DOI:

МЕТОД ВЕРТИКАЛЬНОЙ ФРАГМЕНТАЦИИ ТАБЛИЦ ДАННЫХ И РАЗМЕЩЕНИЕ ФРАГМЕНТОВ В РАСПРЕДЕЛЕННЫХ СИСТЕМАХ

[1]Микрин Е.А.,[2]Кульба В.,[2]Сомов С.К.

[1] ПАО «Ракетно-космическая корпорация «Энергия» им. С.П. Королева»,
г. Королев, ул. Ленина, 4а
Eugeny.Mikrin@rsce.ru
[2]Институт проблем управления им. В.А. Трапезникова РАН, Россия, г. Москва
ул. Профсоюзная д.65, стр.1.
ssomov2016@ipu.ru

Аннотация: В работе представлен метод и формальная модель вертикального фрагментирования таблиц баз данных, используемых в распределенных системах различного класса и назначения. Представлен процесс проектирования и синтеза логической структуры фрагментов таблиц, основанный на анализе предметных областей пользователей и множества запросов пользователей. Сформулирована задача синтеза логической структуры фрагментов таблиц данных. Предложен эвристический алгоритм размещения в узлах сети полученных вертикальных фрагментов.

Ключевые слова: распределенные системы обработки данных, вертикальная фрагментация таблиц данных, распределенные базы данных.

Введение

За несколько последних лет было выполнено множество исследований, посвященных разработке эффективных методов улучшения характеристик распределенных систем обработки данных (РСОД) и систем управления распределенными базами данных (РБД). Основной акцент в данных работах уделялся таким наиболее важным характеристикам систем как высокая надежность работы, приемлемое время доступа к данным, высокая производительность систем, минимизация затрат на функционирование и высокий уровень сохранности информации, используемой в системе.

Из-за огромного объема накопленных цифровых данных, хранящихся в базах данных, одна централизованная база данных не может обеспечивать хорошую производительность и доступность данных, которые используются большим количеством пользователей. Поэтому использование в РСОД распределенных баз данных является эффективным методом для преодоления проблемы производительности и доступности данных путем фрагментации таблиц базы данных и размещения фрагментов в правильно выбранных узлах сети. Во многих исследованиях представлены разнообразные алгоритмы оптимизации фрагментации таблиц, распределения и репликации фрагментов таблиц на начальном этапе проектирования распределенной базы данных с использованием различных методов, которые влияют на производительность РБД [1,2].

Проектирование и создание распределенных систем это масштабный, ресурсоемкий и сложный проект, в ходе которого решается множество важных задач, в списке которых одно из первых мест занимает задача обеспечения высокой производительности системы.

Для обеспечения высокопроизводительной работы распределенных систем используется несколько методов. В частности, это использование качественных и быстродействующих аппаратных средств (каналы связи с большой пропускной способностью и высоким уровнем надежности, высокопроизводительные компьютеры, быстродействующие и надежные устройства хранения информации большой емкости с возможностями восстановления информации в случае ее повреждения). Еще одним направлением повышения быстродействия и надежности работы распределенных систем являются эффективные методы проектирования оптимальной логической и физической структуры распределенных баз данных, которые используются распределенными системами. Наконец, это использование различных методов поиска оптимального размещения таблиц и фрагментов таблиц распределенных баз данных в узлах компьютерной сети, за счет чего обеспечивается максимальное приближение необходимых данных к их потребителям - прикладным процессам, обрабатывающих запросы пользователей системы к информации, хранящейся в таблицах базы данных.

В последнем случае для приближения данных к пользователям могут использоваться следующие методы [1-5]:

- репликация массивов данных, таблиц распределенных баз данных;
- распределение по нескольким узлам сети горизонтальных или вертикальных фрагментов таблиц распределенных баз данных.

В результате использования перечисленных методов удается увеличить не только производительность системы, но и надежность ее работы, так как при сбое обработки запроса пользователя в одном узле системы запрос можно переадресовать в другой работоспособный узел, имеющий идентичную реплику или фрагмент таблицы данных.

