УДК 519.7

С.Ю. Шабанов-Кушнаренко, Кудхаир Абед Тамер

Харьковский национальный университет радиоэлектроники, Харьков

РАЗРАБОТКА МЕТОДА ФОРМИРОВАНИЯ ПРЕДИКАТНЫХ МОДЕЛЕЙ ПРОТОТИПОВ СТРУКТУРИРОВАННЫХ ОБЪЕКТОВ

Разработанный метод предназначен для формирования набора прототипов структурированных объектов. В качестве исходных данных для метода используются модели несвязанных подмножеств структурируемых объектов. Каждое подмножество содержит объекты одного класса. При построении прототипа сравнение вершин и дуг графовых моделей из обучающего набора и их возможное объединение выполняется на основе применения правил сравнения свойств соответствующих понятий.

Ключевые слова: *алгебра конечных предикатов, предикат, структурированный объект, прототип, представление знаний.*

Введение

В последнее время значительно вырос интерес к представлению структурированных объектов в виде графов в областях, связанных с распознаванием и обработкой изображений, а также машинным обучением. Такое представление обеспечивает удобные возможности для отображения элементов исследуемых объектов, связей между этими элементами, а также свойств указанных элементов. В частности, графовое представление структурированных объектов используется при распознавании разнообразных текстовых символов в различном начертании, распознавании изображений, аутентификации людей по отпечаткам пальцев и изображению лица, анализе документов, интеллектуальном анализе данных и процессов [1 – 6].

Преимущество графового представления структурированных объектов состоит в возможности быстрой адаптации графовой модели с учетом размера и сложности исходного объекта.

Недостатки данного представления по сравнению с традиционным векторным представлением свойств структурированных объектов определяются необходимостью адаптации существующих методов интеллектуального анализа данных, машинного обучения и распознавания. При этом вычислительная сложность адаптированных методов значительно повышается [7].

Теория прототипов базируется на использовании абстрактного структурированного прототипа для хранения образа объекта. В отличие от сравнения с эталоном, распознавание объекта по прототипу осуществляется на основе сходства – т.е. выполняется неточное сравнение. Прототип формализует сходство структур или форм объектов. Экспериментальные исследования показали возможность сопоставления прототипа и искаженного объекта [8].

Основные подходы к построению прототипов:

- прототип представляет собой среднее всех экземпляров класса [8];
- прототип отражает наиболее часто встречающееся сочетание признаков [9].

В первом случае прототип рассматривается как абстракция некоторого объекта, обладающая средними значениями признаков. Во втором случае существенной является информация о соотношении признаков. Обе теории получили экспериментальное подтверждение, поэтому можно считать, что при формировании образа объекта важны как усредненные значения признаков, так и соотношения этих значений.

Постановка задачи

Разработанный метод предназначен для формирования набора прототипов структурированных объектов. Формализация полученных прототипов выполняется на основе представленной в работе [10] предикатной модели.

В качестве исходных данных для метода используются модели несвязанных подмножеств структурируемых объектов. Каждое подмножество содержит объекты одного класса. При выполнении разбиения можно использовать предварительно определенные набор понятий, которые описывают объект.

Общая идея метода заключается в следующем: необходимо собрать обобщенный граф, охватывающий максимальное количество элементов обучающего множества моделей структурированных объектов и представленный в виде системы предикатов. При построении прототипа сравнение вершин и дуг графовых моделей из обучающего набора и их возможное объединение выполняется на основе применения правил сравнения свойств соответствующих понятий. При исключении необходимо не потерять общности (т.е. удовлетворить ограничению полноты) и точности (т.е. удовлетворить ограничению непротиворечивости).

Основная часть

Проверить идентичность вершин исходных графов для предложенной в предыдущем параграфе модели можно путем сопоставления характеризующих их понятий. При объединении вершин сравниваются предикаты понятий. Поскольку понятия определены на основе их свойств, то сравниваются наборы свойств и взаимосвязи между ними. Вершины объединяются в следующих случаях:

– характеризующие вершины предикаты полностью совпадают;

– предикат для i – й вершины может быть получен дополнением предиката для j – й вершины;

– предикаты совпадают частично.

