УДК 519.685.1

КОНТРОЛЬ ДИАЛОГА ОБЪЕКТНЫХ ПРОГРАММ НА ОСНОВЕ ПОЗИТИВНО-ОБРАЗОВАННЫХ ФОРМУЛ

М.И. Бутаков, В.И. Курганский

ГОУ ВПО «Иркутский государственный университет», г. Иркутск

Рецензент д-р физ.-мат. наук, профессор Ф.И. Иванов

Ключевые слова и фразы: контроль диалога; КС-грамматики; позитивно-образованные формулы; синтез транслирующих программ; формальный язык.

Аннотация: Разработана логико-лингвистическая модель диалога объектных программ, которая позволяет задавать требования к диалогу, синтезировать программы диагностирования и исправления ошибок. Для построения модели и ее интеграции с диалоговыми программами и программами контроля диалога разработано инструментальное средство.

Позитивно-образованные формулы (**ПОФ**) представляют собой аппарат автоматического доказательства теорем [5, 16]. Рассмотрим применение ПОФ для решения задачи контроля диалога. Основу такого применения составляют представления грамматического вывода и распознавания входной цепочки процессом доказательства (вывода) позитивно-образованной формулы. Для решения задачи контроля диалога разработаны транслирующие ПОФ (**ТПОФ**), при доказательстве которых синтезируются так называемые транслирующие программы. Результатом исполнения транслирующих программ является преобразование (трансляция) заданной входной цепочки или диагностика и обработка ошибок.

Диалог в современных прикладных программных системах зачастую представляет собой цепочки взаимосвязанных входных воздействий (нажатий кнопок клавиатуры и мыши, перемещения мыши и т.п.). Каждое входное воздействие порождает событие, которое может быть обработано одним из объектов программной системы. Штатные средства программи-

Бутаков М.И. – аспирант кафедры «Теория алгоритмов и программирования» Института математики, экономики и информатики, e-mail: ime@math.isu.ru; Курганский В.И. – кандидат физико-математических наук, доцент кафедры «Теория алгоритмов и программирования» Института математики, экономики и информатики, ИркутГУ, г. Иркутск.

рования обработки событий в современных системах объектно-ориентированного программирования не обеспечивают анализа контекста входного воздействия [10].

Первые попытки формально специфицировать входные воздействия в диалоговых системах появились в 80-х годах XX столетия [13]. В работе [11] представлен обзор различных моделей спецификации входных воздействий. Наибольшую популярность получила модель, основанная на формальных грамматиках и предложенная Рейснером [15]. Основная идея модели заключается в описании правильных последовательностей входных воздействий с помощью формальных грамматик [14].

Ниже представлен аппарат транслирующих позитивно-образованных формул (п. 1); задача контроля диалога сводится к задаче трансляции (п. 2); описание средства проектирования и программной реализации контроля диалога (п. 3).

Отметим, что использование методик искусственного интеллекта при контроле диалога предложено Р.М. Янгом [17].

1. Транслирующие позитивно-образованные формулы

Рассмотрим задачу грамматического разбора как задачу синтеза транслирующей программы, которая преобразует заданную входную цепочку, а также диагностирует ошибки в этой цепочке.

Теорема о существовании такой программы в чистом исчислении предикатов [5, 9] имеет вид

$$\begin{gathered}((\forall X_1 : (P_1(X_1) \to \exists Y_1 : R_1(X_1, Y_1))) \wedge \ldots \wedge (\forall X_n : (P_n(X_n) \to \exists Y_n : R_n(X_n, Y_n)))) \to \\ \to (\forall X : (P(X) \to \exists Y : R(X, Y))).\end{gathered} \quad (1)$$

Поясним обозначения. Формула $\forall X : P(X) \to \exists Y : R(X, Y)$ является спецификацией транслирующей программы, а формулы $\forall X_1 : (P_1(X_1) \to \exists Y_1 : R_1(X_1, Y_1)), \ldots, \forall X_n : (P_n(X_n) \to \exists Y_n : R_n(X_n, Y_n))$ – спецификациями транслирующих подпрограмм.