Большое количество исследований, выполненных за последние годы в области распределенных систем, проводилось в следующих направлениях [1-4]:

- Фрагментирование таблиц данных РБД:
- горизонтальное фрагментирование, при котором множество строк таблицы делится на несколько непересекающихся подмножеств строк (фрагментов);
- вертикальное фрагментирование, когда множество столбцов таблицы делится на несколько подмножеств столбцов (вертикальных фрагментов);
- смешанное (гибридное) фрагментирование, при котором используется комбинация первых двух методов фрагментирования.
- Размещение таблиц базы данных или их фрагментов:
- неизбыточное размещение данных (таблиц данных или их фрагментов), при котором один экземпляр таблицы или фрагмента размещается в одном из узлов распределенной системы;
- избыточное размещение, при котором в нескольких узлах распределенной системы размещается соответствующее количество идентичных копий (реплик) таблиц или их фрагментов;
- Репликация данных (используется при избыточном размещении данных):
- использование алгоритмов создания и удаления реплик для оптимизации работы системы;
- миграция реплик из одного узла в другой узел системы;
- использование методов обеспечения идентичности и непротиворечивости данных
- в репликах одних таблиц и их фрагментов (согласованность реплик).

За несколько предыдущих лет было опубликовано много исследований, посвященных проблемам и методам повышения производительности РБД [1-5]. Значительная часть этих работ касаются вопросов фрагментации таблиц баз данных и распределения фрагментов по узлам сети распределенной системы.

Фрагментация таблиц данных это сложная процедура, которая, как правило, применяется только для баз данных с очень большими объемами данных. К наиболее распространенным ситуациям, при которых использование фрагментации целесообразно, можно отнести следующие:

- прикладные процессы, работающие с базой данных, настолько интенсивно заполняют таблицы новыми записями, что прогнозируется переполнение таблиц баз данных, расположенных в одном из узлов сети;
- трафик операций записи, чтения и модификации данных интенсивно увеличивается, что приводит к недопустимому увеличению времени отклика со стороны базы данных;
- пропускная способность сети не удовлетворяет требованиям приложений к доступу данных, что приводит к увеличению времени ожидания запросов в очереди на обработку.

Основными преимуществами использования методов фрагментации являются следующие:

1) Фрагментация позволяет упростить масштабирование распределенных информационных систем, т.е. добавление новых компьютеров к существующей архитектуре распределенной системы. Это в свою очередь позволяет распределить нагрузку между большим количеством компьютеров и быстрее обрабатывать больший объем трафика запросов к данным.
2) Использование нескольких фрагментов таблиц данных позволяет сократить время обработки запросов, поскольку во фрагментированной базе данных для получения ответа на запрос требуется просмотреть меньшее количество записей, чем в исходной таблице, не разбитой на несколько фрагментов. Кроме того, размещение фрагментов одной таблицы на нескольких компьютерах системы позволяет выполнять параллельную обработку нескольких запросов к разным сегментам одной таблицы. Такая технология работы с запросами также позволяет сократить время обработки запросов.
3) Использование фрагментов таблиц данных повышает надежность работы распределенных систем. Если система будет использовать монолитную базу данных, то любой серьезный сбой в базе данных приведет к остановке всей системы. При использовании системой фрагментированной базы данный сбой в одном из фрагментов приведет к неработоспособности только некоторых частей системы. Общее воздействие сбоя в одном из фрагментов окажет на систему гораздо меньшее негативное влияние, чем в случае сбоя в монолитной базе данных.

В данной работе рассматривается комплексный метод проектирования логической структуры фрагментов таблиц данных совместно с синтезом программных модулей, используемых в прикладных процессах, использующих данные из фрагментов таблиц. Данный метод является модификацией подхода, который был предложен для проектирования и создания информационно-технологического резерва в рамках распределенных систем обработки данных [5,6]. Метод основывается на результатах анализа предметных областей пользователей.

Предлагаемый в работе метод проектирования логической структуры фрагментов таблиц и синтеза программных модулей состоит из трех этапов.

1 Этап 1. Анализ предметных областей пользователей.

На первом этапе проводится анализ предметных областей пользователей распределенной системы. При этом уточняются входные, промежуточные и итоговые данные таблиц, перечень процедур обработки этих данных, а также последовательности выполнения процедур в процессе обработки запросов пользователей. Каждый запрос пользователей должен быть отнесен к определенному типу запросов, который характеризуется своим набором используемых данных таблиц и множеством процедур их обработки.

