Exam Assignments V09

chengchengguo 183090

1. How do **bandwidth-bound** computations **differ** from **compute-bound** computations?

Bandwidth bound

- kernels approach the physical **limits of the device in terms of access to global memory**
- **bottleneck is cpu**

compute-bound:

- its **boundaries** in time being more defined **either by computing** (for example, numerical calculations) or **communicating** (mainly just waiting to receive inputs or to send outputs). [1]
- bottleneck is memory access

2. Explain why **temporal locality** and **spatial locality** can improve program performance.

Temporal Locality[2]

In a program with **good temporal locality**, a **memory location** that is referenced once is likely to be **referenced again multiple times** in the near future. (reuse data in caches)

Spatial Locality

In a program with **good spatial locality**, if **a memory location** is referenced once, then the program is likely to **reference a nearby memory location** in the near future. (prefer unit stride memory access)

program with good temporal and spatial locality can save time in the process when it access data from memory and save it in then cacheline,

vector arithmetics is cheap, memory access is expensive [3]
Modern computers make heavy use of SRAM-based **caches** to try to bridge the processor–memory gap. This **approach works** if programs tend to exhibit good **temporal locality** and **spatial locality**.

---

[1] Compute-bound vs i/o-bound (slikts.github.io)
[2] folie v09 page 8
[3] folie v09 page 7

3. What are the **differences** between **data-oriented design** and **object-oriented design**?

"Data-Oriented design shifts the perspective of programming from objects to the data itself: The type of the data, how it is laid out in memory, and how it will be read and processed in the game."[4]

Data-oriented design pays attention to how complex data is stored; for cache effectiveness, or for eliminating lock contention by generating copies. Column- vs row- stores of data are a database application of DoD. As a side effect, DoD seems to cross swords with OOD; where the latter tries to hide object base data. DoD exposes data, so the user can choose to process structured collections in a way that best works with actual storage[5]

- comes from the **game industry**
- **efficient usage** of the **CPU cache**
- **easier to parallelize**

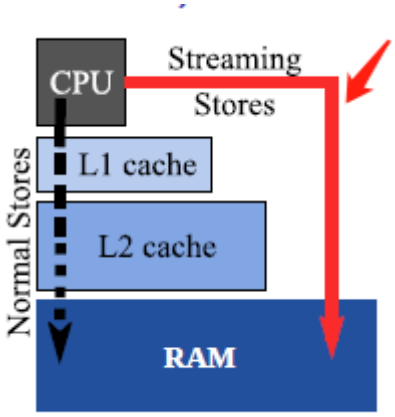
OOP's main focus is on attaching functionality to particular pieces of data.[6]
OOP has many object and merge them, it is less efficient than DoD

4. What are **streaming stores**?

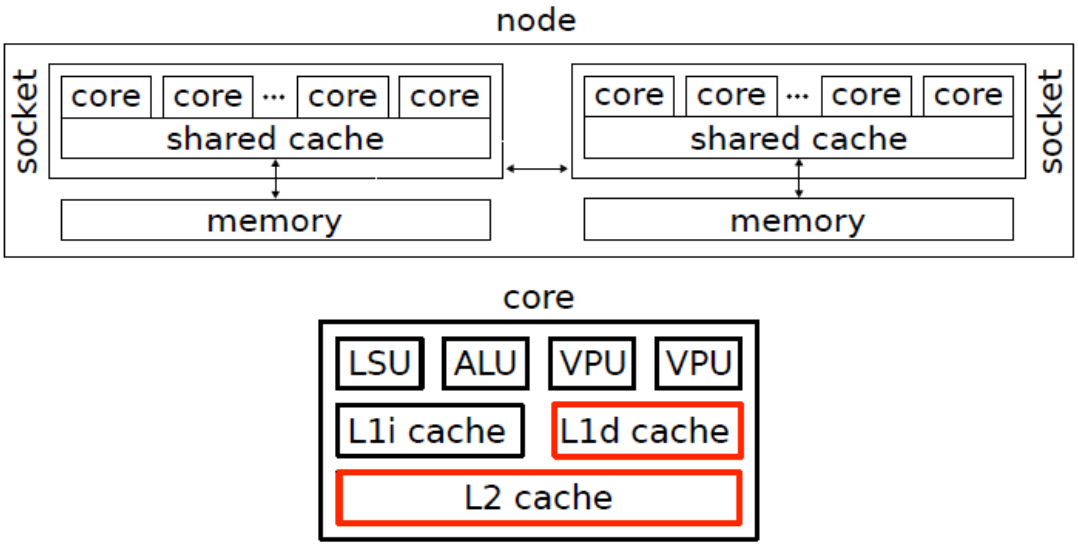
Streaming stores (writes) executed with the non-temporal move instructions[7]
Temporal vs. Non-temporal:
„Data referenced by a program can be temporal (data will be used again) or non-temporal (data will be referenced once and not reused in the immediate future)

- with **streaming stores** it's possible to **bypass cache** and **write directly to RAM** [8]
- saves cache for other data

[4] Data-Oriented vs Object-Oriented Design | by Jonathan Mines | Medium
[5] oop - What's the difference between Data Oriented and Data Driven Programs? - Stack Overflow
[6] Programming Paradigms: Object Oriented vs Data Oriented – PERPETUAL ENIGMA (prateekvjoshi.com)
[7] PowerPoint Presentation (cern.ch)
[8] folie v09 page 13



5. Describe a **typical cache hierarchy** used in **Intel CPUs**.



a typical cache hierarchy is like it shows above

- core has different calculate unit (LSU load-store unit, ALU arithmetic logic unit, VPU Video Processing Unit…)
- in a core we have 2 level of cache, L1 cache, and L2 cache, L2 is normally bigger than L1 and cheaper
- in a socket some cores are linked to a same shared cache (L3 cache much bigger and cheaper)
- a socket is linked to memory
- core normally read cacheline(some continuous data from its nearest

L1 cache) , it takes the least time

- streaming load/store, load/store data from L3 to core
- the nearer the cache is, the shorter access time is

6. What are **cache conflicts**?

cache conflicts are caused by hash table conflicts: CPU **caches** are basically **hardware hash tables**, so they **suffer from hash table conflicts**[9]

if you **access memory** in **specific address intervals** and repeatedly put more cache lines into the bucket than it can hold, you can observe significant slowdown since the **cache will thrash** and evict a cache line on every access

A sequence of accesses to memory repeatedly overwriting the same cache entry. This can happen if two blocks of data, which are mapped to the same set of cache locations, are needed simultaneously.[10]

[9] In computer science, a **hash collision** or clash is when two pieces of data in a hash table share the same hash value. The hash value in this case is derived from a hash function which takes a data input and returns a fixed length of bits.

[10] Cache conflict | Article about cache conflict by The Free Dictionary