УДК 681.3.057.51-7.311.17

© 2013 г. **А.С. Клещев**, д-р физ.-мат. наук,
В.А. Тимченко, канд. техн. наук
(Институт автоматики и процессов управления ДВО РАН, Владивосток)

ОГРАНИЧЕННАЯ СИНТАКСИЧЕСКАЯ ЭКВИВАЛЕНТНОСТЬ И АДАПТИВНАЯ УНИФИКАЦИЯ ВЫРАЖЕНИЙ В РАСШИРЯЕМОЙ МОДЕЛИ МАТЕМАТИЧЕСКОГО ДИАЛЕКТА*

Рассмотрена задача проверки на ограниченную синтаксическую эквивалентность математических выражений, а также задача адаптивной унификации для расширяемой модели математического диалекта и приведены алгоритмы их решения.

Ключевые слова: ограниченная синтаксическая эквивалентность, адаптивная унификация, математическое утверждение, пропозициональное утверждение, метаматематическое утверждение, предметная переменная, пропозициональная переменная, синтаксическая переменная, модифицированная синтаксическая переменная.

Введение

В системах доказательств теорем, основанных на формальных системах, приближенных к используемым в математической практике, язык представления математических знаний является, как правило, избыточным. Это означает, в частности, что:

одно и то же (по смыслу) утверждение может быть представлено различными способами (т.е. утверждения могут синтаксически не совпадать, но быть эквивалентными);

утверждения могут содержать коммутативные и ассоциативные операции.

В процессе построения доказательства в таких системах должны решаться следующие задачи:

1) предотвращение повторного доказательства эквивалентных между собой математических утверждений;

2) проверка достижения условия окончания процесса вывода (что на очередном шаге вывода получено математическое утверждение, эквивалентное доказываемому).

---

* Работа выполнена при финансовой поддержке РФФИ, проект 11-07-00460-а, и ДВО РАН, проект 12-III-А-01И-018, проект 12-I-П15-03.

Аналогичные задачи необходимо решать в расширяемой модели математического диалекта [1], где математические знания представляются с помощью трех языков – ограниченного языка прикладной логики (языка математических утверждений), а также языка пропозициональной логики и метаязыка (для представления правил вывода). Кроме того, проверка унифицируемости двух утверждений [2] в данной модели сводится к выяснению существования математического утверждения, эквивалентного частному и унифицируемого с общим.

Стоит отметить, что возможность обращаться к функциональности систем компьютерной математики (СКМ) в процессе построения доказательств теорем для решения задач, требующих большой вычислительной сложности, представляется авторам весьма перспективной. Однако в настоящее время задачи проверки двух выражений на эквивалентность и унифицируемость с помощью современных широко распространенных СКМ [3 – 5] непосредственно решены быть не могут.

Целью настоящей работы, таким образом, является разработка алгоритма проверки на эквивалентность двух математических утверждений (в том числе при унификации) для расширяемой модели математического диалекта.

Постановка задачи

В общем виде математическое выражение можно записать следующим образом: $E = (v_1\text{: } t_1(v_1, \ldots, v_n)) \ldots (v_n\text{: } t_n(v_1, \ldots, v_n))\ s(v_1, \ldots, v_n)$, где $s(v_1, \ldots, v_n)$ – математическая формула (f) или математический терм (t), содержащая (-ий) вхождения предметных переменных $v_1, \ldots, v_n$; $(v_i\text{: } t_i(v_1, \ldots, v_n))$ $(i = 1, \ldots, n)$ – описание предметной переменной v_i; $t_i(v_1, \ldots, v_n)$ – математический терм, значением которого является множество (область возможных значений переменной v_i).

Обозначим одно математическое выражение через $E_m = (v_{11}\text{: } t_{11}(v_{11}, \ldots, v_{1m})) \ldots (v_{1m}\text{: } t_{1m}(v_{11}, \ldots, v_{1m}))\ s_m(v_{11}, \ldots, v_{1m})$, а второе через $E_n = (v_{21}\text{: } t_{21}(v_{21}, \ldots, v_{2n})) \ldots (v_{2n}\text{: } t_{2n}(v_{21}, \ldots, v_{2n}))\ s_n(v_{21}, \ldots, v_{2n})$. Напомним определение (семантической) эквивалентности математических выражений.

Определение 1. Два математических выражения являются *семантически эквивалентными* ($E_m \sim E_n$), если выполняются следующие условия:

1) существует взаимно-однозначное соответствие между множеством входящих в s_m переменных – υ_m и подмножеством (возможно несобственным) υ_n' множества входящих в s_n переменных – υ_n, такое, что значения s_m и s_n одинаковы при любых подстановках значений вместо соответствующих переменных, входящих в υ_m и υ_n', из областей их возможных значений и произвольных значениях переменных, входящих в υ_n'', из областей их возможных значений ($\upsilon_n = \upsilon_n' \cup \upsilon_n''$, $\upsilon_n' \cap \upsilon_n'' = \varnothing$);

2) для всех пар термов, представляющих области возможных значений переменных, между которыми установлено взаимно-однозначное соответствие, справедлив п.1 при том же соответствии между переменными.

