УДК 004.8
ББК 32.973.202:20.18

МАТЕМАТИЧЕСКИЕ АСПЕКТЫ ПРОГРАММИРОВАНИЯ И РЕАЛИЗАЦИЯ МАШТАБИРУЕМЫХ ИНФОРМАЦИОННЫХ КОМПОНЕНТОВ ГИС

Черный С. Г.[1]
(*Керченский государственный морской технологический университет, Керчь*)

Произведен анализ информационных продуктов, которые используют key-value data base технологию. Предложен механизм по улучшению масштабируемости и работоспособности крупномасштабных компонентов анализа для геоинформационных систем в аспектах глубоководной добычи.

Ключевые слова: база данных, геоинформационные системы, глубоководная добыча, оптимизация.

1. Введение

Эксплуатация океана и его ресурсов на протяжении существенного периода времени ограничивалась рыболовством и прибрежной деятельностью. Технологические ограничения не позволяли в более глубоких водах организовывать рентабельные предприятия. В первые десятилетия 21-го века существуют причины, которые препятствуют разработкам в данной области. Отметим, что геологоразведка и подводное строительство осуществляется с помощью флотов автоматизированных и дистанционно управляемых аппаратов. Быстрый рост спроса на редкоземельные элементы, используемые в широком спектре электроники и других высокотехнологичных приложений,

[1] *Сергей Григорьевич Черный*, *кандидат технических наук*, *доцент* (*sergiiblack@gmail.com*).

создавал все более острый дефицит сырья, что превратило их в ресурсы стратегического значения и подняло на один уровень с добычей нефти и природного газа в предыдущие десятилетия. Это стало особенно очевидным в Азии, такие страны, как Индия, Япония, Южная Корея и Индонезия наращивают усилия. Широкое применение, хотя и с большим риском опасности – добыча гидрата метана. Информатизация данного комплекса работ начинает активно развиваться и приобретать более ориентированных характер специализированных комплексов.

2. Постановка проблемы

Возникает существенная необходимость в информационной поддержки принятия решений процессов глубоководной добычи, которые комбинируют аппарат геоинформационных систем (ГИС) и компонентов надстроек анализа, и дополнительных специализированных программных комплексов.

Т.к. большинство информации хранится в виде цифровых карт с несущей информацией визуально и практически реализованных в таблицах данных (хранилищах). Как пример можно привести информацию получаемую отделом гидроакустических систем и наносимую на карту (работа с гидроакустическими данными, сонограммами, полученными от эхолотов, гидролокаторов бокового обзора, акустических профилографов (АП) донных осадков и многолучевых эхолотов); данные обрабатываются при помощи специализированного программного обеспечения. Далее строится мозаика, а данные эхолотов служат для построения батиметрических карт в различных ГИС. Затем мозаики отдаются на интерпретацию специалиста. В конечном итоге может быть сформирована 3D модель области. Количество таблиц, которые могут быть привязаны к слою и может быть достаточно существенным, в рамках ограничения системы, но также мы можем использовать и подключаемые модули серверных групп.

В рамках данной работы, будет освещена тема масштабируемости системы, идентификации параметров,

пропускной способности канала и извлечение необходимых данных с минимальными затратами временного барьера.

Одним из важнейших свойств программных систем на сегодняшний день является масштабируемость. Под термином «масштабируемость» чаще всего понимают способность системы справляться с увеличением нагрузки, сохраняя пропускную способность, при увеличении определенных ресурсов системы.

При разработке проектов в ГИС используются базы данных (БД), в частности такие как MySQL, но использование данной технологии является не совсем оправданно, так как анализируя архитектурный уровень приложения, в большинстве случаях базу используют просто как некое хранилище обычных данных. Ярким примером может служить система кэширования базы, которая используется во избежание дополнительных запросов (Memcached - распределенная система хранения данных на основе хеш-таблицы). В более общем случае можно сказать, что это хранилище пар «ключ-значение», над которыми можно производить только общие операции — запись, чтение, удаление и проверку на присутствие.

Доступ к i-тому элементу массива определяется, как смещение i от начала массива:

$$p[i] = p + i * sizeof(p[0]),$$

где p - адрес начала массива.

3. Результаты исследования

Проблемно-ориентированные информационные пространства ГИС разного уровня и назначения, являются одним из элементов создаваемой платформы развития информационного сообщества. Они обеспечивают новый уровень информационного обслуживания и широкое использование современных методов и средств аналитики, системного анализа, моделирования и оптимизации в решении глобальных, стратегических и текущих задач [8]. Современные информационные проекты (системы) хранящие не просто пары «ключ-значение», а и дополнительную мета-информацию об объекте, которые уже приближаются по возможностям к БД и

их иногда называют документ-ориентированными базами (хранилищами), так как единицей информации, над которой происходит работа, является документ и ассоциированные с ними данные.

Хранилища данных строятся на основе Distributed Hash Table (DHT) и изначально готовы к распределенной работе, обеспечивая масштабируемость и устойчивость к отказам отдельных узлов. В различных системах это решается за счет среды (например, если хранилище работает поверх Erlang VM), другие используют встроенные средства распределенной работы (JGroups для систем на Java) или собственные решения, как Memcached. Особенности технологии key-value имеют положительные и отрицательные аспекты одновременно.

Доступ к любому элементу массива осуществляется за постоянное время M(t). Если в проектируемой БД используются индексы от 0 до N, и предполагается, что большинство из них будет заполнено, тогда можем в качестве хранилища использовать только массивы, но следует учесть, что индексы бывают как цифровые так и символьные (цифровые индексы могут иметь большую размерность и реальное заполнение такого массива будет не более 10-15%, что будет совсем не эффективно с точки зрения использования свободной памяти).

