УДК 004.001

В.Ф. Гузик, С. М. Гушанский, И.А. Судаков

ПОСТРОЕНИЕ МОДЕЛИ КВАНТОВОГО ВЫЧИСЛИТЕЛЯ

В настоящее время квантовые вычислители существуют в виде экспериментальных установок и прототипов [1], т.е. не существует устройств, способных производить квантовые вычисления на уровне, когда производительность данных устройств, превзойдет современные компьютеры. Для исследователей это означает, что для изучения особенностей квантовых алгоритмов необходимо проводить моделирование. В статье приводится анализ характеристик существующих моделей квантового вычислителя и сравнение некоторых особенностей с целью выявления и постановки требований для построения собственной модели. Для выявления лучших моделей представлена попытка построения метода численной оценки моделей по реализованному функционалу. Метод оценки позволяет ранжировать модели, выделить лучшие и составить требования к разрабатываемой модели.

Квантовый компьютер; модели квантового компьютера; сравнение моделей.

V.F. Guzik, S.M. Gushansky, I.A. Sudakov

BUILDING A MODEL OF QUANTUM COMPUTER

In our days quantum computers are presented in the form of experimental equipment and prototypes [1] and there is no quantum computing devices, which will surpass the performance of modern computers. That means it's necessary to perform simulation to study the peculiarities of quantum algorithms. In this article the characteristics of existing models are analyzed and the purpose of this article is to give the requirements for building own model. Also this work is presented to construct a methodic of the numerical evaluation for searching models with best functionality. Method of evaluation allows to rank models, distinguish the best one and make requirements for developed model.

Quantum Computer; quantum computer mode; a comparison of models.

Введение. Под моделированием обычно понимают процесс исследования (познания) объектов на их моделях. Однако не существует единой классификации видов моделирования, например, можно выделить следующие виды моделирования: информационное, компьютерное, математическое и др. виды моделирования [2]. Компьютерная модель или численная модель – программа, реализующая абстрактную модель некоторой сложной для аналитики системы. Программные модели проще и удобнее исследовать в силу простоты проведения экспериментов и широких возможностях настройки [3]. В данной статье под моделированием квантовых вычислений будем понимать математическое (программное) моделирование системы кубит и преобразований над ними.

К настоящему времени разработано большое количество различных программных моделей (как консольных, так и графических), библиотек API, а также моделей отдельных алгоритмов, большинство распространяется свободно и доступно для широкого круга пользователей. Предоставляемые возможности моделирования и подходы к управлению настолько различны, что оценить их качество и выбрать подходящую модель – сложная задача. Для анализа были отобраны различные модели, но наиболее интересными являются модели с графическим интерфейсом.

Анализ и сравнение существующих моделей. Рассматривая и анализируя возможности существующих моделей, можно произвести сравнение и суммарную оценку моделей по всем характеристикам. Сравнение производится по категориям характеристик, наиболее важные из которых затрагиваются в пунктах сравнения «Моделирование» и «Архитектурные и программные характеристики».

I. Тип модели.

Можно выделить три категории моделей и вообще приложений для моделирования квантовых вычислений:

1. Самостоятельное приложение. Модели этой группы – самодостаточные и полнофункциональные, позволяют строить и моделировать квантовые алгоритмы. Не зависят от другого ПО и построены с оптимальным процессом моделирования, но зачастую ориентированы только для тех целей, которые преследовал разработчик, небольшой уровень гибкости при моделировании и возможностей построения алгоритмов [4, 5, 6, 7]. Сюда же можно отнести модели отдельных квантовых алгоритмов [8, 9].

2. Надстройка над математическими средами. Представителем данной группы могут быть toolkit или библиотека для MatLab, Maple, MathCAD и Mathematica, которые дополняют возможности основной среды для легкого построения моделей квантовых алгоритмов [10, 11]. Встречаются готовые модели алгоритмов [12]. Имея навыки моделирования в данных средствах, можно легко изменить модель, но моделирование неоптимизированное, возможностей настройки нет.

