УДК 004.2.004.8

Є. І. ЦИБАЄВ, В. Г. ЗАЙЦЕВ, М. В. ПЛАХОТНИЙ

НТУУ«КПІ», Україна

ОЦІНКА ЧАСУ ВИКОНАННЯ ПРОГРАМ

Прогнозування часу виконання програми пов'язано з використанням цілого ряду статичних і статистичних методів оцінювання, а також із імітаціонним моделюванням роботи пристроїв обчислювальної системи. В даній роботі розглядається задача оцінки часу виконання програм для спеціалізованих комп'ютерних систем. Запропонована модель роботи програми у вигляді поглинаючого марківського ланцюга із дискретними станами і дискретним часом. Модель враховує сукупність історій виконання програми та використовує метод оцінки часу виконання лінійних блоків програми, що запрограмовані мовою високого рівня. Наведена блок-схема алгоритму програми та можлива її інтерпретація графом, а також приклад її застосування.

Ключові слова: *поглинаючий марківський ланцюг, історії виконання програми, лінійний блок програми, прогнозування часу виконання програми.*

Обрання процесора та оцінка часу виконання програм на стадії проектування програмного забезпечення спеціалізованих комп'ютерних систем (СКС) викликає великий інтерес у розробників.

Такі системи часто вимагають не тільки безпомилкового виконання певних функцій і алгоритмів системи, але додатково накладають при виконанні певні часові обмеження.

Ці обмеження можуть стосуватися як виконання програм у монопольному режимі, так і загального часу виконання програми з урахуванням часу очікування у черзі в багатозадачних системах.

Ми розглянемо питання щодо часу виконання програм у монопольному режимі.

Під оцінкою часу виконання програм можна розуміти одну з трьох альтернатив [2]:

– вимір значень характеристик функціонування обчислювальної системи (або її частини);

– прогнозування часу виконання програми без виконання на обраному обчислювачі;

– прогнозування часу виконання програми за наявності обраного обчислювача.

За першим підходом мають на увазі дослідження виконання створюваного варіанту програми, виявлення трас, побудову програмних та апаратних емуляторів, створення моделей зовнішнього середовища,тощо[3]. Тобто, підхід пов'язано з виявленням динаміки програми.

Перевага методу полягає у тому, що може бути досягнута висока точність отриманих результатів. Однак, використання цього методу може виявитися неможливим з декількох причин [4]:

– відсутність реального обчислювача;

– реалізація емулятора обчислювача – дуже дорогий та трудомісткий процес;

– значне уповільнення (10-1000 разів) виконання програми завдяки великим накладним витратам, втратам при вимірюваннях і, в результаті, втраті необхідної точності вимірювання.

Прогнозування часу виконання програми пов'язано з використанням цілого ряду статичних і статистичних методів оцінювання, а також із імітаційним моделюванням роботи пристроїв обчислювальної системи. Наприклад, у роботі [4] вивчається така проблема: по тексту програми та інформації про поводження програми, опису архітектури обчислювача оцінити час виконання програми, опису архітектури обчислювача оцінити час виконання програми на цьому обчислювачі. У роботі показано, що специфіка вирішення задачі суттєво змінюється при зміні типу обчислювача. Це проілюстровано на прикладах: обчислювача фон-Нейманівського типу, векторно - конвейєрного типу, обчислювача на процесорах з RISC архітектурою. Окрім того, час виконання програми буде також залежати від особливостей реалізації обчислювачів вказаного типу. Наприклад, від вибору методів боротьби з конфліктами у конвеєрних обчислювачів: структурними, за даними, конфліктами по керуванню, використанням централізованої схеми управління чи станцій резервування у суперскалярних обчислювачів та інш.

З появою багатоядерних мікропроцесорів виникає проблема розподілу обчислень по декільком ядрам і чи буде такий розподіл сталим при повторному вирішенні задачі.

Якщо тип обчислювача попередньо обрано, задача прогнозування суттєво спрощується але виникає питання, наскільки такий вибір вдалий та відпо-

відає задачам СКС. Як стверджується у [4], для обчислювальних систем, що суттєво відрізняються за архітектурою, дуже важко знайти єдину міру оцінки продуктивності для порівняння. У роботі [2] автори роблять висновок: єдиною надійною мірою виміру продуктивності є час виконання програми на конкретному обчислювачі. Всі інші метрики призводять тільки до невірних висновків. Але і в цьому випадку часові критерії вибору можуть бути різними. Наприклад, середній час вирішення окремої задачі, часові обмеження на вирішення задачі, повний середній час вирішення задачі у багатозадачній СКС, враховуючи час перебування у черзі і т.п.

