УДК 004.93.51

Т.М. Косовская, М.А. Власова

ИСПОЛЬЗОВАНИЕ ЯЗЫКОВ СЕМЕЙСТВА PROLOG ДЛЯ РАСПОЗНАВАНИЯ ИЗОБРАЖЕНИЙ

Косовская Т.М., Власова М.А. **Использование языков семейства Prolog для распознавания изображений.**

Аннотация. Статья посвящена исследованию возможностей языков семейства Prolog для их использования при решении задач распознавания изображений на экране дисплея. Отмечены трудности, возникшие при реализации подхода на языках семейства Prolog. Показано, как использование оценок числа шагов работы алгоритма поиска вывода для рассматриваемой задачи позволило преодолеть возникшие трудности. Приведены примеры применения написанных программ к выделению эталонного изображения на сложном изображении. Проанализированы особенности использования различных форматов изображения, предъявленного к распознаванию.

Ключевые слова: распознавание изображений, сложность алгоритмов, язык Prolog.

Kosovskaya T.M., Vlasova M.A. **The use of Prolog language family for image recognition.**

Abstract. The paper is devoted to the investigation of Prolog language family possibility for its use in the solving of the display image recognition. Difficulties appeared in the Prolog implementation of the approach are marked out. It is shown the way in which the number of deduction algorithm steps allows to overcome the appeared difficulties. Examples of the Prolog programs implementation to the separation of an etalon image from a complex one are done. Peculiarities of the use of different recognizable image formats are analized.

Keywords: image recognition, complexity of algorithm, Prolog.

1. Введение. В работе [1] предложен логико-предметный подход к решению задач распознавания образов. Одним из достоинств этого метода является возможность его применения к решению задачи анализа сложного объекта, заключающейся в выделении и распознавании частей объекта, а также в определении их взаимного расположения. Большинство подходов к решению задач распознавания образов плохо приспособлено (или вовсе не приспособлено) к решению такой задачи.

В работе [1] рассмотрены три типа задач распознавания образов: задача идентификации, задача классификации и задача анализа сложного объекта. Первые две задачи не только являются вспомогательными при решении задачи анализа сложного объекта, но и представляют самостоятельный интерес.

Для случая, когда исходные признаки задают свойства частей распознаваемых объектов или отношения между ними, доказана NP-трудность задач идентификации, классификации и анализа сложного объекта в общей постановке. Доказано также, что при фиксировании одного из параметров задач эти задачи имеют полиномиальный алгоритм решения. Доказаны оценки степени полиномов, ограничивающих число шагов решения указанных задач при использовании различных алгоритмов.

В настоящей работе излагается применение указанного подхода к решению задачи идентификации изображения. Отмечены трудности, возникшие при реализации подхода на языках семейства Prolog. Показано, как использование оценок числа шагов работы алгоритма поиска вывода для рассматриваемой задачи позволило преодолеть возникшие трудности.

2. Логико-предметный подход к решению задач распознавания образов. Пусть имеется множество Ω конечных множеств $\omega = \{\omega_1, \ldots, \omega_t\}$, которые в дальнейшем будем называть распознаваемыми объектами. Частью τ объекта ω будем называть любое его подмножество (не обязательно собственное). Пусть также на частях τ задан набор предикатов $p_1, \ldots, p_n$, характеризующих свойства элементов объекта ω и отношения между ними.

Пусть задано разбиение множества Ω на M (возможно пересекающихся) классов $\Omega = \bigcup_{k=1}^{M} \Omega_k$.

Логическим описанием $S(\omega)$ объекта ω называется набор всех истинных постоянных атомарных формул вида $p_i(\overline{\tau})$ или $\neg p_i(\overline{\tau})$, выписанных для всех возможных частей τ объекта ω. Здесь и далее посредством $\overline{x}_\Pi$ будем обозначать упорядочение конечного множества x, соответствующее перестановке Π номеров его элементов. Чтобы не загромождать формулы индексами будем писать $\overline{x}$ вместо $\overline{x}_\Pi$.

Логическим описанием класса (ОК) Ω_k называется такая формула $A_k(\overline{x})$, что

1. $A_k(\overline{x})$ содержит в качестве атомарных только формулы вида $p_i(\overline{y})$, где $y \subseteq x$;

2. $A_k(\overline{x})$ не содержит кванторов;

3. если истинна формула $A_k(\overline{\omega})$, то $\omega \in \Omega_k$.

ОК всегда может быть записано в виде дизъюнкции элементарных конъюнкций атомарных формул.

С помощью построенных описаний предлагается решать следующие задачи распознавания образов.

