Информационный анализ

УДК 519.688

С. О. Кузнецов

БЫСТРЫЙ АЛГОРИТМ ПОСТРОЕНИЯ ВСЕХ ПЕРЕСЕЧЕНИЙ ОБЪЕКТОВ ИЗ КОНЕЧНОЙ ПОЛУРЕШЕТКИ

Предлагается алгоритм построения всех элементов полурешетки по множеству порождающих элементов G. Задача вычисления числа всех элементов $\#P$-полна даже в случае булевой полурешетки, т. е. когда G представляет собой семейство подмножеств некоторого множества U. Само количество пересечений может быть экспоненциально от размера множества порождающих элементов. Предложенный алгоритм может считаться оптимальным, поскольку порождает все пресечения за время, линейное от их числа.

1. ВСТУПЛЕНИЕ. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

Ряд информационных проблем, связанных с обработкой данных требует нахождения сходства описаний объектов. В том случае, когда объекты представляются множествами, сходство может выражаться через теоретико-множественное пересечение их описаний. Для данных более сложной природы сходство может задаваться полурешеточной операцией, т. е. операцией, обладающей следующими свойствами для произвольных объектов X, Y, Z из области ее действия:

$X\sqcap X=X$ (идемпотентность),

$X\sqcap Y=Y\sqcap X$ (коммутативность),

$(X\sqcap Y)\sqcap Z=X\sqcap(Y\sqcap Z)$ (ассоциативность).

$X\sqcap 0=0$ для некоторого элемента 0 полурешетки, называемого нулевым.

Операция $\sqcap$ задает естественным образом порядок $\sqsubseteq$: $X\sqsubseteq Y\Leftrightarrow X\sqcap Y=X$.

В некоторых информационных системах поиск сходства ограничивается попарными или n-арными пересечениями для фиксированного n [1], однако в ряде исследований [2—6] необходимо найти все пересечения, образованные произвольным количеством объектов из базы данных. Алгоритмы, решающие эту задачу, имеют экспоненциальную сложность даже в случае представления данных множествами в связи с экспоненциальным числом возможных пересечений [7], а определение числа различных пересечений есть $\#P$-полная задача [8]. Эти обстоятельства делают необходимым построение алгоритмов, решающих указанную проблему самым эффективным образом. Ниже мы предложим алгоритм нахождения пересечений для произвольной полурешетки, оценка быстродействия которого позволяет говорить о его оптимальности в некотором смысле.

При определении основных понятий в целях удобства воспользуемся обозначениями из [3].

Пусть S — конечное множество объектов, для которых определена идемпотентная, коммутативная и ассоциативная операция $\sqcap$.

Определение 1.1. Пусть $X_1, \ldots, X_k \in S$, тогда $(\{X_1, \ldots, X_k\})'=X_1\sqcap\ldots\sqcap X_k$. Если $h\sqsubseteq X_i$ для некоторого $X_i\in S$, то, по определению, $h'=\{X\mid X\in S,\ X\sqsupseteq h\}$. Положим также $(\{X_1, \ldots, X_k\})''=\{X\in S\mid X\sqsupseteq X_1\sqcap\ldots\sqcap X_k\}$. Множество $(\{X_1, \ldots, X_k\})''$ будем называть замыканием множества $\{X_1, \ldots, X_k\}$ (нетрудно проверить, что операция $''$ действительно обладает свойством оператора замыкания: экстенсивностью, идемпотентностью и изотонностью).

Если множество Y замкнуто, т. е. $(Y)''=Y$, то множество объектов $(Y\cup\{X_i\})''=Y\cup\{X_i\}\cup\{X_{i_1},\ldots, X_{i_m}\}$ будем также обозначать как $[(Y, X)X_{i_1}, \ldots, X_{i_m}]$.

