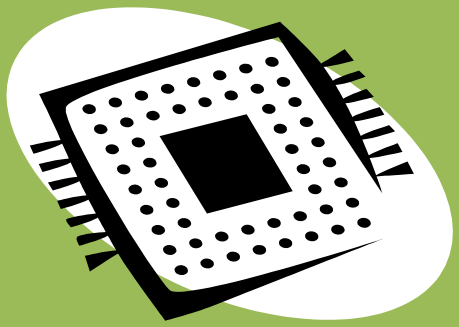
华中科技大学

2021

计算机系统结构 ·实验报告·

专　　业：	计算机科学与技术
班　　级：	CS1807
学　　号：	U201814745
姓　　名：	朱槐志
电　　话：	15623851632
邮　　件：	2607840393@qq.com
完成日期：	2021-4-29



计算机科学与技术学院

目 录

1 CACHE 模拟器实验 2

1.1 设计要求 2

1.2 方案设计 2

1.3 实验步骤 6

1.4 故障与调试 7

1.5 测试与分析 7

2 总结与心得 9

2.1 实验总结 9

2.2 实验心得 9

参考文献 10

1 Cache 模拟器实验

1.1 设计要求

实验内容：编写一个 200-300 行的 C 程序来模拟 Cache 缓存的行为。

输入：内存访问轨迹；

操作：模拟缓存相对内存访问轨迹的命中/缺失行为；

输出：命中、缺失和（缓存行）淘汰/驱逐的总数。

具体要求：

完成的 csim.c 文件应能接受与参考缓存模拟器 csim-ref 相同的命令行参数并产生一致的输出结果。

命令行格式：csim-ref [-hv] -s <s> -E <E> -b <b> -t <tracefile>

-h: 显示帮助信息（可选）

-v: 显示轨迹信息（可选）

-s <s>: 组索引位数

-E <E>: 关联度(每组包含的缓存行数)

-b <b>: 内存块内地址位数

-t <tracefile>: 内存访问轨迹文件名

1.2 方案设计

1.2.1 设计思路

数据结构：

（1）将 cache 行定义为 cache 的最小结构体；

（2）用 cache 行的数组定义一个 cache 组；

（3）用 cache 组的数组定义整个 cache。

（4）记录次数，输入参数相关变量定义为全局变量；

主函数步骤：

（1）获得命令行输入参数；

（2）初始化 cache；

（3）读取 trace 文件内容，根据文件内容中的参数访问缓存

（4）释放 cache 内存；

（5）调用实验包定义好的 printSummary()函数打印结果，结束。

1.2.2 设计原理

将整个 LRU 缓存模拟程序划分为以下模块：

cache_line：自定义 cache 行数据结构，数据成员为地址 tag，有效标志位 valid，连续未被访问时间计数器。

Initcache(): 该函数对 cache 初始化，根据命令行输入参数给 cache 分配内存

accessData(): 该函数定义了访问缓存时的行为，先判断是否命中，未命中则在对应组中查找空闲区域写入相应 data，如果没有空闲行的话则用 LRU 策略，找到最久未被访问的那行进行替换。

replayTrace(): 读取实验包的测试文本文件，从该文件中接受运行命令，根据命令中的参数对 cache 执行访问

freecache(): 释放 cache 内存；

LRU 算法实现：任务书中给出的一种算法为链表实现，本次实验中，我的数据结构定义为数组且考虑到访问命中时将链表中的块提前这一操作更复杂，因此采用计数器实现，每个缓存行里都记录了该缓存行连续没有被访问过的次数，需要淘汰时则淘汰技术最大的缓存行即可。不过数组的这种写法，效率上会不如链表实现。

1.2.3 设计过程

1. cache_line：结构体定义及各成员含义见下图 1.1

```c
//定义cache行的结构
typedef struct cache_line {
    char valid;
    mem_addr_t tag;
    //lru表示该cache已经连续多少次没有被访问
    unsigned long long int lru;
} cache_line_t;
```

