Информационные процессы, Том 11, № 3, 2011, стр. 394–412.

Нестрогий вывод в системах альтернатив

Н.А.Моросанова

ВМК, МГУ имени М. В. Ломоносова, Москва, Россия
Поступила в редколлегию 7.10.2011

Аннотация—Системы альтернатив позволяют конструировать и использовать базы знаний. В работе исследуются возможности обобщения алгоритма логического вывода в системах альтернатив для работы с противоречивыми данными и предлагается обобщающий алгоритм на основе логики с векторной семантикой.

1. ВВЕДЕНИЕ

Простая система альтернатив является одним из способов реализации баз знаний [1]. В основе этого способа лежит подход к описанию объектов проблемной области с помощью атрибутов, принимающих на каждом объекте ровно одно значение, которое называется признаком объекта. В модели проблемной области полный список допустимых значений каждого из атрибутов фиксируется заранее.

В связи с организацией и использованием баз знаний возникает набор задач, которые можно разделить на две группы: задачи, связанные с извлечением знаний, и задачи использования полученных знаний. Вторая группа, в свою очередь, состоит из задач представления знаний и логического вывода. Рассмотрим вопросы использования систем альтернатив для решения задач второй группы.

Задача представления знаний сводится к конструирования системы альтернатив, описывающей предметную область и знания о ней. При этом считается, что:

– предметная область содержит объекты;

– объекты описываются с помощью атрибутов;

– на каждом объекте атрибуты принимают ровно одно значение.

Значение атрибута, соответствующее некоторому объекту, будем называть признаком объекта, или просто признаком. В системах альтернатив для описания предметной области используются атрибуты, которым сопоставляются альтернативы. Фактически альтернатива для представления атрибута есть набор признаков, допустимых для разных объектов.

Знания о предметной области, то есть знания о связях между признаками объектов, представляются в виде так называемых модулей знаний, каждый из которых может состоять из одной или нескольких альтернатив. Существует несколько типов модулей знаний, в частности, в системах альтернатив можно представлять продукции.

Зафиксируем один из объектов предметной области, относительно которого известны некоторые его признаки. Такой объект будем называть активным объектом. Часть признаков активного объекта считается неизвестной. Задача логического вывода состоит в определении неизвестных признаков активного объекта. Примером такой задачи может служить определение класса объекта по имеющимся признакам.

В связи с задачей логического вывода будем рассматривать пары (признак, оценка достоверности) для активного объекта, которые называются фактами. Поскольку любой признак содержится хотя бы в одной альтернативе, то будем говорить, что соответствующий факт также содержится в этой альтернативе.

Факт может находиться в различных состояниях, среди которых выделены обязательные установленное, опровергнутое и неопределенное состояния. Установленное и опровергнутое состояния факта соответствуют некоторой степени наличия и отсутствия признака, соответственно. Состояние факта определяется значением его оценки достоверности, которое есть элемент некоторого множества данных. В этом множестве зафиксированы значения, соответствующие состояниям факта. Для краткости также будем называть значение оценки достоверности факта просто значением факта.

Стандартный логический вывод [1] в системах альтернатив основывается на трех правилах:

1. если факт установлен, то во всех содержащих его альтернативах остальные факты переходят в опровергнутое состояние;

2. если в некоторой альтернативе остается ровно один факт, то этот факт считается установленным;

3. если в альтернативе имеется более одного установленного факта, либо все факты опровергнуты, то это свидетельствует о наличии противоречий в исходных данных.

2. МОДЕЛЬНАЯ ЗАДАЧА РАСПОЗНАВАНИЯ ЖИВОТНЫХ

Рассмотрим метод альтернатив на примере модельной задачи распознавания животных. Следуя [1], будем записывать систему альтернатив в виде наборов $< A_1 \ldots A_n >$, где A_i являются признаками. Для модельной задачи база знаний с признаками животных может быть представлена в виде системы альтернатив. Рассмотрим часть этой системы:

Пример 1. Альтернативы для описания атрибутов модельной области.

$<$ carniv veg $>$ – тип питания животного

$<$ longleg shortleg mediumleg $>$ – длина конечностей животного

$<$ spot stripe normal $>$ – тип окраски

$<$ white white-black black-white yellow-brown $>$ – окраска

$<$ leopard tiger giraffe zebra ostrich penguin albatros $>$ – животные[1]

В примере 1 атрибутами являются тип питания, длина конечностей, тип окраски, окраска, название животных. Возможные значения атрибутов, перечисленные в соответствующих альтернативах, являются признаками объекта-животного. Полностью система альтернатив приведена в приложении.

В рамках модельной задачи продукция представляется следующим образом:

Пример 2. (продолжение примера 1) Продукция „Если хищник имеет желто-коричневую окраску и полосатый, то это тигр" задается четырьмя альтернативами:

[1] Возможные объекты предметной области

< carniv carnivx3 carnivx4 >

< yellow-brown yellow-brownx3 yellow-brownx4 >

< stripe stripex1 stripex2 >

< tiger carnivx3 yellow-brownx3 stripex1 >,

где carnivx3, carnivx4, yellow-brownx3, yellow-brownx4, stripex1, stripex2 — вспомогательные признаки, не встречающиеся более ни в одной альтернативе.

В примере 2 активным объектом является определенное животное — тигр. С помощью набора альтернатив и алгоритма вывода можно установить класс объекта предметной области, используя известные признаки объекта: по типу питания и внешним признакам животного определить, что это тигр.

3. ЗАДАЧА ОБОБЩЕНИЯ АЛГОРИТМА ВЫВОДА

Входные данные для алгоритма вывода в системах альтернатив представляются в виде двух множеств: множества P признаков, которые присутствуют у объекта, и множества O признаков, которые у него отсутствуют. В этом случае в качестве структуры данных для представления оценки достверности может быть использована скалярная переменная, которая может принимать значения из множества $\{-1,\ 1,\ 0\}$, для признаков, принадлежащих соответственно O, P, либо ни одному из двух множеств.

Для некоторых из признаков реальных объектов такое описание возможно: например, признаками, входящими в атрибут „тип питания” из модельной задачи, объект либо обладает, либо не обладает. Наличие других признаков не является столь определенным. Скажем, для признаков из атрибута „длина конечностей животного” разные наблюдатели могут по-разному оценивать их наличие, более того, могут использовать модификаторы „довольно”, „очень” и др., которые определяют степень наличия. В таких случаях трудно представить входные данные для вывода в виде множеств P и O, и возникает задача выбора подходящего способа представления данных для использования в процессе логического вывода.