Кеширование является одной из основополагающей составляющих хранилищ данных или СУБД. В каждой из реализаций конкретного продукта оно реализованы по своему. Фрагмент простой реализации приведем ниже.

```
void * getData(const char* key) {
    char* data = getDataFromcache(key);
    if (data) return data;
    data = getDataFromDb(key);
    if (data) storeDataToCache(key, data);
    return data;
}
```

На этом фоне актуализируется вопрос о адаптивности данных систем для работы в Cloud-среде. Благодаря простоте API решения отлично масштабируется (как на чтение, так и на запись), в том числе и динамически, без перерыва в работе, как элемент подключенный к ГИС.

С ростом количества ключей, возрастает и занимаемая оперативная память, что не является критическим замечанием, если количество используемых ключей от 10К до 100К. При увеличении количества ключей, разработчикам необходим механизм, который мог ограничивать количество ключей или занимаемую оперативную память. MapInfo и ArcGis позволяют так же работать с извлечением информации из БД, аутентифицировать таблицы, но важным условием являться подключаемые модули и кодирование хранилища. При загрузки информации (подключение сетевых таблиц хранилища) с международных серверов научно-исследовательских предприятий космического зондирования, структуризация потоковых данных по ключам может занимать терабайты информации, что замедляет работу с мобильностью карты, а отображение компонентных элементов анализа, требует наличие существенных ресурсов. Ускорение процессов адаптации карты при программно-логическом анализе, заключается в механизме кэширования.

Существуют следующие алгоритмы реализации кеширования:

MRU (Most Recently Used) – вытесняется последний использованный буфер. Реализуется простая очередь, в которую поступают ключи, по мере их запроса. Реализацию очереди может выполнять массив размера Hsize, с двумя указателями Pw – текущий указатель записи и Pr. При условии, когда один из указателей достигает границы массива Hsize, происходит сбрасывание на начало массива P0.

LRU (Least Recently Used) – вытесняется буфер, неиспользованный дольше всех. Процесс инвалидации ключей осуществляется отдельным потоком или по таймауту, между запросами. Libevent/libev – вполне позволяют реализовать обе схемы. Для хранения времени последнего обращения к элементу можно использовать упорядоченный массив-очередь или дерево BTree. При условии использования упорядоченного массива, разработчик имеет время доступа M(ln(t)), что приводит к проблеме вставки, когда необходимо добавить новый ключ в кеш, если до этого были уже внесены ключи со временем expiretime больше и меньше чем добавляемого - одновременно.

По этому, можно реализовать массив с агрегированными по времени списками ключей. В литературе указаны случаи, когда предложено использовать BTree дерево, где индексом служит время, а данными – значение ключа, тогда сложность доступа можно представить M(ln(t)), где существует также и проблема переодической балансировки дерева, что сравнительно накладная операция: $M(t^2)$. При постоянном добавлении упорядоченных ключей, каковыми является текущее время, получаем разбалансированнное дерево, время поиска в котором M(t).

LFU (Least Frequently Used) – подсчитывает, как часто используется закешированный элемент. Элементы, обращения к которым происходят реже всего, вытесняются в первую очередь. Реализуется, как массив счетчиков с указателем на минимальное значение счетчика. При появлении необходимости вытеснения, вытесняется элемент, на который указывает указатель. При добавлении нового элемента – указатель на минимальное значение счетчика автоматически становится индексом нового элемента. Проблемой является нахождение минимального счетчика, после вытеснения данных. При этом необходимо пройтись по всему массиву и найти элемент с наименьшим счетчиком. Сложность в этом случае – линейная M(t).

ARC (Adaptive Replacement Cache) – алгоритм вытеснения, комбинирующий LRU и LFU, запатентованный IBM.

При работе с морской геологической информацией, часто возникают проблемы, когда приложение увеличивает потребность в ресурсах настолько, что таблица данных не помещается на один сервер, тогда можно выделить отдельный более мощный сервер под рабочую (используемую) таблицу. Таблицу можно портицировать, а фрагментарные части разнести по серверам (тип такого масштабирования – вертикальное).

Проблема масштабируемости серверов БД достаточно актуальная тема, которая в последнее время все чаще оговаривается в мире информационных технологий при внедрении мощных программных продуктов, что влечет рост нагрузки на сервис (когда сервер БД не справляется с количеством запросов на чтение/запись данных).

Проектируя ГИС приложения или компоненты, разработчик учитывает комплексные структурные части компонент. Например, Hash таблица состоит из двух основных компонент: bucket и hash-функции, где Hash-функция на входе принимает значение ключа, а на выходе отдает число, соответствующее индексу массива. При условии, что ключ символьный, тогда вначале вычисляем crc значение, и далее находим остаток от деления. Данная функция для разных значений ключей может иметь одинаковые значения и такое свойство называется коллизией. Если, величина корзины не очень большая, то вероятность коллизии увеличивается. На практике, надо рассчитывать так, чтоб величина корзины соответствовала приблизительному объему хранения данных.

Теоретически, при малых коллизиях, доступ к элементу hash-таблицы считается как постоянная величина M(t), и не зависит от размера таблицы данных, но выбор из этой таблицы может быть только по ключу. На данных, организованных по принципу hash-таблиц возможны операции только сравнения: равно и не равно. Hash-таблицы не позволяют делать выбор по диапазонам ключей или поиск ключей по операциям больше или меньше и для этого, необходимо прибегнуть к другой технологии.

Осуществление процесса поиска некого элемента в дереве, использует временной ресурс системы, который пропорционален глубине дерева. При условии формирования дерева произвольным способом, его вложенность рассчитывается, как log2(n). Для поиска элемента в структуре такого дерева достаточно произвести log2(n) сравнений, где n - это предполагаемая размерность массива данных. Если, разность между уровнями, где листья не имеющие потомков, не превышает 1, то такое дерево называют сбалансированным.

