МЕТОДЫ И ТЕХНОЛОГИИ ПРИНЯТИЯ РЕШЕНИЙ

УДК 004.89, 004.832 DOI: 10.18287/2223-9537-2020-10-4-503-515

Компактное представление ограничений на основе новой интерпретации понятия «кортеж многоместного отношения»

А.А. Зуенко

Институт информатики и математического моделирования – обособленное подразделение Федерального государственного бюджетного учреждения Федерального исследовательского центра «Кольский научный центр Российской академии наук», Апатиты, Мурманской обл., Россия

Аннотация

Рассматриваются различные точки зрения на понятие «кортеж многоместного отношения», используемые в математике и информационных технологиях. Особое внимание уделено эволюции понятия «кортеж» в рамках технологии программирования в ограничениях - *Constraint Programming*, где появление новых интерпретаций понятия «кортеж» связано с попытками разработать более компактное табличное представление качественных зависимостей, чем обычные реляционные таблицы. Подобное компактное представление может служить основой для ускорения процедур удовлетворения качественных ограничений. В работах-прототипах были предложены такие разновидности табличных ограничений как *compressed*-таблицы и *smart*-таблицы. При этом понятия *compressed*- и *smart*- кортежа существенно отличаются от традиционного понятия кортежа многоместного отношения. Однако, известные виды табличных ограничений не одинаково хорошо подходят для моделирования и обработки всех видов качественных зависимостей, например, возникают неудобства при моделировании продукционных правил. В статье предлагается новый вид табличных ограничений – *smart*-таблицы *D*-типа, применение которых позволяет в некоторых случаях существенно сократить расход памяти компьютера по сравнению с использованием известных типов табличных ограничений. В частности, *smart*-таблицы *D*-типа хорошо подходят для моделирования продукционных правил, некоторых типов логических выражений, а также некоторых типов глобальных ограничений.

Ключевые слова: *кортеж многоместного отношения, технология программирования в ограничениях, задача удовлетворения ограничений, табличные ограничения, качественные ограничения.*

Цитирование: *Зуенко, А.А.* Компактное представление ограничений на основе новой интерпретации понятия «кортеж многоместного отношения» / А.А. Зуенко // Онтология проектирования. – 2020. – Т. 10, №4(38). - С.503-515. – DOI: 10.18287/2223-9537-2020-10-4-503-515.

Введение

В современных информационных системах, особенно тех, что связаны с применением методов искусственного интеллекта, для описания связей между объектами часто используют понятие «отношение», а для обозначения элемента отношения – понятие «кортеж». К таким системам, например, относятся системы: моделирования рассуждений [1, 2]; онтологического инжиниринга [3]; управления базами данных (СУБД) [4]; программирования в ограничениях (*Constraint Programming - CP*) [5, 6] и т.п. Однако оказывается, что в известных автору работах присутствуют различные взгляды на такие понятия, как «многоместное отношение» и «кортеж многоместного отношения». Игнорирование этих различий может привести к неверному пониманию выдвигаемых научных идей.

В статье рассматриваются различные трактовки понятия «кортеж многоместного отношения», используемые в математике и информационных технологиях. Особое внимание уделено эволюции понятия «кортеж» в рамках технологии *CP*. Базовым в рамках данной технологии является понятие ограничения, которое по смыслу очень близко понятию многоместного отношения. Основной раздел посвящён развитию технологии *CP* в части повышения эффективности обработки качественных ограничений, таких как: продукционные правила, логические выражения, бинарные и многоместные отношения и т.п. В настоящий момент процедуры удовлетворения качественных зависимостей, поддерживаемые существующими библиотеками *CP*, недостаточно эффективны. Перспективным подходом к представлению и обработке качественных ограничений следует признать подход, основанный на применении их специализированного табличного представления. В рамках *CP* был выделен отдельный класс глобальных ограничений – табличные ограничения. Простейшим примером табличных ограничений являются реляционные таблицы. В виде реляционных таблиц, содержащих множество выполняющих подстановок, может быть представлен любой конечный предикат. Однако, такое эксплицитное представление для многих отношений нецелесообразно, поскольку приводит к экспоненциальному росту сложности процедур удовлетворения ограничений. Поэтому предпринимаются попытки разработать более компактное табличное представление качественных зависимостей [7-10].

Несмотря на предпринимаемые усилия, известные виды табличных ограничений не одинаково хорошо подходят для моделирования и обработки всех видов качественных зависимостей, например, возникают неудобства при моделировании продукционных правил. В работе предлагается новый тип табличных ограничений – *smart*-таблицы *D*-типа, а также приводятся утверждения, позволяющие эффективно редуцировать вводимые табличные структуры, сохраняя множество решений исходной задачи. Применение *smart*-таблиц *D*-типа позволяет в некоторых случаях существенно сократить расход памяти компьютера по сравнению с известными типами табличных ограничений.

1 Разбор понятия «кортеж многоместного отношения»

1.1 Понятия «кортеж» и «многоместное отношение»

«Кортеж», как и понятие «множество», относится к первичным понятиям (т.е. не определяемым строго) [11].

Под кортежем традиционно понимается упорядоченный набор конечного числа объектов любой природы. Число объектов называется длиной кортежа, а сами объекты – компонентами кортежа. Кортеж длиной 2 называется парой, длиной 3 – тройкой, длиной n – n-кой.

Отличие кортежа от конечного множества состоит в следующем. Во-первых, в кортеже его компоненты могут совпадать. Например, кортеж всех букв в слове «математика» – это набор букв (м, а, т, е, м, а, т, и, к, а). Во-вторых, в кортеже очень важен порядок, в котором расположены его компоненты.

Вместо слова «кортеж» в математической литературе употребляются также в качестве синонимов термины «вектор», «набор».

Традиционно в математике под многоместным отношением понимают некоторое подмножество декартова (прямого) произведения множеств.

Определение 1.1. (Декартово произведение конечного семейства множеств).

