Приближенное решение задач комбинаторной оптимизации: алгоритмы и трудность
Лекция 7: PCP теорема

М. Вялый

Вычислительный центр
им. А.А.Дородницына
ФИЦ ИУ РАН

Санкт-Петербург, Computer Science Club, 2016

PCP теорема

Теорема (PCP теорема, комбинаторный вариант)

Для некоторых констант $\alpha > 0$ и q задача MAX-2CSP$_q(1, 1-\alpha)$ является NP-трудной.

Из предыдущих сводимостей получаем следствия.

Следствие 1

Задача MAX-3SAT$(1, 1-\varepsilon)$ является NP-трудной для некоторой константы ε.

Следствие 2

Для некоторого $\varepsilon > 0$ задача MAX-LC$_7(1, 1-\varepsilon)$ является NP-трудной.

PCP теорема

Теорема (PCP теорема, комбинаторный вариант)

Для некоторых констант $\alpha > 0$ и q задача MAX-2CSP$_q(1, 1-\alpha)$ является NP-трудной.

Из предыдущих сводимостей получаем следствия.

Следствие 1

Задача MAX-3SAT$(1, 1-\varepsilon)$ является NP-трудной для некоторой константы ε.

Следствие 2

Для некоторого $\varepsilon > 0$ задача MAX-LC$_7(1, 1-\varepsilon)$ является NP-трудной.

PCP теорема

Теорема (PCP теорема, комбинаторный вариант)

Для некоторых констант $\alpha > 0$ и q задача $\text{MAX-2CSP}_q(1, 1-\alpha)$ является NP-трудной.

Из предыдущих сводимостей получаем следствия.

Следствие 1

Задача $\text{MAX-3SAT}(1, 1-\varepsilon)$ является NP-трудной для некоторой константы ε.

Следствие 2

Для некоторого $\varepsilon > 0$ задача $\text{MAX-LC}_7(1, 1-\varepsilon)$ является NP-трудной.

О доказательстве Irit Dinur

1. $\text{MAX-2CSP}_3(1, 1 - n^{-2})$ является NP-трудной (нарушение 3-раскраски хотя бы на одном ребре даёт потери $1/n^2$).
2. Сводимость $\text{MAX-2CSP}_3(1, 1 - n^{-2}) \to \text{MAX-2CSP}_q(1, 1 - \alpha)$ строится как композиция $O(\log n)$ сводимостей «удвоения щели»:

   $$\text{MAX-2CSP}_q(1, 1 - \varepsilon) \to \text{MAX-2CSP}_q(1, 1 - 2\varepsilon), \quad \varepsilon < \alpha,$$

   здесь достаточно велико ($q = 2^{10}$ вроде бы хватает).
   Для полиномиальности композиции $O(\log n)$ сводимостей нужно, чтобы каждая изменяла размеры графа в $O(1)$ раз.
3. Сводимость удвоения щели получается в четыре шага:
   1. Улучшение графа ограничений ценой сужения щели.
   2. Расширение щели ценой увеличения алфавита.
   3. Уменьшение алфавита ценой увеличения арности и сужения щели.
   4. Возврат к 2-выполнимости ценой увеличения алфавита.

   Волшебным образом в итоге щель удваивается, а остальные параметры сохраняются.

О доказательстве Irit Dinur

1. $\text{MAX-2CSP}_3(1, 1 - n^{-2})$ является NP-трудной (нарушение 3-раскраски хотя бы на одном ребре даёт потери $1/n^2$).
2. Сводимость $\text{MAX-2CSP}_3(1, 1 - n^{-2}) \to \text{MAX-2CSP}_q(1, 1 - \alpha)$ строится как композиция $O(\log n)$ сводимостей «удвоения щели»:

   $$\text{MAX-2CSP}_q(1, 1 - \varepsilon) \to \text{MAX-2CSP}_q(1, 1 - 2\varepsilon), \quad \varepsilon < \alpha,$$

   здесь достаточно велико ($q = 2^{10}$ вроде бы хватает).
   Для полиномиальности композиции $O(\log n)$ сводимостей нужно, чтобы каждая изменяла размеры графа в $O(1)$ раз.
3. Сводимость удвоения щели получается в четыре шага:
   1. Улучшение графа ограничений ценой сужения щели.
   2. Расширение щели ценой увеличения алфавита.
   3. Уменьшение алфавита ценой увеличения арности и сужения щели.
   4. Возврат к 2-выполнимости ценой увеличения алфавита.

   Волшебным образом в итоге щель удваивается, а остальные параметры сохраняются.

О доказательстве Irit Dinur

1. $\text{MAX-2CSP}_3(1, 1 - n^{-2})$ является $\mathbf{NP}$-трудной (нарушение 3-раскраски хотя бы на одном ребре даёт потери $1/n^2$).
2. Сводимость $\text{MAX-2CSP}_3(1, 1 - n^{-2}) \to \text{MAX-2CSP}_q(1, 1 - \alpha)$ строится как композиция $O(\log n)$ сводимостей «удвоения щели»:

$$\text{MAX-2CSP}_q(1, 1 - \varepsilon) \to \text{MAX-2CSP}_q(1, 1 - 2\varepsilon), \quad \varepsilon < \alpha,$$

здесь достаточно велико ($q = 2^{10}$ вроде бы хватает).
Для полиномиальности композиции $O(\log n)$ сводимостей нужно, чтобы каждая изменяла размеры графа в $O(1)$ раз.
3. Сводимость удвоения щели получается в четыре шага:
   1. Улучшение графа ограничений ценой сужения щели.
   2. Расширение щели ценой увеличения алфавита.
   3. Уменьшение алфавита ценой увеличения арности и сужения щели.
   4. Возврат к 2-выполнимости ценой увеличения алфавита.

Волшебным образом в итоге щель удваивается, а остальные параметры сохраняются.

О доказательстве Irit Dinur

1. $\text{MAX-2CSP}_3(1, 1 - n^{-2})$ является NP-трудной (нарушение 3-раскраски хотя бы на одном ребре даёт потери $1/n^2$).
2. Сводимость $\text{MAX-2CSP}_3(1, 1 - n^{-2}) \to \text{MAX-2CSP}_q(1, 1 - \alpha)$ строится как композиция $O(\log n)$ сводимостей «удвоения щели»:

$$\text{MAX-2CSP}_q(1, 1 - \varepsilon) \to \text{MAX-2CSP}_q(1, 1 - 2\varepsilon), \quad \varepsilon < \alpha,$$

здесь достаточно велико ($q = 2^{10}$ вроде бы хватает).
Для полиномиальности композиции $O(\log n)$ сводимостей нужно, чтобы каждая изменяла размеры графа в $O(1)$ раз.
3. Сводимость удвоения щели получается в четыре шага:
   1. Улучшение графа ограничений ценой сужения щели.
   2. Расширение щели ценой увеличения алфавита.
   3. Уменьшение алфавита ценой увеличения арности и сужения щели.
   4. Возврат к 2-выполнимости ценой увеличения алфавита.

Волшебным образом в итоге щель удваивается, а остальные параметры сохраняются.

