

Computer Performance and Multimedia Instructions



Z. Jerry Shi
Department of Computer Science and Engineering
University of Connecticut

CSE268: Microprocessor Laboratory

Execution time

The execution time of a single application can be estimated as:

$$\text{Path Length} \times \text{CPI} \times \text{Cycle Time}$$

Path length: number of instructions needed

CPI: average number of cycles per instruction

Cycle time: length of each cycle

If you have multiple applications, be careful when using a single number to characterize the processor's performance.

The single number may be arithmetic mean, geometric mean, or harmonic mean of the performance of multiple applications.

Arithmetic mean

- Unweighted arithmetic mean

$$A_mean = \sum_{i=1}^n \frac{M_i}{n}$$

- Weighted arithmetic mean

$$A_mean = \sum_{i=1}^n w_i M_i$$

Geometric mean(1)

- Unweighted geometric mean

$$G_mean = \sqrt[n]{\prod_{i=1}^n M_i}$$

- Weighted geometric mean

$$G_mean = \prod_{i=1}^n M_i^{w_i}$$

Geometric mean(2)

- Consistently maintain the performance relationship regardless of the basis
- Does not predict execution time. Thus, may lead to wrong conclusions

Harmonic mean(1)

- Unweighted harmonic mean

$$H_mean = \frac{n}{\sum_{i=1}^n \frac{1}{M_i}}$$

- Weighted harmonic mean

$$H_mean = \frac{n}{\sum_{i=1}^n \frac{w_i}{M_i}}$$

Harmonic mean(2)

- Should be used for summarizing performance expressed as a rate.
- Equivalent to calculating the total number of operations divided by the total time.

$M = F/T$ where M is the rate, and T is the time needed for F operations.

$$H_mean = \frac{n}{\sum_{i=1}^n \frac{1}{M_i}} = \frac{n}{\sum_{i=1}^n \frac{T_i}{F}} = \frac{nF}{\sum_{i=1}^n T_i}$$

Performance characterization with a single number

- Use arithmetic mean for an measure of performance expressed as **time**
- Use harmonic mean for an measure of performance expressed as **rate**
- An aggregate performance measure should be calculated **before** any normalizing is done

Amdahl's law

- The performance improvement to be gained from using some faster mode of execution is limited by the fraction of the time the faster mode can be used
- Speedup is defined as:

$$Speedup = \frac{ExecutionTime_{before_enhancement}}{ExecutionTime_{after_enhancement}}$$

Example:

Using a new method, 40% of an application can execute 10 times faster. The overall speedup can be calculated as:

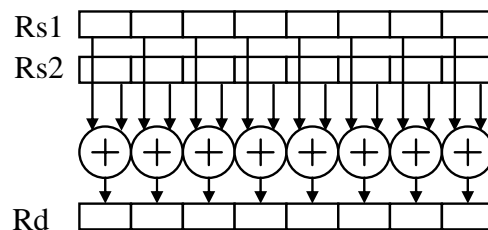
$$Speedup = \frac{1}{0.6 + \frac{0.4}{10}} = \frac{1}{0.64} = 1.56$$

The best speedup you can achieve by optimizing the 40% code is $1/0.6=1.6$

Subword parallelism (microSIMD)

- Multimedia applications use small data items of 8 or 16 bits
- Multiple subwords can be stored in each register
- A single instruction can perform multiple operations

Example: A 64-bit register can store eight bytes. One PADD (parallel add) instruction can eight add operations in one cycle.



Example of subword instructions

Suppose each register can store w subwords.

For each i between $0 \dots w - 1$,

a_i are subwords in Ra,

b_i are subwords in Rb,

and d_i are subwords in Rd.

| | | |
|----------|------------|--|
| PADD | Rd, Ra, Rb | ; parallel add : $d_i = a_i + b_i$ |
| PSUB | Rd, Ra, Rb | ; parallel sub : $d_i = a_i - b_i$ |
| PMIN | Rd, Ra, Rb | ; parallel minimum: $d_i = \min(a_i, b_i)$ |
| PMAX | Rd, Ra, Rb | ; parallel maximum: $d_i = \max(a_i, b_i)$ |
| PCMP.rel | Rd, Ra, Rb | ; parallel compare: $d_i = (a_i \text{ rel } b_i)? 1 \dots 1 : 0 \dots 0$; where rel can be gt, lt, ge, etc. |

Case study: median value

- When n is odd, the median value of n elements is the middle element after they are sorted
- Median filter is used in image processing applications for the smoothing of signals, suppression of impulse noise, and preserving of edges

| | | | | |
|-----|-----|-----|-----|-----|
| 123 | 123 | 126 | 130 | 140 |
| 122 | 124 | 126 | 127 | 135 |
| 118 | 120 | 130 | 125 | 134 |
| 119 | 115 | 119 | 123 | 133 |
| 111 | 116 | 110 | 120 | 130 |

Neighbourhood values:

115, 119, 120, 123, 124,
125, 126, 127, 130

Median value: 124

Courtesy of Department of AI, University of Edinburgh, U.K.

