Characterizing read Latency on different Levels of Cache Hierarchy

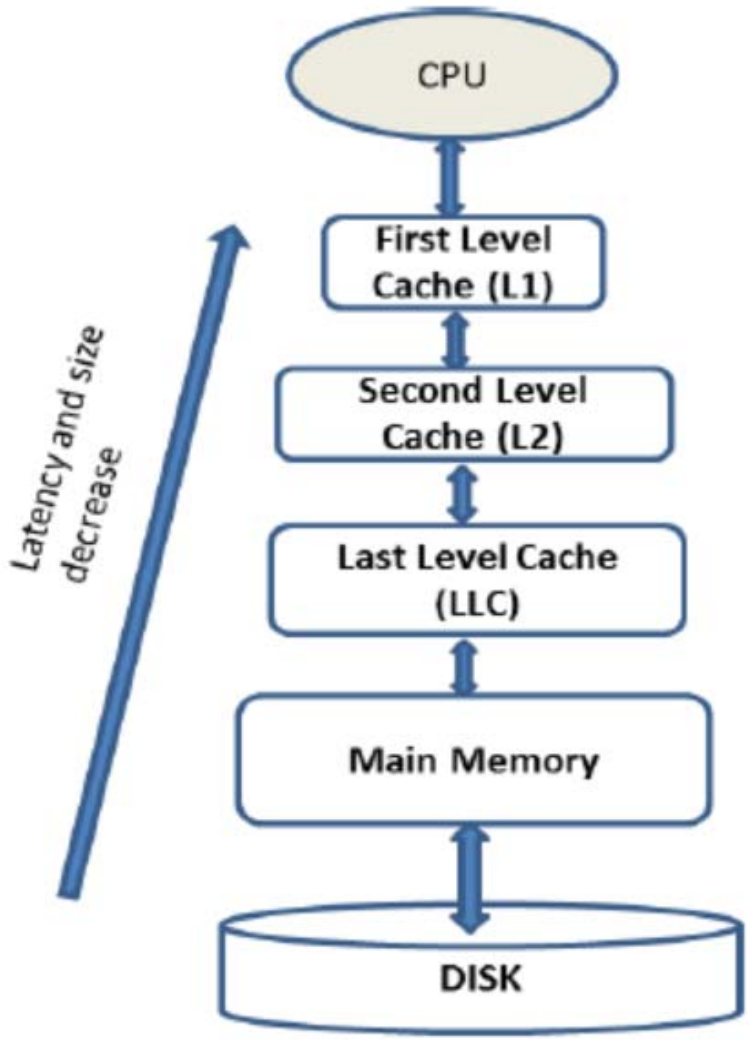
Lorena Ndreu

University of Cyprus

06/03/2017

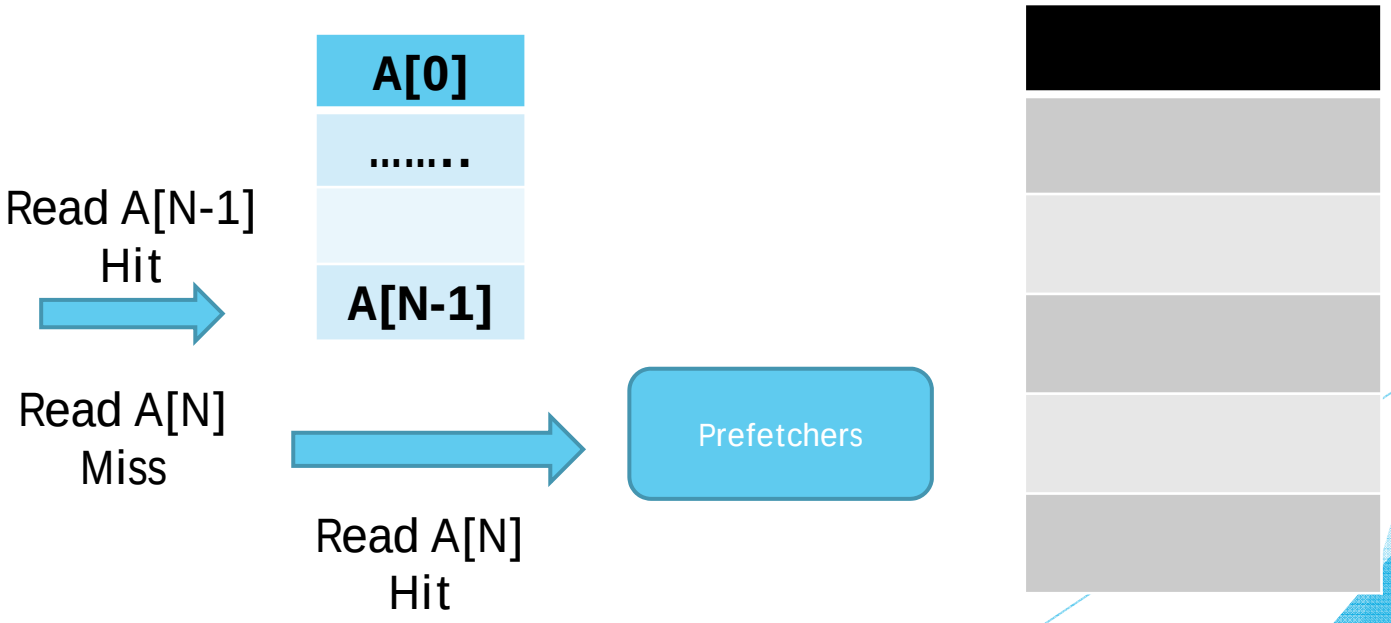
www.cs.ucy.ac.cy/carch/xi/

Memory Hierarchy



What to consider while characterizing Cache Hierarchy

- The benefits from prefetching
  - Bringing data or instructions from memory, into the cache, before they are needed
  - Reduce the latency in case that the data is accessed
- The benefits from Cache locality. Temporal & Spatial
