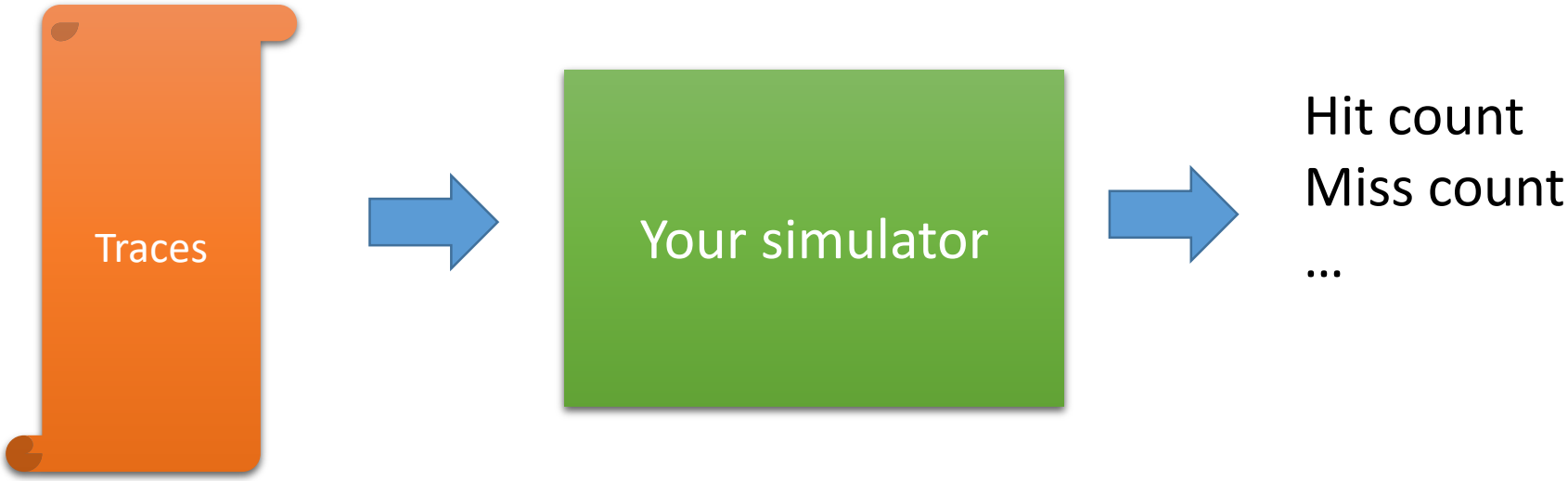
Operating Systems Programming Assignment #5

