



ChatGPT 4o ▾



ChatGPT

Explore GPTs

Today

COW Fork Efficiency Optimization

memmove与uvmunmap顺序

Read Write Request Handling ...

Read Write Thread Handling

Create folder and clone repo

Yesterday

Joint PDF Transformation U V

Update Timed Wait Functions

```
void* read1(void* args) {
    query* q = (query*)args;
    file* use = q->file;
    struct timespec entry;
    clock_gettime(CLOCK_REALTIME, &entry);
    struct timespec timeout;
    timeout.tv_sec = entry.tv_sec + t;
    timeout.tv_nsec = 0;

    if (pthread_mutex_timedlock(&use->users, &timeout) == -1) {
        printf(RED "User %d canceled the request due to no response at
%d seconds\n" RESET, q->id, q->t + t);
        return NULL; }
```

**Use ChatGPT for every search**

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
if (sem_timedwait(&use->conc, &timeout) == -1) {
    printf(RED "User %d canceled the request due to no response at
%d seconds\n" RESET, q->id, q->t + t);
    pthread_mutex_lock(&use->users);
    use->us--;
    if (use->us == 0) {
        pthread_cond_signal(&use->user);
    }
    pthread_mutex_unlock(&use->users);
    return NULL;
}
```

**Use ChatGPT for every search**

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
if (sem_timedwait(&use->conc, &timeout) == -1) {
    printf(RED "User %d canceled the request due to no response at
%d seconds\n" RESET, q->id, q->t + currentsecs);
    sleep(r);
    printf(GREEN "The read request for User %d was completed at %
seconds\n" RESET, q->id, q->t + currentsecs + r);
    pthread_mutex_lock(&use->users);
    use->us--;
    if (use->us == 0) {
        pthread_cond_signal(&use->user);
    }
    pthread_mutex_unlock(&use->users);
    sem_post(&use->conc);
    return NULL; }
```

**Use ChatGPT for every search**

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
use = q->use;
struct timespec entry;
clock_gettime(CLOCK_REALTIME, &entry);
struct timespec timeout;
timeout.tv_sec = entry.tv_sec + t;
timeout.tv_nsec = 0;

if (pthread_mutex_timedlock(&use->users, &timeout) == -1) {
    printf(RED "User %d canceled the request due to no response at
%d seconds\n" RESET, q->id, q->t + t);
    return NULL;
}
use->us++; }
```

**Use ChatGPT for every search**

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
    printf(RED "User %d canceled the request due to no response at\n"
           "%d seconds\n" RESET, q->id, q->t + t);
    pthread_mutex_lock(&use->users);
    use->us--;
    if (use->us == 0) {
        pthread_cond_signal(&use->user);
    }
    pthread_mutex_unlock(&use->users);
    return NULL;
}

if (sem_timedwait(&use->conc, &timeout) == -1) {
    printf(RED "User %d canceled the request due to no response at\n"
           "%d seconds\n" RESET, q->id, q->t + t);
}
```

**Use ChatGPT for every search**

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
    pthread_cond_signal(&use->user);
}
pthread_mutex_unlock(&use->users);
sem_post(&use->write);
return NULL;
}

struct timespec now;
clock_gettime(CLOCK_REALTIME, &now);
int currentsecs = now.tv_sec - entry.tv_sec;
printf(PINK "LAZY has taken up the write request of User %d at %d
seconds\n" RESET, q->id, q->t + currentsecs);
sleep(w);
```

**Use ChatGPT for every search**

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
use->us--;
if (use->us == 0) {
    pthread_cond_signal(&use->user);
}
pthread_mutex_unlock(&use->users);
sem_post(&use->conc);
sem_post(&use->write);
return NULL;
}

void* delete1(void* args) {
    query* q = (query*)args;
    file* use = q->f;
```