- Assume an array of N blocks and L1 Data cache size of N Blocks



What to consider while characterizing Cache Hierarchy

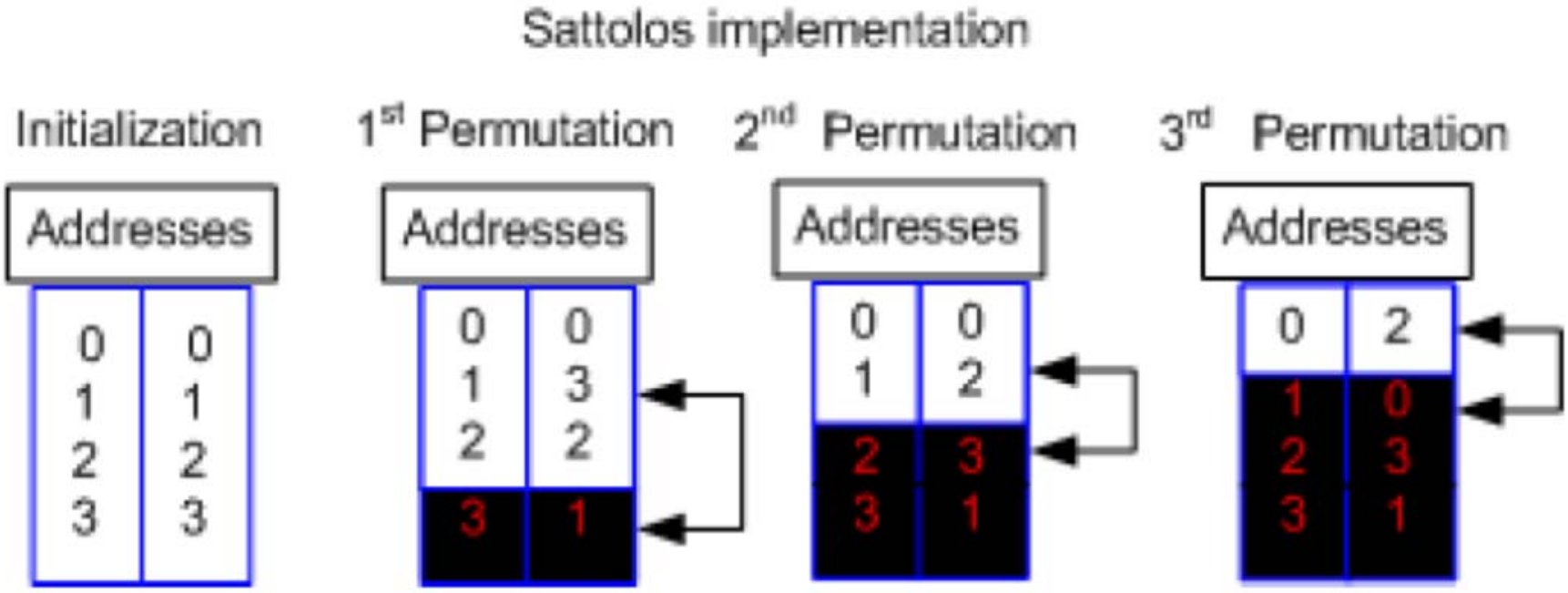
- The program used for characterizing the latency of different levels of Caches
  - Should allocate and perform random accesses to an array
    - in such a way that it should access all the distinct addresses of an array before accessing the same address twice.
  - Performing such random accesses is important because it "disables" Prefetching & Cache Locality

