Доказательство NP-полноты для некоторых задач Карпа.

Хритова Екатерина Андреевна
510 группа

Содержание

1 Введение. **2**

2 Основные определения. **3**
2.1 Класс сложности P. 3
2.2 Класс сложности NP. 3

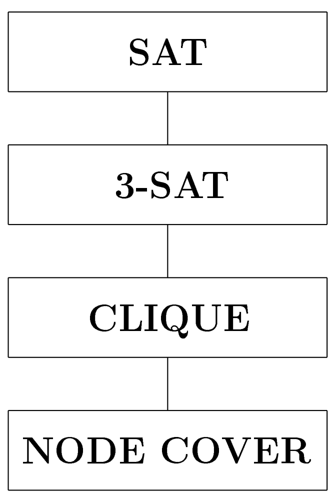
3 Формулировка рассматриваемых задач. **4**
3.1 Формулировака задачи SAT. 4
3.2 Формулировка задачи 3-SAT. 5
3.3 Формулировка CLIQUE. 5
3.4 Формулировка NODE COVER 5

4 Доказательство NP-полноты рассматриваемых задач. **6**
4.1 Сведение SAT к 3-SAT. 6
4.2 Сведение 3-SAT к CLIQUE. 7
4.3 Сведение CLIQUE к NODE COVER. 10

1 Введение.

В этой работе представлены доказательства NP-полноты задачи о выполнимости формул в 3-конъюнктивной нормальной форме, задачи о клике и задачи о вершинном покрытии графа.

В введении приводятся основные определения, а также формулируется лемма о сведении, с помощью которой будет доказываться NP-полнота задач. Далее приводится формулировка каждой задачи. Далее представлено доказательство NP-полноты описанных задач с помощью сведения их к NP-полным задачам. Сведение происходит по следующей схеме:



2 Основные определения.

2.1 Класс сложности P.

Алфавит Σ – это коечное множество символов. **Язык** L, определенный над множеством Σ – это произвольное множество строк, состоящих из символов Σ. Язык всех строк, заданных над множеством Σ, обозначается Σ^*. Любой язык над множеством Σ являетяс подмножеством Σ^*.

Алгоритм A **принимает** строку $x \in \{0,1\}^*$, если для заданных входных данных x выход алгоритма $A(x)$ равен 1. Язык, **принимаемый алгоритмом** A представляет собой множество строк, принимаемых алгоритмом A:

$$L := \{x \in \{0,1\}^* : A(x) = 1\}$$

Язык L **принимается** алгоритмом A **за полиномиальное время**, если он принимается алгоритмом A и, кроме того, существует константа k, такая что для любой n-символьной стоки $x \in L$ алгоритм A принимает строку x за $O(n^k)$. Язык L **распознается** алгоритмом A **за полиномиальное время**, если существует константа k, такая что для любой n-символьной стоки $x \in \{0,1\}^*$ алгоритм за $O(n^k)$ правильно выясняет, принадлежит ли x языку L.

Класс P – это множество языков $L \subseteq \{0,1\}^*$, для которых существует алгоритм A, распознающий язык L за полиномиальное время.

2.2 Класс сложности NP.

Алгоритм верификации A – это алгоритм с двумя аргументами: входная строка x и бинарная строка y, называемая **сертификатом**. Алгоритм A **верифицирует** входную строку x, если существует сертификат y, т.ч. $A(x,y) = 1$. **Язык, верифицируемый алгорит-**

мом A представляет собой множество

$$L = \{x \in \{0,1\}^* : \exists y \in \{0,1\}^*, \text{т.ч. } A(x,y) = 1\}$$

Класс NP – это класс языков, которые можно верифицировать с помощью алгоритма с полиномиальным временем работы.

Будем говорить, что язык L_1 **приводим за полиномиальное время** к языку L_2 ($L_1 \propto L_2$), если существует вычислимая за полиномиальное время функция $f : L_1 \to L_2$, т.ч. $\forall x \in \{0,1\}^*$ выполнено $x \in L_i \Leftrightarrow f(x) \in L_2$.

Язык $L \subseteq \{0,1\}^*$ называется **NP-полным**, если $L \in$NP и $L' \propto L \; \forall L' \in$NP.

Ключевым в данной работе является следующее утверждение.

Лемма	Если $L \in$NP и $L' \propto L$ для некоторого $L' \in$NP, то $L \in$NP

3 Формулировка рассматриваемых задач.

3.1 Формулировака задачи SAT.

