哈爾濱工業大學

实验报告

实 验（六）

题　　目 Cachelab

高速缓冲器模拟

专　　业 计算机科学与技术

学　　号 1170300520

班　　级 1703005

学　　生 郭子阳

指 导 教 师 吴　锐

实 验 地 点 G712

实 验 日 期

计算机科学与技术学院

目 录

第 1 章 实验基本信息

1.1 实验目的

理解现代计算机系统存储器层级结构
掌握 Cache 的功能结构与访问控制策略
培养 Linux 下的性能测试方法与技巧
深入理解 Cache 组成结构对 C 程序性能的影响

1.2 实验环境与工具

1.2.1 硬件环境

Intel Core i7 6700HQ，8GB RAM，128GB SSD

1.2.2 软件环境

Windows 10 专业版 64 位，Ubuntu 18.04.1 64 位

1.2.3 开发工具

Codeblocks 17.12，gcc，cgdb，vscode

1.3 实验预习

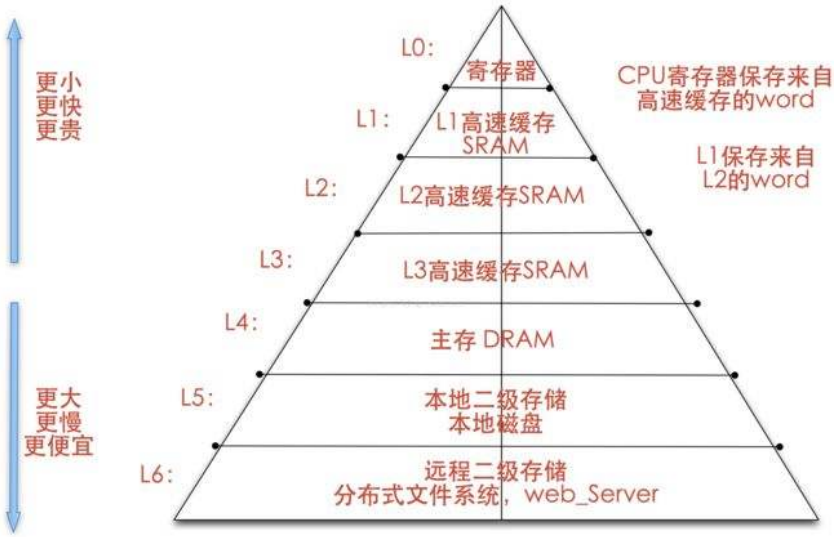
上实验课前，必须认真预习实验指导书（PPT 或 PDF）

了解实验的目的、实验环境与软硬件工具、实验操作步骤，复习与实验有关的理论知识。

画出存储器的层级结构，标识其容量价格速度等指标变化
用 CPUZ 等查看你的计算机 Cache 各参数，写出 C S E B s e b
写出 Cache 的基本结构与参数
写出各类 Cache 的读策略与写策略
掌握 Valgrind 与 Gprof 的使用方法

第 2 章 实验预习

2.1 画出存储器层级结构，标识容量价格速度等指标变化（5 分）



一个存储器层次结构

2.2 用 CPUZ 等查看你的计算机 Cache 各参数，写出各级 Cache 的 C S E B s e b（5 分）

	C	S	E	B	s	e	b
一级数据缓存	32KB*2	64	8	64B	6	3	6
一级指令缓存	32KB*2	64	8	64B	6	3	6
二级缓存	256KB*2	1024	4	64B	10	2	6
三级缓存	3MB	…	12	64B	…	…	6

2.3 写出各类 Cache 的读策略与写策略（5 分）

读策略：

1：命中，则从 cache 中读相应数据到 CPU 或上一级 cache 中。

2：失败，则从主存或下一级 cache 中读取数据，并替换出一行数据，通常采用 LRU 算法。

写策略：

1：命中，又分两种策略

（1）写回法：只写本级 cache，暂时不写数据到主存或下一级 cache，等到该行被替换出去时，才将数据写回到主存或下一级 cache。

（2）写直达：写本级 cache，同时写数据到主存或下一级 cache，等到该行被替换出去时，就不用写回数据了。

2：失败，又分两种策略

（1）按写分配，又分两种：[1]先写数据到主存或下一级 cache，并从主存或下一级 cache 读取刚才修改过的数据，即：先写数据，再为所写数据分配 cache line；[2]先分配 cache line 给所写数据，即：从主存中读取一行数据到 cache，然后直接对 cache 进行修改，并不把数据到写到主存或下一级 cache，一直等到该行被替换出去，才写数据到主存或下一级 cache。

