РОССИЙСКАЯ АКАДЕМИЯ НАУК

ВСЕРОССИЙСКИЙ ИНСТИТУТ НАУЧНОЙ И ТЕХНИЧЕСКОЙ ИНФОРМАЦИИ
(ВИНИТИ)

НАУЧНО • ТЕХНИЧЕСКАЯ
ИНФОРМАЦИЯ

Серия 2. ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ И СИСТЕМЫ

ЕЖЕМЕСЯЧНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ СБОРНИК

Издается с 1961 г. № 3 Москва 2012

ИНФОРМАЦИОННЫЕ СИСТЕМЫ

УДК 004.4 : 004.774

Е. М. Бениаминов, В. А. Лапшин

Уровни представлений онтологий, языки, математические модели и проект Веб-сервера онтологий в стиле Веб 2.0*

Рассматривается проект Веб-сервера онтологий в стиле Веб 2.0, предоставляющий возможности коллективной разработки онтологий. Обсуждаются уровни представления онтологий и математические модели, лежащие в основе этих представлений. Приведен обзор языка алгебраических спецификаций CASL.

Ключевые слова: *онтология, уровни представлений онтологий, Веб-сервер онтологий, алгебраические спецификации, язык CASL*

1. ВВЕДЕНИЕ

В последние годы в Интернет стали очень популярными системы, которые поддерживают процессы объединения людей по интересам, так называемые социальные сети. В таких системах пользователям предоставляется электронная среда в сети Интернет для организации объединения пользователей, для ввода разнородных данных и доступа к данным, а наполнение системы данными производится самими пользователями. Технология создания этих систем получила название технологии Веб 2.0 [1].

Для каждого объединения людей в таких системах определяется область интересов, лежащих в основе их объединения и целей объединения. Степень точности описания области интересов может быть различной и иметь в различной степени развитую онтологию и словарь, которыми владеют участники группы. Онтология – это формальное описание договоренности группы людей о том, что и как у них называется, какими свойствами может обладать, в каких отношениях участвует и каким ограничениям удовлетворяет [2 – 4]. Онтология группы может уточняться и, как правило, уточняется в процессе объединения людей. Более того, по мере определения онтологии предметной области

* Работа выполнена при финансовой поддержке РФФИ (проект 09-07-00079-а)

социальной группы в этой группе формируется свой язык, на котором выражаются данная онтология и способы ее использования. Степень формальности такого языка зависит от традиций, особенностей предметной области, лежащей в основе интересов данной группы, ее близости к общеупотребимой – бытовой области знаний, а также от желания и возможности использования в данной области компьютерных моделей.

В настоящей работе обсуждаются средства, которые могут предоставляться пользователям для компьютерной поддержки процессов коллективного формирования онтологий и языка социальной группы. Развитие этих средств шло в следующем направлении. На некотором этапе развития технологии Веб 2.0 стала понятной потребность пользователей в классификации вводимых ими материалов. Для удовлетворения этой потребности в технологию Веб 2.0 были введены средства формирования и использования таксономий[1]. Пользователям в таких системах была обеспечена возможность определять собственные элементы таксономии и использовать их для разметки вводимых материалов (страниц, фотографий, видеоклипов и т. д.), а также для разметки на формируемых ими страницах ссылок на другие страницы посредством произвольно выбираемых меток, называемых тегами. Такая практика спонтанного сотрудничества группы людей с целью совместной категоризации информации получила название *фолксономии* [5].

Кроме того, пользователям стали предоставлять языки использования таксономий и разметок по таксономиям материалов и ссылок для поиска и отображения материалов. Эти языки предоставляются либо в строгом формате, который пытаются довести до некоторого стандарта, либо в виде графических интерфейсов.

Одновременно с этим, в рамках направления *семантическая паутина* [6], развивалось понимание важности использования формальных моделей онтологий для организации связей между данными в Интернет и построения языков представления онтологий. Использование онтологий в системах стало получать все более широкое распространение. В связи с этим большое значение приобрели работы по стандартизации языков представлений онтологий и построение онтологий, разделяемых большим числом пользователей. Однако большие онтологии обычно являются коллективной разработкой и складываются в процессе согласований и отладки. Поэтому возникает потребность в программных средствах, поддерживающих коллективную разработку онтологий. Таких систем пока немного[2]. Наиболее развитой среди них является система *Ontolingua* [8, 9], разработанная в Стэнфордском университете в лаборатории систем знаний (KSL).

В данной статье описывается проект разработки системы в стиле Веб 2.0 для коллективного формирования баз онтологий и языков работы с ними. В проекте предлагается разработка Веб-сервера в стиле Википедия [10], но для онтологий. То есть предполагается создать среду в Интернет, в которой сами пользователи коллективно наполняют систему онтологиями и создают библиотеки онтологий по различным областям знаний. Онтология представляет собой фиксацию на формальном открытом языке договоренностей пользователей о том, что как называется в их области, какими свойствами обладает и каким ограничениям удовлетворяет. Предполагается, что открытый формальный язык, на котором описываются онтологии, также коллективно формируется пользователями по мере пополнения системы онтологиями и шаблонами языка, введенными в онтологиях. В системе должны поддерживаться модульность, многоверсионность разработки онтологий, управление доступом к версиям онтологий и словарям системы, а также средства поиска и отладки онтологий и языка системы. Онтологии, хранящиеся на разрабатываемом Веб-сервере, предназначены для многоразового многоцелевого компьютерного использования в задачах модульного построения формальных информационных моделей сложных объектов и их анализа.

Особое значение для коллективной разработки онтологий имеют модульная организация сложных онтологий и использование операций над онтологиями. Богатый опыт по модульной организации спецификаций накоплен в алгебраическом подходе к спецификации онтологий. В данной работе обсуждаются основы этого подхода и принципы организации алгебраического языка спецификаций CASL (Common Algebraic Specification Language) [11], который был разработан сообществом CoFI (Common Framework Initiative for algebraic specification and development) [12].

Для эффективного использования онтологий требуется многоуровневое их представление и использование специфических языков и средств на каждом уровне. Уровням представления онтологий посвящен следующий раздел.

2. УРОВНИ И ЯЗЫКИ ПРЕДСТАВЛЕНИЙ ОНТОЛОГИЙ

Использование онтологий в компьютерных системах связано с несколькими уровнями представления онтологий.

2.1. Уровни представления онтологий

Ниже описаны три уровня представления онтологии.

Верхний уровень. Это уровень представления пользователей. На этом уровне онтологии представляются для понимания пользователями и для формирования новых онтологий. В настоящее время верхний уровень в основном отображается в графических редакторах онтологий типа Protege [13] или KAON [14]. Однако сложные аксиомы онтологий в этих редакторах приходится задавать в логических языках, которые пользователи должны изучать, если хотят пользоваться этим инструментом, и никак не могут подстроить его для своей предметной области. Жесткость и чуждость таких языков представления онтологий для специалистов в предметных областях,

[1] Таксономия является простейшим типом онтологий, в которых ключевыми элементами являются понятия класса, элемента класса, свойств класса и элементов, отношений класс – подкласс.

[2] Представляется достойным внимания подход, используемый в проекте Semantic Wiki [7]. Целью этого проекта является создание инструмента, позволяющего пользователям производить семантическую разметку добавляемых в Вики текстов.

далеких от программирования и математики, является основной проблемой, препятствующей массовому использованию этих технологий. В основном этими технологиями пользуются программисты для создания приложений, которые уже разрабатываются для конкретных задач пользователей.

Средний уровень. Это уровень представления онтологий программистами и уровень межмашинного обмена онтологиями. Средний уровень развивался очень бурно. В настоящее время на этом уровне имеются стандарты: для языков представления – OWL 2.0 [15], языков запросов – SPARQL [16] и языков описания ограничений – SWRL [17] и RIF [18]. Этот уровень описан во многих докладах и работах, поэтому здесь он особо обсуждаться не будет.

Нижний уровень. Это уровень представления онтологий в виде набора простейших фактов из бинарных отношений (троек). Для этого уровня разработан язык представления RDF [19]. Язык запросов SPARQL обращается к уровню представления онтологий на языке RDF.

Для работы с онтологиями на уровне выполнения запросов и отображения онтологий, а также для организации взаимодействия онтологий с базами данных используется загрузка онтологий, представленных в виде RDF, в программную среду в виде RDF-хранилищ. Для этого используются специальные программные средства типа JENA [20] и SESAM [21] или программные средства СУБД ORACLE [22], а также некоторые программные среды логического вывода.

2.2. Отображения уровней

Взаимные отображения между уровнями представлений онтологий не всегда однозначны и полны.

Отображение с верхнего уровня на уровень OWL и обратно стараются полностью автоматизировать. Однако некоторые ограничения, сформулированные на уровне пользователя, могут представляться в OWL в виде комментария и не использоваться при компьютерной обработке.

Язык OWL может оказаться недостаточным для отображения пользовательских ограничений. Для расширения языка ограничений язык OWL дополняется различными версиями языков правил SWRL или RIF, выражения которых позволяют генерировать новые факты онтологий из существующих.

Отображение с уровня OWL на уровень RDF также может быть неоднозначным и определяться конкретным программным средством. Например, если некоторое отношение в OWL объявлено транзитивным, то на RDF-уровне могут быть отображены все факты транзитивного замыкания исходного отношения, а могут быть отображены исходные факты и правило транзитивного замыкания в программной среде RDF-хранилища.

В общем случае пользовательское определение онтологии, заданное словарем онтологии и ограничениями (аксиомами), может задать неразрешимую теорию. Это значит, что в такой онтологии не существует общего алгоритма для построения ответов на все вопросы. Это значит, что система, использующая такую онтологию, не сможет ответить на все вопросы, которые мы сможем сформулировать к ней, и теория, которая соответствует такой онтологии, не является полной. Неполнота теорий, соответствующих общим онтологиям, является их специфическим качеством[3].

Однако для каждой онтологии и уровня подробности ее представления в виде RDF-хранилища определяется фрагмент теории (подмножество запросов), для которого в данной программной среде система эффективно строит ответы. Такой фрагмент можно назвать *вычислительной аппроксимацией* данной онтологии. Таким образом, помимо онтологии возникают вычислительные аппроксимации онтологии, определяющие возможности использования онтологии в данной вычислительной среде, но должен быть определен и идеальный объект, приближением которого являются эти аппроксимации.

Аппроксимация зависит от потребностей пользователя, от представления онтологии в RDF-хранилище, программной среды, в которой создано хранилище, от представленных правил и, наконец, от уровня знаний логических выводов в данной онтологии. Правила выражают ограничения (аксиомы) данной онтологии, но, кроме того, могут также представлять достигнутый уровень процедурных знаний в этой онтологии. Совокупность всех аппроксимаций онтологии представляет собой направленное множество вместе с отношением, отражающим, что одна аппроксимация содержит другую (является более точной и отвечает на большее множество запросов). В пределе аппроксимации онтологии приближаются к идеальной (в общем случае недостижимой и потенциально бесконечной) *математической модели* онтологии.

В среде конкретных онтологий пользователями могут быть придуманы особо эффективные алгоритмы для ответов на определенные запросы[4], и эти процедурные знания должны быть как-то отражены в вычислительных моделях. Следовательно, для эффективной работы с онтологиями мы должны иметь как средства для представления непроцедурных знаний о мире онтологии, так и средства представления процедурных знаний для эффективных вычислений в мире онтологий. Процедурные знания могут выражаться, например, в виде последовательностей запросов или правил переписывания.

Заметим, что языки запросов в онтологиях являются процедурными языками (в отличие от языка формул или вопросов). Пользуясь знаниями об онтологии, пользователи выражают информационную потребность, содержащуюся в вопросе, на языке запросов, указывая последовательность действий в данной онтологии и последовательность выполнения подзапросов для получения ответа на вопрос.

[3] В системах, использующих общие онтологии, данные онтологий дополняются более подробной информацией из приложения.

[4] Бывает и наоборот: для особых алгоритмов описывается онтология, в которой этот алгоритм работает.

2.3. Уровень описания математической модели онтологии

Среди уровней представления онтологий дополнительно следует выделить математический (логический) уровень для описания идеальной математической модели онтологии. На этом уровне моделируются потенциальные возможности средств онтологического моделирования, несколько абстрагируясь от языков представления онтологий и удобств пользователей. По существу, на этом уровне представляется семантика возможностей онтологического моделирования. На этом же уровне моделируются операции взаимодействия онтологий. К математическому уровню относятся логические модели онтологий в виде дескриптивных логик, представление онтологий в исчислениях предикатов первого порядка и более высоких порядков. С математической точки зрения описание онтологии (ее декларативной части) есть последовательное определение сигнатуры логической теории и ее аксиом (т. е. описание теории). Имея начальные понятия класса, элементов классов, отношения класс – подкласс, операций и предикатов и связывающие их аксиомы, последовательно вводят новые классы, элементы, предикаты и операции, а также связывающие их аксиомы, отражающие предметную область онтологии и системные конструкции, принятые или вводимые в данной предметной области.

Наравне с логическими моделями онтологий на математическом уровне используются алгебраические модели, подходы и языки (алгебра и логика – взаимно дополняющие друг друга разделы математики). Эти подходы, может быть, не столь известны программистам, использующим онтологии. Они требуют более серьезной математической подготовки, но в них накоплен опыт и достигнуты наиболее существенные результаты для построения идеальных математических моделей онтологий, приближениями которых являются вычислительные аппроксимации онтологий. В частности, здесь продуманы идеи модульной организации онтологий, параметрические онтологии, методы коллективной разработки онтологий [12].

Один из авторов этой работы давно пропагандирует алгебраические методы в теории баз данных и моделировании средств представления знаний. Уже более 10 лет этот курс читается в РГГУ для студентов специальности «Интеллектуальные системы в гуманитарной сфере», написан учебник этого курса [23]. В настоящей работе описывается алгебраический подход к моделированию онтологий.

Алгебраический подход к моделированию онтологий был положен в основу алгебраического языка спецификаций онтологий CASL [11]. Этот язык был разработан большой группой специалистов, применяющих алгебраические методы для моделирования, накопивших большой опыт и объединившихся в группу Common Framework Initiative (CoFI) [12] для построения стандарта алгебраического языка спецификаций.

Следующий раздел статьи посвящен языку CASL и подготовлен на основе доклада А. И. Ильиной, студентки Отделения интеллектуальных систем в гуманитарной сфере РГГУ. Авторы выражают ей благодарность за квалифицированную работу.

3. ЯЗЫК CASL

Язык алгебраических спецификаций CASL (Common Algebraic Specification Language) был разработан сообществом CoFI (Common Framework Initiative for algebraic specification and development) в 1998 г., но работа над ним началась гораздо раньше и CASL вобрал в себя достижения многих других алгебраических языков спецификаций.

В основе языка CASL лежит стандартизованный язык предикатов первого порядка. В настоящее время существуют различные расширения и подъязыки CASL, включая диалекты языка логики высшего порядка, модальной логики, OWL DL. В языке CASL имеются средства структурирования спецификаций, включающие использование модулей, инкапсуляции, объединение спецификаций, расширения и т. д.

Алгебраическим язык CASL называется потому, что моделями его спецификаций являются алгебраические системы. Этот язык включает в себя тщательно отобранные идеи предыдущих языков алгебраических спецификаций и новые оригинальные идеи, чтобы можно было писать спецификации более четко, ясно и в то же время кратко. CASL сразу же привлек к себе всеобщее внимание и уже de facto рассматривается как стандарт. Стоит отметить, что CASL является языком представления, а не языком программирования.

При создании языка спецификаций CoFI руководствовалось следующими принципами:

1) все типы данных могут быть представлены алгебрами, т.е. несущим множеством, операторами на этом множестве и аксиомами;

2) абстрактные типы данных могут быть представлены классами таких алгебр;

3) спецификации программ можно давать с помощью вводимых абстрактных типов данных и описания последовательности выполнения операций.

Исходя из этих соображений, было решено, что абстрактные типы данных и программы могут быть представлены алгебраическими методами. Таким образом появился CASL – язык для спецификаций требований к программным продуктам. Позже он стал рассматриваться как язык описания онтологий, использующий модульную организацию, и стал применяться для описания сложных онтологий (см., например, [24] о применении CASL в разработке онтологии верхнего уровня[5] DOLCE).

Язык CASL состоит из следующих частей:

- **Базовые спецификации:** декларации переменных, определения, аксиомы. В спецификации могут быть определены сорта (классы), подсорта (подклассы), операции (частичные операции) и предикаты. Аксиомы базовых спецификаций являются аксиомами логики первого порядка.

- **Структурные спецификации:** сложные спецификации могут быть легко построены из более мелких с помощью нескольких операций, опреде-

[5] Онтологиями верхнего уровня принято называть такие онтологии, которые создаются для описания абстрактных понятий и используются в основном для согласования онтологий, описывающих конкретные предметные области. Более подробно см. в [4] раздел 2.2 и описание на [25].

ленных в языке CASL: объединения, расширения, переименования и др.

• **Архитектурные спецификации** служат для структурирования сложной модульной онтологии вплоть до реализации онтологии готового программного продукта.

• **Библиотеки спецификаций:** именованные коллекции именованных спецификаций.

3.1. Базовые спецификации

Семантика базовых спецификаций, как правило, состоит из двух частей:

– сигнатуры Σ, соответствующей символам, введенным спецификацией (словарь онтологии);

– класса моделей сигнатуры Σ (class of Σ-models), соответствующих тем интерпретациям сигнатуры Σ, которые удовлетворяют аксиомам и ограничениям спецификации.

Спецификации CASL могут задавать:

- сорта (классы),
- подсорта (подклассы),
- операции,
- предикаты.

Сорта (sorts). Сорта представляют собой множество, которое называется несущим множеством (carrier set). Элементы несущего множества обычно являются абстрактными представлениями какого-либо типа данных: чисел, букв, списков и т. д. Сорт в языке CASL является аналогом типа в языках программирования. В CASL также можно задавать составные сорта (например, List[int] – список из целых чисел).

Подсорта (subsorts). Подсорта в CASL интерпретируются как вложенные (встроенные) типы. Подсорта могут быть и подмножествами сортов (*Nat*<*Int*), и вложенными типами (*Char*<*String*).

Операции (operations). Операции в CASL могут быть определены как полные или частичные. Операции состоят из имени и «совокупности параметров», в которой указываются количество и сорта аргументов, а также сорт результирующего значения операции. Если значение какого-либо параметра операции не определено, то результат этой операции также будет не определен. Операции без аргументов называются константами. Они интерпретируются как элементы несущего множества результирующего сорта. Раздел спецификации CASL, в котором вводятся операции, обозначается словом *ops*.

Приведем пример:

```
ops min(x,y : Elem) : Elem = x when x <
y else y;
max(x,y : Elem) : Elem = y when min(x,y)
= x else x
```

Предикаты (predicates). Предикаты определяются так же, как и операции, только они не имеют сорта результирующего значения. Следует обратить внимание, что не стоит путать предикаты с операциями с результирующим сортом булевского типа. Предикаты, в отличие от операций, не могут иметь неопределенного значения (если аргумент предиката не определен, то предикат считается не выполненным). Предикаты используются для формирования атомарных формул, а не термов.

Приведем пример:

```
sort Elem
pred __<__ : Elem x Elem
pred  __? __(x,y : Elem) <=> (x < y ? x
= y)
```

Имя предиката или операции может быть объявлено с различными наборами типов параметров в одной и той же спецификации. Такой случай называется перегрузкой (overloading). Например: перегрузка константной операции empty: для сорта list и для сорта set. Перегрузка предиката <: для *Int* и для *Char*.

Интерпретация логических связок и кванторов в языке CASL является стандартной. Аксиомы в CASL – это аксиомы логики первого порядка. Все логические связки и кванторы (с добавлением квантора) также берутся из языка предикатов. Переменные в формулах могут иметь значения из несущего множества каждого соответствующего сорта.

Приведем пример спецификации «частичный порядок»:

```
spec Strict_Partial_Order =  %задали
имя%
sort Elem  %задали сорт%
pred__<__: Elem x Elem %задали предикат
«меньше»%
? x , y, z : Elem
. not (x < x )  %некоммутативность%
. x < y => not (y < x )  %асимметрич-
ность%
. x < y and y < z => x < z %транзитив-
ность%
%{стоит отметить, что могут существовать
такие x и y,
что не верно ни x < y, ни y < x.}%
end
```

3.2. Структурные спецификации

Семантика структурных спецификаций та же, что и у базовых спецификаций. Структурные спецификации формируются из базовых с использованием разнообразных конструкций составления спецификаций.

