NP-полнота алгоритмических игр

Sameer Tantry

December 2022

В данном проекте мы формализуем некоторые классические алгоритмические игры и приведем доказательства их **NP**-полноты.

1 Основные определения

Определение 1. Класс **NP**. Язык $A \in \mathbf{NP}$, если $\exists V(x, s)$, работающая за $poly(|x|)$: $\forall x \in A \iff \exists s : V(x, s) = 1$.

Определение 2. Пусть A и B — два языка. Тогда A сводится по Карпу к B, если существует всюду определенная функция $f : \{0,1\}^* \to \{0,1\}^*$, вычислимая за полиномиальное время, такая что $x \in A \iff f(x) \in B$. Обозначение: $A \leq_p B$. Рассмотрим некоторые простые свойства такой сводимости:

Утверждение. Имеют место следующие факты:

- Сводимость по Карпу рефлексивна: $A \leq_p A$;
- Сводимость по Карпу транзитивна: $(A \leq_p B \wedge B \leq_p C) \to (A \leq_p C)$;
- Если $B \in \mathbf{NP}$ и $A \leq_p B$, то $A \in \mathbf{NP}$.

Определение 3. **NP**-полнота. Язык A **NP**-труден, если
$\forall B \in \mathbf{NP} \hookrightarrow B \leq_p A$.
Язык A **NP**-полон, если A **NP**-труден и $A \in \mathbf{NP}$.

Доказательство **NP**-полноты по определению часто является крайне трудной задачей, поэтому в большинстве случаев в доказательстве используется сведение другой **NP**-полной задачи к данной. Стоит отметить, что мы говорим про **NP**-полноту задач, но определение **NP**-полноты ввели для языков.

Замечание. Будем называть задачу **NP**-полной, если язык, состоящий из слов, определенным образом кодирующих входные данные нашей задачи, для которых существует решение задачи, является **NP**-полным.

Рассмотрим некоторые наиболее популярные **NP**-полные задачи:

- KNAPSACK $= \{(n_1, ..., n_k, m_1, ..., m_k, N, M) \mid \exists \alpha \in \{0,1\}^k \ \sum_{i=1}^k \alpha_i n_i \leq N$ и $\sum_{i=1}^k \alpha_i m_i \geq M\}$;
- SAT $= \{\varphi \mid \varphi$ — выполнимая булева формула$\}$;
- CLIQUE $= \{(G, k) \mid$ в графе G есть клика из k вершин$\}$;
- INDSET $= \{(G, k) \mid$ в графе G есть вершинное покрытие из k вершин$\}$;
- VERTEXCOVER $= \{$в графе G есть вершинное покрытие из k вершин$\}$;
- DHAMPATH $= \{(G, s, t) \mid$ в ориентированном графе G есть ориентированный путь из s в t, проходящий ровно один раз через каждую вершину$\}$;
- TSP $= \{(G, w, s, t, l) \mid$ во взвешенном графе (G, w) есть путь коммивожёра из s в t длины не больше l$\}$;
- 3COL $= \{G \mid$ вершины графа можно правильно раскрасить в 3 цвета$\}$;
- INTPROG $= \{(A, b) \in \mathbb{Z}^{n \times m} \times \mathbb{Z}^n \mid \exists x \in \mathbb{N}^m \ Ax = b\}$;
- THEOREMS $= \{(\varphi, 1^n) \mid$ у арифметической формулы φ есть доказательство в аксиоматике Пеано длины не больше $n\}$.

2 NP-полные игры

NP-полные задачи считаются наиболее сложными задачами среди задач из **NP**. Для них, вероятно, не будет найдено быстрого решения дающего точный, а не приближенный ответ. Необходимо упомянуть гипотезу **P** = **NP**? Задача заключается в установлении равенства или неравенства между этими двумя классами. Данная проблема является одной из семи задач тысячелетия, за решение которой Математический институт Клэя назначил премию в миллион долларов.

2.1 Пятнашки

Пятнашки, или игра в 15, такен — популярная головоломка, придуманная в 1878 году Ноем Чепмэном. Головоломка представляет собой набор из 15 одинаковых квадратных костяшек с нанесёнными на них числами, лежащих в квадратной коробке. Длина стороны коробки в четыре раза больше длины стороны костяшки, поэтому в коробке остаётся незаполненным одно квадратное поле. Цель игры — упорядочить костяшки по возрастанию номеров, перемещая их внутри коробки, желательно сделав как можно меньше перемещений.

