NP-полные задачи

Шовкопляс Григорий

Алгоритмы и структуры данных Advanced

Proof by contradiction:

If $P = NP$

$P - NP = 0$

$P(1-N) = 0$

$P = 0$ or $N = 1$

We know $P \neq 0$ and $N \neq 1$

Therefore $P \neq NP$

Введение

Определения и мотивация

- **Класс P** – множество задач, которые решаются за полиномиальное время p(n).
  - n – размер входа
  - p(n) для каждой задачи своё
- **Задача разрешимости** (decision problem) – задача, с произвольными входными данными, и выходными в формате «Да»/«Нет». Например:
  - задача о гамильтоновом цикле
  - задача о связности графа
  - задача о вершинном покрытии размера не более k

Недетерминированная программа

```
HAM(V,E)
  for i = 1 to n
    p[i] ← {v1, v2, ... , vn}
  if check(p, E)
    return 'Yes'
  else
    return 'No'
```

- В недетерминированной программе добавилась операция недетерминированного выбора:
    - в переменную записывается один элемент множества
    - присваивается тот элемент, который приведет к ответу «Да»

Определение NP

- **Класс NP** – множество задач разрешимости, которые решаются за полиномиальное время p(n) с помощью недетерминированной программы
    - n – размер входа
    - p(n) для каждой задачи своё

Определение NP

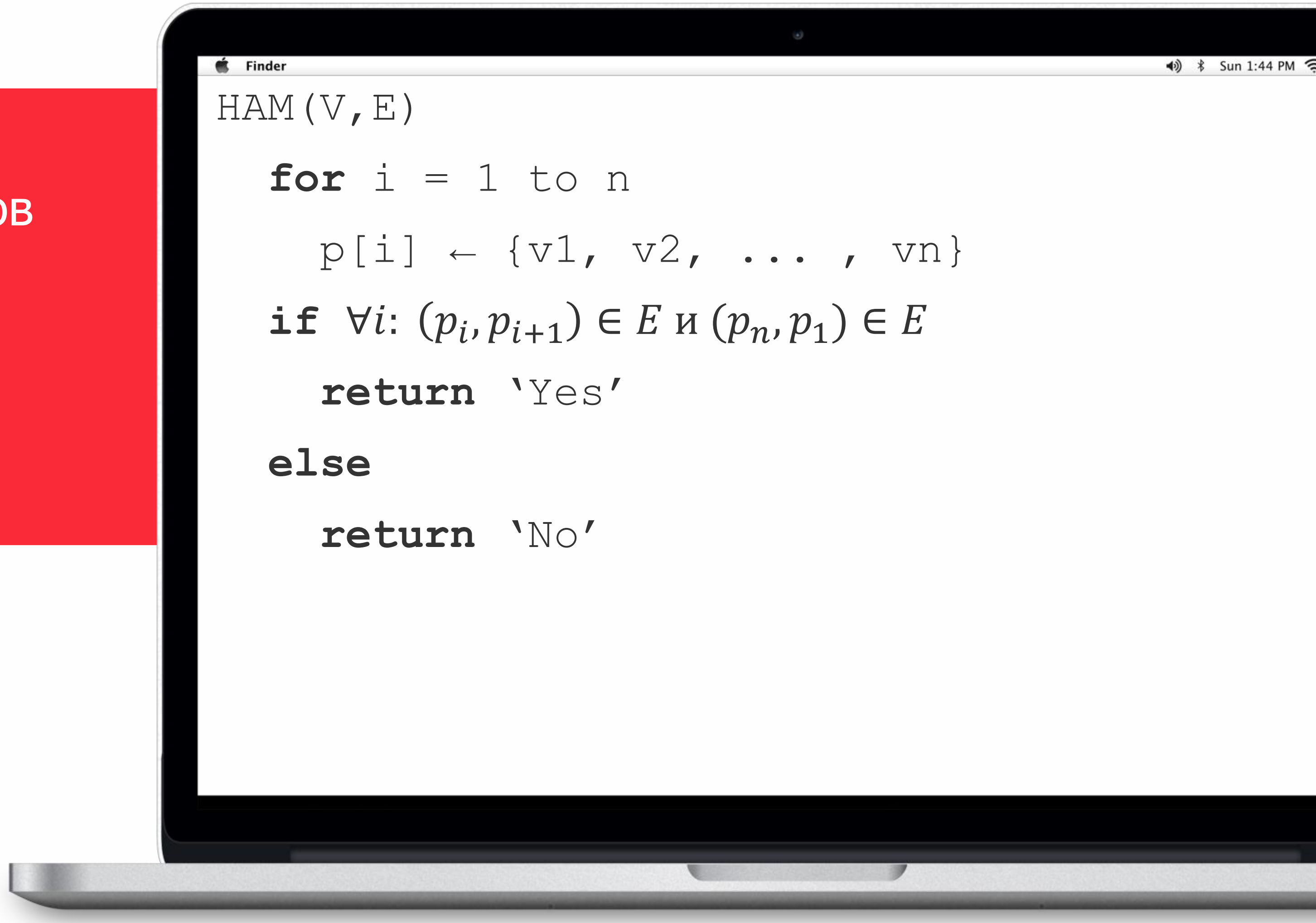
- Альтернативное определение NP:
  - **верификатор** – программа, которая по входным данным и сертификату проверяет, что ответ на данные входные «Да»
  - **сертификат** – данные полиномиального размера;
  - верификатор работает за полиномиальное время.
- **Класс NP** – задачи, для которых можно построить сертификат и верификатор.