Пусть имеется n множеств $X_1, \ldots, X_n$, тогда их декартово произведение определяется как $X_1 \times \ldots \times X_n = \{(x_1, x_2, \ldots, x_n) \mid x_i \in X_i\}$, где запись $(x_1, x_2, \ldots, x_n)$ обозначает упорядоченную n-ку, в которой первая компонента принадлежит X_1, вторая – X_2 и т.д. [11].

Определение 1.2. (Многоместное отношение). Отношением между множествами X_1, …, X_n называется любое подмножество их декартова произведения $R \subseteq X_1\times, \ldots,\times X_n$. Если требуется явно указать число n, то говорят об n-арных или n-местных отношениях. При n = 1, 2, 3 имеем унарное, бинарное, тернарное отношения соответственно. Унарное отношение на множестве X представляет собой подмножество множества X [11].

При табличном представлении многоместного отношения (как подмножества декартова произведения множеств) столбцу таблицы сопоставляется имя некоторого множества, из тех, что участвует в декартовом произведении. При этом, в результате перестановки строк таблицы получается таблица, представляющая то же самое отношение, а при перестановке столбцов получается формально другое отношение.

Пример. Пусть заданы множества $X = \{a, b, c\}$ и $Y = \{1, 2\}$. Необходимо удостовериться, что: $X \times Y \neq Y \times X$. При этом $X \times Y$ и $Y \times X$ являются *универсальными отношениями*.

Тогда: $X \times Y = \{(a, 1), (a, 2), (b, 1), (b, 2), (c, 1), (c, 2)\}$ и $Y \times X = \{(1, a), (2, a), (1, b), (2, b), (1, c), (2, c)\}$. Из сопоставления между собой кортежей этих отношений видно, что это два совершенно разных отношения.

В классическом определении многоместного отношения, в отличие от интерпретации, принятой в реляционной алгебре, отсутствует понятие «атрибут». Поэтому в реляционной алгебре имеют дело с абсолютно другим математическим объектом, обладающим отличающимися свойствами, но тоже называющимся «отношение», что часто приводит к путанице. При табличном задании отношений реляционной алгебры столбцу таблицы сопоставляется некоторый атрибут. Отличается также от классического и интерпретация понятия «кортеж».

1.2 Формализация понятия «кортеж» в реляционной алгебре

В реляционной алгебре схемой *отношения* R называется конечное множество имен атрибутов $\{A_1, A_2, ..., A_n\}$[4]. Каждому имени атрибута A_i ставится в соответствие множество D_i, называемое *доменом* атрибута A_i, $1 \le i \le n$. Домены являются произвольными непустыми конечными или счётными множествами.

Пусть $\boldsymbol{D} = D_1 \cup D_2 \cup ... \cup D_n$. *Отношение* r со схемой R – это конечное множество отображений $\{t_1, t_2,\ldots, t_p\}$ из R в $\boldsymbol{D}$, причём каждое отображение $t \in r$ должно удовлетворять следующему ограничению: $t(A_i) \in D_i$, $1\le i \le n$. Эти отображения называются *кортежами.* В данной формализации понятие отношения вводится через понятие отображения, чтобы избежать любого упорядочения имен атрибутов в схеме отношения. В рассматриваемой формализации расширяется традиционное понимание кортежа как последовательности значений, заданной в определённом порядке. Под кортежем понимается множество значений, взятых по одному для каждого имени атрибута из схемы отношения.

Пусть имеется n ($n > 0$) не обязательно различных доменов $D_1, D_2, ..., D_n$. Пусть имеется n атрибутов (переменных) $A_1, A_2, ..., A_n$.

Поскольку *кортеж* в реляционной алгебре рассматривается как отображение, то он может задаваться как множество пар (A, v), где A – некоторый атрибут, а v – значение, взятое из домена атрибута A.

Таким образом, *отношение* состоит из множества кортежей, и каждый кортеж имеет одно и то же множество атрибутов. Если все домены являются простыми, то такое отношение имеет табличное представление со следующими свойствами:

- не существует дубликатов строк (кортежей);
- порядок строк (кортежей) является несущественным;
- порядок столбцов (атрибутов) является несущественным;
- все элементы таблицы являются атомарными значениями.

1.3 Эволюция понятия «кортеж» в рамках технологии *CP*

Для того, чтобы прикладная задача могла быть решена с применением технологии программирования в ограничениях, она должна быть представлена в форме *задачи удовлетворения ограничений* (*Constraint Satisfaction Problem – CSP*).

З*адача удовлетворения ограничений* состоит из трёх компонент: $<X, D, C>$ [12]. X – множество переменных $\{X_1, X_2, \ldots, X_n\}$, D – множество доменов $\{D_1, D_2, \ldots, D_n\}$, где D_i является доменом переменной X_i; C – множество ограничений $\{C_1, C_2, \ldots, C_m\}$, которые предписывают допустимые комбинации значений переменных. Каждый домен D_i описывает множество допустимых значений $\{v_1, \ldots, v_k\}$ для переменной X_i. Каждое ограничение есть пара $<scope, rel>$, где *scope* – множество переменных, которые участвуют в ограничении, а *rel* – отношение, регламентирующее допустимые комбинации значений, которые переменные из *scope* могут принимать.

Задача описывается как *присваивание* значений некоторым (*частичное присваивание*) или всем переменным (*полное присваивание*): $\{X_i = v_i, X_j = v_j, \ldots\}$. *Решением задачи CSP* является полное присваивание, которое удовлетворяет всем ограничениям.

Для различных типов ограничений разрабатываются свои способы вывода на данных ограничениях, при этом процедура вывода называется *распространением ограничений*. Любая система программирования в ограничениях обладает библиотекой стандартных функций-распространителей для обработки наиболее популярных типов ограничений. Как правило, подобные ограничения содержат нефиксированное количество переменных и носят название глобальных ограничений, например ограничение *Alldiff* $(X_1, \ldots, X_k)$ [13].

