ГЕНЕРАЦИЯ ТЕСТОВЫХ ЗАДАНИЙ В ЭКСПЕРТНО-ОБУЧАЮЩИХ СИСТЕМАХ

И.Л. Братчиков

Кафедра математической теории
микропроцессорных систем управления
Санкт-Петербургский государственный университет
Университетская наб., 7-9, Санкт-Петербург, Россия, 199034

В статье рассматриваются методы генерации тестовых заданий, в которых используются базы знаний учебных дисциплин, формируемые на основе продукционной модели знаний, а также аппарат формальных грамматик. Представлены две экспериментальные экспертно-обучающие системы Formula Tutor и «Теоретик», ориентированные в основном на изучение точных дисциплин. Приведенные в статье примеры иллюстрируют достоинства описанных методов.

Ключевые слова: автоматизированная система обучения, экспертно-обучающая система, продукционная модель знаний, продукционное правило, контекстно-свободная формальная грамматика, каркас задания, шаблон, выборочно-конструируемый ответ.

Несмотря на то, что разработки в области автоматизации образования ведутся давно, недостаток многих существующих систем автоматизированного обучения заключается в слабом развитии средств генерации заданий для обучаемых и анализа ответов на них, в силу чего исследования возможностей увеличения интеллектуальности обучающих систем не потеряли актуальности и сегодня.

Очевидно, что тестирование знаний обучаемых и индивидуализация процесса обучения на основе результатов такого тестирования являются важнейшей особенностью автоматизированных систем обучения (АСО). Пристальное внимание методам тестирование уделено, например, в фундаментальной монографии [1]. Большинство современных обучающих систем обладает удобными средствами создания учебных курсов для преподавателей и работы с системой для обучаемых, однако во многих случаях их слабым местом является механизм формирования вопросов, предлагаемых обучаемому в процессе тестирования. Например, в [2] подробно описан алгоритм подготовки задачи для тестирования: разбиение на подзадачи, формулирование гипотез о том, что знает и чего не знает обучаемый, подбор признаков для подтверждения или опровержения этих гипотез, выставление априорных вероятностей признаков. Но для ответов на вопросы, по которым система делает вывод об уровне знаний обучаемых, применен традиционный метод меню (вопросы с множественным выбором ответа) Между тем именно автоматическое формирование заданий при тестировании обучаемых является основным признаком так называемого генеративного обучения, при котором почти все элементы учебного процесса (учебные задания, решения, обратная связь с обучаемым) не зафиксированы в обучающей программе, а генерируются АСО. На наш взгляд, именно генеративное обучение является наиболее перспективной методикой компьютеризации процесса обучения.

Одним из возможных подходов к автоматизации генерации тестовых заданий при проверке знаний учащихся является метод параметризации вопросов, хорошо

продемонстрированный на примере интерактивной интеллектуальной системы [3]. Данная система создавалась для решения проблемы массового применения электронных учебников по алгебре в школе — для предотвращения списывания правильных ответов учениками необходимо было обеспечить каждого из них индивидуальными, хотя и однотипными задачами, что требовало наличия тысяч таких задач. В основе описываемого подхода лежит метод клонирования шаблона документа, который представляет собой параметризованное задание, а его клоном считается вариант с конкретными значениями параметров. Отдельно для каждого документа задаются условия перебора параметров, что влияет на количество генерируемых клонов.

Пример шаблона документа по теме «Решение квадратных уравнений» может быть таким: «Решить уравнение: $x^2-(a+b)x+ab=0$. Ответ: $x=a, x=b$. Условия перебора: $(1\le a\le 10)\,\&\,(1\le b\le 10)\,\&\,(a<b)$». В ходе клонирования параметры, удовлетворяющие условиям перебора, подставляются в формулы шаблона для задания и ответа на него, после чего полученные формулы при необходимости упрощаются при помощи символьных преобразований. Хотя данная интерактивная система и автоматизирует генерацию вопросов для обучаемых, она вряд ли может быть названа полностью интеллектуальной, и вопрос о применении методов искусственного интеллекта в обучающих системах по-прежнему требует изучения.

Перспективным решением проблемы генерации вопросов является использование логического вывода с целью автоматизации создания вопросов. Для этого необходимо построить базу знаний системы так, чтобы выводимыми в ней оказались только семантически правильные вопросы. Исследования в данном направлении проводились на факультете прикладной математики — процессов управления Санкт-Петербургского государственного университета под руководством автора настоящей статьи. В результате этих исследований были разработаны две экспериментальные системы, получившие названия Formula Tutor и «Теоретик»

Экспертно-обучающая система Formula Tutor

Система Formula Tutor предназначена для обучения работе с формулами и ориентирована на такие дисциплины, как физика, математика, отчасти химия. Система была разработана М.В. Аксёновым [4]. Поясним работу системы на примере физики. Объектами дедуктивной системы в Formula Tutor являются величины, входящие в физические формулы и законы, а правилами — те формулы и законы, которые эти величины связывают. Такие правила удобно представлять так называемыми продукционными правилами или продукциями вида

<антецедент> → <консеквент>.