Если разница между уровнями элементов, которые не имеют обоих родителей более чем 2, то такое дерево не является сбалансированным.

Над деревом возможны следующие операции:

– взятие данных родительской вершины;

– взятие данных одной из дочерних дочерних вершин (правой и левой);

– вставка новой вершины;

– удаление вершины.

Если, взятие одной из соседних вершин – операция сложности O(a), то вставка новой вершины, меняет упорядоченность. Это требует времени O(h), где h – это глубина дерева.

Более сложная операция удаления. Сложность удаления ветви (крайнего нижнего уровня) есть величина постоянная, но если удалить элемент, который является узлом, то после удаления необходимо трансформировать все дерево, что влечет затраты времени не менее, чем O(h). По этому, на практике, данные в индексе не удаляют, а помечают как недоступные.

При построении key-value хранилищ, данные не имеет смысла держать в самом дереве, поэтому в структуре бинарного дерева хранятся только ключи, и ссылка на корзину, в которой хранятся данные. Корзина представляет собой динамический массив данных, с элементами фиксированной длинны, а при условии, что наши данные не влезают в один элемент корзины, тогда мы запрашиваем следующий элемент, а в структуре помечаем, что наши данные должны иметь продолжение.

Примеров проектов, которые сталкиваются с данной проблемой, можно привести большое количество – начиная от сервисов новостей и заканчивая социальным сетями и такими сервисами, как Skype и ICQ.

Используя процедуру репликации, разработчик должен учитывать предельный максимум слейв-серверов (и обычно он связан с тем, что на каком-то этапе уже мастер становится слабым звеном и начинаются проблемы с записью, а не с чтением) или шардинг (суть шардинга заключается в логическом разделении данных по различным ресурсам исходя из требований к нагрузке. Различают вертикальный и горизонтальный шардинг. У каждого из этих методов есть свои характерные особенности, ни один нельзя выделить как идеальное решение проблемы.

Среди информационных продуктов на сегодняшний день известно не мало подобных систем: MemcacheDB, Hadoop HBase, Cassandra, Hypertable, Scalaris, Dynomite, Kai, Ringo, Project Voldemort, ThruDB.

Построение схемы обобщенной элементов ГИС с возможностью масштабирования, можно осуществить на основе анализа входных/исходных информационных потоков, которые функционируют в автоматизированной системе [8].

Построение схемы обобщенной ГИС можно осуществить на основе анализа входных/исходных информационных потоков, которые функционируют в автоматизированной системе [8]. В соответствии с общей теорией систем (ОТС) [5] функциональная система S определяется как отображение входного множества X (множества первичных элементов) на выходное множество Y. В формальном представлении ОТС это будет соответствовать записи

$$S : X \to Y .$$

Любая сложная система, в общем случае, считается гетерогенной, поэтому целесообразно разбить ее на однородные компоненты путем построения стратифицированной (многоуровневой) системы [5, 10]. Математическую модель процесса проектирования ГИС морского глубоководного комплекса можно представить в виде итерационного процесса, состоящего из отдельных взаимодействующих проектных процедур трех множеств [7]:

– множества моделей $M = \{M_1, M_2, ..., M_n\}$;

– множества операций над моделями $O = \{O_1, O_2, ..., O_n\}$;

– множества критериев проектирования $K = \{K_1, K_2, ..., K_n\}$,

где n – количество этапов проектирования.

Весь процесс проектирования представляется последовательностью проектных процедур $\rho = \{\rho_1, \rho_2, ..., \rho_n\}$, где каждая проектная процедура $\rho_j = \langle O_j, M_j, K_j \rangle$ переводит модель проектируемого объекта в следующее состояние [7]:

$\rho_1 : M_1 \to M_2, \rho_2 : M_2 \to M_3, ..., \rho_{n-1} : M_{n-1} \to M_n$.

При этом необходимо соблюдение условия:

$\forall \mathrm{M_j} \in \mathrm{M} \quad \exists \rho_j : \mathrm{M_j} \to \mathrm{M_{j+1}}$,

где $M_j \Rightarrow M_{j+1}, K_j \subseteq K, O_j \subseteq O, j = \overline{1, N}$, где $\Rightarrow$ – символ предшествования.

Выполнение определенного числа проектных процедур ρ_{ij} на *i*-ом витке спиральной модели процесса проектирования позволяет перейти на следующий, более высокий уровень данного проектного решения.

Отметим, что при построении информационных систем при большом количестве критериев проектирования возникает задача обработки обобщенного критерия (критериев) проектирования, как свертки множества (или подмножества) критериев, возникает необходимость идентификации [4]:

$$x^O = \underset{x \in X}{opt}\, G[F(x), \Lambda],$$

где $\Lambda = \{\lambda_1, \lambda_2\}$ – относительная важность локальных критериев; X – количество альтернатив (решений) вида x; $F(x) = \{f_1(x), f_2(x), ..., f_n(x)\}$ – количество критериев.

В случае, когда определен вид оператора $opt\, G$ и задан обобщенный критерий или указано правило, позволяющее упорядочивать возможные решения, получение идентификатора x^O, не вызывает принципиальных трудностей при разработки и вычислении.