Задача идентификации. *Проверить, принадлежит ли объект ω или его часть классу Ω_k.*

Эта задача в [1] сведена к доказательству выводимости логического следования

$$S(\omega) \vdash \exists \overline{y}_{\neq}\ A_k(\overline{y}). \tag{1}$$

Задача классификации. *Найти все такие номера классов k, что $\omega \in \Omega_k$.*

Эта задача в [1] сведена к доказательству выводимости формулы $\bigvee_{k=1}^{M} A_k(\overline{\omega})$ из описания распознаваемого объекта $S(\omega)$ с указанием всех таких номеров k, для которых соответствующий дизъюнктивный член истинен на ω.

Задача анализа сложного объекта. *Найти и классифицировать все части τ объекта ω, для которых $\tau \in \Omega$.*

Эта задача в [1] сведена к доказательству выводимости формулы $\bigvee_{k=1}^{M} \exists \overline{y}_{\neq} A_k(\overline{y})$ из описания распознаваемого объекта $S(\omega)$ с указанием всех частей объекта ω, поддающихся классификации, и идентификации их.

Для практической реализации логико-предметного подхода к решению задач распознавания образов кажется приемлемым язык Пролог — известный язык искусственного интеллекта, использующий при решении задач метод резолюции. Были изучены возможности и ограничения различных разновидностей зыка Пролог. Чтение целевых предикатов из текстовых файлов (т.к. при решении многих задач необходимо, чтобы описания классов хранились в памяти компьютера), бесплатность, объем стека и поддерживаемые платформы и их разрядность - основные критерии отбора. На их основе были выбраны SWI Prolog (подходит по всем параметрам) и Visual Prolog (не поддерживает чтение целевых предикатов из текстовых файлов, требуется написание интерпретатора фактов).

Ниже анализируется, насколько Пролог подходит для решения описанных задач и каким образом можно уменьшить время решения задач распознавания изображений на экране дисплея. Приводятся примеры использования программ на языке Пролог для некоторых задач распознавания изображения.

Для получения исходных данных для программы на языке Пролог при распознавании изображений написана программа на C#. Она обрабатывает сложное изображение *image*, на котором ищем его часть - эталонный объект *class*. Все результаты записываются в текстовый файл. Затем вызывается программа на языке Пролог, которая выполняет поиск *class* на *image*. Если результат найден, он также записывается в текстовый файл. Иначе файл остается пуст.

Ниже описаны программа на C# (подготавливающая исходные данные задачи распознавания изображения), программы на Visual Prolog 7.3 и SWI Prolog (реализующие процесс поиска вывода). Показана структура текстовых файлов, необходимых для взаимодействия программ. Также показан способ составления предикатов, описывающих изображение (*image* и *class*).

3. Исходные данные для распознавания изображения. Работа программы, написанной на языке Пролог и решающей задачу идентификации (1), проверена на задаче идентификации части сложного изображения. При этом изначальное большое изображение, заданное в формате .jpeg (.jpg), .bmp или .png рассматривается как сложный объект ω, а описание его части, в котором различные константы заменены различными переменными, рассматривается как описание класса. Но при сохранении изображений в различных форматах могут возникнуть ситуации, при которых решение задачи не будет найдено. Ситуация может возникнуть в случае с изображениями в формате .jpeg и .bmp. Это объясняется тем, что при сохранении изображения обычно используются методы сжатия изображений, при которых появляются артефакты - пиксели с усредненными цветом и тоном (однако, может применяться режим сжатия lossless JPEG - сжатие без потерь, при котором артефакты не появляются) [9].

Изображения описываются попиксельно. Каждый пиксель характеризуется своими координатами и яркостью, которая вычисляется по формуле $0.3 \cdot R + 0.59 \cdot G + 0.11 \cdot B$ [5], где R, G, B – красная, зеленая и синяя компоненты цвета пикселя соответственно.

4. Уменьшение размерности задачи.

Для сокращения перебора при решении задачи пиксель будет характеризоваться не только собственной яркостью, но и яркостью восьми соседних пикселей (обход слева направо, сверху вниз).

Была предпринята попытка характеризовать пиксель только собственной яркостью (один исходный предикат). Но в связи с NP-трудностью задачи перебор при этом был слишком велик. Полученные в [1, 2, 3] оценки числа шагов решения задачи идентификации показывают, что чем больше количество исходных предикатов, тем меньше показатель степени у экспоненты, ограничивающей сверху число таких шагов.

