УДК 378.16

*В.В. Кручинин*

# ИСПОЛЬЗОВАНИЕ ДЕРЕВЬЕВ И/ИЛИ ДЛЯ ГЕНЕРАЦИИ ВОПРОСОВ И ЗАДАЧ

Рассматриваются вопросы применения деревьев И/ИЛИ для построения генераторов тестовых заданий и вопросов. Предложены оригинальные алгоритмы для генерации текстов заданий при изучении языков программирования. Показано использование деревьев И/ИЛИ для проверки семантики сгенерированной задачи. Описан алгоритм генерации вопросов для классификации, приведен конкретный пример реализации. Для студентов, аспирантов, преподавателей и инженеров, занимающихся тестовым контролем знаний.

Генераторы вопросов и тестовых заданий становятся важным элементом системы контроля знаний в вузе. Они используются при выдаче индивидуальных заданий студентам, проведении тестовых экзаменов и контрольных работ, создании различных тренажеров, компьютерных учебных программ и мультимедийных учебников[1–3].

Центральным узлом генератора является база знаний, которая обеспечивает формализованное представление знаний некоторой предметной области. Структура этой базы определяет основные характеристики генератора: вариантность, сложность, надежность, перечисляемость. Вариантность определяет общее число вопросов и тестовых заданий, которое может быть получено данным генератором. Надежность определяет корректность генерируемых заданий, поэтому необходимо иметь генератор, который генерирует только корректные задания. Перечисляемость определяет возможность однозначного соответствия между некоторым номером и конкретным вопросом, полученным генератором.

При создании генераторов могут использоваться различные модели представления знаний [4]. Одной из возможных моделей является дерево И/ИЛИ, основные понятия и перечислительные свойства которых даны в [5].

Ниже предлагаются оригинальные модели и алгоритмы генераторов вопросов и тестовых заданий, основанные на использовании деревьев И/ИЛИ.

## ГЕНЕРАЦИЯ ТЕКСТА ТЕСТОВОГО ЗАДАНИЯ

Дерево И-ИЛИ можно использовать для многовариантного представления тестового задания. Для этого текст задания разбивается на фрагменты. Обычно фрагменты разбиваются на классы: постоянные $\{T\}$ и переменные $\{V\}$. Для переменных фрагментов записываются множества реализаций, каждый из которых представляет собой конкретный текст. Затем каждый из выделенных фрагментов реализаций анализируется и, если есть возможность, разбивается на переменные и постоянные подфрагменты.

Рассмотрим использование данного метода на конкретном примере. Пусть дано конкретное тестовое задание:

«Дана следующая функция, написанная на языке программирования Си:

```
int Sum(int v[], int n){
int sum=0;
 for(int i=0; i<n; i++){
    sum+=*v++;  }
  return sum;}
```

и следующий фрагмент программы

```
int vec[5]={1, 7, 12, 5, 7};
printf("%s",sum(vec,5));
```

Какое число будет напечатано?»

В этом задании необходимо понять, что функция *Sum* определяет сумму элементов одномерного массива *v*, и в ответ ввести конкретный результат.

Используя выразительные возможности языка программирования Си, можно различными способами написать функцию нахождения суммы. Например: название функции может быть различным (*Sum, Total, CountSum, count_sum* и т.д.); описание параметров также может иметь различные вариации (*int*v, int v*[]): цикл также можно записать с помощью операторов *while* и *for*; тело цикла также может быть записано различными способами. Таким образом, задачу нахождения суммы целочисленного вектора можно записать некоторым множеством функций. Все это множество можно представить деревом И/ИЛИ (рис 1).

