CSED211: Lab. 8

Cache Lab

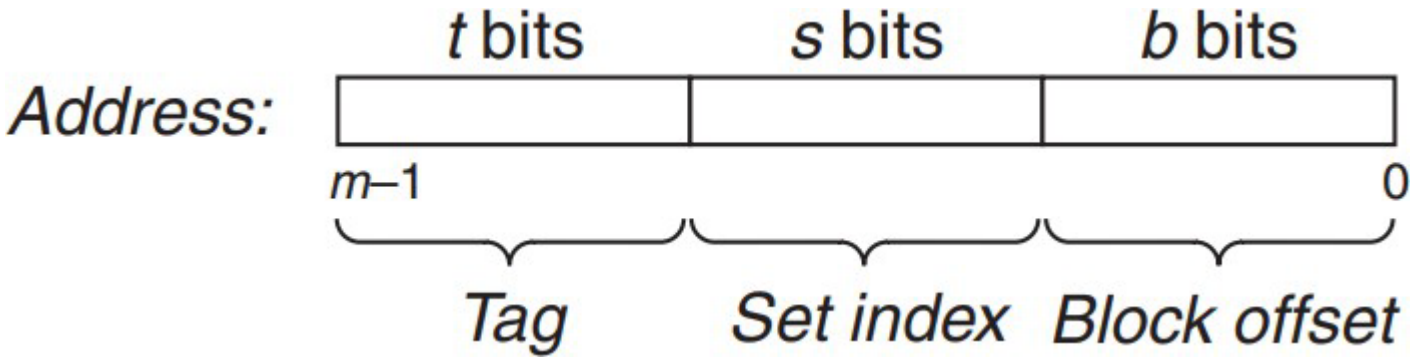
(Part A: Writing Cache Simulator)

