УДК 519.71

О. В. ШУТ

ОЦЕНКИ СЛОЖНОСТИ РЕШЕНИЯ ЗАДАЧ РАСПОЗНАВАНИЯ С ОБУЧЕНИЕМ В КЛАССЕ АЛГОРИТМОВ ИНДУКТИВНОЙ РЕЗОЛЮЦИИ

Рассматриваются три алгоритма решения задач распознавания образов, называемые алгоритмами индуктивной резолюции, для случая, когда все признаки объектов принимают конечное количество значений: алгоритм, основанный на методе объектных резолюций, алгоритм распознавания, использующий функцию близости объектов, и комбинированный алгоритм. Подсчитана вычислительная сложность этих алгоритмов и показано, что сложность алгоритма объектных резолюций и комбинированного алгоритма прямо пропорциональна третьей степени количества объектов, построенных этими алгоритмами. Для уменьшения времени работы алгоритмов предложены модификации всех трех алгоритмов, использующие идею метода ветвей и границ. Приведены оценки сложности модифицированных алгоритмов. Показано, что с помощью предлагаемой модификации время работы алгоритмов индуктивной резолюции уменьшается в экспоненциальной зависимости от структуры информации о рассматриваемом объекте.

Ключевые слова: распознавание образов; метод резолюций; вычислительная сложность.

The paper considers three algorithms for solving pattern recognition problems in the case when all signs of objects have a finite quantity of values: the first algorithm bases on object resolution method, the second one is a recognition algorithm which uses a closeness function of objects, and the third one combines these two algorithms. These algorithms are called inductive resolution algorithms. The computational complexity of all three algorithms is calculated. The complexity of the object resolution and combined algorithms is shown to be directly proportional to the cube of the quantity of objects built by these algorithms. In order to reduce running time of algorithms, modifications, which are based on the idea of branch and bound method, are suggested for all three algorithms. Complexities of modified algorithms are estimated. It is shown that, due to the suggested modification, the running time of inductive resolution algorithms decreases exponentially depending on the structure of information of the object which is being considered.

Key words: pattern recognition; resolution method; computational complexity.

В [1] предложена схема использования метода резолюций для решения задач распознавания образов с обучением. Для этого разработан аналог метода резолюций, работающий с объектами в конечномерном пространстве значений признаков, и построен алгоритм A_1, названный алгоритмом объектных резолюций. Построен также комбинированный алгоритм A_C, объединяющий алгоритм A_1 с алгоритмом распознавания A_2, описанным в [2] и названным в [3] алгоритмом индуктивной резолюции. Комбинация A_C также реализует индуктивный вывод, и поэтому A_C можно отнести к классу алгоритмов индуктивной резолюции.

Главной идеей алгоритмов указанного класса является перебор объектов, выводимых методом объектных резолюций из заданного множества либо составляющих обучающую выборку. Поэтому при больших размерах выборок возникает необходимость сократить время работы этих алгоритмов. С этой целью предлагается использовать технику, описанную в [4], где введены оценки близости значений признаков для уменьшения времени работы алгоритма A_2 путем применения метода ветвей и границ. В данной статье рассматриваются модификации всех данных алгоритмов, основанные на этом методе, приведены оценки их вычислительной сложности и показано, что сложность алгоритмов индуктивной резолюции может быть уменьшена.

1. Основные обозначения. Ниже используются понятия из [1–4]. Рассматривается задача распознавания образов Z, в которой X обозначает множество объектов, $X_1, ..., X_l$ – классы, $X^0 \subset X$ – обучающую выборку. Потребуются также некоторые дополнительные определения и обозначения: объект – отображение $p: s_1 \times ... \times s_n \to D^n$; $S = \{s_1, ..., s_n\}$ – множество признаков, порядок которых зафиксирован; D – конечное множество значений признаков; $k = |D|$; D_j^p – множество значений признака s_j объекта p, где $D_j^p \subset D$, $D_j^p \neq \varnothing$, $j = \overline{1, n}$. Для описания объектов используется кодировка c, где значение признака s_j объекта p представлено в виде k-мерного вектора:

$$c\left(D_j^p\right) = \left(d_{j1}^p, ..., d_{jk}^p\right), \text{ где } d_{ji}^p = \begin{cases} 1, (i-1) \in D_j^p, \\ 0, (i-1) \notin D_j^p. \end{cases}$$

