М.И. Забежайло

О некоторых возможностях управления перебором в ДСМ-методе. Часть II

Аннотация. Обсуждаются возможности оптимизации перебора при интеллектуальном анализе данных средствами ДСМ-метода автоматического формирования гипотез. Рассматриваются некоторые варианты управления перебором за счет использования специально созданных комбинаторных объектов – псевдо-деревьев. Рассмотрены комбинаторные свойства таких объектов. Предложены алгоритмы целенаправленного восстановления псевдо-деревьев по их каркасам. Развивается понятие приближенного ДСМ-метода. Демонстрируются возможности дополнительного ускорения ДСМ-обработки данных за счет использования параллельных алгоритмов, специального типа «облачных» вычислений, а также некоторых проблемно-ориентированных аппаратно-схемных решений.

Ключевые слова: ДСМ-метод автоматического порождения гипотез, вычислительная сложность и оптимизация перебора, методы декомпозиции при сокращении перебора, приближенные вычисления.

Введение

Опыт приложений ДСМ-метода автоматического порождения гипотез [1, 2] и др. показал, что используемые здесь алгоритмы нередко сталкиваются с проблемой сложности вычислений (быстрого роста объемов комбинаторного перебора вариантов при порождении ДСМ-гипотез). В значительном ряде случаев это обстоятельство затрудняет (о порою, вообще, делает невозможным) исчерпывающий анализ вариантов при поиске ДСМ-сходств на заданном для обучения наборе примеров.

Именно поэтому достаточно заманчиво выглядит идея разбиения множества всех ДСМ-гипотез, потенциально порождаемых из каждого конкретного множества примеров, на «самостоятельные» части (подмножества) таким образом, что

- объединение всех подобных «самостоятельных» частей совпадало бы с исходной диаграммой;
- каждая из таких самостоятельных частей порождалась бы «достаточно просто»;

и, наконец.

- число таких «самостоятельных частей» с ростом обучающей выборки и\или исходного алфавита росло бы «не очень быстро».

В первой части этой работы уже было показано, как именно подобная схема может быть реализована. Это делается, прежде всего, за счет выделения в исходной диаграмме ДСМ-сходств отдельных поддиаграмм (подграфов специального вида), каждая из которых в корне имеет один из примеров исходной обучающей выборки. Ветки такой диаграммы ведут к каждой из вершин, образованной минимальными по вложению ДСМ-сходствами – минимальными по вложению неподвижными точками рассматриваемого замыкания Галуа. Представляется уместным именовать эти диаграммы *псевдо-деревьями*, подчеркивая, что поддиаграммы предлагаемого вида, вообще говоря, могут и не быть деревьями (например, за счет существования в них пар вершин во фрагментах типа β - см. ниже, находящихся по крайней мере на двух различных ветках частичного порядка взаимной вложимости соответствующих замыканий Галуа), однако, могут быть сведены к виду дерева удалением тех или иных подмножеств входящих в них ребер. В свою очередь, каждое такое псевдо-дерево может быть представлено как объединение соответствующих деревьев, содержащих то же самое множество вершин и вкладывающихся в рассматриваемое псевдо-дерево.

Также было показано, что каждое из подобных псевдо-деревьев может быть реконструировано (восстановлено по имеющемуся множеству обучающих примеров) достаточно удобно устроенной процедурой. В ней полиномиально быстро (в соотнесении с характерными размерами исходных алфавита и обучающей выборки примеров) может быть построен так называемый каркас – т.е. некоторое специальное приближенное описание этого псевдо-дерева (его подграф, содержащий замыкания всех однобуквенных подмножеств находящегося в корне этого псевдо-дерева объекта). На последующих шагах реконструкция рассматриваемого псевдо-дерева может быть продолжена рекурсивным применением процедуры порождения каркаса на специальным образом выделяемых последовательно вложимых друг в друга подграфах анализируемого псевдо-дерева. При этом порождение множества вершин каждого такого псевдо-дерева может быть выполнено процедурой, вычислительная сложность которой ограничена сверху полиномиальными (от параметров исходных данных – размеров алфавита и обучающей выборки) оценками, а число подобных псевдо-деревьев совпадает с числом примеров в исходной обучающей выборке. Были выделены три типа архитектурных фрагментов, формирующих рассматриваемые псевдо-деревья (архитектура α-типа α, β и γ).

Далее было показано, как организовать анализ всего множества ДСМ-гипотез последовательным перебором вдоль веток частичного порядка в каждом из порожденных представленным образом псевдо-деревьев. При этом оказалось, что порождаемые таким путем псевдо-деревья частичного порядка ДСМ-гипотез удовлетворяют уже сформулированным ранее «пожеланиям» в части декомпозиции исходной диаграммы ДСМ-сходств на «самостоятельные» фрагменты с «удобными» характеристиками.

Показано, что предлагаемая техника организации перебора ДСМ-сходств предоставляет возможности для *выборочного* порождения ДСМ-гипотез (в частности, лишь тех из них, которые имеют отношение к ДСМ-доопределению заданных новых объектов и при этом удовлетворяют тем или иным структурным ограничениям).

Именно таким путем и реализован далее механизм *управляемых приближенных вычислений* в рамках (традиционного) ДСМ-метода: технология *направленного* анализа ДСМ-гипотез в процессе *управляемой навигации* вдоль веток частичного порядка в диаграмме взаимной вложимости ДСМ-сходств. В предельном случае (при анализе всех веток всех псевдо-деревьев частичного порядка, который может быть организован как в последовательном, так и в *параллельном* режиме) эта технология порождает те же результаты, что и обычно используемые в программной реализации ДСМ-метода алгоритмы.

В первой части этой работы наше внимание было сфокусировано на способах формирования, а также на структурных и комбинаторных свойствах псевдо-деревьев и их каркасов. Во второй части мы обратимся к детальному рассмотрению алгоритмики предлагаемого направленного перебора ДСМ-сходств, включая некоторые оценки сложности вычислений. Далее будут также обсуждены некоторые полезные особенности развиваемых алгоритмических конструкций, скоординированное использование которых и на описаниях ДСМ-объектов и на описаниях свойств этих объектов позволяют получить дополнительный выигрыш в части сокращения перебора. Наконец, в заключительном разделе второй части мы поговорим о возможностях дополнительного ускорения ДСМ-обработки данных за счет использования параллельных алгоритмов, специального типа «облачных» вычислений, а также некоторых проблемно-ориентированных аппаратно-схемных решений.

5. Процедуры направленного перебора в диаграммах замыканий Галуа[1]

В первой части этой работы нами уже была продемонстрирована возможность формирования *рекурсивной* процедуры восстановления диаграммы сходств $\mathbf{D\_GC}_{\varphi,f}$ ($\boldsymbol{U}$,$\boldsymbol{\Omega}$), порождаемой из заданного множества примеров $\boldsymbol{\Omega}$ над алфавитом $\boldsymbol{U}$. Процедура позволяет применять ее сперва к исходным данным, затем (по ходу их обработки) к определенным подмножествам исходных данных, и далее (продолжая такие

[1] Нумерация разделов продолжает уже начатую в первой части настоящей работы нумерацию. Забежайло М.И. О некоторых возможностях управления перебором в ДСМ-методе. Часть I // Искусственный интеллект и принятие решений. – 2014, № 2, С. 3-18.

рекурсивные «вложения») - до анализа соответствующих простейших конфигураций (объединение которых и дает исходные анализируемые данные). Именно таким путем формируется естественный вариант декомпозиции исходной решаемой задачи в набор подзадач меньшей сложности. Используемая технология «фокусировки» на менее сложных подзадачах позволяет также переходить от исходного множества примеров **Ω** к его специальным – характеризующим соответствующие гиперкубы – подмножествам. Наконец, дополнительно в рассматриваемой процедуре может быть реализована *целенаправленная навигация* по формирующим анализируемую диаграмму псевдо-деревьям, например, в первую очередь по тем, которые «интересны» с точки зрения тех или иных дополнительных условий (в частности - релевантны некоторым заданным подмножествам исходных примеров, содержат или же, наоборот, не содержат некоторые заданные подмножества образующих из ***U*** и т.п.).

Рассмотрим предоставляемые предлагаемой процедурой возможности целенаправленной навигации в восстанавливаемой диаграмме сходств $\mathbf{D\_GC}_{\varphi,f}(\boldsymbol{U},\boldsymbol{\Omega})$ более подробно. Прежде всего, отметим, что здесь могут комбинироваться два типа «продвижений» по восстанавливаемой диаграмме сходств:

- движения «вширь» (в частности, переход с одного псевдо-дерева в исходной анализируемой диаграмме сходств на другое, переход от одного подмножества исходного множества примеров **Ω,** раскрывающего определенные вершины гиперкуба, содержащегося в заданном псевдо-дереве с одним из примеров исходной выборки **Ω** в корне, к другому такому подмножеству и т.п.); и
- движения «вглубь» (в частности, переход от одного гиперкуба, вложенного в заданное псевдо-дерево с одним из исходных примеров из **Ω** в корне, к другому гиперкубу той же «глубины» вложения (в анализируемое псевдо-дерево) или же переход от гиперкуба одной «глубины» вложения в анализируемое псевдо-дерево к гиперкубу другой «глубины» вложения и т.п. При этом сперва производится переход к соответствующему гиперкубу той же «глубины» вложения, далее в нем - к одному из порождаемых при его раскрытии псевдо-деревьев – поддиаграмм псевдо-дерева, содержащих гиперкуб – исходную точку рассматриваемого перехода, а затем уже в этой поддиаграмме – к одному из вложенных уже в нее гиперкубов).

Исчерпывающую процедуру последовательного обхода (при восстановлении) исходной диаграммы по схеме

сперва - вглубь на сколько это возможно,

затем – назад до последней точки рекурсивного вложения,

где, далее, – вширь на один шаг и, затем,

вновь - вглубь, на сколько это окажется возможным,

и так далее до возвращения к последнему примеру исходной выборки ***Ω*** *в качестве корня только что восстановленного псевдо-дерева,*

предлагает

Алгоритм НПС
(Направленного Перебора Сходств)

ВХОД (Входные данные): алфавит ***U*** и множество примеров для обучения **Ω**.

ВЫХОД (Выходной результат): диаграмма сходств $\mathbf{D\_GC}_{\varphi,f}(\boldsymbol{U},\boldsymbol{\Omega})$.

Описание алгоритма:

ШАГ 1.

