Przemysław Szymoniak

Przetwarzanie współbieżne. Programowanie równoległe i rozproszone.

Sprawozdanie z laboratorium 4.
Celem laboratorium było nabycie praktycznej umiejętności pisania programów z synchronizacją wątków.

W ramach zajęć zrealizowałem następujące kroki:

- Pobranie plików „**pub_sym_1.c**” oraz **„pub_sym_2.c”** i skopiowanie ich do wcześniej utworzonego folderu
- Zaprojektowanie symulacji pubu w sytuacji gdy liczba kufli przewyższa liczbę klientów wraz z jego implementacją w pliku **„pub_sym_1.c”:**

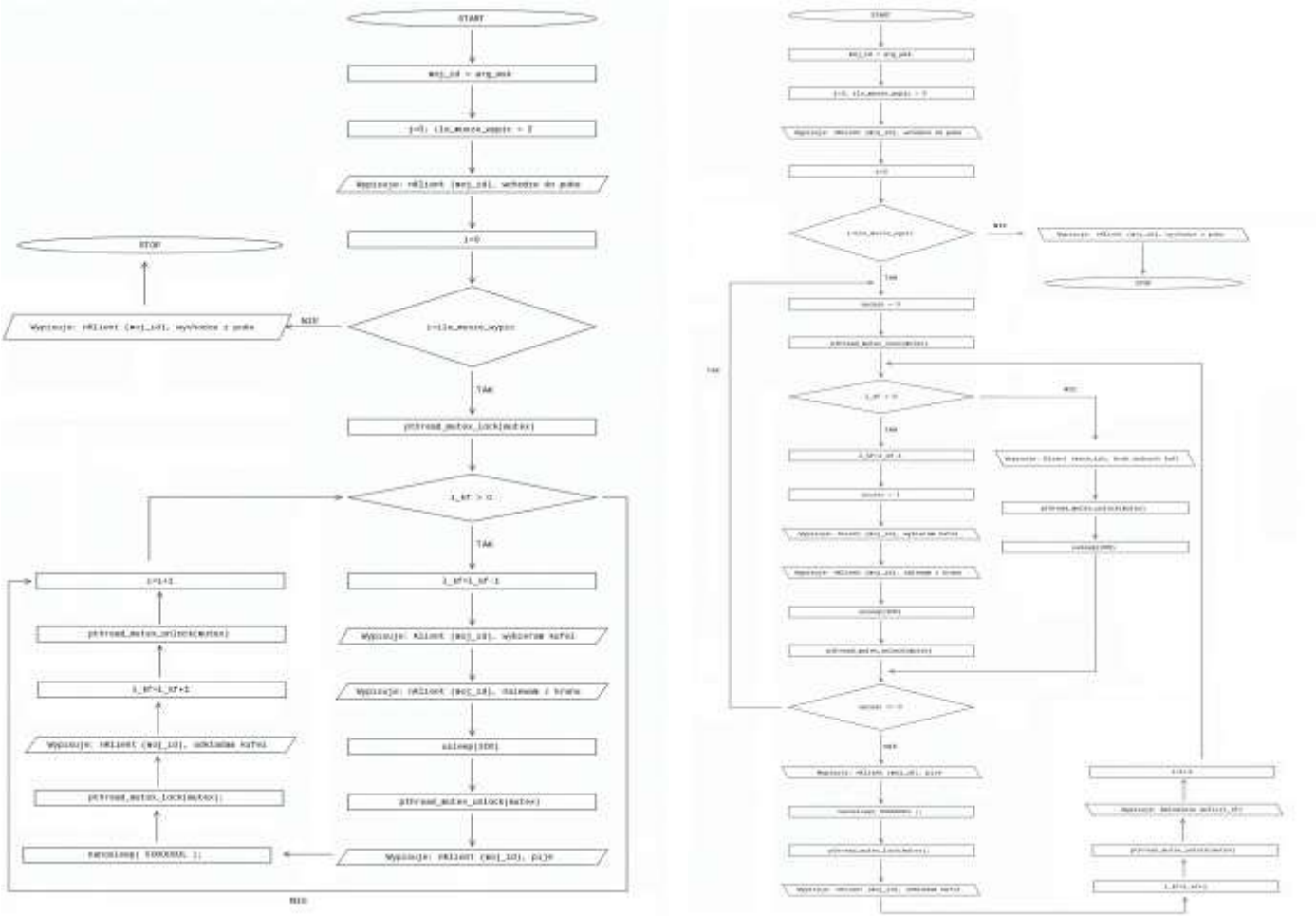
```
void * watek_klient (void * arg);
int l_kf;
pthread_mutex_t
        mutex=PTHREAD_MUTEX_INITIALIZER;

main(){
    pthread_t *tab_klient;
    int *tab_klient_id;
    int l_kl, l_kr, i;
     l_kr = 1;

    printf("\nLiczba klientow: "); scanf("%d",
  &l_kl);
    printf("\nLiczba kufli: "); scanf("%d", &l_kf);

    tab_klient= (pthread_t *)
  malloc(l_kl*sizeof(pthread_t));
    tab_klient_id = (int *) malloc(l_kl*sizeof(int));

    for(i=0;i<l_kl;i++)
        tab_klient_id[i]=i;

    printf("\nOtwieramy pub (simple)!\n");
    printf("\nLiczba wolnych kufli %d\n", l_kf);

    for(i=0;i<l_kl;i++){
        pthread_create(&tab_klient[i], NULL,
            watek_klient, &tab_klient_id[i]);
    }

    for(i=0;i<l_kl;i++){
        pthread_join( tab_klient[i], NULL);
    }
    printf("\nZamykamy pub!\n");
}
```

```
void * watek_klient (void * arg_wsk){
    int moj_id = * ((int *)arg_wsk);
    int i, j=0;
    int ile_musze_wypic = 2;
    printf("\nKlient %d, wchodzę do pubu\n", moj_id);

    for(i=0; i<ile_musze_wypic; i++){
        pthread_mutex_lock(&mutex);

        if(l_kf > 0){
            l_kf--;
            printf("\nKlient %d, wybieram kufel %d\n",
                moj_id), j;
            printf("\nKlient %d, nalewam z kranu
                %d\n", moj_id, j);

            usleep(300);

            pthread_mutex_unlock(&mutex);

            printf("\nKlient %d, pije\n", moj_id);
                nanosleep((struct
                timespec[]){{0, 5000000L}}, NULL);

            pthread_mutex_lock(&mutex);

            printf("\nKlient %d, odkladam kufel\n",
                moj_id);
            l_kf++;
            pthread_mutex_unlock(&mutex);
        }
    }
    printf("\nKlient %d, wychodzę z pubu\n", moj_id);
    return(NULL);
}
```

