Bab 5
Program Benchmark

Ref. [LILJA] Chap 7

Dr. Yeffry Handoko Putra

PROGRAM BENCHMARK:

- Program pengujian kinerja komputer, karakteristik kinerja komputer diukur saat menjalankannya
- Program yang digunakan sebagai standar pembanding kinerja untuk tujuan peningkatan mutu
- Program Benchmark yang baik adalah program yang mudah digunakan pada sistem yang berbeda

JENIS-JENIS PROGRAM BENCHMARK

1. Waktu Eksekusi dari Intruksi Tunggal (Single Instruction Execution Time)→ *Early Day Computing*
2. Intruksi eksekusi campuran (Instruction Execution Mixes)

1. WAKTU EKSEKUSI DARI INTRUKSI TUNGGAL (SINGLE INSTRUCTION EXECUTION TIME)

- Intruksi sederhana yang digunakan adalah penjumlahan (addition)
- Komputer tercepat yang menyelesaikan adalah komputer terbaik

2. INTRUKSI EKSEKUSI CAMPURAN (INSTRUCTION EXECUTION MIXES)

- Komputer diuji dengan mengeksekusi beberapa intruksi campuran.
  Fakta: Intruksi yang menggunakan memori utama akan lebih lama dari intruksi yang menggunakan register
- Intruksi Campuran Gibson (1950, Jack C. Gibson), digunakan pada IBM 704 dan 650. Berdasarkan karakterisik IBM 704 dan 650 diberikan pembobotan spesifik (CPI=*Cycle Per Intruction*).

Waktu eksekusi dari N instruksi :

$$T_{program} = N \; x \; CPI \; x \; T_{clock}$$

INSTRUKSI CAMPURAN GIBSON (13 INSTRUKSI)

No	Jenis Instruksi	Fraksi Instruksi Eksekusi (%)
1.	Load and Store	31,2
2.	Fixed-Point Add and Substract	6,1
3.	Compares	3,8
4.	Branches	16,6
5.	Floating add and substract	6,9
6.	Floating Multiply	3,8
7.	Floating Devide	1,5
8.	Fixed-Point Multiply	0,6
9.	Fixed-Point Devide	0,2
10.	Shifting	4,4
11.	Logical, And, Or	1,6
12.	Instruction not using registers	5,3
13.	Indexing	18,0
	Total	100

Contoh Perhitungan $T_{Program}$ pada komputer dengan Tclock= 8 ns ; Jumlah Intruksi: 23.842.128 dengan persentase

Jenis Instruksi	Fraksi Instruksi Eksekusi(%)	Clock yang diperlukan
1	33,4	2
2	23,2	3
3	18,1	3
4	10,3	4
5	7,8	5
6	7,2	7

Contoh Perhitungan $T_{Program}$ pada komputer dengan Tclock= 8 ns ; Jumlah Intruksi: 23.842.128 dengan persentase

Jenis Instruksi	Fraksi Instruksi Eksekusi(%)	Clock yang diperlukan
1	33,4	2
2	23,2	3
3	18,1	3
4	10,3	4
5	7,8	5
6	7,2	7

CPI = 2(0,334) + 3(0,232) + 3(0,181) + 4(0,103) + 5(0,078) + 7(0,072)
=3,213

Tprogram = 23.842.128 x 3,213 x 8 x 10-9 = 0,61 s

SAAT INI INTRUKSI CAMPURAN TIDAK DIGUNAKAN

- Saat ini komputer menyediakan banyak kelas instruksi yang kompleks yang waktu instruksi rata-ratanya tidak berkaitan dengan instruksi campuran
- Intruction Campuran hanya merepresentasikan kinerja prosesor saja tidak seluruh sistem komputer
- Sistem komputer modern, waktu instruksi tergantung pada:
  - Mode Pengalamatan (Addresing modes)
  - Laju perolehan di memori cache (cache hit rates)
  - Efisiensi Pipeline (pipeline efficiency),
  - Interferensi dari peralatan lain saat siklus akses prosesor ke memori

3. PROGRAM BENCHMARK BUATAN (SYNTETHIC BENCHMARK)

- Benchmark buatan adalah program yang dibuat sesuai dengan yang diharapkan oleh intruksi Gibson dengan mengatur jumlah permintaan dan mengukur layanan OS : Process creation, forking, memory allocation
- Benchmark Buatan Populer:
  - **Sieve**
  - **Ackermann's Function**
  - **Whetstone**
  - **LINPACK**
  - **Dhrystone**
  - **Lawrence Livermore Loops**
  - **Debit-Credit Benchmark**
  - **SPEC Benchmark Suite**

