

Московский физико-технический институт
Физтех-школа прикладной математики и информатики

СЛОЖНОСТЬ ВЫЧИСЛЕНИЙ
IV СЕМЕСТР

Лектор: *Мусатов Даниил Владимирович*

h\nu

Автор: *Киселев Николай*
Репозиторий на Github

весна 2025

Содержание

1 Класс P	2
1.1 Базовые определения	2
1.2 Неконструктивные оценки P	2
1.3 Другие классы задач	3
1.4 Асимптотики различных задач	3
2 Класс NP	3
2.1 Определение через сертификат	4
2.2 Некоторые следствия из определений	5

1 Класс P

1.1 Базовые определения

Будем рассматривать задачи на распознавание, т.е. дан $A \subset \{0, 1\}^*$ и требуется по $x \in \{0, 1\}^* \mapsto \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases}$. Пусть M решает данную задачу, т.е. $\forall x(x \in A \Leftrightarrow M(x) = 1)$.

Определение 1.1. $time_M(x)$ — число шагов $M(x)$ при вычислении ответа.

Определение 1.2. $time_M(n) = \max_{x:|x|=n} time_M(x)$

Определение 1.3. $time_M(n) = O(f(n))$, если $\exists C : \forall n : time_M(n) \leq C \cdot f(n)$

Возникает вопрос: можно ли сказать, что $time_A(n) = \min_{M: M \text{ решает } A} time_M(n)$? Нет, но показать это достаточно сложно (Теорема Блюма).

Поэтому мы приходим к данному определению:

Определение 1.4. $\mathbf{DTIME}(t(n)) = \{A | \exists M : M(x) = 1 \Leftrightarrow x \in A, time_M(n) = O(t(n))\}$

Заметим, что для определения **DTIME**, необходимо задать модель вычислений. Обычно такой моделью выбирают многоленточную машину Тьюринга.

Определение 1.5. $P = \mathbf{DTIME}(poly(n)) = \bigcup_{k=1}^{\infty} \mathbf{DTIME}(n^k)$

Тезис Черча-Тьюринга в сильной форме: Любая задача, эффективно решаемая физическим устройством, решается за полиномиальное время на машине Тьюринга.

Пример (Нетривиальные примеры задач из **P**). 1. \mathbb{P} — множество простых чисел

2. Линейное программирование — как пример, нахождения максимума функции на многограннике. Эта задача не бинарная, но вот задача "достижима ли это число на многограннике" принадлежит классу **P**.
3. Симплекс-метод — алгоритм решения

1.2 Неконструктивные оценки P

Рассмотрим, например, задачу определения графа на планарность. Для этого существует два критерия: критерий Понtryгина-Куратовского: граф планарен \Leftrightarrow в нем нет подграфов, гомеоморфных $K_5, K_{3,3}$. Также, существует критерий Вагнера: граф планарен \Leftrightarrow в нем нет миноров $K_5, K_{3,3}$ (минор — граф, полученный из исходного удалением и стягиванием ребер). Рассмотрим свойства, которые сохраняются при удалении и стягивании ребер.

Теорема 1.1 (Робертсона-Сеймура). 1. Для любого свойства, аналогичного планарности выполнен аналог критерия Вагнера с конечным числом запрещенных миноров.

2. Наличие такого минора проверяется за полиномиальное время

Следствие. Любое такое свойство лежит в классе \mathbb{P} .

Но проблема в том, что мы не знаем миноров, которые необходимо проверить, чтобы найти проверить выполнение данного свойства.

