Task 2

2.1 Priority Schduling

Target:
1.preemtive schedule
2.Avoid busy waiting

Modified(mainly):
1.The function is related with going to ready state.

```
thread_create();
thread_unblock();
thread_yield();
```

Change the uneffective `push back` to `list_insert_ordered` .

And notice when a thread create,if the create priority is less than the new priority,then yield and rescheduled.Just like what `fork()` do in liunx.

2.2 Priority Donation

Target:
1.Normal recursive donation like nest donation
2.The priority queue design for lock,semaphore,condition variable.

Modified(mainly)
1.Heap Design
which fitts the lock ,semaphore and condition variable.

```
typedef struct MaxHeap
{
        int size;
        int heap[10];
}MaxHeap;

...

void swim(MaxHeap *);
void swap(int*, int*);
void sink(int, MaxHeap*);
void heap_push(int, MaxHeap*);
//int extract();
void heap_remove(int, MaxHeap*);
int heap_top(MaxHeap*);
bool heap_empty(MaxHeap*);
void heap_init(MaxHeap*);
```

2.

`atomic modified` during acquiring lock,releasing lock
and set priority

```
void hold_the_lock(struct lock *);
void remove_the_lock(struct lock *);
//bool lock_cmp(const struct list_elem *,const struct list_elem *,void * aux); // Formatted
void donate(struct thread *);
void update(struct thread *);
```

Data Structures

Q2.1

1.

For the priority queue of donation of holding lock,
I design a max heap so that it can get the max donation in time complexity $O(log(n))$,which is better than my list desigin which cost $O(n)$.

```
typedef struct MaxHeap
{
        int size;
        int heap[10];
}MaxHeap;

...

struct thread{
...
MaxHeap lock_heap;
...
};
```

2.

`waiting_locks` mean the lock list that the thread is waiting.And the `raw_priority` is made to deal with the `priority-donate-lower` test,which ensure the highest priority that a thread can own.Both of them are made for recursively donation.

```
struct thread{
    ...
    int raw_priority;
    struct lock * waiting_locks;
    ...
}
```

Q2.2

1.The data structure to track donation
(Previous --- Based on List)

```
struct thread{
...
struct list hloding_locks;
...
};

struct lock
{
    ...
     int max_priority;
    ...
};
```

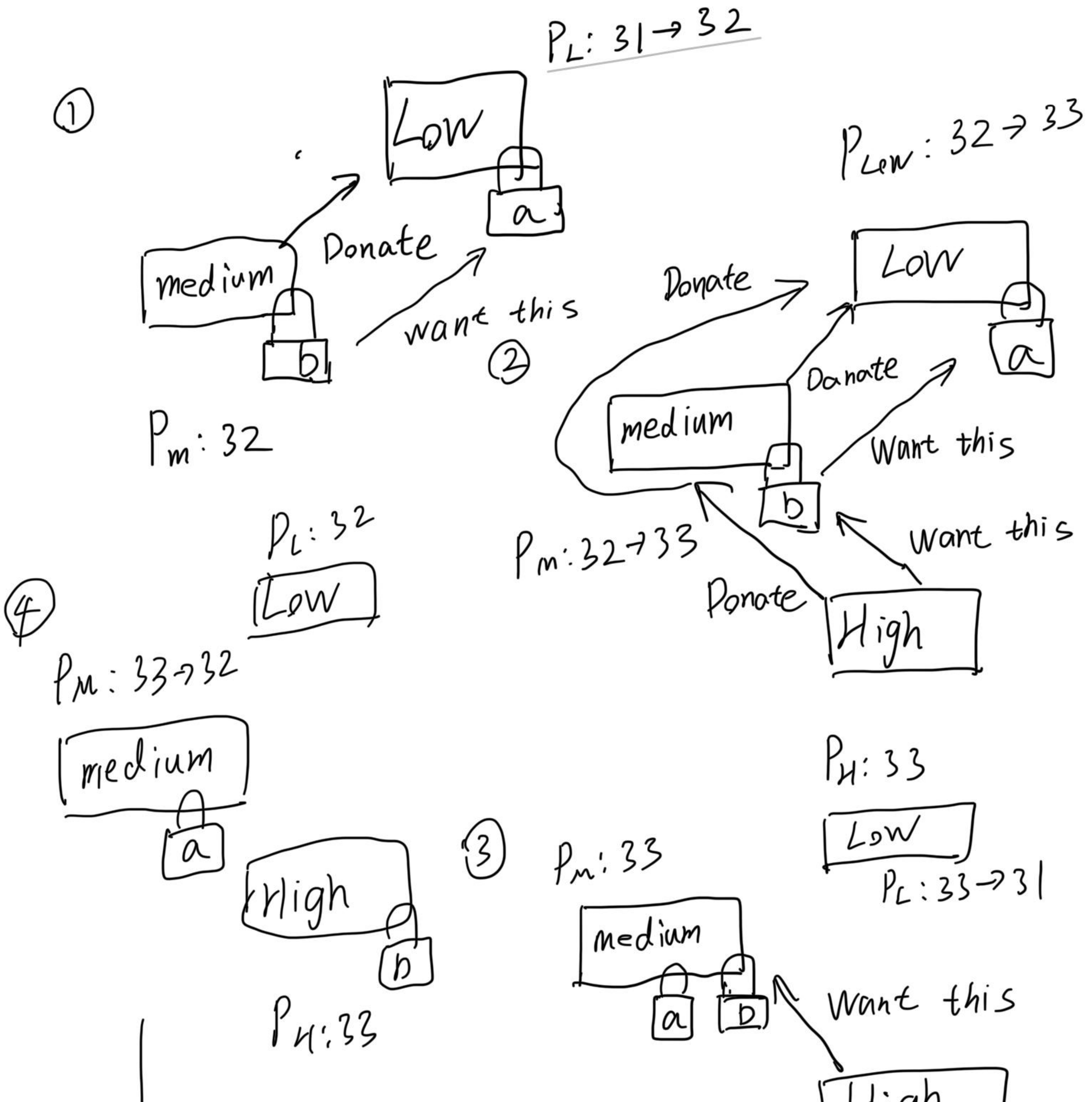
(Improved --- Based on Heap)

```
struct thread{
...
MaxHeap lock_heap;
...
};


struct lock
{
    ...
     int max_priority;
    ...
};
```

