Фундаментальная информатика. Курсовая работа, 1 курс, 1 семестр 2022/23 учебного года

Программирование машин Тьюринга

Когда вице-президента фирмы IBM спросили, почему на персональных компьютерах IBM PS/2 нет кнопки Reset, он ответил: «Наши компьютеры никогда не зависают. Да и шесть долларов, в которые обходится эта кнопка, не лишние!»

Составить программу машины Тьюринга *в четвёрках,* выполняющую заданное действие над словами, записанными на ленте. Отладку и тестирование проводить в среде интерактивного действующего макета **jstu4**. Алфавит МТ определяется заданием. Использование дополнительных (несобственных) букв (кроме λ) нежелательно. Необоснованное использование несобственных букв считается недочётом и является достаточным основанием для снижения оценки.

В начальном состоянии головка МТ находится на пустой ячейке непосредственно справа от записанных на ленте аргументов – слов входного сообщения. В конечном состоянии головка МТ должна находиться на пустой ячейке непосредственно справа от результата (последнего преобразованного или вновь сгенерированного слова результирующего сообщения).

Вычисления в программе, как правило, должны быть нормированными (аргументы после работы программы сохраняются на ленте в неизменном виде и не остаётся промежуточных результатов); ненормированные вычисления, особенно в простых случаях, считаются недочётом и являются достаточным основанием для снижения оценки.

Перед составлением алгоритма для машины Тьюринга необходимо подготовить тесты для него – представительный набор различных входных сообщений, для которых известен правильный ответ, включая значения на границах области определения вычислимой функции и за её пределами.

В отчёте по данной работе необходимо дать сложностную оценку реализованного алгоритма.

Примеры программ: слева – ASCII-крутилка, справа – *ненормированное* сложение двоичных чисел без знака:

`00, ,-,00`
`00,-,\,00`
`00,\,|,00`
`00,|,/,00`
`00,/, ,01`
`01, ,>,00`
	`00, ,<,01`
	`01,0,1,02` *перенос -1*
	`02,1,<,01` *перенос -1*
	`01,1,0,03` *-1*
	`03,0,<,03`
		`03,1,<,03`
		`03, ,<,04`
		`04,0,1,05` *+1*
		`05,0,>,05`
		`05,1,>,05`
			`05, ,>,00`
			`00,1,>,00`
			`00,0,>,00`
			`04,1,0,02` *перенос +1*
			`02,0,<,04` *перенос +1*
				`04, ,1,05`
				`01, ,>,06`
				`06,1,>,06`
				`06, ,<,07`
				`07,1, ,06` *стирание*
					`07, , ,07` *стоп*

Варианты заданий (назначаются преподавателем):

№	Действие (звездочками помечены более трудные задачи)
1*	Вычисление поразрядной конъюнкции двух двоичных чисел.
2*	Вычисление поразрядной дизъюнкции двух двоичных чисел.
3*	*Обмен местами* двух двоичных чисел.
4	Нормированное вычисление суммы двух двоичных чисел без знака.
5**	Вариант 4 с логарифмической сложностью.
6	Генерация двух чисел из чётных и нечётных разрядов двоичного числа.
7*	Генерация двух чисел из разрядов двоичного числа, находящихся на чётных и нечётных позициях.
8	Обмен местами разрядов двоичного числа, находящихся на чётных и нечётных позициях.
9	Зеркальное отражение цифр двоичного числа относительно его середины
10	Вычисление логического произведения (&& в Си) двоичных чисел.
11***	Вычисление наибольшего общего делителя двух чисел в натуральной системе счисления.
12***	Вычисление наименьшего общего кратного двух чисел в натуральной системе счисления.
13***	Проверка делимости на три
14	Вычисление двоичного логического сдвига второго числа влево на число разрядов, равное первому.
15	Вычисление двоичного логического сдвига первого числа вправо на число разрядов, равное второму.
16*	Вычисление двоичного арифметического сдвига второго числа влево на число разрядов, равное первому.
17*	Вычисление двоичного арифметического сдвига первого числа вправо на число разрядов, равное второму.
18*	Вычисление двоичного циклического сдвига второго числа влево на число разрядов, равное первому.
19*	Вычисление двоичного циклического сдвига первого числа вправо на число разрядов, равное второму.
20*	Выделение разрядов первого двоичного числа по маске, заданной вторым числом.
21*	Выделение разрядов второго двоичного числа по маске, заданной первым числом.
22	Закодировать двоичное число азбукой Морзе.
23*	Умножение двух чисел в кардинальной системе счисления {|}.
24	Кодирование числа в римской записи по Цезарю (в алфавите {I, V, X, L, C, D, M})
25	Умножение однозначных чисел в усечённой римской системе счисления.
26*	Проверить палиндромию двоичного числа.
27*	Вычисление двоичного логарифма двоичного числа.
28	Натурализация двоичного числа в позиционной записи (перевод в натуральную систему счисления {|}).
29**	Двоичное сложение двоичного и четверичного числа.
30	Восстановление целого числа в восьмеричной системе счисления по обратному коду.
31	Восстановление целого числа в восьмеричной системе счисления по дополнительному коду.
32	Уменьшение на единицу целого неотрицательного числа в восьмеричной системе счисления.