Эквивалентность определений

- Недетерм. программа $\to$ сертификат:
  - ответы на выборы: сертификат
  - проверка после выборов: верификатор
- Сертификат $\to$ недетерм. программа:
  - сгенерировать сертификат c недетерм. выбором
  - **If** verifier(c) **return** ‘Yes’ **else return** ‘No’

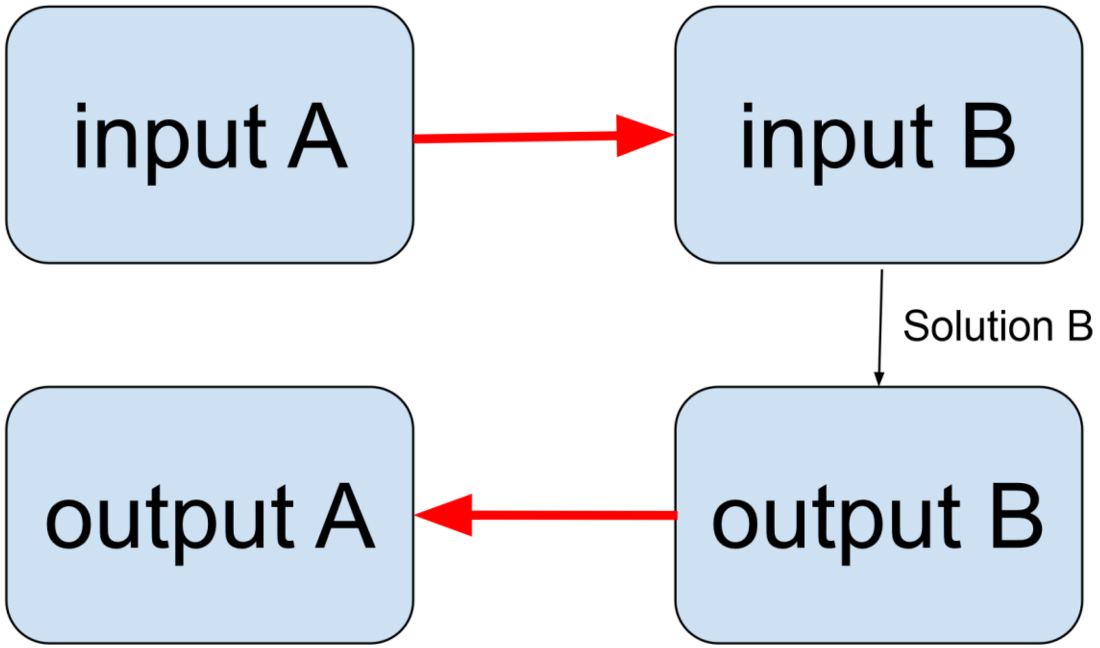
Пример задачи из NP

- Есть ли в графе гамильтонов цикл?
- Сертификат: перестановка



NP-hard и NP-complete

Сведение задачи A к задаче B



NP-hard и NP-complete

- Сведение A к B – программа, которая по входу задачи A получает **эквивалентный** вход B
    - программа полиномиальна
    - записываем: $A \leq_p B$
    - означает, что задача B не проще задачи A.
- $A \in NPH$ (A – NP-трудная), если $\forall X \in NP$: $X \leq_p A$
- $A \in NPC$ (A – NP-полная), если:
    - $A \in NP$
    - $A \in NPH$
- Другими словами: $NPC = NP \cap NPH$

Мотивация

- Большинство оптимизационных задач NP-полны
- Понять, какие задачи сложные
  - научиться это доказывать
- Сложность сведения показывает, какие задачи сложнее интуитивно
- Научиться откладывать сложные задачи

Схема доказательства

- $A \in NPC$ (A – NP-полная), если:
  - $A \in NP$
    - написать недетерм. программу
    - описать сертификат и верификатор
  - $A \in NPH$
    - доказать, что все задачи сводятся к A, сложно
    - найти $\mathrm{Y} \in NPH: Y \leq_p A$
- (но) Хотя бы для одной задачи придется доказать, что к ней сводятся все.

