О развитии техники моделирования логических процессов

А. С. Подколзин

Логические процессы занимают центральное место в поведении любых интеллектуальных систем, и прогресс в развитии искусственного интеллекта в значительной степени определяется прогрессом в развитии техники моделирования логических процессов. Первые шаги в освоении такой техники естественно было предпринимать на материале хорошо формализованных предметных областей, располагающих богатыми ресурсами обучающего материала и позволяющих на первых этапах обособиться от дополнительных трудностей, привносимых нечеткостью, неполнотой и частичной противоречивостью данных в задачах плохо формализованных областей. По этой причине, описываемая работа по моделированию логических процессов была связана с созданием компьютерных решателей математических задач и исследованием возможностей автоматизации процессов математического творчества. Разумеется, многие из найденных здесь конструктивных решений могут быть использованы и в других предметных областях, допускающих различную степень нечеткости своих понятий. Результатом работы явилась компьютерная система решения стандартных задач в следующих разделах элементарной математики и математического анализа:

а) Упрощение алгебраических, тригонометрических и трансцендентных выражений и доказательство тождеств с такими выражениями.

б) Решение алгебраических, логарифмических, показательных и тригонометрических уравнений, а также систем уравнений.

в) Доказательство и решение неравенств.

г) Решение простейших задач по комбинаторике.

д) Решение планиметрических задач на вычисление.

е) Вычисление производных и пределов, с упрощением возникающих здесь выражений и анализом условий на параметры.

ж) Использование производных и пределов для доказательства неравенств, отыскания экстремумов и качественного исследования графика функции.

з) Формальное интегрирование и вычисление определенных интегралов.

и) Исследование сходимости рядов в зависимости от значений параметров. Разложение в ряды Тейлора и Фурье (в виде бесконечных сумм), а также суммирование рядов.

Существенной особенностью системы, выгодно отличающей ее от систем компьютерной алгебры, является возможность пошагового просмотра всего процесса решения задачи с использованием обычной математической записи формул и сопровождением геометрических задач чертежом, автоматически пополняемым по ходу решения. Такого рода семантическая трассировка совершенно необходима для эффективного обучения системы. Для ввода в решатель новых процедур решения задач (приемов) разработаны средства, обеспечивающие столь же высокую степень наглядности, как и для отображения процесса решения. Приемы решения задач представлены в виде теорем предметной области, снабженных некоторой «алгоритмизирующей» разметкой и сгруппированных по тематическому принципу в разделы и подразделы. Достигнутый уровень технологии обучения делает развиваемую систему не просто решателем задач в конкретных разделах математики, но обучаемой компьютерной системой, дальнейшее развитие которой, хотя и трудоемкое, в принципе, доступно пользователям достаточно высокой квалификации.

Исключительное разнообразие, многоуровневый характер и сложность алгоритмических конструкций решателя, возникших при проработке конкретных предметных областей, представляются совершенно необходимыми для его эффективности в этих областях. Вместе с тем, они существенно затрудняют работу над созданием компьютерной логической системы, способной к саморазвитию.

Так как подавляющее большинство приемов решателя представляются в виде теорем предметной области, снабженных алгоритмизирующей разметкой, то процесс автоматического развития базы приемов можно разбить на две составляющие: автоматическое развитие базы теорем и автоматический синтез приемов по новым теоремам. Были проведены исследования по организации циклов автоматической генерации новых теорем и синтеза приемов для этих теорем в нескольких простейших разделах (алгебра логики, алгебра множеств и арифметика). Применение специальной техники редуцирования теорем, возникающих после применения правил логического вывода, а также эвристических методов фильтрации результатов вывода, позволили обеспечить автоматическое порождение в крайне небольшой период (десятки минут) основного теоремного запаса в указанных разделах — идентичного (и даже несколько более полного) тому, на основе которого в них создавались приемы при обучении вручную. Использование простейших целевых функционалов позволило, далее, организовать автоматический синтез вполне удовлетворительных баз приемов для решения задач на упрощение выражений и решение уравнений в алгебре логики и алгебре множеств. Однако, уже при переходе к несколько более развитой предметной области, каковой является элементарная алгебра, возникли трудности с выявлением сколь-нибудь простой системы общих принципов, объясняющих целесообразность сопровождения теорем теми решающими правилами, которые направляют ход решения задач по традиционному руслу и, по-видимому, не имеют практически приемлемых существенных альтернатив. Сложилось впечатление, что автоматическое формирование таких решающих правил «в реальном времени», при обработке потока теорем, возникающих на выходе блока логического вывода, вряд ли возможно — оно требует гораздо более трудоемкого процесса адаптации к предметной области, основанного на отборе лучших вариантов из многочисленных альтернативных версий организации логических процессов. Это означает, что работу над автоматизацией развития компьютерного решателя задач, которую необходимо проводить параллельно с продолжением его обучения, на ближайших этапах следует направить, в первую очередь, на автоматизацию развития базы теорем — продолжить исследования по циклам логиче-