2.Nested Donation

Priority - Donate - Nest Test (Based On)



→ Then run High → Medium → Low

Algorithm

Q2.3

For locks,I made a MaxHeap then it can get the highest from top easily.

(Can be Modified below --- Use Heap)
For semaphore,I made it to insert to list orderly and pick it from head which make same effect.

Condvar...
(Modified Here)

Q2.4

Firstly,we can make up easily in use of the provided List API to deal with the recursively donation in 'lock_acquire()'.Just keep memory of locks each thread hold or waiting.And then search holding lock recursilvely and donate one by one.

Q 2.5

Make the donated thread doante back its priority to raw priority,then push it again to the heap.

Synchronization

Maybe after the thread's priority modifying,the thread
get a switch to anther thread.Then `thread_yield()` is not going to happen immetialy which make a wrong schduled.

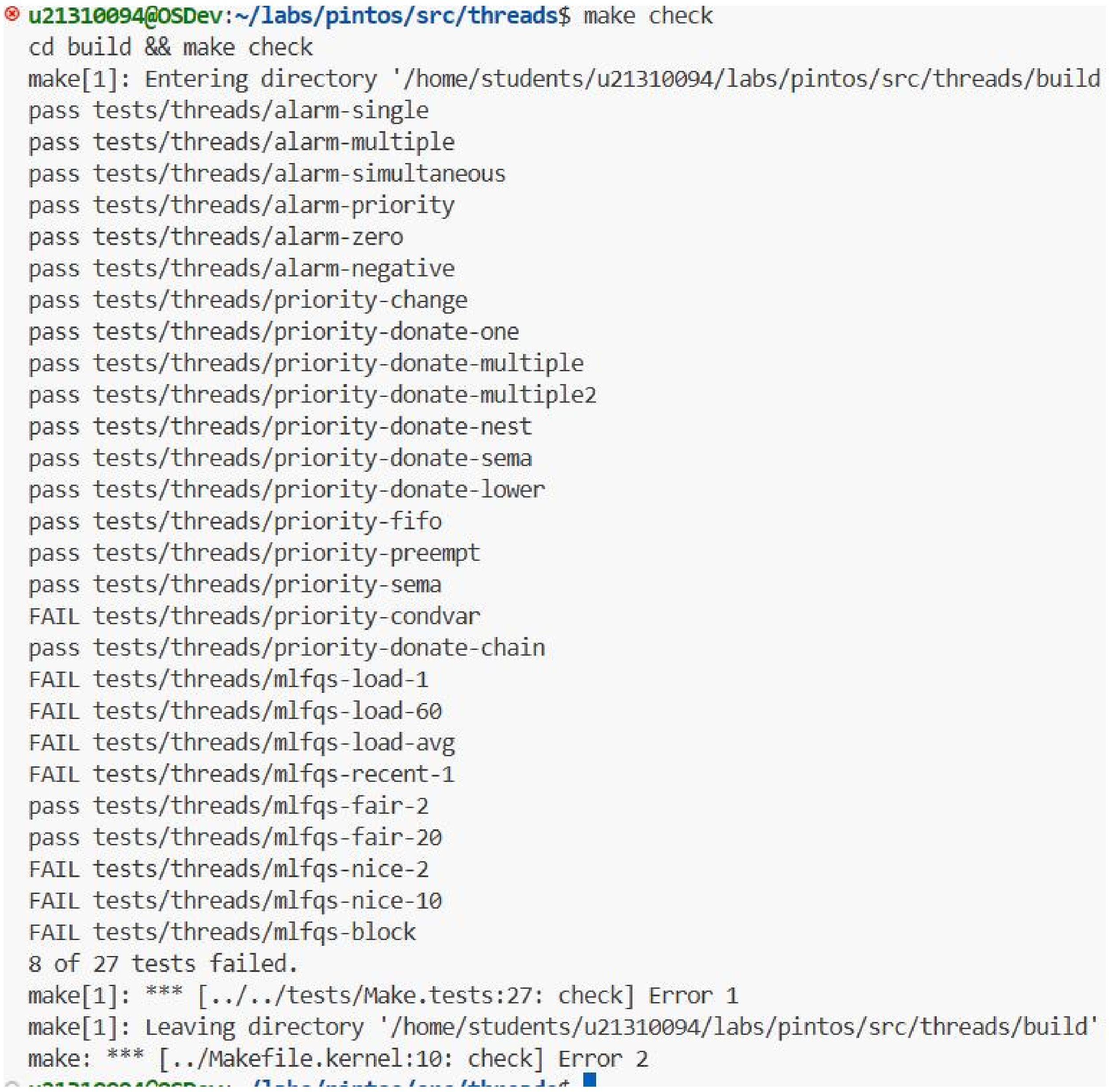
There are much like this.To prevent them,I drive interrupt by `intr_disable()` and roll back by
`intr_set_level(prev_level)` ,which is able to call as
`atomic modified` .It is necessary espiecially in such
basic level.

Rationale

Because heap support insert and pop in time complexity $(log(n)),quite effectively and fit those sceniries.

Maybe Red-Black tree will be better,since it plays quite an vital role in modern OS design.