Примеры NP-полных задач

3SAT

- Задана булева формула в КНФ:
  - В скобке ровно три литерала (переменная или ее отрицание)
  - Можно ли удовлетворить?
- Задача 3SAT В NP:
  - Сертификат: значения $x_1, x_2, \dots x_n$
  - Верификатор: вычисляет формулу
- Задача 3SAT В NPH: пока поверим в это

Задача о независимом множестве (IND)

- Задан неорграф и число k
  - Существует ли $I \subset V\colon \forall\, vu \in E\colon v \notin I$ или $u \notin I$
  - $|I| \geq k$
- Сведем 3SAT к IND, по формуле построим граф
  - Что покажет, что IND не проще 3SAT
- Возьмем формулу, построим для нее граф
  - Каждую скобку преобразуем в треугольник
  - Соединим x_i и $\overline{x_i}$ ребрами
- n переменных и m скобок → сделаем k = m, решим задачу IND

Корректность сведения

- $|I| \leq m$
- $\exists I : |I| = m$
  - назначаем 1 переменным из I, остальным что получится
  - каждая скобка удовлетворилась
  - противоречий нет, в I нет обратных друг другу
- Обратно, если $\exists\{x_1, x_2 \dots x_n\}$, удовлетворяющее формулу
  - в каждой скобке есть хотя бы одна 1
  - возьмем по одной вершине из скобки в I
  - I образует независимое множество
- Сведение полиномиально

Доказательство NPC

- Показали, что IND в NPH
- Осталось показать, что IND в NP
    - Сертификат: независимое множество
    - Верификатор: проверка, что вершины сертификата не соединены

Задача о вершинном покрытии

- Задан неограф и число k
  - Существует ли $C \subset V$: $\forall\, vu \in E$: $v \in C$ или $u \in C$
  - $|C| \leq k$
- Сведем IND к VertexCover
  - Дополнение независимого множества есть вершинное покрытие
  - $G' = G, k' = |V| - k$
- $\exists I$: $|I| \leq k \Leftrightarrow \exists C: |C| \leq |V| - k = k'$

Задача о максимальной клике

- Задан неограф и число k
  - Существует ли $\mathrm{C} \subset V$: $\forall v \in \mathrm{C}$ и $u \in \mathrm{C}$: $vu \in E$
  - $|\mathrm{C}| \geq k$
- $V' = V, E' = \bar{E}$

SAT

SAT

- Задана булева формула в КНФ из n переменных
  - удовлетворима ли она?
- Покажем $SAT \leq_p 3SAT$
- По КНФ формуле надо построить 3-КНФ формулу
- Если в скобке 1 или 2 литерала, то очевидно, иначе:
  - Преобразуем $a = (x_1 \vee \overline{x_5} \vee \overline{x_2} \vee x_3 \vee x_4)$
  - В $b = (x_1 \vee \overline{x_5} \vee y_1) \wedge (\overline{y_1} \vee \overline{x_2} \vee y_2) \wedge (\overline{y_2} \vee x_3 \vee x_4)$
- Если a удовлетворимо, то b удовлетворимо:
  - $y_1 = \overline{x_1} \wedge x_5$
  - $y_2 = \overline{x_1} \wedge x_5 \wedge x_2$

Доказательство сведения

- Преобразуем $a = (x_1 \vee \overline{x_5} \vee \overline{x_2} \vee x_3 \vee x_4)$
- В $b = (x_1 \vee \overline{x_5} \vee y_1) \wedge (\overline{y_1} \vee \overline{x_2} \vee y_2) \wedge (\overline{y_2} \vee x_3 \vee x_4)$
- Если a неудовлетворимо, то b неудовлетворимо:
    - Все литералы a равны 0
    - Переменных y меньше, чем скобок
    - Каждую скобку в b не удовлетворить

3COLOR

3COLOR

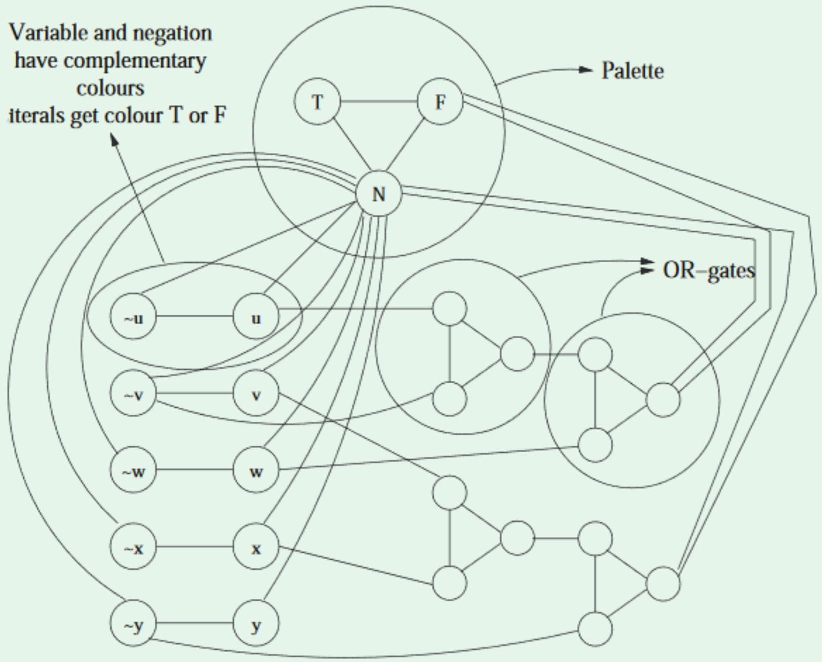
- Задан неограф
    - Можно ли раскрасить его в 3 цвета
    - $color(v) \leftarrow \{1, 2, 3\} \, \forall \, vu \in E: \, color(v) \neq color(u)$
- 3COLOR в NP
    - Сертификат: раскраска
    - Верификатор: аналогично предыдущим
- Покажем $3SAT \leq_p 3COLOR$