Объект p, обладающий n признаками, представлен в виде kn-мерного вектора $c(p) = \left(c\left(D_1^p\right)\right), ..., \left(c\left(D_n^p\right)\right)$. Кроме того, рассмотрим операцию построения объекта $r = Or_h(p_1, p_2)$, где r называется объектной резольвентой:

$$D_j^r = \begin{cases} D_j^{p_1} \bigcup D_j^{p_2}, \ j = h, \\ D_j^{p_1} \bigcap D_j^{p_2}, \ j \neq h \end{cases} \text{ или в кодировке } c\text{: } c\left(D_j^r\right) = \begin{cases} c\left(D_j^{p_1}\right) \vee c\left(D_j^{p_2}\right), \ j = h, \\ c\left(D_j^{p_1}\right) \wedge c\left(D_j^{p_2}\right), \ j \neq h. \end{cases}$$

2. Постановка задачи. Оценим сложность алгоритмов, предложенных для решения Z. Для этого подсчитаем сложность операций, которые используются в этих алгоритмах.

1) *Сравнение двух объектов*. При использовании кодировки c объекты неравны, если у них не совпадает значение хотя бы одной компоненты из kn. Поэтому необходимо сравнить попарно kn компонент. Отсюда следует, что сложность операции равна $O(kn)$.

2) *Построение объектной резольвенты*. Поскольку $c\left(D_j^{p_1}\right)$ и $c\left(D_j^{p_2}\right)$ – k-мерные векторы, то для вычисления значения одного признака нужно совершить k операций над компонентами векторов. Отсюда следует, что сложность операции равна $O(kn)$.

3) *Проверка существования объекта*. Если $D_j^p = \varnothing$, то p не обладает признаком s_j и не рассматривается в рамках задачи Z. Поэтому при построении r нужно сравнить $c\left(D_j^p\right)$ с (0, …, 0) для $j = \overline{1, k}$. Отсюда следует, что сложность операции также равна $O(kn)$.

Исходя из этих оценок, нетрудно подсчитать, что вычислительная сложность алгоритмов A_1, A_2 и A_C составляет $O\left(kQ^3n^2\right)$, $O\left(kq_0n^2\right)$ и $O\left(kQ^3n^2\right)$ соответственно (см. табл. 1–3), где $q_0 = \left|X^0\right|$; Q – общее количество построенных объектов. Предполагается, что множество объектов хранится в виде списка.

Заметим, что k, n и q_0 зависят только от исходных данных задачи Z, т. е. зафиксированы. В худшем случае алгоритмы A_1 и A_C могут построить все множество X, поэтому возникает необходимость уменьшить число объектов, обрабатываемых этими алгоритмами. Исходя из этих соображений, предлагается уменьшить сложность алгоритмов путем уменьшения Q. Это и является основной целью данной работы и представляет собой суть предложенных ниже модификаций в классе алгоритмов индуктивной резолюции.

3. Модификация алгоритма A_1. Предлагаемая модификация алгоритма A_1 основана на следующей идее: будем рассматривать признаки по очереди и после рассмотрения каждого признака удалять те объекты, которые заведомо не могут быть использованы при выводе x из X^0. Для этого введем функцию близости двух объектов по признаку s_j, а также функцию близости объекта к множеству объектов по признаку s_j:

$$\mu_j^1(x, p) = \begin{cases} 1, D_j^x \subset D_j^p, \\ 0, D_j^x \not\subset D_j^p, \end{cases} \quad \mu_j^1(x, Y) = \max_{p \in Y}\left\{\mu_j(x, p)\right\}, \text{ где } Y \subset X.$$

Пусть $L = \{1, ..., l\}$ – множество номеров классов, $X_i^0 = X_i \cap X^0$. Разобьем множество L на l подмножеств, содержащих номера классов. Функция μ_j^1 будет играть роль верхней оценки принадлежности объекта x текущему классу. Рассмотрим следующий алгоритм:

Шаг 1. Введем множества $Y_i = X_i^0$, $L = \{1, ..., l\}$. Для всех n признаков выполним шаги 2–8. Если все признаки уже рассмотрены, перейдем к шагу 9.

Шаг 2. Выберем признак s_j. Для каждого класса X_i, где $i \in L$, выполним шаги 3–5.

Шаг 3. Выберем из Y_i нерассмотренную пару объектов (p_1, p_2). Если все пары объектов уже рассматривались, то возвращаемся на шаг 2.

