УДК 004.415.53

Александрова Елена Григорьевна,
инженер ИВЦ, ФГБОУ ВО «Ангарский государственный технический университет»,
e-mail: qweb@angtu.ru
Добрынина Надежда Николаевна,
к.т.н., доцент, ФГБОУ ВО «Ангарский государственный технический университет»,
e-mail: priem@angtu.ru

МАТЕМАТИЧЕСКИЕ МЕТОДЫ В ТЕСТИРОВАНИИ

Aleksandrova E.G, Dobrynina N.N.

MATHEMATICAL METHODS IN TESTING

Аннотация. *В статье рассмотрено применение дискретных математических методов, с помощью которых тестировщик может анализировать связи между структурой программного продукта и логикой его поведения.*

Ключевые слова: *тестирование, математические методы, тест-кейсы, чек-листы.*

Abstract. *The article discusses the use of discrete mathematical methods, with the help of which a tester can analyze the connections between the structure of a software product and the logic of its behavior.*

Keywords: *testing, mathematical methods, test cases, checklists.*

Каждый день инженеры по обеспечению качества взаимодействуют с логическими формулами, поведением параметров и структурных пластов информации. В их профессиональные задачи входит проверка и контроль за состоянием систем.

А это значит, тестировщику крайне полезно обладать математическим и алгоритмическим мышлением, чтобы хорошо понимать логику проверяемого программного обеспечения.

Если QA применяет математический подход в своей деятельности, все логические операции в процессе тестирования программного обеспечения будут качественно связанны между собой, действия будут последовательными, а любая проверяемая функция будет тщательно анализироваться.

Знания дискретной математики позволяют отыскать оптимальное решение в любой рабочей ситуации. К примеру, можно настроить оптимальный набор тестовых случаев, не включая в них все допустимые сценарии.

Также, дискретная математика позволяет визуализировать точные параметры ПО, которые были проверены и покрыты тестами.

В процессе проведения тестирования ожидаемые результаты поведения ПО могут кардинальным образом отличаться от фактического результата. Из этого следует, что базовая задача отдела тестирования – максимально охватить все допустимые тестовые случаи при проверке программного обеспечения.

Чтобы лучше понять применение познаний дискретной математики в сфере QA, можно обратиться к диаграмме Венна (рисунок 1), которая наглядно иллюстрирует подобную взаимосвязь.

Рисунок 1 - диаграмма Венна

Подобная диаграмма позволяет не просто определить набор необходимых тестовых ситуаций, но и также провести сравнение использования (целесообразности применения) того или иного набора. Только дискретная математика позволяет оптимизировать и анализировать наборы тестовых действий, влияющих на функционирование программного комплекса.

Теорию множеств стоит рассмотреть на примере наглядного тестового случая, когда тестировщику поставлена задача по проверке логики работы приложения «Следующий день» (программа, что показывает какой день будет следующий после введенной даты).

Итак, теория множеств позволяет создать так называемый псевдокод:

M1={month:month has 30 days}

M2={month:month has 31 days except December}

M3={month:month is February}

M4={month:month is December}
D1={day:1<=day<=28}

D2={day:1<=day<=29}

D3={day:1<=day<=30}
D4={day:1<=day<=31}

Y1={year:year is a leap year}

Y2={year:year is not a leap year}

Основываясь на этих данных, тестеровщики могут проводить всевозможные проверки. Такой формат данных не только повышает скорость обработки тестов программным обеспечением, но и снижает риск пропуска глобальной ошибки.

Большая часть дискретной математики основана на так называемых «теориях графов», изучающих графы. Эти графы применяются для качественного представления связей между данными или объектами. Наиболее распространенный пример графа – компьютерная сеть.

Графы играют основополагающую роль для процесса разработки программного обеспечения. К примеру, с их помощью можно разложить некоторые функции (сложные задачи) на мелкие составляющие части, что позволит эффективно понимать необходимые бизнес-процессы.

Вернемся непосредственно к процессу тестирования ПО и представим, что у нас есть некий поток процесса (к примеру, перемещение задачи внутри системы отслеживания задач). Таким образом тестировщик располагает условием некоторой задачи и может перенести ее на другой этап (ориентированный граф), или может достичь определенной точки, где нет возможности редактировать сущность (неориентированный граф).

Следующий пример: тестировщик получил набор сущностей и действий, которые можно выполнять с этими сущностями. А с помощью матрицы смежности у него есть возможность сократить набор тестовых случаев.

Именно процесс сокращения тестовых случаев является наиболее важной частью любого процесса тестирования программного обеспечения. Ведь возникает максимальный охват тестирования и избегается процедура проведения ненужных проверок.

Задача тестировщика – охватить вебпродукт, применяя эффективные контрольные примеры, с помощью которых можно проверять все допустимые комбинации действий с программным обеспечением. QA-специалисты могут добиться максимального успеха, прилагая небольшие усилия, основываясь на базовых принципах дискретной математики (алгоритмы) для поиска оптимальных наборов тестовых случаев.

