Рецензия на работу "Эффективная разрешающая процедура для задачи выполнимости в теории номинальных систем типов с вариантностью"

Косарев Дмитрий
"ИнтеллиДжей Лабс"

6 июня 2020 г.

В своей работе Милова Н.А. занимается поиском процедуры, которая будет использоваться при символьном исполнении .NET программ и будет проверять возможность символьных значений типизироваться определенным образом. Эта задача является критически важной для верификации программ на платформе .NET, так как программы, использующиеся в этой среде, полагаются на объектно-ориентированное программирование как на единственной способ абстракции данных и отсутствие поддержки преобразований типов в верификаторе вообще может свести на нет применимость верификации.

Сутью работы является подбор подходящего инструмента для решения задачи выполнимости подтипирования, вычленение запросов на подтипирование во время работы верификатора V# и передача этих запросов в инструмент.

В работе обозреваются имеющиеся результаты о проверке подтипируемости в номинальных системах типов с вариантностью. В общем случае задача не разрешима. Однако, можно выделить разрешимые фрагменты, хотя они не очень полезны на практике. В результате работы ожидается получение инструмента, который будет показывать хорошие результаты на практически встречающихся, неискусственных примерах.

В обзоре кратко описана система типов .NET, архитектура проекта V#, а также обозреваются четыре инструмента, которые в том числе могут использоваться как разрешающие для решаемой задачи, а именно: Prolog, реляционное программирование на miniKanren с помощью библиотеки OCanren, Mace4-Prover9, и два SMT-решателя: CVC4 и Vampire. Описание OCanren в принципе правильное, но не очень аккуратное: неподготовленный читатель может посчитать некоторые особенности реляционного программирования отличительными для OCanren.

В разделе 3 формально описываются правила типизации и подтипирования номинальной объектно-ориентированной системы типов .NET. В этом разделе меня несколько смутило несколько моментов.

- Перевод ground типов как "закрытых" (я бы предпочел замкнутых).
- Также Милова Н.А. ссылается на определение 3 непосредственно до введения определения 2, что несколько странно.

- Также фразой "За определением нерасширяемости автор отсылается к работе [12]." Милова Н.А. ссылается сама на себя в третьем лице, что тоже несколько необычно.

В разделе 4 приводятся формальные описания алгоритмов, которые преобразуют запросы на приведение типа в формат, понятный инструментам. В начале раздела заявляется, что будут рассматриваться формулы языка L_s, хотя формальное определение языка я не нашел.

Всего в работе присутствуют 9 листингов программ, происходящее в них более менее понятно без вчитывания в текст работы. С первым листингом у меня складывается ощущение, что в нём пропущено возвращение результата. Также при описании листинга перечисляются шаги, которые нумеруются арабскими цифрами, что вносит чехарду, так как номера строк тоже нумеруются арабскими цифрами. В общем и целом данный алгоритм достаточно прост, чтобы его нельзя было не понять.

Во втором листинге описывается алгоритм упрощения запросов на подтипирование. В цикле порождаются новые формулы ϕ_n, но для возвращения результата используется только одна. По тексту понятно, что имелось в виду, но в непосредственно алгоритм закралась опечатка.

В разделе 4.2 дело переходит непосредственно к кодированию формулы в формат, понятный внешнему инструменту. Здесь вводится некоторое количество аксиом, они не выглядят неправильными, но не очень понятно, как автор к ним пришел.

Все листинги сопровождаются по сути одним примером, разбитым на несколько подпримеров, который пошагово показывает как преобразуется исходная формула. К сожалению примеры ссылаются друг на друга и по мере введения новых алгоритмов исходный пример забывается. Также описания листингов попадает на разные страницы с листингами и не очень удобно следить за происходящим. Работа стала бы более читаемой, если бы листинги раскладывались по страницам не автоматически.

В разделе 5 проводится оценка инструментов на некотором наборе разрешимых и неразрешимых запросов на подтипирование. Я не очень знаком с особенностями CVC4 и Mace4-Prover9, но результаты, которые показали OCanren и Prolog соответствуют моим ожиданиям. Судя по описанию системы Vampire, которое было в обзоре, он ожидаемо показывает хорошие результаты, а именно чаще решает невыполнимые запросы. К сожалению, при замерах производительности было опущено описание тестового стенда и количество запусков, по которым бралось среднее время.

