Развитие методов параллельных вычислений для фрагментации данных сетевой базы данных на основе рангового подхода

Е. Г. Андрианова, В. К. Раев, Д. И. Фильгус

МИРЭА — Российский технологический университет
119571, Москва, пр-т Вернадского, 78

e-mail: andrianova@mirea.ru; vkr3708@gmail.com; ar4ers@ya.ru

Аннотация. Анализ задачи фрагментации данных сетевой базы данных, показывает, что формальной моделью является так называемая задача целочисленного линейного программирования с булевыми переменными (ЦЛП БП), которая относится к классу NP (non-deterministic polynomial — недетерминированные с полиномиальным временем). Доминирующее место в методах решения этих задач в настоящее время занимают комбинаторные методы. К ним в первую очередь можно отнести методы полного перебора, динамического программирования, а также локальные алгоритмы. Практическое применение данных методов затруднено при решении задач большой размерности. Попытки уменьшения времени решения задач ЦЛП с БП за счет распараллеливания сталкиваются с другой проблемой теории параллельных вычислений, которая заключается в том, что с точки зрения параллельных алгоритмов данный тип задач относится к классу сильносвязанных задач и поэтому плохо поддается распараллеливанию. Поэтому при разработке параллельных алгоритмов для решения задачи ЦЛП с БП, кроме противоречия между точностью решения задачи и временем ее решения, возникает еще одно противоречие — между сильной связностью, присущей данной задаче и необходимостью ее распараллеливания. Предлагаемый метод параллельных вычислений для фрагментации данных сетевой базы данных основан на ранговом подходе к ее решению, принципе оптимизации по направлению и стратегий отсечения неперспективных вариантов решений. Метод позволяет определять локальные экстремумы в вершинах графа и на их основе определять глобальный экстремум.
Ключевые слова: данные, параллельные вычисления, ранговый подход, фрагментация, сетевая база данных.

1. Введение

Вычислительные комплексы (ВК), которые находятся в эксплуатации, непрерывно развиваются. Наступает время, когда в силу увеличения объемов хранимой информации рост интенсивности решения существующих задач, производительность системы падает. Одним из решений данных проблем может быть замена элементов ВК на более современные и производительные. Поскольку время выполнения за-

просов к ВК в значительной мере определяется количеством операций дискового доступа [12], то повышение производительности можно достичь использованием более быстродействующих дисков или нескольких дисковых устройств [13] для оптимального распределения между ними дисковых операций. Также можно увеличить размер оперативной памяти для хранения большого количества информации [14] и таким образом разгрузить дисковые устройства. Другой путь — применение многопроцессорных вычислительных систем, позволяющих оптимизировать выполнение операций, предусмотренных в запросе [4, 5]. Однако это, как правило, связано с большими затратами. Покупка оборудования далеко не всегда решает проблемы производительности, поскольку «узкие места» могут находиться на другом архитектурном слое. Также часто приобритение дополнительного оборудования экономически невыгодно: избыток мощностей может привести к снижению рентабельности [17].

Анализ [7–9] показывает, что каждая из технологий повышения производительности (ТПП) требует анализа структуры сетевой базы данных, потока решаемых задач (запросов к ВК), а в ряде случаев детального анализа использования отдельных фрагментов СБД (отдельных таблиц, их полей, групп таблиц). Так, например, управление индексной структурой требует определения интенсивности использования одной таблицы и полей. Реструктуризация предусматривает изменение структуры одной или нескольких таблиц. Материализованные представления имеют в виду детальный анализ последовательности запросов и введение специальных таблиц для микропроцессоров (МП).

В статье разрабатывается метод параллельных вычислений для фрагментации данных сетевой базы данных, который основан на ранговом подходе к ее решению, принципе оптимизации по направлению и стратегий отсечения неперспективных вариантов решений.

2. Постановка задачи распределения фрагментов данных сетевой базы данных

Рассмотрим модель размещения фрагментов сетевой базы данных в вычислительном комплексе. В качестве критерия оптимальности определим средний объем пересылаемых по линиям связи данных при выполнении запроса. Рассматривается n — число узлов сети с произвольной структурой; m — число независимых фрагментов сетевой базы даных; K_j — j-й узел сети, $j=\overline{1,n}$; F_i — i-й фрагмент сетевой базы данных, $i=\overline{1,m}$; L_i — объем i-го фрагмента; b_j — объем памяти узла K_j, предназначенной для размещения фрагментов; s — число классов запросов (например, чтение, добавление, обновление, удаление записей базы данных); λ_{ij}^k — интенсивность запросов k-го класса $(k=\overline{1,s})$ до фрагмента F_i, инициированных в

узле K_j; α_{ij}^k — объем запроса k-го класса $(k=\overline{1,s})$ до фрагмента F_i, инициированного в узле K_j; β_{ij}^k — объем запрашиваемых данных при выполнении запроса k-го класса $(k=\overline{1,s})$ до фрагмента F_i, поступившего в узел K_j, p — число таблиц сетевой базы данных; M_z — z-тая таблица РБД, $z=\overline{1,p}$; m_z — число строк таблицы M_z; n_z — число столбцов таблицы M_z; d_{ij}^z — ячейка таблицы M_Z в позиции (строка i, столбец j), $i=\overline{1,m_z}$, $j=\overline{1,n_z}$; w_{ij}^z — объем памяти, занимаемый ячейкой d_{ij}^z. Схема фрагментации сетевой базы данных представлена на рис. 1.

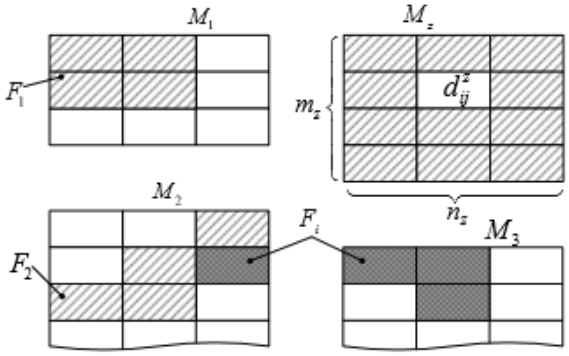

Рисунок 1. Схема фрагментации данных

Объем пересылаемых данных при выполнении запроса k-го класса до фрагмента F_1, инициированного в узле K_j, определяется следующим образом $(\alpha_{ij}^k+\beta_{ij}^k)(1-x_{ij})$. При этом x_{ij}, $i=\overline{1,m}$; $j=\overline{1,n}$ — значения, которые определяются следующим образом:

$$\alpha_{ij}=\begin{cases}1, \text{ если фрагмент } F_i \text{ находится в узле } K_j,\\ 0, \text{ в противном случае}\end{cases}. \quad (1)$$