По той же причине чем больше градаций яркости, тем быстрее работает программа. Максимальное количество градаций - 256, в программе пользователь может выбирать их количество. Следует отметить, что классы градаций яркости могут быть одно-, двух- и трехзначными. Поэтому строка, состоящая из названия и аргументов предиката может быть разной длины. С одной стороны, чем больше градаций яркости, тем быстрее осуществляется поиск и тем больше вероятность того, что стек Пролога не переполнится. С другой стороны, чем больше градаций, тем длиннее название предиката и тем больше времени требуется для записи описаний сложного изображения *image* и эталонного объекта *class* в текстовый файл.

На Рис. 1 представлен пример составления предиката, описывающего конкретный пиксель, при условии, что пользователь выбрал 5 градаций яркости.

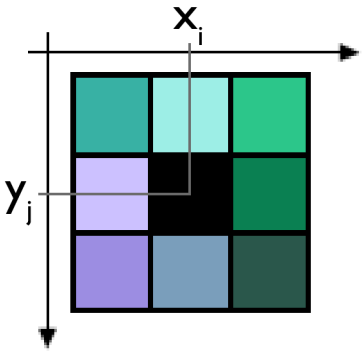

Рис. 1. Характеризация пикселя его окружением.

Для описываемого пикселя и его ближайших соседей вычисля-

ется яркость (I) и номер интервала яркости, в который яркость пикселя попадает. Так как пользователь выбрал 5 градаций, то определены 5 интервалов яркостей: $0 - [0,\ 51)$, $1 - [51,\ 102)$, $2 - [102,\ 153)$, $3 - [153,\ 204)$, $4 - [204,\ 255]$.

$$
\begin{array}{rcl}
I(x_{i-1},\ y_{j-1}) & = & 0.3 \cdot 56 + 0.59 \cdot 177 + 0.11 \cdot 164 \approx 139 \quad 3 \\
I(x_i,\ y_{j-1}) & = & 0.3 \cdot 157 + 0.59 \cdot 238 + 0.11 \cdot 230 \approx 212 \quad 4 \\
I(x_{i+1},\ y_{j-1}) & = & 0.3 \cdot 44 + 0.59 \cdot 198 + 0.11 \cdot 138 \approx 145 \quad 2 \\
I(x_{i-1},\ y_j) & = & 0.3 \cdot 204 + 0.59 \cdot 193 + 0.11 \cdot 255 \approx 203 \quad 3 \\
I(x_i,\ y_j) & = & 0.3 \cdot 0 + 0.59 \cdot 0 + 0.11 \cdot 0 \approx 0 \quad 0 \\
I(x_{i+1},\ y_j) & = & 0.3 \cdot 10 + 0.59 \cdot 128 + 0.11 \cdot 82 \approx 87 \quad 1 \\
I(x_{i-1},\ y_{j+1}) & = & 0.3 \cdot 156 + 0.59 \cdot 141 + 0.11 \cdot 226 \approx 154 \quad 3 \\
I(x_i,\ y_{j+1}) & = & 0.3 \cdot 122 + 0.59 \cdot 158 + 0.11 \cdot 187 \approx 150 \quad 2 \\
I(x_{i+1},\ y_{j+1}) & = & 0.3 \cdot 42 + 0.59 \cdot 87 + 0.11 \cdot 75 \approx 72 \quad 1
\end{array}
$$

Составляется цепочка интервалов яркости: 342301321. Таким образом, предикат, описывающий пиксель $(x_i,\ y_j)$ в SWI Prolog выглядит как $p342301321(i,\ j)$., а в Visual Prolog как $p(''342301321'',\ ''i'',\ ''j'')$. (Подробнее про особенности программ на SWI Prolog и Visual Prolog см. в разделах 5 и 6.)

Для описания *image* размером m x n будет получено $(m-2) \cdot (n-2)$ атомарных формул. Аналогичным образом, для описания *class* размером m_{Cl} x n_{Cl} будет составлено $(m_{Cl}-2) \cdot (n_{Cl}-2)$ атомарных формул, содержащих $m_{Cl} + n_{Cl}$ переменных.

В [1, 2, 3] доказано, что при использовании алгоритма полного перебора число шагов решения задачи распознавания изображений составит $O(t^{m'} \cdot |S|)$, где

t – число элементов в множестве $\omega = \{\omega_1, ..., \omega_t\}$,

m' – количество аргументов в $A(\overline{x})$,

$|S|$ – количество постоянных формул в $S(\omega)$.

В рассматриваемом случае

$t = m \cdot n$,

$m' = m_{Cl} \cdot n_{Cl}$,

$|S| = (m_{Cl}-2) \cdot (n_{Cl}-2)$.

