация канала равна отношению суммы скоростей всех потоков, проходящих через этот канал, к пропускной способности канала и определена для случая недемультиплексированных потоков. Поскольку для разного количества потоков и разных вариантов размещения отправителей и получателей производительность сети может совпадать, то метка на графике может комбинировать результаты для более чем 5 тысяч разных значений входных параметров имитационной модели. Производительность сети для одной метки рассчитывается как среднее значение производительности сети для всех значений входных параметров, соответствующих этой метке. Светлые метки обозначают случаи, где производительность сети с демультиплексированными потоками выше, чем с недемультиплексированными – чем светлее цвет, тем выше преимущество демультиплексирования. Темные метки представляют случаи, где демультиплексирование проигрывает – чем темнее цвет, тем больше падает пропускная способность сети. Серые метки предполагают, что производительность сети для демультиплексированных потоков отличается от производительности сети для недемультиплексированных потоков меньше, чем на один процент.

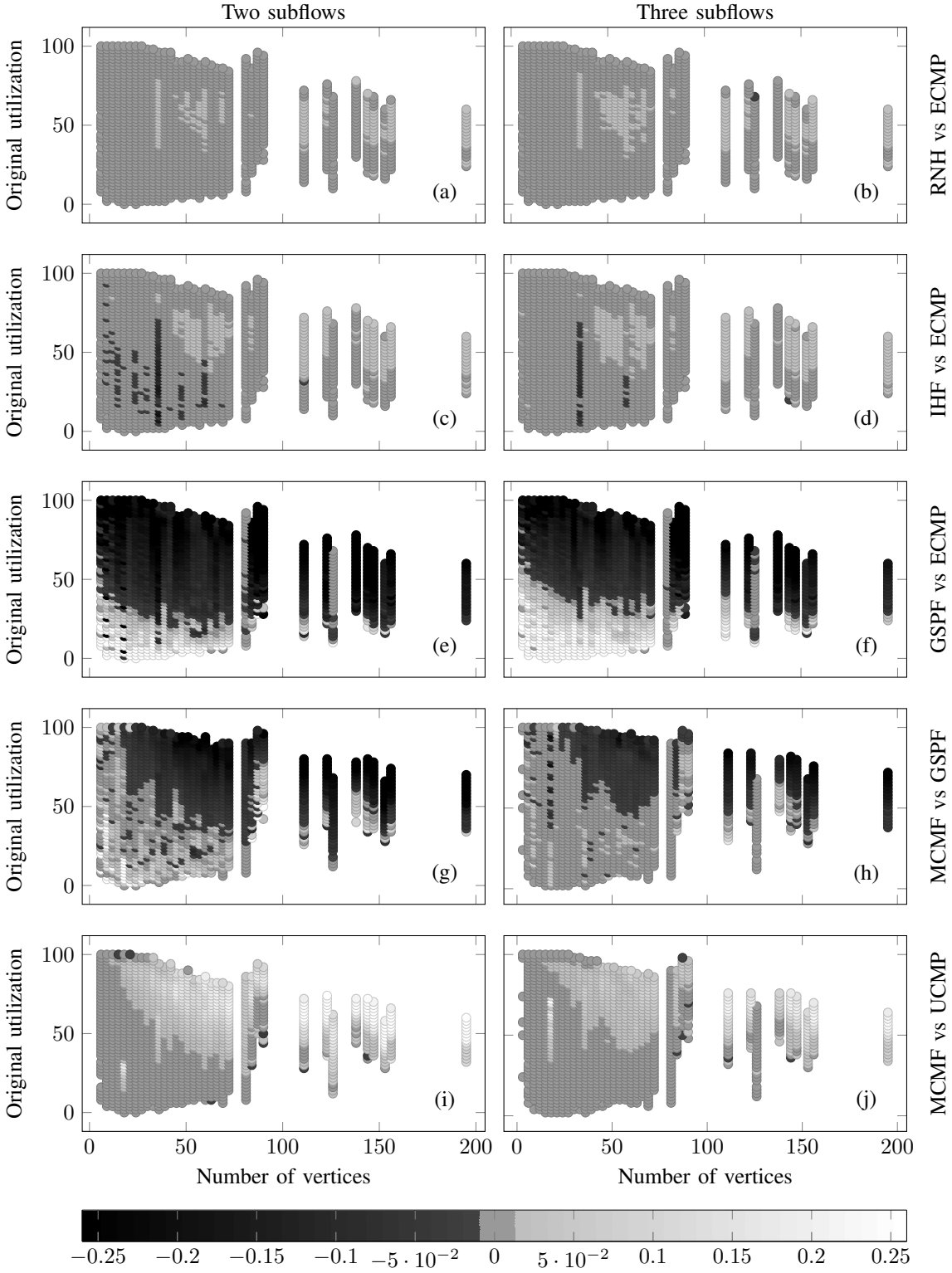

Рис. 1: Сравнение скоростей демультиплексированных и недемультиплексированных потоков с разными политиками маршрутизации