При ближайшем рассмотрении оказывается, что эта задача возникает из-за того, что структура данных, предназначенная для оценки достоверности, недостаточна для описания знаний о предметной области. Традиционный подход к решению этой задачи состоит в том, что выбирается другая структура данных для оценки достоверности, расширяющая возможности прежней структуры. Например, числа из множества $\{-1,\ 1,\ 0\}$ заменяются числами из отрезка $[-1, 1]$. При этом возможны два способа построения алгоритма вывода для работы с новой структурой данных для оценки достоверности факта.

Первый способ заключается в том, что новая структура данных используется только для описания знаний о предметной области, после чего некоторым образом сводится к прежней структуре, которая уже используется в процессе вывода. Таким образом, становится возможным использовать имеющийся алгоритм вывода, работающий с прежней структурой. Например, для оценки достоверности, представляемой в виде числа из $[-1, 1]$, этот способ сводится к тому, что вводятся некоторые пороговые значения из интервала $(-1, 1)$. Если значение оценки достоверности превышает выбранный порог, то наличие признака считается установленным, аналогично для отсутствия признака. Такой способ ведет к появлению дополнительных задач. Во-первых, это выбор порогового значения, которое обычно выбирается или статистически, или на основе дополнительных знаний или предположений относительно предметной области. Во-вторых, это интерпретация полученного результата в рамках новой структуры данных.

Второй способ заключается в построении обобщенного алгоритма вывода, напрямую использующего новую структуру. Различия двух способов работы с исходными данными показаны на рис.1.

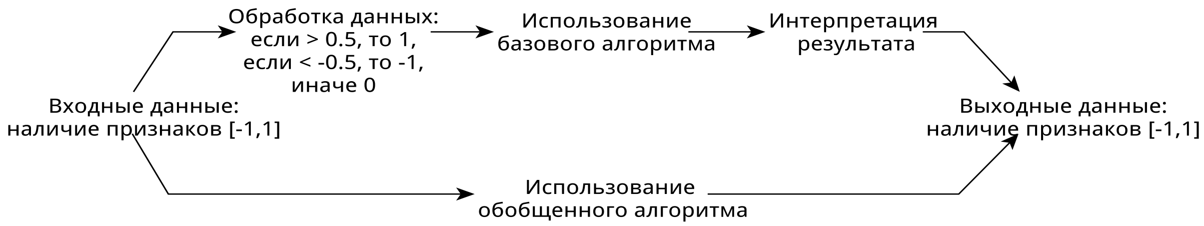

Рис. 1. Способы работы с исходными данными

При втором способе появляется возможность использовать информацию, которую базовый алгоритм, как правило, не учитывает.

4. ПОДХОДЫ К ОБОБЩЕНИЮ АЛГОРИТМА ВЫВОДА В СИСТЕМАХ АЛЬТЕРНАТИВ

Для построения обобщенного алгоритма вывода в системах альтернатив необходимо определить основания, на которых будет строиться этот алгоритм. Во-первых, необходимо определить способ описания исходных данных, прежде всего, структуру данных для представления оценок достоверности. Во-вторых, необходимо определить логику, по законам которой будет происходить вывод.

4.1. Обобщения классической логики

Вывод в системах альтернатив является разновидностью логического вывода в рамках классической логики первого порядка. В случае обобщения алгоритма для работы с фактами, оценка достоверности которых представлена расширенной структурой данных, необходимо основываться на законах какой-либо обобщающей логики. При этом выбор структуры данных для представления оценок достоверности определяется выбранной логикой. Рассмотрим в качестве обобщающих логик многозначные логики и векторные логики, а также логики с векторной семантикой.

Многозначные логики

В многозначных логиках вместо пары значений 0 и 1 используются три и более значений для представления оценки достоверности, что позволяет использовать модификаторы наличия признака у объекта. Такие логики могут быть конечнозначными и бесконечнозначными. Чаще всего в качестве конъюнкции и дизъюнкции используются функции min и max, однако возможно использование и других функций.

Частным случаем бесконечнозначных логик являются нечеткие логики, в которых используются значения из $[0, 1]$. Нечеткая логика определяется выбором основных операторов, таких, как конъюнкция, дизъюнкция, отрицание. Вывод во многом определяется выбором оператора импликации. В [2] описывается применение нечеткой логики при построении экспертных систем, при этом для представления оценок достоверности предлагается использовать две величины: необходимость и возможность. Эти величины определяются следующим образом:

– необходимость n — это та степень, в которой наличие признака подтверждено данными;

– возможность p — это та степень, в которой данные не опровергают наличие признака.

Эти величины позволяют работать с исходными данными, в которых содержатся противоречия относительно наличия признаков. Тем не менее, на их значения накладываются некоторые ограничения (например, $n \leq p$), поэтому не любые противоречия могут быть описатны таким образом.

Векторные логики

Векторные логики [7] основаны на использовании вектора в качестве структуры данных для оценки достоверности и матричного описания основных логических операторов. Векторы принадлежат q-мерному евклидову пространству. Векторная логика представляет собой пятерку:

– целое число k ;
– ортонормальная система k векторов размерности q, каждый из которых представляет собой одно из значений k-значной логики;
– классификаторная матрица, которая определяет разложение произвольного вектора по системе k векторов;
– набор одноместных логических операторов, которые представляются квадратными матрицами;
– набор двуместных операторов, которые представляются матрицами, действующими на результат произведения Кронекера двух операндов-векторов.

Пример 3. Векторная трехзначная логика

– $k = 3$
– система векторов: $s = [1, 0, 0]^T$ – „Истина", $h = [0, 0, 1]^T$ – „Неизвестно", $n = [0, 1, 0]^T$ – „Ложь" ($q = 3$)
– классификаторная матрица

$$F = \begin{bmatrix} 1\,0\,0 \\ 0\,1\,0 \\ 0\,0\,1 \end{bmatrix}$$

– набор одноместных логических операторов:

$$neg = \begin{bmatrix} 0\,1\,0 \\ 1\,0\,0 \\ 0\,0\,1 \end{bmatrix}$$

где neg — отрицание
– набор двуместных операторов: конъюнкция con, дизъюнкция dis, материальная импликация $impl$, импликация Лукашевича $implL$

$con = \begin{bmatrix} 1\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,1\,0\,1\,1\,1\,0\,1\,0 \\ 0\,0\,1\,0\,0\,0\,1\,0\,1 \end{bmatrix}$	$dis = \begin{bmatrix} 1\,1\,1\,1\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,1\,0\,1\,1 \end{bmatrix}$
$impl = \begin{bmatrix} 1\,0\,0\,1\,1\,1\,1\,0\,0 \\ 0\,1\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,0\,0\,1\,1 \end{bmatrix}$	$implL = \begin{bmatrix} 1\,0\,0\,1\,1\,1\,1\,0\,1 \\ 0\,1\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,0\,0\,1\,0 \end{bmatrix}$

Формализм векторных логик удобен для описания проблемных областей, где факт имеет множество параметров и хорошо описывается вектором. При этом многие законы классической логики, такие, как правила Де Моргана, переносятся на построенную векторную логику.

