Дискретные симуляции для дизайна и разработки распределительных систем

Н. А. Ефанов
Финансовый университет при Правительстве Российской Федерации (Финуниверситет), Financial University
naefanoff@yandex.ru

***Аннотация*. На сегодняшний день наблюдается тенденция к усложнению любой системы. Но, ведь чем сложнее система, тем больше в ней структурных и системно-образующих элементов, тем сложнее в них связи, и тем более сложен процесс прогнозирования и анализа подобных систем. В этом случае возрастает риск уязвимости подобной системы, а также возникают вопросы по поводу эффективности ее функционирования. Поэтому важным становится исследование систем относительно этих показателей.**

Ключевые слова: модель; подход; моделирование; система; развитие; аналитические методы

Разработчики распределительных систем создают дискретные симуляции, которые помогают понять и изучить поведение внутри – компонентных протоколов. Обычно эти протоколы написаны на основе императивных программных языков. Основная функциональность этого метода состоит в фокусировке на распределении и его основных свойствах. Эксперты задаются вопросами о возможностях использования симуляций для тестирования приложений распределительных систем. Поскольку симуляции отвечают определенным особенностям, код симуляции может быть рассмотрен как нетрадиционная, формальная модель. Основная модель, использованная экспертами это fault-based аналитика распределительного кода, в которой сет мутантов выявлен с использованием стандартных код-мутантных техник. Этот анализ может быть использован для определения эффективности тестируемой информации, а также для тестирования критериев подбора данных.

I. Применение дискретных симуляция

Использование дискретных симуляции для дизайна и разработки распределительных систем очень популярно. Оно привлекает разработчиков своей эффективностью и рентабельностью.

Симуляции использованы для понимания и тестирования функциональности сложных внутри-компонентных протоколов и алгоритмов. Обычно они написаны с использованием императивных программных языков, таких как C++ и Java. Симуляции помогают разработчику определить основные алгоритмические функциональности. На основе множественных тестирований и наблюдений, эксперты пришли к выводу о том, что код, использованный в симуляциях распределительных систем есть формальная частное функционального поведения этой систему. Поэтому операционный код может быть рассмотрен как операционная, формальная модель для этой системы.

Основываясь на этих наблюдениях, эксперты сделали вывод о том, симуляции могут быть использованы разработчиками для выявления эффективным тестируемых величин. Интуиция позади этого вывода состоит в том, что поскольку симуляционный код – это частное определенного поведения, то он может быть использован для анализа тестируемых сетов и для прогнозирования эффективности выявления ошибки в применении.

Тестируемые данные собраны, используя критерий четкости. Критерий четкости изучен как способ организации тестируемых активностей, служащих как стоп отсечка и одновременно как критерий измерения продвижения к цели. На сегодняшний день критерий точности не был изучен как специально в контексте распределительных систем. На сегодняшний момент эксперты не сколько сосредоточены на выявлении новых критериев, сколько на возможности помочь разработчику в определении возможностей использования данного сета информации для поставленной задачи. Забегая вперед, это практически невозможно доказать универсальную эффективность индивидуального критерия. Очень вероятно, что различные критерии будут более эффективны для различных систем. Это должно быть особенно верно для распределительных систем. Поэтому метод для оценки соревнующихся критериев будет очень полезен для разработчиков. По мнению экспертов, конечная цель тестируемой стратегии состоит в выявлении эффективности и возможности для легкого применения. Основной метод, использованный экспертами – fault based анализ симуляционного кода. Основная идея состоит в выработке сета мутантов из симуляционного кода, используя стандартные код-мутантные техники.

II. Симуляционные тестирование

Симуляционные тестирование использовано для простого и общего процесса. Первый шаг, разработчик определяет тестируемую базу данных. Симуляционный код играет роль в определении особенностей тестирования. Поэтому, симуляции использована для определения точности и применимости тестируемой базы данных. На более высоком уровне симуляционное тестирование основывается на двух идеях. Первая идея заключается в том, что симуляционный код позволяет

определить системные критерии точности. После того как критерий определен, разработчик может протестировать базу данных на основе полученного критерия. Это использование симуляции требует от разработчика определение одного или нескольких критериев точности. Однако, не всегда возможно определить критерий точности. Может из-за недостатка информации или критерий неизвестен. В таком случае вторая идея использования симуляционного тестирования состоит в составление рейтингового механизма. Этот рейтинговый механизм так же основан на применении симуляционного кода и получен на основе fault-based анализа симуляционного кода.