ского вывода с редуцированием и фильтрацией, хорошо зарекомендовавшим себя в проведенных экспериментах, а также исследования по логике постановки задач, ориентированных на целевое пополнение базы теорем. Автоматические синтез и оптимизация решающих правил приемов должны будут, по-видимому, происходить путем формирования тестов при анализе реального и возможных альтернативных поведений решателя на конкретном множестве обучающих задач. Эту работу естественно начинать с создания автоматического анализатора решений, который мог бы быть использован и как дополнительное средство отладки решателя при обучении его вручную.

Общая схема моделирования процесса решения задачи в решателе и языки ЛОС, ГЕНОЛО́Г, используемые для представления приемов, уже были описаны в предыдущих публикациях [1, 2] и будут далее предполагаться известными. В этой заметке мы приведем лишь несколько примеров, иллюстрирующих применение языка ГЕНОЛО́Г для записи приемов решения задач из различных предметных областей. Всего таких приемов в настоящей, третьей версии решателя насчитывается около 9 тысяч.

Описание приема на ГЕНОЛО́Гe складывается из следующих компонент:

а) формулировка теоремы предметной области, на которой основан прием;

б) заголовок приема, указывающий способ применения теоремы для преобразования текущей ситуации, возникшей в процессе решения задачи;

в) решающие правила приема — условия на целесообразность применения рассматриваемого преобразования (будем такие условия далее называть фильтрами приема);

г) указания транслятору, уточняющие способ формирования программы приема;

д) ссылки на пакеты продукций, применяемые для стандартизации вспомогательных выражений, формируемых в процессе применения приема.

В указанном виде определяются как основные приемы решателя, активизируемые при сканировании задачи, так и продукции различ-

ных вспомогательных продукционных пакетов. Продукционные пакеты играют в решателе роль ускорителей вычислений: уменьшение в десятки и сотни раз количества приемов, среди которых требуется найти очередное преобразование, приводит к многократному уменьшению трудоемкости. Такие пакеты, как правило, используются в процессе применения обычного приема, заданного на ГЕНОЛО́Ге. Рассматриваются четыре типа пакетов продукций, аналогичных четырем типам задач:

а) Пакеты для тождественных либо эквивалентных преобразований терма над заданным списком посылок (аналог задач на преобразование);

б) Пакеты для проверки истинности утверждения в предположении истинности заданного списка посылок (аналог задач на доказательство);

в) Пакеты для определения значений переменных, при которых становится истинным утверждение заданного вида (аналог задач на описание);

г) Пакеты для целевого вывода следствий из исходного списка посылок (аналог задач на исследование).

Для краткости, пакеты продукций перечисленных типов будем называть, соответственно, нормализаторами, проверочными операторами, синтезаторами и анализаторами. Заметим, что в проверочных операторах и синтезаторах используется только процедура обратного вывода, дополняющая применяемый при сканировании задачи прямой вывод.

Список иллюстративных примеров начнем с описания на ГЕНОЛО́Ге приема для решения квадратных уравнений. Теорема этого приема имеет вид:

$$\forall abcxd\ (d = b^2 + 4ac \Rightarrow (ax^2 + bx = c \Leftrightarrow \neg(a = 0)\ \&$$
$$\&\ 0 \leqslant d\ \&\ \left(x = \frac{\sqrt{d} - b}{2a} \vee x = -\frac{\sqrt{d} + b}{2a}\right) \vee bx = c\ \&\ a = 0)).$$

Заголовок приема указывает, что применяется эквивалентная замена «слева направо». Решающие правила приема определяют следующие условия его применения:

а) Рассматривается задача на описание; текущий уровень ее сканирования — 1 либо 2; уравнение имеет корневое вхождение в условие задачи.

б) Выражения a, b, c не имеют вхождений неизвестных задачи, а выражение d — имеет вхождение неизвестной.

в) Если число неизвестных задачи более 1, то текущий уровень сканирования равен 1, иначе он равен 2.

г) Либо число входящих в уравнение неизвестных равно 1, либо уравнение не имеет целочисленных неизвестных.