В антецеденте находятся физические величины, позволяющие вычислить величину, стоящую в консеквенте. Функционирование данной дедуктивной системы позволяет формировать сложные формулы различной степени вложенности, что позволяет формулировать задачи на нахождение выражения для некоторой физической величины через другие величины. Так происходит генерация заданий

(от искомой величины к величинам, необходимым для ее вычисления, после чего последние получают числовые значения). Анализ ответов в реализованной версии системы осуществляется путем сравнения с ответами, вычисленными обучающей системой.

Перейдем к более подробному описанию системы.

На рис. 1. показана структура экспертно-обучающей системы Formula Tutor в виде диаграммы компонентов языка UML. Напомним, что на таких диаграммах компоненты изображаются в виде прямоугольников, в одну из сторон которого врезаны два прямоугольника поменьше. Зависимости между компонентами изображаются пунктирными стрелками, направленными от зависимого компонента к независимому. Зависимость здесь понимается в том смысле, что зависимый компонент использует в своей работе услуги, которые предоставляет независимый [5].

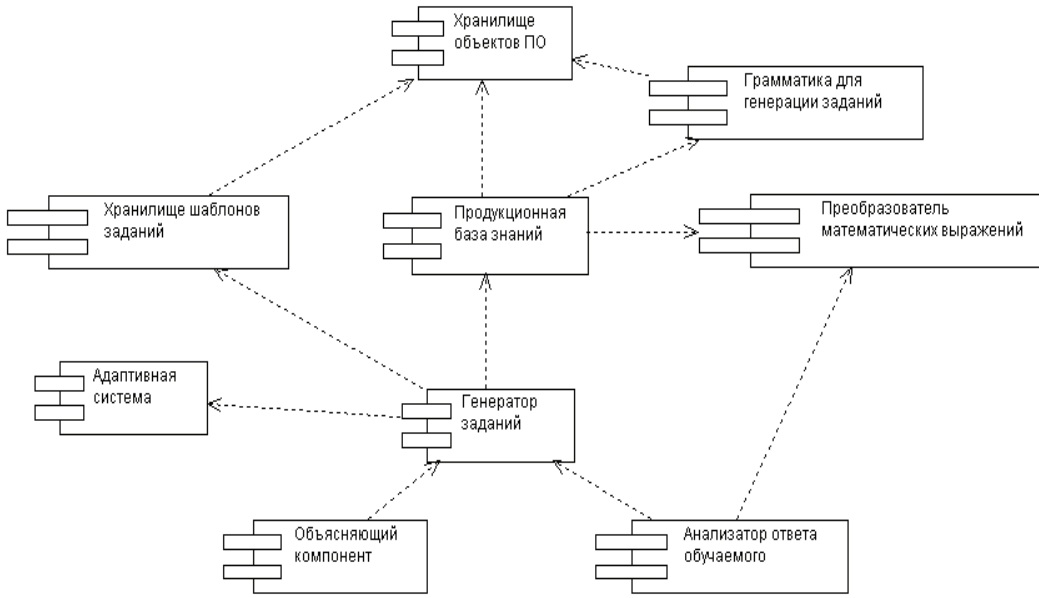

Рис. 1. Структура ЭОС Formula Tutor

Приведем краткую характеристику компонентов системы.

Хранилище объектов предметной области используется для описания объектов предметной области. Информация об объекте состоит из следующих элементов: имени объекта и списка полей. Каждое поле из списка имеет уникальное в пределах данного объекта имя, тип и при необходимости область допустимых значений. По умолчанию в область допустимых значений входят все значения указанного типа. Объекты предметной области могут быть производными от других объектов, в этом случае они наследуют все поля своего базового объекта, а также объект может выступать в качестве поля другого объекта. В совокупности объекты предметной области образуют модель предметной области.

Имя объекта должно быть уникальным, поскольку через имя осуществляется доступ к этому объекту.

Поле может иметь один из следующих типов:

int — 32-разрядное целое число;

double — 64-разрядное число с плавающей точкой;

string — строка символов;

object — ссылка на описание другого объекта предметной области.

Область допустимых значений зависит от типа поля. Так, для численных значений область может быть задана в виде некоторого диапазона или в виде списка значений, если данное поле может принимать только дискретные значения. Для строкового типа область допустимых значений может состоять из списка строк символов. Поле типа object в качестве области допустимых значений содержит имя объекта, на которое ссылается данное поле. Вот простые примеры объектов:

число (значение (int, (–100 – 100))), формула (значение (string)).