A. Анализ неисправностей

В анализе, основанном на ошибках стратегии тестирования, критерии адекватности сравниваются по их способности обнаруживать классы отказов. Классы ошибок обычно проявляются как операторы мутации, которые изменяют правильную спецификацию, используя четко определенные способы создания набора неправильных версий. Эти неправильные версии или мутанты, могут использоваться для сравнения стратегий тестирования.

Например, реализация может иметь ошибку, которая вызывает упущение определенного состояния. Такие изменения представлены как переходы в спецификации конечного состояния. Этот недостающий переход отказа представляется в области спецификации всеми спецификациями, которые могут быть получены в исходную спецификацию, удалив один из переходов. Стратегии тестирования, способные отличить неправильные спецификации от правильных соответствующих этому конкретному классу неисправности. Основные допущения анализа, основанного на ошибках, состоят в том, что простые синтаксические ошибки в спецификации представляют собой ошибки, которые могут возникнуть на практике, поэтому стратегия тестирования охватывает конкретный класс отказов.

Предпосылкой анализа, основанного на ошибках, является наличие набора операторов мутаций, которые могут быть применены к анализируемой спецификации. Моделирование обычно кодируется в императивных языках программирования, и, поэтому, хорошо подходят для операторов кодирования мутаций, разработанных в контексте тестирования.

Эти операторы делают простые синтаксические изменения в коде, что может привести к семантическим различиям.

При проведении анализа, основанного на ошибках, будем применять стандартные кодовые мутационные операторы к коду моделирования, тем самым, получены набора дефектных спецификаций. Для оценки набора тестов запускаются все тестовые примеры по всем мутантным симуляциям. Для каждого прогона может возникнуть один из следующих вариантов?

- имитация заканчивается обоснованными результатами;
- симуляция заканчивается необоснованными результатами;
- симуляция не заканчивается;
- симуляция недействительна и прерывается.

В большинстве мутационных анализов происходит сравнение вывода из исходных и мутационных версий. Но это не всегда возможно в моделировании дискретных событий распределенных систем. Это объясняется тем, что ядро дискретного события использует отдельные потоки для выполнения каждого процесса и поэтому является недетерминированным. На практике мы используется утверждение и проверка здравомыслия в коде моделирования, чтобы определить, какие результаты будут считаться обоснованными.

B. Методы

Наиболее оптимальным является использование анализа на основе ошибок кода моделирования и адекватности критериев, которые используются индивидуально или в сочетании с другими.

Описание метода с использованием шести разных сценариев. В каждом сценарии существует предположение, что код моделирования доступен.

Нет информации.
Первый сценарий: разработчик практически не имеет опыта работы с системой. Поэтому разработчик не может выбрать какой-либо конкретный критерий достаточности и ищет любые рекомендации относительно того, как оценивать тестовые комплекты. В этом случае будет удобным использовать анализ, основанный на ошибках.

Разработчик генерирует несколько наборов тестов, получает свой балл мутантов и выбирает набор с помощью самых высоких баллов мутантов.

Критерии, основанные на входных значениях.
Разработчик выбирает критерий, который зависит исключительно от ввода. Примером является набор, который считается адекватным, т.е. если он охватывает разделы входного пространства. В этом случае код моделирования не имеет смысла при оценке адекватности, и разработчик должен это сделать через другие средства. Тем не менее, тестирование на основе моделирования может по-прежнему способствовать этому сценарию, предоставляя относительный рейтинг адекватных номеров на основе оценки мутанта.

Критерии, основанные на явлениях окружающей среды.
Разработчик выбирает критерий, который не зависит от кода моделирования, но при этом зависит от наблюдаемых событий в окружающей среды. Например, критерий может требовать, чтобы линии связи использовались до максимальной емкости для большего количества времени при выполнении теста. В этом сценарии симуляция может непосредственно оценить адекватность набора тестов, но

разработчик должен выполнить тестовые примеры, используя код моделирования, специально предназначенный для регистрации событий, представляющих интерес.

Оценка критериев по рейтингу.

Разработчик хочет выбрать критерий, однако, для в системе существует множество применимых критериев, относительная эффективность которых неизвестна. Разработчик создает адекватные пакеты для каждого из них, а затем выбирает набор с самый высокий баллом мутантов.

Таким образом, симуляция может использоваться непосредственно для оценки адекватности набора тестов в отношении критериев, основанных на окружающей среде и на порядке моделирования. Что требует запуска тестового набора через инструментальное моделирование. Кроме того, симуляция может быть использована для повышения эффективности критерия, включая критерий «нулевой», и информировать программиста о выборе критерия.

