УДК 681.324
Ю.К. Апраксин, профессор, д-р техн. наук,
В.О. Сикач, магистр
Севастопольский национальный технический университет
ул. Университетская, 33, г. Севастополь, Украина, 99053
E-mail: kvt@sevgtu.sebastopol.ua

АВТОМАТИЧЕСКАЯ ГЕНЕРАЦИЯ СЛУЧАЙНЫХ АЛГОРИТМОВ С ЗАДАННЫМИ ПАРАМЕТРАМИ

Предлагается система генерации случайных алгоритмов с заданными характеристиками. Приводятся примеры сгенерированных алгоритмов и результаты моделирования.
***Ключевые слова**: алгоритм, генерация алгоритмов, автоматизация, случайный алгоритм.*

Введение. Одним из перспективных подходов к созданию систем логического управления является их синтез в виде управляющих автоматов с программируемой логикой. Для исследования эффективности функционирования управляющих автоматов с программируемой логикой наиболее информативными являются такие характеристики как: требуемый объем памяти для хранения микропрограммы функционирования объекта управления, стоимость реализации управляющего автомата, его быстродействие и надежность работы. В зависимости о того, какой выбран способ адресации микрокоманд, какой или какие способы выбраны для кодирования поля микроопераций и т.д., требуется провести анализ реализации алгоритмов с различными параметрами. Для этого необходимо обладать достаточно большим набором исходных данных (в данном случае – алгоритмов логического управления), чтобы провести объективное сравнение вариантов. Число реальных алгоритмов управления, также как и число алгоритмов, которые можно сгенерировать в реальный интервал времени, ограничено и об объективности их сравнения не может быть и речи. В связи с чем вызывает интерес автоматическая генерация заданного количества алгоритмов с установленными параметрами. Следует отметить, что результаты сравнения, полученные на выборке случайных алгоритмов, необходимо проверить и на известных реальных примерах алгоритмов управления.

Целью статьи является описание метода, позволяющего осуществить генерацию алгоритмов с требуемыми характеристиками за приемлемое для пользователя время. При этом следует помнить, что в составе любого корректного алгоритма всегда можно выделить набор типовых фрагментов, из которых он составлен. К ним относятся начальный фрагмент, линейный участок, условное (альтернативное) ветвление, цикл.

Любой алгоритм можно представить как композицию вышеперечисленных фрагментов без изменения порядка следования операций. Для упрощения рассуждений не будут рассматриваться циклы с прерыванием, так как они могут быть получены комбинацией вышеперечисленных фрагментов. Каждый фрагмент может быть представлен как узел дерева. Соответствующие сопоставления приведены на рисунке 1. Здесь приняты следующие обозначения:

BE – сокращение от «Begin» для обозначения начального обязательного блока алгоритма;
LIN – сокращение от «Linear» для обозначения линейного блока;
ALT – сокращение от «Alternative» для обозначения блока условного ветвления;
LOOP – для обозначения цикла с постусловием, который является аналогом ALT, но имеет переход назад;
символом «*» отмечены дуги алгоритма, которые могут быть использованы для вставки любого из перечисленных выше фрагментов при генерации алгоритма. Наличие индексов указывает соответствие дуг в представленном фрагменте схемы алгоритма и в представлении узлов дерева фрагментов.

Не вызывает сомнения, что любой алгоритм, обычно представляемый в виде графа, может быть представлен в виде дерева перечисленных на рисунке 1 фрагментов [1].

