УДК 519.712

Сенченко А.С., Щербак О.В.

[1] *канд. физ.-мат. наук, доцент кафедры алгебры, СГПУ*

[2] *студентка 5 курса физико-математического факультета, СГПУ*

e-mail: senchenko@pisem.net

ИСПОЛЬЗОВАНИЕ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ ДЛЯ ТЕСТИРОВАНИЯ БУЛЕВЫХ ФУНКЦИЙ

В работе предложен генетический алгоритм построения контрольных тестов булевых функций. Этот алгоритм выбирает те наборы значений переменных, на которых тестируемые функции принимают различные значения. Предложена программная реализация алгоритма в среде Delphi.

Ключевые слова: *тестирование, генетический алгоритм, конторольный тест*

Введение

Булевы функции эффективно используются при логическом проектировании цифровой и микропроцессорной техники, в теории кодирования и криптографии, а также в математическом моделировании. Их название происходит от фамилии выдающегося английского математика и логика Джорджа Буля, который впервые использовал их для применения символизма к логике. Первоначально булевы функции рассматривались как логические формулы и использовались прежде всего для решения комбинаторных логических задач. В 1938 году Клод Шеннон показал, что релейные схемы могут быть описаны и промоделированы с помощью булевых функций. В дальнейшем булевы функции стали широко использоваться в разработке различных вычислительных и управляющих систем: законы функционирования системы и вся система вообще, обычно описываются булевыми функциями. В связи с этим возникло направление в дискретной математике, занимающееся исследованием булевых функций.

С увеличением сложности вычислительных и управляющих систем, а также с повышением требований, предъявляющихся к надёжности их функционирования, необходимо уделять большое внимание контролю исправности и диагностике неисправностей этих систем. Если неисправности управляющих элементов систем являются достаточно устойчивыми, то для выявления и проверки таких неисправностей можно использовать проверочные контрольные тесты. Поскольку поведение отдельных элементов системы описывается булевыми функциями, то контрольные тесты чаще всего применяют именно к булевым функциям: их сравнивают с эталоном поведения.

Доказано, что построение таких тестов является NP-сложной задачей [2, 3, 5, 6], поэтому все известные алгоритмы являются эвристическими и дают приближенный к оптимальному результат. Было разработано много методов для более-менее эффективного построения контрольных тестов, которые дают высококачественный результат за небольшое время. Большой вклад в развитие тестирования цифровых схем внесли отечественные ученые С.В. Яблонский, О.Б. Лупанов, С.А. Ложкин, Н.П. Редькин, П.П. Пархоменко, Ю.А. Скобцов и зарубежные исследователи Рот, Зориан, Агравал, Фудживара и многие другие. Исследования в данном направлении продолжаются, поскольку использующиеся методы не всегда позволяют строить тесты необходимой полноты из-за неудовлетворительных показателей быстродействия. Наиболее перспективными являются алгоритмы построения тестов с использованием методов теории искусственного интеллекта, в первую очередь, генетических алгоритмов. Поэтому для эффективного построения контрольных тестов булевых функций мы использовали именно генетические алгоритмы.

Постановка задачи

Задачей работы является разработка генетического алгоритма построения контрольных тестов заданных булевых функций и его программная реализация в среде Delphi. Исходные функции могут быть заданы как таблицей истинности, так и аналитически в виде ДНФ. Необходимо выделить такие наборы значений переменных, на которых все функции принимают различные значения. Количество таких наборов должно быть минимально возможное или близкое к нему. Задаются ограничения на количество переменных (до 8) и количество тестируемых булевых функций (до 10).

Основная часть

Булевыми называют функции, аргументы и значения которых принадлежат множеству $\{0,1\}$. Основными булевыми операциями являются отрицание (обозначается чертой сверху), конъюнкция (обозначается $\wedge$, $\cdot$ или знак пропускается как и для умножения), дизъюнкция (обозначается $\vee$), импликация ($\Rightarrow$), эквиваленция ($\Leftrightarrow$) и сумма по модулю два ($\oplus$) [1]. Каждая, отличная от нуля, булева функция может быть записана в виде дизъюнктивной нормальной формы (ДНФ), эта запись является наиболее удобной и часто используется. Булева функция от N переменных имеет 2^N возможных комбинаций значений переменных, такую функцию можно полностью описать таблицей с 2^N строками. В каждой строке будет указано значение функции для различных комбинаций значений переменных. Такая таблица называется таблицей истинности. Строки таблицы истинности будем всегда располагать в порядке двоичного возрастания значений переменных. Поэтому строки могут быть пронумерованы.

