5. *Зуенко А. А.*, *Фридман А. Я.* Развитие алгебры кортежей для логического анализа баз данных с использованием двухместных предикатов // Известия РАН. Теория и системы управления. 2009. № 2. (В печати).
6. *Зуенко А. А.*, *Фридман А. Я.* Метод семантического анализа нерегламентированных запросов в реляционной базе данных с иерархической структурой (настоящий сборник).
7. *Страуструп Б.* Язык программирования C++. 3-е изд. СПб.: Невский Диалект — БИНОМ, 1999.
8. *Виноградов А. Н.*, *Жилякова Л. Ю.*, *Осипов Г. С.* Динамические интеллектуальные системы. I. Представление знаний и основные алгоритмы // Известия РАН. Теория и системы управления. № 6. М.: Наука, 2002. С. 119–127.
9. *Фридман А. Я.*, *Фридман О. В.* Контроль корректности вычислений и управление выводом в системах продукций // Имитационное моделирование в исследованиях проблем регионального развития. Апатиты: КНЦ РАН, 1999. С. 93–100.

Метод семантического анализа нерегламентированных запросов в реляционной базе данных с иерархической структурой

А. А. Зуенко, А. Я. Фридман

ИИММ ТП КолНЦ РАН, Апатиты

Статья посвящена разработке метода семантического анализа и оптимизации незапланированных (нерегламентированных) адресных (путевых) запросов к базам данных с фиксированной иерархической структурой. Метод основан на учете семантических ограничений в виде специализированной онтологии и последующем использовании этой информации при анализе запросов с целью упрощения их структуры. Применимость метода ограничена классом онтологий, поддерживающих только иерархические связи «часть-целое» и ассоциативные (предметные) связи. Описана реализация метода для базы данных системы моделирования на основе открытой иерархической концептуальной модели предметной области.

1. Введение

В целях обеспечения полноценной интеграции с объектно-ориентированными приложениями разработчики современных СУБД все более стремятся перенести парадигмы объектно-ориентированного программирования на технологию БД. Современные СУБД используют концепцию объекта (экземпляр сложного типа, определяемого пользователем) для представления знаний о предметной области, снижения сложности запросов к данным, хранения объектов программ и так далее [1, 2].

В СУБД объектный подход широко используется при обработке данных, имеющих иерархическую структуру, в частности, при работе с форматом XML, который фактически представляет собой объектно-ориенти-

рованную модель документа. Все данные XML-документов организованы строго иерархически. Древовидная структура документа отображается и в структуре запросов к XML-документам. В связи с этим запросы, как правило, опираются на понятие пути к единице хранимой информации, как элементу иерархии, и, поэтому носят название «путевые запросы». Соответствие модели запросов, используемой XML-документами, объектной модели данных позволяет сильно упростить анализ запросов, так как он производится на основе схемы XML-документа, которая описывает классы элементов документа и отношения между ними в объектном стиле. Однако, набор средств, позволяющих задавать ограничения на отношения между классами элементов XML-документов, достаточно беден и предназначен только для проверки типов этих элементов и всего документа в целом [3]. Семантические же ограничения предметной области удобнее всего описывать средствами WEB-онтологий [2, 4]. Эти онтологии предназначены в основном для эффективного исполнения пользовательских запросов, сформулированных на естественном языке, посредством их расширения и уточнения в ходе процедур логического вывода. Однако современные онтологические системы не позволяют задавать ограничения между классами в виде логических формул над предикатами первого и второго порядка.

Заметим также, что СУБД, предназначенные исключительно для работы с XML-документами, не отвечают в полной мере требованиям безопасности и ссылочной целостности данных [5]. Следовательно, они не могут считаться удачным решением для хранения информации с иерархической структурой. Поэтому часто применяют комбинированный подход к хранению XML-документов: данные хранятся в реляционных таблицах, а доступ осуществляется посредством путевых запросов, причем для согласования модели данных и модели запроса используются специализированные XML-схемы [6].

Цель данной работы — описать метод учета семантических ограничений предметной области уже на этапе анализа и оптимизации путевых запросов.

Предлагаемый метод логического анализа запросов был применен в системе моделирования на основе концептуального ситуационного подхода [7].

2. Онтология структурных ограничений

Для того, чтобы кратко пояснить роль онтологии в рамках предлагаемого подхода, смысл и форму представленной в ней информации, опишем онтологию структурных ограничений, созданную для системы ситуационного концептуального моделирования (ССКМ) [7].

ССКМ предназначена для моделирования и прогнозирования поведения сложных промышленно-природных комплексов (ППК). Концептуальная модель (КМ) исследуемого объекта в рамках ССКМ используется не только для декларативного описания сущностей предметной области и связей между ними, но и является основным средством организации и управления всем процессом моделирования. Такой подход позволяет автоматизировать стадии построения модели предметной области, проверки корректности модели с точки зрения структурных ограничений, проведения вычислительного эксперимента. Множество моделей, которые допустимы в конкретной системе моделирования на основе концептуального подхода, задается схемой КМ, где описываются классы элементов КМ и бинарные отношения между ними, а также процедурами проверки корректности структуры модели. Процедуры проверки корректности опираются на ограничения, которым должны удовлетворять отдельные отношения схемы и группы отношений. Знания об ограничениях на отношения схемы КМ в основном заключены в некоторых процедурах (например, в процедурах проверки логической непротиворечивости модели), что затрудняет повторное использование этой информации в других целях, в частности, для логического анализа запросов. Ниже для систематизации таких знаний предлагается применять специализированную онтологию структурных ограничений.

