РЕГРЕССИОННОЕ ТЕСТИРОВАНИЕ И РАЗРАБОТКА WEB – ПРИЛОЖЕНИЯ ДЛЯ ТЕСТИРОВЩИКА С ФУНКЦИОНАЛОМ МИНИМИЗАЦИИ ЧИСЛА ТЕСТОВЫХ НАБОРОВ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Дума В. Д.

Поттосина С.А. – к. ф.-м. н., доцент

Регрессионное тестирование – это выборочное тестирование, позволяющее убедиться, что изменения не вызвали нежелательных побочных эффектов, или что измененная система по-прежнему соответствует требованиям. После каждой модификации программы необходимо удостовериться, что на функциональность программы не оказал влияния модифицированный код.

Регрессионное тестирование – дорогостоящий род деятельности: процесс регрессионного тестирования может включать исполнение достаточно большого количества тестов на скорректированной программе, даже если изменений очень мало. Несмотря на то, что усилия, требуемые для внесения небольших изменений, как правило, минимальны, они могут требовать достаточно больших усилий для проверки качества изменённой программы. Тем не менее, проведение регрессионного тестирования необходимо. Надёжная и эффективная разработка и сопровождение программного обеспечения невозможна без регрессионного тестирования. Выполнить полное регрессивное тестирование вряд ли возможно. По мере роста и расширения становится все сложнее тестировать отдельные части системы программного обеспечения. Эта проблема усложняется из-за частоты создания сборок программ. Необходимо тестировать предыдущую функциональность, чтобы обеспечить возможность тестирования новых исправлений и новых функций [1].

Целью работы является совершенствование регрессионного тестирования за счет разработки программного модуля, основанного на минимизации наборов тестов с использованием минимаксного алгоритма, позволяющем сократить временные затраты на проведение регрессионного тестирования.

В отличие от задач традиционной математики, где решение получается с помощью целенаправленной вычислительной процедуры, однозначно ведущей к цели, решение комбинаторной задачи сводится зачастую к полному перебору различных вариантов. Перебираются и испытываются конструкции определённого вида, среди которых должно находиться решение задачи. Как только выясняется, что очередная конструкция является решением, процесс поиска решения можно считать завершённым. Комбинаторные задачи характерны ещё тем, что множество, среди элементов которого отыскивается решение, всегда конечно. Реализовав полный перебор, либо найдём решение, либо убедимся в том, что решения нет. Таким образом, всякая подобная задача может быть решена за конечное время [2].

В настоящее время выделяют следующие методы отбора тестов для регрессионного тестирования: случайные методы; безопасные методы; методы минимизации; методы, основанные на покрытии кода. При отборе тестов для проведения регрессионного тестирования безопасным выборочным методом, можно воспользоваться комбинаторной задачей о кратчайшем покрытии булевой матрицы, которая ставится следующим образом. Пусть дано некоторое множество A = {a1, a2, …, an} и совокупность его подмножеств B1, B2, …, Bт, т. е. Bi $\subseteq$ A, i = 1, 2, …, m, причем B1 $\cup$ B2 $\cup$ … $\cup$ Bт = A. Требуется среди данных подмножеств выделить такую совокупность Bi1, Bi2 ,…, Bik минимальным k, чтобы каждый элемент из A попал хотя бы в одно из Bij (j = 1, 2, …, k), т. е. Bi1 $\cup$ Bi2 $\cup$… $\cup$ Bik = A. [3]

В качестве булевой матрицы выступает диагностическая матрица R, столбцы которой соответствуют тестируемым функционалам программы, а строки соответствуют тестам. Элемент матрицы rij равен 1, если для тестирования функционала fi пригоден тест tj, иначе элемент rij. матрицы R равен 0. Необходимо найти минимальной число строк матрицы, покрывающих все столбцы матрицы.

В качестве примера рассмотрен реальный проект «Open Ladger»

Для этого выделены 23 ключевых элемента системы (тестируемый функционал), безотказная работа которых была критична для данного приложения. К ним относятся: 1, 2, 3, 4, 5 и т.д. При тестировании этого проекта были выбраны тесты: a, b, c, d, e, f, g, h.

Диагностическая матрица C приведена на рисунке 1.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	
1	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	a
0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	b
0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	c
1	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	d
0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	e
0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	f
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	g
0	0	0	0	1	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	h

Рис.1 – Матрица C

Далее полученной диагностической матрицы был использован минимаксный алгоритм поиска крат-

чайшего строчного покрытия.

