Логарифмическое решение задачи об опасной близости

Е. А. Скиба

Исследуется задача о поиске движущихся объектов, которые могут столкнуться с движущимся объектом-запросом, где под столкновением понимается нахождение объектов в опасной близости. Для случая фиксированных скоростей объектов описан алгоритм, имеющий логарифмическую сложность.

1. Основные понятия и формулировка результата

Постановка задачи. Сначала дадим неформальную постановку задачи. Внутри квадрата $[0,1] \times [0,1]$ прямолинейно и с фиксированными двигаются точечные объекты, причем появляются они на двух смежных сторонах квадрата и двигаются перпендикулярно той стороне, на которой появились.

Объекты, появляющиеся на одной из сторон, будем называть объектами-данными (они пронумерованы в порядке своего появления и имеют скорость v), а объекты, появляющиеся на другой стороне — объектами-запросами (они имеют скорость $\tilde{v}$).

В момент появления каждый объект сообщает системе свои координаты, а система фиксирует время его появления. В задаче требуется для каждого объекта-запроса перечислить все объекты-данные, которые

- появились до него,
- с которыми он, в течение своего движения внутри квадрата, будет находиться в состоянии опасной близости, то есть на рассто-

янии меньшем, чем ρ. Расстояние в данной задаче измеряется по Манхэттену.

Оси координат расположим вдоль сторон квадрата, таким образом, что у объектов-данных координата по оси X в начальный момент равна 0 и скорость параллельна оси X и равняется v, а у объектов-запросов — координата по оси Y в начальный момент равна 0, а скорость параллельна оси Y и равняется $\tilde{v}$.

Формализуем постановку задачи. Обозначим за t_0 текущий момент времени.

Объектами-данными назовем пары чисел вида $o_i = (t_i, y_i)$, где t_i — время появления на границе квадрата, y_i — координата появления по оси Y. **Объектами-запросами** будем называть пары вида $q = (t, x)$, где t — время появления на границе квадрата, x — координата появления по оси X. Множество всех объектов-запросов, появляющихся в момент времени t_0, обозначим $Q(t_0)$, то есть $Q(t_0) = \{q = (t_0, x) : x \in [0, 1]\}$. Фактически, время появления будет переменным только для объектов-данных, а для объектов-запросов момент появления будет всегда совпадать с текущим моментом времени t_0, поэтому в дальнейшем вместо $Q(t_0)$ будем писать просто Q.

Координата точки появления y_i для любого объекта-данного o_i имеет равномерное распределение на отрезке $[0, 1]$ по оси Y, то есть для любых a, b, таких что $0 \leqslant a \leqslant b \leqslant 1$ выполнено $P(y_i \in [a, b]) = b - a$.

Аналогично координата точки появления x для объекта-запроса q имеет равномерное распределение на отрезке $[0, 1]$ по оси X, то есть для любых a, b, таких что $0 \leqslant a \leqslant b \leqslant 1$ выполнено $P(x \in [a, b]) = b - a$.

Последовательность $\bar{V}_\lambda = \{(t_i, y_i), i = \overline{1, \infty}\}$ такую, что моменты появления объектов-данных t_i образуют пуассоновский поток с параметром λ, то есть $t_{i+1} = t_i + \delta_i$, где δ_i для любого i распределено по показательному закону с параметром λ, назовем последовательностью объектов-данных.

Множество всех $\bar{V}_\lambda$ будем обозначать $\mathbb{V}_\lambda$.

Библиотекой $V_\lambda(t_0)$ назовем множество объектов-данных, находящихся в текущий момент времени t_0 внутри квадрата, то есть:

$$V_\lambda(t_0) = \{(t_i, y_i) \in \bar{V}_\lambda : t_i \in [t_0 - \frac{1}{v}, t_0]\},$$

Поскольку параметры λ и t_0 у нас фиксированы, то в дальнейшем вместо $V_\lambda(t_0)$ будем, как правило, писать просто V.

Множество всех библиотек в текущий момент времени определим как

$$\mathbb{V}_\lambda(t_0) = \{V_\lambda(t_0) : \bar{V}_\lambda \in \mathbb{V}_\lambda\}.$$

Ответом на запрос $q = (t_0, x)$ назовем множество объектов из V, которые будут с этим объектом в опасной близости, то есть:

$$J(\rho, q, V) = \{o_i \in V_\lambda(t_0) \text{ такие, что для каждого } i \text{ существует } t \in [t_0, t_0 + \frac{1}{v}] \text{ такое, что } |v(t - t_i) - x| + |y_i - \tilde{v}(t - t_0)| \leqslant \rho\}.$$

Тройку (ρ, Q, V) будем называть **задачей об опасной близости**, где через ρ обозначено расстояние опасной близости.

В качестве модели базы данных будем использовать **информационные графы (ИГ)** [1] с возможностью поиска, вставки и удаления в них.

Структура ИГ. Пусть F — множество одноместных предикатов, заданных на множестве Q. Пусть G — множество функций, определенных на Q, области значений которых являются отрезками натурального ряда. Назовем их *переключателями*.

Пару $\mathcal{F}(F, G)$ будем называть базовым множеством.

Пусть нам дан произвольный ориентированный граф. Выделим в нем одну вершину и назовем ее корнем ИГ. Также выделим произвольное множество вершин и назовем их листьями. Вершины графа, не являющиеся листьями, будем называть *внутренними вершинами*.

Сопоставим каждому листу графа некоторую запись из множества V.

Выделим некоторое подмножество внутренних вершин, которые назовем *точками переключения*. Каждой точке переключения с полустепенью исхода m поставим в соответствие переключатель $g(x) \in G$, у которого область значений содержит не менее m чисел. Занумеруем все дуги , исходящие из v последовательными числами из

области значений f_v. Такие дуги назовем *переключательными*. Внутренние вершины, не являющиеся переключательными, также назовем *предикатными*. Каждой предикатной дуге сопоставим символ из F.

Полученный граф назовем *информационным графом* над базовым множеством $\mathcal{F}$ с библиотекой V.

Поиск в ИГ. Каждому ИГ U с базовым множеством $\mathcal{F}$ и библиотекой V можно сопоставить процедуру поиска. Предполагается, что эта процедура хранит в своей внешней памяти структуру U. Входными данными является запрос $q \in Q$. Выходными данными является подмножество библиотеки V.

Пусть на вход процедуры поступил запрос $q \in Q$. Введем понятие активного множества вершин и внесем в него в начальный момент корень ИГ U и помечаем его. Далее по очереди просматриваем вершины из активного множества и для каждой из них проделываем следующее:

- если рассматриваемая вершина — лист, то запись, приписанную вершине включаем в ответ,
- если рассматриваемая вершина — точка переключения, то вычисляем на запросе q переключатель, соответствующий данной вершине, и если дуга, нагрузка которой равна значению переключателя существует, и ее конец — не помеченная вершина, то помечаем конец дуги и включаем его в множество активных вершин,
- если рассматриваемая вершина — предикатная, то просматриваем по очереди исходящие из нее дуги и вычисляем значения предикатов, приписанных этим дугам на запросе q. Концы дуг, которым соответствуют предикаты со значением, равным 1, если они непомеченные, помечаем и включаем в множество активных вершин,
- исключаем рассматриваемую вершину из активного множества.

Ответ, полученный на выходе процедуры обозначим $J(U, q)$.