백승훈
habaek4@postech.ac.kr
POSTECH

2023.11.06

Table of Contents

Memory Address

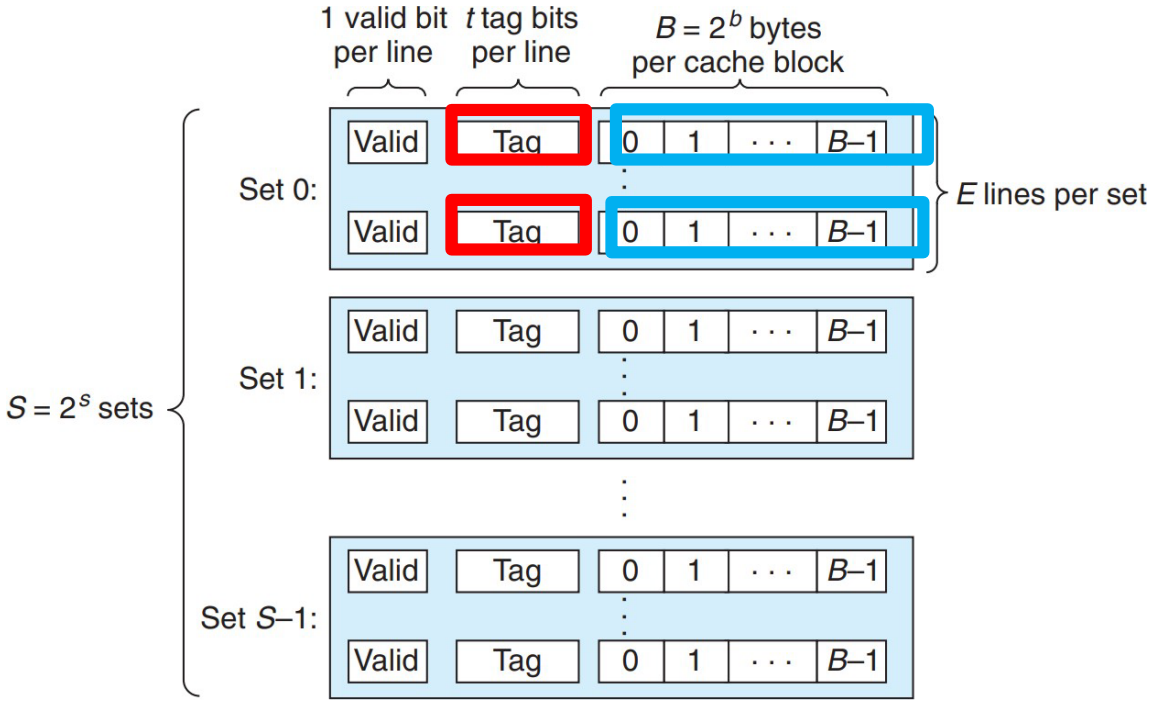


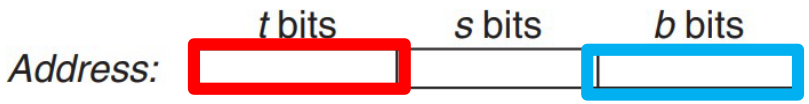
* m-bit machine

- Block offset (B): b bits
- Set index(S): s bits
- Tag: m – (s+b) bits

Cache Memory

- Store data or instruction which *satisfies* principle of *locality* (spatial, temporal)
- A cache set is a set of E(=associativity) cache lines





Associativity (E)	Mapping
1	Direct Mapping
$1 < E < C/2^b$	Set Associative Mapping
$C/2^b$	Fully Associative Mapping

Cache Hit / Miss / Eviction

- If a program attempts to access block X and block X is in cache, then this access is called '**hit**'

- If a program attempts to access block X and block X is not in a cache, then this access is called '**miss**'

- If the address accessed by a program is already filled with another different block (not empty block, called valid), then this need to be given '**eviction**'
  - Need replacement policy

- (In this lab, we don't care about data, so only line matching is enough)

Cache Replacement Policy

- Size of cache memory is limited, thus not all blocks from upper layer are stored in a cache

- So, cache memory needs to replace block in a cache with the new block to be used in the near future
  - How to pick the block to be evicted? (victim block)

- Replacement policy algorithms
  - LRU: Replace Least Recently Used line with the new one
    - Use counter to trace recently accessed line
  - FIFO, LIFO, …

Cache Lab

Part A. Building a cache simulator

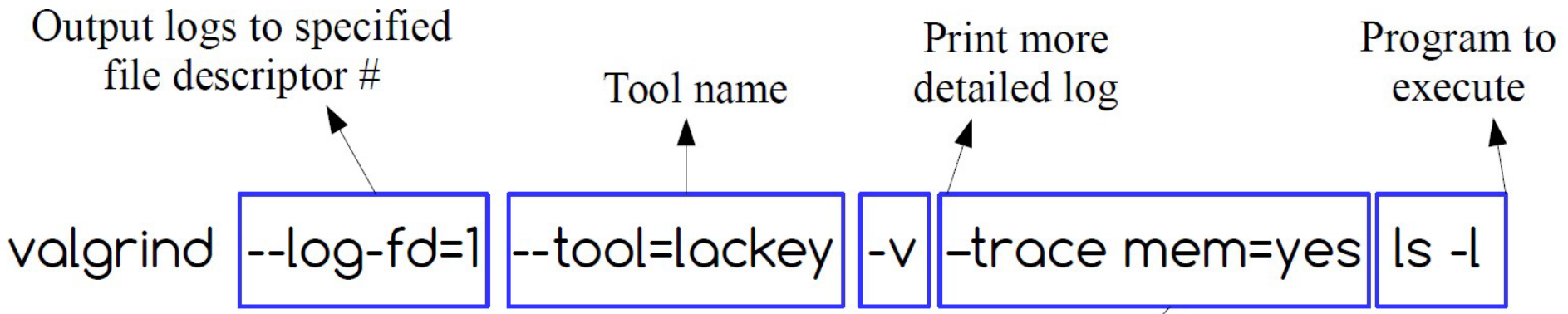
Valgrind

- A suit of tools for debugging and profiling programs to detect memory management problems
- Provided as a command line tool
- Usage:
  - `$ valgrind --tool=[tool_name] [option] [program]`
  - `$ valgrind --tool=lackey ls`
- Install (Ubuntu assumed)
  - `sudo apt install valgrind`
- Reference

