

## ПРОЕКТИРОВАНИЕ И ДИАГНОСТИКА ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

УДК 519.17

В.В. Андреева, К.А. Сорудейкин

### ПРОСТРАНСТВЕННО-ОРИЕНТИРОВАННАЯ ОПТИМИЗАЦИЯ ТЕСТОВЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ С ПРИМЕНЕНИЕМ ИТЕРАТИВНОГО ПОДХОДА ПО МНОГИМ ПЕРЕМЕННЫМ

*Исследование выполнено при финансовой поддержке РНФ в рамках научного проекта № 14-19-00218.*

Предлагается метод сокращения тестовых последовательностей, представленных в виде троичных векторов с помощью дерева декомпозиции, основанного на разложении множеств троичных векторов по нескольким переменным.

**Ключевые слова:** дерево декомпозиции; максимально-совместимые подмножества; троичные векторы.

Рассматривается тестовая последовательность  $T = \{t_1, \dots, t_s\}$ , представленная в виде троичных векторов. Тестовый набор  $b_i$ , полученный из троичного вектора  $t_i$  путем доопределения неопределенных компонент, обнаруживает неисправность  $f_i$  из множества  $F$ . Необходимо найти минимальную либо достаточно близкую к минимальной тестовую последовательность без потери покрытия множества рассматриваемых неисправностей.

В данной работе предлагается модификация алгоритма, описанного нами ранее в работе [3] и до этого развиваемого в работах [1, 2]. Анализируется тестовая последовательность, в которой отсутствуют поглощающие векторы. Сокращение длины тестовой последовательности выполняется за счет операции пересечения троичных векторов. Основная идея сокращения тестовой последовательности заключается в выделении максимально совместимых подмножеств  $T'_i$  из  $T$ , где каждое  $T'_i$  порождает булев вектор (тестовый набор), полученный путем пересечения (совмещения) всех элементов этого подмножества с последующим доопределением неопределенных компонент. Построение максимально совместимых подмножеств предлагается выполнять с помощью дерева разложения по немонотонным переменным, рассматриваемого в [1–3]. В работе [3] предлагается модификация дерева разложения, которая позволяет, во-первых, избавиться от дублирования троичных векторов в дочерних вершинах, а во-вторых, находить решение без использования классической задачи покрытия. Отсутствие дублирования сокращает глубину дерева разложения, а исключение задачи покрытия сокращает вычислительные затраты при поиске решения. Если правильно распределить ограничения и поставить наиболее значительные из них в самом начале, то можно сократить вычислительные затраты. Этому принципа мы придерживаемся в наших работах и называем его *пространственно-ориентированной оптимизацией* [4].

В данной работе развивается подход [5,7], основанный на разложении множества векторов по многим переменным. Такое разложение выполняется в том случае, если в множестве векторов выделяется некоторое *множество максимально определенных немонотонных переменных*, а именно немонотонных переменных с одинаковым минимальным числом неопределенных символов. Предлагается итеративный подход к построению дерева декомпозиции, который позволяет избавиться от избыточных шагов на стадии формирования результата и освободиться в ряде случаев от необходимости использования перебора.

## 1. Типы данных и основные определения

Заданное множество  $T$  троичных векторов представляется в древовидной структуре путем распределения векторов по вершинам на основе некоторого набора признаков. В качестве таких признаков выбираются значения отдельных переменных векторов.

Троичный вектор представляет собой вектор, компоненты (переменные) которого могут принимать значения из множества  $\{0, 1, x\}$ . Напомним, что символ  $x$  в троичном векторе означает неопределенную компоненту, которая может принимать значение либо 0, либо 1. Значения  $\{0, 1\}$  переменной троичного вектора будем называть *определенными* значениями, значение  $x$  – *неопределенным* значением.

Векторы, у которых одноименные переменные принимают значения 0,1 (1,0), будем называть *ортгогональными*.

Переменная для некоторого множества векторов может быть *монотонной*. Это значит, что в векторах этого множества она встречается либо со значением 0(1), либо со значением  $x$ . Переменная для этого множества векторов *не монотонна*, если она встречается как со значением 0, так и со значением 1.

