Тестирование методом комбинаторного покрытия: обзор публикации NIST SP 800-142

А. Барабанов

9 ноября 2010 г.

Аннотация

Данная статья посвящена обзору и анализу публикации Национального Института стандартов и технологий США (The National Institute of Standards and Technology, NIST) SP 800-142 «Practical Combinatorial Testing». В публикации SP 800-142 представлены практические советы по проведению тестирования программного обеспечения (ПО) методом комбинаторного покрытия, который позволяет повысить эффективность данной процедуры. Данный подход может быть использован и при поведении функционального тестирования ПО по требованиям безопасности информации.

Публикация доступна на сайте NIST (http://csrc.nist.gov/groups/SNS/acts/documents/SP800-142-101006.pdf).

Введение

Ошибки, допущенные при разработке ПО, являются основными причинами уязвимостей информационных систем, приводящих к их некорректной работе и, в частности, к реализациям угроз информационной безопасности (ИБ). Таким образом, тестирование ПО является важнейшей стадией жизненного цикла ПО. В соответствии с исследованиями, проведенными институтом NIST, экономика США несет ежегодные убытки от ошибок в работе ПО в размере 59 млрд. долларов, даже несмотря на то, что производители ПО расходуют на тестирование от 50% до 80% бюджета, выделяемого на разработку.

Исчерпывающее тестирование (*exhaustive testing*) – тестирование всех возможных комбинаций входных воздействий, позволяющее полностью исследовать поведение программы,– является очень трудоёмким и на практике применяется для «критичного» ПО, сбой которого может повлечь риск для человеческих жизней (например, ПО бортовых авиационных систем). Для менее критичного ПО бюджет, выделяемый на разработку, ограничивает объем испытаний, что повышает вероятность остаточных ошибок в ПО, которые могут привести к сбоям в работе и нарушениям ИБ.

Тестирование методом комбинаторного покрытия – подход, позволяющий вместе со снижением затрат на тестирование повысить его эффективность, а так же снизить вероятность остаточных ошибок в ПО.

1 Основная идея

Основная идея данного метода состоит в следующем: на ошибку в работе ПО, в большинстве случаев, влияет не значение одного входного параметра, а комбинация двух или более входных параметров (от 2-х до 6-ти параметров в соответствии с эмпирическими результатами, полученными в ходе исследования, проведенного NIST). Проиллюстрируем применение тестирования методом комбинаторного покрытия. Предположим, что необходимо протестировать программу, принимающую 3 бинарных входных параметра a, b, c (факторы тестирования). При применении метода исчерпывающего тестирования необходимо протестировать $2^3 = 8$ различных комбинаций входных значений. Для 2-х факторного тестирования (тестирование всех возможных комбинаций двух переменных) число способов выбрать 2 переменные из 3 равно $C_3^2 = 3$ (таблица 1, столбец «Факторы»).

Поскольку каждая переменная принимает 2 значения, общее число тестов должно быть $C_3^2 \cdot 2^2 = 12$ (таблица 1, столбец «Тестируемые значения»).

Таблица 1: тестовые случаи при 2-х факторном тестировании

Факторы	**Тестируемые значения**
$\{a, b\}$	$\{0, 0\}$
	$\{0, 1\}$
	$\{1, 0\}$
	$\{1, 1\}$
$\{a, c\}$	$\{0, 0\}$
	$\{0, 1\}$
	$\{1, 0\}$
	$\{1, 1\}$
$\{b, c\}$	$\{0, 0\}$
	$\{0, 1\}$
	$\{1, 0\}$
	$\{1, 1\}$

Но поскольку каждый тест $\{a, b, c\}$ в свою очередь покрывает $C_3^2 = 3$ тестовых случая (таблица 2), то общее число тестов при 2-х факторном тестировании методом комбинаторного покрытия равно 4.

Таблица 2: покрытие исчерпывающего тестирования при 2-х факторном тестировании

Исчерпывающее тестирование	**2-х факторное тестирование**		
$\{a, b, c\}$	$\{a, b\}$	$\{a, c\}$	$\{b, c\}$
$\{0, 0, 0\}$	$\{0, 0\}$	$\{0, 0\}$	$\{0, 0\}$
$\{0, 0, 1\}$	$\{0, 0\}$	$\{0, 1\}$	$\{0, 1\}$
$\{0, 1, 0\}$	$\{0, 1\}$	$\{0, 0\}$	$\{1, 0\}$
$\{0, 1, 1\}$	$\{0, 1\}$	$\{0, 1\}$	$\{1, 1\}$
$\{1, 0, 0\}$	$\{1, 0\}$	$\{1, 0\}$	$\{0, 0\}$
$\{1, 0, 1\}$	$\{1, 0\}$	$\{1, 1\}$	$\{0, 1\}$
$\{1, 1, 0\}$	$\{1, 1\}$	$\{1, 0\}$	$\{1, 0\}$
$\{1, 1, 1\}$	$\{1, 1\}$	$\{1, 1\}$	$\{1, 1\}$

Покрытие 2-х факторного тестирования 4 тестами представлено в таблице 3 (данное представлением имеет общепринятое название «покрывающий массив», *covering array*). Таким образом, применение метода комбинаторных покрытий позволило сократить число тестовых случаем в два раза.