В настоящей работе рассматривается такая разновидность глобальных ограничений, как табличные ограничения [14], а также процесс эволюции, которую претерпело понятие кортежа в контексте обработки табличных ограничений.

В форме обычной таблицы, содержащей множество выполняющих подстановок, может быть описан любой конечный предикат. В этом случае трактовка понятия «кортеж» мало чем отличается от традиционной. Теоретически, в форме табличного ограничения можно представить любой тип ограничения, в силу чего данная разновидность глобальных ограничений является полезным инструментом для моделирования задач комбинаторного поиска в рамках парадигмы программирования в ограничениях.

Например, предикат $(X_1 < X_2) \wedge (X_3 \neq t)$, где домены переменных X_1 и X_2 равны множеству $\{1, 2, 3\}$, а доменом переменной X_3 является множество $\{t, k, r\}$, может быть описан следующим образом:

$$\begin{array}{ccc} X_1 & X_2 & X_3 \\ \end{array}$$
$$\begin{bmatrix} 1 & 2 & k \\ 1 & 2 & r \\ 1 & 3 & k \\ 1 & 3 & r \\ 2 & 3 & k \\ 2 & 3 & r \end{bmatrix}.$$

Однако, явное перечисление всех допустимых присваиваний приводит к экспоненциальному росту размера таблиц при росте количества переменных. Поиск на подобных таблицах возможен лишь при малой размерности решаемых задач *CSP*.

В последние годы в рамках технологии *CP* появляется много подходов, посвящённых

более компактному представлению табличных ограничений [7-10, 15, 16]. Один из них состоит в изменении классического понятия кортеж и введении в рассмотрение нового понятия – *compressed*-кортеж (*compressed tuple*) [7-9]. В отличие от обычных кортежей, *compressed*-кортежи содержат в своих компонентах не отдельные значения переменных, а подмножества значений соответствующего домена. Целиком *compressed*-кортеж соответствует декартову произведению компонент-множеств. Таблица, содержащая *compressed*-кортежи, называется *compressed*-таблицей (*compressed table*).

Ранее рассмотренному предикату $(X_1 < X_2) \wedge (X_3 \neq t)$ соответствует следующая *compressed*-таблица:

$$\begin{array}{ccc} X_1 & X_2 & X_3 \\ \end{array}$$
$$\begin{bmatrix} \{1\} & \{2,3\} & \{k,r\} \\ \{2\} & \{3\} & \{k,r\} \end{bmatrix}.$$

Дальнейшая эволюция понятия «кортеж» заключалась в том, что в [10] было предложено обобщить понятия *compressed*-кортежей и коротких поддержек (*short supports*) [15] до понятия *smart*-кортежи (*smart tuple*). В описании компонент таблицы было разрешено вместо явного перечисления подмножества элементов соответствующего домена использовать обозначения некоторых одноместных и двуместных предикатов, моделирующих простые арифметические выражения. Таблицы, состоящие из *smart*-кортежей, называются *smart*-таблицами (*smart table*). Ниже приводится *smart*-кортеж, который соответствует упомянутому ранее предикату:

$$\begin{array}{ccc} X_1 & X_2 & X_3 \\ \end{array}$$
$$\begin{bmatrix} < X_2 & * & \neq t \end{bmatrix}.$$

Обозначение фиктивной компоненты “*” используется здесь для указания того факта, что на месте переменной X_2 может находиться любое значение из домена данной переменной. Стоит заметить, что до введения понятия «*smart*-таблица» подобные структуры рассматривались автором в работе [16].

2 Представление исходной задачи удовлетворения ограничений с использованием *smart*-таблиц *D*-типа и организация логического вывода

Из приведённых примеров *compressed*-таблиц и *smart*-таблиц видно, что эти структуры хорошо подходят для моделирования дизъюнктивных нормальных форм логических формул.

Однако, с помощью известных типов табличных ограничений не всегда целесообразно описывать некоторые виды знаний, например продукционные правила.

Пример 1. Имеется следующее продукционное правило: «Если человеку больше 65 лет и у него имеется основной диагноз, отнесённый к списку M ={сахарный диабет, заболевания органов дыхания, сердечно-сосудистые заболевания}, то риск заражения COVID-19 оценивается как высокий».

Для формализации данного правила в виде ограничения введены три переменные: X – целочисленная переменная, обозначающая возраст человека и имеющая домен [0, 150]; Y – переменная, обозначающая основной диагноз человека и принимающая возможные значения из списка возможных медицинских диагнозов; Z – переменная, обозначающая риск заражения и принимающая значения из множества {a – высокий, b – низкий, c – средний}.

На языке логических формул данное ограничение может быть представлено следующим образом: $(X \geq 65) \wedge (Y \in M) \rightarrow (Z = a)$.

Раскрытие в данном выражении импликации приводит к записи: $(X < 65) \vee (Y \notin M) \vee (Z = a)$.

С помощью *smart*-таблиц это ограничение может быть представлено так:

$$\begin{array}{ccc} X & Y & Z \\ \end{array}$$
$$\begin{bmatrix} < 65 & * & * \\ * & \notin M & * \\ * & * & Z = a \end{bmatrix}.$$

Данное табличное представление при трёх значащих компонентах содержит шесть фиктивных компонент “*”. В данном иллюстративном примере видно, что использование табличных ограничений известных типов является нерациональным с точки зрения компактного представления данных.

Для эффективной обработки качественных ограничений предлагается использовать новый тип представления ограничений, а именно, *smart-таблицы D-типа*.

