Секция 4. **Физико-математические методы в приборостроении**

УДК 510.51+004.42

А. А. Айзикович, канд. физ.-мат. наук, доц.
В. М. Коровин, студент
E-mail: pmi@istu.ru
Ижевский государственный технический университет имени М. Т. Калашникова

Программная реализация машины с неограниченными регистрами

Дан краткий обзор существующих подходов к определению понятия «алгоритм» – одного из ключевых понятий современной математики. Приводятся начальные сведения об абстрактной машине с неограниченными регистрами, представляющей инструмент одной из точных концепций современной теории алгоритмов, и дано описание эмулятора, выполненного на языке программирования высокого уровня, реализующего эту машину как конечный автомат. Разработанный эмулятор может быть использован в учебном процессе в качестве демонстрационной программы.

Ключевые слова: теория алгоритмов, машина с неограниченными регистрами, эмулятор.

Введение

Понятие алгоритма, подобно понятиям множества и натурального числа, принадлежит к числу понятий столь фундаментальных, что оно не может быть выражено через другие (в частности, теоретико-множественные), а должно рассматриваться как неопределяемое. Лишь как пояснения, а не как определения следует расценивать формулировки типа «Алгоритм – это точное предписание, которое задает вычислительный процесс (в широком смысле слова), называемый в этом случае алгоритмическим, начинающийся с произвольного исходного данного, из некоторой совокупности возможных для данного алгоритма исходных данных, и направленный на получение полностью определяемого этим исходным данным результата» [1, 2]. Приведем другие словесные описания алгоритма. Алгоритм по Колмогорову [3] – это всякая система

вычислений, выполняемых по строго определенным правилам, которая после какого-либо числа шагов заведомо приводит к решению поставленной задачи. Алгоритм по Маркову [3, 4] – это точное предписание, определяющее вычислительный процесс, идущий от варьируемых исходных данных к искомому результату.

Формирование теории алгоритмов как самостоятельного раздела математики, изучающего общие свойства алгоритмов, началось в 30-е годы XX века. Математиками предпринимались попытки построить теорию алгоритмов с ответом на вопрос, что считать алгоритмом. К настоящему времени сложилось несколько направлений в теории алгоритмов: 1) рекурсивные функции [5–14]; 2) системы Поста [6]; 3) нормальные алгорифмы Маркова [5, 6, 10]; 4) абстрактные машины [14], машина Тьюринга (и ее модификации) [2, 5–8, 10, 11, 13, 15], машина Поста [16], машина Колмогорова [2, 17], машина Шёнхаге [2, 17], регистровые машины Минского [18], стековые регистровые машины [9, 19], машина с неограниченными регистрами (Шепердсона и Стерджиса) [6, 7, 10]. Каждое из конкретизаций сужает понятие алгоритма, однако поразительным является тот факт, что все теории эквивалентны: функция, вычислимая по одной теории, является вычислимой и по другой.

Вне рамок строгих теорий вводятся неформальные (интуитивные) понятия, опирающиеся на свойства, присущие алгоритмам – эффективным процедурам. Это дискретность, детерминированность, элементарность, результативность (направленность), массовость [7, 10, 13, 20]. Однако Роджерс в [12] отмечает несколько иные признаки алгоритма: набор инструкций, наличие вычислителя, наличие возможности для выделения, запоминания и повторения шагов вычисления, дискретность, детерминированность, а Кнут [21] – конечность, определенность, ввод, вывод, эффективность. Кроме того, Колмогоровым [1] выделены семь характеризующих алгоритм параметров: 1) совокупность возможных исходных данных, 2) совокупность возможных результатов, 3) совокупность возможных промежуточных результатов, 4) правило начала, 5) правило непосредственной переработки, 6) правило окончания, 7) правило извлечения результата.

Что касается программирования, то, как правило, вопрос определения алгоритма здесь не обсуждается, а интерпретируется как эквивалент программы для ЭВМ. Например, в работах [22, 23], содержащих в своих названиях слово «алгоритм», вообще не упоминается, что это такое, или упоминается вскользь. С точки зрения современной практики алгоритм – это программа, а критерием алгоритмичности процесса является возможность его запрограммировать, или алгоритм – это набор команд, необходимый для решения той или иной задачи [24]. Именно благодаря

этой реальности алгоритма, а также потому, что подход инженера к математическим методам всегда был конструктивным, понятие алгоритма в технике за короткий срок стало необычайно популярным (быть может, даже больше, чем в самой математике) [10].