В первом случае понятия, определяемые предикатами, полностью идентичны. Это означает, что совпадает набор используемых переменных, отражающих свойства понятия, логические взаимосвязи между свойствами, наборы значений переменных, входящих в предикат.

$$\begin{gathered}\forall x_{i1}^{p} \in A_{i1}^{p*}, x_{i2}^{p} \in A_{i2}^{p*}, ..., x_{in}^{p} \in A_{in}^{p*}, p = \overline{1,k};\\ \forall x_{j1}^{p} \in A_{j1}^{p*}, x_{j2}^{p} \in A_{j2}^{p*}, ..., x_{jn}^{p} \in A_{jn}^{p*}, p = \overline{1,k};\\ C_i^p(x_{i1}^p, x_{i2}^p, ..., x_{in}^p) = C_j^p(x_{j1}^p, x_{j2}^p, ..., x_{jn}^p),\end{gathered} \quad (1)$$

Во втором случае возможны три отличающихся варианта поглощения:

– один из предикатов содержит подмножество переменных второго:

$$C_i^p(x_{i1}^p, x_{i2}^p, ..., x_{in}^p) \subset C_j^p(x_{j1}^p, x_{j2}^p, ..., x_{jm}^p),\ n < m, \quad (2)$$

где C_i^p - предикат p-го понятия, формализующий вершину V_i^{p**} специализированного графа G_i^{**}, а C_j^p - предикат p-го понятия, формализующий вершину V_j^{p**} специализированного графа G_j^{**}. n и m - количество переменных в предикатах C_i^p и C_j^p, формализующих свойства p-го понятия для вершин V_i^{p**} и V_j^{p**} соответственно графов G_i^{**} и G_j^{**}.

– один из предикатов по некоторым переменным p-го понятия содержит подмножество логических операций с этими переменными второго предиката:

$$\begin{gathered}C_i^p(x_{i1}^p, x_{i2}^p, ..., x_{in}^p) \subset C_j^p(x_{j1}^p, x_{j2}^p, ..., x_{jn}^p);\\ x_{i1}^p \in A_{i1}^{p**}, x_{i2}^p \in A_{i2}^{p**}, ..., x_{in}^p \in A_{in}^{p**};\\ x_{j1}^p \in A_{j1}^{p**}, x_{j2}^p \in A_{j2}^{p**}, ..., x_{jn}^p \in A_{jn}^{p**};\\ A_{i1}^{p**} \subseteq A_{j1}^{p**}, A_{i2}^{p**} \subseteq A_{j2}^{p**}, ..., A_{in}^{p**} \subseteq A_{jn}^{p**}.\end{gathered} \quad (3)$$

В отличие от варианта поглощения (2), случай (3) допускает сокращение подмножеств логических операций по некоторым переменным p-го понятия, формализующих свойства этого понятия для графов G_i^{**} и G_j^{**}, что определяется сокращением областей определения этих переменных:

$$A_{i1}^{p**} \subseteq A_{j1}^{p**}, A_{i2}^{p**} \subseteq A_{j2}^{p**}, ..., A_{in}^{p**} \subseteq A_{jn}^{p**}.$$

Сокращение подмножества логических операций по некоторой переменной p-го понятия означает, что таблицы истинности предикатов C_i^p и C_j^p не совпадают, но выполняется свойство поглощения логических операций p-го понятия: единицы таблицы истинности предиката C_i^p покрываются единицами предиката C_j^p.

