Architettura degli Elaboratori

Lezione 17 – Calcolo delle prestazioni

Giuseppe Cota
Dipartimento di Scienze Matematiche Fisiche e Informatiche
Università degli Studi di Parma

Indice

- Indici per il calcolo delle prestazioni
- Legge di Amdahl

Indici per il calcolo delle prestazioni

Primo indice: tempo richiesto per l'esecuzione di un programma

- $\boldsymbol{T_{CPU}}$ tempo di CPU richiesto per l'esecuzione del programma:

$$T_{CPU} = \frac{N}{f}$$

- $\boldsymbol{N}$ numero totale di cicli di clock consumati nell'esecuzione del programma
- $\boldsymbol{f}$ frequenza di clock $(= 1/T, T$ periodo di clock)
- $\boldsymbol{N_{ist}}$ numero totale di istruzioni eseguite
- $\boldsymbol{C_{PI}}$ numero medio di cicli di clock per istruzione macchina

$$C_{PI} = \frac{N}{N_{ist}}$$

- In totale si ha:

$$T_{CPU} = \frac{N}{f} = \frac{(N_{ist} \cdot CPI)}{f} = N_{ist} \cdot C_{PI} \cdot T$$

Analisi della formula del tempo di esecuzione di un programma

- **Formula:**

$$T_{CPU} = \frac{N}{f} = \frac{(N_{ist} \cdot C_{PI})}{f} = N_{ist} \cdot C_{PI} \cdot T$$

- $\boldsymbol{N_{ist}}$ (numero di istruzioni) dipende dal repertorio di istruzioni e dal grado di ottimizzazione del compilatore.
  - **Un repertorio CISC dovrebbe ridurre N_{ist}.** Tendenzialmente con CISC uso meno istruzioni macchina per un programma.
- $\boldsymbol{T}$ (periodo di clock) dipende dalla tecnologia e dall'architettura
  - **Un repertorio RISC favorisce un basso T**. Eseguire un'istruzione RISC richiede meno tempo di un'istruzione CISC
- $\boldsymbol{C_{PI}}$ (cicli di clock per istruzione) dipende dal repertorio di istruzioni e dall'architettura
  - **Un repertorio RISC favorisce un basso C_{PI}.**
  - Soluzioni architettoniche come la pipeline permette di avere $C_{PI} \simeq 1$
  - Con soluzioni parallele (processori multicore) ho $C_{PI} < 1$

Indice MIPS

- **MIPS: milioni di istruzioni per secondo**

$$MIPS = \frac{N_{ist}}{T_e \cdot 10^6}$$

N_{ist} : numero di istruzioni eseguite
T_e : tempo di esecuzione del programma in secondi

- Dipende dal repertorio di istruzioni della CPU, per cui non è possibile fare confronti tra calcolatori con repertori differenti.
    - Istruzioni CISC ≠ Istruzioni RISC

Indice MFLOPS

- **MFLOPS: milioni di operazioni in virgola mobile per secondo**

$$MFLOPS = \frac{N_{vm}}{T_e \cdot 10^6}$$

N_{vm} : numero di operazioni in virgola mobile eseguite
T_e : tempo di esecuzione del programma in secondi

- Un po' più affidabile di MIPS in quanto considera il numero di operazioni e non di istruzioni.
  - Si basa sul presupposto che lo stesso programma esegua lo stesso numero di operazioni in virgola mobile…
  - … ma il repertorio delle istruzioni per le operazioni in virgola mobile su macchine diverse non è lo stesso.

Misura delle prestazioni

- Le prestazioni complessive di un sistema di elaborazione non dipendono solo dalla CPU, ma anche dalla memoria e dal sottosistema I/O.
- Uno dei modi per misurare le prestazioni di un sistema di elaborazione è tramite l'utilizzo di *programmi campione* chiamati ***benchmark***.
- **Tipi di benchmark:**
  - ***reali***: programma reali;
  - ***ridotti***: codice estratto da programmi reali:
  - ***sintetici***: simulano le operazioni frequenti del sistema:
  - ***suite***: gruppi di benchmark.

Legge di Amdahl

Legge di Amdahl

- Legge definita nel 1967 da Gene Amdahl (ingegnere IBM).
- ***In un sistema composto da più componenti il miglioramento di una componente influenza il sistema in proporzione alla frazione di tempo in cui il componente viene utilizzato***.
- Si definisce ***accelerazione*** o ***speedup*** (S) il rapporto tra le prestazioni ottenute con miglioramento e le prestazioni di prima del miglioramento, ossia il rapporto tra il tempo di esecuzione vecchio (t_v) e quello vecchio (t_n)

$$S = \frac{t_v}{t_n}$$

Esempio

Il mio sistema ha $t_v = 20$, apporto un miglioramento e ottengo $t_n = 16$.
Il mio speedup sarà

$$S = \frac{20}{16} = 1.25$$

Legge di Amdahl

- Lo speedup può essere applicato sull'intero sistema, su un componente del sistema o una sua funzionalità.
- Supponiamo di apportare un miglioramento **ad un singolo componente** del sistema.
- $\boldsymbol{f_u}$: frazione di tempo di utilizzo del componente che voglio migliorare, corrisponde alla % di tempo in cui il componente è utilizzato
  - Esempio: se un programma viene eseguito in 40 secondi ed utilizzo il componente da migliorare per 8 secondi allora $f_u = \frac{8}{40} = 0.2$
- $\boldsymbol{f_i}$: % di tempo in cui il componente è inutilizzato
  - $f_i = 1 - f_u$

Allora, se S_c è lo speedup del componente migliorato, il tempo nuovo (t_n) **dell'intero sistema** sarà:

$$t_n = t_v \cdot f_i + t_v \cdot \frac{f_u}{S_c} = t_v \cdot \left[(1 - f_u) + \frac{f_u}{S_c}\right]$$

Legge di Amdahl

- $\boldsymbol{S_{tot}}$: speed up del intero sistema (con un componente migliorato)

$$S_{tot} = \frac{t_v}{t_n} =$$

$$= \frac{t_v}{t_v \cdot \left[(1 - f_u) + \frac{f_u}{S_c}\right]} =$$

$$= \frac{1}{(1 - f_u) + \frac{f_u}{S_c}} =$$

$$= \frac{1}{f_i + \frac{f_u}{S_c}}$$

Legge di Amdahl
Esempio

- Un miglioramento di un componente migliora le sue prestazioni di 100 volte, ma esso viene usato il 10% del tempo.
- Dati:
  - $S_C = 100$
  - $f_u = 0.1 \Rightarrow f_i = 1 - 0.1 = 0.9$
- Mi chiedo quanto vale S_{tot} (speedup dell'intero sistema)?

$$S_{tot} = \frac{1}{f_i + \frac{f_u}{S_C}} = \frac{1}{0.9 + \frac{0.1}{100}} \simeq 1.1$$

- Anche se il componente è 100 volte più veloce il sistema totale è più veloce solo del 10%

- A casa provare cosa ottengo se ho $S_C = 10$ e $f_u = 0.9$

Domande?

Riferimenti principali

- Capitolo 2 di ***Calcolatori elettronici. Architettura e Organizzazione,*** Giacomo Bucci. McGraw-Hill Education, 2017.