https://doi.org/10.32362/2500-316X-2019-7-4-7-20

УДК 621.311.6:621.3.089.2

Определение кратчайших гамильтоновых путей в произвольных графах распределенных баз данных

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

МИРЭА – Российский технологический университет, Москва 119454, Россия
@Автор для переписки, e-mail: dmif42@ya.ru

Разработан метод поиска кратчайшего гамильтонова пути в произвольном графе на основе рангового подхода, который обеспечивает высокую оперативность и существенное уменьшение погрешности решения задачи организации процесса управления множеством транзакций и запросов при их реализации в сетевых базах данных. Во многих случаях существующие решения не обеспечивают необходимых результатов по времени доступа и точности найденного решения. Использование разработанного метода позволяет минимизировать время простоя вычислительных устройств, сократить объемы и время передачи данных от одних исполнительных устройств к другим, повысить общую масштабируемость, минимизировать время доступа к данным и пр. Важным достоинством предлагаемого метода является уменьшения числа элементарных операций и числа обрабатываемых векторов в процедуре формирования очереди выполнения операций запроса, что приводит к существенному уменьшению времени на реализацию этих процедур. В работе используются методы теории графов. Оценка эффективности решения задачи выполнена с использованием системного подхода, системного анализа и теории исследования операций. Обработка экспериментальных данных, полученных в ходе работы, проведена в соответствии с положениями математической статистики.

Ключевые слова: проблема, классификация, системная инженерия, требования, валидация, верификация.

Determination of the Shortest Hamiltonian Paths in an Arbitrary Graph of Distributed Databases

Elena G. Andrianova,
Vyacheslav K. Raev,
Dmitry I. Filgus@

MIREA – Russian Technological University, Moscow 119454, Russia
@Corresponding author e-mail: dmif42@ya.ru

A method has been developed for finding the shortest Hamiltonian path in an arbitrary graph based on the rank approach, which provides high efficiency and a significant reduction in the error in solving the problem of organizing the process of managing multiple transactions and queries when they are implemented in network databases. In many cases, existing solutions do not provide the necessary results in terms of access time and accuracy of the found solution. Using the developed method allows minimizing the idle time of computing devices, reducing the volume and time of data transfer from one device to another, increases overall scalability, minimizes data access time, etc. An important advantage of the proposed method is to reduce the number of elementary operations and the number of vectors being processed the queue of the operations of the request, which leads to a significant reduction in time to implement the procedures for the formation of echer di operations in the requests. Methods of graph theory were used in this paper. The effectiveness of the task solution was evaluated using a systems approach, system analysis and the theory of operations research. Processing of experimental data obtained during the work was carried out in accordance with the provisions of mathematical statistics.

Keywords: graph, Hamiltonian path, query, rank approach, short tree of paths, transaction, rank, distributed database.

Введение

Распределенные системы управления базами данных (СУБД) являются основным компонентом современных систем обработки информации (СОИ). Все использующиеся в настоящее время модели СОИ – такие, как модель файлового сервера, модель сервера баз данных, модель сервера приложений, модель распределенных вычислений и др. – имеют в своей основе выполнение запросов к базам данных, которые обрабатываются СУБД. Существует множество определений термина «оптимизация запросов», однако наиболее часто встречающееся в литературе формулируется следующим образом: под оптимизацией запросов понимается функция распределенной СУБД, осуществляющая поиск оптимального плана выполнения запросов из всех возможных для заданного запроса, либо процесс изменения запроса и/или структуры баз данных (БД) с целью уменьшения использования вычислительных ресурсов при выполнении запроса [1–5].

Основы метода решения задачи оптимизации запросов, различные модификации которого с успехом используются во многих дорогих современных реляционных СУБД со встроенными оптимизаторами, даны в работе [4]. На практике часто приходится использовать относительно недорогие системы, не имеющие оптимизаторов, где пользователи нередко применяют семантически прозрачные запросы, но не оптимальные с точки зрения их эффективности. Более того, как показывает опыт, и в системах со встроенными оптимизаторами небесполезна предварительная внешняя оптимизация. Возникает потребность в разработке программной надстройки для формирования оптимальных запросов. Здесь следует отметить, что существуют различные подходы к проблеме оптимизации запросов пользователей в СУБД. Это одна из наиболее сложных задач компьютерной обработки информации, которая является NP-полной.

Несмотря на наличие частных результатов, относящихся к специальным классам графов, в общем случае задача определения гамильтоновых циклов и гамильтоновых путей недостаточно изучена [1, 2]. Так, например, до сих пор нет эффективной процедуры нахождения гамильтонова пути в произвольном графе. Более того, нет даже хороших методов доказательства существования такого пути. Но актуальность, определяемая сводимостью множества задач управления информационными системами, обусловливает необходимость нахождения новых решений, соответственно улучшающих существующие [6–11].

Создание полиномиальных алгоритмов для решения NP-полных задач может базироваться не только на идее рангового подхода. Классический подход к решению любой NP-полной задачи заключается в попытке выработать стратегию отсечения неперспективных вариантов решений на экспоненциальном множестве W допустимых решений. Применение «жадных» алгоритмов на множестве W приводит либо к необходимости полного перебора вариантов, либо к эвристическим решениям, которые могут отличаться от оптимальных на 70% и более. Поэтому и возникли алгоритмы на основе идей метода ветвей и границ. Необходимо подчеркнуть, что если на первых шагах работы алгоритма удается найти оптимальное решение, то затем необходимо осуществить неявный полный перебор всех вариантов на множестве W, например, мощности 2^n, чтобы убедиться, что данное решение действительно оптимально. В этом заключается основной недостаток метода.

Не менее важным с точки зрения практического применения разрабатываемых алгоритмов является вопрос о возможности построения параллельных алгоритмов решения задачи поиска кратчайшего гамильтонова пути, что связано с использованием концепций мультипрограммирования и многопроцессорной обработки информации управления [3]. Задача определения кратчайших гамильтоновых путей в графе с точки зрения построения параллельных алгоритмов относится к классу сильно связанных задач [4]. Как показано ранее [12, 13], для эффективного распараллеливания задачи произвольной размерности необходимо определять оптимальное число процессорных элементов, поскольку с их ростом суммарное время ее решения может резко увеличиваться из-за неоправданно растущего числа обменных операций. При этом сама задача определения оптимального числа процессорных элементов также относится к классу NP-полных задач, и эффективные методы ее решения неизвестны [5, 7, 14, 15].

