Описание программной системы DMiner

1 Цель создания системы

Программная система DMiner разрабатывалась как набор инструментов для анализа реляционных баз данных, создаваемых в КарНЦ РАН. Целью ее создания является исследование возможностей и разработка методики практического применения методов Data Mining для решения задач анализа данных как в технических, так и в гуманитарных областях науки. Одной из областей применения системы является анализ реляционной базы данных TORIS, содержащей информацию о топонимах Европейского Севера России. Другим направлением, в котором была применена данная система, является Web Mining – исследование статистики посещений Web-сайтов методами Data Mining, в частности, для анализа посещений Web-серверов КарНЦ РАН.

Задача разработки системы DMiner формулируется следующим образом. Требуется реализовать программную систему, позволяющую выполнять обнаружение значимых множеств различных видов, а также генерировать из них наборы правил, выражающих связи между элементами записей выборки анализируемой части исследуемой базы данных и удовлетворяющих заданным значениям уверенности и поддержки.

В состав системы DMiner должен входить набор инструментов, позволяющих проводить отбор и предварительную обработку исходных данных для последующего анализа. В частности, при работе с реляционной базой данных в качестве исходного материала, пользователь должен иметь возможность определения тех компонент записей, которые должны присутствовать в исследуемой выборке. Кроме того, в связи с тем, что систему DMiner предполагается использовать для проведения Web Mining, в ней должны присутствовать специализированные инструменты для определения структуры анализируемого Web-сайта и разбора регистрационного файла (log-файла) сервера с занесением хранящейся в нем информации о запросах в реляционную базу данных.

Программная система в качестве входных подлежащих анализу данных должна получать определяемую пользователем выборку из исследуемой базы данных. Полученные данные подвергаются обработке с учетом задаваемых пользователем параметров поддержки и уверенности. Результаты анализа должны сохраняться для последующего их просмотра.

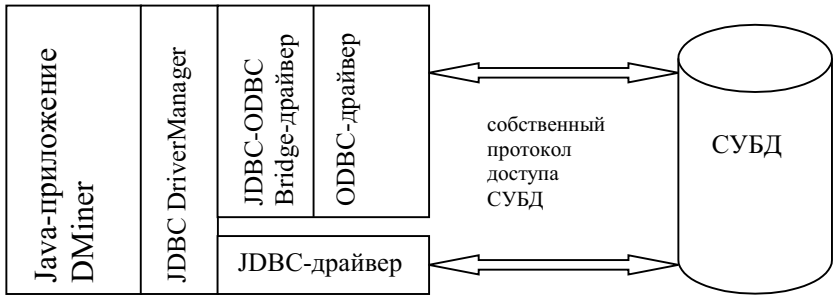

Рис. 1: Архитектура взаимодействия DMiner с СУБД

Система должна предоставлять определенный сервис при просмотре результатов анализа. Она должна позволять пользователю производить нужные ему выборки из найденных значимых множеств или правил по критериям соответствия степеней поддержки и уверенности, а также по вхождению задаваемых элементов в значимые множества или правила.

2 Архитектура системы DMiner

Система DMiner реализована с использованием пакета разработчика на языке Java – JDK 1.3.1 (SDK 2) и представляет собой Java-приложение, выполняющее роль клиента по отношению к системам управления участвующими в работе DMiner базами данных. Работа системы основана на построении и обработке результатов SQL запросов к базам данных: анализируемой, из которой получает исходные данные, и рабочей, в которую помещает промежуточную оперативную информацию и результаты выполнения алгоритма. Доступ из Java-приложения к базам данных реализован с использованием интерфейса JDBC (Java Database Connectivity). Он может осуществляться напрямую к СУБД посредством соответствующих JDBC-драйверов, а также (как альтернативный вариант) путем применения схемы взаимодействия интерфейсов JDBC и ODBC (Open Database Connectivity) с использованием JDBC-ODBC-Bridge и ODBC-драйверов (см. рис. 1). Система DMiner имеет оконный пользовательский интерфейс, для реализации которого были использованы наборы классов для работы с окнами библиотеки Abstraction Window Toolkit (AWT) [1].

DMiner имеет модульную структуру. Все модули системы работают независимо друг от друга, получая входные данные из базы данных и на основе пользовательских настроек. Результаты работы либо выводятся на экран, либо помещаются в базу данных. Запуск модулей осуществляется из главного меню системы.