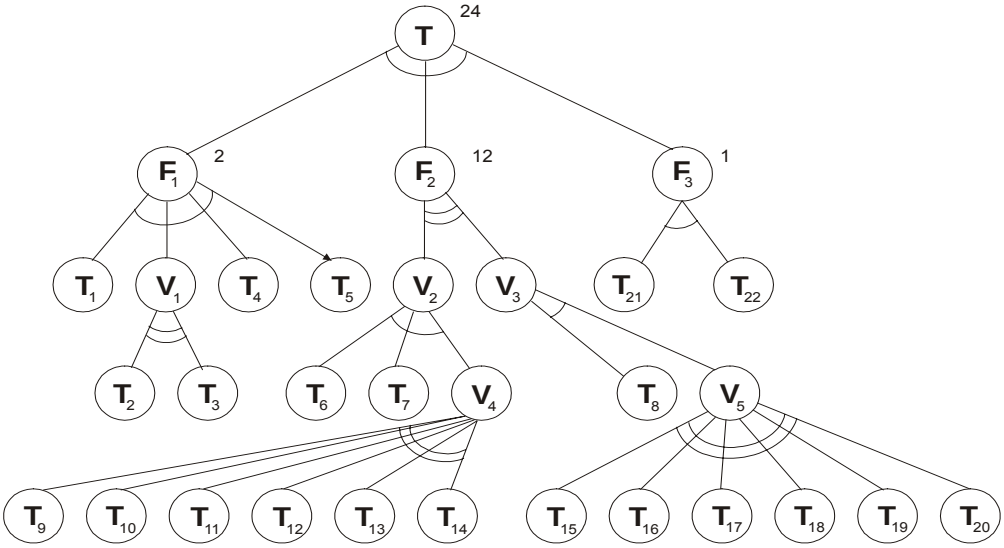


Рис. 1. Дерево И-ИЛИ для представления множества подпрограмм

Весь текст функции можно разделить на три фрагмента $\{F_1, F_2, F_3\}$, тогда узел $T$ будет И-узлом, содержащий сыновей $F_1$, $F_2$, $F_3$. Каждый из фрагментов разделяется на фиксированные и переменные части. Например, узел $F_1$ разбивается на три фиксированных фрагмента $\{T_1, T_4, T_6\}$ и один переменный $V_1$, который имеет два варианта реализации $\{T_2, T_3\}$. Ниже в табл. 1 перечислены значения узлов $\{T_i\}_{i=1}^{22}$.

Таблица 1

| | |
|---|---|
| <T1> := int Sum(int | <T12>:= sum+=v[i]; i++; |
| <T2>:= *v | <T13>:= sum+=*v++; i++; |
| <T3>:= v[] | <T14>:= sum=sum+*(v+i); i++; |
| <T4>:= ,int n){ | <T15>:= sum=sum+v[i]; |
| <T5>:= int sum=0; | <T16>:= sum=sum+v[i]; |
| <T6>:= int i=0; | <T17>:= sum=sum+*v++; |
| <T7>:= while(i<n) { | <T18>:= sum+=v[i]; |
| <T8>:= for(int i=0; i<n; i++){ | <T19>:= sum+=*v++; |
| <T9>:= sum=sum+v[i++]; | <T20>:= sum=sum+*(v+i); |
| <T10>:= sum=sum+v[i]; i++; | <T21>:=}, |
| <T11>:= sum=sum+*v++; i++; | <T22>:= return sum; } |

Используя алгоритм подсчета вариантов в дереве И/ИЛИ [3], подсчитываем общее количество вариантов, которое будет равно 24. Для генерации конкретной функции необходимо получить номер варианта и, используя алгоритм построения варианта [3], – соответствующий вариант. Производя левосторонний обход варианта и записывая листья, получим конкретное описание функции. Приведем примеры двух вариантов.