Основная операция, используемая для сокращения длины тестовой последовательности, – это операция *пересечения*. Троичные векторы, пересечение которых не пусто, будем называть *совместимыми*. Например, рассмотрим следующие троичные векторы:  $t_1 = 1xx10$ ,  $t_2 = x10xx$ , их результат пересечения определяется вектором  $t_1 \cap t_2 = 11010$ . Совместимые векторы монотонны по всем переменным.

Из заданного множества  $T$  необходимо сформировать попарно не пересекающиеся *максимально-совместимые* подмножества.

## 2. Использование дерева для извлечения решения

В данной работе в качестве анализируемой структуры используется дерево. Мы обобщаем его структуру, представленную в [1, 6]. Из вершины дерева может исходить одна, две или три дуги. Каждой дуге сопоставляются значения соответствующих переменных, используемых для разложения множества векторов для рассматриваемой вершины. Множество разбивается на более мелкие подмножества, соответствующие дочерним вершинам. В каждом подмножестве дочерней вершины значения переменных разложения совпадают с комбинацией значений переменных дуги, заходящей в эту вершину. Будем говорить, что векторы дочерних вершин совместимы по переменным разложения родительской вершины. Отметим, что каждой вершине дерева сопоставляется множество троичных векторов, совместимое по значениям переменных разложения, сопоставляемых вершинам пути, ведущим из корня дерева в рассматриваемую вершину. Вершина объявляется концевой, если сопоставляемое ей подмножество монотонно по всем переменным.

Итак, исходное множество  $T$  троичных векторов разделяется на подмножества, сопоставляемые вершинам дерева, на основе свойства совместимости по значениям на подмножестве переменных, определяемых путем, ведущим в рассматриваемую вершину.

Возможны различные правила выбора переменных (различные эвристики) для разложения в очередной вершине дерева. Например, для сокращения глубины дерева при разложении выбирается переменная с минимальным числом неопределенных значений в множестве векторов, сопоставляемых вершине. В этом случае из вершины дерева проводится три дуги – по одной на каждое значение переменной из множества  $\{0, 1, x\}$ . Дочерним вершинам сопоставляются множества троичных векторов, принимающие соответствующие (одинаковые) значения по рассматриваемой переменной. Каждое полученное множество разлагается, если возможно, аналогичным образом. В результате такого разложения строится дерево, в котором концевым вершинам будут соответствовать *совместимые подмножества*, а ветвям со значением  $x$  – множества, элементы которых добавляются в совместимые подмножества с целью построения максимально-совместимых подмножеств. Правила добавления рассматриваются в [1, 2].

Для удобства дальнейшего изложения предлагаемого подхода введем понятие *поддерева решения*. В него включаются пути дерева, помеченные определенными значениями  $\{0, 1\}$  переменных и ведущие в терминальные вершины. В дереве присутствуют дуги, помеченные значением  $x$ . Будем считать, что эти дуги вместе с инцидентными им вершинами не входят в дерево решения. Множество троичных

векторов, сопоставляемое этим вершинам, будем называть **LDC** (local don't care set)-множеством – множеством локальных неопределенных компонент. Множества **LDC<sub>i</sub>**, сопоставляемые некоторым вершинам дерева, используются для добавления их векторов в множества, сопоставляемые концевым вершинам *поддерева решения* с целью получения максимально совместимых подмножеств.

### 3. Метод построения максимально совместимых подмножеств

Максимально совместимые подмножества предлагается строить с помощью дерева декомпозиции. В разделе 3 описан общий механизм его построения на основе исходного множества троичных векторов. Опишем здесь детали общей процедуры.

Одна процедура построения дерева (раскрытие вершины дерева) в нашей работе называется *итерацией*. В процессе решения задачи может быть выполнено несколько итераций для достижения требуемого результата. По выполнении итерации производится проверка на получение нужного решения, и в случае необходимости осуществляется следующая итерация. Существует набор условий, на основании которого мы выполняем эту проверку. Можно выделить начальную, результирующую и промежуточные итерации, располагаемые между ними. В свою очередь, каждая итерация включает в себя ряд действий, в основе которых лежит построение дерева и получение в процессе построения разбиения векторов начального множества. В данной работе полученные на каждой итерации результаты представляем в виде нового начального множества и обрабатываем его той же самой процедурой разложения дерева, повторяя этот процесс, пока не выполняются условия, описанные в работе [6]:

- Если очередной вершине сопоставляется монотонное подмножество, то вершина объявляется концевой. В соответствии с работой [6] считается, что терминальные вершины не относятся к дугам **LDC<sub>i</sub>**, которые являются *вспомогательными*.