Скажем, что ИГ U *решает задачу поиска* для задачи об опасной близости (ρ, Q, V), если ответ, полученный на выходе процедуры поиска, совпадает с ответом на запрос, то есть

$$J(U,q) = J(\rho, q, V).$$

Скажем, что ИГ U *решает задачу поиска с погрешностью первого рода* для задачи об опасной близости (ρ, Q, V), если ответ, полученный на выходе процедуры поиска, является подмножеством ответа на запрос, то есть

$$J(U,q) \subset J(\rho, q, V).$$

Скажем, что ИГ U *решает задачу поиска с погрешностью второго рода* для задачи об опасной близости (ρ, Q, V), если ответ на запрос является подмножеством ответа, полученного на выходе процедуры поиска, то есть

$$J(U,q) \supset J(\rho, q, V).$$

Сложность поиска в ИГ. Пусть нам дан информационный граф U над базовым множеством $\mathcal{F}$. Каждому символу $g \in G$ ($f \in F$) поставим в соответствие неотрицательное число $t(g)$ ($t(f)$), обозначающее сложность переключателя g (предиката f).

Сложностью поиска по запросу q в ИГ U для задачи об опасной близости (ρ, Q, V) назовем сумму сложностей всех переключателей и предикатов, вычисленных в ходе процедуры поиска, минус величина, равная $|J(U,q)|$. Обозначим эту сложность $T(U,q)$. Вычитаемая величина $|J(U,q)|$ характеризует время перечисления ответа и неизбежно будет присутствовать в любом алгоритме поиска.

Вставка и удаление записи в ИГ. Назовем вставкой объекта o в ИГ U с библиотекой V такое преобразование ИГ U, при котором библиотека V преобразуется в $V \cup \{o\}$ и U остается ИГ, решающим задачу поиска.

Удалением объекта o из ИГ U с библиотекой V назовем такое преобразование ИГ U, при котором библиотека V преобразуется в $V \backslash \{o\}$ и U остается ИГ, решающим задачу поиска.

Вставка/удаление представляет собой последовательность преобразований вершин, дуг графа, их пометок, которые мы договорились считать элементарными, то есть на выполнение которых тратиться

небольшое и константное время. Множество элементарных операций обозначим $H = \{h_1, ..., h_m\}$.

Сложность вставки/удаления в ИГ. Сопоставим каждому $h \in H$ величину $t(h)$ и назовем ее сложностью элементарной операции h.

Тогда *сложность вставки* объекта o в ИГ U равна сумме сложностей всех элементарных операций, произведенных в ходе выполнения процедуры вставки. Обозначим ее $S(U, o)$.

Аналогично, *сложность удаления* объекта o из ИГ U определим как сумму сложностей соответствующих элементарных операций и обозначим как $R(U, o)$.

Понятие алгоритма. **Алгоритмом** A решения задачи об опасной близости (ρ, Q, V) будем называть ИГ U с определенными на нем процедурами поиска, вставки и удаления, такой, что ИГ U решает (с погрешностью или без) задачу поиска для задачи об опасной близости (ρ, Q, V).

В этом случае U будем называть информационным графом, **соответствующим** алгоритму A.

Тогда **сложность поиска** по алгоритму A ответа на запрос q для расстояния опасной близости ρ при библиотеке V обозначим как $T_A(\rho, q, V)$ и будем считать равной $T(U, q)$.

Сложность вставки по алгоритму A объекта o в библиотеку V для расстояния опасной близости ρ обозначим $S_A(\rho, o, V)$ и будем считать равной $S(U, o)$.

Сложность удаления по алгоритму A объекта o из библиотеки V для расстояния опасной близости ρ обозначим $R_A(\rho, o, V)$ и будем считать равной $R(U, o)$.

Объемом алгоритма A назовем число ребер в ИГ U и обозначим $Q_A(\rho, V)$.

Ответ на запрос $q \in Q$ для задачи об опасной близости (ρ, Q, V), полученный в результате работы алгоритма A, будем обозначать $J_A(\rho, q, V)$ и считать равным $J(U, q)$.

Понятие погрешности алгоритма. Погрешностью первого рода алгоритма A назовем математическое ожидание от чис-

ла объектов-данных, находящихся в опасной близости с объектом-запросом, но не попавших в ответ, то есть

$$f_A^1(\rho,\lambda) = M_q M_V |J(\rho,q,V)\backslash J_A(\rho,q,V)|,$$

где математическое ожидание берется по всем V из $\mathbb{V}_\lambda(t_0)$.

Аналогично, **погрешностью второго рода** алгоритма A назовем математическое ожидание от числа объектов, не находящихся в опасной близости, но попавших в ответ :

$$f_A^2(\rho,\lambda) = M_q M_V |J_A(\rho,q,V)\backslash J(\rho,q,V)|,$$

где математическое ожидание также берется по всем V из $\mathbb{V}_\lambda(t_0)$.

Формулировка результата. Будем писать, что $f(n) = O(n)$, если существуют такие константы $C_1 > 0, C_2 > 0$, и константы C_3,C_4 такие, что для каждого n выполнено:

$$C_1 n + C_3 \leqslant f(n) \leqslant C_2 n + C_4.$$

В данной статье предлагается алгоритмы A_1, A_2, A_3 решения задачи об опасной близости с помощью сведения к задаче одномерного интервального поиска. Их характеристики описываются следующей теоремой:

Теорема 1. *Пусть $\tilde{v}$ — скорость объектов-запросов, v — скорость объектов-данных, ρ — расстояние опасной близости.*

Пусть также V — библиотека объектов-данных, такая что моменты появления объектов-данных образуют пуассоновский поток с параметром λ и их координаты появления равномерно распределены на отрезке $[0,1]$, а Q — множество объектов-запросов, такое что координаты появления объектов-запросов тоже равномерно распределены на отрезке $[0,1]$.

Тогда алгоритм A_1 решает задачу об опасной близости (ρ, Q, V) для случая $\tilde{v} \geqslant v$ без погрешности. Для сложности поиска ответа на любой запрос q из Q, вставки любого объекта-данного o, удаления любого объекта-данного $\tilde{o}$ по алгоритму A_1 и для объема алгоритма A_1 верны следующие оценки:

$$T_{A_1}(\rho, q, V) = O(\log_2 |V|),$$
$$S_{A_1}(\rho, o, V) = O(\log_2 |V|),$$
$$R_{A_1}(\rho, \tilde{o}, V) = O(\log_2 |V|),$$
$$Q_{A_1}(\rho, V) = O(|V|).$$

Алгоритм A_2 решает задачу об опасной близости (ρ, Q, V) для случая $\tilde{v} < v$ с погрешностью второго рода. Для сложности поиска ответа на любой запрос q из Q, вставки любого объекта-данного o, удаления любого объекта-данного $\tilde{o}$ по алгоритму A_2, для объема и погрешности алгоритма A_2 верны следующие оценки:

$$T_{A_2}(\rho, q, V) = O(\log_2 |V|),$$
$$S_{A_2}(\rho, o, V) = O(\log_2 |V|),$$
$$R_{A_2}(\rho, \tilde{o}, V) = O(\log_2 |V|),$$
$$Q_{A_2}(\rho, V) = O(|V|),$$
$$f^2_{A_2}(\rho, \lambda) = \frac{\lambda \rho^2 (v - \tilde{v})}{2v\tilde{v}}, \quad f^1_{A_2}(\rho, \lambda) = 0.$$