Поскольку определение семантической эквивалентности не является конструктивным, введем понятие синтаксической эквивалентности.

Определение 2. Два математических выражения будем называть *синтаксически эквивалентными* ($E_m \sim_s E_n$), если выполняются следующие условия:

1) существует конечная последовательность $s_m \Rightarrow \ldots \Rightarrow s_i \Rightarrow s_{i+1} \Rightarrow \ldots \Rightarrow s_n$, в которой любые s_i и s_{i+1} (i = m, …, n – 1) связаны некоторым равносильным преобразованием;

2) для всех пар термов, представляющих области возможных значений переменных, между которыми установлено взаимно-однозначное соответствие, справедлив п.1 при том же соответствии между переменными.

Допустимые равносильные преобразования определяются содержащимися в базе знаний утверждениями в форме равенства и равносильности. Следовательно, достоверность установления синтаксической эквивалентности двух выражений зависит от содержимого используемой в системах доказательств теорем базы знаний. Синтаксическая эквивалентность может быть приближена к семантической за счет расширения базы знаний.

Последовательность $s_m \Rightarrow \ldots \Rightarrow s_n$ при этом может быть сколь угодно большой длины, что на практике может приводить к большой вычислительной сложности, а следовательно, неприемлемому времени работы систем доказательств теорем. Введем поэтому понятие *ограниченной синтаксической эквивалентности*.

Определение 3. Два математических выражения будем называть *ограниченно синтаксически эквивалентными* ($E_m \sim_{s'} E_n$), если выполняются следующие условия:

1) s_m *сводимо* к s_m', совпадающему с s_n с точностью до обозначений переменных ($s_m \hookrightarrow s_m'$);

2) существует взаимно-однозначное соответствие между переменными, входящими в s_m', и переменными, входящими в s_n;

3) термы, представляющие области возможных значений переменных, входящих в s_m', должны быть *сводимы* к термам, представляющим области возможных значений переменных, входящих в s_n, при том же установленном соответствии между переменными s_m' и s_n.

Определим отношение *сводимости* ($\hookrightarrow$) с точностью до обозначений переменных на множестве пар формул (термов).

Определение 4.

1) $f(\mathrm{f}_1, \ldots, \mathrm{f}_k, \mathrm{t}_1, \ldots, \mathrm{t}_p) \hookrightarrow f(\mathrm{f}_1', \ldots, \mathrm{f}_q', \mathrm{t}_1', \ldots, \mathrm{t}_s')$, где f – формула, $\mathrm{f}_1, \ldots, \mathrm{f}_k, \mathrm{f}_1', \ldots, \mathrm{f}_q'$ – формулы, непосредственно входящие в f, $\mathrm{t}_1, \ldots, \mathrm{t}_p, \mathrm{t}_1', \ldots, \mathrm{t}_s'$ – термы, непосредственно входящие в f, если между формулами (термами), непосредственно входящими в f, существует такое взаимно-однозначное соответствие, при котором соответствующая пара формул (термов) находится между собой в отношении *сводимости*;

2) $t(\mathrm{f}_1, \ldots, \mathrm{f}_k, \mathrm{t}_1, \ldots, \mathrm{t}_p) \hookrightarrow t(\mathrm{f}_1', \ldots, \mathrm{f}_q', \mathrm{t}_1', \ldots, \mathrm{t}_s')$, где t – терм, $\mathrm{f}_1, \ldots, \mathrm{f}_k, \mathrm{f}_1', \ldots, \mathrm{f}_q'$ – формулы, непосредственно входящие в t, $\mathrm{t}_1, \ldots, \mathrm{t}_p, \mathrm{t}_1', \ldots, \mathrm{t}_s'$ – термы, непосредственно входящие в t, если между формулами (термами), непосредственно входящими в t, существует такое взаимно-однозначное соответствие, при котором

соответствующая пара формул (термов) находится между собой в отношении *сводимости*;

3) $f_m(f_1, \ldots, f_k) \hookrightarrow f_n(f_1', \ldots, f_q')$, где f_m и f_n – формулы, $f_1, \ldots, f_k$ – формулы, непосредственно входящие в f_m, $f_1', \ldots, f_q'$ – формулы, непосредственно входящие в f_n, если в базе знаний существует равносильность $pf_1(v_1, \ldots, v_p) \Leftrightarrow pf_2(v_1, \ldots, v_s)$, где pf_1 – пропозициональная формула, содержащая вхождения пропозициональных переменных $v_1, \ldots v_p$, pf_2 – пропозициональная формула, содержащая вхождения пропозициональных переменных $v_1, \ldots v_s$; такая что (без ограничения общности) f_m унифицируема с pf_1 [2], а между формулами, непосредственно входящими в формулу, являющуюся результатом применения сформированной подстановки – $pf_2\theta$ [6], и формулами, непосредственно входящими в f_n, существует такое взаимно-однозначное соответствие, при котором соответствующая пара формул находится между собой в отношении *сводимости*;