3. Библиотеки API. Такие библиотеки предназначены скорее для облегчения разработки моделей алгоритмов и самодостаточных моделей квантового вычислителя, чем для моделирования**.** При использовании нет необходимости реализовывать весь математический аппарат, но и моделью такие библиотеки назвать нельзя [8, 13, 14, 15].

II. Архитектурные и программные характеристики.

Данный подход к классификации и оценке подразумевает сравнение моделей на основе особенностей структуры приложения, взаимосвязей между различными компонентами среды, а также возможностей, заложенных в модели разработчиками.

1. Модульность. Данный критерий сравнения под модульностью подразумевает разделение среды моделирования на отдельные файлы-компоненты, которые содержат в себе различные части модели. Это могут быть: исполняемые файлы, библиотеки, файлы баз данных и т.д. Можно выделить многомодульные и одномодульные модели. Модели, состоящие из одного файла (модуля), не имеют никаких зависимостей, но и не предоставляют возможностей изменений в архитектуре модели или её функциональности без перекомпиляции всего проекта [4, 5, 16]. Многомодульные модели разделены на модули (компоненты ввода-вывода данных, математическое ядро, наборы гейтов) при наличии API или открытого исходного кода можно легко модифицировать или дополнить. Вынесение математического ядра [6, 17, 18] или набора гейтов [6, 19] может дать возможность модификации-дополнения исходного, использования другого или же включения данного компонента в другую модель. Теоретически возможна реализация модели с системой плагинов, расширяющих функционал модели.

2. Многопоточность. Критерий сравнения «Многопоточность» основывается на том факте, что процесс моделирования квантовых вычислений легко распараллеливается, что теоретически дает заметное ускорение при моделировании на многопроцессорных системах [20, 21].

При однопоточном моделировании [18, 6, 19, 4] нагружается только на одно ядро или процессор, что нерентабельно на современных компьютерах. Многопоточное моделирование может быть реализовано на ядрах одного или нескольких процессоров, для таких приложений существует API Open MP [9]. Современные графические платы позволяют производить параллельные вычисления на своих ядрах, выдавая большой прирост производительности [20] за счет большего количества параллельных вычислительных ядер. Кластеры или GRID-системы могут максимальный прирост производительности при больших объемах вычислений, для создания таких моделей можно использовать MPI .

3. Экономия памяти. Для реализации преобразований в квантовых вычислениях используются матрицы, размер которых экспоненциально зависит от количества моделируемых кубит, объем памяти – одна из главных проблем моделирования квантовых вычислений.

Неоптимизированные матрицы дают возможность задать и хранить любую матрицу, как в SimQubit, и использовать её во время вычислений [19, 17]. Но полноразмерные матрицы операций можно генерировать во время выполнения или применять серию небольших преобразований, суммарное воздействие будет аналогичным [13, 5]. Разработчики Libquantum [8] при вычислениях используют значения только тех амплитуд состояний, которые не равны нулю. В QuIDDPro [16] вычисления организованы на основе графов и переходов, определяются наиболее вероятные состояния, а остальные отбрасываются, что экономит еще больше памяти.

4. Кроссплатформенность. Возможность использовать модель на различном оборудовании и под различными платформами дает преимущество, но многие модели разрабатываются только под платформу UNIX\Linux или Windows, что не очень удобно [16, 6, 18, 19]. Модели, разработанные для нескольких платформ [5], частично решают эту проблему, но кроссплатформенная модель наиболее предпочтительна. Платформа Net может стать кроссплатформенной, а Java уже является таковой [4, 17, 13, 24].

III. Интерфейс, информативность модели.

Пользовательский интерфейс – это главная составляющая впечатления пользователя о модели. Консольный интерфейс проще [15, 16, 18], удобен для организации диалогового режима между моделью и человеком, но графический интерфейс в виду широких возможностей визуального представления и обновления данных о текущем состоянии модели, позволяет отобразить больше информации.

