УДК 621.396.253

И. А. ЛЫСЕНКО, аспирант;
А. А. СМИРНОВ, доктор техн. наук, профессор,
Кировоградский национальный технический университет

ИССЛЕДОВАНИЕ АЛГОРИТМА ВЫЯВЛЕНИЯ ВИДА НЕУЧТЕННЫХ ТЕСТОВЫХ СЛУЧАЕВ ПРИ ПРОЕКТИРОВАНИИ ТЕСТОВЫХ НАБОРОВ

Предложен алгоритм выявления неучтенных тестовых случаев в процессе использования упорядоченных каскадных таблиц решений при проектировании тестовых наборов для программного обеспечения.

Ключевые слова: инфокоммуникационная система; программное обеспечение; таблица решений; упорядоченная каскадная таблица решений; тестовый случай.

Введение

Одной из определяющих фаз жизненного цикла программного обеспечения (ПО) инфокоммуникационной системы (ИКС) является фаза тестирования указанного ПО с целью контроля его качества.

Тестирование ПО — это проверка соответствия между реальным и ожидаемым поведением программного продукта, осуществляемая на конечном множестве тестовых наборов, созданных или выбранных определенным образом. В свою очередь, наиболее полную проверку выполнения требований к ПО можно провести на уровне системного тестирования.

При формировании основы для обнаружения и устранения дефектов в процессе системного тестирования ПО имеет смысл воспользоваться методами формализованного описания требований к программному продукту, который подлежит тестированию. Сегодня такие методы рассматриваются, прежде всего, как **методы проектирования *тестовых наборов***, состоящих из одного или нескольких ***тестовых случаев***.

Проектирование тестовых наборов относится к наиболее трудоемким видам интеллектуальной деятельности, связанной с тестированием ПО ИКС. Важностью и сложностью проектирования тестовых наборов определяется необходимость формализации и, возможно, последующей автоматизации выполнения процессов создания тестов с автоматизированным контролем их корректности.

С точки зрения методологии тестирования ПО под ***таблицей решений (ТР)*** понимается таблица, отражающая комбинации входных данных и/или причин с соответствующими выходными данными и/или действиями (следствиями), которая может быть использована для проектирования тестовых случаев и тестовых наборов [1; 4].

С точки зрения математической формализации ТР определяется набором множеств $T = < C, A, U, W >$, где $C = \{c_i\}$, $i = 1, 2, ..., n$, — множество условий; $A = \{a_r\}$, $r = 1, 2, ..., m$ — множество действий; $U = \{u_j\}$, $j = 1, 2, ..., k$, — множество векторов-столбцов матрицы $\|u_{ij}\|$, описывающих совокупность значений элементов множества C; $W = \{w_j\}$ — множество векторов-столбцов булевой матрицы $\|w_{rj}\|$, описывающей подмножества множества A, которые ставятся в соответствие элементам множества U [3].

Однако классические ТР, используемые для формализации тестовых наборов, имеют ограниченные описательные возможности, обусловленные неупорядоченностью множества условий и действий ТР. Этим объясняется и свойство их непроцедурности. Расширить описательные возможности ТР позволяет их модификация — приведение ***к виду упорядоченных каскадных таблиц решений (УКТР)***. Модификация заключается в использовании совместно с таблицей решений ***матрицы следования***, задающей строгий частичный порядок на множестве рассматриваемых условий и действий.

К недостаткам ТР — как обычных, так и УКТР — относится трудность выявления вида неучтенных ситуаций, представляющих собой условную часть тестовых случаев. Рассмотрим ***приближенный алгоритм выявления вида неучтенных тестовых случаев***.

Разработка приближенного алгоритма выявления вида неучтенных тестовых случаев

Из условия неполноты ТР следует, что поиск неучтенных тестовых случаев посредством поиска неучтенных ситуаций можно представить как многократное построение ситуаций $\vee$, ортогональной троичной матрице $\|u_{ij}\|$. Точное решение этой задачи обеспечивается только экспоненциальными алгоритмами, позволяющими получить множество найденных неучтенных ситуаций в виде минимального или ограниченного наперед заданным числом количества обобщенных ситуаций. Приближением может быть отказ от минимизации или ограничения множества искомых неучтенных ситуаций. Если необходимые ситуации будут иметь высокую степень обобщенности, то получаемый результат можно считать достаточно хорошим приближением.