Шаг 4. Вычислим $r = Or_j(p_1, p_2)$. Если $\exists j\ D_j^r = \varnothing$, то возвращаемся на шаг 3.

Шаг 5. Если $r \notin Y_i$, то добавим r в Y_i: $Y_i := Y_i \cup \{r\}$. Возвращаемся на шаг 3.

Шаг 6. Вычислим $\mu_j^1(x, Y_i)$ для всех $i \in L$.

Шаг 7. Удалим из L те номера классов i, для которых $\mu_j^1(x, Y_i) = 0$.

Шаг 8. Удалим из Y_i те объекты p, для которых $\mu_j^1(x, p) = 0$.

Шаг 9. Алгоритм завершает работу.

Обозначим этот новый алгоритм через A_1^B. Его результаты интерпретируются следующим образом: если после завершения работы алгоритма $L = \varnothing$, то с помощью данного алгоритма нельзя сделать никаких выводов о принадлежности x; если же $\exists i \in L$, то $x \in X_i$. В частности, если $|L| > 1$, то это значит, что $\exists i, j\ X_i^0 \cap X_j^0 \neq \varnothing$.

4. Свойства алгоритма A_1^B. Покажем, что алгоритм A_1^B работает правильно в следующем смысле: он строит те и только те объекты, которые являются логическим следствием X_i^0. Для упрощения рассмотрения пар объектов обобщим операцию построения объектной резольвенты на произвольное количество объектов:

$$Or_j(p_1, ..., p_m) = Or_j\left(Or_j(p_1, ..., p_{m-1}), p_m\right), \text{ где } m \geq 2.$$

Исследуем свойства обобщенной объектной резольвенты, которые следуют из ее определения и будут использованы при описании свойств алгоритма A_1^B.

Свойство 1. $Or_j(p_1, ..., p_m) = Or_j\left(p_{\sigma(1)}, ..., p_{\sigma(m)}\right)$, где σ – произвольная перестановка номеров признаков (1, …, m).

Свойство 2. $p = Or_j(p, p)$.

Свойство 3. $Or_j(p_1, ..., p_m) = Or_j\left(Or_j(p_1, ..., p_t), Or_j(p_{t+1}, ..., p_m)\right)$, где $1 \leq t \leq m$.

Покажем, что исключение классов и объектов на шагах 7 и 8 алгоритма A_1^B правомерно: если объект x может быть выведен из X^0 и $\mu_j^1(x, p) = 0$ после рассмотрения признака s_j, то в дальнейшем объект p не может участвовать в выводе x, поэтому удаление p из Y_i не приведет к тому, что x не будет построен алгоритмом A_1^B.

Теорема 1. *Пусть* $\mu_j^1(x, p_1) = \mu_j^1(x, p_2)$, *где* $p_1, p_2 \in X_i$. *Если* $h \neq j$ *или* $\mu_j^1(x, p_1) = \mu_j^1(x, p_2) = 1$, *то* $\mu_j^1(x, Or_h(p_1, p_2)) = \mu_j^1(x, p_1) = \mu_j^1(x, p_2)$.

Покажем, что порядок рассмотрения признаков в алгоритме A_1^B не влияет на результат.

Теорема 2. *Пусть объект* $r = Or_{j_2}\left(Or_{j_1}(p_1, p_2), Or_{j_1}(p_3, p_4)\right)$ *был построен алгоритмом* A_1^B *из объектов* $p_1, p_2, p_3, p_4 \in Y_i$ *при последовательном рассмотрении признаков* s_{j_1} *и* s_{j_2}. *Тогда объект r будет построен и при рассмотрении* s_{j_1} *и* s_{j_2} *в обратном порядке:*

$$r = Or_{j_1}\left(Or_{j_2}(p_1, p_3), Or_{j_2}(p_2, p_4), Or_{j_2}(p_1, p_4), Or_{j_2}(p_2, p_3)\right).$$

Доказательства теорем 1 и 2 носят технический характер и следуют из определений обобщенной объектной резольвенты и функции μ_j^1.

Покажем, что по окончании работы алгоритма A_1^B $x \in Y_i$ тогда и только тогда, когда x является логическим следствием X_i^0. Заметим, что если $\mathrm{Norm}(x) \subset \mathrm{Norm}(p)$ и $p \in Y_i$, то и $x \in Y_i$. Поэтому будем считать, что если объект p может быть выведен из X_i^0 и $\mathrm{Norm}(x) \subset \mathrm{Norm}(p)$, то и x может быть выведен из X_i^0.