Для выполнения работ на этапе анализа предметных областей пользователей целесообразно использовать получившие широкое распространение формальные модели и методы, разработанные в Институте проблем управления РАН [7-9]. Эти модели и методы представляют собой совокупность взаимосвязанных матричных и графовых моделей, которые используются на этапе анализа предметных областей пользователей распределенных систем обработки данных и РБД. В результате работ на этом этапе определяется перечень и типы входных, промежуточных и выходных данных, а также множество процедур обработки информации, которые используются при обработке запросов к данным системы. Для представления информационных связей между различными процедурами обработки данных используется формат мультиграфа. При этом процедуры обработки данных это вершины данного графа, а дуги, которые связывают эти вершины, раскрашены множеством информационных элементов, общих для смежных процедур.

Задача анализа предметных областей пользователей в графовой интерпретации заключается в разбиении полученного мультаграфа на множество подграфов, имеющих минимальное количество связей между этими подграфами [7].

2 Этап 2. Синтез программных модулей и логической структуры фрагментов

На данном этапе на основе данных, полученных в ходе выполнения первого этапа, проводится синтез логической структуры фрагментов таблиц и прикладного программного обеспечения, использующего данные этих фрагментов при обработке запросов пользователей. Синтез программного обеспечения производится с учетом принципа модульности. Данный принцип позволяет оптимизировать перечень и взаимосвязи между отдельными компонентами прикладного программного обеспечения и фрагментами таблиц данных распределенной системы [9].

В работе [10] дано определение ряда понятий, которые используются для формализации и описания основных положений процесса анализа и синтеза модульных систем обработки данных. С использованием данных понятий далее будем использовать следующие определения:

Запрос – требование (заявка) пользователя или прикладного процесса на обработку данных фрагментов таблиц. Каждый запрос характеризуется определенным набором обрабатываемых данных, перечнем выходных данных, полученных в результате обработки данных, и ограничением на время, затраченное на обработку данных.

Тип запросов – множество запросов, характеризующееся одинаковыми наборами обрабатываемых данных (перечнем используемых фрагментов таблиц) и множеством процедур их обработки данных. К одному типу запросов относятся запросы, для обработки которых используются одинаковый набор данных (фрагментов таблиц) и одинаковое множество процедур их обработки, которые выполняются в РСОД в одинаковой очередности.

Информационный элемент (ИЭ) – атрибут записи фрагмента таблицы - наименьшая, неделимая часть данных, по отношению к которой допускается ее независимое использование в процессе обработки запросов к данным.

Процедура обработки данных – алгоритмическое действие по преобразованию одного атрибута записи фрагмента или группы атрибутов в другой атрибут или группу атрибутов в соответствии с логикой обработки запросов.

Задача РСОД – набор процедур и обрабатываемых ими информационных элементов, необходимых для обслуживания запросов одного или нескольких типов.

При решении задач оптимизации состава фрагментов таблиц, как правило, используются следующие основные критерии оптимизации:

- максимум информационной производительности системы при обработке запросов,
- минимум времени обработки запросов,
- минимум сложности интерфейса между отдельными модулями,
- минимум времени обмена данными между оперативной и внешней памятью в процессе обработки запросов,
- минимум объема неиспользуемых данных при обмене данными между оперативной и внешней памятью серверов системы.

В качестве исходных данных для синтеза логической структуры фрагментов таблиц используются результаты анализа предметных областей пользователей, которые были получены на первом этапе процесса.

На основе этих данных определяются следующие параметры:

- множество задач РСОД по обработке запросов;
- множество процедур обработки данных, используемых при решении множества задач РСОД;
- включение процедур обработки данных в состав тех или иных программных модулей;
- множество атрибутов записей фрагментов таблиц, связанных с процедурами обработки данных и классифицируемых по следующим типам: – входной, промежуточный или выходной элемент данных;
- вхождение атрибутов в состав фрагментов таблиц;
- варианты взаимодействия процедур обработки данных с атрибутами фрагментов;
- технологическая матрица смежности;
- множество допустимых последовательностей выполнения процедур;
- характеристики процедур и атрибутов фрагментов таблиц.

3 Этап 3. Размещение фрагментов таблиц по узлам сети.

Основная задача данного этапа заключается в поиске оптимального распределения фрагментов таблиц по серверам РСОД. Цель оптимального размещения фрагментов - повышение производительности и надежности функционирования РСОД при снижении затрат на ее функционирование. В качестве критериев оптимальности размещения фрагментов таблиц применяют такие наиболее часто используемые критерии, как минимум стоимости функционирования системы или минимум среднего времени обработки запросов пользователей РСОД.