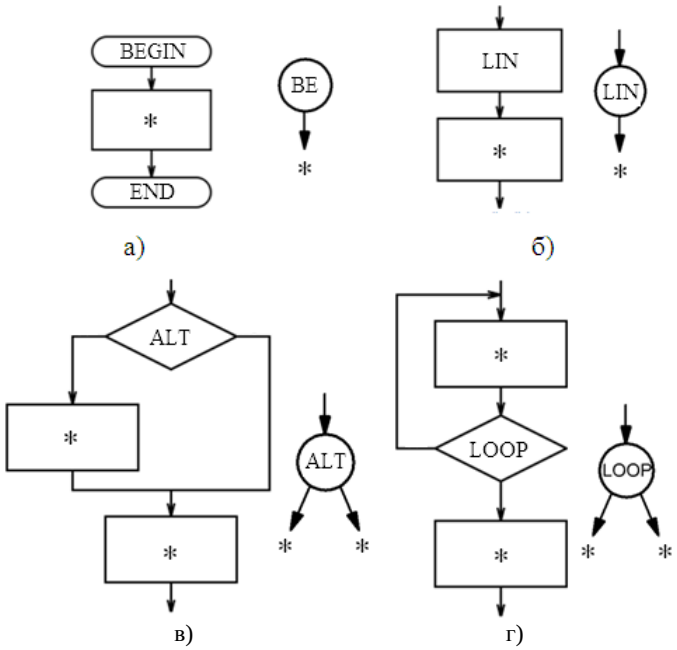

Рисунок 1 – Фрагменты схем алгоритмов:
а) начальный фрагмент; б) линейный фрагмент;
в) фрагмент «ветвление»; г) фрагмент «цикл»

В качестве примера на рисунке 2 показана схема простого алгоритма и её представление в виде дерева фрагментов.

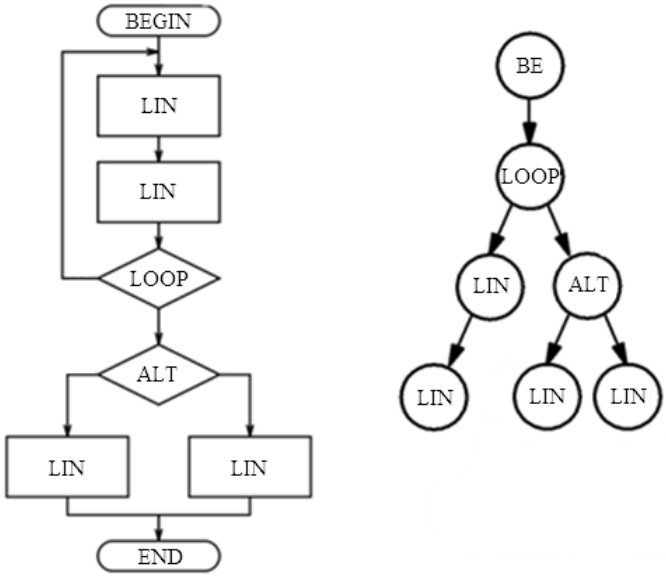

Рисунок 2 – Представление алгоритма в виде дерева фрагментов

На основании перечисленных выше предположений процесс случайной генерации алгоритмов логического управления сводится к двум этапам. Начальным этапом является построение дерева фрагментов сверху вниз (от корня к листьям), завершающим - генерация алгоритма по полученному дереву фрагментов.

Генерацию дерева фрагментов можно представить в виде следующей последовательности шагов:

1. Задать количество линейных блоков и блоков ветвления.
2. Добавить в дерево фрагментов начальный фрагмент.
3. Выбрать среди имеющихся в дереве фрагментов свободную дугу.
4. Выбрать тип добавляемого случайным образом фрагмента, определить его параметры (случайным образом задать состав микроопераций и логических условий, операторных и условных вершин). Добавить фрагмент в дерево.

5. Если условие завершения генерации (достигнуто заданное суммарное количество фрагментов) не выполнено, то перейти к пункту 3.
6. Конец.

На рисунке 3 приведена процедура добавления случайного фрагмента в дерево.

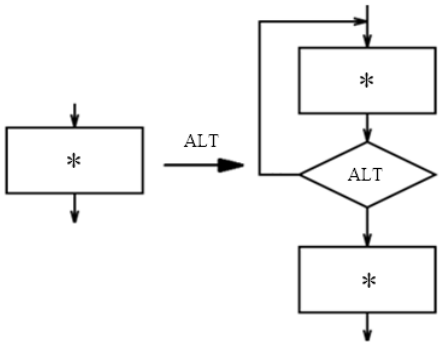

Рисунок 3 – Замена свободной дуги дочерним фрагментом

В результате выполнения первого этапа построено дерево фрагментов, которое может быть использовано непосредственно для синтеза алгоритма в виде графа. Алгоритм второго этапа выглядит следующим образом:

1. Создать список для хранения линеаризованного алгоритма.
2. Выполнить нисходящий обход дерева фрагментов с занесением в список их названий и параметров.
3. Вывести полученный список в листинг.
4. Конец.