1. Вариант {T1, T2, T4, T5, T6, T7,T9,T21,T22}.

```
int Sum(int *v,int n){
int sum=0;
int i=0;
while(i<n) {
  sum=sum+v[i++];
 }
 return sum;
}
```

2. Вариант {T1, T3, T4, T5, T8,T19,T21,T22}

```
int Sum(int v[], int n){
int sum=0;
 for(int i=0; i<n; i++){
   sum+=*v++;
  }
  return sum;
}
```

## ОПРЕДЕЛЕНИЕ НАЛИЧИЯ РЕШЕНИЯ ЗАДАЧИ ПРИ ГЕНЕРАЦИИ

Проблема генерации задач уже обсуждалась в литературе [6, 7]. Ниже предлагается способ проверки наличия решения, который основан на построении и анализе дерева И/ИЛИ. Рассмотрим данный метод на примере генерации задачи на встречу. Общая схема задачи показана на рис. 2.

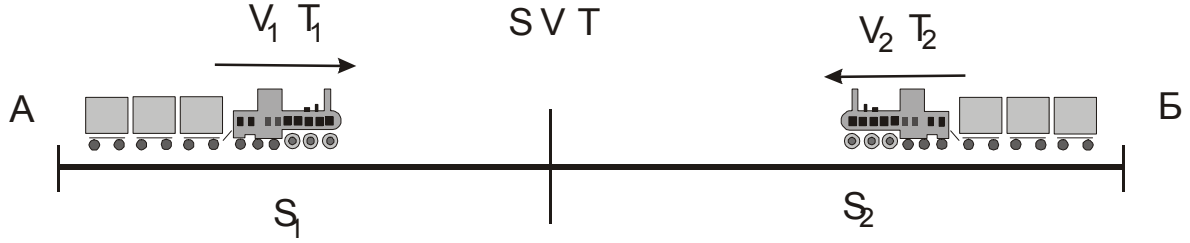


Рис. 2. Схема задачи на встречу

Для решения указанной задачи необходимо записать следующие формулы:

$$S = S_1 + S_2,$$

где $S$ – путь между пунктами; $S_1$ – путь первого и $S_2$ – путь второго «путников»;

$$S = t \times V;\ S_1 = t_1 \times V_1;\ S_2 = t_2 \times V_2;$$

$t$ – общее время движения; $t_1$ – время движения первого и $t_2$ – время движения второго «путников»; $V$ – средняя скорость движения; $V_1$ – скорость первого и $V_2$ – скорость второго «путников»;

$$V = S / t;\ V_1 = S_1 / t_1;\ V_2 = S_2 / t_2;$$
$$t = S / V;\ t_1 = S_1 / V_1;\ t_2 = S_2 / V_2;$$
$$t = t_1 + t_2;\ t_1 = t_2;$$

$K_1$ и $K_2$ – некоторые константы, причем

$$V_1 = V_2 + K_1;\ S_1 = S_2 + K_2.$$

Тогда задача может быть формально сгенерирована следующим образом: известны некоторые переменные (например, $S$, $t$, $K_2$); необходимо найти другие переменные ($V_1$, $V_2$). Если генерировать множество известных параметров и искомых переменных, то можно получить конкретную задачу, которая не будет иметь решения. Поэтому после генерации задачи необходимо осуществить семантическую проверку задачи на предмет наличия решения. Для этого используем метод, основанный на построении и использования дерева И-ИЛИ. Первоначально на основе множества приведенных формул строится дерево И/ИЛИ, далее производится его анализ, определяются варианты, дающие возможность построения уравнения. Рассмотрим кратко основные моменты построения дерева И/ИЛИ. Корнем этого дерева становится искомая переменная. Далее в

базе знаний ищутся все формулы, в которых левая часть содержит данную переменную. Эти формулы записываются как сыновья рассматриваемого узла. А сам узел становится узлом ИЛИ (т.е. для определения переменной, записанной в рассматриваемом узле, необходимо использовать одну из формул). Затем происходит переход на рассмотрение узла, в котором записана одна из формул. Все переменные, которые имеются в формуле, записываются в узлы, которые являются сыновьями рассматриваемого узла (где записана формула). А сам рассматриваемый узел становится узлом И, т.к. для вычисления формулы необходимо знать значения всех переменных. Листом дерева становится узел, в котором записана переменная в следующих случаях:

1) значение переменной известно;

2) переменная является искомой переменной;

3) в процессе построения дерева рассматриваемая переменная использовалась ранее (вариант рекурсии).