Итак, потребность в разработке онтологии структурных ограничений обусловлена следующими причинами:

- необходимость выработать унифицированное понимание структуры информации с целью последующего совместного использования людьми и/или программными модулями;
- целесообразность повторного использования знаний предметной области;
- необходимость сделать допущения в предметной области явными;
- необходимость отделить знания предметной области от управляющих знаний, используемых для их обработки.

В качестве примера приведем упрощенный фрагмент онтологии структурных ограничений ситуационной концептуальной модели [8] (рис. 1):

Рисунок 1 отражает следующие факты о классификации и отношениях элементов КМ. Концептуальная модель (Model) состоит из множеств объектов (Object), процессов (Process) и ресурсов (Resource). Объекты формализуют организационную структуру исследуемого ППК. Каждый объект является оболочкой набора процессов — преобразователей данных (ресурсов). Как объекты, так и процессы связаны между собой потоками

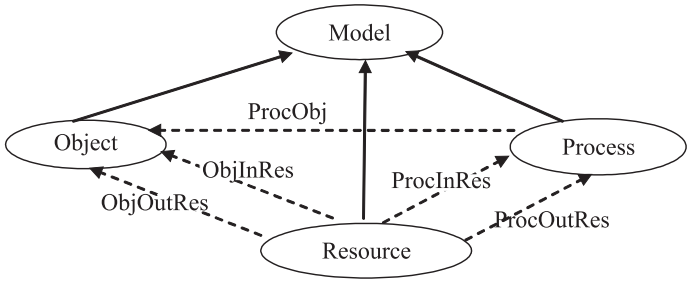

Рис. 1. Фрагмент графа бинарных отношений

ресурсов. Над множествами определены отношения принадлежности (например, ProcObj — процессы, приписанные к объекту), отношения типа «вход-выход» (например, ObjInRes — входные ресурсы объекта, ProcOutRes — выходные ресурсы процесса).

Как видно из рис. 1, между классами приведенной онтологии имеются только два вида связей:

- иерархические связи (имеется в виду иерархия включения);
- ассоциативные связи, которые имеют собственные названия (соответствуют бинарным отношениям схемы КМ);

Каждую связь онтологии можно рассматривать как частичную функцию. Стрелки, изображающие связи онтологии, направлены от областей определения таких частичных функций к областям значений.

Таким образом, сама онтология отвечает некоторой модели. Формальное описание модели онтологии в настоящей работе не приводится (см. [8]). Отметим только, что помимо ограничений, которые свойственны всем концептуальным моделям, задаваемым некоторой схемой КМ наподобие изображенной, в разработанной онтологии учитываются ограничения на отношения, которые специфичны для той или иной конкретной модели. Такие ограничения появляются в результате использования аппарата пользовательских типов для всех элементов модели и могут быть описаны в виде логических формул над предикатами первого и второго рода без кванторов.

Помимо анализа корректности модели, онтологии указанного класса могут быть использованы и для других целей — в этом преимущество декларативного способа представления знаний о структурных ограничениях. Онтология может служить основой для генерации базы данных моделирования (БДМ) и таким образом использоваться для поддержки согласованности схемы КМ и, например, реляционной БДМ. Также онтология может быть применена на этапе семантического анализа корректности запросов к БДМ. Далее в основном рассматривается использование онтологии для семантического анализа запросов к БДМ.

В силу того, что классы элементов упорядочены в иерархию, то и структура запроса должна отражать иерархичность, чтобы можно было строить запросы в терминах онтологии. Поэтому для обращения к БДМ предлагается использовать язык, опирающийся на понятие пути, как способ адресовать данные в рамках иерархии.

3. Путевые запросы

Для доступа к данным БД моделирования используются путевые (адресные) запросы следующего вида:

$$< \text{Имя\_объектной\_структуры} > [< \text{условия\_фильтрации} >]$$
$$\{. < \text{Имя\_объектной\_структуры} > [< \text{условия\_фильтрации} >]\}^{\geqslant 0}$$
$$[.\text{Имя\_атрибута}] \qquad (1)$$

Данное адресуется как свойство некоторой объектной структуры (или как сама объектная структура), которая занимает определенное место в общей иерархии включения. Для формирования условий выборки нужных данных используются фильтры. Условия фильтрации можно разделить на следующие виды:

1. Предикаты 1 порядка.
2. Предикаты 2 порядка ($<$, $>$, $=$, $\neq$), где переменные — атрибуты уровня.
3. Переход к вложенному запросу при помощи заранее оговоренных связей (ролевых связей онтологии).
4. Переход к вложенному запросу по незапланированным (нерегламентированным) связям.

Присутствие в запросе межуровневых связей, оговоренных в онтологии, дает возможность анализировать эти связи с точки зрения ограничений онтологии. Однако такой анализ целесообразно производить только после предварительного преобразования структуры запроса. Это обусловлено следующими причинами. Во-первых, различные уровни запроса могут адресовать одни и те же элементы данных, поэтому условия этих уровней целесообразно анализировать совместно, предварительно выполняя их объединение. Кроме того, наличие в запросе некоторых регламентированных связей накладывает на запрос «скрытые» ограничения, явное указание которых позволяет производить более полный анализ с точки зрения различного рода ограничений. Таким образом, в процессе

исследований выделено два класса предварительных преобразований запроса, направленных на оптимизацию его структуры:

1. Правила «склейки» вершин — выявление и отождествление узлов запроса, описывающих одно и то же множество элементов КМ.
2. Правила доопределения запроса — выявление «скрытых» ограничений запроса.

Для осуществления предварительных преобразований запрос необходимо представить в виде набора конъюнктивных запросов так, чтобы совокупность результатов их исполнения была идентична результату исходного запроса.

