ФГБОУ ВО «Московский государственный
университет имени М.В. Ломоносова»

На правах рукописи

Плетнев Александр Андреевич

Моделирование динамических баз данных

01.01.09 — дискретная математика и математическая кибернетика

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата физико-математических наук

Москва — 2016

Работа выполнена на кафедре Математической теории интеллектуальных систем Механико-математического факультета ФГБОУ ВО «Московский государственный университет имени М.В. Ломоносова».

Научный руководитель:	**Гасанов Эльяр Эльдарович**, доктор физико-математических наук, профессор
Официальные оппоненты:	**Орлов Валентин Александрович** доктор физико-математических наук, профессор кафедры "Информационная безопасность"(Московского государственного технического университета имени Н.Э. Баумана)
	Лялин Илья Викторович кандидат физико-математических наук, Элайн Технолоджи, старший разработчик
Ведущая организация:	**ФГБОУ ВО "Московский технологический университет"**

Защита диссертации состоится **21 октября 2016 г.** в 16 ч. 45 м. на заседании диссертационного совета Д.501.001.84, на базе ФГБОУ ВО МГУ имени М.В. Ломоносова, по адресу: Российская Федерация, 119991, Москва, ГСП-1, Ленинские горы, д.1, ФГБОУ ВО МГУ имени М.В. Ломоносова, Механико-математический факультет, аудитория 14-08.
С диссертацией можно ознакомиться в Фундаментальной библиотеке ФГБОУ ВО МГУ имени М.В. Ломоносова, по адресу: Москва, Ломоносовский проспект, д. 27, сектор А., http://mech.math.msu.su/∼snark/index.cgi, http://istina.msu.ru/dissertations/21275202.

Автореферат разослан 21 сентября 2016 г.

Ученый секретарь
диссертационного совета
Д.501.001.84, на базе
ФГБОУ ВО МГУ имени М.В. Ломоносова,
доктор физико-математических наук,
профессор **Шафаревич Андрей Игоревич**

Общая характеристика работы

Актуальность темы

Диссертация является исследованием в области дискретной математики и математической кибернетики и посвящена изучению динамических баз данных. База данных (БД) — формализованное представление информации, удобное для хранения и поиска данных в нем. Понятие БД возникло в 60-е годы 20 века и связано с развитием вычислительной техники и информатики. Тематика теории БД связана с поиском удобного представления, компактного хранения, быстрого поиска и других свойств данных. Исследование в области баз данных широко известны из работ: Кодд (E.F.Codd)[1] (реляционная модель данных), В.Н.Решетников[2] (алгебраическая модель информационного поиска), Думи (A.Dumey), [3], А.П.Ершов [4](методы хеширования), Бентли (J.L.Bentley)[5], Кнут (D.E.Knuth)[6], Ульман (J.D.Ullman)[7], Хаджеруп (Hagerup T.) и Каммер (Kammer F.) [8], Ружиц (Ruzic M.) [9], Шеймос (M.I.Shamos) и другие [10] (сложность алгоритмов обработки данных), Э.Э.Гасанов [11] (информационно-графовая модель данных, сложность алгоритмов поиска) и многих других.

В диссертации изучено направление БД, связанное с решением одной из задач информационного поиска (ЗИП), а именно, динамическая задача поиска идентичных объектов (ДЗПИО). ЗИП – один из наиболее часто встречающихся на практике видов задач. Самым распространенным примером задачи поиска, встречающейся в любой базе данных, является задача поиска по идентификатору. Суть задачи состоит в том, что любой объект в базе данных имеет свой уникальный идентификатор. Идентификатором может быть уникальное имя или уникальное значение, например, автомобильный номер. Задача состоит в том, чтобы по заданному в запросе

[1]Codd E. F., *A Relation Model of Data for Large Shared Data Banks*, Comm. ACM 13, № 6, ACM, New York, London, Amsterdam, June 1970, 377–387.

[2]Решетников В. Н., *Алгебраическая теория информационного поиска*, Программирование. — 1979. — № 3. — С. 68–74.

[3]Dumey A., *Indexing for Rapid Random Access Memory Systems*, Computers and Automation. (1956) 4, № 12, 6–9.

[4]Ершов А. П., *О программировании арифметических операторов*, ДАН СССР. — 1958. — Т. 118. — С. 427–430.

[5]Bentley J. L., Friedman J. H., *Data structures for range searching*, Comput. Surveys (1979), 11 397–409.

[6]Кнут Д., *Искусство программирования для ЭВМ*,Т. 3. Сортировка и поиск. — М.: Мир, 1978.

[7]Ульман Дж., *Основы систем баз данных*,пер. с англ. — М.: Мир, 1983.

[8]Hagerup T., Kammer F. *Succinct Choice Dictionaries* // CoRR abs/1604.06058 (2016).

[9]Ruzic M. *Uniform deterministic dictionaries* // ACM Trans. Algorithms 4 (2008).

[10]Препарата Ф., Шеймос М., *Вычислительная геометрия: Введение*, М.: Мир, 1989.

[11]Гасанов Э.Э., Кудрявцев В.Б., *Теория хранения и поиска информации*, М.: Физматлит, 2002.

идентификатору найти в базе данных объект с этим идентификатором (если такой объект в базе имеется). Если множество идентификаторов может изменяться со временем, то эту задачу называют ДЗПИО.

Эффективное решение ДЗПИО — это актуальная проблема и на сегодняшний день. Современные ЭВМ позволяют использовать параллельные процессы для решения подобного рода задач. Исследования в области параллельного вычисления широко известны из работ: Гуибас (Leo J. Guibas) [12], Аржоманди (E.A. Arjomandi) [13], Чин (F.Y. Chin) [14], Беркмэн (O. Berkman) [15] и многих других.

Основная сложность параллельного вычисления заключается в считывании и записывании данных в одну и ту же область памяти. Различают несколько типов систем для параллельной обработки данных. Они различаются способами обращения к общей памяти:

- одновременное чтение, одновременная запись (ОЧОЗ). В этой системе разрешается одновременное чтение и одновременная запись разными процессами в общую память;
- одновременное чтение, эсклюзивная запись (ОЧЭЗ). Эта система разрешает одновременное чтение из общей памяти, но при этом одновременная запись разными процессами в общую память запрещена;
- эсклюзивное чтение, эсклюзивная запись (ЭЧЭЗ). В данной системе запрещается писать и читать из общей памяти разными процессам.

Подробно эти системы описаны в работах Гафни (E. Gafni) [16], Снир (Snir M.)[17]. Применительно к задаче поиска идентичного объекта, было предложено множество различных алгоритмов, использующие параллельные вычисления. Они основываются на таких известных структурах данных, как сбалансированное бинарное дерево, красно-черное дерево, 2—3 дерево. В основном в этой области, цель всех исследований заключается в добавлении несколько записей к структуре данных за наименьшее время. Более

[12]Leo J. Guibas, Robert Sedgewick, *A Dichromatic Framework for Balanced Trees*, In Proceedings of the 19th Annual Symposium on Foundations of Computer Science, pages 8-21. IEEE Computer Society, 1978.

[13] E.A. Arjomandi, *A Study of Parallelism in Graph Theory. PhD thesis, PhD thesis*, Dept. Computer Science, University of Toronto, 1975.

