Фоновый алгоритм решения двумерной задачи о доминировании

Э. Э. Гасанов, Д. В. Ефремов

В работе исследуются алгоритмы поиска, используемые в фоновом режиме, и предлагается математическая модель этих алгоритмов, опирающаяся на понятие информационного графа. В работе также предлагается фоновый алгоритм решения двумерной задачи о доминировании с линейными затратами по памяти и константным в среднем временем поиска. Для сравнения отметим, что нефоновый алгоритм, среднее время поиска которого равно константе плюс среднее время перечисления ответа, требует квадратичных затрат по памяти.

Ключевые слова: информационно-графовая модель, задача о доминировании, фоновый поиск.

1. Введение

При решении задач информационного поиска часто возникают ситуации, когда эти задачи можно решать в так называемом фоновом режиме. Приведем два простых примера.

Пользователь некой информационно-справочной системы дал системе запрос на поиск и теперь ждет ответ. При этом вовсе не обязательно, чтобы система подготовила все документы, входящие в ответ, а затем выдала их пользователю. Можно сразу же после нахождения первого документа выдать его пользователю, а пока он осмысливает и обрабатывает этот документ, осуществить поиск второго, третьего и остальных документов и выдавать их по мере обработки пользователем очередного документа из ответа.

Второй пример. Пусть на компьютере имеются два процесса — один выполняет функции информационной системы, то есть осуществляет поиск в базе данных по запросу и подготовку документов

из ответа, а второй отвечает за отсылку найденных документов через сети связи. При этом первый процесс может по мере нахождения и подготовки очередного элемента ответа закладывать его в буфер обмена, а второй процесс может по мере появления в буфере обмена документов отсылать их абонентам.

В обоих примерах в качестве временной сложности алгоритма поиска имеет смысл брать суммарное время ожидания очередного элемента ответа пользователем справочной системы в первом случае и вторым процессом — во втором. В идеальном случае, когда время обработки каждого элемента ответа пользователем или вторым процессом больше времени поиска следующего элемента ответа информационной системой, эта временная сложность равна времени ожидания первого элемента ответа.

Отсюда видно, что к алгоритмам для работы в фоновом режиме предъявляются другие требования, по сравнению с обычными алгоритмами. Для фоновых алгоритмов особенно важна способность быстро получать первый элемент ответа, тогда как все остальные можно получать не так быстро, за отрезки времени, приблизительно равные времени обработки одного элемента ответа пользователем алгоритма.

Заметим также, что в задачах, в которых предполагается, что ответ содержит только один элемент, понятие фонового алгоритма теряет смысл.

В работе предлагается математическая модель, предназначенная для исследования сложности фоновых алгоритмов поиска, и в рамках этой модели исследуется известная геометрическая задача поиска — двумерная задача о доминировании, которая состоит в поиске в конечном подмножестве точек 2-мерного пространства всех тех точек, которые не превышают по обоим координатам 2-мерный вектор-запрос. В данной работе для этой задачи получен алгоритм с константной в среднем временной сложностью и линейными затратами по памяти. Отметим, что обычный, не фоновый, алгоритм с аналогичной временной сложностью требует квадратичных затрат по памяти. В работе [1] предлагался алгоритм с линейными затратами памяти и константной в среднем временной сложностью для большинства баз данных, но там же было показано, что существуют экзотические базы данных, для которых время поиска этим алгоритмом в среднем пропорционально мощности базы данных.

Результаты данной работы были анонсированы в [2].

Авторы выражают благодарность В. Б. Кудрявцеву и А. С. Подколзину за поддержку в работе.

2. Основные понятия и формулировка результата

В соответствии с [3] опишем математическую модель фоновых алгоритмов поиска.

В качестве модели фоновых алгоритмов поиска предлагается использовать информационные графы (ИГ), но сложность ИГ предлагается вводить по-другому, так, чтобы она отражала сложность фоновых алгоритмов.

Пусть X — множество запросов с заданным на нем вероятностным пространством $\langle X, \sigma, \mathbf{P}\rangle$, где σ — алгебра подмножеств множества X, $\mathbf{P}$ — вероятностная мера на σ; Y — множество записей (объектов поиска); ρ — бинарное отношение на $X \times Y$, называемое отношением поиска. Пятерку $S = \langle X, Y, \rho, \sigma, \mathbf{P}\rangle$ будем называть *типом*. Тройку $I = \langle X, V, \rho\rangle$, где V — некоторое конечное подмножество множества Y, в дальнейшем называемое *библиотекой*, будем называть задачей информационного поиска (ЗИП) типа S, и будем считать, что ЗИП $I = \langle X, V, \rho\rangle$ содержательно состоит в перечислении для произвольно взятого запроса $x \in X$ всех тех и только тех записей $y \in V$ таких, что $x\rho y$.

Пусть f — одноместный предикат, определенный на X , то есть $f : X \to \{0, 1\}$. Множество $N_f = \{x \in X : f(x) = 1\}$ назовем *характеристическим множеством* предиката f.

Множество $O(y, \rho) = \{x \in X : x\rho y\}$ назовем *тенью* записи $y \in Y$.

Функцию $\chi_{y,\rho} : X \to \{0, 1\}$ такую, что $N_{\chi_{y,\rho}} = O(y, \rho)$ назовем *характеристической функцией* записи y.

Пусть F — множество символов одноместных предикатов, определенных на множестве X, G — множество символов одноместных переключателей, определенных на множестве X. Под переключателем будем понимать функцию, областью значений которой является начальный отрезок натурального ряда. Пару $\mathcal{F} = \langle F, G\rangle$ назовем базовым множеством.

Понятие *информационного графа* (ИГ) над базовым множеством $\mathcal{F} = \langle F, G \rangle$ определяется следующим образом. Берется конечная многополюсная ориентированная сеть. В ней выбирается некоторый полюс, который называется корнем. Остальные полюса называются листьями и им приписываются записи из Y, причем разным листьям могут быть приписаны одинаковые записи. Некоторые вершины сети (в том числе это могут быть и полюса) называются переключательными и им приписываются переключатели из G. Ребра, исходящие из каждой из переключательных вершин, нумеруются подряд, начиная с 1, и называются переключательными ребрами. Ребра, не являющиеся переключательными, называются предикатными и им приписываются предикаты из множества F. Таким образом нагруженную многополюсную ориентированную сеть называем ИГ над базовым множеством $\mathcal{F} = \langle F, G \rangle$.

Функционирование ИГ определяется следующим образом. Скажем, что предикатное ребро проводит запрос $x \in X$, если предикат, приписанный этому ребру, принимает значение 1 на запросе x, Переключательное ребро, которому приписан номер n, проводит запрос $x \in X$, если переключатель, приписанный началу этого ребра, принимает значение n на запросе x. Ориентированная цепочка ребер проводит запрос $x \in X$, если каждое ребро цепочки проводит запрос x. Запрос $x \in X$ проходит в вершину β ИГ, если существует ориентированная цепочка, ведущая из корня в вершину β, которая проводит запрос x. Запись y, приписанная листу α, попадает в ответ ИГ на запрос $x \in X$, если запрос x проходит в лист α. Ответом ИГ U на запрос x назовем множество записей, попавших в ответ ИГ на запрос x, и обозначим его $\mathcal{J}_U(x)$. Эту функцию $\mathcal{J}_U(x)$ будем считать результатом функционирования ИГ U.

Пусть нам дана ЗИП $I = \langle X, V, \rho \rangle$. Скажем, что ИГ U *решает* ЗИП $I = \langle X, V, \rho \rangle$, если $\mathcal{J}_U(x) = \{y \in V : x \rho y\}$.

Пусть β — некоторая вершина ИГ. Предикат, определенный на множестве запросов, который принимает значение 1 на запросе x, если запрос проходит в вершину β, и 0 — в противном случае, назовем функцией фильтра вершины β и обозначим $\varphi_\beta(x)$.

Объемом $Q(U)$ ИГ U назовем число ребер в графе U.