Основные составляющие систему модули разделяются на три группы по выполняемым функциям:

1. подготовка данных – отбор и преобразование исходных данных в набор

числовых массивов – формат, с которым работает система;

2. выполнение процедур Data Minig обнаружения логических закономерностей: значимых множеств и правил;

3. интерактивный доступ к результатам анализа и их наглядное представление.

В состав первой группы входят:

- набор модулей, предназначенный для загрузки и преобразования данных из исходной реляционной базы данных, с интерактивной настройкой параметров загрузки;

- модули, предназначенные для загрузки и преобразования данных из исходной реляционной базы данных на основе параметров, считываемых из текстового файла.

Вторую группу составляют:

- модули, реализующие пользовательский интерфейс настройки параметров выполнения процедур обнаружения значимых множеств и генерации правил;

- модуль, реализующий процедуру обнаружения значимых множеств по алгоритму Apriory;

- модуль, реализующий процедуру обнаружения значимых множеств по алгоритму Prefix-Span;

- модуль, реализующий процедуру генерации правил.

К третьей группе относятся модули, реализующие пользовательский интерфейс для организации запросов к таблицам рабочей базы данных, хранящим результаты анализа, а также выполняющие преобразование в понятный пользователю вид и отображение результатов запросов.

Кроме вышеописанного базового набора модулей система может дополняться компонентами, связанными со спецификой решаемых задач. В частности, в связи с решением задач Web Mining, в систему был введен соответствующий набор модулей:

- сканер Web-сайта, предназначенный для определения и занесения в базу данных логической структуры сайта;

- модуль, выполняющий разбор log-файла сервера с занесением в базу данных хранящейся в нем информации о запросах;

- модуль, предназначенный для графической визуализации регулярных эпизодов с учетом структуры сайта.

3 Внутреннее представление данных в системе

Организация внутреннего представления данных в системе DMiner основана на формальной модели исходных данных для задач поиска логических закономерностей. Для реализации данной модели были созданы специальные структуры данных, описываемые ниже в терминах объектно-ориентированного программирования.

Базовым объектом в данном представлении является объект $LongSet$, представляющий собой динамический массив длинных целых (long) чисел с набором методов для добавления и удаления элементов, преобразования экземпляра объекта в строку чисел и обратно, сравнения экземпляров объектов. Данный объект используется в качестве основы для представления в закодированном виде совокупности элементов записей. В первоначальном виде записи, составляющие выборку значений определенных полей из базы данных представляют собой объемный набор строк и текстов, и сохранение их в таком виде сильно замедлило бы их дальнейшую обработку – анализ, основой которого является многократно выполняемое сравнение элементов записей. Поэтому на начальном этапе работы системы выполняется числовое кодирование всех элементов записей исходной выборки. Каждое значение поля из набора представляется в виде: “атрибут = значение” и заносится в таблицу кодирования, в которой ему сопоставляется уникальный числовой идентификатор. Таблица кодирования соответствует множеству всех возможных свойств Θ в модели исходных данных. В дальнейшем система работает с числовыми массивами идентификаторов, соответствующими совокупностям свойств объектов, представляемых наборами значений полей записей исходной выборки.

В формальной модели для множеств введено четыре различных типа, определяющие набор выполняемых на них отношений и операций. В реализации системы для задания типа множества и набора выполняемых отношений и операций создан объект $MultiSet$, свойством которого является тип множества, а методы, соответствующие отношениям и операциям над передаваемыми в качестве параметров объектами $LongSet$ выбираются в зависимости от значения типа множества.

Таким образом, пара экземпляров объектов $LongSet$ и $MultiSet$ определяют множество свойств заданного типа. Наборы таких множеств представляют исходный анализируемый и все промежуточные и результирующие наборы данных, строящиеся в процессе работы системы при выполнении процедур Data Mining. Основная часть этих наборов хранится в рабочей базе данных в виде строк чисел, разделенных пробелами. Интерфейс доступа к ним осуществлен путем создания специальных объектов, методы которых реализуют последовательности SQL-запросов, необходимых для инициализации набора, добавления и удаления множеств свойств и обращения к отдельным множествам свойств в наборе. Некоторые наборы данных предположительно небольшого объема и создающиеся на короткое время размещаются в оперативной памяти в целях ускорения работы системы. Для них также реализованы объекты с аналогичным набором методов. К таким

“небольшим” наборам данных относится, например, набор $Candidate_f$, используемый в алгоритме Apriory для хранения генерируемых в данном проходе расширений одного граничного множества в пределах одного множества свойств, являющегося элементом исходного набора данных.