4) $f_m(t_1, \ldots, t_p) \hookrightarrow f_n(t_1', \ldots, t_s')$, где f_m и f_n – формулы, $t_1, \ldots, t_p$ – термы, непосредственно входящие в f_m, $t_1', \ldots, t_s'$ – термы, непосредственно входящие в f_n, если в базе знаний существует равносильность $f_1(v_1, \ldots, v_k) \Leftrightarrow f_2(v_1, \ldots, v_q)$, где f_1 – математическая формула, содержащая вхождения предметных переменных $v_1, \ldots v_k$, f_2 – математическая формула, содержащая вхождения предметных переменных $v_1, \ldots v_q$; такая, что (без ограничения общности) f_m унифицируема с f_1, а между термами, непосредственно входящими в формулу, являющуюся результатом применения сформированной подстановки – $f_2\theta$, и термами, непосредственно входящими в f_n, существует такое взаимно-однозначное соответствие, при котором соответствующая пара термов находится между собой в отношении *сводимости*;

5) $f_m(f_1, \ldots, f_k, t_1, \ldots, t_p) \hookrightarrow f_n(f_1', \ldots, f_q', t_1', \ldots, t_s')$, где f_m и f_n – формулы, $f_1, \ldots, f_k$ – формулы, непосредственно входящие в f_m, $t_1, \ldots, t_p$ – термы, непосредственно входящие в f_m ($k \geq 0$, $p \geq 0$, $k + p > 0$), $f_1', \ldots, f_q'$ – формулы, непосредственно входящие в f_n, $t_1', \ldots, t_s'$ – термы, непосредственно входящие в f_n, если в базе знаний существует равносильность $MF_1(v_1, \ldots, v_{i1}; \mathbf{t_1}, \ldots, \mathbf{t_{k1}}; \mathbf{f_1}, \ldots, \mathbf{f_{s1}}; \mathbf{i_1}, \ldots, \mathbf{i_{p1}}; \mathbf{r_1}, \ldots, \mathbf{r_{q1}}) \Leftrightarrow MF_2(v_1, \ldots, v_{i2}; \mathbf{t_1}, \ldots, \mathbf{t_{k2}}; \mathbf{f_1}, \ldots, \mathbf{f_{s2}}; \mathbf{i_1}, \ldots, \mathbf{i_{p2}}; \mathbf{r_1}, \ldots, \mathbf{r_{q2}})$, где MF_1 – метаматематическая формула, содержащая вхождения предметных переменных $v_1, \ldots, v_{i1}$, а также вхождения синтаксических переменных $\mathbf{t_1}, \ldots, \mathbf{t_{k1}}$ типа **t**, синтаксических переменных $\mathbf{f_1}, \ldots, \mathbf{f_{s1}}$ типа **f**, синтаксических переменных $\mathbf{i_1}, \ldots, \mathbf{i_{p1}}$ типа **i** и синтаксических переменных $\mathbf{r_1}, \ldots, \mathbf{r_{q1}}$ типа **r**, MF_2 – метаматематическая формула, содержащая вхождения предметных переменных $v_1, \ldots, v_{i2}$, а также вхождения синтаксических переменных $\mathbf{t_1}, \ldots, \mathbf{t_{k2}}$ типа **t**, синтаксических переменных $\mathbf{f_1}, \ldots, \mathbf{f_{s2}}$ типа **f**, синтаксических переменных $\mathbf{i_1}, \ldots, \mathbf{i_{p2}}$ типа **i** и синтаксических переменных $\mathbf{r_1}, \ldots, \mathbf{r_{q2}}$ типа **r**; такая, что (без ограничения общности) f_m унифицируема с MF_1, а между формулами (термами), непосредственно входящими в формулу, являющуюся результатом применения сформированной подстановки – $MF_2\theta$, и формулами (термами), непосредственно входящими в f_n, существует такое взаимно-однозначное соответствие, при котором соответствующая пара формул (термов) находится между собой в отношении *сводимости*;

6) $t_m(t_1, \ldots, t_p) \hookrightarrow t_n(t_1', \ldots, t_s')$, где t_m и t_n – термы, $t_1, \ldots, t_p$ – термы, непосредственно входящие в t_m, $t_1', \ldots, t_s'$ – термы, непосредственно входящие в t_n, если в базе знаний существует равенство $t_1(v_1, \ldots, v_k) = t_2(v_1, \ldots, v_q)$, где t_1 – математический терм, содержащий вхождения предметных переменных $v_1, \ldots v_k$, t_2 – математический терм, содержащий вхождения предметных переменных $v_1, \ldots v_q$; такое, что (без ограничения общности) t_m унифицируем с t_1, а между термами, непосредственно входящими в терм, являющийся результатом применения сформированной подстановки – $t_2\theta$, и термами, непосредственно входящими в t_n, существует такое взаимно-однозначное соответствие, при котором соответствующая пара термов находится между собой в отношении *сводимости*;