33 Увеличение на единицу целого неотрицательного числа в восьмеричной системе счисления.
34 Получение двоичного числа, противоположного данному, в обратной кодировке.
35 Получение двоичного числа, противоположного данному, в дополнительной кодировке.
36* Вычисление разности двух двоичных чисел без знака, при условии, что первое число больше второго.
37** Задача 36 с логарифмической сложностью.
38* Задача 36, ответ — модуль разности.
39** Задача 37, ответ — модуль разности.
40 Копирование троичного числа со знаком.
41 Реверс троичного числа со знаком (запись цифр в обратном порядке).
42* Зеркальное отражение двух двоичных слов относительно промежутка между ними.
43 Перевод числа из двоичной системы счисления в восьмеричную.
44 Перевод числа из восьмеричной системы счисления в двоичную.
45 Перевод числа из троичной системы счисления в девятеричную.
46 Перевод числа из девятеричной системы счисления в троичную.
47* Перевод числа из двоичной системы счисления в восьмеричную с логарифмической сложностью.
48* Перевод числа из восьмеричной системы счисления в двоичную с логарифмической сложностью.
49* Перевод числа из троичной системы счисления в девятеричную с логарифмической сложностью.
50* Перевод числа из девятеричной системы счисления в троичную с логарифмической сложностью.
51 Проверка делимости на 9.
52* Проверка делимости на 11.
53** Четверичное сложение двоичного и четверичного числа.

Варианты заданий составлены проф. Зайцевым В.Е., доц. Сошниковым Д.В., ст. преп. Сеницким П.А., Перетягиным И.А. и асп. Макаровым Н.К.

Первый макет МТ разработан в 1987 г. студ. Лукашевичем С. Ю. Описание подготовлено проф. Зайцевым В. Е., доц. Журавлевой Т. Э. и ст. преп. Сеницким П. А. на основе курсовой работы Лукашевича С. Ю. Примеры переработаны Перетягиным И. А., Марухиным А. В., Чечериндой С. В. и др. В последнее время используется макет МТ, реализованный на языке JavaScript Дубининым А.В., и имеющий графический web-интерфейс: **jstu4**. Его, как и другие инструменты (`pytu4`, `turun`), можно найти и скачать на форуме faq8.ru: http://faq8.ru/read.php?2,16248.

Для *домашних* работ могут использоваться и другие системы Тьюринговских вычислений. В рамках проекта GNU существует gturing, хороший экранный (в среде GNOME!) интерпретатор МТ с возможностью редактирования текста программы, однако, задаваемой в пятёрках, причём в качестве разделителя слов используется пробел. Существуют самые разнообразные макеты МТ: для MS Excel, Lego-версия http://legoofdoom.blogspot.com и iPhone-симулятор http://mobile.clauss-net.de/Turing. Видео вполне натуральной машины Тьюринга можно найти здесь http://aturingmachine.com/

Вопросы к защите отчёта по заданию I КР.
ВВОД И ПРЕДСТАВЛЕНИЕ ДАННЫХ

1. Коды ASCII и КОИ-8: характеристики, состав, структура.
2. Альтернативная и основная кодировки (в сравнении с ASCII).
3. Кодировки ISO 8859-5 и CP 1251.
4. Понятие о кодах EBCDIC, ДКОИ.
5. Кодировки Unicode, UTF-8. ISO 8859-5 и 10646.
6. Понятие о клавиатурных раскладках. Основные принципы.
7. Раскладки QWERTY и ЙЦУКЕН и соответствие между знаками кириллицы и латинского алфавита для обычной и фонетической латино-кириллических раскладок.
8. Раскладки Дворака и Diktor.
9. Позиционные системы счисления.
10. Представление целых чисел в ЭВМ.
11. Перевод чисел из одной системы счисления в другую.
12. Особенности целочисленной арифметики в ЭВМ.
13. Научная (экспоненциальная) форма записи числа. Машинное представление с плавающей точкой.
14. Различия представлений числовых и текстовых данных в ЭВМ.
15. Использование калькуляторов ОС UNIX (bc) и MS Windows для операций с числами в различных системах счисления.

Литература

1. Бауэр Ф.Л., Гооз Г. Информатика. –М.: Мир, 1976, 1990. Глава 1 и приложения А; Глава 6 и приложение В (D).
2. Ворощук А.В. Основы ЦВМ и программирование. –М.: Наука, 1978. с. 23-39,59-80.
3. Форсайт Дж., Малькольм М., Моулер К. Машинные методы математических вычислений. –М.: Мир, 1980, с. 22-26
4. Карасев С.Б. и др. Машинные алгоритмы обработки информации. –М.: МАИ, 1987. с. 3-16.
5. Танненбаум Э. Многоуровневая организация ЭВМ. –М: Мир, 1979.
6. UNIX on-line manuals.
7. Уэзерелл Ч. Этюды для программистов. –М.: Мир, 1982.