Экземпляром задачи является булева формула, состоящая только из имён переменных, скобок и операций $\wedge$ (И), $\vee$ (ИЛИ) и $\neg$ (НЕ). Задача заключается в следующем: можно ли назначить всем переменным, встречающимся в формуле, значения ложь и истина так, чтобы формула стала истинной. Задача SAT для булевых формул, записанных в конъюнктивной нормальной форме, является NP-полной.

3.2 Формулировка задачи 3-SAT.

Литерал – это переменная x или ее отрицание $\bar{x}$.
Клоз – это формула вида $l_1 \vee l_2 \vee ... \vee l_m$, где $l_1, l_2, ..., l_m$ - литералы.

Вход: Клозы $C_1, C_2, ..., C_r$, каждый из которых состоит не более чем из трех литералов из множества $\{x_1, x_2, ..., x_m\} \cup \{\bar{x_1}, \bar{x_2}, ..., \bar{x_m}\}$.
Задача: Можно ли назначить всем переменным, встречающимся в формуле, значения ложь и истина так, чтобы формула $C_1 \wedge C_2 \wedge ... \wedge C_r$ стала истинной.

3.3 Формулировка CLIQUE.

Кликой в неориентированом графе называется подмножество вершин, каждые две из которых соединены ребром. Размер клики определяется количеством вершин в ней.

Вход: Граф G, целое положительное число k.
Задача: Существует ли в заданном графе G клика размера k.

3.4 Формулировка NODE COVER

Вершинным покрытием U графа $G = (V, E)$ называется подмножество его вершин, т.ч. у каждого ребра графа G хотя бы один конец входит в вершину из U.

Вход: Граф G, целое положительное число l.
Задача: Существует ли вершинное покрытие графа G, содержащее не более l вершин.

4 Доказательство NP-полноты рассматриваемых задач.

4.1 Сведение SAT к 3-SAT.

Для начала, в качетве примера, рассмотрим клоз длины 5, составленный из литералов (т.е. переменных или их отрицаний) $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5$:

$$\sigma_1 \vee \sigma_2 \vee \sigma_3 \vee \sigma_4 \vee \sigma_5 \tag{1}$$

Необходимо «переделать» клоз длины 5 в несколько клозов длины 3. Введем переменную x_1 и рассмотрим следующую формулу:

$$(\sigma_1 \vee \sigma_2 \vee x_1) \wedge (\bar{x_1} \vee \sigma_3 \vee \sigma_4 \vee \sigma_5) \wedge (\bar{\sigma_3} \vee x_1) \wedge (\bar{\sigma_4} \vee x_1) \wedge (\bar{\sigma_5} \vee x_1) \tag{2}$$

Докажем следующее утверждение.

$\boxed{\textit{Утверждение:}}$ Пусть заданы значения литералов $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5$. Тогда клоз (1) выполнен тогда и только тогда, когда найдется такое значение переменной x_1, что формула (2) будет выполнена.

Доказательство:

1. Пусь значения $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5$ такие, что клоз (1) выполнен. Тогда существует хотя бы один $\sigma_i = 1,\ i \in \{1, 2, 3, 4, 5\}$.

   Пусть $\sigma_1 = 1$ или $\sigma_2 = 1$ и $\sigma_3 = \sigma_4 = \sigma_5 = 0$. Тогда, если $x_1 = 0$, то формула (2) будет выполнена.

   Пусть $\sigma_1 = \sigma_2 = 0$ и хотя бы один из литералов $\sigma_3, \sigma_4, \sigma_5$ равен 1. Тогда, если $x_1 = 1$, то формула (2) будет выполнена.

   В остальных случаях (когда хотя бы один из литералов σ_1, σ_2 равен 1 и хотя бы один из литералов $\sigma_3, \sigma_4, \sigma_5$ равен 1), если $x_1 = 1$, то формула (2) будет выполнена.

2. Пусь значения $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5$ такие, что клоз (1) выполнен. Тогда $\sigma_1 = \sigma_2 = \sigma_3 = \sigma_4 = \sigma_5 = 0$, и $\sigma_1 \vee \sigma_2 \vee x_1 = x_1$ и $\bar{x_1} \vee \sigma_3 \vee \sigma_4 \vee \sigma_5 = \bar{x_1}$. Т.е. какое бы значение переменной x_1 мы не выбрали, формула (2) не будет выполнена!

ч.т.д.

