УДК 004.3:681.518

Д.Е. ИВАНОВ*

РАЗРАБОТКА ШАБЛОНОВ ЭВОЛЮЦИОННЫХ МЕТОДОВ ДИАГНОСТИРОВАНИЯ ЦИФРОВЫХ УСТРОЙСТВ

*Институт прикладной математики и механики НАН Украины, Донецк, Украина

Анотація. *У статті пропонується класифікація існуючих еволюційних методів генерації ідентифікуючих послідовностей цифрових пристроїв, а також розробка на базі цієї класифікації шаблонів побудови таких методів та їх алгоритмічної реалізації. Подібна класифікація та побудовані шаблони є частиною методології синтезу нових ефективних еволюційних методів діагностування цифрових пристроїв, яку направлено на прискорення розробки нових методів за рахунок уніфікації їх компонент.*
Ключові слова: *надійність, діагностування, цифрові пристрої, еволюційні обчислення, класифікація, генетичний алгоритм, симуляція відпалу.*

Аннотация. *В статье предлагается классификация существующих эволюционных методов генерации идентифицирующих последовательностей цифровых устройств, а также разработка на основе данной классификации шаблонов построения таких методов и их алгоритмической реализации. Подобная классификация и построенные шаблоны являются частью методологии синтеза эволюционных методов диагностирования цифровых устройств, которая направлена на ускорение разработки новых методов за счёт унификации их компонент.*
Ключевые слова: *надёжность, диагностирование, цифровое устройство, эволюционные вычисления, классификация, генетический алгоритм, моделирование отжига.*

Abstract. *The paper proposes classification of existing evolutionary methods for generating identifying sequences of digital devices as well as development the templates for constructing such methods and their algorithmic implementation, which are based on this classification. Such classification and built templates are part of the methodology for the synthesis of evolutionary methods of diagnosing digital devices aimed to accelerate the development of new methods by unifying their components.*
Keywords: *reliability, diagnostics, digital device, evolutionary computation, classification, genetic algorithm, simulated annealing.*

1. Введение

В настоящее время весь процесс проектирования цифровых устройств (ЦУ) выполняется с помощью специализированных САПР. Это позволяет проектировать высоконадёжные ЦУ. На различных этапах данного процесса разработчик сталкивается с необходимостью строить входные идентифицирующие последовательности (ИдП) различных классов: характеристические, тестовые и т.д.

Традиционные методы построения таких типов последовательностей для последовательностных ЦУ являются адаптацией соответствующих методов, разработанных для комбинационных устройств. Присущие таким методам внутренние недостатки породили новую эволюционную парадигму разработки методов.

Наиболее часто исследователи для решения задач диагностирования ЦУ используют генетические алгоритмы (ГА) [1–2]. Цель ГА – повторить природный механизм улучшения свойств особей за счёт адаптации к решению задачи природных эволюционных механизмов: скрещивание, выживаемость, приспособленность и т.д.

Одной из сильнейших в данном направлении является школа университета Торино, Италия. Её исследователи, по сути, были пионерами в данной области и предложили целый ряд ГА построения ИдП ЦУ [3–5]. В частности, разработаны алгоритмы генерации проверяющих тестов, инициализирующих последовательностей, верификации эквивалент-

ISSN 1028-9763. Математичні машини і системи, 2014, № 3

ности поведения для синхронных последовательностных ЦУ. В качестве особи в алгоритмах выступает двоичная входная последовательность, а набор особей формирует популяцию. К особям применяется набор эволюционных операций: селекция, скрещивание, мутация.

В таком же ключе работала и группа Иллинойского университета [6–7]. В целом подход оказался очень удачным и получил развитие в многочисленных работах других авторов [9–10].

Значимый вклад в данную область сделали также отечественные авторы [11–12]. При этом заметны усилия в разработке параллельных версий методов [13], расширении ГА на кратную стратегию наблюдения [14] и в тестировании новых типов неисправностей [15].

Часто разрабатываемые методы очень близки идеологически и отличаются только в некоторых деталях: метод моделирования, используемый для оценки особей, применяемые эвристики и т.п.

Ещё одной часто используемой эволюционной парадигмой является метод (стратегия) симуляции отжига (СО) [16]. Основным отличием данной парадигмы от ГА является эволюция одного потенциального решения, которое называется конфигурацией. Процесс поиска оптимального решения строится таким образом, что в его начале происходят частые возмущения решения (ухудшение свойств), а в конце поиска они сходят на нет. Применение СО к решению задач диагностирования прошло, в общем, тот же путь, что и ГА [17–18], а разработанные методы показывают достаточно высокую эффективность в терминах соответствующих задач.

