УДК 004.4'414

ФОРМАЛЬНАЯ ВЕРИФИКАЦИЯ РЕФАКТОРИНГОВ ЯЗЫКА C

Жирков И.О. (Университет ИТМО)
Научный руководитель – доцент, к.т.н Дергачев А.М. (Университет ИТМО)

Введение. При создании сложных программных систем разработчики часто изменяют ее структуру, сохраняя ее поведение. Эти изменения – рефакторинги – бывают глобальными, как изменение набора аргументов функции, и локальные, как создание новой локальной переменной. Трудоемкость и частота рефакторингов мотивируют их автоматизацию с помощью преобразователей кода, таких, как clang-rename.

Существующие преобразователи программ ненадежны [1,2], а ошибки, внесенные ими в программы, могут оставаться незамеченными. Чтобы повысить доверие к преобразователям программ на индустриальных языках программирования используют методы валидации основанные на тестировании[1,2]. Однако для верификации преобразователей программ тестирование неэффективно, в особенности для языков с недетерминированной семантикой, таких, как C. Недетерминизм в исходной программе необходимо в полной мере сохранить в преобразованной программе.

Другой способ повысить доверие к преобразователю – его формальная верификация с написанием машинно-проверяемого доказательства корректности. Это требует полного формального описания семантики языка программирования. Для языка C такое описание создано и апробировано при верификации компилятора CompCert. Это описание также использовалось для верификации переименования глобальных переменных в C [3].

Основная часть. Мы следуем подходу предложенному в [3] и переиспользуем формальную семантику языка C из компилятора CompCert для описания преобразований и построения машинно-проверяемых доказательств их корректности на языке Coq. Кроме того, мы автоматически экстрагируем из описания трансформаций код на языке Ocaml, преобразующий программы на C.

Мы используем метод бисимуляции для формальных систем переходов чтобы доказать взаимно однозначное соответствие между наблюдаемыми поведениями двух программ: любой корректной программы на C и ее преобразованной версией

Мы описали и верифицировали преобразователи для добавления и удаления локальных переменных. В отличие от переименования глобальных переменных, они меняют структуру дерева абстрактного синтаксиса и влияют на память абстрактного вычислителя программы. Для иллюстрации сложности доказательства корректности проследим за соответствующими состояниями памяти абстрактного вычислителя C в исходной программе и в программе с удаленной переменной X в функции F. Исходные состояния памяти этих программ совпадают, но при каждом запуске F выделение памяти под X будет происходить только в исходной программе, поэтому состояния программ будут расходиться с каждым запуском функции F.

Комбинируя доказано корректные преобразования можно реализовывать более сложные преобразователи, для которых доказательство корректности получается композицией.

Выводы. Была показана практическая осуществимость формальной верификации программ для рефакторингов на индустриальных языках, влияющих на структуру дерева абстрактного синтаксиса и состояние абстрактного вычислителя языка. Реализованные преобразователи программ можно включать напрямую в IDE, использовать для разработки программ с повышенными требованиями по надежности.

Список использованных источников:

1. Mongiovi, M., Safira: A Tool for Evaluating Behavior Preservation. // Proceedings of the ACM International Conference Companion on Object Oriented Programming Systems

Languages and Applications Companion, OOPSLA ’11, ACM. Нью-Йорк, США – 2011. – С. 213–214
2. Eilertsen, A.M., Murphy, G.C., "The usability (or not) of refactoring tools." // 2021 IEEE international conference on software analysis, evolution and reengineering (SANER). IEEE. – 2021. – С. 237-248
3. Cohen, J., "Renaming global variables in c mechanically proved correct." // *arXiv preprint arXiv:1607.02226* – 2016.

Жирков И.О. (автор) Подпись

Дергачев А.М. (научный руководитель) Подпись