Определение 1.2. Пару замкнутых множеств $\langle h, \{X_1, \ldots, X_n\}\rangle$ будем называть исчерпывающим пересечением (ИП) относительно множества S, если $\{X_1, \ldots, X_n\}\subseteq S$, $(\{X_1,\ldots, X_n\})'=h$ и $h'=\{X_1, \ldots, X_n\}$. h будем называть пересечением, а $\{X_1, \ldots, X_n\}$ — множеством образующих или Par-множеством и обозначать $\mathrm{Par}(h)$.

В [3] ИП называется понятием (concept), h — его объемом (intent) $\mathrm{Par}(h)$ — содержанием (extent). Пары вида $\langle X, \{X\}\rangle$, $X\in S$ будем называть вырожденными ИП, а X — вырожденным пересечением.

Считаем, что порядок в полурешетке ИП (и, соответственно, направление «сверху вниз» или «снизу вверх», существенное при построении алгоритмов) определяется порядком на пересечениях, так что $\langle h_1, \mathrm{Par}(h_1)\rangle\sqsupseteq\langle h_2, \mathrm{Par}(h_2)\rangle$ тогда и только тогда, когда $h_1\sqsubseteq h_2$ и, соответственно, $\mathrm{Par}(h_1)\supseteq\mathrm{Par}(h_2)$.

Множество всех ИП, т. е. множество пар, состоящих из элементов полурешетки и их образующих, порожденных из множества S, будем обозначать МИП(S).

Таким образом, задача, алгоритму решения которой посвящена данная работа, заключается в порождении МИП(S) по заданному S.

Размер входа алгоритма порождения всех ИП характеризуется числом исходных объектов, т. е. величиной $|S|=m$ и размером максимального объекта N. В частном случае булевой решетки $N=|U|=n$, где U — носитель булеана. Так как размер множества МИП(S) может быть экспоненциален по отношению к S [7], а задача определения этого размера $\#P$-полна [8], в оценку эффективности следует включать размер

МИП(S), т. е. $|$МИП$(S)|=K$. Эффективным при этом можно считать алгоритм, временная сложность которого линейна от K и полиномиальна от размеров входа (т. е. полиномиально-перечисляющий алгоритм по терминологии [9]).

2. АЛГОРИТМ «ЗАМЫКАЙ ПО ОДНОМУ»

Некоторая версия алгоритма «Замыкай по одному» (ЗО), не использующая преимуществ лексикографического упорядочения, была приведена в [10] (алгоритм «ВП-bt дерево») для стратегии «сверху вниз».

Считаем, что объекты из S пронумерованы, а каждое множество объектов X на S представляется упорядоченным множеством, в котором элементы расположены в соответствии с нумерацией на S. Порядок на объектах из S индуцирует также лексикографическое упорядочение множеств из 2^S. Порождение всех пересечений как происходящий сверху вниз процесс построения некоторого дерева, вершинам которого соответствуют ИП. В процессе построения и обхода дерева объекты из S могут стать помеченными или остаться непомеченными независимо в каждой вершине. В соответствии с описанием, приведенным ниже дерево порождается «сперва в глубину», хотя возможны и другие стратегии.

Шаг 0. Корневая вершина дерева связана с m вершинами, каждая из которых взаимно-однозначно соответствует какому-либо объекту из S. Все объекты во всех вершинах непомечены, $Y:=\varnothing$.

Шаг 1. Берем первый из непомеченных элементов S. Пусть — это X_i. Вычисляем $(Y\bigcup\{X_i\})'$ и $(Y\bigcup\{X_i\})''=[(Y,X_i)X_{i_1},\ldots,X_{i_m}]$ для некоторых $X_{i_1},\ldots,X_{i_m}\in S$. Образуем новую вершину, соответствующую $(Y\bigcup\{X_i\})''$. Свяжем ее ребром с вершиной, соответствующей Y.

