Министерство образования и науки Российской Федерации
Государственное образовательное учреждение высшего профессионального образования
“ЮЖНО-УРАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”
Факультет Вычислительной математики и информатики
Кафедра системного программирования

Рецензент
кандидат физ.-мат. наук

_____ Т.Ю. Лымарь

“____”_____ 2011 г.

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой СП

_____ Л.Б. Соколинский

“____”_____ 2011 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
на соискание академической степени магистра прикладной математики и информатики по направлению 010500.68 “Прикладная математика и информатика” (магистерская программа “Системное программирование”)

Интеграция алгоритма кластеризации Fuzzy *c*-Means в СУБД PostgreSQL

Ученый секретарь

_____ М.Л. Цымблер

“____”_____ 2011 г.

Научный руководитель
кандидат физ.-мат. наук, доцент

_____ М.Л. Цымблер

Автор работы
студент группы ВМИ-248

_____ Р.М. Миниахметов

Челябинск–2011

Министерство образования и науки Российской Федерации
Государственное образовательное учреждение высшего профессионального образования
“ЮЖНО-УРАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”
Факультет Вычислительной математики и информатики
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП

_____ Л.Б. Соколинский

“____”_____ 2011 г.

ЗАДАНИЕ

на выполнение выпускной квалификационной работы магистра студенту группы ВМИ-248 Миниахметову Руслану Марсовичу, обучающемуся в магистратуре по направлению 010500.68 “Прикладная математика и информатика” (магистерская программа “Системное программирование”)

1. **Тема работы** (утверждена приказом ректора от 22.03.2011 № 514)
   Интеграция алгоритма кластеризации Fuzzy c-Means в СУБД PostgreSQL.
2. **Срок сдачи студентом законченной работы:** 05.06.2011.
3. **Исходные данные к работе**
   - Ordonez C. Integrating K-Means Clustering with a Relational DBMS Using SQL. IEEE Educational Activities Department, 2006. P. 188–201.
   - Stonebraker M., Rowe L. A., Hirohama M. The Implementation of Postgres. IEEE Computer Society, 1990. P. 125–142.
   - Dunn J. C. A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. Cybernetics and Systems: An International Journal, 1973. P. 32–57.
4. **Перечень подлежащих разработке вопросов**
   4.1. Обзор публикаций, посвященных тематике работы.
   4.2. Проектирование алгоритма нечеткой кластеризации данных на языке реляционных баз данных SQL.
   4.3. Реализация алгоритма адаптированного для PostgreSQL.
   4.4. Разработка тестов и проведение тестирования.
   4.5. Проведение вычислительных экспериментов по исследованию эффективности разработанного алгоритма.
5. **Дата выдачи задания:** 08.02.2011.

Научный руководитель
доцент каф. системного программирования ЮУрГУ
кандидат физ.-мат. наук, доцент М.Л. Цымблер

Задание принял к исполнению Р.М. Миниахметов

Оглавление

Введение

Интеллектуальный анализ данных (Data Mining, Knowledge Discovery in Databases) — процесс обнаружения в сырых данных ранее неизвестных нетривиальных практически полезных и доступных интерпретации знаний, необходимых для принятия решений в различных сферах человеческой деятельности [1].

Интеллектуальный анализ данных включает в себя следующие основные задачи [1]. Выявление групп, имеющих нехарактерные для исходного набора данных признаки, называется *анализом отклонений*. *Поиск ассоциативных правил* представляет собой нахождение закономерностей между связанными событиями в исследуемом наборе данных. Решение *задачи классификации* позволяет выявить признаки, которые характеризуют группу объектов исследуемого набора данных. Нахождение групп объектов (*кластеров*) имеющих некоторые общие признаки назвается *кластеризацией*.

Задача *кластеризации* (или задача *обучения без учителя*) заключается в разбиении выборки на непересекающиеся подмножества, называемые *кластерами*, таким образом, чтобы каждый кластер состоял из объектов, близких по метрике, а объекты разных кластеров существенно отличались. Кластеризация применяется в распознавании образов, в частности, анализе медицинских снимков [2, 3], химии [4], поиске информации и др. [5].

Методы интеллектуального анализа данных на данный момент имеют достаточно большое количество реализаций с открытым исходным кодом на языках программирования высокого уровня [6, 7, 8].

На сегодняшний день системы управления базами данных (СУБД) активно используются практически во всех сферах деятельности человека, связанных с хранением и обработкой информации. Примерами приложений, использующих СУБД, являются научные базы данных, электронная коммерция, индексирование и поиск информации в сети Интернет и др.

В настоящее время разработано достаточно большое количество алгоритмов интеллектуального анализа данных, предполагающих размещение анализируемых наборов данных в оперативной памяти. В то же время ис-

пользование этих методов и алгоритмов совместно с СУБД требует значительных накладных расходов, связанных с предварительным экспортом анализируемых данных из базы данных и импортом результатов анализа данных обратно в базу данных. Поэтому для эффективной обработки и анализа данных требуется работать с СУБД напрямую, без использования посредников [9], а использование эффективных алгоритмов глубинного анализа в реальных базах данных требует их тесной интеграции с современными реляционными СУБД [10, 11].

На сегодняшний день СУБД с открытым исходным кодом [12] получили достаточно широкое распространение [13] и являются надежной альтернативой коммерческим аналогам [14].

В связи с этим является *актуальной задача интеграции алгоритмов интеллектуального анализа данных в реляционные СУБД с открытым исходным кодом*.

Обзор литературы

Исследования по разработке и улучшению алгоритмов интеллектуального анализа данных представлены следующими работами. Эффективная реализация алгоритма поиска ассоциативных правил описана в работе [15]. Описанный алгоритм использует новый метод подсчета и записи состояний для уменьшения количества сканирований данных. Улучшение алгоритма кластеризации Fuzzy c-Means на основе гауссианов описано в работе [16]. В работе [17] описано улучшение базового алгоритма четкой кластеризации k-Means. Авторы предлагают метод определения оптимального количества кластеров, что существенно улучшает качество и скорость кластеризации.

