УДК 004.8
ББК 32.813

ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ НА ОСНОВЕ КОЛОНОК

Чесноков А. М.[1]
(*ФГБУН Институт проблем управления им. В.А.Трапезниникова РАН, Москва*)

В статье рассматриваются интеллектуальные системы на основе колонок. Приводятся основные понятия и определения. Формулируются прямая и обратная задачи для образов в виде конечных неупорядоченных множеств и конечных упорядоченных множеств (конечных последовательностей и векторов). Приводятся решения этих задач и оценки вычислительной эффективности. На этой основе предлагается решение задачи классификации и доказывается возможность реализации произвольных булевых функции. В конце дается оценка классов задач, которые могут решаться с помощью интеллектуальных систем на основе колонок.

Ключевые слова: искусственный интеллект, интеллектуальные системы на основе колонок, колонка, модель «память-прогноз».

1. Введение

В основе интеллектуальных систем на основе колонок лежат идеи и методы двух направлений работ в области искусственного интеллекта (ИИ). Первое из них начало свое развитие вместе с выходом в свет книги Хокинса «On Intelligence» [4, 7], в которой он излагает свои представления о том, как работает кора головного мозга, и предлагает свое понимание того, что такое интеллект. По Хокинсу мозг – это не вычислительная

[1] *Александр Михайлович Чесноков, старший научный сотрудник, кандидат технических наук* (*alex-ches@yandex.ru*).

система, а система прогнозирующей памяти (memory-prediction system), огромные объемы которой используются для построения иерархической модели окружающего мира и формирования прогнозов будущих событий. При построении своей модели интеллекта, получившей название «память-прогноз» (memory-prediction model), Хокинс опирается на теорию Маунткасла о колончатой организации коры головного мозга [2], согласно которой минимальной структурной единицей является не отдельный нейрон, а колонка коры (кортикальная колонка), представляющая собой вертикально расположенную группу нейронов с большим числом связей по вертикальной оси и малым числом связей по горизонтальной оси (Рис. 1).

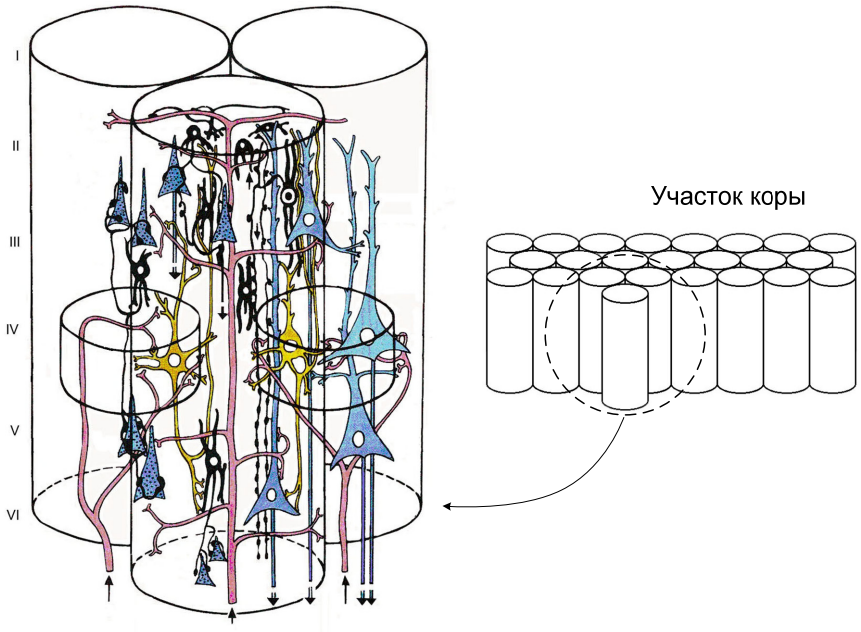


Рис. 1. Кортикальные колонки

В интеллектуальных системах на основе колонок центральным понятием также является понятие колонки. Однако в отличие от подхода Хокинса, где используется модель реальной кортикальной колонки, в данном случае речь идет об абстрактном объекте – упорядоченной паре вида (имя образа, образ). При этом, несмотря на свою абстрактность, системы на основе колонок удовлетворяют основным положениям модели «память-прогноз», а между абстрактными колонками и колонками коры существует достаточно глубокая аналогия [5].

Второе из упомянутых направлений связано с использованием индекса, заданного в виде списка или таблицы, для организации быстрого доступа к данным. Методы индексации получили широчайшее распространение и применяются, в частности, в базах данных, интерпретаторах и компиляторах, при реализации логического вывода [1, 3] и т.п. Таблично заданный индекс используется в поисковых машинах интернета [6], а в области ИИ индекс стал применяться для решения задач распознавания [8, 9].

В системах на основе колонок таблично заданный индекс необходим, в первую очередь, для реализации перехода по ссылке образ $\rightarrow$ имя образа, т.е. для получения для некоторого образа его имени. Разумеется, для этого могут применяться и другие методы. Однако схемы на основе индекса показали достаточную эффективность в различных задачах и, кроме того, обеспечивают для систем на основе колонок ряд интересных особенностей. Это и упоминавшаяся выше аналогия абстрактных и кортикальных колонок, и повышение быстродействия по мере накопления знаний – чем больше элементов использует система для формирования образов, тем она не только точнее отражает объекты окружающего мира, но и быстрее работает.

В данной работе приводятся основные понятия и определения, относящиеся к системам на основе колонок: имя, образ, колонка, индекс как множество колонок. Рассматриваются колонки общего вида и многоуровневый индекс. Показана связь между индексом как множеством колонок и обычным таблично заданным индексом. Знания в рассматриваемых системах представлены в виде образов, а в основе процесса накопления знаний лежит запоминание новых образов под определенными именами (т.е. формирование новых колонок). Рассматриваются связанные с этим базовые задачи для образов в виде конечных неупорядоченных множеств и конечных упорядоченных множеств (конечных последовательностей и векторов). Предлагается метод решения этих задач с использованием таблично заданного индекса и дается оценка его вычислительной эффективности, которая показывает снижение необходимого числа операций при увеличении количества элементов, участвующих в форми-

ровании образов. Базовые задачи важны не только сами по себе, но и как основа для решения других задач. В работе доказывается возможность решения задач классификации и реализации в рассматриваемых системах произвольных булевых функций. Эти результаты позволяют говорить о принципиальной возможности применения интеллектуальных систем на основе колонок для решения различных задач классификационного и синтетического типа.

2. Основные понятия

2.1. ИМЕНА И ОБРАЗЫ

Рассматривается конечное множество U – *множество имен* некоторых объектов произвольной природы. Не ограничивая общности, можно считать, что оно является подмножеством множества целых чисел Z. Множество U конечное, однако никаких ограничений на его размеры не накладывается. Так, например, можно принять, что его мощность равна 10^{20} или 10^{80}. Если по каким-то причинам этого вдруг окажется недостаточным, то его всегда можно увеличить, добавив необходимое число элементов.

Множество имен U имеет разбиение на непересекающиеся подмножества имен. Каждое такое подмножество называется *областью имен* и служит для определенных целей, например, область имен для плоских фигур. При необходимости любая область имен может быть в любой момент расширена, а также может быть введена новая область имен.

На практике удобно рассматривать области имен как независимые множества имен, перенумеровав имена в каждой области так, как это показано ниже для областей U_1 и U_2 ($U_1 \subset U$, $U_2 \subset U$, $U_1 \bigcap U_2 = \varnothing$):

$$\begin{array}{ccc} & \text{область имен } U_1 & \text{область имен } U_2 \\ & / \qquad\qquad \backslash & \\ U = \{ \ldots, \{a_1, a_2, \ldots, a_n\}, \ldots, & \{b_1, b_2, \ldots, b_m\}, \ldots \} \\ & \updownarrow \qquad\qquad \updownarrow & \\ & U_1 = \{1,\ 2, \ldots,\ n\} \qquad \{1,\ 2, \ldots,\ m\} = U_2 & \end{array}$$

Следует подчеркнуть, что несмотря на возможное формальное равенство, имена в независимых областях U_1 и U_2 совершенно разные. Например, имя $2 \in U_1$ не равно имени $2 \in U_2$, так как с точностью до взаимно однозначного отображения $2 \in U_1$ – это имя a_2 общего множества имен U, а $2 \in U_2$ – это имя b_2 того же множества U, причем $a_2 \neq b_2$. Фактически в данном примере неявным образом рассматриваются упорядоченные пары (2, 1) и (2, 2) или имена 2_1 и 2_2, в которых для простоты опущен второй элемент, указывающий на соответствующую область имен.

Под *образом* понимается любое конечное множество имен. Никакие ограничения на типы множеств не накладываются. В первую очередь будут рассматриваться образы в виде неупорядоченных конечных множеств и образы в виде упорядоченных конечных множеств – конечных последовательностей и векторов.

2.2. КОЛОНКИ

Рассмотрим некоторое конечное множество образов P, перенумерованных с помощью элементов некоторой области имен U':

$$P = \{p_i \mid i \in U'\},$$

где $|U'| = |P|$. Здесь через $|\cdot|$ обозначена мощность множества.

То, что образы множества P перенумерованы, означает, что установлено взаимно однозначное соответствие $\varphi : U' \leftrightarrow P$ между областью имен U' и множеством образов P.