Условия в) и г) здесь представляют собой «тесты», введенные в процессе регулировки поведения базы приемов на обучающих выборках задач, и их целесообразность может быть прокомментирована лишь при анализе тех нескольких задач, для которых они оказались существенными (например, условие г) предотвращает применение формулы корней квадратного уравнения в тех целочисленных уравнениях, где применяются соображения делимости). Условия такого типа могут многократно видоизменяться в процессе обучения, аккумулируя в себе все более изощренные описания ситуаций. Для контроля за возникновением нежелательных ситуаций, не учтенных при очередной коррекции решающих правил, используются циклы автоматической «прогонки» решателя через необходимую выборку обучающего материала. Быстродействие системы позволяет за не очень большое время пропускать через решатель сотни задач (например, «прогонка» около 600 планиметрических задач на вычисление требует порядка 6 минут), и таким образом поддерживать в процессе обучения удовлетворительный уровень регулировки взаимодействия приемов. Для определения после прогонки «особых» точек, требующих коррекции приемов, в системе предусмотрены процедуры автоматического анализа решений.

При формировании программы рассматриваемого приема транслятору передаются следующие указания:

а) Единственная посылка теоремы приема (равенство $d = b^2 + 4ac$) представляет собой присвоение переменной d соответствующего значения.

б) Переменные a, b могут принимать вырожденное значение 1. Перед слагаемыми с коэффициентами a, b может находиться знак «минус», который при идентификации относится к этим коэффициентам.

в) Если число уравнений задачи более одного, то преобразуемое уравнение сопровождается после замены комментарием, ускоряющим переход к разбору случаев.

Для стандартизации вспомогательных выражений, формируемых в процессе применения приема, используются обращения к следующим пакетам продукций (нормализаторам):

а) Выражения $b^2, 4ac, -b, 2a, \frac{\sqrt{d}-b}{2a}, \frac{\sqrt{d}+b}{2a}$ обрабатываются нормализующими пакетами, создаваемыми для общей стандартизации выражений с заголовками «степень», «умножение», «минус» и «дробь». Здесь используются лишь самые простые тождества, типа $a \cdot 1 = a, a^1 = a, \neg(c = 0) \Rightarrow \frac{ac}{bc} = \frac{a}{b}$, и т. п. Продукции этих пакетов записываются на ГЕНОЛО́Ге так же, как приемы, активизируемые при сканировании задачи.

б) Выражение $b^2 + 4ac$ обрабатывается последовательно тремя нормализаторами: сначала — нормализатором общей стандартизации выражений с заголовком «плюс» (приведение подобных членов, устранение вложенных сумм и т. п.); затем — нормализатором раскрывания скобок, и наконец — нормализатором разложения на множители. Последний представляет собой уже весьма мощную процедуру, насчитывающую более полутора сотен продукций (непосредственное разложение на множители; попытки группировок; разложение многочленов путем подбора целочисленных коэффициентов; вспомогательные преобразования выражений с логарифмами, показательной и тригонометрическими функциями).

в) Выражение $\sqrt{d}$ обрабатывается нормализатором общей стандартизации выражений с заголовком «степень», а также вырож-

денным пакетом из единственной продукции, обеспечивающей отбрасывание внешней операции «модуль», если она есть.

г) Числители $\sqrt{d}-b, \sqrt{d}+b$ обрабатываются каждый сначала нормализатором общей стандартизации выражений с заголовком «плюс», а затем — нормализатором для разложения на множители. Входящее в знаменатель выражение a также обрабатывается нормализатором разложения на множители.

д) Условия $a = 0, 0 \leqslant d$ обрабатываются нормализаторами общей стандартизации равенств и неравенств, которые, в частности, могут усмотреть их истинность либо ложность и заменить на логическую константу «истина» либо «ложь».

е) Весь заменяющий терм (правая часть теоремы) обрабатывается нормализатором общей логической стандартизации, обеспечивающим устранение логических констант, если таковые возникли при нормализации его подтермов.

Обработка одним или несколькими нормализаторами (а в особых случаях — и вспомогательными задачами на преобразование либо описание) практически каждого подтерма заменяющего терма приводит к тому, что после применения приема редко возникает необходимость в дополнительных простейших упрощающих преобразованиях. Так как основная трудоемкость процесса решения задачи приходится на «холостой ход» при поиске очередного срабатывания приема, то уменьшение числа циклов сканирования, обусловленное применением нормализаторов, существенно повышает быстродействие системы. Принципы расстановки пометок о нормализации подтермов теоремы в большинстве случаев стандартны, что позволяет применять при записи приема процедуру автоматического формирования таких пометок (эта же процедура создает часть указаний транслятору).

