Анализ алгоритмов

22 января 2021 г.

Содержание

1 Скорость роста длины записи коэффициентов при реализации метода Гаусса (10)

$$\left\|b_{ij}^{r}\right\| \leq (r+1)M + r^2$$

- $||a||$ – длина записи числа a
- M – максимальная длина записи элементов матрицы
- r – ранг матрицы
- $||b_{ij}^{r}||$ – длина записи коэффицента после r-й итерации

2 Представление "длинного" числа в файле (массиве, списке) как числа в системе счисления по модулю p ($p = 1000$, если `integer` 2^{16}, если $p = 10000000$ `longinteger` 2^{32}). Запись из файла. Оценка числа шагов. Вывод в файл. Оценка числа шагов (14)

2.1 Представление длинного числа

Всякое целое неотрицательное число x может быть представлено в m-ичной системе счисления (при $m \geq 2$) в виде $x = m^{k-1}x_0 + m^{k-2}x_1 + \ldots mx_{k-2} + x_{k-1}$. При этом k – длина записи m-ичного представления числа $x, 0 \leq x_i \leq m-1$ при $i = 0, \ldots, k-1$

2.2 Запись из файла

Оценка числа шагов: квадратичное от длины записи исходного числа в файле количество "шагов".

Пусть в файле записано десятичное число, заданное словом $a_1 \cdots a_n$ $(0 \leq a_i \leq 9)$. Требуется представить его динамическим массивом (или списком).

Под "шагом" понимается одна из следуюших операций: считывание цифры из файла, запись цифры в целочисленный массив, выделение первой цифры многозначного числа и её удаление из него, приписывание цифры в конец числа. Заметим, что эти "шаги" не равнозначны, т.к. последние два требуют нахождения остатка от деления на 10, а также умножения на 10 и сложения.

2.3 Вывод в файл

Оценка числа шагов: линейное от длины записи исходного числа количество "шагов".

При выводе числа необходимо помнить, что в каждом элементе массива, в котором хранится многоразрядное число, записана не последовательность цифр, а число, записанное этими цифрами. Поэтому число, десятичная запись которого меньше, чем длина записи выбранного нами основания m, необходимо дополнить ведущими нулями.

Под "шагом" будем понимать одну из следующих операций: запись "макроцифры" в символьную переменную, сравнение длины записи "макроцифры" с $\|m-1\|$, дополнение строки ведущим нулём.

3 Сложение двух "длинных" положительных чисел. Оценка числа шагов (17)

Оценка числа шагов: общее число "шагов" при сложении двух неотрицательных чисел не превосходит $3\max\{A[0], B[0]\}+1$, то есть составляет $\mathrm{O}(\max\{A[0], B[0]\})$

Чтобы сложить два неотрицательных многоразрядных числа, записанных в массивы A и B, достаточно последовательно складывать по модулю m числа, записанные в $A[i], B[i]$ и $d[i]$ для $i=1,\ldots,\max\{A[0], B[0]\}$, где $d[1]=0$, при $i>1, d[i]-$ это 1 (если $A[i-1]+B[i-1]+d[i-1]>m$) или 0 в противном случае.

При подсчёте числа шагов в этом разделе под "шагом" понимается одна из следующих операций: вычисление $A[i-1]+B[i-1]+d[i-1] \mod m$, проверка условия $A[i-1]+B[i-1]+d[i-1]>m$ и вычисление $d[i]$

4 Предикаты равенства и неравенств "длинных" положительных чисел. Оценка числа шагов (18)

Оценка числа шагов: если под "шагом" понимать количество сравнений "макроцифр", то обшее число "шагов" такой процедуры не превосходит $A[0]$. В общем случае число "шагов" вычисления каждого из четырёх предикатов не превосходит $\min\{A[0], B[0]\}$.

Оценим число шагов вычисления значений предикатов $x=y$ и $x<y$ для случая, когда $A[0]=B[0]$.