Процесс построения дерева будет завершен, если для всех листьев дерева будут выполняться условия, записанные выше. Таким образом, при построении дерева будет получено некоторое множество вариантов подстановок, которые могут приводить к построению уравнения с одним неизвестным или не приводить к нему. Вариантом поддстановки будет поддерево И/ИЛИ, у которого каждый ИЛИ-узел будет иметь только одного сына. Вариантом подстановки, приводящей к построению уравнения, будет то поддерево, у которого листья содержат известные переменные и искомую переменную.

Рассмотрим работу данного метода на конкретном примере. Пусть дана следующая задача:

Из пунктов А и Б навстречу друг другу вышли два пешехода. Известно расстояние 10 км, время в пути 2 часа. Известно также, что скорость второго пешехода на 1 км/ч больше, чем скорость первого. Необходимо найти скорость первого пешехода.

Для решения этой задачи база знаний должна содержать следующие формулы:

(1) $s$ => $v$*$t$ => $s1$+$s2$;
(2) $v$ => $s/t$;
(3) $t$ => $s/v$;
(4) $s1$ => $v1$*$t1$ => $s$-$s2$;
(5) $s2$ => $v2$*$t2$ => $s$-$s1$;
(6) $v1$ => $s1/t1$ => $v2$-$k$;
(7) $v2$ => $s2/t2$ => v1+k;
(8) t1 => s1/v1;
(9) $t2$ => $s2/v2$.

---

*Дано* $s$=10, $t1$=$t2$=$t$=2, $k$=1.
*Найти* $v1$.

Первым шагом поиска решения является построение дерева решений. Дерево решений для нашего примера показано на рис. 3. Построение начинается с записи искомой переменной *v1* (см. описание алгоритма построения дерева И-ИЛИ). Далее записываются формулы, с помощью которых определяется искомая переменная (это формулы $s1/t1$ и $v2$-$k$). Использование второй формулы приводит к решению через нахождение переменной $v2$. Для краткости этот вариант рассматриваться не будет. Рассмотрим ветвь для формулы $s1/t1$. Для данного узла будут записаны два сына – переменная $s1$ и переменная $t1$. Переменная $t1$ известна по условию задачи. Она становится листом. Далее рассматривается переменная $s1$. Для нее в базе знаний имеется две формулы $s$–$s2$ и $v1$*$t1$. Использование второй формулы не приведет к решению, т.к. происходит сокращение переменных. Для формулы $s$–$s2$ необходимо найти $s2$, т.к. переменная $s$ известна. Для переменной $s2$ в базе знаний имеется две формулы: $v2$*$t2$ и $s$–$s1$. Использование второй формулы приводит к рекурсии, т.к. ранее подстановка для переменной $s1$ уже использовалась. Рассмотрим формулу $v2$*$t2$. Здесь неизвестная переменная $v2$. Для нее производим подстановки. Это формулы $s2/t2$ и $v1$+$k$. Первая и вторая формула приводит к рекурсии. Таким образом, процесс построения дерева заканчивается.

