Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра математической
кибернетики и компьютерных наук

РЕАЛИЗАЦИЯ И ТЕСТИРОВАНИЕ КВАНТОВОГО АЛГОРИТМА ШОРА С ИСПОЛЬЗОВАНИЕМ DOCKER КОНТЕЙНЕРА

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Холкина Александра Витальевича

Научный руководитель
доцент, к. ф.-м. н. _____ В. М. Соловьев

Заведующий кафедрой
к. ф.-м. н. _____ С. В. Миронов

Саратов 2016

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Темой работы является реализация и тестирование квантового алгоритма Шора с использованием Docker контейнера.Квантовые технологии является очень перспективным направлением на сегодняшний день. Квантовый мир со своей возможностью «вмещать в себя» столь грандиозные конструкции на сегодняшний день не имеет конкурентов. Квантовые вычисления имеют корни в узкоспециализированных областях теоретической физики, но их будущее без сомнений лежит в огромном эффекте, которые они окажут на жизнь всего человечества. Но использование квантовых технологий в полной мере требует квантового компьютера, на создание которого брошены большие силы. Но пока он еще не изобретен, то остается возможность знакомиться с квантовыми технологиями на классических компьютерах. Так как квантовые алгоритмы при таком использовании являются очень ресурсоемкими, то для ознакомления лучше использовать изолированную среду в силу того, что при использовании на классических компьютерах поведение и возможности алгоритмов довольно сложно предугадать.

Предметом исследования настоящей дипломной работе является квантовый алгоритм Шора, как пример квантовой технологии. Также в работе используется Docker архитектура, позволяющая повысить надежность при исследовании алгоритма Шора. В ходе работы описывается каким образом использование контейнерной технологии помогает исследовать квантового алгоритма Шора.

Цель дипломной работы — реализовать и протестировать квантовый алгоритм Шора с использованием Docker контейнера.

В данной дипломной работе ставятся следующие задачи:

1. Реализовать квантовый алгоритм Шора;
2. Создать образ Docker контейнера с реализованным алгоритмом;
3. Развернуть Docker контейнер на основе созданного образа и провести тестирования алгоритма;
4. На основе тестирования реализовать скрипт, позволяющий безопасно использовать алгоритм Шора в Docker контейнере.

Дипломная работа состоит из введения, трех разделов, заключения, списка литературы и приложения. Во введении обосновывается актуальность темы квалифицированный работы, связанной с квантовыми вычислениями и их

моделированием на классических компьютерах. В Разделе «Квантовые технологии» рассматриваются основы квантовых технологий вычисления и реализация квантовых алгоритмов. Описаны математические основы факторизации на примере алгоритма Шора. В разделе «Виртуализация» рассматриваются основные концепции виртуализации вычислений. Описывается архитектура Docker контейнера и особенности его использования для организации вычислений. В разделе «Реализация и тестирование квантового алгоритма Шора с использованием Docker контейнера» рассматриваются вопросы реализации квантового алгоритма Шора. Приводится процесс моделирования на классическом компьютере квантового алгоритма Шора. В заключении делается вывод о перспективах использования квантовых вычислений. В Приложении приведен листинг разработанных модулей для создания собственного Docker контейнера и реализация квантового алгоритма Шора.

1 Основное содержание работы

Раздел 1. Квантовые технологии

В данном разделе описываются основы квантовых технологий и компьютеров. Так же рассматривается квантовый алгоритм Шора и приводится его математическое описание. Использование квантовой теории является одним из самых главных направлений развития в науке в современном мире. Разработка квантового компьютера — одно из самых активных направлений в квантовой механике. Это принципиально новый тип вычислений, позволяющий добиться невероятных мощностей.

Квантовая технология в целом — это технология, которая основана на манипуляции сложными квантовыми системами на уровне их индивидуальных компонентов, а не просто технология, основанная на квантовой физике. Цель данной технологии состоит в том, чтобы создать системы и устройства, основанные на квантовых принципах, к которым обычно относят следующие:

1. Дискретность (квантованность) уровней энергии (квантово-размерный эффект, квантовый эффект Холла);
2. Принцип неопределённости Гейзенберга;
3. Квантовая суперпозиция чистых состояний систем;
4. Квантовое туннелирование через потенциальные барьеры;
5. Квантовую сцепленность состояний.