Теорема 3. *Объект x может быть выведен из* X_i^0 *с помощью метода объектных резолюций тогда и только тогда, когда в результате выполнения алгоритма* A_1^B $i \in L$.

Доказательство. 1. Пусть x может быть выведен из X_i^0. Предположим, что в процессе выполнения алгоритма A_1^B признаки рассматриваются в последовательности их номеров S_0. Пусть для вывода x из X_i^0 методом объектных резолюций нужно построить объектные резольвенты по признакам в последовательности J. Если в J имеются повторяющиеся признаки, то добавим их в S_0, что по теореме 1 и свойствам обобщенной объектной резольвенты не повлияет на результат алгоритма A_1^B. Поскольку по теореме 2 порядок рассмотрения признаков в алгоритме A_1^B не имеет значения, то переставим признаки в S_0 в том порядке, в каком они идут в J. Отсюда получаем, что x будет выведен из X_i^0, если при выполнении алгоритма A_1^B признаки рассматриваются в последовательности J, которая применяется при выводе x методом объектных резолюций.

2. Пусть после выполнения алгоритма A_1^B $i \in L$. Из шагов 7 и 8 следует, что после рассмотрения алгоритмом A_1^B всех признаков $\forall j \ \forall i \in L \ \forall p \in Y_i \ \mu_j^1(x, p) = 1$. Тогда любой объект $p \in Y_i$ может быть выведен из X_i^0 путем построения объектных резольвент в последовательности S_0. По определению $\mu_j^1(x, p)$ получаем, что $\forall j \ \mu_j^1(x, p) = 1$ тогда и только тогда, когда $\mathrm{Norm}(x) \subset \mathrm{Norm}(p)$, а тогда и x может быть выведен из X_i^0.

Из теоремы 4 следует правильность работы алгоритма A_1^B. Таким образом, для алгоритма A_1^B справедливы все утверждения, приведенные в [1] для алгоритма A_1.

5. Модификация алгоритма A_2. Опишем модификацию алгоритма A_2, основанную на оценке близости объектов по части признаков. Как и в алгоритме A_1^B, рассмотрим признаки по очереди. Определим функцию ρ_j следующим образом:

$$\rho_j\left(D_h^x, D_h^p\right) = \begin{cases} (-1)^t, h \in \{s_1, ..., s_j\}, \\ 1, h \in \{s_{j+1}, ..., s_n\}, \end{cases} \quad \text{где } t = \begin{cases} 1, \ D_j^x \neq D_j^p, \\ 2, \ D_j^x = D_j^p. \end{cases}$$

Значение ρ_j отражает близость первых j признаков объектов x и p. Введем также функцию верхней оценки частичной близости объектов μ_j^2 по j признакам:

$$\mu_j^2(x, p) = \max\left\{0, \left(\sum_{h \in S} a_{ij}\right)^{-1} \sum_{h \in S} a_{ih} \rho_j\left(D_h^x, D_h^p\right)\right\}, \text{ где } a_{ij} \in R, \ \forall i, j \ \ a_{ij} \geq 0, \ \forall i \ \sum_j a_{ij} > 0.$$

Здесь $\|a_{ij}\|$ – матрица, поставленная в соответствие множеству $\{1, ..., l\} \times |S|$.

Заметим, что если $h = 0$, то $\mu_j^2(x, p) = 1$. Рассмотрим следующую модификацию алгоритма A_2: множество L разбивается на l подмножеств, содержащих номера классов, а функция μ_j^2 применяется в качестве верхней оценки принадлежности x текущему классу.

Шаг 1. Пусть $\forall i = \overline{1, l}$ $j_i = 0$. Выберем произвольный класс X_i.

Шаг 2. Для всех $x^0 \in X_i^0$ вычислим $\mu_{j_i}^2(x, x^0)$.

Шаг 3. Для всех $i = \overline{1, l}$ вычислим $P_i(x) = \max\limits_{x^0 \in X_i^0} \left\{ \mu_{j_i}^2(x, x^0) \right\}$.

Шаг 4. Вычислим $P(x) = \max\limits_i \left\{ P_i(x) \right\}$.