Рассмотрим следующие основные способы построения структурных спецификаций:

- переименование,
- скрытие,
- объединение и расширение.

Переименование. Переименование используется для составления новых спецификаций из уже имеющихся. Это помогает избежать коллизии имен операций и предикатов и подчеркивает отличие смысла операций в новой спецификации. Для переименования в CASL есть оператор `|->`.

Приведем пример:

```
%На основе списка строится спецификация
стек.%
spec Stack[sort Elem] =
List_Selectors [sort Elem]
with
sort List |--> Stack,
ops cons |--> push__onto__, head |-->
top, tail |--> pop
end
```

Скрытие. Ненужные в новой спецификации операторы и предикаты можно скрыть. Для этого используется оператор hide.

Пример:

```
Spec Square_Matrix =
Matrix hide num_of_columns
end
```

Объединение и расширение. Можно конструировать сложные спецификации, объединяя уже существующие более мелкие спецификации. В CASL это делается с помощью оператора and. Логично это делать одновременно с расширением, т. е. новая спецификация расширяет возможности тех спецификаций, из которых она состоит. Для этого в CASL есть оператор then.

Приведем пример:

```
%{Построим спецификацию, объединяющую
спецификации
список и множество.}%
Spec List_Set [sort Elem] =
List_Selectors [sort Elem]
and Generated_Set [sort Elem]
then op elements__of__ : List -> Set
. e:Elem; L:List
. elements_of empty = empty
. elements_of cons(e, L) = { e } V
elements_of L
end
```

3.3. Архитектурные спецификации

Семантика архитектурных спецификаций отражает их модульную структуру, т. е. архитектурные спецификации описывают структуру системы в целом. Они состоят из блоков, которые определяют компоненты системы, и блока результата, который указывает, как компоненты должны быть скомбинированы.

Приведем пример:

```
%Простейшая архитектурная спецификация:%
arch spec System =
units
Unit1 : Specification1
…
UnitN : SpecificationN
result LINK(Uuit1, …, UnitN)
```

Здесь `Unit1...UnitN` – имена блоков, представляющих собой спецификации `Specification1...SpecificationN`; `LINK` – терм, определяющий, что будет результатом этой спецификации. Часто результатом бывает лишь один из блоков спецификации.

Обычно `Specification1...SpecificationN` имеют общие части и используют операции и предикаты друг друга. Для этого в CASL есть оператор given, который указывает, что данная спецификация может быть реализована только тогда, когда реализована какая-либо другая спецификация. С его помощью можно разбивать сложные спецификации на несколько более простых.

Приведем пример:

```
arch spec Matr =
units
V : Vector
M : Matrix given V
MM: MatrixMult given M
result MM
```

3.4. Библиотеки спецификаций

Семантика библиотек спецификаций – это отображение имен спецификаций в их семантику, т. е. библиотеки спецификаций представляют собой именованные коллекции спецификаций. Чаще всего библиотеки являются локальными (все используемые в библиотеке спецификации определены внутри этой библиотеки). Спецификации, которые были определены позднее, могут использовать внутри себя те, что были определены до них.

Приведем пример:

```
library UserManual/Examples
...
spec Natural = ...
...
spec Natural_Order = Natural then ...
```

Библиотеки также могут включать в себя спецификации из других библиотек. Для этого в CASL используется слово from.

Пример:

```
library Basic/StructuredDatatypes ...
from Basic/Numbers get Nat, Int ...
spec List [sort Elem ] given Nat = ...
...
spec Array ... given Int = ...
```

Для библиотеки можно определять инструкции синтаксического разбора, отображения символов, информацию об авторе и дате создания библиотеки.

Пример:

```
library UserManual/Examples  ...
%display union %LATEX V
%prec { + , ? } < { * }
%authors( Michel Bidoit <bidoit@lsv.ens-
cachan.fr>)%
%dates 15 Oct 2003, 1 Apr 2000  ...
```

Директива `%display union %LATEX V` указывает на то, что оператор под именем union следует отображать как V; `%prec { + , ? } < { * }` определяет приоритет операций и таким образом позволяет опускать скобки и вместо $a-(b*c)+d$ просто писать $a-b*c+d$.

В настоящее время CASL имеет много диалектов (см. [11, раздел 1.7.2]), в основе которых лежат язык предикатов первого порядка, второго порядка, высших порядков, модальная логика, теория категорий. Предусматриваются подходы слияния языка CASL и наиболее современного языка функционального программирования Haskell [26], в основе которого лежат теоретико-категорные конструкции λ-исчисления.

4. ОБЩЕЕ ОПИСАНИЕ ПРОЕКТА ПОСТРОЕНИЯ ВЕБ-СЕРВЕРА ОНТОЛОГИЙ

В этом разделе описывается проект [27], разрабатываемый авторами данной статьи. В проекте предлагается построить Веб-сервер в стиле Википедия с использованием технологий Web 2.0 для библиотек онтологий.

Проектируемая система предназначена для коллективного формирования библиотек взаимодействующих онтологий и языков работы с ними. В соответствии с принципами Веб 2.0 система обеспечивает удобную среду в Интернет для объединения пользователей системы по интересам, распределения ролей между поль-

зователями и формирования библиотек онтологий самими пользователями на формальном языке, который они формируют в процессе построения библиотеки.

Онтология, как упоминалось ранее, – это формальное описание договоренности группы людей о том, что и как у них называется, какими свойствами может обладать, в каких отношениях участвует и каким ограничениям удовлетворяет. Онтология группы может уточняться и, как правило, уточняется в процессе объединения людей. Более того, по мере определения онтологии предметной области социальной группы в этой группе формируется свой язык, на котором выражаются данная онтология и способы ее использования. Степень формальности такого языка зависит от традиций и особенностей предметной области, лежащей в основе интересов данной группы, а также от желания и возможности использования в данной области компьютерных моделей.

Первоначально пользователям системы предоставляется некоторая базовая онтология ядра системы и язык ядра, обеспечивающие возможность формировать произвольные онтологии, вводить новые конструкции языка и переопределять старые конструкции – подстраивать язык под конкретную проблемную область.

Далее пользователи могут объявить любую построенную ими онтологию средой для построения новых онтологий. В этом случае онтология среды и конструкции языка, доступные из среды, становятся доступными для новых конструируемых онтологий.

В ядре системы имеются возможности для модульного объектного построения новых онтологий. В результате деятельности пользователей строятся библиотеки взаимосвязанных онтологий и словарей языков системы. В системе имеются средства, поддерживающие процессы обсуждения вариантов онтологий, построения и хранения черновиков онтологий, версий онтологий, тестирования и отладки онтологий, а также средства управления публикацией онтологий (доступностью онтологий для различных групп пользователей).

В системе вместе с текстом онтологии на построенном формальном языке хранится некоторая вычислительная модель онтологии, автоматически построенная системой на этапе проверки правильности построения текста онтологии (откомпилированный файл онтологии). На основании этой вычислительной модели в системе производится анализ правильности построения запросов к онтологии и построение ответов на запросы. В ядре онтологии имеются некоторые языковые конструкции, управляющие вычислительной моделью онтологии, позволяющие задавать вычисления по правилам переписывания выражений в данной онтологии.

Работа системы обеспечивается взаимодействием двух подсистем:

- Веб-сервера онтологий,
- Веб-приложения.

Веб-сервер онтологий обеспечивает многопользовательский режим работы системы, работу с пользователями, библиотеками онтологий и словарями. Для работы с конкретной онтологией или черновиком онтологии Веб-сервер вызывает Веб-приложение в соответствующем режиме и передает ему требуемые для работы в этом режиме параметры.

Веб-приложение выполняет все функции с отдельной онтологией или черновиком: анализирует запрос к онтологии, строит и выводит ответ на запрос; анализирует текст черновика онтологии; строит и выдает сообщения об ошибках; строит вычислительную модель онтологии по тексту онтологии; результаты работы с онтологией отображает в формах пользователя и передает Веб-серверу для сохранения.

В настоящее время Веб-сервер онтологий разрабатывается с помощью программных средств технологии Drupal [28] для создания интернет-порталов. Веб-приложение разрабатывается на языке Пролог с помощью средств Visual Prolog 5.2 [29] и SWI-Prolog [30]. В основу Веб-приложения положены программы, разработанные на Visual Prolog 5.2 для системы представления знаний ЭЗОП, работающей под Windows [31].

Основное отличие проектируемой системы от существующих серверов онтологий (например, Ontolingua [8]) состоит в том, что в проекте предусматривается использование принципов открытого шаблонного языка для представления онтологий, позволяющего пользователям подстраивать язык представлений по мере пополнения библиотеки онтологиями и шаблонами пользователей к языку предметной области.

Для обмена проектируемой системы онтологиями с другими системами предполагается разработка модулей по загрузке и выводу онтологий на языках OWL и CASL.

Кроме того, существенной особенностью разрабатываемой системы является использование вместе с текстами онтологий откомпилированных по этим текстам внутренних представлений онтологий (вычислительных аппроксимаций онтологий). Внутреннее представление онтологии используется при семантическом анализе выражений языка, при формировании ответов на запросы к онтологии и ее отладке, при межмашинном обмене онтологиями в некотором стандарте и при использовании онтологий в приложениях. Внутренние представления онтологий предполагается строить с использованием современных алгебраических средств и модульной организации построения онтологий на основании подхода, близкого к подходу языка спецификаций CASL. Эти средства обладают достаточными возможностями для моделирования сложных онтологий и проведения вычислений на основе построенных моделей.

В соответствии с принципами Веб 2.0 авторы проекта не предполагают сами разрабатывать онтологии, а собираются разработать программную среду в Веб и ядро системы, в которой пользователям было бы удобно коллективно разрабатывать и отлаживать библиотеки онтологий и сопутствующие им языки. Однако некоторые примеры библиотек онтологий авторы предполагают разрабатывать совместно с некоторыми группами пользователей на этапе отладки системы и создания демонстрационных примеров.

Процесс формирования сложных и больших библиотек онтологий в настоящее время сдерживается сложностью формального языка представления онтологий. Сложность языка (несоответствие его языку

прикладной области) ограничивает понимание этих текстов специалистами в прикладных областях и, следовательно, затрудняет процессы проверки, формирования и тестирования онтологий. На преодоление этих недостатков нацелен предлагаемый проект. В качестве основы для решения перечисленных проблем предлагаются разработанные членами коллектива принципы открытого языка представления знаний. Эти принципы обеспечат многослойность представления онтологий:

- на пользовательском уровне онтологии представляются на языке, близком к языку предметной области;
- на уровне межмашинного обмена знаниями онтологии представляются на стандартном языке машинного представления онтологий;
- на внутреннем уровне для задач тестирования онтологий и построения ответов на вопросы к миру построенной онтологии используется алгебраический язык и средства алгебраических вычислений.

В настоящее время в соответствии с проектом создается экспериментальный сервер онтологий с использованием технологий Веб 2.0. На сервере можно будет зарегистрироваться. Пользователям могут быть присвоены различные роли. Система будет работать в многопользовательском режиме. Пользователи смогут искать документацию на сервере, просматривать ее, оставлять комментарии к документам, открывать новые форумы для обсуждений и участвовать в старых, а также открывать блоги для обсуждения работ над онтологиями.

Для формирования онтологии в открытом языке шаблонных выражений, ее отладки и построения ответов на вопросы к онтологии разработана специальная программа (Веб-приложение). Программа работает на сервере с интерфейсом в Веб-браузере и вызывается пользователем с сервера следующими опциями:

«Задать вопрос» – для построения запроса на доступном из текущей онтологии языке, сформированном пользователями, и для получения ответов на запросы к онтологии. Ответы формируются на основании внутреннего представления онтологии, правил переписывания, введенных в онтологии, и алгебраических вычислений.

«Сделать средой» и «Создать новую онтологию» – для создания новых онтологий с возможностью сохранения недоделанных онтологий в виде черновиков.

«Создать новую версию онтологии» – для создания и отладки новых версий существующих онтологий.

«Удалить онтологию», если она не используется другими онтологиями.

Взаимодействие сервера онтологий и программы работы с текущей онтологией находится в стадии отладки.

Сервер онтологий разрабатывается программными средствами системы управления сайтом Drupal 5 [28]. Система Drupal имеет большое количество модулей и удобную организацию для быстрой разработки серверов, использующих технологии Веб 2.0. В дальнейшем предполагается переход на более свежую версию CMS Drupal 6.

Программа грамматического анализа для открытого языка шаблонных выражений, на котором описываются онтологии и задаются вопросы к онтологии, написана на языке Пролог, программа вычисления ответов на запросы к онтологии также разрабатывается на языке Пролог. Основу этих программ составляет система ЭЗОП под Windows (http://ezop-project.ru/?q=WindowsEzop) [9, 31, 32], разработанная программными средствами Visual Prolog 5.2.

Документация к проекту выставлена на специально разработанном средствами CMS Drupal 6 сайте [27]. На этом же сайте приведен список примеров онтологий, разработанных в системе [33].

Основной онтологией системы является онтология с именем «Ядро системы». Эта онтология создана разработчиками системы. В онтологии ядра определены базовые типы, используемые при конструировании онтологий, а также шаблоны базового языка, предоставляемого системой пользователям. Кроме того, в языке ядра определены шаблоны для задания вопросов к онтологии. Наиболее интересные из них:

```
Элементы области @Области?
Подобласти области @Область?
Чему равно @выражение?
```

Здесь выражения, начинающиеся с символа `@`, представляют собой переменные, вместо которых могут подставляться выражения типа, заданного в шаблоне вопроса. Например, в шаблоне `Элементы области @Области?` переменная `@Области` имеет тип «область», который в системе является синонимом типа «класс». Результатом выполнения данного шаблона будет печать списка элементов конкретного класса, имя которого передано в качестве параметра в шаблон. Например, на вопрос `Элементы области материки?` в онтологии, описывающей географические объекты, будет дан ответ: `Евразия, Северная Америка, Южная Америка, Африка, Австралия, Антарктида`.

В [33] представлены примеры иерархических онтологий (таксономий), спецификации булевой алгебры, онтологий, описывающих физические понятия (в частности, понятие равномерного движения), а также приведен пример спецификации вычислимой функции (онтология «Факториал»). Рассмотрим последнюю онтологию подробнее.

Онтология «Факториал» содержит грамматический шаблон, позволяющий задавать в текстах онтологий факториалы различных значений. Значение этого шаблона имеет тип real, а семантика состоит в вычислении значения факториала непосредственно в тексте онтологии. Ниже приведен текст онтологии «Факториал»:

```
Введем шаблон "fact(@n,@i,@y)"
с переменными:
"n,i,y: real_выражение"
и переменной результата "z:
real_выражение ";
Пояснения: [Вычисляет факториал]
Условие применения шаблона:
[]
Действие шаблона:
[
if n == i then z = y else z =
fact(n,i+1,(y)*(i+1))
]
Тип доступа шаблона:[локальный].
```

```
Введем шаблон "@n!"
с переменными:
"n: real"
и переменной результата "z: real " ;
Пояснения: [Вычисляет факториал]
Условие применения шаблона:
[]
Действие шаблона:
[
z=fact(n,1,1)
]
Тип доступа шаблона:[внешний].
```

В этой онтологии определяются два шаблона: `fact(@n,@i,@y)` и `@n!`. Первый шаблон вспомогательный и имеет внутреннюю область видимости. Второй шаблон `@n!` состоит, фактически, из параметра `@n`, который используется для передачи натурального числа с целью последующего вычисления значения факториала. В текстах онтологий выражение вида `5!` будет теперь трактоваться как факториал числа 5 и вычисляться соответствующим образом.

5. ЗАКЛЮЧЕНИЕ

В настоящей статье описан проект построения Веб-сервера онтологий в стиле Веб 2.0. Целью проекта является разработка системы, позволяющей пользователям коллективно создавать онтологии. Разработка такой системы требует решения целого ряда вопросов. Возникающая проблематика также описана в данной статье.

Введено понятие об уровнях представления онтологий. Выделено три уровня представления: верхний, средний и нижний. Верхний уровень доступен пользователям системы, создающим онтологии. На среднем уровне специалисты по разработке систем знаний пишут программы для обмена данными между различными системами. На нижнем уровне разрабатываются модели эффективного доступа к данным.

Кроме того, выделяется уровень математической модели представления онтологии. Приведено описание одного из вариантов уровня математической модели представления онтологии, основанного на алгебраическом подходе к моделированию онтологий. Онтологии, описанные с помощью этого подхода, обычно называют алгебраическими спецификациями. В статье приведен обзор языка алгебраических спецификаций CASL, являющегося сегодня de facto стандартом для такого рода языков.

Вопросы организации вычислений в онтологиях, построенных на основе алгебраического подхода, не были рассмотрены подробно, поскольку эта проблематика требует отдельной статьи. Однако следует сказать, что проект построения Веб-сервера онтологий в стиле Веб 2.0 позволяет производить такие вычисления (см. приведенные выше примеры вопросов к онтологии «Ядро системы»).

СПИСОК ЛИТЕРАТУРЫ

1. Веб 2.0 // Википедия [сайт]. – URL: http://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1_2.0
2. Gruber T. R. Toward Principles for the Design of Ontologies used for Knowledge Sharing // Technical report KSL-93-21 / Knowledge Systems Laboratory, Stanford University. – 1993. – August 23.
3. Guarinio N. Formal Ontology and Information Systems // Proceedings of FOIS'98 (Trento, Italy, 6 – 8 June 1998). – Amsterdam : IOS Press, 1998.
4. Лапшин В. А. Онтологии в компьютерных системах. – М. : Научный мир, 2010.
5. Фолксономия // Википедия [сайт]. – URL: http://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D0%BB%D0%BA%D1%81%D0%BE%D0%BD%D0%BE%D0%BC%D0%B8%D1%8F
6. Semantic Web // Википедия [сайт]. – URL: http://en.wikipedia.org/wiki/Semantic_Web
7. Semantic Wiki // Википедия [сайт]. – URL: http://en.wikipedia.org/wiki/Semantic_wiki
8. Проект Ontolingua [сайт]. – URL: http://www.ksl.stanford.edu/software/ontolingua
9. Бениаминов Е. М., Болдина Д. М. Система представления знаний Ontolingua – принципы и перспективы // НТИ. Сер. 2. – 1999. – № 10. – С. 26 – 32.
10. Свободная энциклопедия Википедия [сайт]. – URL: http://ru.wikipedia.org
11. Bidoit M., Mosses P. D. CASL User Manual. Introduction to Using the Common Algebraic Specification Language. – New York : Springer, 1998.
12. Проект CoFI [сайт]. – URL: http://www.informatik.unibremen.de/cofi/wiki/index.php/CoFI
13. Проект Protege [сайт]. – URL: http://protege.stanford.edu
14. Проект KAON // Википедия [сайт]. – URL: http://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1_2.0
15. Обзор языка OWL // The World Wide Web Consortium (W3C) [сайт]. – URL: http://www.w3.org/TR/owl-features
16. Описание языка SPARQL // The World Wide Web Consortium (W3C) [сайт]. – URL: http://www.w3.org/TR/rdf-sparql-query
17. Обзор языка SWRL // The World Wide Web Consortium (W3C) [сайт]. – URL: http://www.w3.org/Submission/SWRL
18. Обзор языка RIF // The World Wide Web Consortium (W3C) [сайт]. – URL: http://www.w3.org/TR/rif-overview
19. Язык RDF // The World Wide Web Consortium (W3C) [сайт]. – URL: http://www.w3.org/RDF
20. Проект JENA // SourgeForge [сайт]. – URL: http://jena.sourceforge.net/documentation.html
21. Проект Open RDF [сайт]. – URL: http://www.openrdf.org
22. Database Semantic Technologies // ORACLE [сайт]. – URL: http://www.oracle.com/technology/tech/semantic_technologies/htdocs/semtech_training.html
23. Бениаминов Е. М. Алгебраические методы в теории баз данных и представлении знаний. – М. : Научный мир, 2003.
24. Проект DOLCE // Laboratory for Applied Ontology [сайт]. – URL: http://www.loacnr.it/DOLCE.html

25. Upper ontology (information science) // Википедия [сайт]. – URL: http://en.wikipedia.org/wiki/Upper_ontology_(computer_science)
26. Язык Haskell [сайт]. – URL: http://www.haskell.org
27. Проект экспериментального Веб-сервера онтологий ЭЗОП [сайт]. – URL: http://ezop-project.ru
28. Проект Drupal: open source content management platform [сайт]. – URL: http://drupal.org
29. Адаменко А., Кучуков А. Логическое программирование и Visual Prolog. – СПб. : БХВ-Петербург, 2003.
30. Язык SWI-Prolog [сайт]. – URL: http://www.swi-prolog.org
31. Бениаминов Е. М., Болдина Д. М. Система представления и обработки знаний ЭЗОП // Материалы конференции «Диалог'2001». Прикладные проблемы. – М., 2001.
32. Бениаминов Е. М., Манушина М. Ю. Принципы построения открытого языка шаблонных выражений в системе представления знаний // НТИ. Сер. 2. – 2000. – № 7. – С. 10 – 17.
33. Примеры онтологий // Проект ЭЗОП [сайт]. – URL: http://ezop-project.ru/?q=node/166

Материал поступил в редакцию 28.09.11.

