На правах рукописи

ИВАНОВА Елена Владимировна

# МЕТОДЫ ПАРАЛЛЕЛЬНОЙ ОБРАБОТКИ СВЕРХБОЛЬШИХ БАЗ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ РАСПРЕДЕЛЕННЫХ КОЛОНОЧНЫХ ИНДЕКСОВ

05.13.11 — математическое и программное обеспечение
вычислительных машин, комплексов и компьютерных сетей

Автореферат
диссертации на соискание ученой степени
кандидата физико-математических наук

Челябинск — 2015

Работа выполнена на кафедре системного программирования
ФГБОУ ВПО «Южно-Уральский государственный университет»
(национальный исследовательский университет)

**Научный руководитель:** СОКОЛИНСКИЙ Леонид Борисович
доктор физ.-мат. наук, профессор,
проректор по информатизации, ФГБОУ ВПО
«Южно-Уральский государственный университет»
(национальный исследовательский университет)

**Официальные оппоненты:** КУЗНЕЦОВ Сергей Дмитриевич
доктор тех. наук, профессор, главный научный
сотрудник, ФГБУН Институт системного
программирования РАН

ПАН Константин Сергеевич
кандидат физ.-мат. наук, разработчик,
ООО «ПОСТГРЕС ПРОФЕССИОНАЛЬНЫЙ
РАЗРАБОТКА»

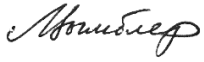
**Ведущая организация:** ФГБОУ ВПО «Московский государственный
университет имени М.В. Ломоносова»

Защита состоится 23 декабря 2015 г. в 12:00 часов на заседании диссертационного совета Д 212.298.18 при ФГБОУ ВПО «Южно-Уральский государственный университет» (национальный исследовательский университет) по адресу: 454080, г. Челябинск, пр. Ленина, 76, ауд. 1001.

С диссертацией можно ознакомиться в библиотеке Южно-Уральского государственного университета и на сайте:
http://susu.ac.ru/ru/dissertation/d-21229818/ivanova-elena-vladimirovna.

Автореферат разослан 20 ноября 2015 г.

Ученый секретарь
диссертационного
совета



М.Л. Цымблер

## Общая характеристика работы

**Актуальность темы.** В настоящее время одним из феноменов, оказывающих существенное влияние на область технологий обработки данных, являются сверхбольшие данные. Согласно прогнозам аналитической компании IDC, количество данных в мире удваивается каждые два года и к 2020 г. достигнет 44 Зеттабайт, или 44 триллионов гигабайт. Сверхбольшие данные путем очистки и структурирования преобразуются в сверхбольшие базы и хранилища данных. По оценке IDC в 2013 г. из всего объема существующих данных потенциально полезны 22%, из которых менее 5% были подвергнуты анализу. К 2020 году процент потенциально полезных данных может вырасти до 35%, преимущественно за счет данных от встроенных систем.

По мнению одного из ведущих специалистов мира в области баз данных М. Стоунбрейкера (Michael Stonebraker) для решения проблемы обработки сверхбольших данных необходимо использовать технологии СУБД. Для обработки больших данных необходимы высокопроизводительные вычислительные системы. В этом сегменте вычислительной техники сегодня доминируют системы с кластерной архитектурой, узлы которых оснащены многоядерными ускорителями. Недавние исследования показывают, что кластерные вычислительные системы могут эффективно использоваться для хранения и обработки сверхбольших баз данных. Однако в этой области остается целый ряд нерешенных масштабных научных задач, в первую очередь связанных с проблемой больших данных.

В последние годы основным способом наращивания производительности процессоров является увеличение количества ядер, а не тактовой частоты, и эта тенденция, вероятно, сохранится в ближайшем обозримом будущем. Сегодня GPU (Graphic Processing Units) и Intel MIC (Many Integrated Cores) значительно опережают традиционные процессоры в производительности по арифметическим операциям и пропускной способности памяти, позволяя использовать сотни процессорных ядер для выполнения десятков тысяч потоков. Последние исследования показывают, что многоядерные ускорители могут эффективно использоваться для обработки запросов к базам данных, хранящимся в оперативной памяти.

Одним из наиболее важных классов приложений, связанным с обработкой сверхбольших баз данных, являются хранилища данных, для которых характерны запросы типа OLAP. Исследования показали, что для таких приложений выгодно использовать колоночную модель представления данных, позволяющую получить на порядок лучшую производительность по сравнению с традиционными системами баз данных, использующими строчную модель представления данных. Эта разница в производительности объясняется тем, что колоночные хранилища позволяют выполнять меньше обменов с дисками при выполнении запросов на выборку данных, поскольку с диска (или из основной памяти) считываются значения только тех атрибутов, которые упоминаются в запросе. Дополнительным преимуществом колоночного представления является возможность использования эффективных алгоритмов сжатия данных, поскольку в одной колонке таблицы содержатся данные одного типа. Сжатие может привести к повышению производительности на порядок, поскольку меньше времени занимают операции ввода-вывода. Недостатком колоночной модели представления данных является то, что в колоночных СУБД затруднено применение техники эффективной оптимизации SQL-запросов, хорошо зарекомендовавшей себя в реляционных СУБД. Кроме этого, колоночные СУБД значительно уступают строковым по производительности на запросах класса OLTP.

В соответствие с этим актуальной является задача разработки новых эффективных методов параллельной обработки сверхбольших баз данных в оперативной памяти на кластерных вычислительных системах, оснащенных многоядерными ускорителями, которые позволили бы совместить преимущества реляционной модели с колоночным представлением данных.

**Цель и задачи исследования**. *Цель* данной работы состояла в разработке и исследовании эффективных методов параллельной обработки сверхбольших баз данных с использованием колоночного представления информации, ориентированных на кластерные вычислительные системы, оснащенные многоядерными ускорителями, и допускающих интеграцию с реляционными СУБД. Для достижения этой цели необходимо было решить следующие *задачи*:

1. На основе колоночной модели хранения информации разработать вспомогательные структуры данных (колоночные индексы), позволяющие ускорить выполнение ресурсоемких реляционных операций.
2. Разработать методы фрагментации (распределения) колоночных индексов, минимизирующие обмены данными между вычислительными узлами при выполнении реляционных операций.
3. На основе использования распределенных колоночных индексов разработать методы декомпозиции основных реляционных операций, позволяющие организовать параллельное выполнение запросов без массовых пересылок данных между вычислительными узлами.
4. Реализовать предложенные модели и методы в виде колоночного сопроцессора СУБД, работающего на кластерных вычислительных системах с многоядерными ускорителями Intel Xeon Phi.
5. Провести вычислительные эксперименты, подтверждающие эффективность предложенных подходов.