В отличие от широко распространенных *smart*-таблиц *C*-типа, которые соответствуют дизъюнктивным нормальным формам логических формул с элементарными одно и двуместными предикатами, *smart*-таблицы *D*-типа соответствуют конъюнктивным нормальным формам таких формул. Происхождение терминов *smart*-таблица *C*-типа и *smart*-таблица *D*-типа обусловлено тем, что первый тип таблиц состоит из *smart*-кортежей *C*-типа (от *Conjunction*), которые служат для представления конъюнкций одно и двуместных предикатов, а второй тип таблиц состоит из *smart*-кортежей *D*-типа (от *Disjunction*), которые моделируют дизъюнкции элементарных предикатов. При описании компонент *smart*-таблиц *D*-типа могут использоваться те же базовые унарные и бинарные предикаты, что используются в *smart*-таблицах *C*-типа.

Автору не известны работы, где в контексте решения задач удовлетворения ограничений рассматривались бы структуры, подобные предлагаемым *smart*-таблицам *D*-типа.

Представленное выше правило может быть смоделировано следующей *smart*-таблицей *D*-типа:

$$\begin{array}{ccc} X & Y & Z \\ ]< 65 & \notin M & = a[\end{array}.$$

Таблицы *D*-типа здесь и далее записаны с помощью перевернутых квадратных скобок. В самой верхней строке записан заголовок отношения (ограничения).

Пример 2. Пусть имеется три вещественных переменных *X*, *Y, Z,* домены переменных равны интервалу целых чисел “[1,4]”, также задан следующий набор правил (семантика правил несущественна):

1. $(X \in \{2,3\}) \wedge (Y \geq X) \rightarrow (Z = 4)$;
2. $(X \in \{2,3\}) \wedge (Y < X) \rightarrow (Z = 2)$;
3. $(X \in \{2,3\}) \wedge (Y \neq X) \rightarrow (Z = 3)$.

Путём избавления от импликации в формулах получены следующие дизъюнкты:

1. $(X \notin \{2,3\}) \vee (Y < X) \vee (Z = 4)$;
2. $(X \notin \{2,3\}) \vee (Y \geq X) \vee (Z = 2)$;
3. $(X \notin \{2,3\}) \vee (Y = X) \vee (Z = 3)$.

Пусть в дополнение к уже заданным ограничениям для переменных *X* и *Y* заданы следующие унарные ограничения: *X*=3, *Y*=3.

Этот набор ограничений может быть представлен в виде следующей *smart*-таблицы *D*-типа.

$$\begin{array}{cccc} X & Y & Z & YX \\ \end{array}$$
$$\begin{bmatrix} \notin \{2,3\} & \varnothing & =4 & < \\ \notin \{2,3\} & \varnothing & =2 & \geq \\ \notin \{2,3\} & \varnothing & =3 & = \\ =3 & \varnothing & \varnothing & \varnothing \\ \varnothing & =3 & \varnothing & \varnothing \end{bmatrix}.$$

Символ «∅» обозначает компоненту, не содержащую ни одного значения. Каждая строка данной таблицы соответствует некоторому дизъюнкту.

Отличительной особенностью предлагаемых *smart*-таблиц *D*-типа по сравнению с обычными таблицами и *compressed*-таблицами является наличие составных атрибутов в заголовке матрицы (схеме отношения). Значениями компонент составных атрибутов могут быть отношения из предопределённого множества. В приведённой в качестве примера таблице имеется составной атрибут *YX*, в качестве компонент могут выступать все возможные отношения сравнения на паре атрибутов *Y* и *X*, а именно: $\{>, <, =, \neq, \leq, \geq\}$.

При организации рассуждений на данных структурах ключевым понятием является понятие кванта информации. Квант – это совокупность значений переменной, рассматриваемая как единое целое: все значения одного кванта элиминируются из домена или компоненты одновременно. Множество квантов для некоторого домена представляет собой разбиение соответствующего домена или компоненты. Процедуру квантования можно показать на примере 2. Домен переменной *YX* разбивается на три кванта. Первым квантом является отношение “$Y<X$”, вторым квантом – отношение “$Y=X$”, третьим – отношение “$Y>X$”. С помощью данной совокупности квантов может быть представлено любое из отношений $\{>, <, =, \neq, \leq, \geq\}$ для переменных (Y, X). Для простых атрибутов X, Y, Z с дискретными областями определения в качестве квантов можно взять элементы домена, то есть значения $\{1, 2, 3, 4\}$.

Результат квантования всей представленной выше матрицы можно записать с использованием секционированных булевых векторов:

$$\begin{array}{cccc} X & Y & Z & YX \\ 1,2,3,4. & 1,2,3,4. & 1,2,3,4. & <,=,> \\ \end{array}$$
$$\begin{bmatrix} 1\,0\,0\,1. & 0\,0\,0\,0. & 0\,0\,0\,1. & 1\,0\,0 \\ 1\,0\,0\,1. & 0\,0\,0\,0. & 0\,1\,0\,0. & 0\,1\,1 \\ 1\,0\,0\,1. & 0\,0\,0\,0. & 0\,0\,1\,0. & 0\,1\,0 \\ 0\,0\,1\,0. & 0\,0\,0\,0. & 0\,0\,0\,0. & 0\,0\,0 \\ 0\,0\,0\,0. & 0\,0\,1\,0. & 0\,0\,0\,0. & 0\,0\,0 \end{bmatrix}.$$

Во второй строке заголовка таблицы перечислены обозначения квантов. Данная матрица показывает как производится разложение каждой компоненты на кванты. Например, четвёртая компонента второй строки 011 соответствует отношению “≥”. Данный булев вектор показывает, что: “$Y\geq X$” = “$Y>X$” ∪ “$Y=X$”.

Любой метод решения задачи *CSP* содержит в себе две компоненты: компоненту, реализующую поиск, и компоненту, реализующую распространение ограничений (логический вывод на ограничениях). Как правило, в качестве компоненты, реализующей поиск, выступает один из методов поиска в глубину с возвратами. Вывод на ограничениях сводится к усечению областей определения переменных и характеризуется полиномиальной временной сложностью. Компонента, реализующая вывод, разрабатывается отдельно под каждый тип ограничений. В случае *smart*-таблиц *D*-типа, помимо редуцирования доменов переменных, в

результате вывода на ограничениях удаляются некоторые строки, столбцы, компоненты, значения компонент таблиц. Вывод на ограничениях, представленных в виде *smart*-таблиц *D*-типа, предлагается осуществлять с использованием следующих утверждений.