3COLOR

- По 3-КНФ формуле построим граф:
  - Сделаем «Палетку»: треугольник T, F, N
  - Соединим x_i и $\overline{x_i}$ ребрами
  - Проведем из всех y ребра в N
  - Для каждой скобки C_j заведем два треугольника
  - Если y в C_j проведем ребро
  - Из всех «выходов» C_j проведем ребра в F и N

3COLOR

$$\varphi = (u \vee \neg v \vee w) \wedge (v \vee x \vee \neg y)$$



Доказательство сведения

- Если формула разрешима → Есть раскраска
- Если есть раскраска → Формула разрешима

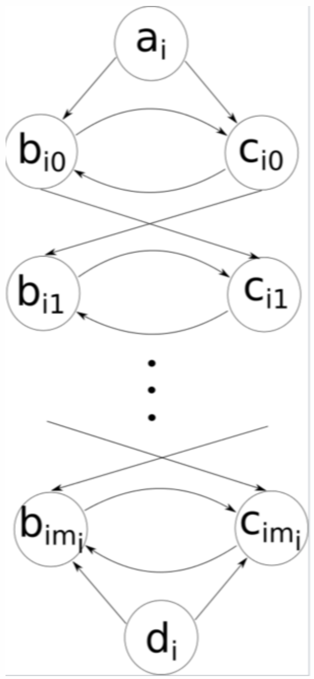
Гамильтонов цикл в ориентированном графе*

Гамильтонов цикл

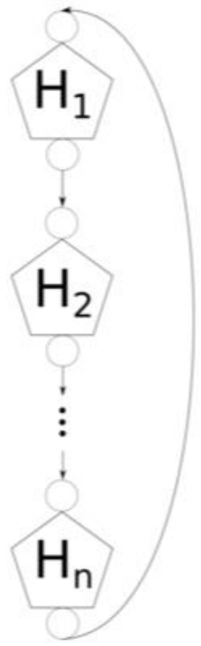
- Дан ориентированный граф G = (V, E)
    - Есть ли в нем гамильтонов цикл?
- HAM в NP
    - Доказали ранее
- Покажем $3SAT \leq_p HAM$

Гамильтонов цикл

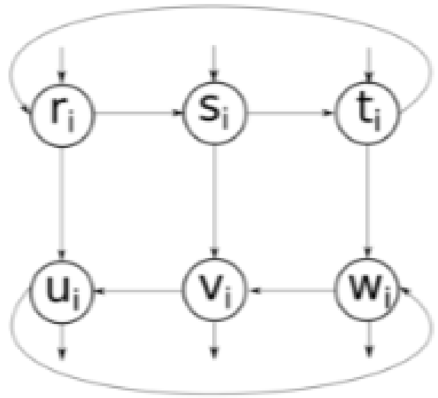
а) подграф для вершин



б) общий вид графа



в) подграф для скобки



Для более подробных картинок смотрите видео-лекции

Сумма подмножества

Сумма подмножества

- Дано множество S из n целых чисел и число s
  - Можно ли выбрать $S' \subset S, \text{что } \sum_{x \in S'} x = s$?
- SSP в NP
  - Сертификат: подмножество
  - Верификатор: вычислить сумму и сравнить с s
- Покажем $3SAT \leq_p SSP$