Логики с векторной семантикой

В [3, 4] развивается аппарат логик с векторной семантикой, в которых векторы используются для оценки достоверности фактов. При этом компоненты векторов рассматриваются как независимые величины, значения которых могут изменяться от 0 до 1. Независимость компонент вектора означает то, что информация о какой-либо компоненте не может быть выведена из информации о другой компоненте.

Частным случаем таких векторов являются двумерные векторы, компоненты которых рассматриваются как оценка достоверности установленности и опровергнутости факта, соответственно. Эти компоненты также называются истинностью и ложностью факта. Истина и ложь в классическом понимании соответствуют значениям $< 1, 0 >$ и $< 0, 1 >$.

Для случая двумерных векторов в логиках с векторной семантикой вводятся операторы конъюнкции, дизъюнкции и отрицания. Для каждого из операторов предлагается две альтернативные формы:

- $< a^+, a^- > \&_1 < b^+, b^- > \stackrel{\mathrm{def}}{=} < min(a^+, b^+), max(a^-, b^-) >$
- $< a^+, a^- > \&_2 < b^+, b^- > \stackrel{\mathrm{def}}{=} < min(a^+, b^+), min(a^-, b^-) >$
- $< a^+, a^- > \vee_1 < b^+, b^- > \stackrel{\mathrm{def}}{=} < max(a^+, b^+), min(a^-, b^-) >$
- $< a^+, a^- > \vee_2 < b^+, b^- > \stackrel{\mathrm{def}}{=} < max(a^+, b^+), max(a^-, b^-) >$
- $\neg_1 < a^+, a^- > \stackrel{\mathrm{def}}{=} < a^-, a^+ >$
- $\neg_2 < a^+, a^- > \stackrel{\mathrm{def}}{=} < 1 - a^+, 1 - a^- >$

В [3] рассматривается также несколько форм импликации: материальная, нестрогая и содержательная.

Материальная импликация определяется традиционно: $\neg a \vee b$.

Нестрогая импликация $a \to b$ определяется равенством[2] $a = b$, или, что то же самое, $b^+ = a^+$, $b^- = a^-$.

Содержательная импликация в логике с векторной семантикой определяется на основе оценок достоверности не только фактов a и b, но и самой импликации. Если импликация $a \to b$ имеет оценку достоверности $i =< i^+, i^- >$, то вектор-оценка достоверности для b принадлежит прямоугольнику, ограниченному векторами $a\&i$ и $\neg a \vee i$ [3]. Вектор i — это степень подтвержденности импликативного закона. Согласно определению, для значений b^+, b^- определяются интервалы, а не конкретные значения. В случае, когда требуются не интервальные, а точные значения, рекомендуется брать середину соответствующего интервала.

Логики с векторной семантикой отличаются от векторных логик тем, что в них:

1. отсутствует конечная система базовых векторов;
2. не выполняется нормировка вектора.

Последнее обстоятельство позволяет гибко оперировать противоречивой информацией.

Как показано в [3], и многозначным, и нечетким логикам свойственны некоторые ограничения классической логики. В них действуют (в соответствующей форме):

- закон исключенного третьего,
- принцип противоречия.

Вместе с тем, логика с векторной семантикой позволяет описывать и работать с противоречивыми и неполными данными.

4.2. Комбинирование оценок достоверности

При наличии различных оценок достоверности одного и того же факта возникает задача комбинирования этих оценок. Задача состоит в том, чтобы определить функцию f, аргументами которой являются две оценки достоверности, представленные в виде значений некотрого множества данных, а значением — новое значение оценки достоверности.

Из общих соображений к f предъявляются некоторые требования:

1. коммутативность $f(x, y) = f(y, x)$;
2. ассоциативность $f(x, f(y, z)) = f(f(x, y), z)$;
3. идемпотентность $f(x, x) = x$.

Выполнение требований коммутативности и ассоциативности позволяет не учитывать порядок поступления информации о фактах. Требование идемпотентности рекомендуется соблюдать в тех случаях, когда источники оценок достоверности, могут быть зависимыми [8]. Примерами функций, удовлетворяющих всем трем требованиям, являются min и max.

[2] В [3] рассматривается два вида нестрогой импликации, однако для обоих видов рекомендуется для b брать значение, равное значению a.

Выполнение всех требований для функции комбинирования является желательным, но не обязательным. Каждый конкретный подход к комбинированию оценок достоверности определяет набор требований к функции комбинирования, которые обязательны в рамках этого подхода. Например, функция $(x+y)/2$ удовлетворяет только первому и третьему требованиям, а для функции $(x+y-xy)$ выполнены первое и второе требования.

Заметим, что использование функции комбинирования в системах альтернатив возможно только при выполнении требования идемпотентности, поскольку независимость значений одних фактов от других не гарантируется. Более того, присутствует явная зависимость значений фактов, входящих в одну альтернативу [3].

Рассмотрим с точки зрения применимости в системах альтернатив основные подходы к комбинированию: байесовский подход, аксиоматический подход, а также подходы, принятые в теории Демпстера-Шафера и в логиках с векторной семантикой.

Байесовский подход

Байесовский подход применим для комбинирования вероятностей событий. Этот подход основывается на формуле Байеса, которая в свою очередь основана на формуле условной вероятности. Формула Байеса может быть записана следующим образом:

$$p(A_k|B) = \frac{p(B|A_k) * p(A_k)}{\sum_i p(B|A_i) * p(A_i)} \qquad (4.2.1)$$

где A_i - все возможные события.