Традиционно к родственным методам искусственного интеллекта также относятся муравьиные алгоритмы [19], алгоритмы роя пчёл (роевого интеллекта) [20] и целый ряд малоизвестных и редко применяемых в задачах диагностики метаэвристик, например, табу поиск [21]. Алгоритмы данного рода применялись к решению задач диагностики цифровых схем [22] и в ряде случаев показывали хорошие результаты [23, 24]. Однако мы не будем рассматривать их в данной работе, поскольку, во-первых, их применение в задачах технической диагностики очень ограничено, а во-вторых, строго говоря, они не являются эволюционными алгоритмами (ЭА) по устоявшейся классификации [2].

Особенностью всех упомянутых выше работ является то, что разработка методов основана на эвристическом подходе. При разработке нового метода и, исходя из условия задачи, выбираются кодирование, эволюционные операции, строятся оценочные функции и т.д. То есть полностью выполняется процесс, характерный для разработки одиночного генетического алгоритма решения конкретной задачи.

С другой стороны, представленная база разработанных методов позволяет сделать соответствующие обобщения и сформировать методологию синтеза эволюционных методов построения ИдП. Исследования в данном направлении не проводились. Между тем такая методология должна за счёт унификации подхода и разработки единых компонент эволюционных методов повысить скорость и качество разработки новых методов.

Частью такой методологии является классификация (иерархия) эволюционных методов, отражающая понимание их места в решении соответствующих задач генерации ИдП. В данной статье предлагается вариант подобной иерархии эволюционных методов, включающий популяционные и непопуляционные парадигмы, а также шаблоны методов компонент такой иерархии.

2. Одно- и двухуровневые эволюционные алгоритмы построения идентифицирующих последовательностей цифровых устройств

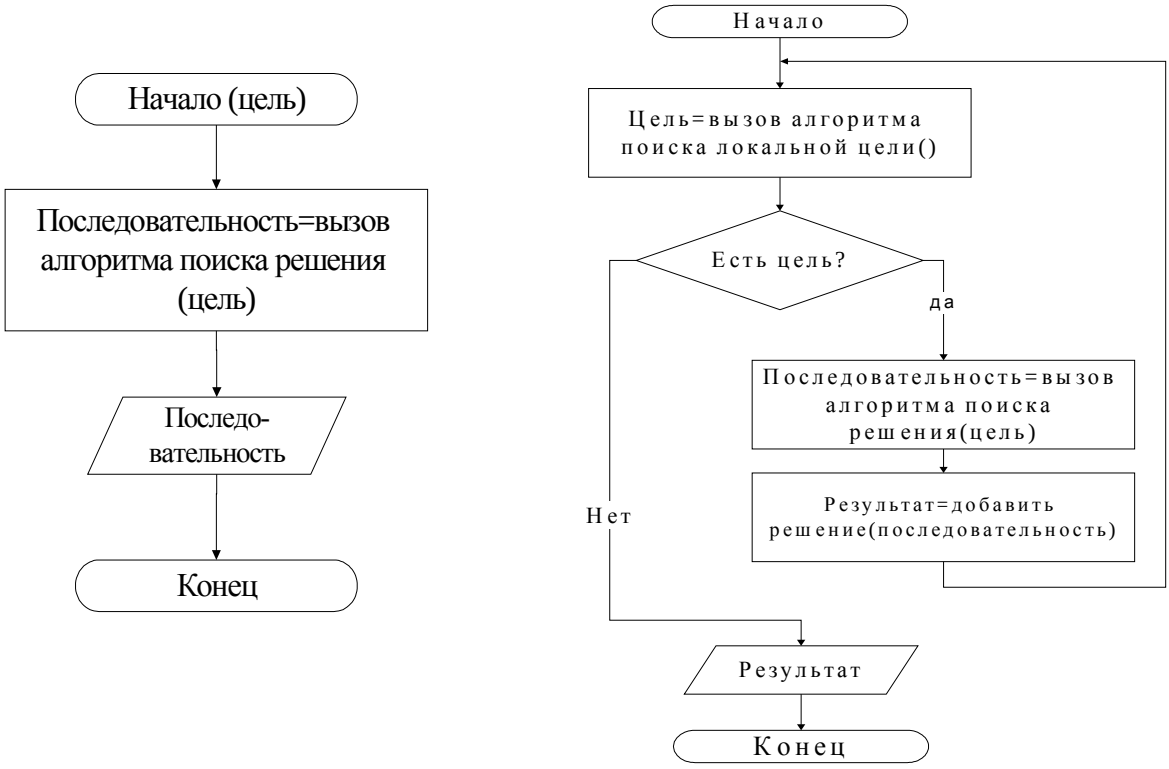
В зависимости от сложности задачи построения ИдП структуры эволюционных алгоритмов для их решения предлагается разделять на два больших класса [12]:

– одноуровневые ЭА;

– двухуровневые ЭА.