Розглянемо задачу оцінки часу виконання задачі на попередньо обраному обчислювачі.

Будемо вважати, що обраний обчислювач однопроцесорний, тобто не використовуються архітектури, що традиційно відносяться до так званих"паралельних". Окрім того, врахуємо, що алгоритми вирішення всіх задач СКС попередньо відомі і можуть бути представлені як за допомогою блок-схем, так і запрограмовані мовою високого рівня.

Лінійна структура усієї програми досить рідкісний випадок. Як правило, вона складається як з лінійних (базових) блоків, так і з циклів, умовних, безумовних переходів та викликів процедур і функцій. Така структура програми визначає досить складну структуру графа передачі управління між базовими блоками. Цей граф фактично визначає множину усіх шляхів (історій виконання) від початку до завершення програми. Кожна з таких історій відповідає певним вхідним даним, тобто залежить від них.

Поводження програми – це, власне, сукупність усіх можливих історій виконання. Можна погодитись з тим [4], що відповідно до мети прогнозу оцінки часу виконання програми, її можна виконати:

– для кожного з можливих шляхів виконання з осередненням результатів,
– для деякої сукупності історій,
– для заданої конкретної історії.

При оцінці за другою альтернативою необхідно визначити репрезентативну сукупність історій [5] або оцінити тільки два шляхи на графі програми, що відповідають найменшому та найбільшому часу виконання програми. При цьому визначення шляхів, що визначають верхню та нижню оцінки, виконується статичним методом аналізу графа управління. Недоліком цього методу є те, що кількість ітерацій у циклах, а також граф виклику функцій тісно пов'язані із вхідними даними. Тому у [6] запропоновані обмеження на структуру програми, яка повинна мати цикли з відомими (або обчисленими статистично) кількістю повторень, та не допускати рекурсивного виклику процедур.

Оберемо модель визначення сукупності історій виконання програм з осередненням історій. Однією з таких моделей, що відповідає умовам, сформульованим у [6], є модель, що ставить у відповідність блок-схемі алгоритму програми орієнтований граф G(i), у якому вузли і співставленні лінійним блокам і розгалуженням, а орієнтованим ребрам – шляхи переходу алгоритму між окремими блоками[7]. При цьому вважається, що процесу виконання програми відповідає перехід з одного вузла до іншого, а знаходженню у певному вузлі – виконання команди переходу і (або) лінійного блоку програми.

Відповідно до особливостей виконання програм можна вважати, що переходи процесу виконання програми із вершини і в деяку вершину j не залежить від того, яким чином програма потрапила у вершину (стан) і. Окрім того, у графі G(i) завжди можна визначити деяку початкову вершину (початок алгоритму) і кінцеву (кінець алгоритму).

Ці особливості виконання програми дають можливість інтерпретувати її виконання як випадковий марківський процес S(t) з дискретними станами і дискретним часом, який має початковий стан (початок програми) та поглинаючий стан, що відповідає завершенню програми.

Граф цього випадкового процесу співпадає з графом. При цьому перехід із стану і в стан j не залежить G(i) від того, яким чином він потрапив у стан і, а тільки залежить від перехідних імовірностей p_{ij} , якими навантажують відповідні ребра графа G(i). Як відомо [8] випадковий процес S(t) переходить із стану в стан у наперед визначені моменти часу, що називають кроками такого процесу. Кожний крок можна пов'язати з часом виконання відповідного лінійного блоку - τ_i [7].

Оскільки система $S(t_k)$ у довільний момент часу може перебувати тільки в одному із станів $S_1,S_2,\ldots S_n$, то при кожному k=1,2… події $S_1(k),\ldots S_n(k)$ несумісні і утворюють повну групу подій.

Основними характеристиками марківських ланцюгів з дискретним часом і дискретним станом є імовірність станів $P_i(k)=P(S_i(k))$ та імовірності переходів P_{ij} , що утворюють квадратну матрицю переходів P порядку n, де n - кількість станів.

$$P=\begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1n} \\ P_{21} & P_{22} & \cdots & P_{2n} \\ P_{n1} & P_{n2} & \cdots & P_{nn} \end{bmatrix}; \qquad (1)$$

$$\sum_{j=1}^{n} P_{ij} = 1 \qquad (2)$$

для кожного кроку K.