Сведения об авторах

БЕНИАМИНОВ Евгений Михайлович – доктор физико-математических наук, профессор, заведующий кафедрой математики, логики и интеллектуальных систем в гуманитарной сфере Института лингвистики Российского государственного гуманитарного университета, Москва
E-mail: ebeniamin@yandex.ru

ЛАПШИН Владимир Анатольевич – кандидат физико-математических наук, руководитель отдела обработки запросов на естественном языке компании ABBYY, Москва
E-mail: mefrill@yandex.ru

ИНФОРМАЦИОННЫЙ АНАЛИЗ

УДК 004.93

В. Г. Мокрозуб, К. В. Немтинов, К. А. Шаронин

Программное обеспечение автоматизированных систем размещения объектов в пространстве, инвариантное к предметной области*

Предложены структура и программная реализация инвариантной к предметной области базы ограничений информационных систем, предназначенных для размещения объектов в пространстве. В качестве программно-независимого способа представления объектов выбран N-ориентированный гиперграф. Ограничения представляются правилами вида «если A, то B». Базовое программное обеспечение – система управления реляционными базами данных. Правила представлены теоретико-множественным описанием и описанием на языке структурированных запросов Transact SQL.

Ключевые слова: *база ограничений, размещение объектов, правила, запросы, SQL, гиперграф*

ВВЕДЕНИЕ

Решение разнообразных задач размещения (компоновки) объектов в пространстве часто становится актуальным в различных областях человеческой деятельности. К таким задачам относятся, например, планировка дачного участка (где и какие деревья и кустарники посадить), создание проекта городского строительства (где и какие здания строить), создание проекта цеха химического или машиностроительного предприятия (где и какие аппараты или станки надо разместить). Общим в перечисленных задачах является то, что в них имеется набор сущностей (объектов), которые надо разместить (деревья, здания, аппараты, станки), и набор сущностей (объектов), в которых размещаются первые (территория дачного участка или города, помещения химического или машиностроительного предприятия). Размещаемые объекты и объекты, в которых происходит размещение, обладают определенными характеристиками или свойствами (размеры, назначение, категория). Объект считается размещенным, если однозначно определено его положение в пространстве. Это могут быть координаты некоторой характерной точки объекта, например, координаты центра ствола дерева на земле или координаты диагональных углов длинного объекта.

На возможные координаты размещаемых объектов накладываются ограничения, например: минимальное расстояние от ствола высокорослого дерева до границы соседнего участка на даче равно 4 м; расстояние между домами при наличии окон в противоположно расположенных зданиях не может быть меньше 15 м; тяжелое оборудование надо размещать на нижних этажах и др. Эти ограничения содержатся в нормативных документах, которые определены в каждой предметной области: сборниках нормативов и правил (СНиП), правилах проектирования безопасных технических объектов (ПБ) и др.

При наличии опыта человек (проектировщик), занимающийся размещением, может накладывать дополнительные ограничения, которых нет в нормативных документах. Кроме того, существует по крайней мере одно ограничение общего характера – непересечение размещаемых объектов друг с другом.

Если имеется много вариантов размещения, то выбор наилучшего осуществляется с использованием некоторого критерия предпочтения (критерия оптимизации), например: максимум освещенной солнцем поверхности под огород, минимум стоимости монтажа оборудования, минимум стоимости соединительных трубопроводов и др.

В работах [1, 2] представлены аналитические и процедурные модели, позволяющие находить оптимальные компоновочные решения оборудования химических производств. В качестве критерия оптимизации предлагается использовать критерий приведенных затрат, включающий в себя капитальные и эксплуатационные затраты, зависящие от принимаемых компоновочных решений (затраты на строительные конструкции, монтаж оборудования, трубопроводную арматуру, электроэнергию, расходуемую на транспортировку веществ и др.).

* Работа выполнена в рамках государственного контракта № 02.740.11.0624 Федеральной целевой программы «Научные и научно-педагогические кадры инновационной России на 2009 – 2013 годы».

В настоящее время существует ряд программных средств, предназначенных для размещения объектов:

- программы ландшафтного дизайна: Наш Сад 9.0 Рубин (Dicomp), Sierra Land Designer 3D 7.0 (ActiVision), Home and Landscape Design (PUNCH!);
- программы автоматизации инженерного проектирования объектов промышленности: PDS (Intergraph), PDMS (AVEVA), CADPIPE (Orange Technologies), CADWORX (COADE), AutoPLANT (Bentley), PLANT-4D (CEA Technology), ABS (Autodesk);
- универсальные программы, например, SketchUp (Google).

Эти программы обладают мощным графическим интерфейсом, множеством инструментов для ручного проектирования, базами данных элементов (деревья, строительные конструкции, арматура). Однако в них отсутствуют ограничения, которые определены нормативными документами, или возможность добавлять новые ограничения, что не позволяет решать оптимизационные задачи размещения в автоматическом режиме (или с минимальным участием человека).

Между тем принципиальная общность перечисленных задач размещения позволяет создать такую автоматизированную систему, которая может настраиваться пользователем на конкретную предметную область: ландшафтный дизайн, градостроительство или проектирование промышленных предприятий. Основным компонентом этой системы является база ограничений.

Цель настоящей работы – разработка инвариантных к предметной области: 1) представлений ограничений в задачах размещения на языке SQL, 2) структуры базы для хранения ограничений, 3) способов обработки базы данных с целью ответа на запросы о соблюдении ограничений предметной области при размещении объектов.

Область применения – разработка программного обеспечения автоматизированных систем, предназначенных для решения компоновочных задач промышленности, градостроительства, ландшафтного дизайна и др.

В качестве программно-независимого способа представления информационных объектов выбран N-ориентированный гиперграф. Ограничения представляются продукционными правилами вида «если A, то B». В качестве базового программного обеспечения использована реляционная база данных. Продукции (правила) представлены теоретико-множественным описанием и описанием на языке структурированных запросов Transact SQL (релиз SQL фирмы Microsoft).

1. ОПРЕДЕЛЕНИЕ N-ОРИЕНТИРОВАННОГО ГИПЕРГРАФА

Обозначим N-ориентированный гиперграф [3] $G(X,U)$, где $X = \{x_i\}$, $i = \overline{1,I}$, – множество вершин гиперграфа, x_i – i-я вершина, $U = \{u_m(X1_m)\}$, $m = \overline{1,M}$ – множество ребер гиперграфа, $u_m(X1_m)$ – m-тое ребро гиперграфа, $X1_m$ – множество вершин, инцидентных m-тому ребру, $X1_m \subseteq X$, $X1_m = \{x_k^{\overline{L}}\}$, $\forall k \in K_m$, $K_m \subseteq \overline{1,I}$, $\overline{L}$ – номер вершины в ребре ориентированного гиперграфа, представляет собой вектор, $\overline{L} = \{l_n\}$, $n = \overline{1,N}$, мощность которого N.

В общем случае номер вершины в ребре не обязательно должен иметь значение номер по порядку и может отражать определенное свойство вершины, которое принимает конкретное значение при включении вершины в ребро. При этом предполагается, что все вершины в ребрах имеют одинаковый набор свойств. Например, при решении задачи размещения элементов в пространстве, элементы имеют одинаковые наборы свойств, а именно координаты элементов.

Требование одинакового набора свойств элементов существенно сужает область прикладных задач. Каждый элемент реальной технической системы обладает собственным набором свойств, который отличается от наборов свойств других элементов, например, каждый размещаемый аппарат имеет собственный набор размеров (для горизонтального емкостного аппарата – диаметр и длина, для вертикального – диаметр и высота).

Обозначим $S = \{s_j\}$, $j = \overline{1,J}$ множество всех возможных свойств вершин и ребер, $SX_i = \{s_{r1}\} \subset S$, $i = \overline{1,I}$, $r1 \in J1_i \subset \overline{1,J}$ – множество свойств i-ой вершины, $SU_m = \{s_{r2}\} \subset S$, $m = \overline{1,M}$, $r2 \in J2_m \subset \overline{1,J}$ – множество свойств m-того ребра. Под свойством здесь понимается контейнер для хранения значения свойства. Например, свойство «вес аппарата» может иметь значение 25000 кг. Таким образом, для каждой вершины множество ее номеров $\overline{L} = \{l_n\}$, $n = \overline{1,N}$ заменяется множеством свойств SX_i, $i = \overline{1,I}$. Кроме того, каждому ребру u_m ($X1_m$) также ставится в соответствие свой набор свойств SU_m, $m = \overline{1,M}$.

Каждое свойство может иметь значения из определенного множества, например, масса аппарата – это числовые значения больше нуля, группа аппарата в зависимости от расчетного давления, температуры стенки и характера среды принимает значения 1, 2, 3, 4, 5а, 5б (ПБ 03-584-03. Правила проектирования, изготовления и приемки сосудов и аппаратов стальных сварных), а категория помещения по взрывопожарной и пожарной опасности – значения А, Б, В1, В2, В3, В4, Г, Д (НПБ 105-03. Нормы пожарной безопасности).

Обозначим $Z_j = \{z_{j,t}\}$, $t = \overline{1,T_j}$ множество возможных значений свойства s_j.

Обозначим $z[s_j,x_i] \in Z_j$ значение свойства s_j вершины x_i. Значение свойства s_j ребра u_m обозначим $z[s_j,u_m] \in Z_j$. Введем логический оператор Θ, один из элементов множества $\Omega = \{=,\neq,<,>,\leq,\geq\}$, $\Theta \in \Omega$. Запись $z[s_{j1},x_{i1}] = z_{j1,t1}$ обозначает, что значение свойства s_{j1} для вершины x_{i1} равно $z_{j1,t1}$ (или принимает значение $z_{j1,t1}$). Запись $z[s_{j1},x_{i1}]\Theta z_{j1,t1}$ означает, что значение свойства s_{j1} для вершины x_{i1} находится в определенном отношении со значением $z_{j1,t1}$, причем это отношение ограничено элементами множества $\Omega = \{=,\neq,<,>,\leq,\geq\}$.

2. СТРУКТУРА БАЗЫ ДАННЫХ ДЛЯ ПРЕДСТАВЛЕНИЯ N-ОРИЕНТИРОВАННОГО ГИПЕРГРАФА

Здесь и далее в качестве примера используется компоновка химического оборудования, но только потому, что эта предметная область более известна авторам, чем, например, градостроительство. Кро-

ме того, предполагается, что размещение аппаратов осуществляется в два этапа [1, 2]. На первом этапе определяется этаж или помещение, в котором будет размещен аппарат, на втором – координаты аппарата на этаже. Подобная двухуровневая декомпозиция исходной задачи не сужает область применения, так как она присутствует и в других предметных областях, например: разделение участка на сад и огород или группировка станков по отделениям (токарное, фрезерное). Теоретически можно предложить N-уровневую декомпозицию, однако для реальных практических задач двухуровневой декомпозиции вполне достаточно.

Структура базы данных для хранения описанного выше N-ориентированного гиперграфа представлена на рис. 1.

Имена таблиц соответствуют именам введенных ранее множеств X, U, G, S, SX, SU, Z. Текст в именах таблиц после подчеркивания следует рассматривать как комментарий. Например, действительное имя таблицы G_(Аппараты_в_помещениях) будет G. Поле Z_Значение_свойства в приведенных ниже примерах обозначено как Z.

Поле S_(Реестр_свойств). **Тип** позволяет различать два типа свойств:

- Тип 1. Свойства, которые могут иметь бесконечное множество значений. Они вводятся вручную (например, масса аппарата, объем аппарата, температура среды в аппарате). Таблица Z для них будет содержать ограничение на допустимое значение свойства (например, значение массы аппарата всегда больше нуля).
- Тип 2. Свойства, значения которых ограничены конечным списком. Например, для свойства «категория помещения» список возможных значений – А, Б, В1, В2, В3, В4, Г, Д. Этот список будет содержаться в таблице Z.

Таблица Z позволяет поддерживать доменную целостность базы. Ссылочная целостность поддерживается первичными ключами по полям X.ID_X, U.ID_U, S.ID_S, Z.ID_Z, SU.ID_SU, SX.ID_SX и внешними ключами, наименование которых на рис. 1 имеет префикс FK. Кроме того, для сохранения доменной целостности базы необходимо создать уникальные индексы по полям:

– SX.ID_X, SX.ID_S и SU.ID_U, SU.ID_S, конкретное свойство вершины или ребра присутствует в таблице SX или SU один раз;

– G.ID_U, конкретное ребро в графе может быть только один раз;

– G.ID_X, аппарат не может находиться одновременно в разных местах.

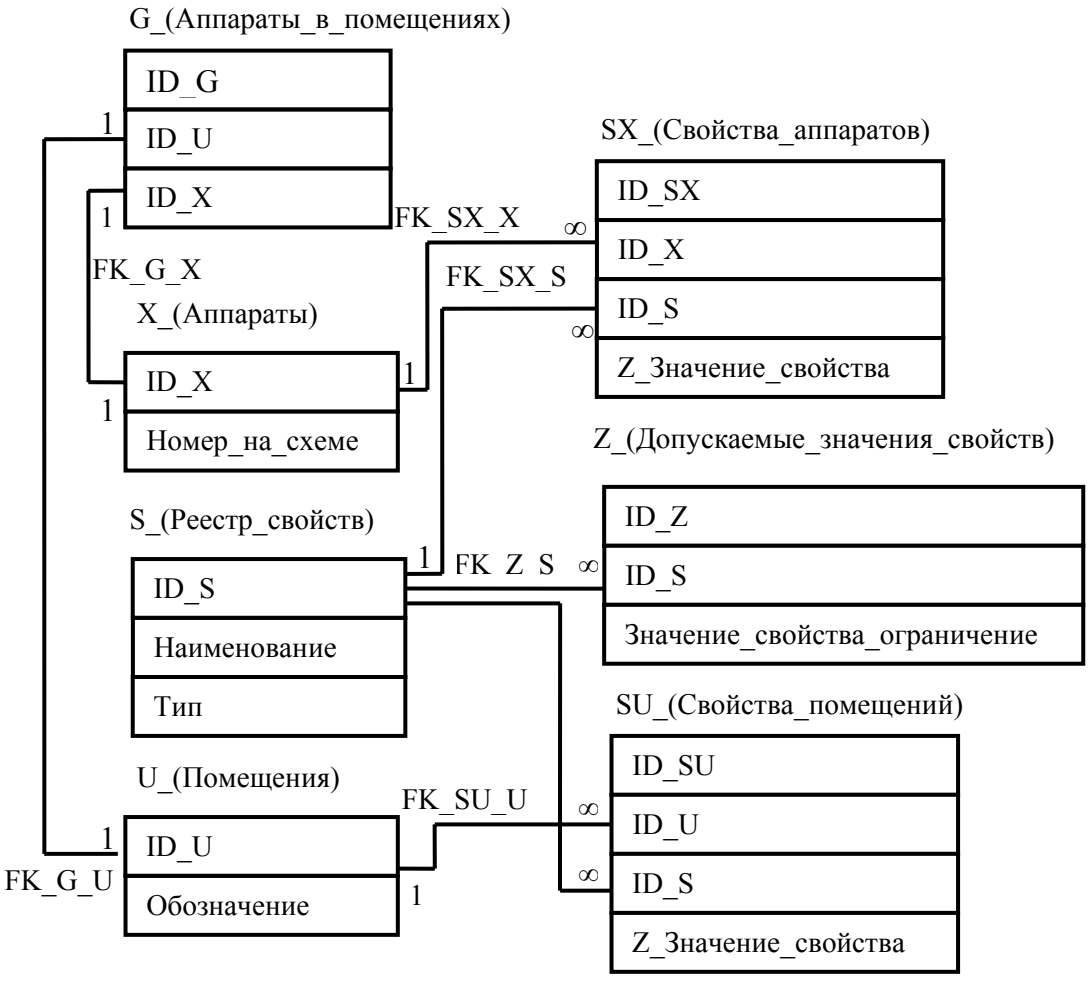

Рис. 1. Структурная схема базы данных для хранения N-ориентированного гиперграфа

Представленная на рис. 1 структура базы данных упрощена. Строго говоря, таблицы SX и SU следует дополнить полем ID_Z для хранения первичного ключа свойств типа 2 (поле Z.ID_Z) . Это усложнит приведенное ниже изложение, но никак не повлияет на суть предлагаемого подхода.

Свойства вершин и свойства ребер дополнительно можно разделить на следующие группы:

1) свойства вершины, которые не зависят от ее принадлежности ребру (например, масса аппарата, размеры);

2) свойства ребра, которые не зависят от вершин этого ребра (например, номер этажа, размеры помещения);

3) свойства ребра, зависящие от вершин этого ребра (например, категория помещения в зависимости от свойств аппаратов, находящихся в помещении);

4) свойства вершины, зависящие от ее принадлежности ребру (например, координаты аппарата в конкретном помещении);

5) свойства вершины, зависящие от собственных свойств или свойств других вершин (при размещении однотипных аппаратов в ряд: после установки первого аппарата определена одна из координат (x или y) всех остальных аппаратов ряда);

6) свойства ребра, зависящие от собственных свойств или свойств других ребер.

Свойства первой и второй группы являются исходными данными и вводятся до начала процесса размещения. Свойства третьей, четвертой, пятой и шестой группы определяются в процессе размещения.

Представленная классификация свойств необходима для дальнейшей программной реализации способов их определения. Предполагается, что свойства третьей, четвертой, пятой и шестой группы в начальный момент содержатся в таблицах SX и SU и имеют значения NULL, т. е. в процессе решения задачи эти свойства не добавляются, а редактируются, точнее, редактируется их значение (поле Z_Значение_свойства).

3. ФУНКЦИИ БАЗЫ ОГРАНИЧЕНИЙ

Под базой ограничений здесь понимается программно-информационный модуль, который в процессе решения задачи размещения должен отвечать на определенные запросы, поступающие из внешней программы, решающей задачу размещения. Ответ на запросы база ограничений осуществляет, используя хранящуюся в ней информацию. Структура этой информации инвариантна к предметной области. Смысловое содержание определяется предметной областью.

В качестве «внешней программы» может выступать и проектировщик, который расставляет объекты «вручную». База ограничений должна предостеречь его о возможных ошибках. Например, если проектировщик разместил тяжелый аппарат не на первом этаже, база ограничений должна сообщить ему, что здесь аппарат размещать нельзя, так как существует правило, согласно которому тяжелые аппараты следует размещать на первом этаже.

Поскольку осуществлена двухуровневая декомпозиция исходной задачи, возможные запросы управляющей программы и ответы базы ограничений также классифицируем по двум уровням. Подобная классификация необходима для представления возможных запросов в SQL-нотации.

На первом уровне возможны следующие запросы управляющей программы:

- Тип 1. Исходные данные – помещение и аппарат. Ответ – можно или нельзя размещать аппарат в помещении.
- Тип 2. Исходные данные – аппарат. Ответ – список помещений, куда ставить аппарат можно.
- Тип 3. Исходные данные – помещение. Ответ – список аппаратов, которые можно ставить в этом помещении.
- Тип 4. Исходные данные – аппарат. Ответ – список помещений, куда ставить аппарат нельзя.
- Тип 5. Исходные данные – помещение. Ответ – список аппаратов, которые нельзя ставить в этом помещении.