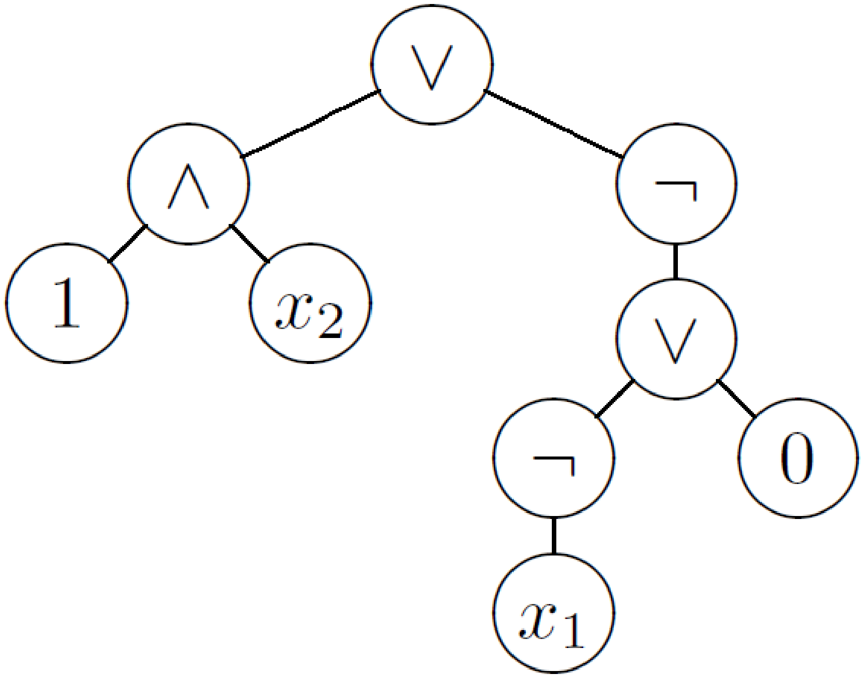
Сумма подмножества

- По 3-КНФ формуле построим множество из 2(n+m) чисел:
  - В каждом числе n + m цифр (n – переменных, m – скобок)
  - Для каждой переменной заведем два числа
    - v_i – 1 в разряде номера переменной и в разрядах тех, скобок, где переменная входит без отрицания
    - w_i – 1 в разряде номера переменной и в разрядах тех, скобок, где переменная входит с отрицанием
  - Для каждой скобки заведем два числа: с 1 и 2 в разряде этой скобки
  - s = 1111…11144..4

Почему есть хотя бы одна NP-полная задача?

Circuit-SAT

- Задана булева схема
- Удовлетворима ли она?



Circuit-SAT

- Покажем $\text{Circuit} - SAT \leq_p SAT$
- По схеме надо построить формулу
- Для каждой вершины заведем переменную x_v

0	$(\overline{x_v})$
1	(x_v)
$v = \neg a$	$(x_v \vee x_a) \wedge (\overline{x_v} \vee \overline{x_a})$
$v = a \wedge b$	$(\overline{x_a} \vee \overline{x_b} \vee x_v) \wedge (x_a \vee \overline{x_v}) \wedge (x_b \vee \overline{x_v})$
$v = a \vee b$	$(x_a \vee x_b \vee \overline{x_v}) \wedge (x_v \vee \overline{x_a}) \wedge (x_v \vee \overline{x_b})$

NP полнота Circuit-SAT

- Неформально покажем идею, как любую задачу $A \in NP$ свести к Circuit-SAT
  - существуют сертификат и верификатор для A
  - по входу задачи A, надо получить схему
  - вход – известный набор бит:
    - превращаем в вершины 0 и 1
  - сертификат – неизвестный набор бит:
    - превращаем в переменные
  - верификатор – программа, программа $\rightarrow$ схема
  - существует сертификат, который принимается верификатором $\Leftrightarrow$ схема удовлетворима

Приближенные алгоритмы: зачем?

Определения

- Задача (дискретной) оптимизации – найти объект, который минимизирует/максимизирует целевую функцию. Например:
  - задача о мин. вершинном покрытии
  - задача о макс. паросочетании
  - задача о рюкзаке
  - задача коммивояжера
- $\alpha(n)$-приближенный алгоритм находит решение оптимизационной задачи, которое отличается от оптимального не более, чем в $\alpha(n)$ раз
  - n – размер входа
  - $\max\left(\frac{\text{C}}{C^*}, \frac{C^*}{C}\right) \leq \alpha(n), C^* - \text{опт.}, \text{C} - \text{найденное}$
  - алгоритм полиномиален от n

Мотивация

- Не умеем решать NP-полные задачи за полином
- Хочется получать хорошие решения опт. задач
- $\alpha(n)$ –метрика сложности задачи
- Полезно в других подходах к решениям этих задач
- Но! Приближённые алгоритмы не единственный способ решать NP-полные опт. задачи.

Еще определения

- Полиномиальная схема аппроксимации – алгоритм, которому на вход подается еще и параметр $\varepsilon > 0$, который описывает, насколько точный ответ нужно найти:
  - для фиксированного ε, схема является $(1 + \varepsilon)$ -приближенным алгоритмом
  - время работы зависит также от ε
  - время работы полиномиально от n, но не от ε
- English: PTAS (polynomial-time approximation scheme)