Объект число содержит одно поле значение типа int с областью допустимых значений [–100, 100]. Объект формула содержит одно поле значение типа string. По умолчанию область допустимых значений — это множество всех строк символов.

Контекстно-свободная грамматика используется для генерации индивидуальных математических выражений. Каждому правилу приписывается параметр в виде неотрицательного целого числа, определяющий стоимость использования данного правила при генерации выражений. Каждому выражению приписывается стоимость, равная сумме стоимостей правил вывода, использованных при его генерации. Стоимость отражает объективную сложность выражений, благодаря чему система генерирует задания, соответствующие текущему уровню знаний обучаемых. В правилах грамматики в качестве терминальных символов могут выступать лексемы (т.е. цепочки литер, представляющие, например, функции и поля объектов из хранилища предметной области). Это позволяет генерировать формулы, содержащие математические функции и элементы некоторых множеств.
Ниже приводится пример грамматики, генерирующей выражения с вещественными числами и тригонометрическими функциями (правила вывода для каждого нетерминала группируются с помощью метасимвола "|"). Считаем, что в хранилище объектов содержится объект число (значение (duble, 0 – 10))

$$G = \{V_N, V_T, S, P\}; V_N = \{S,B,C,D\};$$
$$V_T = \{\sin(x), \cos(x), tg(x), ctg(x), \text{число.значение}\}.$$

Множество правил вывода P:

$$S \to B\ \{0\} \mid S+B\ \{1\} \mid S-B\ \{1\}$$
$$B \to C\ \{0\} \mid B*C\ \{2\} \mid B\backslash C\ \{2\}$$
$$C \to (S)\ \{1\} \mid D\ \{0\}$$
$$D \to \sin(x)\ \{2\} \mid \cos(x)\ \{2\} \mid tg(x)\ \{2\} \mid ctg(x)\ \{2\} \mid \text{число.значение}\ \{0\}.$$

Алгоритму генерации выражений по грамматике задается параметр в виде целого числа, представляющий сложность выражения, соответствующую уровню

знаний обучаемого. При генерации не допускается превышение указанной сложности. В процессе генерации из параметра вычитается стоимость каждого примененного правила. По окончании генерации значение параметра должно быть неотрицательным. Пусть, например, сложность равна 8. Тогда могут быть сгенерированы выражения:

sin(x) – 0.5 + cos(x)*2; tg(x) / (sin(x) + 1.5); (sin(x) + 0.7)/(cos(x) – 0.3) и т.д.

Генерация заданий производится на основе продукционной базы знаний. Так как правила продукционной базы представляются на формальном языке, для преобразования заданий в привычный обучаемому вид система использует шаблоны заданий, о которых речь пойдет ниже.

В используемой системой Formula Tutor модели базы знаний каждая продукция имеет вид

(I) A→B; Comm; Cost.

Здесь I — имя продукции, A→B — ядро продукции, в котором A — антецедент, B — консеквент, Comm — комментарий к правилу (может отсутствовать), Cost — стоимость использования данного правила.

Первые два элемента имеют происхождение из классической продукционной модели. Оставшиеся два связаны со спецификой использования продукционных правил в обучении. Комментарий к правилу необходим для предоставления информации о правиле: сфере его применения, методах использования и т.д. Стоимость правила отражает объективную сложность использования данного правила. Благодаря такой модели система генерирует задания, соответствующие текущему уровню знаний обучаемого.

Антецедент продукционного правила представляется в виде списка условий. Каждое условие состоит из тройки: имя объекта, имя поля, значение поля. Правила продукционной базы знаний строятся на основе объектов предметной области. Имена объекта и поля должны в точности соответствовать описанию какого-либо объекта предметной области. Значение поля может быть либо из области допустимых значений поля, либо быть одним их двух ключевых слов: ValueX или GenerateX, где X — некоторое целое положительное число. Ключевое слово ValueX при генерации задания будет заменено произвольным значением из области допустимых значений данного поля. Ключевое слово GenerateX при генерации задания заменяется выражением, построенным грамматикой для генерации заданий.

Консеквент состоит из списка заключений. Каждое заключение состоит из трех элементов: имени объекта, имени поля, допустимого значения данного поля или ключевого слова Calculate. Ключевое слово Calculate означает, что модуль символьных преобразований должен осуществить преобразования выражения, находящегося в скобках за ключевым словом Calculate. Если в антецеденте находится ключевое слово ValueX или GenerateX, оно может быть помещено в скобки после Calculate, тогда оно будет замещено выражением, полученным в антецеденте, соответственно, подставленное выражение будет подвергнуто символьным преобразованиям.