4 Схема работы системы DMiner

4.1 Подготовка данных

На начальном этапе работы с системой необходимо провести отбор анализируемой информации из исходной базы данных. Выполнение отбора анализируемых данных в рабочую базу обеспечивает значительное ускорение выполнения программы за счет уменьшения количества затрачиваемых операций и снижения объема использования системных ресурсов при обращении к СУБД на следующих этапах. Исходная база данных может иметь сложную структуру, состоять из большого количества связанных между собой таблиц, что обеспечивает оптимальное размещение хранимой информации, но затрудняет и замедляет обращение к данным, которое производится путем построения и выполнения запросов, часто вложенных. Работа с исходной базой данных может происходить удаленно, что будет занимать дополнительные ресурсы. Поэтому эффективнее из исходной базы данных сделать выборку всей подлежащей анализу информации и поместить ее в рабочую базу данных в одну таблицу.

До начала выполнения отбора данных пользователем должна быть создана пустая рабочая база данных. Теоретически, рабочая база данных, как и исходная, может использоваться программой удаленно, но, если позволяют ресурсы платформы, на которой выполняется программа, рекомендуется расположить ее на той же платформе, чтобы не затрачивать дополнительные ресурсы и время на передачу данных по сети.

Для выполнения отбора анализируемых данных из исходной базы данных необходимо определить и задать системе условия работы. При интерактивной настройке системы с использованием графического интерфейса настройка осуществляется в несколько шагов. Прежде всего, должны быть указаны параметры для связи с СУБД исследуемой базы данных: путь к базе данных, имя пользователя, пароль, используемый драйвер. Система пытается связаться с базой данных, и в случае успешной связи отображает на экране ее структуру: весь набор таблиц со списками полей.

Далее пользователь должен определить для системы задачу загрузки данных (см. рис. 2 и 3). Среди всех списков полей таблиц базы данных он должен указать активные поля – те поля, которые будут учитываться при анализе. В случае, если активные поля располагаются в разных таблицах базы данных, пользователю необходимо указать связи между таблицами, содержащими активные поля. Таблица, в которой есть хотя бы одно активное поле, или хотя бы одно из ее полей присутствует хотя бы в одной из определенных связей, считается активной.

После указания всех активных полей и связей необходимо определить

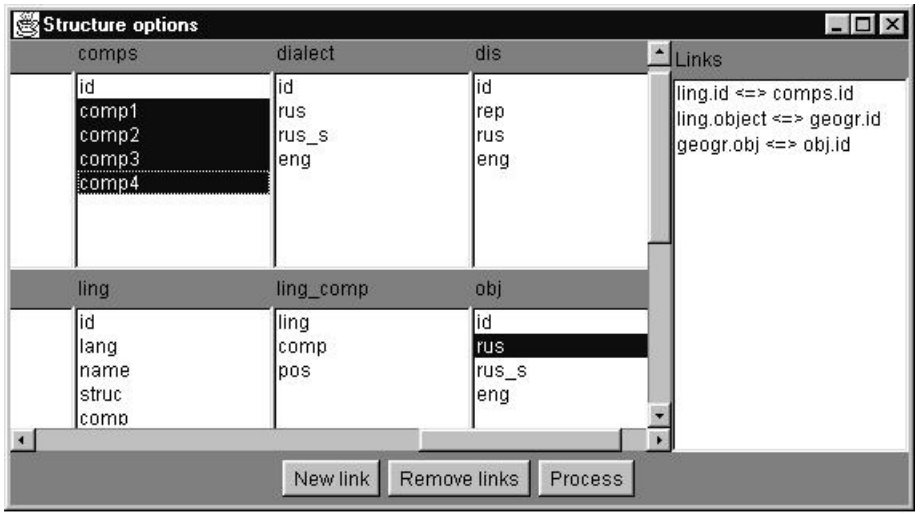

Рис. 2: Интерактивная настройка загрузки данных. Шаг 1.

