УДК 510.52, 519.681.2

PSPACE-полнота задачи о корректности программы с циклом

Секорин В.С.

Тверской государственный университет

Корректность — одно из важнейших свойств программного обеспечения. Одним из способов его установления является формальная верификация, основы которой были заложены в работах Хоара [1]. Для формальной верификации используют те или иные способы описания требований к программному обеспечению, которые затем подлежат проверке. Одним из таких инструментов являются тройки Хоара, описывающие требования ко входным данным и требования к результату выполнения программы. Например, одна из возможных формализаций рассматривается в [2]. Эти методы находят широкое применение как для теоретических исследований [3], так и для практической реализации [4]. Методы, основанные на тройка Хоара, применяются даже в таких сложных предметных областях как графы [5].

К сожалению, в общем случае (например, для программ, которые могут обрабатывать натуральные числа) задача проверки корректности тройки алгоритмически неразрешима [6]. Поэтому важной является проблема нахождения частных случаев, в которых вопрос о корректности может быть решен алгоритмически. При этом практически важно, чтобы алгоритмическое решение было как можно более эффективным, то есть могло быть выполнено с использованием разумного количества ресурсов (прежде всего — времени и памяти). В некоторых случаях проблема верификации может оказаться разрешимой даже за полиномиальное время [7].

Как доказано в [8] даже проверка истинности формулы первого порядка на конечной предметной области является $PSPACE$-полной задачей, поэтому любая задача верификации с использованием логики первого порядка имеет не меньшую сложность. В нашей работе мы попытаемся ограничиться еще более частным случаем, взяв в качестве предметной области только два логических значения $true$ и $false$, а в качестве формул тройки Хоара будем брать только булевы формулы. Основной наш результат заключается в том, что даже при таких ограничениях задача верификации остается $PSPACE$-полной.

Итак, предметная область будет состоять из двух логических значений $true$ и $false$. Булевы формулы мы определяем как обычно. По теореме Поста система булевых связок, состоящая из отрицания и импликации, является полной, то есть с помощью них можно выразить любую булеву функцию. Поэтому в программах мы будем использовать только эти булевы связки.

Пусть зафиксировано некоторое множество переменных $\mathcal{V}$.

Определение 1 (Выражение)**.** *Определим в ы р а ж е н и е как*

- *константы* $true$, $false$ *являются выражением;*
- *переменная* $x \in \mathcal{V}$, *то* x *является выражением;*
- **not** u, $u \to v$, *где* u, v *— выражения, являются выражениями.*

Определение 2 (Программа)**.** *Будем использовать язык программирования, предметной областью которого являются два значения: true и false. Тогда п р о г р а м м о й является*

1) оператор присваивания x **is** u, *где* x *— переменная,* u *— выражение;*

2) оператор полного ветвления: **if** *и* **then** Π_1 **else** Π_2 **end if**, *где и — выражение,* Π_1 *и* Π_2 *— программы;*

3) оператор цикла: **while** *и* **do** Π **end while**, *где и — выражение,* Π *— программа;*

4) последовательность программ: $\Pi_1\Pi_2$, *где* Π_1 *и* Π_2 *— программы.*

Определение 3 (Состояние программы)**.** *Состояние программы (состояние переменных программы) — функция, которая каждой переменной ставит в соответствие значение* $true$ *или* $false$.

Определение 4 (Корректность тройки)**.** *Пусть* Π *— программа,* φ *и* ψ *— булевы формулы. Тогда тройка* (φ, Π, ψ) *называется корректной, если из того, что на состоянии* σ *была истинна формула* φ*, следует, что программа* Π *останавливается на состоянии* σ *и на полученном состоянии будет истина формула* ψ*. Формула* φ *называется предусловием, а* ψ *— постусловием.*

Мы доказываем, что установление корректности тройки является $PSPACE$-полной задачей.

Теорема 1. *Задача установления, является ли тройка* (φ, Π, ψ) *корректной, является* $PSPACE$*-полной, если программа* Π *содержит как минимум один оператор цикла.*

Доказательство. Покажем, что задача принадлежит $PSPACE$. Пусть нам дано слово W. Сначала определим, является ли слово тройкой вида (φ, Π, ψ), если нет, то вернуть $false$.

После этого перебираем все состояния σ переменных программы Π, для каждого из них проверяем предусловие φ. Если φ выполнено, то запускаем программу Π на таком состоянии σ. Так как количество переменных Π конечно (например, равняется n), то программа может иметь 2^n различных состояний. Тогда если программа выполнила 2^n итераций цикла и не завершилась, то она зациклилась, в этом случае возвращаем $false$. В противном случае программа остановилась в некотором состоянии τ, для которого мы проверяем значение формулы ψ. Если ψ ложно, возвращаем $false$. Если после перебора всех состояний не было возвращено $false$, то возвращаем $true$.