Начиная со старшего разряда (то есть с $A[A[0]]$ и $B[B[0]]$) сравниваем значения чисел в $A[i]$ и $B[i]$ до тех пор, пока они совпадают. Если для некоторого $i_0 A[i_0] \neq B[i_0]$, то $x \neq y$. Если при этом $A[i_0] < B[i_0]$, то $x<y$, если $A[i_0] > B[i_0]$, то $x>y$.

5 Вычитание двух "длинных" положительных чисел. Оценка числа шагов (18)

Оценка числа шагов: общее число "шагов" при вычитании двух положительных чисел не превосходит $4\max\{A[0], B[0]\}+1$, то есть составляет $\mathrm{O}(\max\{A[0], B[0]\})$.

При подсчёте числа шагов в этом разделе под "шагом" понимается одна из следуюшци операций: вычисление $A[i-1]-B[i-1]-d[i-1] \pmod m$, проверка условия $A[i-1]+B[i-1]+d[i-1]>0$ и вычисление $d[i]$. Кроме того, предварительно проверяется условие $x \geq y$.

6 Умножение "длинного" числа на короткое. Оценка числа шагов (19)

Оценка числа шагов: общее число операций не превосходит $\max\{1, 2+6A[0]+ \max\{1,3\}\} = 5+6A[0]$, то есть составляет $\mathrm{O}(A[0])$.

Здесь под шагом будем понимать одну из следуюших операций: умножение макроцифр, сложение макроцифр, вычисление неполного частного и остатка от деления результата предыдущих операций на m. В условном операторе после else выполняется одно присваивание и оператор цикла, в котором (помимо двух операций, необходимых для организации цикла) производятся: умножение, сложение, остатка от деления на m, вычисление неполного частного. Всего в операторе цикла 6 "шагов".

7 Умножение "длинных" чисел. Оценка числа шагов (20)

Оценка числа шагов: В предположении, что $B[0] \leq A[0]$ (это условие проверяется за 1 «шаг» и в противном случае можно умножать B на A), получаем оценку $O(A[0]B[0])$.

8 Деление "длинных" чисел. Оценка числа шагов (21)

Оценка числа шагов: $O(A[0]B[0] \cdot (A[0] - B[0]))$

Будем подбирать неполное частное от деления чисел x и y, записанных в массивах A и B, делением промежутка, в котором оно может находиться, пополам. Пусть L и U – нижняя и верхняя границы промежутка соответственно, $M = \left\lfloor \frac{L+U}{2} \right\rfloor$ – целая часть середины промежутка, $z = y \cdot M-$ число, которое будем сравнивать с делимым.

При этом будем предполагать, что число, записанное в A, больше числа, затисанного в B (в противном случае неполное частное равно 0, а остаток совпадает с делимым).

9 Оценки числа шагов метода Гауса при действиях с "длинными" числами

Итоговая асимптотика: $O(\min(n,m) \cdot nm)$

При $n = m$ эта оценка превращается в $O(n^3)$ Для длинных чисел получается $O(M * n^3)$.

10 Сортировки и оценки числа их шагов: Пузырёк. Сортировка вставками. Сортировка слияниями фон Неймана (25)

10.1 Пузырёк

Сложность: $O(n^2)$

В теле циклов сравниваются значения $a[i]$ и $a[j]$. В случае необходимости содержание элементов массива меняются местами. Обмен значениями переменных x и y можно осуществить с помошью трёх операторов присваивания с использованием вспомогательной переменной

$z : z := x; x := y; y := z$.[3] Таким образом, в теле цикла каждый раз выполняется не более четырёх операций.

10.2 Сортировка вставками

Сложность: $O(n^2)$

Элементы входной последовательности просматриваются по одному, и каждый новый поступивший элемент размещается в подходящее место среди ранее упорядоченных элементов.

10.3 Сортировка слияниями фон Неймана

Сложность: $O(n \log_2 n)$

Сортируемый массив разбивается на две части примерно одинакового размера; Каждая из получившихся частей сортируется отдельно, например — тем же самым алгоритмом; Два упорядоченных массива половинного размера соединяются в один.

