Моделирование распределенных вычислений для задачи синтеза фотореалистичных изображений на основе акторов и GRID технологий*

***В.В. Санжаров*[1], *В.А. Фролов*[2,3]**
vadim.sanzharov@gmail.com|vova@frolov.pp.ru
Москва, Россия, [1]Российский государственный университет нефти и газа (национальный исследовательский университет) имени И.М. Губкина;
Москва, Россия, [2]Институт прикладной математики им. М.В.Келдыша РАН;
Россия, [3]Московский государственный университет имени М.В.Ломоносова

В данной работе предлагается новый подход к распределенному рендерингу как на GPU, так и на CPU, использующий идеи GRID-вычислений. Предложена новая модель поведения вычислительных узлов на основе акторного похода. Проведено численное исследование свойств модели.

Ключевые слова: *GRID-вычисления, акторы, распределенный рендеринг, gpu*

1. Введение

Задача синтеза фотореалистичных изображений в приложении к таким областям, как архитектурная и интерьерная визуализация, кинопроизводство требует значительных вычислительных ресурсов. Зачастую, единственным способом получить результат за приемлемое время является распределение расчета освещения на несколько компьютеров. На сегодняшний день, самый распространенный подход – использование так называемых рендер-ферм – высокопроизводительных компьютеров, объединенных в локальную сеть или вычислительный кластер. Однако рендер-фермы мало пригодны для небольших компаний и индивидуальных художников в силу высокой стоимости и определенных неудобств, связанных с использованием внешних сервисов.

2. Обзор

2.1 Распределенный рендеринг

Одним из первых подходов к задаче распределенного рендеринга стали решения для суперкомпьютеров и кластеров [1,2]. В работе [2] было предложено решение на основе интерактивной параллельной трассировки лучей, позволившее достичь 15-20 кадров в секунду в динамических сценах за счет их эффективного разбиения на независимые объекты. В работе [3] предлагается централизованная система рендеринга, в которой все расчеты выполняются на стороне сервера, который, однако, может использовать вычислительные ресурсы нескольких компьютеров. Клиенты же получают доступ к возможностям просмотра и редактирования сцены (в некоторых пределах) через веб-интерфейс. Современные исследования в области распределенного рендеринга в основном направлены на использование GRID-вычислений, основной характеристикой которых является гетерогенность сети – компьютеры, выполняющие вычисления, могут значительно отличаться по производительности и иметь различное географическое расположение. Одно из решений на основе GRID-вычислений – BURP (Big Ugly Rendering Project) [4]. Этот проект представляет собой распределенную систему для рендеринга анимационных сцен. BURP основан на инфраструктуре BOINC (Berkeley Open Infrastructure for Network Computing) [5] и, тем самым, может быть отнесен к виду GRID-вычислений, называемому добровольными вычислениями. Желающий рассчитать анимационную сцену пользователь загружает на сервер проекта свою задачу, которая распределяется между подключенными к серверу пользователями, предоставляющими свои вычислительные ресурсы на добровольной основе. BURP поддерживает две рендер-системы Cycles и встроенный рендер ПО Blender. В работе [6] представлена рендер-система Yafrid, позволяющая распределить расчет изображения между гетерогенными компьютерами через Интернет. Одним из компонентов предлагаемой авторами системы является модуль MAgArRO, использующий мультиагентный подход и нечеткую логику для анализа вычислительной сложности изображения и проведения оптимизации разбиения задачи расчета изображения на подзадачи. В работе [7] представлена рендер-система, реализующая распределенный расчет изображений с помощью алгоритма кэша освещенности. В [8] авторы предлагают подход к рендерингу также с использованием алгоритма кэша освещенности на основе GRID-вычислений в Peer-to-peer (P2P) сетях. Основу метода составляет использование вычислительными узлами результатов друг друга для ускорения процедур интерполяции и заполнения структур данных, специфичных для кэша освещенности. Часть рассмотренных

Работа поддержана Российским фондом фундаментальных исследований (РФФИ) 16-31-60048. Работа опубликована по гранту РФФИ №16-07-20482.

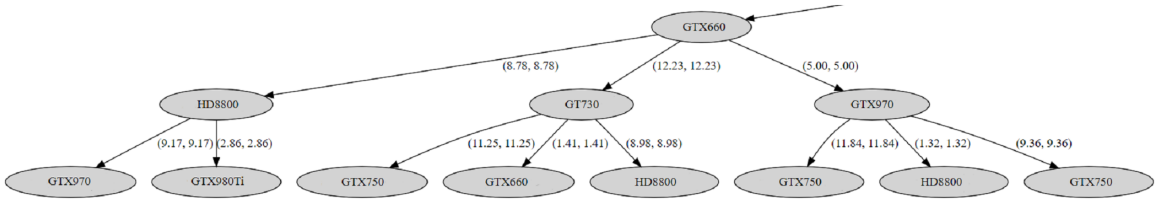

Рис. 1: Фрагмент одной из модельных сетей в виде дерева. В узлах машины с различными GPU.