– один из предикатов содержит подмножество переменных и подмножество логических операций с этими переменными второго предиката:

$$\begin{gathered}C_i^p(x_{i1}^p, x_{i2}^p, ..., x_{in}^p) \subset C_j^p(x_{j1}^p, x_{j2}^p, ..., x_{jm}^p),\ n < m,\\ x_{i1}^p \in A_{i1}^{p**}, x_{i2}^p \in A_{i2}^{p**}, ..., x_{in}^p \in A_{in}^{p**},\\ x_{j1}^p \in A_{j1}^{p**}, x_{j2}^p \in A_{j2}^{p**}, ..., x_{jn}^p \in A_{jm}^{p**},\\ A_{i1}^{p**} \subseteq A_{j1}^{p**}, A_{i2}^{p**} \subseteq A_{j2}^{p**}, ..., A_{in}^{p**} \subseteq A_{jn}^{p**}.\end{gathered} \quad (4)$$

В отличие от вариантов поглощения (2) и (3), случай (4) допускает оба варианта поглощения - по количеству переменных, формализующих свойства p-го понятия, и по множествам допустимых логических операций с этими переменными.

Рассмотрим третий случай, когда предикаты совпадают частично. Здесь также возможны три варианта, аналогичные рассмотренным выше, см. (2)-(4):

– множества переменных $\{x_{i1}^p, x_{i2}^p, ..., x_{in}^p\}$ и $\{x_{j1}^p, x_{j2}^p, ..., x_{jm}^p\}$ предикатов C_i^p и C_j^p пересекаются:

$$\{x_{i1}^p, x_{i2}^p, ..., x_{in}^p\} \cap \{x_{j1}^p, x_{j2}^p, ..., x_{jm}^p\} \neq \varnothing. \quad (5)$$

– множества областей определения некоторых из переменных $\{x_{i1}^p, x_{i2}^p, ..., x_{in}^p\}$ и $\{x_{j1}^p, x_{j2}^p, ..., x_{jm}^p\}$ предикатов C_i^p и C_j^p пересекаются:

$$\begin{gathered}A_{i1}^{p**} \cap A_{j1}^{p**} \neq \varnothing,\ \ A_{i2}^{p**} \cap A_{j2}^{p**} \neq \varnothing, ...,\\ A_{in}^{p**} \cap A_{jn}^{p**} \neq \varnothing.\end{gathered} \quad (6)$$

– множества переменных $\{x_{i1}^p, x_{i2}^p, ..., x_{in}^p\}$ и $\{x_{j1}^p, x_{j2}^p, ..., x_{jm}^p\}$ предикатов C_i^p и C_j^p пересекаются и множества областей определения некоторых из этих общих переменных предикатов C_i^p и C_j^p пересекаются:

$$\begin{gathered}\{x_{i1}^p, x_{i2}^p, ..., x_{in}^p\} \cap \{x_{j1}^p, x_{j2}^p, ..., x_{jm}^p\} = \{x_{is}^p, ..., x_{it}^p\};\\ A_{is}^{p**} \cap A_{js}^{p**} = A_s^{p**}, ..., A_{it}^{p**} \cap A_{jt}^{p**} = A_t^{p**}.\end{gathered} \quad (7)$$

Правила объединения вершин графов из множества $\{G_i^j\}$ подразделяются на следующие группы:

– правило объединения вершин с совпадающими понятиями;

– правила объединения вершин с поглощением понятий;

– правила объединения вершин с пересечением понятий.

Первая группа содержит только одно правило – при совпадении понятий вершины графов из множества $\{G_i^{**}\}_{i=1}^{h}$, (h - количество специализированных графов G_i^{**}) объединяются в обобщающем графе G_i^*:

$$V_i^{p**} \in G_i^{**}, V_j^{p**} \in G_j^{**},$$
$$V_i^{p**} \cup V_j^{p**} = V_{ij}^{p*}, V_{ij}^{p*} \in G_{ij}^*. \quad (8)$$

Вторая группа содержит ряд правил:

– правило поглощения понятий в случае, если один из предикатов содержит надмножество переменных, их значений, логических операций с этими переменными:

$$\begin{aligned} &C_i^p \to V_i^{p**} \in G_i^{**}, C_j^p \to V_j^{p**} \in G_j^{**};\\ &\{x_{i1}^p, x_{i2}^p, ..., x_{in}^p\} \subseteq \{x_{j1}^p, x_{j2}^p, ..., x_{jm}^p\}, n < m;\\ &C_i^p(x_{i1}^p, x_{i2}^p, ..., x_{in}^p) \subset C_j^p(x_{j1}^p, x_{j2}^p, ..., x_{jm}^p);\\ &C_{ij}^p(x_{j1}^p, x_{j2}^p, ..., x_{jm}^p) \to V_{ij}^{p*} \in G_{ij}^*. \end{aligned} \quad (9)$$

– правило поглощения понятий в случае, если один из предикатов содержит надмножество переменных, логических операций с этими переменными:

$$\begin{aligned} &x_{i1}^p \in A_{i1}^{p**}, x_{i2}^p \in A_{i2}^{p**}, ..., x_{in}^p \in A_{in}^{p**};\\ &x_{j1}^p \in A_{j1}^{p**}, x_{j2}^p \in A_{j2}^{p**}, ..., x_{jm}^p \in A_{jm}^{p**};\\ &A_{i1}^{p**} \subseteq A_{j1}^{p**}, A_{i2}^{p**} \subseteq A_{j2}^{p**}, ..., A_{in}^{p**} \subseteq A_{jn}^{p**};\\ &C_{ij}^p(x_{j1}^p, x_{j2}^p, ..., x_{jm}^p) \to V_{ij}^{p*} \in G_{ij}^*. \end{aligned} \quad (10)$$

Как видно из (10), в данном случае предикат понятий для вершины обобщенного графа будет содержать надмножество переменных и логических операций первого предиката, а также объединение подмножеств значений переменных обоих предикатов.

Совпадение подмножеств переменных и различие логических связей между ними означает различную структуру свойств пары понятий, что может свидетельствовать о их противоречивости. В этом случае при объединении формируется обобщенное понятие, представляющее собой логическое произведение базовой пары понятий:

$$\begin{aligned} &C_i^p \to V_i^{p**} \in G_i^{**}, C_j^p \to V_j^{p**} \in G_j^{**};\\ &\{x_{i1}^p, x_{i2}^p, ..., x_{in}^p\} \neq \{x_{j1}^p, x_{j2}^p, ..., x_{jm}^p\};\\ &A_{i1}^{p**} \neq A_{j1}^{p**}, A_{i2}^{p**} \neq A_{j2}^{p**}, ..., A_{in}^{p**} \neq A_{jn}^{p**};\\ &C_i^p(x_{i1}^p, x_{i2}^p, ..., x_{in}^p) \neq C_j^p(x_{j1}^p, x_{j2}^p, ..., x_{jm}^p);\\ &C_{ij}^p(x_{j1}^p, x_{j2}^p, ..., x_{jm}^p) =\\ &= C_i^p(x_{i1}^p, x_{i2}^p, ..., x_{in}^p) \wedge C_j^p(x_{j1}^p, x_{j2}^p, ..., x_{jm}^p). \end{aligned} \quad (11)$$

Иллюстративный пример для поглощения свойств понятий. Первое понятие, характеризующее болезни легких, и заданное унарным предикатом определено через следующий список значений переменной: «абсцесс лёгкого И альвеолит И гемоторакс И плеврит И пневмоторакс».

Второе понятие из этой же области, также заданное предикатом через такой список значений переменной: «абсцесс лёгкого И альвеолит И гемоторакс И плеврит И пневмоторакс И силикоз И силикатоз».