Общий подход к решению задач комбинаторной оптимизации и оптимизационных задач в теории графов

Рассмотрим конечное множество произвольных объектов теории графов или конфигураций комбинаторной оптимизации. В общем случае объект может быть произвольным, но должны быть определены конечное множество элементов $\Omega = \{\omega_1\}$ или подмножества $L_i \in \Omega$ и правила R, позволяющие формировать объекты из исходных элементов или подмножеств L_i, принадлежащие множеству Ω.

Пусть заданы: некоторое разбиение множества Ω на семейства подмножеств $\{L_i\}$ так, что $\bigcup_i L_i = \Omega$, и L_i описывают интересующие нас объекты, состоящие из базовых элементов $\{l_i\}$, таких, что $= \bigcup_i l_i$, а также правило R, позволяющее из базовых элементов определять весовые характеристики произвольных объединений $L_к \cup L_p \in \Omega$, характеризующие свойства $\{v\}$. Требуется определить объект с интересующим нас свойством $v^* \in \{v\}$.

Представим множество всех возможных объединений подмножеств L_i в виде графа D (рис. 1) с параллельно ярусной структурой, состоящей из n горизонтальных линеек с вершинами 1, 2, …n и n ярусами, каждый из которых содержит все вершины графа D. При этом каждой вершине графа D поставим в соответствие базовый элемент l_i. В графе D произвольная вершина i может быть достигнута путями рангов $r = 1$, $r = 2$, … $r = n–1$, а произвольному пути μ_{st}, удовлетворяющему правилам построения R и проходящему через вершины $(j, p, …k, t)$, соответствует объединение базовых элементов $(l_j \cup l_p \cup … \cup l_k \cup l_t)$, определяющих некоторый объект $L_i \in \Omega$. Длина этого пути $d(\mu_{st})$ определяется по правилам, принадлежащим множеству R. Следовательно, множество всех путей $m_{si}(r)$ в графе D, удовлетворяющих правилам R, определяют область допустимых решений исходной задачи по выделению объекта с интересующим нас свойством $v^* \in \{v\}$. В качестве исходной вершины в графе D будем использовать фиктивную вершину S, которую в некоторых случаях удобно отождествлять с нулевым или исходным состоянием системы. Это приводит к тому, что максимальный ранг пути в графе D становится равным n, а добавление вершины S к базовым элементам системы не изменяет их свойств, определяемых правилами R [7].

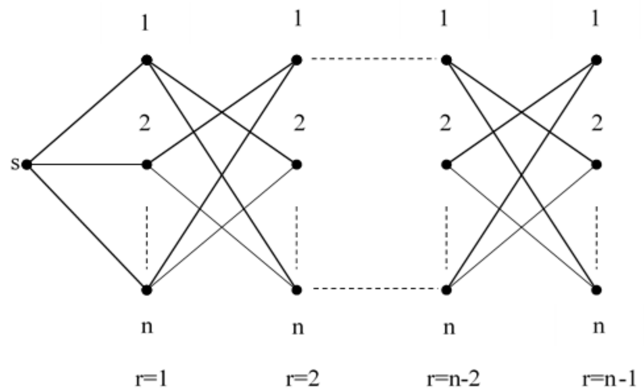

Рис. 1. Стянутое дерево всех путей D графа G,V, E.

Множество сложных систем, обладающих различными свойствами $\{F_i\}$, может быть отображено с помощью некоторого подмножества графов $\{G_i\}$. Рассмотрим произвольный n-вершинный граф $G(V, E) \in \{G_i\}$, который описывает состояние системы $F \in \{F_i\}$ с конечным числом состояний n. Вершины $\{i\} \in V$ графа $G(V, E)$ соответствуют возможным состояниям системы, пути в графе $G(V, E)$, определяемые последовательностью прохождения вершин $\{v_i\}$ и ребер $\{(i, j)\}$, характеризуют возможный порядок достижения состояния $i = p$ из некоторого исходного состояния s. Важной характеристикой пути является ранг пути r – число ребер, образующих путь. В графе $G(V, E)$ максимальное значение ранга $r = n–1$, и в общем случае ранг произвольного пути μ_{sp} характеризует сумму начального состояния, конечного состояния и числа состояний предшествования, через которые может быть достигнуто состояние p из некоторого исходного состояния s. Тогда множества путей m_{sj}^r; $j = (\overline{1, n})$ определяют способы достижения состояния j. В качестве базовых элементов $\{l_i\}$ выберем подмножество $\{vi\} \in V$ вершин графа $G(V, E)$, тогда объектам $\{L_j\}$ будет соответствовать все множество объектов Ω, которое можно породить на множестве V, используя правила R. Например, клики в произвольных графах, независимые множества, циклы графов, вектора и матрицы, задающие некоторый объект в $G(V, E)$. Каждый объект будем характеризовать $(m+1)$ весовой характеристикой, где m – это некоторые второстепенные характеристики объекта, на которые в общем случае могут быть наложены ограничения на то, что они не должны превышать некоторых величин $\{b_i\}$, $i = (1, 2 \ldots m)$. И имеется один определяющий показатель качества объекта, принадлежащий множеству Ω и обладающий определенным свойством v. Правила P определения весовых характеристик объектов естественно должны определяться в соответствии с правилами формирования самих объектов, и $P \in R$. Таким образом, пути μ_{sj}^r в графе D соответствует объект L_j, который может быть построен из r базовых элементов $\{v_j\}$, включая элемент j, на основе правил R. Соответственно, множества путей определяют множество объектов L_j, которые можно построить из r базовых элементов $\{v_j\}$, включая элемент j.