Алгоритм A_3 решает задачу об опасной близости (ρ, Q, V) для случая $\tilde{v} < v$ с погрешностью первого рода. Для сложности поиска ответа на любой запрос q из Q, вставки любого объекта-данного o, удаления любого объекта-данного $\tilde{o}$ по алгоритму A_3, для объема и погрешности алгоритма A_3 верны следующие оценки:

$$T_{A_3}(\rho, q, V) = O(\log_2 |V|),$$
$$S_{A_3}(\rho, V) \leqslant O(\log_2 |V|),$$
$$R_{A_3}(\rho, V) \leqslant O(\log_2 |V|),$$
$$Q_{A_3}(\rho, V) = O(|V|),$$
$$f^1_{A_3}(\rho, \lambda) = \frac{\lambda \rho^2 (v - \tilde{v})}{2v\tilde{v}}, \quad f^2_{A_3}(\rho, \lambda) = 0.$$

2. Вспомогательные утверждения

Лемма 1. *Пусть $\tilde{v}$ — скорость объектов-запросов, v — скорость объектов-данных, ρ — расстояние опасной близости.*

Тогда объект-данное $o_i = (t_i, y_i)$ входит в ответ $J(\rho, q, V)$ на запрос $q = (t_0, x)$ тогда и только тогда, когда o_i принадлежит библиотеке V и выполнено:

1) при $v \leqslant \tilde{v}$

$$\left[\begin{array}{l} \left\{\begin{array}{l} x \leqslant 1-\rho \\ t_0 - \frac{x+\rho}{v} \leqslant t_i - \frac{y_i}{\tilde{v}} \leqslant t_0 - \frac{x-\rho}{v} \end{array}\right. \\ \left\{\begin{array}{l} x > 1-\rho \\ t_0 - \frac{x+\rho}{v} + \frac{1}{\tilde{v}} - \frac{1}{v} \leqslant t_i - \frac{y_i}{\tilde{v}} \leqslant t_0 - \frac{x-\rho}{v}. \end{array}\right. \end{array}\right. \tag{1}$$

2) при $v > \tilde{v}$

$$\left[\begin{array}{l} \left\{\begin{array}{l} t_i \geqslant t_0 - \frac{x}{v} \\ t_0 - \frac{x}{v} - \frac{\rho}{\tilde{v}} \leqslant t_i - \frac{y_i}{\tilde{v}} \leqslant t_0 - \frac{x}{v} + \frac{\rho}{\tilde{v}} \end{array}\right. \\ \left\{\begin{array}{l} t_i < t_0 - \frac{x}{v} \\ t_i - \frac{y_i}{v} \geqslant t_0 - \frac{x+\rho}{v} \end{array}\right. \end{array}\right. \tag{2}$$

Доказательство. Ясно, что $o_i = (t_i, y_i)$ входит в ответ $J(\rho, u, V)$ на запрос $q = (t_0, x)$ тогда и только тогда, когда o_i принадлежит библиотеке V и система

$$\left\{\begin{array}{l} |x - v(t - t_i)| + |y_i - \tilde{v}(t - t_0)| \leqslant \rho \\ t \geqslant t_0 \\ t \leqslant t_i + \frac{1}{v} \\ t \leqslant t_0 + \frac{1}{\tilde{v}} \end{array}\right. \tag{3}$$

имеет решение.

Рассмотрим функцию

$$f(t) = |x - v(t - t_i)| + |y_i - \tilde{v}(t - t_0)|.$$

Обозначим через t_{min} точку, в которой функция $f(t)$ принимает минимальное значение. Очевидно, что t_{min} — это такая точка, в которой один из модулей (с большим перед t коэффициентом) равен 0.

1) Пусть $v \geqslant \tilde{v}$. Тогда

$$t_{min} = t_i + \frac{x}{v}.$$

Заметим сразу, что для $t = t_{min}$ выполняется предпоследнее условие в (3) и, поскольку

$$i \in V \Leftrightarrow t_i \leqslant t_0 \leqslant t_i + \frac{1}{v},$$

то выполняется и последнее условие в (3). Теперь рассмотрим два случая:

а) Для $t = t_{min}$ выполняется второе условие в (3), то есть

$$t_i + \frac{x}{v} \geqslant t_0 \Leftrightarrow (t_0 - t_i)v \leqslant x.$$

Тогда остается для $t = t_{min}$ проверить первое условие в (3):

$$|y_i - \tilde{v}(t_i + \frac{x}{v} - t_0)| \leqslant \rho \Leftrightarrow |y_i + \tilde{v}t_0 - \tilde{v}t_i - \frac{\tilde{v}x}{v}| \leqslant \rho \Leftrightarrow$$
$$\Leftrightarrow \frac{\tilde{v}}{v}x - \rho \leqslant y_i + \tilde{v}(t_0 - t_i) \leqslant \frac{\tilde{v}}{v}x + \rho.$$

б) Для $t = t_{min}$ не выполняется второе условие в (3), то есть

$$(t_0 - t_i)v \geqslant x \tag{4}$$

Тогда

$$f(t) = |v(t - t_i) - x| + |\tilde{v}(t - t_0) - y_i| = |v(t_0 - t_i) - x +$$
$$+ v(t - t_0))| + |y_i - \tilde{v}(t - t_0)|.$$

Первый модуль можно «снять» в силу (3) и (4). Поэтому

$$f(t) \geqslant v(t_0 - t_i) - x + y_i + (v - \tilde{v})(t - t_0) = g(t).$$

Очевидно, что при ограничениях, задаваемых тремя последними неравенствами в (3), $g(t)$ достигает своего минимума в точке $t = t_0$. Кроме того, $g(t_{min}) = f(t_{min})$. Следовательно, $f(t)$ также достигает своего минимума в точке t_0. Тогда проверим первое условие из (3):

$$(t_0 - t_i)v - x + y_i \leqslant \rho \Leftrightarrow y_i + v(t_0 - t_i) \leqslant x + \rho.$$

Таким образом, мы получили совокупность условий, при которых система (3) имеет решение:

$$\left[\begin{array}{l} \left\{\begin{array}{l} (t_0-t_i)v \leqslant x \\ \frac{\tilde{v}}{v}x-\rho \leqslant y_i+\tilde{v}(t_0-t_i) \leqslant \frac{\tilde{v}}{v}x+\rho. \end{array}\right. \\ \left\{\begin{array}{l} (t_0-t_i)v > x \\ y_i+v(t_0-t_i) \leqslant x+\rho \end{array}\right. \end{array}\right.$$

2) Пусть $v < \tilde{v}$. Тогда

$$t_{min} = t_0 + \frac{y_i}{\tilde{v}}.$$

Заметим сразу, что для $t = t_{min}$ верны второе и четвертое условия из (3). Рассмотрим два случая:

а) Для $t = t_{min}$ верно третье условие из (3), то есть

$$t_0+\frac{y_i}{\tilde{v}} \leqslant t_i+\frac{1}{v} \Leftrightarrow \tilde{v}vt_0+vy_i \leqslant \tilde{v}vt_i+\tilde{v} \Leftrightarrow y_i+\tilde{v}(t_0-t_i) \leqslant \frac{\tilde{v}}{v}.$$

Проверим первое условие в (3):

$$|x-v(t_0+\frac{y_i}{\tilde{v}}-t_i)| \leqslant \rho \Leftrightarrow \frac{\tilde{v}}{v}(x-\rho) \leqslant y_i+\tilde{v}(t_0-t_i) \leqslant \frac{\tilde{v}}{v}(x+\rho).$$

б) Для $t = t_{min}$ не верно третье условие из (3), то есть

$$y_i+\tilde{v}(t_0-t_i) > \frac{\tilde{v}}{v} \tag{5}$$

Покажем, что в этом случае верно, что

$$t_i+\frac{1}{v} \leqslant t_0+\frac{1}{\tilde{v}}. \tag{6}$$

От противного: пусть выполнено, что

$$t_i+\frac{1}{v} > t_0+\frac{1}{\tilde{v}} \Leftrightarrow \tilde{v}(t_0-t_i) < \frac{\tilde{v}}{v}-1 \Rightarrow$$
$$\Rightarrow y_i+\tilde{v}(t_0-t_i) < y_i+\frac{\tilde{v}}{v}-1 \leqslant \frac{\tilde{v}}{v}.$$

Следовательно, не верно (5). Пришли к противоречию, то есть (6) верно.