Формальные модели и методы анализа предметных областей пользователей детально представлены в цитированной выше литературе, поэтому более подробно рассмотрим работы второго и третьего этапа процесса проектирования и размещения фрагментов таблиц в узлах РСОД.

4 Синтез программных модулей и логической структуры фрагментов таблиц.

В процессе формализации постановки задачи синтеза программных модулей и структуры фрагментов таблиц будем использовать следующие обозначения:

- $A = \{a_1, a_2, \dots, a_r, \dots, a_R\}$ - множество процедур обработки данных;
- $D = \{d_1, d_2, \dots, d_l, \dots, d_L\}$ - множество информационных элементов (атрибутов записей фрагментов), обрабатываемых процедурами из множества A;
- $W = \|\omega_{rl}\|$ - технологическая матрица смежности. Матрица отражает взаимосвязи атрибутов записей фрагментов таблиц с процедурами их обработки. В этой матрице:

$$\boldsymbol{\omega_{rl}} = \begin{cases} +\mathbf{1}, \text{если } \boldsymbol{l} - \text{й атрибут является исходным для } \boldsymbol{r} - \text{й процедуры;} \\ -\mathbf{1}, \text{если } \boldsymbol{l} - \text{й атрибут является результатом работы } \boldsymbol{r} - \text{й процедуры;} \\ \mathbf{0}, \text{если } \boldsymbol{l} - \text{й атрибут не используется в } \boldsymbol{r} - \text{й процедуре.} \end{cases}$$

- $W^c = \|\omega_{rl}^c\|$ и $W^3 = \|\omega_{rl}^3\|$ - соответственно матрицы взаимосвязей атрибутов записей фрагментов с процедурами при считывании и записи, где

$$\boldsymbol{\omega_{rl}^c} = \begin{cases} \mathbf{1}, \text{если } \boldsymbol{l} - \text{й атрибут считывается (записывается) } \boldsymbol{r} - \text{й процедурой} \\ \mathbf{0} \text{ в противном случае (не используется в } \boldsymbol{r} - \text{й процедуре)} \end{cases}$$

- $\Omega(A)$ – множество допустимых последовательностей исполнения процедур при обработке запросов, которое определено на множестве процедур A.

При этом:

$w \in \Omega(A)$ - это конкретная последовательность выполнения процедур обработки данных.

В дополнение к множеству $\Omega(A)$ определим в виде матрицы P множество всех возможных последовательностей выполнения процедур:

$$\boldsymbol{P} = \|\boldsymbol{p_{r\theta}}\|, \text{где}$$

$$\boldsymbol{p_{r\theta}} = \begin{cases} \mathbf{1}, \text{если процедура } \boldsymbol{r} \text{ может быть выполнена } \boldsymbol{\theta} - \text{й в заданной} \\ \qquad\qquad \text{последовательности реализации процедур при обработке данных,} \\ \mathbf{0}, \text{в противном случае} \end{cases}$$

Также определим следующие усредненные характеристики серверов сети:

- τ_v - среднее время считывания *v*-го программного модуля из внешней памяти в оперативную память сервера;
- t_f^c- это среднее время считывания *f*-го фрагмента данных из внешней памяти в оперативную память сервера;
- $t_f^з$ - среднее время записи результатов обработки данных в *f*-й фрагмент;

В общем случае, компьютерные сети, на основе которых работают распределенные системы, являются неоднородными сетями. Следовательно, компьютеры в узлах сети скорее всего имеют различные характеристики процессоров, оперативной и внешней памяти. Поэтому для величин $(\tau_v, t_f^c, t_f^з)$ будем использовать усредненные оценки на множестве всех серверов сети.

Для оценки значений перечисленных выше величин можно использовать методы, представленные в работах [9,11,12].