решений выполняют распределение задачи между вычислительными узлами только по отдельным кадрам и тем самым подходят только для анимационных сцен. Кроме того, все рассмотренные выше решения используют особенности алгоритмов рендеринга на CPU и не могут быть напрямую применены к GPU рендер-системам (ключевое отличие для GPU - значительно возрастающий объем данных, которыми необходимо обмениваться вычислительным узлам). В распространенных GPU рендер-системах, таких как Vray RT [9], архитектура сетевого взаимодействия проста и мало масштабируема, в силу своей ориентированности на расчеты в локальной сети. Предлагаемая в данной работе модель распределенного рендеринга ориентирована на использование ресурсов как GPU, так и CPU, и рассчитана на распределение работы между вычислительными узлами в рамках расчета одного изображения в гетерогенной сети, состоящей из компьютеров с разной производительностью (GRID) (рис. 1).

2.2 Сетевые технологии

Одна из проблем, встающих в системах GRID-вычислений связана с сетевым взаимодействием вычислительных узлов. Это связано с тем, что в соответствии с парадигмой GRID-вычислений, компьютеры потенциально могут иметь любое географическое расположение. Что приводит к необходимости передачи данных в условиях глобальных сетей, которые значительно отличаются от таковых для локальных сетей. В задаче расчета освещения каждый вычислительный узел должен получить по сети всю информацию о 3d-сцене, включая геометрию и текстуры. В современных производственных задачах сцены могут содержать несколько миллионов треугольников в геометрии и несколько сотен текстур. Таким образом, размер сцены, с учетом архивирования, может достигать нескольких гигабайт. Одним из эффективных подходов к организации сетевого взаимодействия являются P2P технологии [10]. В [11] приводится классификация P2P технологий в контексте их применения к задачам GRID-вычислений. Одна из классификаций делит P2P по структуре на структурированные, неструктурированные и гибридные. В работах [12-14] было показано, что структурирование позволяет снизить трафик и, соответственно, нагрузку на канал в глобальных P2P сетях, а также решает проблему несбалансированного распределения ресурсов между узлами. На сегодняшний день ни одно из существующих решений в области распределенного рендеринга не использует возможностей P2P-технологий, несмотря на их перспективность в GRID-вычислениях.

3. Предлагаемое решение

3.1 Выбор средства моделирования

Предлагаемая в данной работе модель построена с использованием акторного подхода, в основе которого лежит асинхронный обмен сообщениями [15], и модели начальник-подчинённый (или заказчик-исполнитель). Акторный подход позволяет моделировать и анализировать поведение широкого круга систем распределённых параллельных вычислений и находит применение в построении таких систем в силу масштабируемости и устойчивости к отказам [16]. Мы использовали реализацию акторной модели в популярной платформе Akka [17] для создания программной реализации предлагаемого решения.

3.2 Описание модели

В предлагаемой модели вычислительные узлы объединяются в древовидную структуру (рис. 1). Корнем дерева является компьютер, инициировавший задачу рендеринга, другими словами «заказчик». В узлах располагаются компьютеры, предоставляющие свои вычислительные ресурсы для решения задачи - «исполнители». В качестве допущения принимается, что каждый из компьютеров знает своих ближайших (с точки зрения скорости передачи данных) соседей-исполнителей. Таким образом, компьютер-заказчик отсылает вычислительную задачу своим соседям-исполнителям и, те, в свою очередь, получив запрос, начинают выполнять расчеты. Исполнители далее также отсылают вычислительную задачу уже своим соседям-исполнителям, выступая для них в роли заказчика. Результатом вычислений в случае расчета освещения является изображение с некоторым числом сэмплов (Монте-Карло выборок) на пиксель.