Поскольку интенсивность λ_{ij}^k порождает объем данных $\lambda_{ij}^k(\alpha_{ij}^k+\beta_{ij}^k)(1-x_{ij})$, нуждающихся в пересылке, то общий объем данных, которые требуется переслать по каналам связи между узлами вследствие функционирования распределенной системы в течение единицы времени, определяется формулой

$$S=\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{s}\lambda_{ij}^k(\alpha_{ij}^k+\beta_{ij}^k)(1-x_{ij}). \quad (2)$$

Если положить, что $\lambda=\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{s}\lambda_{ij}^k$, то целевая функция задачи оптимального распределения фрагментов по узлам вычислительного комплекса примет вид:

$$V = \frac{1}{\lambda}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{s}\lambda_{ij}^{k}(\alpha_{ij}^{k} + \beta_{ij}^{k})(1 - x_{ij}). \quad (3)$$

Чем меньше значение V, тем выше скорость обслуживания запросов. Все сообщения, поступающие во входные очереди узлов, разделим на два типа: тип 1 — сообщения, составляющие запросы, для обработки которых необходимые фрагменты не содержатся в базе данных узла вычислительного комплекса, и ответы на эти запросы; тип 2 — сообщения, составляющие запросы, для обслуживания которых нужные фрагменты содержатся в БД соответствующего узла вычислительного комплекса. При этом запрос типа 1, прибывший для своего обслуживания в отдаленный узел вычислительного комплекса, превращается в запрос типа 2.

Поскольку каждый фрагмент F_i, $i = \overline{1, m}$ должен находиться, по крайней мере, в одном из узлов вычислительного комплекса, то справедливо условие

$$\sum_{j=1}^{n} x_{ij} \geq 1,\ i = \overline{1,m}. \quad (4)$$

Кроме этого, объем локальной БД каждого узла K_j, $j = \overline{1,n}$ не должен превышать объем памяти этого узла, предназначенный для размещения фрагментов:

$$\sum_{i=1}^{m} L_i x_{ij} \leq b_j,\ j = \overline{1,n}. \quad (5)$$

Запишем ограничения, связанные с распределением ячеек по фрагментам:

$$y_{ijz}^{\sigma} = \begin{cases} 1, \text{ если ячейка } d_{ijz} \text{ принадлежит фрагменту } F_{\sigma}, \\ 0, \text{ в противном случае.} \end{cases} \quad (6)$$

Суммарный объем всех фрагментов должен быть равен суммарному объему всех таблиц сетевой базы данных, т. е.

$$\sum_{i=1}^{m} L_i = \sum_{z=1}^{p}\sum_{i=1}^{m_z}\sum_{i=1}^{n_z} w_{ij}^{z}. \quad (7)$$

При этом объем фрагмента равен

$$L_{\sigma} = \sum_{z=1}^{p}\sum_{i=1}^{m_z}\sum_{i=1}^{n_z} w_{ij}^{z} y_{ijz}^{\sigma}. \quad (8)$$

Таким образом, задача размещения фрагментов данных сетевой базы данных по узлам вычислительного комплекса состоит в том, чтобы определить значения переменных x_{ij}, где $x_{ij} = \{0,1\}$, $i = \overline{1,m}$; $j = \overline{1,n}$, которые удовлетворяют условиям (4), (5) и дают минимум линейной функции (3). Полученная математическая модель предназначена решению задачи целочисленного линейного программирования с булевыми переменными.

К сожалению задача ЦЛП с БП относится к классу NP (non-deterministic polynomial — недетерминированные с полиномиальным временем), которые с трудом поддаются решению даже при использовании современных ЭВМ [10, 11].

Доминирующее место в методах решения этих задач в настоящее время занимают комбинаторные методы. К ним в первую очередь можно отнести методы полного перебора, динамического программирования, а также локальные алгоритмы. Практическое применение данных методов затруднено при решении задач большой размерности.

Попытки уменьшения времени решения задач ЦЛП с БП за счет распараллеливания сталкиваются с другой проблемой теории параллельных вычислений, которая заключается в том, что с точки зрения параллельных алгоритмов данный тип задач относится к классу сильносвязанных задач и поэтому плохо поддается распараллеливанию. Поэтому при разработке параллельных алгоритмов для решения задачи ЦЛП с БП, кроме противоречия между точностью решения задачи и временем ее решения, возникает еще одно противоречие — между сильной связностью, присущей данной задаче и необходимостью ее распараллеливания.

3. Разработка метода параллельных вычислений для решения задачи фрагментации данных сетевой базы данных на основе рангового подхода

Метод размещения фрагментов данных сетевой базы данных на основе рангового подхода состоит из следующих этапов:

- разработка последовательного способа решения задачи распределения фрагментов сетевой базы данных;
- разработка алгоритма параллельных вычислений решения задачи распределения фрагментов сетевой базы данных;
- разработка архитектуры параллельной вычислительной структуры систолического типа для решения задачи распределения фрагментов сетевой базы данных.

3.1. Способ решения задачи распределения фрагментов сетевой базы данных

Размещение фрагментов данных сетевой базы данных вычислительного комплекса будем рассматривать на основе рангового подхода к решению задач дискретной оптимизации, которая определяет представление n-мерного единичного куба B_n в виде графа $G\Delta$. На рис. 2 изображен граф $G\Delta$ для $n = 4$.

Геометрически вершина k графа $G\Delta$ ранга r — это множество векторов $\vec{x}(x_1, x_2, ..., x_k, ..., x_n)$, у которых $x_k = 1$, а на позициях от 1 до k находится r единиц (рис. 3). Ребру, входящему в вершину k графа $G\Delta$, соответствует единичный вектор $\vec{e}_k(0, 0, ..., 0, 1, 0, ..., 0)$ в n-мерном единичном кубе B_n с единицей в k-й позиции.

Тогда пути μ_{sj}^r ранга r в графе $G\Delta$ соответствует вектор $\vec{x}$, равный сумме единичных векторов ребер, по которым он достиг вершину j ранга r, начиная с вершины s.