ключевое поле. В качестве него выбирается одно из полей одной из активных таблиц в базе данных. Ключевое поле определяет принадлежность свойств, определяемых значениями полей записи, к одному объекту. Несколько записей с одним и тем же значением ключевого поля определяют свойства одного и того же описываемого объекта. Пример:

Id	Топоним	Компонент	Семант. формула компонента
13	Варойярви	1	A421 - Птицы
13	Варойярви	2	A211 - Воды, моря и реки

В данном случае топоним выступает в роли описываемого объекта. Каждому его компоненту (их может быть произвольное количество) соответствует запись в таблице приведенной структуры. В качестве ключевого может быть выбрано поле “id” или “Топоним”. Решение вопроса, какое именно поле выбрать, будет в данном случае влиять только на скорость работы. По соображениям оптимальности в качестве ключевого рекомендуется выбирать поле числового типа (а не символьного, текстового и других, для которых операции сравнения сильно замедляют работу используемой СУБД).

На этом же шаге определяются такие параметры как кодировки исходной и рабочей баз данных, имена создаваемых в рабочей базе данных кодовой таблицы и таблицы для размещения записей анализируемой выборки, а также опция булевского типа, определяющая необходимость приведения исходных данных к одному регистру написания.

После определения структуры базы данных и постановки задачи программа проводит частичный анализ корректности определения связей. Схема связей между таблицами может быть представлена в виде ориентированного графа, в котором таблицы являются вершинами, а связи между ними – дугами. С помощью рекурсивного алгоритма поиска путей в графе проверяется наличие путей от таблицы, содержащей ключевое поле до всех остальных

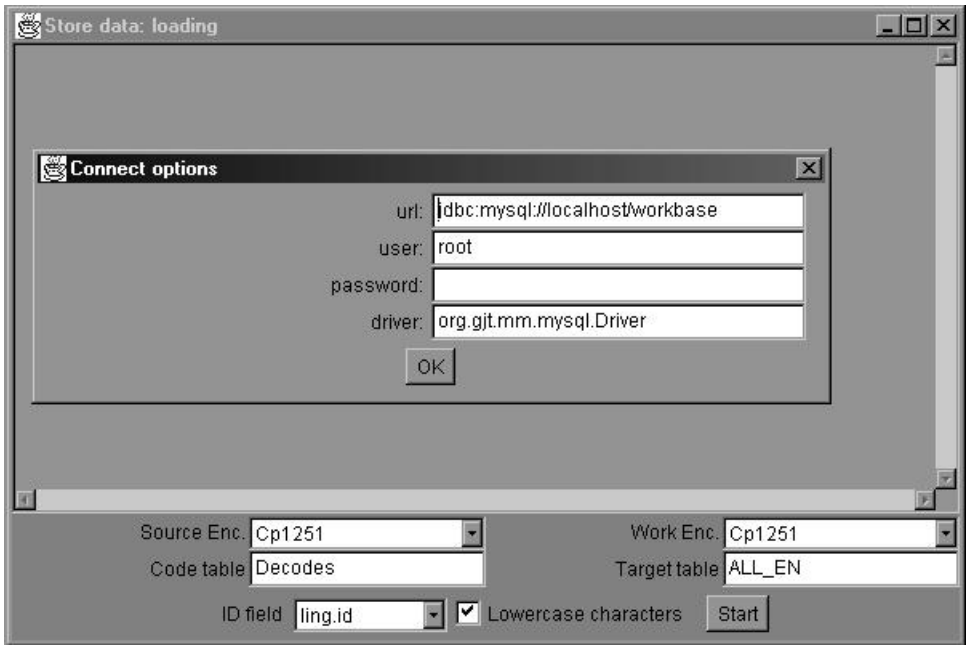

Рис. 3: Интерактивная настройка загрузки данных. Шаг 2.

активных таблиц.

Если связи между таблицами определены корректно, то настроенная таким образом программная система считается готовой к выполнению начального этапа работы – отбора из исследуемой базы данных всего подлежащего анализу объема информации. После ввода параметров связи с СУБД рабочей базы данных начинается процесс загрузки, кодирования и размещения в рабочей базе данных записей выборки анализируемых полей исходной базы данных.

