УДК 004.057.4+004.724

А.С. Краевой

ОБ ОДНОМ МЕТОДЕ ПОСТРОЕНИЯ СИСТЕМЫ МОДЕЛИРОВАНИЯ ДЕЦЕНТРАЛИЗОВАННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

Введение

Запросы современных приложений к вычислительной мощности сред выполнения часто требуют объединения рабочих станций для повышения общей производительности. Создание высокопроизводительных систем подразумевает объединение разнородных вычислительных ресурсов в однородную логическую систему. В качестве примеров можно привести среду Grid-вычислений LHC Computing Grid [1] и систему доставки контента Limelight Networks [2].

При росте размера системы большинство известных централизованных решений в области управления ресурсами оказываются неприменимыми. Следовательно, большую практическую ценность представляют исследования в области децентрализованных вычислительных систем. Передовым направлением таких исследований являются разработки децентрализованных версий различных алгоритмов и протоколов, их качественное описание и возможные сравнения статистиче-ских характеристик.

Имитационное моделирование является популярным средством прогнозирования характеристик поведения децентрализованных систем. Такая популярность связана с тем, что затраты на реализацию системы моделирования существенно меньше цены полномасштабного эксперимента на реальной системе. Также отметим, что статистические данные, полученные имитационным моделированием, часто бывают более значимыми, нежели полученные из абстрактной математической модели.

Целью настоящей работы является выделение типичных задач, связанных с разработкой системы моделирования децентрализованных вычислительных сред и описание авторских решений

 ISSN 1562-9945

некоторых из этих задач. Дополнительно в работе приводится обзор предметной области децентрализованных вычислительных сред и анализ актуальных в настоящий момент программных продуктов-аналогов.

Общие принципы

Так как в децентрализованной системе нет возможности получить информацию о структуре всей вычислительной сети в целом, каждый узел использует для решения специфических прикладных задач децентрализованные протоколы обнаружения других компонент системы. Первые версии таких протоколов были упрощенными и не предполагали определенной логической организации узлов. Поиск в такой системе приводил к экспоненциальному росту передач сообщений при линейном росте количества узлов сети, так как на каждом узле было несколько вариантов возможной ретрансляции поискового запроса.

Следовательно, актуальность приобрели способы улучшения масштабируемости механизмов поиска. Для этого используются определенные адресные пространства, задающие отношения порядка на множестве ключей. Такие пространства позволяют задать направленность поиска в контексте отдельного узла, улучшить сходимость децентрализованного поиска и ограничить размер маршрутных таблиц узла.

Если рассмотреть маршрутную таблицу узла как список ссылок на другие узлы, мы получим ориентированный граф. В этом орграфе роль вершин играют узлы, а ребра соответствуют записям (ссылкам) в маршрутной таблице. По сути, такой граф является сложной сетью логических связей и взаимодействий узлов, часто используемой при оценке и оптимизации работы децентрализованной системы в целом.

Характеристики работы системы тесно связаны со свойствами оверлейной сети. Например, количество ретрансляций поискового запроса связано с распределением длин путей в оверлейной сети, а размер маршрутной таблицы соответствует степени вершины оверлейной сети. В эффективной системе средние значения ранее упомянутых величин ограничены сверху логарифмическими выражениями от количества узлов [3]. Чаще всего есть возможность установить характер таких зависимостей аналитическими методами,

но исследователи часто подтверждают аналитическую гипотезу с помощью имитационного моделирования.

Так как децентрализованные системы часто используют ненадежные компоненты, оверлейная сеть подвергается значительным «возмущениям» в связи со сбоями узлов или соединений. Потому большой интерес представляет вычисление определенной меры затрат, необходимых на нормализацию сети, и измерение вероятности сохране-ния связности сети в зависимости от вероятности сбоев узлов.

В большинстве случаев такие измерения имеют обобщенный статистический характер, где влияние детальных характеристик сетевого и транспортного уровня сказывается достаточно мало. Отметим, что при необходимости учесть такие характеристики возможно использовать данные о топологии сети Интернет [4]. Кроме того, теория сложных сетей в настоящий момент предлагает модели, позволяющие генерировать достоверные топологии глобальных вычислительных сетей.