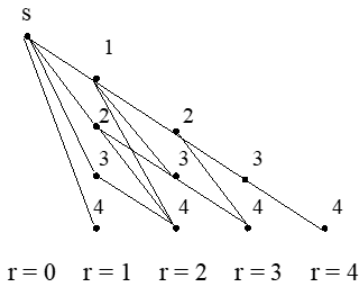

Рисунок 2. Граф $G\Delta$

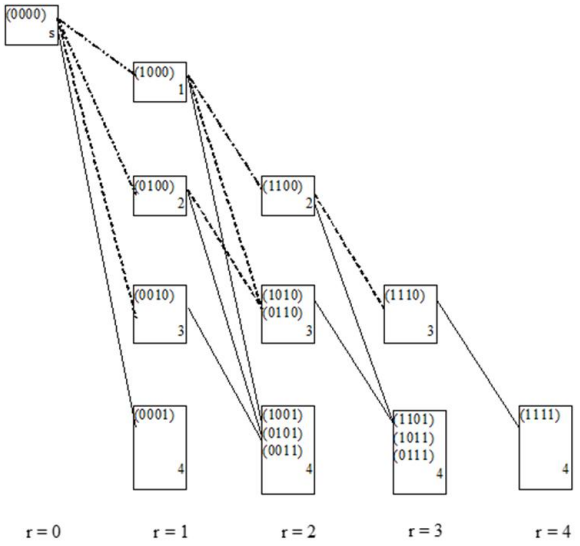

Рисунок 3. Геометрическая интерпретация графа $G\Delta$

Формальной моделью является m-мерная задача 0,1-рюкзак, где m — количество векторов ограничений, которая в общем случае может быть представлена в следующем виде:

$$\sum_{j=1}^{n} c_j x_j \to \min \tag{9}$$

при ограничениях

$$\sum_{j=1}^{n} \alpha_{ij} x_j \le b_i. \tag{10}$$

Пусть в графе $G\Delta$ каждому ребру, входящему в вершину j, $j = \overline{(1, n)}$ соответствует $m+1$ вес: вес c_j, равный коэффициенту при x_j в функционале (9), и m ве-

сов a_{ij}, равных коэффициентам при x_{ij} в ограничениях (10). Тогда, путь μ_{sj}^r в графе $G\Delta$ из вершины s в вершину j характеризуется $m+1$ длинами: $d_c(\mu_{sj}^r)$ — длиной по весам функционала и $d_a(\mu_{sj}^r)_1$ — длинами по весам ограничений. Множество путей $m_s^r(j)$ в графе $G\Delta$ к вершинам j, расположенным на ярусах $r=(\overline{1,n})$ от вершины s, можно представить в виде

$$m_s^r(j)=m_{sj}^{r=1}\cup m_{sj}^{r=2}\cup\ldots\cup m_{sj}^{r=n},\ j=(\overline{1,n}), \tag{11}$$

где m_{sj}^r — множество путей в графе $G\Delta$ от вершины s к вершинам j, расположенным на r-х ярусах графа $G\Delta$ (ранг пути $\mu_{sj}^r\in m_{sj}^r$ определяется числом ребер, образующих этот путь). Следует иметь в виду, что множество путей $m_{sj}^{r=k}$ в графе $G\Delta$ соответствует множеству векторов $\{\vec{x_1}, \vec{x_1}, \ldots, \vec{x_J}\}$, содержащих k единиц. Следовательно, $|m_{sj}^r|=C_n^{r=k}$, т. е. каждому пути в множестве $m_{sj}^{r=k}$ соответствует некоторый вектор $(x_1, x_2, \ldots, x_n)$. Из (11) следует

$$m_s^r(j)=C_n^{r=1}+C_n^{r=2}+\ldots+C_n^{r=n}=2^n-1. \tag{12}$$

Таким образом, граф $G\Delta$ представляет собой упорядоченный по рангам эквивалент n-мерного единичного куба B_n, в котором пути $\mu_{sj}^r\in m_{sj}^r$ соответствуют вершинам B_n (12). Длина каждого пути по весу функционала определяет значение функционала (9) в вершинах единичного куба B_n Длина по весам ограничений определяет, соответствует ли данная вершина B_n ограничениям (10), т. е. принадлежит вершина n-мерного единичного куба B_n соответствующей гиперплоскости. Если $d_a(\mu_{sj}^r)\le b$, то вершина принадлежит соответствующей гиперплоскости (10) и будем говорить, что путь μ_{sj}^r удовлетворяет свойству ν. Если $d_a(\mu_{sj}^r)>b$, то вершина n-мерного куба, соответствующая пути μ_{sj}^r, не принадлежит ни одной гиперплоскости (10), а путь μ_{sj}^r считаем не удовлетворяющим свойству ν.

Решению задачи (9)–(10) в $G\Delta$ соответствует самый длинный путь по весам функционала, удовлетворяющий свойству ν.

В основе математической модели рангового подхода для построения алгоритмов решения задач ЦЛП с БП положен принцип оптимизации по направлению в дискретном пространстве состояний, заданном графом $G\Delta$. Представление n-мерного единичного куба в виде графа $G\Delta$ позволяет разбить множество всех путей графа $G\Delta$ из нулевой вершины s на Ω локальных областей, где $|\Omega|$ не превышает величину $n^2/2$, поскольку число вершин в графе $G\Delta$ определяется суммой чисел натурального ряда

$$\Omega=n+(n-1)+\ldots+1=n(n+1)/2\approx n^2/2, \tag{13}$$

причем Ω-области в графе $G\Delta$ упорядочены по рангам и пути следующего ранга могут быть получены на основе путей предыдущего ранга за счет подсоединения к ним ребра (j, p) в графе $G\Delta$

$$m_{sp}^{r=r+1} = \{(\forall(\mu_{sj}^{r} \in m_{sj}^{r})) \bigcup (j, p)\}. \tag{14}$$

Пусть заданы некоторые правила отсечений путей μ_{sj}^{r} в множествах m_{sj}^{r}:

$$\text{L1: } \mu_{sp}^{r=r+1} = \max_{c_j} \mu_{sj}^{r} \bigcup (j, p)\}; \; p = \overline{1,n}; \; j = \overline{r,n}; \; j \neq p. \tag{15}$$

$$\text{L2: } \mu_{sp}^{r=r+1} = \min_{\alpha_j} \mu_{sj}^{r} \bigcup (j, p)\}; \; p = \overline{1,n}; \; j = \overline{r,n}; \; j \neq p. \tag{16}$$

$$\text{L3: } \begin{cases} \mu_{sp}^{r=r+1} = \max_{c_j} \{\mu_{sj}^{r} \bigcup (j, p)\}, \\ \mu_{sp}^{r=r+1} = \min_{\alpha_j} \{\mu_{sj}^{r} \bigcup (j, p)\}. \end{cases} \tag{17}$$

В соответствии с правилом (15) при построении путей следующего ранга в каждом множестве путей выбирается один максимальный по значению функционала (9), в соответствии с правилом (16) при построении путей следующего ранга в каждом множестве путей выбирается один минимальный по значению ограничения (10), в соответствии с правилом (17) при построении путей следующего ранга в каждом множестве путей выбирается два максимальный по значению функционала (9) и минимальный по значению ограничения (10).

