МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ ПОТОКА ЗАЯВОК К ОБЛАЧНОМУ ВЫЧИСЛИТЕЛЬНОМУ КЛАСТЕРУ

DOI: 10.36724/2072-8735-2020-14-10-39-44

Manuscript received 05 June 2020

Accepted 07 September 2020

Куприянов Дмитрий Олегович,
Московский Технический Университет Связи и Информатики,
Москва, Россия,
dim.kupro@gmail.com

Ключевые слова: облачные вычисления,
вычислительные центры, марковские
процессы, инфраструктура веб-приложения,
Processor Sharing, теория массового
обслуживания, качество обслуживания

При использовании облачных сервисов оценка параметров качества обслуживания при запуске сервиса имеет очень большое значение. В работе предлагается вариант математической модели облачного сервиса с использованием дисциплины Processor Sharing (PS) для оценки качества обслуживания при обработке моносервисного трафика. В рассматриваемой модели поток заявок является пуассоновским с интенсивностью λ . Учитываются ожидание заявок в очереди и рассматривается случай с бесконечной очередью, в котором все заявки будут обслужены. Заявки в рассматриваемой системе представляют собой HTTP запросы с определённой полезной нагрузкой (payload) в формате JSON. Размер полезной нагрузки для всех заявок является разной, но разброс её величины находится в достаточно узком коридоре (десятки байт). Построена модель вычислительного кластера. На основе этой модели произведены расчёты характеристик относительной эффективности обслуживания и относительной пропускной способности потока при различном количестве ресурса, предоставляемого одному запросу. Построены графики зависимостей этих характеристик от коэффициента загрузки кластера. Сделаны выводы об изменении параметров качества обслуживания вычислительного кластера при изменении нагрузки на вычислительный кластер. Предложенная модель позволяет оценить качество обслуживания заявок от пользователей приложением и может быть использована при проектировании инфраструктуры веб-приложения, а также для оценки поведения кластера при изменении его размеров и размеров поступающего на вход потока заявок.

Информация об авторе

Куприянов Дмитрий Олегович, аспирант, Московский Технический Университет Связи и Информатики, Москва, Россия

Для цитирования:

Куприянов Д.О. Математическое моделирование потока заявок к облачному вычислительному кластеру // T-Comm: Телекоммуникации и транспорт. 2020. Том 14. №10. С. 39-44.

For citation:

Kupriyanov D.O. (2020) Mathematical modeling of requests flow to cloud compute cluster. *T-Comm*, vol. 14, no. 10, pp. 39-44. (in Russian)

Введение

Быстрый рост сложности веб-сервисов и их количества повышает требования к вычислительным мощностям и интерфейсам для доступа к ним. Для владельца сервиса наиболее оптимальным способом развертывания своего сервиса являются вычислительные облачные кластеры. Их использование позволяет сильно уменьшить затраты не только на размещение оборудования, но и на его обслуживание, позволяет избежать большого количества сложных работ по запуску, размещению и поддержке работоспособности оборудования. При использовании облачных сервисов, в отличие от собственной капитальной инфраструктуры, оплачиваются часы занятия ресурсов облачного провайдера, поэтому ещё более важной становится оценка параметров качества обслуживания, которые напрямую влияют на количество и загруженность вычислительного ресурса, а следовательно и на отношение пользователя к продукту и стоимость инфраструктуры [1]. Наиболее важными характеристиками при предоставлении вычислительных мощностей облачным провайдером являются количество виртуальных ядер процессора (vCPU), оперативной памяти (RAM, байт), скорость чтения/записи на диск (бит/с), которая зависит от типа используемого накопителя, и пропускная способность сети (бит/с) [1-3]. Количество этих ресурсов в системе зависит от текущей нагрузки на сервис и может меняться при увеличении или уменьшении количества запросов от абонентов в единицу времени автоматически или вручную, системными администраторами, обслуживающими кластер на стороне владельца сервиса [4]. На данный момент многие клиенты облачных сервисов оценивают ресурс экспериментально, исходя из среднего количества запросов в минуту (RPM). В работе предлагается построить математическую модель веб-сервиса для оценки параметров качества обслуживания. Ожидается, что такой подход позволит снизить затраты на инфраструктуру или увеличить стабильность кластера в зависимости от переоценки/недооценки требуемого размера кластера экспериментальным методом. Также, такая модель может помочь при эксплуатации сервиса при изменении величины потока заявок в систему.