В том случае, если метод поиска решения (построения последовательности) может его найти за один вызов ЭА, будем говорить об одноуровневых эволюционных алгоритмах (рис. 1а) или одноуровневой схеме применения ЭА.

Особенностью одноуровневых эволюционных методов является то, что их цель задаётся только один раз и она известна до начала выполнения алгоритма. Для такого ЭА формализация цели выражается в виде оценочной функции потенциальных решений. Достижение этой цели показывает завершение работы алгоритма в целом. Фактически весь метод построения последовательности и является эволюционным алгоритмом. Таким образом, структура методов данного рода такова, что основной цикл эволюции построения новых решений является в нём также самым внешним.



а) одноуровневая б) двухуровневая

Рис. 1. Модели применения ЭА

К задачам, решение которых строится на основании одноуровневой модели, можно отнести частные случаи задач построения характеристических последовательностей различного типа:

– инициализирующие последовательности: переводят устройство из начального неопределённого состояния в заданное стартовое;

– последовательности достижения состояний (ПДС): определяющим здесь является то финальное состояние, которое должно быть достигнуто после приложения входной последовательности;

– верификации эквивалентности поведения двух заданных устройств;

– оценки параметров рассеивания тепла заданным устройством (пиковое однотактное, n-тактное и устойчивое).

В том случае, если сложность задачи не позволяет методу найти решение за один

вызов ЭА поиска, будем говорить о двухуровневых методах (рис. 1б) или двухуровневой схеме применения ЭА. Методы данного класса предполагают итеративную схему построения. Каждая итерация состоит из двух фаз. В первой фазе происходит поиск промежуточной (локальной) цели. Если такая цель найдена, то вызывается ЭА поиска решения для данной локальной цели, который и формирует вторую фазу итерации. Итеративный поиск промежуточных целей и их достижение ведёт к решению общей задачи. В такой постановке будем называть фазу 1 поиска локальной цели верхним уровнем ЭА, а фазу 2 достижения локальной цели – нижним уровнем ЭА. При этом структура фазы 2 алгоритма соответствует одноуровневому ЭА построения ИдП. В задачах построения входных ИдП конечное решение (последовательность) часто строится по аддитивному принципу (является совокупностью промежуточных решений). То есть данные задачи естественным образом проецируются на двухуровневую схему ЭА.

К классу двухуровневых ЭА построения ИдП относятся:

– методы построения проверяющих тестов, использующие различные стратегии (подтверждение состояния на основе ГА или распространение влияния неисправности на основе ГА);

– метод построения диагностических тестов;

– метод построения энергоэффективных тестов ЦУ, который является модификацией метода построения тестов; здесь для строящихся последовательностей изучаются не только проверяющие свойства, но и дополнительные, в частности, параметр рассеивания тепла.

В качестве эволюционных алгоритмов поиска в одно- и двухуровневой схемах применения могут выступать как алгоритмы, основанные на эволюции популяции решений, так и алгоритмы с одиночной эволюцией решений. В качестве популяционных методов будем рассматривать ГА. Методы с эволюцией одиночного решения представлены методом СО.

Таким образом, множество эволюционных алгоритмов построения ИдП может быть представлено в виде иерархии, которая приведена на рис. 2. Классификационными признаками ЭА в данной иерархии являются:

– популяционность эволюции решения: популяция либо одно решение;

– уровневость алгоритма: одно- либо двухуровневый.

