ЛИТЕРАТУРА

1. *Егорушкин О. И., Колбасина И. В., Сафонов К. В.* О совместности систем символьных полиномиальных уравнений и их приложении // Прикладная дискретная математика. Приложение. 2016. № 9. С. 119–121.
2. *Egorushkin O. I., Kolbasina I. V., and Safonov K. V.* On solvability of systems of symbolic polynomial equations // Журн. СФУ. Сер. Матем. и физ. 2016. Т. 9. Вып. 2. С. 166–172.
3. *Глушков В. М., Цейтлин Г. Е., Ющенко Е. Л.* Алгебра. Языки. Программирование. Киев: Наукова думка, 1973.
4. *Salomaa A. and Soitolla M.* Automata-Theoretic Aspects of Formal Power Series. N.Y.: Springer Verlag, 1978.
5. *Семёнов А. Л.* Алгоритмические проблемы для степенных рядов и контекстно-свободных грамматик // Доклады АН СССР. 1973. № 212. С. 50–52.
6. *Сафонов К. В., Егорушкин О. И.* О синтаксическом анализе и проблеме В. М. Глушкова распознавания контекстно-свободных языков Хомского // Вестник Томского государственного университета. 2006. Приложение № 17. С. 63–67.
7. *Сафонов К. В.* Об условиях алгебраичности и рациональности суммы степенного ряда // Матем. заметки. 1987. Т. 41. Вып. 3. С. 325–332.
8. *Safonov K. V.* On power series of algebraic and rational functions in C^n // J. Math. Analysis Appl. 2000. V. 243. P. 261–277.

УДК 519.714 DOI 10.17223/2226308X/10/59

ЗАВЕРШЕНИЕ ЭСКИЗОВ ПРЕДИКАТНЫХ ПРОГРАММ МЕТОДОМ СИНТЕЗА ЧЕРЕЗ КОНТРПРИМЕРЫ[1]

М. С. Чушкин

Программа на языке P представляет собой набор определений предикатов. Для языка P разработана операционная семантика. На базе операционной семантики определена формула тотальной корректности предиката относительно его спецификации. Для незаконченной программы на языке P ставится задача её завершения до корректной относительно спецификации. Метод синтеза выражений на основе контрпримеров успешно адаптирован для этой задачи.

Ключевые слова: *предикатное программирование, формальная операционная семантика, программный синтез, синтез на основе контрпримеров, дедуктивная верификация.*

1. Система предикатного программирования

Программа на языке P_0 определяется следующей конструкцией:

$$<\textit{имя предиката}>\ (<\textit{аргументы}>:<\textit{результаты}>)\{<\textit{оператор}>\},$$

аргументы и результаты — непересекающиеся наборы имён переменных. Набор аргументов может быть пустым.

Операторами языка P_0 являются: $B(x:z);C(z:y)$ — последовательный оператор; $B(x:y)||C(x:z)$ — параллельный оператор и if (e) $B(x:y)$ else $C(x:y)$ — условный оператор. Здесь B и C — операторы, e — логическое выражение [1].

Операционной семантикой программы $H(x:y)$ назовём формулу $R(H)(x,y)$, смысл которой на естественном языке звучит следующим образом: «для значения на-

[1]Работа поддержана грантом РФФИ, проект № 16-01-00498.

бора x исполнение программы H всегда завершается и существует исполнение, при котором результатом вычисления является значение набора y» [2].

Тотальная корректность программы относительно *спецификации* в виде *предусловия* $P(x)$ и *постусловия* $Q(x, y)$ определяется формулой

$$\forall x\ P(x) \Rightarrow [\forall y\ R(H)(x, y) \Rightarrow Q(x, y)]\ \&\ \exists y\ R(H)(x, y).$$

В системе предикатного программирования реализован генератор формул корректности. Используя систему правил вывода, генератор декомпозирует исходное условие тотальной корректности на более простые формулы корректности, которые проходят проверку в SMT-решателе CVC3 и в системе PVS [3].