Шаг 2. Если для какого-либо i_j имеет место $i_j\leqslant i$, то ИП $(Y\bigcup\{X_i\})''$ уже было порождено ранее (см. лемму 2.3). Помечаем все элементы S в вершине $(Y\bigcup\{X_i\})''$. Данная ветвь дерева тем самым не будет продолжаться. Если ИП ранее не порождалось, то пометим дополнительно в вершине Y элемент X_i и в вершине $(Y\bigcup\{X_i\})''$ все элементы множества $(Y\bigcup\{X_i\})''$.

Шаг. 3. Если все элементы S оказались помеченными в $(Y\bigcup\{X_i\})''$, то переходим к шагу 4. Иначе — $Y:=(Y\bigcup\{X_i\})''$, переходим к шагу 1.

Шаг 4. Возвращаемся по дереву вверх вплоть до ближайшей вершины, в которой есть непомеченные элементы S. Если такая вершина есть и она соответствует множеству объектов Z, то $Y:=Z$ и переходим к шагу 1. Если такой вершины нет, то все пересечения построены и алгоритм заканчивает работу.

Приведем пример, в котором алгоритм ЗО порождает некоторые пересечения экспоненциальное число раз.

Рассмотрим стратегию «сверху вниз» для булевой полурешетки на носителе $U=\{a_1,\ldots,a_n\}$ и порождающем множестве $S=\{X_1,\ldots,X_n,X_{n+1}\}$, где $X_i=U\backslash\{a_i\}$, $1\leqslant i\leqslant n$, $X_{n+1}=\{a_n\}$. Напомним, что направление «снизу вверх» или «сверху вниз» определяется порядком на ИП: в первом случае процесс начинается с порождения ИП $\langle h,\mathrm{Par}(h)\rangle$ с самыми большими по вложению пересечениями, т. е. h, и переходит к меньшим пересечениям. Тогда ИП $\langle\{a_n\},\{X_1,\ldots,X_{n-1},X_{n+1}\}\rangle$ можно получить из произвольного большего ИП $\langle h,\mathrm{Par}(h)\rangle$ (т. е. $\mathrm{Par}(h)\subseteq\mathrm{Par}(\{a_n\})$) замыканием $(h\bigcup\{X_{n+1}\})'$. Так как таких ИП в точности 2^n-1, то столько же раз может быть получен ИП $\langle\{a_n\},\mathrm{Par}(\{a_n\})\rangle$.

Несмотря на казалось бы пессимистические выводы которые можно сделать из данного примера, число таких ИП и общее число всех ИП (включая кратные), порождаемых алгоритмом, невелико. Это позволяет утверждать следующая

Лемма 2.1. Количество исчерпывающих пересечений, производимых алгоритмом ЗО в стратегии «сверху вниз», есть $O(mK)$.

Доказательство. Алгоритм ЗО из всех ветвей дерева, приводящих к данному ИП, продолжает лишь одну ветвь (см. шаг 2). Следовательно, для каждого ИП $\langle h,\mathrm{Par}(h)\rangle$ число ИП, получаемых алгоритмом ЗО непосредственно после $\langle h,\mathrm{Par}(h)\rangle$, может быть не больше числа объектов, входящих в $S\backslash\mathrm{Par}(h)$, т. е. не более $m-k<m$. Так как число ИП есть K, то общее количество пересечений, произведенных алгоритмом 30, не превышает $mK\square$.

Лемма 2.2. Пусть $S=\{X_1,\ldots,X_m\}\subseteq 2^U$, $U=\{a_1,\ldots,a_n\}$. Тогда количество применений операции пересечения, производимых алгоритмом ЗО в стратегии «снизу вверх», не превышает nK.

Доказательство. Множество элементов полурешетки МИП (S) (будем ее обозначать $\langle S,\bigcap,0\rangle$), образованной исчерпывающими пересечениями множеств из семейства S, можно получить по стратегии «снизу вверх» как множество элементов полурешетки $\langle S',\bigcup,0\rangle$, т. е. МИП$(S)$ с множеством образующих S', двойственным к $S:S'=\{Y_i|Y_i=\{X_j|a_i\in X_j,\ 1\leqslant i\leqslant n,\ 1\leqslant j\leqslant m\}\}$ — и полурешеточной операцией $\bigcup$ — булевой операцией объединения. Так как $|S'|=n$, то, по лемме 2.1, число применений операции пересечения, производимых алгоритмом ЗО, не превосходит $nK\square$.