Page Replacement Simulation: LRU and LFU

Prof. Li-Pin Chang, NCTU

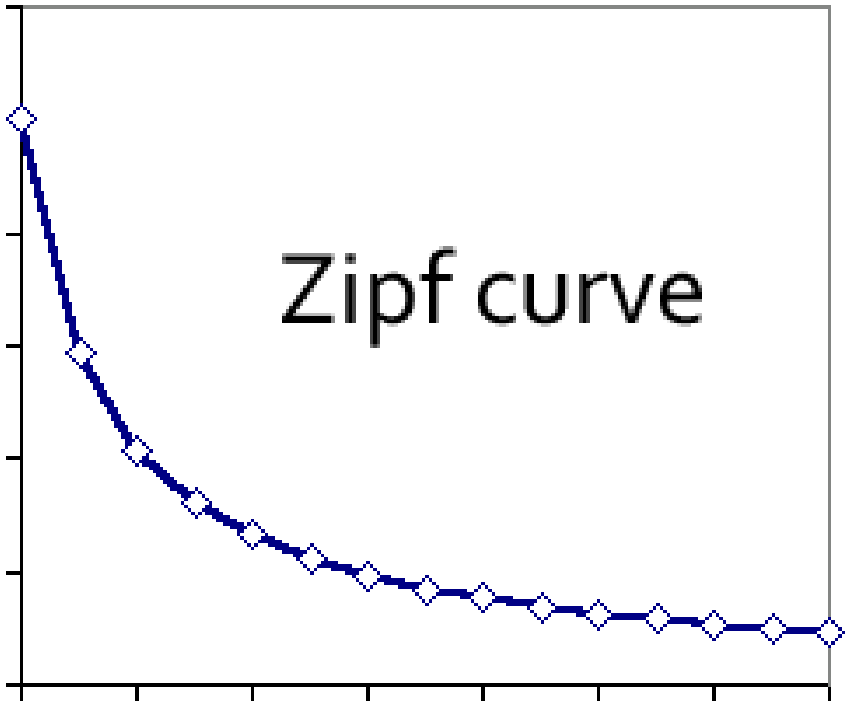
Simulation



Trace File Format

- The trace format and Zipfian distribution

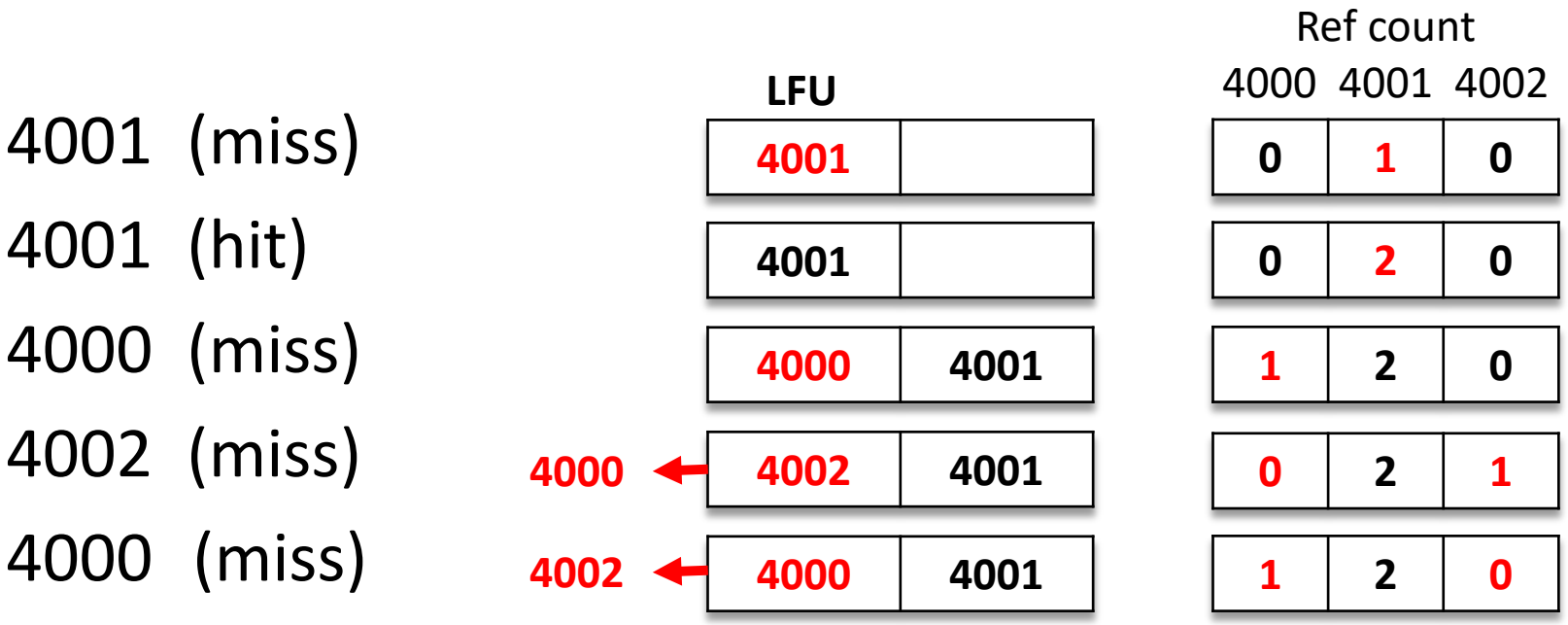
```
1003
1003
9340
1243
1108
1786
1066
1312
1000
1000
1213
1249
2116
```