7) $t_m(f_1, \ldots, f_k, t_1, \ldots, t_p) \hookrightarrow t_n(f_1', \ldots, f_q', t_1', \ldots, t_s')$, где t_m и t_n – термы, $f_1, \ldots, f_k$ – формулы, непосредственно входящие в t_m, $t_1, \ldots, t_p$ – термы, непосредственно входящие в t_m ($k \geq 0$, $p \geq 0$, $k + p > 0$), $f_1', \ldots, f_q'$ – формулы, непосредственно входящие в t_n, $t_1', \ldots, t_s'$ – термы, непосредственно входящие в t_n, если в базе знаний существует равенство $MT_1(v_1, \ldots, v_{i1}; \mathbf{t_1}, \ldots, \mathbf{t_{k1}}; \mathbf{f_1}, \ldots, \mathbf{f_{s1}}; \mathbf{i_1}, \ldots, \mathbf{i_{p1}}; \mathbf{r_1}, \ldots, \mathbf{r_{q1}}) = MT_2(v_1, \ldots, v_{i2}; \mathbf{t_1}, \ldots, \mathbf{t_{k2}}; \mathbf{f_1}, \ldots, \mathbf{f_{s2}}; \mathbf{i_1}, \ldots, \mathbf{i_{p2}}; \mathbf{r_1}, \ldots, \mathbf{r_{q2}})$, где MT_1 – метаматематический терм, содержащий вхождения предметных переменных $v_1, \ldots, v_{i1}$, а также вхождения синтаксических переменных $\mathbf{t_1}, \ldots, \mathbf{t_{k1}}$ типа **t**, синтаксических переменных $\mathbf{f_1}, \ldots, \mathbf{f_{s1}}$ типа **f**, синтаксических переменных $\mathbf{i_1}, \ldots, \mathbf{i_{p1}}$ типа **i** и синтаксических переменных $\mathbf{r_1}, \ldots, \mathbf{r_{q1}}$ типа **r**, MT_2 – метаматематический терм, содержащий вхождения предметных переменных $v_1, \ldots, v_{i2}$, а также вхождения синтаксических переменных $\mathbf{t_1}, \ldots, \mathbf{t_{k2}}$ типа **t**, синтаксических переменных $\mathbf{f_1}, \ldots, \mathbf{f_{s2}}$ типа **f**, синтаксических переменных $\mathbf{i_1}, \ldots, \mathbf{i_{p2}}$ типа **i** и синтаксических переменных $\mathbf{r_1}, \ldots, \mathbf{r_{q2}}$ типа **r**; такое, что (без ограничения общности) t_m унифицируем с MT_1, а между формулами (термами), непосредственно входящими в терм, являющийся результатом применения сформированной подстановки – $MT_2\theta$, и формулами (термами), непосредственно входящими в t_n, существует такое взаимно-однозначное соответствие, при котором соответствующая пара формул (термов) находится между собой в отношении *сводимости*;

8) переменная v считается сводимой к любой другой переменной v' ($v \hookrightarrow v'$);

9) константа c сводима только к самой себе ($c \hookrightarrow c$).

Отметим принципиальное отличие между синтаксической эквивалентностью и ограниченной синтаксической эквивалентностью, состоящее в том, что во втором случае для очередного, полученного в результате предыдущего шага преобразования выражения, следующий шаг преобразования может применяться только к его аргументам-подвыражениям (но не к нему самому). Ограниченная синтаксическая эквивалентность за счет расширения базы знаний, в свою очередь, может быть приближена к синтаксической эквивалентности.

Прежде чем ввести следующее определение, необходимо упомянуть, что в работе [2] рассматривается *простая* унификация выражений. Это означает, что если на некотором этапе выясняется, что частное выражение не унифицируемо с общим, то не предпринимается попыток применить к частному выражению эквивалентные преобразования на основе утверждений в форме равенства или равносильности из базы знаний, чтобы изменить отношение между ними. Очевидно, что если в систему заложена возможность выполнять только простую унификацию, то существенно снижается удобство ее использования и увеличивается нагрузка на пользователя: ему необходимо отвлекаться на доказательство тривиальных промежуточных утверждений. Поэтому введем далее понятие *адаптивной унификации*.

*Определение 5***.** Два выражения – частное математическое E_m и общее E_n (математическое, пропозициональное или метаматематическое) – являются *адаптивно унифицируемыми*, если существует $s_m{}'$ такое, что $s_m \hookrightarrow s_m{}'$, унифицируемое с s_n, т.е. существует подстановка θ такая, что результат ее применения к E_n совпадает с утверждением $E_m{}' = (\mathrm{v}_1\colon \mathrm{t}_1(\mathrm{v}_1, \ldots, \mathrm{v}_{\mathrm{m}'})) \ldots (\mathrm{v}_{\mathrm{m}'}\colon \mathrm{t}_{\mathrm{m}'}(\mathrm{v}_1, \ldots, \mathrm{v}_{\mathrm{m}'}))\ \mathrm{s}_{\mathrm{m}}{}'(\mathrm{v}_1, \ldots, \mathrm{v}_{\mathrm{m}'})$ $(E_m{}' = E_n\theta)$.

Таким образом, для двух выражений E_m и E_n в расширяемой модели математического диалекта можно сформулировать следующие две задачи. Необходимо выяснить, являются ли исследуемые выражения E_m и E_n:

1) *ограниченно синтаксически эквивалентными* ($E_m \sim_{\mathrm{s}'} E_n$);

2) *адаптивно унифицируемыми*, т.е. найти требуемую подстановку θ или убедиться, что такой подстановки нет.