- Наличие множества **LDC<sub>i</sub>** означает необходимость дальнейшей обработки этого множества либо новой итерацией, либо процедурой обратного обхода. Следует иметь ввиду, что все векторы **LDC<sub>i</sub>** необходимо в конце концов распределить по терминальным вершинам.

- Модификация начального множества выполняется в случае формирования совместимых подмножеств мощности больше 1, соответствующих терминальным вершинам дерева разложения. В результате пересечения совместимого подмножества продуцируется новый элемент, который будет не равен какому-либо уже существующему элементу в исходном множестве заданной итерации, но возможно поглощение этим элементом других; в этом случае все поглощенные элементы исключаются и формируется новая итерация.

#### 4.1. Выбор переменных для дерева разложения

В соответствии с правилами построения дерева разложения в каждой вершине выбирается переменная разложения с минимальным числом неопределенных значений на множестве троичных векторов, сопоставляемом вершине. Возможно, что таких переменных несколько. В более ранних работах [1, 2] эта ситуация игнорировалась и разложение проводилось по любой из них. В работе [7] была высказана идея разложения по многим переменным. В данной работе эта идея развивается.

Напомним, что терминальным вершинам дерева декомпозиции соответствуют совместимые подмножества, относительно которых в дальнейшем формируются максимально совместимые подмножества путем добавления в них элементов из **LDC<sub>i</sub>**.

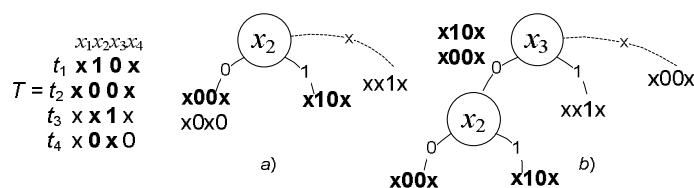


Рис. 1. Пример с двумя альтернативными деревьями декомпозиции

Анализируя множество максимально определенных столбцов, можно выделить следующие ситуации:

1. Из заданного множества троичных векторов можно выделить такое подмножество **D** троичных векторов, у которых переменные с минимальным числом неопределенных значений для множества в целом принимают *только определенные значения*. В этом случае, если выполнять разложение по *любым* из соответствующих переменных, эти троичные векторы будут приписаны терминальным вершинам во всех альтернативных деревьях, полученных по этим переменным. Например, переменным  $x_2, x_3$  (рис. 1) соответствуют максимально определенные столбцы, а троичные векторы  $t_1, t_2$  по этим переменным принимают только определенные значения. Здесь при поиске решения возможны альтернативы, так как присутствуют два максимально определенных столбца. Итак, построив для каждой из альтернатив свое дерево разложения, векторы  $t_1, t_2$  будут сопоставляться терминальным вершинам в каждом дереве, что следует из процедуры разложения.

2. Из заданного множества троичных векторов невозможно выделить подмножество **D** троичных векторов, у которых переменные с минимальным числом неопределенных значений для множества в целом принимают только определенные значения. Если выполнять разложение по *любым* из соответствующих переменных, каждый раз будет построено очередное альтернативное решение. В этом случае воспользуемся максиминной стратегией выбора переменной разложения, а именно выбираем ту переменную, которой соответствует максимально определенный столбец с минимальным количеством определенных значений **0(1)**. Такой выбор переменной обусловлен тем, что одной из ветвей будет соответствовать минимальное по мощности множество, а раз векторов с таким определенным значением по этой переменной мало, то формирование решения можно начать с них.

3. Из заданного множества троичных векторов можно выделить такое подмножество **D\*** троичных векторов, у которых переменные с минимальным числом неопределенных значений для множества в целом принимают *только неопределенные значения*. В этом случае, если выполнять разложение по *любым* из соответствующих переменных, эти троичные векторы будут входить в множество  $LDC_i$ . Следовательно, во всех альтернативных деревьях эти векторы либо расширяют совместимые множества до максимально совместимых, либо формируют новое решение.

