

OS小实验汇报

致理-信计01 单敬博 2020012711

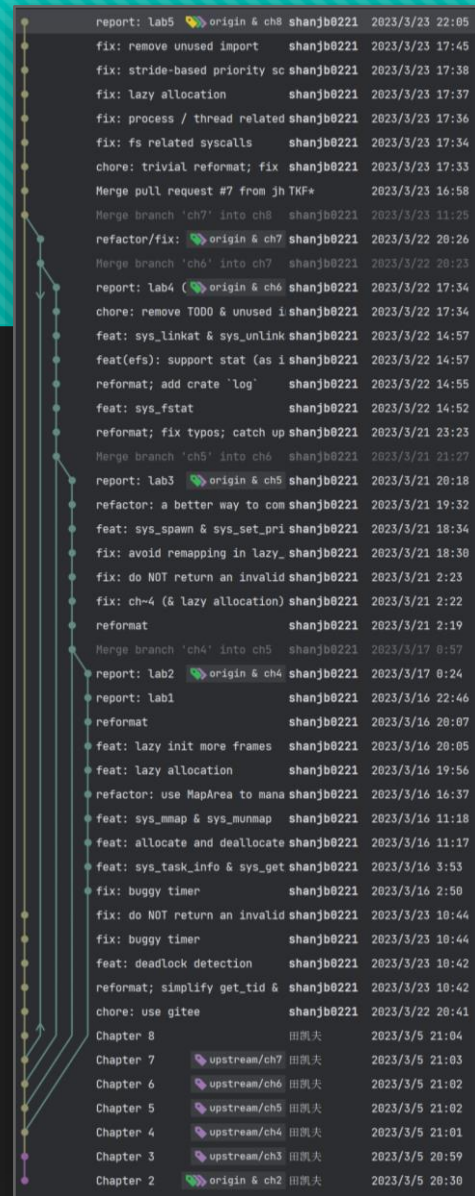
搞定的事情

Chapter[3-8] 编程作业 & 问答作业 ecall

扩展功能：惰性页面分配

Trap Context, User Stack, sbrk, mmap, ...