Ответы на перечисленные запросы выдаются на основании правил (ограничений), которые связывают свойства вершин и свойства ребер. Часть этих свойств не зависит от решаемой задачи (свойства группы 1 и 2), другие (группы 3 – 6) изменяются или определяются в процессе решения задачи. Таким образом, кроме ответа на перечисленные выше запросы база ограничений должна «уметь» изменять (или определять) свойства вершин и ребер в зависимости от своего текущего состояния, получаемого в процессе размещения. Текущее состояние определяется значениями полей G.ID_U, G.ID_X (принадлежность аппаратов помещениям) , SX.ID_S, SX.Z (свойства аппаратов), SU.ID_S, SU.Z (свойства помещений) на определенном итерационном этапе решения задачи размещения.

Ниже рассматриваются только правила первого уровня. Решение задачи второго уровня (размещение аппаратов на этаже) можно свести к задаче первого уровня, если все пространство этажа разделить на зоны (ребра) со своими свойствами. Например, для задачи размещения технологического оборудования такими зонами являются типовые строительные клетки размером 6x6 м (участок между колоннами).

4. ЭЛЕМЕНТАРНЫЕ ОГРАНИЧЕНИЯ И ИХ ПРЕДСТАВЛЕНИЕ В SQL-НОТАЦИИ

Введем понятия ограничения на значение свойства s_{j1} вершины x_{i1} (например, ограничение на массу аппарата) и ограничения на значение свойства s_{j2} ребра u_{m1} (например, ограничение на номер этажа помещения).

4.1. Ограничение на значение свойства вершины

$$z[s_{j1}, x_{i1}] \Theta z_{j1,t1},\ z_{j1,t1} \in Z_j,\ j1 \in J1 \subset J,\ i1 \in I,\ t1 \in \overline{1, T_{j1}} \qquad (1)$$

Запишем проверку ограничения (1) для заданной вершины в SQL-нотации. Будем считать, что заданы @ID_X (вершина), @ID_S (свойство вершины) и @z (значение свойства), которые соответствуют $x_{i1}, j1, z_{j1,t1}$ в выражении (1).

exists (select * from SX
where SX.ID_X=@ID_X and SX.ID_S=@ID_S and
SX.Z Θ @z1 (2)

Запишем выбор всех вершин, удовлетворяющих условию (1):

select * from X where exists (select * from SX where X.ID_X=SX.ID_X and SX.ID_S=@ID_S and SX.Z Θ @z) (3)

Введем понятие элементарного ограничения на свойства вершины Ox1(x), такое, что

Ox1(@ID_X) ≡ exists (select * from SX where SX.ID_X=@ID_X and SX.ID_S=@ID_S and SX.Z Θ @z),

Ox1(X.ID_X) ≡ exists (select * from SX where SX.ID_X=X.ID_X and SX.ID_S=@ID_S and SX.Z Θ @z).

С учетом введенного Ox1(x) выражение (2) запишется как Ox1(@ID_X), выражение (3) – как select * from X where Ox1(X.ID_X).

4.2. Ограничение на значение свойства ребра

$$z[s_{j2},u_{m1}]\Theta z_{j2,t2},\ z_{j2,t2}\in Z_{j2},\ j2\in J2\subset J,$$
$$m1\in M,\ t2\in\overline{1,T_{j2}} \quad (4)$$

Запишем проверку ограничения (4) для заданной вершины в нотации SQL. Заданы @ID_U, @ID_S, @z2, которые соответствуют $m1, j2, z_{j1,t1}$ в выражении (4).

exists (select * from SU where SU. ID_U=@ID_U and SU.ID_S=@ID_S and SU.Z Θ @z) (5)

Запишем выбор всех ребер, удовлетворяющих условию (4):

select * from U where exists (select * from SU where U.ID_U=SU.ID_U and SU.ID_S=@ID_S and SU.Z Θ @z) (6)

По аналогии с Ox1(x) введем обозначение Ou1(u), такое, что

Ou1(@ID_U) ≡ exists (select * from SU where U.ID_U=@ID_U and SU.ID_S=@ID_S and SU.Z Θ @z),

Ou1(U.ID_U) ≡ exists (select * from SU where SU.ID_U=U.ID_U and SU.ID_S=@ID_S and SU.Z Θ @z).

С учетом введенного Ou1(u) выражения (5) и (6) запишутся как Ou1(@ID_U) и select * from U where Ou1(U.ID_U).

4.3. Комбинации ограничений

Введем функцию $F1(z[s_{j1},x_{i1}]\ \Theta_{r1}\ z_{j1,t1})$, соединяющую логическим "И" несколько элементарных ограничений одной вершины:

$$F1(z[s_{j1},x_{i1}]\ \Theta_{r1}\ z_{j1,t1})=\bigcap_{r1=1}^{R1}(z[s_{j1},x_{i1}]\Theta_{r1}\ z_{j1,t1}),$$
$$j1\in J1\subset J,\ i1\in I,\ r1=\overline{1,R1},\ z_{j1,t1}\in Z_{j1},\ t1\in\overline{1,T_{j1}}, \quad (7)$$

где $J1$ – множество свойств соединяемых ограничений, $R1$ – количество соединяемых ограничений. Для упрощения в дальнейшем $F1(z[s_{j1},x_{i1}]\ \Theta_{r1}\ z_{j1,t1})$ будем обозначать $F1()$.

Рассмотрим пример комбинации двух ограничений вершины: «среда в аппарате взрывоопасна и токсична». Для удобного представления в базе данных запишем эти ограничения в виде «свойство вершины взрывоопасность среды=Да И свойство вершины токсичность среды=Да».

Пусть для свойства «взрывоопасность среды в аппарате» ID_S=10, для свойства «токсичность среды в аппарате» ID_S=11, тогда в выражении (7) $J1 = \{10,11\}$, $R1 = 2$, Θ_1 и Θ_2 соответствуют знаку =. Таким образом, для рассматриваемого примера

$$F1() = (z[s_{10},x_{i1}]="Да")\ И\ (z[s_{11},x_{i1}]="Да"),\ i1\in I \quad (8)$$

Запишем проверку условия (8) для аппарата @ID_X в SQL-нотации:

OOx(@ID_X) ≡ Ox1(@ID_X) and Ox2(@ID_X),

где OOx(@ID_X) – комбинация ограничений вершины, Ox1(@ID_X) ≡ exists (select * from SX where SX.ID_X=@ID_X and SX.ID_S=10 and SX.Z="Да"), Ox2(@ID_X) ≡ exists (select * from SX where SX.ID_X=@ID_X and SX.ID_S=11 and SX.Z="Да").

Поиск всех аппаратов, удовлетворяющих условию (8), запишется как

select * from X where OOx(X.ID_X),

где OOx(X.ID_X) ≡ Ox1(X.ID_X) and Ox2(X.ID_X), Ox1(X.ID_X) ≡ exists (select * from SX where SX.ID_X=X.ID_X and SX.ID_S=10 and SX.Z="Да"), Ox2(ID_X) ≡ exists (select * from SX where SX.ID_X=X.ID_X and SX.ID_S=11 and SX.Z="Да").

В общем случае $F1()$, $j1\in J1\subset J$, $i1\in I$, $r1=\overline{1,R1}$, в SQL-нотации запишется в виде

OOx1(x) ≡ Ox1(x) and Ox2(x) … and Oxr(x) … and OxR1(x) (9)

По аналогии с $F1()$ введем функцию $F2() = F2(z[s_{j2},u_{m1}]\ \Theta_{r2}\ z_{j2})$, соединяющую логическим "И" несколько элементарных ограничений одного ребра:

$$F2()=\bigcap_{r2=1}^{R2}(z[s_{j2},u_{m1}]\Theta_{r2}\ z_{j2,t2}),\ j2\in J2\subset J,\ m1\in M,$$
$$r2=\overline{1,R2},\ z_{j2,t2}\in Z_{j2},\ t2\in\overline{1,T_{j2}}, \quad (10)$$

где $J2$ – множество свойств соединяемых ограничений, $R2$ – количество соединяемых ограничений.

В общем случае $F2()$, $j2\in J2\subset J$, $m1\in M$, $r2=\overline{1,R2}$, $z_{j2,t2}\in Z_{j2}$, $t2\in\overline{1,T_{j2}}$ в SQL-нотации запишется в виде

OOu1(u) ≡ Ou1(u) and Ou2(u) … and Our(u)… and OuR2(u) (11)

Введем функцию $F3(F1(z[s_{j1},x_{i1}]\ \Theta_{r1}\ z_{j1,t1}))$, соединяющую логическим "И" группы ограничений разных вершин:

$$F3(F1())=\bigcap_{\forall i1\in I1}F1_{i1}(z[s_{j1},x_{i1}]\Theta_{r1}\ z_{j1,t1})=\bigcap_{\forall i1\in I1}\bigcap_{r1=1}^{R1_{i1}}(z[s_{j1},x_{i1}]\Theta_{r1}\ z_{j1,t1})$$
$$j1\in J1_{i1}\subset J,\ i1\in I1\subset I,\ r1=\overline{1,R1}_{i1},$$
$$z_{j1,t1}\in Z_{j1},\ t1\in\overline{1,T_{j1}}, \quad (12)$$

где $I1$ – множество соединяемых вершин, J_{f1} – множество соединяемых свойств вершины $i1$, $R1_{i1}$ – количество соединяемых ограничений вершины $i1$.

Запишем проверку истинности выражения (12) в SQL-нотации:

```
OOOx ≡  exists (select * from X where OOx1(ID_X))
          and                                              (13)
        exists (select * from X where OOx2(ID_X))
          and
        ……
        exists (select * from X where OOxI1(ID_X))
```

Например, если среда в одном аппарате взрывоопасна, а в другом токсична и для свойства «взрывоопасность» ID_S=10, а для свойства «токсичность» ID_S=11, $R_{i1} = 1$, $R_{i2} = 1$, выражение (12) примет вид

$$F3(F1()) = (z[s_{10}, x_{i1}] = "Да") \;\; И \;\; (z[s_{11}, x_{i2}] = "Да"),$$
$$i1 \neq i2, i1, i2 \in I1 \subset I$$

Ox1(X.ID_X) ≡ exists (select * from SX where SX.ID_X=X.ID_X and SX.ID_S="10" and SX.Z="Да")

Ox2(X.ID_X) ≡ exists (select * from SX where SX.ID_X=X.ID_X and SX.ID_S="11" and SX.Z="Да")

OOx1(X.ID_X) ≡ Ox1(X.ID_X)

OOx2(X.ID_X) ≡ Ox2(X.ID_X)

```
OOOx ≡ exists (select * from X where OOx1(ID_X)) and
       exists (select * from X where OOx2(ID_X))
```

В реальных практических задачах (по крайней мере в предметной области компоновки химического оборудования) авторам неизвестны ограничения, которые объединяли бы свойства нескольких помещений. Несмотря на это для общности введем по аналогии с $F3(F1())$ функцию $F4(F2())$, которая логически связывает группы свойств разных ребер:

$$F4(F2()) = \bigcap_{\forall m1 \in M1} F2_{m1}(z[s_{j2}, u_{m1}]\Theta_{r2}\, z_{j2,t2}) = \bigcap_{\forall m1 \in M1} \bigcap_{r2=1}^{R2_{m1}} (z[s_{j1}, u_{m1}]\Theta_{r2}\, z_{j2,t2})$$
$$j2 \in J2_{m1} \subset J,\; m1 \in M1 \subset M,\; r2 = \overline{1, R2}_{m1},$$
$$z_{j2,t2} \in Z_{j2},\; t1 \in \overline{1, T_{m1}}, \qquad (14)$$

где $M1$ – множество соединяемых ребер, J_{m1} – множество соединяемых свойств ребра $m1$, $R2_{m1}$ – количество соединяемых ограничений ребра $m1$.

По аналогии с (12) и (13) запишем проверку выражения (14):

```
OOOu ≡  exists (select * from U where OOu1(ID_U))
          and                                              (15)
        exists (select * from U where OOu2(ID_U))
          and
        ……
        exists (select * from U where OOuM1(ID_U))
```

Выражения Ox1, Ou1, OOx1, OOu1, OOOx, OOOu представляют собой языковые конструкции, которые легко получить программным путем.

5. ПРАВИЛА, ОГРАНИЧИВАЮЩИЕ ПРИНАДЛЕЖНОСТЬ ВЕРШИНЫ РЕБРУ

Правила (ограничения) первого уровня декомпозиции, которые определяют принадлежность вершины ребру, можно разделить на три группы.

5.1. Принадлежность вершины x_{i1} ребру u_{m1}, $x_{i1} \in X1_m$ определяется одним свойством s_{j1} вершины x_{i1} и одним свойством s_{j2} ребра. Например, если масса аппарата больше 100 000 кг, то аппарат следует располагать на первом этаже.

$$z[s_{j1}, x_{i1}]\Theta_1\, z_{j1,t1} \wedge\; x_{i1} \in X1_{m1}, m1 \in M,\; z_{j1,t1} \in Z_{j1},$$
$$j1 \in J1 \subset J,\; t1 = \overline{1, T_{j1}},\;\; i1 \in I,$$
$$m1 \in M \;\Rightarrow\; z[s_{j2}, u_{m1}]\Theta_2\, z_{j2,t2},$$
$$j2 \in J2 \subset J,\; z_{j2,t2} \in Z_{j2}\;\; t2 = \overline{1, T_{j2}} \qquad (16)$$

Представим синтаксис запросов типов 1 – 3, основанных на правилах (16), в нотации SQL. Синтаксис запросов типов 4 и 5 достаточно легко получается из запросов типа 2, 3 отрицанием ограничений.

Заданы @ID_S1 – $j1$ свойство вершины, @z1 – $z_{j1,t1}$ значение свойства вершины, @ID_S2 – $j2$ свойство ребра, @z2 – $z_{j2,t2}$ значение свойства ребра.

Тип 1. Ответ на вопрос, можно ли размещать аппарат @ID_X в помещении @ID_U.

Введем переменную @ret = 1, если аппарат в заданном помещении располагать можно, @ret = 0 в противном случае.

```
if Ox1(@ID_X)                                              (17)
  if  Ou1 (@ID_U)
      set @ret=1
  else set @ret=0
```

Тип 2. Получить все помещения, в которых можно размещать аппарат @ID_X.

```
If Ox1(@ID_X) select ID_U from U
   where Ou1(U.ID_U)                                       (18)
```

Тип 3. Получить список аппаратов, которые необходимо ставить в помещении @ID_U.

```
if Ou1(@ID_U) select ID_X from X
   where Ox1(ID_X)                                         (19)
```

Представленное разделение запросов на типы позволяет ввести правило один раз и автоматически сформировать рассмотренные конструкции запросов. Рассмотрим это на следующем примере.

Правило «Если масса аппарата больше 100 000 кг, то аппарат надо располагать на первом этаже». В нотации, удобной для ввода в базу, это правило выглядит следующим образом: «Если свойство аппарата = "масса" и значение свойства >100000, то свойство помещения = "этаж" и значение свойства = "первый"».

Предположим, что для свойства "масса" ID_S=6, для свойства "этаж" ID_S=7. Сформировать программно проверку элементарных ограничений нетрудно:

Ox1(@ID_X) ≡ exists (select * from SX where SX.ID_X=@ID_X and SX.ID_S=6 and SX.Z>10000),

Ox1(SX.ID_X) ≡ exists (select * from SX where SX.ID_X=X.ID_X and SX.ID_S=6 and SX.Z>10000),

Ou1(@ID_U) ≡ exists (select * from SU where SU.ID_U=@ID_U and SU.ID_S=7 and SU.Z="первый"),

Ou1(U.ID_U) ≡ exists (select * from SU where SU.ID_U=U.ID_U and SU.ID_S=7 and SU.Z="первый").

Подставив полученные выражения в языковые конструкции (17) – (19), получаем синтаксис запросов, основанных на одном правиле – «Если масса ап-

парата больше 100 000 кг, то аппарат надо располагать на первом этаже».

5.2. Принадлежность вершины x_{i1} ребру $u_{m1}, x_{i1} \in X1_m$ определяется группой свойств вершины $\{s_{j1}\} \subset S$, $j1 \in J1 \subset J$, соединенных логическими условиями "И", и группой свойств ребра $\{s_{j2}\} \subset S, j2 \in J2 \subset J$, соединенных логическими условиями "И".

Пример правила: «Если среда в аппарате взрывоопасна и токсична, то помещение должно быть оборудовано принудительной вентиляцией и иметь эвакуационный выход».

$$F1(z[s_{j1}, x_{i1}]\Theta_{r1}\, z_{j1,t1}) \wedge x_{i1} \in X1_{m1}, m1 \in M,$$

$$z_{j1,t1} \in Z_{j1},\ j1 \in J1 \subset J,\ t1 = \overline{1, T_{j1}},\ \ i1 \in I,\ r1 = \overline{1, R1},$$

$$m1 \in M \Rightarrow F2(z[s_{j2}, u_{m1}]\Theta_{r2}\, z_{j2,t2}),$$

$$z_{j2,t2} \in Z_{j2},\ j2 \in J2 \subset J,\ t2 = \overline{1, T_{j2}},\ r2 = \overline{1, R2}$$

Проверка истинности логических функций $F1()$ и $F2()$ в нотации SQL запишется конструкциями OOx1(x) и OOu1(u), см. выражения (9) и (11).

SQL-выражения для всех типов запросов первого уровня декомпозиции (типы 1 – 3) легко получить из (17) – (19) заменой конструкций Ox1(X.ID_X), Ox1(@ID_X), Ou1(U.ID_U), Ou1(@ID_U) на OOx1(X.ID_X), OOx1(@ID_X), OOu1(U.ID_U), OOu1(@ID_U).

5.3. Принадлежность вершины x_{i1}, ребру u_{m1}, $x_{i1} \in X1_m$, определяется принадлежностью этому ребру другой вершины x_2, $x_2 \in X1_{m1}$.

Вершина x_{i1} обладает свойствами $\{s_{j1}\} \subset S$, $j1 \in J1 \subset J$, вершина x_2 – свойствами $\{s_{j2}\} \subset S$, $j2 \in J2 \subset J$, для которых выполняются ограничения типа $F1()$, см. выражение (7).

Пример правила: «Токарные станки следует располагать в одном помещении». Это правило можно сформулировать и по-другому: «Если имеется станок, тип которого "токарный", и он расположен в помещении u_{m1}, то другие станки с типом "токарный" следует располагать в помещении u_{m1}». Функция $F1()$ в этом случае ограничивает свойство "тип станка" значением "токарный".

$$F1(z[s_{j1}, x_{i1}]\Theta_{r1}\, z_{j1,t1}) \wedge F1(z[s_{j2}, x_{i2}]\Theta_{r2}\, z_{j2,t2}) \wedge (x_{i2} \in X_{m1}),$$

$$z_{j1,t1} \in Z_j,\ j1 \in J1 \subset J,\ t1 = \overline{1, T_{j1}},\ \ i1 \in I,\ z_{j2,t2} \in Z_{j2},$$

$$j2 \in J2 \subset J,\ t2 = \overline{1, T_{j2}},\ \ i2 \in I,$$

$$m1 \in M \Rightarrow x_{i1} \in X_{m1} \qquad (20)$$

Представим синтаксис всех запросов, основанных на правилах (20), в нотации SQL:

Тип 1. Ответ на вопрос, можно ли размещать аппарат @ID_X в помещении @ID_U. Введем переменную @ret = 1, если аппарат в заданном помещении располагать можно, @ret = 0 в противном случае.

```
if Ox1(@ID_X)
  if exists (select * from G,X
    where G.ID_X=X.ID_X and G.ID_U=@ID_U
    and Ox1(X.ID_X))
      set @ret=1
 else set @ret=0
```

Тип 2. Получить все помещения, в которых можно размещать аппарат @ID_X.

```
if Ox1(@ID_X)
  select U.ID_U from U where exists (select G.ID_U
  from G, X
    where G.ID_X=X.ID_X and X.ID_X<>@ID_X
    and U.ID_U=G.ID_U and OX1(X.ID_X)
```

Тип 3. Получить список аппаратов, которые необходимо ставить в помещении @ID_U.

```
select X1.ID_X from X X1 where Ox1(X1.ID_X) and
exists (select G.ID_U
  from G, X where G.ID_U=@ID_U
  and X.ID_X=G.ID_X and Ox1(X.ID_X)
```

6. ПРАВИЛА, ОПРЕДЕЛЯЮЩИЕ СВОЙСТВА РЕБРА В ЗАВИСИМОСТИ ОТ СВОЙСТВ ВЕРШИНЫ

Это правила, позволяющие по известным определенным свойствам вершины x_{i1}, принадлежащей ребру u_{m1}, найти определенные свойства этого ребра. Например: «Если взрывоопасный аппарат находится в помещении, то это помещение относится к категории А».