Отметим одну особенность логической формализации при записи приемов решателя. Для устранения необходимости части проверок, имеющих почти формальный характер, иногда удается использовать изменение стандартной интерпретации операций и предикатов вне области допустимых значений их аргументов. Такое изменение никак не сказывается на процессе решения задачи, которое с самого начала реализуется строго в области допустимых значений. С другой стороны,

если, например, условиться, что значениями арифметических операций сложения, умножения, деления и возведения в степень для любых (даже не числовых) значений аргументов являются вещественные числа, то становятся избыточными специальные проверки типа данных для термов с указанными заголовками, формально необходимые во многих приемах. Практически использование нестандартной вне о.д.з. интерпретации приводит лишь к тому, что в посылках теорем постоянно оговаривается выполнение условий на о.д.з. В частности, приведенная выше теорема для квадратных уравнений должна содержать в своей заменяющей части неравенство $0 \leqslant d$, не являющееся в применяемой интерпретации логическим следствием дизъюнкции в скобках. Разумеется, возможны альтернативные версии формализации при обучения решателя, вполне допускаемые применяемыми языками программирования.

Следующий пример задания приема на ГЕНОЛÓГе связан с вычислением пределов по правилу Лопиталя. Теорема приема здесь имеет следующий вид (в этом и последующих примерах условимся опускать внешний квантор общности):

$$d = \lim_{x \to a \setminus b} f(x) \;\&\; e = \lim_{x \to a \setminus b} g(x) \;\&$$
$$\&\; (d = 0 \;\&\; e = 0 \vee (d = \infty \vee d = -\infty) \;\&\; (e = \infty \vee e = -\infty)) \;\&$$
$$\&\; p = \lambda_x \left(\frac{df(x)}{dx},\ x \in \mathbb{R} \right) \;\&\; q = \lambda_x \left(\frac{dg(x)}{dx},\ x \in \mathbb{R} \right) \;\&$$
$$\&\; c = \lim_{x \to a \setminus b} \frac{p(x)}{q(x)} \;\&\; (c \in \mathbb{R} \vee c = \infty \vee c = -\infty) \;\&\; r = c \Rightarrow$$
$$\Rightarrow \lim_{x \to a \setminus b} \frac{f(x)}{g(x)} = r.$$

Запись $x \to a \setminus b$ используется для уточнения типа рассматриваемой окрестности точки a: указатель b определяет двустороннюю, левую либо правую окрестность. Запись вида $\lambda_x(A, B)$ используется для обозначения функции, значение которой в точке x определяется выражением A, а условия на принадлежность значения аргумента x области определения функции — выражением B. Последняя из посылок теоремы, вводящая на первый взгляд избыточное вспомога-

тельное обозначение r, в действительности обеспечивает возможность обращения к нормализаторам после того, как были выполнены проверки, определяемые предпоследней посылкой. Использование такого рода фиктивных посылок является часто применяемым при программировании на ГЕНОЛО́Ге приемом. По этой (и ряду других) причин так называемые «теоремы приемов» не всегда тождественны теоремам предметной области, хотя и возникают из них минимальными модификациями. По существу, преобразование теоремы предметной области в «теорему приема» представляет собой уже начало программирования этого приема.

Вычисление пределов обеспечивается в решателе специальным макронормализатором, насчитывающим несколько сотен продукций. Это оказалось возможным из-за почти рекурсивного характера преобразований и позволяет существенно повысить быстродействие по сравнению с версией, основанной на сканировании задачи. Описываемый прием входит в состав данного макронормализатора. Заголовок его содержит ссылку на нормализатор, а также указание направления тождественной замены (слева направо).

Условия применения приема таковы:

а) Уровень срабатывания приема равен 5 (приемы упорядочены в программе нормализатора по возрастанию своих уровней; для выполнения очередного преобразования каждый раз находится прием с наименьшим уровнем, применение которого допустимо).

б) Длина выражений $f(x), g(x)$ — числителя и знаменателя рассматриваемой дроби — не превосходит 80.

в) Числитель и знаменатель не имеют сомножителя вида $u(x)^v$, где v — не константа, а $\lim_{x\to a\setminus b} u(x)$ равен нулю либо плюс-минус бесконечности.

г) Числитель и знаменатель не содержат факториала с x.