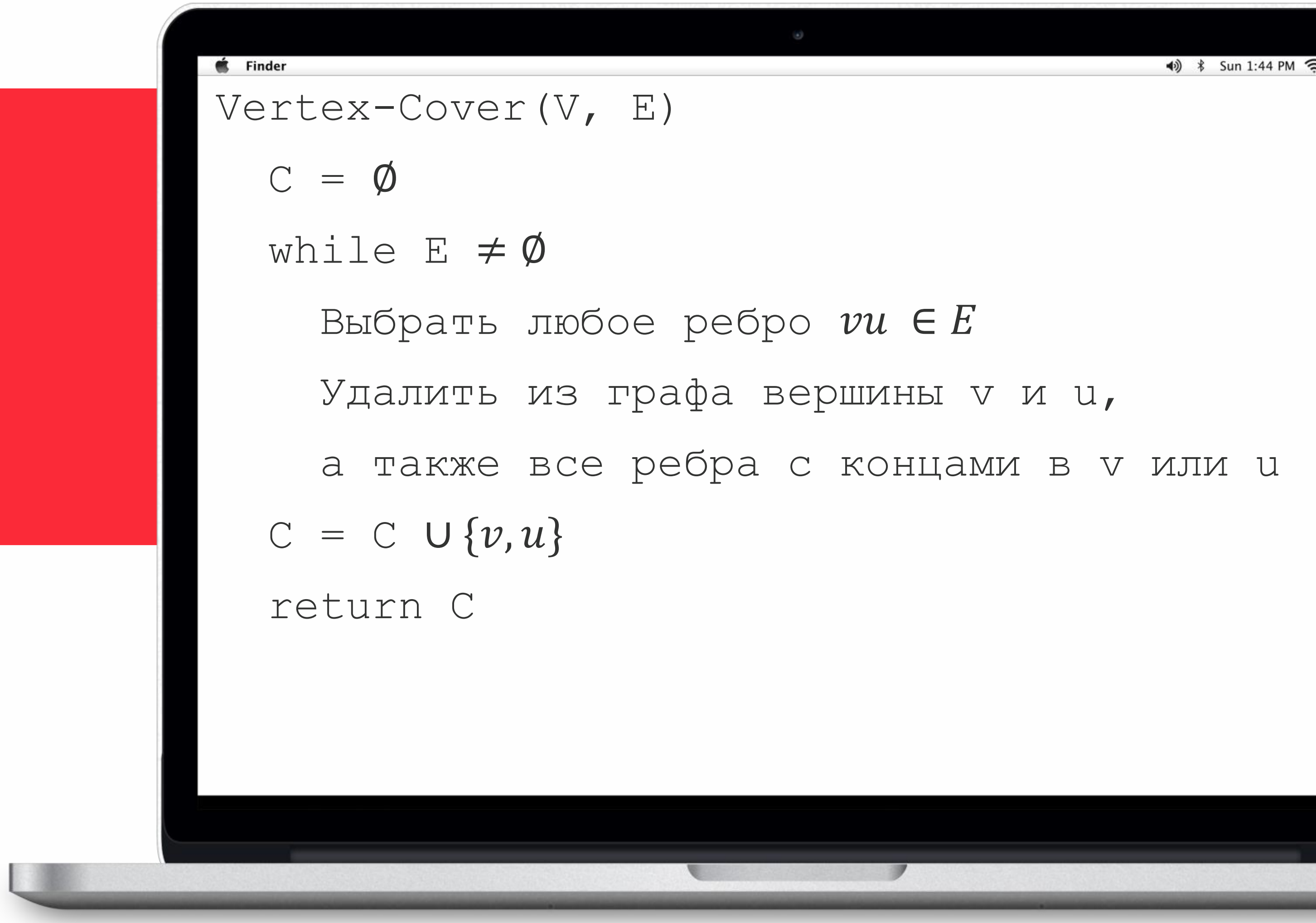
Вершинное покрытие

Задача о вершинном покрытии

- Задан неограф G = (V, E)
    - Существует ли $C \subset V$: $\forall\, vu \in E$: $v \in C$ или $u \in C$
    - $|C| \to min$

Задача о вершинном покрытии

Решение



Анализ

- Пусть А – множество выбранных ребер
- у ребер в А нет общих концов.
- C_0 – оптимальное вершинное покрытие.
- $|C_0| \geq |A|$
    - чтобы покрыть ребра из А нужно $\geq$ $|A|$ вершин
- $|C| = 2|A|$
- Значит, $C \leq 2|C_0|$.
- Предложенный алгоритм 2-оптимальный (2-приближённый).