Час перебування τ_i у стані S_i пов'язують із часом виконання відповідного лінійного блоку програми, а імовірності P_{ij} попередньо обчислюють у

залежності від статистики виконання переходів у програмі. Наприклад, якщо тіло циклу виконується 9 разів, то відповідні імовірності розгалуження будуть дорівнювати 0,9 та 0,1 і т.д.

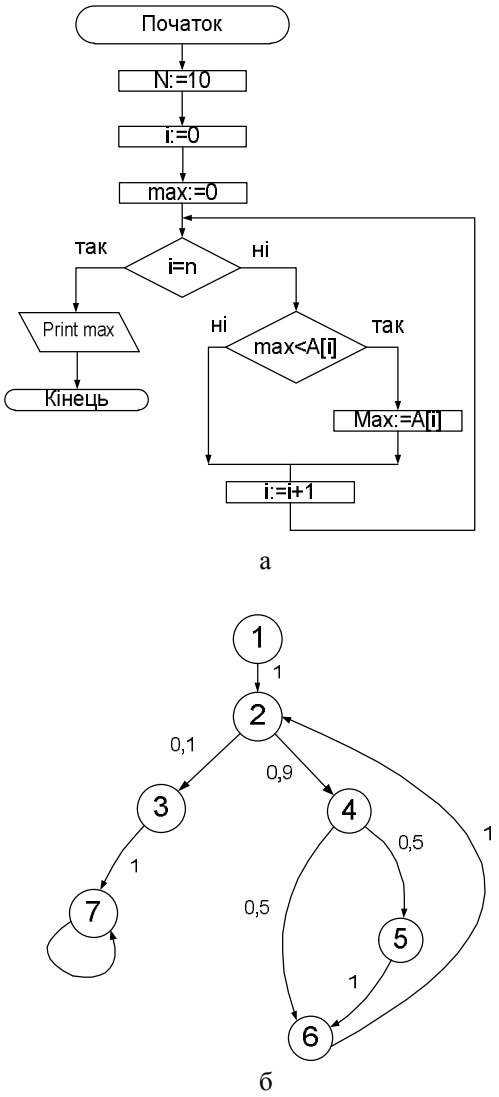

Рис. 1. Блок-схема алгоритму програми та можлива її інтерпретація графом G(i)

На рис. 1 наведено приклад блок-схеми алгоритму та його можлива інтерпретація графом G(i). Як це показано у [7], за допомогою наведеної моделі випадкового процесу можна обчислити середній час знаходження випадкового процесу у групі станів Si до переходу у поглинаючий стан S_n, тобто середній час виконання задачі T сумарну кількість кроків k до завершення програми, сумарну кількість попадання процесу $S(t_k)$ у кожний із станів S_i - K_i . Для такого обчислення попередньо необхідно визначити час виконання кожного з лінійних блоків τ_i . Будемо вважати, що текст програми заданий на мові високого рівня.

Найпоширинішими методами виміру часу фрагментів програми є безпосереднє вимірювання [1].

Наприклад, рекомендовано використання рахівника TSC (Time Stamp Counter), 64 бітного MSR регістру (Model Specific Register), що входить до складу сучасних мікропроцесорів. Цей рахівник може бути ефективно використаний для виміру фрагменту програми із високою точністю (до 50-100 тактів або 10-20нс). Конкретні приклади використання методу з використання мов високого рівня Delfi та C++ наведені у [1] .

На рис. 2 наведено інший приклад запропонованого тут методу, пов'язаного з використанням F: High Rezolution Timer/timer.c, що дозволяє виміряти час час фрагмента програми на мові високого рівня C з використанням Win API- функцій: QueryPerfomanceCounter() та QueryPerfomance-Freguency().

```
1  //high resolution timer
2  #include <windows.h>
3  #include <stdio.h>
4  #include <math.h>
5
6  int main(){
7      LARGE_INTEGER begin, end, freq;
8      double result;
9      int i;
10     QueryPerformanceCounter(&begin);
11
12     //do something();
13
14     for (i = 0; i < 10000000; i++)
15         sqrt(sqrt(sqrt((3.14))));
16     QueryPerformanceCounter(&end);
17     QueryPerformanceFrequency(&freq);
18
19     result = (end.QuadPart - begin.QuadPart) / (double) freq.QuadPart;
20     printf("%s : %4.2f ms\n", "label:", result * 1000);
21 }
```

Рис. 2. Приклад використання high Resolution

Timer для виміру часу виконання фрагменту програми

Висновки

Запропонований метод оцінки часу використання програм може бути ефективно застосовано для визначення середнього часу виконання програм у монопольному режимі.