д) Пусть n — число внешних применений правила Лопиталя. Тогда выполнены следующие условия:

д1) либо $n < 2$, либо в числителе или знаменателе выделяется сумма вида $ux + v$, где коэффициент u не содержит переменной x.

д2) либо $n < 3$, либо числитель и знаменатель дроби имеют длину не более 40.

д3) отсутствует зависящая от x тригонометрическая операция, расположенная в основании степени, имеющей при $n < 2$ не константный, а иначе — не целочисленный показатель.

е) Длина выражений $p(x), q(x)$ не превосходит 70.

Как видно, эти условия учитывают, с одной стороны, возможность непосредственного упрощения числителя либо знаменателя при их дифференцировании, а с другой стороны — отсекают случаи чрезмерного усложнения выражений при последовательных дифференцированиях. Используемые в них константы взяты с некоторым запасом, достаточным для обработки всех рассматривавшихся при обучении примеров. При необходимости работы с более громоздкими выражениями они (как и логика решающих правил приема) легко могут быть изменены.

Перечислим указания транслятору по формированию программы описываемого приема:

а) Для проверки истинности посылок с номерами 3 и 7 используются проверочные операторы (какие именно, определяет по виду входящих в эти посылки атомарных утверждений сам транслятор). Прочие посылки суть равенства, вводящие вспомогательные обозначения. Заметим, что смысл проверки, определяемой 7-й посылкой, состоит в проверке того, что предел отношения производных был фактически найден. Для нахождения этого и других указанных в посылках пределов, а также вычисления производных применяются нормализаторы.

б) Выражение $F(x)$ на используемом в решателе логическом языке представляется с помощью двуместного функционального символа «значение», применяемого к функции F и значению ее аргумента x. Транслятору сообщается, что встречающиеся в теореме выражения $f(x), g(x), p(x), q(x)$ идентифицируются не с термами, имеющими своим заголовком символ «значение», а с произвольными термами. Если бы в теореме встречалось отличное от $f(x)$ выражение $f(A)$, то транслятор попытался бы

идентифицировать переменную x из других фрагментов теоремы, после чего выполнил бы необходимую для получения $f(A)$ подстановку в ранее идентифицированный с $f(x)$ терм.

в) Вводится ограничение трудоемкости на попытку применения правила Лопиталя: если числитель либо знаменатель представляет собой степень с натуральным показателем, то это ограничение составляет 7000000 атомарных шагов интерпретатора ЛОСа, иначе — 1500000 шагов. Счетчик шагов представляет собой внутренние «часы» системы; кроме использования для принятия решений, он позволяет ввести координатную шкалу на моделируемых процессах, удобную для локализации событий при отладке.

Принципы разметки теоремы обращениями к нормализаторам общей стандартизации выражений с заданными заголовками в данном приеме такие же, как в предыдущем. Отметим лишь, что для вычисления производной используется пакет продукций, приемы которого соответствуют основным правилам дифференцирования. Этот пакет осуществляет лишь простейшие стандартизирующие преобразования. Более глубокие упрощения возникающего при дифференцировании выражения обычно реализуются в задаче на преобразование. Здесь, однако, специальное обращение к такой задаче является избыточным, так как оно выполняется нормализатором вычисления предела, применяемым к отношению производных. Все встречающиеся в посылках и решающих правилах теоремы пределы рекурсивным образом обрабатываются нормализатором вычисления пределов, к которому относится и данный прием. Заметим, что рекурсия в этом нормализаторе организована так, что она предусматривает возможность передачи из подпроцесса в надпроцесс указания о необходимости внешнего разбора случаев по условиям на параметры. Такое указание передается сквозным образом до обращения к вычислению исходного предела, где и реализуется собственно разбор случаев. В результате решатель получает возможность находить разветвленные логические описания, определяющие вид найденного предела в зависимости от условий на значения параметров. Еще одной особенностью данного приема является передача используемым в нем нормализато-

рам дополнительной посылки $x \to a \setminus b$, позволяющей выполнять все проверки корректности преобразований лишь в окрестности точки a. Последняя посылка $r = c$ используется для упрощения найденного предела c при помощи вспомогательной задачи на преобразование (если исходное выражение не имело параметров, то данное обращение к задаче отменяется).