Третий вариант не применяется в решаемой задаче, поэтому в данной работе предлагается учитывать первые две ситуации при построении дерева разложения и построении решения.

#### 4.2. Разложение по многим переменным

Разложение по нескольким переменным выполняется в том случае, если выполняются следующие условия:

1. На этапе выбора немонотонной переменной выделяется некоторое множество из двух и более переменных с минимальным числом неопределенных значений для множества в целом.
2. Можно выделить троичные векторы, принимающие определенные значения по этим переменным.

Т а б л и ц а 1

	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
$t_1$	x	1	1	x	x	x
$t_2$	x	1	x	1	x	x
$t_3$	x	1	x	x	1	1
$t_4$	0	x	x	1	x	0
$t_5$	1	0	x	1	x	x
$t_6$	1	x	1	x	x	x
$t_7$	x	0	0	0	0	1
$t_8$	0	x	x	1	0	1
$t_9$	0	x	0	0	x	1
$t_{10}$	x	0	0	0	1	x
$t_{11}$	1	0	x	x	0	x
$t_{12}$	x	1	0	0	x	0

При выполнении этих условий, дерево разложения по многим переменным строится следующим образом: из общего множества троичных векторов выделяется подмножество **T\*** векторов, удовлетворяющих условиям 1, 2. Из оставшихся векторов формируется множество **LDC** по переменным  $\{x_i\}$ , не удовлетворяющим условиям 1, 2. Такое множество в дальнейшем будем обозначать  $LDC^*_i$ .

Множество **T\*** разлагается по выделенным переменным стандартным способом, в результате чего строится дерево разложения. Если множество  $LDC^*_i$  не пусто, то оно приписывается корневому узлу дерева. Графически это можно представить так, что из корня дерева проводится дуга, помеченная значением  $\{x_i\}$ , которой сопоставляется множество  $LDC^*_i$ .

Если терминальным вершинам  $0(1)$  деревьев соответствует хотя бы одно совместимое подмножество мощности больше 1, то формируется *шаг итерации*, в противном случае осуществляется поиск максимально совместимых подмножеств.

Рассмотрим построение разложения по многим переменным на примере табл. 1, где список троичных векторов представлен в виде матрицы. Ее столбцам сопоставляются переменные, а строками являются троичные векторы.

В рассматриваемом примере столбцы, соответствующие переменным  $x_2$  и  $x_4$ , являются максимально определенными (с минимальным числом неопределенных значений), а троичные векторы  $t_2, t_5, t_7, t_{10}, t_{12}$  принимают определенные значения по этим переменным. Сформируем множество  $T^*$ , включив в него выделенные троичные векторы, а все остальные векторы включим в множество  $LDC^*_1$ . Строим дерево разложения по переменным  $x_2, x_4$ , сначала выполняется разложение по  $x_2$ , затем по  $x_4$ . Заметим, что эти переменные являются в множестве  $T^*$  не только максимально, но и полностью определенными, что следует из построения. Этот факт говорит о том, что порядок разложения по переменным  $x_2, x_4$  не влияет на результат разложения. Для рассматриваемого примера и непустого множества  $LDC^*_1$  формируется дуга, выходящая из корня дерева, помеченная  $\{x_2, x_4\}$  (рис. 2).

### 4.3. Процесс построения решения

Если в результате разложения по многим переменным  $LDC$  множество является пустым, тогда троичные векторы, находящиеся в терминальных вершинах, *включаются в решение*, в противном случае выполняется построение *максимально совместимых подмножеств*.