Достаточно большое количество статей по интеллектуальному анализу данных посвящено теме интеграции алгоритмов интеллектуального анализа данных с реляционными СУБД. Основные подходы интеграции алгоритмов интеллектуального анализа данных описаны в работе [10]. В работе [9] рассматриваются подходы сильно- и слабосвязанной интеграции, их преимущества и недостатки. Описана программная среда, в основе которой ле-

жит оптимизация запросов для обеспечения сильносвязанной интеграции с реляционной СУБД. Авторы работ [18, 19] расширяют набор операторов языка SQL для работы с методами интеллектуального анализа данных. Автор работы [11] описывает интеграцию алгоритма кластеризации k-Means в реляционную СУБД, используя язык запросов SQL.

Исследования, посвященные использованию реляционных СУБД с открытым исходным кодом представлены следующими работами. Расширение языка запросов SQL для СУБД MySQL описано в работе [18]. Проект PargreSQL [20] посвящен созданию параллельной СУБД на основе PostgreSQL [21].

В работах [22, 23, 24] приведены подробные сравнительные обзоры и описаные преимущества платформ с открытым исходным кодом перед коммерческими аналогами.

Структура и объем работы

Работа состоит из введения, трех разделов, заключения, библиографии и приложения. Объем работы составляет 28 страниц, объем библиографии — 31 наименование.

Содержание работы

В разделе 1 приводится описание базового алгоритма Fuzzy c-Means.

Раздел 2 описывает реализацию алгоритма Fuzzy c-Means на языке SQL.

В разделе 3 приведены результаты вычислительных экспериментов.

В заключении суммируются основные результаты работы и рассматриваются направления дальнейших исследований в данной области.

В приложении приводится полный текст алгоритма pgFCM на языке *PL/pgSQL*.

1. Алгоритм Fuzzy c-Means

Определим задачу кластеризации и введем необходимые обозначения в соответствии с работой [5].

Объект (*вектор признаков*) $x \in \mathbb{R}^d$ — отдельный элемент данных, которым оперируют алгоритмы кластеризации.

Задача *кластеризации* (или задача *обучения без учителя*) заключается в следующем. Имеется обучающая выборка $X = (x_1, x_2, \ldots, x_n) \subset \mathbb{R}^d$ и функция расстояния между объектами $\rho(x, x')$. Требуется разбить выборку на непересекающиеся подмножества, называемые *кластерами*, таким образом, чтобы каждый кластер состоял из объектов, близких по метрике ρ, а объекты разных кластеров существенно отличались. При этом каждому объекту $x_i \in X$ приписывается метка (номер) кластера c_j.

Алгоритм кластеризации — это функция $a : X \to C$, которая любому объекту $x \in X$ ставит в соответствие метку кластера $c \in C$.

Четкая кластеризация — кластеризация, которая каждому вектору $x_i \in X$ ставит в соответствие только одну метку кластера c_j.

Нечеткая кластеризация — кластеризация, при которой, для каждого $x_i \in X$ определяются *степени принадлежности* $f_{i,j}$. Число $f_{i,j} \in \mathbb{R}$ показывает степень принадлежности вектора x_i кластеру c_j.

Алгоритм Fuzzy c-Means [25, 26] является одним из методов нечеткой кластеризации, который вычисляет степень принадлежности вектора нескольким кластерам.

Введем следующие обозначения:

- $d \in \mathbb{N}$ — размерность пространства векторов данных;
- $l \in \mathbb{N} : 1 \leqslant l \leqslant d$ — номер координаты вектора;
- $n \in \mathbb{N}$ — мощность обучающей выборки;
- $X \subset \mathbb{R}^d$ — обучающая выборка векторов данных;
- $i \in \mathbb{N} : 1 \leqslant i \leqslant n$ — номер вектора обучающей выборки;
- $x_i \in X$ — i-й вектор выборки;
- $k \in \mathbb{N}$ — количество кластеров;
- $j \in \mathbb{N} : 1 \leqslant j \leqslant k$ — номер кластера;
- $C \subset \mathbb{R}^{k \times d}$ — матрица, которая содержит центры кластеров;

- $c_j \in \mathbb{R}^d$ — центр кластера j, вектор размерности d;
- $x_{il}, c_{jl} \in \mathbb{R}$ — l-е координаты векторов x_i и c_j соответственно;
- $U \subset \mathbb{R}^{n \times k}$ — матрица степеней принадлежности, где $u_{ij} \in \mathbb{R}$: $0 \leqslant u_{ij} \leqslant 1$ — степень принадлежности вектора x_i кластеру j;
- $\rho(x_i, c_j)$ — функция расстояния, определяющая степень принадлежности вектора x_i кластеру j;
- $m \in \mathbb{R} : m > 1$ — степень нечеткости целевой функции;
- J_{FCM} — целевая функция алгоритма FCM.

Алгоритм основан на минимизации *целевой функции* J_{FCM}:

$$J_{FCM}(X, k, m) = \sum_{i=1}^{n} \sum_{j=1}^{k} u_{ij}^m \rho^2(x_i, c_j) \tag{1}$$

Нечеткое разбиение входного множества векторов осуществляется в ходе минимизации целевой функции (1). На каждой итерации происходит обновление матрицы U и центров кластеров c_j по следующим формулам:

$$u_{ij} = \sum_{t=1}^{k} \left(\frac{\rho(x_i, c_j)}{\rho(x_i, c_t)} \right)^{\frac{2}{1-m}} \tag{2}$$

$$\forall\, j, l \quad c_{jl} = \frac{\sum\limits_{i=1}^{n} u_{ij}^m \cdot x_{il}}{\sum\limits_{i=1}^{n} u_{ij}^m} \tag{3}$$

Пусть s — номер итерации, $u_{ij}^{(s)}$ и $u_{ij}^{(s+1)}$ — значения u_{ij} на шагах s и $s+1$ соответственно, а $\varepsilon \in (0, 1) \subset \mathbb{R}$ — критерий останова. Тогда условие завершения алгоритма выглядит следующим образом:

$$\max_{ij} \{|u_{ij}^{(s+1)} - u_{ij}^{(s)}|\} < \varepsilon \tag{4}$$

С каждой итерацией целевая функция (1) стремится к локальному минимуму (седловой точке) [27].