1.3 Другие классы задач

Определение 1.6. $\text{QP} = \text{DTIME}(2^{\text{poly}(\log(n))}) = \bigcup_{c=1}^{\infty} \text{DTIME}(2^{(\log n)^c})$

Определение 1.7. $\text{E} = \text{DTIME}(2^{O(n)}) = \bigcup_{c=1}^{\infty} \text{DTIME}(2^{cn})$

Определение 1.8. $\text{EXP} = \text{DTIME}(2^{\text{poly}(n)}) = \bigcup_{c=1}^{\infty} \text{DTIME}(2^{n^c})$

Определение 1.9. $\text{EE} = \text{DTIME}(2^{2^{cn}}) = \bigcup_{c=1}^{\infty} \text{DTIME}(2^{2^{cn}})$

1.4 Асимптотики различных задач

Пример. $\text{LOG-CLIQUE} = \{G \mid \omega(G) \geq \log_2 n\} \in \text{QP}$. Является квазиполиномиальной, т.к. $C_n^{\log n} \leq n^{\log n}$, т.е. полный перебор осуществляется за квазиполином

Пример (Задача о доминирующем множестве в турнире). непон

Пример. $\text{GI} = \{(G_1, G_2) : G_1 \cong G_2\} \in \text{QP}$ (изоморфность графов).

Пример. $\text{3COL} = \{G : \chi(G) \leq 3\} \in E$

Теорема 1.2 (об иерархии по времени). *Если $f \ll g \Rightarrow \text{DTIME}(f(n)) \subsetneq \text{DTIME}(g(n))$*

2 Класс NP

Сейчас будем рассматривать модель вычислений — *недетерминированную машину Тьюринга* или НМТ. В отличие от обычной машины Тьюринга, функция перехода теперь многозначна (по аналогии с ДКА и НКА).

Соответственно, время работы такой машины Тьюринга — $\text{time}_M(x) = \max \# \text{шагов по всем вариантам перехода}$.

Замечание. Можем считать, что дерево переходов двоичное. Действительно, размер ветвлений ограничено мощностью $|\Sigma| \cdot |Q| \cdot |\{N, R, L\}|$ — некоторая константа, не зависящая от входа. Тогда каждое m -ветвление можно заменить $\log_2 m$ 2-ветвлениями.

Ответ данной машины будем понимать следующее:

$$M(x) = \begin{cases} 1, & \text{существует принимающая ветка} \\ 0, & \text{иначе} \end{cases}$$

Замечание. Ответ вычисляется как дизъюнкция по всем результатам работы машины

Определение 2.1. $\text{NTIME}(t(n))$ — класс языков, распознаваемых на НМТ за $O(t(n))$ шагов

Определение 2.2. $\text{NP} = \bigcup_{c=1}^{\infty} \text{NTIME}(n^c)$

Определение 2.3. $\text{NE} = \bigcup_{c=1}^{\infty} \text{NTIME}(2^{cn})$

Определение 2.4. $\text{NEXP} = \bigcup_{c=1}^{\infty} \text{NTIME}(2^{n^c})$

Замечание. $\text{NTIME}(t(n)) \subset \text{DTIME}(2^{t(n)})$

Замечание. $\text{NP} \subset \text{EXP}$ — за время EXP можно построить все дерево и вычислить ответ по определению.

2.1 Определение через сертификат

Теорема 2.1. $A \in \text{NP} \Leftrightarrow \exists V(x, s) - \text{ДМТ, т.ч. } x \in A \Leftrightarrow \exists s : V(x, s) = 1 \text{ и } V(x, s) \text{ работает за } \text{poly}(|x|)$

Доказательство.

- \Leftarrow Рассмотрим следующую НМТ, которая сначала печатает все возможные варианты s (достаточно написать полиномиальное количество символов, т.к. больше V не сможет прочесть), а потом на входе x, s запускает V . Таким образом, Получили машину M , которая в какой-то ветке напечатает нужный сертификат s и выведет $V(x, s) = 1$.
- \Rightarrow Возьмем в качестве сертификата код нужной ветви в машине M (0, если надо идти вправо, 1, если влево). Оно и будет нашим сертификатом s . Машина V будет спускаться, в соответствии с сертификатом, по дереву переходов. Тогда сертификат существует \Leftrightarrow существует принимающая ветвь $\Leftrightarrow A \in \text{NP}$.

