Точные алгоритмы и открытые проблемы

Лекция N 2 курса “Современные задачи теоретической информатики”

Юрий Лифшиц
yura@logic.pdmi.ras.ru

ИТМО

Осень’2005

Закон Хоара (о больших задачах) : "Внутри каждой большой задачи сидит маленькая, которая старается выбраться наружу"

Многие люди думают, что это значит:

Большие задачи на самом деле маленькие

Что имеется в виду:

Большие задачи составлены из меньших задач

Предостережение:

Найти маленькую задачу в центре большой — недостаточно для решения — это только начало исследований...

План лекции

1. Еще NP-полные задачи
2. От табличных сумм к сумме размеров
   - Табличные суммы
   - Применение к сумме размеров
3. Задача о клике и алгоритм Вильямса
   - Поиск треугольников
   - Алгоритм Вильямса для максимального разреза

План лекции

1. Еще NP-полные задачи
2. От табличных сумм к сумме размеров
   - Табличные суммы
   - Применение к сумме размеров
3. Задача о клике и алгоритм Вильямса
   - Поиск треугольников
   - Алгоритм Вильямса для максимального разреза

Сумма размеров

Данные:
Натуральные числа $w_1, \ldots, w_n$ и натуральное s

Выяснить:
Существует ли подмножество $w_1, \ldots, w_n$, сумма элементов которого дает ровно s?

Пример:

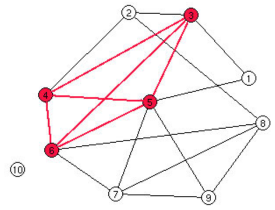
17 43 23 38 14 20 36 47 ; 100

Есть ли подмножество с суммой 100?

Задача о клике

Данные:
Простой неориентированный граф G с n вершинами и число $k \leq n$

Выяснить:
Есть ли в G полный подграф из k вершин?



Максимальный разрез

Данные:
Граф G с n вершинами со взвешенными ребрами

Найти:
Разбиение вершин G на две части с максимальной суммой весов ребер между частями

Факт: задача о клике, максимальный разрез и сумма размеров являются **NP**-полными

План лекции

1. Еще NP-полные задачи
2. **От табличных сумм к сумме размеров**
   - Табличные суммы
   - Применение к сумме размеров
3. Задача о клике и алгоритм Вильямса
   - Поиск треугольников
   - Алгоритм Вильямса для максимального разреза

Табличная-2-сумма

Данные:
Таблица $2 \times n$ заполненная числами и число s

Выяснить:
Есть ли число из первой строчки и число из второй в сумме дающие s?

Тупой алгоритм: n^2

Кто может лучше?

Решение за $O(n \log n)$:
Отсортировать одну строку и делать цикл по другой

Табличная-k-сумма

Данные:
Таблица $k \times n$ заполненная числами и число s

Выяснить:
Существует ли набор из k чисел, по одному из каждой строчки, в сумме дающие s?

Тупой алгоритм: n^k

Кто может лучше?

Поделим пополам: $n^{\lceil k/2 \rceil} \log n$ (сведение к 2-сумме)

Улучшение для k=3: n^2

Быстрее чем n^2 для 3-табличной суммы?

Алгоритм Шреппеля-Шамира

Алгоритм для табличной-4-суммы
Требования по времени $O(m^2 \log m)$, по памяти $O(m)$

Связь с геометрией

Задача "Три-точки-на-прямой"
Данные:
Координаты n точек на плоскости
Определить:
Лежат ли какие-нибудь 3 из них на одной прямой?

Утверждение: табличную-3-сумму с одинаковыми строками можно свести к "трем-точкам-на-прямой"

Более формально: Если существует алгоритм для решения "трех-точек-на-прямой" трудоемкости $f(n)$, то есть и алгоритм для табличной-3-суммы трудоемкости $c \cdot f(n)$

Связь с геометрией II

Конструкция:

Пусть строка - $x_1, \ldots, x_n$, требуемая сумма s

Отметим на графике $f(x) = x^3 - sx^2$ точки

$$(x_1, f(x_1)), \ldots, (x_n, f(x_n))$$

Если три из них (i, j, k) лежат на какой-то прямой $y = ax + b$, то x_i, x_j, x_k являются корнями уравнения $x^3 - sx^2 - ax - b = 0$ т.е. $x_i + x_j + x_k = s$.

Верно и обратное: если $x_i + x_j + x_k = s$, то $(x_1, f(x_1)), (x_2, f(x_2)), (x_3, f(x_3))$ лежат на прямой

Где я вас обманул?