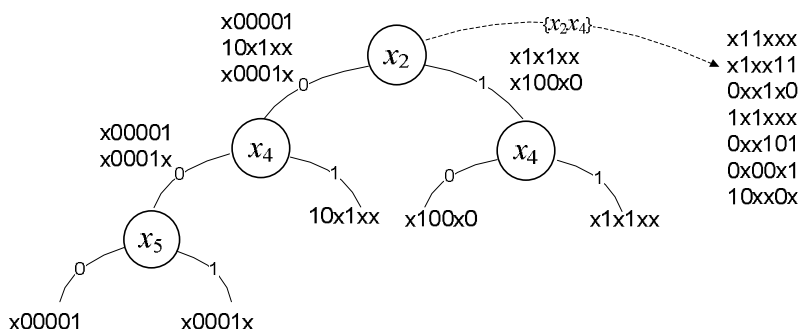


Рис. 2. Пример дерева разложения по многим переменным

Отметим, что терминальным вершинам при разложении по многим переменным сопоставляются отдельные троичные векторы  $q_i$ . К каждому вектору могут быть добавлены векторы из  $LDC^*$ . Полученные множества обозначим  $Q_i$ ,  $q_i \in Q_i$ . Так как множество  $LDC^*_i$  не пусто, для построения максимально совместимых подмножеств необходимо добавить подходящие векторы из  $LDC^*_i$  в одноэлементные множества терминальных вершин дерева. Строим каждое множество  $Q_i$  путем добавления троичных векторов из  $LDC^*_i$ , *совместимых* с  $q_i$ . Заметим, что при таком построении множеств  $Q_i$  возможно появление одинаковых троичных векторов в разных  $Q_i$ , поскольку элементы из  $LDC^*_i$  могут совмещаться с различными терминальными элементами  $q_i$ . Далее необходимо выполнить следующие шаги.

1. Если среди множеств  $Q_i$  выделяются совместимые, то они являются *максимально совместимыми*, так как ни один новый элемент не может быть добавлен в эти множества. В данном случае для каждого совместимого множества формируется свой вектор путем совмещения всех элементов с последующим добавлением его в решение. Так как на этапе построения множеств  $Q_i$  допускались одинаковые векторы в различных множествах, то после добавления полученных векторов в решение у оставшихся подмножеств исключаются те элементы, которые уже участвовали в формировании решения. Таким образом, среди оставшихся множеств  $Q_i$  возможно появление новых совместимых подмножеств. В этом случае действия 1 повторяются, в противном случае переходим к действиям, определенным в разделе 2.

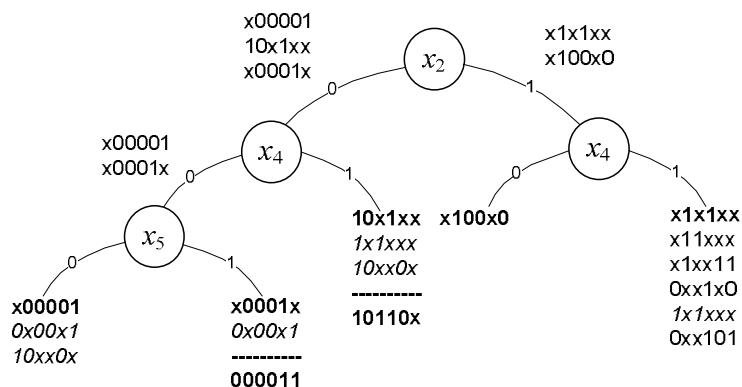


Рис. 3. Формирование множеств для элементов  $q_i$

2. Если множества  $Q_i$  не совместимы, то для каждого множества продолжается разложение. Результат разложения представляется деревом определенного вида: элемент  $q_i$  всегда будет соответствовать дуге, помеченной значениями  $x$ , все остальные векторы, объединённые в совместимые подмножества, – терминальным вершинам. Такое представление связано с тем, что при построении множество  $Q_i$  состоит из элемента  $q_i$  и элементов, совместимых с ним. Следовательно, в общем случае для дерева такого вида формирование максимально совместимого подмножества элемента  $q_i$  выполняется путем добавления этого элемента к любому совместимому подмножеству этого дерева. Предлагается на этом этапе в решении выбирать максимально совместимые подмножества максимальной мощности, в которые входят элементы  $q_i$ , с последующим исключением из оставшихся множеств повторяющихся векторов, уже включенных в решения. В результате этой процедуры все максимально совместимые подмножества, включающие элемент  $q_i$ , будут добавлены в решение, а оставшееся дерево может удовлетворять следующим условиям:

- если множество  $LDC^*_i$  не пусто и в дереве присутствуют максимально совместимые подмножества, то в этом случае формируется *новая итерация*;
- если множество  $LDC^*_i$  пусто и в дереве присутствуют максимально совместимые подмножества, то они включаются в решение.