Дерево всех путей D содержит $(n–1)$ горизонтальную линейку и $(n–1)$ ярус. При определении путей прохождение по каждой горизонтальной линейке допускается только один раз, согласно определению задачи поиска кратчайшего гамильтонова пути. Исходя из стянутого дерева путей, для произвольной вершины j множество путей, ведущих в эту вершину из некоторой вершины S, можно представить в следующем виде:

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

где m_{sj}^r – подмножества путей из произвольной вершины s в некоторую вершину j графа $G(V, E)$, ранга r. Таким образом, используя граф D и введя правила формирования путей следующего ранга, можно из произвольной вершины S поэтапно строить пути произвольного ранга вплоть до ранга $r = n–1$. При таком подходе для решения задачи определения кратчайшего гамильтонова пути в графе D нужно построить кратчайшие пути ранга $r = n–1$ от вершины 1, 2 …n ко всем остальным вершинам графа и затем среди них выбрать кратчайший. То есть, если на основе подмножеств путей $m_{sj}^{r=1}$ в графе D строить подмножества $m_{sj}^{r=2}$ и так далее до $m_{sj}^{r=n-1}$, то мы будем вынуждены построить $(n–1)!$ путей:

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

С целью сокращения алгоритмической и вычислительной сложности разрабатываемых алгоритмов поиска кратчайшего гамильтонова пути (ПКГП) для формирования путей введем процедуру ***A***, позволяющую отсекать неперспективные пути ветвления дерева путей. Для отсечения неперспективных вариантов в процедуре A предлагается использовать принцип оптимизации по направлению к вершине p при формировании путей следующего ранга m_{Sp}^{r+1} на основе путей предыдущего ранга m_{sj}^r.

Разработка обобщенной процедуры A_0

Введем правила формирования путей ветвления с использованием процедуры $\boldsymbol{A}$:

1. Пути, формируемые на рангах $r = 1$ и $r = 2$, зависят от исходных значений весов и числа вершин n графа $D(V, E)$;

2. Пути, формируемые на рангах $r = 3, \ldots r = n–1$, зависят от значений весовых характеристик множества m_{sj}^{r}, где $r = 2$, т. е. путей, сформированных на ранге 2;

3. Выбор подмножества точек исхода ветвления в сформированном множестве m_{sj}^{r}, согласно определенного критерия эффективности, производится по правилу:

$$\min\left\{\mu_{sj}^{r}\right\} \tag{3}$$

4. Выбор точек входа в структуру ветвления по направлению к некоторой вершине p на ранге $r = r+1$, производится по правилу:

$$\min(j, p) \tag{4}$$

где (j, p) – ребро графа D.

Таким образом, сформулированные правила ветвления 1–4 позволяют построить рекуррентное соотношение, определяющее сущность реализации процедуры $\boldsymbol{A}$:

$$\mu_{sp}^{r+1} = \min_{j}\{\{\mu_{sj}^{r}\} \cup (j, p)\}; j = (\overline{1, n}); p = (\overline{1, n}); j \neq p\,, \tag{5}$$

где (j, p) – ребро графа D; n – число различных вершин в графе D.

Стянутое дерево путей с применением процедуры $\boldsymbol{A}$ соответственно будет иметь вид, представленный на рис. 2.

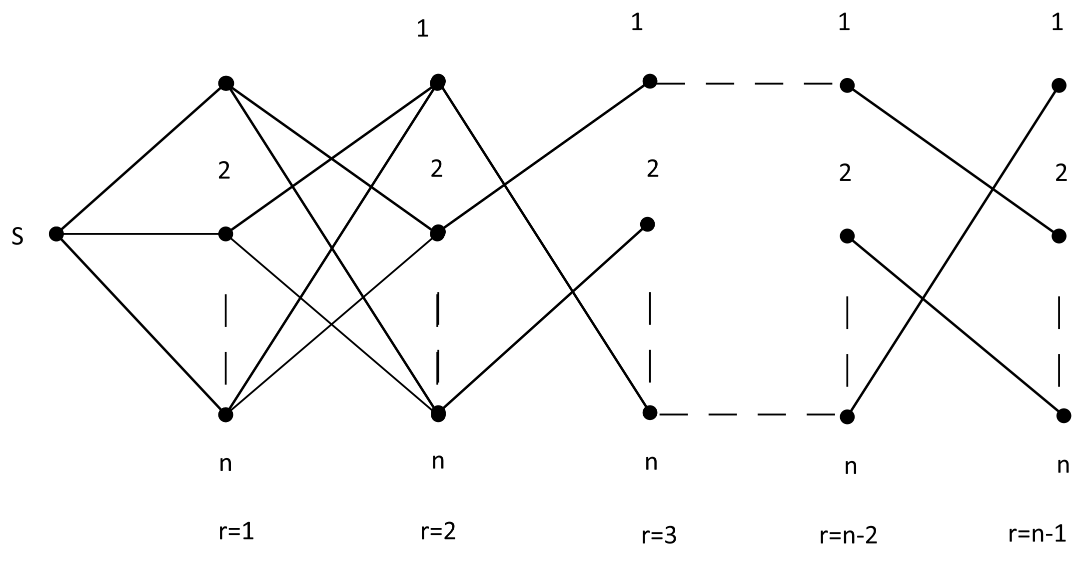

Рис. 2. Стянутое дерево всех путей D графа $G(V, E)$ с применением процедуры $\boldsymbol{A}$.

Рассмотрим порядок реализации процедуры ***А*** в графах поиска кратчайшего гамильтонова пути для произвольного числа вершин.

Исходя из отсутствия какой-либо существующей методики начала поиска кратчайшего гамильтонова пути, примем допущение, что поиск может начинаться с любой вершины без влияния на конечный результат. Такое допущение часто принимается при использовании алгоритмов, основанных на методе ветвей и границ. В нашем случае введение процедуры ***А*** предполагает некоторое отсечение неперспективных направлений поиска. Как следствие приведенного допущения и предназначенности процедуры отсечения, за исходную примем любую из вершин графа, а поиск на первых двух рангах будем производить по направлению ко всем возможным вершинам. То есть, из произвольной вершины S строятся все возможные пути ранга $r = 1$ ко всем вершинам графа D, далее, используя пути ранга $r = 1$, строятся все возможные пути ранга $r = 2$, на их основе формируются пути следующего ранга с использованием рекуррентного соотношения (5), и т. д. до построения путей ранга $r = n–1$.