□

Упражнение. Сформулировать и доказать аналогичную теорему для классов **NE**, **NEXP**

Утверждение 2.1. 1. $A, B \in \text{P} \Rightarrow A \cap B, A \cup B, \overline{A} \in \text{P}$

2. $A, B \in \text{NP} \Rightarrow A \cap B, A \cup B \in \text{NP}$

Замечание. Вообще говоря, $\overline{A} \in \text{NP}$ — открытый вопрос. Нельзя просто инвертировать значение машины M (пусть мы получим машину \overline{M}): тогда $x \in \overline{A} \Leftrightarrow$ все ветки \overline{M} принимающие, а это не то, что мы хотим. Таким образом, мы приходим к следующему определению:

Определение 2.5. $\text{coNP} = \{A \in \{0, 1\}^* : \overline{A} \in \text{NP}\}$.

Замечание. Аналогично можно доказать, что $A \in \text{coNP}$ тогда и только тогда, когда ответ вычисляется как конъюнкция всех результатов работы машины или тогда и только тогда, когда $\exists V : x \in A \Leftrightarrow \forall s V(x, s) = 1$ и V вычисляется полиномиально от длины x .

Пример. 1. $\text{SAT} = \{\varphi : \exists x : \varphi(x) = 1\} \in \text{NP}$.

Также имеет смысл рассмотреть двойственную задачу (к задаче опровергимости формулы):

2. $\text{TAUT} = \{\varphi : \exists x : \varphi(x) = 1\} \in \text{coNP}$

Замечание. Несмотря на доказанную теорему о полноте, вывод не будет являться сертификатом. Действительно, вывод, вообще говоря, не обязан быть полиномиальным и, в таком случае, машина не сможет полностью его прочесть (т.к. работает полиномиально от $|x|$)

Отдельный интерес у людей науки представляет множество $(\text{coNP} \cap \text{NP}) \setminus \text{P}$. Рассмотрим следующую задачу:

Определение 2.6. $\text{FACTORING} = \{(n, a, b) : \exists d \in (a, b) : d \text{ — простое и } n \mid d\}$

Утверждение 2.2. $\text{FACTORING} \in \text{NP} \cap \text{coNP}$.

Доказательство.

$\in \text{NP}$ — сертификат

$\in \text{coNP}$ сертификат — разложение на простые, каждое из которых $\notin (a, b)$.

□

2.2 Некоторые следствия из определений

Утверждение 2.3. $\text{P} = \text{NP} \Leftrightarrow \text{P} = \text{coNP}$

Утверждение 2.4. Следует из того, что $\text{coP} = \text{P}$ (P замкнут относительно дополнения).

Замечание. Тем не менее, может быть, что $\text{P} \neq \text{NP}$, но $\text{NP} = \text{coNP}$

Определение 2.7. $A \leq_p B$ сводится по Карпу (сводится полиномиально), если \exists всюду полиномиально вычислимая от $|x|$ функция $f(x)$, такая, что $x \in A \Leftrightarrow f(x) \in B$.

Определение 2.8. $\text{INDSET} = \{(G, k) : \text{в графе } G \text{ есть антиклика из } k \text{ вершин}\}$

Пример. $\text{CLIQUE} \leq_p \text{INDSET}$. Действительно, $f(G, k) = f(\bar{G}, k)$ (дополнение по ребрам).

Определение 2.9. $\text{4COL} = \{G : \exists \text{правильная раскраска в 4 цвета}\}$

Пример. $\text{4COL} \leq_p \text{SAT}$. Мы так уже делали на матлоге, когда сводили некоторые задачи к задачам выполнимости формулы. Для каждой вершины заведем две переменные p_i, q_i , отвечающие за цвет. Нам нужно для каждого ребра записать, что две вершины, являющиеся его концами, имеют разный цвет, и взять конъюнкцию, т.е:

$$\bigwedge_{(i,j) \in E} (p_i \neq p_j) \vee (q_i \neq q_j)$$

Размер данной формулы будет полиномиальным относительно размера графа.