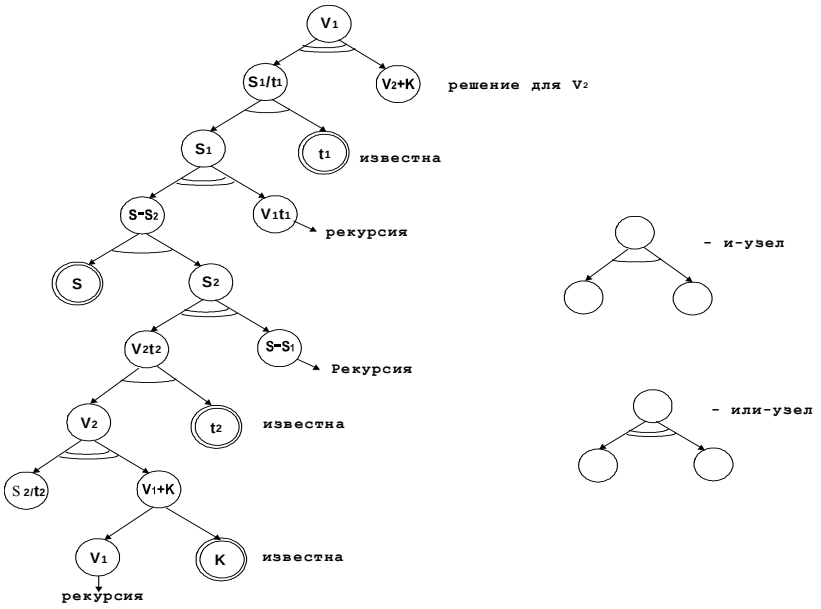


Рис. 3. Дерево решений

После построения дерева решений производится его анализ. Анализ включает: подсчет количества вариантов, анализ каждого варианта с использованием алгоритмов эквивалентных преобразований. Всего вариантов решений, без учета ветви $v2+k$, будет 4 (см. рис. 2). Решение будет получено при использовании варианта с номером 1 (см. алгоритм нумерации И/ИЛИ дерева). Этот вариант показан на рис. 4. В этом дереве записаны подстановки формул, приводящие к построению уравнения.

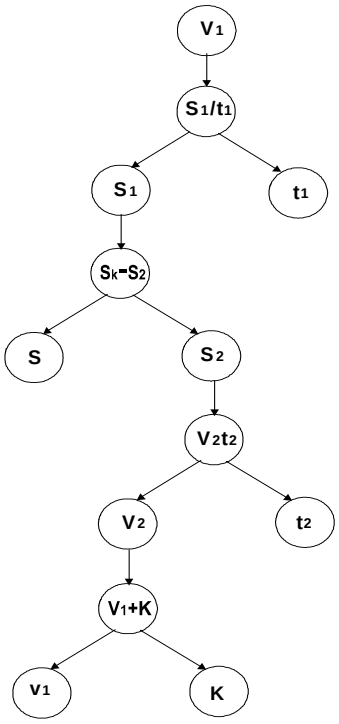


Рис. 4. Вариант решения

Рассмотрим эти подстановки:

(1) $v1$; $v1 => s1/t1$;
(2) $s1/t1$; $s1 => s–s2$;
(3) $(s–s2)/t1$; $s2 => v2*t2$;
(4) $(s–(v2*t2))/t1$; $v2 => v1+k$;
(5) $(s–((v1+k)*t2))/t1$.

## ГЕНЕРАЦИЯ МЕНЮ ВОПРОСОВ

Как известно, используя ограничения рекурсии, контекстно-свободные грамматики можно представить деревьями И/ИЛИ [3].

1. Записывается два дерева И/ИЛИ: первое описывает грамматику правильных, второе – грамматику неправильных выражений.

2. Вычисляем количество правильных и неправильных выражений, получаемых из соответствующих деревьев $m$ и $n$.

3. Вопрос формулируется следующим образом: «Среди перечисленных выражений укажите верные (или неверные).

4. Используя метод нумерации вариантов в дереве И/ИЛИ и алгоритм генерации меню-вопросов, генерируем меню вопрос[3].

4. Общее число вопросов будет вычисляться по формуле

$$v=C_m^k \cdot C_n^l,$$

где $k$ – количество правильных вариантов ответа в вопросе; $l$ – количество неправильных вариантов ответа.