Колонкой называется упорядоченная пара (i, p_i), где i – имя колонки, p_i – образ, содержащийся в колонке. Колонка обозначается как $(i \mid p_i)$. Также будет использоваться обозначение $i \to p_i$. В этом случае будем говорить, что имя колонки i является ссылкой или указателем на содержащийся в колонке образ p_i. В свою очередь, часто будет говориться, что образ p_i имеет имя i. Под этим будет подразумеваться то, что образ p_i содержится в колонке $(i \mid p_i)$. Отображение $\varphi : i \to p_i$ будет называться *отображением наименования*.

Имена множества имен U до их использования в качестве имен колонок можно рассматривать как колонки вида $(i \mid \varnothing)$, содержащие пустой образ. Тогда процесс наименования образа

можно представить как заполнение пустой колонки соответствующим образом. Колонки вида $(i \mid \varnothing)$ будут называться *чистыми* или *пустыми именами*. Далее везде под чистым именем i будет пониматься пустая колонка $(i \mid \varnothing)$ или пустая ссылка $i \to \varnothing$. Пустой образ в обозначении чистого имени обычно будет опускаться.

Как уже говорилось, под образами понимаются состоящие из имен конечные множества. В частности, в образы колонок могут входить имена других колонок, т.е. имя в образе одной колонки может быть именем другой колонки и служить указателем на соответствующий образ. В результате образуется показанная на Рис. 2 сложная структура колонок (для наглядности на рисунке дублируются имена колонок).

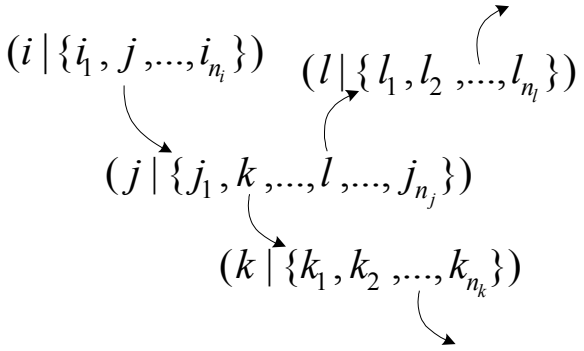


Рис. 2. Структура из колонок

Рассматриваемые простейшие колонки называются *первичными колонками*. Множество первичных колонок будет обозначаться как C^1.

Отличительной особенностью множества первичных колонок C^1 является то, что для наименования образов используются чистые или пустые имена, т.е. колонки вида $(i \mid \varnothing)$, $i \in U'$. Следовательно, любая первичная колонка имеет единичное имя, т.е. имя, не являющееся множеством имен[1]. При этом, так как образы первичных колонок могут содержать только имена первичных колонок (возможно, пустых), то единичному имени

[1] *В колонках общего вида* (*см. далее*) *имя может представлять собой класс эквивалентных имен.*

первичной колонки ставится во взаимно однозначное соответствие образ, также состоящий из единичных имен. Все имена в первичных колонках также будут называться *первичными именами*. Первичное имя – это либо чистое имя, либо имя колонки, которое до его использования для наименования образа, было чистым.

2.3. ФАКТОРИЗАЦИЯ КОЛОНОК

До сих пор рассматривались колонки со взаимно однозначным соответствием между единственным именем и единственным образом. Однако реально может возникнуть необходимость в том, чтобы образ имел несколько имен. Например, если образ описывает некоторый объект, а имя колонки – это имя класса, которому этот объект принадлежит. В общем случае объект может принадлежать одновременно нескольким разным классам. Следовательно, должны иметься несколько колонок с разными именами и одним и тем же образом. Если рассмотреть отображение наименования $\varphi : i \to p$, которое имени ставит в соответствие образ, то все эти имена образуют класс эквивалентности, состоящий из имен с одним и тем же образом, т.е. имеет место *факторизация колонок по образу*. Взаимно однозначное соответствие между именем колонки и образом сохраняется, но как взаимно однозначное соответствие класса имен, имеющих один и тот же образ: $[i] \leftrightarrow p$, где $[i]$ – класс эквивалентных имен. Отображение $\varphi : [i] \to p$, т.е. ссылка на образ, определяется как $\varphi([i]) = \varphi(i_k)$ для $\forall i_k \in [i]$. Таким образом, кроме колонок C^1 необходимо также рассматривать и колонки вида $([i] \mid p_i)$:

$$\begin{array}{ccccc} p & \dots & p & & p \\ \uparrow & \dots & \uparrow & \Rightarrow & \uparrow \\ i_1 & \dots & i_m & & [i] = \{i_1, \dots, i_m\} \end{array}$$

Пусть теперь имеет место противоположный случай, когда несколько образов имеют одно и то же имя. Например, если образы – это описания некоторых объектов, а имя – это имя одного и того же класса, которому эти объекты принадлежат.

Следовательно, должны иметься несколько колонок с одним и тем же именем и разными образами. Теперь класс эквивалентности возникает по отображению φ^{-1}: $p \to i$, которое образу ставит в соответствие его имя. В класс входят все образы с одним и тем же именем, т.е. имеет место *факторизация колонок по имени*. Также, как и в предыдущем случае, взаимно однозначное соответствие сохраняется. Однако на этот раз в виде взаимно однозначного соответствия класса образов, имеющих одно и то же имя: $i \leftrightarrow [p]$, где $[p]$ – класс эквивалентных образов. Отображение φ^{-1}: $[p] \to i$ определяется как $\varphi^{-1}([p]) = \varphi^{-1}(p_k)$ для $\forall p_k \in [p]$. Таким образом, кроме колонок C^1 также должны рассматриваться и колонки вида $(i \mid [p])$:

$$\begin{array}{ccc} p_1 \ldots p_m & & [p] = \{p_1, \ldots, p_m\} \\ \uparrow \ \ldots \uparrow & \Rightarrow & \uparrow \\ i \ \ldots \ i & & i \end{array}$$

2.4. КОЛОНКИ ОБЩЕГО ВИДА

Объединив два рассмотренных выше случая факторизации, получим колонки вида $([i] \mid [p_i])$. Множество таких колонок будет называться *множеством колонок общего вида* и обозначаться через C. Колонки множества C – это колонки, имена которых могут представлять классы эквивалентных имен, а состоящие из них образы – классы эквивалентных образов. При этом причины, на основании которых возникают классы эквивалентности, могут отличаться от тех, что были рассмотрены выше в качестве примера. Очевидно, $C^1 \subset C$.

Колонки множества C можно привести к виду, когда образы колонок не являются классами эквивалентности, т.е. когда классы эквивалентности встречаются только среди имен. Идея преобразования очевидна. Достаточно воспринимать *наши обозначения* образов, входящих в класс, как их имена.

Итак, пусть имеется некоторая колонка $(i \mid [p]) \in C$, где образ колонки – это класс $[p] = \{p_1, \ldots, p_m\}$, $m > 1$.

Используем m чистых имен i_k для того, чтобы наименовать все образы p_k, входящие в класс эквивалентности $[p]$. Получим m колонок $(i_k \mid p_k)$. Сформируем образ $\{i_1, \ldots, i_m\}$, состоящий из

имен этих колонок. Тогда исходную колонку $(i \mid [p])$ можно заменить $m + 1$ колонкой $(i \mid \{i_1, \ldots, i_m\})$, $(i_k \mid p_k)$, $k = 1, \ldots, m$. При этом исходные ссылки $i \to p_k$, имевшиеся в колонке $(i \mid [p])$, заменятся на цепочки ссылок $i \to i_k \to p_k = i \to p_k$:

$$i \to [p] = i \to \{p_1, \ldots, p_m\} \quad \Rightarrow \quad \begin{matrix} p_1 & & p_m \\ \uparrow & \ldots & \uparrow \\ i \to \{i_1, & \ldots, & i_m\} \end{matrix}$$

На месте колонки $(i \mid [p])$ появилась первичная колонка $(i \mid \{i_1, \ldots, i_m\})$ и m колонок $(i_k \mid p_k)$. Если среди элементов в образах p_k есть классы эквивалентности, то действуя аналогичным образом, можно добиться того, что на месте исходной колонки $(i \mid [p])$ образуется конечное множество колонок, у которых классы эквивалентности могут встречаться только среди имен.

Таким образом, колонки множества C образуются с помощью классов имен и состоящих из них образов.

2.5. ЛЕВАЯ И ПРАВАЯ ССЫЛКИ

За счет наименования образов, входящих в класс эквивалентных образов, удалось упростить представление колонок общего вида. Аналогичная замена класса имен $[i]$ на его имя приводит к появлению колонок, имеющих единичное имя с двумя ссылками.

Пусть дана некоторая колонка $([i] \mid p) \in C$, где $[i] = \{i_1, \ldots, i_m\}$, $m > 1$ – класс имен. Возьмем любое чистое имя j и наименуем класс $[i]$. В результате образуется колонка $(j \mid [i])$. Определим ссылку $j \to p$ следующим образом: $j \to p = j \to [i] \to p$. Теперь заменим исходную колонку $([i] \mid p)$ колонкой $(j \mid p)$. В результате вместо колонки $([i] \mid p)$ образовались две колонки $(j \mid [i])$ и $(j \mid p)$ с одним и тем же именем j. Причем несмотря на то, что имя j ссылается на два образа, условий для факторизации по имени нет, так как в данном случае имеются два различных отображения. Отображение φ' служит для на-

именования классов имен, а отображение $\varphi'' = \varphi \circ \varphi'$ [1] для образования ссылки $j \to p = j \to [i] \to p$. Поэтому у имени колонки j сохраняются обе ссылки – *левая* или *именная ссылка* $[i] \leftarrow j$ и *правая* или *образная ссылка* $j \to p$:

$$[i] \to p = \{i_1, \ldots, i_m\} \to p \quad \Rightarrow \quad \{i_1, \ldots, i_m\} \leftarrow j \to p$$

Теперь можно уточнить понятие первичного имени. Легко видеть, что первичное имя – это имя, не имеющее левой ссылки, т.е. левая ссылка пустая, а первичные колонки – это колонки, у которых все без исключения левые ссылки пустые. У чистого или пустого имени пустые обе ссылки – и именная, и образная. В колонках общего вида левые ссылки могут быть как пустыми, так и указывающими на *именные образы*.