От переменной t перейдем к переменной τ, сделав замену

$$\tau = t_i + \frac{1}{v} - t,$$

причем так как по (3), (6)

$$t_0 \leqslant t \leqslant t_i + \frac{1}{v} \Rightarrow 0 \leqslant \tau \leqslant t_i + \frac{1}{v} - t_0.$$

$$f(\tau) = |x - 1 + \tau v| + |y_i + \tilde{v}t_0 - \tilde{v}(t_i + \frac{1}{v} - \tau)| =$$

$$= |1 - x - \tau v| + |y_i + \tilde{v}(t_0 - t_i) - \frac{\tilde{v}}{v} + \tau\tilde{v}|.$$

Второй модуль можем снять по (5), получаем:

$$f(\tau) \geqslant 1 - x - \tau v + y_i + \tilde{v}(t_0 - t_i) - \frac{\tilde{v}}{v} + \tau\tilde{v} = g(\tau).$$

$g(\tau)$ достигает своего минимума на отрезке $[0, t_i + \frac{1}{v}]$ при $\tau = 0$. При этом $g(0) = f(0)$, то есть $f(\tau)$ также достигает своего минимального значения при $\tau = 0$ (или $t = t_i + \frac{1}{v}$). Тогда первое уравнение из (3) можно записать как

$$1 - x + y_i + \tilde{v}(t_0 - t_i) - \frac{\tilde{v}}{v} \leqslant \rho \Leftrightarrow y_i + \tilde{v}(t_0 - t_i) \leqslant \frac{\tilde{v}}{v} + \rho + x - 1.$$

Таким образом, мы получили еще одну совокупность условий, при которых система (3) имеет решение:

$$\left[\begin{array}{l} \left\{ \begin{array}{l} y_i + \tilde{v}(t_0 - t_i) \leqslant \frac{\tilde{v}}{v} \\ \frac{\tilde{v}}{v}(x - \rho) \leqslant y_i + \tilde{v}(t_0 - t_i) \leqslant \frac{\tilde{v}}{v}(x + \rho) \end{array} \right. \\ \left\{ \begin{array}{l} y_i + \tilde{v}(t_0 - t_i) > \frac{\tilde{v}}{v} \\ y_i + \tilde{v}(t_0 - t_i) \leqslant \frac{\tilde{v}}{v} + \rho + x - 1. \end{array} \right. \end{array} \right.$$

Лемма доказана.

Пусть $B \subset [0, \infty) \times [0, 1]$ и пусть $\xi_V(B)$ — случайная величина, обозначающая **число** объектов данных (t_i, y_i) из V, таких что $(t_i, y_i) \in B$, где случайные величины t_i образуют пуассоновский поток с параметром λ, то есть $t_{i+1} = t_i + \delta_i$, где δ_i для любого i распределено

по показательному закону с параметром λ, а случайная величина y_i для любого i равномерно распределена на отрезке $[0,1]$.

Рассмотрим множество

$$B_0 = \{(t,y) : y \in [0,a], t \in [t_1(y), t_2(y)]\},$$

где $t_1(y), t_2(y)$ произвольные непрерывные функции из множества $[0,1]$ в множество $[0,\infty)$, такие что $t_1(y) \leqslant t_2(y)$.

Пример множества B_0 изображен на рисунке 1.

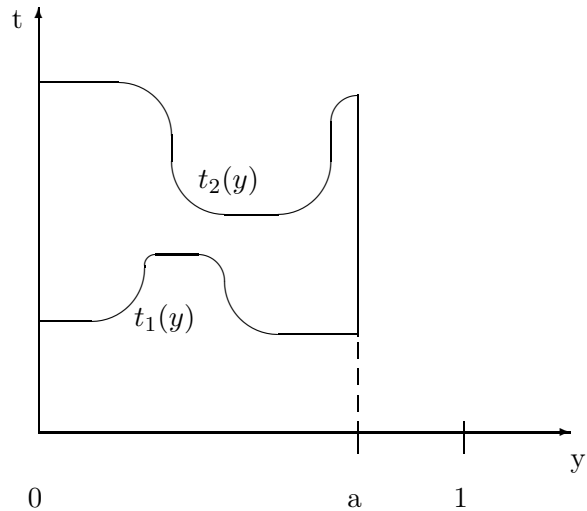

Рис. 1. Область B_0.

Лемма 2.

$$M_V \xi_V(B_0) = \lambda \int_0^a (t_2(y) - t_1(y))dy.$$

Доказательство. Обозначим $P_i^k(y)$ — вероятность, что из i объектов, появившихся на отрезке $[0,1]$, k объектов появится в фиксированном отрезке длины y, то есть

$$P_i^k(y) = C_i^k y^k (1-y)^{i-k}.$$

Заметим, что

$$P(\xi_V(B_0) = k) = \sum_{i=k}^{\infty} \int_0^a p_i(t_2(y) - t_1(y)) dP_i^k(y),$$

где $p_i(t)$ есть вероятность появления i объектов в промежуток времени длины t. В [4] показано, что для пуассоновского потока с параметром λ эта вероятность составляет:

$$p_i(t) = e^{-\lambda t} \frac{(\lambda t)^i}{i!}.$$

Посчитаем математическое ожидание:

$$\begin{aligned} M_V \xi_V(B_0) &= \sum_{k=0}^{\infty} kP(\xi_V(B_0) = k) = \\ &= \sum_{k=0}^{\infty} k \sum_{i=k}^{\infty} \int_0^a p_i(t_2(y) - t_1(y)) dP_i^k(y) = \\ &= \int_0^a \sum_{i=0}^{\infty} p_i(t_2(y) - t_1(y)) \sum_{k=0}^{i} k dP_i^k(y) = (*) \end{aligned}$$

При этом,

$$\sum_{k=0}^{i} k dP_i^k(y) = (\sum_{k=0}^{i} P_i^k(y) k)' = (\sum_{k=0}^{i} C_i^k y^k (1-y)^{i-k} k)' = (yi)' = i,$$

поэтому

$$(*) = \int_0^a \sum_{i=0}^{\infty} p_i(t_2(y) - t_1(y)) i dy = \lambda \int_0^a (t_2(y) - t_1(y)) dy.$$

Лемма доказана.

3. Алгоритм логарифмического поиска

Приведем алгоритмы решения задачи об опасной близости для случая фиксированных скоростей объектов-запросов ($\tilde{v}$) и объектов-данных (v), имеющий логарифмическую сложность.