2. Завершение эскизов предикатных программ

Синтаксис выражений языка P расширяется *произвольным выражением*, для обозначения которого используется слово «??». Программа, содержащая в себе произвольные выражения, называется эскизом [4]. Примеры программ на языке P, формул корректности и эскизов можно найти по ссылке [5].

Пусть дан некоторый эскиз $Sketch(x : y)$, произвольные выражения которого заменены на вызовы предикатов f_i, тела которых неизвестны. Этот эскиз передаётся генератору формул корректности, который строит множество формул корректности.

Для некоторой пары значений (x', y') эскиза $Sketch(x : y)$ строится *конкретный пример* [6] каждого произвольного выражения по следующему алгоритму. Значения (x', y') подставляются в множество формул корректности. Формулы в полученном множестве зависят лишь от значений произвольных выражений, которые можно найти, передав формулы в SMT-решатель. Значения выражений объединяются с парой (x', y'), образуя конкретные примеры произвольных выражений.

Спецификация $[P(x), Q(x, y)]$ эскиза $Sketch(x : y)$ приводится к виду $P(x) \wedge Q(x, y)$ и передаётся SMT-решателю; он находит значения переменных (x_0, y_0), на которых эта формула истинна. Для каждого произвольного выражения строится множество конкретных примеров, начальным элементом которого является конкретный пример для пары значений (x_0, y_0).

Для каждого множества конкретных примеров строится удовлетворяющее ему выражение по следующему алгоритму. Выражения языка P сортируются в порядке возрастания их длины и поочередно проходят проверку на удовлетворение конкретным примерам. Если выражение удовлетворяет всем конкретным примерам заданного множества, то оно считается итоговым.

Построенные таким образом выражения подставляются в множество условий корректности эскиза $Sketch(x : y)$. Полученное множество формул передается SMT-решателю. Если решатель подтверждает истинность формул, то процесс синтеза завершается.

Если SMT-решатель смог найти контрпример (x'', y''), на котором не все формулы корректности истинны, то для этого контрпримера строятся конкретные примеры произвольных выражений, расширяющие существующие множества конкретных примеров. Процесс повторяется для обновлённых множеств.

3. Оценка сложности метода

Сложность алгоритма зависит от количества формул в множестве условий корректности, от количества кандидатов на роль выражения, удовлетворяющего множеству конкретных примеров, и от возможностей решателя.

Зависимость количества формул корректности от объёма программы экспоненциальная. В среднем для программы объёмом в 10 операторов сгенерируется 30 формул корректности.

Зависимость числа возможных кандидатов на роль выражения, удовлетворяющего множеству конкретных примеров, от длины выражения экспоненциальная. В среднем выражение длиной 10 лексем имеет 100 тыс. кандидатов.

В данной реализации используется решатель CVC3. Он корректно завершает проверку выполнимости лишь для простых формул, не содержащих кванторов и сложной арифметики. Таким образом, описанный выше алгоритм целесообразно применять для коротких простых программ.

ЛИТЕРАТУРА

1. *Карнаухов Н. С., Першин Д. Ю., Шелехов В. И.* Язык предикатного программирования Р. Препринт № 153. Новосибирск: ИСИ СО РАН, 2010. 42 с.
2. *Шелехов В. И.* Семантика языка предикатного программирования // ЗОНТ-15. Новосибирск, 2015. С. 15.
3. *Чушкин М. С.* Методы дедуктивной верификации предикатных программ // Тр. 2-й Междунар. конф. «Инструменты и методы анализа программ». Кострома, 2014. С. 205–214.
4. *Solar-Lezama A., Tancau L., Bodik R., et al.* Combinatorial sketching for finite programs // Proc. ASPLOS XII. N. Y.: ACM, 2006. P. 404–415.
5. `https://gist.github.com/mchushkin/8e1a6a28fd6d342623cd97cd5fa395ac`
6. *Udupa A., Raghavan A., Deshmukh J. V., et al.* TRANSIT: specifying protocols with concolic snippets // Proc. PLDI'13. N. Y.: ACM, 2013. P. 287–296.