ПОРІВНЯЛЬНИЙ АНАЛІЗ ШВИДКОДІЇ JAVA НА МІКРОКОМП'ЮТЕРІ RASPBERRY PI

Дідух О. І.; Тищенко В. В.
Національний технічний університет України «Київський політехнічний інститут», м. Київ, Україна

Сьогодні великої популярностті набули мікрокомп'ютери, зокрема *Raspberry Pi*. На їх основі будуються різноманітні системи: керування, контролю, спостереження, розпізнавання і т. д. Розроблений британським Фондом *Raspberry Pi Foundation* мікрокомп'ютер *Raspberry Pi Model B* має 700 МГц процесор *ARM*1176–*JZFS* та 512МБ оперативної пам'яті *LPDDR*2–800. Таке оснащення дозволяє використовувати високорівневі мови програмування. Однією з них є об'єктно-орієнтована мова програмування *Java*.

Метою даної статті є аналіз швидкодії різних версій *Java* на *Raspberry* Pi, визначення оптимальної версії *Java* для застосування.

Практичним методом перевірки швидкодії *Java* є визначення часу виконання певної програми на віртуальній машині *Java* (*JVM*). Для порівняння використаємо дві функціонально відмінні програми. В першому випадку, реалізуємо сучасний швидкий алгоритм пошуку простих чисел до заданого цілого числа *N* на *Java* (решето Аткіна).

В другому випадку, розглянемо програму, яка буде циклічно виконувати операції множення, ділення, та додавання над числами з плаваючою крапкою.

При обчисленні часу роботи проведемо $N = 10$ запусків алгоритму та отримаємо відповідні часи роботи: $(t_1, \ldots, t_N)$. Визначимо мінімально можливий час роботи алгоритму:

$$t_{\min} = \min_{i=1,\ldots,N} t_i,$$

максимальний:

$$t_{max} = \max_{i=1,\ldots,N} t_i$$

та середній:

$$\bar{t} = \frac{1}{N}\sum_{i=1}^{N} t_i$$

Для порівняльного аналізу розглянемо середній час роботи, оскільки при його визначенні враховані впливи сторонніх факторів, так як мінімальний час отримати нелегко.

Для оцінки похибки вимірювання обрахуємо стандартне відхилення (σ) за формулою:

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(t_i - \overline{t}\right)^2}$$

де: N — кількість запусків алгоритму, t_i— тривалість i-го запуску, $\overline{t}$ — середній час роботи алгоритму.

Кількість запусків алгоритму підібрана таким чином, щоб стандартне відхилення не перевищувало 5% від середнього часу роботи, в іншому випадку результат вимірювання не буде достовірним.

Дослідження часу роботи алгоритму проводиться згідно з наступним планом:

1. Підготовка комп'ютера

3. Проведення трьох холостих запусків алгоритму.

4. Запуск програми на виконання.

5. Обрахунок часу виконання програми.

6. Визначення мінімального, максимального та середнього значення часу виконання.

7. Визначення стандартного відхилення та похибки вимірювання.

Для порівняння проведено запуск програми пошуку простих чисел та програми роботи над числами з плаваючою точкою на персональному комп'ютері (ПК) з двох'ядерним процесором *Intel Core i*3-4330 3.5*GHz* під управлінням 64-розрядної операційної системи *Ubuntu* 14.04 *LTS*. Зі встановленою наступною версією *Java* від *Oracle version* "1.8.0_25".

На мікрокомп'ютері *Raspberry Pi* можливо змінити частоту процесора.

Використаємо частину доступних значень: 700 МГц, 900 МГц, 1000 МГц.

Проведемо запуск програми на кожній з цих частот окремо. Далі встановимо *OpenJDK Java version* "1.7.0_65" на *Raspberry Pi*.