3.1. Решение без погрешности

Приведем решение задачи об опасной близости без погрешности для случая, когда скорости объектов-запросов меньше скорости объектов-данных, то есть

$$\tilde{v} \geqslant v.$$

Из леммы 1 следует, что объект $o_i = (t_i, y_i)$ входит в ответ $J(\rho, q, V)$ на запрос $q = (t_0, x)$ тогда и только тогда, когда o_i принадлежит библиотеке V и выполнено:

$$\left[\begin{array}{l} \left\{\begin{array}{l} x \leqslant 1-\rho \\ t_0 - \frac{x+\rho}{v} \leqslant t_i - \frac{y_i}{\tilde{v}} \leqslant t_0 - \frac{x-\rho}{v}. \end{array}\right. \\ \left\{\begin{array}{l} x > 1-\rho \\ t_0 - \frac{x+\rho}{v} + \frac{1}{\tilde{v}} - \frac{1}{v} \leqslant t_i - \frac{y_i}{\tilde{v}} \leqslant t_0 - \frac{x-\rho}{v}. \end{array}\right. \end{array}\right. \tag{7}$$

Данная задача сводится к задаче одномерного интервального поиска. Поскольку данные меняются с течением времени (появляются «новые» объекты, исчезают «старые» объекты), то будем использовать ИГ, основанный на динамической структуре данных 2–3 дерева для упорядоченного множества параметров

$$s_i = t_i - \frac{y_i}{\tilde{v}}.$$

Опишем кратко структуру 2–3 дерева. Линейно упорядоченное множество (например, числа $r_1 \leqslant ... \leqslant r_n$) можно представить в виде 2–3 дерева следующим образом: элементы множества приписываем листьям слева направо по порядку. В каждой внутренней вершине w хранятся две величины : $L(w)$ максимальный элемент в левом поддереве узла w и $M(w)$ — максимальный элемент в среднем (если две дуги, то в правом) поддереве узла w.

В нашем случае будем использовать «прошитое» 2–3 дерево, то есть 2–3 дерево, в котором от каждого листа идет ребро вправо к соседнему листу.

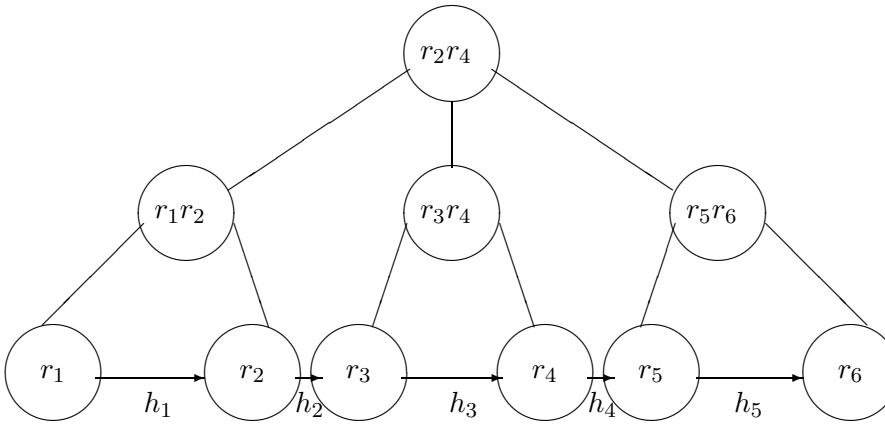

Рис. 2. Пример «прошитого» 2–3 дерева для $r_1 \leqslant r_2 \leqslant \cdots \leqslant r_6$.

«Прошитое» 2–3 дерево позволит осуществлять интервальный поиск в упорядоченном множестве $r_1 \leqslant ... \leqslant r_n$. Пример «прошитого» 2–3 дерева для $n = 6$ показан на рис. 2.

Структура 2–3 дерева подробно изучена в работах [2, 3]. Опишем операции поиска, вставки и удаления для «прошитого» 2–3 дерева применительно к решаемой задаче и оценим их сложность.

1) **Процедура поиска.**

а) Каждому запросу q поставим в соответствие числа A, B, обозначающие концы интервалов из (7), то есть

$$A = \begin{cases} t_0 - \frac{x+\rho}{v}, & \text{если } x \leqslant 1-\rho \\ t_0 - \frac{x+\rho}{v} + \frac{1}{\tilde{v}} - \frac{1}{v}, & \text{если } x > 1-\rho. \end{cases} \qquad (8)$$

$$B = t_0 - \frac{x-\rho}{v}.$$

Сложность вычисления A и B для каждого запроса q будем считать равной 0.

б) Операция поиска листа s_k ближайшего справа к A происходит путем последовательного **прохождения** внутренних

вершин, где под прохождением внутренней вершины w будем понимать следующий набор операций сравнения: если $A \leqslant L(w)$, то переходим к левому сыну w, иначе, если у w два сына, или $A \leqslant M(w)$, то переходим к среднему сыну w, иначе к правому сыну w.

Операцию прохождения внутренней вершины внесем в базовое множество $\mathcal{F}$ в качестве переключателя и обозначим ее сложность за γ.

В работе [3] показано, что операция поиска в 2–3 дереве имеет сложность не меньшую, чем $\gamma]\log_3 V[$ и не большую, чем $\gamma[\log_2 V]$.

в) Сравним s_k с B. Если $s_k \leqslant B$, то выдаем o_k в ответ и переходим по ребру к следующему листу.

Предикат сравнения также внесем в базовое множество $\mathcal{F}$ и будем считать, что этот предикат равен 1, если B меньше, чем число s_k приписанное листу.

г) Повторяем предыдущий пункт, пока значение предикатов, вычисляемых на ребрах, равняется 1.

Заметим, что число операций в этом пункте пропорционально длине ответа, а в алгоритме A_1 мы оцениваем сложность без перечисления ответа. Следовательно, для получения сложности без перечисления ответа нужно только учесть сложность пункта b.

Получаем, что

$$\gamma]\log_3 |V|[\leqslant T_{A_1}(\rho, q, V) \leqslant \gamma[\log_2 |V|].$$

2) **Процедура вставки.**

Для нового объекта-данного $o_k = (t_k, y_k)$ вычислим $s_k = t_k - \frac{y_k}{\bar{v}}$. Чтобы в 2–3 дерево *вставить* новый элемент s_k, нужно найти место для нового листа l, который будет содержать s_k. Для этого ищут элемент s_k в дереве. Если дерево содержит более одного элемента, то поиск s_k окончится в узле f, имеющем двух или трех сыновей, которые являются листами.

Если из узла f выходит только два листа — l_1 и l_2, то делаем l сыном узла f так, чтобы листья остались упорядоченными слева направо. Кроме того, нужно изменить пометки L и M в вершине f, а возможно еще и в нескольких предках f.

Теперь предположим, что у f уже есть три листа — l_1, l_2, l_3. Сделаем l сыном узла f,сохраняя упорядоченность листьев. Чтобы сохранить 2–3 свойство, образуем новый узел g. Два левых сына оставим сыновьями f, а два правых переделаем в сыновей узла g. Затем сделаем g братом узла f, сделав его сыном отца узла f. Эта операция называется **переброской**. Если отец узла f уже имел трех сыновей, то надо рекурсивно повторять переброску до тех пор, пока у всех узлов в дереве останется не более трех сыновей. Если у корня окажется четыре сына, то образуем новый корень с двумя новыми сыновьями, каждый из которых будет иметь в качестве двух своих сыновей двух из четырех сыновей старого корня.

Операцию переброски внесем в базовое множество элементарных преобразований H и обозначим ее сложность через γ_1.