В настоящий момент известны исследования алгоритмов адаптации оверлейной сети к структуре физической сети, стратегий управления нагрузкой и улучшения статистических характеристик структуры оверлейной сети. Таким образом, имитационное моделирование является важным средством исследования децентрализованных стратегий и протоколов, применимым для широкой области практических задач.

Требования к системе моделирования

Основное требование к любой системе моделирования – максималь-ное приближение к реальной среде выполнения. В случае децентрализованных систем это требование связано с возможностью использо-вания большинства компонент системы моделирования в реальной децентрализованной системе. В качестве примера приведем систему моделирования OverlayWeaver [5], в которой исследуемый алгоритм построения оверлейной сети можно выполнить в реальной системе без изменений. Для этого все программными компоненты системы моделирования, абстрагирующие определенные детали среды выполнения просто заменяются реальными компонентами.

Многие сценарии использования существующих систем предполагают одновременное выполнение децентрализованного

протокола на тысячах узлов. Естественно, что в такой ситуации необходимо оценивать характеристики масштабируемости системы моделирования. Следовательно, актуальность приобретает разработка возможных методов параллельной обработки моделируемых сценариев.

Отметим также, что имитация распределенной системы большого масштаба требует оценок реалистичности используемых моделей физической сети с точки зрения ее статистических характеристик, таких как характер распределений степеней связности, длин маршрутов, кластеризация. Потому актуальны интеграции системы моделирования со средствами анализа сложных сетей и возможность исполь-зования заданных топологий аппаратного уровня сетевой среды, на-пример – структуры сети Интернет на уровне автономных систем [4].

Прикладные задачи в децентрализованной системе опираются на распределенные службы: маршрутизацию поисковых запросов, хеширование (т.е. поиск данных по ключу), доставку групповых сообщений. Качество работы этих служб зависит от характеристик структуры оверлейной сети. Следовательно, эффективная реализация этих служб является самостоятельной исследовательской задачей, опосредованно влияющей на характеристики работы задач прикладного уровня.

Отметим, что работа указанных выше базовых алгоритмов требует значимой доли вычислительных ресурсов. Например, алгоритмы обнаружения структуры физического уровня сети используют ICMP-сообщения, объем которых может составлять до 10% от общего объема передаваемой системой информации [6]. Таким образом, повторное использование не только обеспечивает простоту миграции моделируемого алгоритма в реальную среду выполнения, но и позволяет достичь большей эффективности в случае работы нескольких децентрализован-ных приложений в одной сети.

Также отметим следующее важное требование к архитектуре систе-мы моделирования: открытость для изменения существующих и ин-теграции новых реализаций компонент системы: протоколов, алгоритмов, аспектов моделирования с низким риском непредвиденных влияний на поведение системы в целом.

Необходимость поуровневой декомпозиции

Для выполнения вышеизложенных требований используется типичная техника улучшения внутреннего качества программного продукта – поуровневая декомпозиция. При ее применении службы, поддерживаемые децентрализованной системой, разделяются на уровни. Каждый уровень непосредственно использует возможности, предоставляемые нижним уровнем, реализуя собственные службы, абстрагированные от деталей реализации нижележащих (некоторые из которых используются опосредованно).

Таким образом, поуровневая декомпозиция помогает выделить те службы, которые значительно зависят от физического окружения, и в системе моделирования будут соответствовать имитирующим компонентам, равно как и те, которые можно использовать в реальной системе без изменений. Такая декомпозиция является основным способом обеспечения возможности и эффективности повторного использования компонент системы моделирования и децентрализованной системы в целом и открытости программного продукта для изменений.