Следует отметить, что если в процессе применения рекуррентного соотношения (5) возникает несколько кратчайших путей одинаковой длины, то необходимо их все строить на следующем ранге.

Выделим следующие особенности работы процедуры ***А***. В процессе ее реализации может возникать две ситуации. Первая – когда процедура ***А*** на каждом шаге построила пути в множества m_{sj}^{r}, т .е. принцип оптимальности работы процедуры не нарушался, и вторая – когда к одной из вершин ни одного пути построить нельзя. Последнее обстоятельство возможно в двух случаях:

а) если анализируемый граф неполный, и к вершине p не существует пути некоторого ранга $r = k$;

б) когда некоторая вершина p войдет во все пути предыдущего ранга.

Нетрудно видеть, что в первой ситуации процедура ***А*** не теряет оптимальное решение, а во второй ситуации (случай б) оптимальное решение может быть потеряно, поскольку принцип оптимальности работы процедуры нарушается (рис. 3). Это означает, что на основе процедуры ***А*** может быть построен только приближенный алгоритм решения задачи. Однако после нарушения принципа оптимальности последующее продление путей с использованием рекуррентного соотношения (5) позволит минимально отклоняться от оптимального решения. Блок-схема алгоритма процедуры ***А*** представлена на рис. 4.

Разработка алгоритма ПКГП решения задачи определения кратчайших гамильтоновых путей в произвольных графах

В качестве исходных данных для разработки алгоритма ПКГП будем использовать следующие характеристики графа $G(V, E)$:

– матрицу связности, определяющую наличие перехода между вершинами графа G: $G^{св} = \left\| v_{ij}^{св} \right\|$, где $v_{ij}^{св} = 1$, если существует связь между i-й и j-й вершинами графа G, в противном случае $v_{ij}^{св} = 0$.

– весовые характеристики нагрузки на дугах, определяемые некоторой неотрицательной величиной $C > 0$. Данная величина имеет некоторые особенности, которые определяются исходя из следующих соображений:

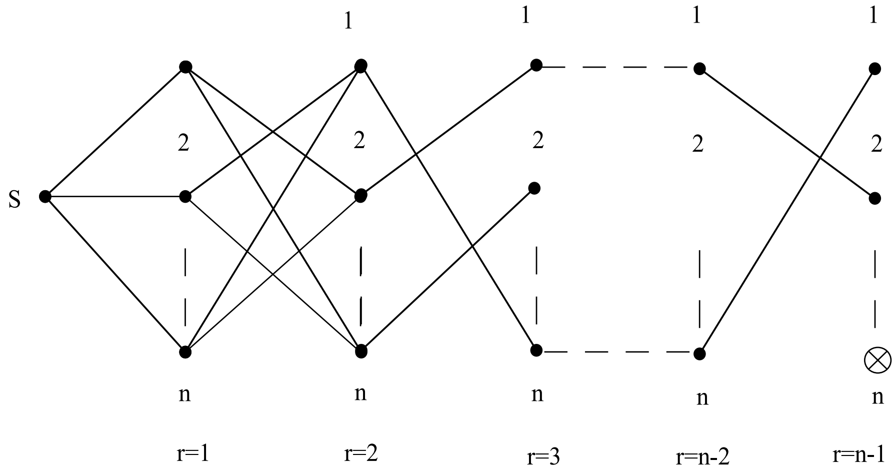

Рис. 3. Стянутое дерево путей *D* графа *G*(*V, E*),
с применением процедуры ***A*** (случай б).

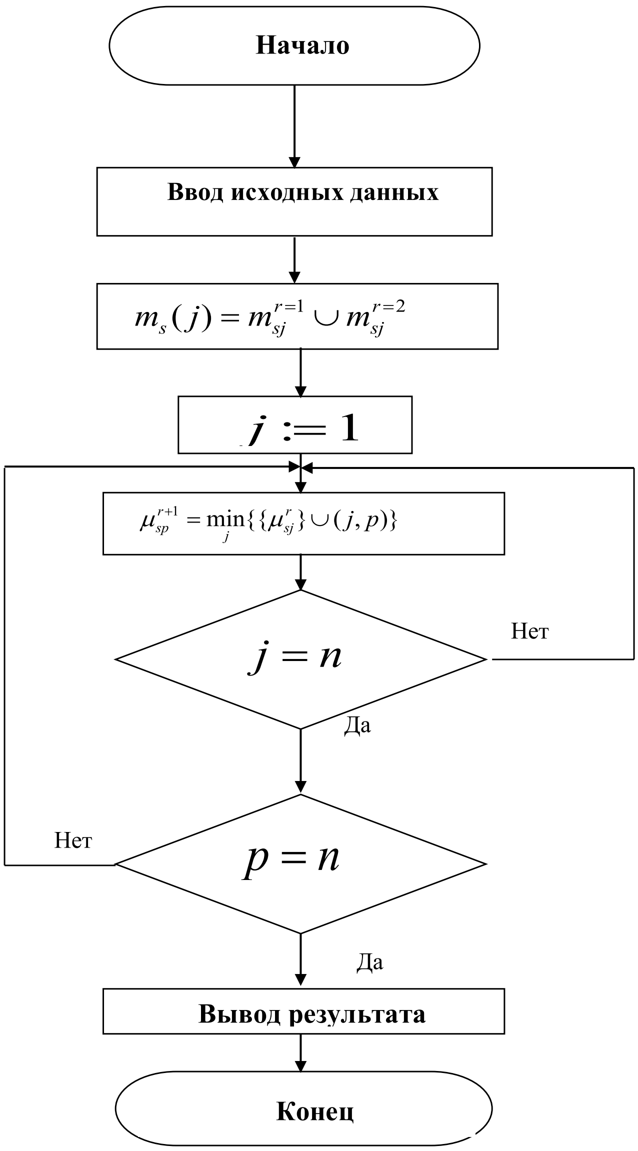

Рис. 4. Блок-схема алгоритма процедуры ***A***.