11 Алгоритмы на графах, различные способы представления графа в компьютере (28)

- V – произвольное конечное множество;
- E – подмножество множества двуэлементных подмножеств множества V;
- $G = (V, E)$ – граф с множеством вершин V и множеством рёбер E;
- A – подмножество множества упорядоченных пар множества V;
- $G = (V, A)$ – орграф с множеством вершин V и множеством дуг A;
- n – количество вершин в графе;
- m – количество рёбер в графе;
- $N(v)$ – окружение вершины v, т.е. множество вершин, смежных с v;
- $OUT(v)$ – множество вершин орграфа, непосредственно достижимых из v;
- $IN(v)$ – множество вершин орграфа, из которых v непосредственно достижима;
- $\deg(v)$ – степень вершины v, т.е. количество вершин в окружении;
- w_{ij} – вес ребра $\{v_i, v_j\}$ или ребра (v_i, v_j) во взвешеном графе.

11.1 Матрица смежности

Матрица смежности графа – это квадратная матрица $A_{n \times n}$, элементы которой определены так:

$$a_{ij} = \begin{cases} 1 & \text{если } \{v_i, v_j\} \in E \\ 0 & \text{иначе} \end{cases}$$

11.2 Списки смежности

Списки смежности – это одномерный массив, i-ым элементом которого является список вершин, смежных с v_i, т.е. окружение вершшны v_i.

11.3 Матрица инцидентности

Матрица инцидентности графа – это матрица $B_{n\times m}$, элементы которой определены так:

$$b_{ij} = \begin{cases} 1 & \text{если } v_i \in e_j \\ 0 & \text{иначе} \end{cases}$$

12 Алгоритм поиска в глубину. Оценки числа шагов в зависимости от способа представления графа (28)

- **Матрица смежности:** $O\left(n^2\right)$ При обходе графа в глубину или в ширину для каждой вершины необходимо проверить все (при обходе в глубину - постепенно, а при обходе в ширину - сразу) вершины, смежные с данной. При использовании матрицы смежности для одной вершины это можно сделать за n проверок того, следует ли помешать вершины в стек или в очередь. Эта процедура выполняется для каждой из n вершин. Этим объясняется то, что алгоритмы, основанные на обходе графа в глубину или в ширину при представлении графа матрицей смежности, имеют оценку числа шагов вида $O\left(n^2\right)$.

  Для орграфа элементы матрицы смежности определяются так

  $$a_{ij} = \begin{cases} 1 & \text{если } (v_i, v_j) \in A \\ 0 & \text{иначе} \end{cases}$$

  Рассуждениями, аналогичными таковым для не ориентированного графа, получаем оценку числа шагов вида $O\left(n^2\right)$.

- **Списки смежности:** $O(n+m)$. Для графов с разными свойствами эта оценка может быть видоизменена.

  Если граф является дереном или лесом, то $m < n$ и оценка принимает вид $O(n)$.

  Ecru граф полный, то $m = \frac{n(n-1)}{2}$ и оценка принимает вид $O\left(n^2\right)$. Если степени всех вершин графа не превосходят некоторой константы C (сушественно меньшей, чем n), то $m \leq Cn$ и оценка принимает вид $O(n)$.

  Если граф связен и степени вершин произвольны, то $m \geq n-1$ и оценка принимает вид $O(m)$.

  Для орграфа список смежности для вершины v состоит из вершин, входяших в $OUT(v)$ (или в $IN(v)$), Поскольку $\sum_{v\in V} \|OUT(v)\| = \sum_{v\in V} \|IN(v)\| = m$, то рассуждениями, аналогичными для не ориентированного графа, получаем оценку числа шагов вида $O(n+m)$.