Полученный список представляет собой линеаризованную запись алгоритма, которую можно впоследствии использовать для дальнейших целей – формировать псевдокод алгоритма с заданными параметрами (рисунок 4), либо производить преобразование в XML-файл специализированного формата (рисунок 5), позволяющий визуализировать алгоритм как блок-схему.

```
BEGIN
  ALT x3
    LOOP x5
      LIN y5 y11 y12 y2 y10
      LIN y8 y9 y1
      LIN y4 y12
    ENDLOOP
    PAR x4
    P
      LIN y10
    P2
      LIN y5 y6 y7
      LIN y4 y6
      LIN y12
      LIN y9 y11 y2 y8
      LIN y1 y10 y2
      LIN y6 y4 y11 y5
    ENDPAR
  ENDALT
END
```

Рисунок 4 – Пример программы на псевдокоде

Ниже приведен список ограничений, соблюдение которых обязательно для разработанной процедуры генерации алгоритмов.

1. Длины линейных и параллельных участков определяются случайным образом в соответствии с равномерным законом распределения.
2. Количество микроопераций в операционных вершинах определяются по дискретному распределению Пуассона. Выбор распределения Пуассона основан на следующих допущениях:

а) число микроопераций в реальных алгоритмах микропрограммного управления не достигает максимально возможных значений, так как это предусматривало бы параллельность исполнения всех микроопераций, исключение критерия стоимости оборудования и исключение из рассмотрения иных способов микропрограммирования, отличных от горизонтального;
б) любой микропрограммный алгоритм является последовательностью микрошагов и в каждый момент времени выполняется малое число микроопераций из всего многообразия доступных микроопераций [2];
в) число микроопераций не может быть отрицательной величиной;
г) минимальное число сигналов микроопераций предполагается равным одному;
д) число микроопераций – целая величина.

3. Количество логических условий в вершинах ветвления равно единице и задано жестко.

Однако принятые допущения не оказывают сколь-нибудь существенного влияния на объективность результатов дельнейших исследований, так как позволяют генерировать как линейные, так и разветвляющиеся алгоритмы, включая циклические. При этом в процессе генерации не нарушаются основные свойства алгоритмов, к числу которых относятся дискретность, детерминированность, элементарность, завершаемость, результативность. В этой связи можно сделать заключение, что разработанная процедура позволяет генерировать псевдокод, который является регулярным выражением и соответствует регулярной грамматике [3]. Следовательно, он может быть реализован на конечном автомате, каковым в конечном итоге и является управляющий автомат с программируемой логикой.

Так как графическое представление алгоритмов является наиболее наглядным и понятным, то в программную среду была введена дополнительная возможность генерации XML-файла, заданной структуры, что позволяет использовать его в стороннем приложении Algorithm Flowchart Editor для визуализации алгоритма в виде схемы. На рисунке 5 показан пример XML-файла и графическое представление схемы алгоритма.

```
<!DOCTYPE AFC><algorithm><branch>
<if cond="ALT" ><branch>
<post cond="LOOP" ><branch>
<process text="LIN" />
</branch></post>
<if cond="ALT" ><branch>
<process text="LIN" />
</branch><branch>
<process text="LIN" />
</branch></if>
</branch><branch></branch></if>
</branch></algorithm>
```