Алгоритм нахождения кратчайшего строчного покрытия:

Ищется столбец с минимальным числом единиц. Если таковых несколько, то выбирается любой.

Среди строк, покрывающих этот столбец, ищется строка с максимальным числом единиц и заносится в покрытие (следовательно, удаляется из матрицы); если же таких строк несколько, то выбирается любая из них.

Удаляются все столбцы, которые покрывает полученная строка.

Действия продолжаем до тех пор, пока не удалится вся матрица.

Результаты оказались достаточно успешными: удалось сократить тестовый набор на 50% (4 теста, а именно, тесты b, c, d, h) без потери процента покрытия. Следовательно, сократились временные затраты на проведение регрессионного тестирования без потери качества программного продукта.

Был разработан программный модуль, позволяющий сократить время, отведенное на проведение регрессионного тестирования за счет разработки программного модуля, основанного на минимизации наборов тестов с использованием минимаксного алгоритма. В программе присутствуют и все инструменты необходимые тестировщику для полноценной работы, весь функционал показан на диаграмме вариантов использования представленный на рисунке 2.

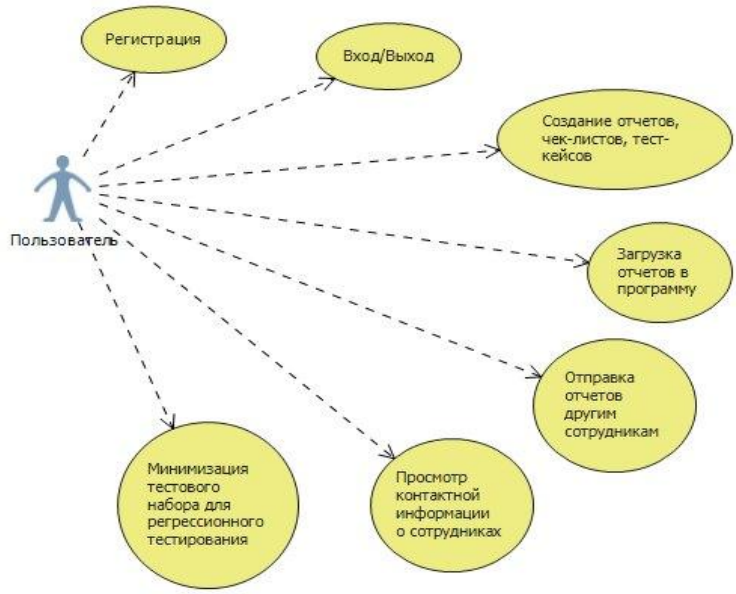

Рис.1 – Спецификация вариантов использования

Для достижения цели в работе были решены следующие задачи:

1. Был проведен анализ существующих методов и средств регрессионного тестирования ПО и выбрать требований к системе.
2. Выбран метод регрессионного тестирования, который обеспечивает выбор минимального набора тестов, также является безопасным и целесообразным, то есть способным выбирать 100% тестов, обнаруживающих ошибки, и оправдывать расходы на собственное применение.
3. Создан инструментарий, поддерживающий процесс выбора минимального набора регрессионных тестов.
4. Проанализирован результат применения инструментального пакета.

Практическая ценность работы заключается в разработке безопасной методики проведения регрессионного тестирования программных средств, обеспечивающей высокую точность и эффективность. Создании программного комплекса поддержки регрессионного тестирования.

Внедрение разработанных методов и средств позволит снизить трудоемкость фазы регрессионного тестирования программного обеспечения и улучшить качество выпускаемого программного продукта.

Список использованных источников:

1. С. С. Куликов. Тестирование программного обеспечения. Базовый курс: практ. пособие. / С. С. Куликов. — Минск: Четыре четверти, 2015. — 294 с

2. А. Д. Закревский, Ю. В. Поттосин, Л. Д. Черемисинова «Основы логического проектирования» Книга 1 «Комбинаторные алгоритмы дискретной математики», Минск, 2004, с.77-82.

3. А. Д. Закревский, Ю. В. Поттосин, Л. Д. Черемисинова «Логические основы проектирования дискретных устройств», Москва, 2007, с. 64.