Определение 1. Адресный запрос вида (1) называется конъюнктивным, если все условия фильтрации имеют вид $F\&P$, где F — конъюнкция элементарных предикатов первого и второго порядка без кванторов (переменные — атрибуты уровня), а P — конъюнкция межуровневых переходов, как регламентированных, так и нерегламентированных.

Для моделирования логических формул над элементарными предикатами первого и второго порядка используется разработанный авторами аппарат условных C-систем [9], а также алгебра кортежей, основы которой приведены в работе [10].

Очевидно, что любой запрос вида (1) может быть разложен в набор конъюнктивных запросов.

В настоящей работе предлагается конъюнктивный адресный запрос представлять в виде иерархического неориентированного графа G, ребра которого соответствуют нерегламентированным связям запроса, а вершины являются своеобразными компонентами связности. Каждая такая компонента отображается древовидным ориентированным графом W, где в качестве вершин выступают уровни запроса, а в качестве дуг — связи, регламентированные онтологией. Ребрам графа G сопоставляются условные C-системы, описывающие предикаты четвертого вида (из перечисленных выше). В графе W имя каждой вершины, кроме имени компоненты связности, включает имя класса вершины и номер вложенности подзапроса. Вершинам в общем случае приписывается C-система, моделирующая условия фильтрации первого вида, а также условная C-система, соответствующая условиям фильтрации второго вида. Дуги графа W помечаются условными C-системами, моделирующими условия фильтрации третьего вида.

Таким образом, часть дуг графа W соответствует иерархическим связям онтологии, а часть — ассоциативным. Иерархические дуги имеют

зарезервированное название «point», поскольку переход вдоль иерархии осуществляется в запросе с использованием оператора «.». Ассоциативные дуги имеют то же название, что и ассоциативные связи онтологии.

Для пояснения изложенного рассмотрим пример адресного запроса «получить все процессы модели с именем „M1“, объекты которых имеют категорию „COMP“ (имеющие подчиненные объекты [7]), и входные ресурсы которых являются входными ресурсами объектов с категорией „GISC“ (состоящих их набора стандартных ГИС-элементов)»:

```
models[name='M1'].processes[(objowner=models[name='M1'].
objects[cat='COMP'].id)and(id=models[name='M1'].resources
[conso=models[name='M1'].objects[cat='GISC'].id].consp)].
```

Граф этого запроса показан на рис. 2. Он не содержит нерегламентированных связей и соответствует, в общем случае, одной компоненте связности.

Очевидно, что предварительные преобразования запроса имеет смысл (имеется возможность) производить только в рамках графа W, то есть тех частей запроса, где уровни соединены регламентированными связями. Причем часть таких преобразований хорошо поддается алгоритмизации, поскольку является следствием только иерархичности онтологии. Эти преобразования выполняются над парами вершин, не соединенных ассоциативными дугами. Другая часть преобразований специфична для каждой конкретной онтологии и не может быть задана раз и навсегда, поскольку касается тех пар вершин, между которыми существует путь, состоящий только из ассоциативных дуг.

Далее предлагается описывать специфичные преобразования запроса декларативно, в виде набора продукций.

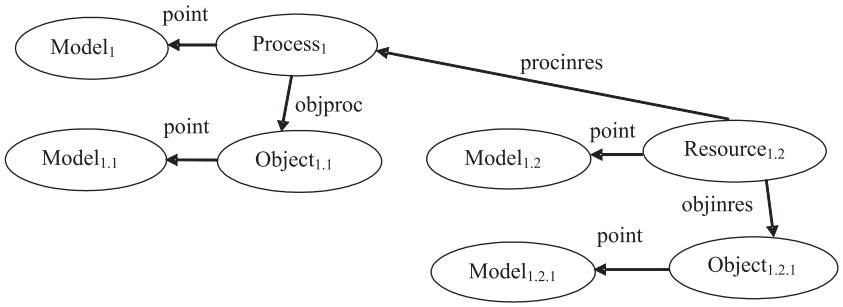

Рис. 2. Пример графа адресных запросов

4. Правила преобразования запроса

Исходя из схемы каждой конкретной концептуальной модели, для нее можно сформулировать набор правил преобразования запроса. Эти правила необходимы на стадии подготовки к проверке корректности запроса с точки зрения ограничений онтологии. Кроме того, в ходе применения правил заранее (до стадии исполнения запроса) выявляются некорректности в описании вершин запроса. Ниже разрабатывается метод представления наборов правил преобразования запроса и условий их применимости в виде систем продукций, причем каждой схеме концептуальной модели (точнее, каждой онтологии) соответствует свой набор правил, а значит, и своя система продукций. При построении формальной модели системы продукций использован аппарат, предложенный в работе [11].

Введем понятия из теории моделей в том объеме, в котором они понадобятся для дальнейшего изложения.

Согласно терминологии работы [11], многосортная алгебра $A = \langle (s^A)_{s\in S}, (f^A)_{f\in F}\rangle$ состоит из семейства множеств-носителей s^A и семейства частичных функций f^A:

$$f^A:\ s_1^A \times s_2^A \times\ \ldots\ \times s_{n-1}^A \to s_n^A, \quad n \geqslant 0. \tag{2}$$

Сигнатура $\Sigma = (S, F)$ состоит из непустого множества S символов для обозначения индексов множеств-носителей, называемых также сортами, и непустого множества F функциональных символов, каждому из которых приписана схема отображения $f : s_1 \times s_2 \times \ldots \times s_{n-1} \to s_n$, $n > 0$, означающая запись арности функции, сортность ее аргументов и результата. Для сигнатуры $\Sigma = (S, F)$ многосортная алгебра $A = \langle (s^A)_{s\in S}, (f^A)_{f\in F}\rangle$ называется Σ-алгеброй, если схемы отображений для всех $f \in F$ согласованы с отображениями f^A.