1. Если характеристическая нагрузка определяется числом элементарных операций, необходимых для выполнения заданной операции в алгоритме процедуры $\boldsymbol{A}$, то величина С принимает следующие значения: $C > 0$, $C = int$.

2. Если характеристическая нагрузка определяется временем выполнения заданной операции, то величина $C > 0$, C – определена на всем множестве действительных чисел.

3. Для вершин, между которыми отсутствуют связь $i \Leftrightarrow j$, принимается $C = \infty$.

В качестве исходной примем любую точку графа G, обозначив ее переменной s, физический смысл которой заключается в следующем: «вершина, от которой производится поиск» в процессе реализации алгоритма, принимает соответствующие значения, равные номерам вершин графа вплоть до некоторой вершины n. Временной точкой изменения состояния системы при переходе поиска от вершины s к вершине s+1 принимается условие окончания работы процедуры $\boldsymbol{A}$ по выполнению поиска кратчайшего гамильтонова пути от вершины s. В итоге работы алгоритма образуется некоторое множество путей с суммарной оценкой каждого:

$$C_{сум} = \sum_{r=1}^{n-1} \{C_{sj}^r\} \tag{6}$$

Данная суммарная оценка и является той характеристикой, которая предполагает нижнюю границу времени реализации каждого из путей. В соответствии с выбранным критерием эффективности определим оптимальный путь Π_{sj}^r из соотношения:

$$\Pi_{sj}^r = \mu_{sj}^r \in m_{sj}^r \equiv \min \sum_{r=1}^{n-1} \left\{C_{sj}^r\right\} \tag{7}$$

Тогда алгоритм ПКГП примет следующую последовательность операций, соответствующую приведенному описанию:

Шаг 1. Присваиваем значение S: = 1 переменной S.

Шаг 2. В графе D, используя процедуру $\boldsymbol{A}$, определяем кратчайшие пути ранга $r = n$–1 от вершины S ко всем остальным вершинам графа, используя рекуррентное соотношение (5).

Шаг 3. Увеличиваем значение S: = S+1.

Шаг 4. Проверяем $S = n$, если нет, то переходим к выполнению шага 2, в противном случае – выполняем следующий шаг.

Шаг 5. Среди всех построенных кратчайших путей ранга $r = n$–1 на шагах 1 и 2 выбираем самый короткий из соотношения (7).

Шаг 6. Вывод результата Π_{sj}^r.

Шаг 7. Конец работы алгоритма. Схема работы алгоритма поиска кратчайшего гамильтонова пути с использованием процедуры $\boldsymbol{A}$ представлена на рис. 5.

Если существует Q вершин i, удовлетворяющих соотношению (5), то это означает, что существует Q минимально равноудаленных вершин от вершины s, определяемой вектором X_s. Определив вершину i входа для графа, можно решать задачу определения кратчайшего гамильтонового пути за один проход процедуры $\boldsymbol{A}_1$. При этом погрешность по сравнению с n-проходной процедурой, как будет показано ниже, увеличивается не более, чем на один процент, а сложность алгоритма понижается в (n–1) раз. Алгоритм с использованием минимально удаленной точки обозначим $\boldsymbol{A}_2$.

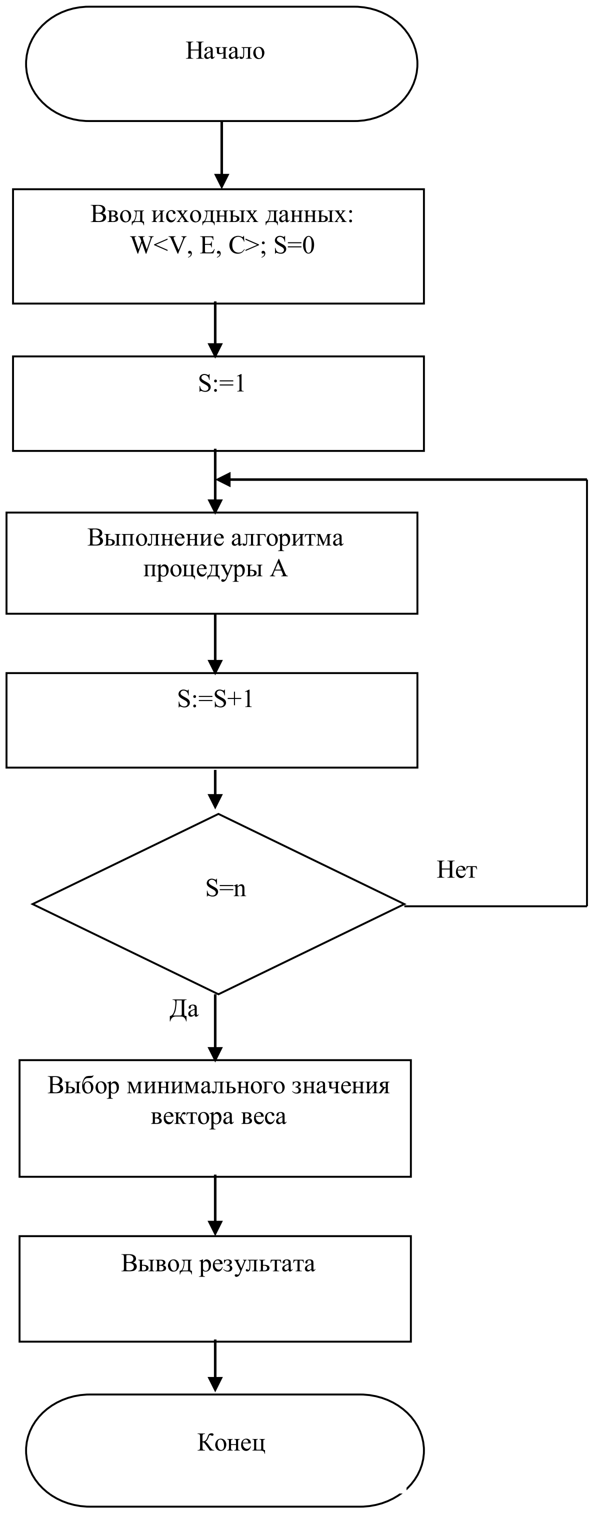

Рис. 5. Схема алгоритма ПКГП.

