1. $\forall \nu_k \in L_k : q_k(\nu_k) \le h_k \Rightarrow \max_{\bar{\nu}_k \in G_k} f_k(\bar{\nu}_k)$;

2. $\forall y < h_k \; \exists \bar{\nu}_k \in L_k : y < q_k(\bar{\nu}_k) \le h_k$.

Эти условия являются необходимыми и достаточными для того, чтобы $\max_{\nu_k \in L_k} q_k(\nu_k) = q_k(\nu_k') = h_k$, $k = \overline{1,r}$. Т.к. по условию каждое из множеств $X_r(\bar{i}_r)$ в (25) при $k = r$ ограничено, то получим искомый максимум $\max_{x \in X_r} c^T x = \max_{\bar{\nu}_r \in G_r} f_r(\bar{\nu}_k) \Rightarrow h_r$, а значит, если условия 1 и 2 установлены при $k = \overline{1,r}$, то их выполнение для $k = r$ доказывает полученный результат.

Список источников

1. Хачиян Л.Г. Полиномиальный алгоритм в линейном программировании. – ДАН СССР, №5, 1979. – 244 с.

2. Судаков Р.С. Теория псевдополуобратных матриц и ее применение к задачам надежности. – М.: Знание. – 1981. – 107 с.

ПРИМЕНЕНИЕ АДАПТИВНИХ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ ДЛЯ ГЕНЕРАЦИИ ТЕСТОВ ЦИФРОВЫХ СХЕМ.

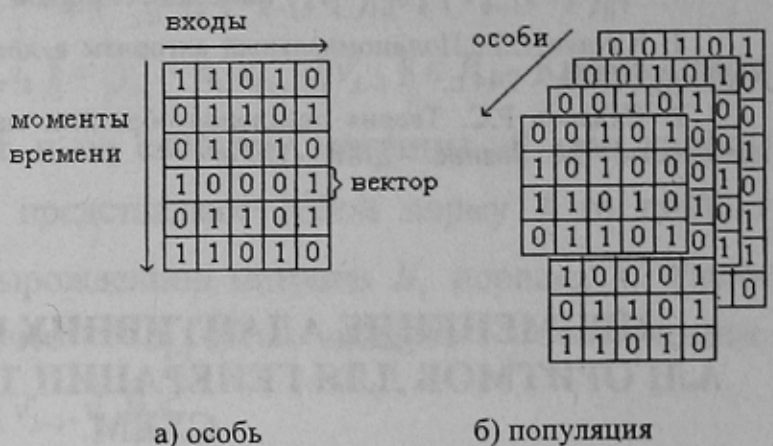
Скобцов Ю.А., Иванов Д.Е., Скобцов В.Ю., Закусило С.А.
Донецкий национальный технический университет,
Институт прикладной математики и механики НАН Украины

В настоящее время по-прежнему остается актуальной задача генерации тестов для последовательных цифровых устройств (ЦУ). При генерации тестов схем с памятью применяются три основных подхода: алгоритмы, основанные на расширении метода ветвей и границ, предложенного для комбинационных схем; алгоритмы символьного моделирования и алгоритмы, основанные на моделировании. Первые два подхода дают неприемлемые результаты для схем большой размерности. Поэтому в последнее время широко разрабатывается третий подход, к которому относятся и генетические алгоритмы (ГА) генерации тестов.

Одним из применений ГА в технической диагностике является построение на их основе генераторов тестов для ЦУ [1].

Современные ЦУ имеют сложную последовательностную структуру, а в качестве модели наибольшее распространение получили синхронные последовательностные ЦУ. Стратегия ГА при построении тестов синхронных последовательностных схем основывается на моделировании ЦУ с неисправностями.

Определим компоненты, которые необходимы для создания ГА. При генерации тестов синхронных последовательностных схем с помощью ГА в качестве особи рассматривают тестовую последовательность (рис. 1а). Популяция состоит из фиксированного числа тестовых последовательностей, возможно, различной длины (рис. 1б).