Колонку общего вида можно рассматривать как колонку, которая имеет имя-колонку $(j \mid [i])$ и образ, состоящий из колонок, т.е. колонка общего вида – это колонка, все элементы которой также являются колонками. Другими словами, колонка общего вида – это приведенная ниже колонка из колонок (колонка колонок):

$$\begin{array}{l} [i] \quad p_k \quad p_l \quad p_m \\ \uparrow \quad \uparrow \ \ldots \ \uparrow \ \ldots \ \uparrow \\ j \ \to \{k, \ldots, l, \ldots, m\} \end{array}$$

или, если показать левые ссылки в образе,

$$\begin{array}{l} \qquad\quad p_k \quad p_l \quad p_m \\ \qquad\quad \uparrow \ \ldots \uparrow \ \ldots \uparrow \\ [i] \leftarrow j \to \{k, \ldots, l, \ldots, m\} \\ \qquad\quad \downarrow \ \ldots \downarrow \ \ldots \downarrow \quad \text{—} \quad \text{именные ссылки} \\ \qquad\quad i_k \quad i_l \quad i_m \end{array}$$

где $p_k, \ldots, p_l, \ldots, p_m$ – образы, $[i]$ и $i_k, \ldots, i_l, \ldots, i_m$ – именные образы.

---

[1] *По определению,* $(f \circ \varphi)(x) = f(\varphi(x))$.

2.6. ИСПОЛЬЗОВАНИЕ КОЛОНОК В КАЧЕСТВЕ ИМЕН

С точностью до взаимно однозначного соответствия любое конечное множество произвольных объектов можно рассматривать как имена[1]. Следовательно, они могут использоваться для построения колонок. В частности, для этих целей можно взять колонки общего вида из множества C. Попробуем с помощью этого получить более сложные колонки. Очевидно, в результате использования колонок множества C в качестве имен будут получены колонки из колонок вида:

$$\begin{array}{lll} i_j & p_k \; p_l \; p_m & \\ \uparrow & \uparrow \ldots \uparrow \ldots \uparrow & \\ j \to p, & p = \{k, \ldots, l, \ldots, m\} & \\ \downarrow & \downarrow \ldots \downarrow \ldots \downarrow & \text{— именные ссылки} \\ p_j & i_k \; i_l \; i_m & \end{array}$$

где $i_j \leftarrow j \to p_j$ – колонка, взятая в качестве имени.

Полученная колонка имеет одну именную $j \to i_j$ и две образные ссылки $j \to p_j$, $j \to p$, где $p = \{k, \ldots, l, \ldots, m\}$. Ее можно легко преобразовать в колонку общего вида. Для этого возьмем два чистых имени i_1, i_2 и образуем колонки $i_1 \to p_j$, $i_2 \to p$, $i_j \leftarrow j \to \{i_1, i_2\}$ и заменим ими исходную колонку. Очевидно, опять получены колонки множества C, одна из которых имеет непустую левую ссылку.

$$\begin{array}{lll} i_j & & p_j \; p \\ \uparrow & & \uparrow \; \uparrow \\ j \to p & \Rightarrow & i_j \leftarrow j \to \{i_1, i_2\} \\ \downarrow & & \\ p_j & & \end{array}$$

Таким образом, любое использование колонок множества C в качестве имен опять порождает колонки множества C.

---

[1] *Наиболее очевидный пример – звуки, знаки, изображения, иероглифы, узелки на веревках и т.п.*

2.7. ИНДЕКС

Индексом называется любое конечное множество колонок. Если индекс состоит из пустых колонок, он также называется пустым. Если это специально не оговаривается, то считается, что индекс – неупорядоченное множество колонок.

Состав любого индекса может меняться за счет добавления или удаления колонок. Одна колонка – это также индекс, следовательно, множество колонок C входит в множество индексов. Поэтому добавление и удаление колонок индекса – это операции на множестве индексов. Эти операции будут называться сложением и вычитанием индексов и обозначаться через + и –.

Как уже говорилось, любая колонка – это индекс. Интересно, что справедливо и обратное – индекс можно представить как колонку. Действительно, как и любое конечное множество колонок, индекс можно наименовать, используя некоторое чистое имя i для образа, который состоит из имен входящих в индекс колонок. Имя этой колонки-индекса можно интерпретировать как имя индекса.

$$\underbrace{\begin{array}{c} p_k \quad p_l \quad\quad p_m \\ \uparrow \ \ldots \uparrow \ \ldots \uparrow \\ k, \ldots, l, \ldots, m \end{array}}_{\text{индекс как безымянное множество колонок}} \quad \Rightarrow \quad \underbrace{\begin{array}{c} p_k \quad p_l \quad\quad p_m \\ \uparrow \ \ldots \uparrow \ \ldots \uparrow \\ i \to \{k, \ldots, l, \ldots, m\} \end{array}}_{\text{индекс как колонка – именованное множество колонок}}$$

Из этих колонок-индексов в свою очередь могут образовываться индексы и т.д. Можно выделить *уровни индекса по построению*: L^1 – индексы из колонок, L^2 – индексы из колонок-индексов уровня L^1, L^3 – индексы из колонок-индексов уровня L^2 и т.д. Принадлежность колонки-индекса к одному из уровней может интерпретироваться как положение индекса в иерархии индексов (Рис. 3).

В качестве примера рассмотрим конечное множество $A = \{A_1, \ldots, A_n\}$, где A_k – индексы уровня L^1. Легко видеть, что A – это индекс уровня L^2. Для этого достаточно воспринимать *наши обозначения* A_k как имена соответствующих колонок-индексов. Можно явным образом наименовать индексы A_k,

используя для этого чистые имена a_k. В результате будем иметь индекс A уровня L^2, состоящий из колонок-индексов $a_k \to A_k$, где ссылка указывает на образ $\{i_1^k, \ldots, i_{m_k}^k\}$, состоящий из имен колонок индекса A_k.

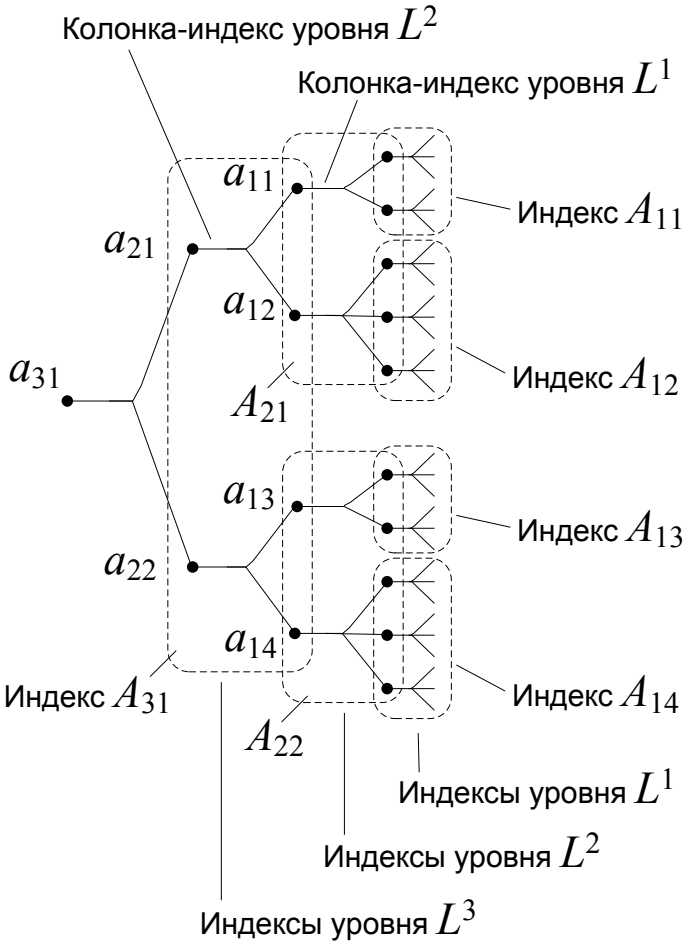


Рис. 3. Уровни индекса

Любой из уровней индекса является множеством колонок и, следовательно, может быть наименован, т.е. представлен в виде колонки. Образ колонки-уровня L^k состоит из имен всех колонок-индексов, образующих уровень L^k. Имя колонки может интерпретироваться как имя уровня L^k.

Вообще, какие бы причины ни приводили к формированию тех или иных множеств колонок, не будет получено ничего кроме колонок. Это демонстрирует одну из основных особенно-

стей модели, основанной на колонках – в мире колонок нет ничего кроме колонок.

2.8. ПРЕДСТАВЛЕНИЕ ИНДЕКСА В ВИДЕ ТАБЛИЦЫ

Индекс A может быть представлен в виде таблицы, состоящей из вертикальных колонок (столбцов) переменной высоты. В нижней строке таблицы, под чертой, имена колонок. Над именем каждой колонки перечислены все имена, входящие в образ колонки. По умолчанию считается, что в таблице имена колонок и имена в образах принадлежат различным областям имен.