Квантовые вычисления — это вычислительная модель, которая отличается от модели Тьюринга и фон Неймана, и предполагается, что для некоторых задач она является более эффективной. По крайней мере найдены задачи, для которых модель квантовых вычислений даёт полиномиальную сложность, в то время как для классической вычислительной модели неизвестно алгоритмов, которые имели бы сложность, ниже экспоненциальной. Квантовая вычислительная модель основана на нескольких довольно простых правилах преобразования входной информации, которые обеспечивают массовую параллелизацию вычислительных процессов. Другими словами, можно одновременно вычислить значение функции для всех её аргументов (и это будет единственный вызов функции). Это достигается специальной подготовкой входных параметров и специальным же видом функции. Модель квантовых вычислений есть математическая абстракция, которая отражает объективный процесс. В частности, числа в векторах и матрицах являются комплексными, хотя это со-

вершенно не увеличивает вычислительную мощность модели (она бы была такой же мощной и с действительными числами), однако выбраны именно комплексные числа потому, что найден объективный физический процесс, который осуществляет такие преобразования, как описывает модель, и в котором используются именно комплексные числа. Этот процесс называется унитарной эволюцией квантовой системы.

В основе квантовой вычислительной модели лежит понятие кубита. Это практически то же самое, что и бит в классической теории информации, однако кубит может одновременно принимать несколько значений. Говорят, что кубит находится в суперпозиции своих состояний, то есть значение кубита есть линейная комбинация его базовых состояний, и коэффициенты при базовых состояниях как раз являются комплексными числами. Базовыми же состояниями являются известные по классической теории информации значения 0 и 1 (в квантовых вычислениях их принято обозначать $|0>$ и $|1>$).

Суперпозиция одного кубита записывается как $A|0> + B|1>$, где A иB — некоторые комплексные числа, единственное ограничение на которые заключается в том, что сумма квадратов их модулей всегда должна равняться 1. Рассмотреть два кубита. Два бита могут получать 4 возможных значения: $00, 01, 1011$. Значит два кубита представляют собой суперпозицию четырёх базовых значений: $A|00> + B|01> + C|10> + D|11>$. Три кубита представляют собой суперпозицию восьми базовых значений. Другими словами, квантовый регистр из N кубитов одновременно хранит в себе $2N$ комплексных чисел. Ну а с математической точки зрения это есть $2N$мерный вектор в комплекснозначном пространстве. Именно этим достигается экспоненциальная мощность модели квантовых вычислений.

Квантовый компьютер — вычислительное устройство, работающее на основе квантовой механики. Квантовый компьютер принципиально отличается от классических компьютеров, работающих на основе классической механики. Полноценный квантовый компьютер является пока гипотетическим устройством, сама возможность построения которого связана с серьёзным развитием квантовой теории в области многих частиц и сложных экспериментов. Эта работа лежит на переднем крае современной физики. Однако, ограниченные (до 512 кубитов) квантовые компьютеры уже существуют.

Упрощённая схема вычисления на квантовом компьютере выглядит так:

берется система кубитов, на которой записывается начальное состояние. Затем состояние системы или её подсистем изменяется посредством базовых квантовых операций. В конце измеряется значение, что и является результатом работы компьютера.

Выделяют несколько основных требований к универсальному квантовому компьютеру:

1. Квантовый компьютер должен состоять из большого числа элементарных кубитов. Таковыми могут быть не только элементарные спины электронов, но и сложные системы, построенные из множества частиц; главное требование при этом — эволюция таких систем должна описываться аналогичными квантовыми состояниями;
2. Операции с кубитами должны быть такими, чтобы обеспечивать возможность «запутывать» состояния каждой пары кубитов и производить изменение квантового состояния каждого отдельного кубита. Должна быть также физическая возможность менять параметры этого воздействия на кубиты во времени по заданному закону;
3. Каждый кубит должен быть с высокой надежностью изолирован от внешнего мира, т.е. время декогерентности должно быть много больше времени выполнения самих вычислений;
4. Необходимо уметь производить в конце процесса вычислений сами измерения, т.е. иметь возможность выяснить, в каком состоянии оказался каждый из кубитов системы к концу действия алгоритма.