Утверждение 1 (**У1**). Если хотя бы одна строка *smart*-таблицы *D*-типа пуста (содержит все пустые компоненты), то таблица пуста (соответствующая система ограничений несовместна, задача *CSP* не имеет решения).

Утверждение 2 (**У2**). Если все компоненты некоторого атрибута пусты, то данный атрибут можно удалить из *smart*-таблицы *D*-типа (удаляются все компоненты, стоящие в соответствующем столбце, а пара “переменная-домен переменной” сохраняется в вектор частичного решения).

Утверждение 3 (**У3**). Если в *smart*-таблице *D*-типа есть строка (*smart*-кортеж), содержащая лишь одну непустую компоненту, то все кванты, не входящие в эту компоненту, удаляются из соответствующего домена.

Утверждение 4 (**У4**). Если строка *smart*-таблицы *D*-типа содержит хотя бы одну полную компоненту, то она удаляется (можно удалить соответствующее ограничение из системы ограничений).

Утверждение 5 (**У5**). Если компонента атрибута *smart*-таблицы *D*-типа содержит квант, не принадлежащий соответствующему домену, то этот квант удаляется из компоненты.

Утверждение 6 (**У6**). Если в *smart*-таблице *D*-типа усечён один или несколько доменов простых атрибутов, которые формируют некоторый составной атрибут, домен составного атрибута должен быть конкретизирован с учётом новых доменов простых атрибутов: из домена составного атрибута исключаются кванты, которые обращаются в пустое множество при новых доменах соответствующих простых атрибутов.

Утверждение 7 (**У7**). В случае конкретизации домена сложного атрибута (если таковая происходит в результате распространения ограничений) должны быть конкретизированы и домены соответствующих простых атрибутов с учётом вновь выведенного домена составного атрибута.

Поскольку две последних строки таблицы в нашем примере содержат по одной непустой компоненте каждая, то в процессе распространения ограничений, согласно **У3**, новые домены переменных *X* и *Y* становятся равными множеству {3}. Строки с номерами 4 и 5 удаляются из рассматриваемой таблицы после применения **У4**. Далее осуществляется “настройка” компонент первого столбца на новый домен переменной *X*. Согласно **У5**, из компонент первого столбца удаляются все значения, поскольку они не содержатся в новом домене. Таким образом, все компоненты первого столбца становятся пустыми, и столбец можно исключить из рассмотрения, пользуясь **У2**. По тем же причинам из рассмотрения исключается столбец *Y*. Дальнейшие вычисления связаны с особенностями обработки сложного атрибута *YX*: поскольку конкретизировались домены переменных *X* и *Y*, то согласно **У6** должна быть выполнена следующая проверка для сложного атрибута *YX*: требуется выяснить, какое из отношений-квантов {<, =, >} перестало выполняться после того, как были уточнены домены простых атрибутов. Ненарушенным оказывается только отношение-квант “*Y=X*”. Тогда домен сложного атрибута *YX* “сужается” до отношения “*Y=X*” и, пользуясь **У5**, можно исключить лишние значения (кванты) из компонент соответствующего столбца:

$$\begin{array}{cc} Z & YX \\ 1,2,3,4 & = \\ \left]\begin{array}{c} =4 \\ =2 \\ =3 \end{array}\right. & \left[\begin{array}{c} \varnothing \\ = \\ = \end{array}\right. \end{array}.$$

В данной таблице вторая строка заголовка содержит описание доменов переменных посредством квантов, оставшихся на данном шаге редукции.

Далее, по **У4** исключаются из рассмотрения строки 2 и 3. Теперь первая строка содержит только одну непустую компоненту (в атрибуте Z). Тогда, согласно **У3**, новый домен атрибута Z будет равен одноэлементому множеству {4} (переменная Z равна 4). Затем, согласно **У4**, вычеркивается из матрицы первая строка. Строк в матрице более не остаётся: они все были вычеркнуты без образования пустых строк (строк, состоящих лишь из пустых компонент). Это является признаком успешного окончания процесса редукции с получением решения поставленной задачи CSP. Окончательные значения атрибутов: X=3, Y=3, Z=4. В процессе вывода конкретизировались значения всех простых атрибутов.

3 Моделирование знаний с помощью *smart*-таблиц нового типа

Представление качественных зависимостей в форме *smart*-таблиц *C*-типа, как правило, интуитивно более понятно, чем представление тех же самых отношений с помощью *smart*-таблиц *D*-типа. Далее на конкретных примерах показано преимущество представления ряда многоместных отношений в форме *smart*-таблиц *D*-типа.

Пример 3. Ограничение *Permutation*(x_1, x_2, …, x_n), задающее множество перестановок из n элементов, где элементы принадлежат множеству $\{a_1, a_2, \ldots, a_n\}$, может быть представлено следующим образом:

$$\begin{array}{cccc} X_1 & X_2 & \ldots & X_n \\ \left]\begin{array}{c} = a_1 \\ = a_2 \\ \ldots \\ = a_n \end{array}\right. & \begin{array}{c} = a_1 \\ = a_2 \\ \ldots \\ = a_n \end{array} & \begin{array}{c} \ldots \\ \ldots \\ \ldots \\ \ldots \end{array} & \left.\begin{array}{c} = a_1 \\ = a_2 \\ \ldots \\ = a_n \end{array}\right[\end{array}.$$

Представление того же самого отношения в виде таблиц *C*-типа слишком громоздко и сводится к явному перечислению всех возможных перестановок.