О доказательстве Irit Dinur

1. $\text{MAX-2CSP}_3(1, 1 - n^{-2})$ является NP-трудной (нарушение 3-раскраски хотя бы на одном ребре даёт потери $1/n^2$).
2. Сводимость $\text{MAX-2CSP}_3(1, 1 - n^{-2}) \to \text{MAX-2CSP}_q(1, 1 - \alpha)$ строится как композиция $O(\log n)$ сводимостей «удвоения щели»:
$$\text{MAX-2CSP}_q(1, 1 - \varepsilon) \to \text{MAX-2CSP}_q(1, 1 - 2\varepsilon), \quad \varepsilon < \alpha,$$
здесь достаточно велико ($q = 2^{10}$ вроде бы хватает).
Для полиномиальности композиции $O(\log n)$ сводимостей нужно, чтобы каждая изменяла размеры графа в $O(1)$ раз.
3. Сводимость удвоения щели получается в четыре шага:
    1. Улучшение графа ограничений ценой сужения щели.
    2. Расширение щели ценой увеличения алфавита.
    3. Уменьшение алфавита ценой увеличения арности и сужения щели.
    4. Возврат к 2-выполнимости ценой увеличения алфавита.

Волшебным образом в итоге щель удваивается, а остальные параметры сохраняются.

О доказательстве Irit Dinur

1. $\mathrm{MAX\text{-}2CSP}_3(1, 1 - n^{-2})$ является NP-трудной (нарушение 3-раскраски хотя бы на одном ребре даёт потери $1/n^2$).
2. Сводимость $\mathrm{MAX\text{-}2CSP}_3(1, 1 - n^{-2}) \to \mathrm{MAX\text{-}2CSP}_q(1, 1 - \alpha)$ строится как композиция $O(\log n)$ сводимостей «удвоения щели»:
$$\mathrm{MAX\text{-}2CSP}_q(1, 1 - \varepsilon) \to \mathrm{MAX\text{-}2CSP}_q(1, 1 - 2\varepsilon), \quad \varepsilon < \alpha,$$
здесь достаточно велико ($q = 2^{10}$ вроде бы хватает).
Для полиномиальности композиции $O(\log n)$ сводимостей нужно, чтобы каждая изменяла размеры графа в $O(1)$ раз.
3. Сводимость удвоения щели получается в четыре шага:
    1. Улучшение графа ограничений ценой сужения щели.
    2. Расширение щели ценой увеличения алфавита.
    3. Уменьшение алфавита ценой увеличения арности и сужения щели.
    4. Возврат к 2-выполнимости ценой увеличения алфавита.

Волшебным образом в итоге щель удваивается, а остальные параметры сохраняются.

О доказательстве Irit Dinur

1. $\text{MAX-2CSP}_3(1, 1 - n^{-2})$ является NP-трудной (нарушение 3-раскраски хотя бы на одном ребре даёт потери $1/n^2$).
2. Сводимость $\text{MAX-2CSP}_3(1, 1 - n^{-2}) \to \text{MAX-2CSP}_q(1, 1 - \alpha)$ строится как композиция $O(\log n)$ сводимостей «удвоения щели»:
$$\text{MAX-2CSP}_q(1, 1 - \varepsilon) \to \text{MAX-2CSP}_q(1, 1 - 2\varepsilon), \quad \varepsilon < \alpha,$$
здесь достаточно велико ($q = 2^{10}$ вроде бы хватает).
Для полиномиальности композиции $O(\log n)$ сводимостей нужно, чтобы каждая изменяла размеры графа в $O(1)$ раз.
3. Сводимость удвоения щели получается в четыре шага:
   1. Улучшение графа ограничений ценой сужения щели.
   2. Расширение щели ценой увеличения алфавита.
   3. Уменьшение алфавита ценой увеличения арности и сужения щели.
   4. Возврат к 2-выполнимости ценой увеличения алфавита.

Волшебным образом в итоге щель удваивается, а остальные параметры сохраняются.

О доказательстве Irit Dinur

1. $\text{MAX-2CSP}_3(1, 1 - n^{-2})$ является NP-трудной (нарушение 3-раскраски хотя бы на одном ребре даёт потери $1/n^2$).
2. Сводимость $\text{MAX-2CSP}_3(1, 1 - n^{-2}) \to \text{MAX-2CSP}_q(1, 1 - \alpha)$ строится как композиция $O(\log n)$ сводимостей «удвоения щели»:

$$\text{MAX-2CSP}_q(1, 1 - \varepsilon) \to \text{MAX-2CSP}_q(1, 1 - 2\varepsilon), \quad \varepsilon < \alpha,$$

здесь достаточно велико ($q = 2^{10}$ вроде бы хватает).
Для полиномиальности композиции $O(\log n)$ сводимостей нужно, чтобы каждая изменяла размеры графа в $O(1)$ раз.
3. Сводимость удвоения щели получается в четыре шага:
   1. Улучшение графа ограничений ценой сужения щели.
   2. Расширение щели ценой увеличения алфавита.
   3. Уменьшение алфавита ценой увеличения арности и сужения щели.
   4. Возврат к 2-выполнимости ценой увеличения алфавита.

   Волшебным образом в итоге щель удваивается, а остальные параметры сохраняются.

Игры и классы сложности

Игровое определение класса NP

Два игрока: Проверяющий V (verifier) и Доказывающий P (prover).
V полиномиально ограничен, а P знает всё.
Цель V — убедиться, что входное слово x принадлежит языку L. Цель P — вынудить V признать, что слово принадлежит языку.
Игра в один раунд: P посылает сообщение y (доказательство) V, а тот изучает его и принимает решение.

Наблюдение

V выигрывает тогда и только тогда, когда язык из NP.
Предикат $R(x, y)$ — это в точности правило принятия решения V.

- если $x \in L$, то $\exists y R(x, y)$. P посылает y, V соглашается признать $x \in L$.
- если $x \notin L$, то $\forall y \neg R(x, y)$. V отказывается признать $x \in L$ при любом сообщении P.

Игры и классы сложности

Игровое определение класса NP

Два игрока: Проверяющий V (verifier) и Доказывающий P (prover).
V полиномиально ограничен, а P знает всё.
Цель V — убедиться, что входное слово x принадлежит языку L. Цель P — вынудить V признать, что слово принадлежит языку.
Игра в один раунд: P посылает сообщение y (доказательство) V, а тот изучает его и принимает решение.

Наблюдение

V выигрывает тогда и только тогда, когда язык из NP.
Предикат $R(x, y)$ — это в точности правило принятия решения V.

- если $x \in L$, то $\exists y R(x, y)$. P посылает y, V соглашается признать $x \in L$.
- если $x \notin L$, то $\forall y \neg R(x, y)$. V откажется признать $x \in L$ при любом сообщении P.

Игры и классы сложности

Игровое определение класса NP

Два игрока: Проверяющий V (verifier) и Доказывающий P (prover).
V полиномиально ограничен, а P знает всё.
Цель V — убедиться, что входное слово x принадлежит языку L. Цель P — вынудить V признать, что слово принадлежит языку.
Игра в один раунд: P посылает сообщение y (доказательство) V, а тот изучает его и принимает решение.

Наблюдение

V выигрывает тогда и только тогда, когда язык из NP.
Предикат $R(x, y)$ — это в точности правило принятия решения V.

- если $x \in L$, то $\exists y R(x, y)$. P посылает y, V соглашается признать $x \in L$.
- если $x \notin L$, то $\forall y \neg R(x, y)$. V откажется признать $x \in L$ при любом сообщении P.