Так как индекс – неупорядоченное множество колонок, то порядок записи колонок в таблице может быть произвольным. Если образы представляют собой неупорядоченные множества имен, то и порядок записи имен в образах колонок также может быть произвольным.

Если образы упорядочены, то запись имен в образах колонок выполняется в определенном порядке, например, снизу вверх, т.е. первое имя образа в первой строке над чертой, второе – во второй и т.д.

Индекс, состоящий из первичных колонок может быть сразу записан в виде таблицы. Если индекс содержит колонки, имена которых представляют собой множества имен, то предварительно для каждой из них выполняется дефакторизация – все имена из множества, являющегося именем колонки, получают свою отдельную колонку.

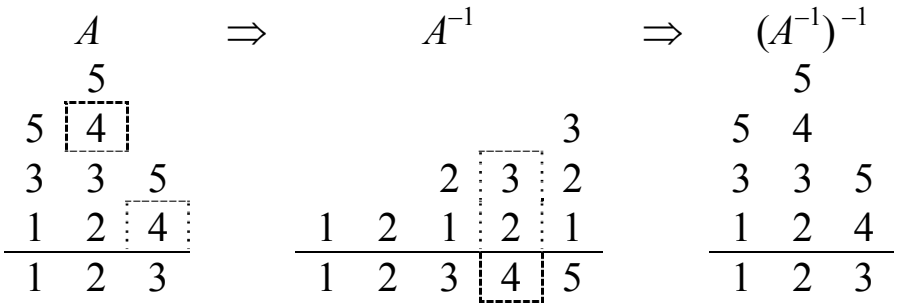
$$\begin{array}{c} p \\ \uparrow \\ \{i_1, \ldots, i_m\} \end{array} \quad \Rightarrow \quad \begin{array}{ccc} p & & p \\ \uparrow & \ldots & \uparrow \\ i_1 & & i_m \end{array}$$

Затем для колонок, имеющих одно и то же имя, выполняется факторизация по имени. При этом в зависимости от задачи образы таких колонок либо объединяются, либо вместо них записываются первичные имена, имеющие ссылки на эти образы. После этого индекс может быть записан в виде таблицы.

Для любого непустого индекса A, представленного в виде таблицы, можно построить *обратный индекс*, который будет обозначаться как A^{-1}. Для этого из A вычитаются (удаляются)

все пустые колонки. Затем берется множество U_p всех имен, содержащихся в образах всех колонок индекса A. Каждое имя, принадлежащее U_p, используются как имя одной из колонок формируемого обратного индекса A^{-1}. При этом в образ колонки индекса A^{-1} с именем $k \in U_p$ включаются те, и только те имена колонок A, которые содержат k в своем образе. Другими словами, если имя $k \in U_p$ содержится только в образах колонок $(i_1 \mid p_1), \ldots, (i_m \mid p_m) \in A$, то колонка обратного индекса A^{-1} с именем k будет иметь вид $(k \mid \{i_1, \ldots, i_m\})$.

Например, в колонку с именем 4 приведенного ниже обратного индекса A^{-1} включены имена колонок 2 и 3 – имена всех колонок индекса A, которые содержат в образе имя 4.



Если для приведенного в примере индекса A^{-1} построить обратный индекс $(A^{-1})^{-1}$, то снова будет получен индекс A. Можно показать [5], что справедливо следующее утверждение: для любого представленного в виде таблицы индекса A, который не содержит пустых колонок, существует единственный обратный индекс A^{-1}, причем $(A^{-1})^{-1} = A$.

2.9. ВЕКТОРА И ПОСЛЕДОВАТЕЛЬНОСТИ

До сих пор тип образов в колонках не конкретизировался. Те образы, которые встречались в тексте и приводились в примерах, представляли собой образы в виде неупорядоченных множеств имен вида $p = \{i_1, \ldots, i_m\}$, $i_k \in U_1$, где U_1 – некоторая область имен. Очевидно, множество всех таких образов P представляет собой множество всех подмножеств области U_1.

Рассмотрим теперь образы, которые представляют собой вектора или конечные последовательности $p = (i_1, \ldots, i_n)$, $i_k \in U_k$, где U_k – область имен k-ой координаты. На области имен U_k не

накладывается никаких ограничений. Они могут как совпадать, так и представлять собой совершенно различные области имен. Множество всех таких образов размерности n представляет собой произведение $P^n = U_1 \times \ldots \times U_n$.

На практике часто необходимо одновременно рассматривать вектора или последовательности различной размерности. В простом варианте, не использующем служебные имена, под множеством всех векторов или последовательностей с размерностью, не превосходящей n, будет пониматься объединение $P = \bigcup_{k=1}^{n} P^k$, где $P^k = U_1 \times \ldots \times U_k$, т.е. любой образ $p \in P$ имеет вид $p = (i_1, \ldots, i_k)$, $i_l \in U_l$, $1 \le k \le n$.

Размерность вектора и последовательности будет обозначаться как $| p |$. Про последовательность также будет говориться, что $| p |$ – это *длина последовательности*.

Несмотря на общее представление векторов и последовательностей, имеются отличия, связанные с интерпретацией *порядковых номеров* координат, т.е. нижних индексов в обозначении координат образа $p = (i_1, \ldots, i_n)$. И для векторов, и для последовательностей порядковые номера позволяют различить координаты, например, имя 2, являющееся 1-й координатой, от имени 2, являющегося 2-й координатой. Однако для последовательностей порядок следования координат в записи $(i_1, \ldots, i_n)$ также может интерпретироваться как порядок следования имен в некотором процессе. Причем, если в векторах для координат обычно используются различные области имен, то в последовательностях все имена могут принадлежать одной и той же области имен, например, области имен состояний.

2.10. ШКАЛА ВРЕМЕНИ, СОБЫТИЯ

Для того чтобы последовательность $p = (i_1, \ldots, i_n)$ можно было рассматривать как временной процесс, достаточно сопоставить порядковым номерам дискретные моменты времени $f_t : k \to t_k$.

Абстрактный образ, который может быть привязан к любому конечному отрезку дискретного времени соответствующей длины, получил название *шкалы времени*. Более точно под

шкалой времени $\tau = (t_1, t_2, \ldots, t_n)$ понимается образ в виде конечной последовательности с именами из области имен моментов времени $T \subset U$. При этом имена, являющиеся координатами шкалы τ, образуют упорядоченную последовательность, т.е. из $k > i$ следует, что $t_k > t_i$, $t_k, t_i \in T$. Также возможны шкалы с обратным порядком, где из $k > i$ следует, что $t_k < t_i$. Тот факт, что момент времени t_k является координатой шкалы времени τ, будет обозначаться как $t_k \in \tau$ и про него будет говориться, что это элемент или момент шкалы τ.

Шкала времени может быть сопоставлена соответствующему отрезку дискретного времени, но это не значит, что она обязательно должна быть сопоставлена такому отрезку. Шкала времени, не связанная с дискретным временем, играет роль *порядковой шкалы*.

Может существовать несколько шкал времени. По умолчанию считается, что элементы различных шкал принадлежат различным областям имен. В качестве одного из вариантов можно представить себе систему шкал, состоящую из основной шкалы времени и нескольких дополнительных шкал. Основная шкала времени имеет очень большую длину и играет роль оси времени. Дополнительные шкалы гораздо меньшей длины служат для тех или иных целей в рассматриваемой проблемной области. Интерпретация имен в различных шкалах, т.е. их соответствие реальным моментам дискретного времени, зависит от рассматриваемых задач.

Для того чтобы привязать колонку $(i \mid p)$ к моменту времени $t \in \tau$, достаточно, используя отображение $\varphi_t : t \to i$, сформировать колонку $(t \mid i)$. В результате образуется цепочка ссылок $t \to i \to p$. При этом факторизация по моменту времени t (факторизация по имени t) приведет к формированию образа из имен всех колонок, относящихся к этому моменту времени:

$$\begin{array}{ccccc} p_1 & p_2 & & p_m & \\ \uparrow & \uparrow & \ldots & \uparrow & \\ i_1 & i_2 & & i_m & \Rightarrow \quad t \to \{i_1, i_2, \ldots, i_m\} \\ \uparrow & \uparrow & \ldots & \uparrow & \\ t & t & & t & \end{array} \qquad \begin{array}{cccc} p_1 & p_2 & & p_m \\ \uparrow & \uparrow & \ldots & \uparrow \end{array}$$

Любая колонка вида $(t \mid p_t)$ называется *событием*, а образ p_t – *состоянием* в момент времени $t \in \tau$. На приведенной выше схеме событиями являются колонки $(t \mid i_k)$ и $(t \mid \{i_1, i_2, \ldots, i_m\})$.

2.11. УПОРЯДОЧЕННЫЙ ИНДЕКС

Наличие образов в виде упорядоченных конечных множеств – векторов и последовательностей, приводит к появлению упорядоченных множеств колонок, т.е. *упорядоченных индексов* или *индексов с упорядоченным множеством колонок*. Ниже показан безымянный упорядоченный индекс, состоящий из колонок $(i_k \mid p_k)$:

$$\begin{matrix} p_1 & p_2 & & p_m \\ \uparrow & \uparrow & \ldots & \uparrow \\ (i_1, & i_2, & \ldots, & i_m) \end{matrix} = \begin{pmatrix} p_1 & p_2 & & p_m \\ \uparrow & \uparrow & \ldots & \uparrow \\ i_1, & i_2, & \ldots, & i_m \end{pmatrix}$$

Практический интерес имеет упорядоченный индекс, колонки которого всегда представляют собой m последних событий.

