УДК 681.3

В. Э. Балтрашевич
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)

Разработка автоматизированных обучающих систем на базе списка атрибутов

Описывается технология разработки автоматизированной обучающей системы, позволяющая легко изменять и тип, и предметную область решаемых задач. Решаемая задача разбивается на блоки, с каждым из которых связывается атрибут с определенной структурой, позволяющей хранить знания об атрибуте, такие, как вопросы к пользователю, номер соответствующей процедуры обработки, тип ответа, признак записи ответа и др. Предметная область описывается с помощью отдельного модуля, который демонстрирует текущее состояние предметной области и содержит процедуры, образующие изучаемый метод. Порядок запуска этих процедур определяется основной программой. Составляющие процедуры кроме выполнения своих функций дополнительно проверяют ответ пользователя и выдают соответствующее сообщение. Описание атрибутов производится на языке эксперта и задается в отдельном текстовом файле, который может редактироваться. Ответы пользователя могут быть разных типов: строковые, целые, вещественные. При получении неправильного ответа пользователя система выдает подсказки, но учитывает факт ошибки, который может использоваться при оценке знаний обучаемого.

Поверхностные и глубинные знания эксперта, контроль знаний обучаемого, блок пояснений

В последнее время получила распространение инструментальная автоматизированная обучающая система (АОС) на базе экспертной системы (ЭС), предназначенная для обучения пользователя процессу решения определенного класса задач из конкретной предметной области (ПрО), а также для контроля знаний [1], [2]. Инструментальность АОС означает возможность легкой смены и класса задач, и ПрО. Процесс решения задачи состоит из основного (поверхностного) процесса, который использует результаты подпроцессов (глубинных процессов). Поверхностный процесс описывается с помощью языка эксперта, использующего правила продукций и описание атрибутов. С помощью ЭС и правил продукций глубинные знания связывались в основной (поверхностный) процесс решения изучаемой задачи (метода). При разработке и использовании данной системы студенты знакомились с разными способами представления знаний. Особенностью АОС является возможность контроля ответов ученика о результатах глубинных процессов и режима помощи при этих ответах; кроме того, в ней реализована система учета ошибок ученика и подсказок ему с последующей оценкой его знаний. Особенностью процедур, описывающих глубинные знания, является необходимость проверки ответов пользователя наряду с традиционной выработкой правильных ответов.

Часто процесс решения изучаемой задачи уже описан на традиционном языке программирования, так что необходимость его описания на языке продукций отпадает. Но хотелось бы сохранить возможность проверки знаний обучаемого, оказания ему помощи и оценки его знаний на основе собранной статистики ответов обучаемого. Так как уже имеется процедура решения задачи и она должна быть модифицирована для указанных целей, то факт инструментальности (т. е. легкой (без перекомпиляции АОС) смены задачи с помощью замены файла со знаниями) отпадает, но появляется возможность разработки технологии создания подобных АОС за счет использования заранее разработанной общей системной части.

Таким образом, целью описываемой работы являлось создание технологии, позволяющей строить АОС на базе готовых программ (процедур) с небольшой их модификацией для целей контроля знаний учащегося и помощи ему в процессе решения задачи.

ЭС состояла из списка правил, списка атрибутов, стека целей, стека фактов, блока логического вывода и блока объяснений. Если список продукций не будет использоваться, то, естественно, упрощается блок логического вывода до обработки списка атрибутов. Упрощенный блок объяснения нужно оставить, и так как он базировался на стеке фактов, то его тоже следует оставить.

Рассмотрим требования к структуре данных для атрибутов, выдвигаемые АОС. С каждым атрибутом можно связать текст перевода, а с некоторыми атрибутами надо связать тексты подсказок. Текст перевода используется для замены имени атрибута при поясняющей распечатке атрибута в стеке фактов. Что касается текста подсказки, то обычно в ЭС он фактически состоит из вопроса о значении атрибута и из возможных ответов на данный вопрос. Очевидно, что в обучающей системе этот параметр должен быть разделен на два, т. е. на вопрос и подсказку.

Так как значение атрибута, заданное обучаемым, должно быть проверено, и для этой проверки используется процедура из глубинных знаний, то среди параметров атрибута должен быть параметр, содержащий имя процедуры или ее номер.

В общем случае значения атрибутов могут быть разных типов (целые, вещественные, строковые и т. п.), поэтому при реализации на языках со строгой типизацией необходим параметр, указывающий тип атрибута. Для объединения разных типов значений атрибутов в один объединенный тип можно использовать структуры данных, подобные записям с вариантами языка Паскаль. Ключ варианта определяет тип значения атрибута в конкретном случае.