**Use ChatGPT for every search**

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
clock_gettime(CLOCK_REALTIME, &timeout);
timeout.tv_sec += t;
sleep(1);

while (use->us > 0) {
    int ret = pthread_cond_timedwait(&use->user, &use->users,
&timeout);
    if (ret == ETIMEDOUT) {
        printf("User %d cancelled their request at %d seconds\n", q-
>id, q->t + t);
        pthread_mutex_unlock(&use->users);
        return NULL;
    }
}
```

**Use ChatGPT for every search**

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
int currentsecs = now.tv_sec - timeout.tv_sec + t;
printf(PINK "LAZY has taken up delete request of User %d at %d
seconds\n" RESET, q->id, q->t + t);
```

```
seconds\n" RESET, q->id, q->t + currentsecs);  
  
    sleep(d);  
  
    printf(GREEN "The delete request for User %d was completed at %d  
seconds\n" RESET, q->id, q->t + currentsecs + d);  
  
    return NULL;  
}
```



#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
scanf("%d %d %d", &r, &w, &d);  
scanf("%d %d %d", &n, &c, &t);  
  
query query_stack[limit];  
int query_count = 0;  
char res[7];  
  
while (1) {  
    scanf("%s", res);  
    if (strcmp(res, "STOP") == 0) break;  
  
    query_stack[query_count].id = atoi(res);  
    scanf("%d" &query_stack[query_count].storefile);  
    ↓
```

#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
else if (res == "D", query_stack[query_count].op = 0,  
else query_stack[query_count].op = -1;  
scanf("%d", &query_stack[query_count].t);  
query_count++;  
}  
  
query queries[query_count];  
file f[n + 1];  
  
for (int i = 0; i <= n; i++) {  
    sem_init(&f[i].del, 0, 1);  
    pthread_mutex_init(&f[i].users, NULL);  
    pthread_cond_init(&f[i].user, NULL);  
    ↓
```

#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
for (int i = 0; i < query_count; i++) {  
    queries[i] = query_stack[i];  
    queries[i].file = &f[queries[i].storefile];  
}  
  
qsort(queries, query_count, sizeof(query), cmp);  
  
pthread_t threads[query_count];  
int last_executed_time = 0;  
  
for (int i = 0; i < query_count; i++) {  
    int sleep_dur. ↓ n = queries[i].t - last_executed_time;
```

#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
if (sem_trywait(&queries[i].file->del) != 0) {  
    printf("User %d has their request declined at %d seconds as  
the requested file was deleted\n", queries[i].id, queries[i].t);  
    last_executed_time = queries[i].t;  
    continue;
```

```
    sem_post(&queries[i].file->del);

    if (queries[i].op == 1 && queries[i].storefile<=n) printf("User %d
has made a request for performing READ operation on file %d at %d
seconds\n", queries[i].id, queries[i].storefile, queries[i].t);
    else if (queries[i].op == 2 && queries[i].storefile<=n) printf("User
%d has made a re↓st for performing WRITE operation on file %d at
```

Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)



```
if (queries[i].op == 1) pthread_create(&threads[i], NULL, read1,
&queries[i]);
else if (queries[i].op == 2) pthread_create(&threads[i], NULL,
write1, &queries[i]);
else if (queries[i].op == 3) pthread_create(&threads[i], NULL,
delete1, &queries[i]);
else if(queries[i].op<0) printf("ERROR: User requested an
unsupported operation\n");
else if(queries[i].storefile>n) printf("ERROR:User requested an
invalid file\n");
last_executed = queries[i].t;
}
```

Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)



```
for (int i = 0; i <= n; i++) {
    sem_destroy(&f[i].del);
    pthread_mutex_destroy(&ff[i].users);
    pthread_cond_destroy(&ff[i].user);
    sem_destroy(&f[i].conc);
    sem_destroy(&f[i].write);
}

printf("\nLAZY is going back to sleep!\n");
return 0;
}
```

Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)



or User u\_i at t seconds [PINK]  
If LAZY declines a request due to an invalid file:LAZY has declined the  
request of User u\_i at t seconds because an invalid/deleted file was  
requested. [WHITE]  
When a request completes:The request for User u\_i was completed at  
t\_j seconds [GREEN]  
If the user cancels their request:User u\_i canceled the request due to  
no response at t\_k seconds [RED]  
Modify above function as per these rules



Here is the modified code with the requested print statements and colors added:



Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)



```
    use = q->use;
    struct timespec entry;
    clock_gettime(CLOCK_REALTIME, &entry);
    struct timespec timeout;
    timeout.tv_sec = entry.tv_sec + t;
    timeout.tv_nsec = 0;

    if (pthread_mutex_timedlock(&use->users, &timeout) == -1) {
        printf(RED "User %d canceled the request due to no response at %d seconds\n" RESET
```

```
        return NULL;
    }
    use->us++;
    pthread_mutex_unlock(&use->users); ↓
```

#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
pthread_mutex_lock(&use->users);
use->us--;
if (use->us == 0) {
    pthread_cond_signal(&use->user);
}
pthread_mutex_unlock(&use->users);
return NULL;
}

struct timespec now;
clock_gettime(CLOCK_REALTIME, &now);
int currentsecs = now.tv_sec - entry.tv_sec;
printf(PINK "LAZY has taken up the r ↓ request of User %d at %d seconds\n" RESET, q->
```

#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
if (use->us == 0) {
    pthread_cond_signal(&use->user);
}
pthread_mutex_unlock(&use->users);
sem_post(&use->conc);
return NULL;
}

void* write1(void* args) {
    query* q = (query*)args;
    file* use = q->file;
    struct timespec entry;
    clock_gettime(CLOCK_REALTIME, &entry); ↓
```

#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
if (pthread_mutex_timedlock(&use->users, &timeout) == -1) {
    printf(RED "
```

don't replace colours with a macro actually print correct symbols for each statement



Here's the modified code with color outputs as symbols (no macros):

[c](#)[d](#)[Copy code](#)

#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
struct timespec entry;
clock_gettime(CLOCK_REALTIME, &entry);
struct timespec timeout;
timeout.tv_sec = entry.tv_sec + t;
timeout.tv_nsec = 0;

if (pthread_mutex_timedlock(&use->users, &timeout) == -1) {
    printf("\033[31mUser %d canceled the request due to no response at %d seconds\033[1m");
    return NULL;
}
use->us++;
pthread_mutex_unlock(&use->users); ↓
```

```
    sleep(1);
```

#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
use->us--;
if (use->us == 0) {
    pthread_cond_signal(&use->user);
}
pthread_mutex_unlock(&use->users);
return NULL;
}

struct timespec now;
clock_gettime(CLOCK_REALTIME, &now);
int currentsecs = now.tv_sec - entry.tv_sec;
printf("\033[35mLAZY has taken up the read request of User %d at %d seconds\033[0m\n",
sleep(r):
```

#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
use->us --;
pthread_cond_signal(&use->user);
}
pthread_mutex_unlock(&use->users);
sem_post(&use->conc);
return NULL;
}

void* write1(void* args) {
query* q = (query*)args;
file* use = q->file;
struct timespec entry;
clock_gettime(CLOCK_REALTIME, &entry)
```

#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
if (pthread_mutex_timedlock(&use->users, &timeout) == -1) {
    printf("\033[31mUser %d canceled the request due to no response at %d seconds\033[0m\n");
    return NULL;
}
use->us++;
pthread_mutex_unlock(&use->users);
sleep(1);

if (sem_timedwait(&use->write, &timeout) == -1) {
    printf("\033[31mUser %d canceled the request due to no response at %d seconds\033[0m\n");
    pthread_mutex_lock(&use->users);
    use->us--;
    if (use->us == 0) {
```

#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
}
```

```
if (sem_timedwait(&use->conc, &timeout) == -1) {
    printf("\033[31mUser %d canceled the request due to no response at %d seconds\033[0m\n");
    pthread_mutex_lock(&use->users);
    use->us--;
    if (use->us == 0) {
        pthread_cond_signal(&use->user);
    }
    pthread_mutex_unlock(&use->users);
    sem_post(&use->write);
    return NULL;
}
```

#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
printf("\033[35mLAZY has taken up the write request of User %d at %d seconds\033[0m\n";
sleep(w);
printf("\033[32mThe write request for User %d was completed at %d seconds\033[0m\n", q
pthread_mutex_lock(&use->users);
use->us--;
if (use->us == 0) {
    pthread_cond_signal(&use->user);
}
pthread_mutex_unlock(&use->users);
sem_post(&use->conc);
sem_post(&use->write);
return NULL;
}
```



**Use ChatGPT for every search**

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
pthread_mutex_lock(&use->users);

struct timespec timeout;
clock_gettime(CLOCK_REALTIME, &timeout);
timeout.tv_sec += t;
sleep(1);

while (use->us > 0) {
    int ret = pthread_cond_timedwait(&use->user, &use->users, &timeout);
    if (ret == ETIMEDOUT) {
        printf("\033[31mUser %d canceled their request at %d seconds\033[0m\n", q->id,
        ...
```