- Na podstawie pliku **„pub_sym_1.c”** wykonanie programu z uwzględnieniem sytuacji gdy liczba kufli jest większa od liczby klientów, dokonanie zmiany (w funkcji wątku) to:
    - Wykorzystanie aktywnego czekania
    - Obsługa braku obecności kufla
    - W wypadku braku kufla wymuszenie odczekania chwili
    - Podział na część kodu odpowiedzialna za odbieranie kufla oraz za picie i odłożenie go
    - Dodanie komunikatów odnośnie liczby dostępnych kufli
    - **ZMODYFIKOWANA CZĘŚĆ PROGRAMU:**

```
do{
    pthread_mutex_lock(&mutex);
    if(l_kf > 0){
            sucess = 1;
            l_kf--;
            printf("\nKlient %d, wybieram kufel %d\n", moj_id, j);
            printf("\nKlient %d, nalewam z kranu %d\n", moj_id, j);
            usleep(300);
            pthread_mutex_unlock(&mutex);
    }else{
            printf("Klient %d, brak wolnych kufli\n", moj_id);
            pthread_mutex_unlock(&mutex);
            usleep(200);
    }
}while(!sucess);
printf("\nKlient %d, pije\n", moj_id);
nanosleep((struct timespec[]){{0, 5000000L}}, NULL);
pthread_mutex_lock(&mutex);
printf("\nKlient %d, odkladam kufel\n", moj_id);
l_kf++;
pthread_mutex_unlock(&mutex);
printf("\nAktualnie kufli:%d\n", l_kf);
```

- Schemat blokowy programu (W większej skali dołączone jako załącznik):



Wnioski:

- Wykorzystanie mutexów pozwoliło znacząco zwiększyć bezpieczeństwo pracy aplikacji, poprzez blokowanie innych wątków chcących dostać się równocześnie do współdzielonych danych
- Użycie algorytmu z aktywnym czekaniem pozwoliło zmniejszyć czas blokowania wątków pozwalając im wykonywać w między czasie inne operacje nie wymagające dostępu do danych współdzielonych
- Zaimplementowane rozwiązanie w działało poprawnie przy każdym testowanym wariancie
- Większa liczba kranów wymagałaby od programu użycia tablicy mutexów oraz kranów
- Blokada mutexem została wykorzystana osobno przy odbieraniu i oddawaniu kufla aby inne wątki mogły wykorzystać ten czas do pracy równoległej