Лекция 10.

Задача одномерного интервального поиска.
Оценки сложности одномерного интервального поиска при вариации базового множества.

1 Базовое множество сверхлогарифмического поиска

Пусть

$$F_1 = \{\chi_{a,\rho_{int}} : a \in Y_{int}\}, \tag{1}$$

$$G_1 = \{g_{\leq,a}(u,v) = \begin{cases} 1, \text{ если } u \leq a \\ 2, \text{ если } u > a \end{cases} \; : \; a \in [0,1]\}, \tag{2}$$

$$G_2 = \{ \; g_{-,m}(u,v) = \\ = \begin{cases} 1, & \text{если } 0 \leq v-u < 1/m \\ 2 & \text{в противном случае} \end{cases} \; : m \in \mathbf{N}\}, \tag{3}$$

$$\mathcal{F}_2 = \langle F_1, G_1 \cup G_2 \rangle. \tag{4}$$

Понятно, что для $N_f \in \sigma$ для любого $f \in \widehat{G}_2$.
Справедлива следующая теорема.

Теорема 1. *Пусть функция плотности распределения вероятностей $p(u,v)$, определяющая меру $\mathbf{P}$ вероятностного пространства над множеством запросов X_{int}, такая, что $p(u,v) \leq c = const$; $I = \langle X_{int}, V, \rho_{int} \rangle$ — произвольная ЗИП типа S_{int}; $\mathcal{F}_2$ — базовое множество, определяемое соотношением (4). Тогда*

$$T(I,\mathcal{F}_2) \leq \sum_{y \in V} \mathbf{P}(O(y,\rho_{int})) + \log_2 \log_2 k + 6 + \log_2 c.$$

Доказательство. Упорядочим записи в библиотеке $V = \{y_1, \ldots, y_k\}$ так, что $y_1 \le y_2 \le \ldots \le y_k$.

Пусть m — некоторый параметр, значение которого определим позже.

Введем множество номеров $S = \{s_1, \ldots, s_{m-1}\}$ такое, что s_i — номер записи из V такой, что y_{s_i} — ближайшая слева запись к точке i/m ($i = \overline{1, m-1}$), а если такой записи не существует, то $s_i = 0$.

Возьмем точку, объявим ее корнем графа и обозначим β_0. Выпустим из корня два ребра, одно из которых будем считать левым, а другое правым. Конец левого ребра обозначим β_1, а конец второго — β_2.

Припишем корню переключатель $g_{-,m}(u, v)$ из множества G_3. Левому ребру припишем 1, а правому 2.

Построим для V граф U_V так, как это было сделано в случае логарифмического поиска, т.е. построим прошитое двоичное дерево, и совместим его корень с вершиной β_1.

Построим ориентированное сбалансированное бинарное дерево с $m-1$ концевыми вершинами с корнем в вершине β_2, все ребра которого ориентированы от β_2 к концевым вершинам. Обозначим это дерево через D_{m-1}. Из каждой внутренней вершины дерева D_{m-1} исходит 2 ребра, одному из них припишем 1, а другому — 2, т.е. все ребра дерева D_{m-1} будут переключательными. Тогда каждой концевой вершине дерева D_{m-1} можно сопоставить слово из символов 1 и 2 в соответствии с ребрами, встречающимися в цепи от корня к данному листу, причем первый символ слова соответствует ребру, исходящему из корня, а последний ребру, входящему в концевую вершину. Занумеруем концевые вершины в соответствии с лексикографическим порядком слов, соответствующих концевым вершинам, и обозначим их через $\beta'_1, \ldots, \beta'_{m-1}$, т.е. вершине β'_1 соответствует слово $11\ldots1$, а вершине β'_{m-1} слово $22\ldots2$. Нагрузку внутренних вершин дерева D_{m-1} переключателями из G_1 определим следующим образом. Пусть β — произвольная внутренняя вершина дерева D_{m-1}. Пусть β'_j — концевая вершина с максимальным номером в ветви, растущей из вершины, в которое ведет ребро с номером 1, исходящее из β. Тогда вершине β припишем переключатель $g_{\le,j/m}$. Такую операцию проделаем для всех внутренних вершин дерева, и полностью определим нагрузку вершин и ребер дерева D_{m-1}.