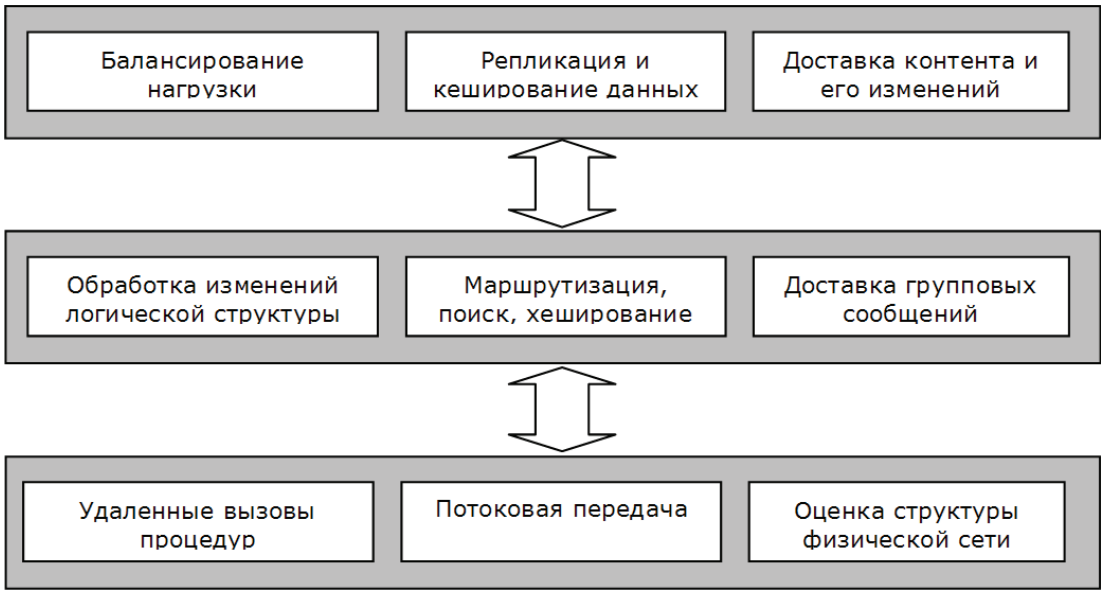

Рис. 1: Поуровневая декомпозиция служб.

Пример поуровневой декомпозиции отображен на рис. 1, на котором также дополнительно показаны типичные высокоуровневые службы, не несущие прикладной специфики. Отметим, что только службы нижнего уровня зависят от физического окружения, остальные возможно использовать повторно.

 ISSN 1562-9945

Следовательно, поуровневая декомпозиция позволяет реализовать многие требования к системе моделирования: открытость для изменений, близость к реальной среде выполнения, возможность повторного использования.

Стандарт Common API

Common API — предполагаемый публичный стандарт базовых операций, используемых для поддержки структуры оверлейной сети [8]. Такой стандарт позволяет обеспечить независимость одновременных исследований протоколов, сервисов и приложений, использующих оверлейные сети, дает возможность непосредственного экспериментального сравнения посредством эксперимента и облегчает разработку компонент сторонними производителями.

Основными механизмами, которые предлагают разработчики Common API, являются интерфейсы Routing и RoutingState, реализующие задачи маршрутизации на уровне всей сети и отдельного узла соответственно (см. рис. 2). Заметим, что концепции адресации и генерации ключей слабо детализированы (Address, Key, NodeHandle). Очевидна некоторая незаконченность и изначальная неэффективность предложений по реализации удаленных вызовов процедур, так как каждое сообщение моделируется с помощью отдельного объекта, реализующего маркерный интерфейс Msg.