В связи с информатизацией (и цифровизацией) общества в программу подготовки специалистов по компьютерным направлениям стали включать такие дисциплины, как «Теория алгоритмов» и «Теоретические основы информатики». При этом встает вопрос: как излагать учебный материал [6, 7, 9, 10, 13, 25, 26]? Одним из таких подходов является изложение материала с использованием абстрактной машины с неограниченными регистрами (МНР), что гораздо нагляднее машины Тьюринга (учитывая «близость» МНР к реальным компьютерам) и проще, чем математическая теория рекурсивных функций. Кроме того, на этой машине автоматически реализуются все семь характерных признаков алгоритма.

В статье описывается эмулятор МНР, созданный с использованием универсального языка программирования высокого уровня. Разработанная программа позволяет демонстрировать работу МНР со всеми возможностями машины: ввод и вывод данных, получение промежуточных результатов, линейные алгоритмы, ветвление, циклы, подключение подпрограмм, реализация рекурсивных вычислений и имитация аварийного останова.

Машина с неограниченными регистрами

МНР содержит счетное число регистров, занумерованных натуральными числами R_1, R_2, …, R_n, … В каждый регистр может быть записано любое неотрицательное целое число. Машина обрабатывает по заданной программе конечную последовательность заданных чисел и выдает единственное значение. Таким образом, МНР работает с конструктивными объектами, которые можно поименовать натуральными числами.

Начальные данные (начальная конфигурация C_0 из n чисел) содержатся в n первых регистрах машины, остальные регистры содержат нули. Результат считывается в первом регистре R_1 после завершения программы.

Программа машины задается конечной последовательностью I_1, I_2, …, I_s из четырех возможных команд I_k: арифметических команд обнуления, добавления единицы и пересылки, а также команды управления – условного перехода, которая может выполнять и функцию безусловного перехода.

Работа программы начинается с первой команды, далее управление передается следующей команде, если предыдущая выполненная коман-

да была арифметической, иначе – по предписанию команды управления. Процесс заканчивается, если номер вызываемой команды отсутствует в программе. В МНР не предусмотрен «аварийный останов» (например, при делении нечетного числа на два); его роль выполняет зацикливание.

Команды МНР

Опишем команды МНР:

– команда $Z(n)$ – в регистр R_n записывается 0, остальные регистры не изменяются;

– команда добавления единицы $S(n)$ – к содержимому регистра R_n прибавляется 1, остальные регистры не изменяются;

– команда пересылки $T(m, n)$ – в регистр R_n записывается содержимое регистра R_m, остальные регистры не изменяются;

– команда условного перехода $J(m, n, q)$ – управление передается команде с номером q, если содержимое регистров R_m и R_n совпадают, иначе – к следующей команде. При выполнении этой команды все регистры не изменяются. В случае $J(m, n, q)$ имеем команду безусловного перехода к команде I_q.

После выполнения текущей команды изменяется (или нет) текущая конфигурация. После завершения работы программы машина находится в заключительной конфигурации C_z, а первый регистр R_1, как отмечалось, содержит результат вычисления.

В качестве примера программы МНР приведем программу вычисления функции знака неотрицательного целого числа:

$$\text{sign}(x) = 0, \text{ если } x = 0; \ \text{sign}(x) = 1, \text{ если } x \neq 0.$$

Пусть начальная конфигурация $C_0 = (x, 0, 0, \ldots)$. Имеем:

I_1 $\quad J(1,2,4) \quad$ $\#x = 0?$

I_2 $\quad Z(1)$

I_3 $\quad S(1)$

Здесь заключительной конфигурацией является $C_z = (0,0,0,...)$ при $x = 0$ или $C_z = (0,0,0,...)$ при $x \neq 0$.

Работа машины

Перейдем к некоторой конкретике программной реализации МНР. В эмуляторе МНР введен особый регистр – счетчик адресов команд (САК), а также команда останова Stop.

