SPb HSE, 1 курс, весна 2024/25
Практика по алгоритмам #23

Приближения и центроиды
19 марта

Собрано 20 марта 2025 г. в 10:37

Содержание

Приближения и центроиды

1. **gcd на отрезке**

   Дан массив целых чисел от 1 до C. Массив не меняется.
   Научитесь в online отвечать на запросы «gcd на отрезке $[L, R]$» за $\mathcal{O}(\log C)$.

2. **Максимальный короткий путь в дереве**

   Дано дерево, каждая вершина имеет неотрицательный вес.
   Среди путей длины ровно L найти путь max веса. $\mathcal{O}(n \log n)$.

3. **Неприближаемость TSP**

   Покажите, что если $\text{P} \neq \text{NP}$, то за полином нельзя найти приближение задачи коммивояжёра на произвольном полном графе. (a) Веса неотрицательны. (b) Веса целые, $\geqslant 1$.

4. **Приближение Bin Packing**

   Пользуясь NP-трудностью задачи Partition, докажите, что $(1.5-\varepsilon)$-приближение задачи Bin Packing NP-трудно.

5. **Приближение Knapsack**

   Докажите, что жадное решение непрерывного рюкзака не является никаким приближением дискретного.

6. **PTAS для Partition**

   Придумайте PTAS для задачи Partition. Что такое PTAS? Что такое partition?

7. **Приближение Set Cover**

   a) Реализуйте жадное $(\ln n)$-приближение максимально быстро. Можно за $\mathcal{O}(|U| + \sum |A_i|)$.
   b) Модифицируйте жадное решение для взвешенного случая:
   каждое множество имеет вес, мы ищем покрытие минимального веса.

8. **Итеративное приближение коммивояжера**

   Задача: Euclidean TSP, строим цикл коммивояжера на точках на плоскости.
   Решение. Начнём с цикла, состоящего из любой одной точки. По одной вставляем в цикл новые точки, каждый раз выбираем точку, ближайшую к циклу: пусть A – вершины цикла, берём $\min_{a \in A, b \in V \setminus A} d(a, b)$. Вставляем b или слева от a, или справа (берём лучшее из двух).

   a) Докажите, что этот алгоритм 2-OPT.
   b) Покажите как его улучшить, чуть изменив принцип выбора b.

9. **Приближённые расстояния в ацикличном графе**

   Дан ацикличный граф, для каждой вершины приближённо найти число достижимых из неё.
   У нас уже есть точное решение за $\mathcal{O}(nm/w)$. Тут хотим приближённо, зато за $\mathcal{O}(n + m)$.

10. **(*) Задача о надстроке**

    Даны строки $s_1, \dots, s_n$, найти строку S min длины: все s_i – подстроки S.

    a) Увидеть в этой задаче Set Cover. $(2\ln n)$-приближение.
    b) Увидеть в этой задаче TSP. 2-приближение.

11. **(*) Пятачок, у тебя есть дома ружье?**

    Даны n непересекающихся кругов на плоскости. Мы стоим в точке $(0,0)$ и можем стелять по прямой. Минимальным числом выстрелов проткнуть все круги.

Разбор задач практики

1. **gcd на отрезке**

   Массив – это дерево. Можно на нем строить Centroid Decomposition. Получится Disjoint Sparse Table. Дальше считать $\gcd[v, d]$ и отвечать на запросы, как обычно: по v, u находить $x\colon u, v \in C(x) \wedge x \in \texttt{path}(u, v)$, отвечать $\gcd(\gcd[u, d_x], \gcd[v, d_x])$. Можно даже за $\mathcal{O}(\log\log n + \log C)$.

2. **Максимальный короткий путь в дереве**

   Переберем центроид x, который будет на пути.
   За `dfs` по $C(x)$ считаем массив $w_x[d]$ – max вес пути до вершины на глубине d, $d \leqslant |C(x)|$.
   Снова `dfs` по $C(x)$. Вот `dfs` стоит в вершине v на глубине d с весом пути s. Обновляем ответ величиной $s + w_x[L - d]$.

   Но это может дать путь с самопересечением.
   Чтобы этого избежать, храним w_x не для всей $C(x)$, а для нескольких рассмотренных поддеревьев.
   Обходя очередное поддерево x, только обновляем ответ.
   Затем обходим поддерево второй раз, обновляя w_x.