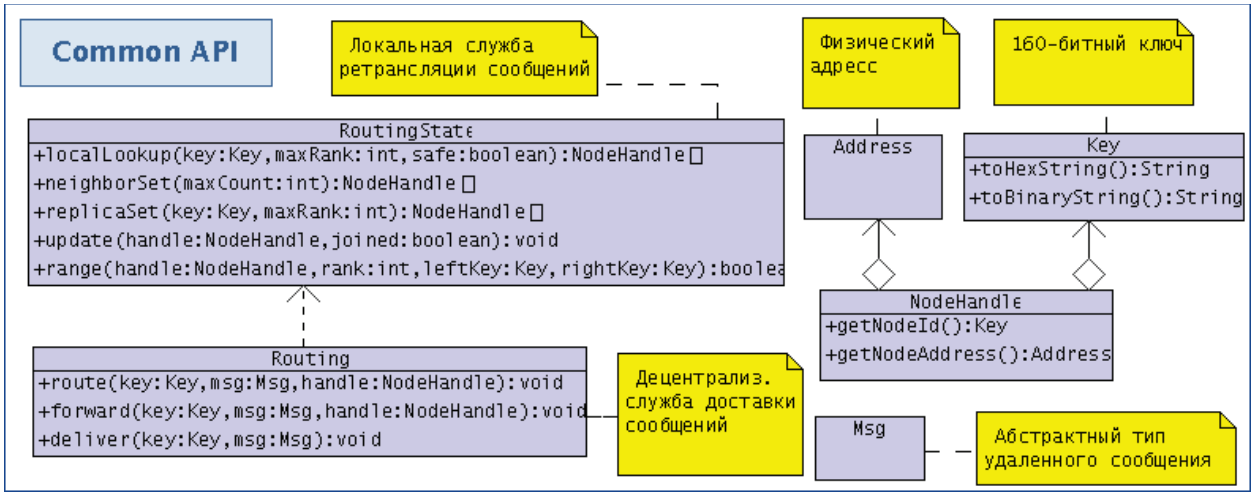

Рисунок 2 - Ключевые компоненты Common API.

Из-за ограничений в объеме материала не будут рассмотрены детали реализации этого стандарта в системах моделирования OverlayWeaver [5] и PlanetSim [7]. Отметим только значительно смешение сущностей в среде PlanetSim, которая приводит к большой

сложности модификации и расширения данной системы. Упомянутая ранее среда OverlayWeaver не реализует Common API, однако в ней представлены многие аналогичные элементы.

Архитектура авторской системы

Предлагаемая автором система моделирования содержит спецификацию Common API, которая в последствии расширяется уточненными службами и типами данных. На рис. 3 представлены службы базового уровня. Одной из основных является служба StorageService, который задает примитивы локального сохранения данных. Отметим, что служба RoutingService в авторской реализации наследует RoutingState, и позволяет обрабатывать большее количество событий.

Отдельно отметим службу RpcService, которая для заданной службы генерирует прокси-объект, обеспечивающий поддержку удаленных вызовов на других узлах. Прокси-объект – объект, генерируемый интроспективными средствами программной среды, позволяющий в обобщенном виде определять способ реализации вызовов определенного интерфейса. В нашем случае вызовы методов этого объекта интерпретируются системой моделирования и отображаются в вызовы соответствующих методов заданной удаленной службы.

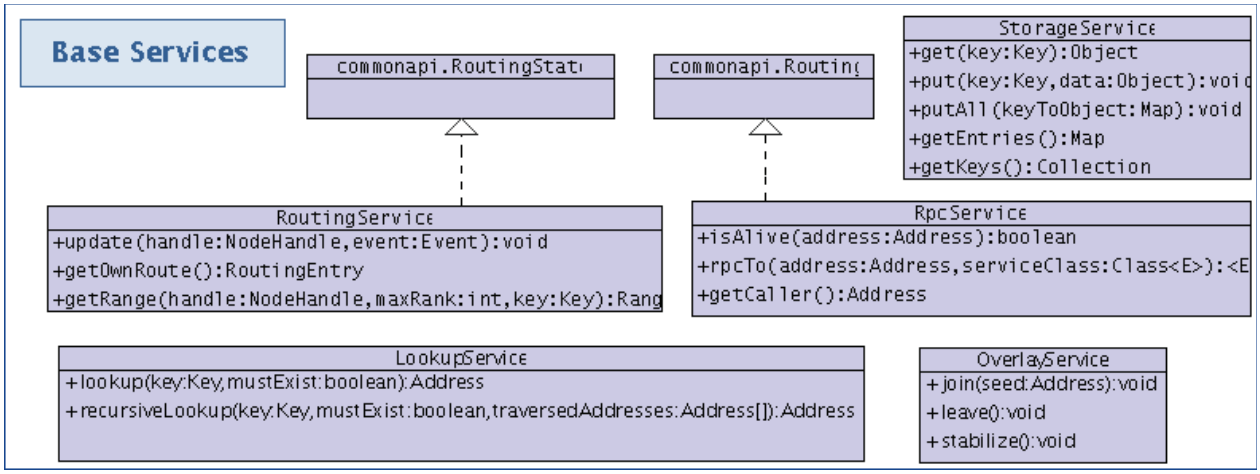

Рисунок 3 - Службы базового уровня

Таким образом, удаленный вызов процедуры в авторской среде моделирования представляет собой вызов метода, а не отдельный объект. Это позволяет сохранить чистоту объектно-ориентированной декомпозиции системы, снизить запросы к подсистеме управления памятью и выделить службу удаленных вызовов, реализация которой

 ISSN 1562-9945

может быть легко изменена при переходе к реальной среде выполнения.

