ICS-LAB8
Dynamic Storage Allocator
动态内存分配器

哈尔滨工业大学
计算机科学与技术学院

2021年6月8日

一、实验基本信息

- **实验类型：设计型实验**
- **实验目的**
  - 理解现代计算机系统虚拟存储的基本知识
  - 掌握C语言指针相关的基本操作
  - 深入理解动态存储申请、释放的基本原理和相关系统函数
  - 用C语言实现动态存储分配器，并进行测试分析
  - 培养Linux下的软件系统开发与测试能力
- **实验指导教师**
  - 任课教师：刘宏伟、史先俊、郑贵滨、吴锐
- **实验分组**
  - 一人一组

一、实验基本信息

- 实验学时：3
- 实验分数：5，本次实验按100分计算，折合成总成绩的5分。
- 实验地点：G712、G709
- 实验环境与工具：
  - X64 CPU；2GHz；2G RAM；256GHD Disk 以上
  - Windows7 64位以上；VirtualBox/Vmware 11以上；Ubuntu 16.04 LTS 64位/优麒麟 64位

一、实验基本信息

- **学生实验准备：禁止准备不合格的学生做实验**
  - 个人笔记本电脑
  - 实验环境与工具所列明软件
  - 参考手册: Linux环境下的命令；GCC手册；GDB手册
  - http://docs.huihoo.com/c/linux-c-programming/ C汇编Linux手册
  - http://csapp.cs.cmu.edu/3e/labs.html CMU的实验参考
  - http://www.linuxidc.com/ http://cn.ubuntu.com/ http://forum.ubuntu.org.cn/

二、实验要求

- **学生应穿鞋套进入实验室**
- **进入实验室后在签到簿中签字**
- **实验安全与注意事项**
  - 禁止使用笔记本电脑以外的设备
  - 学行生不得自行开关空调、投影仪
  - 学生不得自打开窗户
  - 不得使用实验室内其他实验箱、示波器、导线、工具器等
  - 认真阅读消防安全撤离路线
  - 突发事件处理：第一时间告知教师，同时关闭电源开关。
- **遵守学生实验守则，爱护实验设备，遵守操作规程，精心操作，注意安全，严禁乱拆乱动。**
- **实验结束后要及时关掉电源，对所用实验设备进行整理，设备摆放和状态恢复到原始状态。**
- **桌面整洁、椅子归位，经实验指导教师允许后方可离开**

三、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。
- 熟知C语言指针的概念、原理和使用方法
- 了解虚拟存储的基本原理
- 熟知动态内存申请、释放的方法和相关函数
- 熟知动态内存申请的内部实现机制：分配算法、释放合并算法等

四、实验内容与步骤

- **1.环境建立**
  - Ubuntu + gcc
- **2.获得实验包**
  - 从实验教师处获得下 malloc-handout-hit.tar
  - 也可以从课程QQ群下载，也可以从其他同学处获取。
  - **HIT与CMU的不同，**提供了隐式空闲链表的例子代码了mm-implicit.c，但**缺少合并函数**的实现部分。
- **3. 实验报告解压（linux下）**

解压命令 unix>**tar xvf malloc-handout-hit.tar**

四、实验内容与步骤

- **4.0 实验包内容介绍**
  - mm.c 实验需要修改的源代码文件
  - memlib.c 模拟内存系统，为实验提供如下可用函数：
    - void *mem_sbrk(int incr)：将堆增加incr字节，参数 incr是正整数，函数返回新增加堆区域的首字节地址，incr不可以是负数。
    - void *mem_heap_lo(void): 返回指向堆中首字节的指针。
    - void *mem_heap_hi(void): 返回指向堆中末尾字节的指针。
    - size_t mem_heapsize(void): 返回堆大小（字节总数）
    - size_t mem_pagesize(void): 返回系统的页尺寸(字节数，Linux系统是4K）

四、实验内容与步骤

- **4.0 实验包内容介绍(续...)**
  - mm-implicit.c 采用隐式空闲链表的分配器代码：缺少空闲块合并函数coalesce的代码(mm-implicit.c 的第301行处)
  - mdriver.c 性能评测程序，检查mm.c中实现的分配器的正确性、空间利用率、吞吐率
  - 轨迹文件：malloclab-handout\traces
  - malloclab.pdf：实验说明文档（英文）