Пусть имеется некоторая шкала времени $\tau = (1, 2, \ldots, n)$. Рассмотрим упорядоченный индекс A_m, число колонок которого не превосходит m, где $1 < m < n$. Первоначально индекс $A_m = \varnothing$. В момент времени 1 в индекс A_m добавляется колонка $(1 \mid s_1)$, в момент времени 2 – колонка $(2 \mid s_2)$ и т.д., где s_k – состояние в момент времени $k \in \tau$. В момент времени m индекс A_m будет состоять из колонок $(1 \mid s_1)$, $(2 \mid s_2)$, …, $(m \mid s_m)$, т.е. будет содержать m последних событий. Так как индекс A_m не может содержать более чем m колонок, то в момент времени $m + 1$ из него сначала необходимо удалить самую старую колонку $(1 \mid s_1)$, и только потом можно будет добавить самую новую колонку $(m + 1 \mid s_{m+1})$. Индекс A_m по-прежнему будет содержать m последних событий. Аналогично, в момент времени $m + 2$ будет удалена самая старая колонка $(2 \mid s_2)$ и добавлена новая $(m + 2 \mid s_{m+2})$ и т.д. В любой момент времени k, $m \le k < n$, индекс A_m будет содержать m колонок $(k - m + 1 \mid s_{k-m+1})$, …, $(k \mid s_k)$, представляющих собой m последних событий.

3. Прямая и обратная задачи

Знания в рассматриваемых системах представлены в виде образов, а в основе процесса накопления знаний лежит запоминание *новых* образов под определенными именами. При этом базовыми являются задачи, получившие название *прямой* и *обратной задачи* [5]. При решении прямой задачи система пытается определить имя входного образа. Если ей это не удается, то образ является новым и она его запоминает под некоторым именем. При решении обратной задачи система должна по имени образа получить сам образ.

Идея решения этих задач с использованием таблично заданного индекса взята из [9]. Предлагаемый в данной работе метод отличается схемой работы, набором применяемых индексов, а также отсутствием покоординатного сравнения входного образа и образов-кандидатов. Кроме того, для обратной задачи предложен метод решения, в котором для того, чтобы обеспечить компромисс между быстродействием и объемом требуемой памяти, используется упорядоченный индекс.

Сначала рассмотрим решение прямой и обратной задачи для образов в виде конечных неупорядоченных множеств.

3.1. ОБРАЗЫ В ВИДЕ КОНЕЧНЫХ НЕУПОРЯДОЧЕННЫХ МНОЖЕСТВ

3.1.1. ПРЯМАЯ ЗАДАЧА

Будем полагать, что имена, содержащиеся во входных образах, принадлежат некоторой области имен U'. Тогда любой входной образ $p \in P$, где P – множество всех подмножеств множества U', исключая пустое[1].

Для решения прямой задачи система будет использовать индекс A. В начале работы индекс $A = \varnothing$.

Пусть на вход системы пришел некоторый образ $p_1 \in P$. Так как система еще ничего не знает, то p_1 – новый неизвестный

---

[1] *При необходимости образу в виде пустого множества можно сразу присвоить некоторое служебное имя*.

образ. У системы есть область имен U'', которые она может использовать для наименования неизвестных образов. Если через $U_p \subset U''$ обозначить множество имен всех известных образов, то для наименования образа p_1 система может взять любое имя из $U''\backslash U_p$. Пусть выбрано имя i_1. Для запоминания образа p_1 под именем i_1 выполняется сложение $A + (p_1 \mid \{i_1\})$. Это означает, что к индексу A добавляется $n_1 = |\, p_1 \,|$ колонок $(k \mid \{i_1\})$ для всех элементов $k \in p_1$.

Если системе опять будет предъявлен образ p_1, то пересечение n_1 образов колонок индекса A для всех имен $k \in p_1$, очевидно, даст множество, состоящее из одного имени i_1, под которым системе известен образ p_1, т.е. $\bigcap\limits_{k \in p_1} a_k = \{i_1\}$, где a_k – образ колонки $(k \mid a_k) \in A$.

Для произвольного образа $p \in P$ обозначим через $\eta(p)$ пересечение образов колонок $(k \mid a_k) \in A$ для всех имен $k \in p$, т.е. $\eta(p) = \bigcap\limits_{k \in p} a_k$. При этом будем считать, что если в индексе A отсутствует колонка с именем k, то $a_k = \varnothing$, т.е. отсутствующая колонка заменяется пустой колонкой $(k \mid \varnothing)$.

Пусть теперь на вход поступил некоторый образ $p_2 \in P$. Рассмотрим для него пересечение $\eta(p_2)$. Очевидно, равенство $\eta(p_2) = \{i_1\}$ возможно только в том случае, если $p_2 \subset p_1$, $|\, p_2 \,| \le |\, p_1 \,|$. Причем при $|\, p_2 \,| = |\, p_1 \,|$ имеет место равенство $p_2 = p_1$. Если же $p_2 \not\subset p_1$, то всегда $\eta(p_2) = \varnothing$, так как в индексе A просто отсутствуют колонки с именами, принадлежащими $p_2 \setminus p_1$.

Следовательно, возможны три случая:

1. $\eta(p_2) = \{i_1\}$, $|\, p_2 \,| = |\, p_1 \,|$ при $p_2 = p_1$;
2. $\eta(p_2) = \{i_1\}$, $|\, p_2 \,| < |\, p_1 \,|$ при $p_2 \subset p_1$;
3. $\eta(p_2) = \varnothing$ при $p_2 \ne p_1$ и $p_2 \not\subset p_1$.

Если рассматривается вариант задачи, когда система работает с образами одинаковой мощности, возможны только случаи 1 и 3. Тогда равенство $\eta(p_2) = \varnothing$ является признаком того, что образ p_2 неизвестен системе и его надо запомнить. При $\eta(p_2) = \{i_1\}$ образ p_2 известен системе под именем i_1.

Если же на вход системы поступают образы различной мощности, то равенство $\eta(p_2) = \varnothing$ также является признаком того, что образ p_2 неизвестен системе и его надо запомнить. Однако при $\eta(p_2) = \{i_1\}$ возможны случаи 1 и 2 – либо входной образ известен под именем i_1, либо входной образ неизвестен, но представляет собой подмножество образа, известного под именем i_1. Чтобы выделить случай 2, необходимо выполнить сравнение мощностей образов. Для этого в системе используется функция $n(i)$, которая определена на множестве U_p и задана с помощью множества пар (аргумент, значение). Каждый раз, когда система запоминает новый образ p, в определение функции $n(i)$ добавляется новая пара $(i, | p |)$, где i – имя, под которым запоминается образ p, $| p |$ – его мощность.

Вернемся к моменту, когда на вход системы поступил некоторый образ p_2. Если $\eta(p_2) = \{i_1\}$ и $n(i_1) = | p_2 |$, то образ p_2 известен системе, $p_2 = p_1$, и на ее выходе появляется имя i_1.

Если $\eta(p_2) = \varnothing$ или $\eta(p_2) = \{i_1\}$, но $n(i_1) \neq | p_2 |$, то образ p_2 неизвестен и его надо запомнить. Системой выбирается любое имя $i_2 \in U'' \backslash U_p$, под которым теперь будет известен образ p_2, и выполняется сложение $A + (p_2 \mid \{i_2\})$. В определение функции $n(i)$ добавляется новая пара $(i_2, | p_2 |)$, и на выходе системы появляется имя i_2.

Очевидно, при выполнении сложения $A + (p_2 \mid \{i_2\})$ факторизация по имени приведет к тому, что любая колонка индекса A с именем $k \in p_1 \bigcap p_2$ будет иметь вид $(k \mid \{i_1, i_2\})$. Также легко заметить, что если выполнить факторизацию по образу, то полученный в результате сложения индекс A будет состоять из трех колонок $(p_1 \setminus p_2 \mid \{i_1\})$, $(p_1 \bigcap p_2 \mid \{i_1, i_2\})$ и $(p_2 \setminus p_1 \mid \{i_2\})\}$.

Аналогичным образом система будет запоминать новые образы p_3, p_4 и т.д. Запоминание l новых образов можно представить в виде цепочки сложений:

$$A = \sum_{k=1}^{l} (p_k \mid \{i_k\}),\ n(i) = \bigcup_{k=1}^{l} (i_k, | p_k |).$$

Общая схема работы системы при решении прямой задачи для поступившего на вход образа $p \in P$ можно представить в виде:

1. Для входного образа p вычислить пересечение $\eta(p)$.
2. Если $\eta(p) \neq \varnothing$ и для некоторого $i \in \eta(p)$ выполняется равенство $n(i) = |\, p \,|$, то образ p известен системе под именем i. Вернуть имя i в качестве имени образа p. Перейти к пункту 4.
3. Если $\eta(p) = \varnothing$ или $\eta(p) \neq \varnothing$, но $n(i) \neq |\, p \,|$ для $\forall i \in \eta(p)$, то образ p является новым и его необходимо запомнить под некоторым именем. Выбрать для этого любое имя $i \in U''\backslash U_p$ и выполнить сложение $A + (p \mid \{i\})$. Кроме того, в определение функции $n(i)$ добавить пару $(i, |\, p \,|)$. Вернуть имя i в качестве имени образа p.
4. Конец.