При разработке приближенного алгоритма выявления неучтенных ситуаций за основу предлагается взять алгоритм нахождения корней логического уравнения [2], дополненный эвристическими приемами для сокращения перебора.

Сокращение перебора достигается тем, что по возможности перебираются обобщенные ситуации. Для этой цели эвристические приемы выбора значения очередной переменной в ситуации $\vee$ направляют перебор таким образом, чтобы в первую очередь проверялись миноры матрицы $\|u_{ij}\|$, имеющие меньшие размеры и большее количество элементов $u_{ij} = \lambda$. Отсюда следует, что чем больше степень обобщенности ситуаций, описываемых правилами ТР (т. е. чем больше прочерков в ТР), тем быстрее будет работать этот алгоритм.

Для формально корректной ТР справедливо равенство:

$$\sum_{i=1}^{k} 2^{P_i} = 2^n,$$

где k и n — количество соответственно правил и условий в ТР; P_i — степень обобщенности ситуации, описанной в i-м правиле.

Предположим, что P_i мало отличаются друг от друга. Тогда можно ввести среднюю степень обобщенности $P_{\text{ср}}$ ситуаций в правилах ТР (тестовых случаях) и переписать предыдущее выражение в виде

$$k2^{P_{\text{ср}}} = 2^n,$$

откуда $P_{\text{ср}} = n - \log_2 k$.

Чтобы значение $P_{\text{ср}}$ оставалось неизменным при увеличении размеров ТР, необходимо выполнение следующего условия:

$$\Delta n = \log_2 \Delta k,$$

т. е. с увеличением числа условий количество правил должно расти как 2^n. Для реальных тестовых наборов это условие обычно не выполняется, так как рост количества правил с увеличением числа условий в ТР происходит значительно медленнее.

Следовательно, с увеличением размеров ТР значение $P_{\text{ср}}$ имеет тенденцию к возрастанию, что частично компенсирует увеличение затрат времени на выполнение описанного далее алгоритма.

Основные действия согласно указанному алгоритму выявления неучтенных в ТР ситуаций, представляющих собой условные части тестовых случаев, можно представить в виде последовательно выполняемых шагов.

Шаг 1-й. Завершить выполнение алгоритма при наличии в ТР признаков полноты:

- $\exists(j)\forall(i)\left(u_{ij} = \lambda\right)$;
- $(n = 1) \wedge \exists(j, j')\left(R_j \perp R_{j'}\right)$;
- $\exists(j, j')\left(\forall(i \neq i')\left(u_{ij} = u_{ij'} = \lambda\right) \wedge \left(u_{i'j} \neq \lambda\right) \wedge \right.$
  $\left. \wedge \left(u_{i'j} \neq \lambda\right) \wedge \left(u_{i'j} = \overline{u_{i'j'}}\right)\right)$.

Шаг 2-й. Зафиксировать в искомой ситуации $\vee$ значения переменных, которые нельзя варьировать:

- $v_i = \lambda$, если $\exists(i)\forall(j)\left(u_{ij} = \lambda\right)$;
- $v_i = 1$, если $\exists(i)\forall(j)\left(u_{ij} \neq 1\right)$;
- $v_i = 0$, если $\exists(i)\forall(j)\left(u_{ij} \neq 0\right)$;
- $v_i = \overline{u_{i'j}}$, если $\exists(i)\forall(i \neq i')\left(u_{ij} = \lambda\right)$.

Шаг 3-й. Определить количество N значений переменных, которые нельзя варьировать.

Шаг 4-й. Если $N = n$, то

- выполнить обобщение найденной неучтенной ситуации, последовательно подставив $v_i = \lambda$ вместо зафиксированных значений при условии сохранения ортогональности ситуации v матрице $\|u_{ij}\|$;
- завершить выполнение алгоритма.

Шаг 5-й. Если $0 < N < n$, то

- построить минор матрицы $\|u_{ij}\|$, отбросив строки, соответствующие переменным с зафиксированными значениями, и столбцы, ортогональные этим значениям;
- рекурсивно применить этот же алгоритм для получения минора;
- завершить выполнение алгоритма.