[14] F.Y. Chin, J. Lam, and I. Chen, *Efficient parallel algorithms for some graph problems*, Comm. ACM, 25(9):659–665, 1982.

[15]O. Berkman , B. Schieber, and U. Vishkin, *Optimal doubly logarithmic parallel algorithms based on finding all nearest smaller values*, Journal of Algorithms, 14(3):344–370, 1993.

[16]E. Gafni, J. Naor, P. Ragde, *On Separating the EREW and CREW PRAM Models*, Theoretical Computer Science 68 (1989) 343-346.

[17]M. Snir, *On parallel searching*, SIAM J. Computing 14 (1985) 688-708.

детально о полученных результатах можно прочитать в работах: Комер (Comer D.) [18], Паркс (Parks H.)[19] и Ванг (Wang B-F) [20].

Цель работы

Основной целью работы является построение математической модели динамических баз данных, кроме того, решаются следующие задачи.

- Доказать применимость предлагаемой модели для решения задач, возникающих в динамических базах данных.
- Разработать бесконечно распараллеливаемые структуры данных для решения ДЗПИО для любого потока запросов.
- Предъявить бесконечно распараллеливаемую структуру данных для решения ДЗПИО с логарифмической сложностью, для любого потока запросов.
- Получить оценки параметров модели, при которых возможна обработка произвольных потоков запросов.

Методы исследования

В работе используются методы теории автоматов, теории сложности управляющих систем и теории графов.

Научная новизна

В работе представлена модель динамических баз данных. В рамках этой модели можно описывать алгоритмы, использующие параллельные процессы и оценивать их сложность.

В диссертации предложена бесконечно распараллеливаемая ОЧЭЗ структура данных, решающая динамическую задачу поиска идентичного объекта для любого потока запросов с логарифмической сложностью.

Получены минимально возможные значения параметров модели, при которых база данных может обрабатывать произвольный поток запросов.

[18] Comer D., *The Ubiquitous B-tree*, ACM Computing Surveys, 11(2):121-137, 1979.

[19] H. Park, K. Park, *Parallel algorithms for red-black trees*, Theoretical Computer Science 262 (2001) 415-435.

[20] B-F Wang, G-H Chen, *Cost-optimal parallel algorithms for constructing 2-3 trees*, Journal Of Parallel And Distributed Computing 11 (1991) 257-261.

Теоретическая и практическая значимость

Работа имеет теоретический характер.

Введенная модель позволяет сравнивать между собой различные алгоритмы для решения ДЗПИО, в том числе известные классические алгоритмы и алгоритмы, использующие параллельные процессы. Так же полученные в работе результаты являются новыми и представляют интерес для специалистов в области дискретной математики и теории алгоритмов.

Кроме этого, представленный в работе алгоритм решения ДЗПИО для любого потока запросов с логарифмической сложностью может быть использован в прикладных задачах.

Апробация работы

Результаты диссертации докладывались на следующих научных семинарах и всероссийских и международных конференциях.

(1) Семинар "Теория автоматов"под руководством академика, профессора, д.ф-м.н. В.Б. Кудрявцева (2014-2015 гг., неоднократно);

(2) Семинар "Вопросы сложности алгоритмов поиска"под руководством проф., д.ф-м.н. Э.Э.Гасанова (2009-2015 гг., неоднократно);

(3) Международная конференция студентов, аспирантов и молодых ученых "Ломоносов"(7-11 апреля 2014, 13-17 апреля 2015, Москва, МГУ);

(4) X Международный семинар "Дискретная математика и ее приложения"(1-6 февраля 2010, Москва, МГУ);

(5) X Международная конференция "Интеллектуальные системы и компьютерные науки"(21-26 ноября 2011, Москва, МГУ);

(6) XI Международный семинар "Дискретная математика и ее приложения посвященный 80-летию со дня рождения академика О.Б. Лупанова (18-23 июня 2012, Москва, МГУ);

(7) Республиканская научная конференции с участием зарубежных ученых "Современные методы математической физики и их приложения"(15–17 апреля 2015, Ташкент, Национальный университет Узбекистана им. М. Улугбека).

Публикации

Результаты автора по теме диссертации опубликованы в шести работах из списка ВАК РФ, список которых приводится в конце автореферата [1 - 6].

Структура и объем работы

Диссертация состоит из введения, трех глав и заключения. Объем диссертации 131 страница. Список литературы содержит 34 наименования.

Краткое содержание работы

В введении описана предметная область и история вопроса, даны основные используемые определения, сформулированы результаты диссертации.

В первой главе вводится модель динамических баз данных для решения задач информационного поиска. Полученная модель применяется для решения ДЗПИО, а как следствие доказывается, что конечный объект (автомат) может поддерживать бесконечную структуру (информационный граф). В конце первой главы модель обобщается на случай потоковых запросов к базе данных. Структуры данных, полученные в рамках этой модели, называем потоковый динамический информационный граф (ПДИГ).

Во второй главе доказываются верхние оценки на ПДИГ. Первая оценка заключается в том, что существует ПДИГ, который решает ДЗПИО для любого потока запросов. Кроме этого, в этой главе доказывается, что существует ПДИГ, решающий ДЗПИО для любого потока запросов с логарифмической сложностью. Так же в конце второй главы приводится пример минимального по степени ветвления ПДИГ с радиусом видимости один, решающий ДЗПИО для любого потока запросов.

В третьей главе доказываются нижние оценки на ПДИГ. Автор показал, что не существует ПДИГ со степенью ветвления один, решающего ДЗПИО для любого потока запросов. Так же доказывается, что не существует конечного, селекторного ПДИГ со степенью ветвления два и радиусом видимости один, который решает эту же задачу.

Пусть X — *множество запросов*, Y — *множество записей* (объектов поиска), ρ – *бинарное отношение на* $X \times Y$, называемое отношением поиска. Тройку $I = \langle X, V, \rho \rangle$, где V — некоторое подмножество множества Y, в дальнейшем называемой библиотекой, будем называть *задачей информационного поиска* (ЗИП). Будем считать, что ЗИП $I = \langle X, V, \rho \rangle$ содержательно

состоит в перечислении для произвольного взятого запроса $x \in X$ всех тех и только тех записей $y \in V$, что $x\rho y$.

В формальном определении понятия ИГ используются 4 множества: множество запросов X, множество записей Y, множество F одноместных предикатов (заданных на множестве X), множество G одноместных переключателей (заданных на множестве X). *Предикат*— это функция, множество значений которой есть $\{0, 1\}$. *Переключатель* — это функция, множество значений которых является начальным отрезком натурального ряда.

Понятие ИГ 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 могут быть и полюса) называются *переключательными* и им приписываются переключатели из G. Ребра, исходящие из каждой из переключательной 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 назыаем ИГ над базовым множеством $\mathcal{F} = \langle F, G\rangle$, где $F = \{f_j, j \in J\}$, $G = \{g_k, k \in K\}$, J и K — множества индексов.