**Научная новизна** работы заключается в разработке автором оригинальной доменно-колоночной модели представления данных, на базе которой введены колоночные индексы с доменно-интервальной фрагментацией, и выполнением на ее основе декомпозиции основных операций реляционной алгебры. По сравнению с ранее известными методами параллельной обработки больших объемов данных предложенный подход позволяет сочетать эффективность колоночной модели хранения данных с возможностью использования мощных механизмов оптимизации запросов, разработанных для реляционной модели.

**Теоретическая ценность** работы состоит в том, что в ней дано формальное описание методов параллельной обработки сверхбольших баз данных с использованием распределенных колоночных индексов, включающее в себя доменно-колоночную модель представления данных. **Практическая ценность** работы заключается в том, что на базе предложенных методов и алгоритмов разработан колоночный сопроцессор для кластерной вычислительной системы с многоядерными ускорителями, позволяющий во взаимодействии с СУБД PostgreSQL получить линейное ускорение при выполнении ресурсоемких реляционных операций. Результаты, полученные в диссертационной работе, могут быть использованы для создания колоночных сопроцессоров для других коммерческих и свободно распространяемых реляционных СУБД.

**Методы исследования.** Методологической основой диссертационного исследования является теория множеств и реляционная алгебра. Для организации параллельной обработки запросов использовались методы горизонтальной фрагментации отношений и методы организации параллельных вычислений на основе стандартов MPI и OpenMP. При разработке колоночного сопроцессора применялись методы объектно-ориентированного проектирования и язык UML.

**Степень достоверности результатов.** Все утверждения, связанные с декомпозицией реляционных операций, сформулированы в виде теорем и снабжены строгими доказательствами. Теоретические построения подтверждены тестами, проведенными в соответствии с общепринятыми стандартами.

**Апробация работы.** Основные положения диссертационной работы, разработанные модели, методы, алгоритмы и результаты вычислительных экспериментов докладывались автором на следующих международных научных конференциях:

- 38-я международная научная конференция по информационно-коммуникационным технологиям, электронике и микроэлектронике MIPRO'2015, 25-29 мая 2015 г., Хорватия, г. Опатия;
- Международная научная конференция «Суперкомпьютерные дни в России», 28-29 сентября 2015 г., Москва;
- Международная научная конференция «Параллельные вычислительные технологии (ПаВТ'2015)», 30 марта – 3 апреля 2015 г., Екатеринбург;
- Международная научная конференция «Параллельные вычислительные технологии (ПаВТ'2014)», 1-3 апреля 2014 г., Ростов-на-Дону;
- Международная суперкомпьютерная конференция «Научный сервис в сети Интернет: многообразие суперкомпьютерных миров», 22-27 сентября 2014 г., Новороссийск.

**Публикации.** По теме диссертации опубликовано 10 печатных работ. Работы [1-3] опубликованы в журналах, включенных ВАК в перечень изданий, в которых должны быть опубликованы основные результаты диссертаций на соискание ученой степени доктора и кандидата наук. Работа [4] опубликована в издании, индексируемом в SCOPUS и Web of Science. В работах [1-8] научному руководителю Л.Б. Соколинскому принадлежит постановка задачи, Е.В. Ивановой — все полученные результаты. В рамках выполнения диссертационной работы получено одно свидетельство Роспатента об официальной регистрации программ для ЭВМ и баз данных.

**Структура и объем работы.** Диссертация состоит из введения, четырех глав, заключения и библиографии. В приложении 1 приведены основные обозначения, используемые в диссертации. Приложение 2 содержит сводку по операциям расширенной реляционной алгебры. Объем диссертации составляет 143 страницы, объем библиографии — 143 наименования.

## Содержание работы

**Во введении** приводится обоснование актуальности темы и степень ее разработанности; формулируются цели и задачи исследования; раскрываются новизна, теоретическая и практическая значимость полученных результатов; приводятся данные об апробациях и публикациях автора; дается обзор содержания диссертации.

**В первой главе, «Современные тенденции в развитии аппаратного обеспечения и технологий баз данных»,** рассматриваются тенденции развития аппаратного обеспечения и дается обзор научных исследований в области современных технологий баз данных. Особое внимание уделяется методам обработки баз данных на вычислительных системах с многоядерными ускорителями и колоночной модели хранения данных. Анализируются публикации, наиболее близко относящихся к теме диссертации.

**Во второй главе, «Доменно-колоночная модель»,** строится формальная доменно-колоночная модель представления данных. Вводятся колоночные индексы. Описывается оригинальный способ фрагментации колоночных индексов, названный доменно-интервальной фрагментацией. Вводится понятие транзитивной фрагментации одного колоночного индекса относительно другого для атрибутов, принадлежащих одному и тому же отношению.

Рассматриваются методы декомпозиции реляционных операций на основе использования фрагментированных колоночных индексов.

Пусть задано отношение $R(A,B,\ldots)$, $T(R)=n$. Атрибут $A$ в отношении $R$ играет роль суррогатного ключа. Пусть на множестве $\mathfrak{D}_B$, являющемся доменом атрибута $B$, задано отношение линейного порядка. *Колоночным индексом* $I_{R.B}$ атрибута $B$ отношения $R$ будем называть упорядоченное отношение $I_{R.B}(A,B)$, удовлетворяющее следующим свойствам:

$$T(I_{R.B})=n \text{ и } \pi_A(I_{R.B})=\pi_A(R);$$

$$\forall x_1,x_2\in I_{R.B}\left(x_1\le x_2 \Leftrightarrow x_1.B\le x_2.B\right);$$

$$\forall r\in R\left(\forall x\in I_{R.B}\left(r.A=x.A\Rightarrow r.B=x.B\right)\right).$$

С содержательной точки зрения колоночный индекс $I_{R.B}$ представляет собой таблицу из двух колонок с именами $A$ и $B$. Количество строк в колоночном индексе совпадает с количеством строк в индексируемой таблице. Колонка $B$ индекса $I_{R.B}$ включает в себя все значения колонки $B$ таблицы $R$ (с учетом повторяющихся значений), отсортированных в порядке возрастания. Каждая строка $x$ индекса $I_{R.B}$ содержит в колонке $A$ суррогатный ключ (адрес) строки $r$ в таблице $R$, имеющей такое же значение в колонке $B$, что и $x$.