四、实验内容与步骤

4.1 实验任务

实现自定义版本的malloc, free 和 realloc函数：

- int mm_init(void);
- void *mm_malloc(size_t size);
- void mm_free(void *ptr);
- void *mm_realloc(void *ptr, size_t size);

这四个函数在mm.h声明，在mm.c中实现。

四、实验内容与步骤

4.2实验任务——函数说明

- int mm_init(void)

应用程序（例如轨迹驱动的测试程序mdriver）在使用mm_malloc、mm_realloc或mm_free之前，首先要调用该函数进行初始化。例如申请初始堆区域。

返回值：0表示正常，-1表示有错误；

- void *mm_malloc(size_t size)

申请有效载荷至少是参数“size”指定大小的内存块，返回该内存块地址首地址（可以使用的区域首地址）。申请的整个块应该在对齐的区间内，并且不能与其他已经分配的块重叠。返回的地址应该是8字节对齐的（地址%8==0）。

四、实验内容与步骤

4.2实验任务——函数说明

- void mm_free(void *ptr)

释放参数“ptr”指向的已分配内存块，没有返回值。指针值ptr应该是之前调用mm_malloc或mm_realloc返回的值，并且没有释放过。

- void *mm_realloc(void *ptr, size_t size)

–如ptr是空指针NULL,等价于mm_malloc(size)

–如果参数size为0，等价于mm_free(ptr)

–如ptr非空, 它应该是之前调用mm_malloc或mm_realloc返回的数值，指向一个已分配的内存块。

四、实验内容与步骤

- **4.2实验任务——函数说明**
  - void *mm_realloc(void *ptr, size_t size) (续…)

调用mm_realloc是为了将ptr所指向内存块（旧块)的大小变为size，并返回新内存块的地址。

注意：

(1)返回的地址与原地址可能相同，也可能不同，这依赖于算法的实现、旧块内部碎片大小、参数size的数值。

(2)新内存块中，前min(旧块size，新块size)个字节的内容与旧块相同，其他字节未做初始化。

四、实验内容与步骤

4.2 实验任务——函数说明

堆的一致性检查int mm_check(void)

建议重点关注的方面：

- 空闲列表中的每个块是否都标识为free（空闲）？
- 是否有连续的空闲块没有被合并？
- 是否每个空闲块都在空闲链表中？
- 空闲链表中的指针是否均指向有效的空闲块？
- 分配的块是否有重叠？
- 堆块中的指针是否指向有效的堆地址？

int mm_check(void)函数，检查重要的不变量和一致性条件。当且仅当堆是一致的，才能返回非0值。

★提交代码文件mm.c的时候，将mm_check的所有调用注释，以免影响速度，降低吞吐率。

四、实验内容与步骤

- **4.3 空闲块合并函数★**
  - 函数原型：static void *coalesce(void *bp)
  - 参　　数：bp是要回收的空闲块指针
  - 功　　能：将要回收的空闲块和临近的空闲块（如果有的话）合并成一个大的空闲块。
  - 返 回 值：合并后的空闲块指针
  - 代码实现：
    - 参考教材9.9.11节的相关内容，针对空闲块合并的4种情况，进行合并处理。

四、实验内容与步骤

- **5 注意事项**
  - 不能修改mm.c中的接口函数（函数声明不能改）
  - 不可以使用内存管理相关的库函数或系统调用，如malloc, calloc, free, realloc, sbrk, brk等
  - 不可以定义任何全局或静态的复合数据结构，例如：数据、结构体、树或链表
  - 可以定义全局标量型变量，例如整型、浮点型、指针。
  - 对齐：为了和libc的malloc一致，使用8字节边界对齐，即实验实现的malloc函数、remalloc函数应该返回8字节对齐的边界（指针值%8==0)
    - 测试驱动程序会强制检查这一点