Пусть, например, в хранилище имеются объекты:

слагаемое (значение (int, 0 – 100)); сумма (значение (string)).

Тогда на их основе можно построить следующее ядро продукционного правила:

слагаемое.значение = Value1, слагаемое.значение = Value2 →
сумма.значение = Calculate(Value1+Value2).

При генерации задания система выполнит следующие действия: заменит ключевые слова Value1 и Value2 некоторыми значениями из области допустимых значений поля значение объекта слагаемое (например, значениями 17 и 86), затем Value1 и Value2 в консеквенте будут заменены теми же значениями, после чего получившееся выражение «17+86» будет обработано преобразователем символьных выражений, в результате получим 103.

Если поле объекта A ссылается на другой объект (объект B), то к полям объекта B можно обратиться, используя имя объекта A. Такой метод доступа к полям используется в условиях и заключениях правил. Формат обращения к таким полям следующий:

имя объекта.имя поля.имя поля. …. имя поля=значение.

Пусть в хранилище объектов имеются следующие объекты:

автомобиль(тип_движения (string (едет));
скорость (object (скорость_автомобиля)))
скорость_автомобиля(единица_измерения(string (км/ч));
значение(int (1 – 300))).

Для того чтобы обратиться к полю значение объекта скорость_автомобиля через объект автомобиль нужно построить следующую конструкцию:

автомобиль.скорость.значение=Value1.

Продукционные правила из базы знаний и грамматика используются для генерации так называемых каркасов заданий. Для того чтобы представить каркас задания в привычном обучаемому виде, создаются особого вида структуры, названные шаблонами заданий. Они состоят из текста и специальных символов. Шаблоны заданий характеризуется набором исходных данных и неизвестных величин. Сопоставляя каркас задания с шаблоном, можно преобразовать его в привычный обучаемому вид.

Рассмотрим пример использования шаблонов заданий. Считаем, что хранилище объектов содержит следующие объекты:

транспортное_средство (object (автомобиль, корабль)
автомобиль (тип_движения (string (едет));
скорость (object (скорость_автомобиля)));
корабль(тип движения (string (плывет));
скорость (object (скорость_корабля)))
скорость автомобиля(единица_измерения(string (км/ч));
значение(int (1 – 300)), коэфф(double (1)))
скорость корабля(единица измерения(string (узл.);
значение(int (1 – 70)), коэфф(double (1.852)

время(значение(int (1 – 100)))
расстояние(значение(string))

В базе знаний имеется продукционное правило:

транспортное_средство.скорость.значение=Value1,
транспортное_средство.скорость.коэфф=Value2, время.значение=Value3 →
расстояние. значение=Calculate(Value1*Value2* Value3); Cost = 2.

В списке шаблонов имеется шаблон:

объект_движения=instance(транспортное_средство) (объект_движения объект_движения.тип_движения) со скоростью {объект_движения.скорость.значение!} (объект_движения.скорость.единица измерения) [объект_ движения.скорость.коэфф] в течение {время.значение!} часов, определить расстояние, которое прошел (объект_движения). [расстояние.значение?].

При генерации задания сначала строится каркас задания, который формируется приведенным правилом:

автомобиль.скорость.значение=Value1, автомобиль.скорость.коэфф=Value2,
время.значение=Value3 →
расстояние.значение=calculate(Value1*Value2* Value3).

При формировании каркаса было выполнено так называемое инстанцирование объекта транспортное_средство, в результате чего получен объект автомобиль. Рассмотрим структуру шаблона. Равенство объект_движения=instance(транспортное_средство) вводит имя объект_движения, которое в качестве значения может принять автомобиль или корабль. Текст в фигурных или квадратных скобках определяет значения, при этом, если он заканчивается знаком «!», эти значения должны быть определены при генерации задания, а в случае «?» — вычисляться обучаемым. При этом значения в квадратных скобках не включаются в формулировку задания. Текст вне скобок переносится в формулировку задания, а текст в круглых скобках заменяется соответствующими значениями. Чтобы проверить, подходит ли шаблон к каркасу, выполняется их сопоставление. Оно заключается в сравнении значений из каркаса с значениями, определенными текстами в фигурных и квадратных скобках шаблона.

В нашем случае имеем:

1) в каркасе автомобиль. скорость. значение, в шаблоне объект_движения. скорость. значение.

Так как одно из значений имени объект_движения есть автомобиль, сравнение выполнено успешно, а имени объект_движения присваивается нужное значение;

2) в каркасе автомобиль.скорость.коэфф, в шаблоне то же значение (с учетом значения имени объект_ движения);

3) в каркасе время.значение, в шаблоне то же;

4) в каркасе расстояние.значение, в шаблоне — то же.