Шаг 5. Пусть $P_{i_0}(x) = \max\limits_i \left\{ P_i(x) \right\}$. Если $j_{i_0} = n$, то переходим на шаг 7.

Шаг 6. Перейдем к следующему признаку в X_{i_0}: $j_{i_0} := j_{i_0} + 1$. Возвращаемся на шаг 2.

Шаг 7. Занесем объект x в класс X_{i_0}. Алгоритм завершает работу.

Обозначим этот алгоритм через A_2^B. Заметим, что в отличие от алгоритма A_1^B алгоритм A_2^B определяет принадлежность любого объекта $x \in X$.

6. Модификация алгоритма A_C. Рассмотрим модификацию алгоритма A_C, объединяющую алгоритмы A_1^B и A_2^B.

Шаг 1. Пусть $L = \{1, ..., l\}$, $\forall i = \overline{1, l}$ $j_i = 0$, $Y_i = X_i^0$. Выберем произвольный класс X_i.

Шаг 2. Если $i \in L$, то выполним шаги 3–8 алгоритма A_1^B для класса X_i и признака j_i, иначе перейдем к шагу 4.

Шаг 3. Если $\exists p \in Y_i$ $\mathrm{Norm}(x) \subset \mathrm{Norm}(p)$, то обозначим $i_0 = i$ и перейдем к шагу 7.

Шаг 4. Выполним шаги 2–4 алгоритма A_2^B.

Шаг 5. Пусть $P_{i_0}(x) = \max\limits_i \left\{ P_i(x) \right\}$. Если $j_{i_0} = n$, то перейдем к шагу 7.

Шаг 6. Перейдем к следующему признаку в X_{i_0}: $j_{i_0} := j_{i_0} + 1$. Возвращаемся на шаг 2.

Шаг 7. Занесем объект x в класс X_{i_0}. Алгоритм завершает работу.

Обозначим этот алгоритм через A_C^B. Поскольку A_2^B выдает результат для всех $x \in X$, то с помощью алгоритма A_C^B также можно определить принадлежность любого $x \in X$.

7. Сложность модифицированных алгоритмов. Подсчитаем сложность алгоритмов A_1^B, A_2^B и A_C^B. В худшем случае их сложность равна сложности алгоритмов A_1, A_2 и A_C соответственно, так как здесь в этих трех алгоритмах будут осуществлены все шаги для всех классов и признаков, что и происходит при выполнении немодифицированных алгоритмов. Более детальный расчет сложности алгоритмов описан в табл. 1–3.

Таблица 1

Сложность алгоритмов A_1 и A_1^B

Шаг A_1	Шаг A_1^B	Сложность
Шаги 1–3	Шаг 1	Шаги 2–8 будут выполнены n раз
	Шаг 2	Шаги 3–5 будут выполнены l раз
	Шаг 3	Шаги 4–5 будут выполнены q_i^2 раз, где $q_i = \lvert Y_i \rvert$
Шаг 4	Шаг 4	Построение и проверка существования r: $O(kn)$
Шаг 5	Шаг 5	Сравнение r с объектами из Y_i: $O(kq_in)$. Добавление r: $O(1)$
	Шаг 6	Поскольку сложность вычисления $\mu_j^1(x, Y_i)$ равна $O(kn\lvert Y_i \rvert)$, то сложность шага $O(kQn)$, где $Q = \sum\limits_{i=1}^{l} q_i$
	Шаг 7	$O(l)$
	Шаг 8	$O(Q)$
Шаг 6	Шаг 9	$O(1)$
Общая сложность		$O(kQ^3n^2)$

Таблица 2

Сложность алгоритмов A_2 и A_2^B

Шаг A_2	Шаг A_2^B	Сложность
Обучение: шаги 1–4		$O(q_0 n)$, $O(ln)$, $O(ln)$, $O(1)$ соответственно
Распознавание:		
Шаг 1	Шаги 1–2	Поскольку сложность вычисления $\mu_j^2(x, p)$ составляет $O(kn)$, то сложность шагов $O(kq_{0i}n)$, где $\lvert X_i^0 \rvert = q_{0i}$
Шаг 2	Шаг 3	$O(q_{0i})$
	Шаги 4–5	$O(l)$
Шаги 1–2	Шаг 6	Шаги 2–5 будут выполнены не более чем ln раз
Шаг 3	Шаг 7	$O(1)$
Общая сложность		$O(kq_0 n^2)$