Литература

1. Методика измерения времени выполнения заданного фрагмента программы [Электронный ресурс]. – Режим доступа: http//evatutin.narod.ru.

2. Калиниченко, Д. В. Методы и средства прогнозирования времени выполнения последовательных программ [Текст] / Д. В. Калиниченко, А. П. Капитонова, П. В. Ющенко // Методы математического моделирования. – М. : МГУ, 1997. – № 2.

3. Davidson, N. W. Relating Static and Dyanamic Machine Code Measurments [Text] / N. W. Davidson, N. R. Rabung, D. B. Whalley // IEETrans. on Computers. – Apr. 1992. – V. 41, № 4. – P. 444 – 454.

4. Капитонова, А. П. Методы и средства прогнозирования времени выполнения последовательных фрагментов программ на вычислителях с различной архитектурой [Текст] : дис. на соискание ученой степени канд. физ.–мат. наук. МГУ им. М.В. Ломоносова. – М., 1997. – 99 с.

5. Nilson, K. D. Worst–Case Execution Time Analysison Modern Processors [Text] / K. D. Nilson, B. Rygg // Second ACM SIGPLAN Workshop on Zanguages Compilers and Tools for Real–Time Systems, June, 1995.

6. Pushnir, P. Calculating the Maximum Execution Time of Real–Time Programs [Text] / P. Pushnir, C. H. Koza // The International Jornal of Time–Critical Computer Systems. – 1989. – V. 1(2). – P. 159 – 176.

7. Зайцев, В. Г. Математична модель для оцінки часу виконання програм [Текст] / В. Г. Зайцев, М. В. Плахотний, Є. І. Цибаєв // Управління розвитком складних систем. – 2013. – Вип. 15. – С. 126 – 130.

8. Романовский, Дискретный анализ.Учебное пособие для студентов [Текст] / Романовский. – 3–е изд. – СПб. : «Невский диалект», БХВ Петрбург, 2003. – 320 с.

Поступила в редакцию 19.02.2014, рассмотрена на редколлегии 25.03.2014

Рецензент: д-р техн. наук, проф. О. В. Поморова, Хмельницкий национальный университет, Хмельницкий, Украина.

ОЦЕНКА ВРЕМЕНИ ВЫПОЛНЕНИЯ ПРОГРАММ

Е. И. Цибаев, В. Г. Зайцев, М. В. Плахотный

Прогнозирование времени выполнения программы связано с использованием целого ряда статических и статистических методов оценки, а также с имитационным моделированием работы устройств вычислительной системы. В данной статье рассматривается задача оценки времени выполнения программ для специализированных компьютерных систем. Предложенная модель работы программы, в виде поглощающей Марковской цепи с дискретными состояниями и дискретным временем. Модель учитывает совокупность историй выполнения программы и использует метод оценки времени выполнения линейных блоков программы, запрограммированные на языке высокого уровня.Приведенная блок-схема алгоритма программы и возможная ее интерпретация графом, а также пример ее применения.

Ключевые слова: поглощающая Марковская цепь, истории выполнения программы, линейный блок программы, прогнозирования времени выполнения программы.

EVALUATION OF PROGRAM EXECUTION TIME

E. I. Tsybaev, V. G. Zaitsev, M. V. Plakhotny

Predicting execution time associated with the use of a number of static and statistical evaluation methods and simulation modeling of the devices of the computer system as well. In thir article we consider problem of estimating execution time of programs for specialized computer systems. The proposed model of work program in absorbing Markov chain form with discrete states and discrete time. The model takes into account totality of stories run program uses method of evaluation of execution time of a linear block of code programmed in high level language. Also it shows a block diagram of the algorithm of the program, its possible interpretation of the graph and an example of its use.

Key words: absorbing Markov chain, run history of program, linear block program, time prediction of program execution.

Цибаев Евгений Игоревич – аспирант кафедры Специализированных компьютерных систем и системного программирования Национального Технического Университета Украины «Киевский Политехнический Институт», Киев, e-mail: etsybaev@gmail.com.

Зайцев Владимир Григорьевич – д-р техн. наук, профессор, профессор кафедры Специализированных компьютерных систем и системного программирования Национального Технического Университета Украины «Киевский Политехнический Институт», Киев, e-mail: v_zaitsev@bigmir.net.

Плахотный Николай Викторович – канд. техн. наук, доцент, доцент кафедры Специализированных компьютерных систем и системного программирования Национального Технического Университета Украины «Киевский Политехнический Институт», Киев.