Из лемм 2.1 и 2.2 вытекает, что объем памяти, необходимый алгоритму ЗО, линеен от элементов полурешетки, т. е. от K. Оценка быстродействия алгоритма складывается из времени порождения ИП и времени проверки того, что данное ИП не было порождено раньше. Для такой проверки «в лоб» путем просмотра всего списка ИП, порожденных ранее, требуется $O(K^2)$ операций (в данном случае, проверок на совпадение). Можно показать, однако, что проверку уникальности порожденных ИП можно провести за время, линейное от K.

Пусть произвольное ИП $H=\langle h,\mathrm{Par}(h)\rangle$ было получено алгоритмом ЗО по стратегии «сверху вниз», т. е. замыканием по одному элементу из S (для булевого случая и стратегии «снизу вверх», т. е. замыкания по одному элементу U, дальнейшие рассуждения проводятся аналогичным образом).

Выводом ИП $\langle h,\mathrm{Par}(h)\rangle$ назовем путь в дереве ЗО, ведущий в какую-либо вершину, соответствующую $\langle h,\mathrm{Par}(h)\rangle$. При этом путь соответствует последовательности элементов по которым происходит замыкание в вершинах пути.

Порядок на элементах S индуцирует лексикографическое упорядочение выводов: вывод $W_1=\langle w_1,\ldots,w_k\rangle$ для ИП $\langle h,\mathrm{Par}(h)\rangle$ лексикографически раньше вывода $W_2=\langle v_1,\ldots,v_s\rangle$, если для первого i, такого, что $w_i\neq\neq w_i$ имеет место $w_i\prec v^i$ (в смысле упорядочения на номерах элементов S).

Каноническим выводом ИП $\langle h,\mathrm{Par}(h)\rangle$ назовем лексикографически первый вывод.

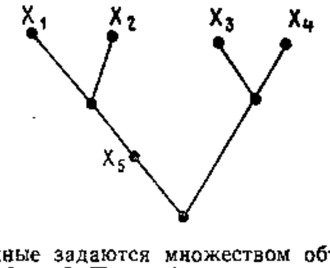
Рассмотрим путь $X_1,\ldots,X_k$ в дереве ЗО, ведущий к ИП $\langle h,\mathrm{Par}(h)\rangle$. Обозначим пересечения, получаемые в результате этих замыканий, как $h_1,\ldots,h_k$, соответственно. Пусть на l-м шаге пути порождается ИП $\langle h_l,\mathrm{Par}(h_l)\rangle$, где

$$h_l=(\mathrm{Par}(h_{l-1})\bigcup\{X_l\})'\ \text{и}$$

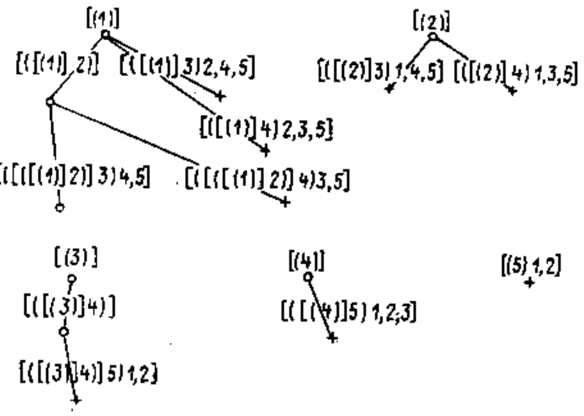
$$\mathrm{Par}(h_l)=\mathrm{Par}(h_{l-1})\bigcup\{X_{i_1},\ldots,X_{i_s}\}\bigcup\{X_l\}\ \text{для некото-}$$