Fig. 1. Comparison of demultiplexed and non-demultiplexed flow rates under different routing policies

Панель на рисунке 1 снизу проводит соответствие между оттенком цвета и величиной преимущества демультиплексирования.

Результаты экспериментального исследования показали, что демультиплексирование обеспечивает увеличение производительности сети больше чем на 1% лишь в случае небольшого числа наборов значений входных параметров. Высокую эффективность метод массового статического демультиплексирования показывает в следующем случае: утилизация сети от 40% до 70%, средняя степень вершин в топологии больше 2.3 и среднее количество кратчайших маршрутов между вершинами больше 1.3. В этом случае производительность сети с применением демультиплексирования будет больше производительности сети без его использования более чем на 1% с вероятностью 78% при демультиплексировании на 2 подпотока и с вероятностью 85% при $k = 3$. Большое количество каналов между узлами сети обеспечивает ECMP балансировщики большим множеством альтернативных следующих скачков по направлению к каждой точке назначения. Например, доминирование демультиплексирования на топологиях с 36 вершинами связано с экспериментами на плотной топологии с названием «Bt North America 2010». Однако лишь небольшое количество рассмотренных топологий обладает достаточным разнообразием маршрутов, которое может эффективно использовать ECMP. Среди 1421 меток на рис. 1, показанных для случая двух подпотоков, демультиплексирование явно выигрывает только 179 раз.

1.4. Обоснование корректности

Для проверки корректности результатов, полученных при помощи имитационной модели, мы провели эмуляцию сети и нагрузки при тех же значениях входных параметров, что и для имитационных экспериментов. Сетевые топологии эмулировались с помощью виртуальных коммутаторов OpenVSwitch и Veth-интерфейсов на экспериментальном стенде, который состоял из одного сервера с 24 ядрами 2 ГГц и оперативной памятью с объемом 64 ГБ. Каждый виртуальный канал между коммутаторами обеспечивал 1 Гбит/с пропускной способности, задержку в 2 мс и 0% потерь. Для задачи генерации трафика (потоков) на сервере размещались 4 гостевые виртуальные машины, выполняющие роль полюсов. Эти виртуальные машины были развернуты с модифицированным ядром, поддерживающим работу по протоколу MPTCP. Каждая гостевая виртуальная машина была подключена к эмулируемой топологии одним сетевым интерфейсом через выделенный под эту задачу коммутатор OpenVSwitch. Этот коммутатор перенаправлял потоки от гостевой виртуальной машины к узлам топологии сети и возвращал потоки обратно к гостевым виртуальным машинам.

Родительские потоки трафика генерировались при помощи клиент-серверной программы iperf3. Одна пара клиент-сервер у iperf3 позволяет открывать до 128 потоков. Чтобы запустить необходимое количество потоков, запускались множественные пары клиент-серверов, которые были равномерно распределены по четырем гостевым виртуальным машинам. Для осуществления демультиплексирования потоков на каждой виртуальной машине выделялось k интерфейсов с IP-адресами из разных подсетей.

Для упрощения настройки экспериментального стенда использовался разработанный нами модуль порождения родственных подпотоков, основанный на страте-

гии full-mesh [6]. Он порождал один подпоток на каждую пару IP адресов внутри одной подсети. Кроме того, порождение дополнительных родственных подпотоков происходило на стороне сервера, а не клиента. Тогда родственные подпотоки имели разные транспортные порты на стороне сервера, но гарантированно имели один и тот же транспортный порт на стороне клиента. Так как программе iperf3 в параметрах можно указать, какие транспортные порты использовать на стороне клиента, то можно узнать принадлежность каждого подпотока к родственным потокам по паре IP-адресов и клиентскому транспортному порту. Порт на стороне сервера для этой задачи не подойдет, так как iperf3 и так запускал до 128 параллельных родительских потоков с одним серверным портом для одной пары клиент-сервера. Используя информацию об IP-адресах и клиентском транспортном порте подпотока, можно реализовать любую политику маршрутизации, вычисляя для каждого подпотока подходящие правила пересылки и загружая их в коммутаторы.