Sieve

- Sieve Kernel digunakan untuk membandingkan microprosesor, PC dan High-level language
- Didasari oleh Eratosthenes's sieve algorithm yang digunakan untuk mencari bilangan prima yang lebih kecil dari bilangan n
- Algoritma ini secara manual menuliskan semua bilangan integer dari 1 sampai n lalu menghapus kelipatan dari k untuk k = 2, 3, .., $\sqrt{n}$

- *Contoh :*

1. Write down all numbers from 1 to 20. Mark all as prime:

[1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11]

[12] [13] [14] [15] [16] [17] [18] [19] [20]

2. Remove all multiples of 2 from the list of primes:

[1] [2] [3] 4 [5] 6 [7] 8 [9] 10 [11] 12 [13] 14 [15] 16 [17] 18 [19] 20

3. The next integer in the sequence is 3. Remove all multiples of 3:

[1] [2] [3] 4 [5] 6 [7] 8 9 10 [11] 12 [13] 14 15 16 [17] 18 [19] 20

4. The next integer in the sequence is 5, which is greater than the square root of 20. Hence, the remaining sequence consists of the desired prime numbers

Ackermann's Function

- Kernel ini digunakan untuk menilai efisiensi dari mekanisme pemanggilan prosedur dalam ALGOL-like languages.
- Fungsi Ackermann terdiri dari dua parameter dan didefinisikan rekursif. Fungsi Ackermann (3,*n*) mengevaluasi nilai n dari 1 sampai 6
- Nilai dari fungsi Ackermann (3,*n*) adalah $2^{n+3} - 3$. Fungsi ini digunakan untuk memverifikasi benchmark
- Kemudian benchmark yang dihasilkan adalah :
  - The average execution time per call
  - the number of instructions executed per call
  - the amount of stack space required for each call

- Jumlah pemanggilan rekursif dari Ackermann (3,2) dinyatakan dengan (Wichmann ,1976) :

$$(512 \times 4^{n-1} - 15 \times 2^{n+3} + 9n + 37)/3$$

This expression is used to compute the execution time per call. For Ackermann (3,n), the maximum depth of the procedure calls is $2^{n+3} - 4$. Hence, the amount of stack space required doubles when n is increased by 1.

Whetstone Benchmark

- Synthetic Benchmark untuk mengevaluasi kinerja komputer, mula-mula dikembangkan pada National Physical Lab., UK. 1972 dengan bahasa Algol 60.
- Whetstone Benchmark terdiri dari 11 modules
- Selain sebagai syntehtic mix, Whetstone digunakan sebagai floating point benchmark pada cache memory. Namun hanya bertujuan pada optimisasi compiler sehingga parameter input I/O tidak mempengaruhi kinerja yang diukur
- Kernel Exercises whetstone (processor feature):
  - Array addressing
  - Fixed and floating point arithmetic

  - Subroutine calls
  - Parameter passing
- Whetstone kernel telah diterjemahkan ke FORTRAN, PL/I, dll

Whetstone benchmark diukur dalam KWIPS (kilo Whetstone Instruction per second) juga MWIPS (Million Whestone Instruction per second)

LINPACK Benchmark

- Dikembangkan oleh Jack Dongarra (1983) dari Argonne National Laboratory.
- Benchmark LINPACK terdiri dari sejumlah program yang menyelesaikan persamaan linier rumit menggunakan UNPACK subroutine. UNPACK berisi penjumlahan floating point dan perkalian.
- Contoh subroutinenya yang menyita waktu (time consumed) adalah BLAS (Basic Linear Algebra Subroutines)
- Diukur dalam MFLOPs

Popular for:

– 100x100 system of equations

- 1000x1000 system of equations

Represent:

- finite element analysis, simulation
- calls for high computation speed

graphics processing

Dhrystone Benchmark
pun from whetstone

- Synthetic computing benchmark yang dikembangkan oleh Reinhold P. Weicker pada Siemens [1984] dengan tujuan merepresentasikan pemrograman integer
- Kernel memiliki banyak procedur dan pemanggilan (call procedure) yang percabangannya rendah (low dynamic nesting) dan jumlah instruction per function call yang rendah tetapi waktu komputasi banyak digunakan untuk meng-copy character-string dan membandingkannya
- Weicker mengumpulkan meta-data dari broad range of software, kemudian program-program itu dikarakteristikan dalam bentuk umum: procedure call, pointer indirection, assignment, etc.