Так как другие значения в каркасе и шаблоне отсутствуют, сопоставление выполнено успешно.

Теперь выполняется конкретизация значений Value1, Value2 и Value3, после чего строится окончательная формулировка задания, например: Автомобиль едет

со скоростью 70 км/ч в течение 5 часов, определить расстояние, которое прошел автомобиль.

Охарактеризуем остальные компоненты структуры ЭОС (см. рис. 1). Адаптивная система хранит информацию о том, насколько успешно обучаемый решает поставленные системой задания, в использовании каких правил и формул допускает ошибки и т.д. Это позволяет системе адаптироваться к текущему уровню знаний обучаемого и предлагать ему задания, соответствующие этому уровню.

Назначение преобразователя математических выражений — выполнять символьные преобразования математических выражений. Это необходимо для приведения формул к стандартному виду, к которому приводятся как формулы, порожденные грамматикой, так и формулы, представляющие ответы обучаемых.

Функция объясняющего компонента — помогать обучаемому, если он не может решить задачу. Анализатор ответа обучаемого призван помочь обучаемому в поиске ошибки, возникшей в ходе решения задания.

Экспертно-обучающая система «Теоретик»

Экспертная обучающая система «Теоретик» (разработчик Ю.В. Юрчишко) предназначена для тестирования знаний обучаемых по теоретическим дисциплинам. Она представляет собой оболочку, позволяющую реализовать несколько подходов к формированию баз знаний, на основе которых осуществляется обучение.

Сперва рассмотрим метод генерации вопросов и анализа ответов обучаемых, реализованный в системе «Теоретик». Ранее были рассмотрены типы тестовых заданий, среди которых наиболее интересным представляется программа «Словарь», которая, во-первых, предоставляет возможности для генерации вопросов, а во-вторых, приближает ответы обучаемых к свободно конструируемым.

Основная идея программы такова. Текст формулировки, представляющей правильный ответ, должен быть собран учащимся из фрагментов-«кубиков», причем среди них присутствуют лишние «кубики», и они никак не сгруппированы (при делении набора фрагментов на группы, в каждой из которых верным является лишь один фрагмент, задача обучаемого значительно облегчается). В системе «Теоретик» генерация подобных вопросов должна осуществляться автоматически, поэтому варьирование фрагментов определения с последующим их перемешиванием можно заменить на смешивание фрагментов двух различных определений, что даст тот же эффект. За счет произвольности выбора второй формулировки, элементы которой будут перемешаны с элементами спрашиваемого у обучаемого определения, число различных вариантов вопроса, относящихся к одному и тому же понятию, лишь на единицу меньше количества всех понятий в базе знаний. При оценивании ответа, данного обучаемым, необходимо учесть как правильность выбора фрагментов из списка предложенных «кубиков», так и порядок их расположения.

Анализ правильности подбора элементов ответа можно провести с помощью метода контрольных сумм: фрагментам спрашиваемого определения назначить положительные веса в соответствии с их важностью в данном определении, а фрагментам дополнительной формулировки — отрицательные, после чего сравнить сумму весов в ответе обучаемого с контрольной суммой верного ответа. На-

значение весов поможет понять, насколько существенны ошибки, допущенные обучаемым в формулировке ответа.

Перейдем к рассмотрению методов формирования баз знаний. В подавляющем большинстве дисциплин существует ряд базовых понятий, через которые затем определяются новые понятия, а для них, в свою очередь, формулируются свойства и признаки — этот процесс можно продолжать и далее. Таким образом, для каждого тематического раздела конкретной дисциплины выстраивается иерархическая структура, состоящая из теоретических сведений. В качестве простого примера можно привести иерархию понятий в геометрии: точка и прямая (базовые понятия), пересечение прямых и параллельные прямые (понятия, определяемые через базовые), признаки параллельности прямых (теоремы).

На данном свойстве основан самый очевидный способ формирования базы знаний. Формируется множество базовых понятий, играющих роль фактов, а остальные понятия определяются с помощью правил вывода. Они составляются на основе тех формулировок, которые будут предлагаться обучаемому в качестве тестовых заданий. Идея проста: если в определении некоторого понятия встречается какой-либо термин, то нужно удостовериться в том, что обучаемый этот термин знает. Этот способ подходит для случая, когда в учебном материале имеется большое количество специфической для предмета терминологии, представляющей сложность для учащихся. Данная модель базы знаний может быть представлена моделью исчисления предикатов с 0-местными предикатами и правилами вывода вида «Предикат1, Предикат2, ..., ПредикатM → ПредикатN». Она должна быть дополнена формулировками понятий, используемых при генерации вопросов, которые удобно поместить в качестве пользовательских данных у соответствующих предикатов. Пользовательскими данными может быть произвольная строка символов, вводимая при заполнении базы знаний, а также веса, роль которых была обсуждена выше.