Нумеруем образующие алфавита $\boldsymbol{U}=\{\mathbf{a_1}, \dots, \mathbf{a_n}\}$ и примеры из множества $\boldsymbol{\Omega}=\{\mathbf{A_1}, \dots, \mathbf{A_m}\}$.

Счетчику *ГЛУБИНА ВЛОЖЕНИЯ*, контролирующему глубину вложения текущей анализируемой поддиаграммы (диаграммы сходств $\mathbf{D\_GC}_{\varphi,f}(\boldsymbol{U},\boldsymbol{\Omega})$), присваиваем значение 0.

Параметру *МНОЖЕСТВО ПРИМЕРОВ* присваиваем значение <1,1, … , 1> (кортеж, сформированный из в точности m единиц), кодирующее включение в текущее анализируемое множество примеров всех m элементов исходного множества **Ω** (фактически положив текущее анализируемое множество примеров $\boldsymbol{\Omega}^* = \boldsymbol{\Omega}$).

Счетчику *ТЕКУЩИЙ ПРИМЕР*, контролирующему перебор в исходной диаграмме сходств $\mathbf{D\_GC}_{\varphi,f}(\boldsymbol{U},\boldsymbol{\Omega})$ корней текущих анализируемых псевдо-деревьев, присваиваем значение 0.

Счетчику *ТЕКУЩИЙ ГИПЕРКУБ*, контролирующему перебор гиперкубов в текущем анализируемом псевдо-дереве, присваиваем значение 0.

Параметру *РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ПРИМЕРОВ*, фиксирующему

число примеров в текущем *МНОЖЕСТВЕ ПРИМЕРОВ* присваиваем значение m.

Параметру *РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ГИПЕРКУБОВ*, фиксирующему число гиперкубов в текущем анализируемом псевдо-дереве, присваиваем значение 0.

В исходно пустой стек **TRACK** помещаем (как единую запись) шестерку <*ГЛУБИНА ВЛОЖЕНИЯ, МНОЖЕСТВО ПРИМЕРОВ, ТЕКУЩИЙ ПРИМЕР, ТЕКУЩИЙ ГИПЕРКУБ, РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ПРИМЕРОВ, РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ГИПЕРКУБОВ* >.

ШАГ 2.

Значение ***i*** счетчика *ТЕКУЩИЙ ПРИМЕР* элементов из **Ω*** присваиваем значение ***i***+1. Если значение ***i*** после этого превосходит значение параметра *РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ПРИМЕРОВ* (т.е. все элементы множества **Ω*** - как корни соответствующих псевдо-деревьев на текущем уровне вложения анализируемых поддиаграмм исходной диаграммы сходств $\mathbf{D\_GC}_{\varphi,f}$ (***U*,Ω**) – перебраны), переходим на ШАГ 7. В противном случае начинаем восстановление псевдо-дерева $\mathbf{D\_GC}_{\varphi,f}$ $(\boldsymbol{U},\boldsymbol{\Omega}^*)\big|_{\{Ai\}}$, переходя на

ШАГ 3.

Строим каркас $GK_{\mathbf{D,Ai}}$ псевдо-дерева $\mathbf{D\_GC}_{\varphi,f}$ $(\boldsymbol{U},\boldsymbol{\Omega}^*)\big|_{\{Ai\}}$.

Выделяем в этом псевдо-дереве фрагмент типа α и проверяем наличие архитектурных фрагментов типа β. В случае их отсутствия переходим к ШАГу 2. В противном случае значение ***j*** счетчика *ТЕКУЩИЙ ГИПЕРКУБ* таких фрагментов типа β обращаем в 0, одновременно вычисленное (при построении каркаса $GK_{\mathbf{D,Ai}}$) число фрагментов типа β присваиваем в качестве текущего значения параметру *РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ГИПЕРКУБОВ*, текущее значение d счетчика *ГЛУБИНА ВЛОЖЕНИЯ* заменяем на значение d+1, в стек **TRACK** в качестве верхнего элемента помещаем обновленную шестерку <*ГЛУБИНА ВЛОЖЕНИЯ, МНОЖЕСТВО ПРИМЕРОВ, ТЕКУЩИЙ ПРИМЕР, ТЕКУЩИЙ ГИПЕРКУБ, РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ПРИМЕРОВ, РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ГИПЕРКУБОВ*>, после чего переходим на

ШАГ 4.

Значение ***j*** счетчика *ТЕКУЩИЙ ГИПЕРКУБ* соответствующих фрагментов типа β присваиваем значение ***j***+1. Если вновь полученное значение ***j*** превосходит текущее значение параметра *РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ГИПЕРКУБОВ* (т.е. уже перебраны все такие фрагменты для текущего анализируемого псевдо-дерева, причем *текущего* - т.е. не обязательно *верхнего* - уровня вложения поддиаграмм исходной диаграммы сходств $\mathbf{D\_GC}_{\varphi,f}$ (***U*,Ω**)), удаляем из стека **TRACK** последний элемент и возвращаемся к ШАГу 2. (Тем самым «курсор» актуального состояния анализа исходной диаграммы сходств $\mathbf{D\_GC}_{\varphi,f}$ (***U*,Ω**) переводится к предпоследнему - с точки зрения движения *вширь* и *вглубь* по этой диаграмме – положению, которое фиксируется соответствующими значениями трех используемых счетчиков - глубины вложения, примеров из исходной выборки **Ω** а также гиперкубов в текущей анализируемой поддиаграмме). В противном случае обращаемся к анализу той части псевдо-дерева $\mathbf{D\_GC}_{\varphi,f}(\boldsymbol{U},\boldsymbol{\Omega}^*)\big|_{\{Ai\}}$, которая лежит на гиперкубе с вновь выбранным текущим номером ***j*** и переходим на

ШАГ 5.

Выделяем то подмножество **Ω**** исходного множества примеров **Ω**, все элементы которого раскрываются в текущий анализируемый гиперкуб.

Проверяем выполнимость условия **Утверждения 4.3**, гарантирующего полное покрытие соответствующим фрагментом текущего анализируемого псевдо-дерева всего текущего анализируемого гиперкуба. В случае его выполнимости, восстанавливаем соответствующий анализируемому фрагменту текущего анализируемого псевдо-дерева гиперкуб, затем переходим на ШАГ 4. В противном случае

ШАГ 6.

Переходим к анализу подзадачи с текущими входными данными <***U*,Ω****> (рекурсивно вкладывая описанные ШАГами 1-5 действия в описываемый исходный алгоритм: в качестве исходных данных на ВХОДЕ такого рекурсивного вложения текущая пара <***U*,Ω****> заменит предыдущую анализируемую пару <***U*,Ω***>).

Параметру *МНОЖЕСТВО ПРИМЕРОВ* присваиваем значение $<x_1, x_2, \ldots, x_m>$, где для каждо-

го $i \in \{1,2, \ldots, m\}$ значение $x_i = 1$, если A_i содержится в множестве **Ω****. В противном случае полагаем $x_i=0$.

Счетчику *ТЕКУЩИЙ ПРИМЕР*, контролирующему перебор в исходной диаграмме сходств $\mathbf{D\_GC}_{\varphi,f}(\boldsymbol{U},\boldsymbol{\Omega}^*)$ корней текущих анализируемых псевдо-деревьев, присваиваем значение 0.

Счетчику *ТЕКУЩИЙ ГИПЕРКУБ*, контролирующему перебор гиперкубов в текущем анализируемом псевдо-дереве, присваиваем значение 0.

Параметру *РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ПРИМЕРОВ*, фиксирующему число примеров в текущем *МНОЖЕСТВЕ ПРИМЕРОВ* присваиваем значение $|\boldsymbol{\Omega}^{**}|$ - число единиц в текущем виде кортежа $<x_1, x_2, \ldots, x_m>$, отражающее число элементов в текущем множестве примеров **Ω****.

Параметру *РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ГИПЕРКУБОВ*, фиксирующему число гиперкубов в текущем анализируемом псевдо-дереве, присваиваем значение 0.

В стек **TRACK** помещаем (как единую запись) шестерку текущих значений *<ГЛУБИНА ВЛОЖЕНИЯ, МНОЖЕСТВО ПРИМЕРОВ, ТЕКУЩИЙ ПРИМЕР, ТЕКУЩИЙ ГИПЕРКУБ, РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ПРИМЕРОВ, РАЗМЕР ТЕКУЩЕГО МНОЖЕСТВА ГИПЕРКУБОВ>*. Далее переходим на ШАГ 2 (имея ввиду при этом, что роль **Ω*** теперь будет играть **Ω****).

ШАГ *7*.
СТОП

Свойства корректности **Алгоритма НПС** демонстрирует следующее

Утверждение 5.1[2]

Алгоритма НПС характеризуется полнотой и точностью при восстановлении диаграммы сходств $\mathbf{D\_GC}_{\varphi,f}(\boldsymbol{U},\boldsymbol{\Omega})$:

1) порождает все принадлежащие ей сходства (с указанием для каждого из них всех ближайших к нему по вложимости элементов восстанавливаемой диаграммы), при этом

2) не порождает никаких других (не являющихся элементами восстанавливаемого множества сходств) объектов.

Доказательство

ПОЛНОТА. Полнота **Алгоритма НПС** – естественное следствие полноты организованного в нем последовательного перебора всех возможных (в рамках восстанавливаемой диаграммы сходств) псевдо-деревьев, их каркасов а также (вложенных в эти псевдо-деревья и диагностируемых при формировании соответствующих каркасов) гиперкубов.

ТОЧНОСТЬ. Точность **Алгоритма НПС** – очевидное следствие использования им для восстановления каждой рассматриваемой диаграммы сходств процедуры порождения каркасов (и только ее - !) соответствующих поддиаграмм: каждой вершине каждого порождаемого каркаса соответствует то или иное замкнутое подмножество образующих исходного алфавита $\boldsymbol{U}$.

Дополнительную полезную информацию о необходимых **Алгоритму НПС** ресурсах памяти дает

Утверждение 5.2

Глубина заполнения стека **TRACK** при работе **Алгоритма НПС** ограничена сверху величиной $|\boldsymbol{U}|$ (числом образующих исходного алфавита $\boldsymbol{U}$).

Доказательство

Глубина стека **TRACK** на каждом шаге **Алгоритма НПС** ограничена сверху длиной соответствующей ветки частичного порядка в каждой из анализируемых по ходу его работы псевдо-деревьев. Таким образом, доказываемое **Утверждение** есть очевидное следствие двух фактов:

1) длина каждой ветки частичного порядка (задаваемого исходной диаграммой вложимости сходств $\mathbf{D\_GC}_{\varphi,f}$ $(\boldsymbol{U},\boldsymbol{\Omega})$) не может превышать числа образующих, которыми сформирован лежащий в корне соответствующего псевдо-дерева пример из исходного множества **Ω**.