На Рис. 1 представлен алгоритм FCM. На вход алгоритма поступают

Вход: $X,\ m,\ \varepsilon,\ k$.

Выход: U.

Шаг 1. $s := 0$.

Шаг 2. $U^{(0)} := (u_{ij})$.

Шаг 3. (вычисление новых координат центроидов)

Вычислить $C^{(s)} := (c_j)$, используя формулу (3), где $u_{ij} \in U^{(s)}$.

Шаг 4. (обновление значений матриц)

Вычислить $U^{(s)}$ и $U^{(s+1)}$ по формуле (2).

Шаг 5. $s := s + 1$.

Шаг 6. Если условие (4) не выполняется, то перейти на шаг 3.

Шаг 7. Стоп.

Рис. 1. Алгоритм Fuzzy c-Means

множество векторов данных $X = (x_1, x_2, \ldots, x_n)$, количество кластеров k, степень нечеткости m и критерий останова ε. На выходе алгоритма матрица степеней принадлежности U.

2. Реализация алгоритма Fuzzy c-Means на языке SQL

В данном разделе описывается реализации алгоритма Fuzzy c-Means на языке SQL. Подраздел **“Общие определения”** содержит определения, используемые для реализации алгоритма. В подразделе **“Схема базы данных”** приводится описание реляционных таблиц. Подраздел **“Алгоритм pgFCM”** содержит общее описание предложенного алгоритма, а также подробное описание каждого шага алгоритма.

2.1. Общие определения

Для интеграции алгоритма FCM с реляционной СУБД необходимо обеспечить хранение данных, которыми оперирует алгоритм (U, X), в виде реляционных таблиц. Идентификация элементов реляционных таблиц осуществляется с использованием номеров, указанных в Табл. 1 (где числа n, k и d определены ранее в разделе 1).

Табл. 1. Нумерация элементов данных

Номер	Интервал	Семантика
i	$\overline{1,n}$	номер вектора данных
j	$\overline{1,k}$	номер кластера
l	$\overline{1,d}$	номер координаты вектора

В качестве функции расстояния $\rho(x_i, c_j)$ нами без ограничения общности используется евклидова метрика:

$$\rho(x_i, c_j) = \sqrt{\sum_{l=1}^{d}(x_{il} - c_{jl})^2} \tag{5}$$

Для нахождения максимального значения $|u_{ij}^{(s+1)} - u_{ij}^{(s)}|$ определим функцию δ следующим образом:

$$\delta = \max_{ij}\{|u_{ij}^{(s+1)} - u_{ij}^{(s)}|\} \tag{6}$$

Значение функции δ используется при проверке условия завершения (4).

2.2. Схема базы данных

Опишем схему базы данных для алгоритма pgFCM.Таблицы Краткое описание и семантика таблиц, приведены в Табл. 2. Атрибуты, являющиеся первичными ключами, подчеркнуты.

Табл. 2. Схема базы данных алгоритма

№	Таблица	Семантика	Атрибуты	Кол-во записей
1	SH	Выборка векторов данных	$\underline{i}, x1, x2, \ldots, xd$	n
2	SV	Выборка векторов данных	$\underline{i, l}, val$	$n{\cdot}d$
3	C	Координаты центроидов	$\underline{j, l}, val$	$k{\cdot}d$
4	SD	Расстояния между x_i и c_j	$\underline{i, j}, dist$	$n{\cdot}k$
5	U	Степени принадлежности вектора X_i кластеру j на шаге s	$\underline{i, j}, val$	$n{\cdot}k$
6	UT	Степени принадлежности вектора X_i кластеру j на шаге $s{+}1$	$\underline{i, j}, val$	$n{\cdot}k$
7	P	Значение функции (6) на текущей итерации	$\underline{d, k, n}, s, delta$	число итераций

Для хранения выборки векторов множества X требуется определить таблицу $SH(\underline{i}, x1, x2, \ldots, xd)$, каждая запись которой хранит вектор данных размерности d с номером i. Таблица SH имеет n записей и первичный ключ i.

В ходе выполнения вычислений, предусмотренных алгоритмом FCM, требуется выполнять агрегирование (подсчет суммы, максимума и др.) координат векторов множества X. Однако, в силу своего определения, таблица SH не позволяет применять функции агрегирования языка SQL. В соответствии с этим нами определяется таблица $SV(\underline{i, l}, val)$, состоящая из $n{\cdot}d$ записей и имеющая составной первичный ключ (i, l). Таблица SV представляет собой выборку данных из таблицы SH. Структура таблицы позволяет применять агрегирующие функции языка SQL, например, функции $max()$ и $sum()$.

Для хранения данных о координатах центроидов кластеров необходимо создать отдельную временную таблицу $C(\underline{j,l},val)$, которая имеет $k{\cdot}d$ записей и составной первичный ключ (j,l). Как и у таблицы SV, структура таблицы C позволяет применять агрегирующие функции.

Согласно алгоритму на Рис. 1, на шаге 5 алгоритма требуется определять степень принадлежности вектора i кластеру j, что включает в себя вычисление расстояний $\rho(x_i, c_j)$. Для хранения расстояний используется таблица $SD(\underline{i,j},dist)$ с количеством записей $n{\cdot}k$ и составным первичным ключом (i,j).