Пусть на множестве значений домена $\mathfrak{D}_B$ задано отношение линейного порядка. Разобьем множество $\mathfrak{D}_B$ на $k>0$ непересекающихся интервалов:

$$V_0=[v_0;v_1),V_1=[v_1;v_2),\ldots,V_{k-1}=[v_{k-1};v_k);$$

$$v_0<v_1<\ldots<v_k;$$

$$\mathfrak{D}_B=\bigcup_{i=0}^{k-1}V_i.$$

Отметим, что в случае $\mathfrak{D}_B=\mathbb{R}$ будем иметь $v_0=-\infty$ и $v_k=+\infty$. Функция $\varphi_{\mathfrak{D}_B}:\mathfrak{D}_B\to\{0,\ldots,k-1\}$ называется доменной функцией фрагментации для $\mathfrak{D}_B$, если она удовлетворяет следующему условию:

$$\forall i\in\{0,\ldots,k-1\}\left(\forall b\in\mathfrak{D}_B\left(\varphi_{\mathfrak{D}_B}(b)=i\Leftrightarrow b\in V_i\right)\right).$$

Другими словами, доменная функция фрагментации сопоставляет значению $b$ — номер интервала, которому это значение принадлежит.

Пусть задан колоночный индекс $I_{R.B}$ для отношения $R(A,B,\ldots)$ с атрибутом $B$ над доменом $\mathfrak{D}_B$ и доменная функция фрагментации $\varphi_{\mathfrak{D}_B}$. Функция $\varphi_{I_{R.B}}:I_{R.B}\to\{0,\ldots,k-1\}$, определенная по правилу $\forall x\in I_{R.B}\left(\varphi_{I_{R.B}}(x)=\varphi_{\mathfrak{D}_B}(x.B)\right)$, называется *доменно-интервальной функцией фрагментации* для индекса $I_{R.B}$. Другими словами, функция фрагментации $\varphi_{I_{R.B}}$ сопоставляет каждому кортежу $x$ из $I_{R.B}$ номер доменного интервала, которому принадлежит значение $x.B$.

Определим $i$-тый фрагмент $(i=0,\ldots,k-1)$ индекса $I_{R.B}$ следующим образом:

$$I_{R.B}^i=\left\{x\mid x\in I_{R.B};\ \varphi_{I_{R.B}}(x)=i\right\}.$$

Это означает, что в $i$-тый фрагмент попадают кортежи, у которых значение атрибута $B$ принадлежит $i$-тому доменному интервалу. Будем называть фрагментацию, построенную таким

образом, *доменно-интервальной*. Количество фрагментов $k$ будем называть *степенью фрагментации*.

Пусть для отношения $R(A,B,C,\ldots)$ заданы колоночные индексы $I_{R.B}$ и $I_{R.C}$. *Транзитивной фрагментацией* индекса $I_{R.C}$ относительно индекса $I_{R.B}$ называется фрагментация, задаваемая функцией $\ddot{\varphi}_{I_{R.C}} : I_{R.C} \to \{0,\ldots,k-1\}$, удовлетворяющей условию: $\forall x \in I_{R.C}$

$$\ddot{\varphi}_{I_{R.C}}(x) = \varphi_{I_{R.B}}\left(\sigma_{A=x.A}(I_{R.B})\right).$$

Транзитивная фрагментация позволяет разместить на одном и том же узле элементы колоночных индексов, соответствующие одному кортежу индексируемого отношения.

Далее рассматривается декомпозиция реляционных операций на основе использования фрагментированных колоночных индексов. Декомпозиция заключается в разбиении ресурсоемких вычислений на отдельные подзадачи, которые могут выполняться в виде независимых процессов, не требующих обменов данными.

***Декомпозиция естественного соединения.*** Пусть заданы два отношения $R(A,B_1,\ldots,B_u,C_1,\ldots,C_v)$ и $S(A,B_1,\ldots,B_u,D_1,\ldots,D_w)$. Пусть имеется два набора колоночных индексов по атрибутам $B_1,\ldots,B_u$: $I_{R.B_1},\ldots,I_{R.B_u}$; $I_{S.B_1},\ldots,I_{S.B_u}$. Пусть для всех этих индексов задана доменно-интервальная фрагментация степени $k$: $I_{R.B_j} = \bigcup_{i=0}^{k-1} I^i_{R.B_j}$; $I_{S.B_j} = \bigcup_{i=0}^{k-1} I^i_{S.B_j}$. Положим

$$P^i_j = \pi_{I^i_{R.B_j}.A \to A_R,\, I^i_{S.B_j}.A \to A_S}\left(I^i_{R.B_j} \underset{I^i_{R.B_j}.B_j = I^i_{S.B_j}.B_j}{\bowtie} I^i_{S.B_j}\right)$$

для всех $i = 0,\ldots,k-1$ и $j = 1,\ldots,u$. Определим $P_j = \bigcup_{i=0}^{k-1} P^i_j$. Положим $P = \bigcap_{j=1}^{u} P_j$. Построим отношение $Q(B_1,\ldots,B_u,C_1,\ldots,C_v,D_1,\ldots,D_w)$ следующим образом:

$$\begin{aligned} Q = \{ & (\&_R(p.A_R).B_1,\ldots,\&_R(p.A_R).B_u, \\ & \&_B(p.A_B).C_1,\ldots,\&_B(p.A_B).C_v, \\ & \&_S(p.A_S).D_1,\ldots,\&_S(p.A_S).D_w) \mid p \in P\}. \end{aligned}$$

Тогда $Q = \pi_{*\setminus A}(R) \bowtie \pi_{*\setminus A}(S)$.