Напомним, что в графе U_V листья обозначаются $\alpha_1, \ldots, \alpha_k$ и им приписаны соответственно записи $y_1, \ldots, y_k$. Возьмем k новых точек, объявим их листьями и обозначим $\alpha'_1, \ldots, \alpha'_k$. Каждому листу α_i ($i = \overline{1, k}$) припишем запись y_i (тем самым каждая запись y_i будет приписана двум

листьям — α_i и α'_i). Из каждого листа α'_i ($i = \overline{2,k}$) выпустим ребро, ведущее в лист α'_{i-1}, и припишем ему предикат $\chi_{y_{i-1},\rho_{int}} \in F_1$.

Это множество из $(k-1)$-го ребра назовем левосторонней концевой цепью.

Теперь для каждой вершины β'_i проделаем следующие действия. Если $s_i \neq 0$, то из β'_i выпустим ребро, ведущее в лист α'_{s_i}, и припишем ему предикат $\chi_{y_{s_i},\rho_{int}} \in F_1$. Если $s_i < k$, то из β'_i выпустим ребро, ведущее в лист α_{s_i+1}, и припишем ему предикат $\chi_{y_{s_i+1},\rho_{int}} \in F_1$.

Это множество ребер, исходящих из вершин β'_i ($i = \overline{1,m-1}$), назовем правой предикатной частью.

Полученный таким образом ИГ обозначим через U_m.

Покажем, что граф U_m решает одномерную задачу интервального поиска $I = \langle X_{int}, V, \rho_{int} \rangle$.

Рассмотрим произвольную запись $y_i \in V$. По построению $L_{U_m}(y_i) = \{\alpha_i, \alpha'_i\}$. В каждый из листьев α_i и α'_i ведут только ребра, которым приписан предикат $\chi_{y_i,\rho_{int}}$, следовательно,

$$N_{\varphi_{\alpha_i} \vee \varphi_{\alpha'_i}} \subseteq O(y_i, \rho_{int}). \tag{5}$$

Пусть $x = (u, v)$ — произвольный запрос, такой что $u \leq y_i \leq v$, то есть $x \in O(y_i, \rho_{int})$. Покажем, что $\varphi_{\alpha_i}(x) \vee \varphi_{\alpha'_i}(x) = 1$.

Обозначим $A_a = \{x = (u, v) : 0 \leq v - u \leq a\}$.

Рассмотрим сначала случай, когда $x \in A_{1/(m+1)}$.

Тогда проводимость ребра (β_0, β_1) равна единице, а ребра (β_0, β_2) — нулю. Поскольку любая цепь, ведущая в лист α'_i проходит через ребро (β_0, β_2), то $\varphi_{\alpha'_i}(x) = 0$.

Обозначим через G'_β подграф графа U_m, состоящий из цепей, исходящих из вершины β.

Так как проводимость ребра (β_0, β_1) на запросе x равна 1, то мы выходим в вершину β_1. По построению граф G'_{β_1} совпадает с U_V и, следовательно, $\varphi_{\alpha_i}(x) = 1$, так как U_V решает ЗИП I.

Теперь рассмотрим случай, когда $x \notin A_{1/(m+1)}$.

В этом случае $g_{-,m}(x) = 2$ и проводимость ребра (β_0, β_1) будет равна 0, а ребра (β_0, β_2) равна 1, то есть мы попадем в вершину β_2, а проводимость всех цепей, проходящих через (β_0, β_1) равна 0. Из вершины β_2 растет дерево D_{m-1}, и оно обладает тем свойством, что из каждой его внутренней вершины исходит 2 переключательных ребра, и, следовательно, всегда проводимость ровно одного из ребер равна 1. Отсюда

следует, что для любого запроса всегда существует единственная цепь, соединяющая β_2 с некоторой вершиной β'_j, проводимость которой равна 1. Таким образом, после прохождения дерева мы попадем в некоторую вершину β'_j, причем по построению этого дерева эта вершина, такая, что j является минимальным числом, таким что $j/m \geq u$.

Так как $v - u \geq 1/m$, то точка j/m принадлежит отрезку $[u, v]$.

Рассмотрим два случая.

1) $y_i \leq j/m$.

