Санкт-Петербургский государственный университет
Кафедра системного программирования

Автоматический двунаправленный синтез объектов для символьного исполнения

Максим Алексеевич Паршин

Научный руководитель: к.ф.-м.н. Д.А. Мордвинов, доцент кафедры системного программирования

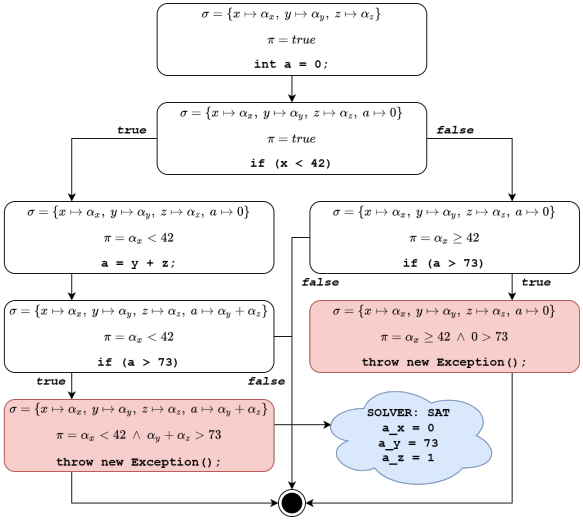
Санкт-Петербург
2022

Введение

- Символьное исполнение
  - Техника статического анализа кода
  - Тестирование «белого ящика»
  - Исполнение кода не на конкретных значениях входных данных, а на символьных переменных
- V# — символьная виртуальная машина для .NET Core и .NET
  - Открытый исходный код на F#
  - Генерация автотестов с наибольшим покрытием
  - *Поиск ошибок*

Пример (1)

```
1  void Foo(int x, int y, int z)
2  {
3          int a = 0;
4
5          if (x < 42)
6          {
7                  a = y + z;
8          }
9
10         if (a > 73)
11         {
12                 throw new Exception();
13         }
14 }
```



Пример (2)

```
1  public class StaticsForType<T> where T:class
2  {
3      private readonly List<T> myList = new
           List<T>();
4      private event Action? Changed;
5      ...
6      public void ForEachValue(Action action)
7      {
8        lock (myList)
9        {
10         Changed += action;
11       }
12       action();
13     }
14     ...
15 }
```

- Пример кода из библиотеки JetBrains.Lifetimes[a]
- V# при запуске на методе *ForEachValue* находит две ошибки
  - *NullReferenceException* на строке 12 при *action* == *null* – разумно
  - *ArgumentNullException* на строке 8 при *myList* == *null* – неразумно
- *myList* никак не может быть равен *null* при реальном использовании данного класса!
  - *Как «убедить» в этом символьную машину?*

[a] https://github.com/JetBrains/rd

Постановка задачи

Целью данной работы является реализация автоматического синтеза объектов на основе механизма двунаправленного символьного исполнения в символьной виртуальной машине V#

Задачи (практика):

- Провести обзор техник автоматического синтеза объектов в существующих инструментах тестирования

Задачи (ВКР):

- Разработать алгоритм синтеза объектов на основе механизма двунаправленного символьного исполнения
- Реализовать разработанный алгоритм в символьной виртуальной машине V#
- Провести эксперименты для определения эффективности работы реализованного алгоритма

Подходы к генерации объектов

- Синтезировать объект с помощью рефлексии, проставляя явным образом значения полей
  - Хорошо «дружит» с символьным исполнением
  - Никак не учитывает внутренние инварианты объекта — *List* с отрицательным *Count*
- Генерировать последовательность вызовов методов из публичного API, создающих данный объект
  - Готовый, корректный, красивый тест
  - Как генерировать последовательность?

Search-Based Software Testing

- Техники генерации тестов, основанные на эвристических алгоритмах оптимизации
  - Генетические алгоритмы
- Сначала генерируются тесты, а потом проверяется, насколько они «интересны»
- *EvoSuite*

Подход к реализации

- Методы, комбинирующие символьное исполнение и SBST
  - Символьное исполнение последовательностей методов (*Symstra, Evacon*)
  - Поиск последовательности методов по условию пути (*SUSHI*)
- Механизм двунаправленного символьного исполнения
  - Комбинация прямого символьного исполнения и обратного – движения от целей к точке входа
  - Позволяет учитывать семантику программы
  - Предоставляет более эффективный способ таргетированного исполнения

Результаты

В ходе данной работы были получены следующие результаты

- Проведён обзор методов синтеза объектов, используемых в различных инструментах генерации тестов

В рамках ВКР планируется

- Разработать алгоритм синтеза объектов на основе механизма двунаправленного символьного исполнения
- Реализовать разработанный алгоритм в символьной виртуальной машине V#
- Провести эксперименты для определения эффективности работы реализованного алгоритма