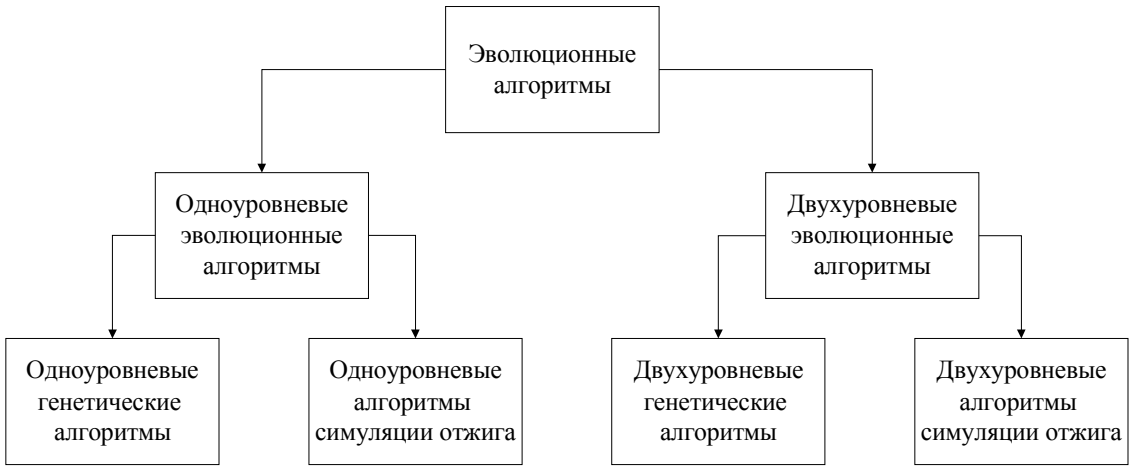


Рис. 2. Классификация эволюционных алгоритмов построения ИдП

Также далее для разрабатываемых методов, которые используют ГА в качестве метода поиска, будем использовать название ГА-метод, а для алгоритмов с поиском на основе алгоритма симуляции отжига – СО-метод.

Далее в статье будут разработаны шаблоны одно- и двухуровневых ГА-методов построения ИдП ЦУ. Аналогично разрабатываются шаблоны соответствующих СО-методов.

3. Шаблон одноуровневых ГА построения идентифицирующих последовательностей цифровых устройств

Формально ГА задаётся следующим образом. Пусть Ind – множество особей; $Pop = \{pop \mid pop \subseteq Ind, |pop| < \infty\}$ – множество возможных популяций конечного размера. Тогда ГА есть упорядоченная совокупность объектов:

$$\text{ГА}=(Ind, Sel, Cross, Mut, O, Fit, pop_{нач}, N_{особ}, l, p_{скр}, p_{мут}),$$

где $Sel : Pop \to Ind$ – операция селекции: выбирает из заданной популяции одну (или несколько) особь(ей) для выполнения генетических операций; обычно более высокую вероятность быть выбранными в качестве родителей имеют особи с большей фитнес-функцией;

$Cross : Ind \times Ind \to Ind$ – операция скрещивания: по двум заданным особям строит новую особь в соответствии с выбранным правилом скрещивания;

$Mut : Ind \to Ind$ – операция мутации: строит новую особь, применяя к заданной особи правило мутации;

$O : Ind \to R$ – оценочная функция;

$Fit : Ind \times Pop \to R$ – фитнес-функция; оценочную функцию следует отличать от фитнес-функции, поскольку последняя показывает качество особи относительно других в популяции. Для её вычисления необходимо знать не только оценку особи, но и оценку всех других особей в популяции;

$pop_{нач} \subseteq Ind$ – начальная популяция особей; часто при реализации строится случайным образом;

$N_{особ}(pop) = |pop|$ – размер популяции, который задаёт число особей, входящих в популяцию pop;

l – длина особи в битах при двоичном кодировании;

$p_{скр}$ и $p_{мут}$ – вероятности применения операторов скрещивания и мутации соответственно.

Цель ГА – поиск особи с наивысшей оценкой: $O \to \max$.

Видно, что для реализации конкретного ГА-ориентированного метода построения ИдП целый ряд его компонент необходимо задавать конструктивно: оператор репродукции, оценочная функция, способ построения фитнес-функции и т.д. Часто при построении соответствующего метода ГА конкретная реализация таких компонент алгоритма предполагает их экспериментальное обоснование. В конечную реализацию метода входит тот вид оператора, функции и т.п., при котором получаются наилучшие числовые результаты. Будем называть такие компоненты зависящими от реализации.

Абстрагируясь от реализации таких компонент, можно разработать обобщённый шаблон ГА-методов построения ИдП, использующих одноуровневую модель применения. Такой ГА-шаблон будет включать как кодирование особей, набор генетических операций (селекция, скрещивание, мутация), так и базовую структуру метода ГА. При этом зависящие от реализации компоненты должны быть только названы, а их конкретное наполнение вынесено за данный шаблон и отнесено к построению конкретного метода, его алгоритмической реализации и настройке.

Пусть на основании целей метода выбраны кодирование особей-последовательностей, кодирование популяций и эволюционные операции (селекция, скрещивание, мутация). Данные компоненты хорошо известны и описаны, например, в [12].

