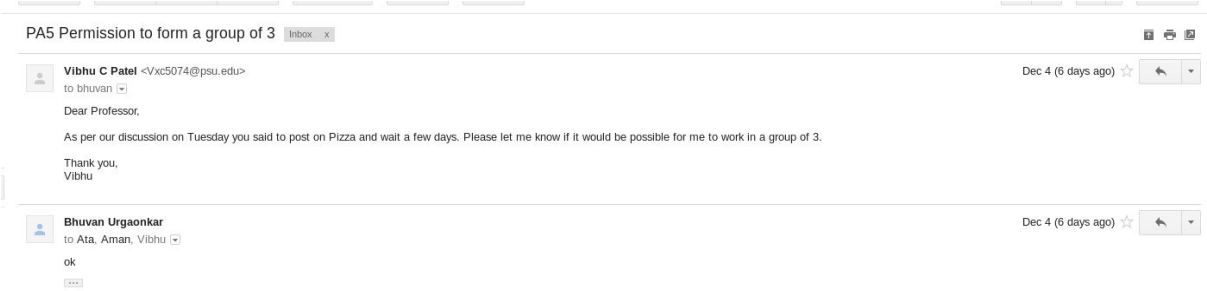
Vibhu C. Patel
Scott Mitchell
Parth Patel

Permission to work in a group of three for PA5 by the Professor:


PA5 Permission to form a group of 3 Inbox x

Vibhu C Patel <Vxc5074@psu.edu> Dec 4 (6 days ago)
to bhuvan

Dear Professor,

As per our discussion on Tuesday you said to post on Pizza and wait a few days. Please let me know if it would be possible for me to work in a group of 3.

Thank you,
Vibhu

Bhuvan Urgaonkar Dec 4 (6 days ago)
to Ata, Aman, Vibhu

ok

Pseudocode:

Synchronization:

```
while (1) {
        pthread_mutex_lock(&lock);  // lock this critical section where you're adding a request to the buffer to make sure there are no race conditions
        if (buff_count < limit) {          //Only add to buffer if it's not full
                if(b_head == NULL && b_tail == NULL)
                {
                        b_head = b_tail = temp1;
                }
                else{
                        b_tail->next = temp1;
                        b_tail = temp1;
                }
                buff_count ++;

                temp1->req_id = req;
                req++;
                struct timespec startTime;
                clock_gettime(CLOCK_MONOTONIC, &startTime);
                printf("REQUEST %d issued at %lld.%ld\n", temp1->req_id,(long long) startTime.tv_sec, startTime.tv_nsec);                    //for timing
                pthread_mutex_unlock(&lock);        //unlock after finishing critical section
                alarm(1);       //this is for the buffer timer, alarm every 1 sec
                break;          //this will break the while loop after it added the request
        }
        pthread_mutex_unlock(&lock);        //unlock critical section and loop again
 }
```

Buffer Timer:

In driver.c:

```
void alarm_event() {    //our alarm event will just set a variable to 1, this will be used in disk.c to allow buffers that are not full.
  should_run = 1;
 }
```

In init:

```
  struct sigaction action;          //we declare our alarm variables.
  action.sa_handler = &alarm_event;
  action.sa_flags = 0;  // Restart interrupted system calls
  sigemptyset(&action.sa_mask);
  sigaction(SIGALRM, &action, NULL);
```

In disk_ops:

```
while(buff_count < limit && !should_run); //Spin lock, if should_run == 1 then it will be able to get out of the while loop and run the rest of the diskops code.
should_run = 0;           //set should run back to 0.
```

Average service times:

FCFS:

sample_input.txt and limit = 1
(1.2216763 + 1.2137493 + 1.2116819)/3 = 1.2157025 seconds

sample_input_2.txt and limit = 2
(1.2128576+0.4205336+1.2137376+1.2135907)/4 = 1.015179875 seconds

sample_input_3.txt and limit = 3
(1.2363571+0.6272314+1.2115981+1.4117936+1.2131415+2.2018593)/6 = 1.31699683333 seconds

sample_input_4.txt and limit = 5
(0.6341529+0.10486104+0.8388603+1.7906341+0.14071732+0.10366184+1.5729920+1.2024585+1.2356684+1.7781254)/10 = 0.94021318 seconds

Elevator:

sample_input.txt and limit = 1
(0.2163500+0.2108185+0.2103575)/3 = 0.21250866666 seconds

sample_input_2.txt and limit = 2
(1.2099612+0.4194091+1.2127129+1.2140469)/4 = 1.014032525 seconds

sample_input_3.txt and limit = 3
(1.2254884+1.2136569+1.2112544+1.4182945+0.4213924+2.1965836)/6 = 1.2811117 seconds

sample_input_4.txt and limit = 5
(0.20322112+1.15652683+0.18243819+0.24575916+0.13522070+0.4159240+1.9341466+1
.22177757+0.996875970+1.6717149)/10 = 0.816360504 seconds