В задаче принятия решений в условиях неопределенности для каждого из наборов исходных данных In_i требуется определить наиболее эффективные варианты построения элементов модели ГИС морского предприятия s_i, $i = \overline{1, n}$. В упрощенной форме, при идентификации выбора в аспекте принятия решений, выбор может осуществляться по критериям [2]:

– Лапласа: $s = \arg\max_i \left\{ \frac{1}{n} * \sum_{j=1}^{n} U(s_i / p_j) \right\}$;

– максимакса: $s = \arg\max_i \max_j \{U(s_i/p_j)\}$;

– Вальда $s = \arg\max_i \min_j \{U(s_i/p_j)\}$;

– Гурвица

$$s = \arg\max_i \left[\gamma * \max_i U(s_i/p_j) + (1-\gamma) * \min_i U(s_i/p_j)\right],$$

где γ – адаптационный коэффициент «риска - осторожности», $0 \le \gamma \le 1$;

– Сэвиджа $s = \arg\min_i \max_j \{\overline{U}(s_i/p_j)\}$, где $\overline{U}(s_i/p_j) = 1 - U(s_i/p_j)$, $\overline{U}(s_i/p_j) = \max_i U(s_i/p_j) - U(s_i/p_j)$ – функции потери полезности;

– Ходжеса-Лемана

$$s = \arg\max_i \left[\nu * \sum_{j=1}^{n} p_j * U(s_i/p_j) + (1-\nu) * \min_i U(s_i/p_j)\right],$$

где ν – адаптационный параметр, $0 \le \nu \le 1$. При $\nu = 1$ критерий переходит в критерий Байеса-Лапласа, при $\nu = 0$ – в минимаксный.

Выбор единственного решения должен производиться исходя из максимального соответствия требованиям задания на проектирование ГИС для поддержки глубоководной добычи; следует стремиться к выбору "устойчивого" решения, являющегося лучшим по нескольким критериям, с учетом фактора безопасности и экономичности.

В общем случае исследуемая система представляется данным математическим аппаратом следующим образом [1]:

$$S = <\{Res\} L \{Rel\} L \{B\}>, \{Res\} \subseteq U, \{Cond\} \subseteq U,$$

где S – теоретико-множественная модель объекта-системы, представленная в виде композиции; {Res} – множество «первичных элементов» (элементов, которые должны быть определены перед определением объекта-системы S); {Rel} – множество отношений, определенных на элементах множества {Res}; {B} – множество ограничений, ограничивающих отношения {Rel} на множестве {Res}; U –

универсум, из которого выделяются «первичные элементы» res множества $\{\text{Res}\}$; $\{\text{Cond}\}$ – множество условий, определяющих правила выделения «первичных элементов» множества $\{\text{Res}\}$; L – совокупность законов композиции, определяющих правила синтеза из рассмотренных выше множеств $\{\text{Res}\}$, $\{\text{Rel}\}$ и $\{\text{B}\}$ объекта-системы S.

Обоснование предпочтения данного математического аппарата:

– данный аппарат рассматривает исследуемую систему как результат выбора «первичных элементов» по некоторым условиям из универсума всех возможных элементов, причем тип условий (временные, технические и т.д.) не ограничен;

– данный аппарат одинаково хорошо может описывать статические и динамические состояния моделируемой системы за счет изменений детализированных описаний элементов множества $\{\text{Res}\}$, отношений множества $\{\text{Rel}\}$, ограничений множества $\{B\}$ и условий множества $\{\text{Cond}\}$;

– нежестко определенная совокупность законов композиции компонентов метамодели позволяет сравнительно легко адаптировать данную метамодель для описания практически любого варианта реализации ГИС.

Таким образом, применительно к ГИС глубоководной добычи, множество «первичных элементов» {Res} является конечным и счетным, и идентифицируется из подмножеств [1]:

$$\{\text{Res}\} = \{\text{Res}_{\text{ч}}\} \cup \{\text{Res}_{\text{И}}\},$$

где $\{\text{Res}_{\text{ч}}\}$ – конечное и счетное подмножество «первичных элементов», описывающее человеческие ресурсы проектирования ГИС глубоководной добычи, которое в общем случае состоит из следующих наборов элементов

$$\{\text{Res}_{\text{ч}}\} = \{\text{res}_{\text{З}}\} \cup \{res_{\text{Р}}\} \cup \{\text{r}es_{\text{П}}\},$$

$\{\text{res}_{\text{З}}\}$ – набор элементов, описывающих заказчиков;

$\{res_{\text{Р}}\}$ – набор элементов, описывающих разработчиков системы;

$\{\text{r}es_{\text{П}}\}$ – набор элементов, описывающих пользователей

системы;

$\{\text{Res}_{\text{И}}\}$ – конечное и счетное подмножество «первичных элементов», описывающее инструментальные ресурсы проектирования ГИС глубоководной добычи, которое в общем случае состоит из следующих наборов элементов

$$\{\text{R}es_{\text{И}}\} = \{\text{r}es_{\text{ПР}}\} \cup \{\text{r}es_{\text{АП}}\} \cup \{\text{r}es_{\text{ДОК}}\} \cup \{\text{r}es_{\text{ТЕСТ}}\} \cup \{\text{r}es_{\text{ФИН}}\}$$

$\{\text{r}es_{\text{ПР}}\}$ – набор элементов, описывающих программные ресурсы для разработки ГИС глубоководной добычи;

$\{\text{r}es_{\text{АП}}\}$ – набор элементов, описывающих аппаратные ресурсы для разработки ГИС глубоководной добычи;

$\{\text{r}es_{\text{ДОК}}\}$ – набор элементов, описывающих документальные ресурсы для разработки ГИС глубоководной добычи (среды и средства документирования);

$\{\text{r}es_{\text{ТЕСТ}}\}$ – набор элементов, описывающих среды и средства тестирования ГИС глубоководной добычи;

$\{\text{r}es_{\text{ФИН}}\}$ – набор элементов, описывающих финансовые ресурсы для разработки ГИС глубоководной добычи.