- **Матрица инцидентности:** $O(nm)$

13 Алгоритм поиска в ширину. Оценки числа шагов в зависимости от способа представления графа (28)

- **Матрица смежности:** $O\left(n^2\right)$ При обходе графа в глубину или в ширину для каждой вершины необходимо проверить все (при обходе в глубину - постепенно, а при обходе в ширину - сразу) вершины, смежные с данной. При использовании матрицы смежности для одной вершины это можно сделать за n проверок того, следует ли помешать вершины в стек или в очередь. Эта процедура выполняется для каждой из n вершин. Этим объясняется то, что алгоритмы, основанные на обходе графа в глубину или в ширину при представлении графа матрицей смежности, имеют оценку числа шагов вида $O\left(n^2\right)$.

  Для орграфа элементы матрицы смежности определяются так

  $$a_{ij} = \begin{cases} 1 & \text{если } (v_i, v_j) \in A \\ 0 & \text{иначе} \end{cases}$$

  Рассуждениями, аналогичными таковым для не ориентированного графа, получаем оценку числа шагов вида $O\left(n^2\right)$.

- **Списки смежности:** $O(n+m)$. Для графов с разными свойствами эта оценка может быть видоизменена.

  Если граф является дереном или лесом, то $m < n$ и оценка принимает вид $O(n)$.

  Ecru граф полный, то $m = \frac{n(n-1)}{2}$ и оценка принимает вид $O\left(n^2\right)$. Если степени всех вершин графа не превосходят некоторой константы C (сушественно меньшей, чем n), то $m \leq Cn$ и оценка принимает вид $O(n)$.

  Если граф связен и степени вершин произвольны, то $m \geq n-1$ и оценка принимает вид $O(m)$.

  Для орграфа список смежности для вершины v состоит из вершин, входяших в $OUT(v)$ (или в $IN(v)$), Поскольку $\sum_{v \in V} \|OUT(v)\| = \sum_{v \in V} \|IN(v)\| = m$, то рассуждениями, аналогичными для не ориентированного графа, получаем оценку числа шагов вида $O(n+m)$.

- **Матрица инцидентности:** $O(nm)$

14 Задачи, решаемые с помощью этих алгоритмов: выделение компонент связности; проверка на двудольность и выделение долей; выделение остова графа

14.1 Выделение компонент связности

Компонента связности графа G – максимальный (по включению) связный подграф графа G. Другими словами, это подграф $G(U)$, порождённый множеством $U \subseteq V(G)$ вершин, в котором для любой пары вершин $u, v \in U$ в графе G существует (u, v)-цепь и для любой пары вершин $u \in U, w \notin U$ не существует (u, w)-цепи.

Для выделения компонент связности можно использовать поиск в ширину или поиск в глубину. При этом затраченное время будет **линейным** от суммы числа вершин и числа рёбер графа.

14.2 Проверка на двудольность и выделение долей

Двудольный граф или биграф в теории графов – это граф, множество вершин которого можно разбить на две части таким образом, что каждое ребро графа соединяет какую-то вершину из одной части с какой-то вершиной другой части, то есть не существует рёбер между вершинами одной и той же части.

Для того, чтобы проверить граф на предмет двудольности, достаточно в каждой компоненте связности выбрать любую вершину и помечать оставшиеся вершины во время обхода графа (например, поиском в ширину) поочерёдно как чётные и нечётные. Если при этом не возникнет конфликта, все чётные вершины образуют множество U, а все нечётные – V.

14.3 Выделение остова графа

Остовное дерево графа – это дерево, подграф данного графа, с тем же числом вершин, что и у исходного графа. Неформально говоря, остовное дерево получается из исходного графа удалением максимального числа рёбее, входящих в циклы, но без нарушения связности графа. остовное дерево включает в себя все n вершин исходного графа и содержит $n-1$ ребро.

Остовное дерево может быть построено практически любым алгоритмом обхода графа, например поиском в глубину или поиском в ширину. Оно состоит из всех пар рёбер (u, v), таких, что алгоритм, просматривая вершину u, обнаруживает в её списке смежности новую, не обнаруженную ранее вершину v.