Опишем кратко децентрализованные службы, заданные в предлагаемой автором системе моделирования (см. рис. 3). Служба децентрализованного поиска LookupService позволяет отображать пространство ключей в адреса таким образом, чтобы соответствующий адрес соответствовал узлу, хранящему данный ключ, либо первому из узлов, ответственных за сохранение сегмента пространства ключей, содержащего искомый ключ. Таким образом, эта служба является реализацией службы децентрализованного хеширования.

Служба OverlayService задает базовые события, которые должен обрабатывать узел оверлейной сети. В случае реализации конкретного протокола, задаются дополнительные методы, которые могут вызываться удаленно другими узлами, участвующими в сети. Например, в случае реализации протокола P-Grid [9], могут быть заданы специфичные методы сбалансированных разделений и слияний сегментов пространства ключей, заданные этим протоколом.

Способы имитации дискретных событий

Для обеспечения максимальной правдоподобности необходимо реализовать многопоточную обработку запросов в контексте одного узла, так как в реальных системах параллельный доступ к данным оказывает значительное влияние на архитектуру и поведение системы. Однако, в случае имитации одновременного выполнения нет необходимости запускать для каждого узла определенное количество потоков: как было показано в [10], такая реализация ограничивает размер моделируемой системы (около 1200 узлов). Для достижения необходимого параллелизма достаточно использовать небольшое количество потоков в контексте глобальной очереди событий.

Каждый узел содержит определенное количество данных – чаще всего эти данные содержат информацию о соседних узлах: IP-адрес, позицию соседнего узла в общей логической структуре, задержку передачи сообщений и пр. Так как система моделирования предоставляет базовые механизмы для управления этими данными, необходимо обеспечить безопасность использования таких механизмов в многопоточном контексте.

Система использует несколько потоков, выполняющих задания из одной глобальной очереди событий, что позволяет выполнять

имитационное моделирование в многопоточном режиме, не перегружая ресурсы системы.

Задание сценариев, сбор статистики

Для воспроизводимого и управляемого выполнения моделирования необходимо обеспечить возможность задания сценариев выполнения. Сценарии представляют собой определенную последовательность событий, разделенную на фазы инициализации системы и сбора статистики. Система в настоящий момент поддерживает следующие события: добавление/удаление/сбой узла оверлейной сети, добавление/удаление/запрос элемента данных, и децентрализованный поиск элементов данных.

При этом, события могут быть сгруппированы в последовательности и циклы. Сбой одной из операций цикла приводит к прекращению последовательности, что, обеспечивает однородность собираемой статистики по составным событиям. Для задания сценариев используется язык XML, что обеспечивает некоторую универсальность и удобство программной обработки. Отметим, что в сценариях используются подстановки значений параметров (т.е. ссылки на значения, изменяющиеся в определенном диапазоне), что позволяет разделить описание исследуемого пространства параметров от самих сценариев (см. рис. 4, выделения).

```
<testbench>
      <initEvent>
            <sequence>
                  <join nodeCount="${nodeCount}"/>
                  <loop loopCount="${nodeMappings.static.min}">
                        <putMapEntry entryCount="${nodeCount}"/>
                  </loop>
                  <stabilizeAll/>
            </sequence>
      </initEvent>
      <runEvent>
            <loop loopCount="${loop.static.max}">
                  <loop loopCount="${nodeCount}">
                        <lookupMapEntry/>
                  </loop>
            </loop>
      </runEvent>
</testbench>
```

Рисунок 4 - Пример задания тестового сценария, в котором добавляется варьируемое количество узлов, фиксированное количество элементов данных (в расчете на узел) и производится фиксированное количество случайных поисковых запросов

 ISSN 1562-9945

Пространство параметров также задается посредством документа XML, в котором последовательно описаны параметры и их возможные значения. Такой подход позволяет легко произвести моделирование некоторого тестового сценария для различных наборов параметров, что обеспечивает легкость обнаружения зависимостей между характеристиками системы.

Также система поддерживает сбор статистических данных о структуре логической сети и работе базовых сервисов: модельное время, количество ретрансляций и успешность поисковых запросов, дисперсия количества хранимых элементов данных, степень кластеризации, диаметр оверлейной сети.