По аналогии с элементами метамодели ГИС глубоководной добычи, множество типов связей ГИС является конечным и счетным и, в общем случае, состоит из следующих подмножеств [1]:

$$\{\text{Rel}\} = \{\text{Rel}_{\text{ч}}\} \cup \{\text{Rel}_{\text{И}}{}^{\text{ч}}\} \cup \{\text{Rel}_{\text{И}}\} \cup \{\text{Rel}_{\text{ч}}{}^{\text{и}}\},$$

$\{\text{Rel}_{\text{ч}}\}$ – счетное подмножество, описывающее типы связей между элементами подмножества $\{\text{Res}_{\text{ч}}\}$;

$\{\text{Rel}_{\text{И}}{}^{\text{ч}}\}$ – счетное подмножество, описывающее типы связей, направленных от элементов подмножества $\{\text{Res}_{\text{ч}}\}$ к элементам подмножества $\{\text{Res}_{\text{И}}\}$;

$\{\text{Rel}_{\text{И}}\}$ – счетное подмножество, описывающее типы связей между элементами подмножества $\{\text{Res}_{\text{И}}\}$;

$\{\text{Rel}_{\text{ч}}{}^{\text{и}}\}$ – счетное подмножество, описывающее типы связей, направленных от элементов подмножества $\{\text{Res}_{\text{И}}\}$ к элементам подмножества $\{\text{Res}_{\text{ч}}\}$.

Особенностью ГИС глубоководной добычи, является выделенные «первичные элементы», которые могут и должны рассматриваться как в статическом представлении, так и в динамическом представлении. Иными словами, каждый из элементов выделенных подмножеств $\{\text{Res}_\text{Ч}\}$ и $\{\text{Res}_\text{И}\}$ может быть описан как набором статических описаний, так и совокупностью правил (законов, зависимостей или функций), описывающих динамику поведения данных элементов. Поэтому множество условий выбора элементов {Cond} можно рассматривать как совокупность следующих подмножеств [1]:

$$\{\text{Cond}\} = \{\text{Cond}_\text{Ч}\} \cup \{\text{Cond}_\text{И}\},$$

где $\{\text{C}ond_\text{Ч}\}$ – счетное подмножество статических (на определенном промежутке времени выполнения проектных работ) условий отбора «первичных элементов», описывающих человеческие ресурсы проектирования ГИС, которое в общем случае состоит из следующих наборов элементов:

$$\{\text{Cond}_\text{Ч}\} = \{\text{cond}_\text{З}\} \cup \{\text{cond}_\text{Р}\} \cup \{\text{cond}_\text{П}\},$$

где $\{cond_\text{З}\}$ – набор условий (статических характеристик), выдвигаемых к заказчикам ГИС глубоководной добычи;

$\{cond_\text{Р}\}$ – набор условий (статических характеристик), выдвигаемых к разработчикам (администраторам) ГИС глубоководной добычи;

$\{cond_\text{П}\}$ – набор условий (статических характеристик), выдвигаемых к пользователям ГИС глубоководной добычи.

$\{\text{C}ond_\text{И}\}$ – счетное подмножество статических (на определенном промежутке времени выполнения проектных работ) условий отбора «первичных элементов», описывающих инструментальные ресурсы проектирования ГИС, которое в общем случае состоит из следующих наборов элементов:

$$\{\text{Cond}_\text{И}\} = \{\text{cond}_\text{ПР}\} \cup \{\text{cond}_\text{АП}\} \cup \{\text{cond}_\text{ДОК}\} \cup \{\text{cond}_\text{ТЕСТ}\} \cup \{\text{cond}_\text{ФИН}\},$$

где $\{cond_\text{ПР}\}$ – набор условий (статических характеристик), выдвигаемых к программным ресурсам ГИС;

$\{cond_\text{АП}\}$ – набор условий (статических характеристик), выдвигаемых к аппаратным ресурсам ГИС;

$\{cond_{ДОК}\}$ – набор условий (статических характеристик), выдвигаемых к документальным ресурсам (средам и средствам документирования) ГИС;

$\{cond_{ТЕСТ}\}$ – набор условий (статических характеристик), выдвигаемых к средам и средствам тестирования ГИС;

$\{cond_{экон}\}$ – набор условий (статических характеристик), выдвигаемых к экономическим ресурсам проектирования ГИС.

В свою очередь множество ограничений на связи между элементами {B} можно рассматривать как совокупность следующих подмножеств [1]:

$$\{B\} = \{B_ч\} \cup \{B_И{}^{ч}\} \cup \{B_И\} \cup \{B_ч{}^{И}\},$$

где $\{B_ч\}$ – счетное подмножество, описывающее статические и динамические ограничения связей подмножества $\{Rel_ч\}$;

$\{B_И{}^{ч}\}$ – счетное подмножество, описывающее статические и динамические ограничения связей подмножества $\{Rel_И{}^{ч}\}$;

$\{B_И\}$ – счетное подмножество, описывающее статические и динамические ограничения связей подмножества $\{Rel_И\}$;

$\{B_ч{}^{И}\}$ – счетное подмножество, описывающее статические и динамические ограничения связей подмножества $\{Rel_ч{}^{И}\}$.

Что касается универсума U, из которого выбираются «первичные элементы», то процессы его формирования и модификации в общем случае определяются композицией характеристик наборов подмножеств «первичных элементов» и их изменений под влиянием внешней среды.