Тогда $u \leq y_i \leq y_{s_j} \leq v$, так как y_{s_j} — ближайшая слева к j/m запись из библиотеки V. Отсюда следует, что проводимость ребра, ведущего из β'_j в лист α'_{s_j} равна 1. Осталось заметить, что проводимость части левосторонней концевой цепи, ведущей из α'_{s_j} в α'_i, также будет равна 1, так как $0 \leq y_i \leq y_{s_j} \leq v$.

Отметим также, что в этой ситуации $\varphi_{\alpha_i}(x) = 0$, так как в лучшем случае мы попадем в вершины правосторонней концевой цепи в точке α_{s_j+1}, которая в этой цепи находится после точки α_i.

2) $y_i > j/m$.

Тогда $u \leq y_{s_j+1} \leq y_i \leq v$. Отсюда следует, что проводимость ребра, ведущего из β'_j в лист α_{s_j+1}, равна 1, и проводимость части правосторонней концевой цепи, ведущей из α_{s_j+1} в α_i, также равна 1 на запросе x.

Аналогично предыдущему случаю $\varphi_{\alpha'_i}(x) = 0$.

Тем самым мы показали, что для любой записи $y_i \in V$ и любого запроса $\in X_{int}$ такого, что $x \rho_{int} y_i$ в графе U_m существует проводящая на запросе x цепь, ведущая из корня в какой-либо (но ровно в один) из листьев α_i и α'_i.

Что и доказывает, что граф U_m решает задачу I.

Подсчитаем сложность графа U_m.

Рассмотрим сначала произвольный запрос $x \in A_{1/m}$.

В этом случае

$$T(U_m, x) \leq 1 + (\rfloor \log_2 k[-1) + 2 + |\mathcal{J}_{U_m}(x)|.$$

Здесь первое слагаемое соответствует вычислению переключателя $g_{-,m}$ в вершине β_0. Второе слагаемое дают переключатели, входящие в единственную проводящую цепь, пролегающую через переключательную часть дерева D. Третье слагаемое соответствует вычислению одного или двух предикатов, соответствующих предикатным ребрам из предикатной части дерева D, растущим из вершины, в которую ведет проводящая цепь.

Четвертое слагаемое соответствует вычислению предикатов, соответствующих ребрам, исходящим из листьев, записи которых входят в ответ (по одному на каждую запись).

Рассмотрим случай, когда $x \in X \backslash A_{1/m}$.

Тогда

$$T(U_m, x) \leq 1 +] \log_2 (m-1) [+ 2 + |\mathcal{J}_{U_m}(x)|.$$

Здесь первое слагаемое соответствует вычислению переключателя $g_{-,m}$. Второе слагаемое дают переключатели, входящие в единственную проводящую цепь, пролегающую через дерево D_{m-1}. Третье слагаемое соответствует вычислению одного или двух предикатов, приписанных ребрам, исходящим из той вершины β'_j, в которую ведет проводящая цепь дерева D_{m-1}. И, наконец, четвертое слагаемое, как и ранее, соответствует вычислению предикатов, соответствующих ребрам, исходящим из листьев, записи которых входят в ответ. Как мы показали ранее, для любой записи y_i, вошедшей в ответ, ровно у одного из листьев α_i и α'_i функция фильтра будет равна 1, и, следовательно, каждой записи соответствует ровно одно ребро и соответственно ровно один вычисленный предикат.

Подсчитаем сложность графа U_m.

$$\begin{aligned}
T(U_m) &= \mathbf{M}_x \, T(U_m, x) = \mathbf{P}(A_{1/m}) \cdot (2 +] \log_2 k [) + \\
&\quad + \mathbf{P}(X \backslash A_{1/m}) \cdot (3 +] \log_2 (m-1) [) + \mathbf{M}_x \, |\mathcal{J}_{U_m}(x)| \leq \\
&\leq \mathbf{P}(A_{1/m}) \cdot [\log_2 k] + 3 + [\log_2 m] + \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})) \leq \\
&\leq c \cdot [\log_2 k] \left(\frac{2}{m} - \frac{1}{m^2} \right) + 3 + [\log_2 m] + \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})) \leq \\
&\leq \frac{2c \cdot [\log_2 k]}{m} + 3 + [\log_2 m] + \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})).
\end{aligned}$$

Поскольку граф U_m решает задачу I, то

$$\mathcal{J}_{U_m}(x) = \{ y \in V : x \rho_{int} y \},$$

и следовательно, используемое выше равенство

$$\mathbf{M}_x \, |\mathcal{J}_{U_m}(x)| = \sum_{y \in V} \mathbf{P}(O(y, \rho_{int}))$$

доказывается простой сменой порядка суммирования.