2) Любой из элементов исходной выборки **Ω** может быть сформирован лишь образующими из исходного алфавита $\boldsymbol{U}$.

Следует обратить внимание на две особенности обсуждаемого **Алгоритма**. Во-первых, при восстановлении целевой диаграммы взаим-

[2] Ради упрощения «сквозной навигации» в сводном варианте текста нумерация примеров и утверждений продолжает ту, которая уже использована в первой части этой работы.

ной вложимости сходств основным рабочим инструментом **Алгоритма** оказывается (полиномиально сложная - !) процедура порождения каркасов соответствующих псевдо-деревьев. Именно это обстоятельство оставляет возможности при организации целенаправленной навигации по восстанавливаемой диаграмме сходств, например: при проверке условий каузальной полноты или же при поиске ДСМ-гипотез, способных обеспечить доопределение новых τ-примеров и т.п., быстро проверять (восстанавливать) подходящие фрагменты целевой диаграммы всех сходств, получая (быстрыми алгоритмами) таким способом ответы на соответствующие вопросы (выполнения условий каузальной полноты, доопределимости рассматриваемых τ–примеров и т.п.).

Во вторых, если рассматривать транзитивные замыкания анализируемых нами здесь графов – диаграмм сходств, псевдо-деревьев и их каркасов, то лежащая в основе **Алгоритма НПС** техника порождения каркасов и достраивания гиперкубов позволяет поступить следующим образом: для каждого содержащего в своем корне один из примеров – элементов исходной выборки **Ω** – псевдо-дерева стартуем с порождения его каркаса. Затем, как мы только что условились, достраиваем его до соответствующего транзитивного замыкания. Далее, порождая утоняющие «картину» фрагменты – гиперкубы и их поддиаграммы, «запускаем» процесс монотонно возрастающего пополнения «стартового» графа каркаса, итерируя этот процесс до восстановления соответствующего псевдо-дерева. (Разумеется, подобное пополнение может также выполняется на уровне транзитивного замыкания каждого порождаемого подграфа, и, как нетрудно убедиться, исходные графы и их транзитивные замыкания будут различаться лишь числом ребер, имея совпадающие множества вершин).

Таким образом, используя эти два свойства можно, соответствующим образом организовав последовательность перебора восстанавливаемых в каждом псевдо-дереве подграфов, наладить порождение в первую очередь полезных результатов (например, ДСМ-гипотез, обеспечивающих выполнение условий каузальной полноты или же доопределимость заданных τ-примеров), а уже далее, если это потребуется, восстановить целиком все множество ДСМ-гипотез, формируемых из заданных алфавита ***U*** и набора примеров **Ω**. Другими словами, это позволяет организовать процесс формирования ***управляемых приближений*** при порождении множества всех задаваемых каждой парой <***U***,**Ω**> ДСМ-гипотез. Чуть позднее мы подробнее рассмотрим ряд интересных особенностей порождения полезных гипотез (разумеется, уточнив в явном виде формальное значение этого термина) средствами обсуждаемой здесь техники.

6. Некоторые архитектурные особенности анализируемых псевдо-деревьев

Оценивая возможные объемы комбинаторного перебора, возникающего при восстановлении соответствующих диаграмм вложимости ДСМ-сходств, порождаемых заданными парами вида <***U***,**Ω**>, представляется естественным попытаться выделить наиболее сложные комбинации (в том числе – и случаи рекурсивной вложимости) расположения гиперкубов на каркасах соответствующих псевдо-деревьев.

Здесь, рассматривая различные комбинации архитектурных фрагментов типа α и типа β в структуре соответствующих псевдо-деревьев, прежде всего, заметим, что в части возможно большего числа вершин (и ребер) в таких архитектурах конструкции, где на одном уровне «глубины» находятся два и более гиперкубов сопоставимого размера, уступают конструкциям, содержащим лишь один гиперкуб (на соответствующем уровне глубины). Действительно, для каждого уровня «глубины» необходим соответствующий «запас» образующих из ***U***, позволяющий (за счет комбинирования подмножеств этого «запаса») обеспечить «строительным материалом» достаточное количество вершин рассматриваемого псевдо-дерева, лежащих на этих – разных – гиперкубах. Таким образом, наихудший (с точки зрения объемов вычислений) вариант архитектуры – чередование (при движении по вертикали от корня рассматриваемого псевдо-дерева к его листьям) линейных участков (фрагментов типа α) и единичных фрагментов типа β (гиперкубов). Попробуем аппроксимировать конфигурации такого вида, своего рода «предельной» конструкцией – иерархией лишь из заданного числа гиперкубов (пренебрегая в рассматриваемом «предельном» случае необходимостью

зарезервировать хотя бы часть образующих из алфавита $\boldsymbol{U}$ на формирование архитектурных фрагментов типа α в анализируемом псевдо-дереве, и осознавая также, что возможное число вершин восстанавливаемого псевдо-дерева можно максимально увеличить за счет комбинирования всех различных подмножеств образующих, помещаемых на нижней границе соответствующего гиперкуба).

В рамках предлагаемой аппроксимации попробуем оценить, для какого количества гиперкубов в такой конфигурации число вершин в псевдо-дереве анализируемой архитектуры окажется максимально большим. Положим, что в алфавите $\boldsymbol{U}$ у нас (как и ранее) ровно n элементов. Рассмотрим выражение вида

$$N(n) = O(k2^{n/k}),$$

где k – параметр, характеризующий число гиперкубов в анализируемом («модельном») псевдо-дереве выбранной нами архитектуры, $O(2^{n/k})$ – дает оценку числа вершин гиперкуба, построенного на n/k образующих, а N(n) таким образом позволяет оценить общее число вершин этого псевдо-дерева. Переходя к функции вида

$$f(x) = O(x\ 2^{n/x}),$$

поведение которой (в частности – участки роста и убывания, а также экстремумы) нас будет интересовать для целых и положительных значений переменной x, вычислим производную f '(x) для f(x) и приравняем её нулю:

$$f'(x) = 2^{n/x} - n\ ln2\ (2^{n/x})\ /x = 2^{n/x}\ (1- (n\ ln2)/x) = 0$$

Разрешая полученное уравнение относительно x, получаем искомое экстремальное значение

$$x = n\ ln2\ .$$

При этом вычисление второй производной для f(x) показывает, что мы имеем дело с минимумом этой функции в найденной точке экстремума.

Принимая во внимание, что

$$ln2 \approx 0{,}69 > 0{,}5\ ,$$

заключаем, что на интервале значений $x \in [1, n/2]$ рассматриваемая нами функция f(x) монотонно убывает. Таким образом, при заданном размере алфавита $\boldsymbol{U}$ оцениваемая величина N(n) – число вершин восстанавливаемого псевдо-дерева – достигает максимальных возможных значений для простейшей конфигурации – с одним гиперкубом. А в другом предельном случае (для x = n/2) каждый из попадающих в рассматриваемое псевдо-дерево гиперкубов вырождается до трех (соответствующих непустым подмножествам множества из двух образующих) вершин, что в свою очередь минимизирует количество требующих рассмотрения вершин анализируемого псевдо-дерева, ограничивая его сверху величиной 3n/2.

Наглядное представление о характере поведения функции $f(x) = O(x\ 2^{n/x})$ на сегменте положительных значений переменной x дают диаграммы на Рис. 5 и Рис. 6.

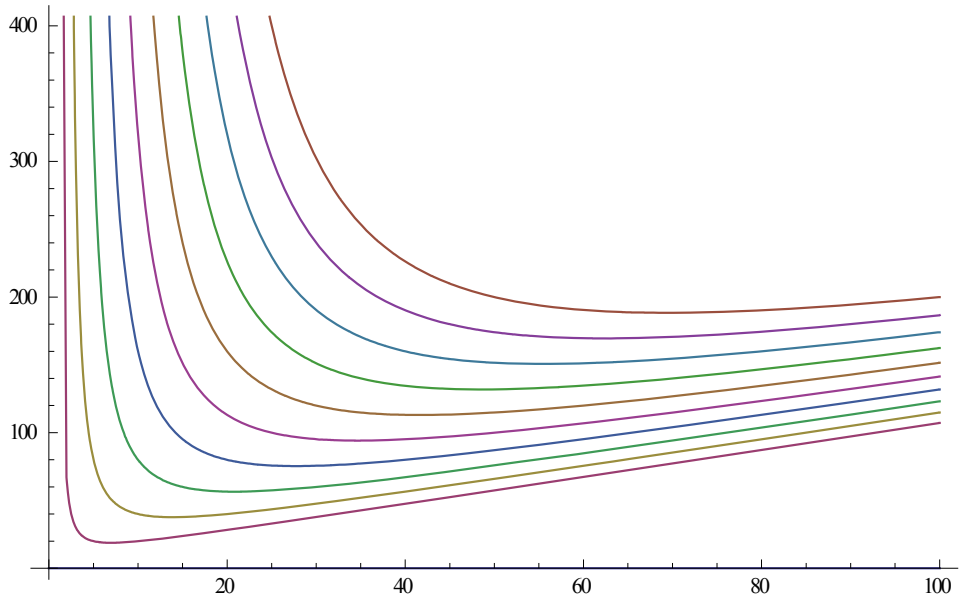

Рис. 5. Графики функции $f(x) = O(x\ 2^{n/x})$ для $n \in \{10, 20, \ldots 100\}$ и $x \in (0,100]$