Введем множество функций преобразования индексов $\mathcal{C}$,

$$\mathcal{C} = \{c_m : J^{d_{1,m}} \times K^{d_{2,m}} \times Y^{d_{3,m}} \to J^{a_m} \times K^{b_m} \times Y^{R_m},$$
$$m \in M, M \subseteq \mathbb{N};\ d_{1,m},\ d_{2,m},\ d_{3,m} \in \mathbb{N};$$
$$a_m, b_m, R_m \in \{0, 1\} \text{ и } a_m + b_m + R_m = 1\}. \quad (1)$$

Введем три счетных множества переменных $\mathcal{P}_J, \mathcal{P}_K, \mathcal{P}_L$:

- $\mathcal{P}_J = \{p_J^j\}$, $j \in \mathbb{N}$, p_J^j принимают значения из J;
- $\mathcal{P}_K = \{p_K^k\}$, $k \in \mathbb{N}$, p_K^k принимают значения из K;
- $\mathcal{P}_L = \{p_L^l\}$, $l \in \mathbb{N}$, p_L^l принимают значения из Y.

Рассмотрим произвольный ИГ U. Он может содержать предикатные, переключательные и листовые вершины. Обозначим $F(U)$ — множество индексов предикатов, $G(U)$ — множество индексов переключателей и $Y(U)$ — множество записей, входящих в ИГ U. Заменим нагрузку всех входящих в U вершин и ребер по следующему правилу.

- Каждый предикат с индексом $j \in F(U)$ заменим на некоторую переменную из $\mathcal{P}_J$. Причем эта замена задается инъективной функцией $\Omega_F : F(U) \to \mathcal{P}_J$. Это означает, что предикаты с различными индексами $j \in F(U)$ заменим на различные переменные из $\mathcal{P}_J$, а с одинаковыми индексами $j \in F(U)$ заменим на одинаковые переменные из $\mathcal{P}_J$.

- Каждый переключатель с индексом $k \in G(U)$ заменим на переменную из $\mathcal{P}_K$. Причем эта замена задается инъективной функцией $\Omega_G : G(U) \to \mathcal{P}_K$.

- Каждую запись (приписанную листовой вершине) $y \in Y$ заменим на переменную из $\mathcal{P}_L$. Причем эта замена задается инъективной функцией $\Omega_Y : Y(U) \to \mathcal{P}_L$.

Рассмотрим множество $\mathcal{P}(U) = \Omega_F(F(U)) \cup \Omega_G(G(U)) \cup \Omega_Y(Y(U))$, то есть $\mathcal{P}(U)$ — множество переменных, которые получились в результате замены нагрузки всех вершин и ребер из U.

Рассмотрим функцию

$$\Omega : F(U) \cup G(U) \cup Y(U) \to \mathcal{P}(U),$$

где $\Omega = \Omega_F$ на множестве $F(U)$ ($\Omega \mid_{F(U)} = \Omega_F$), $\Omega \mid_{G(U)} = \Omega_G$ и $\Omega \mid_{Y(U)} = \Omega_Y$.

Очевидно из определения, что Ω — биективная функция, поэтому существует функция Ω^{-1}. Обозначим $I = \Omega^{-1}$ и будем называть *интерпретацией*, возникшей в результате замены индексов на переменные.

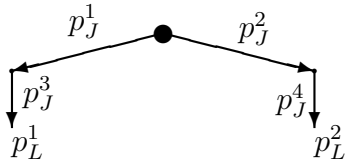

Рис. 1: Пример простого шаблона

После этого сопоставления (замены нагрузки вершин и ребер на переменные в ИГ U) получим нагруженный граф. Выделим в нем множество вершин $V_{\mathcal{T}}$, которые назовем вершинами прикрепления, и занумеруем их числами от 1 до $|V_{\mathcal{T}}|$. Полученный граф назовем *простым шаблоном* $\mathcal{T}$. При этом будем писать $\mathcal{T} = \mathcal{T}(U, I, V_{\mathcal{T}})$, когда хотим подчеркнуть, что $\mathcal{T}$ получен из ИГ U и I, где I — интерпретация, возникшая в результате замены индексов на переменные, $V_{\mathcal{T}}$ — упорядоченное множество вершин прикрепления. Пример простого шаблона изображен на рисунке 1 (жирным кружком обозначена единственная вершина прикрепления).

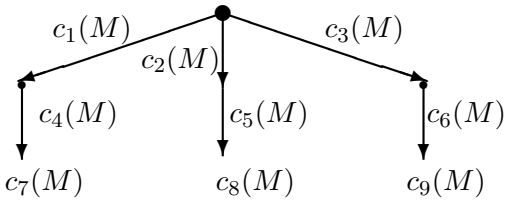

Рис. 2: Пример шаблона $\mathcal{T}_2$

Рассмотрим ИГ $U^{'}$, выделим в нем множество вершин $V_{U'}$, которые назовем вершинами прикрепления, и занумеруем их числами от 1 до $|V_{U'}|$ (вершины прикрепления образуют упорядоченное множество). Будем говорить, что ИГ $U^{'}$ и простой шаблон $\mathcal{T}$ *согласованы*, если выполнены следующие условия:

- $U^{'}$ и $\mathcal{T}$ совпадают как графы, и если в ИГ $U^{'}$ встречаются одинаковые индексы предикатов и записи, то в соответствующих местах шаблона $\mathcal{T}$ находятся одинаковые переменные из $\mathcal{P}_J$ и $\mathcal{P}_L$;

- i—ая вершина прикрепления ИГ $U^{'}$ совпадает с i—ой вершиной прикрепления простого шаблона $\mathcal{T}$, $i = 1, \ldots, |V_{U'}|$ и $|V_{U'}| = |V_{\mathcal{T}}|$.

То есть $\mathcal{T} = \mathcal{T}(U^{'}, I, V_{\mathcal{T}})$ для некоторой интерпретации I, и будем писать $\mathcal{T} = \mathcal{T}(U^{'}, I, V_{U'})$, когда хотим подчеркнуть, что ИГ $U^{'}$ и простой шаблон $\mathcal{T}$ согласованы.

Рассмотрим простой шаблон $\mathcal{T}_1$ и заменим в нем каждую переменную, на формулу над множество функций $\mathcal{C}$ и каким-либо множеством переменных $M \subseteq \mathcal{P}_J \cup \mathcal{P}_L$. В результате, полученный граф назовем *шаблоном* $\mathcal{T}_2$. При этом будем писать $\mathcal{T}_2 = \mathcal{T}_2(\mathcal{T}_1, M, V_{\mathcal{T}_1})$, когда хотим подчеркнуть, что $\mathcal{T}_2$ получен из простого шаблона $\mathcal{T}_1$, множества переменных M и упорядоченного множеств вершин прикрепления $V_{\mathcal{T}_1}$.

На рисунках запись $R_i(M)$ будет означать формулу над над множество функций $\mathcal{C}$ и множеством переменных из M.

Пример шаблона приведен на рисунке 2 (жирным кружком обозначена единственная вершина прикрепления).

Рассмотрим простой шаблон $\mathcal{T}_1 = \mathcal{T}_1(U, I, V_{\mathcal{T}_1})$. Обозначим множество входящих в него переменных $M(\mathcal{T}_1)$. Пусть $p_L^0 \in \mathcal{P}_L \setminus M(\mathcal{T}_1)$.