Будем считать, что множества переменных, входящих в E_m и E_n, не пересекаются, а константные выражения (т.е. выражения, в которые входят только константные значения) уже заменены их вычисленными значениями.

Алгоритм решения поставленных задач

Из постановок этих двух задач можно заметить, что их решение подразумевает выполнение достаточно большого количества одинаковых (общих) действий; поэтому опишем общий алгоритм решения поставленных задач, обозначая те места, где действия должны отличаться в зависимости от конкретной решаемой задачи. Далее, если выполняется проверка на *ограниченную синтаксическую эквивалентность* двух математических выражений E_m и E_n, то для краткости будем говорить, что решается *задача 1*; если же выясняется, являются ли частное математическое выражение E_m и общее выражение E_n *адаптивно унифицируемыми*, – решается *задача 2*.

Введем информационную структуру – *таблицу соответствий* (σ). Она состоит из двух столбцов и устанавливает либо взаимно однозначное соответствие между переменными, входящими в выражения s_m ($s_m{}'$) и s_n (если решается *задача 1*); либо соответствие между переменными, входящими в общее выражение s_n, и формулами или термами, входящими в частное выражение s_m ($s_m{}'$) (если решается *задача 2*, т.е. в этом случае данная таблица фактически представляет собой *подстановку*). Каждая строка таблицы в первом столбце содержит имя переменной (v_{ni}) из s_n, а во втором столбце – либо имя переменной (v_{mj}), которая встретилась в s_m ($s_m{}'$), когда в s_n встретилась переменная v_{ni} при их одновременном просмотре (если решается *задача 1*), либо формулу (f_{mj}) или терм (t_{mj}), встретившийся в s_m ($s_m{}'$), когда в s_n встретилась переменная v_{ni} при их одновременном просмотре (если решается *задача 2*; в этом случае переход к подформулам (подтермам) f_{mj}

или t_{mj} уже не осуществляется).

Отметим, что при решении *задачи 1* взаимно однозначное соответствие между переменными, входящими в s_m', и переменными, входящими в s_n, устанавливается через таблицу соответствий σ в процессе проверки сводимости s_m к s_m', совпадающему с s_n с точностью до обозначений переменных. Будем считать, что выражения E_m и E_n представлены своими деревьями грамматического разбора, которые в силу корректной расстановки скобок в текстовом представлении E_m и E_n могут быть построены однозначно.

Будем считать также, что всем вхождениям подвыражений в выражениях s_m и s_n сопоставлены их позиционные номера. Принцип назначения вхождению некоторого подвыражения (вершина в дереве выражения) позиционного номера следующий. Корневой вершине выражения s назначается номер 1. Префикс номеров у непосредственных вершин-потомков совпадает с номером вершины-предка, а в качестве суффикса добавляется порядковый номер вхождения вершины-потомка в вершину-предок. По существу позиционный номер, сопоставленный некоторой вершине-подвыражению, однозначно определяет путь к ней от корневой вершины; верно и обратное. В дальнейшем существенно, что на позиционных номерах, имеющих такой вид, естественным образом определяется лексикографический порядок.

Опишем сначала вспомогательную процедуру "*корректировка таблицы соответствий*".

<u>*Процедура*</u> ***"корректировка таблицы соответствий"*** (σ).

Данная процедура вызывается, когда необходимо отказаться от некоторой конструкции (s*) в s_m (s_m') и заменить ее другой допустимой конструкцией (если такая имеется). При этом должны быть произведены следующие действия.

При решении *задачи 1*. Из σ должны быть удалены переменные, входящие в s* (если такие есть в σ). Решение принимается на основе позиционных номеров: позиционные номера вершин, представляющих переменные, входящие в s*, больше позиционного номера корневой вершины s*.

При решении *задачи 2*. Из σ должны быть удалены переменные, входящие в конструкцию выражения s_n (если такие есть в σ), которая является текущей тогда, когда в s_m (s_m') текущей является конструкция s*.

Вход:

p_vertex – вершина, представляющая текущую конструкцию в структурном представлении s_m (s_m') (при решении *задачи 1*) или s_n (при решении *задачи 2*).

Описание процедуры.

<u>Начало</u>

В <u>цикле</u> по элементам σ выполняются следующие действия.

При решении *задачи 1*. Если позиционный номер вершины *p_vertex* меньше (лексикографически предшествует) позиционного номера переменной v_{mj} (входящей в s_m (s_m')) для текущего элемента – пары $<v_{ni}, v_{mj}>$, то $<v_{ni}, v_{mj}>$ удаляется из σ.

При решении *задачи 2*. Если позиционный номер вершины *p_vertex* меньше (лексикографически предшествует) позиционного номера переменной v_{ni} (входящей в s_n) для текущего элемента – пары $<v_{ni}, f_{mj}(t_{mj})>$, то $<v_{ni}, f_{mj}(t_{mj})>$ удаляется из σ.

Конец (процедуры *корректировка таблицы соответствий*).