Введем следующие переменные:

$$\boldsymbol{y_{vl}^{c(з)}} = \begin{cases} \mathbf{1}, \text{если } \sum_{r=1}^{R} \boldsymbol{\omega_{rl}^{c(з)} x_{rv}} \geq \mathbf{1}, \\ \mathbf{0}, \text{если } \sum_{r=1}^{R} \boldsymbol{\omega_{rl}^{c(з)} x_{rv}} \geq \mathbf{0}; \end{cases}$$

$$\boldsymbol{z_{vl}^{c(з)}} = \begin{cases} \mathbf{1}, \text{если } \sum_{l=1}^{L} \boldsymbol{y_{vl}^{c(з)} z_{lf}} \geq \mathbf{1}, \\ \mathbf{0}, \text{если } \sum_{l=1}^{L} \boldsymbol{y_{vl}^{c(з)} z_{lf}} \geq \mathbf{0}. \end{cases}$$

Где:

$$\boldsymbol{x_{rv}} = \begin{cases} \mathbf{1}, \text{если } \boldsymbol{r} - \text{я по порядку выполнения процедура} \\ \quad \text{включается в состав } \boldsymbol{v} - \text{го модуля,} \\ \mathbf{0}, \text{в противном случае.} \end{cases}$$

$$\boldsymbol{z_{lf}} = \begin{cases} \mathbf{1}, \text{если } \boldsymbol{l} - \text{й атрибут включается в состав } \boldsymbol{f} - \text{го фрагмента, } \boldsymbol{f} = \overline{\mathbf{1}, \boldsymbol{F}}; \boldsymbol{F} \leq \boldsymbol{L} \\ \mathbf{0}, \text{в противном случае} \end{cases}$$

Определенные выше переменные $y_{vl}^{c(з)}$ и $z_{vl}^{c(з)}$ служат для формализации взаимосвязи программных модулей с атрибутами и фрагментами таблиц при считывании/записи данных в процессе обмена с внешней памятью серверов.

С использованием определенных выше переменных можно сформулировать задачи поиска оптимальной структуры фрагментов таблиц для заданной предметной области и множества запросов пользователей РСОД [8,10-12].

В частности, задача синтеза системы программных модулей и обрабатываемых этими модулями фрагментов таблиц с использованием в качестве критерия оптимизации минимального общего времени обмена с внешней памятью серверов при использовании фрагментов таблиц для обработки запросов, имеет следующую формулировку:

$$(1)\ \min_{x_{rv}, z_{lf}} \left\{ \sum_{r=1}^{R} \sum_{v=1}^{V} x_{rv}\left(1 - x_{r+1,v}\right) \left[\tau_v + \sum_{f=1}^{F} \left(z_{vf}^c t_f^c + z_{vf}^з t_f^з \right) \right] \right\}$$

С использованием следующих ограничений:

- На общее число процедур обработки данных в составе каждого модуля.

$$(2)\ \sum_{r=1}^{R} x_{rv} \le \bar{M}, \qquad v = 1,2,\dots,V,$$

где $\bar{M}$ – это максимальное количество процедур, которое может входить в состав одного модуля.

- На число атрибутов записей фрагментов, обрабатываемых процедурами каждого модуля.

$$(3)\ \sum_{l=1}^{L} y_{vl} \le \bar{L}, v = 1,2,\dots,V$$

Здесь $\bar{L}$ – это максимальное количество атрибутов, обрабатываемых v -м модулем.

- На сложность взаимодействия между отдельными модулями

$$(4)\ \sum_{l=1}^{L}\sum_{v=1}^{V-1}\sum_{v'=v+1}^{V} y_{vl}y_{v'l} \le \bar{S},$$

где $\bar{S}$ – это максимально допустимая мощность информационного интерфейса между модулями, т.е. максимальное допустимое число переменных, общих для программных модулей.

- На сложность информационного интерфейса между отдельными парами модулей

$$(5)\ \sum_{l=1}^{L} y_{vl}y_{v'l} \le S_{vv'},$$

Здесь $S_{vv'}$ – это максимально допустимое число общих атрибутов фрагментов таблиц, обрабатываемых модулями v и v'.

- На однократность включения процедур в программные модули.

$$(6)\ \sum_{v=1}^{V} x_{rv} = 1, r = 1,2,\dots,R$$

- На включение отдельных процедур в состав одного модуля.

$$(7)\ x_{rv} + x_{r'v} \le 1$$

для конкретных процедур α_r и $\alpha_{r'}$ при $v = 1,2,\dots,V$.