Таким образом, метамодель ГИС глубоководной добычи в общем случае может быть описана следующим образом [1]:

$$S = \left\langle \begin{matrix} \left[\{res_З\} \cup \{res_Р\} \cup \{res_П\} \cup \{res_{ПР}\} \cup \{res_{АП}\} \cup \{res_{ДОК}\} \cup \{res_{ТЕСТ}\} \cup \{res_{экон}\}\right] L \\ \left[\{Rel_ч\} \cup \{Rel_И{}^{ч}\} \cup \{Rel_И\} \cup \{Rel_ч{}^{И}\}\right] L \left[\{B_ч\} \cup \{B_И{}^{ч}\} \cup \{B_И\} \cup \{B_ч{}^{И}\}\right] \end{matrix} \right\rangle,$$

то есть

$$F_U^{[\{\text{Res}_\text{ч}\}\cup\{\text{Res}_\text{И}\}]}: U \xrightarrow{[\{Cond_\text{ч}\}\cup\{Cond_\text{И}\}]} [\{\text{Res}_\text{ч}\}\cup\{\text{Res}_\text{И}\}].$$

Существует мнение, что не стоит решать проблему масштабируемости раньше, чем она возникнет. Данное положение с одной стороны достаточно оправданно, ведь эту проблему обеспечивает популярность сервиса у пользователей (при появлении большого количества пользователей возрастает большая нагрузка, что влечет возникновение проблем). Масштабируемость для традиционных систем управления БД – прежде всего финансовая проблема, и инвестор (или рекламодатель) не захочет вкладывать средства в непопулярные ресурсы, ведь существует вероятность того, что сервис не наберет критического количества аудитории. Под DHT понимается класс децентрализованных распределённых систем, которые обеспечивают поисковый сервис, похожий по принципу работы на таблицу хешей (рис. 1), и имеют структуру: (имя, значение), хранящиеся в DHT, а каждый участвующий узел может рационально искать значение, ассоциированное с данным именем (рис. 2). Ответственность за поддержку связи между именем и значением распределяется между узлами, таким образом изменение набора участников является причиной минимального количества разрывов. Это позволяет DHT изменять масштаб до значительно большого количества узлов и постоянно отслеживать добавление/удаление узлов и ошибки в их работе [5].

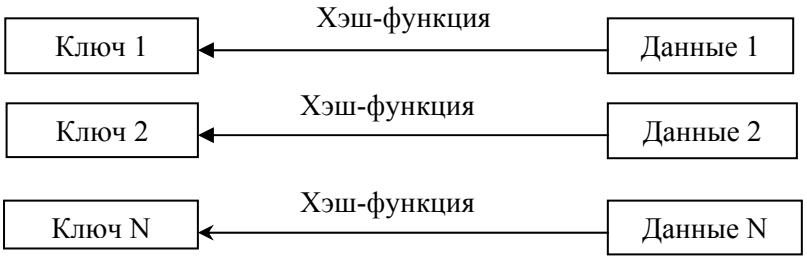

Рис. 1. Структура хэш-таблицы

Необходимо выделить отличительные черты key-value БД и таких привычных реляционных БД. Отвечая на вопрос, что такое БД мы понимаем прежде всего специальное хранилище структурированных данных, обеспечивающее эффективный поиск и обработку данных. Хранятся данные в файлах, имеющих собственную структуру и кешем в памяти.

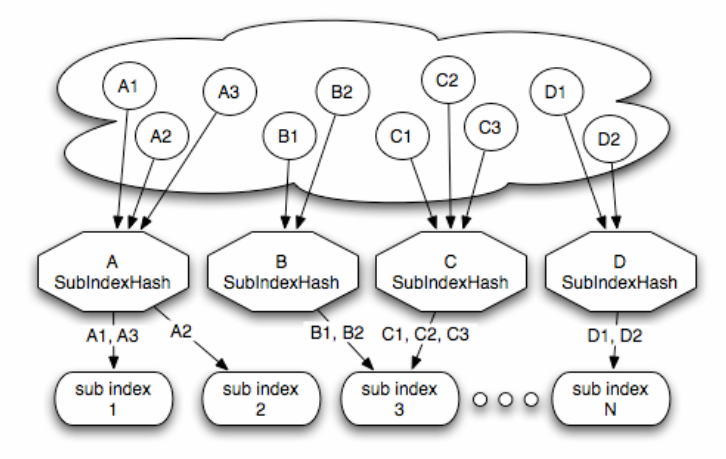

Рис. 2. Распределенная хэш-таблица

Важное свойство хеш-таблиц состоит в том, что при некоторых разумных допущениях, все три операции (поиск, вставка, удаление элементов) в среднем выполняются за время O(1). Поэтому БД, построенные на хэш-таблицах, отличаются высокой производительностью.

Кроме того, масштабируемость при таком решении является неограниченной благодаря свойствам DHT. При этом в программном коде взаимодействия сервиса и БД отсутствует необходимость в редактировании/корректировке, как пришлось бы при использовании любого из вариантов масштабирования обычной БД.

При условии идентификации и масштабируемости проектируемой ГИС глубоководной добычи, важным остается вопрос и извлечения данных из структуры, который был описан выше. Например, ситуация $s \in S$ может задаваться с помощью

формул специализированного языка *L*. Каждой поставленной ситуации s может соответствовать несколько решений. Таким образом, можно допустить, что существуют прецеденты вида $\langle s, r \rangle$ и $\langle s, r' \rangle$, которые отличаются при условии что $r \neq r'$. Данные по анализу формирования выбора области, поступающие в модуль информационно-аналитической надстройки модели, представим множеством прецедентов *M*:

$$M = \{\langle s_1, r_1, h_1, z_1 \rangle, \langle s_2, r_2, h_2, z_1 \rangle, \ldots, \langle s_i, r_i, h_i, z_i \rangle, \ldots, \langle s_n, r_n, h_n, z_n \rangle\}.$$

Каждый прецедент $e_i \in \{e\}$ может рассматриваться как условная импликация

(1) $s_i \Rightarrow r_i,\ i = \overline{1,n}$

Таким образом, если задана некоторая ситуация

$s_i \approx s_j,\ j = \overline{1,n},\ i \neq j$,

и существует прецедент $e_i = \langle s_i, r_i, h_i, z_i \rangle$, то можно утверждать, что r_j является приближенным решением для ситуации s_i. Чем ближе ситуация s_i к ситуации s_j, тем правдоподобнее является решение r_j, которое так же является и решением для s_i [6]. Для нахождения степени близости ситуации s_i к ситуации s_j и, соответственно, оценки близости решения r_j к искомому используется функция подобия ζ, на основе которой строится отношение подобия между прецедентами и выводится мера подобия *SM*.