Далее в виде рекурсивной процедуры "*проверка_сводимости*" приводится алгоритм, позволяющий найти выражение s_m' такое, что:

(1) $s_m \hookrightarrow s_m'$, и

(2) s_m' совпадает с s_n с точностью до обозначений переменных (если решается *задача 1*) или унифицируется с s_n (если решается *задача 2*),

либо показать, что такого выражения нет.

Процедура ***"проверка_сводимости".***

В основе данной процедуры лежит *определение 4.* При этом в процессе рекурсивного обхода в глубину одновременно выражений s_m и s_n, начиная с их корневых вершин:

(1) s_m может трансформироваться; трансформируемое выражение будем обозначать s_m' (изначально s_m' совпадает с s_m, далее это обозначение сохраняется для всех последующих этапов трансформации);

(2) случаи с коммутативными и ассоциативно-коммутативными операциями, входящими в выражения, обрабатываются отдельно для того, чтобы не хранить избыточное количество утверждений в форме равенства и равносильности в базе знаний (считается, что в языке представления математических знаний выделены все коммутативные и ассоциативно-коммутативные операции).

Вход:

$vertex_n$ – вершина, представляющая текущую конструкцию в структурном представлении s_n (в первом вызове это корневая вершина);

$vertex_m$ – вершина, представляющая текущую конструкцию в структурном представлении s_m' (в первом вызове это корневая вершина).

Выход:

result – признак, имеющий значение *false*, если на некотором этапе фиксируется, что s_m' не является искомым утверждением, и *true* – в противном случае.

Описание процедуры.

Начало

При рекурсивном обходе возможны следующие случаи:

1. Тип $vertex_n$ *совпадает* с типом $vertex_m$. Возможны следующие подслучаи:

1.1. $vertex_n$ представляет *константу – c* и $vertex_m$ представляет *константу – c**. В этом случае, если $c \neq c^*$, то *result = false*, иначе выйти из *процедуры*, вернув *true*.

1.2. $vertex_n$ представляет *переменную – v* и $vertex_m$ представляет *переменную – v**. В этом случае:

при решении *задачи 1*. Если v отсутствует в σ, то в σ добавляется пара $v \mid v^*$ (выйти из *процедуры*, вернув *true*); иначе, если сопоставленная v переменная v' не совпадает с переменной v^*, то *result = false*, иначе выйти из *процедуры*, вернув *true*;

при решении *задачи 2*. Если v отсутствует в σ (или в *таблице унификации*, если v является *модифицированной синтаксической переменной* (*МСП*)), то выполняются действия, описанные в работе [2] для аналогичного случая (выйти из <u>*процедуры*</u>, вернув *true*). Если сопоставленная *v* (или ее вхождению, если *v* является *МСП*) переменная v' не совпадает с переменной v^*, то *result* = *false*, иначе выйти из <u>*процедуры*</u>, вернув *true*.

1.3. $vertex_n$ и $vertex_m$ представляют собой одну и ту же операцию, не принадлежащую к классу *коммутативных* или *ассоциативно-коммутативных* операций. В этом случае:

<u>*процедура*</u> ***"проверка_сводимости"*** вызывается для каждой пары потомков-аргументов вершин $vertex_n$ и $vertex_m$ соответственно, у которых совпадают порядковые номера. Если при этом некоторая пара потомков-аргументов не сопоставлена или вызов процедуры для некоторой пары дал отрицательный результат, то *result* = *false*, иначе (т.е. все вызовы увенчались успехом) выйти из <u>*процедуры*</u>, вернув *true*.

1.4. $vertex_n$ и $vertex_m$ представляют собой одну и ту же *коммутативную* операцию (в данном случае у вершин $vertex_n$ и $vertex_m$ всегда два потомка-аргумента).

1.4.1. Вызывается <u>*процедура*</u> ***"проверка_сводимости"*** для прямого порядка потомков-аргументов вершин $vertex_n$ и $vertex_m$ соответственно. Если данный вариант увенчался успехом, то выйти из <u>*процедуры*</u>, вернув *true*. Иначе:

1.4.2. Вызывается <u>*процедура*</u> ***"корректировка таблицы соответствий"*** для $vertex_m$ (при решении *задачи 1*) или $vertex_n$ (при решении *задачи 2*). Вызывается <u>*процедура*</u> ***"проверка_сводимости"*** для прямого порядка потомков-аргументов $vertex_n$ и обратного порядка потомков-аргументов $vertex_m$. Если данный вариант увенчался успехом, то выйти из <u>*процедуры*</u>, вернув *true*, иначе *result* = *false*.

1.5. $vertex_n$ и $vertex_m$ представляют собой одну и ту же *ассоциативно-коммутативную* операцию. В этом случае:

1.5.1. Если количество потомков-аргументов у $vertex_n$ и $vertex_m$ не совпадает, то выйти из <u>*процедуры*</u>, вернув *false*.

1.5.2. Порядок расположения для потомков-аргументов вершины $vertex_n$ фиксируется, а для $vertex_m$ рассматриваются в общем случае все возможные перестановки ее потомков-аргументов.

<u>Цикл</u> по всем перестановкам потомков-аргументов.