В случае применения байесовского подхода для вывода в системах альтернатив A_k — это факт, о котором имеется некоторая информация. При этом A_i — все факты в системе, а $p(A_i)$ — оценки достоверности этих фактов. При появлении информации о факте B, где $B = A_i$ для некоторого i возникает задача ее комбинирования с уже имеющейся информацией, которая представлена в виде известных $p(A_i)$. С помощью формулы (4.2.1) можно найти $p(A_i|B)$ для всевозможных A_i, которые будут далее использоваться как новые значения $p(A_i)$. Таким образом происходит комбинирование оценок достоверности.

Основная проблема байесовского подхода заключается в изначальном отсутствии информации о фактах, которая, тем не менее, требуется для применения формулы (4.2.1). Поэтому обычно используют некоторые простые правила для получения начальной информации. Можно, например, оценки всех фактов в альтернативе считать одинаковыми. Важно отметить, что условием применимости данного подхода является независимость оценок $p(A_i)$, которая не всегда гарантируется на практике.

Аксиоматический подход

Аксиоматический подход к комбинированию свидетельств [5] основан на поиске функции комбинирования, удовлетворяющей заранее определенным требованиям. Например, поиск функции производится только среди многочленов не более n степени.

При подборе функции, удовлетворяющей предварительно сформулированным требованиям, в [5] получены две формулы: $a+b-ab$ и $a+b-ab+ab(1-a)(1-b)$, где a и b – числа из отрезка $[0,1]$. Первая из них получена в предположении о том, что f - многочлен второй степени, а вторая — соответственно, четвертой. Первая из формул обладает свойствами ассоциативности и коммутативности, а вторая — только свойством коммутативности.

Достоинством аксиоматического подхода является тот факт, что свойства функции комбинирования известны и могут быть заданы необходимым образом. Его недостаток в том, что

[3] См. правило 1 в разделе Введение

налагаемые условия не всегда могут быть обоснованы. Отметим, что рассмотренные формулы не обладают свойством идемпотентности, что не позволяет их использовать в рамках вывода в системах альтернатив.

Комбинирование свидетельств в теории Демпстера-Шафера

Теория Демпстера-Шафера[4] [6] определяет правило для комбинирования свидетельств. В этой теории рассматриваются события и их вероятности, иногда ее считают также обобщением теории вероятностей. Отличие теории Демпстера-Шафера от последней состоит в том, что она позволяет рассматривать множества событий как единичные события. Например, если имеется информация о вероятности множества событий, но нет информации об единичных событиях, то не делается никаких предположений о распределении известной вероятности по элементам множества. Для каждого из событий или множества событий вводятся меры доверия и правдоподобия, которые сути являются понятиями, иденичными необходимости и возможности в многозначных логиках [2]. Правило комбинирования свидетельств позволяет объединять информацию из независимых источников; при этом игнорируются все противоречия, и, наоборот, усиливается доверие к данным, подтвержденным несколькими источниками. Такое усиление доверия ведет к отказу от требования идемпотентности.

В [8] изучаются возможности по построению других правил комбинирования свидетельств в рамках теории Демпстера-Шафера с сохранением требования идемпотентности.

Комбинирование свидетельств в логиках с векторной семантикой

Для двумерной логики с векторной семантикой предлагается [4] использовать правило комбинирования:

$$a = f(a_1, a_2) \stackrel{\text{def}}{=} < F_1(a_1^+, a_2^+), F_2(a_1^-, a_2^-) >$$

где a — вектор-результат комбинирования. В качестве F_1 и F_2 рекомендуется использовать вторую форму дизъюнкции [5].

В [4] функция комбинирования для аргументов-векторов определяется как вектор из функций, одинаковых для всех компонент вектора. Можно отметить, что функция $max(a, b)$ удовлетворяет требованиям коммутативности, ассоциативности и идемпотентности.

Анализ основных подходов к моделированию правдоподобных рассуждений позволяет заключить, что для построения обощенного алгоритма вывода в системах альтеннатив наилучшим образом походит логика с векторной семантикой, так как она позволяет оперировать противорчивыми данными и содержит адекватный задаче подход к комбинированию оценок достоверности фактов.

5. ОБОБЩЕННЫЙ АЛГОРИТМ ВЫВОДА В СИСТЕМАХ АЛЬТЕРНАТИВ

В системах альтернатив обобщенный алгоритм вывода должен обладать следующими свойствами:

1. Свойство совместимости. В строгом случае наличия/отсутствия признаков обобщенный алгоритм вывода должен приводить к тем же результатам, что и базовый алгоритм.
2. Свойство корректности. Обобщенный алгоритм вывода должен быть корректен относительно логики с векторной семантикой.

[4] также называется теорией свидетельств

[5] см. раздел Логики с векторной семантикой

3. Свойство завершимости. Обобщенный алгоритм вывода должен завершать работу через конечное число шагов.

5.1. Построение алгоритма

В качестве структуры данных для оценок достоверности будем использовать вектор из двух независимых компонент, принимающих значения из отрезка $[0, 1]$, так же, как в двумерной логике с векторной семантикой.

Для того, чтобы обеспечить совместимость с базовым алгоритмом вывода, необходимо определить, какие значения истинности и ложности фактов считать наличием и отсутствием признаков.

Значение оценки достоверности $< a^+, a^- >$ факта $(a, < a^+, a^- >)$ интерпретируется следующим образом:

- если $< a^+, a^- >=< 1, 0 >$, то признак a строго присутствует у объекта;
- если $< a^+, a^- >=< 0, 1 >$, то признак a строго отсутствует у объекта;
- если $a^+ > a^-$, то факт установлен;
- если $a^+ < a^-$, то факт опровергнут;
- если $a^+ = a^-$, то факт не определен.

В соответствии со свойством 1, результаты обобщенного алгоритма в строгом случае должны быть такими же, как и результаты базового алгоритма. Рассмотрим описание базового алгоритма более подробно. В этом описании выделим два его основных действия: установление факта и опровержение факта. Первое действие приводит к опровержению всех фактов альтернативы за исключением установленного. Второе действие приводит к установлению последнего из неопровергнутых фактов альтернативы (см. правила в разделе Введение).

С точки зрения логики с векторной семантикой, установление и опровержение факта A_n соответствуют следующим содержательным импликациям:

$$\begin{gathered} A_n \to \neg A_i, \text{ где i=1} \ldots \text{n-1}, \\ \neg A_1 \& \neg A_2 \& \ldots \& A_{n-1} \to A_n. \end{gathered} \tag{5.1.1}$$

Это импликативное представление позволяет описать механизм изменения значений фактов в альтернативе.

