ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ ВЫСОКОПРОИЗВОДИТЕЛЬНОЙ СИСТЕМЫ, ОСНОВАННОЙ НА ПРОГРАММНО-КОНФИГУРИРУЕМОЙ СЕТИ

П.Н. Полежаев, А.С. Кириллов, Ю.А. Ушаков, Л.В. Легашев (Оренбург)

В настоящее время перспективным направлением в области компьютерных сетей является использование программно-конфигурируемых сетей (ПКС – Software Defined №etwork). Принципы ПКС впервые возникли в исследовательских лабораториях Стэнфорда и Беркли [1] и в настоящее время развиваются в рамках консорциума Open №etwork Foundation и европейского проекта OFELIA. В основе подхода ПКС лежит возможность динамически управлять пересылкой данных в сети с помощью открытого протокола OpenFlow. Все активные сетевые устройства объединяются под управлением контроллера OpenFlow, который обеспечивает приложениям доступ к управлению сетью.

Каждый коммутатор OpenFlow имеет таблицу потоков, содержащую правила обработки пакетов. Любое правило включает две части – признаки заголовков пакетов и набор действий. При поступлении в коммутатор нового пакета, происходит сопоставление его заголовков с признаками правил в таблице. В случае совпадения выполняются все действия из соответствующего набора. Если подходящее правило в таблице отсутствует, то пакет передается контроллеру OpenFlow. Контроллер принимает решение о дальнейших действиях над пакетом, которое реализуется в виде команды передачи пакета на определенный порт коммутатора и/или в установке для пакета нового правила в таблицу данного и, возможно, других коммутаторов.

Признаки заголовков позволяют управлять пакетами на уровнях L1-L3 модели OSI. В число возможных действий над пакетом входит его передача на заданный порт, на все порты, удаление, изменение его заголовка и т.п. Это дает возможность реализовывать эффективные протоколы маршрутизации, включая многопутевую маршрутизацию, управлять потоками на основе приоритетов, осуществлять виртуализацию сети, обеспечивать QoS, эффективно распределить нагрузку на сеть и т.п.

В данном исследовании рассматриваются высокопроизводительные вычислительные системы, использующие ПКС. Для них в рамках этого проекта разрабатываются [2]: алгоритмы планирования задач с учетом топологии системы, сетевой конкуренции и существующих коммуникационных схем исполняемых задач; методы управления потоками данных между процессами вычислительных задач, основанные на использовании ПКС.

Основной целью алгоритма планирования задач является использование информации о топологии системы, состоянии сети и коммуникационных схемах вычислительных задач для компактного назначения их процессов на вычислительные узлы. Это обеспечивает снижение сетевой конкуренции, как следствие, уменьшение времени выполнения задач и повышение загруженности вычислительных ресурсов системы.

Главной идеей предлагаемых методов управления потоками данных является использования коммуникационных схем вычислительных задач с целью прокладки маршрутов передачи данных между их процессами до запуска. Подобный подход реализует проактивную схему маршрутизации данных в ПКС, а также обеспечивает снижение сетевой конкуренции. Отсутствие аналогов в научных источниках определяет новизну предлагаемых алгоритмических решений.

Для экспериментального исследования эффективности разрабатываемых алгоритмов планирования задачи и методов маршрутизации данных необходимо создание симулятора высокопроизводительной системы, основанной на ПКС. В его основе имитационная модель, описываемая в рамках настоящей статьи. В этом заключается практическая значимость данной модели.

На рис. 1 приведена имитационная схема управляющей системы.