При использовании секвенциального исчисления предикатов или метода резолюций для исчисления предикатов число шагов решения поставленной задачи составлит $O(D \cdot s^a)$, где

D – количество дизъюнктов в описании классов,

s – максимальное количество вхождений одного и того же предиката в описание объекта $S(\omega)$,

a – максимальное количество вхождений атомарных формул в элементарные конъюнкции, составляющие описание $A(\overline{x})$.

В рассматриваемом случае

$D = 1$,

$a = m_{Cl} \cdot n_{Cl}$.

Величина s зависит от того, насколько зашумлено изображение (чем больше шум, тем меньше s).

Понятно, что объем данных очунь велик. Для уменьшения количества предикатов и облегчения поиска эталона $class$ на сложном изображении $image$ предлагается два варианта:

1. Добавить шум к изображениям, чтобы избежать больших областей постоянной яркости.

2. Уменьшить размер эталонного объекта $class$ и сложного изображения $image$.

Ниже представлен способ уменьшения размера изображения.

Вводится дополнительный натуральный параметр k, задающий степень сжатия изображения. (В рассматриваемой реализации нет универсального k, пользователь сам выбирает подходящее число.)

Пусть

$$m = k \cdot m^{'} + ost_m, \quad n = k \cdot n^{'} + ost_n,$$

$$m_{Cl} = k \cdot m^{'}_{Cl} + ost_{Cl_m}, \quad n = k \cdot n^{'}_{Cl} + ost_{Cl_n},$$

где $0 \leq ost_m < k$ и $0 \leq ost_n < k$, $0 \leq ost_{Cl_m} < k$ и $0 \leq ost_{Cl_n} < k$.

Тогда можно уменьшить размер $image$ до $m^{'}$ x $n^{'}$, а размер $class$ до $m^{'}_{Cl}$ x $n^{'}_{Cl}$.

При этом все «квадраты» k x k пикселей преобразуются в один пиксель, яркость которого – среднее арифметическое яркостей всех пикселей в этом квадрате.Оставшиеся справа и снизу пиксели отбрасываются. На рис. 2 продемонстрированы оставшиеся пиксели на $image$ (отбрасываются пиксели с координатами $(x,\ y)$, где $m^{'} \cdot k \ \leq \ x \ < m$ и $n^{'} \cdot k \ \leq \ y \ < \ n$).

При уменьшении размеров $class$ и $image$ в k^2 раз уменьшится показатель степени в оценке числа шагов решения поставленной задачи как в случае полного перебора, так и в случае метода резолюций для исчисления предикатов. Программа, реализующая поиск эталонного объекта $class$ на сложном изображении $image$,

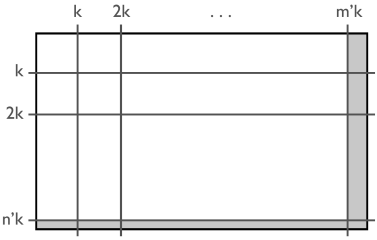

Рис. 2. «Сжатие» изображения с отбрасыванием части пикселей.

написана на Прологе. Поиск решений в Прологе основан на методе резолюций, поэтому посмотрим, как изменится оценка числа шагов решения поставленной задачи при помощи метода резолюций для исчисления предикатов:

a – максимальное количество вхождений атомарных формул в элементарные конъюнкции, составляющие описание $A(\overline{x})$, вычислялось как $a = m_{Cl} \cdot n_{Cl}$. С учетом уменьшения и m_{Cl} и n_{Cl} в k раз, получаем улучшение в k^2 раз: $a^{'} = \frac{m_{Cl}}{k} \cdot \frac{n_{Cl}}{k} = \frac{a}{k^2} = m^{'}_{Cl} \cdot n^{'}_{Cl}$.

Если распознаваемый эталонный объект *class* действительно был частью сложного изображения *image*, то каждому пикселю *class* ставился в соответствие один пиксель *image*. Поэтому одного прогона программы было достаточно, чтобы найти *class*, если он есть, или убедиться в его отсутствии. В случае с уменьшением *image* и *class* в k раз, одного прогона недостаточно, потому что 'новые пиксели' могут не совпадать у *class* и *image*. В таком случае нужно вызвать программу k^2 раз, чтобы найти эталонный объект *class* или убедиться в его отсутствии. Пример «совпадения и несовпадения новых пикселей» показан на рис. 3.