В символике позитивно-образованных формул эта теорема имеет вид

$$\forall X : P(X) \begin{cases} \forall X_1 : P_1(X_1) \exists Y_1 : R_1(X_1, Y_1) \ldots \\ \forall X_n : P_n(X_n) \exists Y_n : R_n(X_n, Y_n) \\ \forall Y : R(X, Y) \exists \text{False}. \end{cases} \quad (2)$$

Элементы $P_1(X_1), \ldots, P_n(X_n), P(X)$ и $R_1(X_1, Y_1), \ldots, R_n(X_n, Y_n), R(X, Y)$ формулы (2) являются конъюнктами, которые строятся из атомов, символизирующих возможность разбора элемента входной цепочки, успешный разбор элемента входной цепочки, разбор языковой конструкции, обозначенной нетерминальным символом грамматики, а также успешное завершение разбора.

В случае разбора входной цепочки слева направо по одному символу и ограничения входной строки $C = c_1, ..., c_n$ справа специальным символом конца входной строки #, спецификация транслирующей программы в чистом исчислении предикатов имеет вид $\forall u : u = 1, P(c_1, u), Q(A) \to \exists : RT(\#)$, где u – переменная вывода; A – аксиома грамматики; $Q(A)$ – атом, символизирующий разбор соответствующей языковой конструкции; $P(c_1,u)$ – атом, символизирующий возможность разбора входного символа c_1, расположенного в позиции u входной цепочки; $RT(\#)$ – атом, символизирующий успешный разбор символа конца строки #.

Спецификации транслирующих подпрограмм $\forall X_i : P_i(X_i) \to$ $\to \exists Y_i : R_i(X_i, Y_i)$ строятся по правилам грамматики или по условиям, исключающим применение правил грамматики при разборе. В последнем случае механизм трансляции на основе ПОФ обеспечивает диагностирование ошибки во входной строке и/или исправление ошибки. Все это осуществляется исполнением соответствующей транслирующей подпрограммы.

Особенности транслирующих позитивно-образованных формул также заключаются в том, что они включают данные о порядке следования входных символов в анализируемой цепочке и нелогические элементы – команды запуска транслирующих подпрограмм. Команды запуска транслирующих подпрограмм связываются со специфицирующими их постусловиями.

Рассмотрим пример. Пусть дана регулярная грамматика $G = \langle V_T, V_N, P, S \rangle$, где $V_T = \{0, 1, \#\}$, $V_N = \{A, B\}$, $S = A$, $P = \{A \to 1B, B \to 1B,$ $B \to 0B, B \to \#\}$. Данная грамматика описывает язык цепочек ненулевой длины, составленных из 0 и 1, начинающихся символом 1 и заканчивающихся символом #.

Транслирующая позитивно-образованная формула, для анализа цепочки 01# по данной грамматике и синтеза транслирующей программы, которая диагностирует ошибки в случае некорректной цепочки, имеет вид

$$\exists u : u = 1, P(0, u), Q(A) \begin{cases} \forall s : P(1, s), Q(A) \exists : R(1, s), Q(B) \\ \forall s : P(1, s), Q(B) \exists : R(1, s), Q(B) \\ \forall s : P(0, s), Q(B) \exists : R(0, s), Q(B) \\ \forall s : PT(\#), Q(B) \exists : RT(\#) \\ \forall s : P(0, s), Q(A) \exists : RT(\#) [\text{Error}(0, s)] \\ \forall : RT(\#) \exists : \text{False} \\ \forall s : s = 1, R(0, s) \exists : s = 2, P(1, s) \\ \forall s : s = 2, R(1, s) \exists : PT(\#) \end{cases}$$

Часть ТПОФ, следующую за квантором существования и до фигурной скобки, называют ее базой U_B. Элементы, расположенные за фигурной скобкой и выделяемые кванторами всеобщности, называют вопросами. Квадратными скобками выделен нелогический элемент – команда запуска транслирующей подпрограммы Error для диагностирования ошибки.