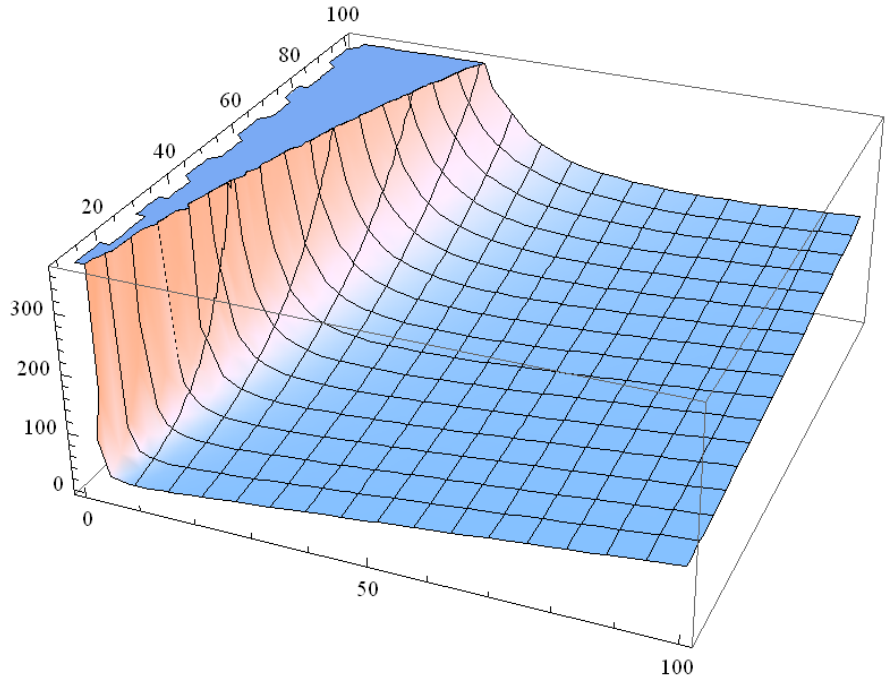

Рис. 6. Графики функции $f(x) = O(x\, 2^{y/x})$ для $x \in (0, 100]$ и $y \in [10,100]$

7. Переход к работе с различными стратегиями ДСМ-рассуждений

Полезные дополнительные (по отношению к традиционной алгоритмике ДСМ-метода например [2] и др.) возможности развиваемая здесь техника целенаправленного перебора вариантов предоставляет также и при порождении ДСМ-гипотез средствами более сильных, чем простое сходство (в частности, [1, 2] и др.) стратегий рассуждений. Целенаправленная навигация вдоль соответствующих псевдо-деревьев помогает ускорить проверку соответствующих дополнительных условий в используемых решающих предикатах и правилах правдоподобного вывода. Наиболее существенные особенности анализа подобного рода дополнительных условий мы рассмотрим на примере проверки выполнимости условий так называемого запрета на контрпримеры [1] и др. Критически важной особенностью этого расширения стандартных предикатов и ППВ простого сходства является необходимость оперировать одновременно примерами обоих знаков из исходного множества ДСМ-примеров **Ω** (т.е. одновременно – и примерами, и контрпримерами).

Логическое условие запрета на контрпримеры (будем именовать его также как условие ***ЗКП***) выражается следующим соотношением[3]:

$$\forall X \forall Y\ (\ ((v \subset X)\ \&\ (w \subseteq Y)) \rightarrow (J_{(1,n)}(X \Rightarrow_1 Y) \vee J_{(\tau,n)}(X \Rightarrow_1 Y))\ ), \qquad (\textbf{\textit{ЗКП}})^{+}$$

и утверждает, что (для позитивного ***ЗКП***) искомая ДСМ-причина **v** ее следствие **w** не содержатся в (–)-примерах (т.е. в контрпримерах из исходного множества **Ω,** возможно, расширенного вновь доопределенными на предыдущих шагах ДСМ-рассуждения примерами) а также в примерах с оценкой ⟨0, n⟩ (фактическое противоречие на текущем n-ном шаге ДСМ-рассуждения).

Обсудим некоторые «технические» особенности рассматриваемой ситуации. Прежде всего, дадим соответствующие определения:

Определение 7.1

1. Задачей *о существовании ДСМ-сходства, удовлетворяющего условию запрета на контрпримеры* (Задачей ***С-ЗКП***), будем называть следующую задачу:

[3] Здесь представлен случай позитивного условия ЗКП. Случай негативного условия формулируется аналогичным образом (например [1] и др.)

Дано: конечный алфавит $\boldsymbol{U} = \{a_1, a_2, \dots, a_n\}$, конечные множества примеров $\boldsymbol{\Omega}^{\alpha} = \{A_1, A_2, \dots, A_n\}$ и контрпримеров $\boldsymbol{\Omega}^{-\alpha} = \{A^-_1, A^-_2, \dots, A^-_m\}$ для ДСМ-обучения.

Найти: существует ли в множестве $GC(\boldsymbol{U}, \boldsymbol{\Omega}^{\alpha})$ ДСМ-сходств, порожденных на примерах из множества $\boldsymbol{\Omega}^{\alpha}$, хотя бы один элемент, удовлетворяющий условию запрета на контрпримеры (т.е. не вкладывающееся ни в один из элементов множества $\boldsymbol{\Omega}^{-\alpha}$ – подробнее [1] и др.).

2. Задачей *о перечислении всех ДСМ-сходств, удовлетворяющих условию запрета на контрпримеры* (Задачей ***Все_С-ЗКП***), будем называть следующую задачу:

Дано: конечный алфавит $\boldsymbol{U} = \{a_1, a_2, \dots, a_n\}$, конечные множества примеров $\boldsymbol{\Omega}^{\alpha} = \{A_1, A_2, \dots, A_n\}$ и контрпримеров $\boldsymbol{\Omega}^{-\alpha} = \{A^-_1, A^-_2, \dots, A^-_m\}$ для ДСМ-обучения.

Найти: перечислить в множестве $GC(\boldsymbol{U}, \boldsymbol{\Omega}^{\alpha})$ ДСМ-сходств, порожденных на примерах из множества $\boldsymbol{\Omega}^{\alpha}$, все элементы, удовлетворяющие условию запрета на контрпримеры (т.е. не вкладывающиеся ни в один из элементов множества $\boldsymbol{\Omega}^{-\alpha}$).

Определение 7.2

При заданных
- конечном алфавите $\boldsymbol{U} = \{a_1, a_2, \dots, a_n\}$ а также
- конечных множествах примеров $\boldsymbol{\Omega}^{\alpha} = \{A_1, A_2, \dots, A_n\}$
и контрпримеров $\boldsymbol{\Omega}^{-\alpha} = \{A^-_1, A^-_2, \dots, A^-_m\}$ для ДСМ-обучения

стоп-листом SL_j для каждого контрпримера A^-_j из множества $\boldsymbol{\Omega}^{-\alpha}$ будем называть соответствующее множество $\{ x^j_1, x^j_2, \dots, x^j_{n(j)} \}$ образующих из $\boldsymbol{U}$, которые не входят в A^-_j.

Далее убедимся, что имеет место следующее полезное

Утверждение 7.1

Элемент $\mathbf{x}$ множества $GC(\boldsymbol{U}, \boldsymbol{\Omega}^{\alpha})$ всех ДСМ-сходств, порожденных из множества примеров $\boldsymbol{\Omega}^{\alpha}$ удовлетворяет условию запрета на контрпримеры (условию ***ЗКП***) тогда и только тогда, когда его пересечение с каждым из стоп-листов $SL_1, SL_2, \dots, SL_m$ не пусто:

$$\forall j \ [(j \in \{1,2, \dots, m\}) \ \& \ (\mathbf{x} \cap SL_j \neq \varnothing)]$$

Доказательство

=>. Если $\mathbf{x} \in GC(\boldsymbol{U}, \boldsymbol{\Omega}^{\alpha})$ и выполнено условие запрета на контрпримеры, то для каждого контрпримера A^-_j выполняется условие $\mathbf{x} \not\subset A^-_j$, т.е.

$$\mathbf{x} \cap A^-_j \neq \mathbf{x},$$

т.е. во множестве образующих $\mathbf{x}$ найдется по крайней мере один элемент из множества $\boldsymbol{U}$, который не входит в A^-_j, что означает выполнимость условия:

$$\mathbf{x} \cap SL_j \neq \varnothing .$$

<=. В случае, когда для каждого стоп-листа SL_j имеем

$$\mathbf{x} \cap SL_j \neq \varnothing,$$

следствием этого соотношения будет соотношение

$$\mathbf{x} \cap A^-_j \neq \mathbf{x},$$

которое и означает, что

$$\mathbf{x} \not\subset A^-_j .$$

Перейдем к обсуждению комбинаторных свойств рассматриваемых нами переборных задач. Имеет место

Утверждение 7.2

Задача *о существовании ДСМ-сходства, удовлетворяющего условию запрета на контрпримеры* (задача ***С-ЗКП***), принадлежит к классу полиномиально разрешимых (например [3] и др.).

Доказательство.

1. Построим по заданным алфавиту $\boldsymbol{U}$ и множеству контрпримеров $\boldsymbol{\Omega}^{-\alpha}$ соответствующее множество стоп-листов $SL_1, SL_2, \dots, SL_m$. (Такое построение – полиномиально сложная процедура). Если среди сформированных стоп-листов имеется $\varnothing$, то множество решений задачи ***С-ЗКП*** пусто.

2. Если ни один из стоп-листов не оказался пустым множеством, выберем A_1 – первый из элементов множества примеров $\boldsymbol{\Omega}^{\alpha}$, чтобы затем (полиномиально быстро - !) построить каркас $GK_{\mathbf{D,A1}}$ соответствующего ему псевдо-дерева $\mathbf{D\_GC}(\boldsymbol{U}, \boldsymbol{\Omega}^{\alpha})\big|_{\{A1\}}$ в диаграмме $\mathbf{D\_GC}(\boldsymbol{U}, \boldsymbol{\Omega}^{\alpha})$ всех возможных ДСМ-сходств, порождаемых на множестве примеров $\boldsymbol{\Omega}^{\alpha}$. Далее последовательно рассмотрим вершины (которых не более чем $n = |\boldsymbol{U}|$) псевдо-дерева $\mathbf{D\_GC}(\boldsymbol{U}, \boldsymbol{\Omega}^{\alpha})\big|_{\{A1\}}$, непосредственно доминируемые его корнем – A_1: для каждой такой вер-

шины непустота ее пересечений (как множества образующих из $\boldsymbol{U}$) со всеми стоп-листами $SL_1, SL_2, \ldots, SL_m$ является достаточным условием существования в $\mathbf{D\_GC}(\boldsymbol{U},\boldsymbol{\Omega}^{\alpha})\big|_{\{A1\}}$ вершины, порождающей решение задачи ***С-ЗКП***. Действительно, выбрав из каждого такого пересечения по одной образующей и вычислив замыкание объединения порожденного таким образом множества образующих, мы получим сходство (как элемент множества всех сходств $\mathbf{D\_GC}(\boldsymbol{U},\boldsymbol{\Omega}^{\alpha})$), удовлетворяющее (по построению – см. общие со стоп-листами образующие) условию запрета на контрпримеры.