***Декомпозиция операции группировки*** $\gamma_{B,C_1,\ldots,C_u,\mathrm{agrf}(D_1,\ldots,D_w)\to F}(R)$. Пусть задано отношение $R(A,B,C_1,\ldots,C_u,D_1,\ldots,D_w,\ldots)$ с суррогатным ключом $A$. Пусть для атрибутов $D_1,\ldots,D_w$ задана агрегирующая функция **agrf**. Пусть имеется колоночный индекс $I_{R.B}$. Пусть также имеются колоночные индексы: $I_{R.C_1},\ldots,I_{R.C_u}$; $I_{R.D_1},\ldots,I_{R.D_w}$. Пусть для индекса $I_{R.B}$ задана доменно-интервальная фрагментация степени $k$: $I_{R.B} = \bigcup_{i=0}^{k-1} I^i_{R.B}$. Пусть для индексов $I_{R.C_1},\ldots,I_{R.C_u}$ и $I_{R.D_1},\ldots,I_{R.D_w}$ задана транзитивная относительно $I_{R.B}$ фрагментация

$$\forall j \in \{1,\ldots,u\}\left(I_{R.C_j} = \bigcup_{i=0}^{k-1} I^i_{R.C_j}\right);\ \forall j \in \{1,\ldots,w\}\left(I_{R.D_j} = \bigcup_{i=0}^{k-1} I^i_{R.D_j}\right).$$

Положим $P_i = \pi_{A,F}\left(\gamma_{\min(A)\to A,B,C_1,\ldots,C_u,\text{agrf}(D_1,\ldots,D_w)\to F}\left(I^i_{R.B} \bowtie I^i_{R.C_1} \bowtie \ldots \bowtie I^i_{R.C_u} \bowtie I^i_{R.D_1} \bowtie \ldots \bowtie I^i_{R.D_w}\right)\right)$ для всех $i = 0,\ldots,k-1$. Определим $P = \bigcup_{i=0}^{k-1} P_i$. Построим отношение $Q(B,C_1,\ldots,C_u,F)$ следующим образом: $Q = \left\{\left(\&_R(p.A).B, \&_R(p.A).C_1,\ldots,\&_R(p.A).C_u, p.F\right) \mid p \in P\right\}$. Тогда $Q = \gamma_{B,C_1,\ldots,C_u,\text{agrf}(D_1,\ldots,D_w)\to F}(R)$.

***Декомпозиция операции пересечения*** $\pi_{B_1,\ldots,B_u}(R) \cap \pi_{B_1,\ldots,B_u}(S)$. Пусть заданы два отношения $R(A,B_1,\ldots,B_u)$ и $S(A,B_1,\ldots,B_u)$, имеющие одинаковый набор атрибутов. Пусть имеется два набора колоночных индексов по атрибутам $B_1,\ldots,B_u$: $I_{R.B_1},\ldots,I_{R.B_u}$; $I_{S.B_1},\ldots,I_{S.B_u}$. Пусть для всех этих индексов задана доменно-интервальная фрагментация степени $k$: $I_{R.B_j} = \bigcup_{i=0}^{k-1} I^i_{R.B_j}$; $I_{S.B_j} = \bigcup_{i=0}^{k-1} I^i_{S.B_j}$. Положим

$$P^i_j = \pi_{I^i_{R.B_j}.A\to A_R,\, I^i_{S.B_j}.A\to A_S}\left(I^i_{R.B_j} \underset{(I^i_{R.B_j}.B_j = I^i_{S.B_j}.B_j)}{\bowtie} I^i_{S.B_j}\right)$$

для всех $i = 0,\ldots,k-1$ и $j = 1,\ldots,u$. Определим $P_j = \bigcup_{i=0}^{k-1} P^i_j$. Положим $P = \bigcap_{j=1}^{u} P_j$. Построим отношение $Q(A,B_1,\ldots,B_u)$ следующим образом: $Q = \left\{r \mid r \in R \wedge r.A \in \pi_{A_R}(P)\right\}$. Тогда $\pi_{B_1,\ldots,B_u}(Q) = \pi_{B_1,\ldots,B_u}(R) \cap \pi_{B_1,\ldots,B_u}(S)$.

Аналогичным образом в диссертации выполняется декомпозиция для следующих операций: *проекция*, *выбор*, *удаление дубликатов* и *объединение*.

Далее вводится понятие *колоночного хеш-индекса*. Пусть задано отношение $R(A,B_1,\ldots,B_u,\ldots)$. Пусть задана инъективная хеш-функция $h: \mathfrak{D}_{B_1} \times \ldots \times \mathfrak{D}_{B_u} \to \mathbb{Z}_{\geq 0}$. *Колоночным хеш-индексом* $I_h(A,H)$ атрибутов $B_1,\ldots,B_u$ отношения $R$ будем называть упорядоченное отношение, удовлетворяющее тождеству: $I_h = \tau_H\left(\pi_{A,h(B_1,\ldots,B_u)\to H}(R)\right)$. *Фрагментация колоночного хеш-индекса* осуществляется на основе доменно-интервального принципа с помощью функции фрагментации $\varphi_{I_h}: I_h \to \{0,\ldots,k-1\}$, определенной следующим образом: $\forall x \in I_h\left(\varphi_{I_h}(x) = \varphi_{\mathbb{Z}_{\geq 0}}(x.H)\right)$, где $\varphi_{\mathbb{Z}_{\geq 0}}: \mathbb{Z}_{\geq 0} \to \{0,\ldots,k-1\}$ — доменная функция фрагментации для домена $\mathfrak{D}_H = \mathbb{Z}_{\geq 0}$. Колоночный хеш-индекс позволяет использовать один колоночный индекс для индексирования нескольких атрибутов одного отношения.

***Декомпозиция операции естественного соединения с использованием распределенного колоночного хеш-индекса.*** Пусть заданы два отношения:

$$R(A,B_1,\ldots,B_u,C_1,\ldots,C_v) \text{ и } S(A,B_1,\ldots,B_u,D_1,\ldots,D_w).$$

Пусть имеются два колоночных хеш-индекса $I_{R,h}$ и $I_{S,h}$ для атрибутов $B_1,\ldots,B_u$ отношений $R$ и $S$, построенные с помощью одной и той же инъективной хеш-функции $h: \mathfrak{D}_{B_1} \times \ldots \times \mathfrak{D}_{B_u} \to \mathbb{Z}_{\geq 0}$. Пусть для этих индексов задана доменно-интервальная фрагментация степени $k$: $I_{R,h} = \bigcup_{i=0}^{k-1} I^i_{R,h}$; $I_{S,h} = \bigcup_{i=0}^{k-1} I^i_{S,h}$. Положим

$P^i = \pi_{I^i_{R,h}.A \to A_R,\, I^i_{S,h}.A \to A_S}\left(I^i_{R,h} \underset{(I^i_{R,h}.H = I^i_{S,h}.H)}{\bowtie} I^i_{S,h}\right)$ для всех $i = 0,\ldots,k-1$. Определим $P = \bigcup_{i=0}^{k-1} P^i$. Построим отношение $Q(B_1,\ldots,B_u,C_1,\ldots,C_v,D_1,\ldots,D_w)$ следующим образом:

$$Q = \{(\&_R(p.A_R).B_1,\ldots,\&_R(p.A_R).B_u,\ \&_B(p.A_B).C_1,\ldots,\&_B(p.A_B).C_v,\ \&_S(p.A_S).D_1,\ldots,\&_S(p.A_S).D_w) \mid p \in P\}.$$

Тогда $Q = \pi_{*\setminus A}(R) \bowtie \pi_{*\setminus A}(S)$.