Замечание (Свойства \leq_p).

1. $A \leq_p B, B \leq_p C \Rightarrow A \leq_p C$
2. $A \leq_p B, B \in \text{P} \Rightarrow A \in \text{P}$
3. $A \leq_p B, B \in \text{NP} \Rightarrow A \in \text{NP}$
4. $A \leq_p B \Rightarrow \bar{A} \leq_p \bar{B}$

Определение 2.10. Задача $B \in \text{NPH}$ (NP -трудной), если $\forall A \in \text{NP} : A \leq_p B$.

Определение 2.11. $\text{NPC} = \text{NP} \cap \text{NPH}$ (NP -полные)

Следствие.

1. $B \in \text{NPH}, B \leq_p C \Rightarrow C \in \text{NPH}$
2. $B \in \text{NPC}, B \leq_p C, C \in \text{NP} \Rightarrow C \in \text{NPC}$

Утверждение 2.5.

1. $\mathbf{P} \cap \mathbf{NPH} \neq \emptyset \Rightarrow \mathbf{P} = \mathbf{NP}$
2. $\mathbf{coNP} \cap \mathbf{NPH} \neq \emptyset \Rightarrow \mathbf{NP} = \mathbf{coNP}$

Доказательство.

2. $B \in \mathbf{NPH}, \mathbf{coNP} \Rightarrow \overline{B} \in \mathbf{NP}$. Теперь, если $A \leq_p B \Rightarrow \overline{A} \leq_p \overline{B}$. Отсюда получаем, что $\overline{A} \in \mathbf{NP}$ и тогда $\mathbf{NP} \subset \mathbf{coNP}$. Тогда:

$$S \in \mathbf{coNP} \Rightarrow \overline{S} \in \mathbf{NP} \Rightarrow \overline{S} \in \mathbf{coNP} \Rightarrow S \in \mathbf{NP}$$

□

Утверждение 2.6. $A \in \mathbf{P}, B, \overline{B} \neq \emptyset \Rightarrow A \leq_p B$

Доказательство. Рассмотрим

$$f(x) = \begin{cases} \in B, x \in A \\ \notin B, x \notin A \end{cases}$$

□

Следствие. $\mathbf{P} = \mathbf{NP} \Rightarrow \mathbf{NPC} = P \setminus \{\emptyset, \Sigma^*\}$

Определение 2.12. $\text{TMSAT} = \{(M, x, 1^t) : \exists y \ M(x, y) \text{ и работает за } \leq t \text{ шагов}\}$.

Утверждение 2.7. $\text{TMSAT} \in \mathbf{NP}$

Доказательство. Сертификат — y , верификатор — УМТ

□

Утверждение 2.8. $\text{TMSAT} \in \mathbf{NPC}$

Доказательство. Принадлежность \mathbf{NP} уже доказали, докажем принадлежность \mathbf{NPC} . Пусть $A \in \mathbf{NP}$. По определению: $x \in A \Leftrightarrow \exists s : V(x, s) = 1$. Положим M — машину Тьюринга, вычисляющую V , $t(n)$ — время работы V для $|x| = n$. Тогда положим $f(x) = (M, x, 1^{t(|x|)})$ и получим, что $A \leq_p \text{TMSAT}$.

□

Определение 2.13. $\text{3SAT} = \{\varphi | \varphi \text{ — выполнимая 3-КНФ}\}$.

Теорема 2.2 (Преобразование Цейтина). $\text{SAT} \leq_p \text{3SAT}$.

Доказательство. Для каждой подформулы φ_i заведем свою переменную p_i . Тогда достаточно записать конъюнкцию формул следующего вида: $q_i \equiv q_j * q_k$, если $\varphi_j = \varphi_j * \varphi_k$. Осталось для каждой такой формулы написать 3-КНФ.