Шаг 6-й. Если $N = 0$, то

- выбрать столбец j^* матрицы $\|u_{ij}\|$, имеющий максимальное значение

$$x_j = \sum_{i=1}^{n} r_{ij},$$

где

$$r_{ij} = \begin{cases} 1, & \text{если } u_{ij} = \lambda, \\ 0 — & \text{в противном случае;} \end{cases}$$

- выбрать строку j^* матрицы $\|u_{ij}\|$, имеющую максимальное значение

$$y_j = \sum_{j=1}^{k} q_{ij},$$

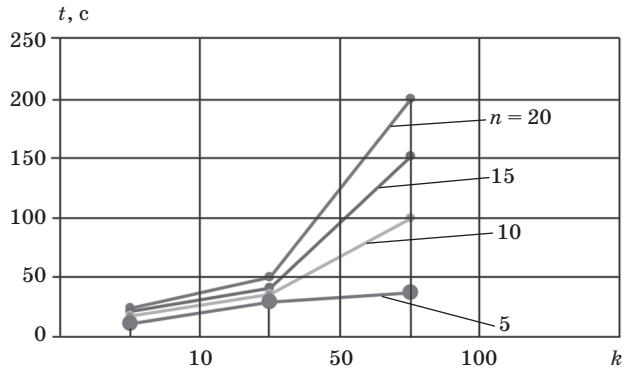
где

$$q_{ij} = \begin{cases} 1, & \text{если } \left(u_{ij^*} \neq \lambda\right) \wedge \left(u_{ij} \neq \lambda\right) \wedge \left(u_{ij^*} \neq u_{ij^*}\right), \\ 0 — & \text{в противном случае;} \end{cases}$$

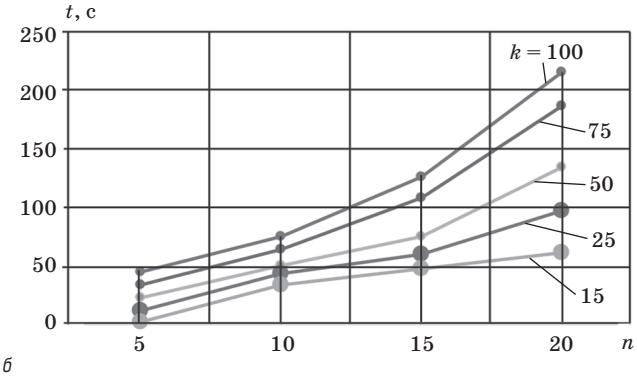
- присвоить $v_i = u_{i^*j^*}$;
- построить минор матрицы $\|u_{ij}\|$, отбросив j^*-ю строку и столбцы, ортогональные значению v_i;
- рекурсивно применить этот же алгоритм для полученного минора;
- если неучтенная ситуация найдена, то выполнить ее обобщение и завершить выполнение алгоритма;
- присвоить $v_i = u_{i^*j^*}$.

Шаг 7-й. Завершить выполнение алгоритма.

Эффективность разработанного алгоритма в различных ситуациях иллюстрируют рис. 1–3.



а



б

Рис. 1. Затраты времени на выявление одной неучтенной элементарной ситуации: *а* — в зависимости от количества *k* правил в ТР при различных значениях *n*; *б* — в зависимости от количества *n* условий в ТР при различных значениях *k*

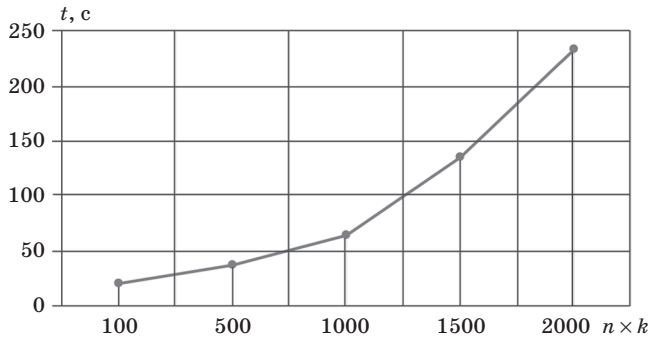


Рис. 2. Затраты времени на выявление одной неучтенной элементарной ситуации в зависимости от объема ($n \times k$) ТР