Также для дальнейшего построения шаблона одноуровневого ГА-метода будем считать, что для каждой особи-последовательности S известен метод вычисления её оценочной функции $O(A_0, S)$, конструктивно заданный процедурой *ОценитьОсобь*(A_0, S), где A_0 – обрабатываемое ЦУ. Наиболее часто при вычислении оценочных функций происходит явное моделирование поведения устройства A_0 при подаче на его вход последовательности S. В зависимости от конкретной задачи может использоваться как исправное моделирование, так и моделирование с неисправностями [25].

ГА в целом представляет собой итеративное построение новых популяций потенциальных решений, которые выполняются до выполнения критерия остановки. Укрупненный псевдокод шаблона одноуровневого ГА-метода построения ИдП может быть представлен в следующем виде.

Алгоритм А1

Одноуровневый_ГА_построения_ИдП(A_0 *, Параметры)*
{
ПредварительнаяОбработка(A_0 *);*
$Pop_{нач}$ *=ПостроениеНачальнойПопуляции(* $N_{особ}, L$ *);*
ОценитьПопуляцию($Pop_{нач}, A_0, N_{особ}$ *);*
$Pop_{тек} = Pop_{нач}$ *;*
НомерПопуляции=0;
// основной цикл по поколениям
while(НеДостигнутКритерийОстановки())
{
ВычислитьФитнес-Функцию($Pop_{тек}, N_{особ}$ *);*
// цикл построения промежуточной популяции
while(СтроитсяНоваяПопуляция ())
{
Родители=ОперацияСелекции();
Потомки=ОперацияСкрещивания(Родители);
Потомки=ОперацияМутации(Потомки);
ДобавлениеВНовуюПопуляцию(Потомки);
} // конец while – построение новой популяции
$Pop_{тек}$ *=ПостроитьНовуюПопуляцию();*
ОценитьПопуляцию($Pop_{тек}, A_0, N_{особ}$ *);*
НомерПопуляции++;
АдаптацияПараметров();
} // конец while – достигнут критерий остановки
СортироватьПопуляциюПоОценке($Pop_{тек}$ *);*
// решение=лучшая особь в посл. популяции
Решение=ТекущаяПопуляция[0];
} // конец одноуровневого ГА построения ИдП

Дадим пояснения к псевдокоду метода. Здесь использованы следующие переменные: A_0 – обрабатываемое цифровое устройство, $Pop_{нач}$, $Pop_{тек}$ – начальная популяция и текущая популяция итерации соответственно, L – начальная длина строящихся особей-последовательностей, показывающая число входных наборов, *Родители* – особи, выбранные для генетических операций скрещивания и мутации, *Потомки* – особи, являющиеся результатом генетических операций, *НомерПопуляции* – счётчик итераций, показывает номер текущей популяции.

В коде функции имеют следующую нагрузку:

ПостроениеНачальнойПопуляции() – реализует стратегию построения начальной популяции особей;

ОценитьПопуляцию() – для всех особей в заданной популяции вычисляет оценочную функцию;

ОперацияСелекции(), ОперацияСкрещивания(), ОперацияМутации() – реализуют генетические операции селекции, скрещивания и мутации соответственно;

ДобавитьОсобьВПопуляцию() – добавляет особь в заданную популяцию;

ПредварительнаяОбработка() – вычисляет статические параметры оценочной функции [26].

Видно, что приведённый псевдокод реализует основной цикл ГА поиска решения. Итерации построения новых популяций внутри ГА прекращаются при выполнении одного из следующих условий: найдено точное решение задачи, достигнуто предельное число итераций, заданное число итераций не происходит улучшения оценки лучшей особи. После остановки итераций решением задачи является последовательность-особь с лучшей оценкой в последней достигнутой популяции.

При таком построении шаблона ГА-методов все зависящие от реализации механизмы являются скрытыми в соответствующих процедурах и конкретизируются при реализации.

Таким образом, на основании этого шаблона далее частные ГА-методы построения ИдП будут строиться путём задания оценочной функции и наполнения зависящих от реализации компонент. Именно на основании данного шаблона построена реализация всех одноуровневых ГА-методов, перечисленных в разд. 2. Фактически они различаются видом оценочной функции для особи-последовательности и применяемыми эвристиками.