Таблиця 1

Платформа для тестування	$\frac{t_{\text{RaspberryPi}}}{t_{\text{ПК}}}$ Решето Аткіна	$\frac{t_{\text{RaspberryPi}}}{t_{\text{ПК}}}$ Операції з плаваючою крапкою
ПК, Ubuntu 14.04 LTS	1	1
Raspberry Pi, Oracle, 700 MHz	38,5	18,7
Raspberry Pi, Oracle, 900 MHz	30,5	14,5
Raspberry Pi, Oracle, 1000 MHz	26,1	12,9
Raspberry Pi, JDK, 700 MHz	540	256
Raspberry Pi, JDK, 900 MHz	432	198
Raspberry Pi, JDK, 1000 MHz	355	176

Повторимо вищеописані кроки з програмою для обчислення операцій з плаваючою крапкою.

Аналіз результатів показує, що на мікрокомп'ютері *Raspberry Pi* час виконання програм є більшим в порівнянні з тестовим персональним

комп'ютером. На прикладі програми пошуку простих чисел, бачимо, що час виконання в 26,1–38,5 разів більший при встановленій *Java* від *Oracle* в порівнянні з часом роботи на ПК. Даний алгоритм при зміні Java на *OpenJDK* виконується в 355–540 разів довше (табл. 1). Програма для роботи над числами з плаваючою крапкою при встановленій *Java* від *Oracle* виконується в 12,9–18,7 рази довше, а при встановленні *OpenJDK* час виконання збільшується в 176–256 разів в порівнянні з часом виконання на ПК (табл. 1).

На мікрокомп'ютері *Raspberry Pi* елементарні опрації над числами з плаваючою крапкою виконуються швидше в порівнянні зі складним алгоритмом пошуку простих чисел. Також очевидно, що *Java* від *Oracle* працює в 13–14 разів швидше від *OpenJDK*.

Отже, в порівнянні з *Java* на ПК *Oracle Java SE* 8 на *Raspberry Pi* дає хороші показники роботи і рекомендується для встановлення та використання на мікрокомп'ютерах.

Перелік посилань

1. Могильний С. Б. Мікрокомп'ютер Raspberry Pi — інструмент дослідника : посібник. — К. : «Талком», 2014. — 340 с. — ISBN 978-617-7133-48-2

2. Monk S. Raspberry Pi Cookbook / Simon Monk — O'Reilly Media, Inc., 2014. — 408 p.

3. Horan B. Praktical Raspberry Pi / Brendan Horan — Apress Media, 3014. — 260 p.

Анотація

Для визначення швидкодії *Java* на мікрокомп'ютері *Raspberry Pi* застосовано метод порівняння часів виконання двох функціонально відмінних програм на різних версіях віртуальної машини *Java*. Реалізовано програму сучасного швидкого алгоритму пошуку простих чисел до заданого цілого числа *N* (решето Аткіна) та програму для виконання елементарних операцій над числами з плаваючою крапкою.

Ключові слова: *Raspberry Pi, Java,* мікрокомп'ютер, порівняння швидкодії, *Oracle Java, OpenJDK*.

Аннотация

Для определения быстродействия *Java* на микрокомпьютере *Raspberry Pi* применен метод сравнения времен выполнения двух функционально отличных программ на разных версиях виртуальной машины *Java*. Реализована программа современного быстрого алгоритма поиска простых чисел до заданного целого числа *N* (решето Аткина) и программа для выполнения элементарных операций над числами с плавающей точкой.

Ключевые слова: *Raspberry Pi, Java,* микрокомпьютер, сравнение быстродействия, *Oracle Java, OpenJDK*.

Abstract

To determine the performance of Java on Raspberry Pi microcomputer used method of comparing execution times of two functionally different programs on different versions of Virtual Machine Java. Implemented application of modern fast search algorithm primes up to a given integer N (Atkin sieve) and a program to perform basic operations on floating point numbers.

Keywords: Raspberry Pi, Java, microcomputer, performance comparison, Oracle Java, OpenJDK.