При этом, в случае, когда корень A_1 рассматриваемого псевдо-дерева $\mathbf{D\_GC}(\boldsymbol{U}, \boldsymbol{\Omega}^{\alpha})\big|_{\{A1\}}$ есть «наибольшая» вершина некоторого гиперкуба в этом псевдо-дереве (т.е. A_1 лежит на архитектурном фрагменте типа β), рассматриваемое выше множество непосредственно доминируемых им вершин анализируемого псевдо-дерева полиномиально быстро восстанавливается по (найденной при построении каркаса $GK_{\mathbf{D,A1}}$) нижней границе этого гиперкуба и (также полиномиально быстро восстанавливаемому – **Утверждение 4.2** в первой части настоящей работы) подмножеству $\mathbf{H}(\boldsymbol{U},\boldsymbol{\Omega}^{\alpha},A_1)$ всех примеров из $\boldsymbol{\Omega}^{\alpha}$, раскрывающих вершины рассматриваемого гиперкуба. (Действительно, для восстановления искомых непосредственно доминируемых A_1 вершин, лежащих на рассматриваемом гиперкубе, достаточно: сформировать попарные пересечения A_1 и элементов подмножества $\mathbf{H}(\boldsymbol{U},\boldsymbol{\Omega}^{\alpha},A_1)$, затем удалить из полученного перечня каждое такое пересечение, которое вкладывается хотя бы в один из других элементов этого перечня, не совпадая с ним. Предлагаемая процедура редактирования рассматриваемого перечня пересечений, как несложно убедиться, может быть реализована не более чем попарными сравнениями его элементов, т.е. - полиномиально быстро).

В случае принадлежности корня A_1 к архитектурному фрагменту типа α, непосредственно доминируемые этим корнем вершины анализируемого псевдо-дерева лежат на (только что построенном) каркасе $GK_{\mathbf{D,A1}}$.

3. Если для текущего элемента A_i исходного множества примеров представленная выше в п.2 последовательность процедур (из-за пустоты пересечений сопоставленных доминируемым вершинам множеств образующих из $\boldsymbol{U}$ с какими-либо из рассматриваемых стоп-листов), переходим от примера A_i к примеру A_{i+1} , и так до завершения (полиномиально быстро реализуемого) перебора всех элементов множества $\boldsymbol{\Omega}^{\alpha}$.

Утверждение 7.3

Задача *о числе ДСМ-сходств, удовлетворяющих условию запрета на контрпримеры* (***ВСЕ_ С-ЗКП***), принадлежит к классу ***#РС*** перечислительно полных (подробнее [3-6] и др.) комбинаторных проблем.

Доказательство.

Предыдущим **Утверждением 7.2** показана полиномиально быстрая разрешимость задачи ***С-ЗКП***. Продемонстрируем возможности полиномиальной сводимости известной перечислительно полной задачи ***МОНОТОННАЯ ВЫПОЛНИМОСТЬ*** (см., например, [3-6] и др.) к задаче ***ВСЕ_С-ЗКП.*** Для этого мы покажем, как по заданной булевской функции **Ф** в виде 2-КНФ будут порождены множества $\boldsymbol{U}(\mathbf{Ф})$, $\boldsymbol{\Omega}^{\alpha}(\mathbf{Ф})$ и $\boldsymbol{\Omega}^{-\alpha}(\mathbf{Ф})$, а также будет показано, что множество таких сходств из $\mathbf{D\_GC}(\boldsymbol{U}(\mathbf{Ф}),\boldsymbol{\Omega}^{\alpha}(\mathbf{Ф}))$, которые удовлетворяют также условию запрета на контрпримеры, изоморфно множеству единиц функции **Ф**.

Обратимся к более подробному описанию предлагаемого механизма сводимости рассматриваемых комбинаторных задач. Прежде всего, уточним, что при упоминании задачи о выполнимости монотонной булевской функции, представленной в виде 2-КНФ, речь идет о следующей комбинаторной проблеме:

Дано: монотонная булевская функция **Ф** в виде 2-КНФ

$$\mathbf{Ф}(x_{\mathbf{1}},x_{\mathbf{2}},\ldots,x_{\mathbf{n}}) = \mathbf{Ф_1} \,\&\, \mathbf{Ф_2} \,\&\, \ldots \,\&\, \mathbf{Ф_m} = (x_{11} \vee x_{12}) \,\&\, (x_{21} \vee x_{22}) \,\&\, \ldots \,\&\, (x_{m1} \vee x_{m2}) \quad \mathbf{(!)}$$

Найти: число наборов значений переменных $x_{\mathbf{1}},x_{\mathbf{2}},\ldots,x_{\mathbf{n}}$, выполняющих функцию **Ф**.

Пусть функция **Ф** имеет представленный выше вид - (**!**). Определим:

- алфавит $\boldsymbol{U}$ для формирования примеров и контрпримеров в соответствующей задаче ***ВСЕ_С-ЗКП*** (**Ф**) как
$$\boldsymbol{U} = \{x_1, x_2, \ldots, x_n\},$$
- множество $\boldsymbol{\Omega}^{\alpha}$ примеров для ДСМ-обучения в соответствующей задаче ***ВСЕ_С-ЗКП*** (**Ф**) как
$$\boldsymbol{\Omega}^{\alpha} = \{A_1, A_2, \ldots, A_{n+2}\},$$

- множество $\mathbf{\Omega}^{-\alpha}$ контрпримеров для ДСМ-обучения в соответствующей задаче ***ВСЕ_С-ЗКП*** (**Ф**) как

$$\mathbf{\Omega}^{-\alpha} = \{A^-_1, A^-_2, \dots, A^-_m\},$$

где

- $A^-_1 = \boldsymbol{U} \setminus \{ x_{11}, x_{12}\}$
- $A^-_2 = \boldsymbol{U} \setminus \{ x_{21}, x_{22}\}$
- …
- $A^-_m = \boldsymbol{U} \setminus \{ x_{m1}, x_{m2}\}$

и также

- $A_1 = \{ x_2, x_3, \dots, x_n \}$
- $A_2 = \{ x_1, x_3, \dots, x_n \}$
- ...
- $A_n = \{ x_1, x_2, \dots, x_{n-1} \}$
- $A_{n+1} = A_{n+2} = \{ x_1, x_2, \dots, x_n\}$,

и, наконец,

- $SL_1 = \{ x_{11}, x_{12}\}$
- $SL_2 = \{ x_{21}, x_{22}\}$
- …
- $SL_m = \{ x_{m1}, x_{m2}\}$.

Таким образом:

1) множество всех ДСМ-сходств, порождаемых из примеров $\mathbf{\Omega}^{\alpha} = \{A_1, A_2, \dots, A_{n+2}\}$ есть

$$\{\varnothing; \{x_1\}, \{x_2\}, \dots, \{x_n\}; \{x_1, x_2\}, \{x_1, x_3\}, \dots, \{x_{n-1}, x_n\}; \{x_1, x_2, x_3\}, \dots, \{x_1, x_2, \dots, x_n\}\},$$

что позволяет перечислить (кодируя единицей на позиции i вхождение соответствующей переменной x_i в порождаемое ДСМ-сходство) все 2^n булевских наборов значений переменных $x_1, x_2, \dots, x_n$:

<00...00>, <00...01>, ... , <11...10>, <11...11>;

2) В соответствии с **Утверждением 7.1** условие ***ЗКП*** выполняется для тех и лишь для тех наборов значений переменных $x_1, x_2, \dots, x_n$, которые имеют непустое пересечение (общие переменные) с каждым из стоп-листов SL_1, SL_2, …, SL_m . Но это, в свою очередь, означает, что для каждой из подформул

$$\mathbf{\Phi}_i = (x_{i1} \vee x_{i2}), \quad \text{где } i \in \{1, 2, \dots, m\},$$

выражения

$$\mathbf{\Phi}(x_1, x_2, \dots, x_n) = \mathbf{\Phi}_1 \,\&\, \mathbf{\Phi}_2 \,\&\, \dots \,\&\, \mathbf{\Phi}_m = (x_{11} \vee x_{12}) \,\&\, (x_{21} \vee x_{22}) \,\&\, \dots \,\&\, (x_{m1} \vee x_{m2})$$

имеется по крайней мере одна (общая с соответствующим стоп-листом) из пары переменных $\langle x_{i1}, x_{i2}\rangle$, определяющая также и выполнимость подформулы $\mathbf{\Phi}_i$. Таким образом, выполнимость функции **Ф** имеет место тогда и только тогда, когда в соответствующем «контексте» $\langle \boldsymbol{U}(\mathbf{\Phi}), \mathbf{\Omega}^{\alpha}(\mathbf{\Phi})$ и $\mathbf{\Omega}^{-\alpha}(\mathbf{\Phi})\rangle$ выполняется условие ***ЗКП***.

Предлагаемый «контекст» $\langle \boldsymbol{U}(\mathbf{\Phi}), \mathbf{\Omega}^{\alpha}(\mathbf{\Phi})$ и $\mathbf{\Omega}^{-\alpha}(\mathbf{\Phi})\rangle$ строится по соответствующей формуле **Ф** полиномиально быстро, что и определяет полиномиальную сводимость задачи ***МОНОТОННАЯ ВЫПОЛНИМОСТЬ*** к задаче ***ВСЕ_С-ЗКП***.

Уже подробно рассмотренная нами для случая работы с простыми ДСМ-сходствами техника управления перебором *восстановление псевдо-деревьев по их каркасам* (сокращенно: *ПД + Каркасы*) также может быть полезна и при обработке дополнительных логических условий – запрета на контрпримеры, единственности причины и др. (например [1] и др.) обоих знаков. Особенностью применения подобной техники оптимизации перебора в данном случае являются как (уже названная выше) необходимость оперировать примерами обоих знаков (примерами и контрпримерами), так и дополнительные ограничения на допустимость порождаемых гипотез, обусловленные структурными особенностями τ-примеров, доопределяемых в каждом конкретном случае (например, при проверке выполнимости Условия Каузальной Полноты - например [1] и др.).

Конкретные алгоритмы перечисления всех сходств, удовлетворяющих условию запрета на контрпримеры легко сформировать, например, дополнив уже подробно представленную выше технику *ПД + Каркасы* процедурами направленного перебора стоп-листов, рассмотренную при доказательстве **Утверждения 7.2**.