Доказательство позитивно-образованной формулы представляет собой ее многократные преобразования по правилу ω [5, 16]. Применение правила ω заключается в поиске подходящего вопроса и соответствующей обработке базы. В нашем случае параметрами однократного применения правила ω являются подстановка $u \to s$ и один из вопросов. Здесь и далее подстановки будем обозначать символом θ.

Исходя из особенностей задач трансляции [1], выделим частный случай конструктивного фрагмента исчисления ПОФ путем дополнения стандартного правила вывода ω [16] модификатором вывода τ. Использование модификатора предотвращает зацикливание и устраняет неопределенность при доказательстве ПОФ.

Элементы модификатора τ обозначены ρ, ω_1 и ω_2.

Элемент ρ модификатора τ заключается в разделении базы U_B позитивно-образованной формулы F на два конъюнкта – конъюнкт синтеза K_S и конъюнкт вывода K_D. Конъюнкт K_S составляют атомы, уже использованные при преобразовании по правилу ω формулы F, а K_D – атомы которые можно использовать в данный момент при применении правила ω. После каждого успешного применения правила ω изменяются конъюнкт синтеза и конъюнкт вывода. Конъюнкт синтеза дополняется атомами из конъюнкта вывода: а именно, в конъюнкт синтеза K_S добавляются только те атомы, которые участвовали при обработке базы U_B и одного из вопросов U_Q. В конъюнкт вывода копируется постусловие обрабатываемого по правилу ω вопроса. Формальные параметры постусловия при этом должны быть заменены переменными вывода в соответствии с подстановкой θ.

Элемент ω_1 модификатора τ заключается в поиске подходящего вопроса для обработки по правилу ω путем просмотра конъюнкта вывода слева направо. Глубина просмотра при проверке выполнимости вопроса определяется мощностью множества атомов, составляющих конъюнкт предусловия сканируемого вопроса. Критерием применимости правила ω к данному вопросу является принадлежность каждого атома предусловия вопроса, обработанного подстановкой θ, конъюнкту вывода.

Элемент ω_2 модификатора τ заключается в пополнении конъюнкта вывода атомами постусловия, примененного в рамках правила ω вопроса, путем их приписывания к конъюнкту вывода только слева.

Таким образом, однократное применение правила ω с учетом модификатора τ заключается в следующей обработке ПОФ:

1) применение элемента ω_1 модификатора τ и выполнение подстановки $u \to s$ для выделенного вопроса Q;

2) применение элемента ω_2 модификатора τ, то есть копирование постусловия вопроса Q в конъюнкт вывода K_D в случае принадлежности атомов, составляющих конъюнкт предусловия обрабатываемого вопроса Q, конъюнкту вывода K_D;

3) применение элемента ρ модификатора τ, то есть перемещение атомов предусловия обработанного вопроса Q из конъюнкта вывода K_D в конъюнкт синтеза K_S в случае принадлежности атомов, составляющих

конъюнкт предусловия обрабатываемого вопроса Q, конъюнкту вывода K_D.

Невозможность применения элемента ω_1 модификатора вывода τ означает недоказуемость ПОФ. Случай, когда конъюнкт вывода содержит False, означает, что позитивно-образованная формула успешно доказана.

Процесс доказательства без возвратов транслирующей позитивно-образованной формулы можно представить последовательностью состояний ее базы. Для рассматриваемого примера эта последовательность имеет следующий вид

$$u = 1, P(0,u), Q(A) \Rightarrow u = 1, RT(\#)[\text{Error}(0,1)] \Rightarrow u = 1, \text{False}.$$

Транслирующая программа представляет собой последовательность нелогических элементов, попавших в процессе вывода в конъюнкт синтеза. В нашем примере это программа, которая состоит из одной команды Error (0, 1).