Покажем, что для построенных таким образом индекса A и функции $n(i)$ при появлении любого образа $p \in P$ всегда можно однозначно сказать, известен ли этот образ, и что существует взаимно однозначное соответствие между известными образами и их именами.

Пусть $\eta(p) = \varnothing$. Это означает, что образ p не предъявлялся системе, так как в противном случае, по построению, существовали бы колонки для $\forall k \in p$ и пересечение $\eta(p)$ содержало бы имя, под которым известен образ p.

Пусть теперь $\eta(p) \neq \varnothing$ и для $\forall i \in \eta(p)$ выполняется $n(i) \neq |\, p \,|$. Это также означает, что образ p не предъявлялся системе, так как в противном случае, по построению, пересечение $\eta(p)$ содержало бы имя i, под которым известен образ p, и выполнялось бы равенство $n(i) = |\, p \,|$.

Наконец, пусть $\eta(p) \neq \varnothing$ и существует по крайней мере одно имя $i \in \eta(p)$, для которого $n(i) = |\, p \,|$. Для любого такого i это означает, что образ p уже известен под этим именем. Предположим, что $\exists i, i' \in \eta(p)$, $i \neq i'$ и $n(i) = n(i') = |\, p \,|$. Так как образ p не может получить сразу два имени, то это означает, что он сначала получил одно имя, а потом в одном из повторных предъявлений – второе. Это невозможно по построению, т.е. $i = i'$, и если существует имя $i \in \eta(p)$, для которого $n(i) = |\, p \,|$, то такое имя единственное.

Таким образом, доказано следующее утверждение: можно построить такую систему на основе колонок, которая решает прямую задачу для $\forall p \in P$.

ЗАМЕЧАНИЕ. Выбор для нового образа имени из множества $U''\backslash U_p$ обеспечивает выполнение условия, которое является *необходимым условием* для правильной работы приведенного метода. Это условие состоит в том, что каждый образ получает единственное уникальное имя, т.е. разные образы не могут иметь одно и то же имя.

Легко показать, что при нарушении указанного условия система не сможет различать образы.

Действительно, пусть два разных образа p_1 и p_2 имеют одно и тоже имя i и $|p_1| = |p_2| = n$. Рассмотрим любой образ p_3, $|p_3| = n$, который имеет часть элементов из образа p_1, а оставшуюся часть из образа p_2. Очевидно, образ p_3 не равен образам p_1 и p_2, но при этом $\eta(p_3) = \eta(p_1) = \eta(p_2) = i$ и $|p_3| = |p_1| = |p_2| = n(i)$, т.е система не сможет отличить образ p_3 от образов p_1 и p_2.

3.1.2. ОБРАТНАЯ ЗАДАЧА

При решении обратной задачи система должна по имени образа i получить сам образ p_i.

Итак, пусть на входе имеется некоторое имя $i \in U''$. Мы можем проверить $i \in U_p$, т.е. установить, принадлежит ли имя i множеству имен известных образов[1]. Если $i \notin U_p$, такой образ неизвестен системе, и на выходе появляется множество из одного служебного имени $i_\varnothing$, означающего в данном случае, что входное имя i – это чистое имя $(i \mid \varnothing)$.

Если $i \in U_p$, то образ под этим именем известен и его необходимо показать на выходе системы. Имея индекс A и функцию $n(i)$, искомый образ p_i можно восстановить, просмотрев колонки индекса A. Если $i \in a_k$ для колонки $(k \mid a_k) \in A$, то, по построению A, имя $k \in p_i$. Поэтому имя k добавляется в формируемый

---

[1] *Это достаточно просто выполнить, например, с помощью функции* $n(i)$.

образ p_i. Найдя все $n(i)$ имен образа p_i, процесс просмотра колонок индекса A можно остановить. Схема решения обратной задачи для $\forall i \in U''$ примет вид:

1. Пусть $i \in U''$ – имя, появившееся на входе. Проверить условие $i \in U_p$.
2. Если $i \notin U_p$, образ под таким именем неизвестен системе, на выходе появляется $i_\varnothing$. Перейти к пункту 4. Если $i \in U_p$, перейти к следующему пункту, чтобы восстановить образ p_i.
3. Просмотреть колонки индекса A. Если $i \in a_k$ для колонки $(k \mid a_k) \in A$, то добавить имя k в искомый образ p_i. Если найдено $n(i)$ имен, входящих в p_i, показать p_i на выходе и перейти к следующему пункту. В противном случае продолжить просмотр колонок индекса A.
4. Конец.

Таким образом, доказано следующее утверждение: можно построить систему на основе колонок, которая решает обратную задачу для $\forall i \in U''$, причем для $\forall i \in U_p$ на выходе можно получить единственный образ p_i, который известен машине под именем i.

Несмотря на простоту приведенной схемы решения обратной задачи, при больших размерах индекса A для восстановления образа p_i может потребоваться значительное время. Чтобы не восстанавливать образ, его можно сохранить.

Для этого необходимо внести изменения в схему решения прямой задачи. Теперь кроме сложения $A + (p \mid \{i\})$ и добавления пары $(i, \mid p \mid)$ в определение функции $n(i)$ будет выполняться и сохранение образа p. Для этого система будет использовать еще один индекс B. В начале работы, когда система ничего не знает, $A = \varnothing$ и $B = \varnothing$. При сохранении образа p будет выполняться сложение $B + (i \mid p)$. Таким образом, процесс запоминания и сохранения l неизвестных образов можно представить цепочкой сложений:

$$A = \sum_{k=1}^{l} (p_k \mid \{i_k\}),\ B = \sum_{k=1}^{l} (i_k \mid p_k),\ n(i) = \bigcup_{k=1}^{l} (i_k, \mid p_k \mid).$$

Использование индекса B существенно упрощает решение обратной задачи. Для этого, очевидно, системе достаточно для $\forall i \in U_p$ вернуть образ колонки $(i \mid b_i) \in B$.

Нетрудно заметить, что $B = A^{-1}$. Для доказательства достаточно использовать определение обратного индекса по отношению к колонкам $(p_k \mid \{i_k\})$ и $(i_k \mid p_k)$.

ПРИМЕР. Пусть индексы A, B и функция $n(i)$ имеют вид:

A

3	3	3	
1	2	1	2
1	2	3	4

$n(i)$

i	1	2	3
n	2	2	3

B

			3
3	4	2	
1	2	1	
1	2	3	

Легко видеть, что система знает три образа: образ $\{1, 3\}$ под именем 1, образ $\{2, 4\}$ под именем 2 и образ $\{1, 2, 3\}$ под именем 3.

Пусть на входе системы появился образ $p = \{1, 2, 3\}$. Пересечение $\eta(p) = \{3\}$ и $n(3) = |p| = 3$. Следовательно, образ p известен системе под именем 3.

Пусть теперь на входе появился образ $p = \{2, 3\}$. Пересечение $\eta(p) = \{3\}$, но $n(3) = 3 \neq |p|$. Следовательно, образ p неизвестен. Для него выбирается имя $4 \in U''\backslash U_p$. Выполняется сложение $A + (p \mid \{4\})$ и $B + (4 \mid p)$, а в определение функции $n(i)$ добавляется пара (4, 2). В результате получим:

A

	4	4	
3	3	3	
1	2	1	2
1	2	3	4

$n(i)$

i	1	2	3	4
n	2	2	3	2

B

			3	
3	4	2	3	
1	2	1	2	
1	2	3	4	

Если на входе опять появится образ $p = \{2, 3\}$, то $\eta(p) = \{3, 4\}$, причем $n(4) = |p| = 2$, что означает, что образ p известен системе под именем 4.

Пусть теперь решается обратная задача и на вход системы пришло имя $i = 3$. Так как $i \in U_p$, то образ p_i равен образу колонки 3 индекса B, т.е. $p_i = \{1, 2, 3\}$. ■

Приведенное выше быстрое решение обратной задачи имеет один недостаток – в индексе B должны сохраняться *все* известные образы. В результате значительно возрастает объем требуемой памяти. Очевидным выходом является сохранение в индексе B лишь части известных образов, например, тех, которые встречаются на отрезке из m последних моментов времени. При этом в качестве индекса B используется упорядоченный индекс, содержащий m последних событий, каждое из которых связано с появлением некоторого образа на входе системы. Отличие схемы пополнения такого индекса, от схемы, рассмотренной в разделе 2.11, состоит лишь в том, что в данном случае индекс не содержит повторяющихся состояний. Если образ p уже появлялся на входе в момент времени t, где $k - m \le t < k$, и хранится в индексе B, то сначала соответствующая колонка удаляется, а затем образ p заносится в индекс для текущего момента времени k. В результате для образов, встречающихся на отрезке из m последних моментов времени, обратная задача будет решаться быстро. Для остальных будет использоваться более медленная схема восстановления.

3.2. ОБРАЗЫ В ВИДЕ КОНЕЧНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ ИЛИ ВЕКТОРОВ

Рассмотрим теперь прямую и обратную задачи для образов в виде конечных упорядоченных множеств – конечных последовательностей или векторов. Будем полагать, что в этом случае, образы $p = (i_1, i_2, \ldots, i_m)$, поступающие на вход системы, принадлежат множеству $P = \bigcup_{m=1}^{n} P^m$, где $P^m = U_1 \times \ldots \times U_m$, U_k – область имен k-ой координаты.

