Логические исследования
2017. Т. 23. № 1. С. 121–139
УДК 16

Logical Investigations
2017, Vol. 23, No. 1, pp. 121–139
DOI: 10.21146/2074-1472-2017-23-1-121-139

В.И. Шалак

Аналитический подход к решению задач

Шалак Владимир Иванович
Сектор логики, Институт философии РАН
Российская Федерация, 109240, г. Москва, ул. Гончарная, д. 12, стр. 1.
E-mail: shalack@gmail.com

Настоящая работа посвящена формализации аналитического подхода к решению задач. Обычно считается, что задача включает две составляющие — условия и цели. Условия A — это то, что дано, а цели B — то, что требуется найти или построить. В этом случае при формальном анализе решение рассматривается как некоторый вывод $A \vdash B$ цели из условий задачи. Такое представление широко распространено, но слишком узко для применения в реальной практике. Возьмем, например, задачу построить железную дорогу между двумя городами. Очевидно, что существует много вариантов прокладки дороги, и условия реализации каждого из этих вариантов будут различаться. Это означает, что в момент постановки задачи нет точных формулировок ни цели, ни условий, чтобы ее можно было представить в стандартном виде. Необходим дополнительный аналитический этап решения задачи. Он заключается в последовательном уточнении цели и ее редукции к более простым подцелям, которые на заключительных шагах образуют совокупность достаточных условий решения задачи. В предлагаемой работе построено аналитическое исчисление, которое в определенной степени формализует этот процесс.

Ключевые слова: решение задач, логическая редукция, аналитические таблицы, теория определений

1. Преамбула

Вспомним школу и одну из типичных задач арифметики, которую в разных вариантах всем нам приходилось решать.

Есть бассейн объемом V кубических метров и две трубы. Через одну трубу в него поступает вода со скоростью n литров в минуту, а через вторую — вытекает со скоростью m литров в секунду. Требуется определить, через сколько минут бассейн будет наполнен до краев.

Мы составляем уравнение $n \times t - m \times t \times 60 = V \times 1000$ и из него находим время $t = V \times 1000/(n - m \times 60)$. Задача решена. Если $t \geq 0$, то бассейн наполнится водой через t минут, а если $t < 0$, то никогда.

Задача довольно рафинированная. Для ее решения достаточно составить уравнение, связывающее указанные в условии задачи параметры, и после этого произвести вычисление.

Решение подобных задач, но в общем виде, исследуется в теории синтеза программ [2]. Есть описание $A(x)$ предусловий выполнения программы, и есть описание постусловий $B(x, y)$, при которых она должна завершиться. Средствами интуиционистской логики ищется доказательство формулы $\forall x(A(x) \Rightarrow \exists y B(x, y))$. Если доказательство найдено, то по его шагам синтезируется соответствующая программа. Это возможно благодаря реализуемостной интерпретации интуиционистской логики.

С момента возникновения интереса к Искусственному Интеллекту отдельным направлением исследований в его рамках стало планирование действий. Одной из стандартных тестовых задач является планирование действий в мире кубиков [8, с. 337–363], когда начальную конфигурацию кубиков необходимо преобразовать в целевую. При этом набор возможных элементарных действий ограничен. В качестве эвристики можно ввести функцию расстояния от текущего состояния мира до целевого и при перестановке кубиков стремиться минимизировать ее значение.

Подобные задачи интересны, но очень ограничены. Начальное и целевое состояния в них описаны полностью и остается лишь найти путь от первого ко второму в некотором пространстве промежуточных состояний. Гораздо больший интерес представляют задачи, решение которых требует выхода за рамки изначальных условий. Это один из отличительных признаков творческих задач.

Начнем с простого, вспомним, как мы планируем летний отпуск. Цель — хорошо его провести. Мы садимся и начинаем думать, как этого достичь? Цель расщепляется на альтернативы. У нас есть выбор между отдыхом дома на диване с книжкой, отдыхом на дачных грядках, пляжным отдыхом, отдыхом на природе, активным отдыхом, музейным и т. д. Наши предпочтения определяют, какой из ва-

риантов мы будем рассматривать в первую очередь. Допустим, мы выбрали пляжный отдых. После этого мы задумываемся о месте, где его лучше всего провести. Опять появляются альтернативы по странам и затем по городам, и опять наши предпочтения определяют, какие варианты мы будем рассматривать в первую очередь. После того как страна и город выбраны, мы должны выбрать тип размещения — отель или апартаменты. Выбор альтернатив большой и осуществляется на основе предпочтений по расположению, цене, объему предоставляемых услуг. Информация об этом берется из справочников и специализированных сайтов Интернета. На конечном шаге мы должны решить, либо самим бронировать места и авиабилеты, либо обратиться в туристическое агентство. Исходная цель сведена к простым действиям, которые мы уже знаем, как осуществить. Таким образом, планирование действий путем анализа сложной и недостаточно четко определенной цели сведено к простым действиям, которые ограничены рамками частичных описаний текущего состояния мира. Так же мы можем рассмотреть и другие варианты отдыха, сравнить их и выбрать наиболее предпочтительный.