4. Шаблон двухуровневых ГА построения идентифицирующих последовательностей цифровых устройств

Одноуровневые ГА-методы построения ИдП следует рассматривать как первый этап в применении ГА к решению задач технической диагностики. Однако целый ряд задач не может быть сведён к однократному вызову ГА поиска в силу того, что решение задачи требует поиска нескольких промежуточных решений. Поскольку в процессе поиска в разное время ставятся различные подзадачи, то в оценочной функции появляется параметр (параметры), который зависит от текущей локальной цели (рис. 1б). Мы рассматриваем случаи, когда поиск решения для локальных целей сводится к вызову процедуры, в той или иной степени использующей эволюционный поиск на основе ГА. Такую схему построения решения задачи будем называть двухуровневой схемой применения ГА, а разработанные на её основе методы – двухуровневыми ГА-методами.

Разработаем шаблон верхнего уровня двухуровневых методов построения ИдП, в котором в качестве эволюционной поисковой процедуры выступает ГА.

Укрупнённая блок-схема двухуровневых ГА-методов приведена на рис.1б. Центральным элементом всех методов, основанных на данной схеме, является объект *Цель*. Природа данного объекта уточняется при построении конкретного метода. Главный цикл верхнего уровня таких методов включает два шага:

– поиск текущей *Цели*; если невозможно найти такую цель, то работа метода завершается;

– поиск решения, которое ведёт к достижению текущей *Цели*.

Пусть заданы ЦУ A_0 и кодирование решений. Шаблон верхнего уровня для двухуровневых ГА-методов построения ИдП может быть представлен следующим образом.

Алгоритм А2

```
Двухуровневый_ГА_построения_ИдП( A0 , Параметры)
{
 ПредварительнаяОбработка( A0 );
 while( НеДостигнутКритерийОстановки() )
 {
   Цель =ВыборТекущейЦели( A0 , Параметры);
  if( Цель == НетЦели )
   return; // нет цели – завершение работы алгоритма
  else
  {
   // цель найдена–вызов ГА поиска локального решения
    S =ГА_Построение_ИдП( A0 , Цель ,НачальнаяПопуляция);
   if( S ==NULL ) // последовательность не построена
    ОтметитьКакНедостижимую( Цель );
   else
   {
    ДобавитьВТест( S );
    ДополнительнаяПроверка( A0 , S );
   } // конец else – последовательность построена
  }   // конец else – выбрали цель
 }    // конец while – не достигли критерия остановки
}     // конец двухуровневого ГА-метода
```

Предварительная обработка, представленная в коде одноименной процедурой, может включать в себя построение множества промежуточных *Целей*, подготовку структуры данных для работы с ними в дальнейшем (списки, массивы и т.д.), а также вычисление статических параметров оценочных функций [26].

Если в итерации текущая *Цель* найдена, то вызывается ГА-метод, который строит промежуточную последовательность S. При этом структурно такой ГА является одноуровневым и соответствует шаблону метода А1. Начальная популяция для ГА нижнего уровня может строиться, например, в процессе поиска *Цели* на верхнем уровне. Такой подход реализован в [3, 9, 12].

После построения промежуточной последовательности S и в зависимости от конкретного метода может быть выполнено дополнительное моделирование [3, 12]. Введение такой эвристики связано с тем, что построенная для некоторой локальной цели промежуточная последовательность S может обладать более широкими идентифицирующими свойствами. Именно тот факт и устанавливается в процедуре *ДополнительнаяПроверка()*.

Результирующее решение строится по аддитивному принципу: итоговая последовательность состоит из совокупности промежуточных последовательностей. К ней могут применяться различные процедуры оптимизации.

Ещё одной особенностью данного шаблона является то, что ГА используется именно для достижения *Цели*. Вообще говоря, ГА может использоваться и на этапе её поиска, но для нас такое его применение является вторичным.

Данный шаблон является обобщающим для достаточно широкого класса методов. Применение различных типов объектов *Цель* и различных процедур ГА-поиска на нижнем уровне порождает целое множество различных методов.

Покажем, как это работает, на примерах. В задаче построения проверяющих тестов множество *Целей* – множество неисправностей. Здесь на верхнем уровне в качестве *Цели*

выбирается одна непроверенная неисправность, для которой на нижнем уровне строится проверяющая последовательность. Фактически на нижнем уровне рассматривается задача верификации эквивалентности поведения двух ЦУ (исправного и неисправного), метод решения которой строится по одноуровневому шаблону А1.

В задаче построения диагностических тестов в качестве объекта *Цель* выступает множество неразличимых неисправностей. На нижнем уровне с помощью одноуровневого ГА строится последовательность, которая разбивает данное множество на меньшие, вплоть до таких, которые содержат ровно одну неисправность. Отличием здесь также является то, что оценочная функция показывает различие поведения не для пары, а для множества устройств.