Задача коммивояжера

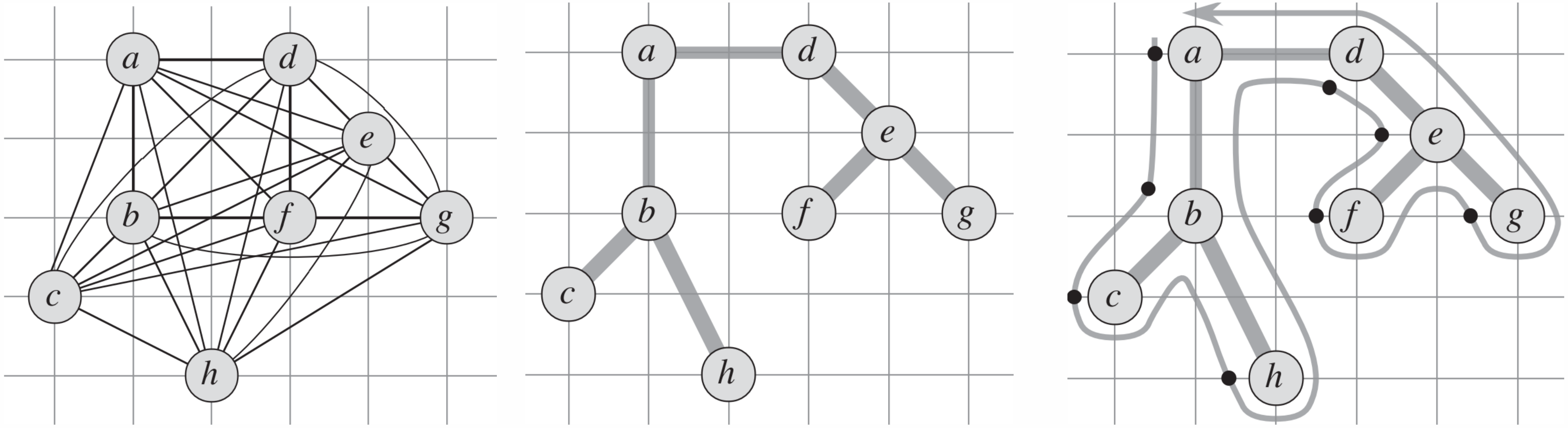
Задача коммивояжера

- Задан взвешенный неограф G = (V, E, w)
  - $n = |V|, w(u,w) \geq 0$
  - Найти перестановку $p_1, p_2, \dots p_n$
    - $p_i \in V, p_i \neq p_j$ при $i \neq j$
    - $w(p_n, p_1) + \sum_{i=1}^{n-1} w(p_i, p_{i+1}) \to min$
  - English: TSP (Travelling Salesman Problem)
  - Две вариации:
    - С условием неравенства треугольника
    - Без условия неравенства треугольника

Задача коммивояжера

- Есть неравенство треугольника
    - Построим минимальное остовное дерево
    - Обойдем обходом в глубину остовное дерево
    - Выпишем вершины в порядке посещения
    - Оставим только первые вхождения вершин

Задача коммивояжера



Анализ

- H^* – оптимальный цикл, H – найденный цикл
- T – мин. остовное дерево
  - Заметим, что $2w(T) \geq w(H)$
    - в обходе каждое ребро пройдено два раза; $2w(T)$
    - в цикле пропускали вершины + неравенство треуг.
  - Убрать ребро из цикла $\rightarrow$ остовное дерево
  - $w(T) \leq w(H^*)$
  - Из чего следует: $w(H) \leq 2w(T) \leq 2w(H^*)$
- Алгоритм 2-оптимальный

А если неравенства треугольника нет?

- Предположим: $\exists\alpha$-приближённый алгоритм
- Решим задачу о гамильтоновом цикле за полином
- В $G = (V, E)$ надо найти гамильтонов цикл
  - $G' = \{V, E' = V \times V, w\}$
  - $w(vu) = 1, \text{если } vu \in E$
  - $w(vu) = \alpha|V| + 1, \text{если } vu \notin E$
- G – гамильтонов $\Rightarrow w(H^*) = |V|$
- Алгоритм найдет $H, \text{что } w(H) \leq \alpha w(H^*) \leq \alpha|V|$
- G – гамильтонов $\Leftrightarrow w(H) \leq \alpha|V|$

Задача коммивояжера

- Альтернативный алгоритм
  - Построим минимальное остовное дерево T
  - Найти паросочетание минимальной стоимости M на множестве вершин T с нечетными степенями
  - Добавить ребра M к T и получить Эйлеров граф
  - Найти Эйлеров обход в новом графе
  - Построить Гамильтонов цикл, посещая вершины графа G в том порядке, в котором они встречаются Эйлеровом обходе

Анализ

- H^* – оптимальный цикл, H – найденный цикл,
- M – паросочентание, T – минимальное остновное дерево
- $\mathrm{w}(M) \leq \frac{1}{2} w(H^*)$
- $w(H) \leq w(T) + w(M) \leq \ w(H^*) + \frac{1}{2} w(H^*) = \frac{3}{2} w(H^*)$
- Алгоритм $\frac{3}{2}$ –оптимальный