Из [3] следует, что сложность вставки нового объекта o удовлетворяет соотношению:

$$\gamma] \log_3 |V|[\leqslant S_{A_1}(\rho, o, V) \leqslant (\gamma + \gamma_1)[\log_2 |V|].$$

3) **Процедура удаления.**

Для того, чтобы удалить объект o ищем лист, содержащий запись s, соответствующую объекту o и удаляем ее. Для этого начиная от корня ищем s в дереве. Пусть f — вершина, имеющая в качестве сына лист, соответствующий записи s. Удалим этот лист.

Если после удаления у вершины f осталось два сына, то заканчиваем процедуру.

Если же у вершины f остался только один сын, то **«склеим»** f с любым из ее братьев, объединив их листья. После этого, возможно f (точнее новая «склеенная») вершина будет единственным сыном своей родительской вершины. В этом случае повторим операцию «склейки» одним ярусом ближе к корню. Наконец, если после этих операций корень имеет только одного сына, то объявим этого сына новым корнем 2–3 дерева, а старый корень удалим.

Получившееся дерево все еще может не обладать 2–3 свойством, так как при «склейке» вершин количество сыновей может ока-

заться больше трех. Поэтому необходим еще один поход от листа (брата удаленного листа) до корня, проверяя число сыновей у вершин и по необходимости совершая переброски.

Операцию склейки внесем в базовое множество элементарных преобразований H и обозначим ее сложность через γ_2.

Из [3] следует, что сложность удаления любого объекта o удовлетворяет соотношению:

$$\gamma]\log_3|V|[\leqslant R_{A_1}(\rho,o,V)\leqslant(\gamma+\gamma_1+\gamma_2)[\log_2|V|].$$

Объем 2–3 дерева с $|V|$ вершинами варьируется от $\frac{3|V|-1}{2}$ до $2|V|-1$, а объем «прошитого» 2–3 дерева больше на $|V|-1$, следовательно,

$$\frac{5|V|-1}{2}\leqslant Q_{A_1}(\rho,V)\leqslant 3|V|-2.$$

Утверждение 1. *Пусть $\tilde{v}$ — скорость объектов-запросов, v — скорость объектов-данных, ρ — расстояние опасной близости.*

Пусть также V — библиотека объектов-данных, а Q — множество объектов-запросов.

Тогда алгоритм A_1 решает задачу об опасной близости (ρ,Q,V) для случая $\tilde{v}\leqslant v$ без погрешности. Для сложности поиска ответа на любой запрос q из Q, вставки любого объекта-данного o, удаления любого объекта-данного $\tilde{o}$ по алгоритму A_1 и для объема алгоритма A_1 верны следующие оценки:

$$\begin{gathered}\gamma]\log_3|V|[\leqslant T_{A_1}(\rho,q,V)\leqslant\gamma[\log_2|V|],\\ \gamma]\log_3|V|[\leqslant S_{A_1}(\rho,o,V)\leqslant(\gamma+\gamma_1)[\log_2|V|],\\ \gamma]\log_3|V|[\leqslant R_{A_1}(\rho,\tilde{o},V)\leqslant(\gamma+\gamma_1+\gamma_2)[\log_2|V|],\\ \frac{5|V|-1}{2}\leqslant Q_{A_1}(\rho,V)\leqslant 3|V|-2,\end{gathered}$$

где для ИГ, соответствующего алгоритму A_1, γ есть сложность операции прохождения внутренней вершины, γ_1 — сложность операции переброски, γ_2 — сложность операции склейки.

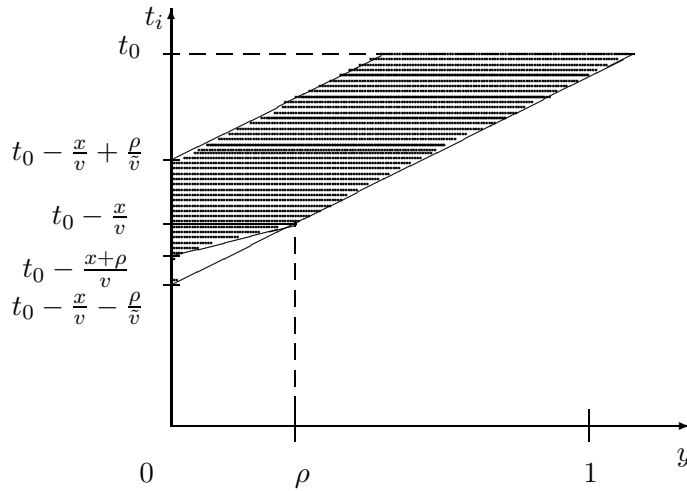

Рис. 3. Множество точек, удовлетворяющих запросу, для случая $v < \tilde{v}$.

3.2. Решение с погрешностью второго рода

Приведем решение задачи об опасной близости с погрешностью второго рода для случая, когда скорости объектов-запросов больше скорости объектов-данных, то есть

$$\tilde{v} < v.$$

Из леммы 1 следует, что объект $o_i = (t_i, y_i)$ входит в ответ $J(\rho, q, V)$ на запрос $q = (t_0, x)$ тогда и только тогда, когда o_i принадлежит библиотеке V и выполнено:

$$\left[\begin{array}{l} \left\{\begin{array}{l} t_i \geqslant t_0 - \frac{x}{v} \\ t_0 - \frac{x}{v} - \frac{\rho}{\tilde{v}} \leqslant t_i - \frac{y_i}{\tilde{v}} \leqslant t_0 - \frac{x}{v} + \frac{\rho}{\tilde{v}}. \end{array}\right. \\ \left\{\begin{array}{l} t_i < t_0 - \frac{x}{v} \\ t_i - \frac{y_i}{v} \geqslant t_0 - \frac{x+\rho}{v}. \end{array}\right. \end{array}\right. \tag{9}$$

На рисунке 3 изображено это множество. Рассмотрим следующие неравенства:

$$t_0 - \frac{x}{v} - \frac{\rho}{\tilde{v}} \leqslant t_i - \frac{y_i}{\tilde{v}} \leqslant t_0 - \frac{x}{v} + \frac{\rho}{\tilde{v}}. \tag{10}$$

Область, соответствующая неравенствам (10) заштрихована на рисунке 3.

Алгоритм A_2 построим таким образом, что множество объектов $o_i = (t_i, y_i)$ из библиотеки V, удовлетворяющих неравенствам (10), будет ответом $J_{A_2}(\rho, q, V)$ на запрос $q = (t_0, x_0)$, полученным в результате работы этого алгоритма.