Тогда если в множествах содержатся пути, удовлетворяющие свойству ν и правилам (15)–(17), то под оптимизацией по направлению в графе $G\Delta$ к вершине p будем понимать формирование множеств $m_{sp}^{r=r+1}$ следующего ранга, которые получаются за счет выделения в m_{sj}^{r} путей, подсоединение к которым ребра (j, p) позволит в множестве $m_{sp}^{r=r+1}$ получить пути, удовлетворяющие правилам (15)–(17) на основе следующего рекуррентного соотношения:

$$\forall(\mu_{sj}^{r} \in m_{sj}^{r})[\mu_{sp}^{r=r+1} = L_w\{\mu_{sj}^{r} \bigcup (j, p)\}], \; p = \overline{r+1,n}; \; j = \overline{r,n}, \tag{18}$$

где $\mu_{sj}^{r} \bigcup (j, p)$ — путь из вершины s графа $G\Delta$ в вершину p, проходящий через промежуточную вершину j и удовлетворяющий правилам (15)–(17), т. е. получаемый за счет подсоединения к пути μ_{sj}^{r} ребра (j, p), если такое соединение не противоречит правилам (15)–(17). В дальнейшем для упрощения изложения, если путь $\mu_{sp}^{r=r+1} = \mu_{sj}^{r} \bigcup (j, p)$ удовлетворяет правилам (15)–(17), то будем говорить, что он удовлетворяет и свойству ν.

Введем обобщенную процедуру A_0, позволяющую на основе выбранного правила отсечений (15)–(17) решать задачу (9)–(10).

Шаг 1. Из вершины s строятся множества путей $m_{si}^{r=1}, i = (\overline{1,n})$, удовлетворяющие свойству ν.

Шаг 2. Формируются множества путей m_{sp}^{r+1}, $p=(\overline{r+1,n})$ следующего ранга, удовлетворяющих свойству ν, на базе множеств путей m_{sp}^{r} текущего ранга в соответствии с рекуррентным соотношением (18). В образованных множествах m_{sp}^{r+1} производится отсев путей согласно выбранным правилам отсечения (15)–(17) и выделяются пути, определяющие локальные экстремумы областей Ω_p, $p=(\overline{r+1,n})$.

Шаг 3. Проверяем, все ли множества m_{sp}^{r+1} следующего ранга пустые. Если это так, то переходим к шагу 4, иначе увеличиваем значение текущего ранга $r:=r+1$ и выполняем шаг 2.

Шаг 4. Выделяем среди множеств локальных экстремумов $\{\Omega_i\}$ глобальный, и процедура заканчивает работу.

Данная обобщенная процедура A_0 позволяет определять локальные экстремумы в Ω областях графа $D\Delta$ каждый раз на шаге 2, и затем на шаге 4 выделять глобальный экстремум из $n^2/2$ локальных, полученных на основе принципа оптимизации по направлению в графе $D\Delta$ с использованием вводимых правил отсечения путей (15)–(17) во множествах m_{si}^{r}.

3.2. Алгоритм параллельных вычислений решения задачи распределения фрагментов сетевой базы данных

Параллельным вычислением называется множественное число операций:

$$\Phi=\{\theta_j\},\ j=1,\ ...,\ n, \tag{19}$$

где $\theta_j: S_i^{'}\to S_i^{''}$, $S_i^{'}=(a_i,q_i)$ — локальное состояние, которое называется условием готовности операции $\theta_j: S_i^{''}=(b_i,q_i)$ — локальное состояние, которое называется условием завершения операции θ_j. В результате применения θ_j происходит изменение состояний $(a_i,q_i)\in S_i$ на $(b_i,q_i)\in S_i^{''}$ в соответствии с правилами (15)–(17).

Вычислительным процессом называется последовательность вида $\pi=S_0;\ S_1;\ \theta_{i-1};\ S_2;\ \theta_{i-2};\ ...;\ S_{i-1};\ \theta_1,$ где S_i — глобальные состояния; θ_{i_j} — операции или подмножества операций, которые превращают S_i в S_{i+1} на основе правил (15)–(17).

Вычислительный процесс параллельного алгоритма для случая реализации на n (для $n=5$) процессорных элементах имеет такой вид:

S_0: установление начальных значений.

$$S_1:\theta_1: q_1=S^{'}\{(a_1,q_1)+(a_2,q_2)\}\to S_{L_1}^{''}(b_1,q_1);$$

$$q_2=S^{'}\{(a_2,q_2)+(a_3,q_3)\}\to S_{L_1}^{''}(b_2,q_2);$$

$$q_3=S^{'}\{(a_3,q_3)+(a_4,q_4)\}\to S_{L_1}^{''}(b_3,q_3);$$

$$q_4=S^{'}\{(a_4,q_4)+(a_5,q_5)\}\to S_{L_1}^{''}(b_4,q_4);$$

$$\theta_2: q_1=q^{*};$$

$$q_2 = S'\{(a_1,q_1)+(a_3,q_3)\} \to S''_{L_1}(b_2,q_2);$$

$$q_3 = S'\{(a_2,q_2)+(a_4,q_4)\} \to S''_{L_1}(b_3,q_3);$$

$$q_4 = S'\{(a_3,q_3)+(a_5,q_5)\} \to S''_{L_1}(b_4,q_4).$$

$$\theta_3 : q_1 = q^*;\ q_2 = q^*;$$

$$q_3 = S'\{(a_1,q_1)+(a_4,q_4)\} \to S''_{L_1}(b_3,q_3);$$

$$q_4 = S'\{(a_2,q_2)+(a_5,q_5)\} \to S''_{L_1}(b_4,q_4);$$

$$\theta_4 : q_1 = q^*;\ q_2 = q^*;\ q_3 = q^*;$$

$$q_4 = S'\{(a_1,q_1)+(a_5,q_5)\} \to S''_{L_1}(b_5,q_5);$$

$$S'(a_1,q_1) := S''(b_1,q_1);\ S'(a_2,q_2) := S''(b_2,q_2);$$

$$S'(a_3,q_3) := S''(b_3,q_3);\ S'(a_4,q_4) := S''(b_4,q_4).$$

$$ext(f) = \max\{S''(a_1,q_1) \vee S''(a_2,q_2) \vee S''(a_3,q_3) \vee S''(a_4,q_4)\}.$$