Ограничение *Permutation*(x_1, x_2, …, x_n) можно рассматривать как частный случай глобального ограничения *Alldiff*(x_1, x_2, …, x_n), когда домены всех переменных равны одному и тому же множеству, и количество переменных совпадает с суммарным количеством возможных значений для этих переменных. Следует заметить, что и в общем случае ограничение *Alldiff*(x_1, x_2, …, x_n) также может быть успешно смоделировано с помощью предложенных smart-таблиц *D*-типа.

Пример 4. Глобальное ограничение *NotAllEqual*(x_1, ..., x_m): $\exists 1 \le i, j \le m : x_i \ne x_j$.

Данное ограничение может быть представлено с помощью *smart*-таблицы *C*-типа:

$$\begin{array}{cccc} X_2X_1 & X_3X_1 & \ldots & X_nX_1 \\ \left[\begin{array}{c} \ne \\ * \\ \ldots \\ * \end{array}\right. & \begin{array}{c} * \\ \ne \\ \ldots \\ * \end{array} & \begin{array}{c} \ldots \\ \ldots \\ \ldots \\ \ldots \end{array} & \left.\begin{array}{c} * \\ * \\ \ldots \\ \ne \end{array}\right] \end{array}.$$

Сходным образом данное ограничение представлено в работе [10]. Однако подобную формализацию ограничения *NotAllEqual*(x_1, ... , x_m) нельзя признать удовлетворительной, поскольку в данном случае табличное представление содержит много фиктивных компонент.

Данное ограничение можно представить в виде *smart*-таблицы *D*-типа, состоящее из единственной строки:

$$\begin{array}{cccc} X_2X_1 & X_3X_1 & \dots & X_nX_1 \\ ]\neq & \neq & \dots & \neq [. \end{array}$$

Далее приведены примеры предметных областей, где важно эффективно представлять и обрабатывать правила и логические выражения.

Пример 5. Задача кластерного анализа, где есть возможность учитывать знания о предметной области в форме пользовательских ограничений *Constrained Clustering* [17].

Пусть стоит задача разбить объекты $\{G_1, G_2, \dots, G_n\}$ на $k < n$ кластеров. В этом случае домены всех переменных равны множеству $\{1, \dots, k\}$. Пусть критерием кластеризации служит нахождение разбиения с минимальным диаметром, где диаметр разбиения – это максимальный диаметр для всех кластеров разбиения. Диаметр кластера – это максимальное расстояние между любыми двумя точками, принадлежащими данному кластеру.

При поиске разбиения с минимальным диаметром для каждого i и j формулируется ограничение: $(G_i = G_j) \rightarrow (D \geq d_{ij})$ [6]. Это выражение равносильно $(G_i \neq G_j) \vee (D \geq d_{ij})$, где d_{ij} – константа, показывающая расстояние между точками i и j, D – переменная, обозначающая диаметр разбиения и принимающая значения из множества $[d_{min}, d_{max}]$, причём d_{min} и d_{max} – минимальное и максимальные расстояния между парой объектов, принадлежащих множеству $\{G_1, G_2, \dots, G_n\}$.

Данное ограничение может быть выражено в виде следующей *smart*-таблицы *D*-типа:

$$\begin{array}{cc} G_iG_j & D \\ ]\neq & \geq d_{ij}[. \end{array}$$

Аналогичным образом может быть формализовано ограничение, которое задаётся в случае, если критерием кластеризации служит максимизация межкластерного расстояния (*split* - критерий): $(S > d_{ij}) \rightarrow (G_i = G_j)$ [17].

Здесь переменная S обозначает минимальное расстояние между точками двух различных кластеров, входящих в одно разбиение.

Пример 6. Задачи *Constraint-Based Scheduling,* рассмотренные в [18]. Переменные, используемые для описания задач:

$X(A_i, t)$ – булевы переменные, служащие для указания исполняется (значение 1) или нет (значение 0) задача (работа, задание) A_i в текущий момент времени t;

$start(A_i)$ и $end(A_i)$ – обозначают моменты начала и окончания работы A_i, изначальные домены переменных $start(A_i)$ и $end(A_i)$ равны соответственно $[r_i, lst_i]$ и $[eet_i, d_i]$, где lst_i and eet_i – наиболее позднее время начала и наиболее раннее время завершения задания A_i; $[r_i, d_i]$ – временное окно, в которое должна быть выполнена работа A_i.

В различных постановках задач *Constraint-Based Scheduling* используются различные виды качественных ограничений, которые могут быть представлены в форме *smart*-таблиц *D*-типа.

Например, ограничение *Time-Table Constraint*, представляющее собой условия на временное окно, в которое должно быть выполнено задание A_i, записано так:

$$[X(A_i, t) = 0] \wedge [t < eet_i] \rightarrow [start(A_i) > t].$$

$$[X(A_i, t) = 0] \wedge [lst_i \leq t] \rightarrow [end(A_i) \leq t].$$

Оно сводится к совокупности логических импликаций и может быть легко представлено с помощью *smart*-таблиц *D*-типа.

Ограничение *Disjunctive Constraint*, которое предписывает, что два задания A_i и A_j не могут перекрываться по времени: $[end(A_i) \leq start(A_j)] \vee [end(A_j) \leq start(A_i)]$, также представляет собой логическое выражение, легко моделируемое с помощью *smart*-таблиц *D*-типа.

Заключение

При решении задач, связанных с комбинаторным поиском, требуется компактно представлять знания в памяти компьютера, а также уметь их эффективно обрабатывать. Технология *Constraint Programming* нацелена на решение задач высокой вычислительной сложности. Вне зависимости от того, используются ли для решения задач удовлетворения ограничений методы систематического обхода дерева поиска, методы локального поиска или гибридные методы, основное внимание разработчиков всегда сосредоточено на механизмах логического вывода на ограничениях (алгоритмах распространения ограничений), обладающих низкой вычислительной сложностью и обеспечивающих редукцию пространства поиска за полиномиальное время. Понятийный аппарат, применяемый в рамках парадигмы *Constraint Programming*, близок к понятийному аппарату реляционной алгебры, а методы решения задач удовлетворения ограничений можно отнести к группе алгебраических методов логического вывода.