а) особь б) популяция

Рисунок 1 - Кодирование особей и популяций в ГА

Для выбранного таким образом кодирования особей и популяций будут применяться следующие генетические операции:

- Скрещивание, вероятность применения операции P_c. Операция реализуется в виде двух независимых операций (рис. 2а-б): вертикального и горизонтального скрещивания, выбор в процессе работы между которыми происходит с вероятностью P_{c1} и $P_{c2}=1-P_{c1}$. Все «пустоты», возникающие в популяции после применения данных операций, которые образуются из-за различной длины участвующих особей, заполняются случайным образом. Также следует отметить, что при такой реализации длина особей-потомков в случае применения первого оператора скрещивания равна максимальной из длин особей, участвующих в операции, а при использовании второго – длины особей могут как уменьшаться, так и увеличиваться.

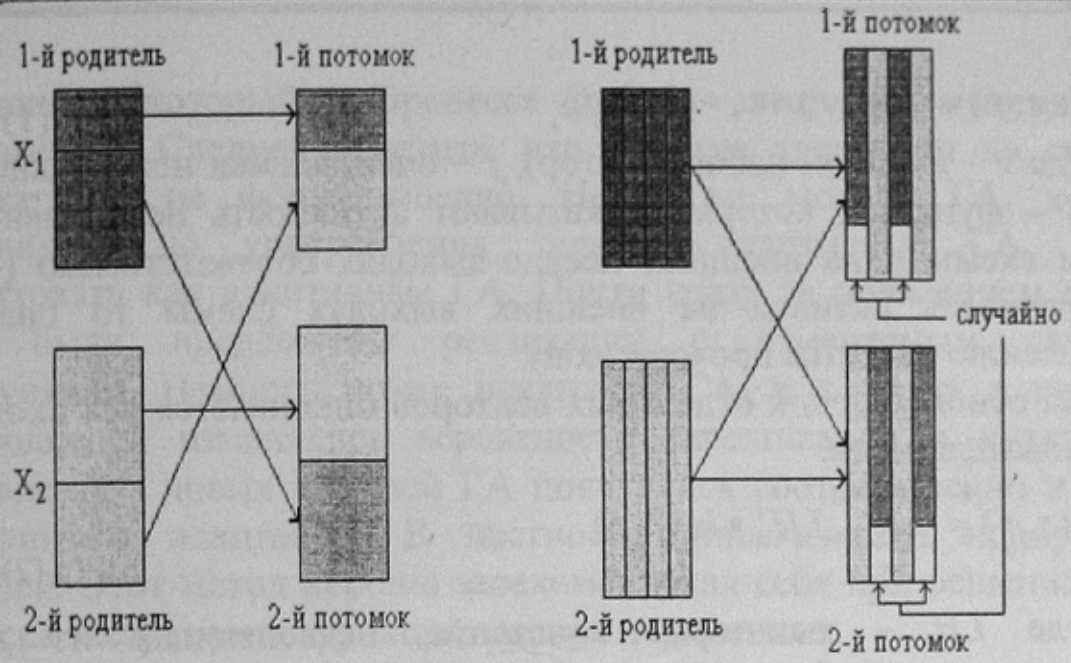


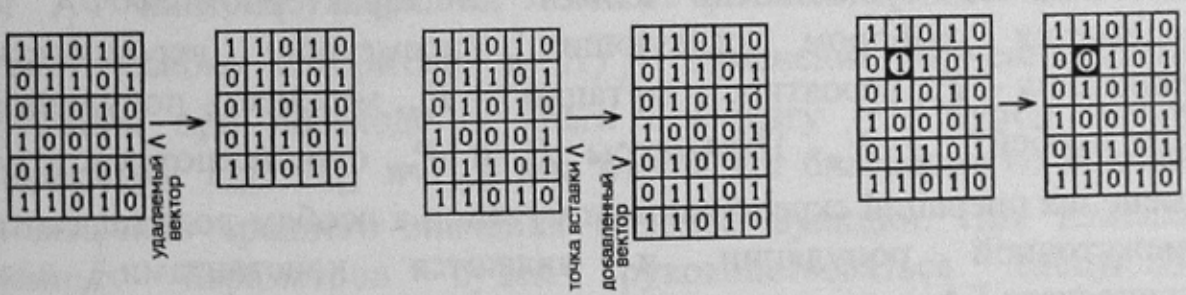
Рисунок 2 - Операции горизонтального и вертикального скрещивания ГА