В [3, 4, 8] представлены алгоритмы построения позитивно-образованных формул по заданным контекстно-свободным грамматикам – LL(1), регулярным и LR(1). Там же доказана эквивалентность грамматического разбора, выполняемого по этим грамматикам, процессу доказательства (вывода) построенной соответствующим алгоритмом позитивно-образованной формулы. Алгоритмы построения ПОФ по грамматикам указанных типов разработаны на основе известных алгоритмов преобразования этих грамматик в соответствующие распознающие автоматы.

2. Задача контроля диалога как задача трансляции формальных языков

Программная реализация диалога представляет собой комплект экземпляров визуальных классов, называемых формами и элементами управления. Обычно эти экземпляры обеспечивают обзор и редактирование значения одного из своих свойств. Редактирование значений свойств диалоговых объектов осуществляется выполнением процедур обработки событий, которые как раз и возникают в связи с манипуляциями пользователя клавиатурой, мышью и другими входными устройствами.

С одной стороны, диалог представляет собой цепочку входных воздействий пользователя (нажатий клавиш клавиатуры и кнопок мыши, перемещения мыши и т.п.) и может рассматриваться как цепочка символов, принадлежащая определенному формальному языку. С другой стороны, при диалоге ассоциации элементов управления и форм могут рассматриваться как языковые конструкции более высокого уровня. Правильность формирования таких конструкций иногда требуется контролировать.

Таким образом, формальная лингвистическая модель диалога имеет двухуровневую структуру и состоит из двух разновидностей формальных грамматик. Нижний уровень этой структуры составляют грамматики, которые задают правила формирования редактируемых значений. Верхний

уровень формальной модели, описывающей диалог, составляет КС-грамматика, которая задает требования к ассоциациям элементов управления и последовательности формирования этих ассоциаций. Эта модель подобна формальной модели языков программирования, предложенной в [1]. Так как формальная лингвистическая модель диалога реализована ТПОФ, назовем ее логико-лингвистической моделью.

Применение транслирующих позитивно-образованных формул позволяет использовать единый механизм и для распознавания принадлежности цепочек входных воздействий заданному формальному языку, и для диагностирования ошибок в этих цепочках независимо от типа используемых формальных грамматик.

Моделирование диалога формальными грамматиками затруднено тем обстоятельством, что элементы множеств их терминальных символов соответствуют событиям. События обрабатываются объектами, которые реализуют формы и элементы управления. Так как событие дополнительно характеризуется набором значений его параметров, то зачастую в логико-лингвистической модели диалога оно может быть представлено не одним, а несколькими терминальными символами или правилом построения этих символов из обозначения события и значений его параметров. Нетерминальные символы формальных грамматик, описывающих диалог, вводятся обычным для формальных грамматик способом.

3. Проектирование и программная реализация средств контроля диалога

Традиционно программное обеспечение контроля диалога разрабатывают в виде процедур обработки событий, сопровождающих диалог. В случае применения логико-лингвистической модели диалога требуются:

– разработка элементов диалога и диалога в частности, как формальных языков;

– программная реализация анализа цепочек входных воздействий на предмет их принадлежности формальным языкам, которые являются моделями элементов диалога;

– интеграция программ анализа цепочек входных воздействий с объектной реализацией диалоговой системы.

Для решения этих задач разработан диалоговый компонент синтеза транслирующих программ (**КСТП**). Под компонентом понимается визуальный объект, который применяется как на этапе проектирования объектной программы в системе визуального объектно-ориентированного программирования, так и при исполнении проектируемой программы. В нашем случае под проектируемой программой понимается диалоговая система, требования к допустимым цепочкам входных значений в которой заданы двухуровневой системой формальных языков.

Компонент синтеза транслирующих программ предназначен для диалогового построения и отладки транслирующей позитивно-образованной

формулы на этапе разработки диалоговой системы и для анализа цепочек входных воздействий при исполнении программ, составляющих диалоговую систему.

Транслирующая позитивно-образованная формула может быть получена преобразованием заданной формальной грамматики или задана непосредственно редактированием.