**Use ChatGPT for every search**

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
sem_wait(&use->del);
struct timespec now;
clock_gettime(CLOCK_REALTIME, &now);
int currentsecs = now.tv_sec - timeout.tv_sec + t;
printf("\033[35mLAZY has taken up delete request of User %d at %d seconds\033[0m\n",
sleep(d);

printf("\033[32mThe delete request for User %d was completed at %d seconds\033[0m\n",
...
```



**Use ChatGPT for every search**

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
scanf("%d %d %d", &n, &c, &t);

query query_stack[limit];
int query_count = 0;
char res[7];

while (1) {
    scanf("%s", res);
    if (strcmp(res, "STOP") == 0) break;

    query_stack[query_count].id = atoi(res);
    scanf("%d", &query_stack[query_count].storefile);
    scanf("%s", res);
```



**Use ChatGPT for every search**

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)

```
scanf("%d", &query_stack[query_count].t);
```

```
        query_count++;
    }

    query queries[query_count];
    file f[n + 1];

    for (int i = 0; i <= n; i++) {
        sem_init(&f[i].del, 0, 1);
        pthread_mutex_init(&f[i].users, NULL);
        pthead_cond_init(&f[i].user, NULL);
        sem_init(&f[i].conc, 0, c);           ↓
        sem_init(&f[i].write, 0, 1);
```

Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)



```
for (int i = 0; i < query_count; i++) {
    queries[i] = query_stack[i];
    queries[i].file = &f[queries[i].storefile];
}

qsort(queries, query_count, sizeof(query), cmp);

pthread_t threads[query_count];
int last_executed_time = 0;

for (int i = 0; i < query_count; i++) {
    int sleep_duration = queries[i].t - last_executed_time;
    if (sleep_duration > 0) {           ↓
        sleep(sleep_duration);
    }
}
```

Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)



```
printf("\033[37mUser %d has their request declined at %d seconds as the request\n"
      "last_executed_time = queries[i].t;\n"
      "continue;\n"
}
sem_post(&queries[i].file->del);

if (queries[i].op == 1 && queries[i].storefile <= n) printf("\033[33mUser %d has made a request to storefile %d\n"
else if (queries[i].op == 2 && queries[i].storefile <= n) printf("\033[33mUser %d has made a request to write to storefile %d\n"
else if (queries[i].op == 3 && queries[i].storefile <= n) printf("\033[33mUser %d has made a request to delete from storefile %d\n"

if (queries[i].op == 1) pthread_create(&threads[i], NULL, read1, &queries[i]);
else if (queries[i].op == 2) pthread_create(&threads[i], NULL, write1, &queries[i]);
else if (queries[i].op == 3) pthread_create(&threads[i], NULL, delete1, &queries[i]);
```

Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)



```
}
```

  

```
for (int i = 0; i < query_count; i++) {
    pthread_join(threads[i], NULL);
}

for (int i = 0; i <= n; i++) {
    sem_destroy(&f[i].del);
    pthead_mutex_destroy(&f[i].users);
    pthead_cond_destroy(&f[i].user);
    sem_destroy(&f[i].conc);
    sem_destroy(&f[i].write);
}
```

Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)



- RED (Cancelled): \033[31m
- YELLOW (Request Initiated): \033[33m
- PINK (Processing): \033[35m
- GREEN (Completed): \033[32m
- WHITE (Declined): \033[37m

This code uses ANSI color codes directly for specific conditions in each log message.



#### Use ChatGPT for every search

Download the Chrome extension to switch your default search engine to ChatGPT, and get instant answers from trusted sources with every search.

[Get Extension](#)



Condition Variable Wait Timeout

Add Team workspace  
Collaborate on a Team plan

Message ChatGPT



ChatGPT can make mistakes. Check important info.

?