Множество формул V составляет некоторую базу знаний о предметной области, полученную экспертным путем.

Для каждого прецедента e_i можно с помощью оценки подобия вычислить степень уместности решения r_i в ситуации, близкой к s_i. В случае, если для этого можно также использовать имеющиеся знания о предметной области, можно утверждать, что формула $V \to \left(s \to \Diamond_{SM_i} s_i\right)$ выполнима для класса ситуаций s_{e_i}. ХП *M* задает экстраполяцию отношения импликации для (1), и фактически является базой знаний,

содержащей приближенные импликации: $M^* = \left\{ s_i \Rightarrow_{\zeta_i} r_i \middle| (s_i, r_i) \in M \right\}$. Соответственно формула

$$M^* \to \left(s_i \Rightarrow_{\zeta_i} r_i \right),$$

также выполнима для класса ситуации s_{e_i}. В данной работе принято, что система прецедентов представляет собой структуру:

$\left\langle DM, SM_{G(L)}, U_L \right\rangle$,

где DM – БД прецедентов; $SM_{G(L)}$ – мера подобия, заданная на множестве интерпретаций G языка L, описывающего входные ситуации; U_L – множество формул языка L.

Список условий P, запишем в виде:

$P = \{(p_1, w_1), (p_2, w_2), \ldots, (p_k, w_k)\}$,

где $w_1, \ldots, w_k$ – весовые факторы условий; $p_1, \ldots, p_k$ – предикаты, характеризующие условия.

Идентификация интервальных групповых предпочтений на основе множества индивидуальных точечных оценок. Рассмотрим случай, когда исходная информация задана в виде точечных оценок ЛПР: по каждому весовому коэффициенту a_i задан набор точечных значений a_{ij}, $i = \overline{1,n}$, $j = \overline{1,m}$.

Для определения интервала возможных значений весовых коэффициентов, необходимо из множества a_{ij} выбрать минимальное и максимальное значение

$a_i^{\min} = \min_j a_{ij}$; $a_i^{\max} = \max_j a_{ij}$.

В результате по каждому весовому коэффициенту получим интервал возможных значений:

$a_i^{\min} \le a_i \le a_i^{\max}$.

Выделенные интервалы корректны только в том случае, если выполняются условия

$\sum_{i=1}^{n} a_i^{\min} \le 1$; $\sum_{i=1}^{n} a_i^{\max} \ge 1$,

так как в противном случае гиперпараллелепипед G возможных значений весовых коэффициентов может не содержать ни одного кортежа, который бы удовлетворял условиям нормирования

$$(2) \quad \sum_{i=1}^{n} a_i = 1, \quad a_i > 0, \quad i = \overline{1,n},$$

$$(3) \quad G = \underset{i=1}{\overset{n}{G}} \left[a_i^{\min}; a_i^{\max} \right], \quad 0 \le a_i^{\min} < a_i^{\max} \le 1.$$

При задании гиперпараллелепипеда (3) в пространстве весовых коэффициентов (ГВК) возможны следующие варианты:

1) заданный ГВК не содержит избыточных значений коэффициентов;

2) ГВК не пересекается с гиперплоскостью (2), т.е. в нем не существует нормированных значений весовых коэффициентов, которые бы удовлетворяли условию (2);

3) в заданном ГВК есть значения коэффициентов, которые удовлетворяют условию (2), но еще содержатся избыточные значения.

В случае 2 и случае 3 возникает необходимость коррекции оценок. Это предлагается сделать непосредственно экспертам или с помощью формальных процедур, которые описаны ниже [9, 10].

Для указанных выше случаев рассмотрим процедуру коррекции весовых коэффициентов. Осуществляется преобразование вида

$$a_i^{\min} = 2 * a_i^{\min} \Big/ \sum_{i=1}^{n} \left(a_i^{\min} + a_i^{\max} \right); \; a_i^{\max} = 2 * a_i^{\max} \Big/ \sum_{i=1}^{n} \left(a_i^{\min} + a_i^{\max} \right),$$

которое перемещает центр параллелепипеда (3) на гиперплоскость (2) и максимизирует «площадь» пересечения гиперпараллелепипеда с этой гиперплоскостью.

Дальше реализуется процедура отсева избыточных значений коэффициентов ГВК, которая является модификацией метода последовательного анализа для задач линейного программирования большой размерности [3]. Суть этого метода

состоит в сужении интервалов изменения коэффициентов $a_i^{\min}$, $a_i^{\max}$ на основе анализа вырожденного многогранника вида (3).

Для определения избыточных значений в области нижней границы по i-му ограничению необходимо требовать выполнение соотношения

(4) $a_i^{\min} + \sum_{j \neq i} a_j^{\max} \geq 1$.

Если это условие не выполняется, т.е. если $a_i^{\min} + \sum_{j \neq i} a_j^{\max} < 1$, необходимо нижнюю границу увеличить на некоторою величину $a_i^{\min} = a_i^{\min} + \delta_i$, для $\delta_i > 0$, чтобы в соотношении (4) выполнялось хотя бы равенство $a_i^{\min} + \sum_{j \neq i} a_j^{\max} = 1$, тогда $\delta_i^i = 1 - a_i^{\min} - \sum_{j \neq i} a_i^{\max}$.

