EECE 7352: Computer Architecture

Assignment 5

Jiayun Xin

NUID: 001563582

College of Engineering

Northeastern University Boston, Massachusetts

Spring, 2022

# Q1

*Why do the authors propose to use hashing for inverted page tables in this article “Virtual memory: issues of implementation”? What is the purpose of Hash Anchor Table?*

Normally, operation system would look for a page frame number through page table depending on the given virtual page number. Using hashing for inverted page tables is to make page frame number available and avoid page number collision by using a collision-chain mechanism.

Hash Anchor Table is used to keep the table small and increase memory accesses per lookup as a trade-off. At the same time, it reduces the average collision-chain length.

# Q2

*Why do the authors of this paper “Effective Mimicry of Belady’s MIN Policy, HPCA’22 (published in February’22)” state that their MockingJay policy is more effective than prior approaches? What is the difference between inferring at priority at the time of insertion vs. at the time eviction?*

On both single-core and multi-core platforms and both with and without a prefetcher, the new MockingJay policy shows improvement with different extent comparing with LRU, SHiP and Hawkeye. In addition, it can provide accurate reuse distance predictions. When reuse distance prediction in unavailable, the age information will be considered.

Inferring at priority at the time of eviction takes subsequent lines as a factor to compute line’s priority but insertion not.

# Q3

*What is the key idea behind entangled instruction prefetcher (first appeared in the Instruction Prefetching Championship)?*

The entangled instruction prefetcher (EPI) is designed to meet three properties of timeliness, coverage and accuracy. Firstly, it searches for the instruction that triggers the prefetch for the consecutive instruction and computes the latency of each prefetch. As the process of tracking each pair of entangled cache lines, EPI merges those following basic blocks and entangles head of basic blocks to reduce entangled lines.

# Q4

*Describe the two TLB prefetchers proposed in the following paper: (1) Leader-Follower, and (2) Distance-based Cross-Core prefetchers. Clearly articulate the motivation and benefits of inter-core TLB prefetching.*

As growing of chip multiprocessors (CMPs), it is necessary to examine TLB performance on parallel workloads. In fact, it is similar between uniprocessor TLB miss pattern and multi processors. To reduce cache access time and miss rates, two new TLB prefetchers were designed.

1. Leader-Follower pushes the common TLB misses from the “leader” to other cores to avoid missing on the same virtual page entry. But identifying miss patterns and prevent bad prefetch are the challenges that leader-follower should focus on.
2. Distance-based Cross-Core prefetchers is designed to detect and adapt to the stride patterns. It focuses on the succussive missing virtual pages on one core and pass the same distance patterns to other cores to eliminate miss rates in total.

# Q5

(a)

Page offset length = log2(16k) = 14

Virtual page length = virtual address length – page offset length = 40 – 14 = 26

Physical page length = physical address length – page offset length = 32 -14 = 18

TLB entry size = virtual page length + physical page length + other purposes = 26 + 18 + 4 = 48bits

Total size of TLB = #TLB entries \* TLB entry size = 8 \* 48 = 384bits

(b)

PTE size = PTE entry for other purposes + physical page length = 4 + 18 = 22bits

# PTEs = 2^26

The total size of the page table = 22 \* 2^26 bits

# Q6

P = 512bytes

VPO = PPO = log2(512) = 9bits

VPN = virtual address space – VPO = 30 – 9 = 21bits

PPN = physical address space – PPO = 22 – 9 = 13bits

P = 1KB

VPO = PPO = log2(1KB) = 10bits

VPN = virtual address space – VPO = 30 – 10 = 20bits

PPN = physical address space – PPO = 22 – 10 = 12bits

P = 2KB

VPO = PPO = log2(1KB) = 11bits

VPN = virtual address space – VPO = 30 – 11 = 19bits

PPN = physical address space – PPO = 22 – 11 = 11bits

# Q7

Virtual address = operation system length

Physical address = log2(RAM size)

Offset = log2(page size)

Virtual page number = virtual address – offset

Physical page number = physical address – offset

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Config | Vir. Addr. | Phy. Addr. | VPN | PPN | Offset |
| a | 32 | 30 | 20 | 18 | 12 |
| b | 32 | 31 | 18 | 17 | 14 |
| c | 64 | 34 | 50 | 20 | 14 |

# Q8

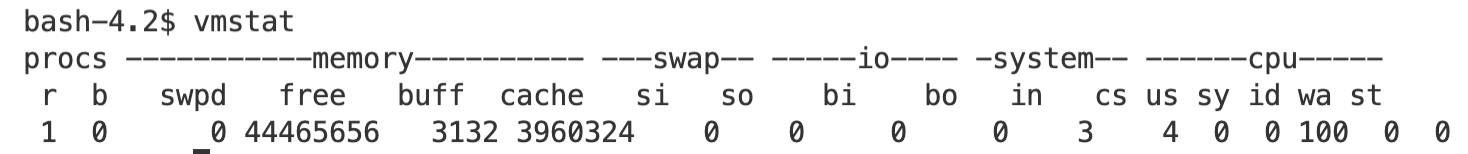
*a. How will you get the pagesize on your system?*

A picture containing text

Description automatically generated

Use “getconf” command.

*b. What is “vmstat” command? What does it report?*



The “vmstat” command is a built-in monitoring utility. It reports real-time information about memory, system processes, paging, block I/O and CPU scheduling.

*c. What is the output of “cat /proc/meminfo”?*

Table

Description automatically generated with medium confidence

It outputs a list of memory usedness. Through “meminfo”, we can determine how much available memory the machine has.

*d. What does the output of “sar –B” tell you?*

Table

Description automatically generated

“sar -B” command outputs the paging statistics of coe linux server.