Замечание: Достаточно рассматривать формулы следующего вида:

$$(\sigma_1 \vee \sigma_2 \vee x_1) \wedge (\bar{x_1} \vee \sigma_3 \vee \sigma_4 \vee \sigma_5) \tag{3}$$

Будем повторять описанную оперцию до тех пор, пока клоз (1) не «переделается» в формулу, в которой все клозы будут состоять не более чем из 3 литералов:

$$(\sigma_1 \vee \sigma_2 \vee x_1) \wedge (\bar{x_1} \vee \sigma_3 \vee x_2) \wedge (\bar{x_2} \vee \sigma_4 \vee \sigma_5) \tag{4}$$

Приведенные рассуждения легко обобщаются на случай клозов длины r.

Таким образом, задача SAT остается сложной, даже если мы ограничиваем каждый клоз тремя литералами. Заметим, что задача 2-SAT решается за линейное время.

4.2 Сведение 3-SAT к CLIQUE.

Будем строить граф $G = (V, E)$ следующим образом: для каждой тройки $C_r = l_1^r \vee l_2^r \vee l_3^r$ в φ добавим в множество V вершины v_1^r, v_2^r, v_3^r. Вершины v_i^r и v_j^q соединим ребром, если $r \neq q$ (т.е. относятся к разным тройкам) и соответсвующие этим вершинам литералы не являются отрицанием друг друга. Такой граф строится для формулы φ за полиномиальное время.

Выполнящим набором для формулы φ назовем набор таких значений ее переменных, при которых формула становится истинной.

Докажем следующее утверждение.

$\boxed{\textit{Утверждение:}}$ Для формулы $\varphi = C_1 \wedge C_2 \wedge ... \wedge C_p$ имеется выполняющий набор тогда и только тогда, когда в соответсвующем ей графе G найдется клика длины p.

Доказательство:

1. Пусть для формулы φ имеется выполняющий набор. Тогда для каждой тройки $C_r = l_1^r \vee l_2^r \vee l_3^r$ значение некоторого литерала $l_i^r = 1$. Рассмотри множество V' вершин, соответсвующих этим литералам, причем, если в одной тройке сразу несколько литералов равны 1, то случайным образом выберем только один из них. Тогда V' - клика. В самом деле, при нашем преобразовании 2 вершины, соответсвующие литералам из разных троек, оказываются не соединены ребром только в том случае, если литералы являются отрицанием друг друга. Но тогда хотя бы один из них должен бы был быть равен 0, а такие литералы не входят в множество V'.

2. Пусь граф G содержит клику V' размера k. Тогда каждому соответсвующему литералу присвоим значение 1. Если некоторые переменные оказались не учтены - присвоим им произвольное значение. По построению, вершины, соответсвующие одной тройке, не соединены ребром, и следовательно, клика V' содержит не более 1 вершины из каждой тройки. Т.е., если размер клики равен количеству троек (и равен p), то это означает, что мы рассмторели все тройки, составляющие формулу φ. Таким образом, каждая тройка C_r, а значит и вся формула φ, будет принимать истинное значение. При этом одной и той же переменной

не могут быть назначены различные значения, т.к. по построению вершины, соответсвующие литералу и его дополнению, не соединены ребром, а значит, не составляют клику.

ч.т.д.

<u>*Замечание:*</u> Данное утверждение было доказано для графов, имеющих специфическую структру. Однако этого достастачно для доказательства NP-полноты задачи CLIQUE т.к., если бы существовал алгоритм с полиномиальным временем работы, решающий задачу CLIQUE с графами общего вида, то должен бы был существовать алгоритм, решающий эту задачу и для специфических графов.