Краткое описание действующего макета машин Тьюринга

Интерактивный макет машин Тьюринга jstu4 реализован как одностраничная web-программа без серверной части и, таким образом, способен работать в любой системе, где имеется html5-совместимый браузер. Например, чтобы запустить его в браузере firefox, можно выполнить следующие команды в командной оболочке ОС UNIX или их аналоги в графическом интерфейсе:

```
unzip jstu4-2.3.zip
firefox jstu4/index.html
```

Макет реализует следующий вариант определения МТ:

- лента МТ «полубесконечна» вправо;
- программа МТ состоит из *четвёрок* `<q,a,v,q'>` где `q` – *символ* старого состояния, `a` – обозреваемая *буква*, `v` – *символ* действия: записываемая буква или команда перемещения, `q'` – *символ* нового состояние;
- поддерживаются следующие команды перемещения:
  - `>` – сдвиг на одну ячейку вправо (знак > реализует символ движения вправо на одну ячейку ***r***);
  - `<` – сдвиг на одну ячейку влево (знак < реализует символ движения влево на одну ячейку ***l***);
  - `=` – сохранение текущего положения;
  - `#` – останов макетной МТ (классическая команда *останова* (`q,a,a,q`) также реализована).

Особенности реализации:

- четвёрки `q,a,v,q'` записываются без лишних пробелов. Элементы четвёрки разделяются запятыми;
- для обозначения состояний МТ допускается использовать неотрицательные десятичные числа, а также произвольные слова, состоящие из букв латинского алфавита, цифр и знака _. Осмысленные названия состояний могут улучшить читаемость кода программы МТ;
- выполнение МТ всегда начинается с нулевого состояния (0 или 00 или 000 и т.д.);
- буквами рабочего алфавита могут быть любые знаки, присутствующие в Юникоде, за исключением знаков `< > = #`, изображающих символы действия МТ;
- несобственная буква **λ** кодируется пробелом;
- для удобства форматирования кода, одна строка программы может содержать несколько четвёрок, разделённых одним или несколькими пробелами. Допустимы также и пустые строки;
- в тексте программы допустимы комментарии, начинающиеся с // или #
- ввод и редактирование текста программы МТ можно осуществлять с помощью самого интерпретатора МТ, однако, сохранить текст программы в файл, ввиду web-страничной природы интерпретатора, невозможно: для этого нужно скопировать текст программы в любой текстовый редактор.

Макет МТ предоставляет пользователю следующие возможности:

- пошаговая интерпретация команд программы;
- быстрая интерпретация программы без остановок;
- редактирование программы МТ;
- подробное описание ошибок, возникающих при выполнении программы МТ.

Опционально выдаётся пространственная и временная сложности текущего исполнения алгоритма.

Макет МТ поддерживает следующие горячие клавиши:

- запуск программы на выполнение **Ctrl+Enter**
- переключение между быстрой и пошаговой интерпретацией программы **Enter**
- выполнение очередного шага **<пробел>**

Для протоколирования работы МТ во избежание появления нетекстового мусора в протоколе можно использовать утилиты `turun` или `pytu4`.

`pytu4` полностью совместима с jstu4 по формату программ, однако, для работы требует установки python 3.9 или выше.

`pytu4` поддерживает запуск с пакетом тестов (один тест — одна строка). В этом режиме pytu4 завершается после первой ошибки:

`./pytu4` *файл-с-программой.tu файл-с-тестами.txt*

или в интерактивном режиме (данные вводятся с клавиатуры):

`./pytu4` файл-с-программой.tu

В интерактивном режиме `pytu4`, для удобства отладки программ МТ, заново считывает *файл-с-программой.tu* перед каждым тестом и не завершается при ошибках.

`turun` является скомпилированной бинарной программой и не требует установки дополнительных системных средств, но он более требователен к форматированию текста программы:

- состояния должны обозначаться двухзначными десятичными или шестнадцатеричными числами (произвольные слова недопустимы);
- комментарии в тексте не поддерживаются;
- одна строка должна содержать ровно одну команду МТ.

`turun` необходимо запускать с пакетом тестов:

`./turun` файл-с-программой.tu файл-с-тестами.tst

При этом каждый тест в файле занимает две строки: в первой — начальные данные на ленте, во второй — знак ^, отмечающий позицию головки МТ в начале работы. Например:

```
011101 10110
       ^
```

Также для протоколирования хорошо подходила Питон-версия интерпретатора МТ со строчным текстовым интерфейсом, утраченная в результате нашествия ядовитых змей на faq8.