Как и вычисление пределов, формальное интегрирование в решателе осуществляется специальным нормализующим макропакетом. Для иллюстрации ограничимся одним из простейших приемов, выполняющим замену переменной интегрирования «через тангенс половинного аргумента». Теорема приема имеет вид:

$$\lambda_x(f(x), x \in \mathbb{R}) = \lambda_x\left(g\left(\tan\frac{x}{2}\right),\ x \in \mathbb{R}\right)\ \&$$
$$\&\ \int \frac{g(x)}{x^2+1}dx = \lambda_x\left(h(x),\ x \in \mathbb{R}\right) \Rightarrow$$
$$\Rightarrow \int f(x)dx = \lambda_x\left(2h\left(\tan\frac{x}{2}\right),\ x \in \mathbb{R}\right).$$

Условий применения приема всего два: требуется, чтобы внутри подынтегрального выражения $f(x)$ встречался $\sin x$ либо $\cos x$, причем все зависящие от x тригонометрические аргументы в $f(x)$ были кратны x. Уровень срабатывания приема достаточно высок, чтобы решатель обращался к нему лишь по исчерпании менее трудоемких средств интегрирования. Транслятору сообщается, что первая посылка используется для обращения к нормализатору, предпринимающему попытку выразить подынтегральное выражение через тангенс половинного аргумента (около 20 простых продукций), и последующего извлечения выражения $g(x)$; $g(\tan\frac{x}{2}) = f(x)$. Вторая посылка используется для рекурсивного обращения к нормализатору формального интегрирования; $h(x)$ — результат этого обращения. Как и в предыдущем примере, уточняется, что для $f(x), g(x), h(x)$ применяется идентификация, связывающая с этими функциональными переменными произвольные выражения, встречающиеся на соответствующей позиции в нормализуемом терме. Кроме уже указанных, в приеме используются следующие обращения к нормализаторам: для упрощения подынтегрального выражения во второй посылке решает-

ся вспомогательная задача на преобразование, целевая установка которой содержит указание на упрощение для последующего интегрирования по переменной x; $h(\tan\frac{x}{2})$ обрабатывается с помощью специального нормализатора упрощения результатов формального интегрирования. Применение такого нормализатора перед выходом во внешнюю задачу оказалось целесообразным для ускорения вычислений. В нем собраны около сотни тождеств, вероятность применения которых к выражению, возникшему после применения стандартных приемов интегрирования, существенно выше, чем в общем случае.

Приведем простые примеры продукций, возникающих в проверочных операторах, синтезаторах и анализаторах. Для иллюстрации остановимся на пакетах продукций, используемых для проверки неравенств. Здесь имеются два проверочных оператора. Первый из них обеспечивает быстрое усмотрение неравенства в ситуациях, когда оно либо имеет место почти наверняка и должно усматриваться простыми средствами (например, при контроле корректности преобразований), либо когда нет каких-либо априорных оснований ожидать выполнение проверяемого неравенства, и более трудоемкие процедуры проверки существенно увеличат время решения. Второй проверочный оператор используется, если явно поставлена задача на доказательство неравенства, причем активизируется он лишь после попытки применить первый. Оператор быстрого усмотрения неравенств насчитывает около сотни продукций; оператор усиленной проверки неравенств — около двух десятков продукций. В действительности используются четыре оператора: по два для строгих и для нестрогих неравенств. В первых версиях решателя использовались объединенные операторы проверки строгих и нестрогих неравенств; раздельная их реализация привела к упрощению записи приемов и практически не сказалась на быстродействии. Так как интерфейс редактора ГЕНОЛÓГа сводит к минимуму процедуру дублирования приема и внесения в нее необходимых модификаций, то такого рода «тиражирование» приемов оказалось удобным и стало применяться в последней версии решателя достаточно часто. По-видимому, по мере развития автоматизации процесса обучения решателя многократное дублирование приема в различных модификациях (использование одной и той же теоремы как для приема сканирования задачи, так и

для различных пакетов продукций; создание серий приемов, отличающихся лишь решающими правилами, и т. п.) станет естественной нормой.

В качестве простейшего приема первого проверочного оператора для нестрогих неравенств приведем процедуру установления неотрицательности суммы путем установления неотрицательности всех слагаемых. Теорема приема имеет здесь вид:

$$0 \leqslant a \;\&\; 0 \leqslant b \Rightarrow 0 \leqslant a + b.$$

Заголовок приема проверочного оператора содержит лишь указание названия этого оператора. Решающие правила рассматриваемого приема определяют его срабатывание на уровне 2 при условии, что выражение $a + b$ не является константой (для проверки неравенств с константными частями используются другие приемы). Указания транслятору определяют рекурсивное обращение к этому же оператору для проверки обеих посылок. Кроме того, имеется указатель, определяющий обработку в общем цикле программы приема всех слагаемых правой части неравенства. При отсутствии такого указателя программа приема осуществляла бы обычное рекурсивное «расщепление» правой части — в качестве a выбиралось бы одно слагаемое, а b идентифицировалось бы с суммой остальных слагаемых. Разумеется, это замедляло бы процесс проверки.