Таблица $U(\underline{i,j},val)$ хранит степени принадлежности, полученных на шаге s. Для хранения степеней принадлежности на шаге $s{+}1$ потребуется еще одна, аналогичная по структуре, таблица $UT(\underline{i,j},val)$. Обе таблицы имеют количество записей $n{\cdot}k$, а также составной первичный ключ (i,j).

Таблица $P(\underline{d,k,n},s,delta)$ хранит номер итерации s и значение формулы (6) для этого номера. Количество записей в таблице зависит от числа итераций, которые понадобились для завершения алгоритма

2.3. Алгоритм pgFCM

Выполнение алгоритма инициируется вызовом хранимой процедуры на языке *PL/pgSQL*. На Рис. 2 показаны основные шаги алгоритма pgFCM.

Входное множество векторов данных X хранится в таблице SH. Степень нечеткости m, критерий останова eps и количество кластеров k являются входными параметрами функции $pgFCM$. Конечный результат работы алгоритма pgFCM находится в таблице U.

2.3.1. Реализация шага “Подготовка”

Команды создания временных таблиц SV, U и P (см. Табл. 2) приведены на Рис. 3. Для создания таблиц используется ключевое слово `TEMP`, которое указывает, что создается специальная временная таблица. Временные таблицы создаются в специальном табличном пространстве и уничтожаются по завершению SQL-сессии. Кроме того, временные таблицы с

Вход: $m,\ eps,\ k,\ SH$.
Выход: U.

Шаг 1. (инициализация таблиц) Создать и инициализировать временные таблицы U, P, SV и др.

Шаг 2. (вычисления)

- Вычислить координаты центроидов. Обновить таблицу C.
- Вычислить расстояния $\forall\ y_i,\ c_j\ \|y_i - c_j\|$. Обновить таблицу SD.
- Вычислить $UT = (ut_{ij})$. Обновить таблицу UT.

Шаг 3. (обновление таблиц) Обновить таблицы P и U.

Шаг 4. (проверка завершения) Если условие $\max\limits_{ij} \|ut_{ij} - u_{ij}\| < \varepsilon$ не выполняется, то перейти на шаг 2.

Шаг 5. Стоп.

Рис. 2. Алгоритм pgFCM

одинаковыми именами могут независимо использоваться разными пользователями.

Команды создания других таблиц приводятся в Приложении.

```
CREATE TEMP TABLE U (i int, j int, val numeric,
  PRIMARY KEY (i,j));

CREATE TEMP TABLE P (d int, k int, n int, s int,
  delta numeric, PRIMARY KEY (d,k,n));

CREATE TEMP TABLE SV (i int, l int, val numeric,
  PRIMARY KEY (i,l));
```

Рис. 3. Создание таблиц SV, U и P алгоритма

2.3.2. Реализация шага “Инициализация”

Перед выполнением вычислительной части алгоритма необходимо проинициализировать таблицы SV, U и P. Инициализация таблиц SV, U и P алгоритма pgFCM показана на Рис. 4. Таблица SV формируется путем выборки записей из таблицы SH. Существует несколько подходов к

инициализации координат центроидов, в данном документе используется следующий. Для таблицы U за степень принадлежности вектора x_i кластеру j принимается случайное число, которое затем нормируется.

```
-- инициализация таблицы SV
 INSERT INTO SV
   SELECT SH.i, 1, x1 FROM SH;
 ...
 INSERT INTO SV
   SELECT SH.i, d, xd FROM SH;

-- инициализация таблицы P
 INSERT INTO P(d, k, n, s, delta)
   VALUES (d, k, n, 0, 0.0);

-- инициализация таблицы U
 INSERT INTO U (i, j, val)
   VALUES (1, 1, random());
 ...
 INSERT INTO U (i, j, val)
   VALUES (i, j, random());
 ...
 INSERT INTO U (i, j, val)
   VALUES (n, k, random());

-- нормирование степеней принадлежности
 UPDATE U SET val = val / U1.tmp
 FROM (SELECT i, sum(val) AS tmp
       FROM U
       GROUP BY i) AS U1
 WHERE U1.i = U.i ;
```

Рис. 4. Инициализация реляционных таблиц алгоритма pgFCM

Таким образом, после нормирования соблюдаются следующие свойства [26] алгоритма Fuzzy c-Means:

$$\forall\, i, j \quad u_{ij} \in [0; 1] \tag{7}$$

$$\forall\, i \quad \sum_{j=1}^{k} u_{ij} = 1 \tag{8}$$

При инициализации таблицы P количество кластеров k задается процедурой $pgFCM$ и является ее параметром. Размерность пространства векторов d и мощность обучающей выборки n задаются на этапе подготовки. Номер итерации s и $delta$ инициализируются нулевыми значениями.

2.3.3. Реализация шага “Вычисление”

На шаге вычислений алгоритма $pgFCM$ Рис. 2 производятся вычисления степеней принадлежности, центров кластеров и расстояний по формулам (2), (3), и (5) соответственно. Соответствующий исходный код приведен на Рис. 5.

```
-- вычисление центров кластеров
 INSERT INTO C
   SELECT R.j, SV.l, sum(R.s * SV.val) / sum(R.s) AS val
   FROM (SELECT i, j, U.val^m AS s
         FROM U) AS R, SV
         WHERE R.i = SV.i
         GROUP BY j, l;

-- вычисление расстояний
 INSERT INTO SD
   SELECT i, j, sqrt(sum((SV.val - C.val)^2)) as dist
   FROM SV, C
   WHERE SV.l = C.l;
   GROUP BY i, j;

-- вычисление степеней принадлежности
 INSERT INTO UT
   SELECT i, j, SD.dist^(2.0^(1.0-m)) * SD1.den AS val
   FROM (SELECT i, 1.0 / sum(dist^(2.0^(m-1.0))) AS den
         FROM SD
         GROUP BY i) AS SD1, SD
   WHERE SD.i = SD1.i;
```