Описание функциональной модели

Для построения функциональной модели и выбора моделируемых параметров рассмотрим простейшее приложение, обслуживающее поток запросов от абонентов. Каждый запрос представляет собой HTTP GET запрос с полезной нагрузкой (payload), который генерируется переходом по ссылке, открытием страницы сайта, различными консольными утилитами или программными библиотеками для работы с HTTP запросами (например, curl в системах с Linux). GET используется для получения информации с сервера и позволяет передавать в запросе параметры, указывающие на нужный пользователю ресурс [1]. GET Запрос посылается на сервер с URL-encoded параметрами [4]. Пример URL запроса к серверу приложения:

`http://test.web-service.com/main.html?key1=value1&key2=value2`

В современных веб-сервисах в связке с сервером приложения используют базу данных. Таким образом, запрос к программному интерфейсу приложения (API), через URL выше, делает запрос в базу данных, обрабатывает его и возвращает клиенту ответ. Ответ на запрос к API веб-сервиса в рассматриваемой системе представляет собой текстовую полезную нагрузку в формате JSON. Таким образом, все заявки в системе имеют один формат, поэтому при построении модели следует учитывать моносервисность клиентского трафика.

Инфраструктура сервиса представляет собой облачный кластер, состоящий из серверов на базе одного из дистрибутивов OS Linux. Сервер со стороны клиента облачного сервера представляет собой совокупность физических вычислительных ресурсов, находящихся в дата-центре облачного провайдера с интерфейсом для доступа к ним и предустановленной операционной системой и может восприниматься как простой ресурсный блок. Распределение заявок между серверами кластера осуществляет балансировщик нагрузки (Load Balancer). Простейшая схема сервиса с одноступенчатой балансировкой изображена на рис. 1.

Рис. 1. Схема облачного сервиса

В представленной схеме через Instances обозначены облачные виртуальные машины, на которых запущен веб-сервер для обслуживания потока заявок. Они представляют собой определённое количество физических вычислительных ресурсов, выделенных по требованию пользователя из ресурсной емкости датацентра провайдера [1, 12]. Также, стоит отметить, что провайдеры часто создают готовые типы виртуальных машин, включающие определённое количество ресурсов и предназначенные для определённых задач, например с увеличенным количеством RAM для требовательных к оперативной памяти задач.

В работе предлагается рассмотреть стандартный кластер и оценить распределение заявок, используя в качестве основного критерия количество виртуальных ядер процессора, выделяемых для обслуживания заявки в единицу времени, как основной вычислительный ресурс. Описанная выше специфика «типов серверов» в таком подходе означает, что не у всех провайдеров есть возможность предоставить гибкое изменение размеров кластера. То есть, в большинстве случаев оно осуществляется добавлением серверов (группы ядер), а не отдельных ядер.

Распределение заявок осуществляется по системе Dynamic Round Robin, критерием для выбора сервера является текущая статистика использования его ресурсов. Приоритет отдается наименее нагруженным машинам в кластере. Добавление новых серверов осуществляется при увеличении средней нагрузки на центральный процессор (CPU) серверов выше 60%. Удаление серверов осуществляется при уменьшении средней нагрузки на CPU до 20% в течение долгого времени. Такие пороговые значения позволяют сохранять стабильность кластера и дают время администраторам на изменение размера «флота» при увеличении или уменьшении нагрузки.

Также, стоит отметить, что в целях безопасности и по экономическим причинам провайдеры облачных сервисов вводят квоты на вычислительные ресурсы, поэтому размер кластера не может увеличиваться бесконечно. Квоты позволяют провайдеру защитить клиента от чрезмерных затрат при несанкционированном доступе к кластеру и более точно планировать свои вычислительные ресурсы и ресурсы передачи и хранения данных. При достижении квоты дальнейшее увеличение кластера невозможно. Таким образом, все заявки, при увеличивающейся интенсивности потока и превышении квоты на размер инфраструктуры, будут получать отказ в связи с перегрузкой инфраструктуры сервиса. Предлагается рассмотреть случай, когда квота не превышена, а кластер заведомо спроектирован, чтобы обслуживать определённое количество поступающих запросов в единицу времени.

Функциональная модель сервиса изображена на рис. 2, где поток заявок на входе в систему – пуассоновский, с интенсивностью λ ; r – вычислительная мощность, выраженная в числе виртуальных ядер (vCPU) в момент времени, предоставленная одной заявке, а C – суммарная вычислительная мощность, имеющаяся в кластере, выраженная в числе виртуальных ядер (vCPU) в момент времени [5-9].