При использовании процедур, проверяющих ответы пользователя, возможны 2 режима. При первом режиме в стек фактов записывается признак правильности задания значения атрибута, а при втором – проверенный ответ пользователя, по которому происходит ветвление в процессе, описанном поверхностными знаниями. Поэтому среди параметров атрибута должен быть признак записи ответа пользователя.

Для представления атрибутов используется класс АТРИБУТ, т. е. декларативно-процедурные структуры данных, состоящие из следующих полей (слотов): <имя атрибута>; <перевод>, т. е. более полное смысловое обозначение атрибута; <вопрос>, текст которого выдается пользователю, если значение атрибута должно быть задано пользователем; <подсказка>, т. е. варианты ответов или описание метода определения значения атрибута; <номер обслуживающей процедуры>; <признак записи ответа>, который определяет, нужно ли записывать в базу данных ЭС сам ответ или признак его наличия; <рекомендация> – многострочные пояснения; <список возможных значений>, позволяющий пользователю выбирать, а не вводить значения атрибутов. Для конкретного атрибута часть указанных параметров (слотов) может отсутствовать.

Описание синтаксиса представления параметров атрибутов на языке представления поверхностных знаний эксперта приведено ниже. Указанные конструкции языка эксперта могут использоваться в любом порядке.

Зададим в нотации Бэкуса–Наура язык эксперта:

<предложение> ::= <вопрос> | <подсказка> | <перевод> | <параметры> | <help> | <рекомендация> | <список возможных значений>

<атрибут> ::= <лексема>

< вопрос> ::= вопрос <атрибут> <конец строки> < текст вопроса > <конец строки>

<подсказка> ::= подсказка <атрибут> <конец строки> < текст подсказки> <конец строки>

<перевод> ::= перевод <атрибут> < конец строки > <текст перевода> < конец строки >

<параметры> ::= параметры <атрибут> <конец строки> <номер процедуры> <признак типа ответа> <признак записи ответа> <конец строки>

<признак типа ответа> ::= <word> | <integer> | <real>

<признак записи ответа> ::= <yes> | <no>

<help> ::= help <атрибут> <конец строки> <номер индекса help > < конец строки >

<рекомендация> ::= рекомендация <атрибут><конец строки> <тексты рекомендаций>

<тексты рекомендаций>::= < текст рекомендации > < конец строки > |<*> < текст рекомендации >

<конец строки >

<список возможных значений > ::= список <атрибут> <конец строки> <значения (через пробел)>

<конец строки>

Кроме этих полей в классе АТРИБУТ должны быть поля для подсчета числа ошибок и числа подсказок о значении конкретного атрибута.

Глубинные знания записываются на традиционном языке программирования и включают описания структур данных, образующих модель

ПрО, и процедуры по обработке этих структур данных, определяющие подпроцессы, происходящие в модели ПрО. Описание глубинных знаний осуществляется программистом (инженером по знаниям). Процедуры, описывающие подпроцессы решения изучаемой задачи, взаимодействуют между собой под управлением процедуры изучаемого метода через глобальные структуры данных, описывающие модель ПрО.

Форма представления глубинных знаний, организованных в виде процедуры EVAL_PROC [1], следующая:

```
const
    nword = 30;
type
    word = packed array[1..nword] of char;
    tips = (integ, rel, wrd);
    domen = record case kind: tips of integ: (di: integer);
                rel: (dr:real); wrd: (dw: word);
            end;
    (* описание структур данных, образующих модель ПрО *)
    procedure EVAL_PROC(number: integer; par: domen; var val: word);
    (* описание процедур ПрО *)
    begin (* EVAL_PROC *)
            case number of
            1: (* *)
            ….
            end (* case *)
    end (* EVAL_PROC *);
```

Пользователю для ответа на вопрос системы надо знать состояние модели ПрО, для чего в системе реализована «доска объявлений» ПрО с помощью одного интерфейсного окна. Второе окно используется для организации диалога пользователя с системой.

Ответ пользователя проверяется соответствующей процедурой, входящей в состав глубинных знаний, и в случае ошибки система может по желанию пользователя выдать подсказку и затем повторит вопрос. Числа ошибок и подсказок дифференцированно запоминаются для каждого атрибута, что в дальнейшем используется для оценки знаний пользователя, т. е. формируется модель пользователя.