$$S_2 : \theta_1 : q_1 := S'\{(a_1,q_1)+(a_2,q_2)\} \to S''_{L_1}(b_1,q_1);$$

$$q_2 := S'\{(a_2,q_2)+(a_3,q_3)\} \to S''_{L_1}(b_2,q_2);$$

$$q_3 := S'\{(a_3,q_3)+(a_4,q_4)\} \to S''_{L_1}(b_3,q_3);$$

$$q_4 := 0.$$

$$\theta_2 : q_1 = q^*;$$

$$q_2 := S'\{(a_1,q_1)+(a_3,q_3)\} \to S''_{L_1}(b_2,q_2);$$

$$q_3 := S'\{(a_2,q_2)+(a_4,q_4)\} \to S''_{L_1}(b_3,q_3);$$

$$q_4 := 0.$$

$$\theta_3 : q_1 = q^*;\ q_2 = q^*;$$

$$q_3 := S'\{(a_1,q_1)+(a_4,q_4)\} \to S''_{L_1}(b_3,q_3);$$

$$q_4 := 0.$$

$$S'(a_1,q_1) := S''(b_1,q_1);\ S'(a_2,q_2) := S''(b_2,q_2);\ S''(a_3,q_3) := S''(b_3,q_3).$$

$$ext(f) = \max\{S''(a_1,q_1) \vee S''(a_2,q_2) \vee S''(a_3,q_3)\}.$$

$$S_3 : \theta_1 : q_1 := S'\{(a_1,q_1)+(a_2,q_2)\} \to S''_{L_1}(b_1,q_1);$$

$$q_2 := S'\{(a_2,q_2)+(a_3,q_3)\} \to S''_{L_1}(b_2,q_2);$$

$$q_3 := 0;\ q_4 := 0.$$

$$\theta_2 : q_1 = q^*;$$

$$q_2 := S^{'}\{(a_1, q_1) + (a_3, q_3)\} \to S^{''}_{L_1}(b_2, q_2);$$

$$q_3 := 0;\ q_4 := 0.$$

$$S^{'}(a_1, q_1) := S^{''}(b_1, q_1);\ S^{'}(a_2, q_2) := S^{''}(b_2, q_2);$$

$$ext(f) = \max\{S^{''}(a_1, q_1) \vee S^{''}(a_2, q_2)\}.$$

$$S_4 : \theta_1 : q_1 := S^{'}\{(a_1, q_1) + (a_2, q_2)\} \to S^{''}_{L_1}(b_1, q_1);$$

$$q_2 := 0;\ q_3 := 0;\ q_4 := 0.$$

$$S^{'}(a_1, q_1) := S^{''}(b_1, q_1);$$

$$ext(f) := \max\{S^{''}(a_1, q_1)\}.$$

3.3. Архитектура параллельной вычислительной структуры систолического типа для решения задачи распределения фрагментов сетевой базы данных

Современные ПВС, как правило, узкоспециализированые. Трудность создания проблемно-ориентированных ПВС обусловлена нерешенностью целого комплекса проблем, определяющих взаимосвязь между архитектурой ПВС и структурой параллельных алгоритмов, предназначенных для реализации на этих ПВС. Особенно остро эта проблема возникает при решении так называемых сильносвязанных задач, разбиение которых на подзадачи приводит к необходимости интенсивных обменов на каждом шаге решения отдельных подзадач. Именно к такому классу задач относятся оптимизационные задачи, которые приходится многократно решать в процессе принятия решений. Трудность создания ПВС, ориентированных на широкий класс оптимизационных задач, связана еще и с большой размерностью решаемых оптимизационных задач и довольно жесткими требованиями к оперативности принятия решений при управлении сложными системами.

Анализ основных тенденций развития параллельных вычислительных систем показывает, что здесь можно выделить следующие направления развития ПВС:

- создание ПВС для решения задач первичной обработки информации на нижнем уровне систем, которые сводятся к масштабированию, перемножению, дифференцированию, интегрированию, фильтрации и другим различным преобразованиям сигналов;
- разработка проблемно-ориентированных ПВС для решения задач линейной алгебры, а также дифференциальных уравнений в частных производных;
- разработка ПВС для решения задач комбинаторной оптимизации и теории графов. Это направление является наиболее сложным и наименее разработанным на сегодняшнее время, а также наиболее

важным с точки зрения построения систем динамического управления в сетях и автоматизации процесса принятия решения.

Предлагаемая архитектура параллельной вычислительной структуры систолического типа состоит из *n*-процессорных элементов. Каждый процессорный элемент характеризуется именем и алфавитом. Символы алфавита могут иметь интерпритацию как константы или переменные. Парой будем называть состояние процессорного элемента. Множественное число состояний всех процессорных элементов системы в определенный момент времени будем называть глобальным состоянием системы, а некоторое его подмножество — локальным состоянием. При данных процессорный элемент свое состояние, и, связанных с ним элементов, т. е., делает локальное преобразование что будем называть операцией (рис. 4).

В блоке сортировки данных 1 (рис. 4) параллельной вычислительной структуры систолического типа выполняется сортировка данных по формулам:

$$\alpha_1 \le \alpha_2 \le \ldots \le \alpha_n, \tag{20}$$

$$c_1 \ge c_2 \ge \ldots \ge c_n, \tag{21}$$

$$c_1/a_{11} \ge c_2/a_{21} \ge \ldots \ge c_n/a_{n1} \tag{22}$$

в случае одномерной задачи. Выражение (20) выполняет сортировку данных в порядке убывания значений ограничения; выражение (21) выполняет сортировку данных в порядке возрастания значений ограничения; выражение (22) выполняет сортировку данных в порядке убывания значений ограничения.

В случае *m*-мерной задачи такая сортировка невозможна, поскольку вместо одного ограничения используется m ограничений. Поэтому введем параметр Ψ_j, по которому следует осуществлять сортировку коэффициентов в функционале. Значения Ψ_j предлагается вычислять по одному из следующих соотношений:

$$\Psi_j = c_j \Big/ \max_i \overset{*}{a}_{ij}, \tag{23}$$

$$\Psi_j = c_j \Big/ (\max_i \overset{*}{a}_{ij} - \min_i \overset{*}{a}_{ij}), \tag{24}$$

$$\Psi_j = c_j \Big/ ((\max_i \overset{*}{a}_{ij}) \sum_{i=1}^{m} \overset{*}{a}_{ij}), \tag{25}$$

$$\Psi_j = c_j \Big/ \sum_{i=1}^{m} \overset{*}{a}_{ij}, \tag{26}$$

где $a_{ij} = a_{ij}/b_i$.

Блок процессорных элементов 3 (см. рис. 4) архитектуры параллельной вычислительной структуры систолического типа осуществляет вычисления локальных экстремумов при заданном значении функционала (3) и ограничении (4), (5), а так-

же определения (вычисления) номера вершины, в которой локальный экстремум (ЛЭ) определен по правилам (15)–(17).