Рис. 2. Функциональная схема облачного сервиса

Описание математической модели

Рассмотрим модель кластера, изображённого на рис. 2. В качестве обслуживающего ресурса предлагается CPU, как основной ресурс при обработке HTTP запросов. В зависимости от типа системы CPU можно интерпретировать как vCPU/Core – виртуальные ядра облачного сервера или hyperthreads – в «железной» версии кластера в центре обработки данных, построенных на процессорах Intel [2]. Так как в работе рассматривается облачный кластер, в качестве ресурса далее CPU следует понимать как vCPU.

В современных реальных системах есть возможность лимитировать ресурсы под отдельные приложения. В качестве единиц вычислительного ресурса при таком ограничении выступает vCPU. В общем случае эта величина может быть дробной и является абсолютной, а не относительной, вне зависимости от типа используемого оборудования. Обозначим через r максимальную вычислительную мощность, предоставляемую для обслуживания одной заявки. Для r должно выполняться условие: $r \leq C$, где C – вычислительная мощность всего кластера.

Поступление заявок в кластер описывается пуассоновским законом с интенсивностью λ . Величина запрашиваемого на обслуживание заявки ресурса имеет экспоненциальное распределение со средним значением F vCPU. Время обслуживания с предоставлением вычислительного ресурса в r (vCPU) также распределено экспоненциально, со средним значением $\frac{1}{\mu} = \frac{F}{r}$. Для удобства расчётов рассматривать случай, когда C нацело делится на r . Через $v = \frac{C}{r}$ обозначим максимальное число заявок, которое может обработать кластер, предоставляя фиксированное количество ресурсов r (vCPU). Через i обозначим число заявок, уже находящихся на обработке в системе, на момент поступления в неё новой. Новая заявка допускается к обслуживанию, если выполняется неравенство $i < v$. Длительность обслуживания в этом случае имеет экспоненциальное распределение со средним временем $\frac{1}{\mu}$. Если выполняется неравенство $i \geq v$, то заявка также принимается к обслуживанию, но в этом случае вычислительная мощность делится поровну между $(i+1)$ -й заявкой, обрабатываемыми в системе. Таким образом, в случае, когда $i \geq v$, каждая заявка получает вычислительную мощность в размере $\frac{C}{i+1}$. Время обслуживания также имеет экспоненциальное распределение со средним значением $(i+1) \cdot \frac{F}{C} = (i+1) \cdot \frac{1}{v\mu}$ [10].

Через $i(t)$ обозначим количество заявок, находящихся в системе на обслуживании в момент времени t . Тогда динамика изменения состояний рассматриваемой модели задаётся случайным процессом $r(t) = i(t)$, определённым на бесконечном пространстве состояний $S = \{(i), i = 0, 1, \dots\}$. В такой модели все случайные величины имеют экспоненциальное распределение, а текущее состояние системы не зависит от её предыдущих состояний, поэтому процесс $r(t)$ можно считать марковским. На рисунке 3 изображены направления переходов между состояниями процесса $r(t)$ и их интенсивности.

Анализируя диаграмму можно заметить, что она идентична диаграмме переходов для модели $M/M/v$. Обозначим

её через $M/M/v - PS$. Таким образом, можно предполагать и статистическую эквивалентность случайных процессов, описывающих число заявок на обслуживании в момент времени t в моделях $M/M/v$ и $M/M/v - PS$. Поэтому можно считать, что значения стационарных вероятностей $p(i)$, когда система находится в состоянии (i) , для этих моделей будут совпадать [8-11, 13].

Рис. 3. Диаграмма переходов между состояниями случайного процесса $r(t)$, отражающая изменения этих состояний при обслуживании заявок кластером в режиме Processor Sharing с ограничением обслуживающего ресурса

Расчет показателей обслуживания заявок

Через $\beta = \lambda F$ обозначим интенсивность поступающего потока заявок, выраженную в RPS (requests per second). Выразим эту величину через среднее число занятых ресурсных блоков (vCPU) в единицу времени. Таким образом, исходя из определения предложенного трафика, $a = \frac{\beta}{r}$. Для стационарного режима нужно, чтобы выполнялось неравенство $a < v$. Обозначим через $\rho = \frac{\beta}{c}$ коэффициент потенциальной загрузки вычислительного кластера. Через i обозначим количество потоков заявок, находящихся на обслуживании. Тогда каждому потоку предоставляется фиксированная вычислительная мощность, равная r (vCPU), если $ir \leq C$. При $ir > C$ значение фиксированной вычислительной мощности для каждого потока будет равным $\frac{C}{i}$ (vCPU). Построим систему уравнений равновесия для оценки ненормированных вероятностей $P(i)$.