Отметим, что все описанные условия достаточно плохо согласуются друг с другом в реальном физическом мире. Так, изоляция кубитов от внешней среды в процессе вычислений противоречит возможности точно управлять значениями параметров воздействия на эти кубиты, а также возможности эффективно измерять конечные состояния. Именно поэтому задача реализации квантового компьютера является исключительно сложной.

Алгоритм Шора — квантовый алгоритм факторизации (разложения числа на простые множители), позволяющий разложить число M за время $O(\lg^3 M)$, используя $O(\lg M)$ логических кубитов.

Алгоритм Шора был разработан Питером Шором в 1994 году. Семь лет спустя, в 2001 году, его работоспособность была продемонстрирована группой специалистов IBM. Число 15 было разложено на множители 3 и 5 при помощи

квантового компьютера с 7 кубитами.

Алгоритм Шора основан на возможности быстро вычислить собственные значения унитарного оператора с высокой точностью, если можно эффективно вычислять любые его степени. Взяв в качестве такого оператора умножение на x по модулю N (этот оператор действует в $2n$ мерном пространстве, где , преобразуя базисный вектор, соответствующий числу a, в базисный вектор, соответствующий числу $x * a(modN)$), мы сможем вычислить такое n, что $x^n = 1(modN)$, что позволяет (с высокой вероятностью) разложить N на множители на обычном компьютере.

Раздел 2. Виртуализация

Данный раздел состоит из двух частей. В первой части рассматриваются концепции виртуализации вычислений, описывается архитектура Docker контейнера. Виртуализация оптимизирует использование ИТ-ресурсов, рассматривая физические ресурсы компании в качестве резервуаров, из которых можно динамически черпать виртуальные ресурсы.

В современном мире виртуализации доминируют технологии гипервизора виртуальных машин, например, VMware, KVM и Xen, хотя есть и альтернативные подходы. К числу последних относится и набирающая популярность технология контейнеров компании Docker на базе открытого исходного кода. Модель виртуализации на базе гипервизора, используемая VMware, Xen и KVM, предполагает запуск виртуальной машины, включающей целую ОС. Docker же придерживается иного подхода, в котором приложение помещается в виртуальный контейнер, функционирующий поверх единственной операционной системы базового уровня, что значительно экономит ресурсы.

Docker — это открытая платформа для разработки, доставки и эксплуатации приложений. Docker используется для автоматизации развертывания и управления приложениями в среде виртуализации на уровне операционной системы. С помощью Docker можно отделить приложение от инфраструктуры и обращаться с инфраструктурой как с управляемым приложением. Docker помогает выкладывать программный код быстрее, быстрее тестировать, быстрее выкладывать приложения и уменьшить время между написанием кода и запуска кода. Docker делает это с помощью легковесной платформы контейнерной виртуализации, используя процессы и утилиты, которые помогают управлять и выкладывать приложения.

В своем ядре Docker позволяет запускать практически любое приложение, которое будет безопасно изолированно в контейнере. Безопасная изоляция позволяет запускать на одном хосте много контейнеров одновременно. Легковесная природа контейнера, который запускается без дополнительной нагрузки гипервизора, позволяет добиваться больше от имеющегося железа.

Docker использует архитектуру клиент-сервер. Docker клиент общается с демоном Docker, который берет на себя тяжесть создания, запуска и распределения контейнеров. И клиент, и сервер могут работать на одной системе, а можно подключить клиент к удаленному демону docker-a. Клиент и сервер общаются через сокет или через RESTful API. Docker-демон, как показано на диаграмме, запускается на хост-машине.