Третья группа содержит следующие правила (в случае вершин с пересечением понятий):

– правило объединения понятий по переменным;

– правило объединения понятий по переменным и их значениям;

В 1-м случае выполняется сравнение множеств переменных для пары реализаций понятия C^p в графах G_i^{**} и G_j^{**} - предикатов C_i^p и C_j^p. В том случае, если множества переменных предикатов C_i^p и C_j^p пересекаются, множество переменных для реализации понятия C^p предикатом C_{ij}^p вершины $V_{ij}^{p*} \in G_i^*$ задаем объединением множеств переменных предикатов C_i^p и C_j^p:

$$\begin{aligned} &C_i^p \to V_i^{p**} \in G_i^{**}, C_j^p \to V_j^{p**} \in G_j^{**};\\ &\{x_{i1}^p, x_{i2}^p, ..., x_{in}^p\} \neq \{x_{j1}^p, x_{j2}^p, ..., x_{jm}^p\};\\ &\{x_{i1}^p, x_{i2}^p, ..., x_{in}^p\} \cap \{x_{j1}^p, x_{j2}^p, ..., x_{jm}^p\} = \{x_{is}^p, ..., x_{it}^p\};\\ &C_{ij}^p(x_{i1}^p, x_{i2}^p, ..., x_{in}^p, x_{is}^p, ..., x_{it}^p) \to V_{ij}^{p*} \in G_i^*. \end{aligned} \quad (12)$$

Во втором случае, если пересекаются и множества переменных предикатов C_i^p и C_j^p и области определения некоторых из этих переменных, то, как и в первом случае, множество переменных для реализации понятия C^p предикатом C_{ij}^p (соответствующем вершине $V_{ij}^{p*} \in G_i^*$) задаем объединением множеств переменных предикатов C_i^p и C_j^p. Кроме

того, учитывая различия в областях определения переменных предикатов C_i^p и C_j^p, объединяем различающиеся области определения:

$$\begin{aligned} &C_{ij}^p(x_{i1}^p, x_{i2}^p, ..., x_{in}^p, x_{is}^p, ..., x_{it}^p) \to V_{ij}^{p*} \in G_i^*; \\ &A_{is}^{p**} \cap A_{js}^{p**} = A_s^{p*}, ..., A_{it}^{p**} \cap A_{jt}^{p**} = A_t^{p*}; \\ &x_{i1}^p \in A_{i1}^{p*}, x_{i2}^p \in A_{i2}^{p*}, ..., x_{in}^p \in A_{in}^{p*}; \\ &x_{is}^p \in A_s^{p*}, ..., x_{it}^p \in A_t^{p*}. \end{aligned} \quad (13)$$

При наличии отличающихся логических связей между переменными поглощения и слияния предикатов не происходит. Действительно, в данном случае мы наблюдаем пересечение переменных понятий, а также различия между логическими связями между переменными, что свидетельствует об отличии понятий, представляющих вершины исходного графа:

$$\begin{aligned} &x_{i1}^p \in A_{i1}^{p**}, x_{i2}^p \in A_{i2}^{p**}, ..., x_{in}^p \in A_{in}^{p**}; \\ &x_{jl}^p \in A_{jl}^{p**}, x_{j2}^p \in A_{j2}^{p**}, ..., x_{jn}^p \in A_{jm}^{p**}; \\ &A_{il}^{p**} \neq A_{jl}^{p**}, A_{i2}^{p**} \neq A_{j2}^{p**}, ..., A_{in}^{p**} \neq A_{jn}^{p**}; \\ &C_i^p(x_{i1}^p, x_{i2}^p, ..., x_{in}^p) \neq C_j^p(x_{jl}^p, x_{j2}^p, ..., x_{jm}^p). \end{aligned} \quad (14)$$

Разработанный метод включает в себя следующие основные шаги:

1.Инициализация множества M прототипов M_i.

2. Установка индекса начального прототипа i.

3.Выделение подмножеств графов $\{G_i^j\}'$, представляющих модели i – класса объектов.

4. Проверка выбранного подмножества. Если список пуст - $|\{G_i^j\}'| = 0$, то переход к этапу 8.

4. Формирование прототипа M_i путем объединения графов отдельных объектов G_i^j в обобщенный граф G_i^*.