Итак, переформулируем условие принадлежности пары $\langle v,w\rangle$ множеству всех сходств из $\mathbf{D\_GC}(\boldsymbol{U}, \mathbf{\Omega})$, дополнительно удовлетворяющих (α)-условию ***ЗКП*** (т.е. условию принадлежности множеству $\boldsymbol{С\text{-}ЗКП}^{\alpha}$ – позитивных, когда α есть +, и негативных, когда α есть -):

$[(\langle v,w\rangle \in (\boldsymbol{С\text{-}ЗКП}^{\alpha})]$ ***iff***[4]

iff $\forall\omega$ ($(\omega \in \mathbf{\Omega}^{-\alpha}) \rightarrow \exists v_\omega [(v_\omega \subseteq SL_\omega) \,\&\, (v_\omega \subseteq v)]$ $\&\ (v \in \mathbf{D\_GC}(\boldsymbol{U}, \mathbf{\Omega}))$)

т.е. пара $\langle v,w\rangle$ соответствует условию ***ЗКП*** тогда и только тогда, когда для каждого контр-

[4] Т.е. ***тогда и только тогда, когда***

примера ω из множества $\boldsymbol{\Omega}^{-\alpha}$ найдется некоторое непустое подмножество v_ω образующих из ***U***, которое содержится одновременно в стоплисте SL_ω для контрпримера ω и в сходстве v из пары <v,w>.

Алгоритм перечисления всех элементов соответствующего множества ***С-ЗКП***$^{\alpha}$ (**Алгоритм *С-ЗКП***$^{\alpha}$) определяет следующая

Схема **Алгоритма *С-ЗКП***$^{\alpha}$

1. Рассмотрим все непосредственно доминируемые корнем A_i вершины текущего псевдодерева номер i. (Их не более чем $|A_i| = \{a_1, a_2, \ldots, a_r\} = r \le n = |\boldsymbol{U}|$).

2. Для каждой из таких вершин с текущим номером j рассмотрим кортеж ее пересечений (как множества образующих из ***U***) со всеми элементами имеющегося множества стоплистов $\boldsymbol{SL}(\boldsymbol{\Omega}^{-\alpha})$:

$$\boldsymbol{U\_SL} = < U\_SL_1, U\_SL_2, \ldots, U\_SL_s >.$$

3. Если кортеж $\boldsymbol{U\_SL}(A_{ij})$ не содержит пустых элементов, то искомая ДСМ-гипотеза из ***С-ЗКП***$^{\alpha}$ существует (т.е. множество ***С-ЗКП***$^{\alpha}$ в этом случае не пусто).

4. Возьмем в каждом из элементов кортежа $\boldsymbol{U\_SL}(A_{ij})$ непустое подмножество $U\_SL_k\ (A_{ij})^*$ и рассмотрим (как соответствующее замыкание)

$$v^*(A_{ij}) = [\ \bigcup_{k=1}^{s} U\_SL_k\ (A_{ij})^*\ ]_{(\boldsymbol{U},\boldsymbol{\Omega})}.$$

5. Нетрудно убедиться, что по построению $v^*(A_{ij})$ есть элемент множества ***С-ЗКП***$^{\alpha}$.

Дополнительный комментарий (ДК).
При этом все множество ***С-ЗКП***$^{\alpha}$ будет располагаться в данном случае между

- верхней по вложению границей, определяемой замыканием объединения всех элементов кортежа $\boldsymbol{U\_SL}(A_{ij})$ и
- нижней границей, определяемой всеми замыканиями объединений всех одноэлементных подмножеств всех компонентов кортежа $\boldsymbol{U\_SL}(A_{ij})$ (когда из каждого такого компонента берется ровно одна образующая).

Опираясь на использованную при доказательстве **Утверждение 7.3** «техническую» конструкцию, можно показать, что размеры нижней границы множества ***С-ЗКП***$^{\alpha}$ определяет

Следствие 7.1

Задача о числе элементов нижней границы (*ДК* выше) множества ***С-ЗКП***$^{\alpha}$ перечислительно полна.

Таким образом, для «быстрого» поиска подходящих элементов множества вновь можно воспользоваться техникой *ПД+Каркасы*:

- построить необходимые стоп-листы и
- перейти к систематическому обзору вершин, непосредственно доминируемых корнями соответствующих псевдо-деревьев (например, воспользовавшись возможностями полиномиально быстрого порождения каркасов или же - в случае попадания в соответствующие гиперкубы - возможностями быстрого порождения «верхней границы» - т.е. множества вершин вида *единственный 0 в кортеже 1* – в каждом из таких гиперкубов);
- далее, используя явный вид уточнения условий «полезности» соответствующих элементов ***С-ЗКП***$^{\alpha}$ (например, перечень «обязательных» образующих и т.п.) сперва породить «полезные» элементы множества ***С-ЗКП***$^{\alpha}$, а затем, если потребуется, - и все остальные.

8. Работа с ДСМ-правилами второго рода

Перейдем к обсуждению возможностей предлагаемой техники сокращений перебора при реализации ДСМ-правил правдоподобного вывода второго рода (ППВ-II - например [1] и др.). Предметом анализа здесь будет техника поиска причинных зависимостей, позволяющих формировать заключения (ДСМ-прогноз) о свойствах новых объектов (исходно недоопределенных примеров).

Пусть заданы:

Ex = $\{Ex_1, Ex_2, \ldots, Ex_m\}$ – исходное множество примеров (в том числе – примеров и контрпримеров для ДСМ-обучения. При этом, если не оговорено противное, будем считать, что в множестве ***Ex*** перечислены примеры одного «знака». Для разделения примеров разных знаков будем использовать дополнительный верхний индекс - $\boldsymbol{Ex}^+$ и $\boldsymbol{Ex}^-$). Будем также считать, что каждый пример

Ex_i из ***Ex*** характеризуется описанием соответствующего объекта A_i (формирующего его множества образующих исходного алфавита

объектов U_L) а также перечнем его свойств C_i (представляемым соответствующим множеством образующих алфавита свойств U_R), связанными отношением $=>_1$ (отношением «объект обладает свойствами»)

$$(A_i =>_1 C_i), \quad i = 1,2, \ldots ,m;$$

$U = U_L \cup U_R$ - объединенный алфавит, а

$\Omega_L = \{A_1, A_2, \ldots , A_m\}$ - множество описаний объектов из $\boldsymbol{Ex}$: $A_i \in 2^{UL}$,

$\Omega_R = \{C_1, C_2, \ldots , C_m\}$ - множество описаний свойств этих объектов: $C_i \in 2^{UR}$, кроме того:

$\boldsymbol{Ex}^\tau = \{Ex^\tau_1, Ex^\tau_2, \ldots , Ex^\tau_s\}$ – исходное множество примеров, предложенных для ДСМ-доопределения (т.е. множество примеров Ex^τ_i вида $A^\tau_i =>_1 C^\tau_i$, где i = 1,2, ... ,s).

По определению ППВ-II для каждого Ex^τ_r из $\boldsymbol{Ex}^\tau$ необходимо найти соответствующий набор $\boldsymbol{H}$ гипотез h_j вида

$$(v_j =>_2 w_j), \quad j = 1,2, \ldots , p(r)$$

таких, что

$$\bigcup_{j=1}^{p(r)} w_j = C^\tau_r .$$

(При этом последнее равенство – ключевое условие в определении решающих предикатов и соответствующих правил правдоподобного вывода по аналогии – подробнее [1] и др., - где, в частности, для позитивной аналогии

$$\boldsymbol{\Pi}^+_{\boldsymbol{n}}(A^\tau_r, C^\tau_r, k) =$$
$$\exists w_1 \ldots \exists w_k \{([\exists v_1 (\mathbf{J}_{<1,r>}(v_1 ==>_2 w_1) \& (v_1 \subset A^\tau_r)] \&$$
$$\& [\exists v_2 [\, \mathbf{J}_{<1,r>}(v_2 ==>_2 w_2) \& (v_2 \subset A^\tau_r)] \& \ldots$$
$$\ldots \& [\exists v_k [\mathbf{J}_{<1,r>}(v_k ==>_2 w_r) \& (v_k \subset A)] \&$$
$$\& (w_1 \cup w_2 \cup \ldots \cup w_r = C^\tau_r)) \&$$
$$\& \forall U ((U \subseteq C^\tau_r) \& (U \neq \varnothing)) \rightarrow$$
$$\rightarrow \neg \exists Z ([\, \mathbf{J}_{<-1,r>}(Z ==>_2 U) \vee \mathbf{J}_{<0,r>}(Z ==>_2 U)\,] \&$$
$$\& (Z \subseteq A^\tau_r))\},$$

а предикаты $\boldsymbol{\Pi}^-_{\boldsymbol{n}}$ и $\boldsymbol{\Pi}^0_{\boldsymbol{n}}$ формулируются путем перехода, соответственно, к учету негативных зависимостей и зависимостей одновременно обоих знаков по отношению к доопределяемому примеру Ex^τ_r).

Искать необходимые для проверки целевого (напомним, речь идет о покрытии C^τ_r множествами w_j , где j пробегает значения от 1 до p(r)) соотношения w_j удобно в псевдо-деревьях $\mathbf{D\_GC}(\boldsymbol{U}_R, \boldsymbol{\Omega}_R)\big|_{\boldsymbol{Ci}}$, построенных на диаграмме сходств $\mathbf{D\_GC}(\boldsymbol{U}_R, \boldsymbol{\Omega}_R)$ примеров (в данном случае – свойств изучаемых объектов) из Ω_R.

Для этого: последовательно для каждого C_i из Ω_R строим каркас $GK_{\mathbf{D},Ci}$ соответствующего псевдо-дерева $\mathbf{D\_GC}(\boldsymbol{U}_R, \boldsymbol{\Omega}_R)\big|_{\boldsymbol{Ci}}$. На каждом таком каркасе выбираем все те $w_{j(i)}$, для которых

$$w_j \subseteq C^\tau_r ,$$

затем вычисляем $\bigcup_j w_j$.

Если это объединение строго вкладывается в C^τ_r, переходим к следующему элементу C_{i+1} из Ω_R а также следующему псевдо-дереву $\mathbf{D\_GC}(\boldsymbol{U}_R, \boldsymbol{\Omega}_R)\big|_{\boldsymbol{Ci+1}}$ (т.к. для текущих множества свойств C_{i+1} и псевдо-дерева $\mathbf{D\_GC}(\boldsymbol{U}_R, \boldsymbol{\Omega}_R)\big|_{\boldsymbol{Ci}}$ имеющиеся ППВ-II не работают).

Если же

$$\bigcup_j w_j = C^\tau_r ,$$

обращаемся к перечислению всех таких подпокрытий найденного нами покрытия $\boldsymbol{w}(r)$ для анализируемого множества свойств C^τ_r, которые, как и прежде, сохраняют найденное качество - покрывать C^τ_r.