Каждый запуск экспериментального стенда был настроен на создание 1024 одноминутных родительских потоков. Для каждого потока была рассчитана его скорость как среднее значение от ежесекундных замеров скорости потока. При этом статистика за первые 15 секунд каждого запуска была отброшена, чтобы отсечь период стабилизации нагрузки (например, стадия медленного старта у алгоритмов управления перегрузок).

Множество всех значений входных параметров, где алгоритмом маршрутизации был CCC, включало в себя несколько тысяч различных комбинаций сетевых топологий, нагрузки трафика и т.д.

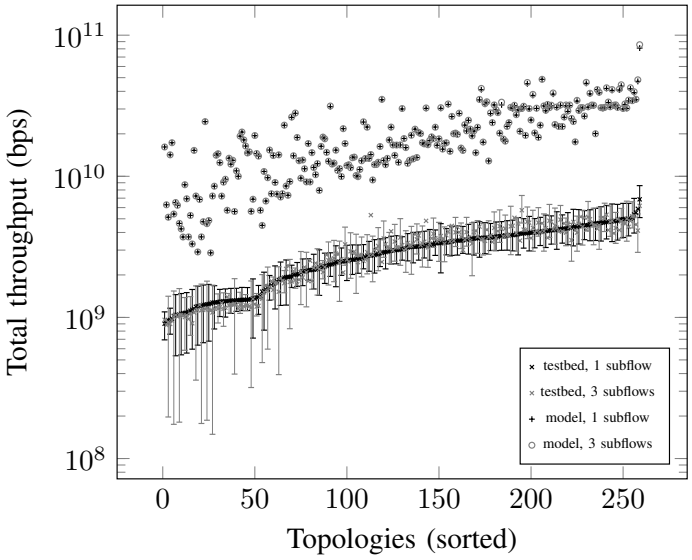

Рис. 2: Производительность сети для CCC, измеренная на тестовом стенде и в имитационной модели

Fig. 2. Total network performance as it is measured by the testbed and estimated by simulation model for RNH

Рисунок 2 показывает среднее значение производительности сети для каждой из тестируемых топологий. Производительность сети в случаях недемультиплексированных потоков и демультиплексирования на три подпотока показана черными и

серыми крестами соответственно. Вертикальный отрезок для каждой отметки ограничивает доверительный интервал для значения производительности сети с уровнем доверия 50 %.

Результаты для демультиплексирования на два подпотока были опущены, потому что они оказались аналогичны случаю с тремя подпотоками. Рисунок 2 также содержит оценку, предоставленную имитационной моделью из раздела 1.3. Оценки в случаях недемультиплексированных потоков и демультиплексирования на три подпотока показаны черными плюсами и серыми окружностями соответственно.

Линейная корреляция между измеренной производительностью сети и оценкой имитационной модели составила 0.79, что является высокой степенью соответствия по шкале Чеддока [19]. Пониженная точность связана с тем, что прогнозы нашей модели чрезмерно оптимистичны – на практике TCP потоки не задействуют все предоставленные ресурсы сети из-за неэффективности работы алгоритмов управления перегрузкой. Тем не менее, модель применима для сравнения производительности сети в случаях демультиплексирования потоков и его отсутствия. И экспериментальный стенд, и имитационная модель не показали существенных улучшений или ухудшений в производительности сети в случае массового демультиплексирования.

2. Иные политики маршрутизации

В предыдущем разделе был сделан вывод, что массовое применение протокола MPTCP дает преимущество лишь для небольшого количества рассмотренных значений входных параметров модели. Причиной этому может быть неэффективность используемой политики маршрутизации. Например, из-за случайности выбора следующего скачка несколько родственных подпотоков могут использовать одно и то же узкое место одного и того же маршрута. В этом разделе приводится исследование альтернативных вариантов политик маршрутизации, которые могут быть использованы при массовом демультиплексировании.