图 1.1cache_line

其中 valid 表示该缓存行是否是空闲，tag 表示地址，lru 表示该缓存行连续未被访问的计数。

2. Initcache():实现见下图 1.2

```c
//cache初始化，分配空间
void initCache()
{
    int i,j;
    cache = (cache_set_t*) malloc(sizeof(cache_set_t) * S);//一个组×组数
    for (i=0; i<S; i++){
        cache[i]=(cache_line_t*) malloc(sizeof(cache_line_t) * E);
        for (j=0; j<E; j++){
            cache[i][j].valid = 0;
            cache[i][j].tag = 0;
            cache[i][j].lru = 0;
        }
    }

    //一个address分成三部分：tag index offset
    //                      数据    s    b
    //用来做与运算得到一个内存地址对应的组编号，
    set_index_mask = (mem_addr_t) (pow(2, s) - 1);
}
```

图 1.2 Initcache()

就是根据 cache 结构来给 cache 分配给定的内存大小

3. accessData(): 定义缓存访问行为，如图 1.3 所示

```
//E:相联度 即每一个组内有几块
for (i = 0; i < E; i++)
{
    //在如果tag相等且cache行不为空，表示命中
    if(cache_set[i].tag == tag && cache_set[i].valid == 1)
    {
        printf("hit ");
        hit_count++;
        //命中之后就把lru置为0，表示刚被访问过
        cache_set[i].lru = 0;
        hit = 1;
    }
    else
    {
        if(cache_set[i].valid == 1)
        {
            //没命中的话就需要将lru++，表示该cache行又没有被访问
            cache_set[i].lru ++;
        }
    }

}
//不在cache中，未命中
if(hit == 0)
```

图 1.3 accessData()

由于函数体略长，不方便截下来，说明如下:

根据地址 addr 获取组号，然后遍历该缓存组中的所有缓存行，如果找到了，则将 hit++，退出即可；如果没找到，则需要存入到缓存中，这时需要判断是否还有空闲缓存行。如果有空闲缓存行的话，将该缓存行 valid 置为 1，表示被占用；否则的话，找到最久未被访问的 cache 行进行替换。相应的 miss 和 eviction 自增。

4. replayTrace(): 以 M 指令为例，分别执行，扫描文件获取访存地址，打印信息，模拟访问 2 次。见图 1.4

```
//数据修改
case 'M':{
            fscanf(trace_fp,"%X, %d",(unsigned int*) &addr, &len);
            printf("%c %x %u ",operation, (int)addr,len);
            accessData(addr);
            accessData(addr);
            printf("\n");
            break;
}
```

图 1.4 M 指令

5. freeCache()：释放 cache 的内存空间，相当于 initCache 的逆操作，如图 1.5

```
//释放cache内存
void freeCache()
{
    int i;
    for(i=0;i<S;i++)
    {
        //释放cache组指针
        free(cache[i]);
    }
    //释放cache本身指针
    free(cache);
}
```

图 1.5 freeCache()

1.3 实验步骤

（1）实验环境：

阿里云服务器 ECS：linux64bit；

Xshell7：远程连接；

Xftp7：文件传输。

WSL 2：本机编译测试

（2）实验步骤

1. 根据实验指导文档登上阿里云网站，获取 ip 和密码
2. 在本机上通过 xshell 远程连接到主机
3. 在本机上通过 xftp 远程连接到主机，并把实验包复制到本地
4. 在本机上用 wsl 作为实验环境，编写 cism.c 函数
5. 编写完成后，make 编译，给 test-cism 增加执行权限
6. 运行 test-cism
7. 根据结果调试

1.4 故障与调试

（1）如图 1.6 编译错误，无法识别 pow 函数。

```
[root@iZuf6brcie45792fsmlph0Z cachelab-handout]# gcc csim.c cachela
/tmp/ccIMjOib.o: In function `initCache':
csim.c:(.text+0x154): undefined reference to `pow'
/tmp/ccIMjOib.o: In function `main':
csim.c:(.text+0x820): undefined reference to `pow'
csim.c:(.text+0x851): undefined reference to `pow'
```