Пользователь не взаимодействует с сервером на прямую, а использует для этого Docker-клиент. программа Docker — главный интерфейс к Docker. Она получает команды от пользователя и взаимодействует с Dockerдемоном.

Docker написан на Go и использует некоторые возможности ядра Linux, чтобы реализовать приведенный выше функционал. Docker использует технологию namespaces для организации изолированных рабочих пространств, которые называется контейнерами. Когда запускается контейнер, Docker создает набор пространств имен для данного контейнера. Это создает изолированный уровень, каждый аспект контейнера запущен в своем простанстве имен, и не имеет доступ к внешней системе. Docker также использует технологию cgroups или контрольные группы. Ключ к работе приложения в изоляции, предоставление приложению только тех ресурсов, которые предоставлены. Это гарантирует, что контейнеры будут хорошими соседями. Контрольные группы позволяют разделять доступные ресурсы железа и если необходимо, устанавливать пределы и ограничения.

Во второй части раздела описывается варианты разворачивая Docker контейнера. Эти варианты можно разделить на две группы. В первом случае пользователю необходимо иметь технические навыки, так как при создании контейнера будут использоваться специальные команды в терминале. Для разворачивания Docker контейнера необходимо создать образ, на основе которого и будет строиться контейнер. Можно скачать готовый образ из Docker реестра, что позволит сэкономить время. Но в таком случае придется иметь дело с настроенной конфигураций. Если необходимо что-то изменить, то придется

это делать кажый раз развернув контейнер. Можно создать собственной образ. Данный вариант позволит настроить контейнер так, как необходимо, иметь все необходимые пакеты и настройки.

Во втором случае можно использовать проект Openstack Murano. Данный проект позволяет разворачивать Docker контейнер без обладания какими-либо техническими навыками. С помощью дружеского интерфейса пользователь без труда сможет развернуть контейнер.

Раздел 3. Реализация и тестирование квантового алгоритма Шора с использованием Docker контейнера

В данном разделе описывается моделирования работы квантового алгоритма Шора на классическом компьютере с использованием Docker контейнера. Использование квантового алгоритма Шора на классическом компьютере происходит возможно в том случае имитации работы квантового компьютера. Но это приводит к экспоненциальному росту используемых ресурсов. Тогда запуске алгоритма Шора для факторизации числа 200 будет неудачный из-за нехватки ресурсов.

Проведем тестирование алгоритма Шора, с целью узнать до какого числа разумно запускать алгоритм Шора без появления ошибки об нехватки ресурсов. Но запуск в обычном виде алгоритма не является безопасным, так как из-за ошибки об нехватке ресурсов операционная система вынуждена аварийно убивать запущенный процесс. При этом работа компьютера становится медленной, а с точки зрения работы алгоритма пользователь получает информацию только о том, что произошла ошибка нехватки памяти.

Тогда для проведения тестирования воспользуемся Docker контейнером, чтобы изолировать работу алгоритма Шора. Для начала создадим Docker образ, в котором будем иметься тестируемый алгоритм и все необходимое для его запуска. Так как процесс тестирования подразумевает, что запуск будет происходит неоднократно, то реализуем скрипт, который поможет при тестировании. Скрипт должен уметь разворачивать Docker контейнер, запускать в нем квантовый алгоритм Шора и после завершения его работы уничтожать контейнер.

При запуске алгоритма в контейнере будем перенаправлять стандартный поток вывода, чтобы иметь результат работы алгоритма в отдельном файле, и поток вывода ошибок, что позволит отслеживать ошибки работы алгоритма.

Для уничтожения контейнера необходимо рассматривать успешный и неудачный запуски алгоритма. Для отлавливания неудачного случая будет использовать поток вывода ошибок. Если данный поток содержит некоторую информацию, то можно прерывать работу алгоритма и уничтожать контейнер. Чтобы отслеживать удачную работу алгоритма будем мониторить загруженность контейнера. Если CPU контейнера 0.00%, то следовательно контейнер завершил работу алгоритма и может быть уничтожен.