Память, которая требуется для перечисленных действий — n для хранения значений переменных во время выполнения программы и n для подсчета количества выполненных итераций.

Теперь покажем, что задача определения корректности является $PSPACE$-трудной. Для этого возьмем произвольную проблему A из класса $PSPACE$ и сведем к нашей проблеме. Иными словами для каждого слова x мы должны построить тройку (φ, Π, ψ), которая будет корректной тогда, и только тогда, когда $x \in A$.

Так как A — проблема из класса $PSPACE$, то существует детерминированная машина Тьюринга $\mathfrak{M} = (Q, \Sigma, P, q_0)$, где Q — множество состояний, Σ — алфавит ленты, P — функция перехода, q_0 — начальное состояние, которая распознает проблему A в памяти $p(|x|)$ для некоторого полинома p, где x — входное слово, а $|x|$ — его длина. Как известно, при вычислении на машине Тьюринга можно обойтись двухсимвольным алфавитом, поэтому мы будем считать $\Sigma = \{\wedge, |\}$, а в конце вычисления в начальной ячейке будет написано $|$, если исходное слово принадлежит A, и $\wedge$ иначе. Будем считать, что машина $\mathfrak{M}$ всегда останавливается и имеет единственное заключительное состояние q_f с наибольшим номером.

Покажем, как для любого входного слова x построить тройку (φ, Π, ψ), которая будет корректна тогда и только тогда, когда результатом работы машины Тьюринга является $|$, то есть исходное слово принадлежит A.

Будем использовать переменные: $h_i,\ q'_j,\ a_i$, где $i \in [-p(|x|); p(|x|)]$, $j \in [0; f]$, для которых выполнены следующие условия:

1. h_i будет истинна на некоторой итерации t цикла программы П тогда и только тогда, когда головка машины Тьюринга находится в i-ой ячейке на шаге с номером t (например, если h_2 истинна, то головка находится во второй ячейке, а все остальные $h_i, i \neq 2$ ложны),

2. q'_j будет истинна на некоторой итерации t цикла программы П тогда и только тогда, когда машина Тьюринга находится в состоянии q_j на шаге с номером t,

3. a_i будет истинна на некоторой итерации t цикла программы П тогда и только тогда, когда в i-ой ячейке ленты машины Тьюринга на шаге с номером t находится $|$.

Для каждой команды машины Тьюринга $\mathfrak{M}$ определим действия, которые будет выполнять программа. Введем еще одну переменную cc, отвечающую за то, что никакая команда еще не была промоделирована на очередной итерации цикла.

Каждой команде машины Тьюринга будет соответствовать фрагмент программы $C_{j,a}$, где j — номер состояния, в котором находимся до выполнения команды, a — значение символа.

Команде вида $q_j,\ a \to q_k,\ b,\ 0$ будет соответствовать $C_{j,a}$:

a_i **is** b'
q'_j **is not** q'_j
q'_k **is not** q'_k

Здесь b' — это константа соответствующая символу b.

Команде вида $q_j,\ a \to q_k,\ b,\ +1$ будет соответствовать $C_{j,a}$:

a_i **is** b'
h_i **is not** h_i
h_{i+1} **is not** h_{i+1}
q'_j **is not** q'_j
q'_k **is not** q'_k

Команде вида $q_j,\ a \to q_k,\ b,\ -1$ будет соответствовать $C_{j,a}$:

a_i **is** b'
h_i **is not** h_i
h_{i-1} **is not** h_{i-1}
q'_j **is not** q'_j
q'_k **is not** q'_k

Теперь определим программу П:

```
while not q'_f do
    cc is false
    ...
    if q'_j then
        if not cc then
            ...
            if h_i then
                if not cc then
                    if a_i then
                        C_{j,|}
                    else
```

```
                    C_{j,∧}
                end if
                cc is not cc
            else
                cc is cc
            end if
        else
            cc is cc
        end if
        ...
    else
        cc is cc
    end if
    ...
end while
```

В теле цикла мы перебираем все возможные значения $i \in [-p(|x|); p(|x|)]$, $j \in [0; f]$. Таким образом длина программы будет $O(p(|x|))$.

В условии цикла проверяется, что машина Тьюринга не перешла в заключительное состояние. Дальше последовательно проверяется: состояние и положение головки машины Тьюринга, так же после каждой проверки проверяется, что никакое действие еще не выполнялось. Затем в зависимости от значения переменной a_i выполняем один из блоков $C_{j,a}$.