Для примера рассмотрим построение максимально совместимых подмножеств (рис. 2). Для каждой терминальной вершины, которой сопоставлен троичный вектор  $q_i$ , формируем множество  $Q_i$  путем добавления совместимых с  $q_i$  векторов из множества  $LDC^*_i$ . На рис. 3 показаны множества  $Q_i$  для элементов  $q_i$ , курсивом отмечены те элементы, которые включены в разные подмножества.

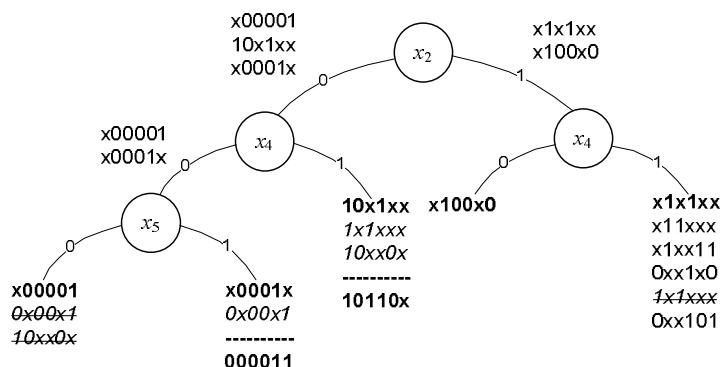


Рис. 3а. Выполнение операции совмещения

Если после добавления вновь получившиеся множества являются совместимыми, то применяем операцию *пересечения* к элементам *совместимых* подмножеств, а получившиеся векторы добавляем в решение. Так как при формировании подмножеств возможны повторения одинаковых векторов в тер-

минальных вершинах, то после формирования решения нужно исключить из оставшихся подмножеств векторы, которые уже участвовали в формировании решений (рис. 3а).

В рассматриваемом примере текущее решение будет иметь вид  $T = \{10110x, 000011, x100x0\}$ , а дерево после упрощения будет иметь вид, представленный на рис. 4, а.

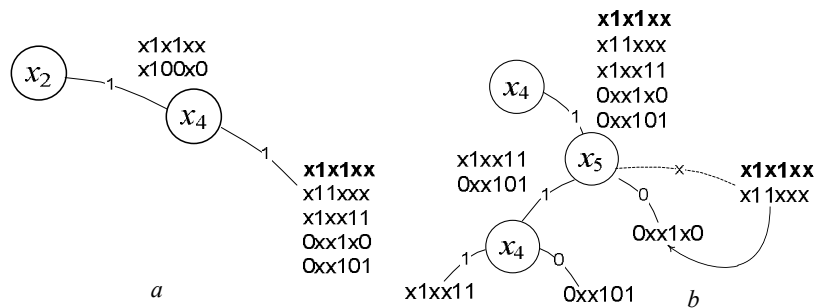


Рис. 4. Сокращенное дерево (а), дальнейшая процедура (b)

В примере оставшееся множество не является совместимым, следовательно, разложение этого множества продолжается традиционным способом (рис. 4, b), так как здесь всего один максимально определенный столбец.

Формируем максимально совместимое подмножество и полученные троичные векторы добавляем в окончательное решение. Если после выполнения такой процедуры в множестве остаются несовместимые подмножества, то они объединяются и разлагаются, а полученные векторы включаются в решение.

Получаем следующее результирующее множество:  $T = \{10110x, 000011, x100x0, x1xx11, 0xx101, 01110\}$ .

## 5. Экспериментальные результаты

Предложенный алгоритм программно реализован и испытан. В качестве оценки рассматриваемого подхода к сокращению длины тестовой последовательности предлагается сравнить полученные результаты с результатами сокращения длины алгоритмом [2]. Основные отличия предложенного подхода заключаются в том, что в алгоритме [2] выполняется построение максимально совместимых подмножеств на основе дерева декомпозиции, строящегося без формирования множеств  $LDC_i$ . В этом случае все элементы, принимающие значение  $x$  по переменной разложения, сопоставляются как левой, так и правой ветви. Кроме того, поиск окончательного решения сводится к задаче покрытия.