Необходимо обратить внимание на важное отличие решения задач по планированию действий, связанных с *преобразованием* текущего состояния мира, от задач, которые можно назвать *статическими*. В последних задачах мы сводим их решение к решению более простых задач и в конечном счете к нахождению значений неизвестных параметров. После этого мы просто синтезируем решение. В задачах планирования действий мы тоже сводим решение сложных задач к решению более простых, но на конечном этапе приходим к элементарным ситуациям, которые пока что не имеют места в мире. И теперь, чтобы решить такую задачу, мы должны выполнить некоторые действия, направленные на то, чтобы эти элементарные ситуации, к которым мы свели задачу, имели место. Например, мы свели задачу об отдыхе к тому, чтобы забронировать места в отеле и купить билеты для поездки. Это мы и должны сделать, чтобы отдых состоялся.

Рассмотрим еще одну творческую задачу. Ночью на берегу моря мы нашли чугунную на вид гирю G и хотим узнать, может ли

внутри нее быть спрятано золото? Никакого очевидного решения не просматривается, на зуб не проверишь. Поэтому мы редуцируем начальную задачу к двум альтернативам, которые первыми пришли нам в голову. Первая альтернатива — распилить и посмотреть, а вторая — сравнить физические параметры G с параметрами гири G_0 в том случае, если бы она действительно была изготовлена из чугуна. Первую альтернативу выбрал Шура Балаганов, мы выбираем вторую. Из всех параметров наибольший интерес представляет вес гири, поскольку нам известно, что плотность золота почти в два с половиной раза больше плотности чугуна, и поэтому вес гирь G и G_0 должен заметно отличаться. Вес гири G легко определить путем взвешивания. Это мы сделать можем, у нас под рукой оказались весы-безмен. Второй такой же, но полностью чугунной гири у нас нет. Поэтому нам остается прибегнуть к теории. Весь гири G_0 вычисляется по формуле $P_{G0} = m_{G0} \times g = V \times \rho_{ch} \times g$. Плотность чугуна ρ_{ch} и ускорение земного тяготения g мы помним со школьной скамьи, осталось лишь определить объем V. Но как это сделать? Одна альтернатива — набрать в кастрюлю воду до краев, опустить в нее гирю, собрать в мензурку воду, которая выльется из кастрюли, и узнать объем V. Но у нас на берегу моря нет ни кастрюли, ни мензурки. Значит, эта альтернатива нам не подходит. Но есть другая альтернатива. Мы опять подвешиваем гирю на безмене и опускаем ее в воду, чтобы измерить вес P_{Gw} гири в воде. Согласно закону Архимеда $P_{Gw} = P_G - V \times \rho_w \times g$. Отсюда мы получаем $V = (P_G - P_{Gw})/(\rho_w \times g)$ и, подставив в первую формулу, узнаем вожделенное значение веса $P_{G0} = (P_G - P_{Gw}) \times \rho_{ch}/\rho_w$, которое с точностью до погрешности измерений нашего безмена совпадает с P_G.

Задачи планирования отпуска и поиска золота в чугунной гире являются действительно творческими, поскольку для их решения требуется выйти за границы того, что дано изначально. При этом поиск решения не является бессистемным, а заключается в уточнении используемых понятий и их последующей логической редукции. Благодаря этому мы не удаляемся от цели случайным образом, а релевантно расширяем поле поиска вокруг нее. Очевидно, это сильно

отличается от методов синтеза программ и планирования действий в мире кубиков.

2. В поисках метода

Проблема поиска регулярных методов решения творческих задач имеет давнюю историю. Можно вспомнить знаменитые *«Правила для руководства ума»* Р. Декарта [1], где перечисляются и объясняются правила, которым должен следовать ученый в своей работе. Можно вспомнить Лейбница и его идею универсального исчисления, которое позволяло бы единообразным способом находить ответы на любые разумно поставленные вопросы.

В наше время на тему поиска общих методов решения задач много писал Д. Пойя. Широкую известность получили его книги *«Математическое открытие»* [4] и *«Математика и правдоподобные рассуждения»* [3]. Отталкиваясь от идей Декарта, он обращает внимание на эвристическое использование индуктивных рассуждений и рассуждений по аналогии, которые очень часто помогают в поиске решений.

Эта проблема интересовала В.А. Смирнова. В статье *«Творчество, открытие и логические методы поиска доказательства»* [7], он хотел *«. . . обратить внимание на новые возможности использования логических методов и идей в исследовании исключительно сложной проблематики поиска, открытия и творчества»* [7, с. 447], считая, что *«одна из целей науки — создание типовых методов, позволяющих стандартным образом решать целые классы задач. В этом отношении характерно соотношение нестандартного, творческого и стандартного, рутинного моментов в поисках решения задач. Без наличия методов задача может быть сугубо творческой. Создание метода решения превращает творческую задачу в стандартную. Творческая деятельность переходит от решения самой задачи к созданию методов ее решения. Это характерно не только для познавательной, научной деятельности. Создание техники, орудий труда, технологии основано на том же “обращении” творческой деятельности»* [7, с. 446].