Для этого можно использовать, например, следующий

Алгоритм 8.1

Вход: множество C^τ_r (свойств примеров из $\boldsymbol{Ex}^\tau$) и покрывающее в точности его множество

$$\boldsymbol{w}(r) = \{w_{j1(r)}, w_{j2(r)}, \ldots, w_{jp(r)}\}$$

наборов свойств в ДСМ-гипотезах h_j вида

$$v_j =>_2 w_j, \quad j = 1,2, \ldots , p(r) ,$$

порожденных применением ППВ-I из примеров множества $\boldsymbol{Ex}$.

Выход: множество всех подмножеств $\boldsymbol{w}'(r)$ множества $\boldsymbol{w}(r)$, таких, что объединение элементов каждого из них попрежнему покрывает целевое множество свойств C^τ_r.

*Описание **Алгоритма**:*

Комментарий: Перенумеруем элементы исходного покрытия $\boldsymbol{w}$ и поместим их в исходный перечень **ТЕКУЩЕЕ ПОКРЫТИЕ**. Стек **ИЗБЫТОЧНЫЕ ЭЛЕМЕНТЫ** используем для перечисления (в лексикографическом порядке - !) комбинаций избыточных (т.е. таких,

удаление которых не нарушает покрытие множества свойств C^{τ}_{r} элементами перечня **ТЕКУЩЕЕ ПОКРЫТИЕ**) элементов исходного покрытия.

ШАГ 1. В список **ТЕКУЩЕЕ ПОКРЫТИЕ** заносим все r элементов исходного покрытия $\boldsymbol{w}(r)$. Переменной **КОРЕНЬ** присваиваем значение w[1].

ШАГ 2. Переменной **ТЕКУЩИЙ ЭЛЕМЕНТ ПОКРЫТИЯ** присваиваем текущее значение переменной **КОРЕНЬ**. Стек **ИЗБЫТОЧНЫЕ ЭЛЕМЕНТЫ** назначаем пустым.

ШАГ 3. Переносим элемент **ТЕКУЩИЙ ЭЛЕМЕНТ ПОКРЫТИЯ** из списка **ТЕКУЩЕЕ ПОКРЫТИЕ** в стек **ИЗБЫТОЧНЫЕ ЭЛЕМЕНТЫ**.

ШАГ 4. Проверяем условие «*входящие в* **ТЕКУЩЕЕ ПОКРЫТИЕ** *элементы исходного перечня* $\boldsymbol{w}(r)$ *покрывают целевое множество свойств C*». Если это условие выполнено, отправляем текущий вариант списка **ТЕКУЩЕЕ ПОКРЫТИЕ** на **ВЫХОД Алгоритма** (например, на печать) и переходим на ШАГ 6. В противном случае

ШАГ 5. Возвращаем текущее значение **ТЕКУЩИЙ ЭЛЕМЕНТ ПОКРЫТИЯ** из стека **ИЗБЫТОЧНЫЕ ЭЛЕМЕНТЫ** в список **ТЕКУЩЕЕ ПОКРЫТИЕ**, затем переходим на ШАГ 7.

ШАГ 6. Переменной **ТЕКУЩИЙ ЭЛЕМЕНТ ПОКРЫТИЯ** присваиваем следующее за текущим значение (заносим сюда следующий элемент исходного покрытия $\boldsymbol{w}(r)$). Если это невозможно (т.е. на данном шаге Алгоритма **ТЕКУЩИЙ ЭЛЕМЕНТ ПОКРЫТИЯ** содержит последний элемент исходного покрытия $\boldsymbol{w}(r)$),

ШАГ 7. Откатываемся в списке **ИЗБЫТОЧНЫЕ ЭЛЕМЕНТЫ** к предпоследнему элементу w[i] (удаляем из стека **ИЗБЫТОЧНЫЕ ЭЛЕМЕНТЫ** в его текущем состоянии последний элемент - w[j]). Если это невозможно (т.е. w[j] – корень этого стека: его значение совпадает с текущим значением переменной **КОРЕНЬ**), переходим на ШАГ 9, в противном случае

ШАГ 8. Переходим к следующему (за помещенным в **ТЕКУЩИЙ ЭЛЕМЕНТ ПОКРЫТИЯ** на данный момент w[j] в исходном перечне $\boldsymbol{w}(r)$) элементу w[j+1]: помещаем в **ТЕКУЩИЙ ЭЛЕМЕНТ ПОКРЫТИЯ** этот элемент w[j+1] и возвращаемся на ШАГ 3.

ШАГ 9. Переменной **КОРЕНЬ** присваиваем следующее за текущим значение (помещаем в нее следующий элемент исходного покрытия $\boldsymbol{w}(r)$). Если это невозможно (т.е. все элементы исходного покрытия уже были рассмотрены в качестве корня стека избыточных элементов покрытия), переходим на ШАГ 10. В противном случае переходим на ШАГ 2.

ШАГ 10. **СТОП**

Корректность предложенного **Алгоритма** определяет

Утверждение 8.1

В части перечисления всех удовлетворяющих условию «*входящие в* **ТЕКУЩЕЕ ПОКРЫТИЕ** *элементы исходного перечня* $\boldsymbol{w}(r)$ *покрывают целевое множество свойств C*» подпокрытий для заданного исходного покрытия $\boldsymbol{w}(r)$ **Алгоритм 8.1** характеризуют свойства полноты и точности.

Доказательство

Реализованная в **Алгоритме 8.1** тактика перебора подпокрытий исходного покрытия $\boldsymbol{w}(r)$ в *лексикографическом порядке* (т.е. начиная с самых «длинных» стеков избыточных элементов, и далее - ко все более и более коротким – с выбором в качестве корня каждого такого стека избыточных элементов только таких элементов исходного покрытия $\boldsymbol{w}(r)$, которые имеют все более и более «старшие» номера; при этом переход от текущего корня стека к следующему организован так, что после каждого такого перехода рассматриваются лишь те элементы исходного покрытия $\boldsymbol{w}(r)$, номера которых строго больше, чем номер элемента, стоящего в корне на текущем шаге) гарантирует, что каждая целевая последовательность отбрасываемых элементов исходного покрытия $\boldsymbol{w}(r)$ будет сформирована лишь один раз. В дополнение к этому, постоянный контроль выполнимости условия «*входящие в* **ТЕКУЩЕЕ ПОКРЫТИЕ**

элементы исходного перечня ***w***(r) *покрывают целевое множество свойств C*» гарантирует, что ни одно из подходящих подпокрытий не будет пропущено. Наконец, никаких других (отличных от перечисленных) объектов **Алгоритм 8.1** не строит.

Будем называть покрытия, порождаемые **Алгоритмом 8.1**, *полезными*. Для каждого такого покрытия ***w'***(r) выделим из множества ***Ex*** подмножество ***Ex***(r) всех формирующих ***w'***(r) примеров:

$$\boldsymbol{Ex}(r) = \{ Ex_{i1(j)}, Ex_{i2(j)}, \dots , Ex_{im(j)} \} .$$

Далее на всех элементах $\Omega_{L(r)} = \{ A_{i1(j)}, A_{i2(j)}, \dots , A_{im(j)} \}$, т.е. на левых частях этих примеров будем перечислять соответствующие ДСМ-сходства (замыкания Галуа). Для этого: для каждого порождаемого на элементах множества $\Omega_{L(r)}$ псевдо-дерева $\mathbf{D\_GC}(\boldsymbol{U}_L, \boldsymbol{\Omega}_L(r))$ строим каркас $GK_{\mathbf{D,A}}$ и для каждого его элемента проверяем, лежит ли он в порождаемом всеми примерами из Ω_L множестве ДСМ-сходств (т.е., сперва: лежит ли он на обоих каркасах, далее, - шаг за шагом по расширениям этих каркасов – возможно, вплоть до проверки его принадлежности одновременно обоим соответствующим псевдо-деревьям, – последовательно перебирая, если потребуется их «промежуточные» приближения). При этом для каждого элемента w_j текущего (анализируемого) покрытия для C^{τ}_r достаточно найти в обсуждаемом здесь процессе анализа хотя бы одну использующую его ДСМ-гипотезу. Любой набор таких гипотез (для каждого элемента покрытия – хотя бы по одной), очевидно, удовлетворяет условиям выполнимости ППВ-II.

9. Учет особенностей работы с множествами свойств исходных ДСМ-примеров

При проверке попадания текущего анализируемого сходства v_j из текущей диаграммы $\mathbf{D\_GC}(\boldsymbol{U}_L, \boldsymbol{\Omega}_L(r))$, построенной на левых частях отобранных в ***Ex***(r) из ***Ex*** примеров, в диаграмму $\mathbf{D\_GC}(\boldsymbol{U}_L, \boldsymbol{\Omega}_L)$, построенную на всех левых частях примеров из ***Ex***, возможны две ситуации:

a) орбиты обоих сходств совпали (искомый нами случай);

b) орбита в диаграмме $\mathbf{D\_GC}(\boldsymbol{U}_L, \boldsymbol{\Omega}_L)$ больше, чем орбита в диаграмме $\mathbf{D\_GC}(\boldsymbol{U}_L, \boldsymbol{\Omega}_L(r))$ (меньше она быть не может, т.к. ***Ex***(r) отбирают из ***Ex***). В этом случае следует проверить, как дополнительные (к выделенным в ***Ex***(r)) примеры из ***Ex*** сокращают (ведь их правые части – множества свойств, в отличие от правых частей примеров из ***Ex***(r), не входят в рассматриваемое *полезное* покрытие для C^{τ}_r - !) размеры текущего w_{j0}: удастся ли оставшимися зависимостями (более точно – множествами свойств w_j в их правых частях) все же «покрыть» удаляемые из w_{j0} (учетом дополнительных к ***Ex***(r) примеров из ***Ex***) образующие. В случае положительного ответа на этот вопрос – на текущем шаге перебора получено искомое решение, в противном случае следует переходить к следующему полезному покрытию для C^{τ}_r.