Для обеспечения надежного функционирования системы необходимо решить задачу контроля исправности её элементов. Для решения этой задачи С.В. Яблонским предложены [6] логические методы контроля, суть которых заключается в том, что на входы системы подаются некоторые специальным образом подобранные «проверочные» наборы значений переменных и на основе выходных значений системы делается вывод о её исправности и характере неисправностей (при их наличии). Как правило, возможные виды неисправности известны заранее. Таким образом, мы можем выделить булеву функцию, соответствующую исправной системе и функции, соответствующие неисправностям, то есть можно сформировать некоторое конечное множество исследуемых булевых функций M мощности k. Для определения, является ли система исправной и выявления типа неисправности необходимо различить между собой все функции множества M. Таким образом, тестирование булевых функций в нашем случае состоит в том, что для них мы подставляем некоторый набор значений переменных и исследуем значение функций.

Функция $f(x) = 2^x$ при возрастании x растет очень быстро. Поэтому, для функций от среднего количества (40-50) переменных, используемых в реальных вычислительных и управляющих системах, протестировать значения на всех наборах вычислительно сложно. Вследствие этого необходимо выбрать лишь некоторые наборы значений переменных, на которых исследуется поведение функции. Для того, чтобы различить между собой k функций необходимо не менее, чем $\lceil \log_2 n \rceil$ наборов значений (функция $\lceil x \rceil$ означает дополнение до ближайшего целого). Наша задача состоит в том, чтобы выбрать такие наборы значений, которые различают между собой заданные булевы функции. При этом в первую очередь интересуют тесты минимальной длины, количество наборов в которых равно $\lceil \log_2 n \rceil$ и тесты, у которых длина близка к минимальной.

Для решения этой задачи полный перебор всех вариантов таких наборов неэффективен ввиду огромного количества операций. Одним из подходов к решению подобных задач является эволюционное моделирование, впервые сформулированное в 1966 году Л. Фогелем, А. Оуэни и М. Уолшем в работе «Искусственный интеллект и эволюционное моделированне». Этот подход использует принцип природной эволюции процесса. Суть этого подхода заключается в том, что из некоторого начального множества возможных решений (начальной популяции) выбираются наилучшие, из которых специальным образом получаем дополнительные решения. Затем отобранные решения объединяются с новыми в первое поколение решений. Из первого поколения аналогичным образом получаем второе и последующие поколения. Считается, что через определенное достаточно большое количество поколений мы получим наилучший результат.

Этот подход послужил основой появления генетических алгоритмов. Рассмотрим основные понятия, связанные с генетическими алгоритмами. Хромосомой будем считать отдельный вариант решения задачи – некоторую последовательность символов, каждый символ называется геном. Генетический алгоритм случайно или особым образом генерирует начальную популяцию хромосом. Считаем, что все популяции состоят из w хромосом. Для каждой хромосомы с помощью «фитнесс-функции» подсчитывается насколько она подходит для решения задачи. Из начальной популяции выбираем $\frac{w}{2}$ по наилучшим значениям их фитнесс-функции. Эти хромосомы переходят в следующее поколение. Из выбранных хромосом получаем еще $\frac{w}{2}$ хромосом с помощью одной из трех операций: мутации, сдвига или кроссинговера. Мутация состоит в том, что некоторый ген меняет свое значение. Сдвиг состоит в циклическом изменении позиции всех генов на определенное количество единиц. Кроссинговер (в литературе о генетических алгоритмах также используется название кроссовер или скрещивание) – это операция, при которой из двух хромосом порождаются две новые хромосомы. Одноточечный кроссинговер работает так: сначала случайным образом выбирается одна из точек разрыва – участок между соседними генами в строке. Обе родительские хромосомы разрываются на два сегмента по этой точке. Затем первый сегмент первой хромосомы склеивается со вторым сегментом второй хромосомы, а первый сегмент второй хромосомы склеивается со вторым сегментом первой хромосомы. В результате получаются две новые хромосомы. Аналогично работает двухточечный кроссинговер, у которого две точки разрыва и т.д. Например хромосомы 011111011 и 110000111 в результате одноточечного кроссинговера по третьей точке разрыва дадут хромосомы 011000111 и 110111011.