В практических задачах существуют следующие группы правил, определяющих свойства ребра в зависимости от свойств вершин.

6.1. Свойство s_{j2} ребра u_{m1} определяется одним свойством s_{j1} вершины x_{i1} этого ребра $x_{i1} \in X1_{m1}$.

$$z[s_{j1}, x_{i1}]\Theta\, z_{j1,t1} \wedge x_{i1} \in X1_{m1}, m1 \in M,\ z_{j1,t1} \in Z_{j1},$$

$$j1 \in J1 \subset J,\ i1 \in I,\ t1 \in \overline{1, T_{j1}} \Rightarrow$$

$$z[s_{j2}, u_{m1}] = z_{j2,t2},\ z_{j2,t2} \in Z_{j2},\ j2 \in J2 \subset J,$$

$$t2 \in \overline{1, T_{j2}}$$

SQL-оператор изменения свойств ребер запишется следующим образом. Заданы @ID_S1 – $j1$ свойство вершины, @z1 – $z_{j1,t1}$ значение свойства вершины, @ID_S2 – $j2$ свойство ребра, @z2 – $z_{j2,t2}$ значение свойства ребра. Необходимо во всех ребрах, которые имеют свойство @ID_S2, изменить значение этого свойства на @z2, если ребра содержат вершины, для которых справедливо элементарное ограничение Ox1(X.ID_X).

```
update SU set Z=@z2 where ID_S=@IZ_S2 and
ID_U in (select G.ID_U                    (21)
    from G, X  where X.ID_X=G.ID_X and
Ox1(X.ID_X))
```

6.2. Свойство s_{j2} ребра u_{m1} определяется группой свойств $\{s_{j1}\} \subset S, j1 \in J1 \subset J$ вершины x_{i1}.

$$F1(z[s_{j1}, x_{i1}]\Theta_{r1}\, z_{j1,t1}) \wedge x_{i1} \in X1_{m1}, m1 \in M,\ z_{j1,t1} \in Z_j,$$

$$\forall j1 \in J1 \subset J,\ i1 \in I,$$

$$t1 \in \overline{1, T_j},\ r1 = \overline{1, R1} \Rightarrow z[s_{j2}, u_{m1}] = z_{j2,t2},\ z_{j2,t2} \in Z_j,$$

$$j2 \in J,\ t2 \in \overline{1, T_j},$$

где $F1(z[s_{j1}, x_{i1}]\Theta_{r1}\, z_{j1,t1}) = \bigcap_{r1=1}^{R1}(z[s_{j1}, x_{i1}]\Theta_{r1}\, z_{j1,t1})$ – логическая функция, связывающая значения свойств вершины, $R1$ – количество связанных свойств.

В нотации SQL корректировка свойства ребра в этом случае получается из выражения (21), в котором Ox1(ID_X) заменяется на OOx1(ID_X), см. выражение (9).

6.3. Свойство s_{j2} ребра u_{m1} определяется группой свойств $\{s_{j1}\} \subset S$, $j1 \in J1 \subset J$ группы вершин $X1 = \{x_{i1}\}$, $i1 \in I1 \subset I$ этого ребра (т.е. свойства могут принадлежать разным вершинам ребра)**.**

$$\bigcap_{\forall i1 \in I1} F_{i1}(z[s_{j1}, x_{i1}]\Theta_{i1,r1}\, z_{j1,t1}) \wedge \bigcap_{\forall i1 \in I1} x_{i1} \in X1_{m1},\ m1 \in M,$$

$$z_{j1,\,t} \in Z_{j1},\ \forall j1 \in J1_{i1} \subset J,\ \forall i1 \in I1 \subset I,\ r1 = \overline{1, R1_{i1}},$$

$$t1 = \overline{1, T1_{i1}} \Rightarrow\ z[s_{j2}, u_{m1}] = z_{j2,t2},\ z_{j2,t2} \in Z_{j2},$$

$$j2 \in J1_{m1} \subset J,\ t2 \in \overline{1, T2}$$

Корректировка значения свойства ребер в нотации SQL в этом случае получается из выражения (21), в котором Ox1(ID_X) заменяется на OOOx, см. выражение (13).

7. ПРАВИЛА, ОПРЕДЕЛЯЮЩИЕ СВОЙСТВА ВЕРШИН В ЗАВИСИМОСТИ ОТ СВОЙСТВ РЕБРА

Эти правила, позволяют по известным определенным свойствам ребра u_{m1} найти определенные свойства принадлежащих ему вершин $x_{i1} \in X_m$. В практических задачах существуют следующие группы правил, определяющих свойства вершин в зависимости от свойств ребра.

7.1. Свойство s_{j1} вершин $x_{i1} \in X_m$ ребра u_{m1} определяется одним свойством ребра s_{j2}.

$$z[s_{j2}, u_{m1}]\Theta\, z_{j2,t2},\ m1 \in M,\ z_{j2,t2} \in Z_j,$$

$$j2 \in J2 \subset J,\ t2 \in \overline{1, T2} \Rightarrow$$

$$z[s_{j1}, x_{i1}] = z_{j1,t1},\ z_{j1,t1} \in Z_j,\ j1 \in J1 \subset J,$$

$$t1 \in \overline{1, T1},\ x_{i1} \in X1_{m1}$$

SQL-оператор изменения свойств ребер запишется следующим образом. Заданы @ID_S2 – $j2$ свойство ребра, @z2 – $z_{j2,t2}$ значение свойства ребра, @ID_S1 – $j1$ свойство вершины, @z1 – $z_{j1,t1}$ значение свойства вершины. В случае истинности элементарного ограничения Ou1(U.ID_U) необходимо у всех вершин ребра, у которых есть свойство SX.ID_S=@ID_S1, изменить его значение SX.Z=@z1.

```
update SX set Z=@z1 where ID_S=@ID_S1
 and ID_X in (select G.ID_X                    (22)
from U, G where U.ID_U=G.ID_U and Ou1(U.ID_U))
```

7.2. Свойство s_{j1} вершин $x_{i1} \in X_{m1}$ ребра u_{m1} определяется группой свойств $\{s_{j2}\} \subset S, j2 \in J2 \subset J$ этого ребра.

$$F2(z[s_{j2}, u_{m1}]\ \Theta_{r2}\, z_{j2,t2}), m1 \in M,\ z_{j2,t2} \in Z_j,$$

$$j2 \in J2 \subset J,\ t2 \in \overline{1, T2},\ r2 = \overline{1, R2} \Rightarrow$$

$$z[s_{j1}, x_{i1}] = z_{j1,t1},\ z_{j1,t1} \in Z_j,\ j1 \in J1 \subset J,$$

$$t1 \in \overline{1, T1},\ x_{i1} \in X1_{m1},$$

где $F2(z[s_{j2}, u_{m1}]\Theta_{r2}\, z_{j2,t2}) = \bigcap_{r2=1}^{R2}(z[s_{j2}, u_{m1}]\Theta_{r2}\, z_{j2,t2})$ – логическая функция, связывающая значения свойств ребра, R2 – число элементарных ограничений (число свойств ребра, определяющих свойства вершины).

SQL-оператор изменения свойств вершин получается заменой Ou1(U.ID_U) в (22) на OOu1(U.ID_U), см. выражение (11).

8. ПРАВИЛА, ОПРЕДЕЛЯЮЩИЕ СВОЙСТВА ВЕРШИНЫ В ЗАВИСИМОСТИ ОТ СВОЙСТВ ДРУГИХ ВЕРШИН

Правила, позволяющие найти значение свойства $j3$ одной вершины x_{i3} по известным свойствам $\{s_{j1}\} \subset S$, $j1 \in J1 \subset J$ другой вершины x_{i1}, причем вершины принадлежат одному ребру $x_{i1}, x_{i3} \in X1_{m1}$. В общем случае может быть, что $i1 = i3$, т.е. некоторое свойство вершины определяется через другие ее свойства, тогда определяемое свойство $j3$ не должно принадлежать множеству определяющих свойств $J1, j3 \notin J1$.

Пример правила: «Однотипные аппараты следует располагать в ряд». Это означает, что если аппараты располагаются в ряд по оси y и найдена (предложена) координата x одного аппарата, то и другие аппараты этого типа будут иметь такую же координату x.

$$F1(z[s_{j1}, x_{i1}]\Theta_{r1}\, z_{j1,t1}) \wedge x_{i1} \in X1_{m1}, m1 \in M,\ z_{j1,t1} \in Z_j,$$

$$j1 \in J1 \subset J,\ i1 \in I,\ t1 \in \overline{1, T_j},$$

$$r1 = \overline{1, R1} \Rightarrow\ z[s_{j3}, x_{i3}] = z_{j3,t3},\ z_{j3,t3} \in Z_j,\ j3 \in J,$$

$$t3 \in \overline{1, T_{j3}},\ i3 \in I,\ x_{i3} \in X1_{m1},$$

если $i3 = i1,\ j3 \notin J1$**,** где

$$F1(z[s_{j1}, x_{i1}]\Theta_{r1}\, z_{j1,t1}) = \bigcap_{r1=1}^{R1}(z[s_{j1}, x_{i1}]\Theta_{r1}\, z_{j1,t1}).$$

Для записи этого правила в нотации SQL введем обозначения. Известны @ID_X – определяющая вершина, набор свойств $J1$ и их значения для определяющей вершины, @ID_S3 – определяемое (изменяемое) свойство, @Z3 – новое значение определяемого свойства. Запишем правило для условия $i3 \neq i1$:

```
if OOx1(@ID_X)
  update SX set Z=@Z3 where ID_S=@ID_S3 and
    ID_X in (select G.ID_X from G where
      G.ID_X<>@ID_X
      and G.ID_U in (select G1.ID_U from G1 where
        G1.ID_X=@ID_X))
```

Условие G.ID_X<>@ID_X соответствует условию $i3 \neq i1$**.**

9. ХРАНЕНИЕ И ОБРАБОТКА ПРАВИЛ

На рис. 2 представлена структура базы данных для хранения правил. Правила записаны в нотации SQL и хранятся в таблицах PP, PR и PW в поле Текст_на_SQL в текстовом формате (nvarchar(max)). В поле Текст_на_ЕЯ хранится текст правила на естественном языке, который необхо-

дим для выдачи пояснения, почему принято то или иное решение.

Таблица PP содержит правила, определяющие принадлежность вершин ребру. Поле PP.Тип определяет характер запроса, на который отвечает правило (типы 1 – 5). В таблицах PR и PW хранятся правила, которые позволяют найти свойства ребер и вершин в зависимости от текущего состояния базы данных. Поле ID_S в таблицах PR и PW содержит первичный ключ искомого свойства.

Таблицы SPP, SPR и SPW содержат первичные ключи свойств ID_S, которые присутствуют в тексте правил. Эти таблицы нужны для обеспечения целостности базы, они не позволяют удалить свойство из таблицы S, если оно присутствует в тексте правила.

После поступления запроса любого типа сначала изменяются свойства вершин и ребер в зависимости от текущей ситуации решения задачи размещения. Для этого в цикле обрабатываются последовательно все правила в таблицах PR и PW до тех пор, пока не будет выполнено ни одно правило. Затем осуществляется однократный проход по правилам таблицы PP. Если запрос первого типа, то результат будет 1 или 0 (можно или нельзя размещать вершину в ребре). Для остальных типов правил результаты работы каждого правила запоминаются во временной таблице, из которой затем удаляются дубликаты.

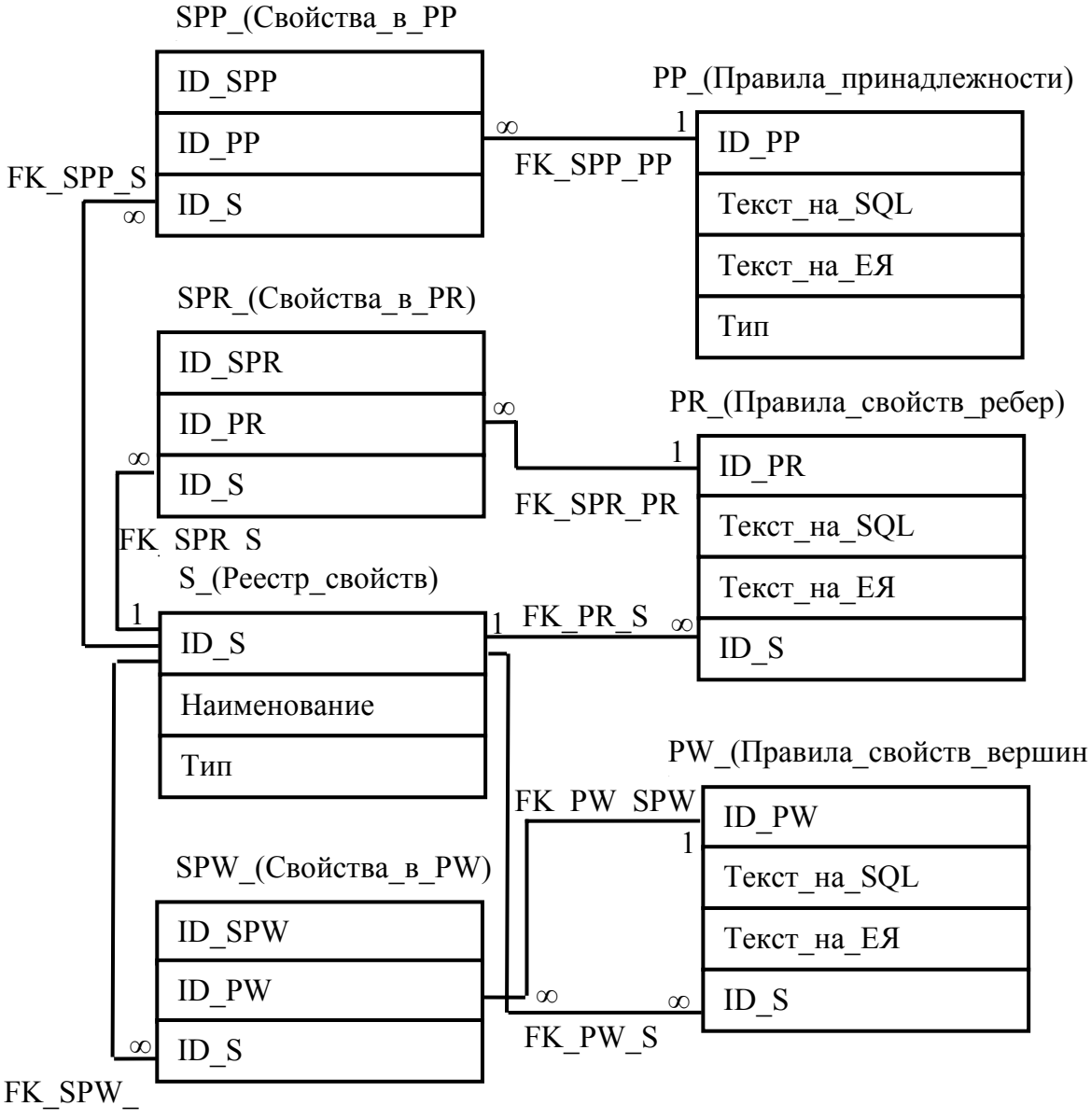

Рис. 2. Структура базы данных для хранения правил

Процедура формирования ответа для запроса первого типа (можно ли размещать вершину в ребре) представлена ниже:

```
create procedure PP_1 @ID_X1 int, @ID_U1 int, @ret1 int output
-- @ret1 = 1, если вершину @ID_X1 МОЖНО размещать в ребре @ID_U1
-- @ret1 = 0, если вершину @ID_X1 НЕЛЬЗЯ размещать в ребре @ID_U1
as
  declare @PP nvarchar(max)  -- текст правила
  declare @param nvarchar(max) -- параметры правила
  -- cursor_PP набор правил первого типа
  declare cursor_PP  cursor for select Текст_на_SQL from PP where Тип=1
  set @param=N'@ID_X int, @ID_U int, @ret int output'
  set @ret1=1 -- предполагаем, что размещать вершину в ребре можно
  open cursor_PP
  Fetch Next from cursor_PP into @PP -- чтение первого правила
  -- функция @@fetch_status = 0, если чтение правила прошло успешно
  while @@fetch_status=0 and @ret1=1
      begin
          -- выполнение правила
          execute sp_executesql @PP, @param,
      @ID_X=@ID_X1,
              @ID_U=@ID_U1, @ret=@ret1 output
          Fetch Next from cursor_PP into @PP -- чтение следующего правила
      end
  close cursor_PP
  deallocate cursor_PP
```

Представленная процедура существенно упрощена по сравнению с реальной (промышленной). Например, в ней не приведен механизм формирования пояснения, почему вершину нельзя размещать в указанном ребре. Кроме того, процедура прекращает работать, как только найдено первое правило, запрещающее размещать вершину @ID_X в ребре @ID_U. Просмотр всех правил позволяет пользователю получить не одно, а все условия, которые запрещают размещение вершины @ID_X в ребре @ID_U, что необходимо для последующего анализа. В реальной процедуре пользователь сам выбирает режим работы: выход по первому запрещающему условию или получение всех запрещающих условий.

ЗАКЛЮЧЕНИЕ

Рассмотренный подход может быть полезен при разработке новых автоматизированных систем размещения объектов в пространстве или при разработке дополнений к уже существующим информационным системам размещения объектов. Этот подход был использован авторами при создании автоматизированной системы размещения химического оборудования в производственных помещениях.

СПИСОК ЛИТЕРАТУРЫ

1. Егоров С. Я., Немтинов В. А., Мокрозуб В. Г., Милованов И. В. Автоматизированная информационная система поддержки проектных решений по компоновке промышленных объектов. Ч. 1. Аналитические и процедурные модели // Информационные технологии в проектировании и производстве. – 2009. – № 4. – С. 3 – 11.
2. Егоров С. Я., Немтинов В. А., Мокрозуб В. Г., Громов М. С. Автоматизированная информационная система поддержки проектных решений по компоновке промышленных объектов. Ч. 2. Структура и функционирование системы // Информационные технологии в проектировании и производстве. – 2010. – №1. – С. 33 – 39.
3. Мокрозуб В. Г., Немтинов В. А., Мордвин А. С., Илясов А. А. Применение N-ориентированных гиперграфов и реляционных баз данных для структурного и параметрического синтеза технических систем // Прикладная информатика. – 2010. – № 4 (28). – С. 115 – 122.

Материал поступил в редакцию 18.02.11.

Сведения об авторах

МОКРОЗУБ Владимир Григорьевич – кандидат технических наук, доцент, член-корреспондент РАЕН, профессор кафедры автоматизированного проектирования технологического оборудования (АПТО) Тамбовского государственного технического университета (ТГТУ)
E-mail: mokrozubv@yandex.ru

ШАРОНИН Кирилл Анатольевич – студент 5-го курса кафедры САПР ТГТУ

НЕМТИНОВ Кирилл Владимирович – студент 4-го курса кафедры АПТО ТГТУ

ИНФОРМАЦИОННЫЕ ЯЗЫКИ

УДК 004.92/.93

Т. Г. Петров

Графическое отображение процессов эволюции составов поликомпонентных объектов любой природы

Описан способ графического отображения процессов эволюции составов. Подготовка материалов для расчетов заключается в ранжировании содержаний компонентов по их снижению и стандартизации длины получаемых последовательностей с отбрасыванием «избыточных» содержаний. Для сохранившегося распределения содержаний рассчитывают три характеристики: 1) информационную энтропию $\boldsymbol{H} = -\Sigma p_i * \lg p_i$ *как меру сложности состава системы, 2) анэнтропию* $\boldsymbol{A} = -\Sigma \lg p_i$ *как меру чистоты состава, 3) толерантность* $\boldsymbol{T} = \lg[(\Sigma 1/ p_i)/n]$ *как меру особой чистоты. Для отображения процесса изменения состава используют парные диаграммы с осями En, An, T. Получаемые энтропийные диаграммы наиболее адекватно описывают процессы разделения и смешения, идущие в природе, технологиях и социуме.*

Ключевые слова: *эволюция составов, информационная энтропия, сложность состава, энтропия смешения, анэнтропия, энтропия разделения, чистота состава, энтропийная диаграмма, толерантность, теорема Шурубора, горные породы, национальный состав, коэффициент Джини*

ВВЕДЕНИЕ

Термодинамика рассматривает три модели систем по степени их открытости. Открытые, которые обмениваются со средой энергией и веществом, закрытые, обменивающиеся со средой только энергией, и изолированные – при отсутствии любых видов обмена со средой. Лингвистический барьер – «термо» исключает из внимания термодинамики системы, для которых значим и доступен для изучения обмен со средой *веществом с изменением состава*, в то время как обмен энергией *не значим, не доступен, или весьма затруднен для исследования*. Такой подход широко распространен в разных отраслях знания. Примером может служить геология с ее многочисленными видами эволюционирующих во времени и изменчивых в пространстве поликомпонентных объектов (систем): химических, минеральных, горно-породных, гранулометрических, изотопных. Аналогично положение с биологическими, национальными, финансовыми и иными системами. Одним из важных генетически значимых свойств составов является их сохранность при изменении структуры, наследование новой структурой компонентов бывшего состава даже при существенном изменении.