   Теперь хотим пути длины $\leqslant L$. Нужно поддерживать префиксные максимумы w_x.
   Способ 1. Сортируем поддеревья x по возрастанию их размеров $C_1, C_2, \ldots$ Обходим их в таком порядке.
   Тоже два обхода каждой C_i. На первом релаксруем ответ, на втором обновляем w_x.
   После второго обхода надо обновить префиксные максимумы. Длина массива w_x сейчас станет $\leqslant |C_i|$, т.е. сделали $\mathcal{O}(|C_i|)$ операций.
   Итого $\mathcal{O}(|C(x)|)$, в сумме $\mathcal{O}(n \log n)$.
   Способ 2. Храним по два максимума в $w_x[d]$.
   Второй max среди путей, ведущих не в то поддерево, где первый max.
   Обходим всю $C(x)$ и считаем w_x.
   Второй раз обходим всю $C(x)$ и обновляем ответ.
   Если max путь пересекается в путем в текущую v, надо брать второй максимум.
   Чтобы определить, есть ли пересечение, запомним в $w_x[d]$ не только вес пути, но и соседа x, с которого начался этот путь. То же самое помним и во втором `dfs`.

3. **Неприближаемость TSP**

   Сведём гамильтонов цикл в неорграфе к приближению коммивояжера. Берём граф. Создаём матрицу весов: нет ребра $\Rightarrow w_{ij} = 1$, есть ребро $\Rightarrow w_{ij} = 0$. $\exists$ гамильтонов цикл $\Rightarrow$ вес для комивояжера $= 0$, иначе 1, что в бесконечность раз больше.

   Если нас просят тест, где все веса положительны, заменим $\langle 0, 1\rangle$ на $\langle 1, Cn + 1\rangle$.

4. **Приближение Bin Packing**

   Такое приближение позволит отличать ответы 2 и 3 для Bin Backing. А это позволит решать задачу Partition.

5. **Приближение Knapsack**

$w_1 = 1,\ c_1 = 2,\ w_2 = S,\ c_2 = S$. Жадность возьмёт первый предмет, оптимально брать второй. Ошибка в $S/2$ раз $\Rightarrow$ не ограничена.

6. **PTAS для Partition**

Переберём за $\binom{n}{k}$, куда кладём k максимальных по весу предметов, остаток разложим жадно (очередного кладём в меньший рюкзак).

Нам нужно минимизировать $f = \max(\sum a_i, \sum b_i)$. При жадной стратегии части $\sum a_i$ и $\sum b_i$ будут отличаться на $\leqslant$ веса одного предмета $w \Rightarrow \text{GREEDY} \leqslant \text{OPT} + \frac{w}{2}$

Пусть мы угадали k максимальных по весу. В один из рюкзаков упало хотя бы $\frac{k}{2}$ из них. $w \leqslant$ этих k весов. Итого: $\frac{k}{2}w \leqslant \text{OPT} \Rightarrow \text{GREEDY} \leqslant \text{OPT}(1+\frac{1}{k})$.

Тогда вес большей части $\leqslant \frac{\sum w_i + w}{2} \leqslant \text{OPT} + \frac{w}{2}$.
Пусть распределили $2k$ предметов оптимально. Если в большую часть больше ничего не добавится, то это оптимум.
Иначе разница между частями, аналогично жадной стратегии, будет не более w_{2k+1}.
Тогда $w_{2k+1} \leqslant \frac{\sum^{2k+1} w_i}{2k+1} \leqslant \frac{\sum w_i}{2k+1} \leqslant \frac{2\text{OPT}}{2k+1} \leqslant \frac{\text{OPT}}{k} \Rightarrow$ погрешность $\leqslant \frac{w/2}{\text{OPT}} \leqslant \frac{1}{2k}\text{OPT}$.

7. **Приближение Set Cover**

Будем считать, что $|A|$ – **число ещё не покрытых элементов A, а не исходный размер**.

a) Предподсчитаем `ids`$[x]$ – номера множеств, содержащих элемент x.
Поддерживаем `size`$[j]$ – число ещё не покрытых элементов в множестве j.
Поддерживаем `sets`$[i]$ – номера множеств, в которых ровно i ещё не покрытых элементов.
При изменении `size`$[j]$ удаляем j из `sets`[`size`$[j]$] лениво (не удаляем).

```
for (int k = n; k >= 1; k--)
  for (int i : sets[k]) // пытаемся выбрать множество, которое покрое ещё k элементов
    if (size[i] == k) // ленивое удаление в действии
      answer.add(i)
      for (int x : A[i]) // все элеметы множества
        if (!used[x]) // тогда нужно пометить элемент
          used[x] = 1
          for (int j : A[x])
            size[j]--
            if (i != j) sets[size[j]].add(j)
```

Каждый элемент x мы удалим ровно один раз $\Rightarrow$ время $\mathcal{O}(n + \sum |A_i|)$.