рых $X_{i_1}, \ldots, X_{i_s} \in S$. Будем называть элемент X_i ведущим, а элементы $X_{i_1}, \ldots, X_{i_s}$ — ведомыми для i-го шага вывода $X_1, \ldots, X_k$. Введем следующее обозначение: если множество Y замыкается относительно объекта X_i, а ведомыми при этом оказываются объекты $X_{i_1}, \ldots, X_{i_s}$, то будем записывать результат замыкания, т. е. $(Y \bigcup \{X_i\})''$ как $[(Y, X_i)\, X_{i_1}, \ldots, X_{i_s}]$.

Приведем пример действий алгоритма ЗО в случае полурешетки с порождающим множеством $\{X_1, X_2, X_3, X_4, X_5\}$ и диаграммой



Входные данные задаются множеством объектов с номерами 1, 2, 3, 4, 5. По соображению кратности объекты в дереве представлены соответствующими номерами, а корневая вершина дерева опущена:



Вершины дерева, соответствующие каноническим путям, отмечены кружком, остальные (т. е. неканонические) — крестом. Заметим, что канонический вывод объектов из порождающего множества, которые как X_5, содержатся в других объектах из порождающего множества, может начинаться с вершин, соответствующих другим объектам.

Лемма 2.3. Вывод $\langle X_1, \ldots, X_k \rangle$ в ЗО дереве является каноническим тогда и только тогда, когда для произвольного i все ведомые объекты $X_{i_1}, \ldots, X_{i_s}$ имеют номера большие, чем номера ведущих объектов $X_1, \ldots, X_{i-1}$, соответствующих первым $i-1$ шагам вывода.

Доказательство.

1) Необходимость. Пусть условие леммы не выполняется и для некоторых i, j, t $(1 \leqslant t \leqslant i,\ 1 < j < s)$ номер X_{i_j} меньше номера X_t в нумерации объектов из S. Если $X_t \in (X_1 \bigcup \ldots \bigcup X_{t-1} \bigcup X_t)''$, то путь $\langle X_1, \ldots, X_{t-1}, X_{i_j}, X_{t+1}, \ldots, X_t \rangle$ приводит к ИП $\langle h_i, \mathrm{Par}(h_i) \rangle$, так как ведомые для X_t объекты будут ведомыми и для X_{i_j}. Если $X_t \notin (X_1 \bigcup \ldots \bigcup X_{t-1} \bigcup \bigcup X_t)''$, то к ИП $\langle h_i, \mathrm{Par}(h_i) \rangle$ приводит путь $\langle X_1, \ldots, X_{t-1}, X_{i_j}, X_t, X_{t+1}, \ldots, X_i \rangle$. В любом случае, каждый из этих путей лексикографически раньше, чем $\langle X_1, \ldots, X_t, \ldots, X_i \rangle$. Следовательно, вывод $\langle X_1, \ldots, X_t, \ldots, X_i \rangle$ не канонический.

2) Достаточность следует непосредственно из определения канонического вывода. В самом деле, если условие из формулировки леммы выполняется, то замыкание на каждом шагу происходило по элементу с самым меньшим номером □.

Следствие. Вывод $\langle X_1, \ldots, X_{k-1}, X_k \rangle$ в ЗО дереве является каноническим тогда и только тогда, когда вывод $\langle X_1, \ldots, X_{k-1} \rangle$ — канонический и номер каждого ведомого объекта для k-го шага больше, чем номер объекта X_{k-1}.

Проверяя на каждом шагу построения дерева ЗО условие леммы 2.3, мы можем обойтись без просмотра списка порожденных ИП, чтобы выяснить, было ли построено данное ИП ранее или нет: достаточно контролировать каноничность вывода и не продолжать путей от вершин, к которым приводит неканонический вывод. Каноничность вывода в каждой вершине проверяется, в силу следствия леммы 2.3, сравнением номера каждого $X_{i_1}, \ldots, X_{i_j}$ с числом $i-1$, т. е. за $O(m)$ операций с 0-1 векторами.

