Задачи теории сложности вычислений и алгоритм Гровера

Владимир Подольский

Математический институт им. В.А. Стеклова РАН
Высшая Школа Экономики, Факультет компьютерных наук

3 марта 2022

План

- Сложностные классы P и NP
- Алгоритм Гровера решения задач перебора
- Алгоритм Гровера и NP-полные задачи

Сложность вычислений

- Изучает и классифицирует алгоритмические задачи с точки зрения их сложности
- Задача разрешения: по данному входу проверить, удовлетворяет ли он некоторому свойству
- Входы: конечные последовательности нулей и единиц (а также то, что ими можно закодировать)
- Множество всех конечных последовательностей нулей и единиц: $\{0,1\}^* = \bigcup_{n\in\mathbb{N}}\{0,1\}^n$
- Задача разрешения: зафиксировано множество $L \subseteq \{0,1\}^*$, требуется по данному $x \in \{0,1\}^*$ проверить, верно ли $x \in L$

Сложностной класс P

Задача: $L \subseteq \{0,1\}^*$.

Алгоритм A решает задачу L, если по всякому входу $x \in L$ он выдает 1, а на всяком входе $x \notin L$ – выдает 0.

Алгоритм A работает за полиномиальное время, если для всякого $x \in \{0,1\}^n$ алгоритм A останавливается на входе x за время, не большее $O(n^k)$ для некоторого фиксированного k. Такой алгоритм называется полиномиальным.

Класс P: L лежит в сложностном классе P, если есть полиномиальный алгоритм A, решающий L.

Примеры задач из P

Разрешимость системы линейных уравнений над $\mathbb{Q}$. Дана система линейных уравнений над $\mathbb{Q}$, требуется проверить, есть ли у нее решение.

Разрешимость системы линейных неравенств над $\mathbb{Q}$. Дана система линейных неравенств над $\mathbb{Q}$, требуется проверить, есть ли у нее решение.

Задача о проверке на простоту. Дано натуральное число n, требуется проверить, является ли n простым

Сложностной класс NP

Неформально: задача L лежит в NP, если для утверждения $x \in L$ можно предъявить доказательство, проверяемое за полиномиальное время.

NP: Задача L лежит в NP если есть полиномиальный алгоритм A и многочлен $p(n)$, такие что для всякого n и всякого $x \in \{0,1\}^n$

$$x \in L \Leftrightarrow \exists y \in \{0,1\}^{p(n)}\ A(x,y) = 1.$$

Строка y называется доказательством или сертификатом для $x \in L$.

Неформальный пример

Как может выглядеть задача, которую сложно решить, но у которой легко проверить решение?

Даны два числа $a, n \in \mathbb{N}$
Вопрос: существует ли $t \in \mathbb{N}$, такое что $t^2 \equiv a \pmod{n}$?

Мы не знаем полиномиального алгоритма

Легко проверяемое доказательство: t

Примеры задач в NP

NP: Задача L лежит в NP если есть полиномиальный алгоритм A и многочлен $p(n)$, такие что для всякого n и всякого $x \in \{0,1\}^n$

$$x \in L \Leftrightarrow \exists y \in \{0,1\}^{p(n)}\ A(x,y) = 1.$$

Разрешимость системы линейных неравенств над $\mathbb{Z}$. Дана система линейных неравенств над $\mathbb{Z}$, требуется проверить, есть ли у нее решение.

Разрешимость системы линейных неравенств над $\mathbb{Z}$ в $x \in \{0,1\}^n$. Дана система линейных неравенств над $\mathbb{Z}$, требуется проверить, есть ли у нее решение, все координаты которого равны 0 или 1.

NP-полные задачи

- NP-полные задачи — самые трудные задачи в NP
- Формально, задача в NP является NP-полной, если любая другая задача из NP к ней эффективно сводится
- Пример: задачи о системах линейных неравенств над $\mathbb{Z}$ являются NP-полными
- Есть много важных задач в самых разных областях, являющихся NP-полными
- Проще назвать NP-полную задачу, чем ту, для которой не известна ни NP-полнота, ни принадлежность P.
- Зачем изучать NP-полные задачи: вопрос о равенстве классов P и NP сводится к полиномиальной разрешимости (любой) одной NP-полной задачи

Задача существования vs. задача поиска

Есть задача о существовании решения линейной системы над $\mathbb{Z}$ в множестве $x \in \{0,1\}^n$, есть задача о поиска решения. Есть ли связь?