Разнообразие систем и накопленные объемы аналитической информации требуют для описания составов и их эволюции использования характеристик, которые обеспечивают сжатие информации о единичном составе и ее систематизацию, а также позволяют увидеть общность между процессами в разных областях знания. Эта общность выявляется на интуитивном уровне, например, когда говорят об усложнении составов или повышении чистоты при процессах, весьма различающихся в материальном отношении (обогащение руды, очистка зерна, воды, химического реактива, «национальные чистки» и пр.).

Для решения этой задачи были предложены три характеристики [1, 2], которыми можно описать составы, сумма компонентов которых нормирована к 100% или к 1. Это следующие: 1) список *главных* компонентов, 2) *сложность* состава, 3) его *чистота*. Первая характеристика принадлежит конкретному объекту. Вторая и третья характеристики «обезличены», они свободны от конкретизации состава. После процедуры выбора компонентов для расчетов их имена утрачивают какое-либо значение, как это происходит в статистических и иных математических методах. Сложность и чистота общеизвестны как качественные оценки объекта или ситуации, но понятие сложности не имело меры до 1928 г. [3], а чистоты – до 1971 г. [1]. Позднее [4] была предложена характеристика для описания особо чистых составов – *стерильность или толерантность*. Все три последние характеристики связаны с широко известными процессами интеграции и дифференциации, которые Г. Спенсер [5, 6] назвал фундаментальными. При рассмотрении вещества и во многих других случаях более удобны термины *смешение* и *разделение.* Заметим, соединение неразличимых частей НЕ является смешением, так же как не является разделением разобщение системы без изменения составов в получающихся частях.

Многократное изложение метода ***RHA*** [1, 2, 4, 7-11] в основном было направлено на проблему систематизации материала и сравнительно слабо затрагивало энтропийные характеристики. С другой сто-

роны, термин «энтропия», можно сказать, довольно сильно дискредитирован многозначностью, разнообразием вариантов его толкований и распространением вплоть до песенного репертуара. Это обстоятельство в некоторых областях знания стало вызывать недоверие к его использованию и тем самым тормозить развитие науки. В связи с этим представляется необходимым изложить смысл, который вкладывается в используемые далее термины – «информационная энтропия» и «анэнтропия», начиная с интуитивного уровня, с рассмотрения понятий и их отношений в естественном языке, далее перевести их на математический язык и показать возможности использования энтропийной диаграммы как инструмента содержательного описания процессов изменения составов поликомпонентных объектов – ценозов, сообществ.

Целью работы является описание диаграммы, предназначенной для отображения процессов изменения ранжированных распределений, в которых интенсивности (вероятности, частоты) могут относиться к разным классам свойств и параметров, т. е. диаграммы для описания процессов изменения составов, абстрагированных от свойств их конкретных компонентов.

«СЛОЖНОСТЬ» И «ЧИСТОТА» СОСТАВОВ НА ИНТУИТИВНОМ УРОВНЕ

Под составом объекта или системы понимается перечень компонентов и их содержаний (концентраций). Если в системе присутствует один или мало компонентов, ее называют простой, ее *сложность* низка. Когда компонентов много, и тем более соизмеримых по содержаниям – сложной. Поэтому система, в которой присутствует один компонент, может считаться минимально сложной, система в которой всех компонентов поровну – предельно сложной для имеющегося числа компонентов. В природе не существует ни систем, состоящих строго из единственного компонента, ни систем, в которых всех компонентов строго поровну. Любое вещество, будь то минерал, или химический реактив, имеет примеси, которые искажают точное равенство содержаний элементов.

С другой стороны, предельно *чистую* систему логично считать состоящей из одного-единственного «простого вещества», не содержащего никаких примесей. Такая система тоже идеал, *абсолютно чистых составов объектов не существует*. Так, в чистейших, специально очищаемых («особо», или «высоко чистых») веществах, что достигается с большими затратами времени и средств, отражающимися в ценах на эти вещества, устанавливается наличие десятков элементов [12]. За минимально чистую систему примем такую, в которой компонентов поровну, т. е., такую, которая выше была названа максимально сложной.

Между «простым» и «сложным», а также между «чистым» и «предельно грязным» существуют непрерывные переходы, которые в естественном языке расчленены, имеют собственные, не очень определенные имена. Между интенсивностями свойств существует, в общем, обратная связь. Если привести в соответствие качественные оценки той и другой характеристик, получится схема, изображенная на рис. 1.

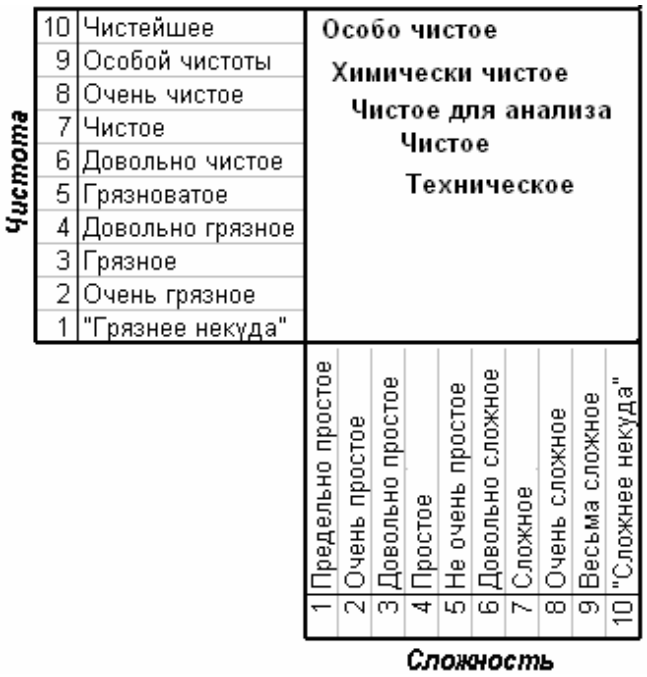

Рис. 1. Соотношения между интуитивными оценками сложности, чистоты и их приблизительное соответствие квалификации химических реактивов

Из рассмотрения схемы следует, с одной стороны, невозможность существования составов в областях, где предельная высокая сложность сочетается с предельно высокой чистотой и наоборот. С другой стороны, предельно низкая сложность соответствует предельной чистоте, и предельно высокая сложность соответствует предельно низкой чистоте. Предложим непрерывные количественные характеристики для описания поля сложности и чистоты, уйдя от качественных оценок, свойственных естественным языкам.

ОТ ЕСТЕСТВЕННОГО ЯЗЫКА К МАТЕМАТИЧЕСКОМУ

Компонентами могут быть совокупности одинаковых в некотором отношении частиц (элементов) системы, четко отличимых от иных и стабильных в течение времени исследования. Содержания (p) компонентов (i) выражены в долях единицы, так что $\Sigma p_i = 1$.

Потребность в получении характеристик сложности состава, не привязанных к определенной отрасли знания, определила ориентацию поиска математического аппарата в направлении теории информации как отрасли математики, занимающейся порождением, кодированием, передачей, приемом и хранением сигналов любой природы. Обращение к этой области знания при разработке метода было обусловлено и еще тем, что были установлены глубокие связи между процессами кристаллизации из сложных сред – растворов, вездесущих на Земле [13] и процессами информационными [14].

Согласно методу ***RHA***, предназначенному для описания составов, их систематизации и описания процессов их изменений [1, 2, 4, 8; 10], все компоненты априорно равноценны для изучения. В то же время, используемая разными авторами длина списка компонентов в анализе различна. Различны и списки компонентов. Поскольку сложность зависит от количества компонентов [15], для сопоставимости расчетов необходим их

выбор в каждом анализе. Выбор не может опираться на «ценность», «важность», «общеизвестность» и тому подобные профессиональные центры внимания. В используемом методе выбор компонентов формализован. Он производится в два этапа.

Этап 1. Ранжирование компонентов по снижению их содержаний. Соответственно, на первое место всегда будет выходить компонент с максимальным содержанием. На последнем месте – с минимальным из числа имеющихся в анализе. Полученная последовательность символов компонентов – ***ранговая формула*** – является первым и необходимым шагом для всех иных дальнейших действий по описываемому методу. Ранговая формула это имя ранжированного распределения – его качественная и, одновременно, полуколичественная характеристика, использованная, в частности, при создании ***R***-словаря химических составов минералов [16]. В контексте задачи статьи ранговая формула играет важную, но только служебную роль.

Этап 2. Стандартизация длины перечня компонентов. Она производится посредством отсечения «избыточных» компонентов с их содержаниями. Отсечение производится с учетом баланса между двумя противоположными интенциями. Согласно первой – приближение суммы анализа к 1 или к 100 процентам. Чем больше учитывается элементов, тем более полна информация об объекте, выше информированность о нем, выше различимость составов, соответственно, у него меньше аналогов, и потому обобщения охватывают меньшие области сходных составов. При этом больше нагрузка на память, внимание, зрение. Согласно второй – минимизация количества учитываемых компонентов. Здесь значима необходимость платы за каждое элементоопределение при химическом анализе, за каждый вопрос в статистическом протоколе, и то, что чем меньше элементов учтено, тем беднее, «грубее» информация о составе, тем больше обнаруживается «сходных» объектов, а поэтому больше вероятность ошибок при идентификации. При этом нагрузка на память снижается. Стандартизированная длина ранговой формулы ***n*** является ***мерой детальности*** описания (изучения) составов. Выбор количества учитываемых элементов совершенно аналогичен выбору оптимального увеличения микроскопа, выбору необходимой и достаточной точности измерения.

На заре развития теории информации Р. В. Л. Хартли [3] сделал первый шаг, он предложил в качестве меры сложности (I) использовать логарифм числа (n) разновидностей сигналов в сообщении, т.е. $I = \lg n$. Таким образом, мера сложности I различала системы лишь по длине списка различающихся сигналов (компонентов). При одинаковых по количеству символов в их наборах (алфавитах), сложности систем при любом характере распределений были неразличимы.

Этот недостаток был преодолен К. Шенноном [17], который предложил меру неопределенности результата испытания (***H)*** – информационную энтропию $\boldsymbol{H} = -\Sigma p_i \lg p_i$, где p_i частота (или вероятность) i-го события. При условии p=1, если она имеет один компонент и абсолютно свободна от «примесей», имеем $\boldsymbol{H}_{min} = 0$. Эта система имеет простейший состав. При $p_1=p_2=p_3=\ldots=p_n=1/n$ имеем $\boldsymbol{H}_{Max}=\lg n$, т.е. состав этой системы максимально сложен. Как видим, это соответствует интуитивному представлению о сложности, изложенному выше. Зависимость вкладов отдельных компонентов от p, в интервале от 0 до 1 проходит через максимум при p=0.368, причем положение максимума от основания логарифмов не зависит. Таким образом, величину энтропии определяют в основном «средние» (в районе 30-50%) по содержаниям компоненты. Для приведения энтропии к интервалу 0÷1 реальное значение энтропии делится на ее максимум (т.е. на $\lg n$). При этом величина энтропии становится независимой от основания логарифмов. В случае работы с химическими составами для устранения совпадения символов водорода и энтропии, нормированное ***H*** обозначается через ***En***.

Разнообразие обликов и применений энтропии можно представить по предметным указателям к книгам в период ее наиболее интенсивного осмысления и развития [18-20] (см. *Приложение*). Опираясь на идею Э.Т. Джейнса [21], М. Трайбус в книге [22] изложил новую концепцию термодинамики на базе понятия информационной энтропии

В термодинамике с точностью до постоянного множителя (газовой постоянной R) информационной энтропии соответствует *энтропия смешения*, которая записывается как $\Delta\mathbf{S} = -R\Sigma x_i \lg x_i$, где x_i – мольные доли [18, с. 558].

Энтропия Шеннона как эквивалент сложности состава в геологии, насколько нам известно, впервые была использована C.R. Pelto [23] при создании геологических карт пляжных отложений. О задачах, решаемых с использованием информационной энтропии, писал А.Б. Вистелиус [24]. Как мера неоднородности состава горных пород энтропия использована в работе [25]. В [15] энтропия была использована как мера сложности геохимических систем, соответственно, переменной p был назначен смысл частоты встречаемости атомов i-го сорта. Позднее энтропию было предложено использовать как универсальную характеристику оценки сложности химического, минерального и гранулометрического состава горной породы [26]. Многостороннее современное обсуждение смыслов, вкладываемых в термин энтропия, см. [27].

При расчетах сложности различение «главных» и «примесных» компонентов теряет смысл, в них участвуют все компоненты до n включительно, но вклады малых резко падают с уменьшением p_i (рис. 3). В тоже время, в некоторых отраслях знания, в частности в геологии, наряду с рассмотрением поведения «больших» и «средних» компонентов значимы малые («ценные») компоненты. Разновидности минералов нередко различаются по малым компонентам, редкие элементы, представляющие особый интерес, обычно имеют низкие концентрации, состав минералов весьма чувствителен к условиям кристаллизации, что важно при поиске полезных ископаемых. Но не менее важно следующее.

При количестве компонентов больше двух существует несоответствие между количеством уравнений (энтропия и нормировочное: Σp_i=1) и количеством компонентов в составе. Последнее приводит к тому, что одной величине энтропии при нарастании количества компонентов соответствуют все более сильно различающиеся составы, т.е.

свертка становится чрезмерно «жесткой». (Заметим, что такая ситуация является общей для всех сверток многокомпонентных составов.) Для детальности, принятой при изучении большинства горных пород и минералов, равной 10, имеем размахи содержаний, показанные на рис. 2. Как видим, различия уже по первому компоненту составляют 82–47=35% (!), что можно считать приемлемым очень редко. Это обстоятельство потребовало ввести характеристику, дополняющую энтропию в процедуре свертывания информации о составе и зависящую от малых содержаний так, чтобы она росла при уменьшении содержаний, т. е. могла быть мерой чистоты.

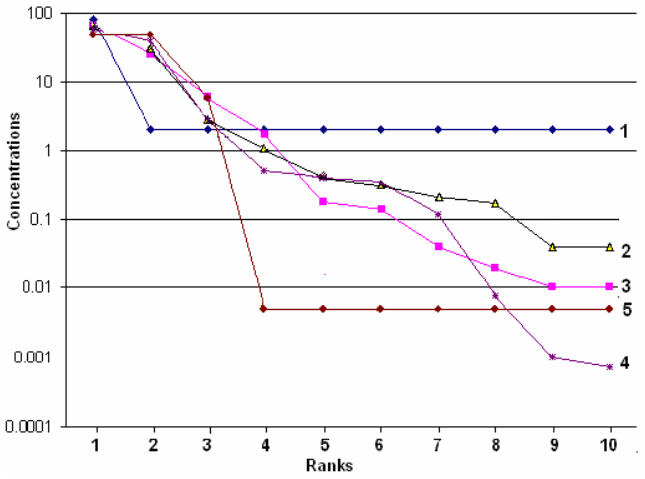

Рис.2. Варианты составов при *En* =0.376.
1- теоретический состав – первый компонент – 82% , остальные по 2%; 2, 3 – горные породы; 4 – минерал «простого» состава; 5 – теоретический состав – первые два компонента по 47%, третий – 5,96%, остальные по 0,005%.

Такая характеристика состава была предложена [1] как средство и некоторого смягчения «жесткости» свертки и, одновременно, как ***мера чистоты*** состава. Величина, получившая название анэнтропия, в простейшем варианте равна: $A^{***} = -\Sigma \lg p_i$, т. е. вкладом в анэнтропию одного элемента является отрицательный логарифм частоты события ($-\lg p$). Неудобство использования A^{***} при построении диаграмм в том, что ее минимум равен $\ln n$, она резко зависит от числа учитываемых компонентов (n), а максимум уходит в положительную бесконечность при устремлении содержаний хотя бы одного элемента к нулю. Для улучшения сравнимости A, получаемых для составов с разными n, удобнее иметь дело с анэнтропией на один элемент системы $A^{**} = A^{***}/n = -(\Sigma \lg p_i)/n$. Далее, для приведения минимума к 0 из A^{**} вычтем $\lg n$, получим: $A^{*} = -(\Sigma \lg p_i)/n - \lg n$. Последнюю особенность – уход максимума A в бесконечность при p=0 – не обязательно устранять, но при желательности приведения всего интервала значений A к 0–1 производится деление A^{*} на состав, который в конкретной области знаний можно принять за максимально чистый [10, 28]. Так, в области составов горных пород, в которых главные 8-10 элементов имеют содержания не меньше 0,01% (0,0001), принят за «аналитически идеально чистый» состав, в котором девять элементов имеют содержания 0,005% (0,00005). Последняя величина примерно равна половине чувствительности «мокрого» метода определения компонента в петрохимическом анализе. Нормирующий делитель в этом случае при использовании десятичных равен 2,871.

Условие стандартизации числа компонентов, сравнительно мягкое для энтропии, при расчетах анэнтропии становится жестким. Заметим, что вклады в анэнтропию в отличие от вкладов в энтропию, однозначно зависят от содержаний, увеличиваясь при снижении содержаний.

При описании процессов получения особо чистых веществ вклады в анэнтропию могут оказаться недостаточно значимыми. Для еще более значительного выделения роли малых компонентов может быть полезной характеристика, названная нами толерантностью $\boldsymbol{T} = \lg[\Sigma(1/p_i)/n]$, где обозначения те же [4]. При необходимости нормирование $\boldsymbol{T}$ производится также, как и в случае анэнтропии.

Между вкладами в энтропию и анэнтропию существует связь: $d(p\lg p)/dp = \lg p + 1$. Вклад в $\boldsymbol{T}$ также равен производной, но уже от вклада в анэнтропию по содержаниям: $d(\lg p)/dp = 1/p$.

Зависимости вкладов от содержаний для всех трех характеристик составов приведены на рис.3.

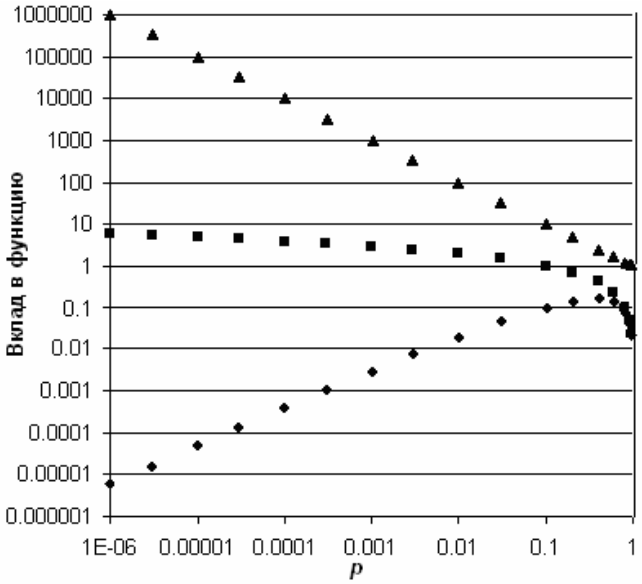

Рис. 3. Вклады в функции: ◆ - энтропии ($-p_i * \lg p_i$), ■ - анэнтропии ($-\lg p_i$); ▲ стерильности ($1/p_i$)

ЭНТРОПИЙНАЯ ДИАГРАММА *НА (EnAn)*

Несовместимость значений максимумов и минимумов энтропии и анэнтропии требует установить границы совместных допустимых значений этих характеристик. Единого уравнения, описывающего границы поля ***EnAn*** для поликомпонентных составов, не существует[1]. Поэтому границы для (их) нормированных значений (***EnAn***) были определены моделированием смешения при двух условиях – максимизации энтропии при минимизации анэнтропии и наоборот. Способ описан ранее [2]. Полученная диаграмма с врезками, на которых показаны составы в виде ранжированных распределений, относящихся к различным точкам диаграммы (ось X – энтропия, ось Y – анэнтропия), показана на рис. 4.