Введем понятие *фоновой сложности ИГ*.

Поскольку для фоновых алгоритмов важны порядок и моменты появления записей в ответе, определим некоторую процедуру обхода

графа U, который в дальнейшем поможет определить новое понятие сложности нашего графа.

Для того, чтобы описать эту процедуру, сделаем следующие предположения:

- будем считать, что каждому ребру мы можем временно приписывать номер, который назовем путевым;
- можем отмечать вершины некоторым образом;
- назовем множество ребер, исходящих из одной вершины, пучком и будем считать, что в каждом пучке определен порядок следования ребер, то есть существует ребро первое, второе и т. д.

Теперь мы можем перейти к описанию процедуры обхода графа U, которая представляет собой линеаризацию во времени процесса поиска при помощи графа U. Входными данными процедуры обхода $\mathcal{B}$ будут информационный граф U и запрос $x \in X$. Выходными данными — упорядоченное множество $\mathcal{J}$ записей, входящих в ответ на запрос x, и последовательность $\mathcal{T}$ моментов появления записей в ответе.

Итак, пусть нам даны ИГ U и запрос x, тогда процедуру обхода $\mathcal{B}$ можно описать следующим образом.

1) (Инициализация процесса обхода)
   а) Сотрем со всех вершин отметки.
   б) Объявим текущей вершиной корень графа U.
   в) Обнулим счетчик времени $t = 0$.
   г) Присвоим счетчику путевых номеров значение $m = 1$.
   д) Присвоим начальное значение множествам $\mathcal{J}$ и $\mathcal{T}$ $\mathcal{J} = \varnothing$ и $\mathcal{T} = \varnothing$.
   е) Перейдем к пункту 2.
2) (Прямой ход)
   а) Если текущая вершина — непомеченный лист графа U, то
      – заносим в конец множества $\mathcal{J}$ запись, принадлежащую текущей вершине;
      – заносим в конец множества $\mathcal{T}$ значение счетчика времени t;

– переходим к пункту 2(b).

б) Отмечаем текущую вершину.

в) Если текущая вершина есть точка переключения, то:
   – вычисляем значение переключателя, приписанного текущей вершине, на запросе x (пусть это значение равно n);
   – увеличиваем на 1 значение счетчика времени: $t = t + 1$;
   – если среди ребер, исходящих из текущей вершины, нет ребра с номером n, то переходим к пункту 3;
   – если конец ребра с номером n отмеченная вершина, то переходим к пункту 3;
   – ребру с номером n, исходящему из текущей вершины, приписываем путевой номер m, объявляем конец этого ребра текущей вершиной, присваиваем $m = m + 1$ и возвращаемся к пункту 2.

г) Если текущая вершина не является точкой переключения, то:
   – если из текущей вершины не исходит ни одно ребро или из нее исходит ребро, имеющее путевой номер, и это ребро последнее по порядку следования в пучке текущей вершины, то идем к пункту 3;
   – если из текущей вершины исходит ребро, имеющее путевой номер, то стираем его с этого ребра и выбираем следующее по порядку следования в пучке ребро; если же из текущей вершины не исходит ребро, имеющее путевой номер, то выбираем первое по порядку следования в пучке текущей вершины ребро;
   – присваиваем выбранному ребру путевой номер m;
   – вычисляем приписанный выбранному ребру предикат на запросе x (пусть его значение равно n);
   – увеличиваем на 1 значение счетчика времени: $t = t + 1$;
   – если $n = 0$, то возвращаемся к пункту 2;
   – если конец выбранного ребра отмеченная вершина, то возвращаемся к пункту 2;

– если $n = 1$, то объявляем конец выбранного ребра текущей вершиной, присваиваем $m = m+1$ и возвращаемся к пункту 2.

3) (Обратный ход)

а) Если из текущей вершины исходит ребро, имеющее путевой номер, то стираем с ребра этот номер.

б) Уменьшаем значение счетчика путевых номеров: $m = m - 1$.

в) Если $m = 0$, то заносим в конец множества $\mathcal{T}$ значение счетчика времени и завершаем работу процедуры.

г) Если $m > 0$, то, значит, существует ребро, входящее в текущую вершину и имеющее путевой номер m, выбираем его.

д) Объявляем текущей вершиной начало выбранного ребра.

е) Если текущая вершина является точкой переключения, то возвращаемся к пункту 3.

ж) Иначе возвращаемся к пункту 2.

Легко видеть, что данная процедура обхода позволяет посетить все вершины, функции фильтров которых принимают значение 1 на запросе x. Значит, на выходе алгоритма мы получаем множество $\mathcal{J} = \{y_{i_1}, \ldots, y_{i_m}\}$, совпадающее с ответом $\mathcal{J}_U(x)$ графа U на запрос x, и множество

$$\mathcal{T} = \{t_1, t_2, \ldots, t_m, t_{m+1}\},$$

где t_i — время появления в ответе i-ой записи ($i = 1, \ldots, m$), t_{m+1} — время работы процедуры, выраженное в количестве вычисленных предикатов и переключателей. При желании определить его более точно надо сопоставить каждому предикату и переключателю число, равное времени его вычисления. Тогда время работы процедуры будет равно сумме времен вычисления всех переключателей и предикатов. Можно также учитывать время обратного хода, изменяя соответствующим образом значение счетчика времени на шаге 3 процедуры. Но в данной работе мы рассматриваем лишь приближение к реальному времени вычисления, то есть мы пренебрегаем всеми затратами времени, кроме вычисления предикатов и переключателей,

причем считаем, что каждый из них вычисляется за время, равное одному такту.

Введем теперь сложность графа U так, чтобы она отражала сущность фонового алгоритма.

Пусть нам даны ИГ U, запрос x, а также некоторый субъект, которого назовем *пользователем*. Пусть на обработку i-ой записи ответа пользователю требуется время τ_i, $i = 1, \ldots, m$. Предполагаем, что $\tau_1, \tau_2, \ldots, \tau_m$ — независимые одинаково распределенные действительные случайные величины. Тогда за сложность графа U целесообразно принять не все время работы процедуры t_{m+1}, а только то время, которое пользователь тратит «впустую», ожидая появления очередной записи. Обозначим эту величину через $T(U, x, \tau_1, \ldots, \tau_m)$.

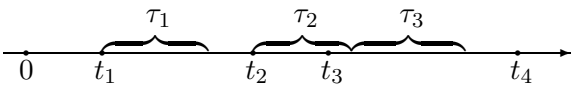

Рис. 1. Моменты появления элементов ответа и интервалы «раздумий».

Рассмотрим пример, изображенный на рисунке 1. В этом примере ответ содержит 3 записи. Пользователь находится в состоянии ожидания трижды: сначала в ожидании первой записи в течение t_1 тактов времени, затем в ожидании второй записи в течение $t_2 - (t_1 + \tau_1)$ и, наконец, в ожидании завершения работы алгоритма (чтобы убедиться, что больше записей в ответе нет) в течение $t_4 - (t_2 + \tau_2 + \tau_3)$. Таким образом, в этом примере

$$T(U, x, \tau_1, \tau_2, \tau_3) = t_1 + t_2 - (t_1 + \tau_1) + t_4 - (t_2 + \tau_2 + \tau_3) = t_4 - \tau_1 - \tau_2 - \tau_3.$$

Этот пример иллюстрирует следующий очевидный факт, если t_j — это последний момент времени, перед которым пользователь находился в состоянии ожидания, то

$$T(U, x, \tau_1, \ldots, \tau_m) = t_j - \sum_{i=1}^{j-1} \tau_i.$$

Или это можно записать иначе:

$$T(U, x, \tau_1, \ldots, \tau_m) = \max_{1 \leqslant j \leqslant m+1} (t_j - \sum_{i=1}^{j-1} \tau_i).$$

Функция $T(U, x, \tau_1, \ldots, \tau_m)$ является случайной величиной, как измеримая функция от случайных величин $\tau_1, \ldots, \tau_m$. Тогда мы можем определить сложность графа U на запросе x как математическое ожидание этой случайной величины $T(U, x, \tau_1, \ldots, \tau_m)$, то есть

$$T(U, x) = \mathbf{M}_{\tau_1,\ldots,\tau_m} T(U, x, \tau_1, \ldots, \tau_m).$$

Скажем, что базовое множество $\mathcal{F}$ *измеримое*, если каждая функция из $\mathcal{F}$ — измеримая (относительно алгебры σ).