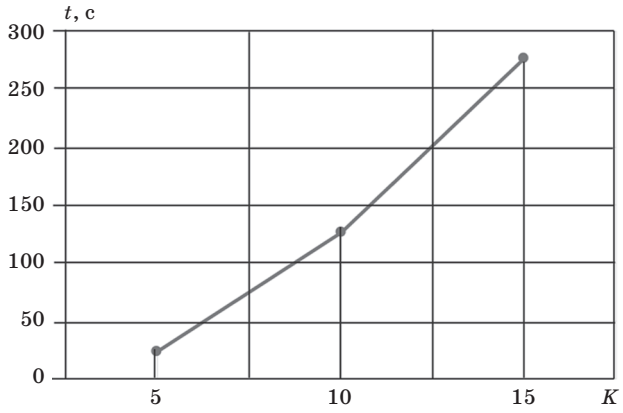


Рис. 3. Затраты времени на выявление неучтенных ситуаций в ТР при $n = 15$ и $k = 25$ в зависимости от количества K получаемых обобщенных ситуаций

Выводы

Благодаря высокой степени обобщенности находимых неучтенных ситуаций, содержащихся в разработанном алгоритме, их количество оказывается либо минимальным, либо близким к нему, что говорит о ***достаточно хорошем приближении полученного результата***. При этом ***выявляемые неучтенные ситуации оказываются взаимно ортогональными***.

Еще одним важным достоинством данного алгоритма является его ***нечувствительность к избыточности и противоречивости в ТР***.

Указанные свойства позволят в дальнейшем успешно использовать алгоритм при разработке тестовых наборов.

Полученные оценки показывают, что ***затраты времени на выполнение разработанного алгоритма с увеличением размеров ТР растут полиномиально***. Этот рост не хуже чем в квадратичной зависимости, по крайней мере для ТР, содержащих до восьми условий и до 256 правил.

Литература

1. ***Степанченко, И. В.*** *Методы тестирования программного обеспечения: учеб. пособие / И. В. Степанченко.— Волгоград: ВолгГТУ, 2006.— 74 с.*

2. ***Орлов, С. А.*** *Технологии разработки программного обеспечения: учебник / С. А. Орлов.— СПб: Питер, 2002.— 464 с.*

3. ***Ахо, А.*** *Теория синтаксического анализа, перевода и компиляции: в 2 т. / А. Ахо, Дж. Ульман.— М.: Мир, 1978.*

Т. 1: Синтаксический анализ.— 612 с.

4. ***Милованов, И. В.*** *Основы разработки программного обеспечения вычислительных систем: учеб. пособие / И. В. Милованов, В. И. Лоскутов.— Тамбов: Изд-во ТГТУ, 2011.— 88 с.*

І. А. Лисенко, О. А. Смірнов

ДОСЛІДЖЕННЯ АЛГОРИТМУ ВИЯВЛЕННЯ ВИДУ НЕВРАХОВАНИХ ТЕСТОВИХ ВИПАДКІВ У ПРОЦЕСІ ПРОЕКТУВАННЯ ТЕСТОВИХ НАБОРІВ

Запропоновано алгоритм виявлення неврахованих тестових випадків у разі використання впорядкованих каскадних таблиць рішень при проектуванні тестових наборів для програмного забезпечення.

Ключові слова: інфокомунікаційна система; програмне забезпечення; впорядкована каскдна таблиця рішень; тестовий випадок.

I. A. Lysenko, A. A. Smirnov

THE RESEARCH OF ALGORITHM FOR DETECTING UNACCOUNTED TEST CASES IN THE PROCESS OF DESIGNING TEST KITS

The algorithm for detecting unaccounted test cases in order to use cascade decisions tables in case of designing test kits for software was proposed.

Keywords: infocommunication system; software; ordered cascade decisions tables; test case.

ЗВ'ЯЗОК

Наукове видання

Редакційна обробка та коректура
О. П. Бондаренко, Т. В. Ількевич

Комп'ютерна верстка да дизайн
Г. С. Тимченко, О. Ю. Апухтіна

Підписано до друку 13.10.2015 р.
Формат 60×84/8. Друк офсетний. Папір друкарський.
Гарнітура SchoolBookC. Наклад 100 прим.

Редакційно-видавничий центр
Державного університету телекомунікацій
03110, м. Київ, вул. Солом'янська, 7
Тел. 249-25-75
E-mail: *zviaz-ok@ukr.net*

Надруковано в РВЦ Державного університету телекомунікацій