Для корректности осуществления выборки и последующего анализа исследуемые данные должны удовлетворять следующим ограничениям:

- структурное ограничение – в исследуемой части базы данных для каждого активного поля в графе связей должен быть определен единственный путь между содержащей его таблицей и таблицей, содержащей ключевое поле;

- семантическое ограничение – все рассматриваемые компоненты записей выборки считаются независимыми внутри записи, поэтому между ними не должно быть никакой явной иерархии, группировки.

Первое ограничение связано с тем, что при загрузке данных система автоматически строит SQL-запросы к исходной базе данных, выбирая необходимые связи между таблицами из набора связей, определенных пользователем при настройке. В случае возникновения неоднозначности выбора связи корректность загрузки данных может быть нарушена. Второе ограничение вводится для того, чтобы избежать обработки непосредственно зависящих друг от друга значений и получения очевидных зависимостей со степенью уверенности

равной 1, но не представляющих интереса (например, если в некоторой базе данных включить в активные такие поля как “время года” и “месяц”, то могут быть получены правила вида: “февраль => зима”).

Кроме интерактивной настройки параметров, определяющих условия задачи загрузки исходных данных, в системе DMiner реализована возможность загрузки этих параметров из текстового файла специального формата, определяющегося следующими правилами:

- строки, помеченные в начале символом #, считаются комментариями и игнорируются;
- первая строка содержит путь к исходной базе данных и имя соответствующего JDBC-драйвера;
- вторая строка определяет входное имя пользователя и пароль доступа к исходой базе данных;
- третья и четвертая строки содержат аналогичную информацию для рабочей базы данных;
- пятая строка определяет кодировки для исходной и рабочей баз данных;
- в шестой строке содержится имя таблицы для размещения записей анализируемой выборки и имя кодовой таблицы;
- седьмая строка определяет ключевое поле;
- в восьмой строке определяется необходимость приведения данных к одному регистру написания;
- в девятой строке определяется число активных полей;
- оставшиеся строки определяют активные поля в формате: свойство, соответствующее данному полю = часть SQL SELECT-запроса, определяющая получение значения данного свойства из исходной базы данных.

Ниже приводится пример входного файла для загрузки записей, содержащих компоненты топонимов, из исходной базы данных TORIS.

```
#Source database host and JDBC driver
jdbc:mysql://localhost/toris org.gjt.mm.mysql.Driver
#Source database login and password
dminer ******
#Target database host and JDBC driver
jdbc:mysql://localhost/workbase org.gjt.mm.mysql.Driver
#Target database login and password
dminer ******
#Source and Target database encoding
ISO8859-5 Cp1251
```

```
#Table for entries, being analized, and code table
ALL_EN Decodes
#Key table.field in source database
ling.id
#Lowercase characters (1 - yes, 0 - no)
1
#Amount of fields, being analized
5
#Fields, being analized, and how to recieve them, having certain
#entry from key table
Объект = obj.rus FROM ling, geogr, obj WHERE (geogr.obj = obj.id) AND (geogr.id =
ling.object) AND (geogr.dis = 61)
Компонент = comp.name FROM ling, comp, geogr WHERE (comp.id = ling.comp1) AND
(geogr.id = ling.object) AND (geogr.dis = 61)
Компонент = comp.name FROM ling, comp, geogr WHERE (comp.id = ling.comp2) AND
(geogr.id = ling.object) AND (geogr.dis = 61)
Компонент = comp.name FROM ling, comp, geogr WHERE (comp.id = ling.comp3) AND
(geogr.id = ling.object) AND (geogr.dis = 61)
Компонент = comp.name FROM ling, comp, geogr WHERE (comp.id = ling.comp4) AND
(geogr.id = ling.object) AND (geogr.dis = 61)
```

Вариант загрузки исходных данных на основе параметров, заданных в текстовом файле, является более гибким в использовании. Он позволяет снять структурное ограничение на базу данных, так как в данном случае пользователь самостоятельно указывает отдельно для каждого активного поля связи между таблицами, необходимые для получения его значения. Однако, этот вариант предъявляет более высокие требования к квалификации пользователя: знание языка запросов SQL и точной структуры анализируемой базы данных.