В [3, стр. 188] доказана справедливость следующей леммы.

Лемма 1. *Если базовое множество $\mathcal{F}$ измеримое, то для любого ИГ U над базовым множеством $\mathcal{F}$ функция $T(U, x)$, как функция от x, является случайной величиной.*

Далее всюду будем предполагать, что базовое множество измеримое.

Сложностью ИГ U назовем математическое ожидание величины $T(U, x)$, то есть число

$$T(U) = \mathbf{M}_x T(U, x).$$

Пусть нам дана некая ЗИП I. Сложностью задачи I при базовом множестве $\mathcal{F}$ и заданном объеме q назовем число

$$T(I, \mathcal{F}, q) = \inf\{T(U) : U \in \mathcal{U}(I, \mathcal{F}) \text{ и } Q(U) \leqslant q\},$$

где $\mathcal{U}(I, \mathcal{F})$ — множество всех ИГ над базовым множеством $\mathcal{F}$, решающих ЗИП I.

Число

$$T(I, \mathcal{F}) = \inf\{T(U) : U \in \mathcal{U}(I, \mathcal{F})\}$$

назовем сложностью задачи I при базовом множестве $\mathcal{F}$.

Таким образом, введя новое понятие сложности ИГ и соответствующее ей понятие сложности ЗИП, мы завершаем определение математической модели фоновых алгоритмов поиска.

На множестве $[0, 1]^2 \times [0, 1]^2$ введем отношение «$\leqslant$» следующим образом:

$$(y^1, y^2) \leqslant (x^1, x^2) \Longleftrightarrow y^1 \leqslant x^1 \& y^2 \leqslant x^2.$$

Пусть $Y = [0,1) \times [0,1]$ — множество записей; $V = \{y_1, \ldots, y_k : y_i = (y_i^1, y_i^2),\ i = 1, \ldots, k\} \subseteq Y$ — библиотека; $X = [0,1) \times [0,1]$ — множество запросов; отношение поиска ρ задается на $X \times Y$ следующим соотношением:

$$x \rho y \Longleftrightarrow y \leqslant x,$$

где $x = (x^1, x^2)$, $y = (y^1, y^2)$, то есть $x \rho y \Longleftrightarrow y \in [0, x^1] \times [0, x^2]$. Тогда ЗИП $I = \langle X, V, \rho \rangle$ называется двумерной задачей о доминировании.

Здесь у квадратов X и Y выброшена правая сторона только в целях удобства дальнейшего изложения.

Пусть $0 \leqslant a < b \leqslant 1$,

$$A_{a,b}^{i,m} = [a + (i-1)(b-a)/m, a + i(b-a)/m),\ i = 1, \ldots, m. \quad (1)$$

Пусть $g_{a,b}^m(x^1, x^2)$ — переключатель такой, что

$$g_{a,b}^m(x^1, x^2) = [(x^1 - a) \cdot m/(b-a)] + 1, \quad (2)$$

где $[a]$ — наибольшее целое, не большее, чем a.

Отметим, что

$$g_{a,b}^m(x^1, x^2) = i, \text{ если } x^1 \in A_{a,b}^{i,m}\ (i = 1, \ldots, m).$$

Пусть

$$g_a(x^1, x^2) = \begin{cases} 1, & \text{если } x^1 < a \\ 2, & \text{в противном случае} \end{cases},\ a \in [0,1], \quad (3)$$

$$f_a(x^1, x^2) = \begin{cases} 1, & \text{если } x^2 \geqslant a \\ 0, & \text{в противном случае} \end{cases},\ a \in [0,1]. \quad (4)$$

Пусть

$$G_1 = \{g_{a,b}^m(x^1, x^2) : 0 \leqslant a < b \leqslant 1, m \in \mathbf{N}\}, \quad (5)$$

где $\mathbf{N}$ — множество натуральных чисел,

$$G_2 = \{g_a(x^1, x^2) : a \in [0,1]\}, \quad (6)$$

$$F = \{f_a(x^1, x^2) : a \in [0,1]\}. \quad (7)$$

Пусть

$$\mathcal{F} = \langle F, G_1 \cup G_2 \rangle - \quad (8)$$

базовое множество

Будем писать $A(n) \precsim B(n)$ при $n \to \infty$, если существует такая положительная константа c, что, начиная с некоторого номера n_0, $A(n) \leqslant c \cdot B(n)$. Если $A \precsim B$ и $B \precsim A$, то будем говорить, что A и B *равны по порядку* при $n \to \infty$ и обозначать $A \asymp B$.

Справедлива следующая теорема.

Теорема 1. *Пусть $I = \langle X, V, \rho \rangle$ — двумерная задача о доминировании, где $|V| = k$, $\mathcal{F}$ — базовое множество, определяемое соотношениями (2)–(8). Пусть на X задано вероятностное пространство $\langle X, \sigma, \mathbf{P} \rangle$, и функция плотности вероятности $p(x, y)$, определяющая вероятностную меру $\mathbf{P}$, — ограничена. Тогда если время обработки пользователем любой записи из ответа не меньше, чем 5, то существует такой информационный граф $U \in \mathcal{U}(I, \mathcal{F})$, что*

$$T(U) < 3, \quad Q(U) \asymp k,$$

при $k \to \infty$.

3. Решение задачи для однослойных библиотек

Скажем, что пара точек $(y^1, y^2), (x^1, x^2) \in [0, 1]^2$ *сравнима*, если $(y^1, y^2) \leqslant (x^1, x^2) \vee (x^1, x^2) \leqslant (y^1, y^2)$, в противном случае эта пара точек называется *несравнимой*.

Множество точек $S \subseteq [0, 1]^2$ назовем *слоем*, если любая пара точек из S не сравнима.

В этом разделе мы предложим решение двумерной задачи о доминировании при условии, что библиотека V является слоем.

Дадим неформальное описание алгоритма. Пусть на вход алгоритма поступает запрос (x^1, x^2).

В библиотеке, упорядоченной по возрастанию абсцисс точек из библиотеки, находится наибольшая по абсциссе точка, абсцисса которой не больше, чем x^1. Эта задача соответствует второй задаче о близости, описанной, например, в [3, раздел 2.3] или [4]. Решать эту задачу будем методом, описанным в [3, Теорема 13] или [4, Теорема 2]. Согласно этим теоремам ее можно решить за константное в среднем время с линейными затратами памяти.

Далее, начиная с найденной точки, просматриваем точки библиотеки в направлении убывания абсцисс и включаем их в ответ пока просматриваемые точки по ординате не больше чем x^2. Процесс прекращается, если очередная точка по ординате больше чем x^2 или если просмотрены все точки библиотеки.

Пусть $a, b, c \in [0,1]$, $a < b$. Рассмотрим область

$$A_{a,b,c} = \{(x^1, x^2) : a \leqslant x^1 < b, c \leqslant x^2 \leqslant 1\}. \tag{9}$$

Если вероятностная мера на $[0,1]^2$ задается функцией плотности вероятности $p(x^1, x^2)$, то вероятность этой области равна

$$\mathbf{P}(A_{a,b,c}) = \int_a^b \int_c^1 p(x^1, x^2) dx^1 dx^2.$$

Описанная задача для однослойной библиотеки, лежащей в области $A_{a,b,c}$, будет использоваться как вспомогательная при условии, что запрос находится в области $A_{a,b,c}$. Поэтому определяем вероятностную меру на области $A_{a,b,c}$ как условную вероятность при условии, что запрос принадлежит $A_{a,b,c}$, и она определяется функцией плотности вероятности

$$p_{a,b,c}(x^1, x^2) = p(x^1, x^2)/\mathbf{P}(A_{a,b,c}).$$

Здесь естественно предполагается, что $\mathbf{P}(A_{a,b,c}) > 0$.