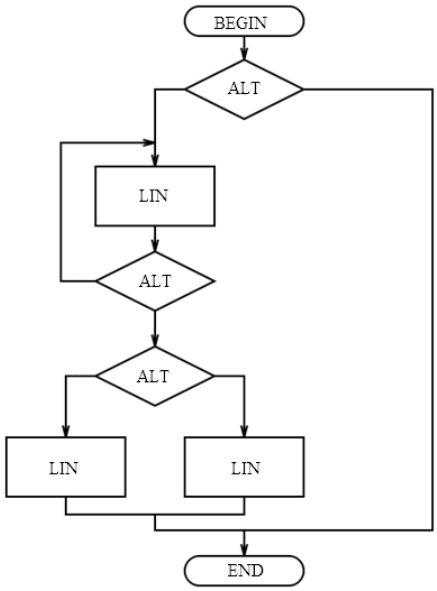

Рисунок 5 – Визуализация алгоритма

Генерация алгоритмов предусматривалась для исследования критериев эффективности системы проектирования управляющих автоматов с программируемой логикой. Предложенный вариант генерации случайных алгоритмов может быть адаптирован для иной программной системы, требующей в качестве исходных данных большого числа разнообразных алгоритмов. На рисунке 6 показан график времени генерации алгоритма в зависимости от его сложности. Под сложностью алгоритма подразумевается характеристика, зависящая от числа операций, от числа проверяемых условий и от числа сигналов управления. На рисунке приняты следующие обозначения: |y| – число микроопераций в сгенерированном алгоритме, m – число операционных блоков алгоритма, T – время генерации алгоритма с заданными характеристиками в секундах.

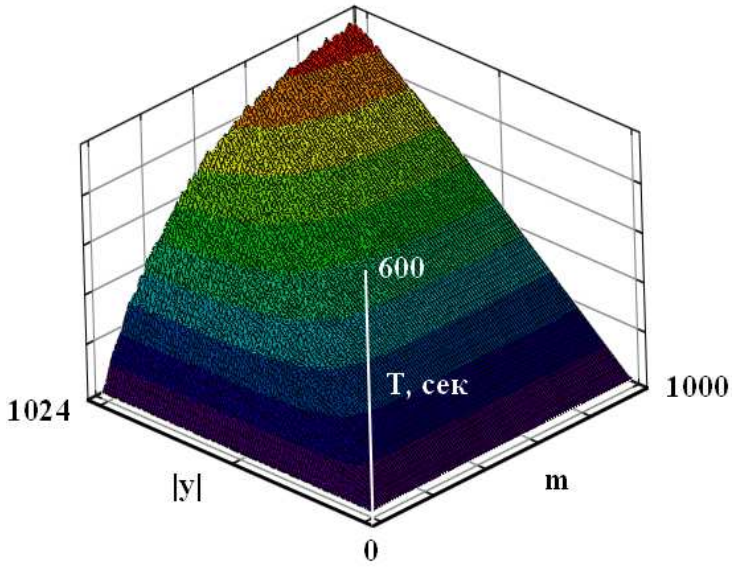

Рисунок 6 – График зависимости времени генерации от сложности алгоритма

В дальнейшем предполагается проведение исследований по классификации анализируемых в [4] алгоритмов функционирования управляющих автоматов с точки зрения структурной организации и функционального назначения.

Библиографический список использованной литературы

1. Майоров С.А. Принципы организации цифровых машин / С.А. Майоров, Г.И. Новиков. — Л.: Машиностроение, 1974. — 432 с.

2. Кормен Томас Х. Алгоритмы: построение и анализ / Х. Кормен Томас, И. Лейзерсон Чарльз, Л. Ривест Рональд, Штайн Клиффорд. — 2-е изд.: пер. с англ. — М.: Вильямс, 2005. — 1296 с.

3. Ахо Альфред В. Компиляторы: принципы, технологии и инструментарий / В. Ахо Альфред, С. Лам Моника, Сети Рави, Д. Ульман Джефри. — 2-е изд.: пер. с англ. — М.: Вильямс, 2008. — 1184 с.

4. Апраксин Ю.К. Программная среда разработки управляющих автоматов с программируемой логикой / Ю.К. Апраксин, Т.В. Волкова, В.О. Сикач // Оптимизация производственных процессов: сб. науч. тр. — Севастополь, 2013. — Вып. 14. — С. 189–192.

Поступила в редакцию 08.11.2013 г.

Апраксін Ю.К., Сікач В.О. Автоматична генерація випадкових алгоритмів із заданими параметрами

Пропонується система генерації випадкових алгоритмів із заданими характеристиками. Наводяться приклади згенерованих алгоритмів і результати моделювання.

Ключові слова: алгоритм, генерація алгоритмів, автоматизація, випадковий алгоритм.

Apraksin Yu.K., Sikach V.O. Automatic generation of random algorithms with preset parameters

A system of generating random algorithms with preset characteristics. The examples of the generated algorithms and simulation results.

Keywords: algorithm, algorithms generator, automation, random algorithm.