Работа на МНР состоит из следующих этапов:

1) в регистры памяти данных вводятся числа, т. е. задается начальная конфигурация;

2) вводятся команды в ячейки программной памяти;

3) записывается в САК единица, запускается МНР, и она начинает работать, выполняя команды программы, пока не встретит команду Stop.

При этом выполнение каждой команды МНР состоит из четырех этапов:

1) считывается адрес из САК;

2) считывается команда из памяти по этому адресу;

3) САК увеличивается на единицу в случае, если команда арифметическая или команда не осуществляет условный переход, иначе – изменяется содержимое САК;

4) выполняется прочитанная команда.

МНР выполняет каждую команду программы до команды Stop. Далее считается результат в регистре R_1.

Выбор технологии для реализации МНР

Опираясь на постановку задачи, сделан вывод, что наилучшим способом реализации данной программы будет разработка GUI приложения. Данный способ является наилучшим, поскольку:

1) графический интерфейс более дружественный и приятный для глаз,

2) обеспечивает легкий доступ к функциям системы и приложений,

3) может легко использоваться начинающими пользователями.

Отталкиваясь от основных требований задачи, среди которых графическая визуализация и объектная ориентированность, было принято решение остановиться на языке программирования C++. Выбор среды разработки осуществлялся на основе выбранного языка – это среда разработки C++ Builder.

Отметим, наконец, что хотя МНР и является «идеализированным компьютером» с отсутствием ограничений по размеру памяти и величине чисел, поступающих на вход, программа реализует конечный автомат – математическую модель с конечной памятью, число регистров которой не будет превышать 20, а значение каждого – 4 294 967 295.

Подпрограммы и рекурсивные вычисления

В обсуждаемом проекте реализованы подключение подпрограмм и рекурсивные вычисления. Подпрограммы выполняются с сохранением значений регистров основной программы, и обеспечивается возврат в основную программу на нужную команду.

Для вызова подпрограмм была введена команда Call. Данная команда принимает единственный операнд – номер вызываемой команды. При выполнении команды Call осуществляется добавление номера следующей команды в стек адресов возврата, а номеру следующей команды присваивается значение из первого операнда.

Одна из необходимых особенностей теории алгоритмов – это обеспечение механизма вычисления рекурсивных функций. Описанная реализация команды Call с использованием стека адресов возврата позволяет создавать рекурсивные программы и программы со сложной схемой вызовов подпрограмм. При каждом вызове подпрограммы в стек добавляется адрес возврата (т.е. адрес команды, следующей за командой Call). При завершении подпрограммы из стека извлекается адрес следующей команды. Подобная схема использования стека при вызове подпрограмм используется во многих реальных процессорах, включая процессоры Intel и AMD архитектуры x86, используемые в персональных компьютерах.

Получение начальной конфигурации

Файл с исходным кодом указывается пользователем с помощью диалогового окна выбора файла при нажатии на кнопку «Открыть файл». При этом очищается массив исходных значений регистров, производится загрузка исходного кода в память и его парсинг, после чего массиву текущих регистров присваиваются значения исходных регистров, очищается стек.

Парсинг представляет собой обработку строк исходного кода, игнорирующую: пустые строки, пробелы и табуляции в начале и в конце строк, а также между элементами команды; комментарии, начинающиеся с символа «#» и заканчивающиеся в конце строки. Первая значимая строка загружаемого файла может содержать разделенные пробелами целые неотрицательные числовые значения регистров R_1, R_2, R_3 и т. д. Если количество значений меньше количества возможных регистров, то оставшиеся регистры содержат нулевые значения.

При парсинге текстовые представления команд преобразуются в так называемый байт-код, т. е. структуру, хранящую код команды и его операнды, используя которую можно исполнять МНР-программу очень быстро. Кроме того, такой подход позволяет обнаружить синтаксические ошибки в коде уже на этапе загрузки программы. При обнаружении синтаксических ошибок на экран выводятся соответствующие сообщения.