2.1. Обратимые функции хэширования

Чтобы проверить гипотезу о неэффективности алгоритма маршрутизации CCC, была изменена процедура генерации маршрутов. В измененном варианте предотвращается маршрутизация родственных подпотоков через один и тот же следующий скачок, если это возможно. Реализовать предложенный подход в сетях ISP можно при помощи стратегии, предложенной в [20], которую будем называть *Обратимой Функцией Хэширования* (ОФХ) или Invertible Hash Function (IHF). Идея ОФХ заключена в том, чтобы выбор среди множества альтернативных следующих скачков осуществлялся при помощи обратимой хэш-функции от заголовков пакетов. Благодаря обратимости хэш-функции отправитель транспортного потока сможет подобрать значение своего порта таким образом, чтобы на маршрутизаторе хэш-функция от заголовка пакета соответствовала индексу желаемого следующего скачка. В частности, для демультиплексированного потока можно обеспечить распределение i-го подпотока на i-й следующий скачок из множества альтернативных.

Рисунки 1c и 1d сравнивают производительность сети для стратегии ОФХ. Неожиданным результатом оказалось, что ОФХ обеспечило лишь минимальное преимущество для статического демультиплексирования – демультиплексированные потоки выигрывают на 243 метках на рис. 1c из 1421, а проигрывают на 119 из них. На тех значениях входных параметров, на которых демультиплексирование обладало преимуществом в случае ССС, производительность сети в случае ОФХ оказалась намного ниже. Наше исследование показало, что поскольку отправитель не обладает информацией о количестве альтернативных следующих скачков на маршрутизаторе, то он всегда выбирает следующие скачки с наименьшими индексами. Поскольку в этом случае транспортные потоки никогда не используют следующий скачок с высоким индексом, ОФХ не может использовать все доступные альтернативные маршруты, в то время как ССС равномерно распределяет потоки по альтернативным следующим скачкам.

2.2. Маршруты с минимальными пересечениями

До сих пор рассматривались только те политики маршрутизации, которые распределяли подпотоки среди маршрутов с одинаковой наименьшей стоимостью. С использованием ССС и ОФХ могли возникнуть ситуации, когда маршруты родственных подпотоков совпадают или частично накладываются друг на друга, так как нет альтернативных маршрутов той же стоимости. Использование неоптимальных по длине, но непересекающихся маршрутов может повысить производительность сети. В данном разделе проверяется эта гипотеза при помощи нескольких алгоритмов построения маршрутов, которые предпочитают непересекающиеся маршруты кратчайшим. При этом мы предполагаем, что можем контролировать маршрут каждого подпотока (такое возможно, например, в сети SDN), а детали осуществления этого предположения лежат за рамками этой статьи.

Стоит отметить, что построение полностью непересекающихся маршрутов не всегда возможно. Топология сети может быть устроена таким образом, что какой бы маршрут от заданных отправителя и получателя ни был выбран, он обязательно пройдет через некоторый канал в сети. В этом случае предпочтительнее будут маршруты с минимальным пересечением, так как узкое место может находиться в другом месте сети. Под n маршрутами с минимальным пересечением понимается любое множество из n маршрутов, у которых сумма пересечений каждой пары маршрутов этого множества минимальна по сравнению со всеми другими множествами из n маршрутов.

Наше исследование начнем с выбора алгоритмов построения соответствующих маршрутов для родственных подпотоков. Многие статьи [9, 21, 22] предлагают использовать простой, но в то же время эффективный *Greedy Shortest Paths First* (GSPF) алгоритм. На каждой итерации этого алгоритма строится маршрут для одного из родственных подпотоков при помощи одного из алгоритмов построения кратчайшего маршрута (например, [23]). По окончании очередной итерации GSPF вес каждого ребра построенного маршрута увеличивается на определенное штрафное значение. Обычно это значение достаточно велико, чтобы маршруты с повторно используемыми ребрами были менее предпочтительными, чем маршруты без пересечений.

Достоинства GSPF – простота и явный способ вычисления дополнительных маршрутов родственных подпотоков по требованию. Однако алгоритм не даст оптимального решения – количество совпадающих ребер в построенных маршрутах не всегда минимально. В какой-то момент жадный выбор (построение кратчайшего маршрута на каждой итерации) может привести к ситуации, когда не существует еще одного непересекающегося маршрута, хотя можно получить аналогичное количество непересекающихся маршрутов при другом алгоритме построения (подробнее в [24]).