Sattolo's Algorithm to characterize the read access latency of different levels of caches

- Sattolo's algorithm allows performing
  - Random number generation of a cyclic permutation on a fixed number of elements.
  - The basic idea of this algorithm is that giving an array of size N (Bytes etc.)
    - it permutes array entries using random cyclic permutations to produce access patterns.
    - each array entry is accessed exactly once before the access is repeated, in a random order.

Sattolo's Algorithm to characterize the read access latency of different levels of caches

- The trick on using Sattolo's algorithm is in
  - initializing the array to contain pointers into the array itself
  - so that pointers chase results in the desired random access pattern

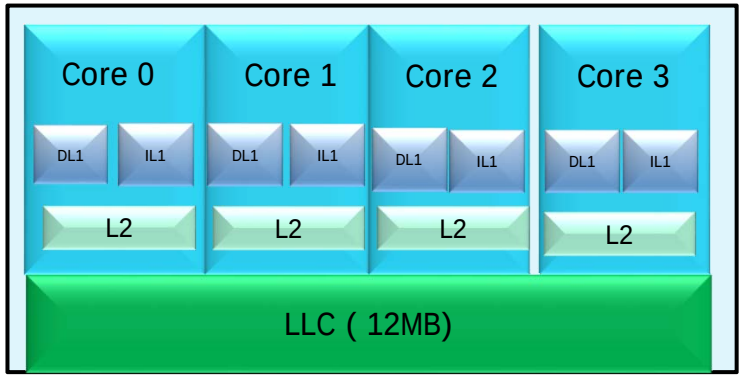


Characterize Different levels of Caches using Random VS Consecutive Read Accesses

