|  |
| --- |
| #include<stdio.h> |
|  | #include<stdlib.h>//malloc |
|  | #include<sys/stat.h> |
|  | #include<assert.h> |
|  | #include<sys/mman.h>//mmap |
|  | #include<semaphore.h> |
|  | #include<pthread.h> |
|  | #include<sys/sysinfo.h>//get\_nprocs |
|  | #include<string.h> |
|  | #include<unistd.h> |
|  | #include<sys/stat.h> |
|  | #include<fcntl.h> |
|  | #include<sys/time.h> |
|  | #include<stdatomic.h> |
|  | #include"definitions.h" |
|  | #include"producer\_consumer.c" |
|  | #include"zipping.c" |
|  |  |
|  | struct arguments \* init\_arg() |
|  | { |
|  | struct arguments \*temp=malloc(sizeof(struct arguments)); |
|  | temp->argc=0; |
|  | temp->argv=NULL; |
|  | return temp; |
|  | } |
|  |  |
|  | struct node\* init() |
|  | { |
|  | struct node\* ptr=malloc(sizeof(struct node)); |
|  | ptr->identity='a'; |
|  | ptr->flag=0; |
|  | ptr->num=0; |
|  | ptr->left=NULL; |
|  | ptr->right=NULL; |
|  | return ptr; |
|  | } |
|  |  |
|  | struct bufferobj \* init\_buff() |
|  | { |
|  | struct bufferobj \*temp= malloc(sizeof(struct bufferobj)); |
|  | temp->filenum=0; |
|  | temp->pagenum=0; |
|  | temp->pageinram=NULL; |
|  | temp->flag=0; |
|  | temp->lastpagesize=0; |
|  | return temp; |
|  | } |
|  |  |
|  | struct compobj\* init\_comp() |
|  | { |
|  | struct compobj \*temp=malloc(sizeof(struct compobj)); |
|  | temp->data=NULL; |
|  | temp->size=0; |
|  | return temp; |
|  | } |
|  |  |
|  | int main(int argc, char\* argv[]) |
|  | { |
|  | struct timeval start\_time, end\_time; |
|  | filenames=argv; |
|  | // printf("%s",filenames[1]); |
|  | //initialize semaphores |
|  | sem\_init(&empty,0,20);//initialized to 20 so that when we run the producer first, it does not sleep |
|  | sem\_init(&full,0,0);//initialized to 0 so that even if a consumer thread runs before the producer, it will be put to sleep |
|  | sem\_init(&mutex,0,1);//only one thread should be able to acquire the lock at a time, hence initialized to a 1 |
|  |  |
|  | //create a thread for producer |
|  | pthread\_t p; |
|  | struct arguments \*ptr=init\_arg(); |
|  | ptr->argc=argc; |
|  | ptr->argv=argv; |
|  | gettimeofday(&start\_time, NULL); |
|  | pthread\_create(&p,NULL,producer,(void \*)ptr); |
|  |  |
|  | //create threads for consumers |
|  | int n\_threads=get\_nprocs(); |
|  | pthread\_t c[n\_threads]; |
|  | for(int i=0;i<n\_threads;i++) |
|  | { |
|  | pthread\_create(&c[i],NULL,consumer,NULL); |
|  | } |
|  |  |
|  | //join all the threads |
|  | //consumer |
|  | for(int i=0;i<n\_threads;i++) |
|  | { |
|  | pthread\_join(c[i],NULL); |
|  | } |
|  | //producer |
|  | pthread\_join(p,NULL); |
|  | gettimeofday(&end\_time, NULL); |
|  | long diff=(end\_time.tv\_sec-start\_time.tv\_sec)\*1000000 + (end\_time.tv\_usec-start\_time.tv\_usec); |
|  | printf("time: %f seconds\n",diff\*0.000001); |
|  |  |
|  | create\_compressed\_files(compressed,argc); |
|  |  |
|  | //free the mallocs |
|  | free(pagecnt); |
|  | for(int i=0;i<argc-2;i++) |
|  | { |
|  | free(compressed[i]); |
|  | } |
|  |  |
|  | return 0; |
|  | } |