Tools	Function
Memcheck	Memory Profiler
Cachegrind	Cache Profiler
Callgrind	Extension of Cachgrind
Massif	Heap Profiler
Helgrind	Multi-threaded Program Debugger
Lackey	Simple profiler

Valgrind Example

- Check more file descriptors by “lsof”



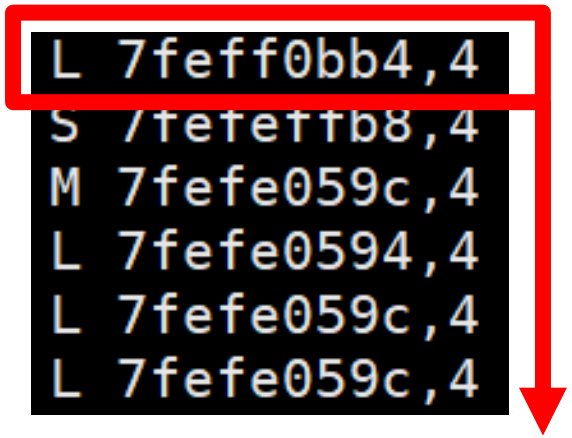
File Descriptor	#
standard input	0
standard output	1
standard error	2

prints the size and address of almost every memory access made by the program.

```
I  04018ac0,7
 L 042270b0,8
I  04018ac7,4
I  04018acb,4
I  04018acf,2
I  04018ad8,4
I  04018adc,7
 L 042270a0,8
I  04018ae3,5
 S ffefff7e8,8
I  0401b4f0,5
I  0401b4f5,2
I  0401b4f7,6
```

Semantics of the output

- [space] operation address, size
- Operation: I (instruction load), L(data load), S(data store), M (data modify)
- Address: virtual memory address that program accessed (in hex format)
- Size: number of bytes



File I/O APIs

- Low level file access
  - `open(), read(), write(), close()`
- High-level standard I/O library
  - `fopen(), fclose(), fread(), fwrite()`
  - **`fscanf()`**: formatted input (**for reading line from trace file**)
  - `fgets()` : read a string with n bytes
- Usage pattern
  - `fp= open("text.txt","r");`
  - //do something with the file
  - `fclose(fp);`

Dynamic Memory Allocation / Deallocation

- `malloc()`: allocate consecutive dynamic memory space in heap
- `free()`: de-allocates specified memory space allocated by `malloc`
- E.g.
  - `int *p =(int*) malloc (16);`
  - `free(p)`