- На передачу управления из программного модуля до завершения обработки данных всеми процедурами, входящими в состав данного модуля.

$$(8)\ \sum_{r=1}^{R} x_{rv}\left(1 - x_{r+1,r}\right) = 1, \qquad v = 1,2,\dots,V$$

- На дублирование атрибутов во фрагментах таблиц.

$$(9)\ \sum_{f=1}^{F} z_{lf} = k, \qquad k = 1,2,\dots,K$$

где K – допустимая степень дублирования атрибутов в записях фрагментов таблиц.

- На размер записи каждого фрагмента таблиц.

$$(10)\ \sum_{l=1}^{L} z_{lf} \le N, \qquad f = 1,2,\dots,F$$

Здесь N – максимально допустимое число атрибутов в записи фрагмента.

Если существуют затруднения в получении оценок временных характеристик серверов сети с программными модулями и фрагментами таблиц, то тогда возможна постановка сформулированной выше задачи по критерию обеспечения минимального числа обращений к внешней памяти серверов при обработке запросов пользователей к данным фрагментов таблиц. Такая задача является вариантом задачи (1) – (10), использующим другой критерий оптимизации и имеет следующую формулировку:

$$(11) \quad \min_{(x_{rv} z_{lf})} \left\{ \sum_{r=1}^{R} \sum_{v=1}^{V} x_{rv}(1 - x_{r+1,v}) \left[1 + \sum_{f=1}^{F} (z_{vf}^{c} + z_{vf}^{3}) \right] \right\}$$

при ограничениях(2) – (10).

Рассмотренные задачи (1-11) синтеза модульной структуры программного обеспечения и логической структуры вертикальных фрагментов таблиц данных это нелинейные задачи целочисленного программирования комбинаторного типа. Для решения данных задач целесообразно использовать алгоритмы, приведенные в работе [12].

5 Размещение фрагментов таблиц по узлам сети.

Для решения задачи размещения фрагментов таблиц по узлам сети предлагается использовать представленную ниже формальную модель и эвристический алгоритм перераспределения фрагментов таблиц в зависимости от текущих параметров распределенной системы.

Предполагается, что РСОД использует распределенную базу данных, функционирующую на базе компьютерной сети из K узлов: $N = \{N_1, \dots N_k \dots, N_K\}$. Топология сети задана взвешенным графом $G = (K, \Gamma)$. Задана матрица $CoC = \{CoC_{ij}\}, (i, j = \overline{1, K})$ стоимости передачи единицы данных между узлами сети. На ее основе построена матрица $DTC = \{DTC_{ij}\}, \ i, j = \overline{1, K}$ - затрат на передачу запросов по кратчайшим путям между узлами сети. Согласно изложенной выше методике получено множество из *M* фрагментов $F = \{F_1, \dots F_m, \dots, F_M\}$. Для каждого фрагмента известна его мощность $Crd(F_m)$ и длина $Len(F_m)$ его кортежей, равная сумме длин атрибутов кортежа. Размещение фрагментов таблиц по узлам сети описывается матрицей $Y = \{y_{mk}\}$, в которой $y_{mk} = 1$, если фрагмент *m* размещен в узле *k*, иначе $y_{mk} = 0$. Используется неизбыточное размещение, т.е. один фрагмент может быть размещен только в одном узле сети. Задан кортеж *DSC(k),*), элементы которого задают стоимость хранения единицы данных в узлах. Задано множество прикладных процессов $P = \{P_p, p = 1, \dots, P\}$. При работе каждого процесса в узле сети генерируются запросы к фрагментам таблиц (информационные и запросы на модификацию данных).

Элементы матрицы $PF = \|PF_{kp}\|$ $(k = \overline{1, K};\ p = \overline{1, P})$ определяют частоту решения процессов в узлах сети. Где pf_{kp} - это частота решения *p*-го процесса в *k*-м узле сети. Элементы матрицы $PQE = \|PQE_{pm}\|$, определяют частоту генерации *p*-м процессом информационных запросов к фрагменту *m*. Аналогично элементы матрицы $PQU = \|PQU_{pm}\|$ задают частоту генерации *p*-м процессом запросов на модификацию данных фрагмента *m*. Предположим, что в системе бесконечные очереди на обслуживание запросов, система работает в установившемся режиме, без сбоев, а все запросы обрабатываются успешно.