Преобразованием R назовем тройку

$$R = (\mathcal{T}_1(U, I, V_{\mathcal{T}_1}), \mathcal{T}_2(\mathcal{T}, M(\mathcal{T}_1) \cup \{p_L^0\}, V_{\mathcal{T}}), \xi),$$

где $\mathcal{T}_1$ — простой шаблон, $\mathcal{T}_2$ — шаблон (полученный из простого шаблона $\mathcal{T}$) в формулах которого встречаются только переменные из множества

$M(\mathcal{T}_1) \cup p_L^0$, $V_\mathcal{T}$ упорядоченное множество вершин прикрепления простого шаблона $\mathcal{T}$, ξ — биективная функция сопоставления вершин прикрепления ($\xi : V_{\mathcal{T}_1} \to V_\mathcal{T}$). Левой частью преобразования R будем называть простой шаблон $\mathcal{T}_1$.

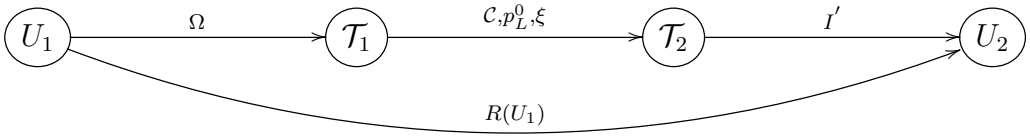

Рис. 3: Преобразование ИГ U_1 в ИГ U_2

Пусть ИГ U_1 и простой шаблон $\mathcal{T}_1$ согласованы, то есть $\mathcal{T}_1 = \mathcal{T}_1(U_1, I, V_{U_1})$ для некоторой интерпретации I, и пусть $I^{'}$ – интерпретация множества переменных $M(\mathcal{T}_1) \cup p_L^0$, причем $I^{'} = I$ на множестве переменных $M(\mathcal{T}_1)$. Тогда применением преобразования $R = (\mathcal{T}_1(U, I, V_{U_1}), \mathcal{T}_2(\mathcal{T}, M(\mathcal{T}_1) \cup p_L^0, V_\mathcal{T}), \xi)$ к ИГ U_1 назовем ИГ U_2, получающийся из шаблона $\mathcal{T}_2(\mathcal{T}, M(\mathcal{T}_1) \cup p_L^0, V_\mathcal{T})$ подстановкой вместо каждой формулы значения данной формулы в интерпретации $I^{'}$. Заметим, что упорядоченное множество вершин прикрепления V_{U_2} в ИГ U_2 однозначно соответствует упорядоченному множеству вершин прикрепления V_{U_1} в ИГ U_1, так как ξ — биективная функция, и простой шаблон $\mathcal{T}_1$ согласован с ИГ U_1.

Результат применения преобразования R к ИГ U_1 будем обозначать $R(U_1) = U_2$. Преобразование ИГ U_1 в ИГ U_2 изображено на рисунке 3.

Следующие понятия будут введены для определения *кода информационного графа на запросе*. Код информационного графа на запросе будет использоваться для описания передвижения автомата по вершинам ИГ.

Рассмотрим ИГ U, $N(U)$ — множество входящих в него вершин. Пусть $\beta, \beta^{'} \in N(U)$, тогда путем $\pi(\beta, \beta^{'})$ назовем последовательность вершин и ребер ИГ U, которая начинается в вершине β, заканчивается в вершине $\beta^{'}$, и в этой последовательности два любых соседних элемента инциденты. Длиной пути $l(\pi(\beta, \beta^{'}))$ назовем количество ребер, участвующих в нем. Пусть $\Pi(\beta, \beta^{'})$ — множество всех путей из β в $\beta^{'}$. Тогда расстоянием между вершинами $\beta, \beta^{'} \in N(U)$ назовем минимальную из всех путей длину $l(\pi(\beta, \beta^{'})), \pi(\beta, \beta^{'}) \in \Pi(\beta, \beta^{'})$, и обозначим $(\beta, \beta^{'}) = \min\{l(\pi(\beta, \beta^{'})) : \pi(\beta, \beta^{'}) \in \Pi(\beta, \beta^{'})\}$. Эксцентриситетом вершины $\beta \in N(U)$ назовем число $\varepsilon(\beta) = \max\limits_{\beta^{'} \in N(U)} (\beta, \beta^{'})$. *Радиусом* ИГ U назовем число $r(U) = \min\limits_{\beta \in N(U)} \varepsilon(\beta)$.

Через $\mathcal{G}(N, R)$, $N, R \in \mathbb{N}$ обозначим класс ИГ радиуса не больше R таких, что количество ребер инцидентных любой вершине графа не превосходит N.

Рассмотрим ИГ U, $E(U)$ — множество входящих в него ребер. Ребро инцидентное вершинам $\beta_1, \beta_2 \in N(U)$ будем обозначать $e(\beta_1, \beta_2)$.

Рассмотрим ИГ U_2 и его подграф (подмножество ребер и инцидентным им вершин) U_1 (пишем $U_1 \subseteq U_2$). Множество $\Sigma(U_1, U_2) = \{\beta : \beta \in N(U_1)$ и $\exists \beta_1 \in N(U_2 \backslash U_1), \exists\ \beta_2 \in N(U_1),\ e(\beta_1, \beta) \in E(U_2),\ e(\beta_2, \beta) \in E(U_1)\}$ назовем множеством *граничных вершин информационных графов* U_1 *и* U_2.

Пусть количество различных типов предикатов и переключателей конечно и равно K_T. Сопоставим им в биективном соответствие натуральные числа от 1 до K_T. Пусть $U_1, U_2 \in \mathcal{G}(N, \infty)$ и $U_1 \subseteq U_2$ ($N < \infty$). Назовем *кодом вершины на запросе* $x \in X$ *относительно пары* (U_1, U_2) пятерку $(k_1, k_2, k_3, k_4, k_5)$, где $k_1 = 0$, если вершина предикатная; $k_1 = 1$, если она переключательная; $k_2 = 0$, если вершина корень; $k_2 = 1$, если вершина листовая; $k_2 = 2$ в остальных случаях; $k_3 \in \{1, \ldots, N+1\}$ и в случае переключательной вершины принимает значение соответствующего ей переключателя на запросе $x \in X$, если это значение принадлежит $\{1,\ldots,\mathrm{N}\}$, и $k_3 = N + 1$ во всех остальных случаях; $k_4 \in \{0, 1\}$, $k_4 = 1$, если вершина принадлежит множеству граничных вершин $\Sigma(U_1, U_2)$ и $k_4 = 0$, если не принадлежит; $k_5 \in \{0, 1, \ldots, K_T\}$ принимает значение 0 если вершина предикатная или номер типа соответствующего ей переключателя иначе.

Кодом ребра на запросе $x \in X$ *типа поиск относительно пары* (U_1, U_2) назовем тройку (k_1^r, k_2^r, k_3^r), где $k_1^r = 0$, если ребро предикатное; $k_1^r = 1$, если ребро переключательное; $k_2^r \in \{0, 1, \ldots, N\}$; $k_3^r \in \{0, 1, \ldots, K_T\}$ принимает значение 0 если ребро переключательное или тип предиката соответствующее этому ребру иначе. У предикатного ребра k_2^r равен значению переключателя на запросе $x \in X$. Таким образом предикатным ребрам приписан код $(0, 0, t)$ или $(0, 1, t)$, где t - тип предиката, а каждому переключательному ребру приписан код $(1, i, 0)$, где $i \in \{1, \ldots, N\}$ номер переключательного ребра.