***Декомпозиция операции пересечения с использованием распределенного колоночного хеш-индекса*** вида $\pi_{B_1,\ldots,B_u}(R) \cap \pi_{B_1,\ldots,B_u}(S)$. Пусть заданы два отношения $R(A,B_1,\ldots,B_u)$ и $S(A,B_1,\ldots,B_u)$, имеющие одинаковый набор атрибутов. Пусть имеются два колоночных хеш-индекса $I_{R,h}$ и $I_{S,h}$ для атрибутов $B_1,\ldots,B_u$ отношений *R* и *S*, построенные с помощью одной и той же инъективной хеш-функции $h: \mathfrak{D}_{B_1} \times \ldots \times \mathfrak{D}_{B_u} \to \mathbb{Z}_{\geq 0}$. Пусть для этих индексов задана доменно-интервальная фрагментация степени $k$: $I_{R,h} = \bigcup_{i=0}^{k-1} I^i_{R,h}$; $I_{S,h} = \bigcup_{i=0}^{k-1} I^i_{S,h}$. Положим $P^i = \pi_{I^i_{R,h}.A \to A_R,\, I^i_{S,h}.A \to A_S}\left(I^i_{R,h} \underset{(I^i_{R,h}.H = I^i_{S,h}.H)}{\bowtie} I^i_{S,h}\right)$ для всех $i = 0,\ldots,k-1$. Определим $P = \bigcup_{i=0}^{k-1} P^i$. Положим $Q = \{\&_R(p.A_R) \mid p \in P\}$. Тогда

$$\pi_{B_1,\ldots,B_u}(Q) = \pi_{B_1,\ldots,B_u}(R) \cap \pi_{B_1,\ldots,B_u}(S).$$

Аналогичным образом в диссертации выполнена декомпозиция операции *объединения* с использованием фрагментированных колоночных хеш-индексов. Для всех методов декомпозиции, приведенных во второй главе, даны математические доказательства их корректности.

**В третьей главе, «Колоночный сопроцессор КСОП»,** описывается процесс проектирования и реализации программной системы «Колоночный сопроцессор КСОП» для кластерных вычислительных систем, представляющей собой резидентную параллельную программу, взаимодействующую с реляционной СУБД. Колоночный сопроцессор КСОП — это программная система, предназначенная для управления распределенными колоночными индексами, размещенными в оперативной памяти кластерной вычислительной системы. Назначение КСОП — вычислять таблицы предварительных вычислений (ТПВ) для ресурсоемких реляционных операций по запросу СУБД. Общая схема взаимодействия СУБД и КСОП изображена на рис. 1. КСОП включает в себя программу «*Координатор*», запускаемую на узле вычислительного кластера с номером 0, и программу «*Исполнитель*», запускаемую на всех остальных узлах, выделенных для работы КСОП. На SQL-сервере устанавливается специальная программа «Драйвер КСОП», обеспечивающая взаимодействие с координатором КСОП по протоколу TCP/IP. КСОП поддерживает следующие основные операции, доступные СУБД через интерфейс драйвера КСОП: CreateColumnIndex (создание распределенного колоночного индекса), Execute (выполнение запроса на вычисление ТПВ), Insert (добавление в колоночный индекс нового кортежа), TransitiveInsert (добавление в колоночный индекс нового кортежа по транзитивному значению), Delete (удаление из колоночного индекса кортежа), TransitiveDelete (удаление из колоночного индекса кортежа по транзитивному значению).

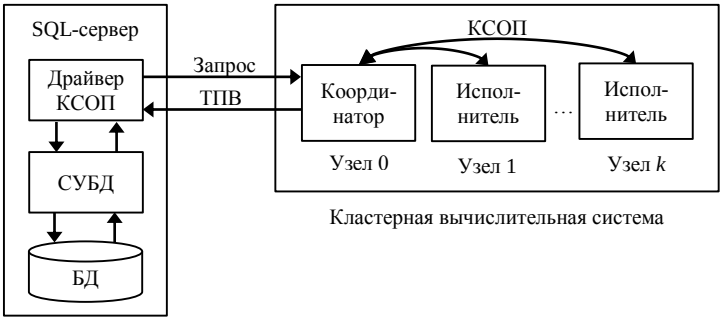



**Рис. 1.** Взаимодействие SQL-сервера с КСОП.

Для организации взаимодействия между драйвером и КСОП в ходе выполнения диссертационного исследования был разработан язык CCOPQL (CCOP Query Language), базирующийся на формате данных JSON.

Поясним общую логику работы КСОП на простом примере. Пусть имеется база данных из двух отношений *R*(*A,B,D*) и *S*(*A,B,C*), хранящихся на SQL-сервере (см. рис. 2). Пусть нам необходимо выполнить запрос:

```
SELECT D, C
FROM R, S
WHERE R.B = S.B AND C<13.
```

Предположим, что КСОП имеет только два узла-исполнителя и на каждом узле имеется три процессорных ядра (процессорные ядра на рис. 2 промаркированы обозначениями $P_{11},\ldots, P_{23}$). Положим, что атрибуты *R.B* и *S.B* определены на домене целых чисел из интервала [0; 120). Сегментные интервалы для *R.B* и *S.B* определим следующим образом: [0; 20), [20; 40), [40; 60), [60; 80), [80; 100), [100; 120). В качестве фрагментных интервалов для *R.B* и *S.B* зафиксируем: [0; 59] и [60,119]. Пусть атрибут *S.C* определен на домене целых чисел из интервала [0; 25]. Изначально администратор базы данных с помощью драйвера КСОП создает для атрибутов *R.B* и *S.B* распределенные колоночные индексы $I_{R.B}$ и $I_{S.B}$. Затем для атрибута *S.C* создается распределенный колоночный индекс $I^B_{S.C}$, который фрагментируется и сегментируется транзитивно относительно индекса $I_{S.B}$. Распределенные колоночные индексы $I_{R.B}$, $I_{S.B}$ и $I^B_{S.C}$ сохраняются в оперативной памяти узлов-исполнителей. Таким образом мы получаем распределение данных внутри КСОП, приведенное на рис. 2. При поступлении SQL-запроса он преобразуется драйвером КСОП в план, определяемый следующим выражением реляционной алгебры:

$$\pi_{I_{R.B}.A\to A_R, I_{S.B}.A\to A_S}\left(I_{R.B}\underset{\boxed{\substack{I_{R.B}.B=\\ I_{S.B}.B}}}{\bowtie}\left(I_{S.B}\bowtie\sigma_{C<13}\left(I^B_{S.C}\right)\right)\right).$$