Организация диалога пользователя с системой осуществляется процедурой ДИАЛОГ, с помощью которой можно показать значение вопроса и получить от пользователя значение атрибута. Перед выдачей ответа пользователь может с помощью системы помощи (help) уточнить недостающие знания. Таким образом, процедура ДИАЛОГ имеет 2 режима работы – режим получения ответа и режим объяснений.

В АОС необходимо обеспечить режим помощи, при котором должны даваться пояснения по определению значений конкретного атрибута; при этом должен быть зафиксирован факт помощи. После этого система входит в цикл проверки, признаком окончания которого является правильный ответ обучаемого на заданный вопрос. С учетом имеющихся пояснений пользователь в принципе всегда сможет дать правильный ответ.

Процедура ДИАЛОГ, поддерживающая диалог с пользователем, использует процедуры EVAL_PROC, ВОПРОС и ПРОВЕРКА. Под атрибутом понимается объект класса АТРИБУТ с описанными ранее полями:

```
procedure ДИАЛОГ(атрибут, значение атрибута)
 begin error_verify := false;
Если поле вопроса пусто //АТРИБУТ.вопрос
 то EVAL_PROC(атрибут, ответ, значение) //виртуальный диалог
 else
 begin ВОПРОС;
 Если нет процедуры проверки //АТРИБУТ.номер_процедуры
 то ввод значения атрибута
 иначе begin
        ввод и проверка значения атрибута;
      ПРОВЕРКА(атрибут, ответ, error_verify);
        если не error_verify, то
         begin
 если признак записи ответа существует, //АТРИБУТ.признак_записи_ответа
 то запись ответа пользователя в поле значение атрибута
 иначе запись ДА в поле значение атрибута;
         запись значения атрибута в стек фактов;
 end;
 end;
 end;
end;
```

Процедура ВОПРОС обеспечивает печать вопроса пользователю и предоставляет возможные варианты ответов, а также в зависимости от его желания осуществляет или подсказку, или ответы на вопросы пользователя, запуская систему помощи:

```
procedure ВОПРОС(атрибут)
begin Печать вопроса;
 В зависимости от пожеланий пользователя либо
        запускается процедура ДИАЛОГ для получения ответа, либо
        осуществляется печать подсказки с наращиванием числа подсказок, либо
        запускается система помощи;
end;
```

Процедура ПРОВЕРКА запускается при наличии проверяющей процедуры.

```
procedure ПРОВЕРКА(атрибут, ответ, error_verify)
begin
 p := false;
 while (not p) do //т. е. крутимся до правильного ответа
 begin
 с учетом типа ответа (целый, вещественный, строковый)  //АТРИБУТ.тип_ответа;
            получение значения ответа;
 EVAL_PROC (АТРИБУТ.номер_процедуры, ответ, значение_проверки); //запуск процедуры
 p := значение_проверки = 'ДА'; //при правильном ответе проверяющие процедуры выдают ДА
 if p then {ShowMessage('М О Л О Д Ч И Н А')}
 else
 begin
 error_verify:=true; атрибут.ErrorCount:=атрибут.ErrorCount+1;
 if MessageDlg('ВЫ ЗНАЕТЕ КАК ОТВЕЧАТЬ?',…)=IDYES
 then MessageDlg('ТОГДА ДУМАЙТЕ О РАБОТЕ, А НЕ О...',…)
 else
 begin
 печать подсказки;
 атрибут.HintCount:=атрибут.HintCount+1;
 end;
 EXIT; // ДИАЛОГ; запуск диалога с пользователем в отдельном окне
 end;
 end;
end;
```

Основу процедуры ПРОВЕРКА составляет цикл ввода ответа пользователя о значении атрибута и проверка ответа соответствующей атрибуту процедурой. Признаком окончания цикла является правильный ответ пользователя. В начале процедуры ПРОВЕРКА в зависимости от значения признака типа ответа (хранящегося во фрейме атрибута) осуществляется ввод ответа необходимого типа и запись его в соответствующее поле переменной ОТВЕТ, которая является входным параметром процедуры EVAL_PROC. Процедура EVAL_PROC запускает проверяющую процедуру, соответствующую атрибуту. При правильном ответе пользователя все проверяющие процедуры выдают значение ДА.

При успешной проверке пользователю выдается поощрительное сообщение, иначе фиксируется факт ошибки и пользователя спрашивают, знает ли он, какое значение должен иметь атрибут. В случае положительного ответа (т. е. ошибка обусловлена небрежностью ввода) пользователю выдается рекомендация не отвлекаться от работы, иначе выдается подсказка об определении значения атрибута и фиксируется факт помощи.