<u>Начало</u>

1.5.2.1. Вызывается <u>*процедура*</u> ***"проверка_сводимости"*** для каждой пары потомков-аргументов вершин $vertex_n$ и $vertex_m$ соответственно, у которых совпадают порядковые номера.

1.5.2.2. Если результаты вызова для всех пар оказались успешными, то выйти из <u>*процедуры*</u>, вернув *true*. Иначе (т.е. вызов процедуры для некоторой пары дал отрицательный результат):

1.5.2.3. Вызывается <u>*процедура*</u> ***"корректировка таблицы соответствий"*** для $vertex_m$ (при решении *задачи 1*) или $vertex_n$ (при решении *задачи 2*) и берется следующая перестановка (если уже все перебраны, то *result* = *false*).

<u>Конец</u> (цикла по перестановкам потомков-аргументов)

2. Тип $vertex_n$ *не совпадает* с типом $vertex_m$ <u>или</u> результат рассмотрения

подслучаев п.1 является отрицательным (*result* = *false*).

2.1. Отдельно рассматривается случай, когда $vertex_n$ представляет *переменную* и при этом решается *задача 2*. Возможны два подслучая:

2.1.1. *v* отсутствует в σ (или в *таблице унификации*, если *v* является *МСП*). В этом случае выполняются действия, описанные в [2] для аналогичного случая.

2.1.2 *v* содержится в σ (или в *таблице унификации*, если *v* является *МСП*). В этом случае решается задача проверки сводимости выражения *value** к выражению *value′* с точностью до обозначений переменных, где *value′* – значение, сопоставленное *v* в σ (или вхождению *v* в *таблице унификации*, если *v* является *МСП*), *value** – текущая конструкция в s_m', представленная вершиной $vertex_m$.

result = вызов процедуры ***"проверка_сводимости"*** для *value** и *value′*. Если *result* = *true*, то выйти из процедуры, вернув *true*.

2.2. Во всех остальных случаях несоответствия типов вершин или когда типы вершин соответствуют, но *result* = *false* в базе знаний (БЗ) ищутся утверждения в форме равносильности – $F_1 \Leftrightarrow F_2$ (если $vertex_n$ является формулой) или равенства – $T_1 = T_2$ (если $vertex_n$ является термом) такие, что утверждение, представленное вершиной $vertex_m$ (без ограничения общности), унифицируемо с $F_1(T_1)$ (здесь выполняется *простая* унификация).

2.2.2. Если таких утверждений не нашлось, то выйти из процедуры, вернув *false*. Иначе:

2.2.3. Цикл по всем найденным утверждениям из БЗ.

Начало

2.2.3.1. К $F_2(T_2)$ применяется сформированная в п. 2.2 подстановка θ.

2.2.3.2. Вызывается процедура ***"корректировка таблицы соответствий"*** для $vertex_m$ (при решении *задачи 1*) или $vertex_n$ (при решении *задачи 2*).

2.2.3.3. В s_m', конструкция, представленная вершиной $vertex_m$, заменяется конструкцией, представленной вершиной $vertex_n'$, – корневой вершиной выражения (формулы или терма), являющего результатом применения подстановки – $F_2(T_2)\theta$ (тип $vertex_n'$ совпадает с типом $vertex_n$).

2.2.3.4. *result* = вызов процедуры ***"проверка_сводимости"*** для $vertex_n$ и $vertex_n'$. Если *result* = *true*, то выйти из процедуры, вернув *true*, иначе берется следующее утверждение (если уже все перебраны, то выйти из процедуры, вернув *false*).

Конец (цикла по утверждениям из БЗ).

Конец (процедуры "***проверка_сводимости"***).

Если в результате выполнения процедуры ***"проверка сводимости"*** требуемое выражение s_m' найдено, то выполняются следующие действия.

При решении *задачи 1*. Решается множество задач по проверке отношения сводимости между термами, представляющими области возможных значений переменных, входящих в s_m', и термами, представляющими области возможных значений переменных, входящих в s_n, при уже установленном в σ соответствии между переменными (σ при решении этих задач не модифицируется). Если некоторая пара термов не находится в отношении сводимости, то исходные выражения E_m и E_n не являются ограниченно синтаксически эквивалентными, в противном случае $E_m \sim_{s'} E_n$.

При решении *задачи 2*. Выполняется обработка *таблицы унификации* аналогично случаю простой унификации [2] и в σ добавляются *модифицированные синтаксические переменные* вместе с их значениями.

Пример 1 (на ограниченную синтаксическую эквивалентность). Пусть в результате вывода необходимо получить математическое выражение $(a: R)(b: R)(c: R)\ (a + b) * ((b + c) * a)$ (1). Пусть также, на некотором шаге этого вывода было получено математическое выражение $(a: R)(b: R)(c: R)\ ((a * b) + (a * c)) * (a + b)$ (2). В данном случае процесс вывода считается завершенным, поскольку на очередном шаге вывода получено выражение, ограниченно синтаксически эквивалентное цели вывода.

При проверке сводимости выражения (2) к выражению (1) сначала используется свойство коммутативности операции умножения ($(x: R)(y: R)\ x * y = y * x$), а затем используется утверждение из БЗ:

$(x: R)(y: R)(z: R)\ x * y + z * y = (x + z) * y$ (правило выноса общего множителя за скобки). Пункты 2 и 3 *определения 3* выполняются.