В рассматриваемой задаче в качестве множества сортов S многосортной алгебры примем множество классов онтологии, например, объекты, процессы, ресурсы и так далее. Семейство частичных функций f^A формализует связи, заданные в онтологии. Так как связи онтологии определяются бинарными отношениями схемы концептуальной модели и ее фрагментов, то эти функции имеют схему $f : s_1 \to s_2$.

Пусть задана сигнатура $\Sigma = (S, F)$. С каждым сортом $s_i \in S$ свяжем счетное множество X_s переменных и определим множество термов TR и функцию тип: $TR \to S$ следующим образом:

- всякая переменная $x \in X_s$ есть терм, причем тип$(x) = s$;
- всякий нуль-арный функциональный символ (константа) $f \in F$ со схемой отображения $f :\to s$ есть терм, причем тип$(f) ::= s$;

- если $f \in F$ имеет схему отображения

$$f:\ s_1 \to s_2 \tag{3}$$

и t_1, t_2 — термы, у которых тип$(t_1) = s_1$, тип$(t_2) = s_2$, то

$$f(t_1, t_2) \text{ — терм, тип } (f(t_1, t_2)) = s_2. \tag{4}$$

Определение 2. Назовем фактом упорядоченный список вида (f, t_1, t_2), где $f \in F$, $f:\ s_1 \to s_2$, $t_i \in TR$, тип $(t_i) = s_i$, $i = \overline{1,2}$ или список вида (тип, t, s), где $t \in TR$, $s \in S$, тип$(t) = s$. В этих обозначениях *ситуацией* назовем конечную конъюнкцию фактов. В дальнейшем через D будем обозначать множество возможных ситуаций.

Каждый граф W может быть описан в виде ситуации, все факты которой содержат только константы. Например, ситуация, описывающая запрос из рис. 2, выглядит следующим образом:

$$\begin{aligned}
&\text{point}(\text{Process}_1, \text{Model}_1) \wedge \text{point}(\text{Object}_{1.1}, \text{Model}_{1.1}) \wedge \\
&\wedge \text{point}(\text{Resource}_{1.2}, \text{Model}_{1.2}) \wedge \text{point}(\text{Object}_{1.2.1}, \text{Model}_{1.2.1}) \wedge \\
&\wedge \text{objproc}(\text{Process}_1, \text{Object}_{1.1}) \wedge \text{procinres}(\text{Resource}_{1.2}, \text{Process}_1) \wedge \\
&\wedge \text{objinres}(\text{Resource}_{1.2,}\text{Object}_{1.2.1}) \wedge \text{тип}(\text{Model}_1, \text{модель}) \wedge \\
&\wedge \text{тип}(\text{Model}_{1.1}, \text{модель}) \wedge \text{тип}(\text{Model}_{1.2}, \text{модель}) \wedge \\
&\wedge \text{тип}(\text{Model}_{1.2.1}, \text{модель}) \wedge \text{тип}(\text{Process}_1, \text{процесс}) \wedge \\
&\wedge \text{тип}(\text{Resource}_{1.2}, \text{ресурс}) \wedge \text{тип}(\text{Object}_{1.1,}\text{объект}) \wedge \\
&\wedge \text{тип}(\text{Object}_{1.1.2,}\text{объект}).
\end{aligned}$$

Отметим, что факты вида (тип, t, s) описывают вершины графа запроса W (компоненты связности запроса), а факты вида (f, t_1, t_2) описывают дуги этого графа.

Определенная таким образом ситуация представляет собой множество конъюнктивно-связанных фактов, и поэтому в дальнейшем будем обращаться с нею как со множеством, используя операции объединения $\cup$, пересечения $\cap$, разности $\setminus$ и отношение включения $\supseteq$.

Обозначим через $var(d)$ множество имен переменных, входящих в ситуацию d, а через $T(d) ::= \{\text{тип}(x) | x \in var(d)\}$ — множество типов переменных, входящих в факты ситуации d.

Введем также понятие подстановки:

$$\theta ::= \{t_1/v_1, t_2/v_2, \ldots, t_m/v_m\},$$

где t_i — термы, v_i- переменная, тип $(t_i) = \text{тип}(v_i)$, $i = \overline{1,m}$. Запись $d\theta$ означает результат одновременной замены всех вхождений переменной v_i

в d на соответствующий терм t_i. По умолчанию, термы t_i будем полагать константами.

Подстановку $\theta_a ::= \{x_1/y_1, x_2/y_2, \dots, x_m/y_m\}$, где x_i, y_i $(i = \overline{1, m})$ — переменные, назовем *алфавитным вариантом*. Для каждого алфавитного варианта определена *обратная подстановка*: $\theta_a^{-1} = \{y_1/x_1, y_2/x_2, \dots, y_m/x_m\}$.

Две ситуации d и d' назовем *эквивалентными* $(d = d')$, если найдется алфавитный вариант θ_a такой, что: $d \supseteq d'\theta_a$ и $d' \supseteq d\theta_a^{-1}$.

Определим элементарные операции преобразования ситуации, к которым относятся добавление фактов и переобозначение термов. Для фиксированного $d' \in D$:

a) операция добавления:

$$A[d'] : D \to D : A[d'](d) = d \cup d'; \tag{5}$$

b) операция переобозначения термов

$$RW[\theta_T] : D \to D : RW[\theta_T](d) = d\theta_T, \tag{6}$$

где $\theta_T = (t_1/e_1, t_1/e_2)$, причем t_1, e_1, e_2 — имена термов (все константы или все переменные), терм t_1 назовем замещающим, а термы e_1, e_2 — замещаемыми.