Результат анализа цепочки входных воздействий представляется в виде так называемой транслирующей программы, которая синтезируется компонентом в процессе контроля входной цепочки. В случае правильной входной цепочки транслирующая программа может быть вырожденной – «пустой». В случае входной цепочки с ошибками транслирующая программа представляет собой последовательность команд запуска транслирующих подпрограмм, которые диагностируют и корректируют ошибку. Корректировка ошибки заключается в установке умалчиваемых и подразумеваемых значений редактируемых при диалоге свойств.

Для конкретного диалогового приложения транслирующие подпрограммы группируются в класс, на который указывает одно из свойств компонента. Ниже в таблице приведены основные функциональные члены компонента.

Интеграция компонента синтеза транслирующих программ с диалоговой системой осуществляется включением в ее объектную модель соответствующих экземпляров компонента, запуском методов анализа входных воздействий и исполнения синтезированной транслирующей программы при обработке определенных событий в процессе диалога.

Основные функциональные члены КСТП

Название	Тип значения	Описание
Свойство Designer	Форма	Вызов диалога редактирования ТПОФ
Свойство InputString	Коллекция	Предоставляет доступ для работы с входной строкой
Метод Prove	Boolean	Осуществляет доказательство ПОФ. В случае успешного доказательства возвращает True
Свойство SynExecProve		Задает способ синтеза и исполнения транслирующей программы – пошаговый или после завершения доказательства ПОФ
Метод Synthesize		Осуществляет синтез программы. В случае успеха возвращает True, иначе False
Метод Execute		Осуществляет исполнение синтезированной программы. В случае успеха возвращает True

Компонент синтеза транслирующих программ разработан для платформы .Net и опробован при разработке специализированной диалоговой системы в среде Microsoft Visual Studio.Net.

Заключение

Проектирование и разработка программного обеспечения контроля диалога на основе его логико-лингвистической модели в виде транслирующих позитивно-образованных формул представляется более предпочтительными по сравнению с автоматным подходом к проектированию интерфейса [10]. Причина заключается в том, что не всякий диалог можно описать конечным автоматом. В то же время транслирующая позитивно-образованная формула на основе регулярной грамматики является одной из реализаций конечного автомата. Транслирующие позитивно-образованные формулы обеспечивают контроль и более сложных лингвистических конструкций, создаваемых из современных диалоговых объектов.

Непосредственное применение моделей трансляции, разрабатываемых с помощью компиляторов [12], проигрывает в технологичности визуальному проектированию контроля диалога и интеграции программ контроля диалога с прикладной диалоговой системой с помощью компонента построения транслирующих позитивно-образованных формул и синтеза транслирующих программ.

Аппарат транслирующих позитивно-образованных формул успешно опробован при решении двух практических задач. Одна из них – задача контроля диалога в специализированной диалоговой системе «Лесной склад». Программное обеспечение в этой системе эксплуатируется на мобильном вычислительном устройстве и служит для сбора данных о поступлении и отгрузке лесоматериалов, а также для обеспечения контрольных мероприятий в связи с хранимыми и отгруженными со склада лесоматериалами [2].

Транслирующие позитивно-образованные формулы составили основу учебной технологии разработки учебных трансляторов. Технология внедрена и эксплуатируется несколько лет в институте математики, экономики и информатики Иркутского государственного университета [6, 7]. Технология предназначена для разработки учебных трансляторов для формальных языков, лексика которых задается регулярными грамматиками, а синтаксис – LL(1)-грамматикой. Грамматики учебного языка задаются транслирующими позитивно-образованными формулами, которые в виде экземпляров КСТП внедряются в разрабатываемый учебный транслятор и обеспечивают лексический и синтаксический анализ.

Список литературы

1. Ахо, А. Компиляторы : принципы, технологии и инструменты : пер. с англ. / А. Ахо, Р. Сети, Д. Ульман. – М. : Вильямс, 2001. – 768 с.

2. Бутаков, М.И. Компонент построения и исполнения транслирующих программ на основе позитивно-образованных формул / М.И. Бутаков // Прикл. информац. технологии и системы – Иркутск : Изд-во Иркут. гос. ун-та, 2009. – С. 4–27.