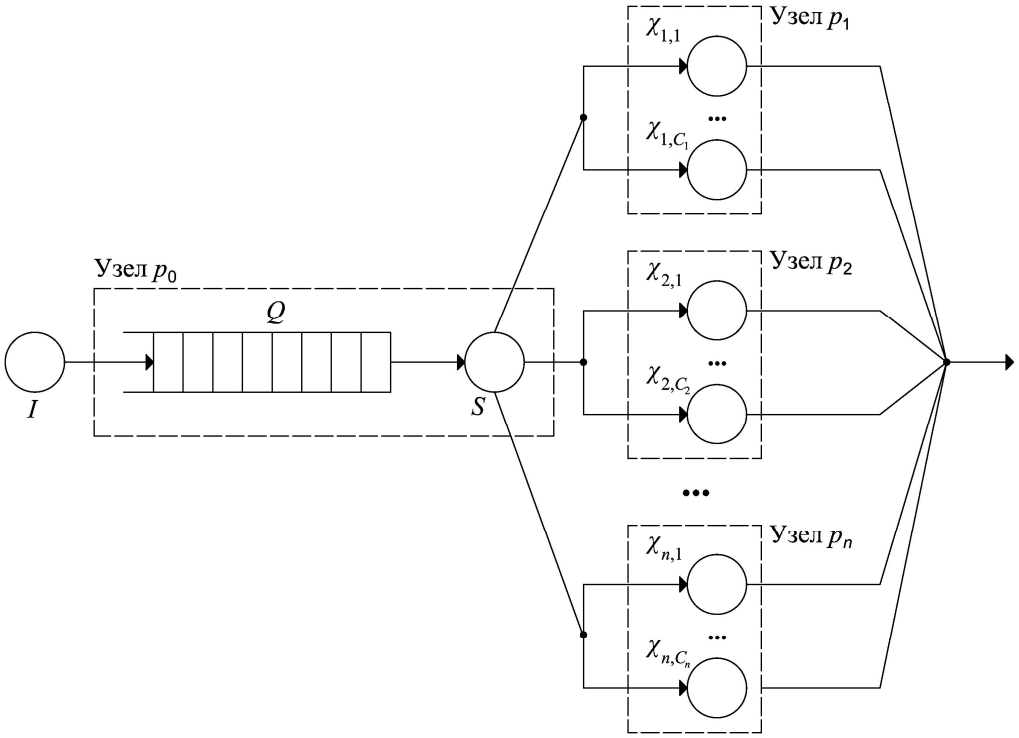

Рис. 1. Имитационная схема управляющей системы

Источник I генерирует поток параллельных задач, отправляемых пользователями в очередь Q управляющего узла p_0. Канал S представляет собой планировщик, который в соответствии с заложенным в него алгоритмом осуществляет извлечение задач из очереди Q и назначение их на свободные вычислительные ядра подходящих по конфигурации узлов.

Моделируемая сеть высокопроизводительной системы может быть описана неориентированным графом:

$$G = (D, L), \tag{1}$$

где D – множество вершин, представляющих собой сетевые устройства: вычислительные узлы p_i, управляющий узел p_0, обычные коммутаторы k_i (без поддержки OpenFlow), коммутаторы OpenFlow f_i и контроллер OpenFlow Ω. L – множество сетевых связей межу устройствами.

Имитационная схема вычислительного узла p_i приведена на рис. 2, p_i соединен с другими узлами и коммутаторами вычислительной сети с помощью r_i дуплексных связей. Все входящие пакеты, а также пакеты сообщений, генерируемых процессами, выполняющимися на локальных вычислительных ядрах, сначала поступают в очередь $Q_{inp.i}$, а затем маршрутизируются каналом обслуживания R_i. Если соответствующий пакет предназначен для локального вычислительного ядра, то он передается ему непосредственно, иначе – помещается в одну из очередей $Q_{out.i,1}, Q_{out.i,2}, \ldots, Q_{out.i,r_i}$, соответствующую выбранной алгоритмом маршрутизации исходящей связи.

При выполнении маршрутизации каналом R_i моделируется временная задержка. Заметим, если два процесса одной задачи выполняются на соседних ядрах узла, то обмен пакетами между ними также осуществляется через очередь $Q_{inp.i}$ и канал R_i.