Игры и классы сложности

Игровое определение класса NP

Два игрока: Проверяющий V (verifier) и Доказывающий P (prover). V полиномиально ограничен, а P знает всё.
Цель V — убедиться, что входное слово x принадлежит языку L. Цель P — вынудить V признать, что слово принадлежит языку.
Игра в один раунд: P посылает сообщение y (доказательство) V, а тот изучает его и принимает решение.

Наблюдение

V выигрывает тогда и только тогда, когда язык из NP.
Предикат $R(x, y)$ — это в точности правило принятия решения V.

- если $x \in L$, то $\exists y R(x, y)$. P посылает y, V соглашается признать $x \in L$.
- если $x \notin L$, то $\forall y \neg R(x, y)$. V откажется признать $x \in L$ при любом сообщении P.

Класс PCP

Проверяющий спешит. Хочет не читать всё доказательство, а посмотреть «на ключевые места» и после этого принять решение.

Машина Тьюринга с произвольным доступом к памяти

Имеет три ленты: входная, рабочая и адресный регистр.
С рабочей лентой и адресным регистром работает по обычным правилам. В любой момент может выполнить специальное действие: запросить значение ячейки входной ленты, номер которой записан в адресном регистре (в двоичной системе).

Класс PCP

Проверяющий спешит. Хочет не читать всё доказательство, а посмотреть «на ключевые места» и после этого принять решение.

Машина Тьюринга с произвольным доступом к памяти

Имеет три ленты: входная, рабочая и адресный регистр.
С рабочей лентой и адресным регистром работает по обычным правилам. В любой момент может выполнить специальное действие: запросить значение ячейки входной ленты, номер которой записан в адресном регистре (в двоичной системе).

Модификация игры

V полиномиальный вероятностный алгоритм, а P знает всё.
Игра в один раунд: P посылает сообщение y, а V читает из него $O(q(n))$ битов, используя $O(r(n))$ случайных битов, после чего принимает решение.

Определение класса $\mathrm{PCP}_{c,s}(r(n), q(n))$

$L \in \mathrm{PCP}_{c,s}(r(n), q(n)) \Leftrightarrow$ у V есть алгоритм (стратегия), при которой

- (полнота) для $x \in L$ существует y, который V примет с вероятностью $\geqslant c$;
- (корректность) для $x \notin L$ любое доказательство принимается V с вероятностью $< s$.

Модификация игры

V полиномиальный вероятностный алгоритм, а P знает всё.
Игра в один раунд: P посылает сообщение y, а V читает из него $O(q(n))$ битов, используя $O(r(n))$ случайных битов, после чего принимает решение.

Определение класса $\mathrm{PCP}_{c,s}(r(n), q(n))$

$L \in \mathrm{PCP}_{c,s}(r(n), q(n)) \Leftrightarrow$ у V есть алгоритм (стратегия), при которой

- (полнота) для $x \in L$ существует y, который V примет с вероятностью $\geqslant c$;
- (корректность) для $x \notin L$ любое доказательство принимается V с вероятностью $< s$.

Вторая формулировка PCP теоремы

PCP теорема (сложностной вариант)

Для некоторой константы $\varepsilon > 0$ выполняется

$$\mathrm{PCP}_{1,1-\varepsilon}(\log n, 1) = \mathrm{NP}.$$

Лёгкая часть

$\mathrm{PCP}_{1,1-\varepsilon}(\log n, 1) \subseteq \mathrm{NP}$ (PCP алгоритм смотрит только $2^{c \log n} = \mathrm{poly}(n)$ мест в доказательстве).

Вторая формулировка PCP теоремы

PCP теорема (сложностной вариант)

Для некоторой константы $\varepsilon > 0$ выполняется

$$\mathrm{PCP}_{1,1-\varepsilon}(\log n, 1) = \mathrm{NP}.$$

Лёгкая часть

$\mathrm{PCP}_{1,1-\varepsilon}(\log n, 1) \subseteq \mathrm{NP}$ (PCP алгоритм смотрит только $2^{c \log n} = \mathrm{poly}(n)$ мест в доказательстве).

Сложностной вариант следует из комбинаторного

Комбинаторный вариант: задача MAX-2CSP$_q(1, 1-\alpha)$ является NP-трудной.
Пусть $L \in \mathrm{NP}$, тогда $L \leqslant_p$ MAX-2CSP$_q(1, 1-\alpha)$.
Сводимость $x \mapsto G_x$, причём

$$(x \in L) \Rightarrow (\textsc{unsat}(G_x) = 0), \qquad (x \notin L) \Rightarrow (\textsc{unsat}(G_x) > \alpha).$$

PCP алгоритм:

P предъявляет присваивание для графа G_x, то есть q-ичное слово длины $\ell = \mathrm{poly}(|x|)$, где $\ell = |V(G_x)|$.
V выбирает случайно и равновероятно ребро G_x ($O(\log n)$ случайных битов), запрашивает значения переменных в концах ребра ($O(1)$ битов) и проверяет выполнение ограничения для этих значений переменных.
V принимает доказательство, только если ограничение выполнено.
Если $x \in L$, то P предъявит присваивание с $\textsc{unsat}(G_x) = 0$.
Если $x \notin L$, то любое присваивание нарушает хотя бы долю α ограничений. V обнаружит нарушение с вероятностью $\geqslant \alpha$.

Сложностной вариант следует из комбинаторного

Комбинаторный вариант: задача $\text{MAX-2CSP}_q(1, 1-\alpha)$ является NP-трудной.
Пусть $L \in \mathrm{NP}$, тогда $L \leqslant_p \text{MAX-2CSP}_q(1, 1-\alpha)$.
Сводимость $x \mapsto G_x$, причём

$$(x \in L) \Rightarrow (\textsc{unsat}(G_x) = 0), \qquad (x \notin L) \Rightarrow (\textsc{unsat}(G_x) > \alpha).$$

PCP алгоритм:
P предъявляет присваивание для графа G_x, то есть q-ичное слово длины $\ell = \mathrm{poly}(|x|)$, где $\ell = |V(G_x)|$.
V выбирает случайно и равновероятно ребро G_x ($O(\log n)$ случайных битов), запрашивает значения переменных в концах ребра ($O(1)$ битов) и проверяет выполнение ограничения для этих значений переменных.
V принимает доказательство, только если ограничение выполнено.
Если $x \in L$, то P предъявит присваивание с $\textsc{unsat}(G_x) = 0$.
Если $x \notin L$, то любое присваивание нарушает хотя бы долю α ограничений. V обнаружит нарушение с вероятностью $\geqslant \alpha$.

Сложностной вариант следует из комбинаторного

Комбинаторный вариант: задача $\text{MAX-2CSP}_q(1, 1-\alpha)$ является NP-трудной.
Пусть $L \in \mathrm{NP}$, тогда $L \leqslant_p \text{MAX-2CSP}_q(1, 1-\alpha)$.
Сводимость $x \mapsto G_x$, причём

$$(x \in L) \Rightarrow (\textsc{unsat}(G_x) = 0), \qquad (x \notin L) \Rightarrow (\textsc{unsat}(G_x) > \alpha).$$

PCP алгоритм:
P предъявляет присваивание для графа G_x, то есть q-ичное слово длины $\ell = \mathrm{poly}(|x|)$, где $\ell = |V(G_x)|$.
V выбирает случайно и равновероятно ребро G_x ($O(\log n)$ случайных битов), запрашивает значения переменных в концах ребра ($O(1)$ битов) и проверяет выполнение ограничения для этих значений переменных.
V принимает доказательство, только если ограничение выполнено.
Если $x \in L$, то P предъявит присваивание с $\textsc{unsat}(G_x) = 0$.
Если $x \notin L$, то любое присваивание нарушает хотя бы долю α ограничений. V обнаружит нарушение с вероятностью $\geqslant \alpha$.