При подсчете временной сложности необходимо учесть сложность порождения каждой вершины. Для замыкания требуется выполнение одной операции $\sqcap$ и не более m проверок на вложение. Для проверки каноничности вывода, приводящего в данную вершину, в соответствии с леммой 2.3, требуется выполнение не более m^2 операций с 0-1 строками. Общая временная сложность алгоритма ЗО есть $O(mK)\$ + O(m^2K)\% + O(m^2K)@$, где $\$$ — время выполнения операции $\sqsubseteq$, $\%$ — время проверки на вложение (точнее, проверки выполнимости отношения частичного порядка $\sqsubseteq$, соответствующего полурешеточной операции $\sqcap$ на паре объектов из полурешетки), @ — время выполнения операций с 0-1 строками. В частности, в булевом случае общая временная сложность есть $O(m^2K)@$. В стратегии «снизу вверх» необходимо проведение m проверок на вложение, а также m пересечений при замыкнии и n операций при проверке каноничности вывода, что составляет в сумме $O((m+n)nK)\ @$ времени.

Заметим, что, в случае большой сложности операции пересечения, реальные программы, основанные на алгоритме ЗО могут при продолжении ветви дерева пересекать текущее ИП не с примерами, а с некоторыми ИП, образованными из них и имеющими в дереве ЗО общего родителя с данным ИП (в силу ассоциативности операции сходства): пересечение с подобъектами, т. е. с меньшими объектами, потребует меньших затрат, чем пересечение с исходными объектами, как это имеет место, например, с множествами графов.

Лемма 2.4. Для произвольного входа $\langle S, \sqcap\, 0 \rangle$ дерево ЗО полно, т. е. каждое ИП порождается этим алгоритмом, по крайней мере, один раз.

Доказательство этого факта сводится к построению канонического вывода для произвольного ИП $\langle h, \mathrm{Par}(h) \rangle$. Для стратегии «сверху вниз» при этом из множества $\mathrm{Par}(h)$ выбирается лексикографически первый элемент X, а затем производится последовательность действий, приведенная в описании алгоритма ЗО, но с заменой множества S на множество $\mathrm{Par}(h)$. Таким образом, на каждом шаге производится замыкание объединения множества примеров, полученного на предыдущем шаге с примером из $\mathrm{Par}(h)$, имеющим наименьший номер среди оставшихся. Для каждого ИП из возникающей при этом последовательности множество примеров, содержась в $\mathrm{Par}(h)$, содержит в качестве собственного подмножества множество примеров для предыдущего ИП, а в силу того, что $(\mathrm{Par}(h))'' = \mathrm{Par}(h)$, на некотором шаге в этой последовательности будет порождено $\langle h, \mathrm{Par}(h) \rangle$. Для стратегии «снизу вверх» в случае булевой решетки рассуждения проводятся аналогично.

Все результаты, полученные в данном разделе, можно суммировать в следующей

Теореме 2.5. Множество всех элементов произвольной полурешетки $\langle S, \sqcap, 0\rangle$, где S — множество порождающих полурешетку элементов, можно построить, используя объем памяти $O(mNK)$ и время $O(mK)\$+ +O(m^2K)\%+O(m^2K)@$ где K — число элементов в полурешетке, N — размер максимального элемента полурешетки, $\$$ — время выполнения операции $\sqcap$, $\%$ — время проверки выполнимости отношения частичного порядка, соответствующего полурешеточной операции $\sqcap$ на паре объектов из полурешетки, $@$ — время выполнения операций с 0—1 строками. Для булевой решетки размерности n порождение возможно при использовании $O(mnK)$ памяти и $O((m+n)nK)$ операций с булевыми векторами.