Рассмотрим пример простейшей базы знаний по школьной геометрии, составленной согласно данным правилам.

1. Факты:
   a) Точка
   b) Прямая
   c) Угол.
2. Правила вывода:
   a) Точка, Прямая → Параллельные прямые
   b) Угол → Накрест лежащие углы
   c) Накрест лежащие углы, Параллельные прямые → 1-й признак параллельности прямых.

Пользовательская информация должна быть занесена следующим образом:

— для предиката Параллельные прямые: «Две прямые | 2 | на плоскости | 3 | не | 3 | имеющие | 1 | общих точек | 2»;

— для предиката Накрест лежащие углы: «Углы | 1 | лежащие между двумя прямыми | 3 | пересеченными секущей | 2 | по разные стороны | 3 | от секущей | 2»;

— для предиката 1-й признак параллельности прямых: «Если | 1 | при пересечении | 2 | двух прямых секущей | 2 | накрест лежащие углы | 3 | равны | 3 | то | 1 | прямые | 1 | параллельны | 3».

Другой подход к разработке базы знаний состоит в задании для понятий учебного курса связей, описывающих последовательность подачи теоретического материала. К примеру, учитель геометрии может счесть разумным тот факт, что усвоению 3-го признака параллельности прямых должно предшествовать изучение первых двух признаков, и запишет соответствующее правило вывода: «1-й признак параллельности прямых, 2-й признак параллельности прямых → 3-й признак параллельности прямых». Аналогичным образом можно задать связи между понятиями по возрастанию уровня их сложности, например переход от аксиом к теоремам: «Аксиома о параллельных прямых → 1-й признак параллельности прямых». Опытный преподаватель наверняка найдет и другие варианты структурирования учебного материала.

Система «Теоретик» имеет в своем составе все необходимые средства разработки учебных курсов, интерфейсы пользователей и пр. Мы рассмотрим те из них, которые непосредственно связаны с генерацией вопросов. Это модуль генерации вопросов с выборочно-конструируемым ответом и модуль анализа ответов обучаемого. Будет кратко охарактеризован и модуль статистики успеваемости.

Модуль генерации вопросов содержит следующие пункты.

1. Извлечение строки пользовательских данных у предиката, соответствующего спрашиваемому понятию; удаление из извлеченной строки лишних пробелов и проверка ее корректности процедурой (строка должна иметь вид «название понятия / 1-й фрагмент | вес 1-го фрагмента | 2-й фрагмент | вес 2-го фрагмента...»). Если все в порядке, то случайный выбор второго (сопутствующего) предиката из числа еще не усвоенных обучаемым понятий (выбирается с помощью статистики успеваемости), иначе окончание работы с отрицательным результатом.
2. Извлечение строки пользовательских данных у сопутствующего предиката, удаление из нее лишних пробелов и проверка на корректность аналогично 1-му пункту.
3. Если все удачно, то разбор двух строк с фрагментами вопроса, причем для фрагментов спрашиваемого понятия веса положительны, а для второго, дополнительного понятия — отрицательны, иначе окончание работы с отрицательным результатом.
4. Удаление повторяющихся фрагментов вопроса (если в формулировках обоих понятий присутствует один и тот же элемент, то удаляется дубликат с отрицательным весом).
5. Перемешивание фрагментов вопроса (порядковые номера фрагментов создаваемого вопроса распределяются случайным образом с помощью генератора случайных чисел).
6. Занесение сформированного вопроса в сборник вопросов и окончание работы с положительным результатом.

После окончания работы модуля с положительным результатом вопрос с выборочно-конструируемым ответом для выбранного для тестирования понятия успешно создан и может быть передан обучаемому.

Анализ ответа, полученного от обучаемого, осуществляется модулем анализа ответов. Ответ учащегося на вопрос, сгенерированный обучающей системой «Теоретик», представляет собой набор чисел — порядковые номера фрагментов вопроса (так как вопрос является выборочно-конструируемым), перечисленные через запятую.

Для анализа ответа выполняются следующие действия.