Сложностной вариант следует из комбинаторного

Комбинаторный вариант: задача $\text{MAX-2CSP}_q(1, 1-\alpha)$ является NP-трудной.
Пусть $L \in \mathrm{NP}$, тогда $L \leqslant_p \text{MAX-2CSP}_q(1, 1-\alpha)$.
Сводимость $x \mapsto G_x$, причём

$$(x \in L) \Rightarrow (\text{UNSAT}(G_x) = 0), \qquad (x \notin L) \Rightarrow (\text{UNSAT}(G_x) > \alpha).$$

PCP алгоритм:
P предъявляет присваивание для графа G_x, то есть q-ичное слово длины $\ell = \mathrm{poly}(|x|)$, где $\ell = |V(G_x)|$.
V выбирает случайно и равновероятно ребро G_x ($O(\log n)$ случайных битов), запрашивает значения переменных в концах ребра ($O(1)$ битов) и проверяет выполнение ограничения для этих значений переменных.
V принимает доказательство, только если ограничение выполнено.
Если $x \in L$, то P предъявит присваивание с $\text{UNSAT}(G_x) = 0$.
Если $x \notin L$, то любое присваивание нарушает хотя бы долю α ограничений. V обнаружит нарушение с вероятностью $\geqslant \alpha$.

Комбинаторный вариант сильнее сложностного

Разница:

- в PCP алгоритме Проверяющий может выбирать места для чтения адаптивно, учитывая значения уже прочитанных битов доказательства;
- комбинаторный вариант отвечает неадаптивным проверкам: Проверяющий выбирает биты, которые он собирается прочесть, читает их и выносит решение, основываясь только на значениях прочитанных битов.

Комбинаторный вариант сильнее сложностного

Разница:

- в PCP алгоритме Проверяющий может выбирать места для чтения адаптивно, учитывая значения уже прочитанных битов доказательства;
- комбинаторный вариант отвечает неадаптивным проверкам: Проверяющий выбирает биты, которые он собирается прочесть, читает их и выносит решение, основываясь только на значениях прочитанных битов.

Расширение щели

Очевидное из определений включение: при $\varepsilon > \delta$ PCP алгоритм с параметрами $(1, \delta)$ является также и алгоритмом с параметрами $(1, \varepsilon)$, т.е.

$$\mathrm{PCP}_{1,\delta}(\log n, 1) \subseteq \mathrm{PCP}_{1,\varepsilon}(\log n, 1).$$

В обратную сторону

$$\forall \varepsilon > 0\ \forall \delta > 0 \quad \mathrm{PCP}_{1,1-\delta}(\log n, 1) \subseteq \mathrm{PCP}_{1,\varepsilon}(\log n, 1).$$

V повторяет k раз свой алгоритм, принимает доказательство, если все проверки положительные.

k-кратное повторение алгоритма с параметрами $(1, 1-\delta)$ даёт параметры $(1, (1-\delta)^k)$.

Для любых ε, δ при некотором k выполняется

$$(1-\delta)^k < \varepsilon.$$

Расширение щели

Очевидное из определений включение: при $\varepsilon > \delta$ PCP алгоритм с параметрами $(1,\delta)$ является также и алгоритмом с параметрами $(1,\varepsilon)$, т.е.

$$\mathrm{PCP}_{1,\delta}(\log n, 1) \subseteq \mathrm{PCP}_{1,\varepsilon}(\log n, 1).$$

В обратную сторону

$\forall \varepsilon > 0\ \forall \delta > 0 \quad \mathrm{PCP}_{1,1-\delta}(\log n, 1) \subseteq \mathrm{PCP}_{1,\varepsilon}(\log n, 1).$

V повторяет k раз свой алгоритм, принимает доказательство, если все проверки положительные.

k-кратное повторение алгоритма с параметрами $(1, 1-\delta)$ даёт параметры $(1, (1-\delta)^k)$.

Для любых ε, δ при некотором k выполняется

$$(1-\delta)^k < \varepsilon.$$

Расширение щели

Очевидное из определений включение: при $\varepsilon > \delta$ PCP алгоритм с параметрами $(1, \delta)$ является также и алгоритмом с параметрами $(1, \varepsilon)$, т.е.

$$\mathrm{PCP}_{1,\delta}(\log n, 1) \subseteq \mathrm{PCP}_{1,\varepsilon}(\log n, 1).$$

В обратную сторону

$\forall \varepsilon > 0 \ \forall \delta > 0 \quad \mathrm{PCP}_{1,1-\delta}(\log n, 1) \subseteq \mathrm{PCP}_{1,\varepsilon}(\log n, 1).$

V повторяет k раз свой алгоритм, принимает доказательство, если все проверки положительные.

k-кратное повторение алгоритма с параметрами $(1, 1-\delta)$ даёт параметры $(1, (1-\delta)^k)$.

Для любых ε, δ при некотором k выполняется

$$(1-\delta)^k < \varepsilon.$$

Расширение щели: комбинаторный вариант

Лемма

$\forall \varepsilon\ \forall t$ MAX-kCSP$_q(1, 1-\varepsilon) \leqslant_p$ MAX-(kt)CSP$_q(1, (1-\varepsilon)^t)$.

«Возведение графа в степень»

Рёбра графа $G(V, E, \Sigma, c)$ «возводим в степень t», получаем граф $G^t(V, E_t, \Sigma, c')$.

Ребро $\hat{e} \in E_t$ — это последовательность $(e_1, \ldots, e_t) \in V^{kt}$ рёбер графа G.

Ограничения: $c'_{\hat{e}}(\sigma') = \bigwedge_{i=1}^{t} c_{e_i}(\sigma)$ (должны выполняться ограничения исходного графа на каждом ребре последовательности).

Сводимость $G \mapsto G^t$ полиномиальная (количество рёбер возводится в степень t).

Осталось проверить корректность изменения щели.

Расширение щели: комбинаторный вариант

Лемма

$\forall\varepsilon\ \forall t$ MAX-kCSP$_q(1, 1-\varepsilon) \leqslant_p$ MAX-(kt)CSP$_q(1, (1-\varepsilon)^t)$.

«Возведение графа в степень»

Рёбра графа $G(V, E, \Sigma, c)$ «возводим в степень t», получаем граф $G^t(V, E_t, \Sigma, c')$.

Ребро $\hat{e} \in E_t$ — это последовательность $(e_1, \ldots, e_t) \in V^{kt}$ рёбер графа G.

Ограничения: $c'_{\hat{e}}(\sigma') = \bigwedge_{i=1}^{t} c_{e_i}(\sigma)$ (должны выполняться ограничения исходного графа на каждом ребре последовательности).

Сводимость $G \mapsto G^t$ полиномиальная (количество рёбер возводится в степень t).

Осталось проверить корректность изменения щели.

Расширение щели: комбинаторный вариант

Лемма

$\forall\varepsilon\ \forall t$ MAX-kCSP$_q(1, 1-\varepsilon) \leqslant_p$ MAX-(kt)CSP$_q(1, (1-\varepsilon)^t)$.