Таблица 2

№ п/п	%DC	$L$	$L^*$	$T^*$	$L^{**}$	$t^{**}$
1	60	32	<b>23</b>	0,2020	<b>22</b>	0,0051
2	65	32	<b>19</b>	0,1853	<b>17</b>	0,0046
3	75	32	12	0,4656	12	0,0070
4	66	32	<b>19</b>	0,2703	<b>17</b>	0,0050
5	56	32	26	0,2033	25	0,0073
6	76	32	9	1,0400	10	0,0060
7	86	32	7	1,9730	6	0,0050
8	90	32	5	1,7371	5	0,0090
9	71	32	13	0,5360	15	0,0080
10	67	32	17	1,2851	18	0,0070

В качестве тестовых примеров рассматривались короткие тестовые последовательности, анализ которых не требует существенных вычислительных затрат. Количество входных переменных и начальная длина теста принимают значение 32. Примеры генерируются с помощью программы PlaGenerator 2.1 [8]. Экспериментальные результаты представлены в табл. 2, в которой введены следующие обозначения:

- $\%DC$  – данный параметр определяет количество символов  $X$  в процентах, генерируемых программой PlaGenerator 2.1;
- $L$  – начальная длина теста;
- $L^*$  – длина теста, полученная методом из работы [2];
- $t^*$  – время работы алгоритма из работы [2] в секундах;
- $L^{**}$  – длина теста, полученная предложенным методом;
- $t^{**}$  – время работы предложенного алгоритма в секундах.

Анализируя полученные результаты, видим, что разработанный подход позволят получить неплохие по качеству решения, а в некоторых случаях наблюдаются улучшения результатов при существенном сокращении временных затрат.

## Заключение

В данной работе предложен подход к сокращению длины тестовой последовательности, основанный на построении дерева декомпозиции. В очередной вершине дерева используется, если возможно, разложение подмножеств троичных векторов по многим переменным. Предлагаемый подход позволяет на порядки сокращать временные затраты при сохранении в целом качества сжатия исходных тестовых последовательностей.

## ЛИТЕРАТУРА

1. Andreeva V. Test minimization technique for multiple stuck-at faults of combinational circuits // Proc. 8th East-West Design&Test International Symposium, Saint Petersburg, Russia, 2010. P. 168–170.
2. Andreeva V. Test Set Compaction Procedure for Combinational Circuits Based On Decomposition Tree. // Proc. 9th East-West Design &Test Symposium, Sevastopol, Ukraine, 2011. P. 251–254.
3. Andreeva V., Sorudeykin K. A research of heuristic optimization approaches to the test set compaction procedure based on decomposition tree // Proceeding of IEEE EWDTS. 2012. P. 382–387.
4. Sorudeykin K. Irrespective Priority-Based Regular Properties of High-Intensity Virtual Environments // 20<sup>th</sup> IEEE Telecomm. Forum TELFOR 2012, Belgrade, Serbia, Nov 2012. P. 510–513.
5. Андреева В.В., Сорудейкин К.А. Пространственно-ориентированная оптимизация тестовых последовательностей с применением итеративного подхода // Материалы 10-й российской конференции с международным участием «Новые информационные технологии в исследовании сложных структур» ICAM 2014. Томск : Изд-во ТГУ, 2014. С. 46–47.
6. Sorudeykin K., Andreeva V. Decomposition Tree - based Compaction Procedure with Iteration Steps for Interconversional Layouts of Tasks // Proc. 12th IEEE East-West Design & Test Symposium, 2014. Kiev, Ukraine, 2014. P. 173–178.
7. Андреева В.В., Сорудейкин К.А. Сокращение длины тестовой последовательности на основе дерева декомпозиции // Известия высших учебных заведений. Физика. 2013. Т. 56, № 9/2. С. 187–190.
8. URL: [http://ddd.fit.cvut.cz/prj/Circ\\_Gen/](http://ddd.fit.cvut.cz/prj/Circ_Gen/)

**Андреева Валентина Валерьевна**, канд. техн. наук, доцент. E-mail: [valentina.andreeva@mail.tsu.ru](mailto:valentina.andreeva@mail.tsu.ru)  
Томский государственный университет

**Сорудейкин Кирилл Александрович**. E-mail: [Kirill.Sorudeykin@relvecorp.com](mailto:Kirill.Sorudeykin@relvecorp.com)  
Харьковский национальный университет радиоэлектроники

Поступила в редакцию 20 ноября 2014 г.