Лемма 2. *Если $I = \langle A_{a,b,c}, V, \rho\rangle$ — двумерная задача о доминировании, такая что V — слой мощности k, и $V \subset A_{a,b,c}$, $\mathbf{P}(A_{a,b,c}) > 0$, время обработки пользователем любой записи из ответа не меньше, чем 1, $\mathcal{F}$ — базовое множество, определяемое соотношениями (2)–(8), функция плотности вероятности $p(x^1, x^2)$, определяющая вероятностную меру $\mathbf{P}$ на $[0,1]^2$, ограничена константой C, то существует ИГ $U^V_{a,b,c} \in \mathcal{U}(I, \mathcal{F})$ такой, что $Q(U^V_{a,b,c}) \leqslant k(4 + C(b-a)(1-c)/\mathbf{P}(A_{a,b,c}))$, $T(U^V_{a,b,c}) \leqslant 3$.*

Доказательство. Пусть $V = \{(y^1_1, y^2_1), (y^1_2, y^2_2), \ldots, (y^1_k, y^2_k)\}$ и $y^1_1 < y^1_2 < \cdots < y^1_k$. Пусть $B = \{y^1_1, y^1_2, \ldots, y^1_k\}$ — упорядоченное по возрастанию множество абсцисс точек из V (у точки (y^1_i, y^2_i) число y^1_i — абсцисса, y^2_i — ордината). В множестве V все точки имеют различные абсциссы, поскольку V — слой.

Пусть $m =]kC(b-a)(1-c)/\mathbf{P}(A_{a,b,c})[$. Так как $C \geqslant 1$, то $m \geqslant k$. Отметим, что для равномерной вероятностной меры $C = 1$ и $C(b-a)(1-c)/\mathbf{P}(A_{a,b,c}) = 1$.

Построим ИГ U_m^1, который будет решать вторую задачу о близости для множества B.

Возьмем вершину β_0 и объявим ее корнем графа. Выпустим из β_0 m ребер, припишем им числа от 1 до m, объявим β_0 точкой переключения и припишем ей переключатель $g_{a,b}^m$.

Пусть $A_{a,b}^{i,m}$ — множества, определяемые соотношениями (1), $B_i = A_{a,b}^{i,m} \cap B,\ l_i = |B_i|,\ i = 1, \ldots, m$.

Конец ребра с номером i обозначим β_i.

Для всех i таких, что $B_i \neq \varnothing$, выполним следующую процедуру. Выпустим из вершины β_i бинарное сбалансированное дерево D^{B_i} с $l_i + 1$ концевыми вершинами и высоты $]\log_2(l_i + 1)[$.

Здесь и далее $]a[$ — наименьшее целое, не меньшее, чем a, *бинарное дерево* — это ориентированное дерево, в каждую вершину которого, кроме корня, входит одно ребро, и из каждой вершины которого, кроме концевых вершин, исходит два ребра; *корень* — это вершина, в которую не входит ни одно ребро; *концевые вершины* — это вершины из которых не исходит ни одно ребро; *высота вершины* — это длина пути от корня к данной вершине; *сбалансированное дерево* — это дерево, высота концевых вершин которого отличается не более чем на 1; *высота дерева* — это максимальная высота концевых вершин дерева.

Сопоставим концевым вершинам этого дерева D^{B_i}, кроме первой (самой левой), слева направо в порядке возрастания элементы из B_i.

Здесь и далее будем представлять дерево в виде некоторой его укладки на плоскости и понимать направления «вправо», «влево» как направления на этой плоскости относительно данной укладки.

Для произвольной внутренней вершины β дерева D^{B_i} обозначим через V_β множество чисел, соответствующих концевым вершинам, достижимым из β. Здесь и далее вершина α *достижима* из вершины β, если существует ориентированный путь ведущий из β в α.

Для произвольной внутренней вершины β дерева D^{B_i} обозначим

$$b_\beta = \min_{b \in V_{\beta'}} b,$$

где β' — конец правого ребра, исходящего из β.

Объявим все внутренние вершины дерева D^{B_i} вершинами переключения и для каждой внутренней вершины β левому ребру, из нее выходящему, припишем 1, а правому — 2, а самой вершине припишем переключатель $g_{b_\beta}(x)$.

Пусть $i \in \{1, \ldots, k\}$. Обозначим $j(i)$ такой номер, что $j(i) < i$, $l_{j(i)} > 0$ и не существует j' такого, что $l_{j'} > 0$, $j' < i$ и $j' > j(i)$, то есть $j(i)$ — индекс ближайшего снизу непустого множества B_j. Если такого множества нет, то $j(i) = 0$.

Теперь для каждого дерева D^{B_i} самую левую концевую вершину дерева отождествим с самым правым листом дерева $D^{B_{j(i)}}$, если $j(i) \neq 0$.

Для каждого такого i, что $l_i = 0$, вершину β_i отождествим с самым правым листом дерева $D^{B_{j(i)}}$, если $j(i) \neq 0$.

Полученный граф и будет ИГ U^1_m.

Обозначим через α_i концевую вершину, которой приписано число y^1_i $(i = 1, \ldots, k)$, а через α_0 самую левую концевую вершину в самом левом дереве D^{B_i}. Легко проверить, что функция проводимости между корнем и вершиной α_i $(i = 0, \ldots, k)$ имеет вид:

$$h_i(x^1, x^2) = \begin{cases} 1, & \text{если } y^1_i \leqslant x^1 < y^1_{i+1}, \\ 0, & \text{в противном случае,} \end{cases}$$

где $y^1_0 = 0$, $y^1_{k+1} = 1$.

Подсчитаем объем графа U^1_m.

Поскольку каждое из D^{B_i} есть дерево, в котором полустепень исхода любой вершины равна 2, то в D^{B_i} будет ровно $2 \cdot (l_i + 1) - 2 = 2\, l_i$ ребер. Отсюда следует, что

$$Q(U^1_m) = m + 2 \sum_{i=1}^{m} l_i = m + 2\, k. \tag{10}$$

Подсчитаем сложность графа U^1_m.

Обозначим

$$L(u) = \begin{cases} u, & \text{если } 0 \leqslant u \leqslant 3 \\ \log_2(u+1) + 1, & \text{если } u \geqslant 3 \end{cases},$$

$$X_i = \{(x^1, x^2) \in A_{a,b,c} : x^1 \in A^{i,m}_{a,b}\}.$$

Рассмотрим произвольный запрос $x = (x^1, x^2) \in A_{a,b,c}$. Пусть $x^1 \in A^{i,m}_{a,b}$, то есть $x \in X_i$. При $l_i = 0$ выполняется $T(U^1_m, x) = 1 = 1 + L(l_i)$.

