Memory Performance of Algorithms

CSE 373
Data Structures
Lecture 16

Algorithm Performance Factors

- Algorithm choices (asymptotic running time)
  - $O(n^2)$ or $O(n \log n)$...
- Data structure choices
  - Binary heap or linked list priority queue
- Language and Compiler
  - C, C++, Java, Fortran
- Memory performance
  - How near is the data to the processor

Moore’s Law

Processor-Memory Performance Gap

- x86 CPU speed (100x over 10 years)

Levels in the Memory Hierarchy

The Cache

- Direct mapped cache
  - Cache hit: data accessed is in the cache.
  - Cache miss: data accessed is not in the cache.
Memory Blocks

- Addressable unit, usually a byte
- Memory block – unit of memory transferred as a whole from memory to cache. Sometimes called “cache line”. Usually, 32-64 bytes (but growing in size).

Why Memory Blocks

- Time to transfer x bytes is given by $T(x) = a + bx$. ($a$ is latency, $b$ ~ 1/bandwidth)
- Because $a$ is large relative to $b$, it pays to transfer more than one byte at a time.
  - The hope is that bytes near the accessed byte will be accessed soon – good spatial locality.

Locality

- **Spatial locality**: addresses near a recently accessed byte are accessed also.
- **Temporal locality**: the same address that was accessed recently is accessed again.

Examples of Locality

- **Good spatial locality**
  - Quicksort – the array is scanned
  - Poor spatial locality
  - Binary search – jump around the array

Examples of locality

- **Good temporal locality**
  - For loop index $i$ in a tight loop.
    
    ```
    for i = 1 to n do {...}
    ```
- **Poor temporal locality**
  - Repeated long scans that exceed the cache size, like in iterative merge sort.

Classifying Cache Misses

- **Compulsory misses** – first time a block is accessed
  - Can never be avoided
- **Capacity misses** – data structure does not fit in cache
  - Can be avoided by algorithmic design.
- **Conflict misses** – several accessed blocks map to the same location in cache
  - Conflict misses are not much of a problem because modern caches are set associative.
Set Associative Cache

- Two-way set associative cache
- Two blocks of the cache can hold blocks from the same parts of memory
- Replacement policy needed.
- Reduces conflict misses

Cache Misses for Scans

In cache

Not in cache

1/B misses per access where B is number of access per line

Repeated Long Scans

- Have good spatial locality
- Poor temporal locality
- If there are B accesses per memory block then 1/B of the accesses are cache misses.

Iterative Mergesort

Recursive Mergesort
Recursive Mergesort

Multi-Mergesort

Multi-Mergesort Cache Behavior

Quicksort

Radix sort

Sorting Study from 1996

- Compared sorting algorithms
  - Cache misses
  - Instruction count
  - Execution time
- The study is still valid today, because the gap between processor speed and memory speed is even larger.
Algorithms

- Iterative mergesort
- Multi-mergesort
- Quicksort
- Heapsort
- Radix sort
  - Parameters chosen for large data set.
  - 4 passes for 64 bit integers.

Notes on Memory Performance

- Memory performance may matter.
- Tips
  - Sacrifice instructions to get better cache performance.
  - Smaller memory footprint is good.
  - Divide and conquer is good.
  - Processing data into cache sized pieces is good.
  - Fully utilize memory blocks if possible
    - Short scans are good.
    - Multilane trees are good.