Запишем условие для объектов $o_i = (t_i, y_i)$ из библиотеки V, при котором они попадают в множество $J_{A_2}(\rho, q, V) \backslash J(\rho, q, V)$, то есть для них (9) выполняется, а (10) — нет:

$$\begin{cases} t_i < t - \frac{x}{v} \\ t_0 - \frac{x}{v} + \frac{y_i - \rho}{v} \geqslant t_i \geqslant t_0 - \frac{x}{v} + \frac{y_i - \rho}{\tilde{v}}. \end{cases}$$

Заметим, что при $y_i > \rho$ эта система не имеет решения, поскольку первые 2 неравенства не могут выполняться одновременно. Получаем, что данная система эквивалентна:

$$\begin{cases} y_i \leqslant \rho \\ t_0 - \frac{x}{v} + \frac{y_i - \rho}{v} \geqslant t_i \geqslant t_0 - \frac{x}{v} + \frac{y_i - \rho}{\tilde{v}}. \end{cases}$$

Обозначим

$$B_0 = \{(t_i, y_i) : y_i \in [0, \rho], t_i \in [t_0 - \frac{x}{v} + \frac{y_i - \rho}{\tilde{v}}, t_0 - \frac{x}{v} + \frac{y_i - \rho}{v}]\}$$

и посчитаем погрешность второго рода по лемме 2:

$$f^2_{A_2}(\rho, \lambda) = M_u M_{\bar{V}} \xi(J(\rho, u, V) \backslash J_{A_2}(\rho, u, V)) = M_u M_V \xi(B_0) =$$
$$= \lambda \int_0^\rho (\rho - y)(\frac{1}{\tilde{v}} - \frac{1}{v}) dy = \frac{\lambda \rho^2 (v - \tilde{v})}{2v\tilde{v}}.$$

Также как в предыдущем случае в качестве ИГ будем использовать «прошитое» 2–3 дерево для множества параметров

$$s_i = t_i - \frac{y_i}{\tilde{v}}.$$

Операции вставки, удаления и поиска происходят абсолютно также как и в предыдущем случае, только по-другому определяются числа A, B (в соответствии с (10)):

$$A = t_0 - \frac{x}{v} - \frac{\rho}{\tilde{v}},$$
$$B = t_0 - \frac{x}{v} + \frac{\rho}{\tilde{v}}.$$

Утверждение 2. *Пусть $\tilde{v}$ — скорость объектов-запросов, v — скорость объектов-данных, ρ — расстояние опасной близости.*

Пусть также V — библиотека объектов-данных, такая что моменты появления объектов-данных образуют пуассоновский поток с параметром λ и их координаты появления равномерно распределены на отрезке $[0,1]$, а Q — множество объектов-запросов, такое что координаты появления объектов-запросов тоже равномерно распределены на отрезке $[0,1]$.

Тогда алгоритм A_2 решает задачу об опасной близости (ρ, Q, V) для случая $\tilde{v} < v$ с погрешностью второго рода. Для сложности поиска ответа на любой запрос q из Q, вставки любого объекта-данного o, удаления любого объекта-данного $\tilde{o}$ по алгоритму A_2, для объема и погрешности алгоритма A_2 верны следующие оценки:

$$\gamma]\log_3|V|[\leqslant T_{A_2}(\rho,q,V)\leqslant\gamma[\log_2|V|],$$
$$\gamma]\log_3|V|[\leqslant S_{A_2}(\rho,o,V)\leqslant(\gamma+\gamma_1)[\log_2|V|],$$
$$\gamma]\log_3|V|[\leqslant R_{A_2}(\rho,\tilde{o},V)\leqslant(\gamma+\gamma_1+\gamma_2)[\log_2|V|],$$
$$\frac{5|V|-1}{2}\leqslant Q_{A_2}(\rho,V)\leqslant 3|V|-2,$$
$$f^2_{A_2}(\rho,\lambda)=\frac{\lambda\rho^2(v-\tilde{v})}{2v\tilde{v}},\quad f^1_{A_2}(\rho,\lambda)=0,$$

где для ИГ, соответствующего алгоритму A_2, γ есть сложность операции прохождения внутренней вершины, γ_1 — сложность операции переброски, γ_2 — сложность операции склейки.

3.3. Решение с погрешностью первого рода

Приведем решение задачи об опасной близости с погрешностью первого рода для случая, когда скорости объектов-запросов больше скорости объектов-данных, то есть

$$\tilde{v} < v.$$

Рассмотрим следующую систему:

$$\left[\begin{array}{l}\left\{\begin{array}{l} y_i>\rho \\ t_0-\frac{x}{v}-\frac{\rho}{\tilde{v}}\leqslant t_i-\frac{y_i}{\tilde{v}}\leqslant t_0-\frac{x}{v}+\frac{\rho}{\tilde{v}}.\end{array}\right. \\ \left\{\begin{array}{l} y_i\leqslant\rho \\ t_0-\frac{x}{v}-\frac{\rho}{v}\leqslant t_i-\frac{y_i}{v}\leqslant t_0-\frac{x}{v}+\frac{\rho}{\tilde{v}}.\end{array}\right.\end{array}\right. \tag{11}$$

Решение системы (11) является подмножеством решения системы (9).

Алгоритм A_3 построим таким образом, что множество объектов $o_i = (t_i, y_i)$ из библиотеки V, удовлетворяющих неравенствам (11), будет ответом $J_{A_3}(\rho, q, V)$ на запрос $q = (t_0, x_0)$, полученным в результате работы этого алгоритма.

Запишем условие для объектов $o_i = (t_i, y_i)$ из библиотеки V, при котором они попадают в множество $J(\rho, q, V) \backslash J_{A_3}(\rho, q, V)$, то есть для них (9) выполняется, а (11) — нет:

$$\begin{cases} y_i \leqslant \rho \\ t_0 - \frac{x}{v} + \frac{y_i - \rho}{\tilde{v}} \leqslant t_i \leqslant t_0 - \frac{x}{v} - \frac{\rho}{\tilde{v}} + \frac{y_i}{v}. \end{cases}$$

Обозначим

$$B_0 = \{(t_i, y_i) : y_i \in [0, \rho], t_i \in [t_0 - \frac{x}{v} + \frac{y_i - \rho}{v}, t_0 - \frac{x}{v} + \frac{y_i - \rho}{\tilde{v}}]\}$$

Посчитаем погрешность первого рода по лемме 2:

$$f^2_{A_3}(\rho, \lambda) = M_u M_V \xi(B_0) = M_u M_V \xi(J(\rho, u, V) \backslash J_A(\rho, u, V)) =$$
$$= \lambda \int_0^\rho y_i (\frac{1}{v} - \frac{1}{v}) dy = \frac{\lambda \rho^2 (v - \tilde{v})}{2 v \tilde{v}}.$$

Опишем построение ИГ. Возьмем вершину, назовем ее корнем, припишем ей переключатель сравнения, действующий на множестве объектов-данных. Определим значение этого переключателя на объекте-данном $o = (t, y)$:

$$f(o) = \begin{cases} 1, & \text{если } y \leqslant \rho \\ 2, & \text{иначе} \end{cases}$$

и выпустим из корня 2 ребра с номерами $1, 2$.

Добавим переключатель сравнения в базовое множество $\mathcal{F}$ и обозначим его сложность через γ_0.

Будем считать, что на множестве объектов-запросов данный переключатель всегда принимает значение 1 и, соответственно, его вычисление не требуется, то есть запрос проходит по всем ребрам, выходящим из корня.

На концах обоих ребер в процессе решения задачи будем строить и модифицировать 2–3 деревья, аналогичные тем, которые использовались в предыдущих случаях. Обозначим через U_1, U_2 соответственно левое и правое «прошитые» 2–3 деревья, а через V_1, V_2 — множество листьев U_1, U_2 соответственно, причем $V_1 \cup V_2 = V$ и $V_1 \cap V_2 = \varnothing$.

1) **Процедура поиска**.

   а) Каждому запросу u поставим в соответствие числа A_1, B, A_2, обозначающие концы интервалов из (11), то есть

$$A_1 = t_0 - \frac{x}{v} - \frac{\rho}{\tilde{v}},$$
$$B = t_0 - \frac{x}{v} + \frac{\rho}{\tilde{v}},$$
$$A_2 = t_0 - \frac{x}{v} - \frac{\rho}{v}.$$

   б) Ищем ближайший к A_1 справа лист в U_1 — s_k, сравниваем s_k с B и переходим к перечислению ответа, затем ищем ближайший справа к A_2 — $s_{k'}$, сравниваем $s_{k'}$ с B и переходим к перечислению ответа.