В прямой задаче, как и ранее, если поступивший на вход системы образ $p \in P$ неизвестен, то его необходимо запомнить под некоторым именем из области имен U'' и вернуть это имя в качестве имени образа. Если же образ p известен системе, то его

необходимо узнать и вернуть его имя. В обратной задаче необходимо по имени восстановить соответствующий образ.

При решении прямой задачи будет использоваться индекс $A = \{A_1, A_2, \ldots, A_n\}$, который представляет собой индекс уровня L^2, состоящий из n колонок-индексов A_k (см. раздел 2.7), где A_k – индекс для k-ой координаты.

Кроме индекса A также будет использоваться функция $m(i)$, которая будет содержать длину известных последовательностей. В исходном состоянии $A_k = \varnothing$ $(k = 1, \ldots, n)$, $m(i) = \varnothing$. Запоминание нового образа $p = (i_1, i_2, \ldots, i_m)$ под некоторым именем i выполняется как покоординатное сложение $A_k + (i_k \mid \{i\})$, $k = 1, \ldots, m$, где i_k – имя, являющимся k-ой координатой входного образа p:

$$A + (p \mid \{i\}) = \{A_1 + (i_1 \mid \{i\}),\ \ldots,\ A_m + (i_m \mid \{i\})\}.$$

Кроме этого, в определение функции $m(i)$ добавляется пара $(i, \mid p \mid)$. Таким образом, запоминание системой l неизвестных образов можно представить в виде цепочки сложений:

$$A = \sum_{k=1}^{l} (p_k \mid \{i_k\}),\ m(i) = \bigcup_{k=1}^{l} (i_k, \mid p_k \mid),$$

где i_k – имя образа p_k.

Пусть на вход системы пришел некоторый образ $p = (i_1, i_2, \ldots, i_m)$. Рассмотрим покоординатное пересечение $\eta(p) = \bigcap_{k=1}^{m} a_{i_k}$, где a_{i_k} – образ колонки $(i_k \mid a_{i_k}) \in A_k$, i_k – имя, являющееся k-ой координатой входного образа p.

Очевидно, если образ p неизвестен и не является подпоследовательностью известных образов, то $\eta(p) = \varnothing$. Если $\eta(p) \neq \varnothing$ и $\mid p \mid \neq m(i)$ для $\forall i \in \eta(p)$, то образ p также неизвестен, но является подпоследовательностью известных образов с именами из множества $\eta(p)$. Наконец, если $\eta(p) \neq \varnothing$ и существует имя $i \in \eta(p)$, такое, что $\mid p \mid = m(i)$, то образ p известен под именем i. Причем в последнем случае такое имя единственное, так как повторное запоминание известного образа под другим именем невозможно.

Как и ранее необходимым условием для правильной работы метода является то, что каждый образ получает единственное

уникальное имя, что обеспечивается выбором этого имени из множества $U''\backslash U_p$. При нарушении этого условия система не сможет различать образы.

Действительно, пусть образы p_1 и p_2, имеющие одно и то же имя i, отличаются всеми координатами и $|p_1| = |p_2|$. Тогда, взяв образ той же размерности, у которого одна часть координат равна координатам образа p_1, а вторая часть – координатам образа p_2, получим образ p_3, отличный от p_1 и p_2. При этом, очевидно, $\eta(p_3) = \eta(p_1) = \eta(p_2) = i$ и $|p_3| = |p_1| = |p_2| = m(i)$, т.е система не сможет отличить образ p_3 от образов p_1 и p_2.

Аналогично тому, как это делалось для неупорядоченных множеств, решается и обратная задача. По имени образа i можно либо восстановить образ p, либо использовать дополнительный индекс B, с которым при запоминании образа p складывается колонка $(i \mid p)$. В этом случае запоминание l неизвестных образов можно представить цепочкой сложений:

$$A = \sum_{k=1}^{l} (p_k \mid \{i_k\}),\ B = \sum_{k=1}^{l} (i_k \mid p_k),\ m(i) = \bigcup_{k=1}^{l} (i_k, \mid p_k \mid),$$

где i_k – имя образа p_k.

Итак, справедливо следующее утверждение: можно построить систему на основе колонок, которая решает прямую задачу для любого образа $p \in \bigcup_{m=1}^{n} P^m$ и обратную задачу для любого имени $i \in U''$, где $P^m = U_1 \times \ldots \times U_m$, U_k – область имен k-ой координаты, U''– область имен для наименования образов.

ПРИМЕР. Пусть для $n = 3$ имеются следующие индексы A, B и функция $m(i)$:

A_1			A_2			A_3			B			
									3	2	2	
4		2	2			4			3	1	1	2
1		3	3	4	1		3	2	1	3	3	1
1	2	3	1	2	3	1	2	3	1	2	3	4

$m(i)$

i	1	2	3	4
m	2	3	3	3

Легко видеть, что в индексе A под именем 1 запомнен образ (1, 3), под именем 2 – образ (3, 1, 3), под именем 3 – образ (3, 1, 2), а под именем 4 – образ (1, 2, 2).

Пусть на вход пришел образ $p = (3, 1, 2)$. Пересечение $\eta(p)$, равное пересечению образа колонки индекса A_1 с именем 3, образа колонки индекса A_2 с именем 1 и образа колонки индекса A_3 с именем 2, будет состоять из одного имени 3, при этом $m(3) = 3$. Это означает, что на входе образ, известный системе под именем 3.

Пусть теперь на входе появился образ $p = (3, 1)$. Пересечение $\eta(p) = \{2, 3\}$, но $m(2) = m(3) \neq 2$. Следовательно, p – неизвестный образ. Из множества $U'' \backslash U_p$ для него выбирается имя 5. Чтобы запомнить образ p под этим именем, выполняются сложения $A + ((3, 1) \mid 5)$ и $B + (5 \mid (3, 1))$, а в определение функции $m(i)$ добавляется пара (5, 2). В результате получим:

A_1			A_2			A_3			B				
		5	5							3	2	2	
4		2	2				4		3	1	1	2	1
1		3	3	4	1		3	2	1	3	3	1	3
1	2	3	1	2	3	1	2	3	1	2	3	4	5

$m(i)$

i	1	2	3	4	5
m	2	3	3	3	2

Если на входе снова появится образ $p = (3, 1)$, то пересечение $\eta(p) = \{2, 3, 5\}$, причем $m(5) = 2$. Следовательно, на входе образ, известный системе под именем 5.

Пусть теперь решается обратная задача и на вход пришло имя $i = 3$. Так как $i \in U_p$, то образ p_i равен образу колонки 3 индекса B, т.е. $p_i = (3, 1, 2)$.

3.3. ВЫЧИСЛИТЕЛЬНАЯ ЭФФЕКТИВНОСТЬ

Пусть решается прямая задача для образов в виде конечных последовательностей или векторов одинаковой размерности. Пусть также l – число запомненных образов, n – размерность вектора, m – число возможных значений каждой координаты.

Рассмотрим число операций, которые необходимо выполнить при решении прямой задачи обычным методом покоординатного сравнения с образцом и с помощью индекса.

Метод покоординатного сравнения предполагает, что в памяти хранятся l известных системе образов, которые выступают в качестве образцов. Если на входе появился некоторый вектор $p = (i_1, i_2, \ldots, i_n)$, то для поиска наиболее подходящего кандидата выполняется покоординатное сравнение этого вектора со всеми образцами. Если такой образец найден, то его имя является решением прямой задачи. Максимальное число операций сравнения, которые при этом необходимо выполнить, равно $O_M = nl$.

Число операций можно сократить, используя схему индексации, например, на основе вычисления хэш-функции по первой координате вектора. Так, если при поиске хэш-функция разбивает l образцов на k_h групп, то необходимое число операций становится равным $O_{Mh} = nl / k_h$ [1].

Оценим теперь число операций, необходимое для решения прямой задачи с помощью индекса $A = \{A_1, A_2, \ldots, A_n\}$. Каждая из m колонок индекса A_k ($k = 1, \ldots, n$) содержит в среднем l / m имен. При вычислении пересечения $\eta(p)$ необходимо выполнить $O_A = nl / m$ операций, т.е. в m раз меньше чем при покоординатном сравнении. Сравнение с оценкой O_{Mh} показывает, что индекс можно рассматривать как достаточно эффективную схему индексации, для которой чем больше разнообразие возможных имен для значений координат, тем меньше операций надо вы-

---

[1] *К указанной оценке необходимо добавить число операций, необходимое для однократного вычисления самой хэш-функции для входного вектора p. В силу того, что, как правило, используются достаточно простые хэш-функции, этой добавкой можно пренебречь.*

полнить[1]. Другими словам, *чем больше имен использует система для представления значений координат, т.е. чем точнее она отражает реальные объекты, тем она быстрее работает*. При большом m выигрыш по сравнению с покоординатным сравнением может быть довольно значительным. Например, пусть запомнено $l = 10^5$ образов размерности $n = 10^2$, каждая координата которых может иметь $m = 10^3$ значений. Тогда при покоординатном сравнении необходимо выполнить 10 млн. операций, а при вычислении пересечения $\eta(p)$ только 10 тыс. операций.

Оценки памяти, необходимой для этих двух методов, совпадают. И в том, и в другом случае в памяти должно храниться nl целых чисел.

ЗАМЕЧАНИЕ. При сравнении оценок быстродействия следует иметь в виду, что методы с использованием таблично заданного индекса очень хорошо распараллеливаются. Все множество колонок разбивается на части по числу параллельных потоков. Для каждой из этих частей выполняются необходимые вычисления, например, определяется пересечение $\eta(p)$, затем результаты объединяются.