В результате тестирования было выявлено, что для данного классического компьютера запуск алгоритма Шора для факторизации числа 53 срабатывает успешно редко. Так как данный алгоритм имеет вероятностный характер, то запуск для одного значения не всегда будет успешен или неудачен. Тогда в нашем случае, можно сказать, что установив границу до 50 будет довольно высокая вероятность того, что алгоритм будет работать без вылета с ошибкой о ресурсах.

Тогда можно модифицировать написанный ранее скрипт, добавив в него проверку значения для факторизации с учетом найденной границы. В итоге получим, что на любом классическом компьютере, который поддерживает использование Docker технологии, можно загрузить созданный ранее Docker образ и запустить модифицированный скрипт. В результате работы скрипта будет развернут Docker контейнер, в нем безопасно отработает квантовый алгоритм Шора, после чего контейнер будет уничтожен. Пользователь обращаясь к файлам, в которые были перенаправлены описанные ранее потоки, может получать информацию о работе алгоритма.

ЗАКЛЮЧЕНИЕ

Основной целью работы являлось моделирование работы квантового алгоритма Шора на классическом компьютере с использованием Docker контейнера. В данной работе были выполнены следующий задачи:

1. рассмотрены квантовые технологии;
2. изучены принципы работы квантовых технологий;
3. изучена архитектура Docker контейнера;
4. реализован и протестирован квантовый алгоритма Шора с использованием Docker контейнера;
5. на основе тестирования был реализован скрипт, позволяющий безопасно запускать алгоритм Шора в Docker контейнере.

Данная работа может быть использована для изучений квантовых технологий. Результат данный работы может использоваться в качестве примера изучения работы квантовых алгоритмов на классических компьютерах.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Understanding docker [Электронный ресурс]. — 2016. — URL: https://www.packtpub.com/books/content/understanding-docker (Дата Обращения 08.06.2016).

2 *Docker, T.* Docker Documentation [Электронный ресурс] / T. Docker. — 2015. — URL: https://media.readthedocs.org/pdf/dhrpdocs/stable/dhrpdocs.pdf (Дата Обращения 08.06.2016).

3 Понимая docker [Электронный ресурс]. — 2015. — URL: https://habrahabr.ru/post/253877/ (Дата Обращения 08.06.2016).

4 Quantum algorithms) [Электронный ресурс]. — URL: http://www.cs.berkeley.edu/~vazirani/algorithms/chap10.pdf (Дата обращения 18.06.2016).

5 An Introduction to Quantum Algorithms) [Электронный ресурс]. — URL: http://people.cs.umass.edu/~strubell/doc/quantum_tutorial.pdf (Дата обращения 18.06.2016).

6 Квантовые алгоритмы и их влияние на безопасность современных классических криптографических систем) [Электронный ресурс]. — URL: http://crypto.rsuh.ru/papers/bogdanov-kizhvatov-quantum.pdf (Дата обращения 18.06.2016).

7 Квантовые алгоритмы: возможности и ограничения) [Электронный ресурс]. — URL: http://logic.pdmi.ras.ru/csclub/sites/default/files/20110403_quantum_algorithms_vyali_lecture_notes.pdf (Дата обращения 18.06.2016).

8 Quantum Computation and Quantum Information) [Электронный ресурс]. — URL: http://www.johnboccio.com/research/quantum/notes/QC10th.pdf (Дата обращения 18.06.2016).

9 D-Wave confirmed as the first real quantum computer by new research) [Электронный ресурс]. — URL: http://www.extremetech.com/computing/184242-d-wave-confirmed-as-the-first-real-quantum/computer-by-new-research (Дата обращения 18.06.2016).

10 quatrum technology: the second quantrum revolution) [Электронный ресурс]. — URL: http://arxiv.org/ftp/quant-ph/papers/0206/0206091.pdf (Дата обращения 18.06.2016).

11 Quipper: A Scalable Quantum Programming Language) [Электронный ресурс]. — URL: http://arxiv.org/pdf/1304.3390.pdf (Дата обращения 18.06.2016).

12 Murano Documentation) [Электронный ресурс]. — URL: https://wiki.openstack.org/wiki/Murano (Дата обращения 18.06.2016).