15 Нахождение остова минимального веса. Метод Р. Прима. Оценки числа шагов (32)

Оценки числа шагов работы этого алгоритма аналогичны оценкам числа шагов алгоритма Дейкстры за исключением того, что в п. 4 не выполняется операция сложения. В результате получаем

$$O\left(n^2 + m\right) = O\left(n^2\right)$$

16 Алгоритм Дейкстры поиска кратчайшего пути. Оценки числа шагов (31)

Сложность: $O(n^2 + m) = O(n^2)$

Задан взвешенный орграф $G = (V, A)$ (все веса w_{ij} положительны) и две выделенные вершины s – старт и f – финиш. Требуется найти кратчайший путь из s в f.

17 Нахождение циклов и мостов в графе. Оценки числа шагов

17.1 Циклы

Замкнутый обход состоит из последовательности вершин, начинающейся и заканчивающейся в той же самой вершине, и каждые две последовательные вершины в последовательности смежны.

Сложность: $O(n+m)$

Неориентированный граф имеет цикл в том и только в том случае, когда поиск в глубину (DFS) находит ребро, которое приводит к уже посещённой вершине (обратная дуга). Таким же образом, все обратные рёбра, которые алгоритм DFS обнаруживает, являются частями циклов. Для неориентированных графов требуется только время $O(n)$ для нахождения цикла в графе с n вершинами, поскольку максимум $n-1$ рёбер могут быть рёбрами дерева.

17.2 Мосты

Мост – ребро в теории графов, удаление которого увеличивает число компонент связности. Эквивалентное определение – ребро является **мостом** в том и только в том случае, если оно не содержится ни в одном цикле.

Сложность: $O(n+m)$

18 Эйлеров цикл. Оценки числа шагов

Эйлеров путь – это путь в графе, проходящий через все его рёбра.

Эйлеров цикл – это эйлеров путь, являющийся циклом.

Граф называется эйлеровым, если он содержит эйлеров цикл. Мультиграф - граф, в котором разрешается присутствие кратных рёбер. Асимптотика задачи поиска эйлерова пути O(m) при использовании списков смежности.

19 Гамильтонов цикл. Оценки числа шагов (33)

Гамильтонов цикл – цикл , который проходит через каждую вершину данного графа ровно по одному разу.

Гамильтонов граф – граф, содержащий гамильтонов цикл.

Асимптотика задачи нахождения гамильтонова цикла в графе O (d^(n − 1)). Где d это максимальная степень вершины в графе - 1.

20 Алгоритм генерации всех независимых множеств. Оценки числа шагов (не будут спрашивать)

21 Теорема о НМ, ВП, КЛИКА. Оценки числа шагов (34)

Алгоритм для поиска клик подойдет для поиска НМ. Ищем клики в G (дополнение), в G это будут независимые множества Алгоритм Брона-Кербоша – метод ветвей и границ для поиска всех клик. Вычислительная сложность алгоритма линейна относительно количества клик в графе. В худшем случае алгоритм работает за $(3^{n/3})$ шагов.

Определение. *Множество V' называется вершинным покрытием графа $G = (V, E)$, если всякое ребро графа инцидентно вершине из V'.*

$$\forall uv(\{u, v\} \in E \to (u \in V' \vee v \in V'))$$

Определение. *Множество V' называется независимым множеством графа $G = (V, E)$, если никакие две вершины из V' не смежны.*

$$\forall uv((u \in V' \ \& \ v \in V') \to \{u, v\} \notin E)$$

Определение. *Множество V' называется кликой в графе $G = (V, E)$, если любые две вершины из V' смежны.*

$$\forall uv((u \in V' \ \& \ v \in V') \to \{u, v\} \in E)$$

Теорема 2.2.1. *Следующие утверждения равносильны:*

1. *V' является вершинным покрытием в $G = (V, E)$;*
2. *$V \setminus V'$ является независимым множеством в $G = (V, E)$;*
3. *$V \setminus V'$ является кликой в $\overline{G} = (V \setminus V', \overline{E})$;*