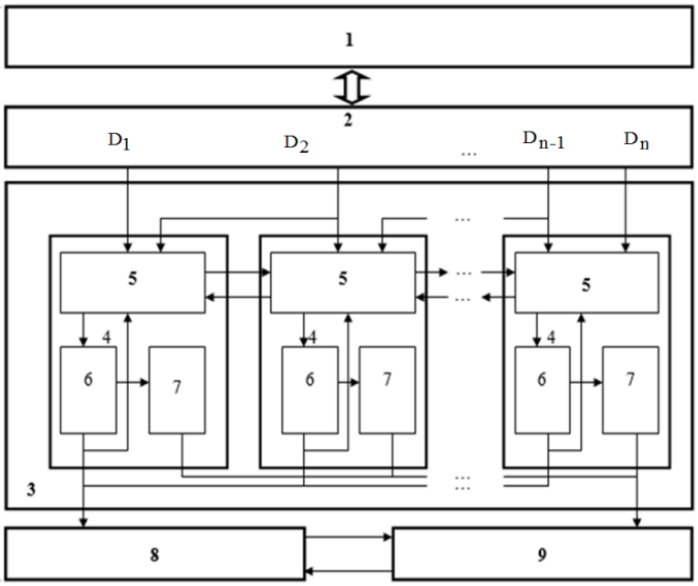

Рисунок 4. Архитектура параллельной вычислительной структуры систолического типа [12, 13]

Каждый процессорный элемент 4 блока процессорных элементов 3 (см. рис. 4) выполняет вычисления параллельно и осуществляет обмен данными между соседними процессорными элементами после завершения вычислений. Блок регистров 5 каждого процессорного элемента 4 сохраняет и обеспечивает микрооперации передачи данных между регистрами блока регистров соседних процессорных элементов. Арифметический вычислитель 6 вычисляет локальные экстремумы на основании данных, поступающих с блока регистров, выбирает локальный экстремум по правилам (15)–(17) и пересылает его в вычислительное устройство формирования вектора пути 8 для вычисления глобального экстремума и формирования вектора пути. Блок идентификации 7 определяет номер вершины, в которой локальный экстремум определен (см. рис. 4). Модуль памяти 9 (см. рис. 4) сохраняет номера вершин локальных экстремумов на каждом ранге вычислений. Данные $D_1, D_2, ..., D_n$ поступают одновременно в каждую систолическую ячейку вычислительного устройства, в которых осуществляется вычисления. Ввод данных осуществляется с помощью блока управления систолическим процессором 2 (см. рис. 4) с блока сортировки данных 1.

4. Результаты экспериментального исследования

В ходе решения тестовых задач генерировались по равномерному закону распределения с помощью датчика случайных чисел коэффициенты в функционале в диапазоне $[1 \div 100]$ и в ограничениях в диапазоне $[1 \div 50]$. Выбор других диапазонов для функционала изменил только его абсолютное значение, но в среднем не повлиял на параметры алгоритмов. Изменение диапазона в ограничениях влияет только на ранг пути (ранг пути — количество вершин графа $D\Delta$, образующих этот путь).

Структура исследования каждого алгоритма заключалась в следующем. Решалось 100 тестовых задач с заданными входными параметрами n выбранным алгоритмом с различной сортировкой коэффициентов при функционале и ограничении. В ходе решения определялись показатели эффективности алгоритмов. Результаты решения задачи приближенным алгоритмом сравнивались с результатом ее решения точным алгоритмом. Полученные значения представлены на графиках рис. 5.

Как показали результаты экспериментального исследования, количественные значения выбранных показателей существенно зависят от ранга получаемого решения, определяют число единиц в оптимальном решении. Так, диапазон изменения rcp можно условно разбить на три условно выделенные зоны: зона — $rcp = [0 \div n/3]$; 2 зона — $rcp = [n/3 \div 2n/3]$; 3 зона — $rcp = [2n/3 \div n]$.

Из анализа графиков можно сделать вывод, что наилучшей точностью обладает алгоритм MAX-MIN, погрешность которого при $n > 50$ не превышает 0,5%. Наиболее эффективной сортировкой является сортировка в порядке убывания отношений коэффициентов при функционале к соответствующим коэффициентам в ограничениях.

Величина располагаемого времени и времени расчетов на практике зависят от множества непредвиденных условий и реально являются случайными, оценим показатель оперативности каждой модели распределения фрагментов сетевых баз данных как вероятность своевременного решения задачи:

$$P = 1 - e^{-T_p/T}. \qquad (27)$$

Используя среднее значение времени цикла моделирования и размещаемого времени (3 минуты), найдем значение показателя вероятности своевременного решения задачи. В качестве алгоритмов для исследования использовались:

- точные алгоритмы: алгоритм Балаша; многоэтапный алгоритм на основе рангового подхода;
- приближенные алгоритмы: алгоритмы с временной сложностью $O(n)$; алгоритмы с временной сложностью $O(n^2)$ при сортировке по ограничению; алгоритмы с временной сложностью $O(n^2)$ при сортировке по функционалу; алгоритмы с временной сложностью $O(n^2)$ при сортировке по отношению; алгоритмы с временной сложностью $O(n^3)$; парал-

лельные алгоритмы; алгоритм, который обеспечивает максимальную точность вычислений при допустимых временных и ресурсных затратах.