图 1.6 编译错误

解决：结尾添加 “-lm”。

（2）如图 1.7，无法执行 test-csim 文件。

```
[root@iZuf6brcie45792fsmlph0Z cachelab-handout]# ./test-csim
-bash: ./test-csim: Permission denied
```

图 1.7 无法执行

解决：使用 chmod 指令增加文件的执行权限，具体指令为：chmod +x test-csim

1.5 测试与分析

测试结果如图 1.8 所示，测试输出结果与期望相符，说明自己实现的 LRU 替换算法正确。

```
root@silence:/mnt/d/cachelab-handout# ./test-csim
                        Your simulator     Reference simulator
Points (s,E,b)    Hits  Misses  Evicts    Hits  Misses  Evicts
     3 (1,1,1)       9       8       6       9       8       6  traces/yi2.trace
     3 (4,2,4)       4       5       2       4       5       2  traces/yi.trace
     3 (2,1,4)       2       3       1       2       3       1  traces/dave.trace
     3 (2,1,3)     167      71      67     167      71      67  traces/trans.trace
     3 (2,2,3)     201      37      29     201      37      29  traces/trans.trace
     3 (2,4,3)     212      26      10     212      26      10  traces/trans.trace
     3 (5,1,5)     231       7       0     231       7       0  traces/trans.trace
     6 (5,1,5)  265189   21775   21743  265189   21775   21743  traces/long.trace
    27

TEST_CSIM_RESULTS=27
root@silence:/mnt/d/cachelab-handout# code .
```

图 1.8 运行 test-csim 结果

2 总结与心得

2.1 实验总结

其实这次实验难在一开始的无从下手，因为实验包给的资料还是蛮少。不过一旦思路正确，缓存行的结构定义完成之后其实 LRU 算法本身并不难，只是细节部分需要注意。另外比较麻烦的点就是命令行参数的处理，因为实验要求中要求程序支持指定的多个参数配置，这部分使用 getopt 函数来实现比较方便。

2.2 实验心得

在本次实验中，我第一次使用了阿里云服务器，也算是一种别样的体验吧。

另外在本次实验中，在亲手实现 LRU 缓存算法的过程中，有许多的细节问题需要我们考虑，比如缓存行结构的定义，如何标志最久未被使用的缓存行，如何从文本文件中读取运行的指令参数等问题，这和我们看书了解 LRU 算法是完全不同的体验，还是老生常谈的问题，纸上得来终觉浅，光是看书的话，我们获得的知识是很浅的，只有自己亲手实践，在实践过程中不断思考和解决问题，掌握算法实现的细节，将知识吃透。进而将书本中的知识转换为自己的经验和阅历，成为自己的产出。

总之，这次实验是一次很好的体验，亲身实现过 LRU 算法之后，帮助我了解了计算机缓存实现算法的细节，收获很大。

参考文献

[1]《计算机系统结构教程（第二版）》------张晨曦，王志英

[2]《计算机组成原理（微课版）》------谭志虎

[3]https://blog.csdn.net/weixin_42294984/article/details/80738945------CSAPPCacheLab实验

•指导教师评定意见•

一、原创性声明

本人郑重声明本报告内容，是由作者本人独立完成的。有关观点、方法、数据和文献等的引用已在文中指出。除文中已注明引用的内容外，本报告不包含任何其他个人或集体已经公开发表的作品成果，不存在剽窃、抄袭行为。

特此声明!

作者签字:

二、对课程实验的学术评语（教师填写）

三、对课程实验的评分（教师填写）

评分项目 （分值）	报告撰写 （30 分）	课设过程 （70 分）	最终评定 （100 分）
得分			

指导教师签字: _____ **2021-05-02**