## ПОСТРОЕНИЕ МЕНЮ-ВОПРОСОВ НА ОСНОВЕ КЛАССИФИКАЦИЙ

Классификации являются одним из важных элементов знаний преподаваемого курса или дисциплины. Обычно классификации представляются в виде некоторой таблицы или иерархии. Ниже предлагается метод генерации меню вопросов на основе классификации, представленной в виде И/ИЛИ дерева (рис. 5).

Этот метод заключается в следующем:

1. По данной классификации признаков строится И/ИЛИ дерево.

2. Определяется количество вариантов в И/ИЛИ дереве.

3. Строится таблица соответствий между вариантом И/ИЛИ дерева и конкретным классифицируемым объектом или явлением. Эта таблица содержит название понятия, объекта или явления и соответствующие номера вариантов.

4. Выбирается одна строчка таблицы и строится вариант, соответствующий номеру данного названия (понятия, объекта или явления).

5. Случайно выбираются еще несколько строк таблицы.

6. Вопрос формулируется так: «Даны следующие признаки: <вариант, полученный на шаге 4>. Укажите правильное название (понятия, объекта или явления)». Перечисляются случайно перемешанные названия, полученные на шаге 4 и 5. Правильным ответом будет название, полученное на шаге 4.

Рассмотрим данный метод на конкретном примере. Пусть дано дерево И-ИЛИ, классифицирующее легковые автомобили (это дерево является условным, поскольку реальное дерево содержит тысячи узлов, а таблица содержит десятки тысяч строк).

Используя алгоритм подсчета вариантов, получим:

$$v = (9) \times (3) \times (1 + (1 + 1) + 1) \times 3 \times 2 = 648.$$

Вариант дерева с номером 0 будет:

Автомобиль:
кузов: хэтчбек,
колеса: 19 дюймов,
двигатель: электрический,
привод: задний,
коробка: механика.

Предположим, что этот вариант соответствует автомобилю марки «Мерседес» модели 123. Аналогично ставим соответствие с другим вариантом классификационного дерева. Затем строим табл. 2 соответствий:

Таблица 2

| Название модели автомобиля | Номера вариантов |
|---|---|
| «Мерседес-123» | 0 |
| «Ауди-530» | 1 |
| «Тойота-127» | 2 |
| «Запорожец» | 3 |
| «Москвич-413» | 4 |