Если задача существования лежит в P, то можно и находить решение за полиномиальное время
То же самое для остальных задач

Как:
Проверяем есть ли решение
Если есть, фиксируем $x_1 = 0$ и проверяем, есть ли решение у получившейся системы
Если да, переходим к следующей переменной
Если нет, фиксируем $x_1 = 1$ и переходим к следующей переменной

Все это работает за полиномиальное время

Квантовые вычисления

- Получаем вход $x \in \{0,1\}^n$ как $|x\rangle$
- Последовательно применяем к $|x\rangle$ базовые унитарные операторы
- В результате получаем результирующий вектор

$$|\varphi\rangle = \sum_{y \in \{0,1\}^n} c_y |y\rangle$$

- Проводим измерение, получаем ответ $y \in \{0,1\}^n$ с вероятностью $|c_y|^2$
- Требуется получить правильный ответ с вероятностью не меньше $2/3$

Квантовые вычисления: идея

- Как квантовые алгоритмы могли бы помочь?
- n кубитов дают 2^n-мерное пространство
- Пусть мы хотим решать систему линейных неравенств в $x \in \{0,1\}^n$
- Сопоставим каждому $x \in \{0,1\}^n$ одну из размерностей этого пространства
- Можем работать со смешанными состояниями в этом пространстве
- Если сможем преобразовывать состояние так, чтобы координаты-решения росли, а остальные нет, то решим задачу

Алгоритм Гровера, постановка

- Пусть для задачи $L \subseteq \{0,1\}^*$ известно, что

$$x \in L \Leftrightarrow \exists y \in \{0,1\}^{p(n)} \; A(x,y) = 1.$$

- Хотим решать задачу L
- Предполагаем, что A — это черный ящик или оракул
- Мы предполагаем, что ничего не знаем о его структуре
- Мы можем делать запросы к A и мгновенно получать ответ

Формально, наряду со стандартными операциями мы можем использовать оператор

$$U_x \left\langle y \right| = \begin{cases} \left\langle y \right|, & \text{если } A(x,y) = 0, \\ -\left\langle y \right|, & \text{если } A(x,y) = 1. \end{cases}$$

Алгоритм Гровера

Лемма

Классический алгоритм поиска y, такого что $A(x, y) = 1$, требует 2^n обращений к оракулу

Теорема (Гровер, '96)

Существует квантовый алгоритм, находящий y, такой что $A(x, y) = 1$, и делающий $O(2^{n/2})$ обращений к оракулу

Алгоритм Гровера, идея доказательства

$$U_x |y\rangle = \begin{cases} |y\rangle, & \text{если } A(x,y) = 0, \\ -|y\rangle, & \text{если } A(x,y) = 1. \end{cases}$$