四、实验内容与步骤

- **6 优化方案建议**
  - 方案1：显式空闲链表 + 基于边界标签的空闲块合并 + 首次适配
  - 方案2：使用红黑树（最优的方法）

四、实验内容与步骤

- **6 优化方案建议**
  - 参考教材，使用宏函数实现一些指针的算术运算
  - 分阶段完成、测试
    - 阶段1：前9个trace文件包含malloc和free的测试
    - 阶段2：最后2个trace文件包含对realloc、malloc和free的测试

    可以利用malloc和free实现realloc的功能，但如果获取高性能，需要写单独的realloc函数。
  - 使用profiler，工具gprof对优化性能很有帮助

四、实验内容与步骤

- **7 性能评测方法**
  - 实验目标：能正确、高效、快速地运行
  - 生成可执行评测程序文件的方法

    linux>make
  - 评测方法:

    mdriver [-hvVa] [-f <file>]

    选项：

    -a　　不检查分组信息

    -f <file>　使用 <file>作为单个的测试轨迹文件

    -h　　显示帮助信息

    -l　　也运行C库的malloc

    -v　　输出每个轨迹文件性能

    -V　　输出额外的调试信息

四、实验内容与步骤

- **7 性能评测方法**

轨迹文件：指示测试驱动程序mdriver以一定顺序调用mm_malloc, mm_realloc 和mm_free

- amptjp-bal.rep
- cccp-bal.rep
- cp-decl-bal.rep
- expr-bal.rep
- coalescing-bal.rep
- random-bal.rep
- random2-bal.rep
- binary-bal.rep
- binary2-bal.rep
- realloc-bal.rep
- realloc2-bal.rep

amptjp-bal.rep
3000000//推荐的堆尺寸
2847//alloc、realloc的id数量
5694 //操作总数
1
a 0 2040
a 1 2040
a 2 48
a 3 4072
a 4 4072
a:alloc
r:realloc
f: free

四、实验内容与步骤

每个人用不同的机器，速度不同，评分仅供参考。统一评测才有意义。

- 8 性能评分

性能分pindex是空间利用率和吞吐率的线性组合：

```
p1 = UTIL_WEIGHT * avg_mm_util;
if (avg_mm_throughput > AVG_LIBC_THRUPUT) {
   p2 = (double)(1.0 - UTIL_WEIGHT);
}
else {
   p2 = ((double) (1.0 - UTIL_WEIGHT)) *
            (avg_mm_throughput/AVG_LIBC_THRUPUT);
}
pindex = (p1 + p2)*100.0;
```

- UTIL_WEIGHT = 0.6
- AVG_LIBC_THRUPUT=600000
- avg_mm_util 测试实验代码malloc得到的平均空间利用率
  avg_mm_util = hwm/heapsize = brk/heapsize //hwm is high water mark
- avg_mm_throughput 测试实验代码malloc得到的平均吞吐率(次/秒, ops/sec)

9 教材示例说明

- **mem_heap:堆的起始位置**
- **mem_brk:堆的已用边界（最大地址）**
  - [mem_heap,mem_brk）：已经动态分配过的内存（即便是malloc之后free的也算）
  - [mem_brk,]未分配的虚拟内存
- **mem_max_addr:系统内存最大地址（模拟值）**
- **void mem_init(void):内存系统的初始化**
- **void *mem_sbrk(int incr): 模拟系统的sbrk()函数**

五、实验报告格式与评分

- **实验报告格式**

  按照实验报告模板所要求的格式与内容书写。

- **评　　分**

  本次实验成绩按100分计

  - 按时上课，签到5分
  - 按时下课，不早退5分
  - 课堂表现：10分，不按操作规程、非法活动扣分。
  - 实验报告：80分。具体参见实验报告各环节的分值

- **在实验报告中，对每一任务，按照要求用文字详细描述**

- **杜绝抄袭！发现 全 0 分！**

9.实验提交

- **提交内容——3个文件：**
  - mm.c文件（非压缩格式）
  - 教师将使用自动评分工具，对代码进行自动评测。
  - 实验报告文件word版（填写4.4 自测试评分）
  - 实验报告pdf版
- **提交时间：实验后 2周内提交**
- **提交方式：**
  - 学生提交1个压缩包给课代表
  - 课代表将自己班级的实验打包后，提交1个包给授课教师/助教