- Dalam bahasa C, Ada and Pascal
- Hasil dinyatakan dengan DIPS (Dhrystone Instructions Per Second)
- Benchmark ini digunakan untuk mengukur integer performance

Lawrence Livermore Loops

- Workload terdiri dari 24 pengujian terpisah yang didominasi oleh perhitungan vektor.
- Diawali di Lawrence Livermore National Laboratories, kemudian dijalankan pada beberapa sistem dari superkomputer sampai PC
- Benchmark yang dihasilkan tidak pernah nilai tunggal karena itu nilai yang diambil berupa maximum, minimum dan tiga mean (Arithmetic, geometric dan harmonic)
- Hasilnya dinyatakan dalam MFLOPS (Million of Floating Point Operation per second)
- Contoh pada :

- Large Scale science application: floating point calculation in single and double precision arithmetic
- Large Scale computational fluid dynamic, astrophysic
- Monte Carlo Simulation

Debit-Credit Benchmark

- Berupa benchmark berlevel aplikasi sebagai standar Transaction Prrocessing System sejak 1973 sering diserbut TPC Benchmark

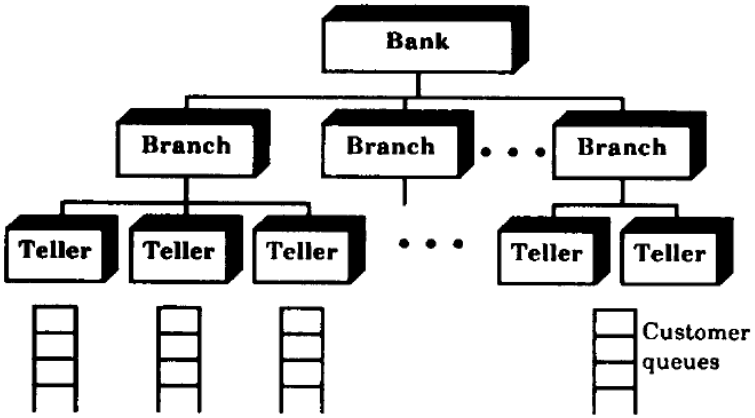

FIGURE 4.4 Banking environment.

```
Begin-Transaction
      Read message from the terminal (100 bytes)
      Rewrite account                (100 bytes, random)
      Write history                  (50 bytes, sequential)
      Rewrite teller                 (100 bytes, random)
      Rewrite branch                 (100 bytes, random)
      Write message to the terminal  (200 bytes)
Commit-Transaction
```

FIGURE 4.5 Debit-credit transaction pseudo-code.

SPEC Suite 1992, 1995, 2000...
(System Performance Evaluation Cooperative)

1. a nonprofit corporation formed by leading computer vendors to develop a standardized set of benchmarks.
2. Release 1.0 of the SPEC benchmark suite (SPEC 1990) consist of 10 benchmark submitted by scientists and engineers.
    1. *GCC*: The time for the GNU C Compiler to convert 19 preprocessed source files into assembly language output is measured. This benchmark is representative of a software engineering environment and measures the compiling efficiency of a system.
    2. *Espresso*: Espresso is an Electronic Design Automation (EDA) tool that performs heuristic boolean

function minimization for Programmable Logic Arrays (PLAs). The elapsed time to run a set of seven input models is measured.

3. *Spice 2g6*: Spice, another representative of the EDA environment, is a widely used analog circuit simulation tool. The time to simulate a bipolar circuit is measured.
4. *Doduc*: This is a synthetic benchmark that performs a Monte Carlo simulation of certain aspects of a nuclear reactor. Because of its iterative structure and abundance of short branches and compact loops, it tests the cache memory effectiveness.
5. *NASA7*: This is a collection of seven floating-point intensive kernels performing matrix operations on double-precision data.

6. *LI*: The elapsed time to solve the popular 9-queens problem by the LISP interpreter is measured.
7. *Eqntom*: This benchmark translates a logical representation of a boolean equation to a truth table.
8. *Matrix300*: This performs various matrix operations using several LINPACK routines on matrices of size 300 × 300. The code uses double-precision floating-point arithmetic and is highly vectorizable.
9. *Fpppp*: This is a quantum chemistry benchmark that performs two electron integral derivatives using double-precision floating-point FORTRAN. It is difficult to vectorize.

10. *Tomcatv*: This is a vectorized mesh generation program using double-precision floating-point FORTRAN. Since it is highly vectorizable, substantial speedups have been observed on several shared-memory multiprocessor systems.