«Возведение графа в степень»

Рёбра графа $G(V, E, \Sigma, c)$ «возводим в степень t», получаем граф $G^t(V, E_t, \Sigma, c')$.
Ребро $\hat{e} \in E_t$ — это последовательность $(e_1, \ldots, e_t) \in V^{kt}$ рёбер графа G.
Ограничения: $c'_{\hat{e}}(\sigma') = \bigwedge_{i=1}^{t} c_{e_i}(\sigma)$ (должны выполняться ограничения исходного графа на каждом ребре последовательности).

Сводимость $G \mapsto G^t$ полиномиальная (количество рёбер возводится в степень t).
Осталось проверить корректность изменения щели.

Расширение щели: комбинаторный вариант

Лемма

$\forall \varepsilon\ \forall t$ MAX-kCSP$_q(1, 1-\varepsilon) \leqslant_p$ MAX-(kt)CSP$_q(1, (1-\varepsilon)^t)$.

«Возведение графа в степень»

Рёбра графа $G(V, E, \Sigma, c)$ «возводим в степень t», получаем граф $G^t(V, E_t, \Sigma, c')$.
Ребро $\hat{e} \in E_t$ — это последовательность $(e_1, \ldots, e_t) \in V^{kt}$ рёбер графа G.
Ограничения: $c'_{\hat{e}}(\sigma') = \bigwedge_{i=1}^{t} c_{e_i}(\sigma)$ (должны выполняться ограничения исходного графа на каждом ребре последовательности).

Сводимость $G \mapsto G^t$ полиномиальная (количество рёбер возводится в степень t).
Осталось проверить корректность изменения щели.

Расширение щели: корректность сводимости

Наблюдение

Если $\textsc{unsat}(G) = 0$, то $\textsc{unsat}(G') = 0$ (достигается на том же самом присваивании).

Утверждение

$\textsc{unsat}(G') = 1 - (1 - \textsc{unsat}(G))^t$.

Доказательство

Сравним долю выполненных ограничений в G и G^t на присваивании σ. В G^t ограничения выполнены в точности на тех гиперрёбрах (т.е. последовательностях рёбер исходного графа), которые состоят только из тех рёбер G, на которых присваивание σ выполняет ограничение. Поэтому

$$1 - \textsc{unsat}(G')(\sigma) = (1 - \textsc{unsat}(G)(\sigma))^t.$$

Расширение щели: корректность сводимости

Наблюдение

Если $\textsc{unsat}(G) = 0$, то $\textsc{unsat}(G') = 0$ (достигается на том же самом присваивании).

Утверждение

$\textsc{unsat}(G') = 1 - (1 - \textsc{unsat}(G))^t$.

Доказательство

Сравним долю выполненных ограничений в G и G^t на присваивании σ. В G^t ограничения выполнены в точности на тех гиперрёбрах (т.е. последовательностях рёбер исходного графа), которые состоят только из тех рёбер G, на которых присваивание σ выполняет ограничение. Поэтому

$$1 - \textsc{unsat}(G')(\sigma) = (1 - \textsc{unsat}(G)(\sigma))^t.$$

Расширение щели: корректность сводимости

Наблюдение

Если $\textsc{unsat}(G) = 0$, то $\textsc{unsat}(G') = 0$ (достигается на том же самом присваивании).

Утверждение

$\textsc{unsat}(G') = 1 - (1 - \textsc{unsat}(G))^t$.

Доказательство

Сравним долю выполненных ограничений в G и G^t на присваивании σ. В G^t ограничения выполнены в точности на тех гиперрёбрах (т.е. последовательностях рёбер исходного графа), которые состоят только из тех рёбер G, на которых присваивание σ выполняет ограничение. Поэтому

$$1 - \textsc{unsat}(G')(\sigma) = \left(1 - \textsc{unsat}(G)(\sigma)\right)^t.$$

Неаппроксимируемость 2-выполнимости

Теорема

Для любого $\varepsilon > 0$ существует такой размер размер алфавита k, что задача $\text{MAX-LC}_k(1, \varepsilon)$ является NP-трудной.

Для какого-то $\delta < 1$ задача $\text{MAX-LC}_7(1, \delta)$ NP-трудна.
Поэтому для доказательства теоремы достаточно построить сводимость

$$\text{MAX-LC}_k(1, \delta) \leqslant_p \text{MAX-LC}_{k^t}(1, \eta^t).$$

Граф повторений

Граф исходной задачи: $G(U, V, E, \Sigma, f)$ (граф с долями U, V).
Сводим к графу «повторений»: $G^t(U^t, V^t, E', \Sigma^t, \hat{f})$.
U^t, V^t: последовательности длины t.
$(u_1, \ldots, u_t)$ и $(v_1, \ldots, v_t)$ связаны ребром $\bar{e}$ в графе G^t, если для каждого i вершины u_i и v_i связаны ребром e_i в графе G.
Ограничения функциональные и задаются функциями

$$\hat{f}_{\bar{e}}(\sigma_1, \ldots, \sigma_t) = (f_{e_1}(\sigma_1), f_{e_2}(\sigma_2), \ldots, f_{e_t}(\sigma_t)).$$

Легко видеть, что если $\mathrm{UNSAT}(G) = 0$, то и $\mathrm{UNSAT}(G^t) = 0$. Если π выполняет все ограничения в графе G, то определим $\bar{\pi}$ как

$$\bar{\pi}(u_1, \ldots, u_t) = (\pi(u_1), \ldots, \pi(u_t)).$$

В обратную сторону (трудная часть):

Для любого $\delta > 0$ существуют такие $\eta > 0$, t_0 что если $\mathrm{UNSAT}(G) > \delta$, то $\mathrm{UNSAT}(G^t) > 1 - \eta^t$ при всех $t > t_0$.

Граф повторений

Граф исходной задачи: $G(U, V, E, \Sigma, f)$ (граф с долями U, V).
Сводим к графу «повторений»: $G^t(U^t, V^t, E', \Sigma^t, \hat{f})$.
U^t, V^t: последовательности длины t.
$(u_1, \dots, u_t)$ и $(v_1, \dots, v_t)$ связаны ребром $\bar{e}$ в графе G^t, если для каждого i вершины u_i и v_i связаны ребром e_i в графе G.
Ограничения функциональные и задаются функциями

$$\hat{f}_{\bar{e}}(\sigma_1, \dots, \sigma_t) = (f_{e_1}(\sigma_1), f_{e_2}(\sigma_2), \dots, f_{e_t}(\sigma_t)).$$

Легко видеть, что если $\text{UNSAT}(G) = 0$, то и $\text{UNSAT}(G^t) = 0$. Если π выполняет все ограничения в графе G, то определим $\bar{\pi}$ как

$$\bar{\pi}(u_1, \dots, u_t) = (\pi(u_1), \dots, \pi(u_t)).$$

В обратную сторону (трудная часть):

Для любого $\delta > 0$ существуют такие $\eta > 0$, t_0 что если $\text{UNSAT}(G) > \delta$, то $\text{UNSAT}(G^t) > 1 - \eta^t$ при всех $t > t_0$.