После выполнения некоторого минимального объема вычислений, регламентируемого компьютером-

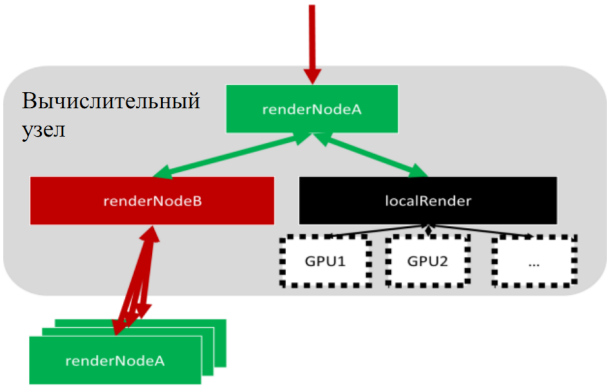

Рис. 2: Схема взаимодействия акторов в рамках одного вычислительного узла. Всё что внутри серого прямоугольника расположено на 1 машине.

заказчиком, и по запросу вышестоящего в иерархии компьютера, вычислительные узлы передают текущие результаты вычислений вверх по иерархии. Изображения, полученные от дочерних узлов и в результате локальных расчетов, объединяются в одно (при помощи усреднения сумм Монте-Карло выборок в акторе renderNodeA) в каждом узле графа и передаются на уровень выше. Таким образом, вышестоящая в иерархии машина (заказчик) «видит» каждого исполнителя как одну, но очень мощную машину, не имея информации о том, что на самом деле происходит ниже в иерархии. В реализованной модели учитывается скорость передачи данных между узлами сети, которая влияет на время, через которое каждый из компьютеров получит сцену и сможет начать расчет, а также на время передачи результатов. В соответствии с акторным подходом, в модели выделены три типа акторов, условно обозначенных как renderNodeA, renderNodeB, localRender. Эти три актора присутствуют на каждом вычислительном узле сети. Схема взаимодействия этих акторов представлена на рис. 2. Актор renderNodeA является родительским для двух остальных типов акторов в системе. Этот актор получает через дочерний актор renderNodeB результаты вычислений с более низких уровней дерева, а также от локальной рендер-системы, и объединяет их. Результат объединения renderNodeA отправляет на уровень вверх актору renderNodeB другого вычислительного узла. Актор renderNodeB, помимо передачи результатов расчетов, занимается установлением соединения с другими вычислительными узлами.

3.3 Анализ результатов моделирования

В качестве исходных данных мы взяли архитектурную сцену (рис. 3), вместе с текстурами, занимающую на диске порядка 500 мегабайт. Данная сцена была рассчитана с использованием алгоритма трассировки пути [18] в рендер-системе Hydra Renderer [19] на GPU от Nvidia модели GTX 670. По результатам времени расчета было получено, что на данном GPU для сцены на рис. 3. скорость расчета составляет приблизительно 0.5 сэмпла на пиксел в секунду.

Рис. 3: Сцена, характеристики которой были использованы для моделирования.

В первом численном эксперименте был зафиксирован GPU для каждого компьютера сети (GTX 670) и канал связи между соседними компьютерами (download 10 мегабайт/сек. и upload 5 мегабайт/сек). Рассмотрены три топологии - бинарное и тернарные деревья (с глубиной 3), и плоское дерево (все узлы подключены к корню). Один узел – один компьютер с одним GPU, всего доступных компьютеров - 15. Моделирование проводилось до достижения числа сэмплов равному 1024 для трех сетей. Сначала рассмотрим случай, когда архив со сценой необходимо передавать по сети между узлами (рис. 4, обозначены пунктирными линиями). Видно, что меньшее число подключений на один узел приводит к лучшему результату. И в случае подключения всех вычислительных узлов напрямую к корневому компьютеру (обозначен синим на графике), к моменту завершения передачи сцены, бинарная конфигурация (красный пунктирный график) практически успевает посчитать такую сцену дважды.

Далее рассмотрим случай, когда вычислительные узлы скачивают сцену с некоторого сервера или облачного хранилища, т.е. нагрузка на каналы между узлами связана исключительно с передачей результатов (показаны на рис. 3 точечными графиками). В этом случае время, требуемое на расчет, падает более чем в два раза. Меньшее количество подключений на узел все также дает наилучший результат.

Если увеличить временной интервал, через который компьютер-заказчик запрашивает из сети промежуточные результаты, то расчет также ускоряется (рис. 4, розовый график). Таким образом, в случае, когда важен только финальный результат,

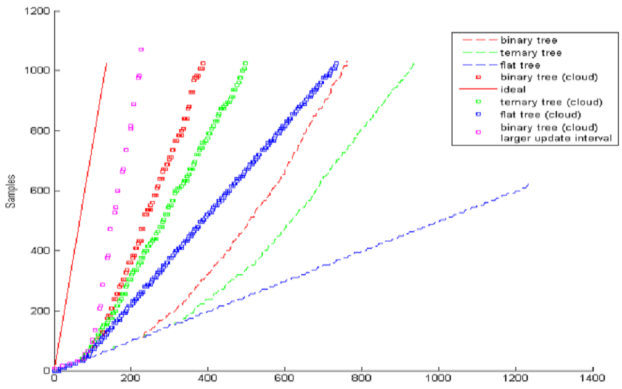

Рис. 4: Рост числа сэмплов для 3-х топологий с одинаковыми GPU и сетевыми каналами.