Стремительное развитие современных технологий привело к тому, что поиск общих методов решения задач переместился из теоретической плоскости в практическую. В качестве примера можно привести работы Т. Саати [5, 6]. Он предложил метод анализа иерархий целей, который сразу привлек к себе внимание. Заслуга Саати заключается в том, что его метод позволяет одновременного учитывать многие критерии предпочтения при оценке иерархии целей [6, с. 24–27], но в нем не содержится никаких правил построения самих иерархий.

На практике не существует установленной процедуры генерирования целей, критериев и видов деятельности для включения в иерархию или даже в более общую систему. Это зависит от тех целей, которые мы выбираем для декомпозиции сложной системы. Обычно эта процедура начинается с изучения литературы для обогащения мыслями, и часто, знакомясь с чужими работами, мы как бы проходим через стадию мозгового штурма для составления перечня всех концепций, существенных для задачи, независимо от их соотношения или порядка [5, с. 20].

Несмотря на это, работы Саати получили широкую известность и множество практических приложений, что говорит о востребованности такого рода исследований.

3. Основная идея

Если цель, или задача, описана в виде предложения логики предикатов, то ее можно редуцировать к набору подцелей. Но в большинстве действительно интересных случаев цель не бывает описана настолько полно, чтобы для ее решения было достаточно одной лишь логической редукции. В первоначальном описании цели могут встречаться термины, которые нуждаются в дополнительном уточнении. Это приводит к расширению языка описания задачи новыми понятиями и терминами. Они добавляются в язык не случайным образом, а имеют то или иное отношение к основной цели или ее подцелям.

Логическим инструментарием, который позволяет совместить анализ целей с расширением радиуса поиска дополнительной информации, является теория определений. Определения позволяют уточ-

нять (эксплицировать) цели-предикаты, разлагая их на логически связанные подцели, затем на под-подцели и т. д. При этом допускается, чтобы в определяющей части присутствовали термины, которые в будущем также потребуют экспликации.

На шаге уточнения терминов языка происходит вмешательство субъективного фактора. От того, насколько удачно выбрано определение терминов, может зависеть успех в решении всей задачи.

Для простоты дальнейшего изложения мы ограничимся конечными предметными областями, в которых каждый индивид может быть поименован. При задании семантики языка это позволит использовать подстановочную интерпретацию кванторов [9].

4. Схема языка

Так как язык описания задачи в ходе ее решения может расширяться, мы зададим не сам язык, а схему, указав, что может входить в его состав.

Исходные символы

1. $Sort$ — конечное множество сортов;
2. Var — конечное множество индивидных переменных;
3. $Const$ — конечное множество индивидных констант;
4. $Pred$ — конечное множество предикатных констант;
5. $\&, \vee, \neg$ — логические связки;
6. $\forall, \exists$ — кванторы;

Каждой индивидной переменной $x \epsilon Var$ и каждой константе $c \in Const$ сопоставлен их сорт $s \in Sort$. При необходимости будем обозначать это посредством x/s и c/s.

Каждой n-местной предикатной константе P^n также сопоставлен сорт, имеющий вид кортежа сортов ее аргументных мест $\langle s_1, \ldots, s_n \rangle$. Это мы будем обозначать посредством $P^n/\langle s_1, \ldots, s_n \rangle$.

Формулы

1. Если $t_1/s_1, \ldots, t_n/s_n \in Const \cup Var, P \in Pred, P/\langle s_1, \ldots, s_n\rangle$, то $P(t_1, \ldots, t_n)$ — (атомарная) формула.

2. Если A и B — формулы, то $\neg A, (A\&B), (A \vee B)$ — формулы.

3. Если $x \in Var$ и A — формула, то $\forall xA, \exists xA$ — формулы. В тех случаях, когда необходимо явное указание на сорт подквантор-ной переменной, мы будем использовать запись $\forall x/sA, \exists x/sA$.

4. Ничто другое формулой не является.

Определения

Если $P/\langle s_1, \ldots, s_n\rangle$ — предикатный символ, не имеющий вхождений в формулу A, все свободные индивидные переменные которой содержатся среди $\mathbf{x} = \langle x_1/s_1, \ldots, x_n/s_n\rangle$, то определением будем называть формулы вида $\forall\mathbf{x}(P\mathbf{x} \equiv A)$, где "$P\mathbf{x} \equiv A$" — обычное сокращение для $(P\mathbf{x}\&A) \vee (\neg P\mathbf{x}\&\neg A)$.