1. Удаление пробелов в переданной строке и проверка ее корректности (числа должны быть положительны и введены через запятую).
2. Если все в порядке, то перенос данных из строки с ответом в числовой вектор, иначе окончание работы с отрицательным результатом.
3. Подсчет контрольной суммы для верного ответа и для ответа обучаемого; отношение полученных чисел, переведенное в проценты, является основанием для выставления оценки по методу контрольных сумм.
4. Нахождение общего числа фрагментов в правильном ответе и подсчет максимального количества подряд идущих элементов ответа обучаемого, совпавших с правильным ответом; отношение данных двух чисел будет использовано для определения оценки за расположение фрагментов ответа.
5. Вычисление оценок обучаемого.
6. Добавление среднего арифметического двух оценок в конец вектора, хранящего оценки учащегося.
7. Поиск проходного балла для понятия из текущего вопроса и внесение пометки о зачете/незачете после сравнения с найденным проходным баллом среднего арифметического двух оценок, поставленных обучаемому; окончание работы с положительным результатом.

Подсчет контрольных сумм производится путем сложения весов соответствующих фрагментов ответа. В случае, если обучаемый ошибся и включил в свой вариант фрагмент определения сопутствующего понятия, контрольная сумма его ответа уменьшится, и, таким образом, ошибка будет замечена, даже если остальная часть ответа совершенно правильна.

Подсчет количества элементов ответа обучаемого, совпадающих с правильным ответом, аналогичен поиску длины максимальной последовательности символов одной строки, входящей в другую строку, и осуществляется с использованием цикла тройной вложенности. Переменная внешнего цикла пробегает по всем элементам ответа обучаемого, промежуточного — по элементам правильного ответа, а внутренний цикл используется для постепенного увеличения смещения начала сравнения элементов двух ответов. В теле последнего цикла элементы обоих ответов (номера фрагментов текста вопроса) сравниваются между собой с учетом смещения точки начала отсчета. При их совпадении счетчик числа одинаковых элементов увеличивается на единицу, при несовпадении — обнуляется, а найденная длина входящей в правильный ответ числовой последовательности (если она превосходит найденную ранее максимальную длину) запоминается в переменной, возвращаемой функцией в качестве результата своей работы.

После завершения анализа ответа обучаемого на сгенерированный вопрос последний получает информацию о выставленных ему оценках и успешности сво-

его ответа, т.е. было ли зачтено понятие, на основе которого данный вопрос формировался.

Дополнительное средство просмотра успеваемости учащегося, доступное как преподавателю, так и самому обучаемому, — обращение к статистике результатов обучения в целом. Модуль статистики успеваемости позволяет получить сведения обо всех понятиях учебного курса, подлежащих усвоению обучаемым: какие из них успешно пройдены, а какие еще придется изучать. Специальная функция возвращает полную информацию об одном из вопросов, на которые учащийся дал ответ (обучаемый имеет право прервать выполнение предложенного ему задания, будучи не в состоянии ответить на текущий вопрос, для того чтобы повторить материал учебного курса). Информация, сообщаемая данной функцией, включают в себя номера задания и вопроса в задании, название теоретического понятия в вопросе, обе оценки за ответ обучаемого и отметку о зачете или незачете.

В заключение приведем фрагмент базы знаний для курса «Геометрия», тема «Параллельные прямые», и пример работы с обучаемым.

Факты базы знаний:

1. Точка
2. Прямая
3. Плоскость
4. Геометрическая фигура

Правила вывода:

1. Прямая, Точка → Луч
2. Прямая, Точка → Секущая
3. Луч, Точка, Геометрическая фигура → Угол
4. Угол → Равные углы
5. Прямая, Точка, Плоскость → Параллельные прямые
6. Угол, Прямая, Секущая → Накрест лежащие углы
7. Угол, Прямая, Секущая → Соответственные углы
8. Угол, Прямая, Секущая → Односторонние углы
9. Секущая, Накрест лежащие углы, Равные углы, Параллельные прямые → 1-й признак параллельности прямых
10. Секущая, Соответственные углы, Равные углы, Параллельные прямые → 2-й признак параллельности прямых
11. Секущая, Односторонние углы, Равные углы, Параллельные прямые → 3-й признак параллельности прямых
12. Параллельные прямые → Аксиома параллельности

Пользовательские данные к различным предикатам:

— Луч: «Часть прямой | 3 | исходящая из | 1 | точки | 2»

— Секущая: «Прямая | 3 | имеющая | 1 | общую точку | 4 | с некоторой другой прямой | 2»

— Угол: «Геометрическая фигура | 2 | состоящая из | 1 | точки | 3 | и двух лучей | 3 | исходящих из | 2 | данной точки | 2»

— Равные углы: «Два угла | 2 | градусные меры которых | 4 | одинаковы | 3»

— Параллельные прямые: «Две прямые | 2 | на плоскости | 4 | не | 3 | имеющие | 1 | общих точек | 2»

— Накрест лежащие углы: «Углы | 1 | лежащие между | 3 | двумя прямыми | 2 | пересеченными секущей | 2 | по разные стороны | 4 | от секущей | 2»