При выполнении драйвером операции Execute указанный запрос передается координатору КСОП в виде оператора CCOPQL в формате JSON. Запрос выполняется независимо процессорными ядрами узлов-исполнителей над соответствующими группами сегментов. При этом за счет доменной фрагментации и сегментации не требуются обмены данными как между узлами-исполнителями, так и между процессорными ядрами одного узла. Каждое процессорное ядро вычисляет свою часть ТПВ, которая пересылается на узел-координатор. Координатор объединяет фрагменты ТПВ в единую таблицу и пересылает ее драйверу,

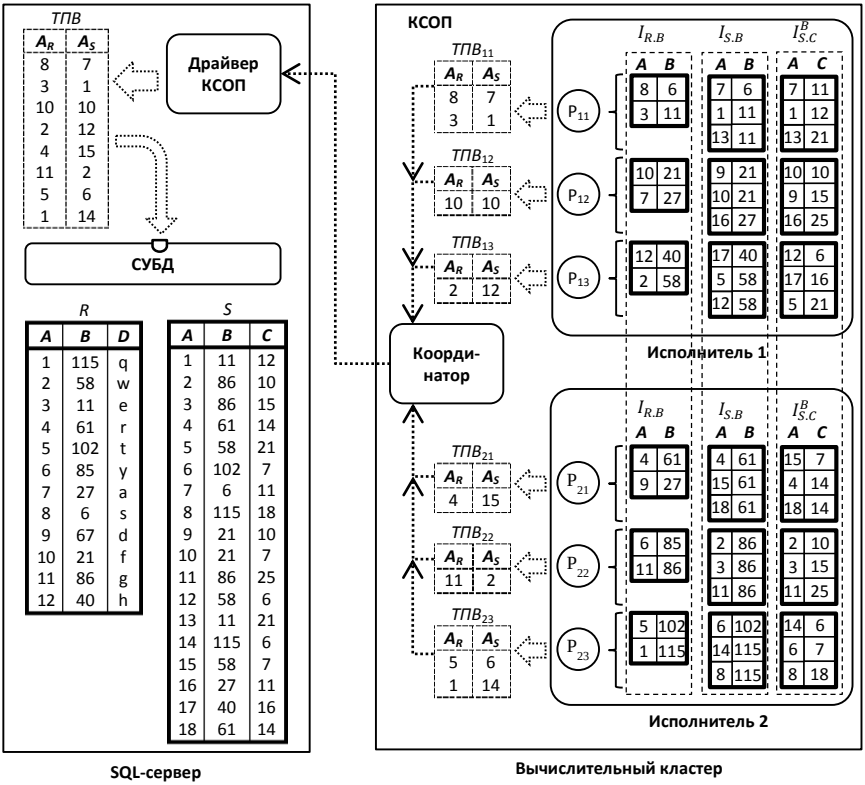



**Рис. 2.** Вычисление ТПВ с использованием КСОП.

который выполняет материализацию этой таблицы в виде отношения в базе данных, хранящейся на SQL-сервере. После этого SQL-сервер вместо исходного SQL-оператора, приведенного на стр. 10, выполняет следующий оператор:

```
SELECT D, C
FROM
    R INNER JOIN (
        ТПВ INNER JOIN S ON (S.A = ТПВ.AS)
    ) ON (R.A = ТПВ.AR).
```

При этом используются обычные кластеризованные индексы в виде B-деревьев, заранее построенные для атрибутов *R.A* и *S.A*.

Колоночный сопроцессор КСОП был реализован на языке Си с использованием аппаратно независимых параллельных технологий MPI и OpenMP. Он может работать как на ЦПУ Intel Xeon X5680, так и на сопроцессоре Intel Xeon Phi. Объем исходного кода составил около двух с половиной тысяч строк. Исходные тексты КСОП свободно доступны в сети Интернет по адресу: https://github.com/elena-ivanova/colomnindices/.

**В четвертой главе, «Вычислительные эксперименты»,** приводятся результаты вычислительных экспериментов по исследованию эффективности разработанных в диссертации моделей, методов и алгоритмов обработки сверхбольших баз данных с использованием распределенных колоночных индексов.

Эксперименты проводились на двух вычислительных комплексах с кластерной архитектурой: «Торнадо ЮУрГУ» и «RSC PetaStream» МСЦ РАН. Система «*Торнадо ЮУрГУ*» включает в себя 384 процессорных узлов, соединенных сетями InfiniBand QDR и Gigabit Ethernet. В состав процессорного узла входит два шестиядерных ЦПУ Intel Xeon X5680, ОЗУ 24 Гб и сопроцессор Intel Xeon Phi SE10X (61 ядро по 1.1 ГГц), соединенные шиной PCI Express. Система «*RSC PetaStream*» состоит из 8 модулей, включающих в себя 8 сопроцессоров Intel Xeon Phi 7120, каждый из которых имеет 61 процессорное ядро и 16 Гб встроенной памяти GDDR5. Модули соединены между собой сетями InfiniBand FDR и Gigabit Ethernet. Непосредственно на каждом сопроцессоре (на одном ядре) загружается операционная система Linux CentOS 7.0.