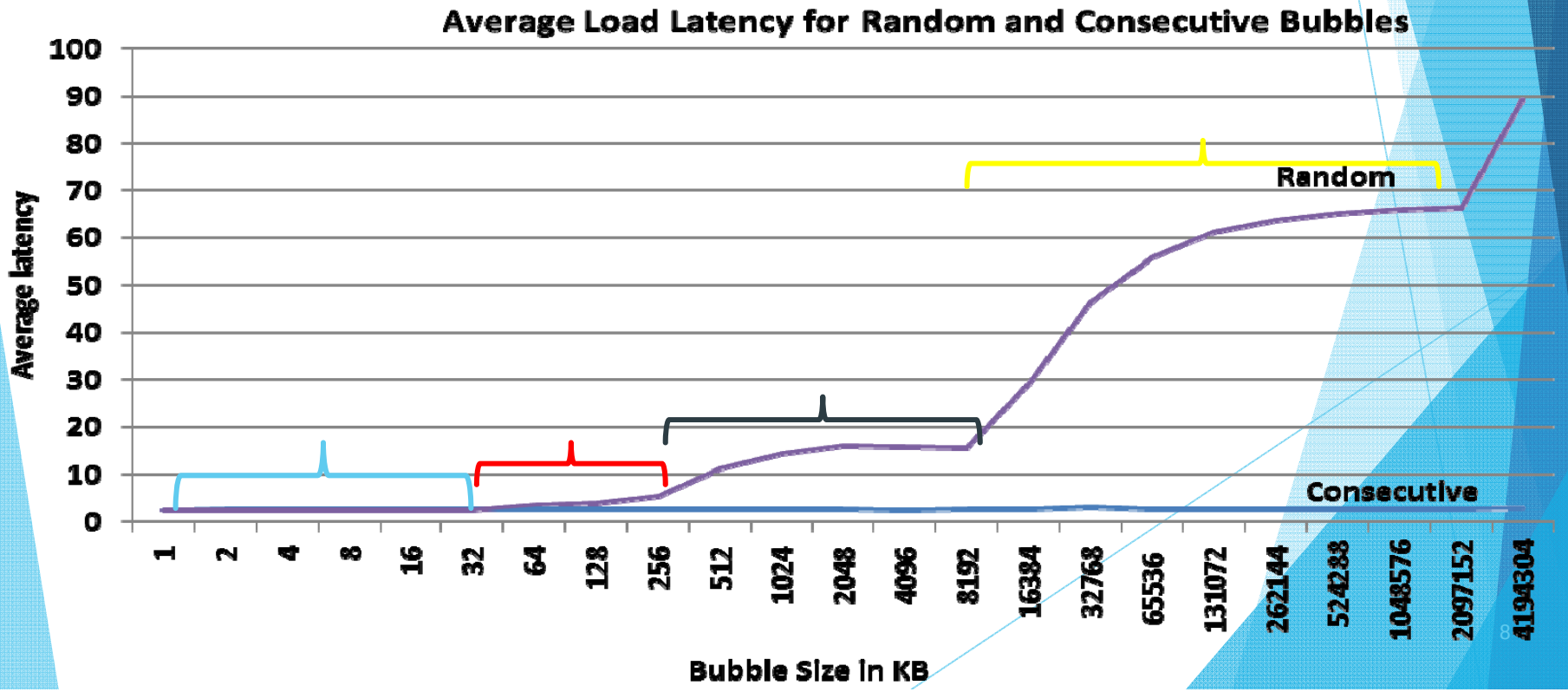
- Why to use Sattolo's algorithm for cache characterization?
  - It shows the read latency as far as the array size increases
  - If array size is more than L1 cache then the read latency should increase, the same for other levels of caches

- Consecutive array accesses do not show the accesses latency as far as the array size increases
  - Due to data locality
  - Prefetching