— Соответственные углы: «Углы | 1 | один из которых | 2 | лежит в области между | 3 | двумя прямыми | 1 | пересеченными секущей | 2 | а другой | 2 | вне этой области | 4 | причем оба угла | 3 | находятся по одну сторону | 4 | от секущей | 2»

— Односторонние углы: «Углы | 1 | лежащие между | 3 | двумя прямыми | 2 | пересеченными секущей | 2 | по одну сторону | 4 | от секущей | 2»

— 1-й признак параллельности прямых: «Если | 1 | при пересечении | 2 | двух прямых секущей | 2 | накрест лежащие углы | 4 | равны | 3 | то | 1 | прямые | 1 | параллельны | 3»

— 2-й признак параллельности прямых: «Если | 1 | при пересечении | 2 | двух прямых секущей | 2 | соответственные углы | 4 | равны | 3 | то | 1 | прямые | 1 | параллельны | 3»

— 3-й признак параллельности прямых: «Если | 1 | при пересечении | 2 | двух прямых секущей | 2 | сумма | 4 | односторонних углов | 4 | равна 180 градусам | 4 | то | 1 | прямые | 1 | параллельны | 3»

— Аксиома параллельности: «Через точку | 3 | не | 4 | лежащую на данной прямой | 2 | проходит | 1 | только одна | 4 | прямая | 2 | параллельная данной | 3».

При выборе определения для генерации вопроса выбирается продукция, в антецеденте которой содержатся понятия, уже усвоенные обучаемым (соответствующие данные формирует модуль статистики успеваемости). Например выбор для вопроса продукции 8 возможен, если в обучаемый уже усвоил понятия «угол», «прямая», «секущая».

Пример генерации вопроса:

Из приведенных элементов выберите нужные и составьте формулировку для понятия «односторонние углы»:

1) при пересечении
2) равны
3) по одну сторону
4) если
5) параллельны
6) прямые
7) двумя прямыми
8) от секущей
9) пересеченными секущей
10) то
11) двух прямых секущей
12) углы
13) накрест лежащие углы
14) лежащие между.

Правильный ответ составляет следующая последовательность: 12, 14, 7, 9, 3, 8. Для генерации вопроса было использовано сопутствующее определение понятия «1-й признак параллельности прямых».

ЛИТЕРАТУРА

[1] *Башмаков А.И., Башмаков И.А.* Разработка компьютерных учебников и обучающих систем. — М.: Информационно-издательский дом «Филинъ», 2003.
[2] *Пустынникова И.Н.* Технология использования экспертных систем для диагностики знаний и умений // Образовательные технологии и общество. — 2001. — № 4. — С. 77—101.
[3] *Левинская М.А.* Автоматизированная генерация заданий по математике для контроля знаний учащихся // Образовательные технологии и общество. — 2002. — № 5. — С. 214—221.
[4] *Аксенов М.В., Братчиков И.Л.* Экспертно-обучающая система «Formula Tutor» // Процессы управления и устойчивость. — СПбГУ, 2004. — С. 111—115.
[5] *Буч Г., Якобсон А., Рамбо Дж.* UML. Классика CS. 2-е изд. / Пер. с англ.; Под общ. ред. С. Орлова. — СПб.: Питер, 2006.
[6] *Палант Ю.А.* Управляющие программы в обучении математике: Учеб. пособие. — Донецк: Изд-во ДонГУ, 1978.

GENERATION OF TEST TASKS IN THE EXPERT-TRAINING SYSTEMS

I.L. Bratchikov

Chair of the mathematical theory of microprocessor control systems
The St.-Petersburg State University
University Str., 7-9, St.-Petersburg, Russia, 199034

The article describes methods of generating test tasks that use the knowledge bases of academic disciplines that are formed on the basis of a knowledge production model as well as the apparatus of formal grammars. We present two experimental expert-training systems “Formula Tutor” and “Teoretik”, focused mainly on studying the exact sciences. In the “Formula Tutor” system generation of tasks carried out in two stages: first, by applying the production rules, and then, if necessary, using the outputs in context-free formal grammar. To give tasks to the student familiar forms the apparatus of templates is used. The analysis of student answers is performed by comparing them with the correct answers. If the answer is a mathematical formula it is converted то standard form before comparison. In the “Teoretik” system production model is also applied. Unlike the first system production rules are used to fix the dependencies of terms of educational discipline. For example, in the course “Geometry” Ray Dot, Geometric figure shows that to understand the definition of “angle” the student should know the definition of the ray, dot and geometric figure. The tasks in the system formed by mixing pieces of definitions and provide selectively-constructed answers. Examples in the paper illustrate advantages of described methods.

Key words: automated training system, expert-training system, knowledge production model, production rule, context-free formal grammar, task frame, template, selectively-constructed answer.