Будем считать, что если $|V_i| = 0$, следовательно $T_{A_3}(\rho, q, V_i) = 0$. Тогда из предыдущих оценок следует, что сложность поиска по алгоритму A_3 составляет:

$$\gamma(\max(\log_3 |V_1|, 0) + \max(\log_3 |V_2|, 0)) \leqslant T_{A_3}(\rho, q, V) \leqslant$$
$$\leqslant \gamma(\max(\log_2 |V_1|, 0) + \max(\log_2 |V_2|, 0).$$

Ясно, что $\max(\log_3 |V_1|, 0) + \max(\log_3 |V_2|, 0)$ принимает свое наименьшее значение $\max(0, \log_3 |V|)$ при $|V_1| = 0, |V_2| = 2$.
Можно заметить также, что наибольшее значение

$$\max(\log_2 |V_1|, 0) + \max(\log_2 |V_2|, 0)$$

равняется наибольшему по всем $|V_1| + |V_2| = |V|$ значению от

$$\max(\log_2 |V_1| + \log_2 |V_2|, \log_2 |V_1|)$$

которое достигается в точке $|V_1| = |V_2| = \frac{|V|}{2}$ и равняется $\max(2 \log_2 |V| - 2, \log_2 |V|)$.

То есть, при $|V| \leqslant 3$

$$0 \leqslant T_{A_3}(\rho, q, V) \leqslant \gamma \log_2 |V|,$$

а при $|V| \geqslant 4$

$$\gamma \log_3 |V| \leqslant T_{A_3}(\rho, q, V) \leqslant \gamma(2 \log_2 |V| - 2).$$

2) **Процедура вставки**.
Пусть появился новый объект $o = (t, y)$. Тогда вычислим переключатель сравнения $f(o)$.

а) Если значение переключателя равно 1, то будем производить вставку в прошитое 2–3 дерево U_1. Из пункта 3.1 следует, что

$$\gamma \max(\log_3 |V_1|, 0) \leqslant S_{A_3}(\rho, o, V_1) \leqslant (\gamma + \gamma_1)[\log_2 |V_1|].$$

б) Если значение переключателя равно 2, то будем производить вставку в прошитое 2–3 дерево U_2. Из пункта 3.1 следует, что

$$\gamma \max(\log_3 |V_2|, 0) \leqslant S_{A_3}(\rho, o, V_2) \leqslant (\gamma + \gamma_1)[\log_2 |V_2|].$$

Так как

$$\gamma_0 + \min(S_{A_3}(\rho, o, V_1), S_{A_3}(\rho, o, V_2)) \leqslant S_{A_3}(\rho, q, V) \leqslant \gamma_0 + \\ + \max(S_{A_3}(\rho, o, V_1), S_{A_3}(\rho, o, V_2)),$$

то, получаем

$$\gamma_0 + \leqslant S_{A_3}(\rho, o, V) \leqslant \gamma_0 + (\gamma + \gamma_1)[\log_2 |V|].$$

3) **Процедура удаления**. Пусть требуется удалить объект $o = (t, y)$. Тогда вычислим переключатель сравнения $f(o)$.

а) Если значение переключателя равно 1, то будем производить удаление в прошитом 2–3 дереве U_1. Из пункта 3.1 следует, что

$$\gamma \max(\log_3 |V_1|, 0) \leqslant R_{A_3}(\rho, o, V_1) \leqslant (\gamma + \gamma_1 + \gamma_2)[\log_2 |V_1|].$$

б) Если значение переключателя равно 2, то будем производить удаление в прошитом 2–3 дереве U_2. Из пункта 3.1 следует, что

$$\gamma \max(\log_3 |V_2|, 0) \leqslant R_{A_3}(\rho, o, V_2) \leqslant (\gamma + \gamma_1 + \gamma_2)[\log_2 |V_2|].$$

Так как

$$\gamma_0 + \min(R_{A_3}(\rho, o, V_1), R_{A_3}(\rho, o, V_2)) \leqslant R_{A_3}(\rho, q, V) \leqslant \gamma_0 + \\ + \max(R_{A_3}(\rho, o, V_1), R_{A_3}(\rho, o, V_2)),$$

то, получаем

$$\gamma_0 \leqslant R_{A_3}(\rho, o, V) \leqslant \gamma_0 + (\gamma + \gamma_1 + \gamma_2)[\log_2 |V|].$$

Утверждение 3. *Пусть $\tilde{v}$ — скорость объектов-запросов, v — скорость объектов-данных, ρ — расстояние опасной близости.*

Пусть также V — библиотека объектов-данных, такая что, моменты появления объектов-данных образуют пуассоновский поток с параметром λ и их координаты появления равномерно распределены на отрезке $[0, 1]$, а Q — множество объектов-запросов, такое что координаты появления объектов-запросов тоже равномерно распределены на отрезке $[0, 1]$.

Тогда алгоритм A_3 решает задачу об опасной близости (ρ, Q, V) для случая $\tilde{v} < v$ с погрешностью первого рода. Для сложности поиска ответа на любой запрос q из Q по алгоритму A верны следующие оценки:
при $|V| \leqslant 3$

$$0 \leqslant T_{A_3}(\rho, q, V) \leqslant \gamma \log_2 |V|,$$

а при $|V| \geqslant 4$

$$\gamma \log_3 |V| \leqslant T_{A_3}(\rho, q, V) \leqslant \gamma(2 \log_2 |V| - 2).$$

Для сложности вставки любого объекта-данного o, удаления любого объекта-данного $\tilde{o}$ по алгоритму A_3, для объема и погрешности алгоритма A_3 верны следующие оценки:

$$\gamma \log_3 |V| \leqslant T_{A_3}(\rho, q, V) \leqslant \gamma(2 \log_2 |V| - 2),$$
$$\gamma_0 \leqslant S_{A_3}(\rho, o, V) \leqslant \gamma_0 + (\gamma + \gamma_1 + \gamma_2)[\log_2 |V|],$$
$$\gamma_0 \leqslant R_{A_3}(\rho, o, V) \leqslant \gamma_0 + (\gamma + \gamma_1 + \gamma_2)[\log_2 |V|],$$

$$\frac{5|V|+3}{2} \leqslant Q_{A_3}(\rho, V) \leqslant 3|V|,$$
$$f^2_{A_3}(\rho,\lambda) = 0, \quad f^2_{A_3}(\rho,\lambda) = \frac{\lambda\rho^2(v-\tilde{v})}{2v\tilde{v}},$$

где для ИГ, соответствующего алгоритму A_3, γ есть сложность операции прохождения внутренней вершины, γ_1 — сложность операции переброски, γ_2 — сложность операции склейки, а γ_0 — сложность вычисления переключателя сравнения.

Осталось заметить, что Теорема 1 является следствием Утверждений 1, 2 и 3.

Автор выражает благодарность профессору Э. Э. Гасанову за постановку задачи и помощь в работе.

Список литературы

[1] Гасанов Э. Э., Кудрявцев В. Б. Теория хранения и поиска информации. М.: ФИЗМАТЛИТ, 2002.

[2] Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. М.: Мир, 1979.

[3] Лапшов И. С. Динамические базы данных с оптимальной по порядку временной сложностью // Дискретная математика, в печати.

[4] Булинский А. В., Ширяев А. Н. Теория случайных процессов. М., 2001.