Граф повторений

Граф исходной задачи: $G(U, V, E, \Sigma, f)$ (граф с долями U, V).
Сводим к графу «повторений»: $G^t(U^t, V^t, E', \Sigma^t, \hat{f})$.
U^t, V^t: последовательности длины t.
$(u_1, \ldots, u_t)$ и $(v_1, \ldots, v_t)$ связаны ребром $\bar{e}$ в графе G^t, если для каждого i вершины u_i и v_i связаны ребром e_i в графе G.
Ограничения функциональные и задаются функциями

$$\hat{f}_{\bar{e}}(\sigma_1, \ldots, \sigma_t) = (f_{e_1}(\sigma_1), f_{e_2}(\sigma_2), \ldots, f_{e_t}(\sigma_t)).$$

Легко видеть, что если $\mathrm{UNSAT}(G) = 0$, то и $\mathrm{UNSAT}(G^t) = 0$. Если π выполняет все ограничения в графе G, то определим $\bar{\pi}$ как

$$\bar{\pi}(u_1, \ldots, u_t) = (\pi(u_1), \ldots, \pi(u_t)).$$

В обратную сторону (трудная часть):

Для любого $\delta > 0$ существуют такие $\eta > 0$, t_0 что если $\mathrm{UNSAT}(G) > \delta$, то $\mathrm{UNSAT}(G^t) > 1 - \eta^t$ при всех $t > t_0$.

Граф повторений

Граф исходной задачи: $G(U, V, E, \Sigma, f)$ (граф с долями U, V).
Сводим к графу «повторений»: $G^t(U^t, V^t, E', \Sigma^t, \hat{f})$.
U^t, V^t: последовательности длины t.
$(u_1, \ldots, u_t)$ и $(v_1, \ldots, v_t)$ связаны ребром $\bar{e}$ в графе G^t, если для каждого i вершины u_i и v_i связаны ребром e_i в графе G.
Ограничения функциональные и задаются функциями

$$\hat{f}_{\bar{e}}(\sigma_1, \ldots, \sigma_t) = (f_{e_1}(\sigma_1), f_{e_2}(\sigma_2), \ldots, f_{e_t}(\sigma_t)).$$

Легко видеть, что если $\mathrm{UNSAT}(G) = 0$, то и $\mathrm{UNSAT}(G^t) = 0$. Если π выполняет все ограничения в графе G, то определим $\bar{\pi}$ как

$$\bar{\pi}(u_1, \ldots, u_t) = (\pi(u_1), \ldots, \pi(u_t)).$$

В обратную сторону (трудная часть):

Для любого $\delta > 0$ существуют такие $\eta > 0$, t_0 что если $\mathrm{UNSAT}(G) > \delta$, то $\mathrm{UNSAT}(G^t) > 1 - \eta^t$ при всех $t > t_0$.

Граф повторений

Граф исходной задачи: $G(U, V, E, \Sigma, f)$ (граф с долями U, V).
Сводим к графу «повторений»: $G^t(U^t, V^t, E', \Sigma^t, \hat{f})$.
U^t, V^t: последовательности длины t.
$(u_1, \dots, u_t)$ и $(v_1, \dots, v_t)$ связаны ребром $\bar{e}$ в графе G^t, если для каждого i вершины u_i и v_i связаны ребром e_i в графе G.
Ограничения функциональные и задаются функциями

$$\hat{f}_{\bar{e}}(\sigma_1, \dots, \sigma_t) = (f_{e_1}(\sigma_1), f_{e_2}(\sigma_2), \dots, f_{e_t}(\sigma_t)).$$

Легко видеть, что если $\text{UNSAT}(G) = 0$, то и $\text{UNSAT}(G^t) = 0$. Если π выполняет все ограничения в графе G, то определим $\bar{\pi}$ как

$$\bar{\pi}(u_1, \dots, u_t) = (\pi(u_1), \dots, \pi(u_t)).$$

В обратную сторону (трудная часть):

Для любого $\delta > 0$ существуют такие $\eta > 0$, t_0 что если $\text{UNSAT}(G) > \delta$, то $\text{UNSAT}(G^t) > 1 - \eta^t$ при всех $t > t_0$.

Игровая формулировка

Игра в допрос

Три игрока: два Доказывающих (provers) и один Проверяющий (verifier).
Проверяющий допрашивает Доказывающих порознь. Он удовлетворён, если ответы согласованы.
Формально: игра $G(U, V, \Gamma, \mathrm{Win})$ — это вероятностное распределение G на рёбрах (взвешенный граф), алфавит Γ, и предикат выигрыша

$$\mathrm{Win} \subseteq U \times \Gamma \times V \times \Gamma.$$

(выделяет четвёрки «вопрос 1му, ответ 1го; вопрос 2му, ответ 2го», которые устраивают Проверяющего).

Цена игры и задача 2-выполнимости

Стратегии Доказывающих — это отображения $x\colon V \to \Gamma$, $y\colon U \to \Gamma$.
Цена игры для выбранных стратегий — это вероятность выигрыша

$$\mathrm{val}(G; x, y) = \Pr_G[\mathrm{Win}(u, x(u), v, y(v))].$$

Цена игры $\mathrm{val}(G)$ — это максимум по стратегиям Доказывающих вероятности выигрыша.

Задача 2-выполнимости и игра в допрос

Вычисление цены игры — частный случай взвешенной задачи $\mathrm{MAX\text{-}2CSP}_q$ на двудольном графе.
Обратно, по двудольному графу ограничений определим распределение на $U \times V$ так: выбираем случайное ребро, один из его концов — вопрос к первому, второй — вопрос ко второму. Предикат Win для данной пары u, v совпадает с ограничением $c_{uv} \subseteq \Gamma^2$ на выбранном ребре графа G.

Цена игры и задача 2-выполнимости

Стратегии Доказывающих — это отображения $x\colon V \to \Gamma$, $y\colon U \to \Gamma$.
Цена игры для выбранных стратегий — это вероятность выигрыша

$$\mathrm{val}(G; x, y) = \Pr_G[\mathrm{Win}(u, x(u), v, y(v))].$$

Цена игры $\mathrm{val}(G)$ — это максимум по стратегиям Доказывающих вероятности выигрыша.

Задача 2-выполнимости и игра в допрос

Вычисление цены игры — частный случай взвешенной задачи MAX-2CSP_q на двудольном графе.
Обратно, по двудольному графу ограничений определим распределение на $U \times V$ так: выбираем случайное ребро, один из его концов — вопрос к первому, второй — вопрос ко второму. Предикат Win для данной пары u, v совпадает с ограничением $c_{uv} \subseteq \Gamma^2$ на выбранном ребре графа G.

Параллельные игры

Параллельная игра (repetition game)

Параллельная игра G^t:
«вопросы» — это последовательности $(u_1, \ldots, u_t)$, $(v_1, \ldots, v_t)$ вопросов в игре G,
«ответы» — последовательности ответов $(\alpha_1, \ldots, \alpha_t)$, $(\beta_1, \ldots, \beta_t)$.
Распределение G^t состоит в независимом выборе для каждого i пары (u_i, v_i) по распределению G.
Результат игры определяется предикатом

$$\mathrm{Win}^t = \bigwedge_i \mathrm{Win}(u_i, \alpha_i, v_i, \beta_i)$$

(Доказывающим нужно выиграть во всех копиях игры).

Цена параллельной игры

Вопрос

Верно ли, что $\mathrm{val}(G^t) = \mathrm{val}(G)^t$?

Ответ

Нет!

Цена параллельной игры

Вопрос

Верно ли, что $\mathrm{val}(G^t) = \mathrm{val}(G)^t$?

Ответ

Нет!