В общем и целом в работе нет подозрительных моментов, но она выглядит несколько незаконченной. Ниже мои вопросы и предложения:

1. Так как большинство запросов на подтипирование из бенчмарков были успешно решены, то хочется предположить, что все "полезные" программы на .NET попадают в некоторый разрешимый фрагмент, но в работе не было попыток сформулировать разрешимый фрагмент и никак не обсуждается разрешимый фрагмент из работы [Misonizhnik2019].

2. В работе создается некоторый тестовый набор, но никак не обговаривается наличие или отсутствие уже существующих наборов. Если никто из упомянутых в разделе 2.4 не смог формулировать подходящий набор бенчмарков, то создание соответствующего набора стоило бы вынести в результаты этой работы. А если у них уже есть свои наборы, то стоило бы объяснить, почему не были взяты они, а придуманы свои.

3. Было бы неплохо понять, какие именно особенности конкретных инструментов мешают или помогают эффективно решать поставленную задачу, так как это может поставить

интересные задачи по улучшению упомянутых инструментов. По схожей причине было бы интересно увидеть те два запроса, с которыми не справился ни один упомянутый инструмент.

4. Исследовались ли в процессе работы альтернативные способы преобразования запросов на подтипирование в формулы соответствующей теории?

5. В примере 1 мне не совсем понятно откуда взялся второй конъюнкт `Derived <: IComparable<T>`, может быть имелось в виду, что `F` это метод класса `Derived`?

```
interface IComparable<in T> {}
class Base {}
sealed class Derived : Base, IComparable<Derived> {}

void F<T>(Base arg1, Derived arg2)
{
  if (arg2 is IComparable<T> && arg1 is T)
  {
  ...
  }
}
```

6. Можно ли было воспользоваться для упрощения возможностями, которые предоставляет SMT-решатель, а не реализовывать вручную алгоритм упрощения на листингах 2-3?

По ходу чтения были выявлены следующие **неаккуратности**, не связанные с русским языком, которые частично упомянуты выше и не влияют на общую оценку работы (можно не зачитывать).

1. В листинге 4 комментарий к 11 строке должен на самом деле быть комментарием к 10й
2. Ссылки вперед на определения
3. Автор пишет о себе в третьем лице "За определением нерасширяемости автор отсылается к работе [12]."
4. Ссылка на язык L_s определение которого я не нашел.
5. В листинге 1 некоторые строчки потерялись, хотя на них есть ссылки в тексте.
6. В листинге 2 строятся на каждой итерации цикла формулы ϕ_n, а после окончания цикла используется только одна
7. В Листинге 4 комментарий к 11й строчке на самом деле должен быть комментарием к 10й (сбилась нумерация).

По ходу чтения были выявлены следующие **неаккуратности и опечатки, связанные с русским языком** и стилистическим оформлением, которые не влияют на общую оценку работы (можно не зачитывать).

1. Стр. титульная: старш**ы**й

2. Стр. 5 ”Разработка эффективной разрешающ**ий** процедуры”
3. Стр. 7 ”интерф**ес**ов”
4. Стр. 10 ”brunch”
5. Стр. 10 ”формулу логики первого порядка, которая описывает условия, выполненными по ветвям взятыми вдоль этого пути”
6. Стр. 10 ”Композициональность”
7. Стр. 11 ” подсистема, кодирующая и выполняющ**ий** запросы к SMT-решателю.”
8. Стр. 13 нет пробела ”(atoms).П”
9. Стр. 16 нет пробела ”VAMPIRE—”
10. Стр. 30 ”Функция кодировани**е**”
11. Стр. 40 нет пробела ”.NETиз”
12. В 10м пункте списка литературы появилась буква S, которая там не должна быть.

В итоге я считаю, что поставленные задачи были вполне выполнены, опечаток достаточно мало, работа не содержит подозрительных мест и Милова Н.А. заслуживает оценки ”*отлично*”.

программист
ООО «ИнтеллиДжей Лабс»
Д. С. Косарев