Установив значение параметра $m =]2c \log_2 k[$, получим

$$T(U_m) \le \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})) + \log_2 \log_2 k + 6 + \log_2 c.$$

Теорема 1 доказана. □

Алгоритм поиска, соответствующий графу U_m, представляет собой следующее.

Пусть нам дан некий запрос $x = (u, v) \in X_{int}$. Поиск по этому запросу будем осуществлять следующим образом.

Сначала вычислим длину интервала x.

Если $v - u < 1/([\log_2 k] + 1)$, то ответ будем искать с помощью прошитого двоичного дерева. Если $v - u \ge 1/([\log_2 k] + 1)$, то мы за время $\log_2 \log_2 k$ найдем номер j точки j/m, попадающей в интервал $[u, v]$. Затем, начиная с записи с номером s_j, просматриваем справа налево записи из V и сравниваем с левым концом запроса — точкой u. Как только очередная запись окажется меньше u, мы, начиная с записи с номером $s_j + 1$, просматриваем слева направо записи из V и сравниваем с правым концом запроса — точкой v до тех пор, пока очередная запись не станет больше v.

Таким образом, в первом случае помимо перечисления ответа мы тратим время порядка $\log_2 k$, а во втором — $\log_2 \log_2 k$. Поскольку в подавляющем большинстве интервал x будет достаточно большим, то мы получим оценку теоремы 1.

2 Мгновенное решение

Пусть

$$G_3 = \{g_{\cdot,m}(u, v) = \max(1,]u \cdot m[) : m \in \mathbf{N}\}, \tag{6}$$

$$\mathcal{F}_3 = \langle F_1, G_1 \cup G_2 \cup G_3 \rangle, \tag{7}$$

где F_1, G_1, G_2 определяются соотношениями (1), (2), (3).

Понятно, что для $N_f \in \sigma$ для любого $f \in \widehat{G}_3$.

Справедлива следующая теорема.

Теорема 2. *Пусть ЗИП $I = \langle X_{int}, V, \rho_{int} \rangle$ — одномерная задача интервального поиска, то есть задача типа S_{int}, $\mathcal{F}_3$ — базовое множество,*

определяемое соотношением (7), и $R(I) \stackrel{\text{def}}{=} \sum_{y \in V} \mathbf{P}(O(y, \rho_{int}))$. *Тогда, если функция плотности вероятности* $p(x)$, *определяющая вероятностную меру* $\mathbf{P}$, *ограничена константой* c, *то*

$$R(I) < T(I, \mathcal{F}_3, 4\,|V| - 3 + 6\,c[\log_2 |V|]) \leq R(I) + 5.$$

Доказательство. Нижняя оценка следует из мощностной нижней оценки.

Построим ИГ, на котором достигается верхняя оценка одномерной задачи интервального поиска.

Пусть m — некоторый параметр, значение которого определим позже.

Возьмем ИГ U_m, построенный в теореме 1. В нем из вершины β_2 исходит дерево D_{m-1}, концевыми вершинами которого являются вершины $\beta'_1, \ldots, \beta'_{m-1}$. Удалим все ребра дерева D_{m-1} и все вершины, не совпадающие с вершинами β_2 и $\beta'_1, \ldots, \beta'_{m-1}$. Выпустим из вершины β_2 $m-1$ ребро, припишем этим ребрам числа от 1 до $m-1$, и заменим нагрузку вершины β_2 на переключатель $g_{\cdot,m} \in G_3$. Полученный информационный граф обозначим через U'_m.

Отметим, что для любого запроса $x = (u, v) \in X_{int}$ если $j = g_{\cdot,m}(x)$, то j является минимальным числом, таким что $j/m \geq u$. Следовательно, вершина β_2 с исходящими из нее ребрами функционально совпадает с деревом D_{m-1}, и, значит, ИГ U'_m решает ЗИП I.