Если $l_i > 0$, то запрос x выйдет к некоторой концевой вершине дерева D^{B_i} и при этом он пройдет не более $1+]\log_2(l_i+1)[$ переключательных вершин и, значит,

$$T(U_m^1, x) \leqslant 1+]\log_2(l_i+1)[\leqslant 1 + L(l_i).$$

По определению

$$\begin{aligned} T(U_m^1) = \mathbf{M}_x\, T(U_m^1, x) = \int_X T(U_m^1, x)\, \mathbf{P}(dx) = \\ = \sum_{i=1}^{m} \int_{X_i} T(U_m^1, x)\, \mathbf{P}(dx) \leqslant \sum_{i=1}^{m} (1 + L(l_i)) \cdot \mathbf{P}(X_i) = \\ = 1 + \sum_{i=1}^{m} L(l_i) \cdot \mathbf{P}(X_i) \leqslant 1 + \frac{(b-a)(1-c)}{m} \cdot \frac{C}{\mathbf{P}(A_{a,b,c})} \sum_{i=1}^{m} L(l_i). \end{aligned}$$

Так как $m \geqslant k$, то в силу вогнутости функции $L(u)$ и согласно лемме 14 из [3] максимум $\sum_{i=1}^{m} L(l_i)$ достигается, когда l_i максимально одинаковые, то есть k из них равны 1, а остальные $m - k$ — 0. Следовательно

$$T(U_m^1) \leqslant 1 + \frac{(b-a)(1-c)}{m} \cdot \frac{C}{\mathbf{P}(A_{a,b,c})} \cdot k \leqslant 2.$$

Теперь из каждой вершины α_i $(i = 1, \ldots, k)$ выпустим ребро, припишем этому ребру предикат $f_{y_i^2}$ и конец этого ребра обозначим γ_i. Вершину γ_i объявим листом и припишем ей запись y_i. Поскольку попадание в вершину α_i гарантирует, что $y_i^1 \leqslant x^1$, то равенство предиката $f_{y_i^2}(x^1, x^2)$ единице гарантирует, что $y_i^2 \leqslant x^2$, и тем самым запись y_i удовлетворяет запросу $x = (x^1, x^2)$.

И наконец, из каждого листа γ_i $(i = 2, \ldots, k)$ выпустим ребро в лист γ_{i-1} и припишем этому ребру предикат $f_{y_{i-1}^2}$. Данная последовательность ребер позволяет двигаться от листа к листу и, тем самым, от записи к записи в порядке убывания абсцисс, а значит, в порядке возрастания ординат, до тех пор пока записи удовлетворяют запросу x. Эту цепь, состоящую из $k-1$ ребра, будем называть *цепью перечисления ответа*.

Получившийся ИГ и будет искомым графом $U_{a,b,c}^V$.

Поскольку мы добавили $2k-1$ ребро, то с учетом (10) объем ИГ $U^V_{a,b,c}$ будет равен

$$Q(U^V_{a,b,c}) = k(4 + C(b-a)(1-c)/\mathbf{P}(A_{a,b,c})).$$

Поскольку из вершины α_i после проверки $f_{y_i^2}(x^1, x^2)$ мы сразу попадаем в лист и далее двигаемся только вдоль листьев, то с учетом того, что каждая запись обрабатывается пользователем по крайней мере 1 такт, получаем, что для фоновой сложности ИГ $U^V_{a,b,c}$ справедливо

$$T(U^V_{a,b,c}) \leqslant 3.$$

Тем самым лемма 2 доказана.

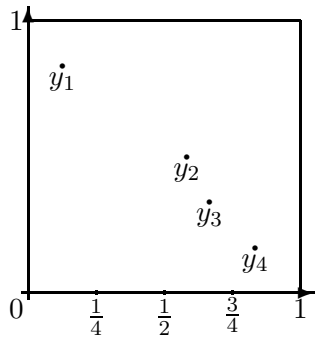

Рис. 2. Библиотека V.

На рисунках 2 и 3 изображен пример, когда область $A_{a,b,c}$ совпадает с X, то есть $a = 0$, $b = 1$, $c = 0$, на X задана равномерная вероятностная мера, то есть $C = 1$, библиотека состоит из 4-х элементов и показана на рисунке 2. Тогда ИГ $U^V_{0,1,0}$, построенный в лемме 2, имеет вид, изображенный на рисунке 3.

Лемма 3. *Если $I = \langle X, V, \rho\rangle$ — двумерная задача о доминировании, такая что V — слой мощности k, время обработки пользователем любой записи из ответа не меньше, чем 1, $\mathcal{F}$ — базовое множество, определяемое соотношениями (2)–(8), то существует ИГ $U^V \in \mathcal{U}(I, \mathcal{F})$ такой, что $Q(U^V) \leqslant 4k-1$, $T(U^V) \leqslant]\log_2(k+1)[+1$.*

Доказательство. $V = \{(y_1^1, y_1^2), (y_2^1, y_2^2), \ldots, (y_k^1, y_k^2)\}$ и $y_1^1 < y_2^1 < \cdots < y_k^1$. Пусть $B = \{y_1^1, y_2^1, \ldots, y_k^1\}$.

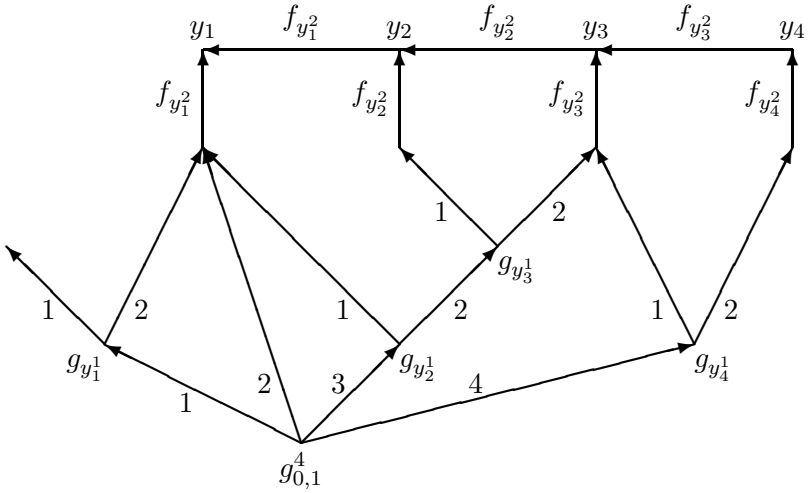

Рис. 3. ИГ $U_{0,1,0}^V$.

Возьмем вершину β_0 и объявим ее корнем графа. Выпустим из вершины β_0 бинарное сбалансированное дерево D^B с $k+1$ концевыми вершинами и высоты $]\log_2(k+1)[$, аналогичное дереву D^{B_i}, которое описано в лемме 2.

Обозначим как и ранее через α_i концевую вершину, которой приписано число y_i^1 $(i = 1, \ldots, k)$, а через α_0 самую левую концевую вершину дерева D^B. Из каждой вершины α_i $(i = 1, \ldots, k)$ выпустим ребро, припишем этому ребру предикат $f_{y_i^2}$ и конец этого ребра обозначим γ_i. Вершину γ_i объявим листом и припишем ей запись y_i. И наконец, из каждого листа γ_i $(i = 2, \ldots, k)$ выпустим ребро в лист γ_{i-1} и припишем этому ребру предикат $f_{y_{i-1}^2}$. Цепь, состоящую из $k-1$ ребра и ведущую от листа к листу, как и ранее будем называть *цепью перечисления ответа*.

Получившийся ИГ и будет искомым графом U^V.

ИГ U^V для библиотеки V мощности 3 приведен на рисунке 4.

Дерево D^B содержит $2k$ ребер, еще k ребер исходят из вершин α_i $(i = 1, \ldots, k)$ и $k-1$ ребер принадлежат цепи перечисления ответа. Тем самым объем ИГ U^V равен $Q(U^V) = 4k - 1$.

Для любого запроса мы вычисляем не более чем $]\log_2(k+1)[$ переключателей, проходя по дереву D^B, далее вычисляем один предикат

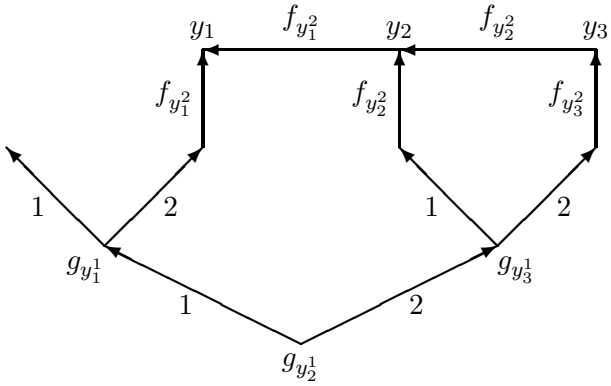

Рис. 4. ИГ U^V.