5. Проверка условия полноты для полученного прототипа. Если условие выполняется, то перейти к этапу 6, иначе – этапу 8.

6. Дополнение множества M вновь созданным прототипом M_i.

7. Удаление подмножества $\{G_i^j\}'$.

8. Переход к формированию следующего прототипа: $i = i + 1$.

9. Переход к этапу 3.

Метод начинает работу с пустого списка прототипов и на основе обработки набора структурируемых объектов пошагово формирует список прототипов. После нахождения очередного прототипа из множества структурируемых объектов удаляются модели тех объектов, из которых сформирован прототип. Затем процесс повторяется для нового прототипа. В том случае, если прототип не может быть получен из заданного набора объектов (т.е. не выполняются условия полноты и непротиворечивости), то метод переходит к формированию следующего прототипа. После исчерпания списка доступных для обработки моделей объектов метод завершает свою работу.

В результате работы алгоритма прототипу M_i будет соответствовать подмножество объектов O_i, прототипу M_j - подмножество объектов O_j и т.д. Каждый из прототипов M_i представлен обобщенными предикатными моделями G_i^*, обобщающими характеристики исходных графов G_i^j. Также при небольшой модификации метода в состав прототипа M_i могут входить специализированные предикатные модели G_i^{**}. Специализированные модели G_i^{**} определяют подкласс объектов O_i^{**}, т.е. G_i^* задается на основе общих понятий, а G_i^{**} - уточненных. Уточнение понятий в рамках предикатной модели означает, что из всех возможных наборов логически связанных свойств элементов модели G_i^* выделены подмножества, ограничивающие применение исходных понятий.

Сформируем критерии полноты и непротиворечивости при неточном сравнении моделей.

Критерий оценки полноты должен обеспечивать возможность максимально охватить набор объектов из обучающей выборки при формировании прототипа. Пусть имеем подмножество моделей объектов O_i в обучающей выборке.

Тогда критерий полноты для полученного прототипа состоит в достижении максимального обобщения, т.е. необходимо, чтобы минимальная часть объектов обучающей выборки (в идеале – все), которая охвачена обобщенным графом G_i^*, относилась к прототипу M_i. Иными словами, наибольшее количество элементов подмножества $\{G_i^j\}'$ должны быть охвачены обобщенным графом:

$$\begin{aligned} &K = \min_j \frac{|\{G^j\}'|}{|\{G_i^j\}'|}; \\ &K \le \varepsilon; \\ &G_i^* \models G_i^j, G_i^* \models G^j, \forall G_i^j, G^j, \end{aligned} \quad (15)$$

где $|\{G_i^j\}|$ - количество объектов O_j из обучающей выборки O, каждый из которых представлен графами G_i^j и включен в прототип M_i;

$|\{G^j\}|$ - количество объектов из обучающей выборки, которые не включены в прототип;

ε - порог для определения общности графа; на практике целесообразно задавать порог близким к нулю.

Условие непротиворечивости прототипов строго выполняется для каждого прототипа на шаге 8 метода путем удаления из обучающего множества всех объектов $O_i^j \in O_i^*$, которые были использованы при построении M_i.

Теперь детализируем этап 4 разработанного метода - формирование прототипа структурированного объекта средствами алгебры конечных предикатов.

На данном этапе выполняются пошаговое дополнение прототипа путем уточнения прототипа по правилам (9)-(14). Шаги данного этапа повторяются до тех пор, пока в прототип не будут внесены свойства всех графов из обучающего набора для данного класса объектов.

Правила (9)-(13) позволяют доопределить понятия в составе прототипа путем уточнения задающих их предикатов.

Правило (14) предназначено для добавления нового элемента в прототип. Данное правило задает ситуацию, когда для рассматриваемого понятия, отражающего элемент текущего объекта из обучающего набора, не существует идентичного понятия в прототипе объекта. Поэтому новое понятие добавляется в прототип.