Операция переобозначения нужна для того, чтобы отождествить пару термов, описывающих одну и ту же сущность. Например, одна и та же константа в ситуации может быть обозначена по-разному, и применение к такой ситуации операции переобозначения термов означает введение единого обозначения этой константы. При выполнении этой операции не происходит как такового удаления фактов, а исключается дублирование фактов, отличающихся только обозначением эквивалентных термов.

Определим множество *программ* R преобразования ситуаций следующим образом. Во-первых, будем считать элементами R программы $A[d'], RW[\theta_T]$ при любых $d' \in D$ и θ_T, во-вторых, если две программы $r_1, r_2 \in R$, то программа $(r_1; r_2)$, определенная посредством равенства $(r_1; r_2)(d) ::= r_2(r_1(d)) \forall\, d \in D$, также есть элемент R.

Определение 3. Программу r^+, содержащую только операции типа $A[d'] \forall\, d' \in D$, назовем *позитивной*. Заметим, что если $d_2 = r^+(d_1)$, то $d_2 \supseteq d_1$.

Через $r\theta$, где $\theta ::= \{t_1/x_1, \dots, t_m/x_m\}$ — произвольная подстановка, обозначим программу r, во всех операциях которой аргументы-переменные x заменены на сопоставленные им в θ термы t_i, $i = \overline{1, m}$. Переменные программы, которым не сопоставлены в подстановке θ никакие термы, заменяются на новые (еще не использованные) переменные из множества X_s, $s \in S$.

Как уже указывалось, весь набор правил преобразования запроса можно разделить на два класса: правила «склейки» вершин и правила доопределения ситуации (запросов). Правила «склейки» вершин призваны выявить вершины графа запроса, которые фактически описывают (адресуют) одно и то же множество однотипных элементов базы данных моделирования, и отождествить их. Такие вершины далее будем называть *эквивалентными*. Для пары вершин A и B отождествление происходит следующим образом: итоговая вершина наследует все входы и выходы исходных вершин, а условия вершин A и B конъюнктивно соединяются. Если в итоге получается тождественно ложная логическая формула, то запрос некорректен.

Основной задачей правил доопределения ситуации является добавление в граф запроса ребер (связей), которые выводятся из связей, уже присутствующих в запросе. В результате этого, при анализе запроса появляется возможность представить в явном виде ограничения на вновь добавляемые связи, то есть «скрытые» ограничения запроса.

Весь набор указанных правил представим в виде набора продукций.

Определение 4. *Продукцией* назовем кортеж $\langle \text{cond}, q, r \rangle$, в котором q — ситуация, называемая условием применимости продукции, r — программа $(r \in R)$ называемая действием, а cond — дополнительное условие применимости продукции (необязательная часть). Дополнительное условие применимости формулируется на языке условных C-систем, которые являются значениями переменных, входящих в q. В частности, в правилах «склейки» такие условия активно используются, а в правилах доопределения ситуации не используются.

Системой продукций назовем конечное множество кортежей

$$Pr = \{\langle \text{cond}, q, r \rangle\}.$$

В полученной вышеописанным способом системе продукций Pr, каждая продукция соответствует либо правилу доопределения запроса, либо правилу «склейки». Факты, входящие как в условия применимости, так и в программы, содержат только переменные, поскольку продукции предназначены для задания схем правил, а не отдельных правил.

Теперь рассмотрим организацию логического вывода на базе такой системы продукций. Будем говорить, что d_2 *непосредственно выводима* из d_1 посредством продукции $pr = \langle \text{cond}, q, r \rangle$, $(d_1 \xrightarrow{pr} d_2)$, если найдется такая подстановка θ, что $d_1 \supseteq d_f \supseteq q\theta(d_f)$ — фрагмент, затрагиваемый программой r, а $d_2 = r\theta(d_f) \cup (d_1 \backslash d_f)$, причем дополнительное условие cond является истинным.

Если найдется последовательность продукций

$$pr_1, pr_2, \ldots, pr_k, pr_i \in Pr, \quad i = \overline{1,k}, \quad k \geqslant 0,$$

и состояний базы $d_0, d_1, \ldots, d_k$ таких, что

$$d_0 \xrightarrow{pr_1} d_1 \xrightarrow{pr_2} \ldots \xrightarrow{pr_k} d_k,$$

то говорим, что d_k выводима из d_0, и пишем $d_0 \xrightarrow{*} d_k$ или

$$d_0 \xrightarrow{pr_1} \ldots \xrightarrow{pr_k} d_k,$$

а $pr_1, pr_2, \ldots, pr_k$ назовем последовательностью применимых к d_0 продукций. Если

$$d_0 \xrightarrow{*} d_k \quad \text{и} \quad \forall\, pr\ (d_k \xrightarrow{pr} d' \Rightarrow d' = d_k),$$

то d_k — *результирующая ситуация* для d_0.

Результирующая ситуация зависит в общем случае (при использовании операций удаления фактов) от выбора подстановки и порядка применения продукций, то есть неоднозначна.

Пусть задано исходное состояние d_0 и система продукций

$$Pr = \{pr_i = \langle \underset{i}{\text{cond}}, \quad q_i, \quad r_i \rangle\}, \quad i = \overline{1,n}, \quad n > 0.$$

Назовем Pr *корректной* на d_0, если не существует бесконечной последовательности применимых к d_0 продукций, и для любых двух результирующих ситуаций d' и d'', выводимых из d_0, выполнено $d' = d''$.

Особо отметим, что в результате применения правил доопределения ситуации исходная ситуация дополняется только фактами вида (f, t_1, t_2), которые соответствуют ребрам графа запроса. При этом удаления каких-либо фактов не происходит. Таким образом, последовательное применение правил (программ продукций) этого класса реализует позитивную программу.