□

Напоминание. Наша модель вычислений — одноленточную ДМТ, где лента бесконечная вправо. Пусть Γ — ленточный алфавит, Q — множество состояний, $Q \cap \Gamma = \emptyset$. Пусть $|Q \cup \Gamma| \in (2^{k-1}, 2^k]$, тогда будем использовать k бит для кодирования элементов $Q \cup \Gamma$. Конфигурация ДМТ — строка $AqaB$, $q \in Q, A, B \in \Gamma^*, a \in \Gamma$.

Определение 2.14. Беспрефиксный код — набор слов, где ни одно из слов не является началом другого

Определение 2.15. Беспрефиксное кодирование — функция, такая, что

Пусть также x — вход, $|x| = n$, машина V работает за $\leq t(n)$ шагов $\Rightarrow V$ использует $\leq t(n)$ ячеек \Rightarrow длина любой конфигурации $\leq t(n) + c$. Также будем считать, что машина имеет два завершающих состояния: q_{accept}, q_{reject} , причем после того, как она пришла в одно из них, она может еще какое-то время передвигаться по ленте, ничего не меняя.

Теорема 2.3 (Кука-Левина). $SAT \in NPC$

Доказательство. Рассмотрим V — верификатор SAT . Покажем, как написать следующую формулу: "из данных x , данных y машина придет в q_a ". Рассмотрим следующую таблицу (индекс столбца — номер ячейки, индекс строки — индекс в конфигурации):

	0	1	2	...	$t(n)$
0					
1					
2					
:					
$t(n)$					

Пусть p_{ij} — k -бит, которые стоят в ij -ой клетке данной таблицы. Запишем формулу, утверждающую, что таблица корректная для V . Тогда наша формула будет иметь вид: $\varphi = \varphi_{start} \wedge \varphi_{accept} \wedge \varphi_{step}$, где:

1. φ_{start} определяет, что изначально машина находилась в корректной конфигурации (изначальная конфигурация $q_1x\#y\#\dots\#$), т.е., что $p_0 = q_1x\#y\#\dots\#$:

$$\begin{aligned} \varphi_{start} = & \underbrace{(p_{00} = q_1)}_{q_1} \wedge \underbrace{(p_{01} = x_1) \wedge \dots \wedge (p_{0n} = x_n)}_x \wedge \underbrace{(p_{0,n+1} = \#)}_{\#} \wedge \\ & \wedge \underbrace{(p_{0,n+2} = 0 \vee p_{0,n+2} = 1) \wedge \dots \wedge (p_{0,n+m+1} = 0 \vee p_{0,n+m+1} = 1)}_y \wedge \\ & \wedge \underbrace{(p_{0,n+m+2} = \#) \wedge \dots \wedge (p_{0,t(n)} = \#)}_{\#\dots\#} \end{aligned}$$

2. φ_{accept} проверяет, что в таблице есть хотя бы одно завершающее состояние, т.е., что наша машина завершилась и приняла слово.

$$\varphi_{accept} = \bigvee_{j=0}^{t(n)} (p_{t(n),j} = q_{accept})$$

3. φ_{step} проверяет, что машина совершила все переходы корректно. Действительно, для таблицы символ p_{ij} однозначно восстанавливается по $p_{i-1,j-1}, p_{i-1,j}, p_{i-1,j+1}, p_{i-1,j+2}$:

$p_{i-1,j-1}$	$p_{i-1,j}$	$p_{i-1,j+1}$	$p_{i-1,j+2}$
p_{ij}			

Тогда

$$\varphi_{step} = \bigwedge_{i=1}^{t(n)} \bigwedge_{j=1}^{t(n)} (p_{ij} = f(p_{i-1,j-1}, p_{i-1,j}, p_{i-1,j+1}, p_{i-1,j+2}))$$

Где f — функция, которая восстанавливает по $p_{i-1,j-1}, p_{i-1,j}, p_{i-1,j+1}, p_{i-1,j+2}$ значение p_{ij} .