Соглашение об обозначениях

1. Если S — множество формул языка, то посредством $L(S)$ обозначим язык, который содержит лишь те сорта, индивидные переменные, константы и предикатные символы, которые имеют вхождения в формулы множества S.

2. Запись $\forall\mathbf{x}P\mathbf{x}$ будет использоваться как сокращение для $\forall x_1 \ldots \forall x_n P(x_1, \ldots, x_n)$, а $P\mathbf{t}$ — сокращение для $P(t_1, \ldots, t_n)$.

5. Правила исчисления

Исчисление мы будем строить в виде аналитических таблиц, имеющих вид дерева. Это самый естественный способ представления редукции формул. Обычно различают аналитические таблицы в виде дерева формул, как у Смалльяна, и таблицы в виде дерева множеств формул, как у Фиттинга. В нашем случае более удобны таблицы, узлы которых имеют вид множеств формул. Кроме этого, множествам

формул узла будет сопоставлен их язык. Для этого нам потребуются три метапеременные, которые могут изменяться в процессе построения таблиц:

1. L — язык;
2. Def — множество определений;
3. Mod — множество замкнутых атомарных формул или их отрицаний (модельное множество).

Для представления узлов мы будем использовать запись, содержащую указание на множество целей Aim и метапеременные L, Mod и Def:

$$L, Mod, Def \vdash Aim$$

Знак "$\vdash$" используется, чтобы при записи просто отделить множество целей Aim от языка L, модельного множества Mod и множества определений Def. Никакого дополнительного смысла он не несет. Начальный узел дерева имеет вид $L(Aim), \varnothing, \varnothing \vdash Aim$. Формулы множества Aim замкнуты. Конечной задачей редукции является построение модели, в которой будут истинны все формулы Aim. Правила редукции делятся на четыре группы:

1. правила логических связок;
2. правила кванторов;
3. правила литералов;
4. правила замыкания.

В формулировках правил редукции мы будем указывать метапеременные L, Mod и Def лишь в тех случаях, когда их значения являются условиями применения правил или изменяются в результате такого применения. При формулировке правил редукции запись $\vdash S, A$ следует понимать как $\vdash S \cup \{A\}$, где S — множество формул

(возможно, пустое), а запись S_A следует понимать как сокращение для $S \setminus \{A\}$.

Правила связок

$$(\neg\neg)\ \frac{\vdash S, \neg\neg A}{\vdash S_{\neg\neg A}, A}$$

$$(\&)\ \frac{\vdash S, (A\&B)}{\vdash S_{(A\&B)}, A, B} \qquad (\neg\&)\ \frac{\vdash S, \neg(A\&B)}{\vdash S_{\neg(A\&B)}, \neg A \mid \vdash S_{\neg(A\&B)}, \neg B}$$

$$(\vee)\ \frac{\vdash S, (A \vee B)}{\vdash S_{(A\vee B)}, A \mid \vdash S_{(A\vee B)}, B} \qquad (\neg\vee)\ \frac{\vdash S, \neg(A \vee B)}{\vdash S_{\neg(A\vee B)}, \neg A, \neg B}$$

Правила для логических связок не требуют особых комментариев. Если нам нужно построить модель для множества формул $S \cup \{(A\&B)\}$, то для этого достаточно построить модель для множества формул $S_{(A\&B)} \cup \{A, B\}$. Аналогично, если нам нужно построить модель для множества формул $S \cup \{(A \vee B)\}$, то узел дерева редукций расщепляется на две ветви, т. к. достаточно построить модель для множества формул $S_{(A\vee B)} \cup \{A\}$ или для множества $S_{(A\vee B)} \cup \{B\}$.

Правила кванторов

$$(\forall)\ \frac{L \vdash S, \forall x/sA}{L \cup Const/s \vdash S_{\forall x/sA} \cup \{A[c/x]\}_{c\in Const/s}}$$

$$(\neg\forall)\ \frac{\vdash S, \neg\forall x A}{\vdash S_{\neg\forall xA}, \exists x \neg A}$$

$$(\exists)\ \frac{L \vdash S, \exists x/sA}{\ldots \mid L \cup \{c_i/s\} \vdash S_{\exists x/sA}, A[c_i/x] \mid \ldots} \qquad Const/s = \{c_1, \ldots, c_n\}$$

$$(\neg\exists)\ \frac{\vdash S, \neg\exists x A}{\vdash S_{\neg\exists xA}, \forall x \neg A}$$

Смысл правила $(\forall)$ заключается в том, что если мы хотим снять квантор всеобщности, то необходимо указать набор всех индивидных констант, которые имеют тот же сорт s, что и подкванторная переменная x/s, и добавить их к языку. Формула $\forall x/sA$ замещается множеством всех формул $\{A[c/x]\}_{c\in Const/s}$, получаемых подстановкой всех возможных констант сорта s вместо переменной x. После

первого случая добавления к языку $L \cup Const/s$ всех констант сорта s они уже не изменяются. То есть при их повторном введении в результате применения правила $(\forall)$ к другим формулам с подквантворной переменной x/s имеет место $L = L \cup Const/s$.