[1] Личное сообщение проф. А.М. Вершика (Математический ин-тут им. Стеклова, СПб)

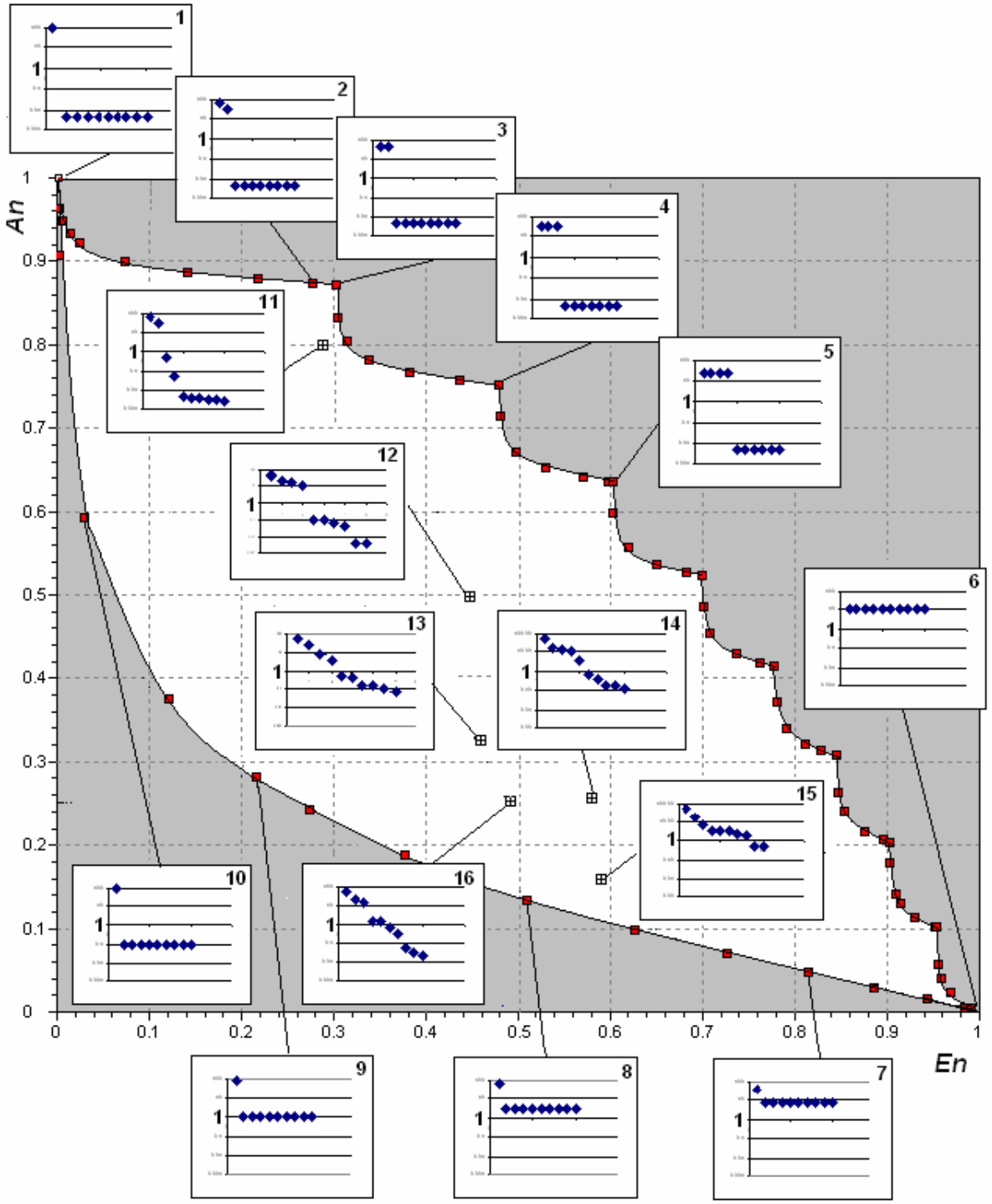

Рис. 4. Виды ранжированных распределений при ***n***=10 на поле сложность-чистота.
1 – состав «аналитически идеально чистого вещества»: главный – 99,955%, остальные 9 по 0,005% (все «простые вещества» – элементы); 2 – стехиометрия 1:2 и 8 примесей по 0,005% (тип H_2O, SiO_2); 3 – стехиометрия 1:1, остальные по 0,005% (тип NaCl); 4 – «чистые» вещества типа NaOH; 5 – «чистые» SOFCl, NaSCN, KCNO; 6 –10 – теоретические составы; 11 – то же, что и 2, но со значимыми примесями, например, ***реальные*** составы воды, или кварца (SiO_2); 12 – состав минерала из группы колумбита-танталита $(Fe,Mn)(Nb,Ta)_2O_6$; 13 – тело человека; 14 – слюда мусковит; 15 – базальт; 16 – мантия Земли. В точке с координатами ***En***=0,7, ***An***= 0,53 находится единственное (!?) «чистое» вещество, в котором пять элементов поровну: PSFClBr.

В каждой точке диаграммы, кроме относящихся к границам поля допустимых значений, показанное распределение – одно из многих возможных, что является упомянутым выше следствием неравенства числа уравнений (3) количеству учитываемых в анализе компонентов (10). Это не мешает построению и интерпретациям конкретных траекторий процессов изменения составов, типичные формы которых показаны ниже.

В представленном на рис. 4 виде поле допустимых совместных значений ***EnAn*** занимает около 43% общей площади диаграммы. Эта площадь растет при увеличении детальности *n* (числа учитываемых компонентов в составе).

ПРОЦЕССЫ, ОТОБРАЖАЕМЫЕ НА ДИАГРАММЕ EnAn

Каждый состав – продукт какого-то процесса изменения состава предыдущего. Поэтому все составы лишь фиксация временного состояния на пути их вечной эволюции. Изменения состава объектов косной природы могут рассматриваться как процессы смешения-разделения и замещения. Последний, как было показано ранее [29], отвечает результату двух идущих последовательно процессов: смешения и разделения.

Известно, но не всегда учитывается, что один и тот же состав может быть получен противоположными путями. Так, состав A:B=99:1 может быть результатом *смешения* двух компонентов в указанной пропорции, а также – удаления большей части B из смеси состава A:B=1:1, т. е. продуктом *разделения*. Точно также состав 1:1 может быть как результатом смешения двух веществ в такой пропорции, так и продуктом разделения любой смеси этих компонентов, т. е. имея состав на диаграмме (или вне ее), нельзя сказать, какой процесс его породил. Более того, последовательность составов также не позволяет прямо определить, какие компоненты изменяются активно, а какие содержания изменяются в результате изменения суммы остальных. Это связано с тем, что состав вещества, выражается в долевых единицах, а не в абсолютных количествах компонентов. Решение задачи существует [30, 31], но это отдельная тема.

В то же время, существуют положения – теоремы, доказанные Ю.В. Шурубором [32], касающиеся изменений энтропии при процессах смешения и разделения. Первая: при смешении энтропия результирующей системы больше ***по крайней мере*** одной из исходных систем. Вторая: при разделении энтропия ***по крайней мере*** одной из результирующих систем меньше исходной. На рис. 5 дана графическая иллюстрация этих положений. Там же отражено статистическое эмпирическое правило, согласно которому между энтропией и анэнтропией в подавляющем большинстве случаев существует обратная зависимость. В достаточно развитых однонаправленных (без внешних возмущений) процессах прямая зависимость наблюдается на относительно коротких отрезках траекторий, что видно и на рис.6.

Смешение составов, использованное при построении внешнего контура поля ***EnAn***, дает возможность построить траектории процессов смешения и разделения для любых пар исходных веществ. На рис.6 показаны траектории изменения *идеализированных* составов типа показанных на рис. под №№ 1, 3-5 и других, содержащих компоненты в равных концентрациях (точки вдоль верхней границы №№ 1-10).

Символика типа 2+3 отвечает кривой, вдоль которой идет смешение системы, состоящей из двух компонентов с соотношением 1:1, с системой, состоящей из трех – с соотношением 1:1:1. При этом все компоненты различны, соотношение между компонентами внутри исходных систем не изменяется при смешении, «примеси» находятся в постоянных содержаниях по 0,005%.

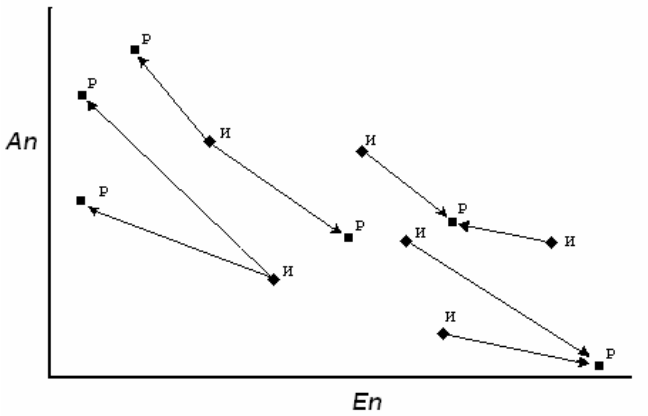

Рис.5 Типичные направления изменений энтропийных характеристик при разделении (два варианта слева) и смешении (то же справа). И – исходные системы, Р - результирующие.

Важно отметить, что достижение точки 1 как конечной *при смешении* проблематично в том смысле, что для этого нужно, чтобы соотношение количеств, например, в исходно двухкомпонентной системе 49,98+49,98 (плюс 8 элементов по 0,005%) стало соотношением 99,955 плюс 9 компонентов по 0,005% в «аналитически идеально чистой», приближающейся к однокомпонентной. Иными словами, цель смешения: перевести один из компонентов в положение «примеси». Для этого нужно, чтобы добавляемого компонента, пусть это будет B, в систему A+B было примерно в 10 000 раз больше чем A, который оставался при смешении в постоянном абсолютном количестве. (Это цена разбавления загрязненных вод или атмосферы.) Тривиально, но весьма значимо, что смешение – это рост размеров результирующей системы, при условии отсутствия ухода части пришедшего или бывшего материала из системы. В то же время, в условиях литостатического давления в земной коре, согласно принципу Ле Шателье, процессы преимущественно идут с сокращением объема.

При разделении процессы идут иначе. Движение, пусть от точки 4 в точку 1 может происходить последовательным уменьшением содержаний компонентов через точки 3, 2, или одновременным удалением трех компонентов. При этом размер остающейся системы снижается. Вообще, траектории составов, показанные на рис.6, в природе не реализуются, для этого они слишком далеки от идеальных, использованных при построении (равенство содержаний компонентов в системе, постоянное количество примесей). Однако характер кривых при реальных процессах, в подавляющем большинстве случаев такой же.

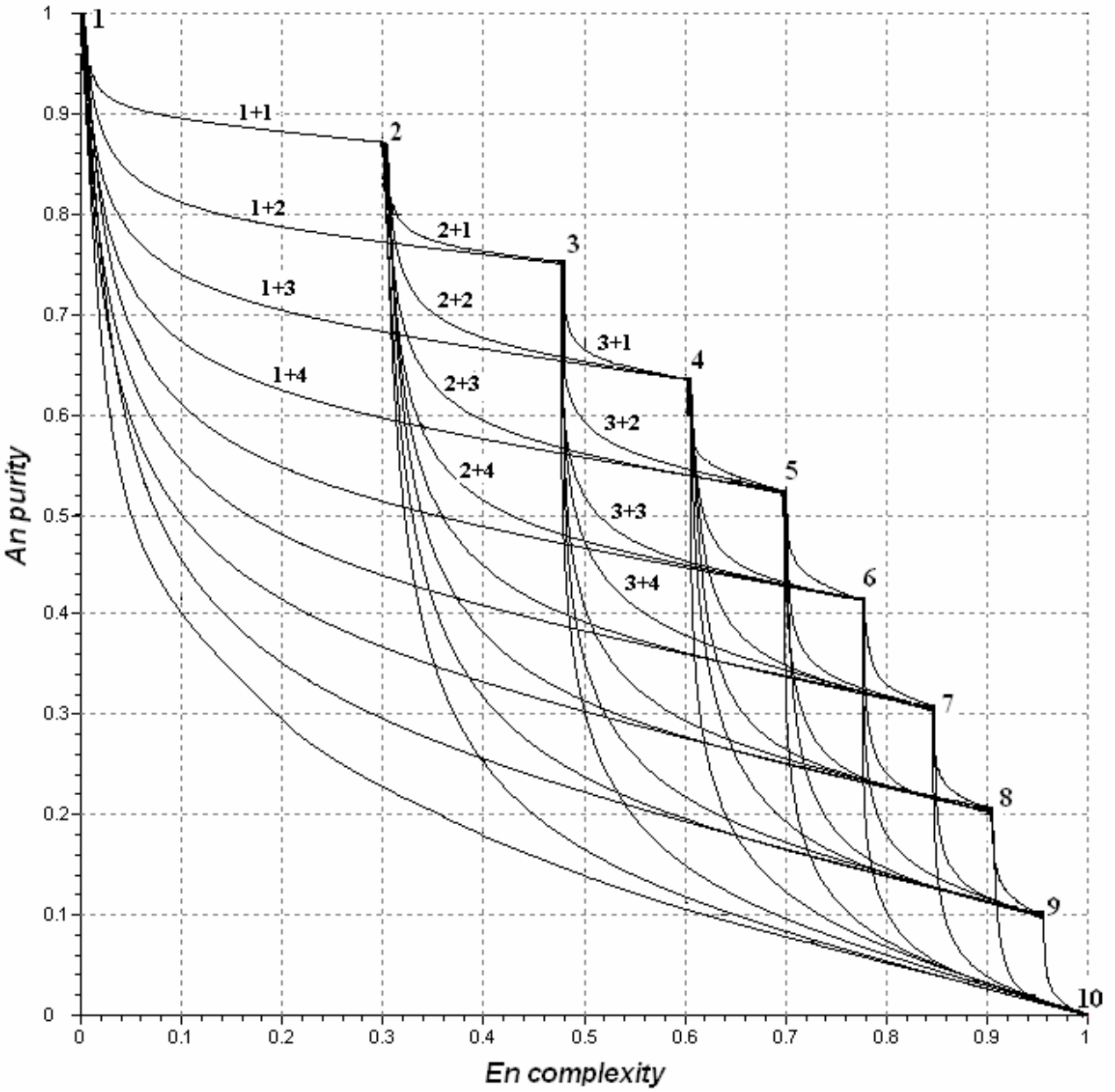

Рис. 6. Траектории смешения и разделения составов, типа находящихся в точках изломов на границах поля диаграммы на рис.2. (*Объяснение в тексте*)

Диаграмма (рис. 7) иллюстрирует правило, о преимущественно обратной пропорциональности между сложностью и чистотой. Как легко видеть, участки с прямо пропорциональными зависимостями занимают небольшую долю от общей длины каждой траектории. Траектории типа показанных на рис.7, вообще, присущи «горячим» контактам, т. е. таким, вдоль которых происходит частичное плавление твердой породы под влиянием тепла жидкой магмы и частичное перемешивание их веществ, а также процессам эволюции составов при кристаллизации. Траектория от долерита до диорита (не показана) и показанная далее – до гранита – весьма близки траекториям, вдоль которых располагаются составы реальных горных пород.

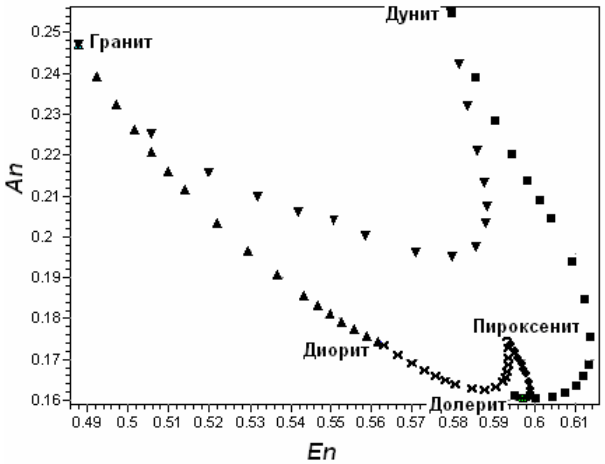

Рис. 7. Траектории теоретического смешения средних составов некоторых горных пород (исходные материалы [33]).

Непрерывные процессы отражаются на диаграмме непрерывными траекториями. Это означает, что ничтожно малому изменению энтропийных характеристик отвечает ничтожно малое изменение исходных данных. Поэтому все возможные разрывы на диаграммах, где отображены реальные процессы, – следствия разрывов непрерывности информации о процессе. Для *данного* процесса, расстояния между точками на диаграмме положительно коррелируют со степенями различий между соответствующими исходными составами.

Траектории *однонаправленных* процессов (в смысле изменения содержаний компонентов с сохранением знака), отображаемых на энтропийной диаграмме имеют следующую особенность. Траектория не может проходить через минимум энтропии и максимум анэнтропии *без начала нового* процесса. Появление такого экстремума на кривой – свидетельство начала нового процесса [29]. Направление роста энтропии (и снижения анэнтропии) может сменяться на противоположное через этап прямой зависимости между энтропией и анэнтропией (рис.7, 8).

Ниже приведены примеры (рис.8) нескольких траекторий реальных процессов из разных областей знания.

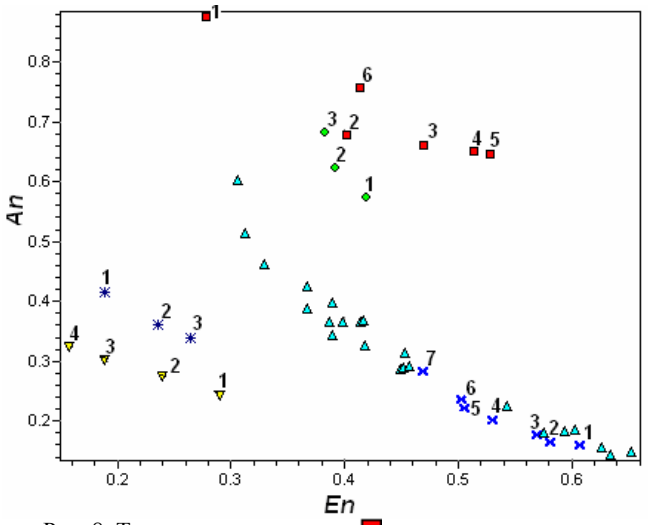

Рис. 8. Траектории процессов: ■ – растворения в воде: (точка 1) азотно-кислого калия (точка 6), 2, 3, 4, 5 – точки растворимости при 0, 20, 40, 60^{0}C; ● – составы минерала шеелита (CaWO4) по мере снижения напряженности процесса кристаллизации (по материалам [34]); ✕ – эволюция составов магматических горных пород от габбро до гранита (по материалам [35]); национальные составы по переписям населения: ▽ – Азербайджана (1939-1979) и ✳ – Эстонии (1959-1979) (по материалам [36]),– △ пески (по материалам [37]), наиболее чистые – олигомиктовые, наиболее сложные – граувакки.

Растворимость веществ обычно отображается с помощью таблиц и графиков зависимости растворимости вещества от температуры или давления. Показанные здесь точки лежат на части кривой растворимости, которая отвечает ветви увеличения сложности состава раствора. Пройдя через максимум, кривая (нет данных) придет в точку сухого вещества KNO3.

Кристаллизация в открытых системах при снижении напряженности процесса [11] идет в направлении снижения сложности и роста чистоты. В том же направлении идет изменение состава песков по мере их освобождения от минералов, неустойчивых к химическим и физическим воздействиям при речном, морском и ветровом переносе. Этому же соответствуют графики кристаллизации шеелита и становления горных пород от габбро до гранита. Установлено правило: минералы как составные части их смесей – горных пород, обычно проще и существенно чище, чем горные породы.

В том же направлении (снижение энтропии – рост анэнтропии) за время 1939-1979 гг. происходило изменение национального состава Азербайджана, что отражало повышение роли титульной национальности в условиях относительно свободного, естественного развития этой республики. Как видно на диаграмме, за время 1959-1979 гг. в Эстонии происходил противоположный по направлению процесс, связанный как с высылкой лиц титульной нации, так и с переселением туда населения других национальностей.