22 Отличия между интуитивным и математическим понятиями алгоритма

(ИНТУИТИВНОЕ ОПРЕДЕЛЕНИЕ) Первые попытки дать математическое определение алгоритма привели приблизительно к следующим требованиям:

- Определенность данных: вид исходных данных строго определен.
- Дискретность: процесс разбивается на отдельные шаги.
- Детерминированность: результат каждого шага строго определjн в зависимости от данных, к которым он применен
- Элементарность шага: переход на один шаг прост.
- Направленность: что считать результатом работы алгоритма, если следующий шаг невозможен.
- Массовость: множество возможных исходных данных потенциально бесконечно

Первыми математическими понятиями алгоритма были рекурсивные функции и машины Тьюринга.

Определение. Простейшими называются функции натурального аргумента S, O, ${I_m}^n$, определяемые равенствами: S(x) = x + 1, O(x) = 0, ${I_m}^n(x_1, ..., x_n) = x_m$ при $1 \leq m \leq n$.

Определение. Функция f от n + 1 переменных получена из функции g от n переменных и функции h от n + 2 переменных с помощью оператора примитивной рекурсии, если

$$\begin{cases} f(x_1, \ldots, x_n, 0) = g(x_1, \ldots, x_n) \\ f(x_1, \ldots, x_n, y+1) = h(x_1, \ldots, x_n, y, f(x_1, \ldots, x_n, y)) \end{cases}$$

Определение. Функция называется примитивно рекурсивной, если она может быть получена из простейших с помощью применения операторов подстановки и/или примитивной рекурсии.

Определение. Функция называется частично рекурсивной, если она может быть получена из простейших с помощью применения операторов подстановки, примитивной рекурсии и/или μ-оператора.

Определение. Функция называется общерекурсивной, если она может быть получена из простейших с помощью применения операторов подстановки, примитивной рекурсии и/или обобщенного μ-оператора.

23 Машины Тьюринга и их модификации. Тезис Тьюринга-Чёрча

Машины Тьюринга, основные определения, стр. 40-41

Тезис Тьюринга-Черча Всякая интуитивно вычислимая функция может быть вычислена на машине Тьюринга.

Модификации машин Тьюринга — стр. 45-47

24 Теорема о числе шагов МТ, моделирующей работу k-ленточной МТ

Теорема. По всякой k-ленточной машине Тьюринга MTk, заканчивающей работу с исходными данными X за t шагов, можно построить одноленточную машину Тьюринга MT1, результат работы которой с исходными данными X совпадает с результатом работы исходной и число шагов которой составляет $O(t^2)$.

25 Недетерминированные МТ. Теорема о числе шагов МТ, моделирующей работу недетерминированной МТ

Недетерминированные машины Тьюринга часто используются для проверки истинности утверждений типа $\exists Y P(X, Y)$ (существует такой объект Y , для которого справедливо утверждение P(X, Y)). Для проверки истинности таких утверждений работу недетерминированной машины Тьюринга можно разбить на два этапа:

- этап угадывания, при реализации которого лента в недетерминированном режиме размножается, и на каждой из них выписывается sпретендентї на решение;
- этап проверки, при реализации которого машина работает в детерминированном режиме и проверяет конкретного "претендента"на то, является ли он решением

Теорема. По всякой недетерминированной машине Тьюринга, проверяющей предикат $\exists P(X, Y)$ и заканчивающей работу с исходными данными X за t шагов, можно построить одноленточ- ную машину Тьюринга, результат работы которой с исходными данными X совпадает с результатом работы исходной и число шагов которой составляет $2^{O(t)}$

26 Понятия сложности алгоритма от данных, сложность алгоритма, сложность задачи. Верхняя и нижняя оценки сложности

Под вычислительной сложностью алгоритма понимают функцию, зависящую от ДЛИНЫ записи исходных данных и характеризующую

- число шагов работы алгоритма над исходными данными (временная сложность);
- объем памяти, необходимой для работы алгоритма над исходными данными (емкостная или зональная сложность).

Определение. Сложностью $S_A(P)$ алгоритма A при работе над данными P называется число шагов или объем памяти, затра- ченные в процессе работы алгоритма A над данными P.