Пример Fortnow, Feige

Игра «Угадай вопрос к товарищу»

Проверяющий выбирает два бита u, v случайно и независимо. Бит u посылается первому Доказывающему, бит v — второму.

Алфавит ответов $\{1,2\} \times \{0,1\}$. Доказывающие выигрывают, если их ответы совпали $\alpha = \beta = (i, \sigma)$, причём Доказывающий с номером i получил вопрос σ.

Цена игры $\leqslant 1/2$: при любых стратегиях один из Доказывающих должен предъявить неизвестный ему вопрос второго.

В игре с двумя копиями есть стратегия, которая имеет цену $1/2$. Первый доказывающий на паре вопросов u_1, u_2 даёт ответ $(1, u_1), (2, u_1)$ (пара ответов в исходной игре), а второй на паре вопросов v_1, v_2 даёт ответ $(1, v_2), (2, v_2)$.

Если (вероятность этого), Доказывающие выигрывают

Пример Fortnow, Feige

Игра «Угадай вопрос к товарищу»

Проверяющий выбирает два бита u, v случайно и независимо. Бит u посылается первому Доказывающему, бит v — второму.

Алфавит ответов $\{1, 2\} \times \{0, 1\}$. Доказывающие выигрывают, если их ответы совпали $\alpha = \beta = (i, \sigma)$, причём Доказывающий с номером i получил вопрос σ.

Цена игры $\leqslant 1/2$: при любых стратегиях один из Доказывающих должен предъявить неизвестный ему вопрос второго.

В игре с двумя копиями есть стратегия, которая имеет цену $1/2$. Первый доказывающий на паре вопросов u_1, u_2 даёт ответ $(1, u_1), (2, u_1)$ (пара ответов в исходной игре), а второй на паре вопросов v_1, v_2 даёт ответ $(1, v_2), (2, v_2)$.

Если $u_1 = v_2$ (вероятность этого $1/2$), Доказывающие выигрывают.

Пример Fortnow, Feige

Игра «Угадай вопрос к товарищу»

Проверяющий выбирает два бита u, v случайно и независимо. Бит u посылается первому Доказывающему, бит v — второму.
Алфавит ответов $\{1,2\} \times \{0,1\}$. Доказывающие выигрывают, если их ответы совпали $\alpha = \beta = (i, \sigma)$, причём Доказывающий с номером i получил вопрос σ.
Цена игры $\leqslant 1/2$: при любых стратегиях один из Доказывающих должен предъявить неизвестный ему вопрос второго.
В игре с двумя копиями есть стратегия, которая имеет цену $1/2$.
Первый доказывающий на паре вопросов u_1, u_2 даёт ответ $(1, u_1), (2, u_1)$ (пара ответов в исходной игре), а второй на паре вопросов v_1, v_2 даёт ответ $(1, v_2), (2, v_2)$.
Если $u_1 = v_2$ (вероятность этого $1/2$), Доказывающие выигрывают.

Пример Fortnow, Feige

Игра «Угадай вопрос к товарищу»

Проверяющий выбирает два бита u, v случайно и независимо. Бит u посылается первому Доказывающему, бит v — второму.
Алфавит ответов $\{1,2\} \times \{0,1\}$. Доказывающие выигрывают, если их ответы совпали $\alpha = \beta = (i, \sigma)$, причём Доказывающий с номером i получил вопрос σ.
Цена игры $\leqslant 1/2$: при любых стратегиях один из Доказывающих должен предъявить неизвестный ему вопрос второго.
В игре с двумя копиями есть стратегия, которая имеет цену $1/2$.
Первый доказывающий на паре вопросов u_1, u_2 даёт ответ $(1, u_1), (2, u_1)$ (пара ответов в исходной игре), а второй на паре вопросов v_1, v_2 даёт ответ $(1, v_2), (2, v_2)$.
Если $u_1 = v_2$ (вероятность этого $1/2$), Доказывающие выигрывают.

Пример Fortnow, Feige

Игра «Угадай вопрос к товарищу»

Проверяющий выбирает два бита u, v случайно и независимо. Бит u посылается первому Доказывающему, бит v — второму.

Алфавит ответов $\{1,2\} \times \{0,1\}$. Доказывающие выигрывают, если их ответы совпали $\alpha = \beta = (i, \sigma)$, причём Доказывающий с номером i получил вопрос σ.

Цена игры $\leqslant 1/2$: при любых стратегиях один из Доказывающих должен предъявить неизвестный ему вопрос второго.

В игре с двумя копиями есть стратегия, которая имеет цену $1/2$.

Первый доказывающий на паре вопросов u_1, u_2 даёт ответ $(1, u_1), (2, u_1)$ (пара ответов в исходной игре), а второй на паре вопросов v_1, v_2 даёт ответ $(1, v_2), (2, v_2)$.

Если $u_1 = v_2$ (вероятность этого $1/2$), Доказывающие выигрывают.

Теорема о повторении (repetition theorem)

Теорема

Для любой константы $\delta > 0$ из $\mathrm{val}(G) < 1 - \delta$ следует $\mathrm{val}(G^t) = 2^{-\Omega(t)}$.

История:

1. Доказана Рацем (Ran Raz) в 1995. Очень сложное и очень длинное доказательство.
2. Более понятное и чуть менее сложное доказательство предложено Холенстейном (Thomas Holenstein) в 2007.
3. Браверман и Барг (Mark Braverman, Ankit Barg) в 2014 предложили доказательство, которое основано на теоретико-информационных оценках. Они считают, что так проще.
4. Есть множество работ на тему параллельных игр, целая область науки возникла. Уточняют вид оценок, переносят результаты на квантовые игры и т.п.

Теорема о повторении (repetition theorem)

Теорема

Для любой константы $\delta > 0$ из $\mathrm{val}(G) < 1 - \delta$ следует $\mathrm{val}(G^t) = 2^{-\Omega(t)}$.

История:

1. Доказана Рацем (Ran Raz) в 1995. Очень сложное и очень длинное доказательство.
2. Более понятное и чуть менее сложное доказательство предложено Холенстейном (Thomas Holenstein) в 2007.
3. Браверман и Барг (Mark Braverman, Ankit Barg) в 2014 предложили доказательство, которое основано на теоретико-информационных оценках. Они считают, что так проще.
4. Есть множество работ на тему параллельных игр, целая область науки возникла. Уточняют вид оценок, переносят результаты на квантовые игры и т.п.

Теорема о повторении (repetition theorem)

Теорема

Для любой константы $\delta > 0$ из $\mathrm{val}(G) < 1 - \delta$ следует $\mathrm{val}(G^t) = 2^{-\Omega(t)}$.

История:

1. Доказана Рацем (Ran Raz) в 1995. Очень сложное и очень длинное доказательство.
2. Более понятное и чуть менее сложное доказательство предложено Холенстейном (Thomas Holenstein) в 2007.
3. Браверман и Барг (Mark Braverman, Ankit Barg) в 2014 предложили доказательство, которое основано на теоретико-информационных оценках. Они считают, что так проще.
4. Есть множество работ на тему параллельных игр, целая область науки возникла. Уточняют вид оценок, переносят результаты на квантовые игры и т.п.

Теорема о повторении (repetition theorem)

Теорема

Для любой константы $\delta > 0$ из $\mathrm{val}(G) < 1 - \delta$ следует $\mathrm{val}(G^t) = 2^{-\Omega(t)}$.