- Мутация, вероятность применения операции P_m. Здесь также производится выбор одной из возможных операций (рис. 3а-в):

а) удаление одного входного вектора из случайно выбранной позиции (P_{m1});

б) добавление одного входного вектора в случайную позицию (P_{m2});

в) случайная замена битов в тестовой последовательности (P_{m3}).



а) удаление вектора б) добавление вектора в) инверсия бита

Рисунок 3 - Операции мутации ГА

Выбор между тремя операторами мутации также производится случайно с вероятностями P_{m1} и P_{m2} с распределением $0<P_{m1}<P_{m2}<1$.

Определим сначала оценочную функцию для отдельного входного набора:

$$h(v, f) = f_1(v, f) + c_1 * f_2(v, f), \quad (1)$$

где v – входной набор (вектор), f – оцениваемая неисправность, f_1 и f_2 – функции, которые показывают активность неисправности внутри схемы и на внешних псевдо-выходах соответственно (если неисправность активна на внешних выходах схемы то она по определению является проверяемой).

На основе оценок отдельных векторов оценивается вся входная последовательность.

$$H(s, f) = \sum_{i=1}^{i=длина} LH^i * h(v_i, f), \quad (2)$$

где LH – некоторая константа, позволяющая из двух последовательностей с построенными таким образом оценками выбрать ту, у которой длина меньше. Следовательно, чтобы определить фитнесс функцию некоторой особи-последовательности s для неисправности f необходимо произвести моделирование работы двух схем: исправной схемы на входной последовательности s и неисправной схемы на той же последовательности в присутствии f.

Работу классического ГА можно рассматривать как сбалансированную комбинацию расширения (освоения) новых областей в пространстве поиска и использования (эксплуатации) уже проверенных областей. Баланс между расширением и эксплуатацией таких областей существенно влияет на характеристики ГА и определяется выбором следующих параметров: вероятность скрещивания – P_c, вероятность мутации – P_m, мощность популяции (число особей) – N_p. Параметры P_c и P_m определяют частоту применения операций скрещивания и мутации к особям-родителям из промежуточной популяции и являются константами для классического ГА.

Без сомнения, управляющие параметры следует выбирать с учетом взаимодействия основных операторов генетического алгоритма. Оптимальные управляющие параметры существенно зависят от вида целевой функции.

Для преодоления ограничений классического ГА и придания ему большей гибкости целесообразно использовать адаптационный

механизм, который в процессе работы алгоритма изменяет его параметры. Следует отметить, что понятие адаптации до сих пор достаточно не формализовано. Некоторые модели ГА, которые появились до употребления термина адаптивные ГА, можно трактовать как адаптивные ГА. Почти сразу за появлением первых ГА были предложены реализации с изменяемым размером популяции. Немного позже появились ГА, в которых в процессе выполнения изменялись вероятности скрещивания и мутации. С появлением новых моделей ГА появлялись соответственно и новые механизмы адаптации. В частности динамическое кодирование особей. Этот метод хорошо зарекомендовал себя при решении задач поиска экстремумов функций многих переменных.

Общая схема внесения адаптации в ГА заключается во введении динамики в управляющие параметры ГА, при этом в схеме алгоритма произойдут некоторые изменения.

Внесение адаптационных механизмов в ГА позволяет функционировать им более эффективно на всех стадиях выполнения. В частности ГА со встроенным адаптационным механизмом намного реже попадают в ловушку локального оптимума. Адаптационный механизм в процессе выполнения алгоритма подстраивает его параметры для оптимального функционирования на каждом конкретном этапе выполнения.

