КВАНТОВОЕ ПРЕВОСХОДСТВО И КАК ЕГО ДОСТИЧЬ: ПРОГРАММИРОВАНИЕ КВАНТОВЫХ КОМПЬЮТЕРОВ

Автор: Великанов Евгений Максимович,
ученик 11 класса МОУ «Лицей №5 имени Ю. А. Гагарина Центрального района Волгограда»,
г. Волгоград

Научный руководитель: Ульченко Екатерина Николаевна,
учитель информатики МОУ «Лицей №5 имени Ю. А. Гагарина Центрального района Волгограда»

Со временем мы сталкиваемся со все большим количеством задач, требующих мощные устройств. Проблема в том, что задачи усложняются в геометрической прогрессии, а рост производительности железа практически остановился. Квантовый компьютер может решать такие задачи намного быстрее и эффективнее классических, поэтому на сегодняшний день это крайне актуальное направление.

В чем отличие квантовых компьютеров от обычных? Как все знают, обычный компьютер использует **биты** – наименьшую единицу измерения информации. Их суть проста – 2 состояния, 3 основные операции и сколько угодно производных от них. При этом каждая базовая операция (функция) принимает 2 бита и возвращает 1. Квантовый компьютер использует **кубиты**, которые могут принимать не только противоположные крайние значения 0 и 1, как бит, но и промежуточные. Впрочем, можно сказать, что кубит принимает не промежуточные значения, а одновременно два – как ни странно, верно и первое, и второе. Можно представить, что кубит – это обычный выключатель света, который или замыкает цепь, или размыкает, а кубит – это ручка, которую можно прокрутить в любое положение и от этого будет зависеть яркость света. В любом случае, это состояние называется **суперпозицией**. По сути, именно оно и дает огромный прирост производительности. Так как кубит находится одновременно в двух состояниях, мы можем узнать оба возможных выхода, а если кубитов несколько – все возможные исходы (или решения). Для упрощения понимания приведу простую аналогию. Допустим, что у человека есть лишь два состояния – сонный и бодрый. Мы хотим узнать, как в обоих случаях он отреагирует на бросок мяча. В случае с битами нам надо сделать бросок бодрому человеку и бросок сонному, итого – два броска. В случае с кубитами мы кидаем мяч человеку, который находится в состоянии суперпозиции, и одновременно видим реакцию и сонного, и бодрого человеков. Да, это не соответствует привычной человеку логике, и это серьезная проблема квантовых компьютеров: программисту нужно абстрагироваться от привычной логики и думать в рамках кубитов.

Если вы слышали о **коте Шрёдингера**, то наверняка заметили сходство с этим экспериментом. Квантовая механика, из которой вытекает логика кубитов, исследуется уже давно, существует масса экспериментов, большая часть из которых мысленная, например – **квантовое самоубийство**. Несмотря на это, квантовые компьютеры еще не вошли в нашу жизнь. Главная причина – сложность и необычность концепции кубитов. До этого момента в статье не упоминались конкретные операции над кубитами. Итак, главный элемент квантового программирования – вентили (англ. *Gates*, букв. перев. *Врата*). Вентиль получает один или несколько кубитов и переводит их в какое-то состояние. Само состояние кубитов описывается формулой $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$, где $|\psi\rangle$ - итоговое состояние, а α и β – некоторые коэффициенты. При любой попытке измерить состояние кубита он неизбежно коллапсирует – принимает конкретное значение, поэтому для работы с кубитами применяют аппарат теории вероятностей. α^2 – вероятность получения состояния $|0\rangle$, а, β^2 – состояния $|1\rangle$. К основным вентилям относятся:

- NOT - переводит кубит из состояния $\alpha\,|0\rangle + \beta\,|1\rangle$ в $\alpha\,|1\rangle + \beta\,|0\rangle$.
- Z – переводит состояние $|0\rangle$ в $|0\rangle$, а $|1\rangle$ – в $-|1\rangle$.
- Вентиль Адамара (англ Hadamard gate) – создает суперпозицию состояний $|0\rangle$ и $|1\rangle$.
- CNOT – получает 2 кубита, первый становится «управляющим», второй – «управляемым». Если управляющий кубит находится в состоянии $|0\rangle$, то состояние управляемого кубита не меняется. Если управляющий кубит находится в состоянии $|1\rangle$, то к управляемому применяется вентиль NOT.

Также большое значение имеет состояние квантовой запутанности. Допустим, мы измеряем состояние первого кубита. С вероятностью 0,5 мы получаем результат $|10\rangle$. Это означает, что состояние после измерения $|\psi'\rangle = |00\rangle$, или $|1\rangle$ с той же вероятностью 0,5, а состояние после измерения $|\psi'\rangle = |11\rangle$. Теперь мы измеряем состояние второго кубита. После измерения пара будет находиться в состоянии $|00\rangle$ или $|11\rangle$. Если после этого мы решим снова измерить состояние первого кубита, вероятности уже не будут равны 0,5. Он окажется в состоянии $|1\rangle$ с вероятностью 0 или 1 – в зависимости от результата измерения. Эти результаты связаны между собой, и такое состояние называется квантовой запутанностью.

В ходе исследования мы работали в открытом для всех сервисе IBM quantum experience, дающем доступ к квантовому компьютеру IBM, и пришли к ряду выводов:

- Квантовый компьютер дает огромное количество операций, реализуемых на физическом уровне и недоступных на обычном. Кроме того, состояние квантовой

запутанности и одновременная обработка всех возможных значений кубитов позволяют практически мгновенно обрабатывать огромные объемы данных;

- главные фактором, мешающим свободному программированию квантовых компьютеров, является отсутствие низко- и высокоуровневых языков программирования. Все, что существует, ориентированно на работу напрямую с кубитами. Этого достаточно для понимания преимуществ и недостатков, но недостаточно для того, чтобы программист мог сосредоточиться на алгоритмике;
- необходимо создать общий интерфейс для прямого и удаленного управления квантовым компьютером.

В результате мы разработали концепцию системы управления квантовым компьютером:

– Сам «квантовый компьютер» (сервер) является гибридным. Классическая часть обеспечивает связь данных с квантовым регистром. Квантовая часть только выполняет переданный ей код, применяя входные данные.

– Удаленное устройство (клиент) является связью между программистом и сервером. Пользователь пишет код, определяет входные данные, устройство передает их серверу, обрабатывает возвращенный результат и выводит его в таком виде.

– Язык, на котором пишет пользователь, является гибридным (одновременно низко- и высокоуровневым). Программист имеет доступ к классическим операциям обычных языков программирования (циклы, ветвления и т.д.) и заранее определенным функциям и методам для работы с кубитами, при этом внутренняя часть сервера – квантовый регистр – остается «черным ящиком».

Такая система позволит минимизировать шанс ошибки из-за разницы между классической и квантовой логикой, но при этом оставляет возможность создавать эффективный код.