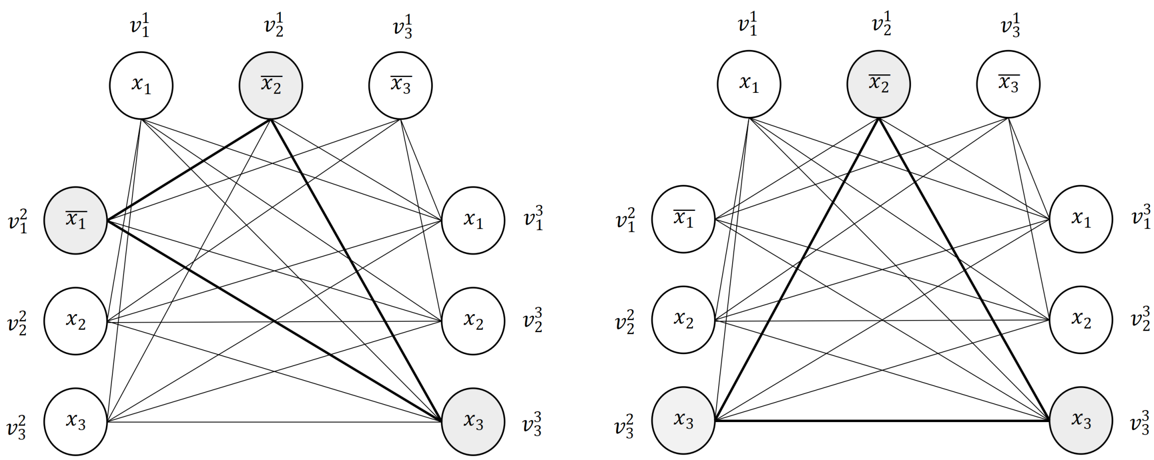


Рис. 1: *Граф, построенный для формулы (5). Серым цветом обозначены вершины, принимающие значение 1 на решающем наборе* $x_1 = 0, x_2 = 0, x_3 = 1$. *Представлено 2 возможных варианта выбора таких вершин.*

Пример: Рассмотрим формулу

$$\varphi = (\underbrace{x_1 \vee \bar{x_2} \vee \bar{x_3}}_{=:C_1}) \wedge (\underbrace{\bar{x_1} \vee x_2 \vee x_3}_{=:C_2}) \wedge (\underbrace{x_1 \vee x_2 \vee x_3}_{=:C_3}) \qquad (5)$$

Для данной формулы существует выполняющий набор (например, $x_1 = 0, x_2 = 0, x_3 = 1$). На рис.1 представены графы, построенные описанным выше способом для формулы φ и данного решающего набора. Серым цветом помечены вершины графа, образующие клику. Представлено 2 варианта таких наборов вершин.

4.3 Сведение CLIQUE к NODE COVER.

Рассмотрим граф $G = (V, E)$. *Дополнение графа* G – это граф $G' = (V, E')$, имеющий то же множество вершин, что и заданный граф G, но в котором две несовпадающие вершины смежны тогда и только тогда, когда они не смежны в G.

Утверждение: В графе G имеется клика размера k тогда и только тогда, когда дополнение графа G' содержит вершинное покрытие размера $l = |V| - k$.

Доказательство:

1. Пусть граф G содержит клику размера k. Тогда вершины U, не входящие в клику, образуют вершинное покрытие дополнения G'. В самом деле, предположим, что в G' найдется ребро, ни один из концов которого не входит в вершины из U. Тогда это ребро соединяет две вершины, принадлежащие клике графа G. Но, по определелению, клику образуют вершины, каждая пара которых соедененна в графе G ребром, а значит в двойственном графе G' такие вершины соеденены ребром быть не могут!

2. Пусть граф G' содержит вершинное покрытие размера l. Тогда вершины U', не входящие в вершинное покрытие, будут образовывать клику в графе G. В самом деле, предположим, найдется пара вершин из U', не соединенных ребром в графе G. Тогда они должны быть соединены ребром в дополнении G', а значит, одна из вершин должна принадлежать вершинному покрытию графа G'.

ч.т.д.

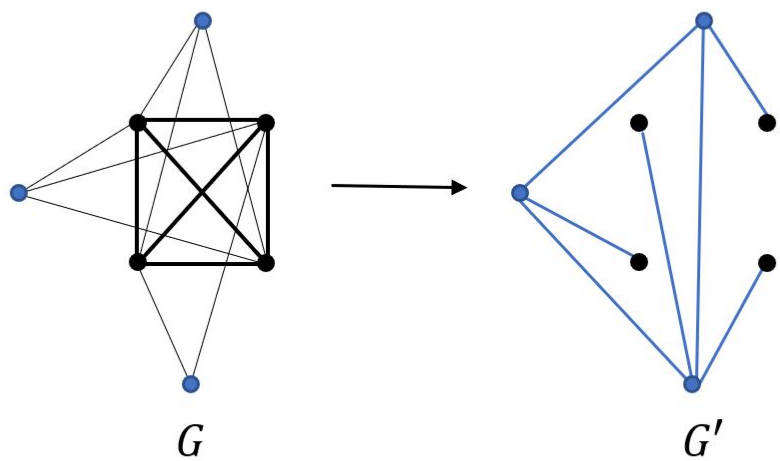


Рис. 2: *Черные вершины образуют клику для графа G, синие вершины образуют вершинное покрытие для графа G'.*

Список литературы.

[1] Karp, Richard. «Reducibility Among Combinatorial Problems». Proceedings of a Symposium on the Complexity of Computer Computations, Plenum Press — 1972.

[2] Кормен, Т., Лейзерсон, Ч., Ривест, Р., Штайн, К. Алгоритмы: построение и анализ — 2-е изд. — М.: Вильямс, 2005.