В общем случае множества запросов X и множество записей Y могут быть разные. Поэтому для того, чтобы определить код вершин и ребер ИГ на запросах типа вставка и удаление введем дополнительную функцию $\eta : Y \to X^l$, $l \in \mathbb{N}$, $l < \infty$, $\eta = (\eta_1, \ldots, \eta_l)$.

Кодом вершины на запросе $y \in Y$ *типа вставка, удаление относительно пары* (U_1, U_2) будет l пятерок кодов вершин, которые соответствуют запросам $\eta_1(y), \ldots, \eta_l(y)$ типа поиск. *Кодом ребра на запросе* $x \in Y$ *типа вставка, удаление относительно пары* (U_1, U_2) будет значение l троек кодов ребер, которые соответствуют запросам $\eta_1(y), \ldots, \eta_l(y)$ типа поиск.

Рассмотрим ИГ U, в нем вершины и ребра имеют нагрузку. Граф $K(U)$ полученный из ИГ U удалением нагрузок вершин и ребер, назовем *каркасом* ИГ U.

Кодом ИГ U_1 относительно ИГ U_2 на запросе, назовем нагруженный каркас $K(U_1)$, где нагрузка каждой вершины $K(U_1)$ это код данной вершин на данном запросе относительно пары (U_1, U_2), а на нагрузка каждого ребра $K(U_1)$ это код данного ребра на данном запросе относительно пары (U_1, U_2). Через $\mathcal{K}(N, R)$ обозначим множество кодов ИГ U_1 относительно ИГ U_2, где U_1 пробегает класс ИГ $\mathcal{G}(N, R)$, U_2 пробегает класс ИГ $\mathcal{G}(N, R+1)$ и $U_1 \subseteq U_2$. Понятно, что $|\mathcal{K}(N, R)| < \infty$.

Пусть $N, R \in \mathbb{N}$, $\mathcal{P}$ — некоторое конечное множество преобразований, шаблоны которых порождены ИГ из $\mathcal{G}(N, R)$, и $P_0 \in \mathcal{P}$, где P_0 — тождественное преобразование такое, что $P_0(U) = U$ для произвольного ИГ U из $\mathcal{G}(N, R)$.

Пусть $N(U_1)$ множество вершин ИГ U_1. Рассмотрим множество вершин $\mathfrak{B} \subseteq N(U_1)$ и обозначим через $U(\mathfrak{B}, U_1)$ — ИГ, полученный из ИГ U_1 как подграф на вершинах $\mathfrak{B}$. Под подграфом на вершинах $\mathfrak{B}$ здесь понимается множество вершин $\mathfrak{B}$ и множество всех инцидентных им ребер.

Пусть $\mathcal{A}$ — конечный автомат. Будем считать, что автомат $\mathcal{A}$ перемещается по вершинам ИГ $U_2 \in \mathcal{G}(N, \infty)$ и в каждый момент времени обозревает окрестность U_1 текущей вершины радиуса R, т.е. будем считать, что входным символом автомата $\mathcal{A}$ является код обозреваемой окрестности относительно U_2. Понятно, что эти коды будут принадлежать $\mathcal{K}(N, R)$ и значит входным алфавитом автомата $\mathcal{A}$ является конечное множество $\mathcal{K}(N, R)$. Результатом этого автомата $\mathcal{A}$ на обозреваемой окрестности текущей вершины будет некоторое преобразование этой окрестности и перемещение в некоторую вершину преобразованной окрестности. Тем самым выходным символом автомата $\mathcal{A}$ будет пятерка $b = (\mathfrak{B}, \pi, R, \beta, e)$, где

- $\mathfrak{B} \subseteq N(U_1)$ определяет множество вершин обозреваемой окрестности к которому будет применено преобразование;
- π — функция нумерации граничных вершин $U(\mathfrak{B}, U_1)$ и U_2 (содержательно это множество граничных вершин $U(\mathfrak{B}, U_1)$ и U_1 объединенное со множеством вершин в коде которых $k_2 = 1$ и одновременно принадлежащих $U(\mathfrak{B}, U_1)$). Пронумерованные граничные вершины назовем вершинами прикрепления;
- R — преобразование из конечного множества $\mathcal{P}$ применяемое к ИГ $U(\mathfrak{B}, U_1)$, причем ИГ $U(\mathfrak{B}, U_1)$ с заданной функцией π нумерацией

вершин прикрепления согласован с левой частью преобразования R;

- $\beta \in N(R(U(\mathfrak{B}, U_1))) \cup \{U_1 \setminus U(\mathfrak{B}, U_1)\}$ — следующая текущая вершина автомата $\mathcal{A}$;

- $e \in \{0, 1, 2\}$ отвечает за продолжение и выдачу ответа автоматом $\mathcal{A}$. Автомат еще не нашел ответ и продолжает функционирование ($e = 0$), либо завершает функционирование и возвращает информацию о том, что искомая запись найдена ($e = 1$), либо завершает функционирование с информацией, что искомой записи нет ($e = 2$).

Множество всех таких выходных символов b образует выходной алфавит B автомата $\mathcal{A}$.

Пусть U — ИГ над базовым множеством $\langle F, G\rangle$ из класса $\mathcal{G}(N, \infty)$, тогда пару $(\mathcal{A}, U)$ назовем динамическим информационным графом (ДИГ) типа (N, R) над $\mathcal{F}{=}\langle F, G, \mathcal{P}, \eta\rangle$ и обозначим $\mathcal{D} = (\mathcal{A}, U)$.

Опишем теперь функционирование ДИГ. Определим функционирование ДИГ $\mathcal{D} = (\mathcal{A}, U)$ типа (N, R) над $\mathcal{F}{=}\langle F, G, \mathcal{P}, \eta\rangle$ на запросе. В начальный момент текущей вершиной объявляется корень ИГ U. Рассматривается ИГ U_1 как подграф $U(i)$, с центром в текущей вершине и радиуса R. На вход автомата $\mathcal{A}$ подается код ИГ U_1 относительно ИГ U на запросе. Пусть $b = (\mathfrak{B}, \pi, R, \beta, e) \in B$ выходная буква автомата $\mathcal{A}$, тогда ИГ U меняется по следующему правилу. В ИГ $U(i)$ ИГ $U(\mathfrak{B}, U_1)$ заменяется на ИГ $U^{'}$, полученной в результате применения преобразования R к ИГ $U(\mathfrak{B}, U_1)$ $(R(U(\mathfrak{B}, U_1)) = U^{'})$ и "прикрепляется" к ИГ $U(i)$ посредством функции π и функции сопоставления вершин прикрепления ξ из преобразования R. После этого текущее положение автомата $\mathcal{A}$ меняется на β и в зависимости от значения последней компоненты выходной буквы b ДИГ $\mathcal{D}_M$ либо продолжает функционирование ($e = 0$), либо завершает функционирование и возвращает информацию о том, что искомая запись найдена (вставлена или удалена)($e = 1$), либо завершает функционирование с информацией, что записи нет ($e = 2$).