Нами был разработан еще один алгоритм построения маршрутов, который в отличие от GSPF дает оптимальное решение по количеству пересечений маршрутов. Этот алгоритм сводит задачу построения маршрутов к классической задаче нахождения *максимального потока минимальной стоимости* (min-cost max-flow), откуда получил название MCMF алгоритма. MCMF алгоритм начинает свою работу с изменения начального графа сети. Он заменяет каждое ребро в графе на множество k параллельных ребер, где k равно ожидаемому числу родственных подпотоков. Для i-го ребра в этом множестве, $0 \leq i < k$, MCMF задает его стоимость как $1+i|E|$, где $|E|$ обозначает количество всех ребер в исходном графе. Стоимость необходима для того, чтобы в решении было как можно меньше пересечений между маршрутами. Для всех ребер в графе MCMF устанавливает пропускную способность в 1. Наконец, MCMF добавляет специальный узел *сток* и соединяет его с полюсом-получателем ребром с пропускной способностью k.

По измененному графу MCMF алгоритм вычисляет максимальный поток минимальной стоимости между отправителем и стоком. С одной стороны, поток всегда может получить по крайней мере k единиц пропускной способности, так как MCMF заменяет каждое ребро в графе на k ребер с единичной пропускной способностью. С другой стороны, скорость потока никогда не сможет превысить k из-за узкого места, образованного ребром на стоке. Таким образом, максимальная скорость потока всегда эквивалентна k, и она распределяется среди k разных маршрутов через измененный граф.

Результаты, полученные алгоритмом нахождения максимального потока минимальной стоимости для измененного графа, отображаются на соответствующие маршруты в исходном графе. Хотя отображение можно провести неоднозначно, каждый полученный набор маршрутов имеет минимальное пересечение и наименьшую суммарную стоимость среди всех маршрутов с минимальным пересечением. MCMF в отличие от GSPF всегда обеспечивает *оптимальный* результат – пока есть возможность уменьшить число разделяемых каналов, MCMF будет её использовать.

2.3. Оценка производительности сети в случае GSPF и MCMF

Описанные выше стратегии выбора маршрутов GSPF и MCMF были проанализированы на имитационной модели из раздела 1.3. На рисунках 1e и 1f показаны результаты сравнения GSPF с ECMP. Рисунки 1g и 1h демонстрируют улучшение в производительности сети, которое MCMF обеспечивает по сравнению с GSPF. Эксперименты показывают, что GSPF и MCMF обеспечивают существенное увеличение производительности сети по сравнению с ECMP, когда значение утилизации низкое. Если утилизация сети ниже 37%, демультиплексирование на три подпотока увели-

чивает производительность сети с вероятностью большей, чем 50%! Очевидно, что GSPF и MCMF обеспечивают достаточное разнообразие маршрутов для демультиплексированных потоков, чтобы обойти перегруженные каналы, появляющиеся вдоль кратчайших маршрутов. Однако, как только сеть становится сильно загруженной, маршруты от GSPF и MCMF ухудшают производительность сети из-за большей длины их маршрутов. Чем длиннее маршрут, тем больше ресурсов сети необходимо для потока, что труднее удовлетворить при большой утилизации сети. Если сеть использует 80% её полосы пропускания, демультиплексирование на три подпотока ухудшит производительность сети на 18% в среднем.

С целью обоснования корректности расчетов была проведена эмуляция на части значений входных параметров для стратегии маршрутизации GSPF на экспериментальном стенде из раздела 1.4. На рисунке 3 показана производительность сети для демультиплексированных и недемультиплексированных потоков, когда доля полюсов составляет 30%. Эмуляция в целом подтвердила наши предсказания – как математическая модель, так и эмуляция сети показали, что демультиплексирование должно увеличить производительность сети на 25% в среднем. Линейная корреляция между результатами на модели и на стенде составила 0.84, что тоже является высокой степенью соответствия по шкале Чеддока.

