Секция 10

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

УДК 004.43, 004.56

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ОПЕРАЦИЙ НАД БОЛЬШИМИ ЧИСЛАМИ В ЯЗЫКЕ ЛЯПАС-Т

А. С. Грибанов, В. А. Сибирякова

Представлена информация о программной реализации в виде функций на языке Ассемблер операций языка ЛЯПАС-Т над большими числами. Эти функции вставляются компилятором в загружаемый модуль (файл с компилируемой программой). Реализованы два способа встраивания — через вызов подпрограммы и в виде макроса.

Ключевые слова: *ЛЯПАС-Т, длинная арифметика.*

В настоящее время в программировании большое внимание уделяется быстродействию программ, особенно в области задач компьютерной криптографии. Все в этой области стремятся к сокращению времени работы программ, ищут для этого альтернативные методы программирования, рассматривают возможности реализации программ на разных языках. Но, несмотря на все усовершенствования высокоуровневых языков, более быстрыми являются программы на языках, которые «ближе» к машинному языку, т. е. являются низкоуровневыми. Среди всех языков программирования язык Ассемблера является самым низкоуровневым, поэтому программы, написанные на нём должным образом, работают быстрее программ, представляющих те же алгоритмы на других языках.

В данной работе описываются функции на языке Ассемблер, реализующие арифметические операции языка программирования ЛЯПАС-Т [1] над большими числами, и способы встраивания их в загружаемый модуль.

Представление данных

В языке ЛЯПАС-Т все операции с длинными операндами выполняются с использованием в качестве одного из операндов собственной переменной τ. В предлагаемой здесь программной реализации операций ЛЯПАСа-Т над большими числами последние и переменная τ представляются логическими комплексами, как описано в [1], с той только разницей, что собственная переменная представляется фиксированным комплексом, называемым собственным. В программе этот комплекс хранится в стеке. Доступ к данным, хранящимся в стеке, осуществляется через смещение от корня стека (регистр `ebp`). Младший элемент комплекса τ расположен по адресу $\texttt{ebp} - 64$; i-й элемент находится по адресу $\texttt{ebp} - 64 - 4i$. Мощность Q комплекса τ хранится по адресу $\texttt{ebp} + 112$.

Когда цепочка операций начинается с некоторого комплекса Li, его значение копируется в комплекс τ. Для любой последующей операции в цепочке комплекс τ является первым операндом и результатом операции.

Далее опишем арифметические операции, интерпретирующие τ как число длины $32Q$ бит. Все они выполняются по модулю 2^{32Q}. Пусть $\circ \in \{-, +, *, /, ;\}$. Тогда соответствующие операции могут быть выполнены следующим образом.

О п е р а ц и я $\circ$

Пример выполнения: L1$\circ$L2$\Rightarrow$L3. Операция выполнения действия $\circ$ над числами L1 и L2 и занесения результата в L3. Выполняется записанная на языке ЛЯПАС-Т строка следующим образом:

1) L1 копируется в τ;
2) над τ и L2 выполняется операция $\circ$. При этом результат находится в τ, а комплекс L2 не изменяется;
3) значение τ копируется в комплекс L3.

Таким образом, после выполнения результат операции $\circ$ над числами L1 и L2 находится в L3 и в τ.

Следующая операция отличается от предыдущих тем, что в результате её выполнения один из параметров изменяется.

Д е л е н и е (о п е р а ц и я «:»)

Пример выполнения: L1:L2$\Rightarrow$L3.

Операция деления числа L1 на число L2 и занесения результата в L3. Выполняется записанная на языке ЛЯПАС-Т строка следующим образом:

1) L1 копируется в τ;
2) τ делится на L2. При этом результат (частное) находится в τ, а в комплекс L2 записывается остаток от деления. Таким образом, L2 в результате выполнения операции «:» изменяется;
3) значение τ копируется в комплекс L3.

Таким образом, после выполнения этой операции результат деления числа L1 на число L2 (частное) находится в L3 и в τ, а в L2 — остаток от деления.

Все представленные операции используют функции, написанные на языке Ассемблер. Эти функции реализуют алгоритмы работы с длинными числами, описанные в книге Дональда Кнута [2], с минимальными изменениями.

Генерация ассемблерного кода, реализующего функции над большими операндами языка ЛЯПАС-Т, может выполняться двумя способами.

В первом случае функция оформляется как подпрограмма на языке Ассемблера с соблюдением всех соглашений о правилах определения подпрограмм в Ассемблере. Компилятором генерируются команды занесения входных аргументов операции в стек, затем команда вызова подпрограммы `call` и команды последующего анализа результатов работы подпрограммы.

Во втором случае в компиляторе определяются функции на языке `C++`, которые формируют строку из команд языка Ассемблера, реализующих действия над длинными операндами. Эта строка встраивается компилятором в общую строку, представляющую результат генерации целевой программы на Ассемблере.

Сравнивая эти подходы, можно сделать следующий вывод. В первом случае сокращается длина выходного кода, но увеличивается время его работы, так как вызов подпрограммы и возврат из неё занимают дополнительное время. Во втором случае имеем экономию времени работы за счёт отсутствия указанных действий, но увеличение длины кода, так как при появлении данных операций повторно соответствующий

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