Сложность и объем ДИГ $\mathcal{D} = (\mathcal{A}, U)$ типа (N, R) над базовым множеством $\mathcal{F}{=}\langle F, G, \mathcal{P}, \eta\rangle$ определяется для фиксированного ИГ U. Объемом ДИГ $\mathcal{D}$ будем называть объем ИГ U (количество ребер в графе) и обозначим $Q(\mathcal{D})$.

Определим сложность ДИГ $\mathcal{D}$ на запросе $x \in X$ типа вставка (удаление, поиск) и обозначим их соответственно $T_i(\mathcal{D}, x)$ ($T_d(\mathcal{D}, x)$, $T_s(\mathcal{D}, x)$). Для этого введем сложность выходного действия автомата $\mathcal{A}$.

Пусть $Z : \mathcal{P} \to \mathbb{N}$, входным параметром, которой является $R \in \mathcal{P}$, а значением сложность выполненного преобразования. Другими словами с помощью L вводится сложность каждого элемента из множества преобразований $\mathcal{P}$.

Последовательность преобразований, выполненных автоматом $\mathcal{A}$, в ходе функционирования ДИГ $\mathcal{D}$ на запросе $x \in X$ типа вставка обозначим $f_{\mathcal{A}}(x, 1)$ (типа удаление $f_{\mathcal{A}}(x, 2)$, типа поиск $f_{\mathcal{A}}(x, 3)$).

Сложностью ДИГ $\mathcal{D}$ на запросе $x \in X$ типа вставка (удаление) назовем

- $T_i(\mathcal{D}, x) = \sum\limits_{R \in f_{\mathcal{A}}(x,1)} Z(R)$ ($T_d(\mathcal{D}, x) = \sum\limits_{R \in f_{\mathcal{A}}(x,2)} Z(R)$, $T_s(\mathcal{D}, x) = \sum\limits_{R \in f_{\mathcal{A}}(x,3)} Z(R)$).

Сложностью ДИГ $\mathcal{D}$ назовем $\max\{T_i(\mathcal{D}), T_d(\mathcal{D}), T_s(\mathcal{D})\}$ и обозначим $T(\mathcal{D})$.

Перейдем к постановке задачи и ее решению с помощью ДИГ. Пусть $V \subset Y \subseteq \mathbb{R}$, $|V| < \infty$, $X = Y$. Скажем, что ДИГ $\mathcal{D}$ решает ЗПИО, если ответ на произвольный запрос $x \in X$ типа поиска равен $\{x\}$, если $x \in V$, и пуст в противном случае, и если на произвольном запросе типа вставки (удаления) записи $y \in Y$ результирующий ДИГ $\mathcal{D}$ выдает ответ $\{x\}$ на произвольный запрос $x \in X$ типа поиск, если $x \in V \cup \{y\}$ ($x \in V \setminus \{y\}$), и пуст в противном случае.

Пусть $J = \{(a) : a \in \mathbb{R}\}$, $K = \{(a, b) : a, b \in \mathbb{R}\}$.

$$G_{id} = \left\{ g_{a,b}(x) = \begin{cases} 1, & \text{если } x \le a; \\ 2, & \text{если, } a < x \le b; , (a,b) \in K \\ 3, & \text{иначе.} \end{cases} \right\}; \tag{2}$$

$$F_{id} = \left\{ f_{=,a}(x) = \begin{cases} 1, & \text{если } x = a; \\ 0, & \text{если } x \ne a. \end{cases} , a \in J \right\}; \tag{3}$$

$$\mathcal{F}^{id} = \langle F_{id} \cup G_{id} \rangle. \tag{4}$$

,

В качестве функции η_{id} всегда рассматриваем тождественную функцию $\eta_{id} : Y \to X$ (в ЗПИО $Y = X$), поэтому ее будем опускать в формулировках в дальнейшем.

Теорема 1. *Существует базовое множество преобразований $\mathcal{R}_{id}$ и существует ДИГ $\mathcal{D}_{id} = (\mathcal{A}_{id}, U_{id})$ типа $(5, 3)$ над базовым множеством $\langle F_{id}, G_{id}, \mathcal{R}_{id}, \eta_{id} \rangle$, который решает ЗПИО, причем*

$$T(D_{id}) \leq (\lambda_{max} + 1)([\log_2 |V|] + 1);$$
$$Q(D_{id}) \leq 3|V|,$$

где $\lambda_{max} = \max\{Z(R) : R \in \mathcal{P}_{id}\}$.

В конце первой главы модель обобщатся для решения задач с использованием параллельных процессов. Обобщенная модель носит название *потоковый динамический информационный граф* (ПДИГ).

Обобщим понятие запроса к базе данных на случай вставки, удаления и пустого действия (запрос не требует действий для него). Для этого рассмотрим алфавит $A = \{\text{П, В, У, }\Lambda\}$. Обобщенным запросом назовем пару $(a, x), (a, x) \in \tilde{X}$, где $\tilde{X} = A \times X$. В данной работе рассматривается задача поиска идентичного объекта поэтому множество запросов и множество записей одинаково и равно X. При этом второй аргумент обобщенного запроса (буква из алфавита A) будет означать необходимое действие, которое нужно осуществить для запроса: П — поиск, В — вставка, У — удаление, Λ — ничего не делать. Далее везде под словом запрос будем понимать обобщенный запрос.

Потоком запросов H, $H : \mathbb{N} \to \tilde{X}$, назовем последовательность запросов, поступающих к базе данных через равные промежутки времени — такты. Другими словами поток запросов H означает, что к базе данных в i-ый такт времени будет подан запрос $H(i)$. Обозначим множество всех потоков через $\mathcal{H}$.

Рассмотрим функцию W, $W : \mathcal{H} \to A^\infty$ (через A^∞ обозначено множество всех сверхслов в алфавите A) сопоставляющую потоку запросов H сверхслово $W(H)$ из алфавита A, причем i-ая буква сверхслова $W(H)$ (обозначим ее через $W(H)(i)$) такова, что $H(i) = (W(H)(i), x_i)$.

Для обработки потоков запросов введем понятие *потоковый динамический информационный граф* (ПДИГ) и обозначим его $\mathcal{D}_\Pi = (\mathcal{A}, U)$. Пусть дано бесконечное множество копий автомата $\mathcal{A}$. Будем считать, что несколько копий автомата $\mathcal{A}$ могут одновременно обслуживать несколько запросов. Например пусть имеется поток запросов на поиск $H = (\text{П}, x_1), (\text{П}, x_2), (\text{П}, x_3), \ldots$, тогда при наличии 3 и более копий автомата $\mathcal{A}$ ПДИГ сможет параллельно обслужить три запроса x_1, x_2, x_3 на поиск, не дожидаясь завершения каждого ДИГ в отдельности. Если в потоке запросов H встречаются запросы на изменение базы данных (вставка и удаление), то при параллельной обработки этих запросов возможны конфликты преобразований (ИГ общий для всех автоматов).