Теперь построим предусловие φ. Оно будет говорить следующее. Во-первых, что машина Тьюринга находится в состоянии q_0:

$$q'_0 \wedge \neg q'_1 \wedge \cdots \wedge \neg q'_f.$$

Во-вторых, что головка находится в нулевой ячейке:

$$\neg h_{-p(|x|)} \wedge \ldots \wedge \neg h_{-1} \wedge h_0 \wedge \neg h_1 \wedge \cdots \wedge \neg h_{p(|x|)}.$$

В-третьих, на ленте написано слово x длины m:

$$\bigwedge_{i<0} \neg a_i \wedge \bigwedge_{x_i=|} a_i \wedge \bigwedge_{x_i=\wedge} \neg a_i \wedge \bigwedge_{i \geq m} \neg a_i,$$

где x_i — i-ый символ слова x. Предусловие φ является конъюнкцией трех указанных формул.

В качестве постусловия ψ берём a_0.

Продемонстрируем, что слово x принимается машиной Тьюринга $\mathfrak{M}$ тогда и только тогда, когда построенная тройка (φ, Π, ψ) является корректной.

Очевидно, что предусловие φ будет истино только на состоянии σ, удовлетворяющим условиям 1, 2, 3 в начальный момент времени. Следовательно, корректность или некорректность построенной тройки будет определяться только результатом работы программы Π на этом состоянии σ.

Пусть в состоянии σ выполнено φ. Индукцией по t легко доказать, что будут выполнены условия 1, 2, 3. Если $x \in A$, то машина $\mathfrak{M}$ принимает x, следовательно, выполнив T команд, она переходит в состояние q_f, когда в нулевой ячейке записано $|$. Значит после T итераций

цикла переменная q'_f станет истинной, поэтому цикл завершит свою работу и переменная a_0 примет значение $true$, т.е. постусловие ψ будет истино.

Если x не принадлежит A, то выполнив T команд, она перейдет в состояние q_f, и в нулевой ячейке будет $\wedge$. Значит после T итераций цикла переменная q'_f станет истинной, поэтому цикл завершит свою работу и переменная a_0 примет значение $false$, т.е. постусловие ψ будет ложно.

Таким образом мы продемонстрировали, что $x \in A$ тогда и только тогда, когда построенная тройка (φ, Π, ψ) является корректной. Поскольку в качестве A мы выбрали произвольную проблему из класса $PSPACE$, то тем самым мы доказали $PSPACE$-трудность проблемы определения корректности. □

Заключение Мы продемонстрировали, что задача верификации циклических логических программ является $PSPACE$-полной при наличии хотя бы одного оператора цикла. Более того можно отметить, что в самой программе из булевых связок используется только отрицание, поэтому реально $PSPACE$-полнота проблемы верификации доказана даже для этого частного случая.

Интерес представляет построение формальных правил верификации для программ, рассмотренного вида. Второй интересующий нас вопрос: до каких пределов следует ограничить рассмотренный язык программирования, чтобы проблема верификации стала разрешимой за полиномиальное время.

Литература

1. *Hoare, C.A.R.* An Axiomatic Basis for Computer Programming. / C.A.R Hoare // Communications of the ACM – 1969, – V.12 - N.10 - October, – C.576–583.

2. *Mili, Ali* Software Testing: Concepts and Operations./ A. Mili, F. Tchier – Wiley, 2015 – 400 c.

3. *Hahnle, Reiner* A Hoare-Style Calculus with Explicit State Updates. / R. Hahnle, R. Bubel // Formal Methods in Computer Science Education(FORMED) – 2008, – C. 49-60.

4. *Sznuk, Tadeusz* Tool Support for Teaching Hoare Logic. / T. Sznuk, A. Schubert // Software Engineering and Formal Methods, Springer – 2014, – C. 332-346.

5. *Poskitt, Christopher M.* Hoare-Style Verification of Graph Programs. / C. M. Poskitt, D. Plump // Fundamenta Informaticae – 2012, – C. 135–175.

6. *Булос, Дж.* Вычислимость и логика. / Дж. Булос, Р. Джеффри – Москва: Мир, 1994 – 396 с.

7. *Yoo, Tae-Sic* Polynomial-time verification of diagnosability of partiallyobserved discrete event systems. / T.S. Yoo, S. Lafortune // IEEE Trans. Autom. Control – 2002, – C. 1491–1495.

8. *Vardi, Moshe Y.* The complexity of relational query languages. / M. Y. Vardi // Proceedings of the 14th Annual ACM Symposium on the Theory of Computing (STOC'82) – 1982, – May, – C. 137–146.

Информация об авторах

ФИО: Секорин Всеслав Станиславович
Должность: студент магистратуры
Место работы: Тверской государственный университет
E-mail: vssekorin@gmail.com
Контактный телефон: +79157063874
Научный руководитель: Дудаков Сергей Михайлович