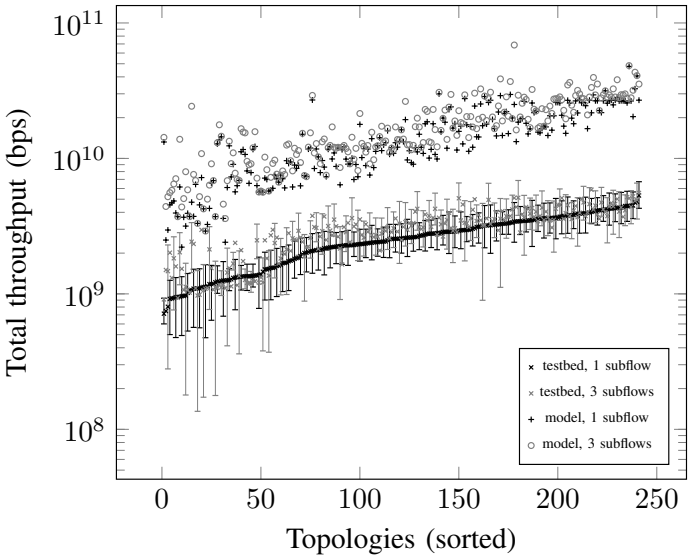

Рис. 3: Производительность сети для GSPF,
измеренная на тестовом стенде и в имитационной модели

Fig. 3. Total network performance as it is measured
by the testbed and estimated by simulation model for GSPF

2.4. Алгоритм выбора маршрутов UCMP

Как было показано выше, демультиплексирование потоков с политикой маршрутизации GSPF или MCMF позволяет увеличить производительность сети в случае её низкой утилизации (до 37%). Однако есть опасение, что этот эффект вызван скорее алгоритмами выбора маршрута, а не техникой демультиплексирования потоков.

Для снятия этого опасения была проведена еще одна серия экспериментов с недемультиплексированными потоками. В этот раз потоки распределялись равномерно по неоптимальным маршрутам, предоставленным GSPF и MCMF, а не только по кратчайшим, как в CCC. Такой алгоритм выбора маршрутов для недемультиплексированных потоков назовем *Unequal-Cost Multi-Path* (UCMP).

Оценка производительности сети с алгоритмом выбора маршрутов UCMP, полученная на имитационной модели из раздела 1.3., была использована как новый эталон для оценки эффективности демультиплексирования. Сравнение производительности сети с алгоритмами выбора маршрутов UCMP для недемультиплексированных потоков и MCMF для демультиплексированных показано на рисунках 1i и 1j. Результаты для GSPF были опущены, так как они схожи с результатами для MCMF. Демультиплексирование улучшает производительность сети, достигнутую балансировкой UCMP, в ряде случаев, однако эти случаи соответствуют сетям с утилизацией выше 60%. Ранее было показано, что маршрутизация по кратчайшему маршруту CCC более эффективна для высоко утилизированных сетей, чем GSPF и MCMF. А при утилизации ниже 60% можно использовать алгоритм выбора маршрутов UCMP для недемультиплексированных потоков, что приводит к тому же результату, как и демультиплексирование с MCMF или GSPF. Поэтому можно сделать вывод, что демультиплексирование не способствует увеличению производительности сети, по крайней мере при неизменяющейся нагрузке.

Интересно, что имитационное моделирование показало уменьшение преимущества демультиплексирования потоков при утилизации сети выше 60%, если количество родственных подпотоков увеличить с 2 до 3. Интуитивно, демультиплексирование на большее количество подпотоков должно увеличивать преимущество в производительности сети над недемультиплексированными потоками. Однако включение третьего маршрута приводит к увеличению утилизации сети и производительности сети для UCMP в среднем на 10%, из-за чего становится намного сложнее увеличить производительность сети в случае демультиплексирования.

Эксперименты показали, что использование маршрутов с минимальным пересечением для родственных подпотоков имеет гораздо более высокий потенциал, чем использование традиционной политики маршрутизации. Однако преимущество демультиплексирования, скорее всего, лежит в более равномерном распределении пропускной способности сети между родительскими потоками, а не в увеличении производительности сети. В зависимости от текущей утилизации каналов вдоль маршрутов скорость транспортных потоков может существенно отличаться. Использование нескольких альтернативных маршрутов родственными подпотоками может смягчить различие в скорости родительских потоков и гарантировать более высокую минимальную скорость для всех потоков в сети. Но исследование этого вопроса находится за рамками данной статьи.

Заключение

В статье проведено исследование, может ли массовое статическое демультиплексирование потоков улучшить производительность сети масштаба ISP. Была предложена математическая модель, на основе которой построена имитационная модель с

целью получения более точных оценок производительности сети с демультиплексированием потоков и без него. Хотя наша имитационная модель и не учитывает всех тонкостей работы алгоритмов управления перегрузкой, её все же можно использовать для прогнозирования того, улучшит демультиплексирование производительность сети или ухудшит. Согласно нашим экспериментам массовое демультиплексирование не обеспечит никакого улучшения для большинства топологий ISP сетей, представленных в библиотеке TopologyZoo.