интервал получения обновлений может быть увеличен для ускорения расчетов.

Рассмотрим результаты моделирования сети с топологией бинарного дерева, с фиксированными каналами связи с высокой пропускной способностью, одинаковой вычислительной мощностью узлов и довольно большим интервалом передачи обновлений. Моделирование проводилось до достижения 32768 сэмплов для разного числа компьютеров – 31, 63 и 127 (глубина дерева 4, 5 и 6 соответственно). В этом случае видно (рис. 5), что для большого числа сэмплов, типичного для финального рендера, и с достаточно большим интервалом отправления результатов, скорость расчета освещения сходится к идеальной (как если бы вся производительность сети была бы доступна на одном компьютере). Есть лишь отставание, связанное с необходимостью загрузить сцену. Однако такая конфигурация сети сама по себе близка к идеальной из-за одинаково высоких скоростей каналов связи у всех узлов (12 мегабайт/сек.), которые без труда обеспечивают беспроблемную доставку результатов раз в 20 секунд (рис. 5).

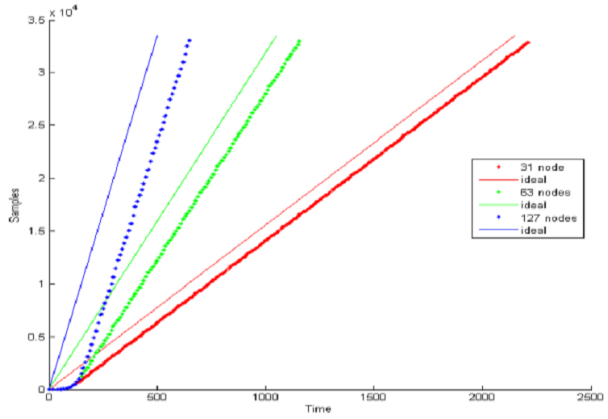

Рис. 5: Рост числа сэмплов для топологии бинарного дерева разной глубины.

Наконец рассмотрим сценарий моделирования со случайными конфигурациями сети. Каждый компьютер сети оборудован GPU, выбранным случайным образом исходя из частоты встречаемости данного GPU в популярном сервисе цифрового распространения компьютерных игр и программ Steam (были взяты 19 самых популярных GPU, а также GTX 670) [20].

Производительность каждого GPU была оценена исходя из известной производительности GTX 670 и баллов, набранных каждым из GPU в системе тестирования 3d Mark [21]. Каналы связи также заданы случайным образом в диапазоне от 0.5 до 12.5 мегабайт/сек. Моделирование проводилось до достижения 16384 сэмплов, максимальная глубина дерева - 3.

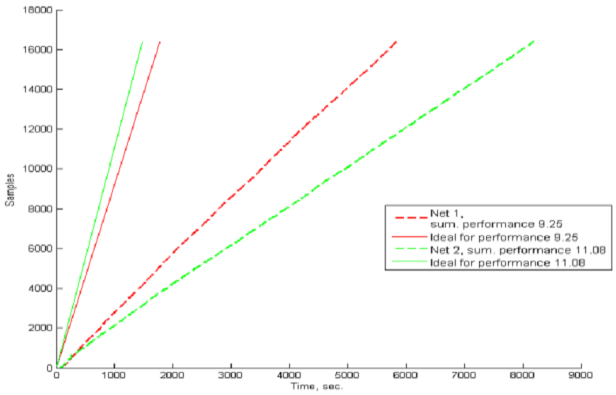

Рис. 6: Сеть со случайными характеристиками.

В итоге, значительную роль играет скорость каналов связи. Сеть с большей суммарной производительностью - 11.08 сэмплов/сек. (рис. 6 зеленый пунктирный график) справляется с задачей медленнее на 40 минут, чем сеть с меньшей производительностью (9.25 сэмплов/сек., красный пунктирный график) именно за счет худших каналов в сети. Для первой сети результат хуже идеального в 5.5 раз, а для второй в 3.3 раза. Суммарная производительность второй сети превосходит производительность корневого компьютера в 18.5 раз, а скорость расчета выше в 5.6 раз. Для первой сети – суммарная производительность больше в 22 раза, скорость в 4 раза.