Для тестирования КСОП использовалась синтетическая база данных, построенная на основе эталонного теста TPC-H. Тестовая база данных состояла из двух таблиц: ORDERS (ЗАКАЗЫ) и CUSTOMER (КЛИЕНТЫ). Для имитирования перекоса данных использовались следующие распределения значения атрибута ORDERS.ID_CUSTOMER (внешний ключ, определяющий идентификатор клиента, сделавшего заказ): равномерное (uniform), «45-20», «65-20», «80-20». Для варьирования размера результирующего отношения использовался *коэффициент селективности Sel*, принимающий значение из интервала ]0;1]. Коэффициент Sel определяет размер (в кортежах) результирующего отношения относительно размера отношения ORDERS. Для масштабирования базы данных использовался *масштабный коэффициент SF* (*Scale Factor*), значение которого изменялось от 1 до 10. При проведении экспериментов размер отношения ORDERS составлял SF×63 000 000 кортежей, размер отношения CUSTOMER — SF×630 000 кортежей.

В первом эксперименте исследовалась балансировка загрузки процессорных ядер Xeon Phi при различных перекосах в распределении значений внешнего ключа ORDERS.ID_CUSTOMER. Результаты эксперимента представлены на рис. 3. Из графиков видно, что при малом количестве сегментов сильный перекос по данным приводит к существенному дисбалансу в загрузке процессорных ядер. В случае, когда количество сегментов равно 60 и совпадает с количеством ядер, время выполнения операции при распределении «80-20» более чем в четыре раза превышает время выполнения той же операции при равномерном (uniform) распределении. Однако при увеличении количества сегментов влияние перекоса по данным нивелируется. Для распределения «45-20» оптимальным оказывается число сегментов, равное 10 000, для распределения «65-20» — 20 000, и для распределения «80-20» — 200 000.

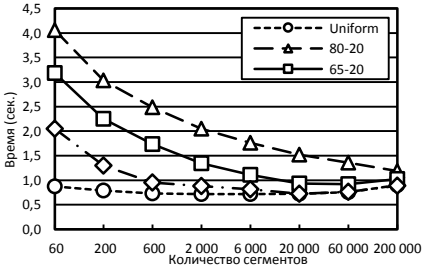



**Рис. 3.** Балансировка загрузки процессорных ядер сопроцессора Xeon Phi.

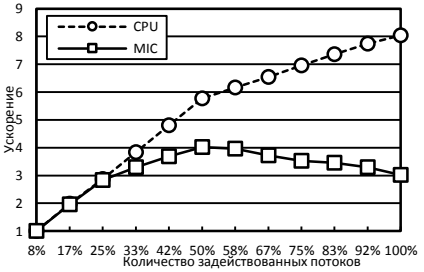



**Рис. 4.** Влияние гиперпоточности на ускорение.

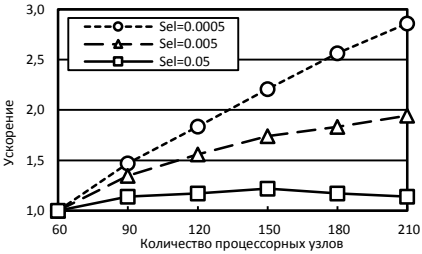



**Рис. 5.** Ускорение вычисления ТПВ на кластере «Торнадо ЮУрГУ» при SF = 10.

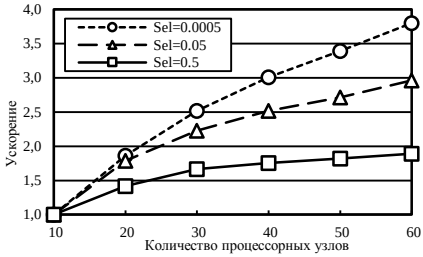



**Рис. 6.** Ускорение вычисления ТПВ на кластере «RSC PetaStream» при SF = 1.

Целью второго эксперимента было определить, насколько эффективно гиперпоточность может быть применена при работе колоночного сопроцессора КСОП. ЦПУ Intel Xeon X5680 имеет аппаратную поддержку двух потоков на ядро, сопроцессор Intel Xeon Phi поддерживает четыре потока на ядро. Результаты эксперимента представлены на рис. 4. Эксперимент показал, что для CPU производительность растет вплоть до максимального числа аппаратно поддерживаемых потоков. Однако, при использовании одного потока на ядро на CPU наблюдается ускорение, близкое к идеальному, а при использовании двух потоков на ядро прирост ускорения становится более медленным. Применительно к MIC картина меняется. При использовании одного потока на ядро на MIC наблюдается ускорение, близкое к идеальному. При использовании двух потоков на ядро прирост ускорения становится более медленным. Использование же большего количества потоков на одно ядро ведет к деградации производительности.

В третьем и четвертом экспериментах была исследована масштабируемость КСОП при работе на вычислительных системах с массовым параллелизмом. На рис. 5 приведены кривые ускорения при вычислении колоночным сопроцессором ТПВ на кластере «Торнадо ЮУрГУ», на рис. 6 — на кластере «RSC PetaStream». Графики на рис. 5 показывают, что селективность запроса *Sel* оказывается фактором, ограничивающим масштабируемость. Так, для *Sel* = 0.0005 кривая ускорения становится практически линейной, а для *Sel* = 0.005 приближается к линейной. Однако, при большом значении коэффициента селективности *Sel* = 0.05 масштабируемость ограничивается 150 процессорными узлами. Причина в том, что при *Sel* = 0.05 время передачи, распаковки и слияния ТПВ меняется мало и существенно превышает время ее вычисления, в то время как при *Sel* = 0.0005 значение времени передачи, распаковки и слияния ТПВ почти на порядок меньше времени ее вычисления. В соответствии с этим можно сделать вывод, что при малой селективности запроса КСОП демонстрирует на больших базах данных ускорение, близкое к линейному.

Аналогичная картина наблюдалась при тестировании КСОП на вычислительном кластере «RSC PetaStream» (рис. 6).

В следующем эксперименте было исследовано в какой мере использование колоночного сопроцессора КСОП может ускорить выполнение запроса класса OLAP в реляционной СУБД. В ходе выполнения эксперимента исследовались следующие три конфигурации:

1) PostgreSQL: выполнение запроса без создания индексных файлов в виде B-деревьев;
2) PostgreSQL & B-Trees: выполнение запроса с предварительным созданием индексных файлов в виде B-деревьев для атрибутов соединения;
3) PostgreSQL & CCOP: выполнение запроса с использованием ТПВ и предварительным созданием индексных файлов в виде B-деревьев для суррогатных ключей.