Определим *функционирование ПДИГ* $\mathcal{D}_П, \mathcal{D}_П = (\mathcal{A}, U)$, *типа* (N, R) *над* $\mathcal{F}, \mathcal{F} = \langle F, \emptyset, \mathcal{P}, \eta_{id} \rangle$, *для потока запросов* H *из* $\mathcal{H}$. Считаем, что время применения любого преобразования R из $\mathcal{P}$, ровно 1 такт. Тем самым автомат $\mathcal{A}$ за один такт делает ровно одно преобразование над ИГ U. Заметим, что мы всегда можем выбрать единицу измерения времени (такт), что это будет выполнено, например, как максимум из времен всех преобразований. Так же мы предполагаем, что такт запроса совпадает с тактом работы автомата.

После каждого такта работы ПДИГ — ИГ U может изменяться, поэтому будем рассматривать $U = U(t)$ $(t \in \mathbb{N} \cup \{0\})$. В начальный момент функционирования $U = U(0)$.

Рассмотрим запрос $H(i), H(i) = (a_i, x_i)$, который поступил для обработки к ПДИГ $\mathcal{D}_\Pi, \mathcal{D}_\Pi = (\mathcal{A}, U)$, $U = U(i)$ в i-ый такт, $i \geq 1$. ПДИГ действует в зависимости от типа запроса a_i. Если $a_i = \Lambda$ (ничего не делать), то ПДИГ ничего не делает для этого запроса и $U(i+1) = U(i)$, иначе ПДИГ функционирует как ДИГ для данного запроса.

Теперь определим понятие *сложности* ПДИГ. Для любого i из $\mathbb{N}$ через $T(\mathcal{D}_\Pi, H(i))$ обозначим количество тактов необходимое для обработки запроса $H(i)$ динамическим информационным графом $\mathcal{D}_\Pi$.

В второй главе доказываются верхние оценки на ПДИГ. Сначала опишем формальную постановку задачи. Пусть H — поток запросов, $H : \mathbb{N} \to X$, где $X, X = \{\text{П}, \text{В}, \text{У}, \Lambda\} \times \mathbb{N}$ — множество запросов. Буква в запросе означает его действие: поиск (П), вставка (В), удаление (У) или пустой запрос (Λ).

Множество всех потоков запросов обозначим через $\mathcal{H}$. Рассмотрим функцию множества записей $V : \{\mathbb{N} \cup \{0\}\} \times \mathcal{H} \to 2^X$, которая удовлетворяет следующим условиям:

$$V(i, H) =$$
$$\begin{cases} \emptyset, & \text{если } i = 0; \\ V(i-1, H), & \text{если } i > 0 \text{ и } H(i) \text{ запрос на поиск или пустой запрос;} \\ V(i-1, H) \cup \{x\}, & \text{если } i > 0 \text{ и } H(i) = (\text{В}, x); \\ V(i-1, H) \setminus \{x\}, & \text{если } i > 0 \text{ и } H(i) = (\text{У}, x). \end{cases}$$

Функция V составляет каждому такту i и потоку запросов H — множество записей, которые образуют базу данных после завершения функционирования ПДИГ для всех запросов до такта i включительно, если обработка любого запроса происходила бы мгновенно (за 1 такт).

Пусть дана функция $L : \mathbb{N} \cup \{0\} \to \mathbb{N}$. Будем говорить, что ПДИГ решает ДЗПИО (*логическую ДЗПИО*) со сложностью L, если для любого потока H и любого такта i выполняются следующие условия

- если $H(i) = (\text{П}, x)$, то результатом функционирования ПДИГ должен быть $\{x\}$ (да), если $x \in V(i, H)$ и пустое множество (нет) в противном случае. При этом количество тактов, необходимое на выдачу ответа, должно не превосходить $L(|V(i, H)|)$;

- если $H(i) = (\text{В}, x)$, то для любого запроса на поиск (П, z), поступившего в такт $i + 1$, результат функционирования ПДИГ должен быть $\{z\}$(да), если $z \in V(i+1, H) = V(i, H) \cup \{x\}$, и пустое множество(нет) в противном случае;

- если $H(i) = (\text{У}, x)$, то для любого запроса на поиск (П,z), поступившего в такт $i + 1$, результат функционирования ПДИГ должен быть $\{z\}$(да), если $z \in V(i+1, H) = V(i, H) \setminus \{x\}$, и пустое множество(нет) в противном случае.

Введем еще два понятие как *конечный* и *селекторный ПДИГ*. Будем говорить, что *ПДИГ конечный*, если для произвольного потока запросов H и произвольного такта времени i существует такая константа C,$C < \infty$, что для потока запросов $H^{'}, H^{'}(1) = H(1), \ldots H^{'}(i) = H(i), H^{'}(i + 1) = \Lambda, \ldots, H^{'}(i + C) = \Lambda$, автомат обслуживающий запрос $H(i)$ завершит свое функционирование.

Неформально говоря, ПДИГ будем называть конечным, если автомат для любого потока запроса и любого такта, функционирует конечное время. ПДИГ, который не является конечным, будем называть бесконечным.

Введем понятие *селекторный ПДИГ*. Рассмотрим произвольную функцию $\phi \in \{\phi_1, \ldots, \phi_n\}$, которую может применить автомат в своем преобразовании. Автомат, зная тип предиката, знает также, сколько у него аргументов. Функция ϕ на вход получает аргументы и типы предикатов , типы вершин, а так же запись, которую обслуживает автомат. После этого эта функция может создать новый предикат или запись. Естественно, функция ϕ может создать только предикат, принадлежащий множеству предикатов F, над которым рассматривается ИГ. Оперируя этими функциями, автомат может в новой окрестности создавать новые предикаты из F.

Преобразование будем называть селекторным, если оно не может создать новый аргумент, кроме тех, что есть во входной окрестности или запроса, который обслуживает автомат.

Пусть

$$f_a(x) = \begin{cases} 1, & \text{если } x = a; \\ 0, & \text{иначе} \end{cases},$$

$$F_{list} = \{f_a(x) : a \in \mathbb{R}\}.$$

Через $]x[$ будем обозначать целую часть сверху от x (наименьшее целое число, которое не меньше x). Справедлива следующая теорема.

Теорема 2. *Существует множество преобразований $\mathcal{R}$ и существует конечный, селекторный ПДИГ типа (2,2) над базовым множеством $\langle F_{list}, \mathcal{R}\rangle$, который решает ДЗПИО для любого потока запросов H из $\mathcal{H}$, со сложностью $L, L(n) =]n/2[+1$.*

Пусть

$$f_a(x) = f_a^h(x) = \begin{cases} 1, & \text{если } x \le a; \\ 0, & \text{иначе} \end{cases}; f_{=,a}(x) = \begin{cases} 1, & \text{если } x = a; \\ 0, & \text{иначе} \end{cases};$$

$$f_a^{\text{П}}(x) = f_a^{\text{В}}(x) = f_a^{\text{У}}(x) = \begin{cases} 1, & \text{если } x = a; \\ 0, & \text{иначе} \end{cases}; f^h(x) \equiv 0;$$

$$F_{log} = \left\{f_a(x), f_a^h(x), f_{=,a}(x), f_a^{\text{П}}, f_a^{\text{В}}, f_a^{\text{У}} : a \in \mathbb{R}\right\} \cup \{f^h\}.$$

Справедлива следующая теорема.