Здесь, k - степень сжатия (и, соответственно, сторона квадрата из пикслей, который преобразовывается в один новый пиксель), c_x и c_y - координаты *class* на *image*. На левом рисунке видно, что 'новый пиксель' с началом в точке $(s_x k,\ s_y k)$ лишь частично перекрывается *class*. На правом рисунке показано 'совпадение' (ситуация, если у *image* обрезать слева $(c_x - s_x k)$(пикселей и сверху $(c_y - s_y k)$ пикселей), c_{x1} и c_{y1} - координаты эталонного объекта *class* на обрезанном сложном изображении *image*. Чтобы получить 'совпадение новых пикселей' нужно совершить k^2 операций:

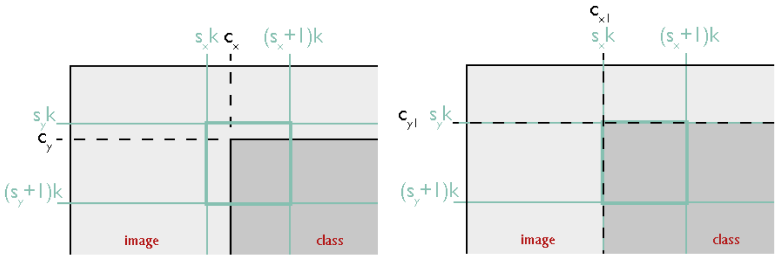

Рис. 3. Пример «совпадения и несовпадения новых пикселей».

$\forall i(0 \le i < k)\forall j(0 \le j < k)$ *{обрезать y image сверху i и слева j пикселей и запустить программу на Прологе}.*

Таким образом, можно привести новую оценку числа шагов решения задачи методом резолюций. Значение s зависело от того, насколько зашумлено изображение. После уменьшения она может измениться, может не измениться, назовем ее $s^{'}$. Значение D не изменилось: $D = 1$. Показатель степени изменился и стал равен $a^{'} = \frac{a}{k^2}$. Также появится множитель $k^2 \cdot time$, где

k^2 – количество прогонов программы,

$time$ – среднее время вызова программы на Прологе, учитывая действия пользователя.

Тогда новая оценка числа шагов примет вид:

$$(k^2 \cdot time) \cdot \left(s^{'}\right)^{\frac{a}{k^2}}.$$

5. Описание программы на языке SWI Prolog 6.0.2. SWI Prolog читает из текстового файла как описание сложного изображения *image*, так и описание эталонного объекта *class*. Предикаты, описывающие *image*, имеют вид: $p_{e_1e_2e_3e_4e_5e_6e_7e_8e_9}(i,\ j)$., где e_l - яркость l-ого пикселя в окружении пикселя с координатами $(i,\ j)$. Предикаты, описывающие *class*, имеют вид: $p_{e_1e_22_3e_4e_5e_6e_7e_8e_9}(X_i,\ Y_j)$., где e_l - яркость l-ого пикселя в окружении пикселя с координатами $(X_i,\ Y_i)$. Цель для поиска решения также содержит в себе вывод результата в текстовый файл в виде $[[x_i,\ y_j], \ldots]$.

6. Описание программы на языке Visual Prolog 7.3. Visual Prolog читает из текстового файла описание сложного изображения *image* и факт, который записан последним и будет интерпретироваться как описание эталонного объекта *class*. Предикаты, описывающие *image* имеют вид: $p(''e_1e_22_3e_4e_5e_6e_7e_8e_9'',\ ''x_i'',\ ''y_i'')$, где e_i - яркость i-ого пикселя, x_i и y_i – его координаты. Предикат, описывающий *class* имеют вид: $f([[''e_1e_22_3e_4e_5e_6e_7e_8e_9'',\ ''X_i'',\ ''Y_i'']\ldots])$., где e_i - яркость i-ого пикселя, x_i и y_i – его координаты. В процессе работы программы составляется список пар вида $(''x_i'',\ ''X_i'')$, $(''y_j'',\ ''Y_j'')\ldots$, который и является ответом на поставленную задачу. Будем в дальнейшем называть этот список *result*.

Агрумент факта f – массив массивов. Каждый внутренний массив соответствует одному пикселю. Интерпретация факта f заключается в следующем: для каждого внутреннего массива (по очереди следования во внешнем массиве), имеющего вид $[''ident'',\ ''X_i'',\ ''Y_j'']$, выполняется вызов предиката $p(''ident'',\ X,\ Y)$. Если находится факт $p(''ident'',\ ''x'',\ ''y'')$, то выполняется проверка для пары $(x,\ X_i)$ (затем аналогичная проверка должна быть выполнена для пары $(y,\ Y_j)$:

1. если в *result* нет пары $(x,\ X_i)$, то добавить эту пару к *result*;
2. если в *result* есть пара $(x,\ X_i)$, то переходим к аналогичной проверке для пары $(y,\ Y_j)$;
3. если в *result* есть пара $(x^{'},\ X_i)$, где $x^{'}$ не совпадает с x, то происходит откат.