Рис. 5. Вычисления

Согласно алгоритму FCM (см. Рис. 1), вычисление степеней принадлежности производится по формуле (2). Поскольку числитель дроби в формуле не зависит от t, то для удобства использования формулу (2) можно переписать в следующем виде:

$$u_{ij} = \rho^{\frac{2}{1-m}}(x_i, c_j) \cdot \left(\sum_{t=1}^{k} \rho^{\frac{2}{m-1}}(x_i, c_t) \right)^{-1}$$

2.3.4. Реализация шага “Обновление”

На шаге обновления алгоритма pgFCM происходит обновления таблиц P и U. Соответствующий код приведен на Рис. 6.

```
-- обновление служебной таблицы
 SELECT max(abs(UT.val - U.val)) INTO tmp
 FROM U, UT
 WHERE U.i = UT.i AND U.j = UT.j;

 INSERT INTO P
   VALUES (d, k, n, steps, tmp);

-- обновление таблицы степеней принадлежности
 TRUNCATE U;
 INSERT INTO U
   SELECT * FROM UT;
```

Рис. 6. Обновление таблиц P и U

В таблице P обновляются значения номера интерации s и значение формулы (6) $delta$. Таблица UT хранит временные значения степеней принадлежности, которые затем вносятся в таблицу U. Для быстрого удаления всех записей таблицы U, полученных на предыдущем шаге, используется оператор `truncate`.

2.3.5. Реализация шага “Проверка”

Шаг проверки является заключительным этапом алгоритма pgFCM. На каждой итерации выполняется проверка условия завершения алгоритма (4). Соответствующий код показан на Рис. 7. Для осуществления про-

```
 IF (tmp < eps) THEN
   RETURN;
 END IF;
```

Рис. 7. Проверка условия завершения

верки используется выборка значения формулы (6) во временную переменную tmp процедуры $pgFCM$.

3. Вычислительные эксперименты

В данном разделе описаны результаты вычислительных экспериментов. В подразделе **“Тестирование алгоритма pgFCM”** представлено тестирование разработанного алгоритма на стандартных наборах данных “Butterfly” и “Iris”. В подразделе **“Быстродействие алгоритма”** представлен график работы алгоритма pgFCM на различных наборах данных.

Эксперименты проводились на следующей аппаратно-программной платформе:

- Процессор AMD ATHLON 64 X2 2,8 ГГц.
- Объем оперативной памяти 3,6 ГБайт.
- Операционная система GNU/Linux 2.6.35 x86_64.
- СУБД PostgreSQL версии 9.0.4.
- Среда для проведения статистических вычислений с открытым исходным кодом KNIME [8, 22, 28] версии 2.3.4.

3.1. Тестирование алгоритма pgFCM

Множество “Butterfly”

Набор данных “Butterfly” [26] содержит 15 векторов размерности 2, предназначен для проверки работы алгоритмов нечеткой кластеризации. На Рис. 8 показан результат работы алгоритма на множестве “Butterfly”. Вектор с координатами $(3; 2)$ принадлежит одновременно 2 кластерам с точностью до $\varepsilon = 0,01$, что наглядно демонстрирует нечеткую кластеризацию.

Множество “Iris”

Набор данных “Iris” [29] содержит 150 векторов размерности 5. Каждый вектор относится к одному из 3-х классов, в каждом классе по 50 точек. Для кластеризации используются первые 4 координаты вектора, 5-я координата содержит класс вектора и используется для проверки результата кластеризации. На Рис. 9 показаны результаты работы алгоритма $pgFCM$.

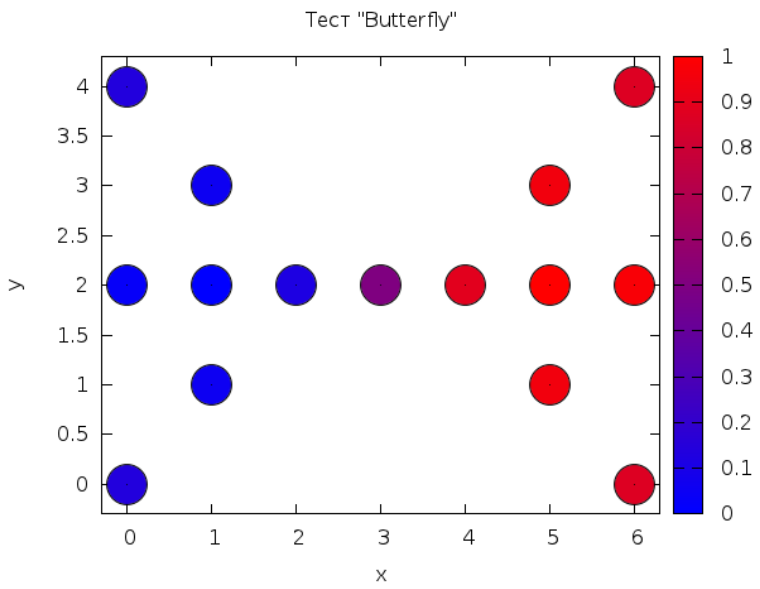

Таблица U

i	j	val
1	1	0.86
1	2	0.13
2	1	0.97
2	2	0.02
...	...	...
8	1	0.49
8	2	0.50
...	...	...
15	1	0.13
15	2	0.86

Рис. 8. Результат кластеризации множества “Butterfly”

Точки имеющие четкий цвет принадлежат одному кластеру. Результаты кластеризации совпадают с эталонными.