В системах, не поддерживающих проверку того, что на очередном шаге вывода получено математическое утверждение, эквивалентное доказываемому, процесс вывода должен быть продолжен.

Пример 2 (на адаптивную унификацию). Пусть проверяется унифицируемость метаматематического выражения $(v: R)\exists v\ \boldsymbol{f}\vdash v\dashv$ (1) и математического выражения $(x_1: R)(x_2: R)(\varepsilon: R)\neg(\forall\varepsilon(|x_1 - x_2| \geq \varepsilon))$ (2). Очевидно, что *простая* унификация этих выражений даст отрицательный результат, поэтому применим к выражению (2) эквивалентные преобразования на основе утверждений в форме равносильности из базы знаний.

Применив утверждение $(v: R)\neg\forall v\ \boldsymbol{f}\vdash v\dashv \Leftrightarrow \exists v\neg\ \boldsymbol{f}\vdash v\dashv$, из (2) получим $(x_1: R)(x_2: R)(\varepsilon: R)\ \exists\varepsilon(\neg(|x_1 - x_2| \geq \varepsilon))$ (3). Применив затем утверждение $(x_1: R)(x_2: R)\ \neg(x_1 \geq x_2) \Leftrightarrow x_1 < x_2$, из (3) получим $(x_1: R)(x_2: R)(\varepsilon: R)\ \exists\varepsilon(|x_1 - x_2| < \varepsilon)$ (4). Выражения (1) и (4) унифицируемы.

Подстановка: $\sigma = (v \mid \varepsilon, \boldsymbol{f} \mid |x_1 - x_2| < \boldsymbol{\tau_1})$; $\upsilon = (x_1: R)(x_2: R)(\varepsilon: R)$.

Множество утверждений Ω: $\{R \subseteq R\}$.

Заключение

В настоящей работе представлен алгоритм решения задачи проверки на ограниченную синтаксическую эквивалентность математических выражений, а также задачи адаптивной унификации для расширяемой модели математического диалекта. Вторая задача рассматривается впервые для систем доказательств теорем.

Предложенный алгоритм позволяет за конечное число шагов установить, являются ли исследуемые выражения ограниченно синтаксически эквивалентными либо адаптивно унифицируемыми (в зависимости от решаемой задачи). Результат при этом может быть не всегда достоверным: алгоритм может давать отрицательный результат, в то время как выражения на самом деле являются ограниченно синтаксически эквивалентными (адаптивно унифицируемыми). Вместе с тем достоверность результата может быть повышена за счет расширения базы знаний необходимыми утверждениями в форме равенства или равносильности.

ЛИТЕРАТУРА

1. *Гаврилова Т.Л., Клещев А.С.* Внутренняя модель математической практики для систем автоматизированного конструирования доказательств теорем // Проблемы управления. – 2006. – № 4. – Ч. 1. – С.32-35; № 5. – Ч. 2. – С.68-73; № 6. – Ч. 3. – С.68-71.
2. *Клещев А.С., Тимченко В.А.* Алгоритм унификации для расширяемой модели математического диалекта // Информатика и системы управления. − 2012. − № 1(31). − С.155-165.
3. Wolfram Mathematica 9. [Электронный ресурс]. URL: http://www.wolfram.com/mathematica (дата обращения 14.12.2012).
4. Maple 16. [Электронный ресурс]. URL: http://www.maplesoft.com/products/Maple (дата обращения 14.12.2012).
5. Matlab. The language for Technical Computing. [Электронный ресурс]. URL: http://www.mathworks.com/products/matlab (дата обращения 14.12.2012).
6. *Клещев А.С., Тимченко В.А.* Задача применения подстановки для расширяемой модели математического диалекта // Информатика и системы управления. − 2011. − № 3(29). − С. 80-88.

E-mail:
Клещев Александр Сергеевич – kleschev@iacp.dvo.ru;
Тимченко Вадим Андреевич – rakot2k@mail.ru.

УДК 004.93

 С.А. Субботин, канд. техн. наук
(Запорожский национальный технический университет, Украина)

ФОРМИРОВАНИЕ ВЫБОРОК ДЛЯ РЕШЕНИЯ ЗАДАЧ КЛАССИФИКАЦИИ ПО ПРИЗНАКАМ*

Решена задача формирования выборок для автоматизации решения задач классификации объектов по признакам. Предложен новый метод формирования обучающих выборок, который обеспечивает сохранение в сформированной подвыборке топологических свойств исходной выборки и не требует при этом загрузки всей выборки в память ЭВМ, что позволяет сократить объем выборки и уменьшить требования к ресурсам ЭВМ.
Ключевые слова: выборка, отбор экземпляров, редукция данных, классификация, распознавание образов, сокращение размерности данных.

* Работа выполнена в рамках госбюджетных научно-исследовательских тем Запорожского национального технического университета "Методы, модели и устройства принятия решений в системах распознавания образов" и "Интеллектуальные информационные технологии автоматизации проектирования, моделирования, управления и диагностирования производственных процессов и систем".