Тогда формула выполнима $\Leftrightarrow V$ приняло изначальную формулу \Rightarrow получили требуемое

□

Определение 2.16. NAE – 3SAT — Даны 3-КНФ. Есть ли набор значений, такой, что в каждой скобке есть истинные и ложные литералы?

Утверждение 2.9. 3SAT \leqslant_p NAE – 3SAT

Доказательство. Рассмотрим преобразование:

$$(a_i \vee b_i \vee c_i) \mapsto (a_i \vee b_i \vee x_i) \wedge (\neg x_i \vee c_i \vee z)$$

Где x_i уникально для каждого i , а z одно для всей формулы.

Покажем, что исходная формула φ выполнима тогда и только тогда, когда построенная формула ψ принадлежит NAE – 3SAT.

В одну сторону: если φ выполнима, выберем выполняющее назначение и положим $z = 0$. Для каждой скобки $(a_i \vee b_i \vee c_i)$ подберём x_i так, чтобы обе скобки $(a_i \vee b_i \vee x_i)$ и $(\neg x_i \vee c_i \vee 0)$ удовлетворяли условию NAE. Значение x_i можно взять из следующей таблицы истинности:

a_i	\vee	b_i	\vee	c_i	$(a_i \vee b_i \vee c_i)$	\wedge	$(\neg x_i \vee c_i \vee z)$
0	0	1			0	0	1
0	1	0			0	1	0
0	1	1			0	1	1
1	0	0			1	0	0
1	0	1			1	1	1
1	1	0			1	0	0
1	1	1			1	1	1

Таким образом, полученное назначение даёт решение NAE – 3SAT для ψ .

В обратную сторону: пусть ψ имеет решение в смысле NAE – 3SAT. Если в этом решении $z = 1$, то инвертируем все переменные (включая z) — полученный набор также будет решением, поскольку условие NAE инвариантно относительно отрицания. Поэтому можно считать, что $z = 0$. Тогда для каждой исходной скобки $(a_i \vee b_i \vee c_i)$ имеем две скобки: $(a_i \vee b_i \vee x_i)$ и $(\neg x_i \vee c_i \vee 0)$. Из условия NAE для второй скобки при $z = 0$ получаем, что $\neg x_i \vee c_i$ истинно (иначе все литералы были бы ложны). Для первой скобки условие NAE означает, что не все три литерала равны. Совокупность этих условий влечёт истинность $a_i \vee b_i \vee c_i$: действительно, если бы $a_i = b_i = c_i = 0$, то из первой скобки следовало бы $x_i = 1$ (чтобы не все нули), но тогда вторая скобка стала бы $(\neg 1 \vee 0 \vee 0) = (0 \vee 0 \vee 0)$ — все ложны, противоречие. Значит, все исходные скобки истинны, т.е. φ выполнима.

Таким образом, построенное сведение полиномиально и корректно, откуда NAE – 3SAT \in NP.

□

Утверждение 2.10. NAE – 3SAT \leqslant_p 3COL.

Доказательство. По формуле φ с переменными x_1, \dots, x_n и дизъюнктами C_1, \dots, C_m (каждый — тройка литералов) строим граф G :

1. Три вершины T, F, B , соединённые попарно (треугольник).
2. Для каждой переменной x вершины x и $\neg x$, соединённые ребром, и каждая из них соединена с B .
3. Для каждого дизъюнкта $C = (l_1 \vee l_2 \vee l_3)$ вершины c_1, c_2, c_3 , образующие треугольник, и для каждого i вершина c_i соединена с l_j, l_k , где $\{i, j, k\} = \{1, 2, 3\}$.

Докажем, что $\varphi \in \text{NAE-3SAT} \iff \chi(G) \leq 3$.