3. Бутаков, М.И. О решении задачи трансляции планированием вычислений на основе позитивно-образованных формул / М.И. Бутаков, В.И. Курганский // Системы упр. и информац. технол. – 2007. – № 3.1(29). – С. 120–123.

4. Бутаков, М.И. Решение учебных задач трансляции на основе позитивно-образованных формул / М.И. Бутаков, О.В. Курганская. – Системы упр. и информац. технологии. – 2008. – № 3.1(33). – С. 124–128.

5. Интеллектное управление динамическими системами / С.Н. Васильев [и др.]. – М. : Физматлит, 2000. – 352 с.

6. Курганский, В.И. Лексический анализ на основе регулярных грамматик и конечных автоматов / В.И. Курганский, М.И. Бутаков. – Иркутск : Изд-во Иркут. гос. ун-та, 2006. – 20 с.

7. Курганский, В.И. Синтаксический анализ на основе LL(1)-грамматик и автоматов с магазинной памятью / В.И. Курганский, М.И. Бутаков. – Иркутск : Изд-во Иркут. гос. ун-та, 2006. – 32 с.

8. Севостьянова, Д.В. Восходящий грамматический разбор на основе позитивно-образованных формул / Д.В. Севостьянова, В.И. Курганский // Материалы науч. студен. конф., Иркутск, 2009 г. / Вестн. Иркут. гос. ун-та. – Иркутск, 2009. – С. 163–165.

9. Тыугу, Э.Х. Концептуальное программирование / Э.Х. Тыугу. – М. : Наука, 1984. – 256 с.

10. Шалыто, А.А. Реализация автоматов при программировании событийных систем / А.А. Шалыто, Н.И. Туккель // Программист. – 2002. – № 4. – С. 74–80.

11. Abowd, G.D. User Interface Languages : A Survey of Existing Methods / G.D. Abowd, J. Bowen. – Technical Report PRG-TR-5-89, Programming Research Group, Oxford University, October, 1989 – P. 65.

12. Anthony A. Aaby. Compiler Construction using Flex and Bison / Aaby A. Anthony // Walla Walla College. February 25. – 2004. – 96 p.

13.Harrison, M.D. A review of formalisms for describing interactive behaviour / M.D. Harrison, D.J. Duke // In Software Engineering and Human-Computer Interaction, volume 896 of Lecture Notes in Computer Science, Springer. – Verlag, 1995. – P. 49–75.

14. A Scalable Formal Method for Design and Automatic Checking of User Interfaces / Berstel Jean [and others] // ACM Transactions on Software Engineering and Methodology. – 2005. – Vol. 14, No 2. – P. 124–167.

15. Phyllis Reisner. Formal Grammar and Human Factors Design of an Interactive Graphics Sysyrem / Reisner Phyllis. – Iee Transactions on Software Engineering. – March, 1981. – Vol. SE-7, No 2. – P. 229–240.

16. Vassilyev, S.N. Intelligent Control via New Efficient Logics / S.N. Vassilyev, A.V. Davydov, A.K. Zherlov // Proc. of the 17th IFAC World Congress. (Korea), 2008 / Seoul, 2008. – P. 13713–13718.

17. Young, R.M. How would your favorite user model cope with these scenarios? / R.M. Young, P. Barnard, T. Simon // SIGCGI Bulletin. – 1989. – Vol. 20. – P. 51–55.

Dialogue Control of Object Programs Based on Positively Formed Formula

M.I. Butakov, V.I. Kurgansky

Institute of System Dynamics and Control Theory of Siberia Department of RAS, Irkutsk State University, Irkutsk

Key words and phrases: dialogue control; context-free grammar; positively formed formula; translation programs synthesis; formal language.

Abstract: The paper presents the developed logical linguistic model of dialogue of object programs. The model enables to set dialogue requirements, synthesize the programs of diagnosis and error correction. To construct this model and integrate it into dialogue programs and programs of dialogue control the tool is developed.