b) **Алгоритм для взвешенного случая.**
Выбираем каждый раз множество S с минимальной удельной стоимостью $\frac{\texttt{cost}(S)}{|S|}$.
Пусть нужно покрыть n элементов, мы выбрали S. Тогда верно $\frac{\texttt{cost}(S)}{|S|} \leqslant \frac{\text{OPT}}{n}$ (*).
Рассмотрим новую задачу для «ещё не покрытых элементов». Вызовемся для неё рекурсивно. Ответы для неё обозначим OPT_1, GREEDY_1. $\text{OPT}_1 \leqslant \text{OPT}$.
По индукции $\text{GREEDY}_1 \leqslant \ln(n-|S|)\text{OPT}_1 \leqslant \ln(n-|S|)\text{OPT}$.
$\text{GREEDY} \leqslant \text{GREEDY}_1 + |S|\frac{\text{OPT}}{n} \leqslant (\ln(n-|S|) + \frac{|S|}{n})\text{OPT} \leqslant (\ln n)\text{OPT}$.

Доказательство (*): рассмотрим множества $A_1, A_2, \dots, A_k$ в оптимальном ответе.
Уменьшим их, чтобы каждый элемент покрывался ровно одним из A_i.
$\frac{\texttt{cost}(S)}{|S|}$ лучше всех удельных стоимостей новых A_i, а максимум из них больше $\frac{\text{OPT}}{n}$.

8. **Итеративное приближение коммивояжера**

   a) Мы очень похожи на алгоритм Прима. Прим тоже выбирает $\min_{a \in A, b \in V \setminus A} d(a,b)$. Собственно $\sum$ выбранных $d(a,b)$ равно MST. При добавлении b к циклу можно представить, что сперва мы получили непростой цикл $a \to b \to a \rightsquigarrow a$, а затем спрямили (уменьшили) его $\Rightarrow$ мы не хуже $2 \cdot \text{MST} \Rightarrow$ 2-OPT.

   b) Более хорошие способы выбирать b – минимизировать
      1) или расстояние не до вершин цикла, а до рёбер цикла,
      2) или стоимость вставки, то есть, $d(x,b) + d(y,b) - d(x,y)$ при вставке в ребро (x,y).

9. **Приближённые расстояния в ацикличном графе**

   Каждой вершине v сгенерируем случайное $x_v = \text{random}[0..1]$.
   Посчитаем динамику по ацикличному графу m_v – минимальное достижимое x_v.
   Если из v достижимо k_v вершин, $\mathsf{E}[m_v] = \frac{1}{k+1}$. Доказательство:
   $\mathsf{E}[\min(x_1, \dots, x_k)] = k \int x(1-x)^{k-1} dx = -x(1-x)^k|_0^1 + \int (1-x)^k dx = \frac{(1-x)^{k+1}}{k+1}|_1^0 = \frac{1}{k+1}$.

   Итого приближение $k_v = \frac{1}{m_v} - 1$. Чтобы алгоритм работал лучше, запустим его z раз, для каждой v получим список из z возможных k_v, вернём медиану.

10. **(*) Задача о надстроке**

    Для начала в любом случае уберём все подстроки (s_i – подстрока $s_j \Rightarrow$ удаляем s_i).
    Зацеплением строк s_i и s_j называют $\max k$: суффикс s_i и префикс s_j длины k совпадают.
    Заметим, что ответ однозначно задаётся порядком строк.

    a) Построим орграф, $i \to j$, а вес ребра – длина максимального зацепления строк s_i и s_j.
       Нам нужно найти max по весу гамильтонов путь, а ответ на задачу $= \sum_i |s_i| - w(path)$.
       Мы умеем 2-приближать только неориентированного коммивояжера с нер-вом $\triangle$.

       Алгоритм. Пока строк больше одной, взять две с max зацеплением и объединить.
       Умеют доказывать, что это 3.5-приближение (статья в формате `ps`).
       Есть гипотеза, что это 2-приближение. Умеют доказывать для строк длины $\leqslant 4$ (статья).

    b) $2 \ln n$-приближение. Стр. 20 в книге Вазирани.
       Посмотрим на все возможные сцепления пар строк: $\sigma_{i,j,k} = s_i + s_j[k{:}]$ при $s_i[{:}{-k}] = s_j[k{:}]$.
       Например, $s_i = xabab,\ s_j = ababcc,\ k = 2 \Rightarrow \sigma_{i,j,k} = xabab\,abcc$.
       $\forall p \in \{\sigma_{i,j,k}\}$ строим множество $S_p = \{i \mid s_i$ является подстрокой $p\}$. Пусть S_p весит $|p|$.
       Хотим все i покрыть множествами из $\{S_p\}$ минимального суммарного веса.
       Пусть $S_{p_1}, \dots, S_{p_m}$ – покрытие min веса $\Rightarrow$ отвечаем строкой $p_1 \dots p_m$.