Интерпретация заканчивается или успешно (все внутренние массивы просмотрены), или неудачно (нет фактов, удовлетворяющих какому-либо внутреннему массиву).

7. Примеры работы программ. Приведем несколько примеров входных данных и результатов работы с программой.

Пример 1. Создание описаний эталонных изображений («чистого» и «зашумленного») и сравнение количества одинаковых входящих в него предикатов.

На рис. 4 и 5 представлены эталонное изображение *class* и его копия с добавленным шумом.

Рис. 4. Эталонное изображение.

Рис. 5. Копия эталона с добавленным шумом.

Описание эталонного изображения содержит очень большое количество одинаковых предикатов, поэтому перебор при решении задачи идентификации оказывается слишком велик, и стек Пролога переполняется.

Например, для $brightness = 10$ и $smoothing = 3$ в описании «чистого» эталонного изображения $class$ предикат $p888888888(X_i,\ Y_i)$ встречается 172 раза. А в описании «зашумленного» эталонного изображения $class$ этот предикат встречается 55 раз. В подавляющем большинстве случаев количество различных имен предикатов возрастает с добавлением шума, а количество одинаковых предикатов, наоборот, уменьшается.

Пример 2. Выделение на Яндекс-карте Санкт-Петербурга заданного эталонного изображения.

На рис. 6 представлено изображение эталонного объекта $class$ $(32px \times 40px)$ и сложного изображения $image$ $(744px \times 480px)$. Результатом работы программы является изображение на рис. 7, где серым выделено место, на котором находится эталонный объект.

Пример 3. Пример работы программы с различными форматами изображений.

Была сформулирована проблема, возникающая при работе программ с изображениями, при сохранении которых использовался алгоритм сжатия с потерей качества. Приведем пример изображений, при которых эта проблема возникает. На рис. 8 представлен эталонный объект $class$ $(40px \times 34px)$ (слева) и сложное изображение $image$ $(300px \times 277px)$ (справа).

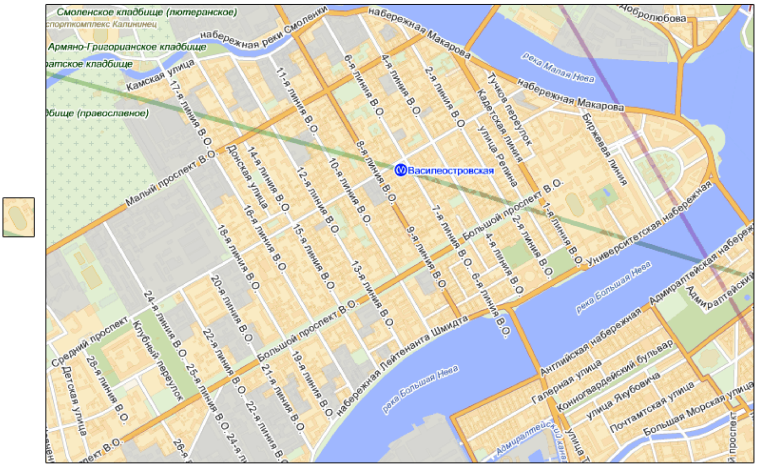

Рис. 6. Изображение эталона (слева) и сложного изображения (справа).

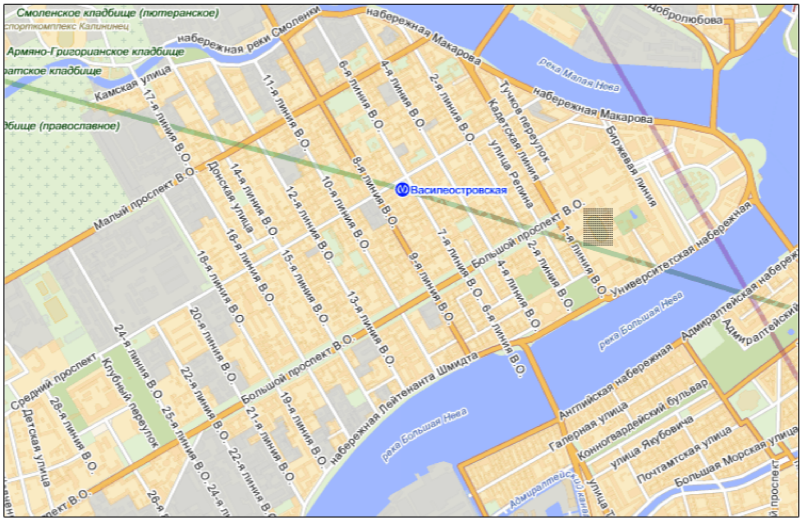

Рис. 7. Выделение эталона на сложном изображении.