Выполнение программы

После загрузки программа переходит в состояние (режим) пошагового исполнения, при котором в окне исходного кода подсвечивается следующая для исполнения команда, выводятся значения текущих регистров, а пользователь имеет возможность выполнить нужную ему операцию:

1) открыть новый файл;

2) запустить программу на непосредственное исполнение (ограничив максимальное количество команд нужным значением во избежание зацикливания);

3) сделать шаг (выполнить одну команду);

4) запустить режим анимации (режим автоматического выполнения шагов с небольшим интервалом: 50, 200 или 500 миллисекунд в зависимости от выбранного режима скорости, при котором автоматически обновляется подсветка следующей команды в окне исходного кода и значения текущих регистров);

5) отобразить строку со следующей для выполнения команды (если пользователь переместил курсор в окне исходного кода);

6) перезапустить программу;

7) очистить данные (выгрузить программу из памяти).

Диалоговое окно программы представляет собой две панели, на одной из которых приведена программа МНР (загружается пользователем из текстового файла со специальным расширением), и при необходимости подсвечивается выполняемая команда, на другой – выводится информация о текущей конфигурации.

Получение результата

Программа завершается в одном из следующих случаев:

1) выполненная команда была последней в списке команд, стек адресов возврата пуст;

2) осуществлен условный переход на команду с номером 0 либо на неимеющуюся команду (с номером, превышающим номер последней команды), стек адресов возврата пуст.

В качестве результата принимается значение, хранящееся в регистре R_1 на момент завершения программы. Данный результат выводится на экран в виде диалогового окна.

После завершения программы ее можно перезапустить, нажав на кнопку «Перезагрузить». При этом в массив текущих регистров будут загружены значения исходных регистров, стек адресов возврата очищен, а программа перейдет в пошаговый режим работы.

Заключение

Результатом разработки является программа, реализованная в среде C++ Builder Community Edition, являющаяся эмулятором абстрактной вычислительной машины с неограниченными регистрами. Эмулятор выводит на экран листинг программы МНР из ранее подготовленного файла в собственном формате и результат работы программы; может в непрерывном режиме и пошагово выполнять инструкции с их подсветкой и отображением изменения регистров, подключать подпрограммы с правильной организацией памяти МНР и передачей управления, реализовывать рекурсивные программы.

Спроектированный эмулятор представляет интерес как сам по себе, так и может быть использован как демонстрационная программа в таких дисциплинах учебного процесса, как «Информатика», «Дискретная математика», «Теория алгоритмов» [25], а также может быть подспорьем при проверке программ МНР, написанных вручную.

Также отметим, что любая реализованная на компьютере программа может быть «запрограммирована» в рамках любой из предложенных теорий алгоритмов, а доказать несуществование компьютерной программы для решения задачи можно только с помощью какой-то из этих теорий. Таким образом, понятие алгоритма является не только центральным понятием теории алгоритмов, не только одним из главных понятий математики вообще, но и одним из главных понятий современной науки. Более того, сегодня, с наступлением эры информатики, алгоритмы становятся одним из важнейших факторов цивилизации [2].

Список литературы

1. *Успенский, В. А.* Алгоритм // Математическая энциклопедия: в 5 томах / В. А. Успенский. – Т. 1. – Москва : Советская энциклопедия, 1977. – С. 202–206.
2. *Успенский, В. А.* Теория алгоритмов: основные открытия и приложения / В. А. Успенский, А. Л. Семенов. – Москва : Наука, 1987. – 288 с.
3. Определение алгоритма. – URL: https://studbooks.net/2237922/informatika/opredelenie_algoritma (дата обращения: 04.10.2020).
4. *Першиков, В. И.* Толковый словарь по информатике / В. И. Першиков, В. М. Савинков. – Москва : Финансы и статистика, 1991. – 543 с.
5. *Герасимов, А. С.* Курс математической логики и теории вычислимости / А. С. Герасимов. – Санкт-Петербург : ЛЕМА, 2011. – 284 с.
6. *Катленд, Н.* Вычислимость. Введение в теорию рекурсивных функций / Н. Катленд. – Москва : Мир, 1983. – 256 с.
7. *Крупский, В. Н.* Теория алгоритмов / В. Н. Крупский, В. Е. Плиско. – Москва : Академия, 2009. – 208 с.
8. *Кузнецов, О. П.* Дискретная математика для инженера / О. П. Кузнецов, Г. М. Адельсон-Вельский. – Москва : Энергоатомиздат, 1988. – 480 с.