Смысл правила $(\exists)$ аналогичен предыдущему, но на этот раз дерево ветвится и потомками узла таблицы является не один, а множество узлов по числу индивидных констант сорта s. При этом нет необходимости добавлять к языку сразу все константы сорта s, а достаточно добавить $L \cup \{c_i/s\}$ лишь ту, на которую снимается квантор. Это очень важное замечание, т. к. служит лазейкой для работы с бесконечными предметными областями.

Правила литералов

Эти правила являются ключевыми в формализации аналитического метода анализа задач.

$$(P.1)\ \frac{Mod, Def \vdash S, P\mathbf{t}}{Mod \cup \{P\mathbf{t}\}, Def \vdash S_{P\mathbf{t}}},$$
при условии $P\mathbf{t} \notin Mod$ и $\forall\mathbf{x}(P\mathbf{x} \equiv A) \notin Def$

$$(P.2)\ \frac{L, Mod, Def \vdash S, P\mathbf{t}}{L \cup L(A), Mod, Def \cup \{\forall\mathbf{x}(P\mathbf{x} \equiv A)\} \vdash S, P\mathbf{t}},$$
при условии $P\mathbf{t} \notin Mod$ и $\forall\mathbf{x}(P\mathbf{x} \equiv A) \notin Def$

$$(\neg P.1)\ \frac{Mod, Def \vdash S, \neg P\mathbf{t}}{Mod \cup \{\neg P\mathbf{t}\}, Def \vdash S_{\neg P\mathbf{t}}},$$
при условии $\neg P\mathbf{t} \notin Mod$ и $\forall\mathbf{x}(P\mathbf{x} \equiv A) \notin Def$

$$(\neg P.2)\ \frac{L, Mod, Def \vdash S, \neg P\mathbf{t}}{L \cup L(A), Mod, Def \cup \{\forall\mathbf{x}(P\mathbf{x} \equiv A)\} \vdash S, \neg P\mathbf{t}},$$
при условии $\neg P\mathbf{t} \notin Mod$ и $\forall\mathbf{x}(P\mathbf{x} \equiv A) \notin Def$

$$(D)\ \frac{Def \cup \{\forall\mathbf{x}(P\mathbf{x} \equiv A)\} \vdash S, P\mathbf{t}}{Def \cup \{\forall\mathbf{x}(P\mathbf{x} \equiv A)\} \vdash S_{P\mathbf{t}}, A[\mathbf{t}/\mathbf{x}]}$$

$$(\neg D)\ \frac{Def \cup \{\forall\mathbf{x}(P\mathbf{x} \equiv A)\} \vdash S, \neg P\mathbf{t}}{Def \cup \{\forall\mathbf{x}(P\mathbf{x} \equiv A)\} \vdash S_{\neg P\mathbf{t}}, \neg A[\mathbf{t}/\mathbf{x}]}$$

$$(El)\ \frac{Mod \cup \{A\} \vdash S, A}{Mod \cup \{A\} \vdash S_A}$$

Если в ходе редукции цели мы пришли к атомарной формуле $P\mathbf{t}$, для предикатного символа которой нет определения вида $\forall \mathbf{x}(P\mathbf{x} \equiv A)$, т. е. два варианта дальнейшего построения дерева. В первом случае $(P.1)$ мы считаем, что дальнейшая редукция не является необходимой, т. к. истинностное значение формулы может быть установлено путем непосредственного соотнесения с фактическим состоянием мира. Поэтому мы добавляем формулу к множеству Mod. Во втором случае $(P.2)$ мы считаем необходимым редуцировать данную формулу и для этого принимаем определение предикатного символа $\forall \mathbf{x}(P\mathbf{x} \equiv A)$. Приняв его, мы должны расширить язык сортами, индивидными переменными, константами и предикатными символами, которые имеют вхождения в дефиниенс определения, т. е. в формулу A.

Смысл правил $(\neg P.1)$ и $(\neg P.2)$ аналогичен предыдущим.

Правила (D) и $(\neg D)$ — это замена предикатного символа на основании ранее принятого определения.

Правило (El) говорит, что если целевая формула A уже содержится в модельном множестве Mod, то ее можно удалить из множества целей.

Правила замыкания

$$(\varnothing.1)\ \frac{\vdash S, A, \neg A}{\varnothing}$$

$$(\varnothing.2)\ \frac{Mod \cup \{P\mathbf{t}, \neg P\mathbf{t}\} \vdash S}{\varnothing}$$

Смысл правил замыкания достаточно очевиден. Если мы пришли к противоречию, то ветвь дерева считается ***замкнутой*** и исключается из дальнейшего рассмотрения.