$$P(i) = P(i-1) \frac{a}{i}, i = 1, 2, \dots, v; \quad (1)$$

$$P(i) = P(i) \rho, i = v+1, v+2, \dots$$

Дополнив приведённые выражения условием нормировки построим систему уравнений для явного нахождения вероятности нахождения заявки, поступившей на обслуживание, в состоянии i [11].

$$p(i) = \begin{cases} p(0) \frac{a^i}{i!}, i = 0, 1, \dots, v; \\ p(0) \frac{a^v}{v!} \rho^{i-v}, i = v+1, v+2, \dots, \end{cases} \quad (2)$$

где

$$p(0) = \frac{1}{\sum_{i=0}^{v-1} \frac{a^i}{i!} + \frac{a^v}{v!} \frac{1}{1-\rho}}.$$

Через π обозначим долю времени, когда кластер находится в режиме «насыщения», т.е. когда ресурс, выделяемый на обслуживание одной заявки составляет менее r (vCPU) [12-13].

Таким образом:

$$\pi = \sum_{i=v+1}^{\infty} p(i) = p(0) \frac{a^v}{v!} \frac{\rho}{1-\rho} = \quad (3)$$

$$= \frac{\frac{a^v}{v!} \frac{\rho}{1-\rho}}{\sum_{i=0}^{v-1} \frac{a^i}{i!} + \frac{a^v}{v!} \frac{1}{1-\rho}} = \rho D(v, a),$$

где $D(v, a)$ – вторая формула Эрланга. $D(v, a)$ можно выразить как:

$$D(v, a) = \frac{vE(v, a)}{v - a(1 - E(v, a))},$$

где $E(v, a)$ – первая формула Эрланга

$$E(v, a) = \frac{\frac{a^v}{v}}{1 + a + \frac{a^2}{2!} + \dots + \frac{a^v}{v!}}$$

Обозначим через L среднее число заявок, находящихся на обслуживании, а через W – среднее время обработки заявки вычислительным кластером. Таким образом, интерпретируя (1), получим выражение для определения L [5, 11]:

$$L = \sum_{i=1}^{\infty} p(i)i = a + \sum_{i=v+1}^{\infty} p(i)(i-v) = \quad (4)$$

$$= a + p(0) \frac{a^v}{v!} \sum_{i=1}^{\infty} p^i i = a + p(0) \frac{a^v}{v!} \frac{\rho}{(1-\rho)^2} =$$

$$= a \left(1 + D(v, a) \frac{1}{v(1-\rho)} \right).$$

Из формулы Литтла можно вывести соотношение для определения среднего времени обработки заявки W [11]:

$$W = \frac{L}{\lambda} = \frac{1}{\mu} \left(1 + D(v, a) \frac{1}{v(1-\rho)} \right) = \quad (5)$$

$$= \frac{F}{r} + D(v, a) \frac{F}{C(1-\rho)}.$$

Стоит отметить, что рассматривается поток заявок с ожиданием и бесконечной очередью. То есть, буфер имеет неограниченный объем. Таким образом, ρ при максимальных значениях будет асимптотически приближаться к 1 [5-8]. Исходя из описанных ранее формул, пропускная способность для каждого потока заявок определяется выражением:

$$\vartheta = \frac{F}{W} = \frac{r}{\mu} \cdot \frac{v\mu(1-\rho)}{v(1-\rho) + D(v, a)} = \quad (6)$$

$$= \frac{F}{v(1-\rho) + D(v, a)}$$

Относительная эффективность обслуживания определяется из соотношения:

$$e = \frac{v(1-\rho)}{v(1-\rho) + D(v, a)} \quad (7)$$

Для расчёта представленных характеристик была написана программа на языке Python. С помощью программы рассчитаны пропускная способность ϑ , выделяемая для заявок каждого потока и относительная эффективность обслуживания для кластера в зависимости от размера поступающей нагрузки. При расчётах учтено, что одна заявка от пользователя требует 0.001 vCPU на обслуживание, что близко к реальным значениям при обслуживании вычислительной нагрузки кластером серверов. В реальных системах наиболее просто эту величину определить экспериментально, так как она зависит от многих параметров, включая размер прикреплённой к запросу нагрузки и сложность цепочки вычислений на серверной стороне приложения, которую запускает поступивший запрос. Также, стоит отметить, что полученные зависимости рассчитывались для кластера суммарной

вычислительной мощностью 10 vCPU и потока поступающих заявок, величины, варьирующейся от 0 до 10000 запросов в секунду с шагом в 500.