Результаты экспериментального исследования разработанных алгоритмов

Для получения аналитических результатов экспериментального исследования алгоритма решения задачи поиска кратчайшего гамильтонова пути разработана программная модель, реализованная в прикладной тестирующей программе с условным названием *Analizator*.

Прикладная тестирующая программа *Analizator* предназначена для проведения тестов разработанного алгоритма и обработки результатов тестирования. Исходя из основного назначения прикладной программы, предложено выделить три основных части программы, выполняющие функциональную нагрузку:

1. Область формирования исходных данных и настройки тестов;
2. Область анализа полученных результатов и их отображения на графиках;
3. Область отображения поранговых результатов тестирования и формирования записей в базе данных.

Использование указанных областей позволяет контролировать промежуточные значения реализации алгоритма, получать конечные результаты в виде графиков и таблиц базы данных. При этом дополнительно обеспечиваются следующие возможности:

- автоматический ввод расчетных данных;
- ручной ввод расчетных данных;
- отображение статистики;
- запись результатов в базу данных;
- простой и интуитивно доступный интерфейс.

При экспериментальном сравнении разработанных алгоритмов с известными веса ребер графа генерировались по равномерному закону распределения в диапазоне (0–50). Для получения среднего значения каждой точки графиков всех анализируемых характеристик решалось по 100 тестовых задач, все результаты статистического анализа получены с доверительной вероятностью 0.95. На всех рисунках разработанные алгоритмы $\boldsymbol{A_1}$ и $\boldsymbol{A_2}$ обозначены как метод 1 и метод 2.

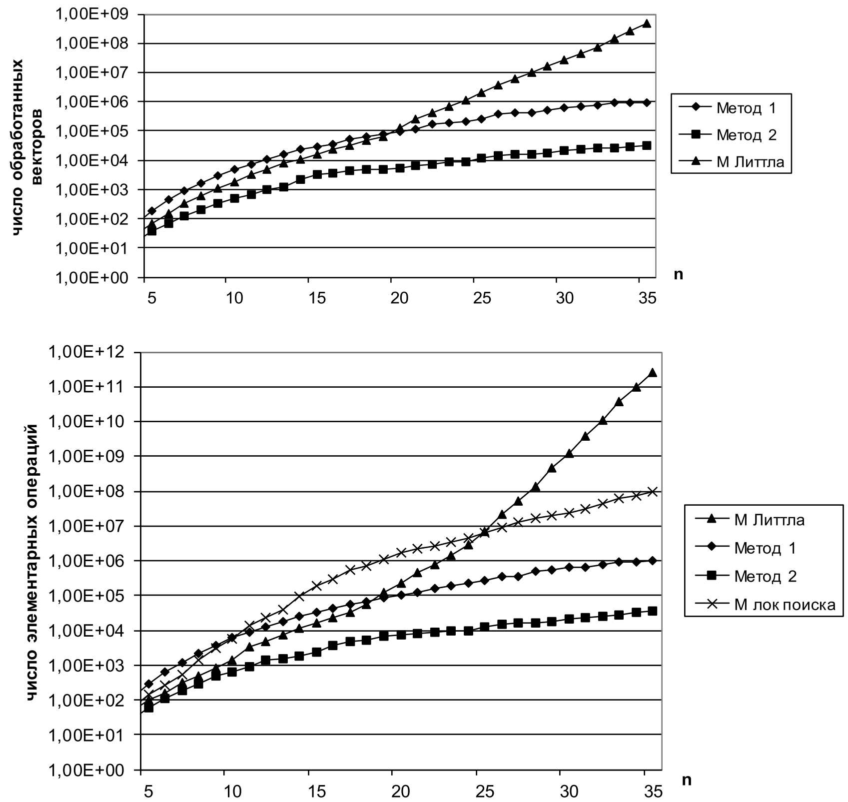

Рис. 6. Зависимости числа обрабатываемых векторов и элементарных операций от размерности решаемой задачи.

Как видно из графиков, приведенных на рис. 6, при $n \geq 27$ алгоритмы Литтла и локального поиска имеют существенно более высокую временную сложность по сравнению с разработанными.

При этом алгоритмы локального поиска имеют погрешность, лежащую в диапазоне от 5 до 27% и возрастающую с увеличением размерности решаемых задач. В то же время в разработанных алгоритмах на основе идей рангового подхода погрешность с увеличением размерности задачи стабилизируется и не превышает 2% (рис. 7).

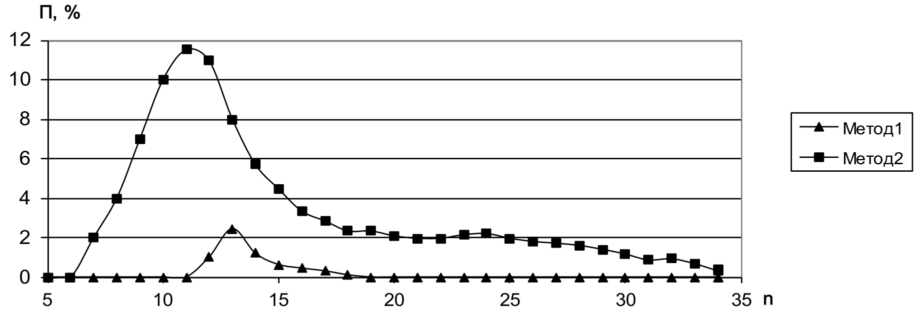

Рис. 7. Зависимость относительной погрешности от размерности решаемой задачи.

Из графиков, приведенных на рис. 8, видно, что процент неточных решений для алгоритма $\boldsymbol{A}_1$ при $n \geq 27$ очень низок.