Правила «склейки» описывают всевозможные пары эквивалентных вершин и способ их отождествления. Каждое правило сводится к выполнению проверки истинности условия cond с последующим проведением операции переобозначения термов и исключением повторяющихся фактов. Если в запросе имеются два различных обозначения одной и той же константы (уровня запроса, вершины графа W), то подстановка θ формируется таким образом, чтобы замещающий терм $c_3 \in \{c_1, c_2\}$ подстановки $\theta_T\theta = (c_3/c_1, c_3/c_2)$ имел наименьший индекс из индексов замещаемых термов c_1 и c_2 (индексы сравниваются как строки).

Приведем примеры продукций (схем правил) преобразования запроса для фрагмента онтологии, изображенного на рис. 1.

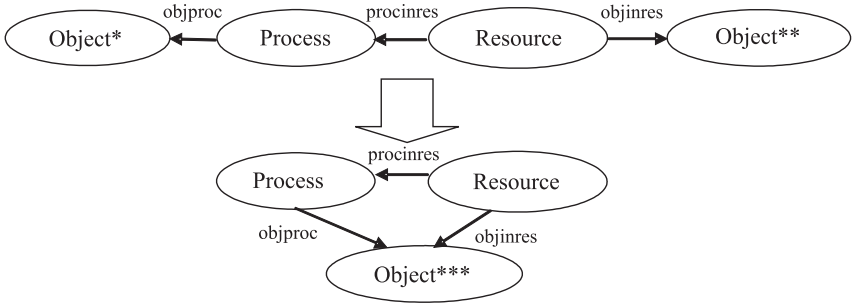

Рис. 3. Пример правила «склейки» вершин

Пример 1. В примере показано, как производится «склейка» пары вершин, сначала в виде графовой схемы (рис. 3.), а затем и на языке продукций.

Запишем продукцию $pr_1 = \langle \text{cond}_1, q_1, r_1 \rangle$, соответствующую этой графовой схеме:

$$\underset{1}{\text{cond}}: \ \text{Object}^{***} = \text{Object}^{*} \cap \text{Object}^{**},$$

где Object^{*}, Object^{**}, Object^{***} — C-системы, соответствующие вершинам графа;

q_1 тип(Object*, объект) $\wedge$ тип(Object**, объект)$\wedge$

$\wedge$ тип(Process, процесс) $\wedge$ тип(Resource, ресурс)$\wedge$

$\wedge$ objproc(Process, Object*) $\wedge$ procinres(Resource, Process)$\wedge$

$\wedge$ objinres(Resource, Object**);

$r_1 = RW[\theta_T](d)$, где d — текущее состояние запроса,

$\theta_T = (\text{Object}^{***}/\text{Object}^{*}, \text{Object}^{***}/\text{Object}^{*})$.

Если условия вершины «Object*» противоречат условиям фильтрации, накладываемым на вершину «Object**», то конъюнктивный запрос считается некорректным. Иначе в качестве «Object***» выбирается вершина с наименьшим индексом.

Пример 2. иллюстрирует вывод дополнительных связей на основе связей, присутствующих в запросе. Конкретно, если в запросе присутствует ситуация «набор ресурсов является входным одновременно для некоторых наборов объектов и процессов» (рис. 4.), то наборы процессов и объектов должны быть связаны отношением «objproc»: каждый процесс приписан некоторому объекту из набора.

Запишем продукцию $pr_3 =< q_3, r_3 >$, соответствующую этой графовой схеме:

q_3 = тип(Object, объект) $\wedge$ тип(Resource, ресурс) $\wedge$ тип(Process, процесс) $\wedge$

$\wedge$ objinres(Resource, Object) $\wedge$ procinres(Resource, Process);

$r_3 = A[d'](d)$, где d — текущее состояние запроса,

d' = objproc(Process, Object).

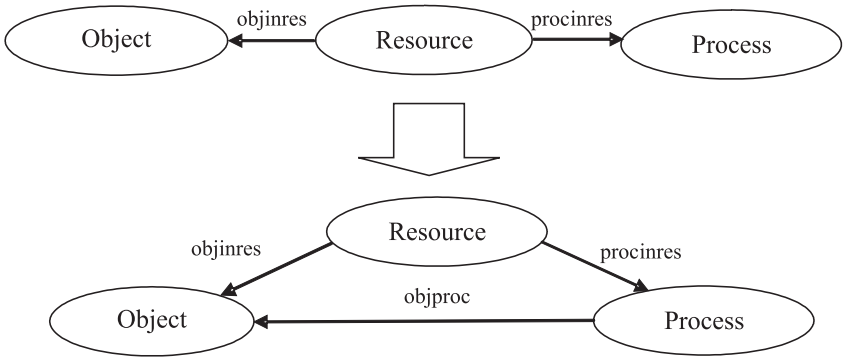

Рис. 4. Пример правила доопределения запроса

Из приводимых примеров видно, что условия применимости продукций представляют собой цепочки вида: $X_1 f_1 X_2 \dots X_{n-1} f_{n-1} X_n$, где $X_1, \dots, X_n$ — попарно различные переменные, а $f_1, \dots, f_{n-1}$ — названия регламентированных связей запроса.

Определение 5. Крайними фактами условия применимости $q = \{\text{fact}\}$, где fact — некоторый факт, входящий в q, назовем множество фактов $F^e = \{\text{fact}_i^e\}$ вида (тип, t, s) таких, что каждая переменная $\text{var}(f_i^e)$ входит ровно в один факт $\text{fact} \in q$ вида (f, t_1, t_2). Множество крайних фактов условия применимости будем обозначать $E(q)$, причем $card(E(q)) = 2$, поскольку q является цепочкой.