Ветвь дерева, конечный узел которой имеет вид $L, Mod, Def \vdash \varnothing$, будем называть ***завершенной***, поскольку к нему не применимы никакие правила редукции.

Будем говорить, что ***цель достижима***, если дерево редукций имеет хотя бы одну завершенную ветвь. ***Цель недостижима***, если все ветви дерева редукций замкнуты.

6. Модели

Моделью будем называть пару вида $M = \langle Mod, Def \rangle$, где

1. Mod — некоторое множество замкнутых атомарных формул или их отрицаний;
2. Def — множество определений;
3. Если $\forall \mathbf{x}(P\mathbf{x} \equiv A) \in Def$, то ни для каких термов $\mathbf{t}$ не верно, что $P\mathbf{t} \in Mod$ или $\neg P\mathbf{t} \in Mod$.

Определим отношение $M \vDash A$ — *«формула A истинна в модели M»*, где A не содержит свободных переменных.

1. $\langle Mod, Def \rangle \vDash P\mathbf{t} \quad \Leftrightarrow \quad (\forall \mathbf{x}(P\mathbf{x} \equiv A) \notin Def$ и $P\mathbf{t} \in Mod)$ или $(\forall \mathbf{x}(P\mathbf{x} \equiv A) \in Def$ и $\langle Mod, Def \rangle \vDash A[\mathbf{t}/\mathbf{x}])$
2. $\langle Mod, Def \rangle \vDash \neg P\mathbf{t} \quad \Leftrightarrow \quad (\forall \mathbf{x}(P\mathbf{x} \equiv A) \notin Def$ и $\neg P\mathbf{t} \in Mod)$ или $(\forall \mathbf{x}(P\mathbf{x} \equiv A) \in Def$ и $\langle Mod, Def \rangle \vDash \neg A[\mathbf{t}/\mathbf{x}])$
3. $M \vDash (A \& B) \quad \Leftrightarrow \quad M \vDash A$ и $M \vDash B$
4. $M \vDash \neg(A \& B) \quad \Leftrightarrow \quad M \vDash \neg A$ или $M \vDash \neg B$
5. $M \vDash (A \vee B) \quad \Leftrightarrow \quad M \vDash A$ или $M \vDash B$
6. $M \vDash \neg(A \vee B) \quad \Leftrightarrow \quad M \vDash \neg A$ и $M \vDash \neg B$
7. $M \vDash \forall x/sA \quad \Leftrightarrow \quad M \vDash A[c/x]$ для каждой константы c/s
8. $M \vDash \neg\forall xA \quad \Leftrightarrow \quad M \vDash \exists x\neg A$
9. $M \vDash \exists x/sA \quad \Leftrightarrow \quad M \vDash A[c/x]$ для некоторой константы c/s
10. $M \vDash \neg\exists xA \quad \Leftrightarrow \quad M \vDash \forall x\neg A$

7. Теорема адекватности

Теорема 1. *Пусть* $L(Aim), \varnothing, \varnothing \vdash Aim$ *— начальный узел дерева, а* $L, Mod, Def \vdash \varnothing$ *— конечный узел одной из завершенных ветвей. Тогда для всякой формулы* $A \in Aim$ *имеет место* $\langle Mod, Def \rangle \models A$.

Доказательство проводим индукцией по построению дерева.

Доказательство. Выделим ветвь дерева с конечным узлом $L, Mod, Def \vdash \varnothing$, и покажем, что для всякого узла $L', Mod', Def' \vdash S'$ этой ветви и всякой формулы $B \in S'$ имеет место $\langle Mod, Def \rangle \models B$.

Базис индукции

Так как узел $L, Mod, Def \vdash \varnothing$ является конечным, то тривиальным образом для каждой формулы $B \in \varnothing$ имеет место $\langle Mod, Def \rangle \models B$.

Индукционный шаг

Рассмотрим узел $L', Mod', Def' \vdash S', B$ в предположении, что для всех нижестоящих узлов наше утверждение выполняется.

Случай 1. Формула B имеет вид $\forall x/sD$ и к ней было применено правило $(\forall)$

$$\frac{L', Mod', Def' \vdash S', \forall x/sD}{L' \cup Const/s, Mod', Def' \vdash S'_{\forall x/sD} \cup \{D[c/x]\}_{c \in Const/s}}$$

Тогда непосредственно следующий узел для всех $c \in Const/s$ содержит формулы вида $D[c/x]$, и по индуктивному опущению для каждой из них имеет место $\langle Mod, Def \rangle \models D[c/x]$. Отсюда по определению истинности в модели получаем $\langle Mod, Def \rangle \models \forall x/sD[c/x]$.