5.2. Описание алгоритма

Входные данные алгоритма — информация о некоторых из признаков. Для каждого из признаков будем записывать эту информацию как пару $A \bigcirc < A^+, A^- >$, где A — признак, а $< A^+, A^- >$ — оценка достоверности. Каждую такую пару будем называть входным свидетельством. Свидетельство отличается от факта тем, что оценка достоверности отражает часть информации о признаке, в то время как оценка достоверности факт содержит всю накопленную на данный момент информацию.

Будем полагать, что каждой альтернативе в соответствие поставлено три очереди. Обозначим их как Q_1, Q_2, Q_3, а длину Q_i, равную количеству элементов в очереди, как $L(Q_i)$. Эти очереди имеют следующую структуру:

- Элементами очередей являются свидетельства $a\textcircled{i} < a_i^+,\ a_i^- >$, где i – номер очереди, a – признак, $< a_i^+,\ a_i^- >$ – значение оценки достоверности его наличия.

- Первая из очередей содержит внешние для данной альтернативы свидетельства [6], вторая и третья очереди - вспомогательные.

Накопление информации в системе альтернатив происходит путем комбинирования значений оценок достоверности фактов и свидетельств. Для этого используется функция комбинирования f. Поскольку оценка достоверности описывается вектором из двух независимых компонент, то функция комбинирования представима в виде двух функций f^+ и f^- для компонент истинности и ложности соответственно, $f =< f^+, f^- >$.

Изначально все вспомогательные очереди пусты, все факты находятся в неопределенном состоянии $< 0, 0 >$, и для каждой альтернативы Q_1 содержит входные свидетельства для тех признаков, которые входят в эту альтернативу.

Шаг 1. Если очереди всех альтернатив пусты, то завершить работу.

Шаг 2. Выбрать очередную альтернативу, имеющую хотя бы одну непустую очередь.

Шаг 3. Обработать выбранную альтернативу, содержащую n фактов:

◇ Пока $L(Q_1) > 0$, то
для очередного свидетельства $a\textcircled{1} < a_1^+,\ a_1^- >$ и факта $(a, < a^+, a^- >)$

- изменить значение факта в альтернативе следующим образом:
  $a^+ = max(a^+, a_1^+),\ a^- = max(a^-, a_1^-)$

- во вторую очередь
  - либо добавить свидетельство $a\textcircled{2} < a^-, a^+ >$, если $a^+ > a^-$,
  - либо добавить свидетельство $a\textcircled{2} < a^-, a^+/(n-1) >$, если $a^+ < a^-$,
  - не добавлять свидетельств, если $a^+ = a^-$

◇ Пока $L(Q_2) > 0$, то
для очередного свидетельства $a\textcircled{2} < a_2^+,\ a_2^- >$ и факта $(a, < a^+, a^- >)$

- изменить значение факта в альтернативе следующим образом:
  - либо $a^+ = min(a^+, a_2^+),\ a^- = max(a^-, a_2^-)$, если $a_2^+ > a_2^-$ и $a^+ \neq a^-$,
  - либо $a^+ = max(a^+, a_2^+),\ a^- = max(a^-, a_2^-)$ иначе

- добавить в третью очередь
  - либо свидетельство $a\textcircled{3} < a^-, a^+ >$, если $a^+ > a^-$,
  - либо свидетельство $a\textcircled{3} < a^-, a^+/(n-1) >$, если $a^+ < a^-$,
  - не добавлять свидетельств, если $a^+ = a^-$

◇ Пока $L(Q_3) > 0$, то
- если в альтернативе
  - установлен ровно один факт, и

[6] например, входные свидетельства или данные, полученные из другой альтернативы

  ◦ нет фактов в неопределенном состоянии,
 то удалить из Q_3 все элементы и перейти к шагу 4

- если в альтернативе
  ◦ опровергнут хотя бы один факт, и
  ◦ есть факты в неопределенном состоянии,
 то удалить из Q_3 все элементы и перейти к шагу 4

- для очередного свидетельства a③ $< a_3^+,\ a_3^- >$
 и факта $(a, < a^+, a^- >)$
 изменить значение факта в альтернативе следующим образом:
 $a^+ = min(a^+, a_3^+),\ a^- = min(a^-, a_3^-)$

Шаг 4. Полученные значения фактов альтернативы добавить в очереди Q_1 для остальных альтернатив.

Шаг 5. Перейти к шагу 1.

В процессе работы алгоритма добавление свидетельства aⓘ $< a_i^+,\ a_i^- >$ в Q_i происходит только в том случае, если a содержится в альтернативе
и если $< a^+,\ a^- > \neq f(< a^+,\ a^- >, < a_i^+,\ a_i^- >)$.

Пример 4. Обработка единственной альтернативы обобщенным алгоритмом.

Система альтернатив представлена одной альтернативой из трех фактов: $< A\ B\ C >$. Известно, что у объекта отсутствует признак A, то есть имеется входное свидетельство A ◯ $< 0, 1 >$. На рис. 2 представлен порядок обработки альтернативы.

1).$A < 0,0 >, B < 0,0 >, C < 0,0 >$ → $Q_1 : A$① $< 0,1 >$; $L(Q_2) = 0$; $L(Q_3) = 0$

2).$A < 0,1 >, B < 0,0 >, C < 0,0 >$ → $L(Q_1) = 0$; $Q_2 : B$② $< 1,0 >$, C② $< 1,0 >$; $L(Q_3) = 0$

3).$A < 0,1 >, B < 1,0 >, C < 0,0 >$ → $L(Q_1) = 0$; $Q_2 : C$② $< 1,0 >$; $L(Q_3) = 0$

4).$A < 0,1 >, B < 1,0 >, C < 1,0 >$ → $L(Q_1) = 0$; $L(Q_2) = 0$; $Q_3 : C$③ $< 0,1 >$, B③ $< 0,1 >$

5).$A < 0,1 >, B < 0,0 >, C < 1,0 >$ → $L(Q_1) = 0$; $L(Q_2) = 0$; $Q_3 : B$③ $< 0,1 >$

6).$A < 0,1 >, B < 0,0 >, C < 0,0 >$ → $L(Q_1) = 0$; $L(Q_2) = 0$; $L(Q_3) = 0$

Рис. 2. Обработка альтернативы $< A\ B\ C >$

5.3. Корректность обобщенного алгоритма

Согласно (5.1.1), альтернативу можно рассматривать как набор импликаций. Пусть оценка достоверности импликации имеет значение $< 1, 0 >$. Тогда, используя определение содержательной импликации, можно вычислить интервалы для значений фактов в альтернативе на основе оценок достоверности свидетельств.