В качестве примера нелинейного процесса рассмотрим изменение распределения доходов населения в России с 1992 по 2007 гг. (см. работу [38]). Для выявления особенностей отображения аналитических данных на диаграмме ***HA*** приведем последовательно: график изменения распределения общего объема денежных доходов населения согласно исходным сведениям в обычном виде (рис. 9) и те же данные на энтропийной диаграмме (рис.10).

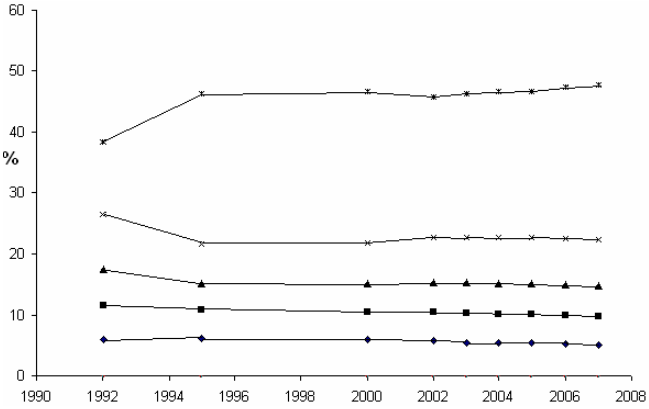

Рис.9. Ход распределения денежных доходов по 5 группам (квантилям) населения России за период 1992-2007гг.

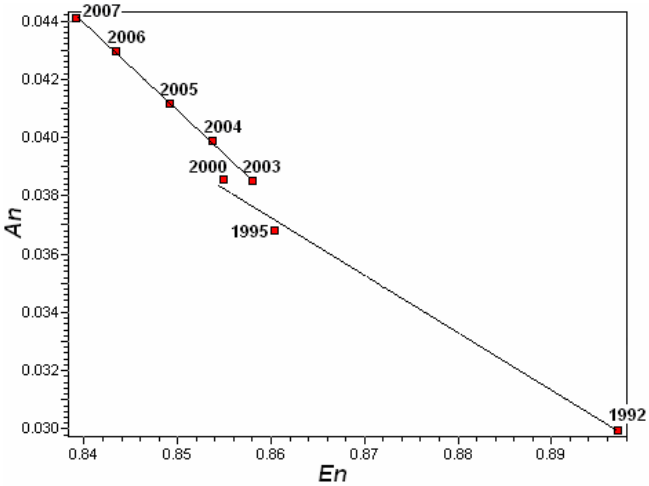

Рис. 10. Кинетика изменения распределения общего объема денежных доходов населения (по материалам [38]) в энтропийных координатах.

Как видим, за время с 1992 по 2007 гг. совокупность точек на энтропийной диаграмме, во-первых, отражает общую тенденцию концентрирования богатств в руках одной группы населения. Во-вторых, данные на диаграмме четко распадаются на две ветви, что свидетельствует о смене в финансовой жизни страны между 2000 и 2003 гг., в-третьих, вторая по времени ветвь начинается со значений энтропии меньших, чем конец первой ветви, в-четвертых, ориентация второй ветви свидетельствует о более резком нарастании разрыва между доходами населения, чем это было ранее.. Это свидетельство резкости изменений – начала процесса с новыми параметрами. В-пятых, новая ветвь находится в области более высоких значения анэнтропии, что свидетельствует о начале процесса, который характеризуется регулярно меньшей долей средств у относительно малооплачиваемого населения. Так, мало отличающиеся по ***En*** данные 2000 и 2004 гг. заметно отличаются по ***An***. Коэффициент Джини [39], использующийся для свертки тех же данных, и имеющий отрицательный коэффициент корреляции с энтропией, равный 0,99, будучи одномерным, подобных заключений сделать не позволяет.

ОБСУЖДЕНИЕ

Информация – отражение действительности, будь она подлинная или искаженная, Само отражение может быть вполне адекватное, преувеличенное или ослабленное. Так или иначе, информация в любом виде – это то, что при достаточно развитых интерпретационных возможностях исследователя может быть использовано в работе.

Энтропийные характеристики, будучи интегрирующими информацию о составе, зависят от всего того, что вошло в их расчеты. Поскольку характеристики непрерывны и зависят от всего включенного в их расчеты, то любое изменение в исходных данных отражается в изменении результатов расчетов, а потому, особенно на первых стадиях освоения, с их помощью удобно производить обзор материалов. Всевозможные невидимые неувязки, уклонения от нормы, неточности группирования, аномальные данные становятся зримыми и могут быть либо исправлены, либо удалены, либо потребуют особого внимания, как значимая новизна.

Известно, что способность к различению прямо связана с объемом уже известной информации о сравниваемых объектах. Этому соответствует то, что возможности различения распределений на диаграмме ***HA*** прямо зависят от количества учтенных компонентов.

Энтропия смешения и ее противоположность – анэнтропия, которую по аналогии и по сути следует назвать энтропией разделения, самими терминами напрямую привязаны к процессу, поэтому направленность изменений сложности и чистоты являются ***генетически значимыми,*** возникают в результате процессов смешения, разделения и замещения.

Энтропия в своем первичном виде была энергетической характеристикой. В этой статье энергия была оставлена в стороне. Но как о самой общей тенденции изменения составов можно сказать, что полярность диаграммы по отношению к смешению-разделению соответствует полярности энергетической. Смесимость и, как следствие, сложность составов растут с повышением температуры и падают с ее снижением. При минимальных температурах все, кроме гелия, – находится в кристаллическом состоянии, т. е. является продуктами разделения с относительно высокой чистотой. При температурах атомного взрыва ситуация противоположна. При достаточно высоких температурах все твердые вещества переходят в жидкость и далее – в газ, что ведет к смешению всего со всем. Заметим, что вообще сведения об «энергии» геологических процессов извлекаются из расчетов особенностей составов сохранившихся систем.

До и после публикации [40] по мере пополнения банка данных нами проводятся наблюдения за публикуемыми аналитическими материалами, касающимися разнообразных геохимических процессов. Общее заключение: в большинстве их, судя по распределению точек анализов на диаграмме ***HA***, изменения химических составов геологических объектов во времени и в пространстве оказываются существенно незакономерными.

Наиболее частыми причинами хаотизации расположения точек, отражающих химико-аналитическую геологическую информацию на энтропийной диаграмме, судя по анализу многочисленных источников, являются следующие:

1. Неполнота геологических данных о процессах формирования объекта.
2. Большая роль случайности при выборе для публикации резко ограниченного объема материалов из массы имеющегося, когда автор не видит материала ***в целом***. Можно думать, что это основная причина затушевывания закономерностей.
3. Многократное наложение последующих процессов на предыдущие, т.е. на то, что считается «первичным» и что обычно наиболее интересует исследователя.
4. Неполнота анализа – игнорирование «неважных» компонентов (в том числе ППП – «потери при прокаливании» – до сих пор обычный «компонент» химических анализов горных пород, за которым скрывается сумма генетически значимых легколетучих компонентов), низкие суммы анализа.
5. Ошибки аналитические. Часто они не замечаются автором, не обнаруживаются и не устраняются.
6. Непроверяемость данных, связанная с неопределенностью места взятия пробы на анализ (особенно, до появления способов точной привязки к местности), труднодоступностью, исчезновением участка, где был взят образец.
7. Разнообразие способов отображения (представления) составов и их изменений в разных областях и даже в разных школах одной области знания.
8. Намеренное искажение данных («политизированная» геология, искажение карт).

Все это затрудняет эвристически значимое выявление общих законов эволюции составов в пределах одной области знания, и, тем более, – в разных.

Эффективным средством выхода за пределы изучения содержаний отдельных компонентов в направлении оценок их целостных распределений является теория информации, и не зря в предисловии к уже довольно давней книге [41] читаем, что эта наука «стала применяться чуть ли не во всех областях научной деятельности человека. В биологии, психоло-

гии, педагогике, истории, эстетике, экономике и ряде других областей…»). Можно выразить уверенность, что использование энтропийной диаграммы помогло бы решению многих задач по обзору и осознанию аналитического материала и, в частности, это способствовало бы более обоснованному выбору материалов для публикаций.

Есть основания считать, что начавшееся после C.R. Pelto [23] временное увлечение энтропией в геологии в 60-х годах прошлого столетия довольно быстро угасло именно из-за неустойчивости, несопоставимости, нечеткости определенности результатов расчетов. Неприятие энтропийных характеристик, как инструмента геологического исследования, проявилось в отсутствии упоминания о нем в весьма полном обзоре [42], когда к этому времени только по методу ***RHA*** уже было опубликовано 16 статей.

При работе по методу ***RHA*** используются разнообразные возможности диаграммы ***EnAn***, представляемые программным комплексом Petros3, составленным С.В. Мошкиным [43].

ЗАКЛЮЧЕНИЕ

Общность отображения результатов процессов, изучаемых в разных областях знания, для узкопрофессиональной работы не представляет интереса. Поэтому сходство «картинок» может вызывать (и вызывает) как раздражение нелепостью самого акта сопоставления у одних, так и любопытство у других. Последнее представляется более продуктивным, так как может побудить к поиску общих, не лежащих на поверхности, причин у разных явлений. Действительно, можно сформулировать как задачу, выявление общих механизмов разделения систем различной природы. Какова природа воздействия общего энергетического фона на эффективность разделения? Как влияет на эффективность разделения степень различия элементов системы? Формальное, графическое сходство – это намеки на возможность выявления содержательной общности, на сходства причин некоей высшей энергетики процессов, на существование более глубоких законов по сравнению с уровнем конкретных научных направлений.

СПИСОК ЛИТЕРАТУРЫ

1. Петров Т. Г. Обоснование варианта общей классификации геохимических систем // Вестник ЛГУ. – 1971. – № 18. – С. 30 – 38.
2. Петров Т. Г. Информационный язык для описания составов многокомпонентных объектов // НТИ. Сер. 2. – 2001. – № 3. – С. 8 – 18.
3. Hartley R. V. L. Transmission of Information // BSTJ 7. – 1928. – № 3. – P. 535 – 563.
4. Петров Т. Г. Рангово-энтропийный подход к описанию составов геологических объектов и их изменений (на примере геологической ценологии) // Общая и прикладная ценология. – 2007. – № 5. – С. 27 – 33.
5. Спенсер Г. Основания биологии. – СПб., 1870. – Т. 1–2.
6. Синтетическая философия Герберта Спенсера. В изложении Г. Коллинза. – Киев : Ника-Центр : Вист-Са, 1997. – 512 с.
7. Петров Т. Г. Метод RHA как решение проблемы систематизации аналитических данных о вещественном составе геологических объектов // Отечественная геология. – 2008. – № 4. – С. 98 – 105.
8. Петров Т. Г. Иерархическая периодическая система химических составов и ее связь с периодической системой элементов // Вестник С.-Петерб. ун-та. Сер. 7. – 2009. – Вып. 2. – С. 21 – 28.
9. Великославинский Д. А., Елисеев Э. Н., Кратц К. О. Вариационный анализ магматических систем. – Л. : Наука, 1984. – 280 с.
10. Петров Т. Г., Фарафонова О. И. Информационно-компонентный анализ. Метод RHA : учеб. пособие. – СПб., 2005. – 168 с.
11. Петров Т. Г., Фарафонова О. И., Соколов П. Б. Информационно-энтропийные характеристики состава минералов и горных пород как отражение напряженности процесса кристаллизации // Записки ВМО. – 2003. – № 2. – С. 33 – 40.
12. Девятых Г. Г., Карпов Ю. А., Ковалев И. Д., Максимов Г. А., Осипова Л. И. Примесный состав образцов выставки-коллекции веществ особой чистоты. III. Простые твердые вещества элементов 3-го и 4-го периодов системы Д. И. Менделеева // Высокочистые вещества. – 1991. – № 2. – С. 22 – 32.
13. Петров Т. Г. О невозможности определения последовательности кристаллизации по индивидуальным характеристикам минералов // Записки ВМО. – 1977. – Ч. 106, № 4. – С. 499 – 502.
14. Петров Т. Г. Теория информации и проблемы кристаллогенезиса // Процессы роста кристаллов и пленок полупроводников: сб. статей / под ред. Л. Н. Александрова, Л. А. Борисовой. – Новосибирск, 1970. – С. 61 – 72.
15. Петров Т. Г. О мере сложности геохимических систем с позиций теории информации // Доклады АН СССР. – 1970. – Т. 191, № 4. – С. 1094 – 1096.
16. Петров Т. Г., Краснова Н. И. R-словарь-каталог химических составов минералов. – СПб. : Наука, 2010. – 150 с.
17. Shannon C. E. A Mathematical Theory of Communication // Bell System Technical Journal. – 1948. – Vol. 27. – P. 379 – 423, 623 – 656.
18. Мелвин-Хьюз Э. А. Физическая химия : в 2 т. – М. : ИЛ, 1962. – 1148 с.
19. Фано Р. Передача информации. Статистическая теория связи. – М. : Мир, 1965. – 440 с.
20. Яглом А. М., Яглом И. М. Вероятность и информация. – М. : Наука, 1973. – 512 с.
21. Jaynes E. T. Information theory and statistical mechanics // Phys. Rev. – 1957. – Vol. 106, № 4. – P. 620 – 630; Vol. 108, № 2. – P. 171 – 190.
22. Трайбус М. Термостатика и термодинамика. – М. : Энергия, 1970. – 504 с.
23. Pelto C. R. Mapping of multicomponent systems // J. Geol. – 1954. – Vol. 62, № 5. – P. 501 – 511.
24. Вистелиус А. Б. Задачи геохимии и информационные меры // Сов. геология. – 1964. – № 12. – С. 5 – 26.
25. Буряковский Л. А. Энтропия как мера неоднородности горных пород // Сов. геология. – 1968. – № 3. – С. 135 – 138.

26. Логвиненко Н. В. Петрография осадочных пород. – М. : Высшая школа, 1984.
27. Бекман И. Н. Информатика. – М. : Рим, 2009.
28. Гордиенко В. В., Петров Т. Г. Исследование редкометальных пегматитов с использованием языка RHA // Записки ВМО. – 1981. – Ч. 110, вып. 5. – С. 46 – 558.
29. Петров Т. Г. Проблема разделения и смешения в неорганических системах // Геология : сб. статей / под ред. В. Т. Трофимова. – М. : МГУ, 1995. – Т. 2. – С. 181 – 186.
30. Gresens R. L. Composition-volume relationships of metasomatism // Chem. Geol. – 1967. – Vol. 2, №1. – P. 47 – 65.
31. Петров Т. Г. Способ равных кратностей для выявления пассивных и характера поведения активных компонентов при геохимических процессах // Записки ВМО. – 1983. – Ч. 112, вып. 6. – С. 641 – 650.
32. Шурубор Ю. В. Об одном свойстве меры сложности геохимических систем // Доклады АН СССР. – 1972. – Т. 205. – С. 453 – 456.
33. Le Maitre R. W. The chemical variability of some common igneous rocks // J. Petrology. – 1976. – Vol. 17. – P. 589 – 637.
34. Спиридонов Э. М., Бакшеев И. А., Середкин М. В. и др. Гумбеиты Урала и сопряженная рудная минерализация // Геология рудных месторождений. – 1998. – Т. 40, № 2. – С. 174 – 190.
35. Соболев А. П. Мезозойские гранитоиды Северо-Востока СССР и проблемы их рудоносности. – М. : Наука, 1989. – 250 с.
36. Демографический энциклопедический словарь. – М. : Сов. энциклопедия, 1985. – 608 с.
37. Шванов В. Н. Петрография песчаных пород. – Л. : Недра, 1987. – 282 с.
38. Россия в цифрах. 2008. – М., 2008. – 512 с.
39. Jini C. Sl Fattori demografici dell'evolucione delle nazioni. – Torino, 1912.
40. Петров Т. Г. Состояние информации о вещественном составе геологических объектов // Сов. геология. – 1976. – № 7. – С. 108 – 118.
41. Мазур М. Качественная теория информации. – М. : Мир, 1974. – 240 с.
42. Ефремова С. В., Стафеев К. Г. Петрохимические методы исследования горных пород. – М. : Недра, 1985. – 512 с.
43. Мошкин С. В., Шелемотов А. С., Петров Т. Г., Краснова Н. И. «PETROS-2» – программный комплекс для обработки петрохимических данных // Геохимия магматических пород / КНЦ РАН. – Апатиты, 2003. – С. 115 – 116.

Материал поступил в редакцию 24.05.11.

Сведения об авторе

ПЕТРОВ Томас Георгиевич – профессор, главный научный сотрудник геологического факультета Санкт-Петербургского государственного университета
E-mail: tomas_petrov@rambler.ru

ПРИЛОЖЕНИЕ

Фрагменты предметных указателей книг, в которых использован термин «энтропия»

Э. А. Мелвин-Хьюз «Физическая химия»	**Р. Фано «Передача информации. Статистическая теория связи»**	**А. М. Яглом, И. М. Яглом «Вероятность и информация»**
Энтропия 34, 235 активации 1106 двумерного газа 837 двухатомных газов 385 жидкостей 243 ионов 769, 771 испарения 247 металлов 578 одноатомных газов 335 плавления 246, 248 поверхностная 815 связь с суммой по состояниям 310 с давлением пара 246 смешения 568, 691, 728 сольватации 774 твердых тел 242 трехатомных газов 415, 417 упорядочения и разупорядочения 606 электронов Всего: 17	Энтропия 59 — ансамбля символов 60, 143 — — событий 123 —условная 105, 123 —сообщений 81, 124 —условная 134 — двоичного ансамбля 61 — источника сообщений 388 — марковского источника 134 — на букву 134 —непрерывного ансамбля 73 — последовательности событий 144 — произведения ансамблей сообщений 81, 124 — — событий 108 — стационарного источника средняя 106, 108 — условная 107 — термодинамическая 59 — условная 61 — шума 65 Всего: 18	Энтропия 7, 72, 73, 79, 105, 121, 128 — безусловная 101 — комбинаторная 272 — опыта 10, 72 — остаточная 167 — предельная 263 — распределения вероятностей 10 — сложного опыта 89 — средняя условная 91 — удельная 217, 228 — условная 91, 241, 248 Всего: 10

Учреждение Российской академии наук

ВСЕРОССИЙСКИЙ ИНСТИТУТ НАУЧНОЙ И ТЕХНИЧЕСКОЙ ИНФОРМАЦИИ РОССИЙСКОЙ АКАДЕМИИ НАУК (ВИНИТИ РАН)

предлагает научным работникам, аспирантам и другим специалистам в области естественных, точных и технических наук, желающим быстро и эффективно опубликовать результаты своей научной и научно-производственной деятельности, использовать способ публикации своих работ через *систему депонирования.*

«Депонирование (передача на хранение) – особый метод публикации научных работ (отдельных статей, обзоров, монографий, сборников научных трудов, материалов научных мероприятий – конференций, симпозиумов, съездов, семинаров) узкоспециального профиля, разрешенных в установленном порядке к открытому опубликованию, которые нецелесообразно издавать полиграфическим способом печати, а также работ широкого профиля, срочная информация о которых необходима для утверждения их приоритета. Депонирование предусматривает прием, учет, регистрацию, хранение научных работ и обязательное размещение информации о них в специальных информационных изданиях».

Подготовка и передача на депонирование научных работ происходит в соответствии с «Инструкцией о порядке депонирования научных работ по естественным, техническим, социальным и гуманитарным наукам» (М., 2003).

Результатом депонирования является публикация информации о депонированных научных работах в информационных изданиях ВИНИТИ РАН – реферативном журнале и библиографическом указателе «Депонированные научные работы».

В соответствии с “Положением о порядке присуждения ученых степеней”, утв. Постановлением Правительства Российской Федерации № 74 от 30.01.2002 г., депонированные научные работы признаны публикациями, учитывающимися при защите кандидатских и докторских диссертаций.

Подать научную работу на депонирование можно, обратившись в Отдел депонирования ВИНИТИ РАН по адресу:

125190, Москва, ул. Усиевича, 20.
ВИНИТИ РАН, Отдел депонирования научных работ.
Тел.: 8 (499) 155-43-28, Факс: 8 (499) 943-00-60.
E-mail: dep@viniti.ru

С инструкцией о порядке депонирования можно ознакомиться на сайте ВИНИТИ РАН: **http://www.viniti.ru**