III. ТЕМАТИЧЕСКИЕ СИСТЕМЫ

Будем использовать моделирование и реализация двух распределенных систем, которые хорошо известные среди распределенных алгоритмов. Первый, GBN, является алгоритмом «обратный-n», который используется для надежной передачи данных по ненадежному уровню связи. Второй, LSR, представляет собой схему маршрутизации состояния канала, которая использует Алгоритм Дейкстра в каждом компоненте децентрализованной коллекции маршрутизаторов для вычисления локальных сообщений поведения.

Реализация обеих систем предоставлена в качестве заданий программирования. Для GBN была использована Java-реализация и создана ошибочная версии с использованием MuJava (инструмент автоматической мутации). Для моделирования использовался механизм моделирования дискретных событий sim-java.

Разработан тонкий слой над ядром дискретного события, который обеспечивает более естественный способ планирования и получения результатов. Этот уровень также включает в себя процесс, реализующий поведение вероятностно ненадежной сети, который используется обеими системами.

3.1 *GBN*

GBN включает в себя два процесса: отправитель, который выводит пакеты данных и ждет подтверждения и получатель, который ждет пакетов данных и ответов с подтверждениями. Эти процессы поддерживают состояние порядкового номера, который обеспечивает получение пакетов данных в правильном порядке. Отправитель также поддерживает скользящее окно отправленных пакетов, из которых данные могут быть повторно переданы, если подтверждения не поступают в течение определенного периода времени.

GBN реализуется в простой клиентской / серверной программе передачи файлов. Вызывается клиентская программа с указанием пути к существующему файлу для чтения и передачи, а серверной программе предоставляется путь к файлу, который будет создан и заполнен полученными данными.

3.2 *LSR*

LSR – это схема маршрутизации состояния канала (link-state routing), в которой каждый маршрутизатор использует полные глобальные знания о сети для вычислить его таблицы переадресации. Система LSR использует Алгоритм Дейкстра для вычисления наименее затратных путей между всеми узлами сети. Эта информация перегоняется для построения таблиц переадресации на каждом узле. Для того, чтобы уменьшить сложность назначения, основная сеть не задерживает или не отбрасывает сообщения.

3.2.1 *Спецификация*

Код моделирования LSR состоит из трех событий, нескольких поддерживающих структур данных, реализации класса Алгоритм Дейкстра, типа процесса маршрутизатора и типа процесса клиента для ввода сообщений, в общей сложности примерно 180 строк кода Java. Маршрутизатор принимает в качестве входных данных список своих прямых соседей и затраты на ссылки на каждый из них.

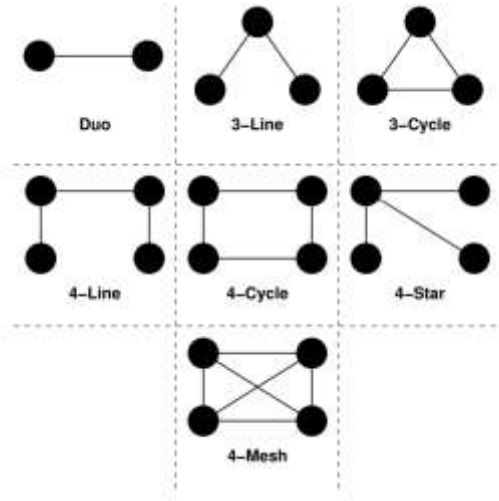

Рис. 1. Топологии LSR

Моделирование LSR определяется следующими параметризованными значениями:

1. топология: целое число в диапазоне [1,7], представляющее конкретное расположение маршрутизаторов и затраты
(На рис. 1 показана каждая возможность графически);

2. количество сообщений: количество отправляемых сообщений, значение в диапазоне [0, 2n], где n - номер маршрутизаторов в топологии;

3. источник и назначение сообщения: для каждого сообщения исходный маршрутизатор и целевой маршрутизатор выбирается случайным образом из диапазона [1, n] с разрешенными локальными сообщениями.

Жестко закодированные детали каждой топологии, включая статически вычисленный кратчайший путь, затраты на все пары маршрутизаторов. При моделировании n-маршрутизаторов создаются и упорядочиваются в соответствии с указанными топологиями. Затем создаются и планируются экземпляры клиента mc (где mc равно количеству сообщений), выполняемые с регулярными интервалами с указанным маршрутизатором источника и получателя. Каждый клиент публикует короткую текстовую строку. Поскольку, каждая публикация распространяется маршрутизатором, переменная-член класса-пути обновляется с помощью затраты на пройденные каналы. В тот момент, когда маршрутизатор получает публикацию, которая будет доставлена локально, она сохраняет строку и ее общую стоимость пути. Когда симуляция завершается, утверждения гарантируют, что каждый полученный маршрутизатор ожидаемое количество публикаций, и что стоимость пути для каждой публикации также верна.