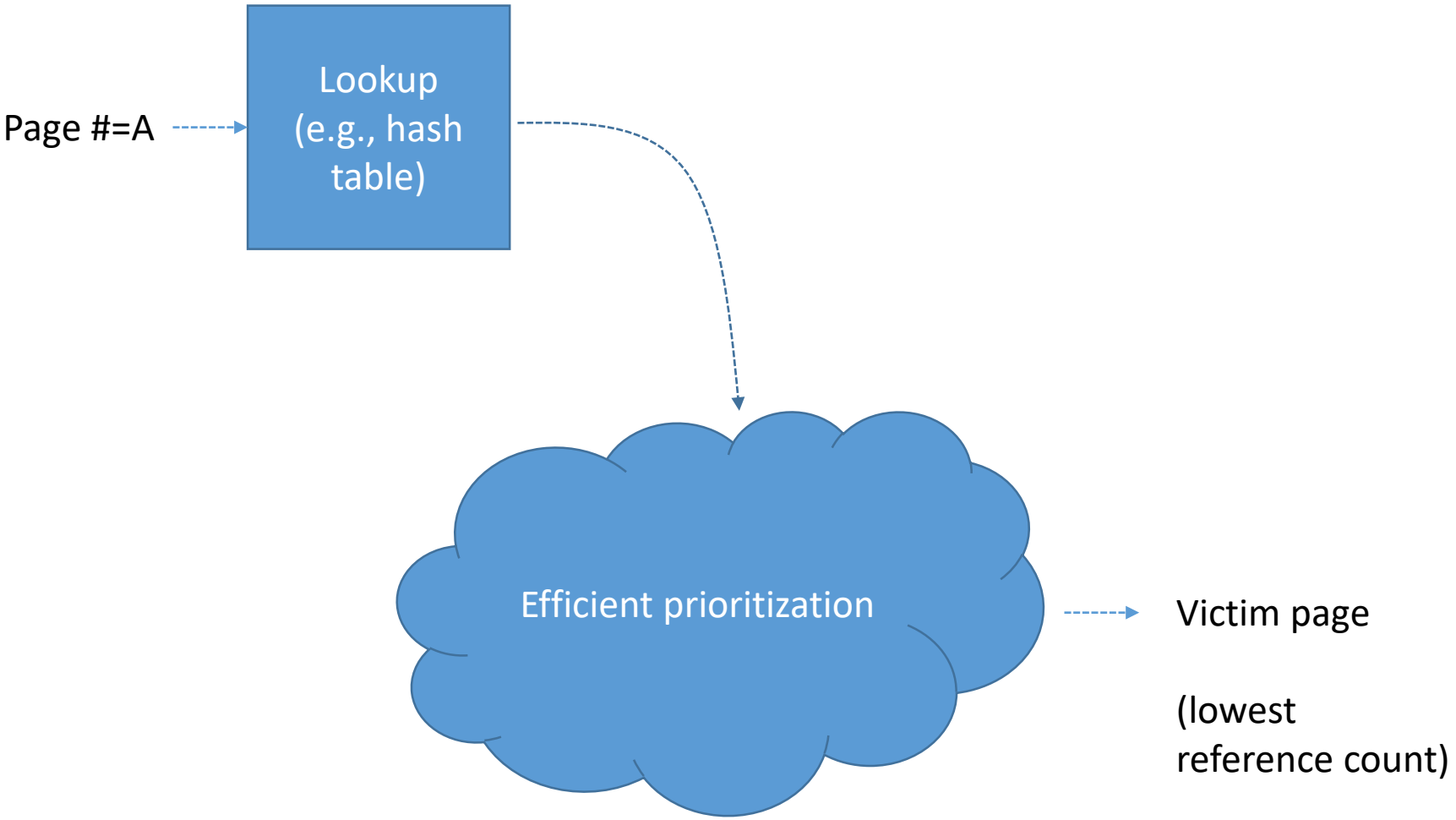
Unsigned integer

Page Replacement(LFU)

- Example: Frame #=2



Simulator Structure (LFU)



Page Replacement(LRU)

- Example: Frame #=2

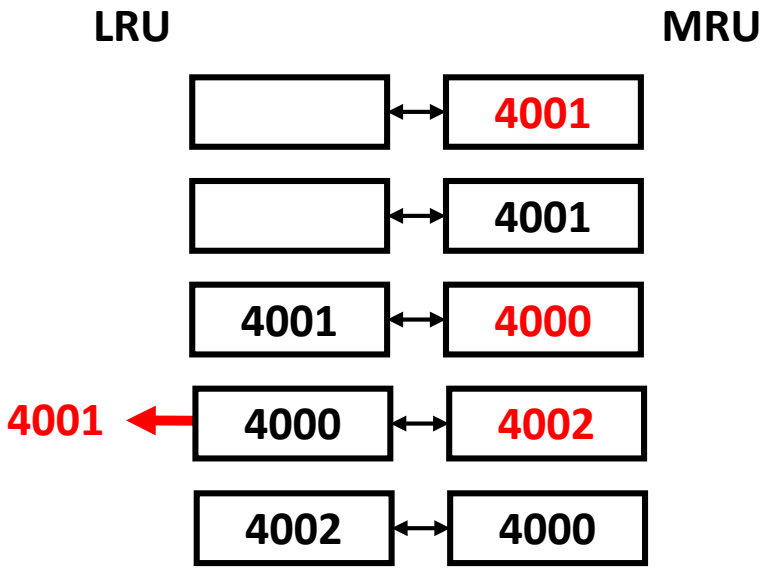
4001 (miss)