Случай 2. Формула B имеет вид $\exists x/sD$ и к ней было применено правило $(\exists)$.

$$\frac{L', Mod', Def' \vdash S', \exists x/sD}{\ldots \mid L' \cup \{c_i/s\}, Mod', Def' \vdash S'_{\exists x/sD}, D[c_i/x] \mid \ldots}$$

Тогда анализируемой ветви дерева принадлежит один из непосредственно следующих узлов вида $L' \cup \{c_i/s\}, Mod', Def' \vdash S'_{\exists x/sD}, D[c_i/x]$ для некоторой константы c_i/s. По индуктивному допущению имеет место $\langle Mod, Def\rangle \models D[c_i/x]$. Отсюда по определению истинности в модели получаем $\langle Mod, Def\rangle \models \exists x/sD$.

Случай 3. Формула B имеет вид $P\mathbf{t}$ и к ней было применено правило $(P.1)$ при выполнении условия $P\mathbf{t} \notin Mod$ и $\forall\mathbf{x}(P\mathbf{x} \equiv A) \notin Def$

$$\frac{Mod', Def' \vdash S', P\mathbf{t}}{Mod' \cup \{P\mathbf{t}\}, Def' \vdash S'_{P\mathbf{t}}}$$

Поскольку $Mod' \cup \{P\mathbf{t}\} \subseteq Mod$ то $\langle Mod, Def\rangle \models P\mathbf{t}$.

Случай 4. Формула B имеет вид $P\mathbf{t}$ и к ней было применено правило $(P.2)$ при выполнении условия $P\mathbf{t} \notin Mod$ и $\forall\mathbf{x}(P\mathbf{x} \equiv A) \notin Def$

$$\frac{L', Mod', Def' \vdash S', P\mathbf{t}}{L' \cup L(A), Mod', Def' \cup \{\forall\mathbf{x}(P\mathbf{x} \equiv A)\} \vdash S', P\mathbf{t}}$$

По индуктивному допущению $\langle Mod, Def\rangle \models P\mathbf{t}$.

Случай 5. Формула B имеет вид $\neg P\mathbf{t}$ и к ней было применено правило $(\neg P.1)$. Рассматривается аналогично случаю 3.

Случай 6. Формула B имеет вид $\neg P\mathbf{t}$ и к ней было применено правило $(\neg P.2)$. Рассматривается аналогично случаю 4.

Случай 7. Формула B имеет вид $P\mathbf{t}$ и к ней было применено правило (D).

$$\frac{L', Mod', Def' \cup \{\forall\mathbf{x}(P\mathbf{x} \equiv A)\} \vdash S', P\mathbf{t}}{L', Mod', Def' \cup \{\forall\mathbf{x}(P\mathbf{x} \equiv A)\} \vdash S'_{P\mathbf{t}}, A[\mathbf{t}/\mathbf{x}]}$$

По индуктивному допущению имеет место $\langle Mod, Def\rangle \models A[\mathbf{t}/\mathbf{x}]$. Поскольку $Def' \cup \{\forall\mathbf{x}(P\mathbf{x} \equiv A) \subseteq Def$, то $\forall\mathbf{x}(P\mathbf{x} \equiv A) \in Def$, и по определению истинности в модели получаем $\langle Mod, Def\rangle \models P\mathbf{t}$.

Случай 8. Формула B имеет вид $\neg P\mathbf{t}$ и к ней было применено правило $(\neg D)$. Рассматривается аналогично случаю 7.

Остальные случаи для логических связок, кванторов $(\neg\forall)$, $(\neg\exists)$ и (El) тривиальны. □

8. Заключение

Построенное исчисление является формализацией проблемы решения задач как последовательного уточнения предикатов и их редукции к подзадачам. В результате такой редукции мы приходим к моделям, которые либо являются решением задачи, либо содержат указание на то, как следует изменить текущее состояние мира, чтобы задача была решена.

Мы ограничились рассмотрением моделей с конечными индивидными областями, поскольку ориентировались в первую очередь на практические задачи, где такие ограничения оправданны. В то же время предложенный метод позволяет решать задачи, относящиеся и к бесконечным предметным областям. Это возможно в тех случаях, когда в ходе построения дерева редукции не применяется правило ($\forall$). Лишь оно может привести к узлу, содержащему бесконечное множество целевых формул по числу констант, именующих бесконечное множество индивидов предметной области. Если же это правило не применяется, то все узлы дерева редукции являются конечными объектами. Бесконечными могут быть лишь ветвления дерева в результате применения правила ($\exists$), но это не является препятствием, поскольку для решения задачи достаточно найти хотя бы одну завершенную ветвь, а это возможно методом поиска вглубь.

Все, кто знаком с методами логического программирования на языке Пролог, могли обратить внимание на то, что представленное исчисление редукций в достаточно общем виде формализует методологию написания логических программ. А именно, планирование и написание логических программ по методу сверху-вниз. Как и в исчислении редукций, программист дает определения предикатам, вводит в рассмотрение индивиды различных сортов, расширяет язык новыми предикатами, чтобы в дальнейшем дать определения уже им, как и в исчислении редукций, использует смысл логических связок для того, чтобы разбить программу на подпрограммы. Отличия лишь в деталях, связанных с эффективной реализацией на компьютере. Для этого в чистом Прологе используются Хорновы формулы, мы же взяли полный язык логики предикатов первого порядка. Множество определений, которые принимаются в ходе построения дерева

редукций, является прямым аналогом логической программы, которая в будущем может использоваться для решения подобных задач. Таким образом, изначально творческая задача переходит в разряд рутинных, которые будут решаться стандартным образом на основе набора однажды принятых определений.