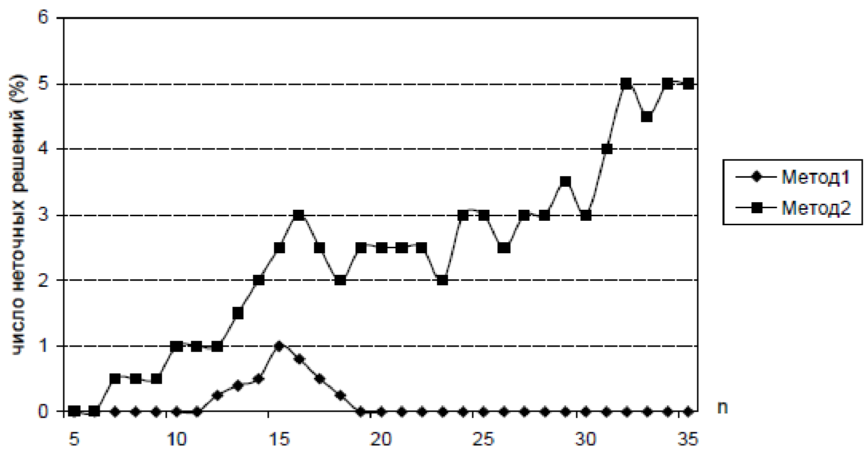

Рис. 8. Зависимость числа неточных решений (в процентах) от размерности решаемой задачи.

Экспериментальное исследование на 2000 тестовых задач показало, что при $n \geq 27$ в среднем только 3 давали приближенное решение, при этом погрешность не превышала 1-2%. Использование эвристического правила в алгоритме $\boldsymbol{A}_2$ позволило существенно уменьшить временную сложность этого алгоритма по сравнению с A_1, но при этом возрастает погрешность и число неточных решений с увеличением размерности решаемой задачи (см. рис. 8).

Заключение

Таким образом, предлагаемый способ решения задачи поиска кратчайшего гамильтонова пути может быть использован в сетевых базах данных для составления оптиматьных расписаний выполнения запросов к базам данных, как на этапах реорганизации, так и на этапах сопровождения функционирования СУБД. При этом предполагается существенное уменьшение времени на реализацию процедуры формирования очереди выполнения операций в запросах за счет уменьшения числа элементарных операций и числа обрабатываемых векторов в данной процедуре.

Из экспериментального исследования разработанного алгоритма решения задач поиска кратчайшего гамильтонова пути следует, что в большинстве случаев удается уменьшить временную сложность решения при незначительном увеличении вносимой погрешности.

Литература:

1. Аббасов М.Э. Методы оптимизации: Учеб. пособие. СПб.: Изд-во «ВВМ», 2014. 64 с.

2. Буй Д.Б., Скобелев В.Г. Модели, методы и алгоритмы оптимизации запросов в базах данных (обзор) // Радиоэлектронные и компьютерные системы. 2014. № 2 (66). С. 43–58. http://nbuv.gov.ua/UJRN/recs_2014_2_8

3. Бердников В.П. Модифицированный алгоритм определения полных областей устойчивости нестационарных нелинейных систем // Российский технологический журнал. 2018. Т. 6. № 3. С. 39–53. https://doi.org/10.32362/2500-316X-2018-6-3-39-53

4. Пастушков А.А., Батоврин В.К. Выбор решений при проектировании сложных систем на основе анализа вариантов со случайными весами // Российский технологический журнал. 2018. Т. 6. № 4. С. 78–88. https://doi.org/10.32362/2500-316X-2018-6-4-78-88

5. Горобец В.В. Облачная модель транзакционной системы // Вестник компьютерных и информационных технологий. 2013. № 4. С. 19–24.

6. Мендкович Н.А., Кузнецов С.Д. Обзор развития методов лексической оптимизации запросов // Труды Института системного программирования РАН. 2012. Т. 23. С. 195–214. https://doi.org/10.15514/ISPRAS-2012-23-12

7. Листровой С.В., Минухин С.В., Листровая Е.С. Разработка метода мониторинга распределенной вычислительной системы на основе определения кратчайших путей и кратчайших гамильтоновых циклов в графе // Восточно-Европейский журнал передовых технологий. 2015. Т. 6. № 4 (78). С. 32–45. https://doi.org/10.15587/1729-4061.2015.56247

8. Замбицкий Д.К., Лозовану Д.Д. Алгоритмы решения оптимизационных задач на сетях. Кишинев: Штиница, 1983. 116 с.

9. Фильгус Д.И., Андрианова Е.Г., Раев В.К. Развитие методов параллельных вычислений для фрагментации данных сетевой базы данных на основе рангового подхода // Cloud of Science. 2018. Т. 5. № 3. С. 532–550. URL: https://cloudofscience.ru/sites/default/files/pdf/CoS_5_532.pdf

10. Listrovoy S.V., Golubnichiy D.Y., Listrovaya E.S. Solution method on the basis of rank approach for integer linear programming problems with Boolean variables // Engineering Simulation. 1999. V. 16. № 6. P. 707–725.

11. Listrovoy S.V., Tretiak V.F., Listrovaya E.S. Parallel algorithms of calculation process optimization for the boolean programming problems // Engineering Simulation. 1999. V. 16. № 5. P. 569–579.

12. Федорин А.Н. Многокритериальные задачи ранцевого типа: разработка и сравнительный анализ алгоритмов: дис. ... канд. техн. наук. Нижний Новгород: Нижегородский государственный университет им. Н.И. Лобачевского, 2010. 132 с.

13. Лыфарь Д.А. Параллельные алгоритмы обработки реляционных баз данных // Вестник НГУ. Серия: Информационные технологии. 2010. Т. 8. Вып. 4. С. 72–80. URI: http://www.nsu.ru/xmlui/handle/nsu/307

14. Фраленко В.П., Агроник А.Ю. Средства, методы и алгоритмы эффективного распараллеливания вычислительной нагрузки в гетерогенных средах // Программные системы: теория и приложения. 2015. Т. 6. № 3(26). С. 73–92. https://doi.org/10.25209/2079-3316-2015-6-3-73-92

15. Жуков В.С. Исследование методов оптимального размещения базы данных по узлам вычислительной сети // Siberian Journal of Life Sciences and Agriculture. 2010. № 4 (10). С. 75–76.