История:

1. Доказана Рацем (Ran Raz) в 1995. Очень сложное и очень длинное доказательство.
2. Более понятное и чуть менее сложное доказательство предложено Холенстейном (Thomas Holenstein) в 2007.
3. Браверман и Барг (Mark Braverman, Ankit Barg) в 2014 предложили доказательство, которое основано на теоретико-информационных оценках. Они считают, что так проще.
4. Есть множество работ на тему параллельных игр, целая область науки возникла. Уточняют вид оценок, переносят результаты на квантовые игры и т.п.

Теорема о повторении (repetition theorem)

Теорема

Для любой константы $\delta > 0$ из $\mathrm{val}(G) < 1 - \delta$ следует $\mathrm{val}(G^t) = 2^{-\Omega(t)}$.

История:

1. Доказана Рацем (Ran Raz) в 1995. Очень сложное и очень длинное доказательство.
2. Более понятное и чуть менее сложное доказательство предложено Холенстейном (Thomas Holenstein) в 2007.
3. Браверман и Барг (Mark Braverman, Ankit Barg) в 2014 предложили доказательство, которое основано на теоретико-информационных оценках. Они считают, что так проще.
4. Есть множество работ на тему параллельных игр, целая область науки возникла. Уточняют вид оценок, переносят результаты на квантовые игры и т.п.

Локальное тестирование

Для сводимостей, сохраняющих щели, важна идея смотреть в малое количество мест, которые выбираются случайно.

Ограничения в задаче 2-выполнимости

Как проверить $(\sigma_1, \sigma_2) \in c_e \subseteq \Sigma^2$, прочитав «дробную часть символов»?

Локальное тестирование

Для сводимостей, сохраняющих щели, важна идея смотреть в малое количество мест, которые выбираются случайно.

Ограничения в задаче 2-выполнимости

Как проверить $(\sigma_1, \sigma_2) \in c_e \subseteq \Sigma^2$, прочитав «дробную часть символов»?

Длинный код

Пусть $|\Sigma| = k$, без ограничения общности $\Sigma = [k]$.
Длинный код: $\Sigma \to \{0,1\}^{2^k}$. Сопоставляет символу k таблицу значений диктатора (проекции), то есть булевой функции от k переменных:

$$k \mapsto f(x_1, \ldots, x_k), \qquad \forall x \; f(x_1, \ldots, x_k) = x_k.$$

Пример

Пусть $k = 3$, упорядочение аргументов функции лексикографическое. Тогда длинные коды таковы:

$$\begin{aligned} 1 &\mapsto \quad (0,0,0,0,1,1,1,1), \\ 2 &\mapsto \quad (0,0,1,1,0,0,1,1), \\ 3 &\mapsto \quad (0,1,0,1,0,1,0,1). \end{aligned}$$

Длинный код

Пусть $|\Sigma| = k$, без ограничения общности $\Sigma = [k]$.
Длинный код: $\Sigma \to \{0,1\}^{2^k}$. Сопоставляет символу k таблицу значений диктатора (проекции), то есть булевой функции от k переменных:

$$k \mapsto f(x_1, \ldots, x_k), \qquad \forall x \; f(x_1, \ldots, x_k) = x_k.$$

Пример

Пусть $k = 3$, упорядочение аргументов функции лексикографическое. Тогда длинные коды таковы:

$$\begin{aligned} 1 &\mapsto (0,0,0,0,1,1,1,1), \\ 2 &\mapsto (0,0,1,1,0,0,1,1), \\ 3 &\mapsto (0,1,0,1,0,1,0,1). \end{aligned}$$

Локальная проверка ограничения (неформальная идея)

Кодируем символы длинным кодом и получаем такую задачу.

1. Есть две битовые строки T_1, T_2 длины 2^k. Хотим запросить как можно меньше битов из них и ответить на вопрос «выполняется ли ограничение?»
2. Требования такие: если строки близки к длинным кодам $T(\sigma_1)$, $T(\sigma_2)$, то ответ обязан совпадать с ответом на вопрос «выполнено ли ограничение на паре (σ_1, σ_2)?»
3. Близость измеряется относительным расстоянием Хэмминга, то есть долей различающихся позиций.
4. Получается сводимость к некоторой задаче, которая определяется видом запросов.
5. Чтобы использовать эту идею в доказательствах, нужны преобразования Фурье на булевом кубе.

Локальная проверка ограничения (неформальная идея)

Кодируем символы длинным кодом и получаем такую задачу.

1. Есть две битовые строки T_1, T_2 длины 2^k. Хотим запросить как можно меньше битов из них и ответить на вопрос «выполняется ли ограничение?»
2. Требования такие: если строки близки к длинным кодам $T(\sigma_1)$, $T(\sigma_2)$, то ответ обязан совпадать с ответом на вопрос «выполнено ли ограничение на паре (σ_1, σ_2)?»
3. Близость измеряется относительным расстоянием Хэмминга, то есть долей различающихся позиций.
4. Получается сводимость к некоторой задаче, которая определяется видом запросов.
5. Чтобы использовать эту идею в доказательствах, нужны преобразования Фурье на булевом кубе.

Локальная проверка ограничения (неформальная идея)

Кодируем символы длинным кодом и получаем такую задачу.

1. Есть две битовые строки T_1, T_2 длины 2^k. Хотим запросить как можно меньше битов из них и ответить на вопрос «выполняется ли ограничение?»
2. Требования такие: если строки близки к длинным кодам $T(\sigma_1)$, $T(\sigma_2)$, то ответ обязан совпадать с ответом на вопрос «выполнено ли ограничение на паре (σ_1, σ_2)?»
3. Близость измеряется относительным расстоянием Хэмминга, то есть долей различающихся позиций.
4. Получается сводимость к некоторой задаче, которая определяется видом запросов.
5. Чтобы использовать эту идею в доказательствах, нужны преобразования Фурье на булевом кубе.

Локальная проверка ограничения (неформальная идея)

Кодируем символы длинным кодом и получаем такую задачу.

1. Есть две битовые строки T_1, T_2 длины 2^k. Хотим запросить как можно меньше битов из них и ответить на вопрос «выполняется ли ограничение?»
2. Требования такие: если строки близки к длинным кодам $T(\sigma_1)$, $T(\sigma_2)$, то ответ обязан совпадать с ответом на вопрос «выполнено ли ограничение на паре (σ_1, σ_2)?»
3. Близость измеряется относительным расстоянием Хэмминга, то есть долей различающихся позиций.
4. Получается сводимость к некоторой задаче, которая определяется видом запросов.
5. Чтобы использовать эту идею в доказательствах, нужны преобразования Фурье на булевом кубе.

Локальная проверка ограничения (неформальная идея)

Кодируем символы длинным кодом и получаем такую задачу.

1. Есть две битовые строки T_1, T_2 длины 2^k. Хотим запросить как можно меньше битов из них и ответить на вопрос «выполняется ли ограничение?»
2. Требования такие: если строки близки к длинным кодам $T(\sigma_1)$, $T(\sigma_2)$, то ответ обязан совпадать с ответом на вопрос «выполнено ли ограничение на паре (σ_1, σ_2)?»
3. Близость измеряется относительным расстоянием Хэмминга, то есть долей различающихся позиций.
4. Получается сводимость к некоторой задаче, которая определяется видом запросов.
5. Чтобы использовать эту идею в доказательствах, нужны преобразования Фурье на булевом кубе.