Теорема. *Пусть* $Pr = \{pr_i\}$ *- система продукций, такая, что каждая продукция* $< \text{cond}_i, \text{q}_i, \text{r}_i >$ *соответствует либо правилу «склейки» вершин, либо правилу доопределения запроса. Если для любой пары продукций* pr_1, pr_2, *где* pr_1 *- правило доопределения или правило «склейки», а* pr_2 *— правило «склейки», выполняется соотношение* $E(q_2\theta_a) \not\subset q_1 \backslash E(q_1)$ (θ_a *— некоторый допустимый алфавитный вариант*), *то система продукций* Pr *корректна.*

Доказательство теоремы вынесено в приложение.

В результате выполнения правил преобразования либо устанавливается некорректность запроса, либо получается конъюнктивный запрос, пригодный для анализа с точки зрения ограничений запроса (связаны с организацией вложенности) и ограничений онтологии.

Такие ограничения моделируются в виде условных C-систем [9], а весь дальнейший анализ сводится к наполнению условных C-систем на основе значений переменных запроса и нахождению пересечений с C-системами [10], моделирующими запрос.

Таким образом, предлагаемый метод позволяет производить анализ корректности запроса до стадии его исполнения.

5. Заключение

В работе представлен метод упрощения нерегламентированных адресных запросов путем отождествления некоторых уровней запроса и явного указания «скрытых» ограничений. Метод позволяет выявлять некорректности в описании запроса не только до стадии исполнения, но и до стадии его основного анализа. Главное отличие предлагаемого метода от существующих состоит в том, что как при анализе запроса, так и в процессе его оптимизации активно используется семантическая информация о содержимом базы данных. Эта информация систематизируется в разработанной авторами специализированной онтологии, которая позволяет задавать структурные ограничения на связи между концептами в виде логических формул. Дополнительная семантическая информация, касающаяся правил преобразования запроса, формализуется в форме системы продукций.

Разработанный метод апробирован при реализации системы ситуационного концептуального моделирования промышленно-природных комплексов. Компонент доступа к данным этой системы моделирования снабжен редактором онтологии структурных ограничений. Редактор позволяет вводить дополнительные ограничения на отношения схемы концептуальной модели и является средством автоматизированного согласования схемы модели и структуры базы данных моделирования. Несмотря на то, что база данных моделирования имеет иерархическую структуру, физически для хранения информации используется реляционная СУБД. Онтология структурных ограничений призвана максимально учесть семантические ограничения на отношения, которые вытекают из схемы концептуальной модели, а также из процедур анализа корректности структуры модели предметной области.

Литература

1. *Дейт К. Дж.*, Введение в системы баз данных. 6-е издание / Пер. с англ. К.; М.; СПб.: Вильямс, 2000. 848 с.
2. OWL, язык веб-онтологий. Руководство.[Электронный ресурс] http://www.sherdim.rsu.ru/pts/semantic_web/REC-owl-quide-20040210_ru.html
3. *Провост У.* За пределами W3C XML Schema.[Электронный ресурс] http://www.iso.ru/journal/articles/174.html
4. *Михаленко П.* Язык онтологий в Web.[Электронный ресурс] http://www.osp.ru/os/2004/02/183921/_p2.html

5. *Доддз Л.* XML и базы данных? Доверьтесь своей интуиции.[Электронный ресурс] http://www.iso.ru/cgi-bin/main/journal.cgi?do_what=details&id=206
6. *Ширшов А.* Использование XML совместно с SQL.[Электронный ресурс] http://www.rsdn.ru/article/db/xmlsql.xml
7. *Фридман А. Я.* Ситуационный подход к моделированию промышленно-природных комплексов и управлению их структурой. Труды IV международной конференции «Идентификация систем и задачи управления» (М., 25–28 января г. 2005). Институт проблем управления им. В. А. Трапезникова. М.: Институт проблем управления им. В. А. Трапезникова, 2005. С. 1075–1108.
8. *Зуенко А. А.*, *Фридман А. Я.* Управление контекстом при организации интеллектуализированного интерфейса БД в системах моделирования на основе концептуального подхода (настоящий сборник).
9. *Зуенко А. А.*, *Фридман А. Я.* Развитие алгебры кортежей для логического анализа баз данных с использованием двухместных предикатов // Известия РАН. Теория и системы управления. 2009. № 2. (В печати).
10. *Кулик Б. А.* Логико-вероятностный анализ интеллектуальных систем на основе алгебры кортежей // Труды международной научной школы «Моделирование и анализ безопасности и риска в сложных системах 2007» (СПб., 4–8 сентября 2007 г.). СПб.: ГОУ ВПО «СпбГУАП», 2007. С. 137–149.
11. *Яхно Т. М.* Системы продукций: структура, технология, применение. Новосибирск: Изд-во ВЦ СО АН СССР, 1990. 128 с.

Приложение

Доказательство теоремы. Сначала докажем *устойчивость*. Программы в системе продукций оперируют только с переменными и фактически представляют собой схемы правил. Правила доопределения запросов только добавляют факты вида (f, t_1, t_2) и не приводят к неприменимости того или иного правила. Правила «склейки» лишь переобозначают константы в ситуациях, соответствующих запросам, в результате чего происходит отождествление идентичных фактов, при этом схема факта остается неизменной. Утверждение $E(q_2\theta_a) \not\subset q_1 \backslash E(q_1)$ говорит о том, что если цепочка, соответствующая условию применимости правила pr_2, не лежит обоими своими концами внутри цепочки, которая соответствует условию применимости правила pr_1, то выполнение правил «склейки» не оказывает влияния на применимость правила pr_1, поскольку не приводит к усечению цепочки q_1.