Выводы

Разработан метод формирования прототипов структурированных объектов. Метод основывается на объединении моделей отдельных экземпляров объектов данного класса с использованием правил объединения понятий. Применение таких правил позволяет учитывать как общность структуры объектов с учетом свойств представляющих их понятий, так и взаимосвязи между свойствами этих понятий.

Список литературы

1. Ratha N. Automatic Fingerprint Recognition Systems [Текст] / Nalini Ratha, Ruud Bolle // Springer Science & Business Media, 2007. – 458 p.

2. Maio D. A structural approach to fingerprint classification [Текст] / D. Maio, D. Maltoni // Proceedings of 13th International Conference on Pattern Recognition, Vienna, Austria, 1996. – P. 578-585.

3. Bunke H. Detection of abnormal change in time series of graphs [Текст] / H. Bunke, M. Kractzl, P. Shoubridge, W.D. Wallis // Journal of Interconnection Networks, 3(1 -2). 2002. – P. 85-101.

4. Schenker A. Classification of web documents using graph matching [Текст] / A. Schenker. M. Last, II. Bunke, A. Kandel // International Journal of Pattern Recognition and Artificial Intelligence 18(3). 2004. – P. 475-496.

5. Holder L.B. Graph-based relational learning: Current and future directions [Текст] / L.B. Holder, D.J. Cook // SIGKDD Explorations, Special Issue on Multirelational Data Mining, 5(1). 2003. – P. 90-93.

6. Mining Graph Data / D. Cook and L. Holder, editors// Wiley-Interscience, 2007. – 51 p.

7. Aggarwal Charu. Managing and Mining Graph Data Editors: [Текст] / Charu Aggarwal, Haixun Wang // Springer Science & Business Media, 2010. – 621 p.

8. Posner M.I. Perceived distance and the classification of distorted patterns [Текст] / M.I. Posner, R. Goldsmith, K.E. Welton // Journal of Experimental Psychology, 1967. – Vol. 73, – P. 28–38.

9. Neumann P.G. Visual prototype formation with discontinuous representation of dimensions of variability. [Текст] / P.G. Neumann. – Memory & Cognition, 1977. – 5(2), – P. 187-197.

10. Кудхаир Абед Тамер. Разработка предикатной модели прототипа структурированного объекта в базе знаний / Кудхаир Абед Тамер // Системи обробки інформації. – Х.: ХУПС, 2015. – Вип. 8 (133). – С. 68-71.

Поступила в редколлегию 21.05.2015

Рецензент: д-р техн. наук, проф. С.Ф. Чалый, Харьковский национальный университет радиоэлектроники, Харьков.

РОЗРОБКА МЕТОДУ ФОРМУВАННЯ ПРЕДИКАТНИХ МОДЕЛЕЙ ПРОТОТИПІВ СТРУКТУРОВАНИХ ОБ'ЄКТІВ

С.Ю, Шабанов-Кушнаренко, Кудхаир Абед Тамер

Розглянуто метод, призначений для формування набору прототипів структурованих об'єктів. В якості вихідних даних для методу використовуються моделі незв'язаних підмножин структуровані об'єктів. Кожна підмножина містить об'єкти одного класу. При побудові прототипу порівняння вершин і дуг графових моделей з навчального набору виконується на основі застосування правил порівняння властивостей відповідних понять.

Ключові слова: *алгебра скінченних предикатів, предикат, структурований об'єкт, прототип, подання знань.*

DEVELOPMENT OF THE PREDICATE MODEL OF STRUCTURED OBJECTS PROTOTYPE FORMING METHODS

S.Yu. Shabanov-Kushnarenko, Kudhair Abed Tamer

A method for generating a set of prototypes structured objects. The initial data for the method used models unrelated subsets of structured objects. Each subset contains objects of a class. When building a prototype comparison vertices and edges of graph models from the training set is performed based on the application of the rules of comparing the properties of the corresponding concepts.

Keywords: *algebra of finite predicate, the predicate, structured object prototype, knowledge representation.*