       Покрытие можем $(\ln n)$-приблизить. Покажем, что покрытие 2-приближает надстроку.
       Интуиция. Грубо говоря, «мы взяли каждое второе зацепление».

       Доказательство. Рассмотрим ответ – строки в порядке $s_{i_1} \dots s_{i_n}$. Пусть s_{i_k} – последняя, пересекающая s_{i_1}. Заметим, что мы могли кусок ответа $\langle s_{i_1}, s_{i_k} \rangle$ взять в качестве p_1. Продолжим процесс для s

       $s_{i_{k_2}}$ – последняя, пересекающая $s_{i_{k_1}+1}$. И так далее.
       Тогда есть покрытие из строк $\sigma_{1,i_1,k_1}, \sigma_{i_1+1,i_2,k_2}, \dots$
       В оптимальное покрытие участки строк от конца s_{i_j+1} до начала $s_{i_{j+1}+1}$ входят один раз, у нас ровно два раза.
       $\Rightarrow$ нашли не хуже, чем 2-приближение.

11. **(*) Пятачок, у тебя есть дома ружье?**

Решим задачу для отрезков на прямой.
Первый выстрел надо сделать в min правый конец. Выкинуть простреленные, решить задачу для оставшихся отрезков.

Вернемся к кругу. Заметим, что после первого выстрела круг разомкнется в прямую.
Более того, для каждого выстрела (в конец отрезка) однозначно определяется следующий.
Найдем двумя указателями для каждого выстрела r_i следующий $r_j : r_i < l_j$.
Если первый выстрел в r_i, то надо переходить к следующему до момента $r_{n+i} \leqslant r_j$ (удвоили круг).
Уже можем за $\mathcal{O}(n^2)$ для каждого варианта первого выстрела посчитать ответ.

За $\mathcal{O}(n \log n)$. Насчитаем `to`$[x][i]$ – прыжок из точки x на 2^i шагов вперед.
Перебираем первый выстрел и бинпоиском считаем, сколько хватит выстрелов.

Домашнее задание

3.1. Обязательная часть

1. **(2) 2-приближение Knapsack**

   Дано n предметов с весами w_i и стоимостями c_i.

   Алгоритм: выкинем предметы, которые не помещаются в рюкзак (т. е. $w_i > W$), остальные упорядочим по убыванию удельной стоимости: $q_1 \geqslant q_2 \geqslant \ldots \geqslant q_n$, $q_i = c_i/w_i$. Выберем наименьшее k, что набор $\{1, \ldots, k\}$ уже не влезает в рюкзак. После этого выберем лучший набор из двух вариантов: $\{1, \ldots, k-1\}$ или $\{k\}$.

   Докажите, что это 2-приближенние для задачи о рюкзаке.

2. **(2) Хорновские формулы**

   SAT-формула называется Хорновской, если в каждом дизъюнкте не более одного отрицания. Найти решение. Оценить время работы.

3. **(2) Размен**

   Есть монеты со стоимостями $c_1, \ldots, c_n$.
   Найти такое минимальное X, что X не представляется в виде суммы набора монет.

   Каждую монету можно брать только один раз (но у разных монет могут совпадать номиналы, то есть может быть $c_i = c_j$ для каких-то $i \neq j$).

4. **(3) Путь с заданным `XOR`**

   Дано дерево и число S. Найти путь, `XOR` на котором равен S. $\mathcal{O}(n \log n)$ времени, $\mathcal{O}(n)$ памяти.

   a) **(1)** веса на рёбрах;
   b) **(2)** веса в вершинах, путь простой.

3.2. Дополнительная часть

1. **(3) Дерево Штейнера**

   Дерево Штейнера для множества вершин T в графе G – такой связный подграф G, содержащий все вершины из T, что суммарный вес всех ребер подграфа минимален. Задача – найти дерево Штейнера. Веса неотрицательны. Граф неориентированный.

   a) **(2)** Для $|T| = k$ найти за $\mathcal{O}(V^3 + 3^k V)$
   b) **(0.25)** Для $|T| = |V|$ найти за $\mathcal{O}(E \log V)$
   c) **(0.25)** Для $|T| = 2$ найти за $\mathcal{O}(E \log V)$
   d) **(0.25)** Для $|T| = 3$ найти за $\mathcal{O}(E \log V)$
   e) **(0.25)** Для $|T| = 4$ найти за $\mathcal{O}(V^3)$

2. **(5) У коммивояжёра нет цели, только путь**

   Построить приближение *пути* коммивояжёра в симметричном полном графе с неравенством треугольника.

   a) **(2)** Путь между любыми двумя вершинами, 3/2-приближение.
   b) **(1)** Путь из данной вершины s, 3/2-приближение.
   c) **(2)** Путь между данными s и t, 5/3-приближение.