IV. Экспериментs

В экспериментах рассматриваются критерии белого квадрата и ввода-разбиения. Критерии белого квадрата – это общеизвестные блоки, где все ветви и всевозможные покрытия, примененные к коду моделирования. В частности, применяя эти критерии в совокупности ко всей базе кодов моделирования, исключая классы, которые являются частью ядра моделирования дискретных событий и наш уровень API.

Критерии ввода-разбиения определяются относительно входной области моделирования.

Сравнение критериев адекватности, основанное на их эффективности, измеряется, как средняя ошибок, обнаруженных соответствующими наборами тестов. Данная метрика более подходит для спецификации тестирования, поскольку, учитывается широта эффективности пакета. Другими словами, поскольку адекватность измеренный в отношении охвата спецификации, адекватный набор тестов должен соответствовать любым реализациям спецификации.

Чтобы определить статистически значимые отношения эффективности, применяется тестирование гипотез к каждой паре критериев и вычислить p-значение. Значение p можно интерпретировать, как наименьшее α-значение, при котором нулевая гипотеза будет отвергнута, где α – вероятность отклонения нулевой гипотезы, когда она фактически зафиксирована.

Например, при определении того, является ли критерий A более эффективным, чем критерий B, предлагается нулевая гипотеза (H0) и альтернативная гипотеза (Ha):

$H0 : A \leq B$

$Ha : A > B$

где “>” означает «более эффективный, чем». Хотя неизвестно фактическое распределение эффективности, мы воспользуемся центральной предельной теоремой статистики и предположения, что распределение нормализованной формы нашей тестовой статистики (E) аппроксимирует нормальное распределение.

Согласно предположению, мы можем использовать эту теорему с размерами выборки большей 30. Поэтому мы вычисляем значение z для этой гипотезы и используем формулу p-значения для тестов, когда область отклонения состоит из высоких значений z:

$$z = EA - EB\ \sigma EA / \sqrt{n}$$

$$p = 1 - \Phi(z)$$

где EA и EB являются средними значениями эффективности для критериев A и B соответственно, σEA является стандартным отклонением значений эффективности A, n – размер выборки, а Φ – стандартная нормальная кумулятивная функция распределения. Как правило, с p-значениями менее 0,05 или 0,01 мы отклоняем H0 и заключаем, что A > B.

Основная угроза заключается в определении эффективности того, что мы предполагаем, что метод тестирования на основе спецификации наиболее эффективен, когда он способен идентифицировать широкий спектр ошибочных реализаций, другие могут видеть это по-другому. Например, неудача внедрения ставки могут быть использованы для включения относительной трудности поиска ошибок в оценке эффективности. Поскольку объем эмпирического исследования ограничен, реализацией двух систем, трудно утверждать, что наши результаты являются действительными.

Список литературы

[1] Интрилигейтор М., Макинтайр Р., Тейлор Л., Эмсден А. «Стратегия эффективного перехода и шоковые методы реформирования российской экономики» // В сб.: Шансы российской экономики / Под ред. Ю.М.Осипова, Е.С.Зотовой. М.: Изд-во ТЕИС, 1997. С. 168-195.

[2] Звягин Л.С. Имитационные методы моделирование экономических процессов в современном информационном обществе // Хроноэкономика. 2018. № 2 (10). С. 31-36.

[3] Звягин Л.С. Системный анализ в исследовании современного управления // Управленческие науки в современном мире. 2018. Т. 1. № 1. С. 443-446.

[4] Звягин Л.С. Возможности использования информационной среды "1С:Предприятие 8" для преподавания дисциплины "Имитационное моделирование" // Новые информационные технологии в образовании: Сб. науч. тр. 19-й международной научно-практической конференции. Под общей ред. Д.В. Чистова. 2019. С. 339-342.

[5] Орлов А.И. «О перестройке статистической науки и её применений» // Журнал "Вестник статистики", 1990. No.1, С.65-71.

[6] Орлов А.И., Федосеев В.Н. «Менеджмент в техносфере»: Учеб. пособие для студ. высш. учеб. заведений. М.: Издательский центр «Академия», 2003. 384 с.

[7] Ратникова Т.А. «Введение в эконометрический анализ панельных данных» // Экономический журнал ВШЭ. 2006. № 2, с. 267-316 с.