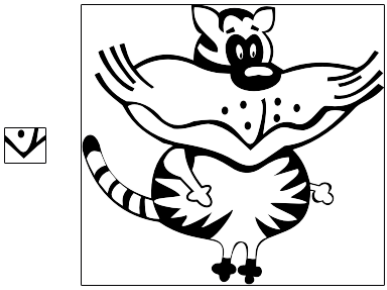

Рис. 8. Эталон и сложное изображение.

Изображения были сохранены в форматах .png и .jpg. Изображение контрастное, при сохранении в .jpg с обычными параметрами (применялся алгоритм сжатия с потерей качества) на нем появились «артефакты». Программы нашли решение только в случае, когда эти изображения были сохранены как .png.

8. Заключение. Таким образом можно выделить следующие особенности логического ядра на языке семейства Пролог:

- слишком сильная зависимость по входным данным;
- необходимость оптимизиции (прebразования, сокращения и т.п.) и правильной сортировки входных данных.

Пролог не идеально подходит для создания ядра для решения задачи идентификации. Но он может быть полезен при рассмотрении задач с «хорошим» набором входных данных.

Список литературы

[1] Косовская Т.М. Доказательства оценок числа шагов решения некоторых задач распознавания образов, имеющих логические описания // Вестн. С.-Петербург.ун-та. Сер. 1. 2007. Вып.(4) С. 82 – 90.

[2] Косовская Т.М. Некоторые задачи искусственного интеллекта, допускающие формализацию на языке исчисления предикатов, и оценки числа шагов их решения // Труды СПИИРАН, 2010. Вып. 14. С. 58 – 75.

[3] Kosovskaya T. Discrete Artificial Intelligence Problems and Number of Steps of their Solution // International Journal on Information Theories and Applications, Vol. 18, Number 1, 2011. P. 93 – 99.

[4] Visual Prolog Version 5.x. // Language Tutorial, Prolog Development Center A/S H.J.Holst Vej 3-5C, DK - 2605 Broendby, Copenhagen, Denmark

[5] Компьютерная Графика и Мультимедиа | Сетевой журнал [Электронный ресурс]. URL: http://cgm.computergraphics.ru/

[6] Strawberry Prolog [Электронный ресурс]: Professional tool for programming in Prolog. Free Light edition with many example programs in it. URL: http://www.dobrev.com/

[7] SWI-Prolog's home [Электронный ресурс]. URL: http://www.swi-prolog.org/

[8] Visual Prolog Programming Language, Compiler, IDE, Download Free Personal Edition [Электронный ресурс] : Visual Prolog is a full-featured programming environment for making commercial applications. Prolog compiler, IDE, tutorials, examples. New: Visual Prolog 7.3. URL: http://www.visual-prolog.com/

[9] JPEG - Wikipedia, the free encyclopedia [Электронный ресурс]. URL: http://en.wikipedia.org/wiki/JPEG

Косовская Татьяна Матвеевна — д.ф.-м.н., доцент; ст. научн. сотр. лаборатории информационных технологий в управлении и робототехнике СПИИРАН, профессор кафедры информатики математико-механического факультета С.-Петербургского государственного университета (СПбГУ), профессор кафедры математики С.-Петербургского государственного морского технического университета (СПбГМТУ). Область научных интересов: логический подход к решению задач искусственного интеллекта, теория сложности алгоритмов. Число научных публикаций - 72. kosovtm@gmail.com; СПИИРАН, 14-я линия В.О., д. 39, г. Санкт-Петербург, 199178, РФ; р.т. +7(812)328-0421.

Tatiana M. Kosovskaya — Doctor of Computer Science, Associate Professor; senior researcher of Laboratory of Information Technologies in Control and Robototechniques, SPIIRAS, Professor of Computer Science Chair, SPbSU, Professor of Mathematics Chair, SPbSMTU. Research area: logical approach to

the solving of Artificial Intelligence problems, theory of complexity of algorithms. Number of publications — 64. kosovtm@gmail.com; SPIIRAS, 14-th line V.O., 39, St. Petersburg, 199178, Russia; office phone +7(812)328-0421.

