код встраивается заново. В дальнейшем экспериментально будет проведён сравнительный анализ изложенных способов генерации.

Результатом работы является «состыковка» компилятора языка ЛЯПАС-Т, написанного на языке C++, с функциями работы с длинной арифметикой, написанными на языке Ассемблер. Таким образом, стала возможной работа с длинной арифметикой стандартными средствами языка ЛЯПАС-Т.

ЛИТЕРАТУРА

1. *Агибалов Г. П., Липский В. Б., Панкратова И. А.* О криптографическом расширении и его реализации для Русского языка программирования // Прикладная дискретная математика. 2013. №3(21). С. 93–104.
2. *Кнут Д.* Искусство программирования. М.: Вильямс, 2001.

УДК 519.681.2

## РАЗРАБОТКА АВТОМАТИЗИРОВАННОГО СРЕДСТВА ДЛЯ ДОКАЗАТЕЛЬСТВА СВОЙСТВ ПРОГРАММ

А. О. Жуковская, Д. А. Стефанцов

Рассматривается метод статической верификации программ, основанный на автоматизированном доказательстве теорем. Моделью программы выбраны функции на парах списков натуральных чисел, упрощённой моделью — функции на натуральных числах. Исследуется свойство безопасности: программа может выдать секретное сведение, только если на вход был подан ключ. С помощью автоматизированного средства доказательства теорем Coq строятся доказательства для примеров функций, выводится общая схема построения доказательств, с помощью которой создаётся тактика Coq. В заключение приводятся идеи дальнейших исследований.

**Ключевые слова:** *верификация программ, автоматизированное доказательство, Coq.*

Для повышения надёжности компьютерных систем актуальна задача автоматической верификации программ. *Верификация программы* — доказательство её соответствия заданным формальным требованиям [1]. Существуют два основных метода верификации: *динамический*, при котором программа исследуется в процессе своей работы, и *статический*, при котором исследование происходит без запуска программы. В данной работе рассматривается статическая верификация. Под *программой* понимается синтаксически правильная последовательность команд на языке программирования, реализующая некоторую функцию. Для решения задачи верификации необходимы методы, позволяющие определить, обладает ли данная программа некоторым формально заданным свойством.

Имеет смысл рассматривать нетривиальное свойство, то есть такое, для которого существуют программы, обладающие им, но не все программы обладают им. В общем случае эта задача неразрешима в силу теоремы Райса — Успенского [2]. Поэтому алгоритм, устанавливающий, обладает ли программа заданным нетривиальным свойством, может не всегда выдавать результат или совершать ошибки первого (второго) рода.

Программа преобразует состояние потоков входа и выхода, состояние потока кодируется последовательностью чисел, поэтому за модель программы примем функцию на произведении списков $f : \mathbb{N}^* \times \mathbb{N}^* \to \mathbb{N}^* \times \mathbb{N}^*$. Для упрощения технических деталей

и в силу существования биективного соответствия между $\mathbb{N}^* \times \mathbb{N}^*$ и $\mathbb{N}$ можно принять упрощённую модель программы $f : \mathbb{N} \to \mathbb{N}$. Будем считать $0 \in \mathbb{N}$ специальным значением.

В работе исследовано следующее важное свойство безопасности: программа $f(x)$ может выдать секретное сведение $s$, только если на вход был подан $x = k$, где $k$ — ключ. В силу принятой модели $s, k \in \mathbb{N}$.

Приведём примеры простейших программ (функций), обладающих данным свойством:

$$f_1(x) = \begin{cases} s, & \text{если } x = k, \\ 0 & \text{иначе;} \end{cases} \qquad f_2(x) = x + (s - k); \qquad f_3(x) = \begin{cases} 0, & \text{если } x = s, \\ x & \text{иначе.} \end{cases}$$

Для доказательства того, что функции обладают заданным свойством, использована система автоматизированного доказательства теорем `Coq` [3]. Для каждой из возможных функций $f(x)$ построено доказательство соответствующей свойству безопасности теоремы:

$$\forall x, s, k \in \mathbb{N} \setminus \{0\}\ (f(x) = s \Rightarrow x = k).$$

Доказательства для разных функций схожи и строятся по следующей схеме:

1) рассматриваются все $x$, $0 < x < k$, для них выполнение условия $f(x) \neq s$ проверяется непосредственно;
2) значение $f(k)$ допускается любым;
3) доказывается, что $f(x)$ при $x > k$ не может принять значения $s$.

Шаги 1 и 2 выполняются одинаково для всех функций; в шаге 3 возможны незначительные различия.

На основе этого написана новая тактика в системе `Coq` для доказательства выбранного свойства, автоматически выполняющая шаги 1 и 2 и для некоторых функций шаг 3. Тактика применима к нерекурсивным функциям.

Для облегчения трансляции с языка программирования удобнее использовать неупрощённую модель представления программ в виде функций. Для функций на списках подходит такой же метод с модификацией способа реализации шагов. Рассмотрены аналогичные приведённым выше функции на списках, для них также построены доказательства соответствующей теоремы.

Наряду с рассмотрением вышеизложенного метода сделано предположение, что поставленную задачу можно решить иным способом. Для формализации программ существует система $\lambda$-исчисления [4]. Каждую функцию можно представить в виде терма $\lambda$-исчисления, которому можно присвоить тип по определённым правилам. Известен изоморфизм между типизированным $\lambda$-исчислением и интуиционистской логикой высказываний, являющийся частным случаем соответствия Карри — Говарда [5]. Таким образом, если сформулировать некое свойство программ в виде импликации в интуиционистской логике и представить исследуемую программу в виде терма типизированного $\lambda$-исчисления, можно проверить, соответствует ли полученный терм доказательству требуемого свойства, и в случае соответствия утверждать, что программа обладает данным свойством. Реализация этого метода — возможная тема для дальнейших исследований.

## ЛИТЕРАТУРА

1. *Hoare T.* The verifying compiler: a grand challenge for computing research // J. ACM. 2003. V. 50. No. 1. P. 63–69.

2. *Верещагин Н. К., Шень А.* Лекции по математической логике и теории алгоритмов. Ч. 3. Вычислимые функции. 4-е изд., испр. М.: МЦНМО, 2012.
3. `http://coq.inria.fr/`
4. *Барендрегт Х.* Ламбда-исчисление. Его синтаксис и семантика. М.: Мир, 1985.
5. *Пирс Б.* Типы в языках программирования. М.: Лямбда пресс & Добросвет, 2011.