Основные особенности предлагаемой процедуры:

- размер $\boldsymbol{U}_R$ - алфавита правых частей (анализируемых примеров и каузальных зависимостей) обычно существенно меньше, чем размер алфавита $\boldsymbol{U}_L$ левых частей. Таким образом, характеризующая объемы перебора при поиске ДСМ-сходств «комбинаторика» правых частей, вообще говоря, менее «тяжела», чем «комбинаторика» анализа их левых частей. Как следствие:
- возникновение гиперкубов в псевдо-деревьях на правых частях исходных примеров и вновь порождаемых зависимостей есть существенно менее частое явление, чем возникновение аналогичных «тяжелых» комбинаторных объектов на левых частях изучаемых пар <*объект (подобъект), свойства*>;
- открывающиеся возможности сокращения «комбинаторики» перебора при работе с правыми частями определяют роль каркасов псевдо-деревьев (как инструмента быстрого порождения приближенных описаний соответствующих псевдо-деревьев): они (такие каркасы) на правых частях примеров и зависимостей более точно аппроксимируют соответствующие псевдо-деревья. Действительно, гиперкубов на правых частях (описаниях свойств) – существенно меньше, чем на левых (описаниях объектов). Да и размер их (если такие гиперкубы все же возникают), соответственно, существен-

но меньше, чем в случае операций с левыми частями;

- при переносе рассмотрения на левые части $=>_1$ (после выделения соответствующих покрытий и соотнесения их с каркасами и др. на правых частях) последовательный перебор по представленной схеме

от примеров обучающей ДСМ-выборки - к каркасам соответствующих псевдо-деревьев, и далее (если потребуется) – ко все более и более точным приближениям соответствующих псевдо-деревьев и всей диаграммы ДСМ-сходств

позволяет, стартовав с быстро порождаемых объектов – каркасов, - вести предлагаемое сравнение на все более точных приближениях для восстанавливаемых псевдо-деревьев так, чтобы завершить его (это сравнение) при первом же получении искомой причинной зависимости вида $v =>_2 w$.

Говоря о возможностях и ограничениях рассматриваемой техники управления перебором, дополнительно также следует заметить, что задействованное в формулировке **Утверждения 3.2** (в первой части настоящей работы) требование об отсутствии дубликатов в исходном множестве примеров Ω не накладывает каких-либо существенных ограничений на общность рассматриваемой здесь конструкции. Действительно, при наличии в Ω пары одинаковых примеров $A_1 = A = A_2$ достаточно удалить их из обработки и добавить к диаграмме $\mathbf{D\_GC}_{\varphi,f}$ (***U*,Ω\{A}**) дополнительную вершину A, чтобы, сократив объем вычислений, необходимый в случае работы с исходным множеством примеров Ω, получить тот же самый финальный результат.

Заключение

Подведем итоги предпринятого нами рассмотрения. Прежде всего отметим, что представленная выше техника декомпозиции множества всех ДСМ-гипотез, порождаемых на заданных алфавите ***U*** и множестве примеров **Ω**, дает возможности последовательно формировать (быстрой процедурой - !) каркасы соответствующих деревьев отдельно для каждого - 1-го, 2-го, …, последнего объектов из исходного множества примеров **Ω**, выбирая (надежно управляемым образом - !) на каждом из каркасов те веточки частичного порядка и те конкретные ДСМ-гипотезы (с учетом входящих в них образующих, с учетом порождающих их примеров – «родителей», …), которые являются «интересными» в смысле тех или иных «внешних» (по отношению к организации процедуры перебора вариантов, в том числе – содержательных, отражающих специфику предметного знания об исследуемой средствами ДСМ-метода прикладной предметной области) соображений. (Например, это могут быть ДСМ-гипотезы, которые непременно содержат образующие из некоторого *специально выделенного подмножества*, или же гипотезы, удовлетворяющие заданным ограничениям на *число входящих в них образующих*, и т.п.).

Не менее интересные возможности предлагаемая техника декомпозиции предоставляет в части проверки выполнимости весьма важного в ДСМ-рассуждении условия так называемой каузальной полноты (которое предполагает для всех элементов заданного конкретного множества примеров наличие ДСМ-«объяснения» порождаемыми из него гипотезами,). В частности, здесь появляются возможности для *ускоренной* проверки выполнимости условия каузальной полноты, причем, если потребуется, - ДСМ-гипотезами необходимого формата (например, - заданного размера, или же – содержащими заданные образующие и т.п.). Аналогичным образом может быть организована ускоренная (т.е. не требующая порождения всех возможных в каждом рассматриваемом случае ДСМ-гипотез) проверка ДСМ-доопределимости заданного нового объекта (примера). Таким образом, можно организовать *приближенные* (однако, снабженные надежным механизмом *управления* порождением рассматриваемых приближений) ДСМ-рассуждения: в первую очередь порождать те или иные «полезные» результаты за счет первоочередного формирования «полезных» ДСМ-гипотез. Далее, перебрав последовательно все ветки частичного порядка во всех рассматриваемых деревьях частичного порядка, (порождаемых в свою очередь в результате декомпозиции описанными выше средствами всего множества ДСМ-гипотез на отдельные подструктуры), сформировать «предельный случай» такого сорта приближений – целиком все множество ДСМ-гипотез.

Наконец, предоставляемая обсуждаемым подходом возможность анализировать отдельно каждое из соответствующих деревьев рассмотренного выше типа, имеющих в корне примеры из исходной выборки **Ω**, позволяет организовать *параллельную обработку* имеющихся ДСМ-данных.

Итак, предлагаемая техника управления перебором (техника *ПД+Каркасы*) позволяет организовать управляемую «навигацию» во множествах соответствующих ДСМ-сходств. Таким образом, появляется возможность

- сначала искать «полезные» (в уже оговоренном выше смысле – т.е. ДСМ-сходства, содержащие, например, некоторые наперед заданные образующие и т.п.) результаты, а далее, если потребуется,
- последовательно достраивать их множество до множества всех соответствующих результатов (простых ДСМ-сходств; гипотез, удовлетворяющих заданной системе дополнительных ограничений ***ЗКП*** и др.; гипотез, полезных для проверки ДСМ-доопределимости элементов заданного множества τ-примеров или же проверки выполнимости Условия Каузальной Полноты).

При этом в каждом из представленных выше направлений организации перебора:

- при поиске простых сходств,
- при анализе дополнительных логических условий (ЗКП, единственность сходства и др.) а также
- при проверке доопределимости τ-примеров и\или проверке Условия Каузальной Полноты

общая «стратегия» последовательного восстановления соответствующих псевдо-деревьев по их каркасам (с порождением в первую очередь «полезных» ДСМ-гипотез) оказывается эффективным инструментом оптимизации объемов необходимых вычислений.

Дополнительные возможности для обсуждаемой оптимизации дает распараллеливание реализуемого ДСМ-поиска: как уже было показано выше, каждое из псевдо-деревьев, составляющих восстанавливаемую диаграмму сходств, может быть «обсчитываемо» в независимом от оставшихся псевдо-деревьев режиме. Более того, каждый из соответствующих каркасов (а также его последовательные расширения до соответствующего псевдо-дерева) может также быть восстанавливаем независимо от других (т.е., в том числе - и в параллельном режиме вычислений).

Одним из вариантов организации подобного распараллеленного «счета» может быть, например, следующий:

- на имеющемся «физическом» системно-техническом ландшафте (вычислительных мощностях, коммуникационном оборудовании и устройствах хранения данных) формируем виртуальную (логическую) «облачную» вычислительную среду;
- в сформированной логической среде, исходя из размеров исходного множества примеров для ДСМ-обучения и уточнения понятие «полезные» гипотезы а также выбора стратегии порождения ДСМ-результатов (например, восстановления лишь «полезных» гипотез, или же восстановления сперва лишь «полезных», а затем достраивания их множества до полного множества ДСМ-гипотез и т.п.), «поднимаем» каждый раз *необходимое* (т.е. определяемое заданными параметрами исходных данных для ДСМ-обучения а также текущими комбинаторными характеристиками анализируемых псевдо-деревьев - их архитектурой, глубиной рекурсии при раскрытии встречающихся в них гиперкубов и т.п.) число параллельно работающих копий «типовой» виртуальной машины (исполнимого «типового» программного модуля ДСМ-анализа каждого такого дерева), реализующей алгоритмику ДСМ-вычислений по представленной выше схеме *ПД+Каркасы* (обеспечивая при этом существенное ускорение ДСМ-вычислений за счет параллельной обработки всех псевдо-деревьев рассмотренного нами выше типа);
- при этом при вычислении базиса замыканий Галуа для рассматриваемых подмножеств образующих определенную пользу (в части сокращения времени вычислений) может оказать использование специализированных аппаратно-схемных технологических комплексов (например, TCAM-решений[5] и т.п.).

Наконец, отметим, что предложенная техника управления перебором при порождении ДСМ-гипотез – возможный вариант решения

[5] TCAM (ternary content addressable memory) – специализированные схемные решения реализации ассоциативной памяти.

проблемы эффективной применимости ДСМ-метода при работе с большими выборками исходных данных. Действительно: описанные нами возможности

- декомпозиции исходной задачи на подзадачи, дополненные очерченными выше перспективами

- организации параллельных вычислений при компьютерной реализации ДСМ-рассуждений, а также возможности

- вести приближенные ДСМ-вычисления (при которых целенаправленным образом порождается лишь некоторое «полезное» подмножество ДСМ-гипотез)

позволяют надеяться на *существенное расширение* сложившихся к настоящему времени рамок использования техники интеллектуального анализа данных средствами ДСМ-метода при решении прикладных задач.

Литература

1. Финн В.К. Индуктивные методы Д.С. Милля в системах искусственного интеллекта // Искусственный интеллект и принятие решений. - 2010. – Часть I: №3, С.3 -21, Часть II: №4, С. 14 -40.
2. Автоматическое порождение гипотез в интеллектуальных системах. (Ред. - Финн В.К.). – М.: Либроком, 2009. – 528 С.
3. Гэри М., Джонсон Д.С. Вычислительные машины и трудно-решаемые задачи. - М.: Мир. - 1982. - 416 С.
4. Simon J. On the difference between one and many. - Lect. Not. Comp. Sci. -V.52 (1977). - Pp. 480-491.
5. Valiant L.G. The complexity of enumeration and reliability problems. - SIAM J.Comput. - 8 (1979), N1. - Pp. 410-421.
6. Valiant L.G. The complexity of computing the permanent. - Theoretical Computer Science. - 8 (1979). - Pp. 189-201.

Забежайло Михаил Иванович. Управляющий директор Центра прикладных исследований компьютерных сетей. Окончил Московский физико-технический институт в 1979 году. Кандидат физико-математических наук, старший научный сотрудник (доцент). Автор более 70 печатных работ. Область научных интересов: интеллектуальный анализ данных.
E-mail: mzabezhailo@arccn.ru