Считаем, что задан кортеж $T^E = (T_1^E, , , , T_n^E, \dots, T_N^E)$ средних времен обработки информационных запросов в узлах сети, и кортеж $S^E = (S_1^E, , , , S_n^E, \dots, S_N^E)$ со средней стоимостью единицы времени обработки запросов в узлах сети. Так же определены аналогичные кортежи для запросов на модификацию данных: $T^U = (T_1^U, , , , T_n^U, \dots, T_N^U)$ и $S^U = (S_1^U, , , , S_n^U, \dots, S_N^U)$.

В модели используется параметр «вес» $FW(N_k, F_m)$ фрагмента *m* в узле *k* сети. Значение данного параметра зависит от суммы *ACSR* средних затрат на передачу запросов по каналам связи, средних затрат *ACPR* на обработку запросов и затрат *FSC* на хранение фрагментов в узлах сети:

$$FW(k, m) = ACSR(k, m) + ACPR(k, m) - FSC(k, m), \quad 1 \le k \le K, 1 \le m \le M.$$

Фактически вес узла $FW(N_k, F_m)$ это сумма, на которую уменьшатся или увеличатся затраты на эксплуатацию системы, если фрагмент F_m будет размещен в узле N_k вместо его размещения в любом другом узле сети.

Сумма затрат *SOC* на функционирование распределенной системы с размещенными в узлах системы фрагментами таблиц данных складывается из следующих трех компонент.

1). Затраты *TSTF* на хранение фрагментов. Определяются с использованием данных матрицы *FSC(k,m)* стоимости хранения фрагментов в узлах сети и матрицы $Y(m, k)$ распределения фрагментов по узлам сети, элемент которой $y_{mn} = 1$, если фрагмент F_m размещен в узле N_k, и равен *0* в противном случае:

$$TSTF = \sum_{m=1}^{M} \sum_{k=1}^{K} y_{mk} \times FSC(k,m)$$

2). Затраты *CTR* на передачу всех запросов, генерируемых во всех узлах системы за единицу времени:

$$CTR = \sum_{k=1}^{K} \sum_{p=1}^{P} PF(k,p) \sum_{m=1}^{M} (PQE(p,m) + PQU(p,m)) \times \sum_{j=1}^{K} y_{mj} DTC(k,j)$$

3). Затраты *RPC* на обработку запросов в узлах с фрагментами данных:

$$RPC = \sum_{j=1}^{K} \sum_{p=1}^{P} PF(j,p) \left(\sum_{m=1}^{M} PQE(p,m) \sum_{k=1}^{K} y_{mk} S^{E}(k,m) + \sum_{m=1}^{M} PQU(p,m) \sum_{k=1}^{K} y_{mk} S^{U}(k,m) \right)$$

Таким образом:

$$SOC = TSTF + CTR + RPC$$

Среднее время *AvRPT* обработки запроса в системе равно результату деления времени *ToRPT* обработки в системе всех запросов, сгенерированных в единицу времени, на общее количество *ToNR* этих запросов*:*

$$AvRPT = ToRPT \,/ ToNR$$

Где:

$$ToRPT = \sum_{k=1}^{K} \sum_{p=1}^{P} PF(k,p) \left(\sum_{m=1}^{M} PQE(p,m) \sum_{j=1}^{K} y_{mj} T^{E}(j,m) + \sum_{m=1}^{M} PQU(p,m) \sum_{j=1}^{K} y_{mj} T^{U}(j,m) \right)$$

$$ToNR = \sum_{k=1}^{K} \sum_{p=1}^{P} PF(k,p) \times \sum_{m=1}^{M} [PQE(p,m) + PQU(p,m)]$$

Для решения задачи размещения фрагментов таблиц необходимо найти такое распределение фрагментов *Y* по узлам распределенной системы, которое обеспечивало бы минимум затрат *OSC* на функционирование системы:

$$\min_{Y} OSC = \min_{Y}(TSTF + CTR + RPC)$$

Найденное решение задачи должно удовлетворять следующим ограничениям:

1) на объем всех фрагментов, размещенных в одном узле сети;

2) на количество фрагментов, размещенных в одном узле;

3) размещение фрагментов должно быть неизбыточным, т.е. один экземпляр фрагмента должен быть размещен в одном из узлов сети;

4) среднее время обработки запроса к фрагментам не должно превышать установленный лимит времени для всей системы.

Для решения сформулированной выше задачи предлагается использовать следующий эвристический алгоритм.