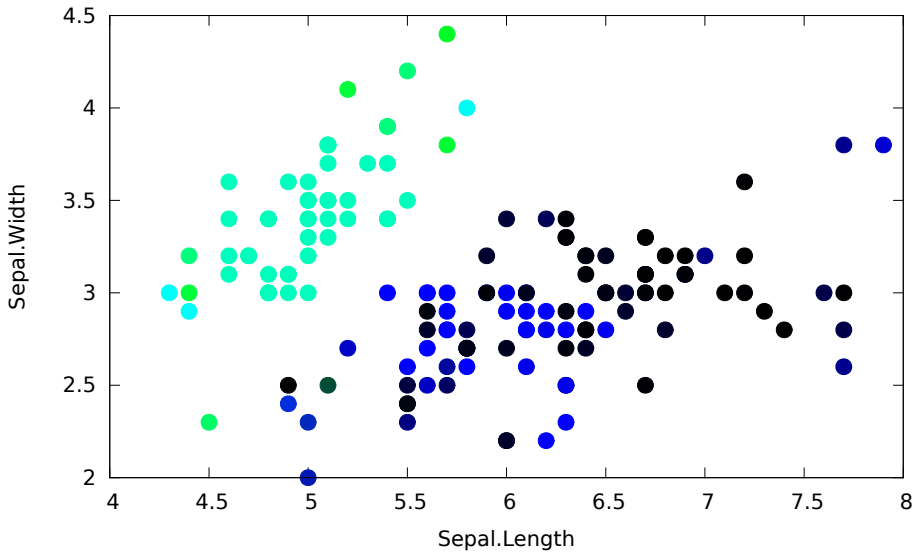

Рис. 9. Результат кластеризации множества “Iris”

3.2. Быстродействие алгоритма

Для исследования быстродействия алгоритма использовались реальные наборы данных (графические изображения) с параметрами $d = 5$,

$k = 3$, $n = \overline{200000,1600000}$. На Рис. 10 показаны результаты работы алгоритма на реальных наборах данных различных размеров.

KNIME был настроен следующим образом:

- количество рабочих потоков для KNIME установлено равным 1;
- максимально доступная память java-машины — 2.5 ГБайт;
- при истечении объема доступной оперативной памяти — использовать жесткий диск;
- для доступа к СУБД PostgreSQL использовался оригинальный драйвер JDBC, который показывает лучшие характеристики времени доступа, чем драйвер ODBC, использованный в работе [30].

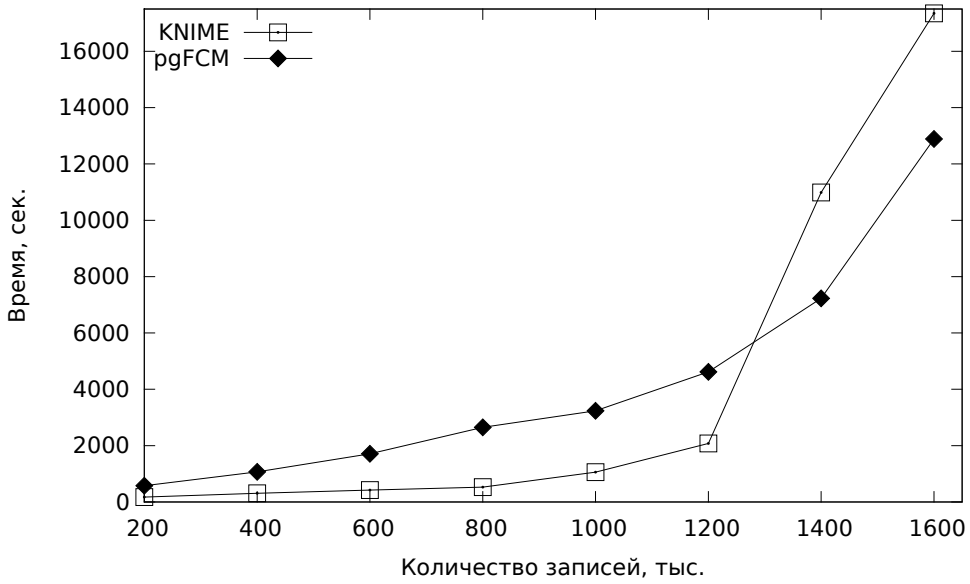

Рис. 10. Производительность алгоритма pgFCM

Как видно из графика эксперимента, начиная с определенного объема данных время работы KNIME существенно превосходит время работы алгоритма pgFCM в СУБД PostgreSQL. При больших объемах исходных данных, количество оперативной памяти для работы KNIME становится недостаточным и используется жесткий диск, скорость обмена данным с которым существенно ниже. Превосходство работы СУБД и алгоритма pgFCM сохранится и на более высоких объемах данных, поскольку СУБД работают в таких условиях более эффективно. Кроме того, с ростом объема ис-

ходных данных увеличивается время их выгрузки из базы данных, а также время внесения результата кластеризации в базу данных.

В таблице Табл. 3 представлены результаты исследования быстродействия, а также время на выгрузку исходного множества векторов из базы данных и время на загрузку ответа в базу данных.

Табл. 3. Временные характеристики

N, тыс.	pgFCM, сек	KNIME, сек	JDBC (из БД), сек	JDBC (в БД), сек
200	578	174	3	25
400	1067	310	9	49
600	1711	423	13	75
800	2648	529	28	100
1000	3238	1061	95	125
1200	4620	2078	123	152
1400	7229	10989	161	178
1600	12888	17347	216	223

4. Заключение

Данная работа посвящена разработке алгоритма нечеткой кластеризации для свободной реляционной СУБД PostgreSQL.

Апробация работы

Результаты работы докладывались автором на международной конференции “The Seventh Spring Researchers Colloquium on Databases and Information Systems”, SYRCoDIS’2011 (June 2-3, 2011, Moscow, Russia).

Публикация работы

Результаты работы опубликованы автором в сборнике трудов *Miniakhmetov R.* Integrating Fuzzy c-Means Clustering with PostgreSQL // Proceedings of the Seventh Spring Researchers’ Colloquium on Databases and Information Systems (SYRCoDIS’2011). Moscow: Moscow State University, 2011. P. 6–10. [31]

Основные результаты работы

1. Выполнен обзор научных публикаций по тематике исследования. Обзор показал актуальность задачи интеграции алгоритмов интеллектуального анализа данных в реляционные СУБД с открытым исходным кодом.