Сразу после определения параметров начинается процесс загрузки данных. Система посылает СУБД исходной базы данных SQL-запрос на получение выборки всех имеющихся значений ключевого поля. Для каждого из полученных значений ключевого поля выполняются запросы на получение всех соответствующих данному объекту значений активных полей. Таким образом из исходной базы данных последовательно извлекаются данные по каждому из объектов, определяемых значениями выбранного ключевого поля. При помещении в рабочую базу данных записи анализируемой выборки преобразуются в вид числовых массивов, более удобных для обработки, по сравнению с их исходным текстовым представлением.

4.2 Анализ данных

Процедуры, реализующие в системе DMiner алгоритмы Data Mining поиска значимых множеств и генерации правил, применяются для анализа данных, отобранных и загруженных в рабочую базу на этапе подготовки данных.

4.2.1 Поиск значимых множеств

Для решения задачи поиска значимых множеств пользователь должен определить для системы анализируемые данные и параметры анализа. При этом должны быть указаны:

- имя таблицы, содержащей записи исходной выборки в закодированном виде;
- имя таблицы, содержащей набор всех возможных элементов, составляющих записи выборки (обычно это таблица кодирования);
- имя таблицы, которая должна быть создана системой автоматически для помещения в нее искомых значимых множеств;
- пороговое значение поддержки для искомых значимых множеств (от 0 до 1);
- тип множества, соответствующий записям исходной выборки;
- алгоритм поиска: Prefix-Span или Apriory – оба алгоритма дают совершенно одинаковые результаты и различаются только скоростью работы;
- опция булевского типа, определяющая, требуется ли приведение анализируемых записей в соответствие типу выбранного множества;
- параметры для связи с СУБД рабочей базы данных.

Процесс выполнения алгоритмов поиска значимых множеств обычно занимает достаточно много времени; в ходе него система отображает долю выполненной работы в процентах от ее общего объема. Результатом поиска является набор найденных значимых множеств, который помещается в таблицу рабочей базы данных в том же закодированном виде, что и исходный набор.

4.2.2 Генерация правил

К набору найденных значимых множеств может быть применена процедура генерации правил. Для ее выполнения пользователю необходимо указать:

- имя таблицы в рабочей базе данных, содержащей набор значимых множеств;
- имя таблицы, которая должна быть создана системой автоматически для помещения в нее искомых правил;
- пороговые значения уверенности и поддержки для правил (от 0 до 1);
- тип множества, соответствующий записям исходной выборки.

Получаемые в результате выполнения процедуры генерации правила помещаются в закодированном виде в рабочую базу данных.