В то же время было показано, что демультиплексированные соединения имели более высокую эффективность, если маршруты их родственных подпотоков имели минимальное пересечение. Была произведена оценка производительности сети при работе с двумя разными алгоритмами построения маршрутов – GSPF и MCMF. При использовании маршрутизации по каждому из этих алгоритмов демультиплексированные потоки обеспечили ощутимое увеличение производительности сетей ISP с уровнем утилизации ниже 37%. Интересно, что можно добиться аналогичного улучшения, используя MCMF или GSPF при реализации алгоритма выбора маршрутов UCMP для недемультиплексированных потоков. Таким образом, улучшение производительности сети является основным достоинством алгоритмов построения маршрутов, а не демультиплексирования.

Кроме того, в ходе экспериментального исследования стало ясно, что необходимо демультиплексировать разные родительские потоки на разное число родственных подпотоков в зависимости от топологии сети и текущей утилизации каналов. Определить, на какое количество необходимо демультиплексировать поток, можно только в динамике в зависимости от текущего состояния сети. Если скорость родительского потока не увеличивается при создании дополнительного подпотока, то этот подпоток лишний и его можно закрыть, чтобы не возникало лишней конкуренции в сети. Это подтверждает преимущество динамических подходов к выбору маршрутов, таких как FDMP [9].

Дополнительное исследование демультиплексированных потоков может показать преимущества по другим показателям. Например, демультиплексирование может приводить к более равномерному распределению пропускной способности сети по сравнению с UCMP. На практике это означает, что потоки между клиентами одной ISP сети будут обеспечивать более стабильную и прогнозируемую скорость независимо от их относительного местоположения. Кроме того, демультиплексирование также может обеспечить определенную выгоду в сетях с быстро меняющимися матрицами трафика. Быстрое перераспределение данных между подпотоками должно реагировать на перегрузки в сети намного быстрее, чем балансировщики, интегрированные в сетевую инфраструктуру. Однако проверка этого утверждения выходит за рамки данной статьи.

Список литературы / References

[1] Kim Bong Ho, Calin Doru, “On the Split-TCP Performance over Real 4G LTE and 3G Wireless Networks”, *IEEE Communications Magazine*, **55**:4 (2017), 124–131.

[2] Chiesa M., Kindler G., Schapira M., “Traffic Engineering With Equal-Cost-MultiPath: An Algorithmic Perspective”, *IEEE/ACM Transactions on Networking*, **25**:2 (2017), 779–792.

[3] Saltzer J. H., Reed D. P., Clark D. D., “End-to-end Arguments in System Design”, *ACM Trans. Comput. Syst.*, **2**:4 (1984), 277–288.

[4] Curtis A. R., Kim W., Yalagandula P., “Mahout: Low-overhead datacenter traffic management using end-host-based elephant detection”, *2011 Proceedings IEEE INFOCOM*, 2011, 1629–1637.

[5] Ford A., et al., *RFC 6824: TCP Extensions for Multipath Operation with Multiple Addresses*, 2013, 64 pp.

[6] Raiciu C., et al., “How Hard Can It Be? Designing and Implementing a Deployable Multipath TCP”, *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*, 2012, 399–412.

[7] Habib S., et al., “The Past, Present, and Future of Transport-Layer Multipath”, 2016, https://arxiv.org/abs/1601.06043.

[8] Boccassi L., Fayed M.M., Marina M.K., “Binder: A System to Aggregate Multiple Internet Gateways in Community Networks”, *Proceedings of the 2013 ACM MobiCom Workshop on Lowest Cost Denominator Networking for Universal Access*, 2013, 3–8.

[9] Chemeritskiy E., Stepanov E., Smeliansky R., “Managing network resources with flow (de) multiplexing protocol”, *Mathematical and Computational Methods in Electrical Engineering*, Recent Advances in Electrical Engineering Series, **53**, 2015, 35–43.

[10] Raiciu C., Handley M., Wischik D., “RFC 6356: Coupled Congestion Control for Multipath Transport Protocols”, 2011, 12 pp.

[11] Khalili R., et al., “MPTCP Is Not Pareto-Optimal: Performance Issues and a Possible Solution”, *IEEE/ACM Transactions on Networking*, **21**:5 (2013), 1651–1665.