2. Выполнено проектирование алгоритма нечеткой кластеризации на языке запросов SQL и схемы соответствующей реляционной базы данных.

3. Выполнена реализация разработанного алгоритма для СУБД PostgreSQL.

4. Выполнено тестирование на стандартных наборах данных Butterfly и Iris.

5. Проведены эксперименты для исследования эффективности разработанного алгоритма на различных наборах данных.

Направления дальнейших исследований

Дальнейшие исследования могут быть направлены на улучшение текущей реализации алгоритма pgFCM, а также на разработку параллельной версии алгоритма.

Литература

1. *Frawley W.J., Piatetsky-Shapiro G., Matheus C.J.* Knowledge Discovery in Databases: An Overview // AI Magazine. 1992. Vol. 13, No. 3. P. 57–70.

2. *Shihab A.I.* Fuzzy Clustering Algorithms and their Applications to Medical Image Analysis: Ph. D. thesis / University of London. 2000.

3. *Zhang D., Chen S.* A Novel Kernelized Fuzzy c-Means Algorithm with Application in Medical Image Segmentation // Artificial Intelligence in Medicine. 2004. Vol. 32. P. 37–50.

4. *Li X., Lu X., Tian J. et al.* Application of Fuzzy c-Means Clustering in Data Analysis of Metabolomics // Analytical Chemistry. 2009. Vol. 81, No. 11. P. 4468–4475.

5. *Jain A.K., Murty M.N., Flynn P.J.* Data clustering: a review // ACM Computing Surveys. 1999. Vol. 31. P. 264–323.

6. *Dimitriadou E., Hornik K., Leisch F. et al.* Machine Learning Open-Source Package ‘r-cran-e1071’. 2010. URL: `http://cran.r-project.org/web/packages/e1071/index.html` (дата обращения: 23.06.2011).

7. *Foundation A.S., Drost I., Dunning T. et al.* Apache Mahout. 2010. URL: `https://cwiki.apache.org/confluence/display/MAHOUT/Fuzzy+K-Means` (дата обращения: 23.06.2011).

8. *Berthold M.R., Cebron N., Dill F. et al.* KNIME — the Konstanz Information Miner: Version 2.0 and Beyond // SIGKDD Explorations Newsletter. 2009. Vol. 11. P. 26–31.

9. *Nestorov S., Tsur S.* Integrating Data Mining with Relational DBMS: A Tightly-Coupled Approach // Proceedings of the 4th International Workshop on Next Generation Information Technologies and Systems. NGIT ’99. London, UK: Springer-Verlag, 1999. P. 295–311.

10. *Sarawagi S., Thomas S., Agrawal R.* Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications // Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data. SIGMOD '98. New York, NY, USA: ACM, 1998. P. 343–354.

11. *Ordonez C.* Integrating K-Means Clustering with a Relational DBMS Using SQL // IEEE Transactions on Knowledge and Data Engineering. 2006. Vol. 18, No. 2. P. 188–201.

12. *Udoh E.* Database Technologies: Concepts, Methodologies, Tools, and Applications (4 Volumes), Ed. by J. Erickson. IGI Global, 2009.

13. *Paulson L.D.* Open Source Databases Move into the Marketplace // Computer. 2004. Vol. 37. P. 13–15.

14. *Evdoridis T., Tzouramanis T.* A Generalized Comparison of Open Source and Commercial Database Management Systems // Database Technologies: Concepts, Methodologies, Tools, and Applications / Ed. by J. Erickson. IGI Global, 2009. P. 13–27.

15. *Wu H., Lu Z., Pan L. et al.* An Improved Apriori-based Algorithm for Association Rules Mining // Fuzzy Systems and Knowledge Discovery, Fourth International Conference on. 2009. Vol. 2. P. 51–55.

16. *Ramathilagam S., Huang Y.-M.* Extended Gaussian Kernel Version of Fuzzy c-Means in the Problem of Data Analyzing // Expert Systems with Applications. 2011. Vol. 38, No. 4. P. 3793–3805.

17. *Pelleg D., Moore A.W.* X-means: Extending K-means with Efficient Estimation of the Number of Clusters // Proceedings of the Seventeenth International Conference on Machine Learning. ICML '00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000. P. 727–734.

18. *Ferro A., Giugno R., Puglisi P., Pulvirenti A.* MySQL Data Mining: Extending MySQL to Support Data Mining Primitives (Demo) // Knowledge-Based and Intelligent Information and Engineering Systems.

Springer Berlin / Heidelberg, 2010. Vol. 6278 of *Lecture Notes in Computer Science*. P. 438–444.

19. *Zakrewicz M.* Databases and Information Systems / Ed. by J. Barzdins, A. Caplinskas. Norwell, MA, USA: Kluwer Academic Publishers, 2001. P. 85–96.

20. *Пан К.С., Цымблер М.Л.* Архитектура и принципы реализации параллельной СУБД PargreSQL // Параллельные вычислительные технологии: труды международной научной конференции. ПаВТ ’2011. Челябинск: Издательский центр ЮУрГУ, 2011. P. 577–584.

21. *Stonebraker M., Rowe L.A., Hirohama M.* The Implementation of POSTGRES // IEEE Transactions on Knowledge and Data Engineering. 1990. Vol. 2. P. 125–142.

22. *Chen X., Ye Y., Williams G., Xu X.* A Survey of Open Source Data Mining Systems // Proceedings of the 2007 International Conference on Emerging Technologies in Knowledge Discovery and Data Mining. PAKDD’07. Berlin, Heidelberg: Springer-Verlag, 2007. P. 3–14.