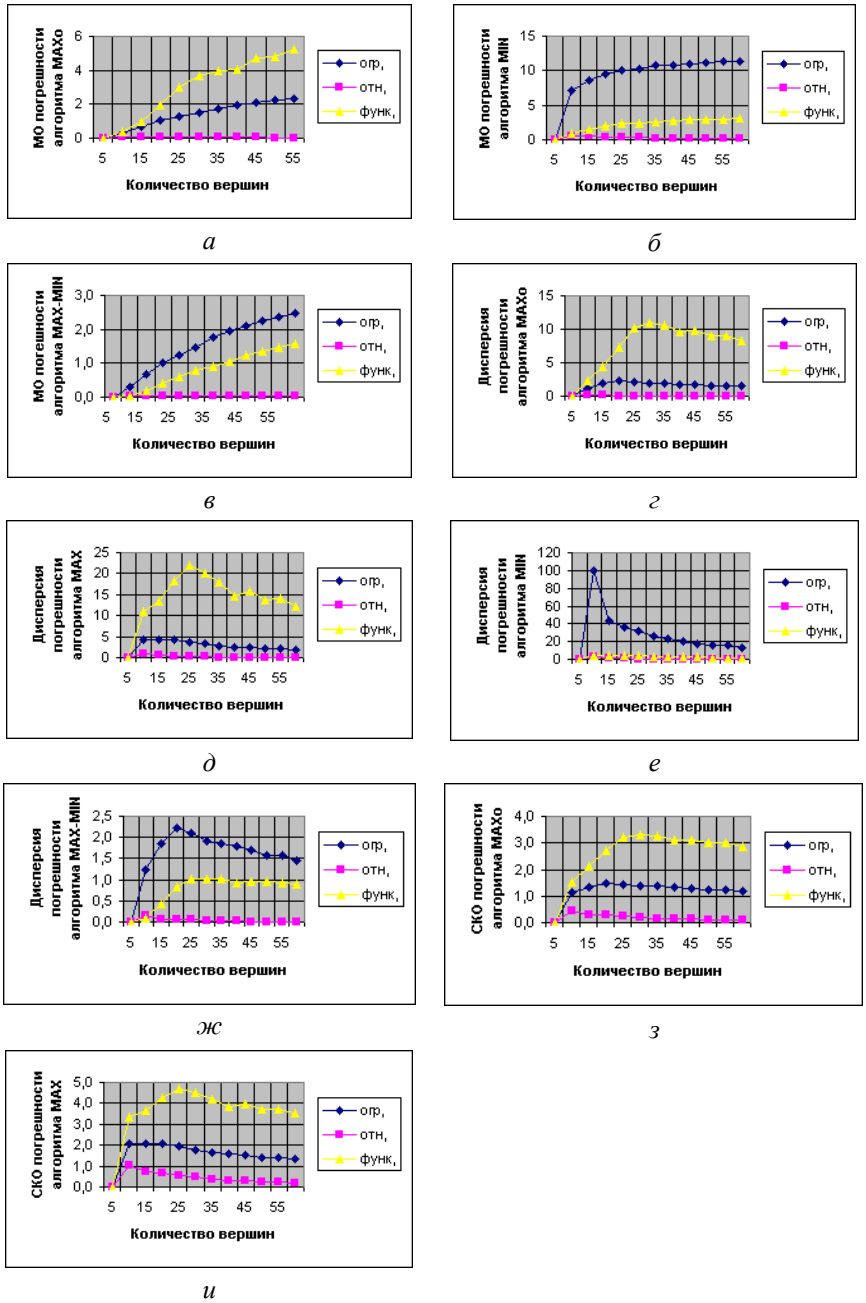

Рисунок 5. Зависимость погрешности и ее характеристик от размерности задачи

Результаты моделирования показывают (рис. 6), что при уровне вероятности своевременного решения задачи в настоящее время обеспечение расчетам этапа оптимального планирования распределения фрагментов сетевых баз данных возможно только методами с временной сложностью $O(n)$ и алгоритмом, который обеспечивает заданную точность вычислений при допустимых временных и ресурсных затратах. Применение точных алгоритмов возможно при небольшой размерности решаемой задачи распределения — до 250 вершин графа, алгоритмов с временной сложностью $O(n^2)$ до 400.

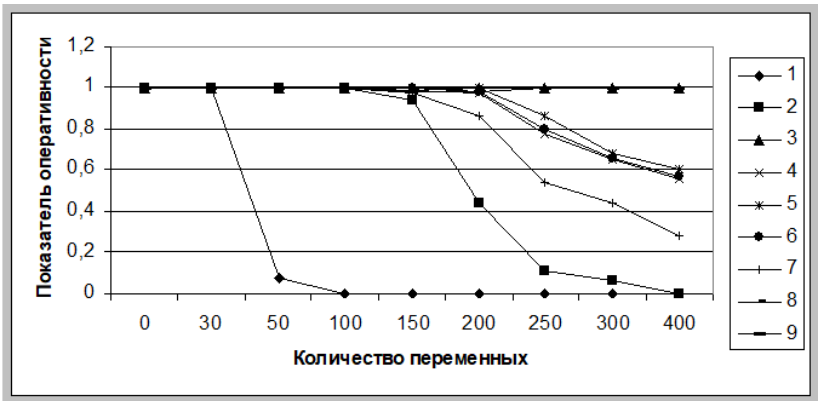

Рисунок 6. График зависимости показателя вероятности своевременного решения задачи от ее размерности

5. Заключение

Разработана математическая модель распределения фрагментов данных сетевой базы данных по узлам вычислительного комплекса является задачей целочисленного линейного программирования с булевыми переменными. Полученные правила отсечения неперспективных вариантов решений и обобщенная процедура A_0 позволили разработать одномерные и *m*-мерные алгоритмы приближенного решения задачи целочисленного линейного программирования с булевыми переменными. Экспериментальное сравнение разработанных алгоритмов с известными, по выбранным показателям эффективности показал, что их временная сложность существенным образом зависит от ранга оптимального решения, который может принадлежать одной из трех условно выделенных зон. Поэтому объективное сравнение алгоритмов возможно только для решений, принадлежащих одной и той же зоне. Из сравнения решений по зонам видно, что наибольший выигрыш предлагаемые алгоритмы дают во второй зоне, где число допустимых решений экспоненциально. Это является важным преимуществом по сравнению с известными методами. Из

экспериментальных исследований видно, что погрешность предложенных алгоритмов существенным образом зависит от способов сортировок коэффициентов в функционале и ограничениях и выявлены наилучшие типы сортировок и стратегии отсечения неперспективных решений. Поскольку формирование множеств путей на ярусах в графе $D\Delta$ можно совмещать во времени, то предлагаемые алгоритмы могут быть эффективно распараллелены, при этом если число процессорных элементов в вычислительной системе равно числу переменных, то временная сложность предложенных алгоритмов может быть понижена в n раз.

Применение метода позволяет обеспечить от 1 до 10% погрешность решения задачи (3)–(5) при использовании правил (15)–(17) для отсечения неперспективных вариантов решения.

Литература

[12] *Новиков Б. А., Левин М. Ю.* Сравнительный анализ производительности SQL и NoSQL СУБД // *Компьютерные инструменты в образовании*, 2017. № 4. С. 48–63.

[13] *Болодурина И. П., Парфенов Д. И., Решетников В. Н.* Моделирование размещения сервис-ориентированных приложений в программно-управляемой инфраструктуре виртуального центра обработки данных // *Программные продукты и системы / Software & Systems*. 2016. № 4(29). С. 15–22

[14] *Майер-Шенбергер В., Кукьер К.* Большие данные. Революция, которая изменит то, как мы живем, работаем, мыслим. — М. : Манн, Иванов и Фербер, 2014.

[15] *Buyya R.* Big Data Analytics-Enhanced Cloud Computing: Challenges, Architectural Elements, and Future Directions // Parallel and Distributed Systems (ICPADS), 2015 IEEE 21st International Conference on. IEEE, 2015. P. 75–84.

[16] *Буй Д. Б., Скобелев В. Г.* Модели, методы и алгоритмы оптимизации запросов в базах данных // *Компьютерные системы и информационные технологии*. 2014, № 2 (66). С. 43–58.

[17] *Дубинин В. Н., Зинкин С. А.* Сетевые модели распределенных систем обработки, хранения и передачи данных: монография. — Пенза : Приволжский Дом знаний, 2013.