Обозначим $\overline{f}_i$ среднее значение фитнесс функции на i-м шаге генетического алгоритма, а $\Delta\overline{f}_i$ – изменение значения фитнесс функции при переходе от шага i к шагу i+1. Будем изменять параметры P_{m1}, P_{m2} и P_{m3} так, чтобы это благоприятно влияло на повышение среднего значения фитнесс функции. При изменении данных параметров будем руководствоваться следующими правилами:

если $\Delta\overline{f}_i$=0, что соответствует случаю, когда генетические операторы не смогли построить ни одной особи со значением фитнесс функции, большим чем у родительских особей, то необходимо увеличивать вероятности P_{m2} и P_{m3}. Вероятность P_{m1} оставим без изменения, поскольку она необходима для сокращения длины получаемых тестовых последовательностей;

если $\Delta\overline{f}_i>0$, что соответствует случаю, когда генетические операторы при данных значениях P_{m1}, P_{m2} и P_{m3} смогли построить особи со значением фитнесс функции, которое превышает значение у родителей, то увеличивать разрушающие свойства (вероятности P_{m2} и P_{m3}) нет необходимости. В этом случае будем увеличивать вероятность P_{m1} удаления вектора из тестовой последовательности, стимулируя тем самим генерацию более коротких тестовых последовательностей.

Данный механизм адаптации был встроен в ГА, который используется для генерации проверяющих тестов в программном комплексе АСМИД-Е [3]. Предложенные выше правила, по которым будет выполняться адаптация, в алгоритме генерации тестов реализованы следующими операциями:

1. Восстановление предварительно заданных значений ($P_{m1}=0.33$; $P_{m1}=0.66$);

2. Увеличение вероятности удаления вектора из тестовой последовательности ($P_{m1}=P_{m1}+0.05$, $P_{m2}=P_{m2}-0.025$, $P_{m3}=P_{m3}-0.025$);

3. Увеличение вероятности добавления вектора к тестовой последовательности и вероятности инверсии вектора ($P_{m1}=P_{m1}-0.05$, $P_{m2}=P_{m2}+0.025$, $P_{m3}=P_{m3}+0.025$);

4. Увеличение вероятности изменения вектора в тестовой последовательности ($P_{m1}=P_{m1}-0.025$, $P_{m2}=P_{m2}-0.025$, $P_{m3}=P_{m3}+0.05$).

В результате при отсутствии прогресса в поиске решения (не увеличивается среднее значение фитнесс функции) максимальное изменение вероятностей мутации добавления вектора и инверсии бита тестовой последовательности достигается достаточно быстро (за 6 поколений). При этом ГА фактически функционирует как алгоритм псевдослучайного поиска.

Рассмотренный выше алгоритм построения тестов, основанный на адаптивной генетической стратегии, является одним из последних, реализованных для системы логического моделирования и диагностики АСМИД-Е. Эксперименты на схемах из каталога [4]

показали, что длина тестов, уменьшилась в среднем на 5% а время их генерации сократилось на 15–25%.

Список источников

1. Rudnick E.M., Holm J.G., Saab D.G., Patel J.H., Application of Simple Genetic Algorithm to Sequential Circuit Test Generation // Proc. European Design & Test Conf. - 1994. - P.40-45.

2. Иванов Д.Е., Скобцов Ю.А. Генерация тестов цифровых устройств с использованием генетических алгоритмов // Труды института прикладной математики и механики НАН Украины. – Т.4. – Донецк, ИПММ. – 1999. – С.82-88.

3. Иванов Д.Е, Скобцов Ю.А. Автоматизированная система моделирования и генерации тестов АСМИД-Е. Техническая диагностика и неразрушающих контроль.- 2000. г.

4. Brgles F., Bryan D., Kozminski K. Combinational profiles of sequential benchmark circuits // International symposium of circuits and systems, ISCAS-89.– 1989.– p.1929-1934.