При условии не выполнения неравенство $a_i^{\max} + \sum_{j \neq i} a_j^{\min} \leq 1$, аналогичными рассуждениями приходим к необходимости поправок на верхних границах интервалов $a_i^{\max} = a_i^{\max} + \Delta_i$, для $\Delta_i > 0$, где $\Delta_i = a_i^{\max} + \sum_{j \neq i} a_j^{\min} - 1$. Полученные в результате использования описанной выше процедуры новые границы параллелепипеда $a_i^{\min}$ и $a_i^{\max}$ определяют результирующий основной весовой коэфициент, который не содержит избыточных значений.

Ярким примером key-value БД является Berkeley DB (BDB). BDB является нереляционной БД – хранит пары «ключ-значение» как массивы байтов и поддерживает множество значений для одного ключа. BDB может обслуживать тысячи процессов или потоков, одновременно манипулирующих базами данных размером в 256 терабайт, на разнообразном оборудовании под различными операционными системами, включая большинство UNIX-подобных систем и Windows, а также на операционных системах реального времени [7]. Berkeley DB входит в состав многих программ с открытым кодом, в частности, система встроена в Web-сервер Apache и в

графическую оболочку Gnome. Все коммерческие дистрибутивы Linux, а также многие разновидности BSD содержат Berkeley DB.

Однако структура данных, дающая огромную производительность и масштабируемость, является плохо приспособленной для привычного SQL-поиска, ведь поиск в key-value БД производится только по ключу, а данные, ассоциированные с ним, могут быть чем угодно. Таким образом, задача обработки данных ложится полностью на приложение.

4. Выводы

Обобщая изложенный материал необходимо отметить ключевые аспекты технологии key-value, масштабируемости и информационного поиска для компонентов ГИС глубоководной добычи.

Большинство подобных хранилищ способны обработать десятки тысяч запросов в секунду на средней серверной конфигурации, данная функция однозначно задает направление использования данных инструментов - в высоконагруженных системах и для приложения ГИС.

Данные хранилища легко могут быть распределены на несколько серверов и тем самым делая процесс обращения к ресурсам более функциональным и эффективным, что очень важно при работе с анализом области по глубоководной добыче и эксплуатации. Распределение информационных ресурсов и компонентов ГИС, как правило, не требует остановки уже существующих серверов, но возможна функция переноса части данных с других серверов на новый на основании применения консистентного метода распределения ключей.

Данные хранилища очень просто обслуживать, процедура резервного копирования не требует остановки серверов и может выполняться в автоматическом режиме, хотя технология анализа для прецедентных структур занимает более значительный промежуток времени и ресурса.

На основании надстроек для ГИС и key-value возможен механизм сложных запросов, элементы математической структуры описаны в работе.

Литература

1. АХМЕД И. АЛЬБАХЛУЛ. *Модели структурно-объектной технологии разработки интерфейсного комплекса корпоративной информационной системы:* Дис. … канд. техн. наук: 05.13.06; – Х., 2005. – 148 с.: ил.
2. БЕСКОРОВАЙНЫЙ В.В., КУЗЬМЕНКО А.А. *Оценка рисков и выбор решений при управлении проектами создания распределенных информационных систем* // Глобальні інформаційні системи. Проблеми й тенденції розвитку: І Міжнародна конференція. – Харків-Туапсе, 2006. – С.113-114.
3. ЗАЙЧЕНКО Ю.П. *Исследование операций: нечеткая оптимизация* / Ю.П.Зайченко: Учеб. пособие.– Киев: Вища школа, 1991. – 191с.
4. КУЗЬМЕНКО С.В. *Информационная технология контроля в дискретных системах обслуживания* // Глобальні інформаційні системи. Проблеми й тенденції розвитку: І Міжнародна конференція. – Харків-Туапсе, 2006. – С.155-156.
5. МЕСАРОВИЧ М., МАКО Д., ТАКАХАРА И. Теория иерархических многоуровневых систем. – М.: МИР, 1973. – 344 с.
6. НЕКРАСОВ О.Б. *Моделі, методи і технологія інформаційної підтримки процесів проектування й модіфікації технічної продукції*: Автореф. дис. канд. техн. наук. – Херсон, 2005. – 20 с.
7. ФИЛАТОВ В.А., СЕМЕНЕЦ Р.В. *Методы и средства проектирования информационных систем и распределенных баз данных* // Вестник Херсонского национального технического университета. – 2007. – №4 (27). – С.203-207.
8. ЦВЕТКОВ В.Я. *Геоинформационные системы и технологии.* – М.: Финансы и статистика, 1998. – 288 с.
9. ЧЕРНЫЙ С.Г. *Элементы технологии групповой классификации многопризнаковых объектов для поддержки принятия решений* // Вестник Херсонского национального технического университета. –2013. – №1 (46). – С. 155-160
10. ШОПИН, А.Г. *Построение функции принадлежности нечеткого множества и оценка его вероятностных*

характеристик // Электронный журнал «ИССЛЕДОВАНО в РОССИИ». – 2003. – с. 453-467.

ARTICLE TITLE, ПЕРЕВОД НАЗВАНИЯ НА АНГЛИЙСКИЙ

Sergey Cherny, Kerch State Marine Technological Univercity, Kerch, Cand.Sc., assistant professor (sergiiblack@gmail.com).

Abstract: The analysis was conducted for the information products that use key-value database technology. A mechanism is proposed to improve the scalability and performance analysis for large-scale components of geographic information systems in aspects of deep-sea mining.

Keywords: database, geographic information systems, deep sea mining, optimization.

*Статья представлена к публикации члено*м *редакционной коллегии* ...заполняется редактором…

Поступила в редакцию ...заполняется редактором…
Опубликована ...заполняется редактором…