Рассмотрим задачу нахождения оптимального решения и покажем ее **NP**-полноту. Эта задача является частым примером использования алгоритмов с приближенным решением и эвристических функций и предлагается в некоторых алгоритмических курсах.
В данной работе мы лишь рассмотрим идею доказательства **NP**-полноты, так как полное доказательство довольно сложное. Для этого попробуем формализовать игру:

Моделировать игру мы будем в общем случае.
Определение 1. Доска — неориентированный граф $G(V, E)$, где $V = \{1,\ ...,\ n^2\}$ и ребра определим следующим образом:
$V_1 = \{1,\ 2,\ ...,\ n\}$, $V_2 = \{1,\ n+1,\ ...,\ n(n-1)+1\}$, $V_3 = \{n,\ 2n,\ ...,\ n^2\}$, $V_4 = \{n(n-1)+1,\ n(n-1)+2,\ ...,\ n^2\}$.
Тогда

$$\forall v \in V \setminus V_1 \hookrightarrow \{v,\ v-n\} \in E,$$

$$\forall v \in V \setminus V_2 \hookrightarrow \{v,\ v-1\} \in E,$$

$$\forall v \in V \setminus V_3 \hookrightarrow \{v,\ v+1\} \in E,$$

$$\forall v \in V \setminus V_4 \hookrightarrow \{v,\ v+n\} \in E.$$

Определение 2. Конфигурация — биекция $F : V \to \{0, ..., |V|-1\}$, где для пустой клетки $v \hookrightarrow F(v) = 0$
Конфигурацию назовем правильной, если в каждой вершине графа кроме одной находится уникальное число от 1 до $|V|-1$. Иначе говоря,

$\exists! v \in V : F(v) = 0, \forall u, w \in V \setminus \{v\} \hookrightarrow F(u), F(w) \in \{1, ..., |V| - 1\}$, причем $F(u) \neq F(w)$.
$u, v \in V$. Определим ход игры как переход от конфигурации F в конфигурацию F', где
$F(u) \neq 0 \wedge F(v) = 0 \wedge F'(u) = 0 \wedge F'(v) \neq 0 \wedge \forall w \in V \setminus \{u, v\} \hookrightarrow F(w) = F'(w)$
Финальную конфигурацию определим следующим образом:
$F_t : \forall n \in \{1, ..., |V| - 1\} \hookrightarrow F_t(n) = n, F_t(16) = 0$
Постановка задачи. Дан граф $G(V, E)$, правильные конфигурации F_s, F_t и целое число k. Существует ли последовательность из $\leq k$ правильных ходов, которая переводит конфигурацию F_s в F_t?

Теорема. Данная задача является **NP**-полной.
Доказательство:

Докажем, что данная задача лежит в классе **NP**.
Заметим, что в качестве сертификата можно предложить саму последовательность конфигураций $F_s, F_1, F_2, ..., F_{m-1}, F_t$. Тогда проверка правильности каждой конфигурации будет полиномиальной. Значит, задача принадлежит классу **NP**.

В доказательстве полноты сведем некоторую **NP**-полную задачу к данной. Рассмотрим задачу $3XC$ (задача о точном покрытии 3-множествами).

Задача. Дано множество $U = \{e_i\}_{i=1...3n}$ и семейство множеств $S = \{s_j\}_{j=1...m}$, где $\forall j \in \{1, ..., m\} \hookrightarrow |s_j| = 3, s_j \subset U$. Существует ли подмножество $S' \subset S$, все множества которого покрывают U таким образом, что каждое e_i лежит ровно в одном s_j?

Ричардом Карпом было доказано, что эта задача является **NP**-полной [4]. Сводимость приведена в статье [2].

Решения, которые применяются на практике:

Можно рассмотреть граф игры и искать в нем кратчайший путь. Для поиска кратчайшего пути часто используется алгоритм A^* [5]. Данный алгоритм является эвристической версией алгоритма поиска в ширину на графе. Вершины для обхода выбираются в соответствии с эвристикой — предполагаемым количеством ходов до цели. Эвристики:

- Манхэттенское расстояние (Manhattan distance)

  Манхэттенское расстояние — это сумма расстояний по строкам и столбцам от текущего расположения костяшки до ее правильной позиции. Пример: на Рис. 1 манхэттенское расстояние равно 4.