Приведем пример для ПО большей сложности. Для 34 бинарных входных параметров при проведении исчерпывающего тестирования необходимо выполнить $2^{34} = 1,7 \cdot 10^{10}$ операций. Для 3-факторного тестирования методом комбинаторного покрытия понадобиться выполнить всего 33 теста.

В общем случае для проведения t-факторного тестирования n входных параметров, принимающих v значений, потребуется количество тестов, пропорциональное $v^t \cdot \log n$. Отметим, что построение покрывающих массивов t-факторного тестирования является NP-полной задачей.

В настоящее время существует ПО, которое позволяет построить массивы для случаев $t = 2...6$ (генератор покрывающих массивов доступен на сайте NIST, http://csrc.nist.gov/groups/SNS/acts/).

Таблица 3: покрытие 2-х факторного тестирования

Факторы	**Тестируемые значения**	**Покрытие** $\{a, b, c\}$
$\{a, b\}$	$\{0, 0\}$	$\{0, 0, 0\}$
	$\{0, 1\}$	$\{0, 1, 1\}$
	$\{1, 0\}$	$\{1, 0, 1\}$
	$\{1, 1\}$	$\{1, 1, 0\}$
$\{a, c\}$	$\{0, 0\}$	$\{0, 0, 0\}$
	$\{0, 1\}$	$\{0, 1, 1\}$
	$\{1, 0\}$	$\{1, 1, 0\}$
	$\{1, 1\}$	$\{1, 0, 1\}$
$\{b, c\}$	$\{0, 0\}$	$\{0, 0, 0\}$
	$\{0, 1\}$	$\{1, 0, 1\}$
	$\{1, 0\}$	$\{1, 1, 0\}$
	$\{1, 1\}$	$\{0, 1, 1\}$

2 Виды тестирования методом комбинаторных покрытий

Метод комбинаторных покрытий на практике может быть применим для следующих видов тестирования.

1. **Тестирование конфигураций**: современное ПО работает под управлением различных операционных систем (ОС), взаимодействует с различным сторонним ПО (например, с СУБД), использует различные сетевые протоколы и т. д. Таким образом, тестирование ПО должно быть проведено под всеми конфигурациями для снижения риска некорректного функционирования программы.
2. **Тестирование входных параметров**: тестирование возможных комбинаций входных параметров, позволяющее исследовать поведение программы, сравнить фактические результаты работы ПО с результатами, описанными в спецификации.

Применение метода комбинаторных покрытий представлено на рисунке 1.

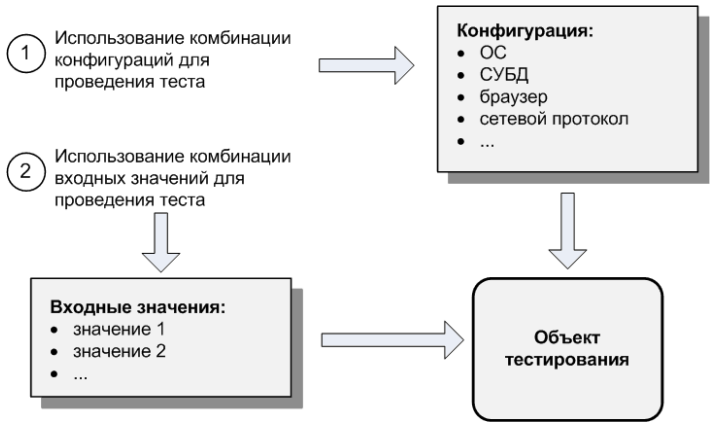

Рис. 1: Применение метода комбинаторного покрытия

3 Некоторые эмпирические результаты

В данном разделе приведены некоторые эмпирические результаты, полученные NIST при проведении исследования. В таблице 4 представлено количество программ (в процентном соотношений), в которых удалось обнаружить ошибки при проведении k-факторного тестирования методом комбинаторного покрытия.

Таблица 4: Число выявленных ошибок при проведении k-факторного тестирования

Количество переменных, k	**ПО мед. учреждений**	**Браузеры**	**Серверное ПО**	**ПО, реализующие сетевые протолы**
1	66	29	42	17
2	97	76	70	62
3	99	95	89	87
4	100	97	96	98
5		99	96	100
6		100	100	

В таблице 5 представлено распределение результатов k-факторного тестирования методом комбинаторного покрытия, которое позволило найти 3045 ошибок в различных программах, способных привести к уязвимости типа «отказ в обслуживании».

Таблица 5: Число выявленных уязвимостей при проведении k-факторного тестирования

Количество переменных, k	**Число выявленных уязвимостей, %**
1	93%
2	99%
3	100%
4	100%
5	100%
6	100%

4 Заключение

Тестирование методом комбинаторного покрытия является хорошей альтернативой исчерпывающему тестированию и позволяет снизить экономические и временные затраты на данном этапе жизненного цикла ПО. Эмпирические результаты, полученные NIST, позволяют сделать вывод об эффективности данного подхода. Применение данного метода позволяет существенно уменьшить вероятность остаточных ошибок в ПО и, как следствие, снизить риск нарушения корректной работы программы. Подход может быть рекомендован для применения при тестировании «некритичного» ПО.