Назовем алгоритм вывода корректным относительно двумерной логики с векторной семантикой, если результаты вывода, получаемые с его помощью, удовлетворяют рассчитанным с помощью импликаций интервалам значений.

Докажем корректность алгоритма относительно двумерной логики с векторной семантикой. Для этого рассмотрим вначале несколько частных случаев, а затем общий случай.

Случай 1. Альтернатива из двух фактов: $< A_1\ A_2 >$ может быть представлена четырьмя импликациями

$$\begin{aligned} &A_1 \to \neg A_2,\\ &\neg A_2 \to A_1,\\ &\neg A_1 \to A_2,\\ &A_2 \to \neg A_1. \end{aligned} \tag{5.3.1}$$

Поэтому если A_1 принимает значение $< A_1^+, A_1^- >$, то согласно [3], A_2 должно получить значение $< A_2^+, A_2^- >$, где

$$0 \le A_2^+ \le A_1^- \text{ и } A_1^+ \le A_2^- \le 1. \tag{5.3.2}$$

В результате работы обобщенного алгоритма A_2 получит значение $< A_1^-, A_1^+ >$, которое удовлетворяет условиям (5.3.2). Таким образом, алгоритм является корректным относительно логики с векторной семантикой в случае с альтернативой из 2 фактов.

Случай 2. Альтернатива из трех фактов $< A_1\ A_2\ A_3 >$ может быть представлена девятью импликациями вида

$$A_i \to \neg A_j, \tag{5.3.3}$$

$$\neg A_i \& \neg A_j \to A_k, \tag{5.3.4}$$

где $i,\ j,\ k \in \{1\ ,2\ ,3\}$ и $i \ne j,\ j \ne k,\ i \ne k$. Импликации вида (5.3.3) аналогичны случаю 1. Импликации вида (5.3.4) описывают случай, когда все факты, кроме одного, были опровергнуты. Рассмотрим одну из импликаций этого вида: $\neg A_2 \& \neg A_3 \to A_1$. Согласно [3], конъюнкция в ее левой части имеет значение $< min(A_2^-, A_3^-), max(A_2^+, A_3^+) >$, тогда должны выполняться неравенства:

$$min(A_2^-, A_3^-) \le A_1^+ \le 1 \text{ и } 0 \le A_1^- \le max(A_2^+, A_3^+). \tag{5.3.5}$$

Результат работы алгоритма зависит от порядка поступления свидетельств $A_2 \bigcirc < A_2^+, A_2^- >$ и $A_3 \bigcirc < A_3^+, A_3^- >$.

Если эти свидетельства поступят одновременно (например, оба являются входными), то шаг 3 будет выполнен только один раз при их обработке. В таком случае A_1 получит значение $< min(A_2^-, A_3^-),\ max(A_2^+/2,\ A_3^+/2) >$. Полученные в результате вывода значения удовлетворяют неравенствам (5.3.5).

Если хотя бы одно из свидетельств не является входным, то возможен случай, когда для обработки двух свидетельств потребуется дважды выполнять шаг 3.

Первый раз, при получении $A_2 \bigcirc < A_2^+, A_2^- >$, где $A_2^+ < A_2^-$, факты A_1 и A_3 получат значение $< A_2^-, A_2^+/2 >$, то есть перейдут в установленное состояние. В результате обработки третьей очереди A_1 и A_3 получат значения $< A_2^+/2, A_2^+/2 >$.

Второй раз, когда в результате обработки других альтернатив будет получено свидетельство $A_3 \bigcirc < A_3^+, A_3^- >$, где $A_3^+ < A_3^-$, то A_1 получит значение $< max(A_3^-, A_2^+/2), max(A_3^+/2, A_2^+/2) >$, которое также удовлетворяет неравенствам (5.3.5), поскольку $A_2^+/2 < A_2^+ < A_2^-$.

Таким образом, алгоритм является корректным относительно логики с векторной семантикой в случае с альтернативой из 3 фактов.

Случай 3. Альтернатива из n фактов может быть представлена импликациями:

$$A_j \to \neg A_i, \tag{5.3.6}$$

$$\neg A_1 \& \neg A_2 \& \ldots \& \neg A_{j-1} \& \neg A_{j+1} \& \ldots \& \neg A_{n-1} \to A_j, \tag{5.3.7}$$

$$\text{где } i = 1 \ldots n,\ j = 1 \ldots n,\ i \neq j.$$

Импликации (5.3.6) аналогичны случаю 1.

Рассмотрим одну из импликаций (5.3.7) для случая $j = n$:

$$\neg A_1 \& \neg A_2 \& \ldots \& A_{n-1} \to A_n. \tag{5.3.8}$$

Согласно (5.3.8), A_n должно получить значение, удовлетворяющее неравенствам

$$\begin{array}{l} min(A_1^-, \ldots, A_{n-1}^-) \le A_n^+ \le 1, \\ 0 \le A_n^- \le max(A_1^+, \ldots, A_{n-1}^+). \end{array} \tag{5.3.9}$$

Аналогично случаю 2, A_n получит либо значение

$$< min(A_1^-, \ldots, A_{n-1}^-),\ max(A_1^+/(n{-}1), \ldots, A_{n-1}^+/(n{-}1)) >$$

в случае одновременного поступления всех свидетельств, либо

$$\begin{array}{l} <max(A_{n-1}^-, min(A_1^+/(n{-}1), \ldots, A_{n-2}^+/(n{-}1))), \\ \quad max(A_{n-1}^+/(n{-}1), min(A_1^+/(n-1), \ldots, A_{n-2}^+/(n{-}1))) >, \end{array}$$

если свидетельства поступают неодновременно. Эти значения удовлетворяют неравенствам (5.3.9). Таким образом, алгоритм является корректным относительно логики с векторной семантикой в случае с альтернативой из n фактов.

Случай 4. Система альтернатив. Для системы из одной альтернативы алгоритм корректен. Для системы из более чем одной альтернативы влияние одной альтернативы на другие заключается только в добавлении свидетельств в очереди других альтернатив на шаге 4. Поскольку порядок поступления свидетельств не влияет на корректность результата алгоритма, то алгоритм корректен для системы альтернатив.