Cache Characterization on a Given Platform



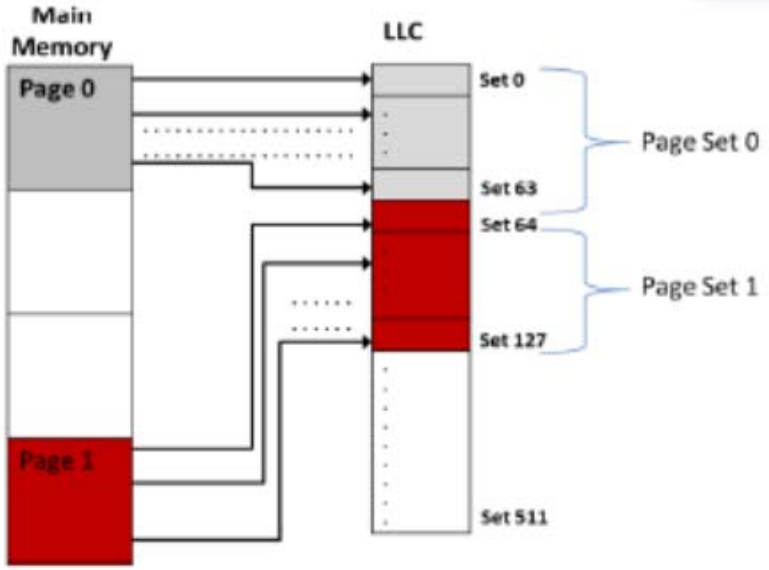
Processor	L1 Cache	L2 Cache	L3 Cache
Family Westmere – 5600 series Servers Intel(R) Xeon(R) CPU E5620 @ 2.40GHz	DL1: 32KB size 8 ways, 64B block size, 64 sets	Unified L2: 256 KB size 8 ways, 64B block size, 512 sets	Shared Unified L3; 12 MB per CPU node, 16 ways, 64B block, 12288 sets
2 CPUs, 4 physical cores per CPU, HyTh: 2 * 8 virtuall cores	IL1: 32KB, 4 ways, 64B block size, 128 sets		

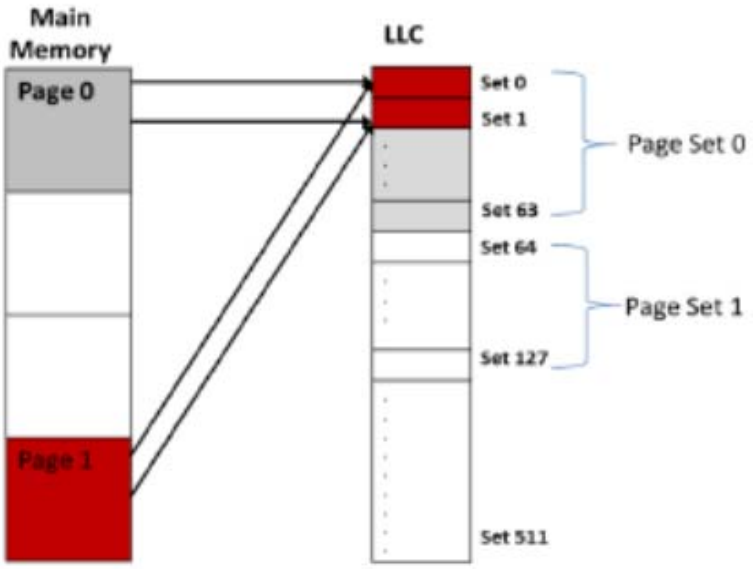


Cache Characterization on a Given Platform

OS PAGES – Small and Huge Pages

- LLC Conflict Misses Due to Page Mapping Performed by OS
  - In today's systems, every application running on an operating system (OS) has its unique virtual address space
  - Giving the impression of a continuous block of memory.
  - For a given application the OS allocates different pages
    - mapped in different memory locations.
    - each page contains blocks with contiguous physical memory addresses.