В статье предложен новый тип табличных ограничений, а именно *smart*-таблицы *D*-типа, а также приводятся правила эквивалентных преобразований данных таблиц, которые лежат в основе процедур редукции ограничений. За счёт новой интерпретации понятия «кортеж многоместного отношения» применение *smart*-таблиц *D*-типа позволяет в некоторых случаях существенно снизить потребление памяти компьютера по сравнению с известными типами табличных ограничений. В частности, *smart*-таблицы *D*-типа хорошо подходят для моделирования продукционных правил, некоторых типов логических выражений, а также некоторых типов глобальных ограничений.

Благодарности

Настоящее исследование выполнено при поддержке Российского фонда фундаментальных исследований (РФФИ), номера проектов 18-07-00615-а, 20-07-00708-а.

Список источников

[1] ***Кандрашина, Е.Ю.*** Представление знаний о времени и пространстве в интеллектуальных системах / Е.Ю. Кандрашина, Л.В. Литвинцева, Д.А. Поспелов. Под ред. Д.А. Поспелова. – М.: Наука, 1989. – 326 с.

[2] ***Осипов, Г.С.*** Методы искусственного интеллекта / Г.С. Осипов. – М.: Физматлит, 2011. – 296 с.

[3] ***Кузнецов, О.П.*** Онтологии в современных информационных системах: состояние, проблемы, перспективы / О.П. Кузнецов, В.С. Суховеров, Л.Б. Шипилина // Датчики и системы. – 2011. – №8 (147). – С.67-77.

[4] ***Maier, D.*** The Theory of Relational Databases. / D. Maier – Computer Science Press, 1983. 665 p.

[5] ***Bartak, R.*** Constraint Programming: In Pursuit of the Holy Grail / R. Bartak // Proceedings of the Week of Doctoral Students (WDS99), Part IV. – Prague: MatFyzPress, 1999. – P.555–564.

[6] ***Ruttkay, Zs.*** Constraint satisfaction a survey / Zs. Ruttkay // CWI Quarterly. – 1998. – Vol. 11. – P.163-214.

[7] ***Xia, W.*** Optimizing STR algorithms with tuple compression /W. Xia, R.H.C. Yap // CP 2013. LNCS, 8124. – 2013. – P.724-732.

[8] ***Perez, G.*** Improving GAC-4 for table and MDD constraints / G. Perez, J.C. Regin // CP 2014. LNCS, 8656. – 2014. – P.606-621.

[9] ***Katsirelos, G.*** A compression algorithm for large arity extensional constraints / G. Katsirelos, T. Walsh // CP 2007. LNCS, 4741, – 2007. – P.379-393.

[10] ***Mairy, J.*** The Smart Table Constraint / J. Mairy, Y. Deville, C. Lecoutre // In: Michel, L. (eds.) Integration of AI and OR Techniques in Constraint Programming. CPAIOR 2015. Lecture Notes in Computer Science. – Springer. Cham, – 2015. – Vol. 9075. – P.271-287.

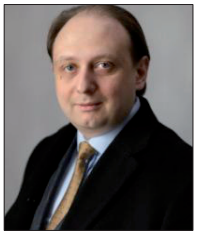
[11] ***Кузнецов, О.П.*** Дискретная математика для инженера / О.П. Кузнецов. – СПб.: Лань, 2009. – 400 с.

[12] ***Russel, S.*** Artificial Intelligence: A Modern Approach. 3rd edition / S. Russel, P. Norvig. – Prentice Hall, 2010. – 1132 P.

[13] ***Regin, J.*** A filtering algorithm for constraints of difference in CSPs / J. Regin // Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI–94). – Seattle: AAAI Press, – 1994. – P.362–367.

[14] ***Bessiere, C.*** Arc consistency for general constraint networks: preliminary results / C. Bessiere, J.C. Regin // Proceedings of IJCAI, – 1997. – P.398-404.
[15] ***Jefferson, C.*** Extending simple tabular reduction with short supports / C. Jefferson and P. Nightingale // Proceedings of IJCAI 2013, – 2013. – P.573-579.
[16] ***Zuenko, A.*** Development of n-tuple algebra for logical analysis of databases with the use of two-place predicates / A. Zuenko, A. Fridman // Journal of Computer and Systems Sciences International. – 2009. – Vol. 48(2). – P.254-261.
[17] ***Dao, T.B.H***. Constrained Clustering by Constraint Programming / T.B.H. Dao, K.C. Duong, C. Vrain // Artificial Intelligence Journal, – 2017. – Vol. 244. – P.70-94.
[18] ***Baptiste, P.*** Constraint-Based Scheduling: Applying Constraint Programming to Scheduling Problems / P. Baptiste, C. Le Pape, W. Nuijten. – Kluwer Academic Publishers, 2001. – 203 P.

Сведения об авторе



Зуенко Александр Анатольевич, 1983 г.р., к.т.н., ведущий научный сотрудник Института информатики и математического моделирования – обособленного подразделения ФИЦ «Кольский научный центр Российской академии наук». Области научных интересов: программирование в ограничениях; моделирование слабо формализованных предметных областей. Author ID (RSCI): 528493; Author ID (Scopus): 26536974000; Researcher ID (WoS): E-7944-2017. *zuenko@iimm.ru.*

Поступила в редакцию 11.11.2020, после рецензирования 07.12.2020. Принята к публикации 11.12.2020.