1. Средство построения алгоритма моделирования. В большинстве моделей пользователю предоставляется возможность построить квантовые алгоритмы с помощью какого-то инструемента: графического редактора квантовой схемы [5, 4, 7] или же написать команды в текстовом редакторе квантового алгоритма [6, 16]. Удобство каждого из подходов зависит от величины алгоритма и квантовой системы, а так же реализации этого инструмента.

2. Ввод/вывод данных. В квантовых вычислениях одна и та же информация о состоянии квантового бита может быть загружена и отображена в различном виде: цветом представления амплитуды вероятности или вероятности состояния кубитов [4,6], числовым значением данных величин [18, 19] (цветовые значения) проще воспринимать, но численные необходимы для точности моделирования. Большое удобство предоставляет возможность отображать значения вероятностей только для заданной группы кубит [5].

3. Визуализация процессов моделирования. Понять квантовые процессы значительно легче, например, с помощью 3D-представления состояния квантового бита или системы кубит на сфере Блоха. Дополнительную информацию можно подчерпнуть из графиков, гистограмм вероятностей состояний регистра [7, 19, 5], они помогают понять суть вычислительных процессов в квантовых алгоритмах [17, 24].

IV. Моделирование.

Данный пункт подразумевает сравнение моделей по возможностям настройки модели и моделированию.

1. Математический подход. Влияет на точность и достоверность результатов моделирования. Моделирование с помощью переходов по графам позволяет значительно экономить память и время, необходимое на моделирование, но достоверность предоставляемых результатов оставляет желать лучшего [16]. Матричный подход предоставляет точные результаты, но потребляя очень большое количество ресурсов [19, 8, 13].

2. Отсутствие ограничений. В некоторых моделях ограничен объем моделируемой системы кубит определенным значением [17, 21], а остальные – только возможностями компьютера. Необычное ограничение в величине моделируемого количества шагов представлено в QCAD [5]. Иногда в процессе моделирования возникает потребность динамического добавления кубит во время моделирования, это реализовано далеко не во всех моделях [13].

3. Приостановка моделирования. Моделирование квантовых вычислений – процесс, требующий большое количество времени, поэтому возможность приостановки и сохранения результатов моделирования очень удобна для продолжения моделирования в другое время или построения различных вариантов последующего моделирования или просто для последующего анализа [19].

4. Физическая реалистичность модели. Моделирование идеального квантового компьютера подходит для изучения алгоритмов, логики их построения или теории квантовых вычислений. Модели физических процессов, например, декогеренции или шумов в NMR квантовом компьютере [8] необходимо для препятствования этим процессам.

5. Режимы выполнения алгоритма. Для отладки алгоритмов удобен пошаговый режим или возможность установки точек останова, а для моделирования алгоритмов больше подходит автоматическое выполнение всего алгоритма [6].

Оценка модели. На основе анализа и классификации можно качественно охарактеризовать каждую модель, выделив имеющиеся преимущества и недостатки, получив субъективную оценку, но она не позволяет четко распределить модели. Численная оценка реализованных функций по каждой из категорий характеристик позволит оценить модель комплексно. Также такой подход к оценке позволяет разбить общую оценку модели на уровневую оценку: оценить категорию характеристик и сами характеристики по отдельности.

В каждом подходе к сравнению модели оцениваются с различных сторон, поэтому суммарные оценки по категориям могут быть просуммированы:

$$S = T + A + I + M \cdots \quad (1)$$

где T,A, I, M, … – оценка категории, по которым выполнялось сравнение моделей, а S – суммарная оценка модели;

В соответствии со степенью значимости или критичностью данной категории характеристик, ей назначается свой весовой коэффициент, таким образом общая оценка имеет следующий вид:

$$S = a * T + b * A + c * I + d * U \cdots, \quad (2)$$

где a, b, c, … – весовые коэффициенты для каждой категории.