9. *Марченков, С. С.* Элементарные рекурсивные функции / С. С. Марченков. – Москва : МЦНМО, 2003. – 112 с.

10. *Матрос, Д. Ш.* Теория алгоритмов / Д. Ш. Матрос, Г. Б. Поднебесова. – Москва : БИНОМ. Лаборатория знаний, 2008. – 202 с.

11. Машины Тьюринга и рекурсивные функции / Г.-Д. Эббинхауз, К. Якобс, Ф.-К. Ман, Г. Хермес. – Москва : Мир, 1972. – 264 с.

12. Роджерс, Х. Теория рекурсивных функций и эффективная вычислимость / Х. Роджерс. – Москва : Мир, 1972. – 624 с.

13. *Судоплатов, С. В.* Математическая логика и теория алгоритмов / С. В. Судоплатов, Е. В. Овчинникова. – Москва : ИНФРА-М ; Новосибирск : Изд-во НГТУ, 2008. – 224 с.

14. *Успенский, В. А.* Лекции о вычислимых функциях / В. А. Успенский. – Москва : ГИФМЛ, 1960. – 492 с.

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

16. *Успенский, В. А.* Машина Поста / В. А. Успенский. – Москва : Наука, 1988. – 96 с.

17. *Успенский, В. А.* Алгоритмы, или машины, Колмогорова / В. А. Успенский, А. Л. Семенов // Теория информации и теория алгоритмов / А. Н. Колмогоров. – Москва : Наука, 1987. – С. 279–289.

18. *Минский, М.* Вычисления и автоматы / М. Минский. – Москва : Мир, 1971. – 367 с.

19. *Бельтюков, А. П.* Машинное описание и иерархия начальных классов Гжегорчика // Записки научных семинаров Ленинградского отделения Математического института им. В. А. Стеклова АН СССР. – 1979. – Т. 88. – С. 30–46.

20. *Горбатов, В. А.* Фундаментальные основы дискретной математики. Информационная математика. – Москва : Наука, Физматлит, 2000. – 544 с.

21. *Кнут, Д. Э.* Искусство программирования : в 4 томах. Т. 1. Основные алгоритмы / Д. Э. Кнут. – Москва : Вильямс, 2002. – 720 с.

22. *Вирт, Н.* Алгоритмы и структуры данных / Н. Вирт. – Москва : Мир, 1989. – 360 с.

23. *Кормен, Т. Х.* Алгоритмы: вводный курс / Т. Х. Кормен. – Москва : Вильямс, 2014. – 208 с.

24. *Стивенс, Р.* Алгоритмы. Теория и практическое применение / Р. Стивенс – Москва : Изд-во «Э», 2016. – 544 с.

25. *Айзикович, А. А.* Построение курса «Теория алгоритмов» / А. А. Айзикович, Л. А. Лещева // Прикладная математика и информатика : сб. ст. науч.-метод. конф. – Ижевск : Изд-во ИжГТУ имени М. Т. Калашникова, 2018. – С. 89–92.

26. *Саидахмедова, М. Б.* Машина с неограниченными регистрами (МНР) / М. Б. Саидахмедова, Ю. М. Юсупов, Г. С. Рагимханова // Модернизация системы непрерывного образования : материалы VI Междунар. науч.-практ. конф. – Махачкала : Изд-во ДГПУ, 2014. – С. 557–561.

A. A. Aizikovich, CSc in phys. and math., associate professor
V. M. Korovin, student
E-mail: pmi@istu.ru
Kalashnikov Izhevsk State Technical University, Izhevsk, Russian Federation

Software Implementation of a Machine with Unlimited Registers

A brief overview of the existing approaches to the definition of "algorithm" – one of the key concepts of modern mathematics – has been given. Initial information about an abstract machine with unlimited registers, representing the instrument of one of the exact concepts of modern algorithm theory, is given, and a description of the emulator made in the high-level programming language that implements this machine as the final machine. The emulator developed can be used in the training process as a demonstration program.

Keywords: algorithm theory, machine with unlimited registers, emulator.