Определение. Верхней (нижней) оценкой сложности алгоритма A при работе над данными длины n называется $S_A^U(n) = \max_{P:\|P\|=n} \{S_A(P)\}$

27 Соотношение между временем работы алгоритма требуемой памятью

28 Классы алгоритмов и задач. Схема обозначений

Класс функций или предикатов	Детерминированная или недетерминированная	Функция сложности	Временна́я или ёмкостная
F	[D] N	LOG LIN QLIN P (Poly) EXP-LIN EXP ⋮	[TIME] SPACE

Определение. Функция сложности это функция от длины записи исходных данных, ограничивающая число шагов или количество ячеек соответствующей машины Тьюринга.

29 Классы P, NP и $P-SPACE$. Соотношения между этими классами

Определение. Класс P это класс предикатов, для которых существует алгоритм, который может быть реализован на детерминированной машине Тьюринга, число шагов которой не превосходит полинома от длины записи исходных данных.

Определение. Класс NP это класс предикатов, для которых существует алгоритм, который может быть реализован на недетерминированной машине Тьюринга, число шагов которой не превосходит полинома от длины записи исходных данных.

Определение. Класс P-SPACE это класс предикатов, для которых существует алгоритм, который может быть реализован на детерминированной машине Тьюринга, число использованных ячеек которой не превосходит полинома от длины записи исходных данных.

$P \subseteq NP \subseteq P - SPACE \subseteq EXP$

Теорема. Если предикат вида $\exists Y P(X, Y)$ принадлежит классу NP, то существует проверяющая его одноленточная машина Тьюринга, число шагов которой составляет $2^{p(n)}$, где p(n) полином от длины записи исходных данных $n = size(X)$.

30 Полиномиальная сводимость и полиномиальная эквивалентность

Определение. Определение. Задача Z_1 вида $\exists Y P_1(X, Y)$ при $X \in D_1$ полиномиалъно сводится к задаче Z_2 вида $\exists Y P_2(X, Y)$ при $X \in D_2$

$$Z_1 \propto Z_2$$

если сушествует функиия f, отображающая D_1 в D_2 и такая, что

- существует машина Тьюринга, вычисляющая функцию f не болеечем за полиномиальное от длиньи записи исходных данных число шагов ($f \in \mathbf{FP}$)
- задача Z_1 имеет решение с исходными данными X тогда и только тогда, когда задача Z_2 имеет решение с исходными данными $f(X)$

$$\forall X_{\in D_1} \left(\exists Y P_1(X,Y) \leftrightarrow \exists Y P_2(f(X),Y)\right)$$

Определение. Задача Z_1 полиномиально эквивалентна задаче Z_2 $(Z_1 \sim_p Z_2)$, если $Z_1 \propto Z_2$ и $Z_2 \propto Z_1$

31 Полиномиальная сводимость задачи ГЦ к задаче КОМИВОЯЖЁР (стр. 80)

32 Классы эквивалентности по отношению полиномиальной эквивалентности. Класс P – пример такого класса (стр. 81-82)

33 NP-полные задачи. Класс NP-полных задач — класс эквивалентности по отношению полиномиальной эквивалентности (стр. 81-82)

34 Задача ВЫПОЛНИМОСТЬ (ВЫП).Теорема Кука

Дано: $U = \{u_1, ..., u_n\}$ множество пропозициональных переменных,

$C = \{c1, ..., cm\}$ множество предложений над U.

Вопрос: выполнимо ли множество C, т.е. существует ли набор значений для переменных из U, для которого истинны все предложения из C?

$\exists u1, ..., un(c1\&...\&cm)$

Теорема. Задача ВЫП NP-полна.

35 Задача 3-ВЫПОЛНИМОСТЬ (3-ВЫП). Её NP-полнота (84)

Пусть U - множество пропозициональных переменных. C - множество предложений над U, где каждое из них состоит из трех переменных. Требуестя определить, существует ли такой набор значений переменных, что $c_1\&...\&c_m$ выполнимо.