При получении правильного ответа процедура ДИАЛОГ в зависимости от значения признака записи ответа записывает в стек фактов в качестве значения атрибута либо ответ пользователя, либо значение ДА.

Рассмотрим режимы работы АОС при различных сочетаниях заполнения полей объекта атрибута.

Процедура ДИАЛОГ получает управление, если в объекте, соответствующем атрибуту, поле вопроса не пусто, либо имеется имя обслуживающей процедуры.

Если поле вопроса не пусто, а имя процедуры отсутствует, то в этом случае в ответ на заданный системой вопрос осуществляется обычный ответ пользователя о значении атрибута.

Если поле вопроса пусто, а имя процедуры есть, то это режим использования имен процедур в качестве атрибутов, т. е. автоматический режим без участия пользователя. В этом случае с помощью процедуры EVAL_PROC осуществляется запуск процедуры, соответствующей атрибуту, т. е. происходит так называемый виртуальный диалог, в отличие от реального диалога при наличии вопроса.

Если же в объекте атрибута имеются и вопрос, и имя процедуры, то этот режим соответствует контролирующему режиму, при котором наряду с процедурой ВОПРОС запускается еще и процедура ПРОВЕРКА.

При создании конкретной АОС, комбинируя различные заполнения слотов фрейма, можно использовать различные режимы работы АОС при обработке конкретного атрибута.

Таким образом, при заполнении АОС знаниями о решении определенного класса задач из конкретной ПрО программист должен описать в текстовом файле знания об атрибутах по указанным выше синтаксическим диаграммам и управляющий процесс на языке программирования в процедуре изучаемого метода.

В процедуре изучаемого метода каждый атрибут используется как фактический параметр обрабатывающей процедуры ОБРАБОТКА. Процедура ОБРАБОТКА осуществляет поиск знаний о конкретном атрибуте в списке атрибутов и осуществляет запуск процедуры ДИАЛОГ. Ветвление процессов в процедуре изучаемого метода происходит на основании ответов процедуры ДИАЛОГ:

Процедура ОБРАБОТКА(атрибут, список атрибутов, ответ) // упрощенный логический вывод
begin
Поиск атрибута в списке атрибутов, т. е. поиск соответствующего объекта АТРИБУТ;
Если есть рекомендации, то выдача рекомендаций;
Если есть вопрос, то ДИАЛОГ(атрибут, значение атрибута) // реальный диалог
Если вопроса нет, но есть номер процедуры, то ДИАЛОГ(атрибут, значение атрибута) // виртуальный диалог
end;

Раньше (в АОС на базе ЭС) поверхностный процесс решения задачи описывался на языке продукций и сохранялся в отдельном файле, который обрабатывался экспертной системой. Глубинные знания записывались на традиционном языке программирования и формировались в виде библиотечного файла. В предлагаемой системе делать библиотеку не обязательно, так как все равно надо будет компилировать основную программу и исполняющую систему. Файл с описанием атрибутов программой трансляции преобразуется в список АТРИБУТОВ, в котором каждый отдельный атрибут представляется объектом класса АТРИБУТ. Можно при описании основной программы создать и заполнить список атрибутов, и тогда после компиляции получим один exe-файл, реализующий АОС конкретной задачи.

Технология разработки АОС (на примере АОС методам оптимизации):

– берется процедура изучаемого метода;

– она разбивается на логически законченные этапы;

– с каждым этапом связывается атрибут;

– формируются знания об атрибутах и записываются в файл или в соответствующую структуру данных;

– формируется процедура EVAL_PROC;

– тело процедуры изучаемого метода в качестве основной программы модифицируется за счет замены каждого логически законченного этапа вызовом процедуры ОБРАБОТКА с соответствующим атрибутом. Атрибуты с признаком записи ответа используются в основной программе для ветвления процесса решения задачи.

Следует особо отметить, что возможно использование одной процедуры ОБРАБОТКА, т. е. упрощенного логического вывода. Но для этого надо обеспечить установку в качестве текущей цели вывода нужного атрибута. Для организации связи между атрибутами, обеспечивающей реализацию процесса изучаемого метода, в структуру класса АТРИБУТ надо добавить поле, в которое будет записываться имя следующего атрибута. С учетом наличия признака записи ответа нужно добавить 2 поля, в каждое из которых будет записываться имя атрибута, которому надо передать управление.