References:

1. Abbasov M. Optimization Methods. St. Petersburg: «VVM» Publ., 2014. 64 p., (in Russ.).

2. Buy D.B., Skobelev V.G. Models, methods and algorithms for query optimization in databases (a survey). *Радіоелектронні і комп'ютерні системи* [Radioelectronic and Computer Systems]. 2014; 2 (66):43-58 (in Russ.). http://nbuv.gov.ua/UJRN/recs_2014_2_8

3. Berdnikov V.P. Modified algorithm for determination of full stability areas in nonstationary nonlinear systems. *Rossiiskii tekhnologicheskii zhurnal* = Russian Technological Journal. 2018; 6(3):39-53, (in Russ.). https://doi.org/10.32362/2500-316X-2018-6-3-39-53

4. Pastushkov A.A., Batovrin V.K. Selection of solutions for designing open systems based on analysis of variants with random weights. *Rossiiskii tekhnologicheskii zhurnal* = Russian Technological Journal. 2018; 6(4):78-88, (in Russ.). https://doi.org/10.32362/2500-316X-2018-6-4-78-88/

5. Gorobets V.V. Cloud model of on-line transaction processing system. *Vestnik komp'iuternykh i informatsionnykh tekhnologii* [Herald of Computer and Information Technologies]. 2013; 4:19-24, (in Russ.).

6. Mendkovich N.A., Kuznetsov S.D. Overview of evolution of lexical query optimization techniques. *Trudy Instituta sistemnogo programmirovaniya RAN* [Proceedings of the Institute for System Programming RAS]. 2012; 23: 195-214, (in Russ.) https://doi.org/10.15514/ISPRAS-2012-23-12

7. Listrovoy S.V., Minukhin S.V., Listrovaya E.S. Monitoring distributed computing systems on the basis of the determined shortest paths and shortest Hamiltonian cycles in a graph. *Eastern-European Journal of Enterprise technologies.* 2015; 6/4(78):32-4,. (in Russ.). https://doi.org/10.15587/1729-4061.2015.56247

8. Zambitsky D.K., Lozovanu D.D. Algorithms for solving optimization problems on networks. Chisinau: Shtinitsa Publ., 1983. 116 p., (in Russ.).

9. Filgus D.I., Andrianova E.G., Raev V.K. Development of parallel computing methods for fragmentation of network database data based on the rank approach. *Cloud of Science.* 2018; 5(3):532-559, (in Russ.). URL: https://cloudofscience.ru/sites/default/files/pdf/CoS_5_532.pdf

10. Listrovoy S.V., Golubnichiy D.Y, Listrovaya E.S. Solutioon Method for the linear programming problems with boolean variables. *Engineering Simulation.* 1999; 16(6): 707-725.

11. Listrovoy S.V., Tretiak V.F., Listrovaya E.S. Parallel algorithms of calculation process optimization for the boolean programming problems. *Engineering Simulation.* 1999; 16(5):569-579.

12. Fedorin A.N. Multi-criteria tasks of the backpack type: development and comparative analysis of algorithms: dis. ... Cand. of Sci. (Engineering). N.I. Lobachevsky Nizhny Novgorod: Nizhny Novgorod State University, 2010. 132 p., (in Russ.).

13. Lyfar D.A. Parallel GPU algorithms of relational databases processing. *Vestnik NGU. Seriya: Informatsionnye tekhnologii* [Herald of NSU. Series: Information Technologies]. 2010; 8(4):72-80, (in Russ.). URI: http://www.nsu.ru/xmlui/handle/nsu/307

14. Fralenko V.P., Agronik A.Yu. Tools, methods and algorithms for efficient parallelization of computational loading in heterogeneous environments. *Programmnye sistemy: teoriya i primenenie* [Program Systems: Theory and Applications]. 2015; 6(3(26)):73-92, (in Russ.). https://doi.org/10.25209/2079-3316-2015-6-3-73-92

15. Zhukov V.S. Study of methods for optimal placement of a database on nodes of a computer network. *Siberian Journal of Life Sciences and Agriculture*. 2010; 4(10):75-76, (in Russ.).

Об авторах:

Андрианова Елена Гельевна, кандидат технических наук, доцент кафедры корпоративных информационных систем Института информационных технологий ФГБОУ ВО «МИРЭА – Российский технологический университет» (Россия, 119454, Москва, пр. Вернадского, д. 78). Scopus author ID 57200555430, ResearcherIDS T-7908-2018, https://orcid.org/0000-0001-6418-6797

Раев Вячеслав Константинович, доктор технических наук, профессор кафедры инструментального и прикладного программного обеспечения Института информационных технологий ФГБОУ ВО «МИРЭА – Российский технологический университет» (Россия, 119454, Москва, пр. Вернадского, д. 78).

Фильгус Дмитрий Игоревич, аспирант кафедры инструментального и прикладного программного обеспечения Института информационных технологий ФГБОУ ВО «МИРЭА – Российский технологический университет» (Россия, 119454, Москва, пр. Вернадского, д. 78). https://orcid.org/0000-0001-6984-2933

About the authors:

Elena G. Andrianova, Cand. of Sci. (Engineering), Associate Professor of the Chair of Corporate Information Systems, Institute of Information Technology, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow 119454, Russia). Scopus author ID 57200555430, ResearcherID T-7908-2018, https://orcid.org/0000-0001-6418-6797

Vyacheslav K. Raev, Dr. of Sci. (Engineering), Professor of the Chair of Instrumental and Applied Software, Institute of Information Technology, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow 119454, Russia).

Dmitry I. Filgus, Postgraduate Student of the Chair of Instrumental and Applied Software, Institute of Information Technology, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow 119454, Russia). https://orcid.org/0000-0001-6984-2933

Для цитирования: Андрианова Е.Г., Раев В.К., Фильгус Д.И. Определение кратчайших гамильтоновых путей в произвольных графах распределенных баз данных // Российский технологический журнал. 2019. Т. 7. № 4. С. 7–20. https://doi.org/10.32362/2500-316X-2019-7-4-7-20

For citation: Andrianova E.G., Raev V.R., Filgus D.I. Determination of the shortest Hamiltonian paths in an arbitrary graph of distributed databases. *Rossiiskii tekhnologicheskii zhurnal* = Russian Technological Journal. 2019; 7(4):7-20, (in Russ.). https://doi.org/10.32362/2500-316X-2019-7-4-7-20