типа $f_{y_i^2}$, после чего начинаем перечислять ответ, и значит, фоновая сложность далее не увеличивается.

Тем самым лемма 3 доказана.

Следствие 1. *Если $I = \langle X, V, \rho\rangle$ — двумерная задача о доминировании, такая что V — слой мощности $k < 4$, время обработки пользователем любой записи из ответа не меньше, чем 1, $\mathcal{F}$ — базовое множество, определяемое соотношениями (2)–(8), то существует ИГ $U^V \in \mathcal{U}(I, \mathcal{F})$ такой, что $Q(U^V) \leqslant 4k - 1$, $T(U^V) \leqslant 3$.*

4. Решение двумерной задачи о доминировании

Пусть $V \subset Y$ — конечное множество точек. Точку $y \in V$ назовем *минимальной точкой множества V*, если не существует другой точки $y' \in V$, такой, что $y' \leqslant y$.

Множество минимальных точек множества V будем обозначать через V_1 и называть *первым слоем множества V*. Понятно, что V_1 есть слой, поскольку любая пара разных минимальных точек множества несравнима.

Теперь индуктивно определим последующие слои множества. Пусть уже определены все слои вплоть до $(n-1)$-го: $V_1, V_2, \ldots, V_{n-1}$. Пусть $V \setminus \bigcup_{i=1}^{n-1} V_i \neq \varnothing$. Тогда *$n$-м слоем множества V* назовем множество V_n минимальных точек множества $V \setminus \bigcup_{i=1}^{n-1} V_i$.

Если $V_n \neq \varnothing$ и $V = \bigcup_{i=1}^{n} V_i$, то множество V назовем *n-слойным*.

Лемма 4. *Если $I = \langle X, V, \rho\rangle$ — двумерная задача о доминировании, такая что $|V| = k < 4$, время обработки пользователем любой записи из ответа не меньше, чем 3, $\mathcal{F}$ — базовое множество, определяемое соотношениями (2)–(8), то существует ИГ $U^V \in \mathcal{U}(I, \mathcal{F})$ такой, что $Q(U^V) \leqslant 4k - 1$, $T(U^V) \leqslant 3$.*

Доказательство. Справедливость случая, когда $k < 4$ и V — однослойная библиотека, следует из следствия 1.

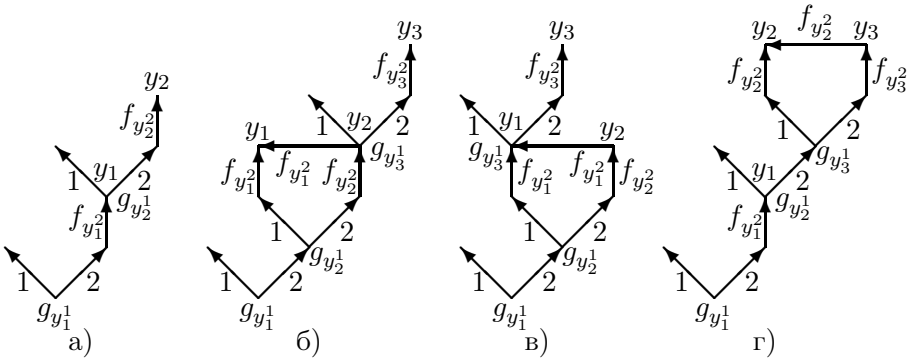

Рис. 5. ИГ U^V.

На рисунке 5 показаны остальные случаи. Так рис. 5 а) описывает ИГ, решающий задачу для двухслойной библиотеки мощности 2. ИГ для случая трехслойной библиотеки мощности 3 строится аналогично, при этом ИГ будет иметь высоту 6, где на последних ярусах будет проверяться последняя запись.

На рис. 5 б) приведен ИГ для двухслойной библиотеки из трех элементов, где на первом слое 2 элемента y_1 и y_2 ($y_1^1 < y_2^1$), а для элемента y_3 со второго слоя выполнено $y_2^1 < y_3^1$.

На рис. 5 в) изображен ИГ для двухслойной библиотеки из трех элементов, где на первом слое 2 элемента y_1 и y_2, а для третьего элемента y_3 выполнено $y_1^1 < y_3^1 < y_2^1$.

И наконец, на рис. 5 г) приведен ИГ для двухслойной библиотеки из трех элементов, где на первом слое 1 элемент y_1, а втором слое элементы y_2 и y_3 ($y_2^1 < y_3^1$).

Условие, что число ребер не больше чем $4k - 1$ проверяется непосредственно. То, что фоновая сложность не больше 3 следует из того, что количество вычислений до достижения первого листа и между листьями не превышает 3.

Тем самым лемма 4 доказана.

Пусть V — n-слойная библиотека, $V_i = \{y_{i,1}, y_{i,2}, \ldots, y_{i,l_i}\}$ — i-й слой библиотеки V, причем V_i упорядочен в порядке возрастания абсцисс, то есть $y^1_{i,1} < y^1_{i,2} < \cdots < y^1_{i,l_i}$ $(i = 1, 2, \ldots, n)$. Обозначим

$$V_{i,j} = \left\{ y_{i+1,t} \in V_{i+1} : y^1_{i,j} \leqslant y^1_{i+1,t} < y^1_{i,j+1} \right\}, j = 1, 2, \ldots, l_i, i < n, \quad (11)$$

где $y^1_{i,l_i+1} = 1$, то есть $V_{i,j}$ — это множество точек $(i+1)$-го слоя, абсциссы которых находятся между абсциссами j-й и $(j+1)$-й точек i-го слоя библиотеки V.

Пусть функция плотности вероятности $p(x^1, x^2)$, определяющая вероятностную меру $\mathbf{P}$ на X, ограничена константой C, V — n-слойная библиотека, где $n > 1$. Обозначим

$$\mathcal{C}_V = \{C\} \cup \left\{ \frac{C(y^1_{i,j+1} - y^1_{i,j})(1 - y^2_{i,j})}{\mathbf{P}(A_{y^1_{i,j}, y^1_{i,j+1}, y^2_{i,j}})} : i \in \{1, \ldots, n-1\}, \right.$$
$$\left. j \in \{1, \ldots, l_i\}, \mathbf{P}(A_{y^1_{i,j}, y^1_{i,j+1}, y^2_{i,j}}) > 0, |V_{i,j}| > 3 \right\}, \quad (12)$$

$$C_V = \max_{C' \in \mathcal{C}_V} C'. \quad (13)$$

Лемма 5. *Если $I = \langle X, V, \rho \rangle$ — двумерная задача о доминировании, время обработки пользователем любой записи из ответа не меньше, чем 5, $\mathcal{F}$ — базовое множество, определяемое соотношениями (2)–(8), C_V — величина, определяемая соотношениями (12)–(13), то существует ИГ $U^V \in \mathcal{U}(I, \mathcal{F})$ такой, что $Q(U^V) < |V|\,(5.25 + C_V)$ и $T(U^V) \leqslant 3$.*

Доказательство. Сначала приведем неформальное описание алгоритма, обеспечивающего требуемую сложность. Разбиваем библиотеку на слои. Далее начинаем поиск на упорядоченном по возрастанию абсцисс первом слое. Для этого бинарным поиском находим запись $y_{1,s}$ с максимальной абсциссой, не превышающей абсциссу запроса. Проверяем попадает ли эта запись в запрос. Если нет, то и никакая другая запись в запрос попасть не может, и значит ответ пуст. Если попадает, то идем вдоль первого слоя, собирая записи, попадающие в ответ. Далее переходим к поиску на втором слое. Но поскольку из

того, что запись $y_{1,s}$ попадает в ответ, и того, что она самая большая по абсциссе среди записей первого слоя, попадающих в ответ, следует, что запрос принадлежит множеству $A_{y^1_{1,s},y^1_{1,s+1},y^2_{1,s}}$. Поэтому поиск максимальной по абсциссе записи второго слоя, не превышающей по абсциссе запрос, можно вести не на всем втором слое, а на множестве $V_{1,s}$. Если же множество $V_{1,s}$ пусто, то искомой записью будет максимальная по абсциссе запись в множестве $V_{1,i}$, где i — максимальный номер, такой, что $i < s$ и $V_{1,i} \neq \varnothing$. После сбора в ответ точек второго слоя мы аналогично переходим к третьему и так далее до тех пор пока на каком-то слое не окажется записей, попадающих в ответ, а это будет означать, что и на слоях с большими номерами записей, удовлетворяющих запросу нет, и поиск можно прекратить.