[18] *Соколинский Л. Б.* Параллельные системы баз данных. — М. : Изд. Московского университета, 2013.

[19] Информационное обеспечение систем организационного управления (теоретические основы): в 3-х ч. / под ред. Е. А. Микрина, В. В. Кульбы. — М. : Физматлит, 2011. Ч. 2. Методы анализа и проектирования информационных систем.

[20] *Фильгус Д. И.* Методы и алгоритмы распределения рабочей нагрузки в сетевых базах данных // *Международный журнал прикладных и фундаментальных исследований*. 2018. № 4.

[21] *Третьяк В. Ф., Пашнева А. А.* Оптимизация структуры хранилища данных в узлах инфокоммуникационной сети облачной среды // *Системы управления, навигации и связи*

Полтавского национального университета имени Юрия Кондратюка. 2017. № 4 (44). С. 122–128.

[22] *Мамедов К. Ш., Мамедов Н. Н.* Алгоритмы построения гарантированного решения и гарантированного приближенного решения многомерной задачи о ранце // *Проблемы управления и информатики*. 2014. № 5. С. 30–37

[23] Патент на полезную модель № 69487, Украина, МПК G06 F15/00. Устройство для решения задач на графах / В. Ф. Третяк, Д. Ю. Голубничий та др. № u201113667; заяв. 21.11.2011; опубл. 25.04.2012; бюл. № 8.

[24] Патент на полезную модель № 92968, Украина, МПК G06 F15/00. Способ обратотки и защиты информации в распределенных хранилищах данных / В. Ф. Третяк, В. В. Баранник и др. № u201403994; заяв. 14.04.2014; опубл. 10.09.2014; бюл. № 17.

Авторы:

Елена Гельевна Андрианова — доцент кафедры корпоративных информационных систем Института информационных технологий, МИРЭА — Российский технологический университет

Вячеслав Константинович Раев — профессор кафедры инструментального и прикладного программного обеспечения Института информационных технологий. МИРЭА — Российский технологический университет

Дмитрий Игоревич Фильгус — аспирант кафедры инструментального и прикладного программного обеспечения Института информационных технологий, МИРЭА — Российский технологический университет

Development of parallel computing methods for fragmentation of network database data based on the rank approach

E. G. Andrianova, V. K. Raev, D. I. Filgus

MIREA — Russian technological university, 78, Prospect Vernadskogo, Moscow, Russia, 119571
e-mail: andrianova@mirea.ru; vkr3708@gmail.com, ar4ers@ya.ru

Annotation. The analysis of the task of fragmentation of the data of the network database showed that the formal model is the so-called integer linear programming problem with Boolean variables. Unfortunately, the problem of CAP with BP belongs to the class NP (nondeterministic polynomial), which are difficult to solve even with the use of modern computers. At present, combinatorial methods occupy a dominant place in the methods of solving these problems. To them, first of all, you can include methods of full busting, dynamic programming, and also local algorithms. The practical application of these methods is difficult in solving problems of large dimension. Attempts to reduce the time for solving the problems of CML with BP due to parallelization face another problem in the theory of parallel computations, which consists in the fact that, from the point of view of parallel algorithms, this type of problem belongs to the class of strongly coupled tasks and therefore can not be paralleled. Therefore, when developing parallel algorithms for solving the problem of CML with BP, in

addition to the contradiction between the accuracy of the solution of the problem and the time of its solution, another contradiction arises — between the strong connectivity inherent in this task and the need for its parallelization.The proposed method of parallel computing for fragmentation of network database data is based on a rank approach to its solution, the principle of optimization in the direction and strategies for cutting off unpromising solutions. The method allows to determine local extrema at the vertices of the graph and on their basis to determine the global extremum.
Keywords: data, parallel computing, rank approach, fragmentation, network database

References

[1] *Novikov B. A., Levin M. Yu.* (2017) *Komp'yuternyye instrumenty v obrazovanii*, 4:48–63. [In Rus]

[2] *Bolodurina I. P., Parfenov D. I., Reshetnikov V. N.* (2016) *Programmnyye produkty i sistemy / Software & Systems*. **4**(29):15–22. [In Rus]

[3] *Mayyer-Shenberger V., Kuk'yer K.* (2014) Bol'shiye dannyye. Revolyutsiya, kotoraya izmenit to, kak my zhivem, rabotayem, myslim. Moscow, Mann, Ivanov i Ferber. [In Rus]

[4] *Buyya R.* (2015) Big Data Analytics-Enhanced Cloud Computing: Challenges, Architectural Elements, and Future Directions. Proceedings of 21st International Conference on Parallel and Distributed Systems (ICPADS, 2015 IEEE). P. 75–84.

[5] *Buy D. B., Skobelêv V. G.* (2014) *Komp'yuternyye sistemy i informat. tekhnologii*. **2**(66):43–58. [In Rus]

[6] *Dubinin V. N., Zinkin S. A.* (2013) Setevyye modeli raspredelennykh sistem obrabotki, khraneniya i peredachi dannykh. Penza, Privolzhskiy Dom znaniy. [In Rus]

[7] *Sokolinskiy L. B.* (2013) Parallel'nyye sistemy baz dannykh. Moscow, Izdatel'stvo Moskovskogo universiteta. [In Rus]

[8] Informatsionnoye obespecheniye sistem organizatsionnogo upravleniya (teoreticheskiye osnovy). v 3 ch. Moscow, Fizmatlit, 2011. Ch. 2. Metody analiza i proyektirovaniya informatsionnykh sistem. [In Rus]

[9] *Fil'gus D. I.* (2018) Mezhdunarodnyy zhurnal prikladnykh i fundamental'nykh issledovaniy, 4. [In Rus]

[10] *Tret'yak V. F., Pashneva A. A.* (2017) Sistemy upravleniya, navigatsii i svyazi Poltavskogo natsional'nogo universiteta imeni Yuriya Kondratyuka, **4**(44):122–128. [In Rus]

[11] *Mamedov K. Sh., Mamedov N. N.* (2014) *Problemy upravleniya i informatiki*. 5:30–37. [In Rus]

[12] Patent 69487, Ukraina, MPK G06 F15/00. Ustroystvo dlya re-sheniya zadach na grafakh / V. F. Tretyak, D. YU. Golubnichiy ta dr. № u201113667; zayav. 21.11.2011; opubl. 25.04.2012; byul. № 8.

[13] Patent na poleznuyu model' № 92968, Ukraina, MPK G06 F15/00. Sposob obratotki i zashchity informatsii v raspredelennykh khranilishchakh dannykh / V. F. Tretyak, V. V. Ba-rannik i dr. № u201403994; zayav. 14.04.2014; opubl. 10.09.2014; byul. № 17.