Теорема 3. *Существует множество преобразований $\mathcal{R}$ и существует конечный, селекторный ПДИГ типа (8,4) над базовым множеством $\langle F_{log}, \mathcal{R}\rangle$, который решает ДЗПИО для любого потока запросов $H, H \in \mathcal{H}$, со сложностью $L, L(n) = \left]\frac{\log_2 \max(n,2)}{3}\right[+ 1$.*

Рассмотрим множество предикатов $F(T) = \{f^0, f^1, f_a^t(x) | t \in T, a \in \mathbb{N}\}$, где $|T| < \infty$ и

$\forall t \in T, a \in \mathbb{N}$:

$$f_a^t(x) = \begin{cases} 1, & \text{если } x = a; \\ 0, & \text{иначе} \end{cases}, f^0(x) \equiv 0, f^1(x) \equiv 1.$$

Пусть

$$T_0 = \{\text{П}, \text{В}_1, \text{В}_2, \text{У}\},$$

то есть $F(T_0) = \{f^0, f^1, f_a^{\text{П}}, f_a^{\text{В}_1}, f_a^{\text{В}_2}, f_a^{\text{У}} : a \in \mathbb{N}\}$.

Справедлива следующая теорема.

Теорема 4. *Существует множество преобразований $\mathcal{R}$ и существует бесконечный, селекторный ПДИГ типа (1,1) над базовым множеством $\langle F(T_0), \mathcal{R}\rangle$, который решает ДЗПИО для любого потока запросов H из $\mathcal{H}$ со сложность $L, L \equiv 3$.*

Пусть

$$T_1 = \{\text{да}, \text{нет}\},$$

то есть $F(T_1) = \{f^0, f^1, f_a^{\text{да}}, f_a^{\text{нет}} : a \in \mathbb{N}\}$.

Теорема 5. *Существует множество преобразований $\mathcal{R}$ и существует конечный, не селекторный ПДИГ типа (1,1) над базовым множеством $\langle F(T_1), \mathcal{R}\rangle$, решающий логическую ДЗПИО со сложностью $L, L \equiv 2$.*

Как будет доказано ниже, не существует конечного ПДИГ типа (1,1), решающего ДЗПИО. В предыдущей теореме конечный ПДИГ типа (1,1) решает логическую ДЗПИО.

Следующая теорема утверждает, что существует конечный не селекторный ПДИГ типа (2,1), который решает логическую ДЗПИО.

Пусть

$$T = \{\text{а}, \text{н}\} \times \{-2, -1, 0, 1, 2\} \times \{\text{П}_{\leftarrow}, \text{П}_{\rightarrow}, \text{В}_{\leftarrow}, \text{В}_{\rightarrow}, \text{У}_{\leftarrow}, \text{У}_{\rightarrow}, \text{н}\},$$

$$F_2(T) = \{f_{l,r,a}^t(x) : t \in T, l, r, a, x \in \mathbb{N}\}, \text{ где}$$

$f_{l,r,a}^t(x) = 1$, если $x = a$ и 0 иначе.

Теорема 6. *Существует множество преобразований $\mathcal{R}$ и существует конечный ПДИГ типа (2,1) над базовым множеством $\langle F_2(T), \mathcal{R}\rangle$, решающий ДЗПИО со сложностью $L, L(n) =]n/2[+5$.*

Как видно из предыдущих теорем ПДИГ достаточно мощный аппарат для решения ДЗПИО. Поэтому найти ограничения на ПДИГ для доказательства не существования алгоритма — еше одна интересная задача для математиков. В частности, предыдущие теоремы 4 — 6 получились в результате поисков ограничений на ПДИГ.

В третьей главе доказываются нижние оценки для ПДИГ.

Теорема 7. *Для любого натурального R, не существует конечного ПДИГ типа $(1, R)$, решающего ДЗПИО для любого потока запросов.*

Теорема 8. *Для любой функции $L, L : \mathbb{N} \to \mathbb{N}$ и для любого множества $T, |T| < \infty$ не существует конечного, селекторного ПДИГ типа (2, 1) над множеством $F(T)$, который решает логическую ДЗПИО со сложностью L.*

Заключение

В диссертации получены следующие основные результаты.

Разработана математическая модель динамических баз данных. Доказана применимость предлагаемой модели для решения задач, возникающих в динамических базах данных. На основе этой модели получены бесконечно распараллеливаемые структуры данных. Получены различные верхние и нижние оценки для решения ДЗПИО и логической ДЗПИО для любого потока запросов.

Построен конечный, селекторный ПДИГ типа (8,4), решающий ДЗПИО для любого потока запросов с логарифмической сложностью.

Доказано, что существует конечный, селекторный ПДИГ типа (2,2), решающий ДЗПИО для любого потока запросов. Так же доказано, что для любого натурального R не существует конечного ПДИГ типа $(1, R)$, который решает ДЗПИО для любого потока запросов, но существует конечный, не селекторный ПДИГ типа (1,1), решающий логическую ДЗПИО.

Показано, что не существует конечного, селекторного ПДИГ типа (2,1), решающего ДЗПИО для любого потока запросов, но существует бесконечный ПДИГ типа (1,1), решающий эту же задачу.

Предъявлен минимально возможный по степени ветвления конечный, не селекторный ПДИГ с радиусом видимости один, решающий ДЗПИО для любого потока запросов.

В качестве перспектив дальнейшей разработки темы диссертации можно предложить применять полученную модель динамических баз данных для решения других динамических задач информационного поиска, например, для решения динамической задачи двумерного интервального поиска.

Благодарность

Автор выражает глубокую благодарность своему научному руководителю — доктору физико-математических наук, профессору Эльяру Эльдаровичу

Гасанову за постановку задачи, постоянное внимание к работе и всестороннюю поддержку, а также заведующему кафедрой академику Валерию Борисовичу Кудрявцеву и всему коллективу кафедры математической теории интеллектуальных систем за доброжелательную и творческую атмосферу.

Публикации автора по теме диссертации

[1] Плетнев А. А. *Моделирование динамических баз данных*, Интеллектуальные системы Т. 17, Вып. 1-4, 2013, 75-79.

[2] Плетнев А. А. *Информационно-графовая модель динамических баз данных и ее применение*, Интеллектуальные системы. Теория и приложения. Т. 18, 2014, 111-140.

[3] Плетнев А. А. *Динамическая база данных, допускающая параллельную обработку произвольных потоков запросов*, Интеллектуальные системы. Теория и приложения. Т. 19, Вып. 1, 2015, 117-142.

[4] Плетнев А. А. *Логарифмическая по сложности параллельная обработка автоматами произвольных потоков запросов в динамической базе данных*, Интеллектуальные системы. Теория и приложения. Т. 19, Вып. 1, 2015, 171-212.

[5] Плетнев А. А. *Нижняя оценка на область видимости автомата, обрабатывающего произвольный поток запросов к динамической базе данных*, Интеллектуальные системы. Теория и приложения. Т. 19, Вып. 4, 2015, 117-151.

[6] Плетнев А. А. *Минимально возможный по степени ветвления информационный граф с радиусом видимости один, обрабатывающий произвольный поток запросов к динамической базе данных*, Интеллектуальные системы. Теория и приложения. Т. 20, Вып. 1, 2016, 223-254.