Таблица 3

Сложность алгоритмов A_C и A_C^B

Шаг A_C	Шаг A_C^B	Сложность
Шаг 1	Шаги 1–3	$O(kq_i^3 n)$
Шаг 2	Шаги 4–5	$O(kq_{0i}n)$
	Шаг 6	Шаги 2–5 будут выполнены не более чем ln раз
Шаг 3	Шаг 7	$O(1)$
Общая сложность		$O\left(k\left(Q^3 + q_0\right)n^2\right)$. Если $Q >> q_0$, то $O\left(kQ^3n^2\right)$

Оценим изменение Q в алгоритмах A_1^B и A_C^B по сравнению с алгоритмами A_1 и A_C соответственно. Пусть перед рассмотрением признака s_j имеется множество объектов Y_i. Обозначим через Y_i^1 и Y_i^B множества объектов, построенные в результате рассмотрения s_j алгоритмами A_1 и A_1^B соответственно. Пусть $\lvert Y_i^1 \rvert = q_i^1$, $\lvert Y_i^B \rvert = q_i^B$. Сравним q_i^1 и q_i^B. Предположим, что в Y_i все значения признаков встречаются с одинаковой частотой. Для упрощения не будем удалять совпадающие и несуществующие объекты из Y_i^1 и Y_i^B.

В алгоритме A_1 множество Y_i^1 состоит из всех резольвент, построенных из объектов Y_i по признаку s_j. В алгоритме A_1^B множество Y_i^B состоит только из тех объектов p, для которых $\mu_j(x, p) = 1$: если $c\left(D_j^x\right) = \left(d_{j1}^x, ..., d_{jk}^x\right)$, $c\left(D_j^p\right) = \left(d_{j1}^p, ..., d_{jk}^p\right)$, то $Y_i^B = \left\{p \mid d_{jt}^x \le d_{jt}^p, t = \overline{1,k}\right\}$. Поэтому если $d_{jt}^x = 1$, то $d_{jt}^p = 1$, а значит, если $c\left(D_j^x\right)$ содержит m_j единиц, то в среднем $q_i^B = 2^{-m_j} q_i^1$. Тогда, если $c(x)$ содержит $m = \sum_{j=1}^{n} m_j$ единиц, то $Q^B = 2^{-m_1} \cdot ... \cdot 2^{-m_n} Q^1 = 2^{-m} Q^1$, где Q^1 и Q^B – общее количество объектов, построенных алгоритмами A_1 и A_1^B соответственно. Следовательно, в среднем алгоритм A_1^B работает в 2^{3m} раза быстрее A_1, а тогда и алгоритм A_C^B работает в 2^{3m} раза быстрее A_C.

Таким образом, рассмотрены три алгоритма решения задачи распознавания Z, представляющие собой модификации алгоритмов, описанных в [1], использующие метод ветвей и границ. Приведены оценки вычислительной сложности предложенных алгоритмов и показано, что в среднем алгоритмы индуктивной резолюции работают быстрее их немодифицированных версий, причем уменьшение времени их работы экспоненциально зависит от структуры информации о рассматриваемом объекте.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Шут О. В. Синтез алгоритмов для решения задач распознавания образов в конечномерных дискретных пространствах // Вестн. БГУ. Сер. 1, Физика. Математика. Информатика. 2014. № 1. С. 56–62.

2. Краснопрошин В. В., Образцов В. А. Распознавание с обучением как задача выбора // Цифровая обраб. изобр. Минск, 1998. С. 80–94.

3. А б л а м е й к о С. В., К р а с н о п р о ш и н В. В., О б р а з ц о в В. А. Распознавание образов и анализ изображений: теория и опыт решения практических задач // Материалы Междунар. конгр. по информатике «Информационные системы и технологии». Минск, 2013. С. 434–444.

4. Г а ф у р о в С. В., К р а с н о п р о ш и н В. В., О б р а з ц о в В. А. Проблема неполноты информации в задаче распознавания образов // Вестн. БГУ. Сер. 1, Физика. Математика. Информатика. 2007. № 2. С. 113–116.

Поступила в редакцию 23.12.2013.

Ольга Викторовна Шут – аспирант кафедры информационных систем управления. Научный руководитель – кандидат физико-математических наук, доцент кафедры информационных систем управления В. А. Образцов.