Литература

[1] *Декарт Р.* Правила для руководства ума // *Декарт Р.* Соч.: в 2 т. Т. 1. М.: Мысль, 1988. С. 77–153.

[2] *Непейвода Н.Н., Свириденко Д.Т.* К теории синтеза программ // Математическая логика и теория алгоритмов М.: Наука, 1982. С. 159–175.

[3] *Пойя Дж.* Математика и правдоподобные рассуждения. М.: Наука, 1975. 464 с.

[4] *Пойя Дж.* Математическое открытие. М.: Наука, 1976. 448 с.

[5] *Саати Т.* Принятие решений. Метод анализа иерархий. М.: Радио и связь, 1993. 278 с.

[6] *Саати Т., Кернс К.* Аналитическое планирование. Организация систем. М.: Радио и связь, 1991. 224 с.

[7] *Смирнов В.А.* Творчество, открытие и логические методы поиска доказательства // Логико-философские труды В.А. Смирнова. М.: Эдиториал УРСС, 2001. С. 438–447.

[8] *Уинстон П.* Искусственный интеллект. М.: Мир, 1980. 520 с.

[9] *Целищев В.В., Бессонов А.В.* Две интерпретации логических систем. Новосибирск: Наука, 1979. 269 с.

V.I. Shalack

Analytical Approach to Problem Solving

Shalack Vladimir Ivanovich
Department of logic, Institute of Philosophy of Russian Academy of Sciences
12/1 Goncharnaya Str., Moscow, 109240, Russian Federation
E-mail: shalack@gmail.com

The work is devoted to the logical analysis of the problem solving. Typically, in each task we highlight the conditions and goals that we have to find or build. In this case the solution of the problem is seen as a kind of deduction from goals to the conditions. This representation of problems and their solutions is too narrow. In actual practice, a task or goal is often formulated in quite general terms as a wish. For example, the task to build a railway between the two cities. Sufficient conditions for the solution of this problem are initially unclear and should be found. For this kind of problems their solution can be represented as a gradual refinement of goals and their reduction to a simpler sub-goals. The methods by which we produce clarification of goals, we took from the theory of definitions. In this paper we construct a calculus in the form of analytical tables, which allows us to represent the whole process algorithmically.

Keywords: Problem solving, logical reduction, analytical tables, theory of definitions

References

[1] Dekart, R. "Pravila dlya rukovodstva uma" [Rules for the Direction of the Natural Intelligence], in: R. Dekart, *Sochineniya v dvukh tomakh* [Works in two volumes], T. 1. Moscow: Mysl', 1988, pp. 77–153. (In Russian)

[2] Nepeivoda, N.N., Sviridenko, D.T. "K teorii sinteza programm" [To the theory of program synthesis], in: *Matematicheskaya logika i teoriya algoritmov* [Mathematical logic and theory of algorithms]. Moscow: Nauka, 1982, pp. 159–175. (In Russian)

[3] Poiya, Dzh. *Matematika i pravdopodobnye rassuzhdeniya* [Mathematics and plausible reasoning]. Moscow: Nauka, 1975. 464 pp. (In Russian)

[4] Poiya, Dzh. *Matematicheskoe otkrytie* [Mathematical discovery]. Moscow: Nauka, 1976. 448 pp. (In Russian)

[5] Saati, T. *Prinyatie reshenii. Metod analiza ierarkhii* [The Analytic Hierarchy Process]. Moscow: Radio i svyaz', 1993. 278 pp. (In Russian)

[6] Saati, T., Kerns, K. *Analiticheskoe planirovanie. Organizatsiya sistem* [Analytical Planning; The Organization of Systems]. Moscow: Radio i svyaz', 1991. 224 pp. (In Russian)

[7] Smirnov, V.A. "Tvorchestvo, otkrytie i logicheskie metody poiska dokazatel'stva" [Creativity, discovery and logical methods for proof search], in: *Logiko-filosofskie trudy V.A. Smirnova* [The logical and philosophical works of V.A. Smirnov] Moscow: Editorial URSS, 2001, pp. 438–447. (In Russian)

[8] Winston, P. *Iskusstvennyi intellekt* [Artificial intellegence], Moscow: Mir, 1980. 520 pp. (In Russian)

[9] Tselishchev, V.V., Bessonov, A.V. *Dve interpretatsii logicheskikh sistem* [Two interpretations of logical systems], Novosibirsk: Nauka, 1979. 269 pp. (In Russian)