\Rightarrow Пусть v — оценка, в каждом дизъюнкте есть и истинный, и ложный литерал. Красим T, F, B в разные цвета. Вершину x красим цветом T , если $v(x) = 1$, и F иначе; $\neg x$ — противоположным цветом. В каждом треугольнике дизъюнкта цвета c_i подбираем так: поскольку среди l_1, l_2, l_3 есть оба цвета, можно назначить c_i цвет, отличный от цветов двух смежных литералов (существование такой раскраски проверяется перебором случаев).

\Leftarrow Пусть G правильно раскрашен в 3 цвета. Можно считать, что цвета T, F, B различны. Тогда x и $\neg x$ имеют цвета T и F (так как оба смежны с B и между собой). Положим $v(x) = 1$, если x цвета T , иначе 0. В каждом дизъюнкте C вершины c_1, c_2, c_3 имеют три разных цвета. Каждая c_i смежна с двумя литералами, поэтому эти литералы не могут иметь цвет c_i . Значит, среди трёх литералов не все одинакового цвета.

Таким образом, сведение корректно и выполняется за полиномиальное время. \square

Определение 2.17. EXACTSETCOVER — следующая задача: пусть даны $S_1, \dots, S_m \subset \{1, 2, \dots, n\}$, надо понять, существует ли i_1, \dots, i_k и $\bigsqcup_{j=1}^k S_{i_j} = \{1, \dots, m\}$

Утверждение 2.11. 3SAT \leq_p EXACTSETCOVER.

Доказательство. По формуле φ с переменными x_1, \dots, x_n и дизъюнктами C_1, \dots, C_m (каждый — тройка литералов) строим множество U и семейство подмножеств \mathcal{F} .

1. Элементы: для каждой переменной x_i : x_i и $\neg x_i$; для каждого дизъюнкта C_j : c_{j1}, c_{j2}, c_{j3} и d_{C_j} .
2. Множества:
 - (a) Для каждого литерала l (т.е. x_i или $\neg x_i$) множество $L_l = \{l\} \cup \{c_{jk} \mid$ литерал l входит в C_j на
 - (b) Для каждого дизъюнкта C_j и каждого непустого $T \subseteq \{1, 2, 3\}$, $T \neq \{1, 2, 3\}$ (т.е. $|T| \leq 2$), множество $M_{j,T} = \{d_{C_j}\} \cup \{c_{jk} \mid k \in T\}$.

Утверждаем: φ выполнима \iff существует точное покрытие U выбранными множествами.

\Rightarrow Пусть v — выполняющая оценка. Выберем L_{x_i} , если $v(x_i) = 1$, иначе $L_{\neg x_i}$. Эти множества покрывают все элементы $x_i, \neg x_i$ и все c_{jk} , соответствующие истинным литералам. Для каждого C_j пусть S_j — множество позиций истинных литералов. Тогда $S_j \neq \emptyset$. Пусть $T_j = \{1, 2, 3\} \setminus S_j$ — позиции ложных литералов ($|T_j| \leq 2$). Выберем M_{j,T_j} (если $T_j = \emptyset$, берём $M_{j,\emptyset} = \{d_{C_j}\}$). Эти множества попарно не пересекаются с выбранными L и между собой и покрывают все оставшиеся c_{jk} и d_{C_j} .

⇐ Пусть есть точное покрытие \mathcal{C} . Для каждого i ровно одно из $L_{x_i}, L_{\neg x_i}$ принадлежит \mathcal{C} (иначе x_i или $\neg x_i$ не покрыты). Задаём $v(x_i) = 1$, если $L_{x_i} \in \mathcal{C}$, иначе 0. Для каждого C_j элемент d_{C_j} покрыт некоторым $M_{j,T}$, $T \subseteq \{1,2,3\}$, $|T| \leq 2$. Тогда c_{jk} при $k \in T$ покрыты этим $M_{j,T}$, а c_{jk} при $k \notin T$ должны быть покрыты L_l , т.е. соответствующие литералы истинны. Так как $T \neq \{1,2,3\}$, имеем $k \notin T$ хотя бы для одного k , значит, в C_j есть истинный литерал.

Следовательно, сведение корректно и полиномиально. □