Двухуровневая схема применима не только для случая, когда целью верхнего уровня выступает неисправность или набор неисправностей. Например, в задаче тестирования функциональных блоков арифметико-логического устройства (АЛУ) [9, 12] в качестве *Цели* выступает отдельная операция функционального уровня. Тогда на нижнем уровне модели для данной операции строится такой набор операндов, при подаче которых на вход при рассматриваемой схеме тестирования будет получено максимальное покрытие множества рассматриваемых неисправностей структурного уровня. То есть задача нижнего уровня снова сведена к задаче верификации поведения устройства.

Анализ известных двухуровневых ГА-методов [1, 3–12] показывает, что на нижнем уровне используются два основных одноуровневых ГА-метода: достижения заданного состояния в ЦУ и верификации эквивалентности поведения двух или множества устройств.

Таким образом, алгоритмическая реализация всех известных двухуровневых ГА-методов строится на основании шаблонов алгоритмов А1 и А2.

Поскольку в шаблоне А2 показывается только место процедур поиска на основе ГА, то без потери общности данный шаблон может быть применён к случаю, когда поиск решения на нижнем уровне будет осуществляться с помощью других ЭА, например СО. С этой точки зрения данный шаблон можно рассматривать общим и для двухуровневых ГА-методов и двухуровневых СО-методов.

5. Выводы

В статье предложена классификация эволюционных методов построения идентифицирующих последовательностей ЦУ различных классов. На основании данной классификации разработаны шаблоны одно- и двухуровневых эволюционных методов, которые служат непосредственной основой их алгоритмической реализации. Это позволяет ускорить разработку новых эффективных методов генерации диагностических последовательностей различных классов, необходимость в которых возникает при проектировании ЦУ.

Очевидным дальнейшим развитием приведённой иерархии должно быть включение в неё параллельных методов. И если для ГА построения ИдП параллельные методы развиты достаточно хорошо (схемы «островов» и «хозяин-рабочий»), то апробация параллельных СО-методов к решению рассматриваемых задач ещё только предстоит.

СПИСОК ЛИТЕРАТУРЫ

1. Goldberg D.E. Genetic Algorithm in Search, Optimization and Machine Learning / Goldberg D.E. – Boston: MA:Addison-Wesley Longman Publishing Co., 1989. – 412 p.
2. Скобцов Ю.А. Основы эволюционных вычислений / Скобцов Ю.А. – Донецк: ДонНТУ, 2008. – 326 с.
3. Corno F. Experiences in the use of evolutionary techniques for testing digital circuits / F. Corno, M. Sonza Reorda, M. Rebaudengo // Proc. of Conf. Applications and science of neural networks, fuzzy systems, and evolutionary computation. – San Diego CA, 1998. – P. 128 – 139.
4. Corno F. Comparing topological, symbolic and GA-based ATPGs: an experimental approach / F. Cor-