Finding the median value

- How to find out the median value?
 - Sort all the elements
 - Pick the one in the middle
 - Only consider the cases where n is odd
- Bubble sort
 - The basic operation is `sort_swap(x, y)`
 - If $x > y$, then swap (x, y)
 - Sorting n elements needs $n(n - 1)/2$ `sort_swap`'s
 - Sum of $1, 2, \dots, n - 1$
 - Do not have to sort all elements to find the median value

Performing `sort_swap` with PMIN and PMAX

- `sort_swap(x, y)` can be done with 2 instructions with register renaming
 - No branch is needed

PMIN R11, R1, R2

PMAX R12, R1, R2

; suppose $R1 = x$ and $R2 = y$, all subwords in $R1$ and $R2$ are sorted simultaneously

; use $R11$ as x and $R12$ as y in following code

The median value of 3 numbers

; Suppose multiple sets of 3 numbers are stored in R1, R2, R3
; Use bubble sort to sort three numbers

```
sort_swap(R1, R2)
sort_swap(R2, R3) ; R3 has the largest numbers
sort_swap(R1, R2) ; R2 has the median values
```

; R2 has 8 median values after three operations
; 6 instructions at most

No need to do full sorting when n is large

Suppose we need to find the median value of n numbers (n is odd)

If a number is greater or less than $(n + 1)/2$ numbers, it can not be the median value.

Steps to find the median value of 9 numbers

| Step | | Description | # of ops |
|------|--|------------------|----------|
| 1 | $(L0, M0, H0) = \text{SORT}(V0, V1, V2)$ | $L0 < M0 < H0$ | 3 |
| 2 | $(L1, M1, H1) = \text{SORT}(V3, V4, V5)$ | $L1 < M1 < H1$ | 3 |
| 3 | $(L2, M2, H2) = \text{SORT}(V6, V7, V8)$ | $L2 < M2 < H2$ | 3 |
| 4 | $L = \text{MAX}(L0, L1, L2)$ | Exclude 2 values | 2 |
| 5 | $M = \text{Median3}(M0, M1, M2)$ | Exclude 2 values | 3 |
| 6 | $H = \text{MIN}(H0, H1, H2)$ | Exclude 2 values | 2 |
| 7 | $M = \text{Median3}(L, M, H)$ | | 3 |

There is a total of 19 operations

Steps to find the median value of 9 numbers (2)

- Step 4: $L = \text{MAX}(L0, L1, L2)$

WLOG, we assume $L = L0$

$L1$ and $L2$ can be excluded because:

$L1 < M1 < H1$ and $L1 < L0 < M0 < H0$

$L2 < M2 < H2$ and $L2 < L0 < M0 < H0$

$$(9 + 1)/2 = 5$$

Steps to find the median value of 9 numbers (3)

- Step 5: $M = \text{Median3}(M_0, M_1, M_2)$

WLOG, we assume $M = M_0$, and $M_1 < M_0 < M_2$

M_1 and M_2 can be excluded because:

$M_1 < H_1$ and $M_1 < M_0 < H_0$ and $M_1 < M_2 < H_2$

$M_2 > L_2$ and $M_2 > M_0 > L_0$ and $M_2 > M_1 > L_1$

Steps to find the median value of 9 numbers (4)

- Step 6: $H = \text{MIN}(H_0, H_1, H_2)$

WLOG, we assume $H = H_0$

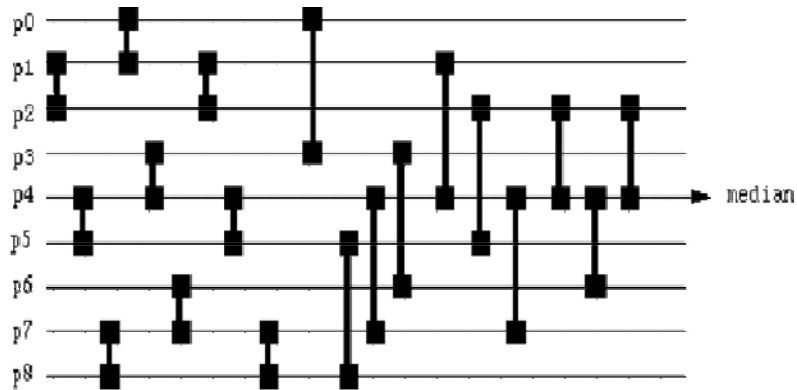
H_1 and H_2 can be excluded because:

$L_1 < M_1 < H_1$ and $L_0 < M_0 < H_0 < H_1$

$L_2 < M_2 < H_2$ and $L_0 < M_0 < H_0 < H_2$

Why can Step 7 find the median value?

Nine values on sorting network



Not all instructions are needed

; Find the median value of three numbers
 ; Suppose sets of 3 values are stored in R1, R2, R3

1. PMIN R11, R1, R2
2. PMAX R12, R1, R2 ; sort_swap(R1, R2)
3. PMAX R13, R12, R3
4. PMIN R12, R12, R3 ; sort_swap(R2, R3)
5. PMAX R2, R11, R12
6. PMIN R1, R11, R12 ; sort_swap(R1, R2)

; Instruction 3 and 6 can be removed because R13 and R1 are not used

Only 4 instructions, not 6 !!!

Online resources for searching for media values

1. <http://ndevilla.free.fr/median/>
2. <http://www.eso.org/projects/dfs/papers/jitter99/node27.html>