Справка по системе верификации Why

Версия 2.19

Введение

В настоящем документе описана основная информация по использованию системы верификации why [1], поддерживающей верификацию последовательных программ на языках Си и Java. Инструмент разработан в LRI (Laboratoire de Recherche en Informatique) [2] и распространяется по лицензии GPL v2.

Общая информация

Система верификации why реализует верификацию последовательных алгоритмов по методам Флойда/Хоара для программ на языках программирования Си и Java.

На вход поступает исходный код верифицируемой программы, аннотированный формальной спецификаций в виде комментариев в специальной нотации. Для программ на языке Си используется нотация ACSL [3], а для программ на языке Java — JML [4]. Этот код вместе с аннотациями трансформируется в код на внутреннем языке системы why. Инструмент выполняющий трансформацию для языка Си называется Caduceus, а для языка Java — Krakatoa.

Затем генератор условий верификации[1] Verification Condition Generator (VCG) создает набор условий верификации (утверждений в логике первого порядка), доказательство корректности которых влечет полную корректность исходной программы относительно спецификации, заданной в виде предусловий и постусловий отдельных функций. Условия верификации изначально формулируются на внутреннем языке why, но в дальнейшем могут быть переведены на один из множества входных языков систем автоматического или интерактивного доказательства теорем. В частности, в число поддерживаемых инструментов входят: PVS, Coq, Isabelle/HOL, HOL 4, HOL Light, Mizar, Simplify, Alt-Ergo, Yices, Z3, CVC3.

В рамках настоящего документа мы будем рассматривать единственный вариант работы системы верификации why, начинающийся с кода программы на языке Си и аннотациями на ACSL, и завершающегося генерацией набора утверждений для доказательства в системе верификации PVS.

---

1 Условия верификации в терминологии системы why включают в себя условия верификации, корректности и завершимости в терминах, использовавшихся в лекциях по методам Флойда [5].

Интерфейс командной строки

Рассмотрим как осуществляется данный вариант работы системы why. Предположим, что у нас есть файл с кодом на языке Си и аннотациями на ACSL и что он называется *myfile.c*. Тогда для трансформации кода в код на внутреннем языке системы why необходимо выполнить команду:

```
> caduceus myfile.c
```

Если в исходном файле не обнаружено ошибок, то в результате генерации появляются следующие файлы:

- *myfile.makefile* — makefile для дальнейшей генерации условий верификации;
- *why/myfile.why* — результат трансляции исходной программы в программу на внутреннем языке инструмента why;
- *why/myfile_spec.why* — результат трансляции вспомогательных предикатов и функций, заданных на ACSL, во внутренний язык инструмента why;
- *myfile.loc* — файл, по которому восстанавливается соответствие между сгенерированными выражениями и их прообразами в исходной программе.

Следующим шагом является генерация условий верификации в виде лемм на языке PVS Definition Language [6]. Для этого необходимо выполнить команду:

```
> make -f myfile.makefile pvs
```

В результате будет создана директория pvs, содержащая следующие файлы:

- *myfile_why.pvs* — теория PVS *myfile_why*, в которой в виде лемм содержатся все условия верификации для функций языка Си, определенных в *myfile.c* (результат трансляции *myfile.why* в PVS);
- *myfile_spec_why.pvs* — теория PVS *myfile_why_spec*, в которой содержатся определения вспомогательных предикатов и функций, заданных на ACSL (результат трансляции *myfile_spec.why* в PVS);
- *caduceus_why.pvs* — теория PVS, в которой определены базовые предикаты и функции, используемые для выражения встроенных операторов языка Си.

Завершающим шагом является доказательство всех лемм из файла *myfile_why.pvs* в системе автоматического доказательства теорем PVS. В случае успешного выполнения этого шага будет доказана полная корректность функций, описанных на языке Си в файле *myfile.c*, относительно спецификации к этим функциям на ACSL в предположении, что целевая функция выполняется последовательно без прерываний сигналами и без интерференции с другими потоками управления.

[5] Лекции по аналитической верификации
[6] PVS Definition Language, http://pvs.csl.sri.com/doc/pvs-language-reference.pdf