Таким образом, алгоритм корректен относительно двумерной логики с векторной семантикой.

5.4. Совместимость с базовым алгоритмом вывода

Рассмотрим случаи установления и опровержения фактов для альтернативы $< A_1 \dots A_n >$. Воспользуемся результатами предыдущего раздела.

Случай установления факта $(A_1, < A_1^+, A_1^- >)$.
Пусть получено входное свидетельство $A_1 \bigcirc < 1, 0 >$. Тогда для фактов $(A_i, < A_i^+, A_i^- >)$ при $i = 2 \dots n$

$$0 \leq A_i^+ 0 \text{ и } 1 \leq A_i^- \leq 1.$$

Таким образом, все факты $(A_i, < A_i^+, A_i^- >)$ при $i = 2 \dots n$ получат значение $< 0, 1 >$, то есть будут опровергнуты. Такой результат обобщенного алгоритма совместим с базовым алгоритмом (см. правило 1 в разделе Введение). Случай опровержения фактов $A_1, \dots, A_{n-1}$.

Пусть получены входные свидетельства $A_i \bigcirc < 0, 1 >$ для $i = 1 \dots n{-}1$. Тогда для факта $(A_n, < A_n^+, A_n^- >)$ должно быть верно

$$min(1, \dots, 1) \leq A_n^+ \leq 1,$$
$$0 \leq A_n^- \leq max(0, \dots, 0).$$

Таким образом, факт $(A_n, < A_n^+, A_n^- >)$ будет установлен, что соответствует правилу 2 для базового алгоритма.

Таким образом, в случае строгого установления и опровержения факта получаются те же результаты, что в базовом алгоритме вывода. Это означает, что предложенный алгоритм совместим с базовым алгоритмом вывода.

5.5. Завершимость обобщенного алгоритма

Обозначим число входных свидетельств $A \bigcirc < A^+, A^- >$ за I, число альтернатив за K. Рассмотрим вначале одну альтернативу из n фактов. Поскольку $L(Q_1) \leq I$, то $L(Q_2) \leq In$ и $L(Q_3) \leq In^2$. После обработки этой альтернативы в первые очереди других альтернатив будет добавлено не более n элементов. Таким образом, очереди всегда имеют конечный размер, и, поскольку K конечно, то обработка каждой из альтернатив всегда завершается.

Обозначим множество значений истинности и ложности во входных данных за S, а множество $\bar{S}$ определим следующим образом:

$$\bar{S} = \{ \frac{s}{n} \mid n \in \mathbb{N},\ s \in S\}.$$

Тогда значения оценок свидетельств, как видно из описания алгоритма, могут состоят только из элементов множества $\bar{S}$. Очевидно, существует максимальный элемент множества $\bar{S}$, так как он совпадает с максимальным элементом множества S, а множество S конечно. При обработке каждого из свидетельств из Q_1 значение истинности или ложности факта должно возрастать, поскольку иначе свидетельство не проходит проверку при добавлении в очередь. Таким образом, число шагов обработки альтернатив конечно, и алгоритм завершает свою работу.

6. ПРИМЕРЫ РАБОТЫ ОБОБЩЕННОГО АЛГОРИТМА

Далее приведены примеры работы алгоритма для задачи распознавания животных.

Пример 5. Вывод типа животного.
Входные данные: желто-коричневый, хищник, полосатый

yellow-brown – $< 1, 0 >$; carniv – $< 1, 0 >$; stripe – $< 1, 0 >$
Результат: tiger

Пример 5 иллюстрирует работу продукции: на основе наблюдаемых признаков выводится тип животного. Отметим, что в этом примере исходные данные непротиворечивы.

Пример 6. Противоречивые данные.
Входные данные: желто-коричневый, хищник, полосатый или пятнистый
yellow-brown – $< 1, 0 >$; carniv – $< 1, 0 >$; stripe – $< 0.5, 0 >$; spot – $< 0.5, 0 >$
Результат: тип животного не определяется

В примере 6 в исходных данных содержится противоречивая информация относительно наблюдаемого признака – окраски животного. Поскольку этот признак является в данном случае определяющим, и оценки достоверности одинаковы, то тип животного определить не удается.

Пример 7. Разрешение противоречия с помощью двух правил.
Входные данные: полосатое, травоядное, бело-черное или черно-белое
veg – $< 1, 0 >$; stripe – $< 1, 0 >$; white-black – $< 0.5, 0 >$; black-white – $< 0.5, 0 >$
Результат: zebra

В примере 7 информация об окраске противоречива, но результат может быть получен. Так происходит потому, что возможные противоречия по вопросу окраски зебры учтены при составлении правил, и имеются два разных правила для каждого из типов окраски.

Пример 8. Разрешение противоречия в сторону наибольшего правдоподобия.
Входные данные: желто-коричневый, хищник, скорее полосатый
yellow-brown – $< 1, 0 >$; carniv – $< 1, 0 >$; stripe – $< 0.7, 0.1 >$; spot – $< 0.5, 0.2 >$
Результат: tiger

В примере 8 при имеющейся противоречивой информации об окраске животного делается выбор в пользу того признака, чья оценка достоверности больше смещена в сторону истины ($< 1, 0 >$).

7. ЗАКЛЮЧЕНИЕ

В результате исследования возможных подходов к обобщению алгоритма вывода в системах альтернатив установлено, что наилучшим образом обобщают алгоритм вывода логики с векторной семантикой.

Практическое применение этого подхода выдвигает задачу разработки принципов реализации подсистемы объяснений.