Пусть V — n-слойная библиотека, удовлетворяющая условиям леммы. Если $|V| < 4$, то справедливость утверждения леммы 5 следует из леммы 4, если $n = 1$, то — из леммы 3.

Рассмотрим случай, когда $|V| > 3$ и $n > 1$.

Пусть $V_1 = \{y_{1,1}, y_{1,2}, \ldots, y_{1,l_1}\}$ — первый слой библиотеки V, причем $y^1_{1,1} < y^1_{1,2} < \cdots < y^1_{1,l_1}$.

Пусть $B_1 = \{y^1_{1,1}, y^1_{1,2}, \ldots, y^1_{1,l_1}\}$ — упорядоченное по возрастанию множество абсцисс точек из V_1.

Если $l_1 > 3$, то построим ИГ U^1_m для множества B_1 точно также, как это делалось в доказательстве леммы 2, где $m =]l_1\, C[\leqslant l_1\, C_V + 1$. Если $l_1 \leqslant 3$, то построим ИГ D^{B_1}, как описано в доказательстве леммы 2.

Как было показано в лемме 2, $Q(U^1_m) = m + 2\,l_1$, $Q(D^{B_1}) = 2\,l_1$, $T(U^1_m) \leqslant 2$ и $T(D^{B_1}) \leqslant 2$ при $l_1 \leqslant 3$ согласно лемме 3.

Обозначим через α^1_i концевую вершину, которой приписано число $y^1_{1,i}$ $(i = 1, \ldots, l_1)$, а через α^1_0 самую левую концевую вершину.

Теперь из каждой вершины α^1_i $(i = 1, \ldots, l_1)$ выпустим по два ребра и каждому из этих двух ребер припишем предикат $f_{y^2_{1,i}}$, а концы этих ребер обозначим γ^1_i и δ^1_i. Вершину γ^1_i объявим листом и припишем ей запись $y_{1,i}$. Попадание запроса $x = (x^1, x^2)$ в вершину α^1_i гарантирует, что $y^1_{1,i} \leqslant x^1$, а предикат $f_{y^2_{1,i}}(x^1, x^2)$ завершает проверку удовлетворяет ли запись $y_{1,i}$ запросу x.

Далее из каждого листа γ^1_i $(i = 1, \ldots, l_1)$ выпустим ребро в лист γ^1_{i-1} и припишем этому ребру предикат $f_{y^2_{1,i-1}}$. Данная последовательность ребер называется цепью перечисления ответа. Она позволяет

двигаться от листа к листу и, тем самым, от записи к записи в порядке убывания абсцисс, а значит, в порядке возрастания ординат, до тех пор пока записи удовлетворяют запросу x.

Вершину δ_i^1 мы будем использовать для продолжения поиска на следующем слое. Легко проверить, что в каждую вершину δ_i^1 $(i = 1, \ldots, l_1)$ могут попасть только запросы из множества $A_{y^1_{1,i}, y^1_{1,i+1}, y^2_{1,i}}$.

Мы организовали поиск на первом слое.

Теперь последовательно будем организовывать поиск на остальных слоях, начиная со второго. Обозначим через q номер очередного слоя для которого мы достраиваем ИГ, чтобы организовать на нем поиск $(q = 2, 3, \ldots, n)$. При этом предполагаем, что ИГ для поиска на предыдущих слоях уже построен и в нем построены вершины δ_i^{q-1} $(i = 1, 2, \ldots, l_{q-1})$, в которые проходят запросы их множества $A_{y^1_{q-1,i}, y^1_{q-1,i+1}, y^2_{q-1,i}}$. Поэтому при условии, что запрос попал в вершину δ_i^{q-1}, поиск подходящих записей из q-го слоя не обязательно вести на всем q-м слое, а можно вести в множестве $V_{q-1,i}$. Напомним, что $V_{q-1,i}$ — это множество точек q-го слоя, абсциссы которых находятся между $y^1_{q-1,i}$ и $y^1_{q-1,i+1}$.

Поэтому для каждого такого $i \in \{1, \ldots, l_{q-1}\}$, что $|V_{q-1,i}| > 0$, мы проделаем следующие действия.

Пусть $V_{q-1,i} = \{y_{q,t_i}, y_{q,t_i+1}, \ldots, y_{q,t_{i+1}-1}\}$, причем $y^1_{q,t_i} < \cdots < y^1_{q,t_{i+1}-1}$. Пусть $B_{q,i} = \{y^1_{q,t_i}, y^1_{q,t_i+1}, \ldots, y^1_{q,t_{i+1}-1}\}$ — упорядоченное по возрастанию множество абсцисс точек из $V_{q-1,i}$.

Если $t_{i+1} - t_i \leqslant 3$ или $\mathbf{P}(A_{y^1_{q-1,i}, y^1_{q-1,i+1}, y^2_{q-1,i}}) = 0$, то из вершины δ_i^{q-1} построим ИГ $D^{B_{q,i}}$, как описано в доказательстве леммы 2. Иначе из вершины δ_i^{q-1} построим ИГ $U^1_{m_i}$ для множества $B_{q,i}$ точно также, как это делалось в доказательстве леммы 2, где $m_i =](t_{i+1} - t_i)\, C(y^1_{q-1,i+1} - y^1_{q-1,i})(1 - y^2_{q-1,i})/\mathbf{P}(A_{y^1_{q-1,i}, y^1_{q-1,i+1}, y^2_{q-1,i}})[\leqslant (t_{i+1} - t_i)\, C_V + 1$.

Как было показано в лемме 2, $Q(U^1_{m_i}) = m_i + 2\,(t_{i+1} - t_i)$, $Q(D^{B_{q,i}}) = 2\,(t_{i+1} - t_i)$, $T(U^1_{m_i}) \leqslant 2$ и $T(D^{B_{q,i}}) \leqslant 2$ при $(t_{i+1} - t_i) \leqslant 3$. Сложность $T(D^{B_{q,i}})$ для случая $\mathbf{P}(A_{y^1_{q-1,i}, y^1_{q-1,i+1}, y^2_{q-1,i}}) = 0$ можно не учитывать, поскольку последнее означает, что вероятность попадания в вершину δ_i^{q-1} равна нулю.

Обозначим через α_i^q концевую вершину, которой приписано число $y^1_{q,i}$ $(i = t_i, \ldots, t_{i+1} - 1)$, а через $\alpha'^q_{t_i-1}$ самую левую концевую вершину.

Построенный фрагмент информационного графа обозначим $U^q_{V_{q-1,i}}$.

Для каждого $i \in \{1, 2, \ldots, l_{q-1}\}$ через $j(i)$ обозначим такой максимальный номер, что $j(i) < i$ и $|V_{q-1,j(i)}| > 0$. Если такого номера нет, то $j(i) = 0$.

Теперь для каждого $i \in \{1, 2, \ldots, l_{q-1}\}$ если $|V_{q-1,i}| > 0$ и $t_i - 1 > 0$, то отождествим вершины $\alpha'^q_{t_i-1}$ и $\alpha^q_{t_i-1}$ (легко понять, что $\alpha^q_{t_i-1}$ самая правая вершина этого типа в ИГ $U^q_{V_{q-1,j(i)}}$, то есть $t_i - 1 = t_{j(i)+1} - 1$), а если $|V_{q-1,i}| = 0$, то отождествим вершины δ^{q-1}_i и $\alpha^q_{t_{j(i)+1}-1}$. В результате этих отождествлений мы добиваемся того, что в вершины α^q_i $(i = 1, \ldots, l_q)$ проходят все запросы $x = (x^1, x^2)$, такие, что $y^1_{q,i} \leqslant x^1 < y^1_{q,i+1}$.