Власова Мария Александровна — аспирант кафедры информатики математико-механического факультета С.-Петербургского государственного университета (СПбГУ). Область научных интересов: языки логического программирования. mvlasova.spb@gmail.com; мат-мех СПбГУ, Университетский пр., д. 28, Старый Петергоф, г. Санкт-Петербург, 198504, РФ.

Maria A. Vlasova — postgraduate student of Computer Science Chair, SPbSU. mvlasova.spb@gmail.com; Math-Mech SPbSU, University av. 28, Stary Petergof, St. Petersburg, 198504, Russia.

Рекомендовано ИТУР СПИИРАН, зав. лаб. Тимофеев А.В.

Статья поступила в редакцию 14.11.2012.

РЕФЕРАТ

Косовская Т.М., Власова М.А. **Использование языков семейства Prolog для распознавания изображений.**

Для изображений на экране дисплея рассмотрена реализация задачи выделения эталоного изображения на сложном изображении средствами языков семейства Prolog. Подробно описаны особенности решения задачи при различных форматах сохранения изображения: .jpeg (.jpg), .bmp или .png. Приведён пример изображения, распознавание части которого удалось осуществить только при его задании в формате .png.

Кратко изложен логико-предметный подход к решению задач распознавания образов и приведены оценки числа шагов алгоритма распознавания изображений, основанного на поиске вывода в исчислении предикатов: Cs^a, где s – масимальное количество вхождений одного и того же исходного предиката, в терминах которых описывается сложное изображение; a – количество вхождений атомарных формул в описание эталона.

Большая размерность задачи распознавания реального изображения на экране дисплея не позволяет непосредственно применить язык Prolog для решения поставленной задачи в силу её NP-трудности.

Использование приведённых сложностных оценок позволило произвести предобработку изображения таким образом, что время решения задач существенно уменьшилось до приемлемого, не изменив при этом качества распознавания.

Для уменьшения параметра s используется описание как эталона, так и сложного изображения не попиксельно с введением одного предиката, характеризующего каждый пиксель его яркостью, а с учётом его окружения из восьми пикселей, что приводит к появлению 256^9 различных предикатов. Измерение их значений занимает столько же времени и столько же места, как и измерение значения попиксельного предиката, но их введение существенно уменьшает значение параметра s (в десятки и сотни раз) до значения s' (своего для каждого изображения).

Для уменьшения значения параметра a введено понятие «сжатия» информации в k раз. Это привело к изменению верхней оценки числа шагов алгоритма распознавания с Cs'^a до $k^2Cs'^{a/k^2}$, где a – размерность эталона.

Приведены примеры задач, решённых с помощью описанного комплекса программ. Выделение на Яндекс-карте Санкт-Петербурга заданного фрагмента изображения. Пример распознавания изображения на гладком фоне, эффективное распознавание которого возможно при его частичном «зашумлении» с целью уменьшения параметра s.

SUMMARY

Kosovskaya T.M., Vlasova M.A. **The use of Prolog language family for image recognition.**

Implementation of an etalon extraction from a complex image on the display screen problem by means of a language Prolog family is considered. Peculiarities of the problem solving while using different image formats (.jpeg (.jpg), .bmp or .png) are described. An example of an image the recognition of which was fulfilled only in rhe case of format .png is done.

The logic-objective approach to the problems of pattern recognition is shortly described and number of steps bounds for an image recognition algorithm based on deduction search in the predicate calculus are brought: Cs^a, where s is a maximal number of the same predicate occurrences in the complex image, a is the number of atomic formulas in the etalon description.

Great dimension of real image on the display screen recognition problem does not allow directly use Prolog in order to solve the mentioned problem because of its NP-hardness.

The use of the above done complexity bounds allows to make such a preprocessing of an image that the problem solving time is decreased up to acceptable one and the quality of recognition does not change.

To decrease the parameter s the description of the etalon (as well as the complex image) is made not in the terms of one predicate characterizing every pixel by means of its brightness, but by means of 256^9 predicates which take in acount the pixel itself and 8 pixels from its enclosing. The measurement of their values takes the same time and memory as the measurement of the pixel-by-pixel predicate value, but their introduction essentially decreases the value of parameter s (in tens and hundreds times) up to the value s' (its own for every image).

To decrease the parameter a the notion of information «compression» in k times is introduced. It leads to the change of the recognition algorithm number of steps upper bound from Cs'^a up to $k^2Cs'^{a/k^2}$ where a is an etalon dimension.

Examples of the problems solved with the use of the described program complex are done. An extraction from the Yandex-map of St.Petersburg of an image fragment. An example of the image situated on the smooth background for which its effective recognition is possible only after its partial «shading» in order to decrease the parameter s.