36 Задачи ВЕРШИННОЕ ПОКРЫТИЕ (ВП), НЕЗАВИСИМОЕ МНОЖЕСТВО (НМ), КЛИКА. NP-полнота задачи ВП. Полиномиальная эквивалентность этих трёх задач (85-86)

Определение вершинного покрытия – множество вершин такое, что для каждого ребра из заданного графа хотя бы один из его концов содержится в этом множестве.

Определение клики Клика - полный подграф исходного графа

Определение независимого множества Независимое множество – подмножество вершин графа такое, что для между любой парой вершин из этого множества нет ребра в исходном графе.

Вершинное покрытие Дан граф и число k, требуется узнать есть ли в графе вершинное покрытие не более чем из k элементов.

Клика Дан граф и число k, требуется узнать есть ли в графе клика не менее чем из k вершин.

Независимое множество Дан граф и число k, требуется узнать есть ли в графе Независимое множество не менее чем из k элементов.

37 NP-полнота задач ГЦ и ГП (без доказательства)

Гамильтонов цикл – NP трудная задача как и Гамильтонов путь.

38 NP-полнота задач 3-С и РАЗБИЕНИЕ (без доказательства)

Задача 3-С (трехмерное сочетание): Дано множество $M = X \times Y \times Z$. X,Y,Z - попарно не пересекающиеся множества мощности q. Требуется узнать существует ли подмножество $M' \in M$ мощности q, такое, что у никаких двух элементов из этого множества значения хотя бы по одной из координат совпадают.

Задача Разбиение: Дано множество A, для каждого $a \in A$ задан вес $s(a)$ – целое положительное число. Требуется разбить исходное множество на два подмножества одинакого веса.

39 Метод сужения доказательства NP-полноты (89-90)

Определение Задача Z_1 называется сужением на множество D_1 задачи Z_2 с исходными данными из множества D_2, если D_1 содержится в D_2 и $\forall X(X \in) D_1 \to (Z_1 \leftrightarrow Z_2)$

Метод сужения: Чтобы доказать NP-полноту задачи Z достаточно доказать, что $Z \in NP$ и что среди известных NP-полных задач найти такую задачу Z_1, что она является сужением задачи Z

40 «Похожие» задачи и их сложность

Этого говна не будет

41 Анализ подзадач (90-91)

Определение Задача Z_1 с исходными данными D_1 является подзадачей Z_2 с исходными данными из множества D_2, если D_1 содержится в D_2 и $\forall X(X \in) D_1 \to (Z_1 \leftrightarrow Z_2)$

Например 3-SAT задача подзадача SAT задачи.

42 Алгоритм решения задачи РАЗБИЕНИЕ

Решение 1: Общая идея: рекурсия. Подсчитаем сумму всего массива – если она нечетная, то задачу не решить. Если она четная, то ок – решаем рекурсивно. Каждый элемент может быть отнесен к одному из двух множеств. Попробуем решить задачу, нахождения множества с суммой равной половине исходной. Для этого рекурсивно будем считать достигается ли эта сумма, если выкинуть или не выкидывать последний элемент из текущего множества. Асимптотика – $\mathcal{O}(n2^n)$

Решение 2: Воспользуемся методом динамического программирования.

Пусть $dp[i][j]$ – ответ на вопрос: правда ли, что сумма на j-ом префиксе равна i? (0/1 = да/нет)

Пересчет дп – $\mathcal{O}(n * sum)$, что может быть долго, если сумма достаточно большая.

43 Задачи с числовыми параметрами. Псевдополиномиальные задачи (92)

Определение. Задача с числовыми параметрами называется псевдополиномиальной, если число шагов решающей ej машины Тьюринга не превосходит полинома от этих числовых параметров и длины записи остальных исходных данных.

Казалось бы алгоритм задачи разбиение, использующий метод динамического программирования, работает за полином, но из-за скорости роста данных он на самом деле Псевдополиномиальный