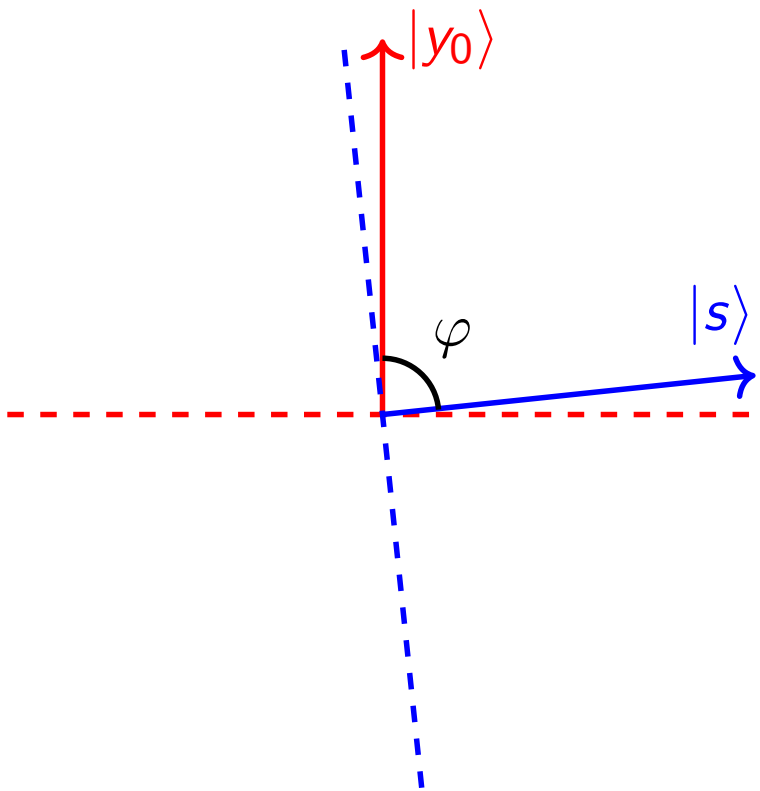
- Для простоты пусть существует единственный y_0, такой что $A(x, y_0) = 1$
- Тогда U_x — оператор отражения относительно гиперплоскости, ортогональной (неизвестному) вектору $|y_0\rangle$
- Рассмотрим вектор

$$|s\rangle = \frac{1}{2^{n/2}} \sum_{y \in \{0,1\}^n} |y\rangle$$

- Рассмотрим оператор V отражения относительно плоскости, ортогональной $|s\rangle$

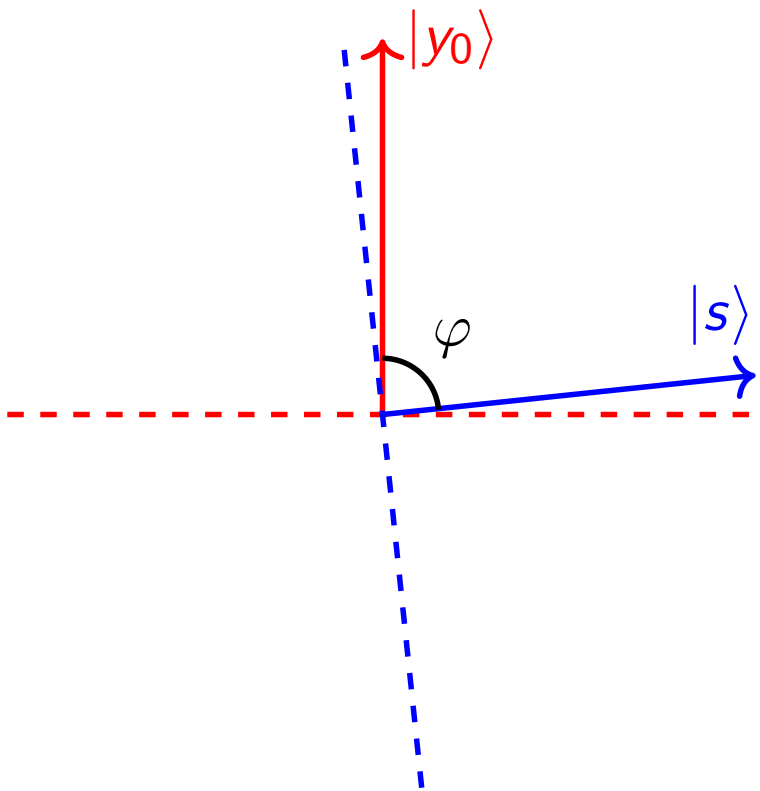
Алгоритм Гровера, идея доказательства

- U_x — оператор отражения относительно гиперплоскости, ортогональной $|y_0\rangle$
- V — оператор отражения относительно гиперплоскости, ортогональной $|s\rangle$