Parsing Command Line Options

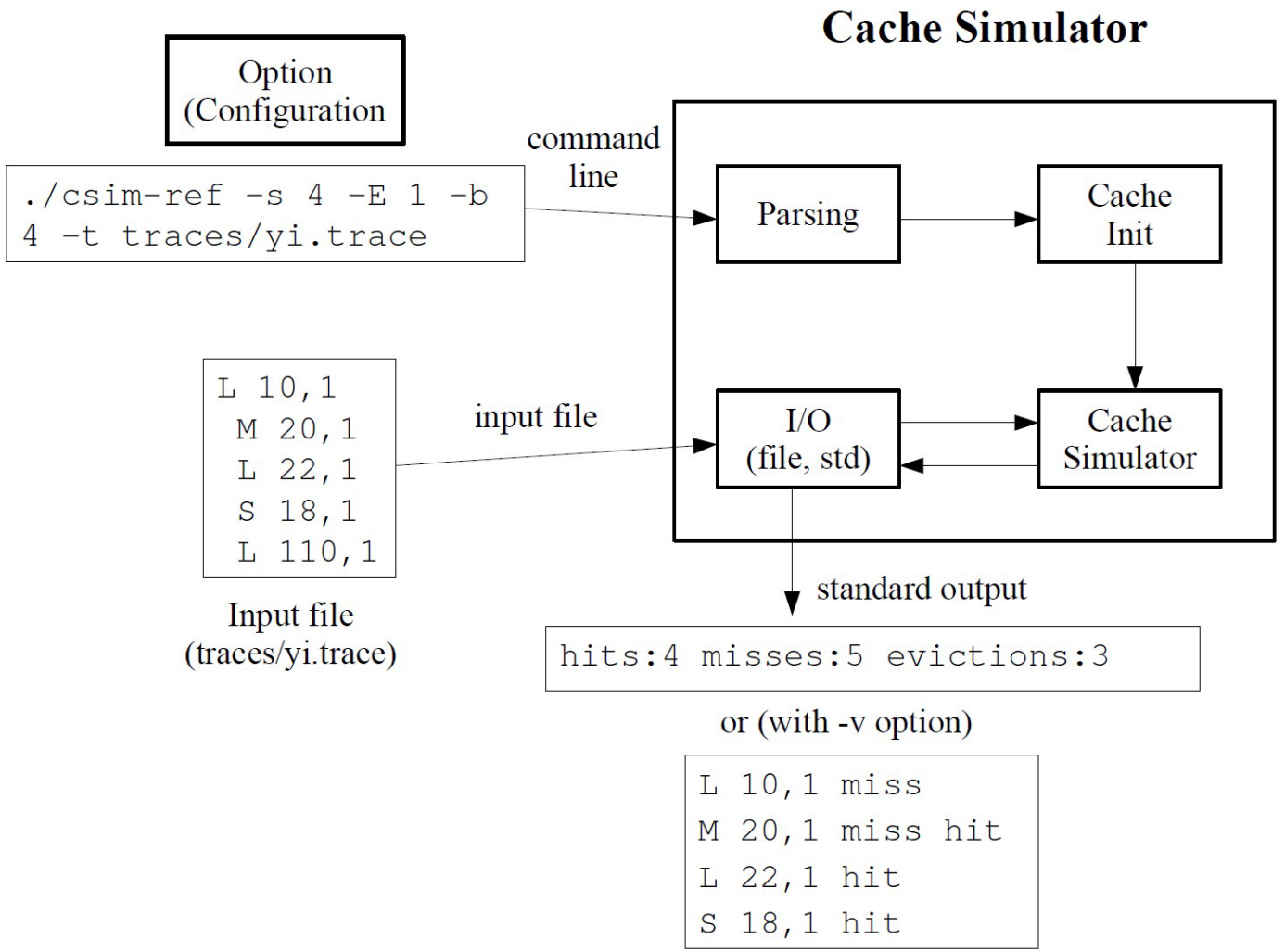
- A program usually need to react differently to different given configuration
- Short options (with/without argument)
  - “`ls -l`”, “`df -h`”, “`gcc -o [arg]`”
- Long options (with/without argument)
  - “`ls -help`”, “`gcc -version`”, “`gcc -std=c11`”
- Two common option parsing library in POSIX implementation
  - **`getopt`**`()`: parse short options
  - `getopt_long()`: parse short + long options

- https://man7.org/linux/man-pages/man3/getopt.3.html

Homework (Part A)

- Understand the mechanism of cache memory system and implement simplified cache simulator
- Write a program which simulates cache memory access and count hit / miss / eviction
- Input file is pre-generated by valgrind tool
- Output contains hit /miss / eviction for each instruction in input file
- It doesn't care about memory content. Only cares about **tag bit** and **set index** bit to find matching line

Homework (Part A)



Homework (Report)

- Deadline: 11/27 (Mon) 23:59
- You need to
  - Explain how did you build your cache simulator in the report.
  - For the report, follow the format [student#].pdf
    - For example, 20170354.pdf (No square brackets in the file name).
  - For the code, follow the format [student#].tar
    - For example, 20170354.tar (No square brackets in the file name).
    - Combining Part A(csim.c) and Part B(trans.c) together.
    - **No zip, No tar.gz!**
- You can find more details in writeup_cachelab.pdf

Quiz