Работа генетического алгоритма представляет собой итерационный процесс, продолжающийся до тех пор, пока не пройдет заданное число поколений или некоторый другой критерий остановки.

Определение операций генетического алгоритма для поставленной задачи

Считаем что в задаче рассматриваются k тестируемых булевых функций от n переменных. Все хромосомы являются последовательностями длиной 2^n, состоящие из нулей и единиц. Каждый ген отвечает за набор значений таблицы истинности с соответствующим номером по правилу: 0 – набор не включается в тест; 1 – включается в тест. Таким образом количеству единиц в хромосоме соответствует количество наборов в тесте. Во всех хромосомах начального поколения количество единиц равно $\lceil \log_2 k \rceil$.

Фитнесс-функция задается как количество различных значений тестируемых функций, то есть хромосома, получившая оценку k, является решением задачи и приводит к останову процесса поиска решения. Для получение хромосом следующих поколений применяются операции сдвига, мутации и одноточечный кроссинговер. При мутации некоторый ген меняет значение 0 на значение 1, а некоторый другой ген - значение 1 на значение 0. При кроссинговере точка разрыва выбирается так, чтобы количество единиц в обеих новых хромосомах совпадало с количеством единиц в родительских хромосомах. Таким образом генетические операции не увеличивают количество единиц в хромосомах.

Если на протяжении 100 поколений ответ не получен, то формируется новое начальное поколение хромосом с количеством единиц на одну больше.

Разработана программная реализация алгоритма в среде программирования Delphi. По результатам тестирования программы выбраны оптимальные параметры вероятностей генетических операций: мутация 0,1 - 0,15; сдвиг 0,1 - 0,15; кроссинговер 0,7 - 0,8. С этими вероятностями количество наборов переменных, участвующих в тесте булевых функций никогда не превышало величину $\lceil \log_2 k \rceil + 1$, то есть никогда не добавлялось более одного набора.

Выводы

В работе предложен генетический алгоритм для тестирования булевых функций. Рассмотрен вариант кодирования исходных данных, оценка фитнесс-функции и способы выполнения генетических операций. Разработана программная реализация алгоритма, выбраны оптимальные на наш взгляд вероятности генетических операций.

Дальнейшие исследования должны рассматривать особенности выполнения генетических операций и их вероятность для отдельных случаев вида тестируемых булевых функций, в частности для случая, когда исходные функции отличаются друг от друга очень слабо.

Литература

[1] *Бондаренко М.Ф.* Комп'ютерна дискретна математика : Підручник / М.Ф. Бондаренко, Н.В. Білоус, А.Г. Руткас. – Х.: «Компанія СМІТ», 2004. – 480 с.

[2] *Коваценко С.В.* Синтез легкотестируемых схем в базисе Жегалкина для инверсных неисправностей / С.В. Коваценко // Вестник Московского университета. Серия 15. Вычислительная математика и кибернетика. – 2000. – №2. – С. 45 – 47.

[3] *Носков В.Н.* Метод синтеза удобных для контроля комбинационных схем / В.Н. Носков // Дискретная математика. – 1993. – Т.5, вып. 4. – С. 3 – 25.

[4] *Рутковская Д.* Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская, М. Пилинский, Л. Рутковский. – М.: Горячая линия – Телеком, 2006. – 452 с.

[5] *Хахулин В.Г.* О проверяющих тестах для счетчика честности / В.Г. Хахулин // Дискретная математика. – 1995. – Т.7, вып. 4. – С. 51 – 59.

[6] *Яблонский С.В.* Некоторые вопросы надежности и контроля управляющих систем / С.В. Яблонский // Математические вопросы кибернетики. – 1988. – №1. – С. 5 – 25.