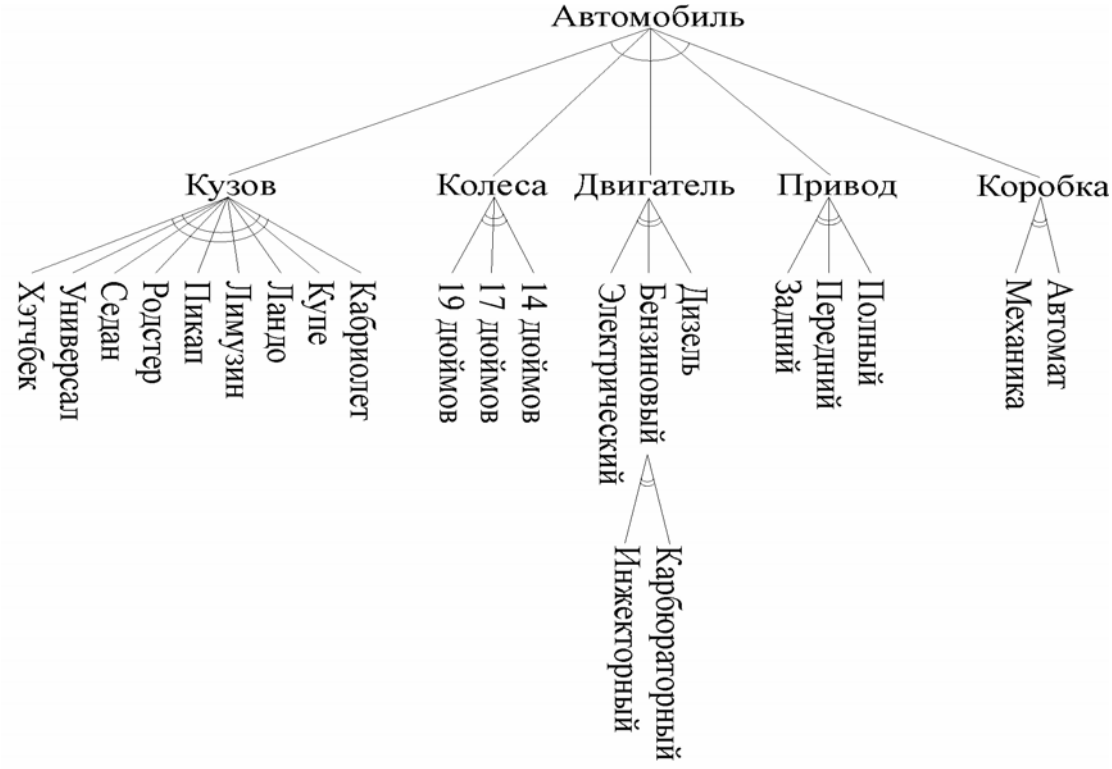


Рис. 4. Дерево И-ИЛИ для представления классификации

Используя описанный метод, можно получить следующий вопрос: «Даны следующие узлы (характеристики) автомобиля: кузов: хэтчбек; колеса: 19 дюймов; двигатель: электрический; привод: задний; коробка: механика.

Укажите марку: 1) «Ауди-530»; 2) «Тойота-127»; 3) «Мерседес-123»; 4) «Москвич-413».

Правильный ответ здесь третий: !Мерседес-123».

Подсчет числа вопросов можно произвести следующим образом:

$$N_v = n \cdot C_{n-1}^3 ,$$

где $N_v$ – общее число вопросов; $n$ – число строк таблицы; $C_{n-1}^3$ – число сочетание из ($n$ – 1) марок автомобилей по 3 для формирования списка неверных ответов.

## ЗАКЛЮЧЕНИЕ

Практическая реализация генераторов и использование их в учебном процессе, как в дистанционных технологиях обучения, так и в очном обучении, показывает их эффективность.

Применение деревьев И/ИЛИ для представления модели знаний предметной области позволяет строить разнообразные алгоритмы генерации вопросов и тестовых заданий. Кроме этого, они дают возможность строить алгоритмы семантической проверки на наличие решений при генерации тестовых заданий.

Предложенные модели и алгоритмы использовались при разработке генераторов контрольных работ и экзаменов, которые успешно эксплуатируются в Томском межвузовском центре дистанционного образования.

ЛИТЕРАТУРА

1. *Кручинин В.В.* Разработка компьютерных учебных программ. Томск: Изд-во Том. ун-та, 1998. 211 с.
2. *Башмаков А.И., Башмаков И.А.* Разработка компьютерных и обучающих систем. М.: Информационно-издательский дом «Филинъ», 2003. 616 с.
3. *Кручинин В.В.* Генераторы в компьютерных учебных программах. Томск: Изд-во Том. ун-та, 2003. 200 с.
4. Представление и использование знаний / Под ред. Х. Уэно, М. Исидзука. М.: Мир, 1989. 220 с.
5. *Кручинин В.В.* Алгоритмы и перечислительные свойства деревьев И/ИЛИ // Вестник Томского гос. ун-та. 2004. № 284. С. 178–181.
6. *Кручинин В.В., Морозова Ю. В.* Модели и алгоритмы генерации задач в компьютерном тестировании // Изв. ТПУ. 2004. № 5. С. 127–131.
7. *Кручинин В.В, Морозова Ю.В*. Модели генераторов вопросов для компьютерного контроля знаний // Открытое и дистанционное образование. Вып. 2(14), 2004. С. 36–42.

Статья представлена кафедрой промышленной электроники факультета электронной техники Томского государственного университета систем управления и радиоэлектроники, поступила в научную редакцию «Кибернетика» 15 мая 2004 г.