Что касается сложности графа U'_m, то по сравнению с ИГ U_m на любом запросе $x \in X_{int} \backslash A_{1/m}$ вместо $]\log_2(m-1)[$ вычислений переключателей, соответствующих проводящей цепи дерева D_{m-1}, будет вычислен один переключатель $g_{\cdot,m}$. Поэтому

$$\begin{aligned}
T(U'_m) &= \mathbf{M}_x\, T(U'_m, x) = \mathbf{P}(A_{1/m}) \cdot (2+]\log_2 k[) + \\
&\quad + \mathbf{P}(X \backslash A_{1/m}) \cdot 4 + \mathbf{M}_x\, |\mathcal{J}_{U'_m}(x)| \leq \\
&\leq \mathbf{P}(A_{1/m}) \cdot ([\log_2 k] - 1) + 4 + \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})) \leq \\
&\leq c \cdot ([\log_2 k] - 1) \left(\frac{2}{m} - \frac{1}{m^2} \right) + 4 + \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})) \leq \\
&\leq \frac{2c \cdot ([\log_2 k] - 1)}{m} + 4 + \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})).
\end{aligned}$$

Подсчитаем объем графа U'_m:

$$Q(U'_m) \le 2 + (2k-1) + (k-1) + (k-1) + (m-1) + 2(m-1).$$

Здесь первое слагаемое соответствует ребрам, исходящим из β_0. Второе слагаемое есть количество ребер в дереве D. Третье и четвертое слагаемые соответствуют ребрам из правосторонней и левосторонней концевых цепочек. Пятое слагаемое — это ребра, исходящие из вершины β_2. И, наконец, шестое слагаемое не меньше чем число ребер, исходящих из вершин β'_i $(i = \overline{1,m})$.

Возьмем в качестве параметра $m = [2\,c\,[\log_2 k]]$ и получим

$$T(U'_m) \le 5 + \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})),$$

$$Q(U'_m) \le 4\,k - 3 + 6\,c[\log_2 k],$$

что и доказывает утверждение теоремы 2. □

Дадим неформальное описание алгоритма, приведенного выше.

Пусть нам дано множество $V = \{y_1, \ldots, y_k\}$, в котором мы должны производить поиск. Сначала упорядочим его в порядке возрастания. Если известна оценка сверху c функции плотности вероятности появления запросов, то в качестве параметра m возьмем $m = 2\,c\,[\log_2 k]$, если же c неизвестна, то вместо нее можно взять любое число, например, $c = 2$. Затем для V построим множество номеров $S = \{s_1, \ldots, s_{m-1}\}$, описанное выше. Отметим, что оно строится только однажды. Теперь поиск по произвольно взятому интервалу-запросу $x = (u, v)$ производится следующим образом.

Сначала вычисляется длина запроса x.

Если она меньше чем $1/m$, то в множестве V бинарным поиском находится ближайшая справа к точке u запись. Далее, начиная с этой записи, просматриваются слева направо все записи из V и сравниваются с правым концом запроса — точкой v — до тех пор, пока очередная запись не станет больше v. Тем самым в этом случае, помимо перечисления ответа, производится порядка $\log_2 k$ действий.

Если $v - u \ge 1/m$, то с помощью функции $g_{\cdot,m}$ получаем номер j точки j/m, попадающей в интервал $[u, v]$. Затем, начиная с записи с номером s_j, просматриваем справа налево записи из V и сравниваем с левым концом запроса — точкой u. Как только очередная запись окажется меньше

u, мы, начиная с записи с номером $s_j + 1$, просматриваем слева направо записи из V и сравниваем с правым концом запроса — точкой v до тех пор, пока очередная запись не станет больше v. Тем самым в этом случае мы, помимо перечисления ответа, производим 4 лишних действия (сравниваем $v - u$ с $1/m$, вычисляем функцию $g_{\cdot,m}$, делаем 1 лишнее действие, идя справа налево, и 1 лишнее действие, идя слева направо).

Осталось заметить, что параметр m подобран так, что средняя сложность первого случая не превышает 1, если известна оценка сверху функции плотности вероятности, и не превышает некоторой константы, если эта оценка точно не известна.

И, наконец, заметим, что данный алгоритм требует дополнительную память порядка $\log_2 k$, чтобы хранить множество S.