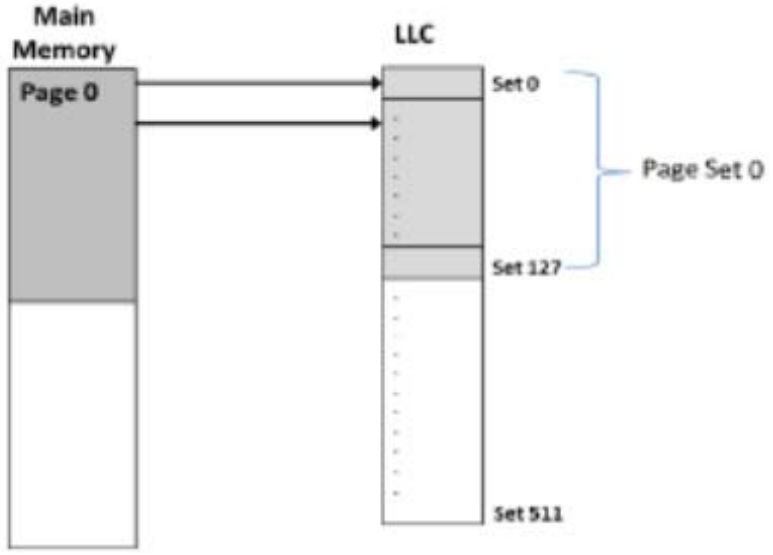




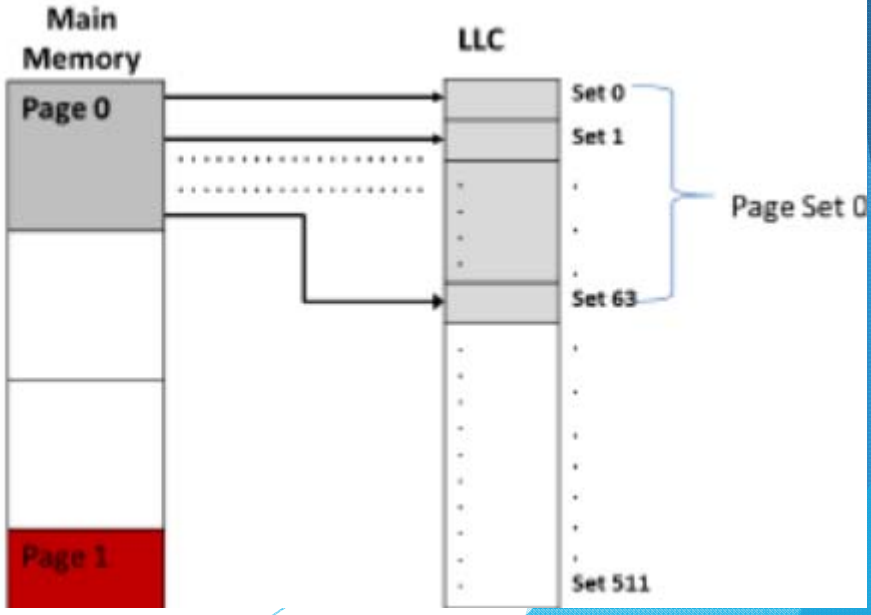
Cache Characterization

Small Pages Vs Huge Pages

- Small Pages 4KB each page
- Huge Pages 2MB each page
- Reduce LLC conflict misses using Huge Pages
  - more consecutive addresses per page