Теперь докажем выполнение условия *коммутативности* продукций. Пусть имеем текущую ситуацию d и применимую к ней пару продукций pr_1 и pr_2 (выбраны произвольным образом). Это означает, что существуют $q_1\theta_1 \subseteq d_1$, и $q_2\theta_2 \subseteq d_f \subseteq d_1$, где d_f — фрагмент, затрагиваемый

переобозначением, причем θ_1 содержит *попарно* различные константы. Пара продукций pr_1 и pr_2 может включать:

a) два правила «склейки»,
b) два правила доопределения ситуаций,
c) правило «склейки» и правило доопределения ситуаций.

В случае (b) r_1 и r_2 — позитивные программы. Для продукций с позитивными программами свойство коммутативности доказано в работе [11].

Рассмотрим случай (c). Для определенности будем считать, что pr_1 — правило доопределения ситуации, а pr_2 — правило «склейки». Тогда

$$d_k = r_1\theta_1(q_1\theta_1) \cup (d \backslash q_1\theta_1), \quad \text{где} \quad r_1(q_1\theta_1) = q_1\theta_1 \cup d^*\theta_1,$$

причем d^* — это ситуация (факт), добавляемая программой r_2. Очевидно, что $d \subseteq d_k$. Тогда

$$d_k = q_1\theta_1 \cup d^*\theta_1 \cup (d \backslash q_1\theta_1) = d \cup d^*\theta_1.$$

Отсюда следует возможность, по-прежнему, выполнить подстановку θ_2, такую, что

$$d' = r_2\theta_2(d \cup d^*\theta_1) = r_2\theta_2(d_f) \cup (d \backslash d_f) \cup r_2\theta_2(d^*\theta_1) = \\ = d_f(\theta_T\theta_2) \cup (d \backslash d_f) \cup d^*\theta_1(\theta_T\theta_2),$$

где запись $\theta_T\theta_2$ означает подстановку, в которой все переменные θ_T заменены на их значения из θ_2. Теперь рассмотрим применение сначала продукции pr_2, а затем pr_1. Тогда

$$d_m = r_2(d) = r_2\theta_2(d_f) \cup (d \backslash d_f) = d_f\theta_T\theta_2 \cup (d \backslash d_f).$$

Согласно свойству устойчивости мы, по-прежнему, имеем возможность добавить ситуацию d^*, но будет другая подстановка $\theta_1' = \theta_1\theta_T\theta_2$, которая означает, что в θ_T все переменные заменены на значения из θ_2, а затем все термы θ_1 переписаны с учетом $\theta_T\theta_2$. Поскольку $E(q_2\theta_a) \not\subset q_1 \backslash E(q_1)$, то в θ_1 происходит перепись всего одной константы, а значит, θ_1', по-прежнему, содержит попарно различные константы. Отличие θ_1' от θ_1 лишь в том, что обозначение некоторой константы заменено на эквивалентное. Таким образом, подстановка θ_1' эквивалентна последовательному применению к ситуации d* подстановок $\theta_1, \theta_T\theta_2$ и не возникает неоднозначности при ее применении. Следовательно:

$$d'' = r_1\theta_1'(d_f\theta_T\theta_2 \cup (d \backslash d_f)) = \\ = d_f\theta_T\theta_2 \cup (d \backslash d_f) \cup d^*\theta_1' = d_f\theta_T\theta_2 \cup (d \backslash d_f) \cup d^*\theta_1(\theta_T\theta_2).$$

Очевидно, что $d' = d''$. Доказательство для случая a) проводится аналогично.

Конечность вычислений обосновывается следующими соображениями. Правила «склейки» соответствуют отождествлению вершин графа запроса, а их число конечно. Новые фрагменты (которых не было в исходной ситуации), пригодные для применения правил «склейки», могут получиться только в результате применения правил доопределения. Если применяются только правила доопределения, то рано или поздно между всеми или только некоторыми вершинами графа запроса будут установлены все допустимые связи, оговоренные в онтологии. Если происходит «переключение» между этими двумя видами правил, то при каждом таком переключении количество вершин уменьшается и (или) количество ребер увеличивается. Этот процесс не может длиться бесконечно в силу конечности числа вершин и количества допустимых связей между парой вершин в онтологии.

Интеграция данных на основе онтологий для обеспечения информационной поддержки управленческих решений

П. А. Ломов, М. Г. Шишаев

Институт информатики КНЦ РАН, Апатиты

1. Введение

В наше время принятие своевременных и обоснованных решений в процессе управления какой-либо макросистемой, будь то социально-экономическая, банковская, государственная система становиться возможным только на основе обработки больших объемов различной информации с использованием информационных технологий и моделирования возможных последствий реализации принимаемых решений. Выполнение данной задачи предполагает создание единого унифицированного интерфейса для некоторой совокупности независимых источников данных, которыми могут выступать традиционные системы баз данных, Web-сайты, файлы структурированных данных. Одной из важных проблем при этом является наличие того факта, что, как правило, обрабатываемые источники данных являются гетерогенными. Их гетерогенность проявляется на различных уровнях: системном, синтаксическом, структурном, семантическом, что является следствием использования различных программных и аппаратных архитектур, на которых реализованы информационные системы, а также применении различных форматов и моделей представления данных. Необходимо также отметить, что множество источников может также расширяться со временем. Все это делает проблему интеграции данных довольно сложной и многоуровневой, для решения которой следует в обязательном порядке принимать во внимание, как структурные и синтаксические различия информационных ресурсов, так и семантику их отдельных информационных элементов, для обеспечения смысловой интероперабельности данных и разрешения семантических конфликтов.