Построим зависимость пропускной способности кластера от коэффициента загрузки линии и размеров вычислительного кластера на основе полученных данных. График этой зависимости изображён на рис.4. Выделенный жирной линией график отражает случай полного доступа заявок одного потока ко всему вычислительному ресурсу. На рис.5 изображена зависимость относительной эффективности обслуживания от ρ и r .

Из построенных графиков можно сделать вывод, что при низкой загрузке кластера каждый запрос, поступающий на обслуживание, получает вычислительную мощность, близкую суммарной вычислительной мощности кластера. При пропускная способность падает и заявка получает близкое к 0 количество vCPU. Так же стоит отметить, что чем меньше количество предоставляемого одной заявке вычислительного ресурса, тем более стабильно ведёт себя характеристика, сохраняя значения, близкие к 1 при достаточно большом диапазоне значений загрузки кластера.

Рис. 4. Зависимость пропускной способности одного потока загрузки линии и размеров вычислительного кластера

Рис. 5. Зависимость относительной эффективности обслуживания заявок кластером от ρ и r . Выделенный график – полное предоставление ресурсов кластера одной заявке

Выводы

В ходе проведенного исследования были рассмотрены особенности функционирования облачного вычислительного кластера. Построена математическая модель, описывающая обслуживание моносервисного трафика веб-приложения

вычислительным кластером. Выбраны параметры, позволяющие сформировать предварительный прогноз по качеству обслуживания заявок кластером и оценить параметры функционирования кластера при разной нагрузке. Написана программа на языке Python для расчёта этих параметров. Рассчитаны параметры относительной эффективности обслуживания запросов кластером и пропускной способности для каждого потока.

Полученные результаты можно использовать для дальнейших исследований в этой области. Программу планируется использовать для расчёта характеристик кластера и выявления закономерностей изменения этих характеристик на основе заданных значений ожидаемой величины потока заявок к приложению и величины вычислительного ресурса, требуемого для обслуживания одной заявки.

В рамках дальнейших исследований планируется выявить закономерности изменения вычислительного ресурса от характеристик заявки для более точного вычисления параметров качества обслуживания. Также есть возможность усложнить модель для оценки параметров качества обслуживания при обработке потока заявок мультисервисного трафика.

Литература

1. RFC 72, HTTP/1.1 Semantics and Content, June 2014
2. Kubernetes Documentation ges. Managing Compute Resources for Containers. Resource units in Kubernetes. 13 апреля (<https://kubernetes.io/docs/concepts/configuration/manage-compute-resources-container/>)
3. Google Kubernetes Engine documentation pages. 18 апреля (<https://cloud.google.com/kubernetes-engine/docs>)
4. Limoncelli T.A., Chalup S.R., Hogan C.J. The Practice of Cloud System Administration – Addison-Wesley. 2015. 524 p.
5. Степанов С.Н. Теория телетрафика: концепции, модели, приложения / Серия «Теория и практика инфокоммуникаций». М.: Горячая линия – Телеком, 2015. 868 с.
6. Степанов С.Н., Иверсен В.Б. Способы уменьшения объема вычислений при расчете моделей систем связи с потерями, основанные на игнорировании маловероятных состояний // Проблемы передачи информации. 2001. Том. 37. Вып.3. С. 82-95.
7. Iversen V.B., Stepanov S.N., Kostrov A.V. Dimensioning of Multi-service Links Taking Account of Soft Blocking. Lecture Notes in Computer Science. 2006. Vol. 4003 LNCS. P. 3-10.
8. Степанов С.Н., Степанов М.С. Планирование ресурса передачи при совместном обслуживании мультисервисного трафика реального времени и эластичного трафика данных // Автоматика и телемеханика. 2017. № 11. С. 79-93.
9. Iversen V.B., Stepanov S.N. The optimal dimensioning of multi-service links. Proceedings from COST285 Mid-term Symposium, Munchen, September 8-10, 2005. Chapter 7 (pp. 151-178) in A. Nejat Ince & Ercan Topuz (editors): "Modeling and simulation tools for emerging telecommunication networks". Springer 2006. 510 p.
10. Bonald T., Roberts J. Internet and the Erlang formula // ACM SIGCOMM Computer Communication Review. 2012. Vol. 42. No. 1. P. 22-30.
11. Bonald T., Feuillet T. Network Performance Analysis – ISTE Ltd 2011, DOI:10.1002/9781118602911. 253 p.
12. Степанов С.Н. Основы телетрафика мультисервисных сетей. М.: Эко-Трендз. 2010. 392 с.
13. Степанов С.Н., Савенков Ю.Ю. Использование кластеризации информационных потоков для повышения эффективности занятия канального ресурса в мультисервисных сетях // Электро-связь. 2010. № 11. С. 22-25.