ch[4-8]均支持 ~~merge~~ 了好久



report: lab5	origin & ch8	shanjib0221	2023/3/23 22:05
fix: remove unused import		shanjib0221	2023/3/23 17:45
fix: stride-based priority sc		shanjib0221	2023/3/23 17:38
fix: lazy allocation		shanjib0221	2023/3/23 17:37
fix: process / thread related		shanjib0221	2023/3/23 17:36
fix: fs related syscalls		shanjib0221	2023/3/23 17:34
chore: trivial reformat; fix		shanjib0221	2023/3/23 17:33
Merge pull request #7 from jh TKF*			2023/3/23 16:58
Merge branch 'ch7' into ch8		shanjib0221	2023/3/23 11:26
refactor/fix:	origin & ch7	shanjib0221	2023/3/22 20:26
Merge branch 'ch6' into ch7		shanjib0221	2023/3/22 20:23
report: lab4	origin & ch6	shanjib0221	2023/3/22 17:34
chore: remove TODO & unused i		shanjib0221	2023/3/22 17:34
feat: sys_linkat & sys_unlink		shanjib0221	2023/3/22 14:57
feat(efs): support stat (as i		shanjib0221	2023/3/22 14:57
reformat; add crate 'log'		shanjib0221	2023/3/22 14:55
feat: sys_fstat		shanjib0221	2023/3/22 14:52
reformat; fix typos; catch up		shanjib0221	2023/3/21 23:23
Merge branch 'ch5' into ch6		shanjib0221	2023/3/21 21:27
report: lab3	origin & ch5	shanjib0221	2023/3/21 20:18
refactor: a better way to com		shanjib0221	2023/3/21 19:32
feat: sys_spawn & sys_set_pri		shanjib0221	2023/3/21 18:34
fix: avoid remapping in lazy_		shanjib0221	2023/3/21 18:30
fix: do NOT return an invalid		shanjib0221	2023/3/21 2:23
fix: ch=4 (& lazy allocation)		shanjib0221	2023/3/21 2:22
reformat		shanjib0221	2023/3/21 2:19
Merge branch 'ch4' into ch5		shanjib0221	2023/3/17 0:57
report: lab2	origin & ch4	shanjib0221	2023/3/17 0:24
report: lab1		shanjib0221	2023/3/16 22:46
reformat		shanjib0221	2023/3/16 20:07
feat: lazy init more frames		shanjib0221	2023/3/16 20:05
feat: lazy allocation		shanjib0221	2023/3/16 19:56
refactor: use MapArea to mana		shanjib0221	2023/3/16 16:37
feat: sys_mmap & sys_munmap		shanjib0221	2023/3/16 11:18
feat: allocate and deallocate		shanjib0221	2023/3/16 11:17
feat: sys_task_info & sys_get		shanjib0221	2023/3/16 3:53
fix: buggy timer		shanjib0221	2023/3/16 2:50
fix: do NOT return an invalid		shanjib0221	2023/3/23 10:44
fix: buggy timer		shanjib0221	2023/3/23 10:44
feat: deadlock detection		shanjib0221	2023/3/23 10:42
reformat; simplify get_tid &		shanjib0221	2023/3/23 10:42
chore: use gitee		shanjib0221	2023/3/22 20:41
Chapter 8		田凱夫	2023/3/5 21:04
Chapter 7	upstream/ch7	田凱夫	2023/3/5 21:03
Chapter 6	upstream/ch6	田凱夫	2023/3/5 21:02
Chapter 5	upstream/ch5	田凱夫	2023/3/5 21:02
Chapter 4	upstream/ch4	田凱夫	2023/3/5 21:01
Chapter 3	upstream/ch3	田凱夫	2023/3/5 20:59
Chapter 2	origin & ch2	田凱夫	2023/3/5 20:30

遇到的困难 (1/6)

- 不太熟悉rust的所有权机制，遇到了很多次：

`panic: BorrowMutError!`

- 打开LOG=TRACE大致定位挂掉的位置

- 及时drop / 直接不保存借用

遇到的困难 (2/6)

- `ch3_taskinfo: sleep(500)` 只睡了 480ms

- `get_time_us` 没整除!

- 改成先乘后除

- 优化: 提前都除掉gcd

遇到的困难 (3/6)

- 实现带溢出处理的Stride调度后:

 - sleep一段时间之后再fork/spawn出来的进程一段时间内不被执行/一直在执行

- Stride初始化成0, 破坏了 $S_{\max} - S_{\min} \leq \text{BigStride}/2$

- fork/spawn的时候把新进程的Stride与当前进程“同步”

遇到的困难 (4/6)

○ `ch8_deadlock_sem1: sleep(500)` 睡死过去力 (悲)

○ `sleep` 依赖 `sys_get_time`, `ch8` 没实现! 说好的不需要merge呢?

○ 变动确实比较大, `cherry-pick` 大失败

○ 于是直接激情merge! 然后修了半天conflict

警告

本次实验框架变动较大, 且改动较为复杂, 为降低同学们的工作量, 本次实验不要求合并之前的实验内容, 只需通过 `ch8` 的全部测例和其他章节的基础测例即可。你可以直接在实验框架的 `ch8` 分支上完成以下作业。

遇到的困难 (5/6)

- 实现扩展功能(惰性页面分配)一时爽，但是.....
 - 内核查询`PageTable(.translate)`也可能触发缺页
 - 这时需要`MemorySet`介入分配分配新页面
 - 于是使用`token`构造PT再`translate`的所有函数(`translated_*`)全寄了
- 把所有`translated_*`修改为`MemorySet`的成员函数
- 将绝大部分`translate`替换为由`MemorySet`提供的惰性分配版本

遇到的困难 (6/6)

- 需要向前兼容
- 点击merge -> 修半天conflict
- 我也许不该手欠去点《重新格式化代码》按钮 :(
- (引用上一个困难) 每次都要修复所有新的translated_*