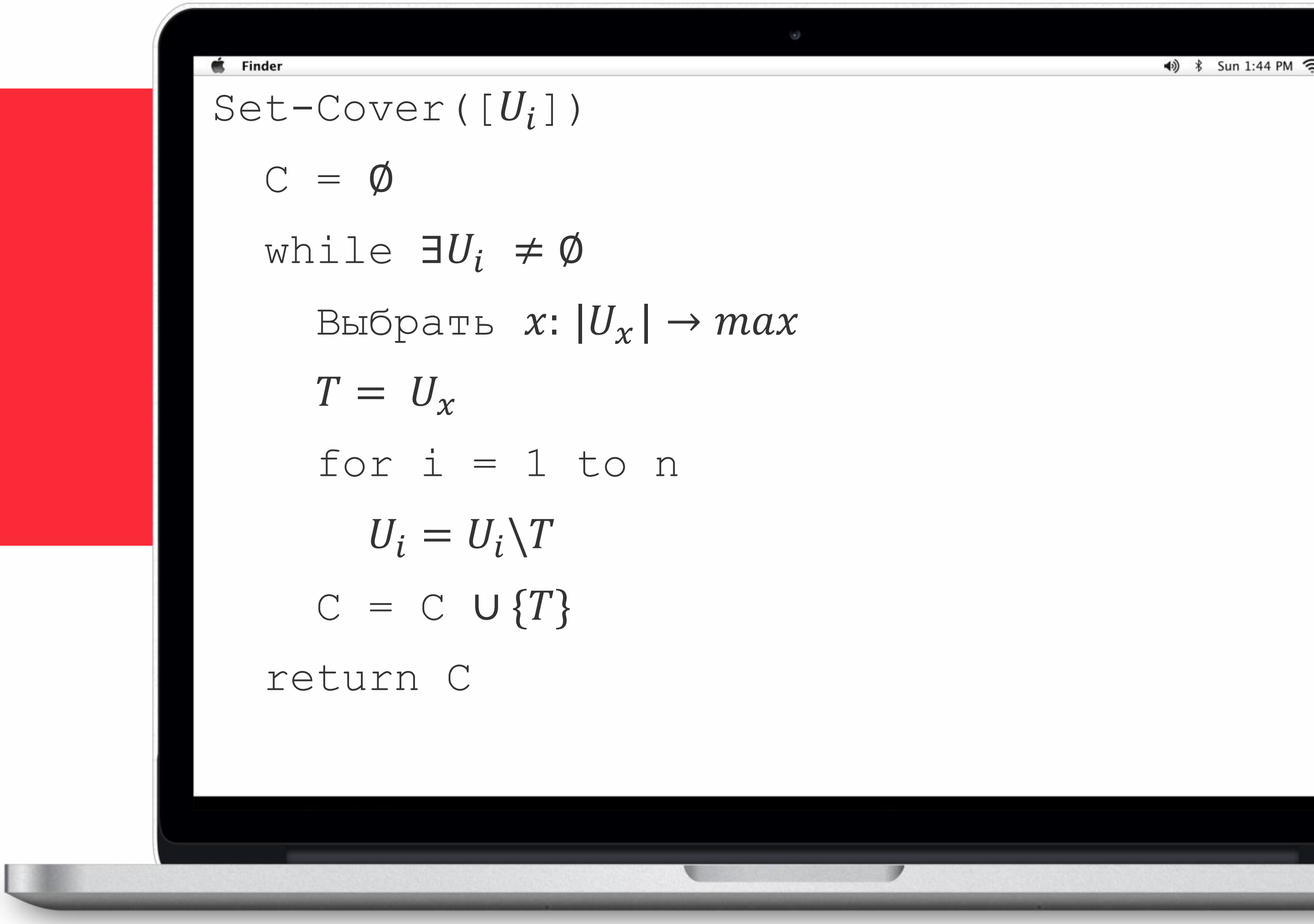
Покрытие множествами

Покрытие множества

- Дано множество U, а также n его подмножеств $U_i \subset U$
  - $\bigcup_{i=1}^{n} U_i = U$
- Найти $\mathrm{C} \subset \{1, 2, 3 \ldots \mathrm{n}\}$
  - $\bigcup_{i \in C} U_i = U$
  - $|C| \to min$
- English: Set Cover

Задача о вершинном покрытии

Решение



Анализ

- Множества, которые выбирал алгоритм $A_1, A_2 \dots A_k$
- $x \in A_i \backslash (A_1 \cup A_2 \cup \dots \cup A_{i-1})$
- $c_x = \frac{1}{|A_i \backslash (A_1 \cup A_2 \cup \dots \cup A_{i-1})|}$
- $\sum_{x \in U} c_x = k$
- $A_1^*, A_2^*, \dots . A_{k^*}^*$ – оптимальный ответ
- $\sum_{x \in U} c_x \leq \sum_{i=1}^{k^*} \sum_{x \in A_i^*} c_x$
  - Каждое слагаемое из левой части, хотя бы один раз есть в правой

Анализ

- Лемма: $\sum_{x \in X} c_x \leq \frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{|X|}$
- $|X| = r, X = \{x_1, x_2, \dots x_r\}$
- x_i пронумерованы в порядке удаления
  - x_1 – последний удаленный, x_r – первый
- Докажем: $c_{x_i} \leq \frac{1}{i}$
- На текущей итерации $X = \{x_1, x_2, \dots x_j\}$ и на ней удаляется x_j
  - $|T| \geq j$ (выбираем жадно)
  - $c_{x_i} = \frac{1}{|T|} \leq \frac{1}{j}, c_{x_{i-1}} = \frac{1}{|T|} \leq \frac{1}{j} \leq \frac{1}{j-1} \dots$
- Т.о. $\sum_{x \in X} c_x \leq \frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{r}$

Анализ

- По лемме: $\sum_{x \in A_i^*} c_x \leq \frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{|A_i^*|} = H(|A_i^*|) \leq H(|U|)$
- $\sum_{i=1}^{k^*} \sum_{x \in A_i^*} c_x \leq H(|U|) \times k^* \leq (\ln|U| + 1) \times k^*$
- $k = \sum_{x \in U} c_x \leq \sum_{i=1}^{k^*} \sum_{x \in A_i^*} c_x \leq (\ln|U| + 1) \times k^*$
- Алгоритм $(\ln|U| + 1)$-оптимален

Все!