В качестве примера приема второго, усиленного оператора проверки неравенств рассмотрим процедуру «поглощения» некоторого, быть может, отрицательного слагаемого правой части неравенства с помощью двух заведомо положительных слагаемых. Здесь применяется обычное неравенство для суммы квадратов и удвоенного произведения. Теорема приема имеет следующий вид:

$$0 \leqslant a \;\&\; 0 \leqslant b \;\&\; c^2 = ab \;\&\; 0 < d \;\&\; 0 < e \;\&\; f = 4de - g^2 \;\&\; \\ \&\; 0 \leqslant f \;\&\; 4di \leqslant fb + 4dh \Rightarrow i \leqslant da + eb + gc + h.$$

Здесь третье слагаемое правой части проверяемого неравенства поглощается первым и частью второго слагаемых, причем $\frac{fb}{4d}$ — оставшаяся после поглощения часть второго слагаемого. Решающие правила определяют срабатывание приема на уровне 3 при условии,

что d, e, g идентифицированы с числовыми константами, причем не усматривается (с помощью первого оператора проверки неравенств) неравенство $0 \leqslant gc$. Указания транслятору определяют обработку первых двух посылок при помощи проверочного оператора быстрого усмотрения неравенств; непосредственные действия с числовыми константами в 4–7 посылках; идентификацию выражения b по ранее идентифицированным a, c из третьей посылки и рекурсивное обращение к усиленному оператору проверки неравенств при обработке последней посылки. Уточняется, что переменные d, e, g могут принимать вырожденное значение 1, а переменная h — идентифицироваться с нулем; если перед gc имеется внешний знак минус, то при идентификации он передается переменной g. Наконец, вводятся оптимизирующие ограничения на порядок идентификации: обработка слагаемого eb начинается только после того, как было идентифицировано b, а обработка слагаемого da — только после того, как было идентифицировано c.

Подобно тому, как обращение к проверочному оператору является мини-версией обращения к решению задачи на доказательство, обращение к синтезатору является мини-версией обращения к решению задачи на описание. По сравнению с общим случаем задач на описание, список условий синтезатора, однако, жестко ограничен. Он определяется посредством шаблона, в котором выделены входные переменные (синтезатору передается набор идентифицируемых с ними «известных» термов), и выходные переменные («неизвестные»). Как и проверочный оператор, синтезатор предпринимает попытку определить значения неизвестных путем обратного вывода. Рекурсивный характер процедуры определяет здесь сначала декомпозицию исходного условия со сложными входными выражениями, а после реализации семейства возникающих таким образом простейших условий — сборку (синтез) значений неизвестных из атомарных объектов.

В качестве примера рассмотрим прием синтезатора получения верхних оценок. Этот синтезатор служит для перечисления некоторых термов x (неизвестных), удовлетворяющих неравенству $a \leqslant x$. Вид таких термов уточняется в передаваемых синтезатору указаниях; по умолчанию в качестве x допускаются только числовые константы. Простейший прием данного синтезатора — получение верхней оценки для суммы выражений — основан на теореме:

$$a \leqslant c \;\&\; b \leqslant d \Rightarrow a + b \leqslant c + d.$$

Этот прием определяет рекурсивное обращение к получению верхних оценок c и d для a и b; после того, как они получены, сумма $c + d$ обрабатывается нормализатором общей стандартизации сумм и выдается в качестве результата. В качестве примера приема, завершающего цепочку рекурсивных обращений, приведем прием для получения оценки линейной комбинации синуса и косинуса:

$$a \sin b + c \cos b \leqslant \sqrt{a^2 + c^2}.$$

По умолчанию, a и c здесь идентифицируются с числовыми константами.