Применение к сумме размеров

Разделим на две части, сосчитаем все подмножества в частях, получим 2-табличную сумму размера $2^{n/2}$

Недостаток: требование по памяти — $2^{n/2}$

$$\underbrace{17 \quad 43 \quad 23 \quad 38} \quad \underbrace{14 \quad 20 \quad 36 \quad 47} \; ; 100$$

$$\begin{array}{l} \{17, 43, 23, \\ 38, 60, 40, 55, \\ 66, 81, 61, 83, \\ 98, 78\} \end{array} \qquad \begin{array}{rcl} 100 - 14 & = & 86 \\ 100 - 20 & = & 80 \\ 100 - 36 & = & 64 \\ 100 - 47 & = & 53 \\ 100 - 34 & = & 66 \\ & \vdots & \end{array}$$

Хитрее:

Разделить на четыре части

Применить алгоритм Шреппеля-Шамира

Время работы — $2^{n/2}$, но по памяти — $2^{n/4}$

Открытые вопросы

Построить алгоритм для табличной-6-суммы, работающий за $O(m^3 \log m)$, и требующий $O(m)$ памяти

Построить алгоритм для табличной-k-суммы, работающий за $O(m^{\lceil k/2 \rceil - \varepsilon})$

Построить алгоритм для суммы размеров, работающий за $O(1.4^n)$

Построить алгоритм для суммы размеров, работающий за $O(1.99^n)$, но *полиномиальный* по памяти

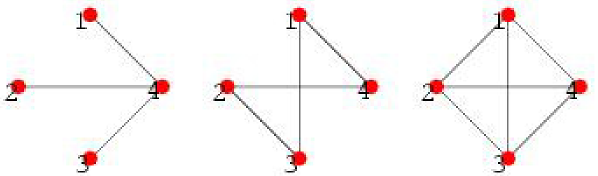
План лекции

1. Еще NP-полные задачи
2. От табличных сумм к сумме размеров
   - Табличные суммы
   - Применение к сумме размеров
3. **Задача о клике и алгоритм Вильямса**
   - Поиск треугольников
   - Алгоритм Вильямса для максимального разреза

Поиск треугольников

Найти 3-клику = найти треугольник

Матрица смежности — какая вершина с какой соединена



$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

Поиск треугольников II

Возведем матрицу смежности A в квадрат $B = A^2$

$$b_{ij} = \sum a_{ik} \cdot a_{kj}$$

Смысл b_{ij} —
количество путей длины 2 из i-ой вершины в j-ую

Как искать треугольники?

Возвести A в куб и посмотреть на диагональ
Трудоемкость: $n^{2.376}$ (быстрое умножение матриц)

Открытые задачи для k-клик

Лучшие алгоритмы для маленьких клик:

3-клика	4-клика	5-клика	6-клика	7-клика
$n^{2.376}$	$n^{3.334}$	$n^{4.220}$	$n^{4.752}$	$n^{5.714}$

Для клики произвольного размера — $O(1.2108^n)$

Улучшить эти алгоритмы

Существует ли алгоритм $O(n^{7.5})$ для 10-клики?

Верно ли, что 3-клика так же трудна, как и умножение матриц?

MAX-CUT

Идеи:

Разделить вершины на три равные части (A, B, C)
Построить огромный трехдольный граф H размера
$$2^{n/3} \times 2^{n/3} \times 2^{n/3}$$
Взаимнооднозначное соответствие треугольники — разрезы

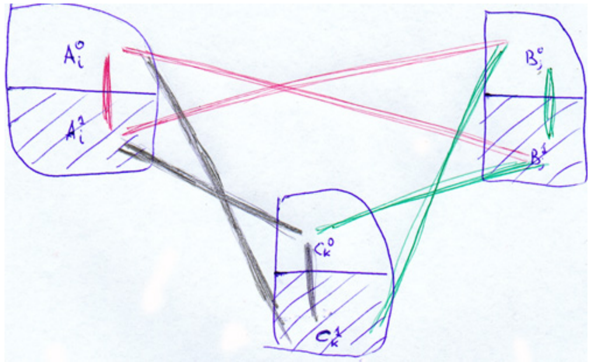
Вершина i в первой доле — разрез A на A_i^0 и A_i^1

Мы хотим так присвоить веса ребрам H чтобы "вес" треугольника (i, j, k) равнялся "весу" разреза
$$A_i^0 \cup B_j^0 \cup C_k^0 \text{ — } A_i^1 \cup B_j^1 \cup C_k^1$$

Вспомогательный граф

Пусть $W(M, N)$ — сумма весов ребер из N в M

Тогда присвоим
ребру (i,j) — вес $W(A_i^0, B_j^1) + W(A_i^1, B_j^0) + W(A_i^0, A_i^1)$
ребру (j,k) — вес $W(B_j^0, C_k^1) + W(B_j^1, C_k^0) + W(B_j^0, B_j^1)$
ребру (k,i) — вес $W(C_k^0, A_i^1) + W(C_k^1, A_i^0) + W(C_k^0, C_k^1)$



Алгоритм Вильямса

Решаем MAX-CUT (найти максимальный разрез)

1. Будем проверять, существует ли разрез больше z (дальше бинарный поиск)
2. Делаем перебор по всем тройкам $z_{ab} + z_{bc} + z_{ca} = z$
3. Оставляем между первой и второй долей H только ребра весом хотя бы z_{ab}, аналогично по другим направлениям
4. Ищем треугольники!

Трудоемкость: $(2^{n/3})^{2.376} = 2^{0.792 \cdot n} < 1.732^n$

Открытые задачи

Придумать алгоритм для MAX-CUT быстрее 1.732^n

Придумать 1.99^n алгоритм для MAX-CUT, но полиномиальный по памяти

Последний слайд

Если не запомните ничего другого:

- Прогресс по **NP**-трудным задачам зависит от эффективных алгоритмов для очень простых задач
- Одинаковый барьер эффективности есть одновременно в табличных суммах и вычислительной геометрии
- Быстрое умножение матриц помогает для оптимизации на графах

Вопросы?