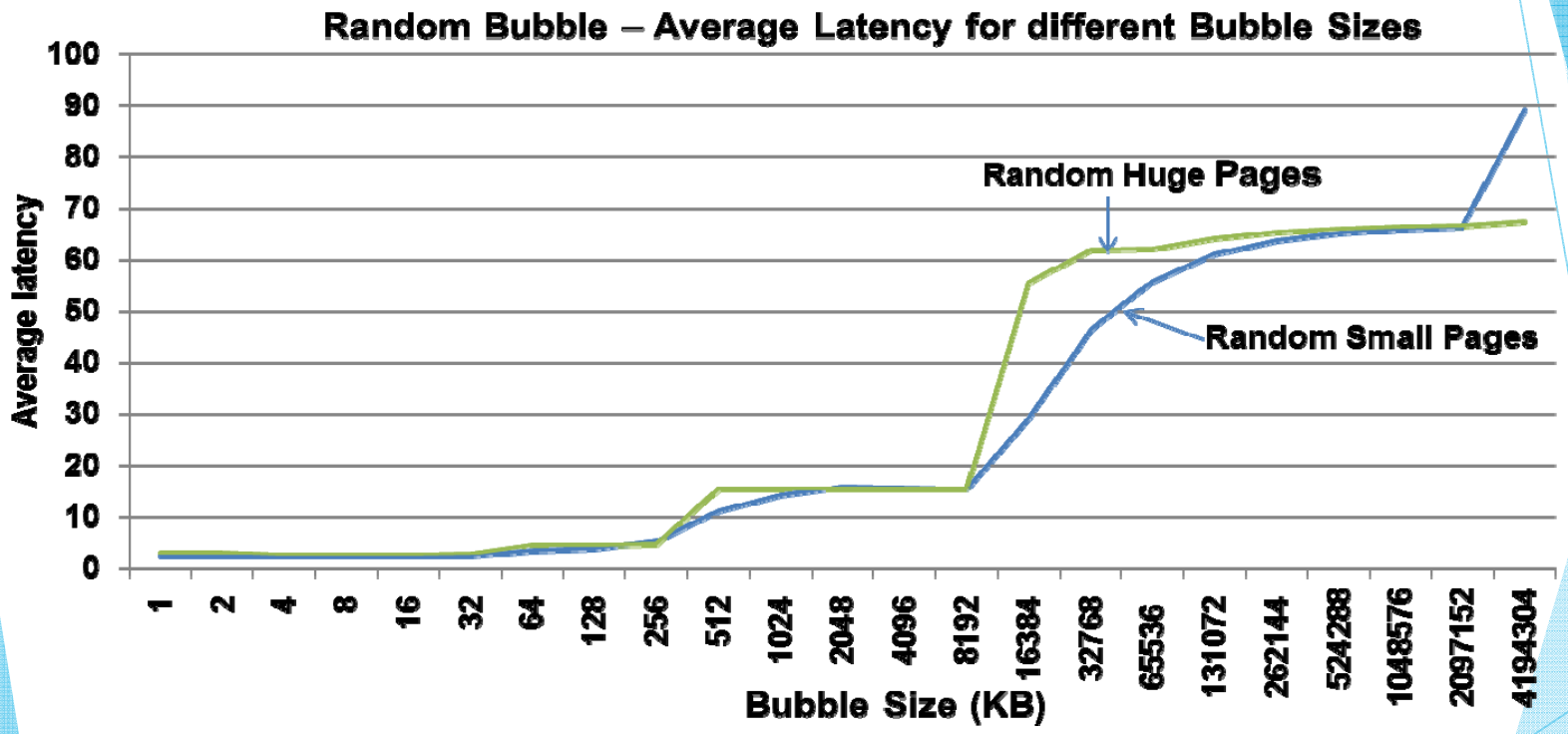


Huge Pages



Small Pages

Evaluate Random Accesses Using Huge/Small Pages

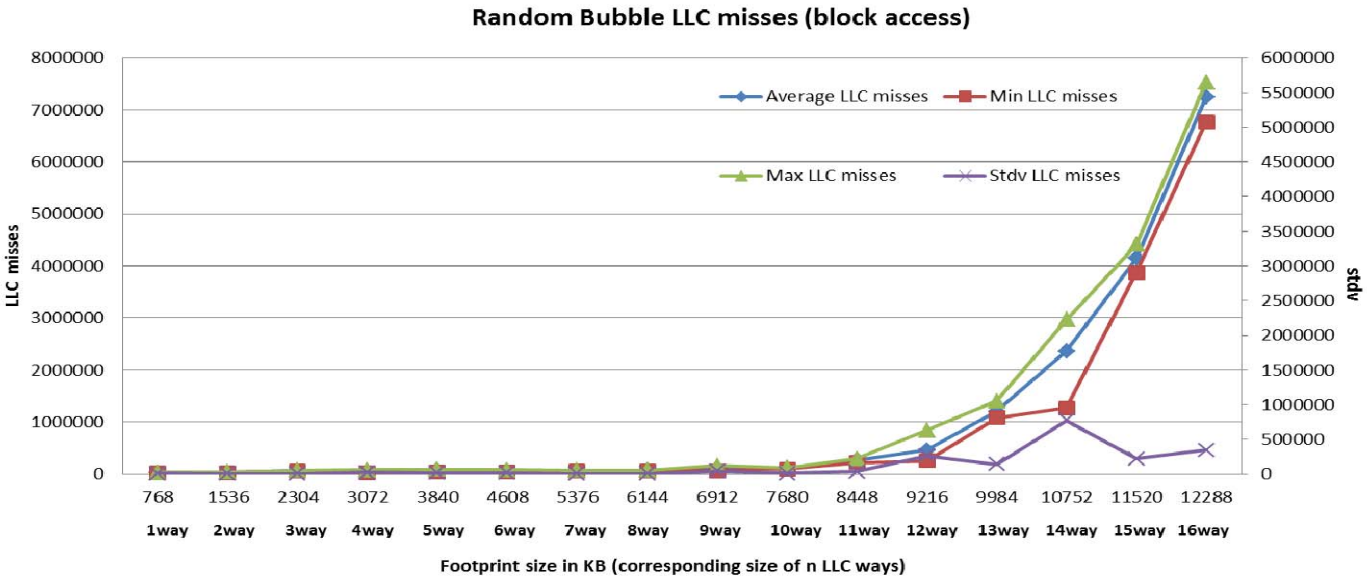


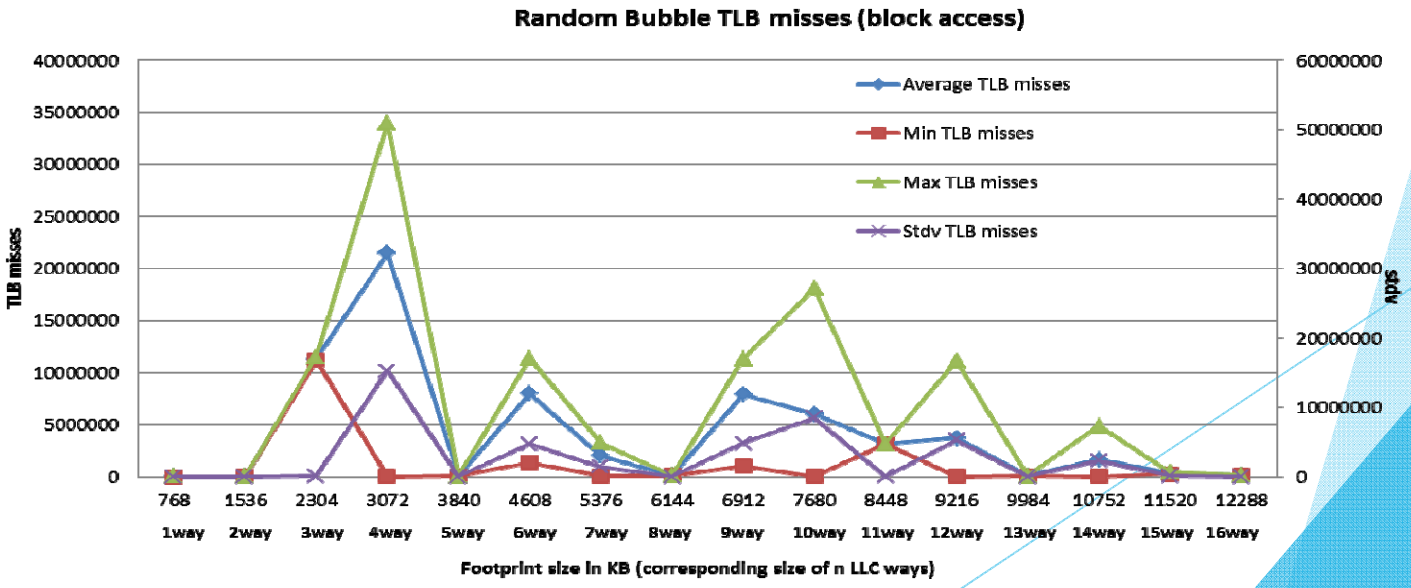
Other parameters to be considered during cache characterization

- Isolation
- Pin program to a given core to avoid context switching
  - taskset
- Replacement policy
- Accurate cycle counting (rdtsc)- avoid OS system calls
- Multiple Runs

Thank You

OS Page Conflicts Affecting Random & Streaming Bubbles





Random Bubble
Different Access Granularity (word & block)