Также дискретная математика позволяет понимать, как на самом деле производится программное обеспечение, ведь любой вебпродукт использует специальные алгоритмы и методы дискретной математики (вместе с логикой). Подобный тезис означает то, что, если мы знаем, как работает программа, мы можем отыскать внутри нее ошибку, которая не будет видна рядовому пользователю.

Рассмотрим, как программное обеспечение работает на основе микросервисной технологии снабженной сетью Петри (рисунок 2).

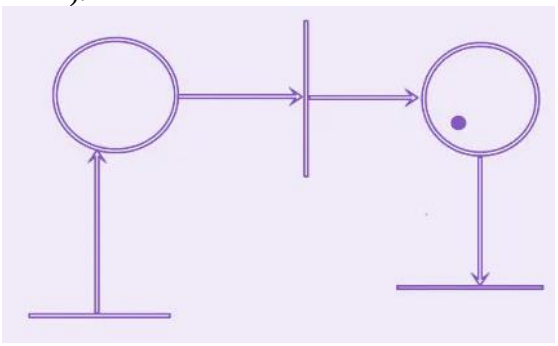

Рисунок 2 - Сети Петри

На картинке видно, что система обладает начальным состоянием и должна получить входящий сигнал, который направляется от другой системы. В зависимости от полученного итога должно выполниться определенное действие.

Другими словами, сети Петри показывают динамику всей системы. Если есть определенная проблема, ее можно очень быстро локализировать.

Все искусственные нейронные сети также базируются на принципах графа. Они обладают способностями обработки данных нейронами человеческого мозга. Любая часть нейронной системы базируется на графе, в содержание которой входят «входные» узлы, «скрытый» слой и «исходящие» узлы.

Определенная часть информации поступает на входной слой, а ранее добавленные алгоритмы скрытого слоя позволяют об-

рабатывать данные с последующей отправкой итогов на исходящий этап. То есть, нейронная сеть может выполнять действия на основе таких данных.

Все модели ИНС требуют обучения или расчёта весов связей. В общем случае, обучение – такой выбор параметров сети, при котором сеть лучше всего справляется с поставленной проблемой. Обучение — это задача многомерной оптимизации, и для ее решения существует множество алгоритмов.

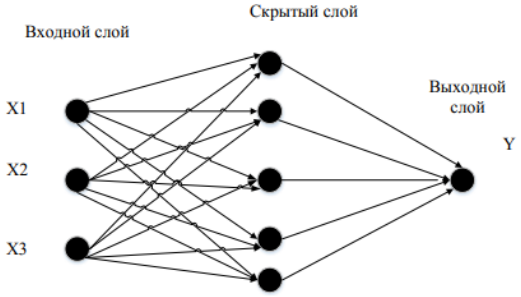

Рисунок 3 – Структура нейронной сети

Технически обучение заключается в нахождении коэффициентов связей между нейронами. В процессе обучения нейронная сеть способна выявлять сложные зависимости между входными данными и выходными, а также выполнять обобщение. Это значит, что в случае успешного обучения сеть сможет вернуть верный результат на основании данных, которые отсутствовали в обучающей выборке.

Последний пример применения дискретной математики в QA рассмотрен на основе построения тестирования программного обеспечения. Есть большое количество методологий проверок и подходов с названием «millennium testing», которые были разработаны и внедрены в обиход еще в начале 2000-х годов.

BDD (от англ. Behavior Driven Development) – часть так называемого миллениум тестирования, которая позволяет тестировщикам выстраивать более тесную связь между приемкой функций и проверками, используемыми для тестирования определенной функциональности.

Базовая структура рабочего функционирования BDD (рисунок 4) выстроена на базе динамического графа (Сети Петри).

Рисунок 4 - QA начинает BDD автоматизацию

Таким образом, базовые преимущества использования наработок дискретной математики в повседневной деятельности компании по обеспечению качества:

- эффективная помощь в понимании бизнес-логики разрабатываемого функционала;
- возможность дробления сложных задач на мелкие компоненты;
- выполнение эффективного тестирования с меньшей затратой выделенного времени;
- наглядная визуализация внутреннего строения архитектуры ПО.
- Однозначно, дискретная математика позволяет повышать эффективность тестирования любого программного обеспечения. Каждый аспект дискретной математики упрощает интерпретацию программного обеспечения и всех его жизненных циклов.

СПИСОК ЛИТЕРАТУРЫ

1. **Куликов С.С.** Тестирование программного обеспечения. Базовый курс / Куликов С.С. – Москва: "Химия", 1981. – 200 с

2. **Александрова Е. Г., Добрынина Н. Н.** Жизненный цикл и основные принципы тестирования // Современные технологии и научно-технический прогресс. 2023. №. 1. С. 95-96.

3. **Александрова Е. Г., Добрынина Н. Н.** Место тестирования в различных моделях разработки программного обеспечения // Современные технологии и научно-технический прогресс. 2023. №. 1. С. 97-98.