4. Задача классификации и булевы функции

Базовые задачи служат основой для решения других задач. В данном разделе это сначала будет показано на примере задачи классификации, а затем будет доказана возможность реализации в рассматриваемых системах произвольных булевых функций. Эти результаты позволят оценить классы задач, для которых могут применяться интеллектуальные системы на основе колонок.

---

[1] *Аналогичное утверждение справедливо и для образов в виде неупорядоченных множеств. В этом случае m – число различных имен, участвующих в формировании образов*.

4.1. ЗАДАЧА КЛАССИФИКАЦИИ

Рассмотрим задачу классификации, в которой любой объект в некоторой предметной области описывается вектором свойств и принадлежит одному или нескольким классам объектов. Необходимо по входному образу в виде вектора определить, к какому классу или классам принадлежит описываемый этим вектором объект. Основное отличие данной задачи от рассмотренных ранее связано с наименованием неизвестных образов. Если ранее система присваивала неизвестному образу любое имя из имеющихся у нее чистых имен, то теперь она должна связать этот образ с совершенно определенным множеством имен классов.

Будем полагать, что для каждого нового образа система имеет возможность получить множество имен классов, к которым этот образ принадлежит. Тогда для решения задачи классификации можно воспользоваться полученным ранее результатами. Действительно, если входной образ p является новым, то достаточно его запомнить под именем i_p, а в индекс B добавить колонку $(i_p \mid q)$, где $q = \{c_1, \ldots, c_r\}$ – множество имен классов, к которым принадлежит образ p, $c_k \in U_q$, U_q – область имен классов. Тогда при предъявлении образа p сначала определяется его имя i_p, а затем по нему колонка $(i_p \mid q)$ индекса B. Образ этой колонки q и есть множество имен классов, к которым принадлежит входной образ p.

Очевидно, индексы A и B в паре образуют средство для решения прямой задачи для колонок вида $q \to p$, где q – множество имен классов, обеспечивая для известных системе векторов p переход по ссылкам $p \to i_p \to q$.

Рассмотрим решение задачи классификации более подробно. Пусть образы на входе системы принадлежат множеству P вида:

$$P = \bigcup_{m=1}^{n} P^m, \; P^m = U_1 \times \ldots \times U_m,$$

где U_k – область имен k-ой координаты.

Для запоминания используются индексы $A = \{A_1, \ldots, A_n\}$, B и функция $m(i)$, которая содержит размерности известных векторов. В исходном состоянии $A_k = \varnothing$ $(k = 1, \ldots, n)$, $B = \varnothing$ и

$m(i) = \varnothing$. Если $p = (i_1, i_2, \ldots, i_m)$ – новый образ, который принадлежит классам $q = \{c_1, \ldots, c_r\}$, то для его запоминания система выбирает некоторое имя $i \in U'' \setminus U_p$ и выполняет сложения:

$$A + (p \mid \{i\}) = \{A_1 + (i_1 \mid \{i\}), \ldots, A_m + (i_m \mid \{i\})\},$$

$$B + (i \mid q),$$

где i_k – имя, являющееся k-ой координатой образа p. Кроме этого, в функцию $m(i)$ добавляется пара (i, m).

Таким образом, запоминание l неизвестных образов можно представить цепочкой сложений:

$$A = \sum_{k=1}^{l} (p_k \mid \{i_k\}),\ B = \sum_{k=1}^{l} (i_k \mid q_k),\ m(i) = \bigcup_{k=1}^{l} (i_k, m_k),$$

где i_k – имя образа p_k, q_k – множество имен классов, которым принадлежит вектор p_k, m_k – размерность p_k.

Пусть на вход системы пришел некоторый вектор $p = (i_1, i_2, \ldots, i_m)$. Для него обычным образом вычисляется покоординатное пересечение $\eta(p) = \bigcap_{k=1}^{m} a_{i_k}$. Если $\eta(p) \neq \varnothing$ и $\exists i \in \eta(p)$, для которого $m = m(i)$, то p – это образ, который известен системе под именем i и принадлежит классам с именами из множества b_i, где $(i \mid b_i)$ – колонка индекса B. Во всех остальных случаях образ p является новым и его необходимо запомнить, используя соответствующее множество имен классов q.

Доказано следующее утверждение: можно построить систему на основе колонок, которая решает приведенную выше задачу классификации.

4.2. БУЛЕВЫ ФУНКЦИИ

Рассмотрим произвольную булеву функцию $f: B^n \to B$, $B = \{0, 1\}$. Она определяет разбиение B^n на классы эквивалентности:

$$B_0^n = \{x \in B^n \mid f(x) = 0\},$$

$$B_1^n = \{x \in B^n \mid f(x) = 1\},$$

где $B_0^n \bigcup B_1^n = B^n$, $B_0^n \bigcap B_1^n = \varnothing$.

Очевидно, для того чтобы реализовать функцию f с помощью системы на основе колонок, достаточно решить задачу

классификации для двух классов B_0^n и B_1^n. Положим для наглядности, что классы B_0^n и B_1^n получили имена 0 и 1 из некоторой области имен U_q. Тогда для любого вектора $p \in B^n$ будем иметь: если p принадлежит классу с именем 0, то $f(p) = 0$; если же p принадлежит классу с именем 1, то $f(p) = 1$.

Так как мы имеем дело с разбиением множества B^n всего на два класса, то задачу можно еще более упростить, оставив только один класс, например, B_1^n. Тогда для $\forall p \in B^n$, если p принадлежит классу 1, то $f(p) = 1$, в противном случае $f(p) = 0$. При этом, так как все вектора имеют одну и ту же размерность, а класс всего один, то при решении задачи классификации функцию $m(i)$ и индекс B можно не использовать. В результате решение, обеспечивающее реализацию булевой функции $f: B^n \to B$, будет выглядеть следующим образом.

Пусть образы представляют собой вектора $p \in P$, $P = U_B^n$, где $U_B = \{0, 1\}$ – область имен для любой координаты.

Сначала при помощи индекса $A = \{A_1, \ldots, A_n\}$ запоминаются все вектора класса B_1^n, что можно представить в виде цепочки сложений:

$$A = \sum_{k=1}^{l} (p_k \mid \{i_k\}),$$

где $p_k \in B_1^n$, i_k – имя вектора p_k, $l = \mid B_1^n \mid$.

В результате только вектора класса B_1^n получают имена и становятся известными системе. Других векторов она на знает.

Далее система работает только в режиме классификации. Пусть $p \in P$ – произвольный входной вектор. Если $\eta(p) \neq \varnothing$, то вектор p известен системе, $p \in B_1^n$ и $f(p) = 1$. В противном случае $f(p) = 0$.

Таким образом, доказано следующее утверждение: любая булева функция $f: B^n \to B$, $B = \{0, 1\}$ может быть реализована в системе на основе колонок.

Этот результат показывает возможность решения с помощью рассматриваемых систем тех задач, для которых может

быть использован логический подход. Это относится и к логическому выводу, используемому, в частности, в синтетических задачах. Вместе с результатами раздела 4.1 это позволяют говорить о принципиальной возможности применения интеллектуальных систем на основе колонок для решения различных задач классификационного и синтетического типа.

Литература

1. ВИШНЯКОВ В.А., БУЛАНЖЕ Д.Ю., ГЕРМАН О.В. *Аппаратно-программные средства процессоров логического вывода*. – М.: Радио и связь, 1991. – 264 с.
2. МАУНТКАСЛ В. *Организующий принцип функции мозга – элементарный модуль и распределенная система*. // Эдельмен Дж., Маунткасл В. Разумный мозг. – М.: Мир, 1981. – С. 15-67.
3. ХОГГЕР К. *Введение в логическое программирование*. – М.: Мир, 1988. – 348 с.
4. ХОКИНС ДЖЕФФ. *Об интеллекте*. – М.: Вильямс, 2007. – 240 с.
5. ЧЕСНОКОВ А.М. *Введение в общую теорию колонок* / Научное издание. – М.: ИПУ РАН. – 2012. – 141 с.
6. BRIN S., PAGE L. *The Anatomy of a Large-Scale Hypertextual Web Search Engine*. // Proceedings of Seventh International Conference on World Wide Web (WWW7) – Amsterdam: Elsevier, 1998. – P. 107-117.
7. HAWKINS JEFF. *On Intelligence*. – New York: Henry Holt and Company, 2005.
8. MIKHAILOV A. *Digital neural cortex*. // Proceedings of the Artificial Neural Networks in Engineering Conference (ANNIE 2007), November, 1-4, 2007, St. Louis, Missouri, USA.
9. MIKHAILOV A. *Biologically Inspired Artificial Neural Cortex and its Formalism*. // World Academy of Science, Engineering and Technology. – August 2009. – Vol.56. – P. 121.

COLUMNS-BASED INTELLIGENT SYSTEMS

Alexander Chesnokov, Institute of Control Sciences of RAS, Moscow, Cand. Sc. (alex-ches@yandex.ru).

Abstract: *The paper considers columns-based intelligent systems and introduces basic definitions and notions. The direct and inverse problems of patterns that are represented by finite unordered and ordered (finite sequences and vectors) sets are introduced. Solutions to these problems and their computational complexity are discussed. Also, a solution to pattern classification problem and a method of implementing Boolean functions are introduced. Finally, a class of problems that can be solved with columns-based intelligent systems is identified.*

Keywords: artificial intelligence, columns-based intelligent systems, column, memory-prediction model.