Теперь из каждой вершины α^q_i $(i = 1, 2, \ldots, l_q)$ выпустим ребро, припишем ему предикат $f_{y^2_{q,i}}$, и конец этого ребра обозначим γ^q_i. Вершину γ^q_i объявим листом и припишем ей запись $y_{q,i}$. Попадание запроса $x = (x^1, x^2)$ в вершину α^q_i гарантирует, что $y^1_{q,i} \leqslant x^1$, а предикат $f_{y^2_{q,i}}(x^1, x^2)$ завершает проверку удовлетворяет ли запись $y_{q,i}$ запросу x. Если исходная библиотека V более чем q-слойная, то есть $n > q$, то из вершины α^q_i выпустим еще одно ребро, которому припишем предикат $f_{y^2_{q,i}}$, и конец этого ребра обозначим δ^q_i. Эту вершину мы будем использовать для продолжения поиска на следующем слое.

Далее из каждого листа γ^q_i $(i = 2, 3, \ldots, l_q)$ выпустим ребро в лист γ^q_{i-1} и припишем этому ребру предикат $f_{y^2_{q,i-1}}$. Данная последовательность ребер есть цепь перечисления ответа. Она позволяет двигаться по q-му слою от записи к записи в порядке убывания абсцисс до тех пор пока записи удовлетворяют запросу x.

После того как мы сделаем описываемые построения для последнего слоя построение требуемого информационного графа U^V завершается.

Легко проверить, что построенный граф в точности соответствует описанному ранее алгоритму поиска и решает нашу задачу $I = \langle X, V, \rho \rangle$.

Подсчитаем фоновую сложность ИГ U^V.

Как мы уже отмечали, согласно леммам 2 и 3 среднее время поиска первой записи на первом слое не превышает 3. Теперь подсчитаем сколько времени может быть потрачено на поиск очередной записи.

Одно вычисление предиката вида $f_{y^2_{q,j-1}}$, чтобы убедиться, что следующая запись на цепи перечисления ответа нам не подходит. Второе вычисление предиката вида $f_{y^2_{q,i}}$, приписанного ребру, ведущему в вершину δ^q_i. Далее в среднем 2 вычисления мы сделаем при прохождении по графу типа U^1_m или дереву типа D^B. И на последок мы вычисляем предикат вида $f_{y^2_{q+1,i}}$, приписанный ребру, ведущему из вершины типа α^{q+1}_i в лист γ^{q+1}_i. Тем самым, если пользователь каждую запись обрабатывает по крайней мере 5 тактов, то все время ожидания выльется в ожидание первой записи. Таким образом мы доказали, что $T(U^V) \leqslant 3$.

Подсчитаем объем ИГ U^V.

Пусть V' некий фрагмент некоторого слоя для которого мы строим ИГ типа U^1_m или D^B, и $|V'| = l$ (то есть V' есть либо V_1, либо $V_{i,j}$ для некоторых $i \in \{1, 2, \ldots, n-1\}$ и $j \in \{1, 2, \ldots, l_i\}$). Поскольку $C_V \geqslant 1$, то $Q(D^B) = 2\,l < Q(U^1_m) \leqslant l\,(2 + C_V) + 1$. Следовательно худшим случаем будет случай, когда для каждого фрагмента будет строиться ИГ типа U^1_m, но последние строятся только если число элементов в фрагменте не меньше 4. Следовательно максимальное число ИГ типа U^1_m не превышает $|V|/4$, а суммарный их объем не превышает $|V|\,(2 + C_V) + |V|/4$. Из каждой вершины типа α^q_i выходит не более двух ребер, а число таких вершин равно $|V|$, поэтому суммарное число таких ребер не превышает $2\,|V|$. Осталось учесть ребра, принадлежащие цепям перечисления ответа, которых не более, чем $|V| - n$. Суммируя все, получаем $Q(U^V) < |V|\,(5.25 + C_V)$.

Тем самым лемма 5 доказана.

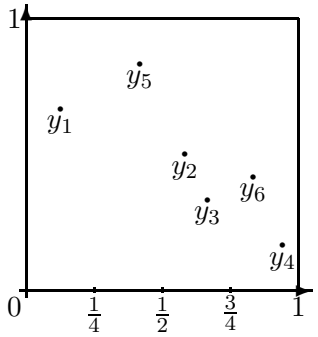

Рис. 6. Библиотека V.

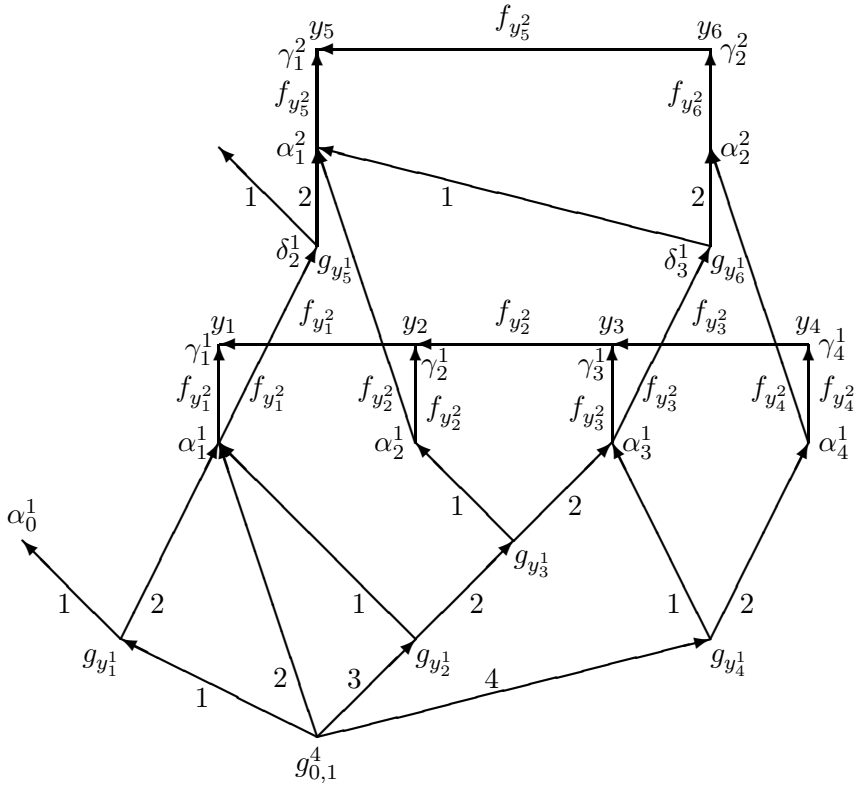

Рис. 7. ИГ U^V.

На рисунке 6 приведен пример двухслойной библиотеки, а на рисунке 7 ИГ U^V для этой библиотеки.

Остается заметить, что утверждение теоремы 1 является простым следствием леммы 5.

Список литературы

[1] Гасанов Э. Э., Мхитарова Т. В. Об одной математической модели фоновых алгоритмов поиска и быстрый фоновый алгоритм двумерной задачи о доминировании // Фундаментальная и прикладная математика. — 1997. — Т. 3, вып. 1. — С. 759–773.

[2] Гасанов Э. Э. Константный в среднем фоновый алгоритм решения двумерной задачи о доминировании // Современные проблемы ма-

тематики, механики и их приложений. Материалы международной конференции, посвященной 70-летию ректора МГУ академика В. А. Садовничего. — М.: Университетская книга, 2009. — С. 356.

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

[4] Гасанов Э. Э. Некоторые задачи поиска, допускающие мгновенное в среднем решение // Фундаментальная и прикладная математика. — 1995. — Т. 1, вып. 1. — С. 123–146.