Суммарная оценка по категории складывается из суммы оценок характеристик, которые можно оценить как реализованная (1) или нереализованная (0) функция, поэтому оценка категории выглядит как

$$K = \sum_{j=1}^{N} p_j, \quad (3)$$

где p_j – оценка реализации j-ой характеристики, а K – суммарная оценка категории.

Следовательно, суммарная оценка модели выглядит как

$$K = a * \sum_{j=1}^{N_T} p_j + b * \sum_{j=1}^{N_A} p_j + c * \sum_{j=1}^{N_I} p_j + d * \sum_{j=1}^{N_M} p_j + \cdots \quad (5)$$

Таким образом, оценка модели будет складываться из суммарных оценок на уровне характеристик, умноженных на соответствующий коэффициент на уровне категории.

Такая организация расчета оценок позволит точно для каждой категории характеристик оценить его значимость и произвести численную оценку модели. Примечательно, что количество уровней в такой системе подсчета может быть увеличено.

Коэффициенты для каждой категории можно подобрать, проанализировав, какие характеристики рассматриваются в данной категории, насколько они важны для модели и их количества. Например, при плохо организованном интерфейсе хорошая архитектура принесет мало пользы, поэтому суммарная оценка по интерфейсу должна быть равна суммарной оценке по архитектуре. Возможности моделирования являются главными характеристиками для оценки модели, поэтому коэффициент должен быть самым большим. Некоторые характеристики фактически являются группами менее крупных, поэтому их оценка складывается в виде суммы оценок всех характеристик в группе, поделенная на их количество. Характеристика должна быть оценена в виде: присутствует (1 балл) и отсутствует (0 баллов), в некоторых случаях могут быть промежуточные варианты.

Пример оценки для модели QCAD:

Тип модели: T = 3 * 1 + 2 * 0 + 1 * 0 = 3.

Архитектура: A = 0 + 0 + 0 + 0 = 0.

Интерфейс: I = ½ *(0 + 1) + ½ * (1 + 1) + ½ * (0 + 1) = 1.5.

Моделирование: M = 1 + ½ * (0 + 1) + 0 + 0 + 0 = 1.5.

Суммарная оценка: 1 * 3 + 1 * 0 + 2 * 1.5 + 1.5 = 7,5.

В заключение выделим некоторые характеристики, которые при реализации модели сделают её лучшей среди существующих моделей, в виде требований к собственной модели. На основе анализа возможностей существующих моделей и оценки важности конкретных характеристик, величины их вклада в общую оценку, можно заключить, что модель должна иметь следующие возможности:

1. Многопоточность вычислений, для максимальной производительности.
2. Разделение на модули, для обеспечения независимости компонентов.
3. Графический интерфейс, позволяющий строить алгоритмы различными способами.
4. Наглядность всех преобразований в виде трехмерных моделей или диаграмм вероятностей системы, с возможностью селекции группы кубит для наблюдения.
5. Реализация функционала по переходу на различные этапы моделирования квантовых алгоритмов.
6. Реализация возможностей по приостановке и продолжению моделирования с перезапуском модели.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. University of Bristol, Multi-purpose photonic chip paves the way to programmable quantum processors URL: http://www.bris.ac.uk/news/2011/8109.html. (Дата обращения: 12.02.12).
2. Моделирование Википедия, URL: http://ru.wikipedia.org/wiki/Моделирование. (Дата обращения: 18.01.12).
3. Компьютерное моделирование Википедия, URL: http://ru.wikipedia.org/wiki/Компьютерное_моделирование (Дата обращения: 10 января 2012).
4. de Vries, jQuantum, URL: http://jquantum.sourceforge.net/. (Дата обращения: 18.01.12).
5. *Watanabe H., Suzuki M., Yamazaki J.* QCAD, University of Tokyo и Nagoya University. URL: http://qcad.sourceforge.jp/ (Дата обращения: 17.01.12).
6. *Raedt H.D., Hams A., Michielsen K., Raedt K.D.* Quantum Computer Emulator (QCE), University of Groningen, URL: http://rugth30.phys.rug.nl/qce/Default.aspx (Дата обращения: 17.01.12).
7. Quantum Algorithm Designer, URL: http://www-users.cs.york.ac.uk/~sok/QAD/.