*Andreeva Valentina V., Sorudeykin Kirill A.* (Tomsk State University, Russian Federation, Kharkov National University of Radioelectronics, Ukraine).

**Spatial-related optimization of test sequences based on iteration approach.**

**Keywords:** Tree of decomposition; triple vectors; maximally compatible subsets.

In this paper, we study structures with source-child relations projection. First we provide (deduce) a specification of several indicators and relations of subject matter. Then, we go through practical implementation. As an example, we consider a modification of algorithm proposed earlier in our research. According to our terminology, we process series of triple vectors containing variables, shorten the data by constructing a decomposition tree applying the operations of intersection, merging, and absorption of vectors within groups according to the structure. Then, we form maximally compatible subsets by a decomposition tree and perform hiping for non-specified variables by merging several vectors into one. The task is specified as a matrix of triple variables, and rows are vectors. Each vector is used in operations as a whole in spite of the vectors are series of variables. Maximally compatible subsets are considered in the scope of the given initial task by the Pareto-effective relation.



Earlier, we have introduced several modifications of an algorithm, making several specifications to avoid duplication of indifferent vectors in child nodes and reducing the necessity of covering task. As usual, we determine maximally defined variable to build a next node for a decomposition tree.

So, we apply the distribution by several maximally defined variables in case of a group with the same amount of defined values. From here, less steps and smaller tree are obtained. Note, an iteration approach has been proposed for the procedure, which allows avoiding excess steps and combinatory actions on the assembling stage. At this, recursion and universalization of method are used here. Also, the principle of the Layout-Related optimization is introduced in our paper.

## REFERENCES

1. Andreeva V. Test minimization technique for multiple stuck-at faults of combinational circuits. *Proc. 8th East-West Design&Test International Symposium*, Saint Petersburg, Russia, 2010, pp. 168-170. DOI: 10.1109/EWDTS.2010.5742046
2. Andreeva V. Test Set Compaction Procedure for Combinational Circuits Based On Decomposition Tree. *Proc. 9th East-West Design &Test Symposium*, Sevastopol, Ukraine, 2011, pp. 251-254. DOI: 10.1109/EWDTS.2011.6116596
3. Andreeva V., Sorudeykin K. A research of heuristic optimization approaches to the test set compaction procedure based on decomposition tree. *Proceeding of IEEE EWDTS*, 2012, pp. 382-387. DOI: 10.1109/EWDTS.2013.6673219
4. Sorudeykin K. Irrespective Priority-Based Regular Properties of High-Intensity Virtual Environments. *20th IEEE Telecomm. Forum TELFOR 2012*, Belgrade, Serbia, Nov 2012, pp. 510-513. DOI: 10.1109/TELFOR.2012.6419259
5. Andreeva V.V., Sorudeykin K.A. [A method for minimizing test sequences based on spatial-related optimization with iteration steps] *Materialy 10 oy rossiyskoy konferentsii s mezhdunarodnym uchastiem "Novye informatsionnye tekhnologii v issledovanii slozhnykh struktur" ICAM 2014* [Proc. of the 10th new information technologies in the study of complex structures ICAM 2014]. Tomsk: TSU Publ., 2014, pp. 46-47.
6. Sorudeykin K., Andreeva V. Decomposition Tree-based Compaction Procedure with Iteration Steps for Interconversional Layouts of Tasks. *Proc. 12th IEEE East-West Design & Test Symposium*, 2014, Kiev, Ukraine, 2014, pp. 173-178. DOI: 10.1109/EWDTS.2014.7027091
7. Andreeva V.V., Sorudeykin K.A. Sokrashchenie dliny testovoy posledovatel'nosti na osnove dereva dekompozitsii [Reducing the length of the test sequence based on tree decomposition]. *Izvestiya vysshikh uchebnykh zavedeniy. Fizika*, 2013, vol. 56, no. 9/2, pp. 187-190.
8. *Random Circuits Generators*. Available at: [http://ddd.fit.cvut.cz/prj/Circ\\_Gen/](http://ddd.fit.cvut.cz/prj/Circ_Gen/)