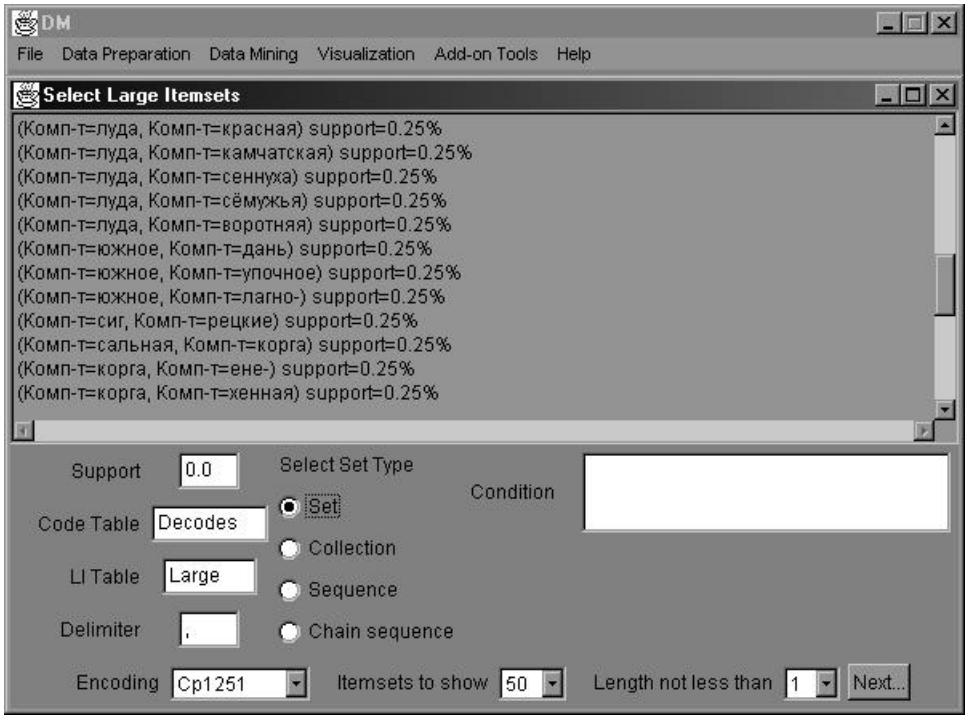

Рис. 4: Просмотр найденных значимых множеств

4.3 Просмотр результатов

Найденные на этапе анализа данных значимые множества и правила хранятся в закодированном виде в таблицах рабочей базы данных. Это позволяет использовать возможности СУБД для организации поиска правил по задаваемым параметрам. Программа в данном случае обеспечивает только удобный пользовательский интерфейс для составления запросов к рабочей базе данных с целью получения интересующей исследователя части результатов анализа данных и их представления в наглядном виде (см. рис. 4 и 5).

Пользователю предоставляется возможность выбора значимых множеств или правил, имеющих поддержку и/или степень уверенности выше задаваемых им пороговых значений, а также содержащих заданные элементы, задаваемые в формате: “атрибут = значение”. Допускается задание нескольких таких элементов, при этом необходимо указать программе символ или последовательность символов, которые будут использоваться в качестве разделителя.

Программа составляет SQL-запрос, соответствующий введенным пользователем параметрам выбора результатов, посылает его СУБД, получаемый от нее результат переводит в соответствии с таблицей кодирования в исходный естественный вид и отображает его в виде списка значимых множеств или правил с указанием степеней поддержки и/или уверенности.

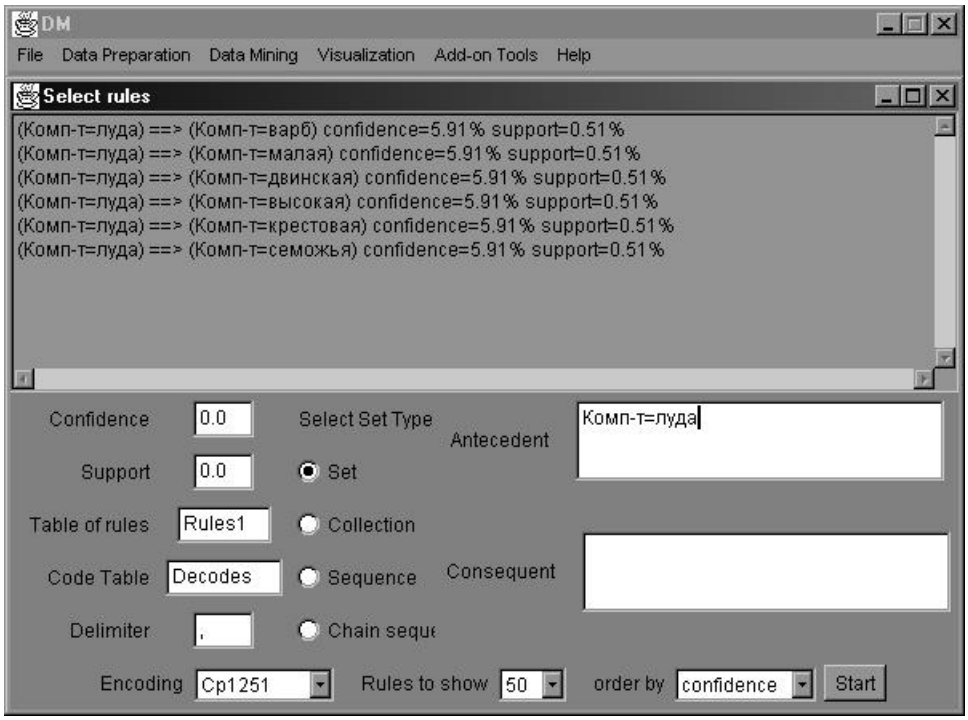

Рис. 5: Просмотр найденных правил

Список литературы

[1] Чуйко Ю., Разработка и реализация автоматированной системы DMiner // Материалы 54-й научной студенческой конференции, Петрозаводский государственный университет, 2002 г., с.171-172.