一些吐槽

- ~~merge~~工作量好大
- 实现扩展功能之后merge更繁琐了 ~~下次不搞了(逃~~
- 每次make都要重新编译所有测例，很费时间 ~~优化空间巨大~~

附录

```
/// Find PageTableEntry by VirtPageNum
fn find_pte(&self, vpn: VirtPageNum) -> Option<&mut PageTableEntry> {
    let idxs = vpn.indexes();
    let mut ppn = self.root_ppn;
    let mut result: Option<&mut PageTableEntry> = None;
    for (i, idx) in idxs.iter().enumerate() {
        let pte = &mut ppn.get_pte_array()[*idx];
        if i == 2 {
            result = Some(pte);
            break;
        }
        if !pte.is_valid() {
            return None;
        }
        ppn = pte.ppn();
    }
    result
}
```

```
115 115
116 116
117 117
118 118
119 119
120 120
121 121
122 122
123 123
124 124
125 125
126 126
127 127
128 128
129 129
130 130
131 131
132 132
```

```
/// Find PageTableEntry by VirtPageNum
fn find_pte(&self, vpn: VirtPageNum) -> Option<&mut PageTableEntry> {
    let idxs = vpn.indexes();
    let mut ppn = self.root_ppn;
    let mut result: Option<&mut PageTableEntry> = None;
    for (i, idx) in idxs.iter().enumerate() {
        let pte = &mut ppn.get_pte_array()[*idx];
        if !pte.is_valid() {
            return None;
        }
        if i == 2 {
            result = Some(pte);
            break;
        }
        ppn = pte.ppn();
    }
    result
}
```

感谢聆听 & 欢迎提问

致理-信计01 单敬博 2020012711

搞定的事情

Lab1

- 在TCB中添加TaskMeta，系统调用次数只记录已支持的
- TaskManager提供新接口：
 - 获取元数据get_current_meta；记录系统调用record_current_syscall
- 在任务第一次被执行时记录开始时间；分发系统调用更新元数据
- 实现sys_task_info系统调用

搞定的事情

Lab2

- 向前兼容：添加`write_buffer<T>`函数，在S态向用户虚存写入
- 在`MapArea`的基础上进行封装，实现`sys_mmap`与`sys_munmap`
- 进行了部分重构使得出错时不产生`panic`而是返回`false`
- 扩展功能：惰性页面分配

搞定的事情

Lab3

- 向前兼容
- 实现了 `sys_spawn` 及 `TCB.spawn`:
参照 `sys_{fork,exec}` 及 `TCB.{fork,exec}` 实现，但避免了复制父进程地址空间
- 实现了 `sys_set_priority` 及 `PriorityTaskManager`
将 `u64` 封装为 `Stride`，支持溢出后正确比较大小
在 `TCB` 中添加 `TaskStride`，每次进入任务前 `stride` 自增 `BigStride/priority`
为 `TCB` 实现了 `[Partial]{Eq,Ord}`，根据 `stride` 进行比较；使用 `BinaryHeap` 实现了等待任务的优先队列

搞定的事情

Lab4

- 向前兼容
- 修改easy-fs，使Inode提供info、link、unlink接口
 - 在DiskInode中添加引用计数nlink
 - 添加回收Inode、「根据block_id与offset计算inode_id」的接口
 - 添加dirent的回收与重分配功能
 - unlink时若引用计数归零则回收文件的数据与Inode，实现文件删除
- 实现sys_{fstat,linkat,unlinkat}

搞定的事情

Lab5

- ⊖ ~~艰难地~~向前兼容
- 在TCB中记录线程持有的资源与正在阻塞等待的资源
- 在PCB中记录是否开启死锁检测，实现
`sys_enable_deadlock_detect`
- 在资源相关的系统调用及资源的分配函数中进行插桩，更新线程的资源使用情况
- 将死锁检测算法实现为PCB的函数，在获取资源前调用进行检查