ПРИЛОЖЕНИЕ

А. СИСТЕМА АЛЬТЕРНАТИВ ДЛЯ МОДЕЛЬНОЙ ЗАДАЧИ РАСПОЗНАВАНИЯ ЖИВОТНЫХ

Таблица 2: Атрибуты

1	<bird mammal>	Животное или птица
2	<carniv veg>	Хищник или травоядное
3	<feather hair>	Перья или шерсть
4	<spot stripe normal>	Типы окраски
5	<white white-black black-white yellow-brown>	Цвета окраски
6	<longleg mediumleg shortleg>	Длина ног
7	<claw hoof>	Когти или копыта
8	<eatmeat eatgrass>	Тип пищи
9	<makemilk makeegg>	Что производит
10	<teeth beak>	Зубы или клюв
11	<fang nofang>	Наличие клыков
12	<longneck mediumneck shortneck>	Длина шеи
13	<nofly badfly mediumfly goodfly excfly>	Способность летать
14	<leopard tiger giraffe zebra ostrich penguin albatros>	Типы животных

Таблица 3: Альтернативы

15	<beak hair>	Очевидные взаимоисключения признаков
16	<beak fang veg>	
17	<hoof feather carniv>	
18	<feather mammal>	
19	<hair bird>	
20	<makemilk bird>	

Таблица 4: Продукции

21	<badfly badflyx1 badflyx2>	Если летает
22	<makeegg makeeggx1 makeeggx2>	и несет яйца,
23	<bird badflyx1 makeeggx1>	то это птица
24	<makeegg makeeggx3 makeeggx4>	Если летает
25	<mediumfly mediumflyx1 mediumflyx2>	и несет яйца,
26	<bird mediumflyx1 makeeggx3>	то это птица
27	<goodfly goodflyx1 goodflyx2>	Если летает
28	<makeegg makeeggx5 makeeggx6>	и несет яйца,
29	<bird goodflyx1 makeeggx5>	то это птица
30	<excfly excflyx1 excflyx2>	Если летает
31	<makeegg makeeggx7 makeeggx8>	и несет яйца,
32	<bird excflyx1 makeeggx7>	то это птица
33	<mammal mammalx1 mammalx2>	Если млекопитающее
34	<eatmeat eatmeatx1 eatmeatx2>	ест мясо,

35	<carniv mammalx1 eatmeatx1>	то это хищник
36	<teeth teethx1 teethx2>	Если животное имеет зубы
37	<fang fangx1 fangx2>	и клыки
38	<mammal mammalx3 mammalx4>	и является млекопитающим,
39	<carniv mammalx3 teethx1 fangx1>	то это хищник
40	<mammal mammalx5 mammalx6>	Если млекопитающее
41	<hoof hoofx1 hoofx2>	имеет копыта,
42	<veg hoofx1 mammalx5>	то это травоядное
43	<mammal mammalx7 mammalx8>	Если млекопитающее
44	<eatgrass eatgrassx1 eatgrassx2>	ест траву,
45	<veg mammalx7 eatgrassx1>	то это травоядное
46	<spot spotx1 spotx2>	Если животное имеет пятна
47	<yellow-brown yellow-brownx1 yellow-brownx2>	и коричнево-желтую окраску
48	<carniv carnivx1 carnivx2>	и является хищником,
49	<leopard carnivx1 yellow-brownx1 spotx1>	то это леопард
50	<stripe stripex1 stripex2>	Если животное имеет полосы
51	<yellow-brown yellow-brownx3 yellobrownx4>	и коричнево-желтую окраску
52	<carniv carnivx3 carnivx4>	и является хищником,
53	<tiger carnivx3 yellow-brownx3 stripex1>	то это тигр
54	<stripe stripex3 stripex4>	Если животное имеет полосы
55	<black-white black-whitex1 black-whitex2>	и черно-белую окраску
56	<veg vegx1 vegx2>	и является травоядным,
57	<zebra vegx1 black-whitex1 stripex3>	то это зебра
58	<stripe stripex5 stripex6>	Если животное имеет полосы
59	<white-black white-blackx1 white-blackx2>	и бело-черную окраску
60	<veg vegx3 vegx4>	и является травоядным,
61	<zebra vegx3 white-blackx1 stripex5>	то это зебра
63	<spot spotx3 spotx4>	Если животное имеет пятна
62	<yellow-brown yellow-brownx5 yellobrownx6>	и коричнево-желтую окраску
64	<veg vegx5 vegx6>	и является травоядным,
65	<longleg longlegx1 longlegx2>	имеет длинные ноги
66	<longneck longneckx1 longneckx2>	и длинную шею,
67	<giraffe vegx5 longlegx1 longneckx1 yellow-brownx5 spotx3>	то это жираф
68	<bird birdx1 birdx2>	Если животное является птицей,
69	<nofly noflyx1 noflyx2>	не летает,
70	<white-black white-blackx3 white-blackx4>	имеет бело-черную окраску,
71	<longleg longlegx3 longlegx4>	длинные ноги
72	<longneck longneckx3 longneckx4>	и длинную шею,
73	<ostrich birdx1 noflyx1 longlegx3 longneckx3 white-blackx3>	то это страус
74	<bird birdx3 birdx4>	Если животное является птицей,
75	<nofly noflyx3 noflyx4>	не летает,
76	<white-black white-blackx5 white-blackx6>	имеет бело-черную окраску,
77	<shortneck shortneckx1 shortneckx2>	короткие ноги
78	<shortleg shortlegx1 shortlegx2>	и короткую шею,
79	<penguin birdx3 noflyx3 shortlegx1 shortneckx1 white-blackx5>	то это пингвин
80	<bird birdx5 birdx6>	Если животное является птицей
81	<excfly excflyx3 excflyx4>	и отлично летает,
82	<albatros birdx5 excflyx3>	то это альбатрос

СПИСОК ЛИТЕРАТУРЫ

1. Соловьев С.Ю., Соловьева Г.М., Вопросы применения метода альтернатив для представления знаний, Изв. АН СССР, Техн. кибернетика, 1987, № 5, стр.80-82.
2. Siler W., Buckley J.J. Fuzzy Expert Systems and Fuzzy Reasoning. Hoboken: Wiley, 2005.
3. Аршинский Л. В., Многозначные логики с векторной семантикой, Деп. в ВИНИТИ 13.02.03, № 281-B2003, Иркутск, 2003
4. Аршинский Л. В., Исследование и разработка математических моделей обработки неполных и противоречивых данных на основе логик с векторной семантикой. Автореферат диссертации на соискание степени доктора технических наук, Иркутск, 2007
5. Johnson, N. and Kotz, S., Axiomatic approach to formulas for combining likelihoods or evidence, Journal of Statistical Computation and Simulation, 1989, No. 31, стр.49-54.
6. G. Shafer, The Dempster-Shafer theory. Encyclopedia of Artificial Intelligence, Second Edition, Stuart C. Shapiro, (ed.), Wiley, 1992, 330-331
7. Misraji E. Vector logic: a natural algebraic representation of the fundamental logical gates. Journal of Logic and Computation, 2008, 18, No. 1, стр. 97-122
8. Destercke, S. and Dubois, D. (2010). Idempotent merging of belief functions: Extending the minimum rule of possibility theory. In T. Denoeux, editor, Workshop on the Theory on Belief Functions (WTBF 2010).