Заключение

Разработанная модель позволила оценить перспективность распределенных GRID-технологий для задачи расчета освещения с учетом особенностей решения этой задачи на GPU. При обеспечении хорошей связи между вычислительными узлами возможно добиться практически идеальной скорости расчета по сравнению с несетевым решением. Если же связь относительно-медленная, скорость расчета может быть увеличена за счет большего вре-

менного интервала между получением результатов расчетов из сети. Скорости каналов связи 12.5 мегабайт/сек (т.е. 100Mb/s) хватает для того, чтобы этот интервал (10-20 сек, рис. 3-5) был достаточен для интерактивного отслеживания рендеринга. Результаты исследования модели не показали наличия предела масштабируемости для такой организации сетевого взаимодействия вычислительных узлов по сети.

Литература

[1] DeMarle D.E., Parker S., Hartner M., Gribble C., Hansen C.: Distributed interactive ray tracing for large volume visualizatiuon. IEEE Symposium on Parallel and Large-Data Visualization and Graphics (2003), IEEE Computer Society, p. 12.

[2] Parker S., Martin W., Sloan P.-P. J., Shirley P., Smits B., Hansen C.: Interactive ray tracing. I3D '99 Proceedings of the 1999 symposium on Interactive 3D graphics (1999), pp. 119-126

[3] Барладян Б.Х., Волобой А.Г., Вьюкова Н.И., Галактионов В.А., Дерябин Н.Б. Моделирование освещенности и синтез фотореалистичных изображений с использованием Интернет технологий // "Программирование №5, 2005, с.66-80.

[4] Открытая распределенная система рендеринга BURP на основе платформы BOINC. 2016 https://burp.renderfarming.net/

[5] Платформа GRID-вычислений BOINC. 2016 https://boinc.berkeley.edu/

[6] Gonzales-Morcillo C., Wei. G.,Vallejo-Fernandez D., Liminez-Linares L., Albusac-Jimenez J. 3D distributed rendering and optimization using free software, European Journal of the Informatics Professional, 6, 8(2008), pp. 45-53

[7] Aggarwal V., Debattista K., Bashford-Rogers T., Dubla P., Chalmers A.: High-fidelity interactive rendering on desktop grids. IEEE Computer Graphics and Applications 32, 3 (2012), pp. 24–36

[8] Bugeja K., Debattista K., Spina S., Chalmers A. Collaborative high-fidelity rendering over peer-to-peer networks, PGV '14 Proceedings of the 14th Eurographics Symposium on Parallel Graphics and Visualization, pp. 9-16

[9] Chaos group, производитель рендер-систем Vray и Vray RT. 2016 http://chaosgroup.com

[10] Foster, I., Iamnitchi, A.: On death, taxes, and the convergence of peer-to-peer and grid computing. In: IPTPS '03: Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (2003)

[11] Zhao H., Liu X., Li X. A taxonomy of peer-to-peer desktop grid paradigms, Cluster Computing, 14, 2(2011), pp.129-144

[12] Cheema, A.S., Muhammad, M., Gupta, I.: Peer-to-peer discovery of computational resources for grid applications. In: GRID'05: Proceedings of the 6th IEEE/ACM International Workshop on Grid Computing, pp. 179–185 (2005)

[13] Darlagiannis, V., Liebau, N., Heckmann, O., Mauthe, A., Steinmetz,R.: Caching indices for efficient lookup in structured overlay networks. Agents and P2P Computing, pp. 81–93 (2006)

[14] Gupta, R., Sekhri, V., Somani, A.K.: CompuP2P: An architecture for internet computing using peer-to-peer networks. IEEE Trans. Parallel Distrib. Syst. 17(11), 1306–1320 (2006)

[15] Hewitt C., Bishop P., Steiger R. A Universal Modular Actor Formalism for Artificial Intelligence. IJCAI'73 Proceedings of the 3rd international conference on Artificial intelligence, pp.235-245

[16] Hewitt C. ORGs for Scalable, Robust, Privacy-Friendly Client Cloud Computing, IEEE Internet Computing, 12, 5(2008), pp. 96-99

[17] Платформа распр. вычислений Akka. 2016 http://akka.io/

[18] Kajiya, J. T.: The rendering equation. Computer Graphics 20(4): 143–150. Proceedings of SIGGRAPH '86

[19] Рендер-система Hydra Renderer. 2016 http://raytracing.ru/

[20] Статистика по комплектующим компьютеров пользователей платформы Steam. 2016 http://store.steampowered.com/hwsurvey

[21] Система тестирования 3dMark. 2016 http://3dmark.com/