(代码运行结果)



```
u21310094@OSDev:~/labs/pintos/src/threads$ make check
cd build && make check
make[1]: Entering directory '/home/students/u21310094/labs/pintos/src/threads/build
pass tests/threads/alarm-single
pass tests/threads/alarm-multiple
pass tests/threads/alarm-simultaneous
pass tests/threads/alarm-priority
pass tests/threads/alarm-zero
pass tests/threads/alarm-negative
pass tests/threads/priority-change
pass tests/threads/priority-donate-one
pass tests/threads/priority-donate-multiple
pass tests/threads/priority-donate-multiple2
pass tests/threads/priority-donate-nest
pass tests/threads/priority-donate-sema
pass tests/threads/priority-donate-lower
pass tests/threads/priority-fifo
pass tests/threads/priority-preempt
pass tests/threads/priority-sema
FAIL tests/threads/priority-condvar
pass tests/threads/priority-donate-chain
FAIL tests/threads/mlfqs-load-1
FAIL tests/threads/mlfqs-load-60
FAIL tests/threads/mlfqs-load-avg
FAIL tests/threads/mlfqs-recent-1
pass tests/threads/mlfqs-fair-2
pass tests/threads/mlfqs-fair-20
FAIL tests/threads/mlfqs-nice-2
FAIL tests/threads/mlfqs-nice-10
FAIL tests/threads/mlfqs-block
8 of 27 tests failed.
make[1]: *** [../../tests/Make.tests:27: check] Error 1
make[1]: Leaving directory '/home/students/u21310094/labs/pintos/src/threads/build'
make: *** [../Makefile.kernel:10: check] Error 2
```