3. ДРУГИЕ АЛГОРИТМЫ

Алгоритм МП получается из алгоритма ЗО изменением описания 1-го шага, на котором алгоритмом МП в каждой вершине дополнительно отбираются те X_i, для которых значения $(Y\bigcup\{X_j\})''$ максимальны по вложению среди всех $(Y\bigcup\{X_j\})''$ с непомеченными ранее X_j. Дерево продолжается лишь по ветвям, соответствующим этим минимальным замыканиям. Алгоритм МП был сформулирован впервые О. М. Аншаковым и К. П. Хазановским [10] в двойственном варианте, т. е. для стратегии «снизу вверх» для булевого случая как алгоритм Минимальных Пересечений. Для этого алгоритма справедлив аналог леммы 2.2 и, следовательно, утверждение о линейности объема памяти, необходимого для работы этого алгоритма. На самом деле, объем памяти, необходимый алгоритму МП, будет меньше, чем этого требует алгоритм ЗО (при сохранении порядка величины $O(mK)$), в силу того, что из каждого ИП осуществляется переход лишь к максимальным по вложению из меньших ИП. При этом, однако, необходимы дополнительные операции для проверки вложимости. Кроме того, для МП канонического вывода в общем случае не существует (так как среди максимальных по вложению ИП могут не быть ИП с лексикографически первыми выводами) и проверять уникальность полученного ИП необходимо путем сравнения его со всеми ИП, порожденными ранее, что приводит к временной сложности алгоритма порядка $O(K^2)$. Заметим, что линейный по памяти и объему алгоритм порождения всех пересечений может быть основан не на проверке каноничности вывода, а на поиске совпадающих пересечений с использованием сбалансированных 2—3 деревьев [11]. Такая техника, однако, не удобна в случае параллельной вычислительной архитектуры, так как предполагает согласование обращений процессоров к централизованному хранилищу пересечений.

* * *

Автор выражает признательность Д. П. Скворцову и О. М. Аншакову за указание неточностей в начальном варианте статьи.

СПИСОК ЛИТЕРАТУРЫ

1. Розенблит А. Б., Голендер В. Е. Логико-комбинаторные методы в конструировании лекарств. — Рига: Зинатне, 1983.
2. Финн В. К. Правдоподобные рассуждения в интеллектуальных системах типа ДСМ // Итоги науки и техники. Сер. Информатика. Т. 15 (Интеллектуальные информационные системы).— М.: ВИНИТИ, 1991.— С. 54—101.
3. Wille R. Restructuring lattice theory: an approach based on hierarchies of concepts // Ordered sets / Ed. I. Rival. Reidel, Dordrecht Boston, 1982.— P. 445—470.
4. Гладун В. П., Ващенко Н. Д. Методы формирования понятий на ЦВМ.— Кибернетика.— 1975.— № 2.— С. 107—112.
5. Duda J. Boolean concept lattices and good concepts // Čas. Pest. Mat.— 1989.— № 114.— P. 165—175.
6. Novotny M., Pawlak Z. Algebraic theory of independence in information systems // Fundamenta Informaticae.— 1991.— № 14.— P. 454—476
7. Забежайло М. И. О некоторых переборных задачах, возникающих при автоматическом порождении гипотез ДСМ-методом // НТИ. Сер. 2.— 1988.— № 1.— С. 28—31.
8. Кузнецов С. О. Интерпретация на графах и сложностные характеристики задач поиска закономерностей определенного вида // НТИ. Сер. 2.— 1989.— № 1.— С. 23—28.
9. Valiant L. G. The Complexity of Enumeration and Reliability Problems // SIAM J. Comput.— 1979. — Vol. 8, № 1.— P. 410—421.
10. Забежайло М. И., Ивашко В. Г., Кузнецов С. О., Михеенкова М. А., Хазановский К. П., Аншаков О. М. Алгоритмические и программные средства ДСМ-метода автоматического порождения гипотез // НТИ. Сер. 2.— 1987.— № 10.— С. 1—14.
11. Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов.— М.: Мир, 1979.

Материал поступил в редакцию 04.12.93.