4001 (hit)

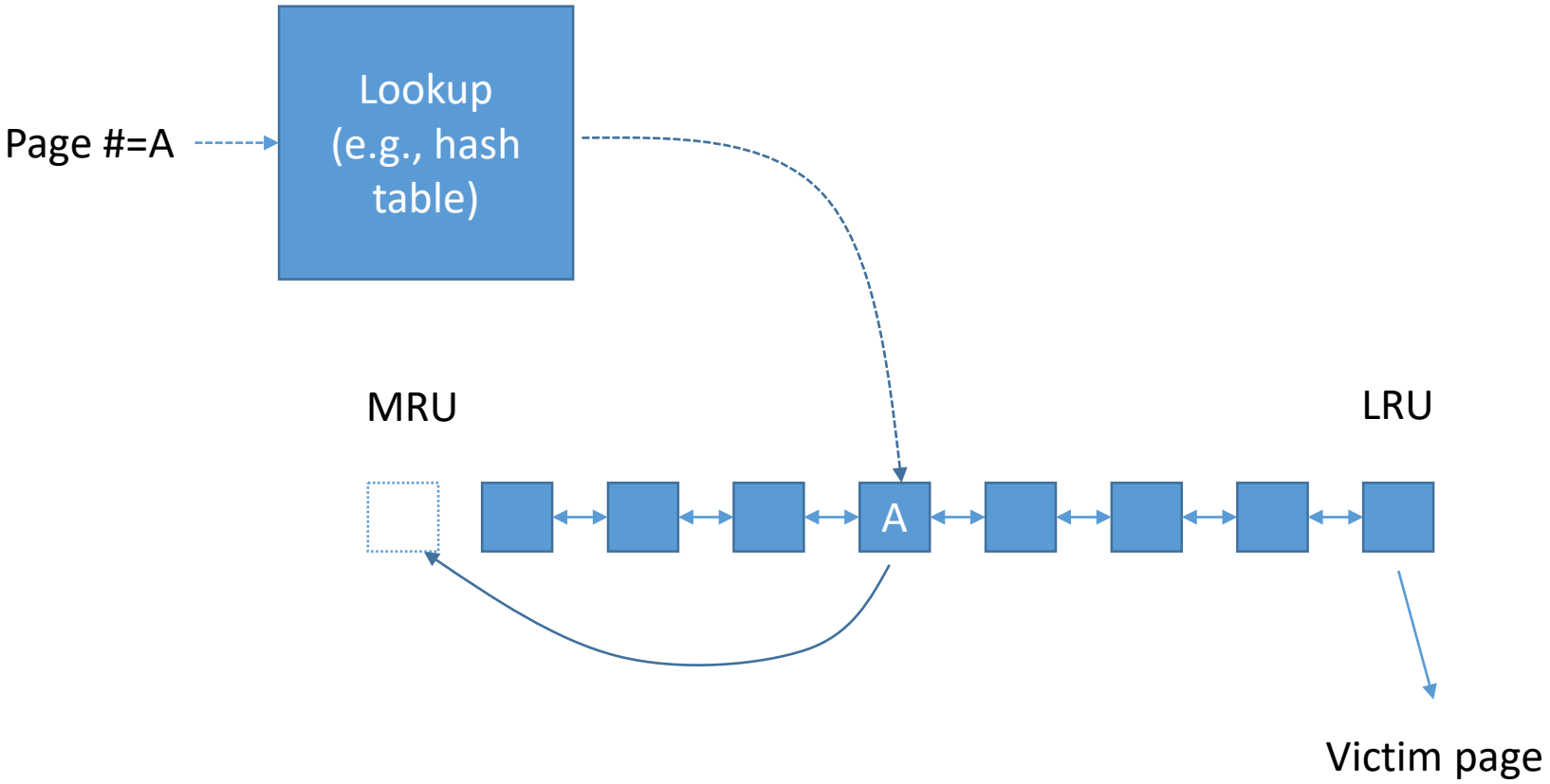
4000 (miss)

4002 (miss)

4000 (hit)



Simulator Structure (LRU)



Page Cache Operations

- Page lookup
  - Check whether or a new reference is a hit or a miss
  - Hash tables, binary search trees, skip lists….