23. *Golfarelli M.* Open Source BI Platforms: A Functional and Architectural Comparison // Proceedings of the 11th International Conference on Data Warehousing and Knowledge Discovery. DaWaK ’09. Berlin, Heidelberg: Springer-Verlag, 2009. P. 287–297.

24. *Thomsen C., Pedersen T.B.* A Survey of Open Source Tools for Business Intelligence // International Journal of Data Warehousing and Mining. 2009. Vol. 5, No. 3. P. 56–75.

25. *Dunn J.C.* A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters // Journal of Cybernetics. 1973. Vol. 3. P. 32–57.

26. *Bezdek J.C.* Pattern Recognition with Fuzzy Objective Function Algorithms. Norwell, MA, USA: Kluwer Academic Publishers, 1981.

27. *Bezdek J., Hathaway R., Sobin M., Tucker W.* Convergence Theory for Fuzzy c-means: Counterexamples and Repairs // IEEE Transactions on Systems, Man, and Cybernetics. 1987. Vol. 17. P. 873–877.

28. *Tiwari A., Sekhar A.K.* Workflow-based Framework for Life Science Informatics // Computational Biology and Chemistry. 2007. Vol. 31, No. 5-6. P. 305–319.

29. *Frank A., Asuncion A.* UCI Machine Learning Repository. URL: `http://archive.ics.uci.edu/ml` (дата обращения: 23.06.2011).

30. *Ordonez C.* Programming the K-means clustering algorithm in SQL // KDD / Ed. by W. Kim, R. Kohavi, J. Gehrke, W. DuMouchel. ACM, 2004. P. 823–828.

31. *Miniakhmetov R.* Integrating Fuzzy c-Means Clustering with PostgreSQL // Proceedings of the 7th Spring Researchers' Colloquium on Databases and Information Systems. Moscow State University, 2011. P. 6–10.

Приложение. Исходный текст алгоритма pgFCM

```
CREATE OR REPLACE FUNCTION pgfcm(d integer, k integer,
  m numeric, eps numeric, dataset text)
 RETURNS void
 LANGUAGE plpgsql
AS $function$
DECLARE
  tmp numeric;
  steps int:=0;
  n int;
  l_iter int:=0;
  qry text:= 'CREATE TABLE SH (i int';
BEGIN
  -- Инициализация --
  CREATE TEMP TABLE C (j int, l int, val numeric,
    PRIMARY KEY (j,l));
  CREATE TEMP TABLE SD (i int, j int, dist numeric,
    PRIMARY KEY (i,j));
  CREATE TEMP TABLE U (i int, j int, val numeric,
    PRIMARY KEY (i,j));
  CREATE TEMP TABLE UT (i int, j int, val numeric,
    PRIMARY KEY (i,j));
  CREATE TEMP TABLE P (d int, k int, n int, s int,
    delta numeric, PRIMARY KEY (d,k,n));

  DROP TABLE IF EXISTS SH;
  CREATE TEMP TABLE SV (i int, l int, val numeric,
    PRIMARY KEY (i,l));

  FOR l_iter IN 1..d LOOP
    qry := qry || ',x' || to_char(l_iter, 'FM999') || ' numeric';
  END LOOP;
  qry := qry || ',PRIMARY KEY (i));';
  EXECUTE qry;

  EXECUTE 'COPY SH FROM ' || quote_literal(dataset) ||
    ' DELIMITER AS ' || quote_literal(';') || ' CSV;';
  -- Заполнение таблицы SV
  FOR l_iter IN 1..d LOOP
    EXECUTE 'INSERT INTO SV SELECT SH.i, '||l_iter||
```

```
    ' as l, x'||l_iter||' as val FROM SH';
END LOOP;

SELECT count(*) INTO n FROM SH;
-- Инициализация таблицы степеней принадлежности
FOR i IN 1..n LOOP
  FOR j IN 1..k LOOP
    EXECUTE 'INSERT INTO U VALUES('||i||', '||j||', random())';
  END LOOP;
END LOOP;
-- Нормирование степеней принадлежности
UPDATE U SET val = val / U1.tmp
  FROM (SELECT i, sum(val) AS tmp
        FROM U
        GROUP BY i) AS U1
WHERE U1.i = U.i ;

INSERT INTO P VALUES (d, k, n, 0, 0.0);

LOOP
-- Вычисления --
-- Вычисление координат центров кластеров
TRUNCATE C;
INSERT INTO C
  SELECT R.j, SV.l, sum(R.s * SV.val) / sum(R.s) AS val
  FROM (SELECT i, j, U.val^m AS s
        FROM U) AS R, SV
        WHERE R.i = SV.i
        GROUP BY j, l;
-- Вычисление расстояний
TRUNCATE SD;
INSERT INTO SD
  SELECT i, j, sum((SV.val - C.val)^2) AS dist
  FROM SV, C
  WHERE SV.l = C.l
  GROUP BY i, j;
-- Вычисление степеней приналежности
TRUNCATE UT;
INSERT INTO UT
  SELECT SD.i, j, SD.dist^(2.0/(1.0-m)) * SD1.den AS val
  FROM (SELECT i, 1.0 / sum(dist^(2.0/(1.0-m))) AS den
```

```
            FROM SD
            GROUP BY i) AS SD1, SD
            WHERE SD.i = SD1.i;
  -- Обновление --
  SELECT max(abs(UT.val - U.val)) INTO tmp
  FROM UT, U
  WHERE UT.i = U.i AND UT.j = U.j;
  -- Обновление служебной таблицы
  INSERT INTO P
    VALUES (d, k, n, steps, tmp);

  -- Обновление таблицы степеней принадлежности
  TRUNCATE U;
  INSERT INTO U
    SELECT * FROM UT;
  -- Проверка --
  IF (tmp < eps) THEN
    RETURN;
  END IF;
  steps := steps + 1;
  END LOOP;
END;
$function$
```