Наконец, приведем пример приема, используемого в анализаторе. Анализирующие пакеты создаются для ускорения вывода следствий в задачах на исследование. Они позволяют накапливать большие массивы утверждений, формируемые с помощью сравнительно небольшого числа приемов и выведенные за рамки трудоемкого процесса сканирования, использующего полную базу приемов. В процессе применения анализатора предпринимается отбор небольшого числа получаемых утверждений, перенесение которых во внешнюю задачу на исследование представляется бесспорно целесообразным. Такое перенесение реализуется либо по окончании цикла вывода (на который установлен определенный лимит времени), либо, в особых случаях, происходит немедленно, с обрывом работы анализатора. Рассмотрим анализатор, генерирующий линейные комбинации алгебраических уравнений при решении систем уравнений (решатель использует два таких анализатора, отличающиеся целевыми установками: в одном случае преследуется цель уменьшить число неизвестных либо получить линейное относительно некоторой неизвестной уравнение; в другом — цель получить уравнение, левая часть которого имеет вид степени либо произведения неизвестных множителей). Простейший прием этого анализатора создает линейную комбинацию двух уравнений, если в них усматриваются неизвестные слагаемые, отличающиеся лишь известными коэффициентами. Теорема приема имеет следующий вид:

$$ag + b = c \;\&\; ah + d = e \;\&\; f = bh - dg - ch + eg \Rightarrow f = 0.$$

Первые две посылки суть исходные уравнения; последняя вводит вспомогательное обозначение f для обработанной нормализаторами линейной комбинации уравнений. Ввод данного обозначения позволяет явно ссылаться на линейную комбинацию в фильтрах приема. Эти фильтры таковы:

а) Выражения g, h не содержат неизвестных, а выражение a — содержит. Заметим, что правые части уранений, в силу обеспечиваемой другими приемами стандартизации, не содержат неизвестных.

б) Левые части уравнений не имеют дробных слагаемых.

в) Либо число неизвестных слагаемых в f меньше, чем в каждом из исходных уравнений, либо число неизвестных в f меньше, чем в каждом из исходных уравнений, либо f линейно относительно некоторой неизвестной.

Прием имеет следующие указания транслятору:

а) Переменные g, h могут вырожденным образом идентифицироваться с единицей; знак минус перед первым слагаемым левой части (если он есть) передается этим переменным. Переменная d может вырожденным образом идентифицироваться с нулем.

б) Если число неизвестных в f равно 1 либо f имеет вид $pX^s + qY^s$ при некоторых неизвестных X, Y и известных p, q, s, то работа анализатора обрывается, и результат $f = 0$ регистрируется в списке посылок внешней задачи на исследование. Это же происходит, если после нормализации f оказалось представлено в виде произведения, и уравнение $f = 0$ приобрело вид дизъюнкции нескольких равенств.

в) Если линейная комбинация уравнений линейна хотя бы по одной из неизвестных, причем ни одно из исходных уравнений не линейно ни по какой неизвестной, то делается пометка о перенесении этой линейной комбинации (после завершения реализуемого анализатором цикла вывода следствий) в список посылок

внешней задачи на исследование. Такая же пометка делается, если f линейно по всем входящим в него неизвестным, а хотя бы одно из исходных уравнений нелинейно.

г) Если удается усмотреть, что $h \neq 0$, то первое уравнение снабжается комментарием, указывающим, что оно является следствием нового уравнения $f = 0$ и прочих (отличных от него самого) посылок, использованных при срабатывании приема. Аналогичным образом комментируется второе уравнение. Эти комментарии могут быть использованы, если при выводе следствий будут найдены значения неизвестных и понадобится усмотреть избыточность части исходных уравнений на этапе проверки.

При обработке выражения f применяются нормализаторы раскрывания скобок; упрощения выражений относительно неизвестных и ускоренного разложения на множители. Уравнение $f = 0$ обрабатывается нормализатором общей стандартизации числовых равенств.

Продолжающееся обучение решателя с применением языка ГЕНОЛО́Г приводит к расширению перечня используемых в нем указателей алгоритмизации теорем. Хотя на первых этапах проработки очередной предметной области такое расширение требует определенных временны́х затрат на развитие транслятора ГЕНОЛО́Га, эти затраты быстро окупаются при накоплении и регулировке больших серий приемов, и общая трудоемкость процесса обучения оказывается значительно меньшей, чем при обучении непосредственно на ЛОСе. Развитие исследований, связанных с языком ГЕНОЛО́Г, по существу, предствляет собой исследование пограничного слоя между логикой и алгоритмами, и необходимо для последующей автоматизации развития решателей задач.

В заключение автор выражает глубокую благодарность В. Б. Кудрявцеву, поддержка которого сделала возможным проведение данной работы.

Список литературы

[1] Подколзин А. С. Компьютерное моделирование процессов решения математических задач. М.: Изд-во мех.-мат. факультета МГУ, 2001. 235 с.

[2] Подколзин А. С. О формализации приемов решения математических задач. // Интеллектуальные системы. М., 1998. Т. 3. Вып. 3–4. С. 51–74.