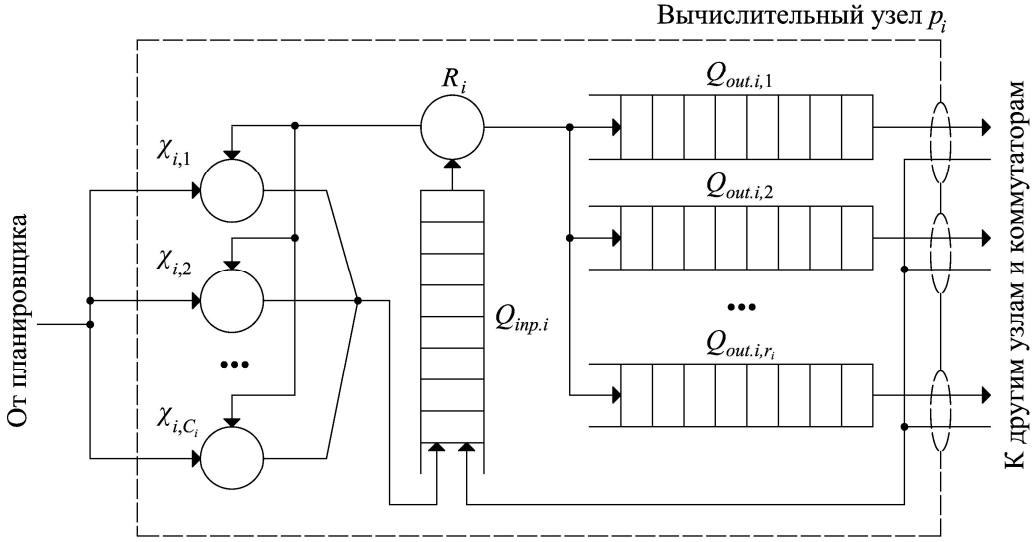

Рис. 2. Имитационная схема вычислительного узла

Имитационная схема коммутатора, приведенная на рис. 3, представляет собой урезанный вариант имитационной схемы узла. Отличие только в том, что коммутатор не может выполнять вычисления и поэтому у него отсутствуют вычислительные ядра.

На рис. 4. приведена схема работы дуплексной связи $L_{ij} = \{E_{ij}, E_{ji}\}$, соединяющей сетевые устройства $d_i \in D$ и $d_j \in D$. Каналы обслуживания E_{ij} и E_{ji} добавляют к времени передачи пакета задержку величиной $b(E_{ij}) = b(E_{ji})$.

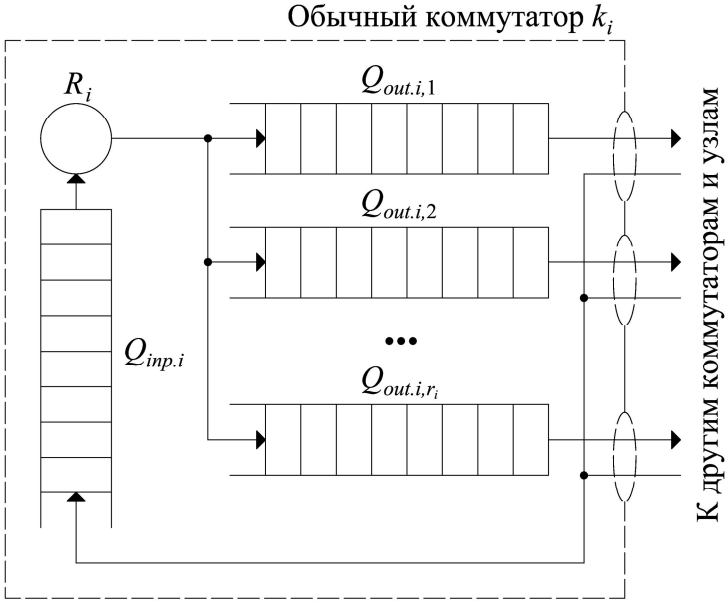

Рис. 3. Имитационная схема обычного коммутатора

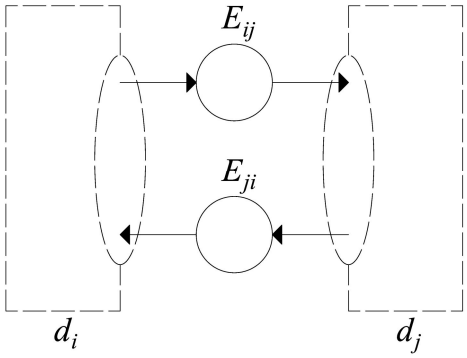

Рис. 4. Имитационная схема сетевой связи

Коммутатор OpenFlow f_i отличается от обычного коммутатора тем, что его канал обслуживания R_i^{of} реализует поведение модуля OpenFlow – при получении нового пакета поиск и выполнение правил в таблице потоков, в случае отсутствия подходящего правила R_i^{of} отправляет пакет контроллеру OpenFlow, поместив его в очередь $Q_{out.i,r_i+1}^{of}$. Ответные команды контроллера в виде пакетов помещаются в очередь $Q_{inp.i}^{of}$, в которой для них задается максимальный приоритет обслуживания.