[12] Peng Q., et al., “Multipath TCP: Analysis, Design, and Implementation”, *IEEE/ACM Transactions on Networking*, **24**:1 (2016), 596–609.

[13] Le T. A., Hong C. S., Lee S., “MPCubic: An extended cubic TCP for multiple paths over high bandwidth-delay networks”, *ICTC 2011*, 2011, 34–39.

[14] Ha B. P., et al., “A hybrid multipath congestion control algorithm for high speed and/or long delay networks”, *2014 International Conference on Advanced Technologies for Communications (ATC 2014)*, 2014, 452–456.

[15] Bich-Phuong Ha, et al., “A Multipath Cubic TCP Congestion Control with Multipath Fast Recovery over High Bandwidth-Delay Product Networks”, *IEICE Transactions on Communications*, **E-95B**:7 (2012), 2232–2244.

[16] Hassayoun S., Iyengar J., Ros D., “Dynamic Window Coupling for multipath congestion control”, *2011 19th IEEE International Conference on Network Protocols*, 2011, 341–352.

[17] Jean-Yves Boudec, *Rate adaptation, Congestion Control and Fairness: A Tutorial*, 2018, 54 pp.

[18] Knight S. et al., “The internet topology zoo”, *IEEE Journal on Selected Areas in Communications*, **29**:9 (2011), 1765–1775.

[19] Carter H., “Chaddock, Robert, Emmet. Principles and Methods of Statistics. Pp. xvi+471. Houghton Mifflin Company, 1925”, *The ANNALS of the American Academy of Political and Social Science*, **123**:1 (1926), 229.

[20] van der Linden S., Detal G., Bonaventure O., “Revisiting Next-hop Selection in Multipath Networks”, *Proceedings of the ACM SIGCOMM 2011 Conference*, 2011, 420–421.

[21] Navin Kukreja, et al., “SDN based automated testbed for evaluating multipath TCP”, *IEEE International Conference on Communication, ICC 2015* (London, United Kingdom, June 8–12, 2015), 2016, 718–723.

[22] Chawanat Nakasan, et al., “A Simple Multipath OpenFlow Controller using topology-based algorithm for Multipath TCP”, 2015, https://arxiv.org/abs/1509.08388.

[23] Dijkstra E. W., “A Note on Two Problems in Connexion with Graphs”, *Numer. Math.*, **1**:1 (1959), 269–271.

[24] Suurballe J. W., Tarjan R. E., “A quick method for finding shortest pairs of disjoint paths”, *Networks*, **14**:2 (1984), 325–336.

Stepanov E. P., "On Analysis of Traffic Flow Demultiplexing Effectiveness", *Modeling and Analysis of Information Systems*, **26**:1 (2019), 170–190.

DOI: 10.18255/1818-1015-2019-1-170-190

Abstract. It is known that the demultiplexing of the individual traffic flow into several independent transport subflows can increase the speed of it. This statement is true for a single flow but its truth for the *massive* case, when demultiplexing technics are applied to all traffic flows in the network of a single Internet Service Provider (ISP), is not obvious. The question arises, what impact the massive demultiplexing of traffic flows will have on the whole ISP network bandwidth. In this paper, this question is considered for the static case, when each flow is demultiplexed statically, i.e. every flow before its launching is demultiplexed into the same number of subflows. We developed a mathematical model that was used to construct a simulation model in order to obtain more accurate estimates of the network performance with and without flow demultiplexing. Using simulation model, network properties are defined, under which the demultiplexing of traffic flows is justified. We proved the correctness of the obtained results by emulating a network load based on a protocol stack virtualization for the same input. We considered various routing policies that can be used for massive demultiplexing. Special attention is paid to algorithms that allow you to build routes with minimal intersections, since using the non-intersecting routes with non-optimal cost can increase the network performance. The routes constructed with these algorithms were used both for the network performance analysis with demultiplexed flows and in the case of balancing non-demultiplexed flows.

Keywords: multipath routing, quality of service, MPTCP

On the authors:
Evgeniy P. Stepanov, orcid.org/0000-0003-3649-0745, graduate student,
Lomonosov Moscow State University,
1, bd. 52 Leninskiye Gory, Moscow 119991, Russia, e-mail: estepanov@lvk.cs.msu.su

Acknowledgments:
The research was supported by the Russian Fund of Basic Research (Project No. 18-07-01255).