Системная часть включает в себя модули:

– ввода и редактирования поверхностных знаний об атрибутах, который осуществляет ввод файла с описанием атрибутов, предоставляет возможность его редактирования, его трансляцию и перевод во внутреннее представление данных, а также позволяет просматривать внутреннее представление списка атрибутов;

– упрощенного логического вывода, который позволяет задавать цель решения, выбор режима работы (автоматический или пошаговый), запуск процедуры изучаемого метода, просмотр содержимого стека фактов;

– получения ответа, который демонстрирует вопрос к пользователю и возможные ответы, получает ответ, по запросу пользователя показывает подсказки и осуществляет переход в режим помощи.

К системной части добавляется модуль (EVAL_PROC) предметной области, демонстрирующий текущее состояние предметной области и позволяющий пользователю принимать решения о продолжении работы. Кроме того к системной части в модуль упрощенного логического вывода добавляется процедура, описывающая решение изучаемой задачи (например, симплекс-метода).

Другими словами, предлагается довольно простая технология создания АОС, которая позволяет создавать АОС студентам старших курсов для обучения студентов младших курсов.

Созданный программный продукт обеспечивает довольно простую разработку АОС. Сам процесс разработки подобных АОС закрепляет программистские навыки и отдельные стадии разработки программных продуктов. Технология легко переводится на различные языки программирования.

С помощью предлагаемой технологии были разработаны АОС по изучению методов оптимизации, по изучению алгоритмов обработки графов и др.

СПИСОК ЛИТЕРАТУРЫ

1. Балтрашевич В. Э. Инструментальная АОС на базе экспертной системы // Управляющие системы и машины. 1994. № 4/5. С. 101–105.

2. Балтрашевич В. Э. АОС по симплекс-методу решения задач линейного программирования. Н91423. Каталог отраслевого фонда алгоритмов и программ. Вып. 9. Каталог программных средств учебного назначения НИИВО, каталог 9. М., 1992.

V. E. Baltrashevich
Saint Petersburg Electrotechnical University «LETI»

DEVELOPMENT OF AOS (COMPUTER-AIDED LEARNING SYSTEM) ON THE BASIS OF ATRIBUTES LIST

The technology of developing an automated learning system is described, which makes it easy to change both the type and the subject area of the problems being solved. The task to be solved is divided into blocks, with each of which an attribute is associated with a specific structure that allows storing knowledge about the attribute, such as questions to the user, the number of the corresponding processing procedure, the type of response, the sign of the answer record, and others. The subject area is described using a separate module, Which demonstrates the current state of the domain and contains the procedures forming the method being studied. The procedure for starting these procedures is determined by the main program. The components of the procedure, in addition to performing their functions, also check the user's response and issue the appropriate message. The description of attributes is made in the expert language and is specified in a separate text file, which can be edited. The user's answers can be of various types: string, integer, real. When the user receives an incorrect response, the system issues hints, but takes into account the fact of error that can be used in assessing the learner's knowledge.

Surface and depth knowledge of the expert, control of the knowledge of the trainee, block of explanations

УДК 629.7

В. А. Шишков, В. В. Макаров
ООО «СТЦ» (Санкт-Петербург)

С. А. Кудряков
Санкт-Петербургский государственный университет гражданской авиации

С. А. Беляев, В. В. Романцев
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)

Управление беспилотным воздушным судном «Орлан-10» на протяженных маршрутах

Рассмотрены актуальные вопросы оптимизации выполнения полетов дистанционно пилотируемых воздушных судов серии «Орлан-10» по протяженным трассам вне пределов прямой видимости. Отражены основные направления применения беспилотных авиационных систем в интересах министерств и ведомств Российской Федерации в различных сферах: военное дело, сельское хозяйство, строительство, геодезия, метеорология, картография, экология, сфера безопасности. Приведены основные технические характеристики дистанционно пилотируемых воздушных судов серии «Орлан-10», состав системы автоматического управления, связного оборудования. Рассматривается опыт реализации полетов по протяженному маршруту с использованием нескольких наземных пунктов дистанционного управления с применением существующей связной инфраструктуры. Отражены принципы организации управления дистанционно пилотируемых воздушных судов серии «Орлан-10» с помощью нескольких наземных пунктов дистанционного управления.

Беспилотные авиационные системы, дистанционно пилотируемые воздушные суда, эксплуатация авиационной техники, пункт дистанционного управления

Беспилотные авиационные системы (БАС) в настоящее время находят все более широкое применение в самых различных сферах: военное дело, сельское хозяйство, строительство, геодезия, метео-