8. *Butscher B., Weimer H.* Libquantum, URL: http://libquantum.de/ (Дата обращения: 17.01.12).
9. *Федотов И.Е.* Модели параллельного программирования. – М.: СОЛОН-ПРЕСС, 2012.
10. *Hertel J.* Quantum Turing Machine Simulator, URL: http://library.wolfram.com/infocenter/Articles/3893/ (Дата обращения: 18.01.12).
11. *Tucci R.R.* Matlab Functions and fun for Quantum Computer Programmers, URL: http://www.ar-tiste.com/m-fun/m-fun-index.html (Дата обращения: 18.01.12).,
12. *Логинов О.В., Цыганов А.В.* Квантовый алгоритм Гровера, Санкт-Петербургский Государственный Университет, URL: http://www.exponenta.ru/educat/systemat/grover/index.asp (Дата обращения: 18.01.12).
13. *Purkeypile M.* Cove: A Practical Quantum Computer Programming Framework, Colorado Technical University, URL: https://cove.purkeypile.com/trac/ (Дата обращения: 17.01.12).
14. *Greve D.* QDD: A Quantum Computer Emulation Library, URL: http://thegreves.com/david/QDD/qdd.html (Дата обращения: 18.01.12).
15. *Marcos E.* PyQu, URL: http://code.google.com/p/pyqu/ (Дата обращения: 19.01.12).
16. *Viamontes G.F. Markov I.L., Haye J.P.* QuIDDPro, University of Michigan, URL: http://vlsicad.eecs.umich.edu/Quantum/qp/ (Дата обращения: 24 Октябрь 2011].
17. *Shary S., Cahay D.M.* Bloch Sphere Simulation, University of Cincinnati, URL: http://www.ece.uc.edu/~mcahay/blochsphere/ (Дата обращения: 17.01.12).
18. *Clark S.* Linear Al University of Bristol, 16 Май 2006. URL: http://linearal.sourceforge.net/ (Дата обращения: 17.01.12).
19. Geeknet, Inc., SimQubit URL: http://simqubit.sourceforge.net/ (Дата обращения: 17.01.12).
20. *Díaz-Pier S., Venegas-Andraca S. E., Gómez-Muñoz J. L.*Classical Simulation of Quantum Adiabatic Algorithms using Mathematica on GPUs, URL: http://uk.arxiv.org/abs/1103.1399v1 (Дата обращения: 18.01.12).
21. *Wasserman J.* Squankum URL: http://www.pha.jhu.edu/~jeffwass/ squankum/ Дата обращения: 17.01.2012

Статью рекомендовал к опубликованию д.т.н., профессор Л.П. Фельдман.

Гузик Вячеслав Филиппович – Технологический институт федерального государственного автономного образовательного учреждения высшего профессионального образования «Южный федеральный университет» в г. Таганроге; e-mail: gvf@favt.tsure.ru; 347928, г. Таганрог, ул. Энгельса,1, ГСП-17А; тел.: 88634371737; кафедра вычислительной техники; зав. кафедрой; д.т.н.; профессор.

Гушанский Сергей Михайлович – кафедра вычислительной техники; к.т.н.; доцент.

Судаков Иван Александрович – студент 1 курса магистратуры.

Guzik Vacheslav Filipovich – Taganrog Institute of Technology – Federal State-Owned Autonomy Educational Establishment of Higher Vocational Education “Southern Federal University”; e-mail: gvf@favt.tsure.ru; GSP-17A, 1, Engels street, Taganrog, 347928, Russia; phone: +78634371737; the department of computer engineering; head the department; dr. of eng. sc.; professor.

Gushnsky Sergey Michailovich – the department of computer engineering; cand. of eng. sc.; associate professor.

Sudakov Ivan Alexandrovich – master of the first course.