Контроллер OpenFlow установлен на выделенный сервер, его иммитационная схема аналогична имитационной схеме вычислительного узла (см. рис. 6). Отличие в том, что какнал R^{con}, помимо стандартной маршрутизации пакетов, помещает запросы к контроллеру OpenFlow в отдельную очередь Q_{req}^{con}. Канал обслуживания L^{con} реализует логику работы контроллера по принятию решений о дальнейших действиях с поступившим пакетом. Результатом его работы являются управляющие пакеты OpenFlow, передаваемые через Q_{inp}^{con} и R^{con} коммутаторам OpenFlow с целью отправки пакета на определенный порт или установки новых правил в их талицы потоков. В Q_{inp}^{con} и очередях портов такие пакеты имеют максимальный приоритет.

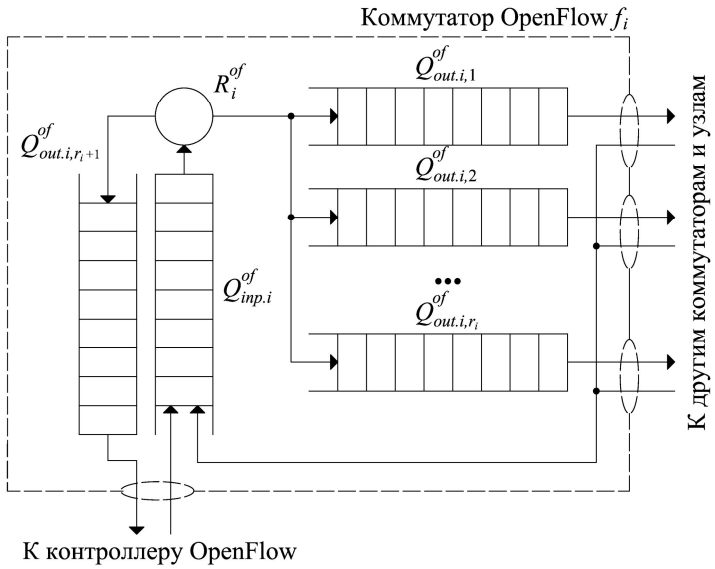

Рис. 5. Имитационная схема коммутатора OpenFlow

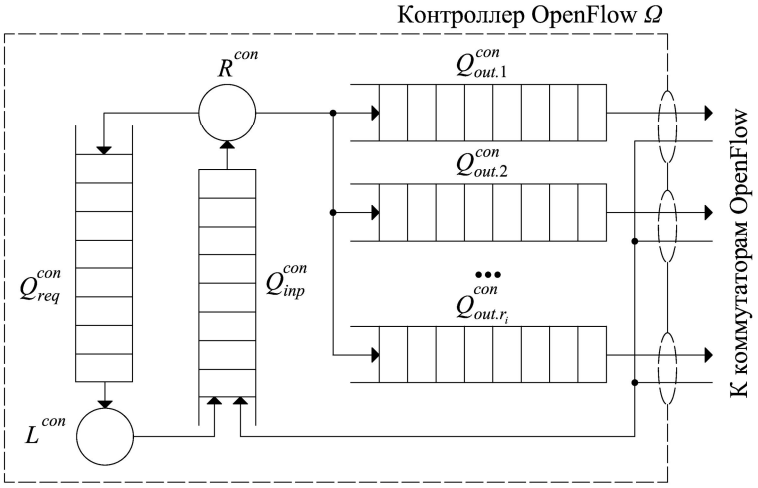

Рис. 6. Имитационная схема контроллера OpenFlow

Поток вычислительных задач, сетевого трафика, характеристики каналов обслуживания являются случайными величинами. Наилучшие подобранные законы их распределения описаны в работе [3].

Разработанная имитационная модель высокопроизводительной системы учитывает передачу пакетов данных по сети и работу протокола OpenFlow. Ее реализация в виде симулятора высокопроизводительной системы позволит исследовать эффективность предлагаемых в рамках проекта алгоритмических решений.

Исследования выполнены при поддержке РФФИ (проекты №12-07-31089 и № 13-07-97046).

Литература

1. **McKeown №., Anderson T., Balakrishnan H., Parulkar G., Peterson L., Rexford J., Shenker S., Turner J.;** Openflow: enabling innovation in campus networks // ACM SIGCOMM Computer Communication Review, 2008, vol. 38, p. 69–74.

2. **Полежаев П.Н**. Математическая модель распределенного вычислительного центра обработки данных с программно-конфигурируемыми сетями его сегментов // Вестник Оренбургского государственного университета. – 2013. – 5(154) – С. 198–204.

3. **Полежаев П.Н.** Разработка симулятора вычислительной грид-системы // Информационные технологии моделирования и управления. – 2012. – № 6(78). – С. 498–505.