Шаг 0. Присваиваем переменной OSC максимально возможное значение.

Шаг 1. Вычисляем сумму частот всех запросов, генерируемых во всех узлах сети к фрагменту m. Результат запоминаем в соответствующем элементе кортежа FQ(m).

Шаг 2. Заполняем элементы кортежа FDS номерами фрагментов, отсортированных в порядке убывания количества запросов к фрагменту из кортежа FQ. Порядок номеров фрагментов в FDS будет определять очередность распределения фрагментов по узлам сети.

Шаг 3. Вычисляем для всех фрагментов и для всех узлов сети значения элементов матрицы » $FW(N_k, F_m)$ с весами фрагментов.

Шаг 4. Распределяем фрагменты F_m по узлам сети последовательно, в цикле по отсортированным номерам фрагментов, сохраненным в кортеже FDS.

Шаг 4.1. Для размещения очередного фрагмента m выбираем узел k сети с наибольшим значением веса FW(k,m). Присваиваем значение 1 соответствующему элементу матрицы Y(m,k), задающей распределение фрагментов по узлам сети.

Шаг 4.2. Проверяем выполнение ограничений задачи.

Если все ограничения выполнены, переходим к поиску узла сети для размещения очередного фрагмента – переход к Шагу 4.

Если одно или несколько ограничений не выполнено, то пробуем разместить текущий фрагмент m в другом узле, имеющим меньшее значение веса узла – переход к Шагу 4.1.

Конец работы алгоритма.

Данный алгоритм реализован на языке C++ в среде MS Visual Studio 2017.

Работа выполнена при финансовой поддержке РФФИ в рамках научного проекта 18-29-16151 «Разработка методов управления процессами трансформации права в условиях цифровой технологии».

Литература

1. *Fuaad H. and oth.* A Survey on Distributed Databases Fragmentation, Allocation and Replication Algorithms// Current Journal of Applied Science and Technology. —2018. —V.27. —No.2. — P.12.
2. *Чернышев Г.А.* Обзор подходов к организации физического уровня в СУБД // Труды СПИИРАН. 2013. – Санкт-Петербург. — 2013. — Вып. 1(24). — С. 222 — 275.
3. *Gupta, S. and Panda, S.* Vertical Fragmentation, Allocation and Re-Fragmentation in Distributed Object Relational Database Systems-(Update Queries Included)//International Journal of Engineering Research and Development.— 2012 .— V.4. — P. 45-52.
4. *Suganya A. and Kalaiselvi, R.* Efficient Fragmentation and Allocation in Distributed Database// International Journal of Engineering Research & Technology (IJERT). 2013—V— 2, P.— 1-7.
5. *Сомов С.К.* Сохранность информации в распределенных системах обработки данных. – М.: ИПУ РАН, .— 2019. – 254 с.
6. *Сомов С.К.* Создание информационно-технологического резерва в распределенных системах обработки данных / Проблемы управления.— 2018.— №3.— С. 40-46.
7. Теоретические основы проектирования оптимальных структур распределенных баз данных / В. В. Кульба, С. С. Ковалевский, С. А. Косяченко, В. О. Сиротюк; Рос. акад. наук, Ин-т проблем упр., — М.:СИНТЕГ. — 1999. — 660 с.
8. *Кузнецов Н.А., Кульба В.В., Ковалевский С.С.* Методы анализа и синтеза модульных информационно-управляющих систем. — М.: ФИЗМАТЛИТ, — 2002. — 800 с.
9. *Мамиконов А.Г, Кульба В.В., Косяченко С.А.* Типизация разработки модульных систем обработки данных. — М.:Наука, — 1989. — 163 с.
10. *Теоретические основы проектирования* информационно-управляющих систем космических аппаратов /В.В. Кульба, Е.А. Микрин, Б.В. Павлов, В.Н. Платонов; под ред. Е.А. Микрина ; Ин-т проблем упр. Им. Трапезникова РАН. — М.:Наука, — 2006. — 579 с.
11. *Мамиконов А.Г, Кульба В.В.* Синтез оптимальных модульных СОД. — М.: Наука, — 1989. — 276 с.
12. *Ашимов А.А., Мамиконов А.Г., Кульба В.В.* Оптимальные модульные системы обработки данных. — Алма-Ата: Наука, — 1981. — 186 с.