no, P. Prinetto, M. Rebaudengo [et al.] // Proceedings of the IEEE International Test Conference on Test and Design Validity. – Washington (USA), 1996. – P. 39 – 47.
5. Corno F. A Genetic Algorithm for the Computation of Initialization Sequences for Synchronous Sequential Circuits / F. Corno, P. Prinetto, M. Rebaudengo [et al.] // Proc. ATS '01 Proceedings of the 10th Anniversary Compendium of Papers from Asian Test Symposium 1992 – 2001. – 2001. – P. 213.
6. Hsiao M.S. Dynamic state traversal for sequential circuit test generation / M.S. Hsiao, E.M. Rudnick, J.H. Patel // ACM Transactions on Design Automation of Electronic Systems (TODAES). – 2000. – Vol. 5, Is. 3. – P. 548 – 565.
7. Yu X. Diagnostic Test Generation for Sequential Circuits / X. Yu, J. Wu, E.M. Rudnick // Proc. of International Test Conferenece. – Atlantic City, NJ, 2000. – P. 225 – 234.
8. Srinivas M. A simulation-based test generation scheme using genetic algorithms / M. Srinivas, L.M. Patnaik // Proc. Int. Conf. VLSI Design. – Bombay, India, 1993. – P. 132 – 135.
9. Skobtsov Y.A. Evolutionary Approach to Test Generation for Functional BIST / Y.A. Skobtsov, D.E. Ivanov, V.Y. Skobtsov [et al.] // 10 European Test Symposium. Informal Digest of Papers. Digest of Papers, (May 22 – 25, 2005). – Tallinn, Estonia, 2005. – P. 151 – 155.
10. Saab D.G. Iterative [Simulation-Based+Deterministic Techniques] = Complete ATPG / D.G. Saab, Y.G. Saab, J. Abraham // Proc. Int. Conf. on Computer Aided Design. – San Jose, CA, 1994. – P. 40 – 43.
11. Иванов Д.Е. Генетические алгоритмы в диагностике и проектировании цифровых схем / Д.Е. Иванов, Ю.А. Скобцов, В.Ю. Скобцов // Искусственный интеллект. – 2002. – № 2. – С. 250 – 258.
12. Иванов Д.Е. Генетические алгоритмы построения входных идентифицирующих последовательностей цифровых устройств / Иванов Д.Е. – Донецк, 2012. – 240 с.
13. Ivanov D.E. Distributed Genetic Algorithm of Test Generation For Digital Circuits / D.E. Ivanov, Y.A. Skobtsov, A.I. El-Khatib // Proc. of the 10th Biennial Baltic Electronics Conference. – Tallinn Technical University, 2006. – P. 281 – 284.
14. Скобцов Ю.А. Генерация тестов для последовательностных схем с использованием кратной стратегии наблюдения выходных сигналов / Ю.А. Скобцов, В.Ю. Скобцов, Ш.Н. Хинди // Науковий вісник Чернівецького університету. – (Серія «Фізика. Електроніка»). – 2008. – Вип. 423. – С. 29 – 36.
15. Skobtsov Yu.A. The crosstalk faults test generation / Yu.A. Skobtsov, V.Yu. Skobtsov // Proc. of IEEE East-West Design&Test Symposium (EWDTS'10). – St. Peterburg, Russia, 2010. – P. 141 – 144.
16. Kirkpatrick S. Optimization by simulating annealing / S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi // Science. – 1983. – Vol. 220. – P. 671 – 680.
17. SAARA: a simulated annealing algorithm for test pattern generation for digital circuits / F. Corno, P. Prinetto, M. Rebaudengo [et al.] // Proc. of the 1997 ACM symposium on Applied computing. – San Jose, California, 1997. – P. 228 – 232.
18. Иванов Д.Е. Применение алгоритмов симуляции отжига в задачах идентификации цифровых схем / Д.Е. Иванов // Вісник Національного технічного університету "Харківський політехнічний інститут". Збірник наукових праць. Тематичний випуск: Інформатика і моделювання. – Харків: НТУ "ХПІ", 2011. – № 17. – С. 60 – 69.
19. Maniezzo V. Ant colony optimization: an overview / V. Maniezzo, A. Carbonaro // Essays and Surveys in Metaheuristics. – Norwell:Kluwer Academic Publishers, 2002. – P. 469 – 492.
20. Blum C. Swarm Intelligence in Optimization. Introduction and Applications. Natural Computing Series / C. Blum, Li Xiaodong. – Springer-Verlag Berlin Heidelberg, 2008. – P. 43 – 85.
21. Diaz E. A Tabu Search Algorithm for Structural Software Testing Computers / E. Diaz, J. Tuya // Operations Research. – 2008. – N 35 (10). – P. 3052 – 3072.
22. Xiaojing H. Ant Colony Optimizations for Initialization of synchronous sequential circuits / H. Xiaojing, S. Zhengxiang // Proc. of IEEE Circuits and Systems International Conf. – Chengdu, China, 2009. – P. 5 – 18.
23. Farah R. An Ant Colony Optimization Approach for Test Pattern Generation / R. Farah, H.M. Harmanani // Canadian Conference on Electrical and Computer Engineering (CCECE). – Niagara Falls. – 2008. – May. – P. 4 – 7, 1397 – 1402.
24. Xin F. A Sequential Circuits Test Set Generation Method Based on Ant Colony Particle Swarm algorithm / F. Xin, F. Shuai // Proc. of National Conference on Information Technology and Computer Science (CITCS 2012). – Atlantis Press, 2012. – P. 205 – 209.

25. Скобцов Ю.А. Логическое моделирование и тестирование цифровых устройств / Ю.А. Скобцов, В.Ю. Скобцов. – Донецк: ИПММ НАНУ, ДонНТУ, 2005. – 436 с.
26. Иванов Д.Е. Применение информации структурного уровня в алгоритмах построения идентифицирующих последовательностей / Д.Е. Иванов // Известия ЮФУ. – (Серия «Технические науки»). – 2013. – № 1. – С. 149 – 160.

Стаття надійшла до редакції 23.10.2013