MATHEMATICAL MODELING OF REQUESTS FLOW TO CLOUD COMPUTE CLUSTER

Dmitriy O. Kupriyanov,

Moscow Technical University of Communications and Informatics, Moscow, Russia,

dim.kupro@gmail.com**Abstract**

Quality of service parameters estimation becomes even more valuable when using cloud compute services. Mathematical model of described cloud-deployed web application in terms of Processor Sharing (PS) policy for mono-service traffic type proposed in this research. This model has Poisson distribution of the incoming requests flow with intensity λ . Requests awaiting in queue, queue length is considered to be unlimited and all requests should be served. Request in this system is HTTP request with a special payload in JSON format. The size of this payload is different for each request but it lies in a narrow band of values (bytes or decades of bytes). A model of cloud compute cluster was built. Characteristics of relative serving efficiency and relative bandwidth of a single requests flow was calculated using this model for different amount of resource provided for processing of a single request. The dependency of these characteristics from cluster load coefficient is demonstrated in charts. Some conclusions on cloud cluster QoS parameters behavior after the change of input requests flow size. Proposed model helps estimating quality of service parameters and adopting the infrastructure to increased or decreased number of requests from customers and could be used for architecting, deploying and administrating web services.

Keywords: cloud computing, web applications infrastructure, Markovian processes, Processor Sharing, queuing theory, qos.

References

1. RFC 7231, HTTP/1.1 Semantics and Content, June 2014
2. Kubernetes Documentation pages. Managing Compute Resources for Containers. Resource units in Kubernetes. April, 13 (<https://kubernetes.io/docs/concepts/configuration/manage-compute-resources-container/>)
3. Google Kubernetes Engine documentation pages. April, 18 (<https://cloud.google.com/kubernetes-engine/docs>)
4. Limoncelli T.A., Chalup S.R., Hogan C.J. The Practice of Cloud System Administration - Addison-Wesley. 2015. 524 p.
5. Stepanov S.N. Teletraffic theory: concepts, models, annexes / Series "Theory and practice of infocommunications". Moscow: Hot Line - Telecom, 2015. 868 p.
6. Iversen V.B., Stepanov S.N. Methods of computation part decrease for calculating telecommunication models with losses, based on ignoring states with low probability. *Information transfer problems*. 2001. Vol. 37. No.3. P. 82-95. (in Russian)
7. Iversen V.B., Stepanov S.N., Kostrov A.V. Dimensioning of Multiservice Links Taking Account of Soft Blocking. *Lecture Notes in Computer Science*. 2006. Vol. 4003 LNCS. P. 3-10.
8. Stepanov S.N., Stepanov M.S. Resource planning for joint multiservice real time traffic and elastic data traffic. *Automatics and telemechanics*. 2017. No. 11. P. 79-93. (in Russian)
9. Iversen V.B., Stepanov S.N. The optimal dimensioning of multi-service links. Proceedings from COST285 Mid-term Symposium, Munchen, September 8-10, 2005. Chapter 7 (pp. 151-178) in A. Nejat Ince & Ercan Topuz (editors): "Modeling and simulation tools for emerging telecommunication networks". Springer 2006. 510 p.
10. Bonald T., Roberts J. Internet and the Erlang formula. *ACM SIGCOMM Computer Communication Review*. 2012. Vol. 42. No. 1. P. 22-30.
11. Bonald T., Feuillet T. Network Performance Analysis - ISTE Ltd 2011, DOI:10.1002/9781118602911. 253 p.
12. Stepanov S.N. Basics of multiservice networks teletraffic. Moscow: Eco- Trends. 2010. 392 p. (in Russian)
13. Stepanov S.N., Savenkov Y.Y. Informational flows clustering for increasing efficiency of channel resource usage in multiservice networks. *Telecommunications*. 2010. No. 11. P. 22-25. (in Russian)

Information about author:

Dmitriy O. Kupriyanov, postgraduate student, Moscow Technical University of Communications and Informatics, Moscow, Russia