- Линейный конфликт (Linear conflict)

  Считается, что костяшка I и костяшка J находятся в линейном конфликте по строке, если они обе стоят в своей строке, но костяшка I находится левее костяшки J, хотя на самом деле должна быть справа. Пример: На Рис. 2 I = 15, J = 13.

  Мы должны подвинуть одну из костяшек со строки, поэтому можем добавить 2 к манхэттенскому расстоянию. Аналогичным образом рассматривается линейный конфликт по столбцу.

- Угловые фишки (Corner tiles)

  Рассмотрим правый верхний угол на Рис. 3. Пусть «3» или «8» стоит на своей позиции, а на месте «4» — любая другая костяшка. В таком случае, чтобы поставить «4» на место, костяшки придется подвинуть. Для этого добавим 2 к манхэттенскому расстоянию. Если «3» или «8» участвует в линейном конфликте (linear conflict), то двойка уже добавлена.

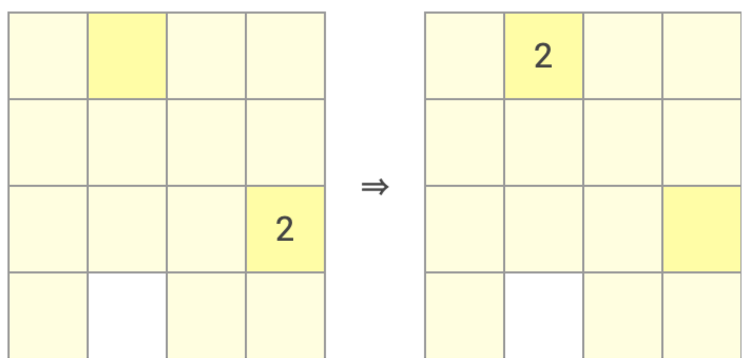

Рис. 1: Манхэттенское расстояние

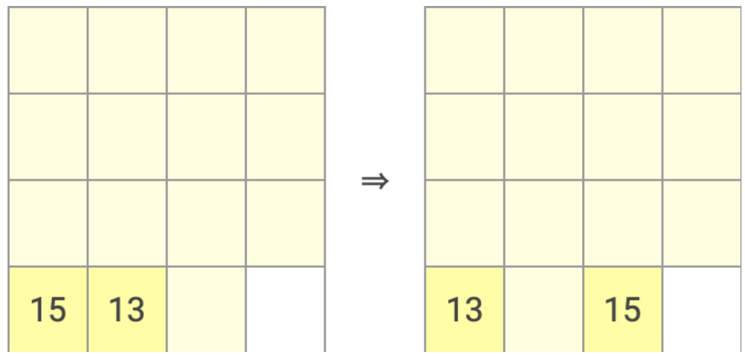

Рис. 2: Линейный конфликт

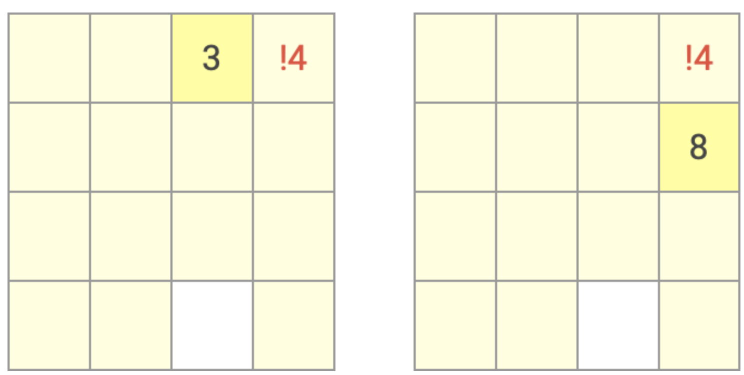

Рис. 3: Угловые фишки

Список литературы

[1] Daniil Musatov: Compl-book

[2] Oded Goldreich: Finding the Shortest Move-Sequence in the Graph-Generalized 15-Puzzle is **NP**-Hard

[3] Solving Sliding Tiles

[4] Richard M. Karp: Reducibility Among Combinatorial Problems

[5] A-star