Кроме этого, система позволяет провести экспорт данных о логической структуре сети в формате GraphML, что дает возможность в дальнейшем провести анализ этой структуры с помощью известных средств анализа сложных сетей [11], и определить характер влияния показателей эффективности оверлейной сети на конкретные показатели работы системы.

Выводы

В настоящей работе представлены основные детали архитектуры авторской системы имитационного моделирования, типичные требования к системам такого типа и некоторые способы их реализации. Практическая новизна реализации состоит в использовании интроспективных средств языка программирования для реализации службы передачи сообщений. Это позволяет достичь большей производительности и открытости прикладной системы для расширения.

В дальнейшем система может быть расширена за счет реализации служб третьего уровня: управления нагрузкой, репликацией, потоковой передачей. Также большой интерес представляет расширение генерируемых показателей структуры оверлейной сети, например – коэффициентов расширения, ассортативности и подобных им.

Показанное автором решение поуровневой декомпозиции системы представляет практический интерес. Отметим возможность сбора статистических данных на всех уровнях реализации, что позволяет проводить анализ эффективности их работы и оценивать степень зависимости уровней. Принципиальным преимуществом системы

можно считать интеграцию со средствами исследования сложных сетей, которое в настоящий момент слабо представлено в системах-аналогах.

ЛИТЕРАТУРА

1. *Bird I. et al*, LHC computing grid: technical design report // Technical Report CERN-LHCC-2005-024. – CERN European Laboratory for Particle Physics, Geneva. – June 2005. – 153 p.
2. *Pathan A.-M.K., Buyya R.*, A taxonomy and survey of content delivery networks // Technical Report, GRIDS-TR-2007-4. – Grid Computing and Distributed Systems Laboratory, The University of Melbourne, Australia. – 12 Feb. 2007. – 44 p.
3. *Stoica I., Morris R., Liben-Nowell D. et al*, Chord: a scalable peer-to-peer lookup protocol for internet applications. // IEEE/ACM Transactions on Networking. – Feb. 2003. – IEEE Press, Piscataway: – Vol. 11, No 1. – pp. 17 – 32. – ISSN 1063-6692.
4. *Mahadevan P., Krioukov D., Fomenkov M. et al*, The internet AS-level topology: three data sources and one definitive metric // SIGCOMM Comput. Commun. Rev. – Vol. 36, No. 1. – ACM. New York. – 2006. – pp. 17-26.
5. *Shudo K., Tanaka Y., Sekiguchi S.*, Overlay Weaver: An overlay construction toolkit // Comput. Commun. – Vol. 31, No. 2. – ACM, New York. – 2008. – pp. 402-412.
6. *Nakao A., Peterson L., Bavier A.*, A routing underlay for overlay networks // SIGCOMM ‘03: Proc. of the 2003 conf. on Applications, technologies, architectures, and protocols for computer comm. – ACM, New York. – 2004. – pp. 11-18.
7. *Ahullo J.P., Lopez P.G.*, PlanetSim: an extensible framework for overlay network and services simulations // Simutools ‘08: Proc. of the 1st intern. conf. on Simulation tools and techniques for communications, networks and systems. – ICST, Brussels. – 2008. – ISBN 978-963-9799-20-2. – 8 p.
8. *Dabek F., Zhao B., Druschel P., Kubiatowicz J., Stoica I.*, Towards a common API for Structured Peer-to-Peer Overlays. // In Proc. of IPTPS’03 Workshop. – Berkeley, CA. – Feb. 2003. – 6 p.
9. *Aberer K., Cudr’e-Mauroux Ph., Datta A. et al*, P-Grid: a self-organizing structured P2P system // SIGMOD Rec. – ACM Press, New York – 2003. – Vol. 32, No. 3. – pp. 29-33.
10. *Naicken S., Livingston B., Basu A. et al,* The state of peer-to-peer simulators and simulations // SIGCOMM Comput. Commun. Rev. – ACM Press, New York. – Vol. 37, No 2. – April 2007. – pp. 95-98.
11. *Juenger, M., Mutzel, P. (Eds.), Batagelj, V., Mrvar, A.*, Pajek - analysis and visualization of large networks. // Graph Drawing Software. – Springer, Berlin. – pp. 77-103.