В последнем случае ко времени выполнения запроса добавлялось время создания ТПВ колоночным сопроцессором КСОП (CCOP). В каждом случае замерялось время первого и повторного запуска запроса. Это связано с тем, что после первого выполнения запроса PostgreSQL собирает статистическую информацию, сохраняемую в словаре базы данных, которая затем используется для оптимизации плана выполнения запроса. Эксперименты показали (см. табл. 1), что при отсутствии индексов в виде B-деревьев использование колоночного сопроцессора позволяет увеличить производительность выполнения запроса в 100 – 150 раз для коэффициента селективности Sel = 0.0005. Однако при больших значениях коэффициента селективности эффективность использования КСОП может снижаться вплоть до отрицательных значений (ускорение меньше единицы).

**Табл. 1.** Время вычисления SQL-запроса и ускорение в сравнении с PostgreSQL при SF = 1.

| Конфигурация | Время в минутах | | | | | |
|---|---|---|---|---|---|---|
| | Sel = 0.0005 | | Sel = 0.005 | | Sel = 0.05 | |
| | 1-й запуск | 2-й запуск | 1-й запуск | 2-й запуск | 1-й запуск | 2-й запуск |
| PostgreSQL | 7.3 | 1.21 | 7.6 | 1.29 | 7.6 | 1.57 |
| PostgreSQL & B-Trees | 2.62 | 2.34 | 2.83 | 2.51 | 2.83 | 2.63 |
| PostgreSQL & CCOP | 0.073 | 0.008 | 0.65 | 0.05 | 2.03 | 1.72 |
| Ускорение | | | | | | |
| $\frac{t_{PostgreSQL}}{t_{PostgreSQL\ \&\ CCOP}}$ | 100 | 151 | 12 | 27 | 4 | 0.9 |
| $\frac{t_{PostgreSQL\&B-Trees}}{t_{PostgreSQL\ \&\ CCOP}}$ | 36 | 293 | 4 | 50 | 1.4 | 1.53 |

**В заключении** в краткой форме излагаются итоги выполненного диссертационного исследования, представляются отличия диссертационной работы от ранее выполненных родственных работ других авторов, даются рекомендации по использованию полученных результатов и рассматриваются перспективы дальнейшего развития темы.

## Основные результаты диссертационной работы

На защиту выносятся следующие новые научные результаты.

1. Разработана доменно-колоночная модель представления данных, на базе которой выполнена декомпозиция основных реляционных операций с помощью распределенных колоночных индексов.
2. Разработаны высокомасштабируемые параллельные алгоритмы выполнения основных реляционных операций, использующие распределенные колоночные индексы.
3. Выполнена реализация колоночного сопроцессора для кластерных вычислительных систем.
4. Проведены вычислительные эксперименты, подтверждающие эффективность предложенных подходов.

## Публикации по теме диссертации

*Статьи в журналах из перечня ВАК*

1. *Иванова Е.В. Соколинский Л.Б.* Колоночный сопроцессор баз данных для кластерных вычислительных систем // Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. 2015. Т. 4, № 4. С. 5–31.

2. *Иванова Е.В. Соколинский Л.Б.* Использование сопроцессоров Intel Xeon Phi для выполнения естественного соединения над сжатыми данными // Вычислительные методы и программирование: Новые вычислительные технологии. 2015. Т. 16, Вып. 4. С. 534–542.

3. *Иванова Е.В. Соколинский Л.Б.* Параллельная декомпозиция реляционных операций на основе распределенных колоночных индексов // Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. 2015. Т. 4, № 4. С. 80–100.

*Статья в издании, индексируемом в SCOPUS и Web of Science*

4. *Ivanova E., Sokolinsky L.* Decomposition of Natural Join Based on Domain-Interval Fragmented Column Indices // Proceedings of the 38th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO. IEEE, 2015. P. 223–226.

*Статьи в изданиях, индексируемых в РИНЦ*

5. *Иванова Е.В., Соколинский Л.Б.* Использование сопроцессоров Intel Xeon Phi для выполнения естественного соединения над сжатыми данными // Суперкомпьютерные дни в России: Труды международной конференции (28–29 сентября 2015 г., Москва). М.: Изд-во МГУ, 2015. С. 190–198.

6. *Иванова Е.В. Соколинский Л.Б.* Использование распределенных колоночных индексов для выполнения запросов к сверхбольшим базам данных // Параллельные вычислительные технологии (ПаВТ'2014): труды международной научной конференции (1–3 апреля 2014 г., Ростов-на-Дону). Челябинск: Издательский центр ЮУрГУ, 2014. С. 270–275.

7. *Иванова Е.В., Соколинский Л.Б.* Декомпозиция операции группировки на базе распределенных колоночных индексов // Наука ЮУрГУ. Челябинск: Издательский центр ЮУрГУ, 2015. С. 15–23.

8. *Иванова Е.В. Соколинский Л.Б.* Декомпозиция операций пересечения и соединения на основе доменно-интервальной фрагментации колоночных индексов // Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. 2015. Т. 4, № 1. С. 44–56.

9. *Иванова Е.В.* Исследование эффективности использования фрагментированных колоночных индексов при выполнении операции естественного соединения с использованием многоядерных ускорителей // Параллельные вычислительные технологии (ПаВТ'2015): труды международной научной конференции (30 марта – 3 апреля 2015 г., Екатеринбург). Челябинск: Издательский центр ЮУрГУ, 2015. С. 393–398.

10. *Иванова Е.В*. Использование распределенных колоночных хеш-индексов для обработки запросов к сверхбольшим базам данных // Научный сервис в сети Интернет: многообразие суперкомпьютерных миров: Труды Международной суперкомпьютерной конференции (22–27 сентября 2014 г., Новороссийск). М.: Изд-во МГУ, 2014. С. 102–104.

*Свидетельства о регистрации программ и баз данных*

11. *Соколинский Л.Б., Иванова Е.В*. Свидетельство Роспатента об официальной регистрации программы для ЭВМ «Система обработки транзакций с использованием распределенных колоночных индексов» № 2015612158 от 13.02.2015, правообладатель: ФГБОУ ВПО "ЮУрГУ" (НИУ).

Работа выполнялась при финансовой поддержке Минобрнауки РФ в рамках ФЦП «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2014—2020 годы» (Госконтракт № 14.574.21.0035).

Подписано в печать «20» октября 2015 г.
Формат 60x84 1/16. Бумага офсетная.
Печать офсетная. Усл. печ. л. 1,0. Уч.-изд. л. 1,2.
Тираж 100 экз.

Типография «Активист»
454080, г. Челябинск, пр. Ленина, 74б