（2）写不分配：直接写数据到主存或下一级 cache，并且不从主存或下一级 cache 中读取被改写的数据，即：不分配 cache line 给被修改的数据。

2.4 写出用 gprof 进行性能分析的方法（5 分）

在编译和链接时，加上-pg 选项。

执行编译的二进制程序

程序正常退出后，在运行目录下 生成 gmon.out 文件。如果原来有 gmon.out 文件，将会被覆盖。

用 gprof 工具分析 gmon.out 文件。

使用 gprof Binary-file gmon.out >report.txt 将报告输出在 report.txt 中。

2.5 写出用 Valgrind 进行性能分析的方法（（5 分）

valgrind 命令的格式如下：

valgrind [valgrind-options] your-prog [your-prog options]

编译（加-g 选项）得到 a.out，再执行 valgrind ./a.out

第 3 章 Cache 模拟与测试

3.1 Cache 模拟器设计

提交 csim.c

程序设计思想：

initCache()和 freeCache()较简单，主要是 accessData()函数，分割请求地址可以得到组索引、标记和块偏移，通过组索引可以找到缓存中对应的组，遍历组中的每一个行，查看是否有相同标记的行，如果有则命中，直接修改缓存行的数据。

如果没有，则是不命中的情况。区分冷不命中和冲突不命中的情况。如果是冷不命中，直接将数据写入无效的空行即可。如果是冲突不命中，则根据 LRU 算法，查找最早被使用的行，每个缓存行中的数据是第几次写入，数据越小使用的事件越早。此时找出行中的 LRU 最小的数据并替换。

测试用例 1 的输出截图（5 分）：

```
➜  cachelab-handout ./csim -s 1 -E 1 -b 1 -t traces/yi2.trace
hits:9 misses:8 evictions:6
```

测试用例 2 的输出截图（5 分）：

```
➜  cachelab-handout ./csim -s 4 -E 2 -b 4 -t traces/yi.trace
hits:4 misses:5 evictions:2
```

测试用例 3 的输出截图（5 分）：

```
➜  cachelab-handout ./csim -s 2 -E 1 -b 4 -t traces/dave.trace
hits:2 misses:3 evictions:1
```

测试用例 4 的输出截图（5 分）：

```
➜  cachelab-handout ./csim -s 2 -E 1 -b 3 -t traces/trans.trace
hits:167 misses:71 evictions:67
```

测试用例 5 的输出截图（5 分）：

```
➜  cachelab-handout ./csim -s 2 -E 2 -b 3 -t traces/trans.trace
hits:201 misses:37 evictions:29
```

测试用例 6 的输出截图（5 分）：

```
➜  cachelab-handout ./csim -s 2 -E 4 -b 3 -t traces/trans.trace
hits:212 misses:26 evictions:10
```

测试用例 7 的输出截图（5 分）：

```
➜  cachelab-handout ./csim -s 5 -E 1 -b 5 -t traces/trans.trace
hits:231 misses:7 evictions:0
```

测试用例 8 的输出截图（10 分）：

```
➜  cachelab-handout ./csim -s 5 -E 1 -b 5 -t traces/long.trace
hits:265189 misses:21775 evictions:21743
```

注：每个用例的每一指标 5 分（最后一个用例 10）——与参考 csim-ref 模拟器输出指标相同则判为正确

3.2 矩阵转置设计

提交 trans.c

程序设计思想：

将矩阵分块，对每一块分别转置，一次取出一个块的所有数据，存放在一个局部变量里，这样每个块访问一次即可，防止转置时抖动的发生。

32×32（10 分）：运行结果截图

```
➜  cachelab-handout ./test-trans -M 32 -N 32

Function 0 (1 total)
Step 1: Validating and generating memory traces
Step 2: Evaluating performance (s=5, E=1, b=5)
func 0 (Transpose submission): hits:1766, misses:287, evictions:255

Summary for official submission (func 0): correctness=1 misses=287

TEST_TRANS_RESULTS=1:287
```

64×64（10 分）：运行结果截图

```
➜  cachelab-handout ./test-trans -M 64 -N 64

Function 0 (1 total)
Step 1: Validating and generating memory traces
Step 2: Evaluating performance (s=5, E=1, b=5)
func 0 (Transpose submission): hits:9002, misses:1243, evictions:1211

Summary for official submission (func 0): correctness=1 misses=1243

TEST_TRANS_RESULTS=1:1243
```

61×67（20 分）：运行结果截图

```
➜  cachelab-handout ./test-trans -M 61 -N 67

Function 0 (1 total)
Step 1: Validating and generating memory traces
Step 2: Evaluating performance (s=5, E=1, b=5)
func 0 (Transpose submission): hits:6194, misses:1985, evictions:1953

Summary for official submission (func 0): correctness=1 misses=1985

TEST_TRANS_RESULTS=1:1985
```

第 4 章 总结

4.1 请总结本次实验的收获

深刻理解了缓存系统以及其运行机制，理解了缓存的读策略、写策略和替换策略。

4.2 请给出对本次实验内容的建议

转置矩阵的优化程度不够