- Do not use linear search!!!
  - You will receive a grade penalty if you do
  - Implement your own search, or reuse any existing libraries/classes for searching
  - TAs will examine your code
  - Duplication in this part does not count

Victim selection

- LFU
  - The least frequently used page
  - If two pages have the same access count, the page having a smaller reference sequence number is replaced
  - You may need to store the reference sequence number when a page is added to the page cache
- LRU
  - The least recently used page

Procedure

1. Algorithm=LFU
2. For (frame # = 128, 256, 512, and 1024) do
   - Read the trace file
   - Run simulation
   - Print out the hit count, miss count, page fault ratio
3. Print out the total elapsed time of Step 2

4. Algorithm=LRU
5. For (frame # = 128, 256, 512, and 1024) do
   - Read the trace file
   - Run simulation
   - Print out the hit count, miss count, page fault ratio
6. Print out the total elapsed time of Step 5

The scenario of program

Output format

```
LFU policy:(\n)
frame   (\t)   hit   (\t\t)   miss   (\t\t)   page fault ratio   (\n)
128     (\t)   %d    (\t\t)   %d     (\t\t)   %.10f             (\n)
256     (\t)   %d    (\t\t)   %d     (\t\t)   %.10f             (\n)
512     (\t)   %d    (\t\t)   %d     (\t\t)   %.10f             (\n)
1024    (\t)   %d    (\t\t)   %d     (\t\t)   %.10f             (\n)
Total elapsed time: %.4f sec (\n)

LRU policy:(\n)
frame   (\t)   hit   (\t\t)   miss   (\t\t)   page fault ratio   (\n)
128     (\t)   %d    (\t\t)   %d     (\t\t)   %.10f             (\n)
256     (\t)   %d    (\t\t)   %d     (\t\t)   %.10f             (\n)
512     (\t)   %d    (\t\t)   %d     (\t\t)   %.10f             (\n)
1024    (\t)   %d    (\t\t)   %d     (\t\t)   %.10f             (\n)
Total elapsed time: %.4f sec (\n)
```

The scenario of program

- Assume the path of the trace file is ./test/sample.txt
  Your output should look like this

```
[tsaichh0619@linux1 assignment5]$ ./bin/ta_5 ./test/sample.txt
LFU policy:
frame   hit             miss            page fault ratio
128     8382341         1617659         0.1617659032
256     8807904         1192096         0.1192096025
512     9134228         865772          0.0865771994
1024    9404369         595631          0.0595631003
Total elapsed time: 2.4703 sec

LRU policy:
frame   hit             miss            page fault ratio
128     7880806         2119194         0.2119193971
256     8425225         1574775         0.1574774981
512     8871064         1128936         0.1128935963
1024    9236112         763888          0.0763887987
Total elapsed time: 2.1897 sec
```

Correctness

- Except the total elapsed time, your results should be exactly the same as ours
- Do not use linear search in anywhere of your program; otherwise, you will receive a score penalty

- Remark: here we do not simulate working set migration, so LFU slightly outperforms LRU; but in real life workloads, LRU usually slightly outperforms LFU

More details

- Total request # <= 500 millions
- Total frame # <= 8192
- The path+file name of the trace file is an argument of your program (see the screen shot), do not hard-coding the pathname of thee trace file
- For each iteration, you should open the file, run the simulation, print the result and close the file
- Do not store the trace data to speed up the next iteration
- Use gettimeofday() to get the total elapsed time

Header of your .c or .cpp

/*

Student No.: <your student id>

Student Name: <your name>

Email: <your email>

SE tag: xnxcxtxuxoxsx

Statement: I am fully aware that this program is not supposed to be posted to a public server, such as a public GitHub repository or a public web page.

*/

About Submission

- We will compile your code on Ubuntu 16.04 or CS linux work station
- You should name your program into xxx_5.c or xxx_5.cpp (xxx is your student ID)
- How we compile your program:

“ gcc -std=gnu99 xxx_5.c -o xxx_5 ” or
“ g++ -std=c++11 xxx_5.cpp -o xxx_5 ”

- If you use a make file, please upload one single zip file that contains your make file and source file (the same naming convention as above)

Plagiarism

- We take plagiarism seriously

- You will get zero points for plagiarism
- “Copying existing code from the internet and revising it for submission” is plagiarism
- “Consulting to existing code from the internet and writing a new program from scratch” is not