Compact representation of constraints based on a new interpretation of the concept "tuple of a multi-place relation"

A.A. Zuenko

Institute for Informatics and Mathematical Modeling – Subdivision of the Federal Research Centre "Kola Science Centre of the Russian Academy of Sciences (IIMM KSC RAS), Apapity, Russia

Abstract

The article discusses various points of view on the concept of "tuple of a multi-place relation" used in mathematics and information technology. Special attention is paid to an evolution of the concept “tuple” within Constraint Programming technology where the emergence of new interpretations for the concept “tuple” is related with attempts to design more “compact” table representation of qualitative relations comparing with typical relational tables. This “compact” representation can serve as a basis for accelerating qualitative constraint satisfaction procedures. In studied prototype-works, such varieties of table constraints as *compressed*-tables and *smart*-tables were proposed. In so doing, the concept of *compressed*- and *smart*- tuple substantially differs from traditional concept of tuple of a multi-place relation. However, the known table constraints types are not well suited for modeling and processing all types of quality relations, for example, there are inconveniences when modeling production rules. The article proposes a new type of table constraints – *smart-tables* of *D-type* and equivalent transformations rules allowing the initial constraint satisfaction problem to be effectively reduced. The application of the *smart*-tables of *D*-type allows in some cases to substantially reduce computer memory consumption comparing with application of the known types of table constraints. In particular, *smart*-tables of *D*-types are well suited for modeling production rules, some types of logical expirations and some types of global constraints.

Key words: *tuple of a multi-place relation, constraint programming technology, constraint satisfaction problem, table constraints, qualitative constraints.*

Citation: *Zuenko AA.* Compact representation of constraints based on a new interpretation of the concept "tuple of a multi-place relation" [In Russian]. *Ontology of designing*. 2020; 10(4): 503-515. DOI: 10.18287/2223-9537-2020-10-4-503-515.

Acknowledgment: This work was supported by the Russian Foundation for Basic Research (RFBR), project numbers 18-07-00615-a, 20-07-00708-a.

References

[1] ***Kandrashina EYu***. Knowledge representation about time and space in intellectual systems [In Russian]. Moscow: Nauka; 1989. 326 p.
[2] ***Osipov GS.*** Methods of artificial intelligence [In Russian]. Moscow: Fizmatlit; 2011. 296 p.
[3] ***Kuznetsov OP, Sukhoverov VS, Shipilina LB.*** Ontologies in modern information systems: state, problems, prospects [In Russian]. Sensors and systems. 2011; 8 (147): 67-77.
[4] ***Maier D.*** The Theory of Relational Databases. Computer Science Press; 1983. 665 p.
[5] ***Bartak R.*** Constraint Programming: In Pursuit of the Holy Grail. Proc. of the Week of Doctoral Students (WDS99), Part IV. Prague: MatFyzPress; 1999; 555–564.
[6] ***Ruttkay ZS***. Constraint satisfaction a survey. CWI Quarterly.1998; 11: 163–214.
[7] ***Xia W, Yap RHC.*** Optimizing STR algorithms with tuple compression. CP 2013. LNCS, 2013; 8124: 724-732.
[8] ***Perez G, Regin JC.*** Improving GAC-4 for table and MDD constraints. CP 2014. LNCS, 2014; 8656: 606-621.
[9] ***Katsirelos G, Walsh T.*** A compression algorithm for large arity extensional constraints. CP 2007, LNCS, 2007; 4741: 379-393.
[10] ***Mairy J.*** The Smart Table Constraint. In: Michel, L. (eds.) Integration of AI and OR Techniques in Constraint Programming. CPAIOR 2015. Lecture Notes in Computer Science. Springer. Cham, 2015; 9075: 271-287.
[11] ***Kuznetsov OP.*** Discrete mathematics for an engineer [In Russian]. SPb.: Lan, 2009; 400 p.
[12] ***Russel S, Norvig P***. Artificial Intelligence: A Modern Approach. 3rd ed. Prentice Hall, 2010. 1132 p.
[13] ***Regin J.*** A filtering algorithm for constraints of difference in CSPs. Proc. of the Twelfth National Conference on Artificial Intelligence (AAAI–94). Seattle: AAAI Press, 1994; 362-367.
[14] ***Bessiere C, Regin JC.*** Arc consistency for general constraint networks: preliminary results. Proc. of IJCAI, 1997: 398-404.
[15] ***Jefferson C, Nightingale P.*** Extending simple tabular reduction with short supports. Proc. of IJCAI 2013. 2013: 573-579.
[16] ***Zuenko A, Fridman A.*** Development of n-tuple algebra for logical analysis of databases with the use of two-place predicates. Journal of Computer and Systems Sciences International. 2009; 48(2): 254-261.
[17] ***Dao TBH, Duong KC, Vrain C.*** Constrained Clustering by Constraint Programming. Artificial Intelligence Journal, 2017; 244: 70-94.
[18] ***Baptiste P, Le Pape C, Nuijten W.*** Constraint-Based Scheduling: Applying Constraint Programming to Scheduling Problems. Kluwer Academic Publishers, 2001. 203 p.

About the author

Alexander Anatolievich Zuenko, (b. 1983) PhD, a leading researcher of Institute for Informatics and Mathematical Modeling – Subdivision of the Federal Research Centre "Kola Science Centre of the Russian Academy of Sciences (IIMM KSC RAS). Research interests: constraint programming; poorly formalized subject domains modeling. Author ID (RSCI): 528493; Author ID (Scopus): 26536974000; Researcher ID (WoS): E-7944-2017. *zuenko@iimm.ru*.

Received November 11, 2020. Revised December 07, 2020. Accepted December 11, 2020.

Примечание редакции для авторов

Статья А.А. Зуенко обнажила нерешённую проблему внедрения в научный оборот, в тексты научных статей заимствованных иностранных терминов, дословный перевод которых не отражает содержание понятий за ними стоящих. Так, в этой статье смешение слов, заимствованных из первоисточников на иностранном языке, с русскими словами приводит к грибоедовской проблеме «*французско-нижегородского языка*». При низкой частотности употребления таких словосочетаний язык интуитивно отторгает и отвергает такие новообразования. Редакция выражает надежду на бережное отношение авторов статей к языку научного общения.