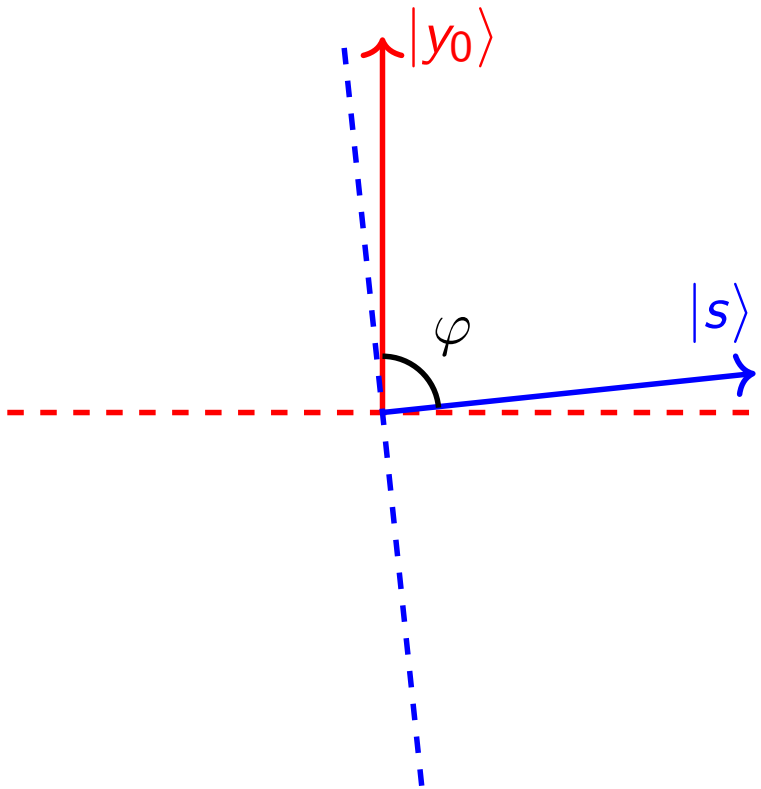
Алгоритм Гровера, идея доказательства

- VU_x — оператор поворота в плоскости, натянутой на $|y_0\rangle$, $|s\rangle$, на удвоенный угол между векторами $|y_0\rangle$ и $|s\rangle$
- Косинус угла φ между векторами равен $\langle y_0|s\rangle = \frac{1}{2^{n/2}}$, угол почти прямой: $\varphi \approx \pi - \frac{1}{2^{n/2}}$



Алгоритм Гровера, идея доказательства

- Удвоенный угол почти развернутый $2\varphi \approx 2\pi - \frac{2}{2^{n/2}}$
- Начинаем с вектора $|s\rangle$, делаем поворот $2^{n/2}/2$ раз
- Получаем вектор очень близкий к $|y_0\rangle$ (с точностью до знака)
- При измерении с большой вероятностью получим y_0



Алгоритм Гровера, оптимальность

Теорема (Залка, '99)

Всякий квантовый алгоритм поиска y, такого что $A(x, y) = 1$, делает не меньше $\Omega(2^{n/2})$ запросов к оракулу

Здесь существенно, что $A(x, y)$ — черный ящик, для которого мы можем лишь применять оператор U_x

Оптимальность, идея

Пусть есть алгоритм, с t обращениями к оракулу. Он имеет вид

$$|\varphi_{x,t}\rangle = U_t U_x U_{t-1} U_x \dots U_0 |\varphi\rangle$$

Рассмотрим также вектор

$$|\varphi_t\rangle = U_t U_{t-1} \dots U_0 |\varphi\rangle$$

Он соответствует работе алгоритма для случая $A(x, y) \equiv 0$
Идея:
Рассмотрим $\| |\varphi_{x,k}\rangle - |\varphi_k\rangle \|$
С одной стороны, для $A(x, y) \equiv 0$ и для остальных A ответ должен быть разный, так что расстояние большое, не меньше некоторой константы.
С другой стороны, его можно оценить сверху как $O\left(\frac{t}{2^{n/2}}\right)$

Заключение

- Алгоритм Гровера позволяет получить квадратичное ускорение перебора в модели с черным ящиком
- Для модели черного ящика дальнейшее ускорение невозможно
- В случае конкретной NP-полной задачи мы имеем дело не с черным ящиком
- Теоретически, для конкретных NP-полных задач дальнейшее ускорение возможно
- Но оно должно использовать внутреннюю структуру $A(x, y)$
- Для этого нужны новые идеи