Լաբորատոր աշխատանք 3

Սեմաֆորներ
Մրցակցային իրավիճակների հանգուցալուծում

SETALL գործողությունը

semctl() ֆունկցիայի **SETALL** հրամանը սկզբնավորում է semid id-ով սահմանված սեմաֆորների հավաքածուի բոլոր սեմաֆորները, օգտվելով arg.array զանգվածում եղած արժեքներից: semnum արգումենտն անտեսվում է:

int semctl(int semid, int semnum, int cmd, ... /* union semun arg */);

Հրամանի կիրառման օրինակը ներկայաված է sem_setall.c ծրագրում, որը որպես հրամանային տողի արգումենտներ ընդունում է սեմաֆորների հավաքածուի id-ն, որը պետք է սկզբնավորել, և սեմաֆորների քանակին հավասար ամբողջ թվեր՝ դրանք սկզբնավորելու համար:

Նախքան այն կրառելը, անհրաժեշտ է ստեղծել նոր սեմաֆորների հավաքածու, որը կպարունակի 1-ից ավելի սեմաֆորներ: Այն իրականացվում է sem_create.c ծրագրի միջոցով, որը որպես հրամանային տողի արգումենտ ընդունում է նոր ստեղծվող սեմաֆորների հավաքածուի սեմաֆորների քանակը: Ծրագիրը կատարելու համար անհրաժեշտ քայլերն են.

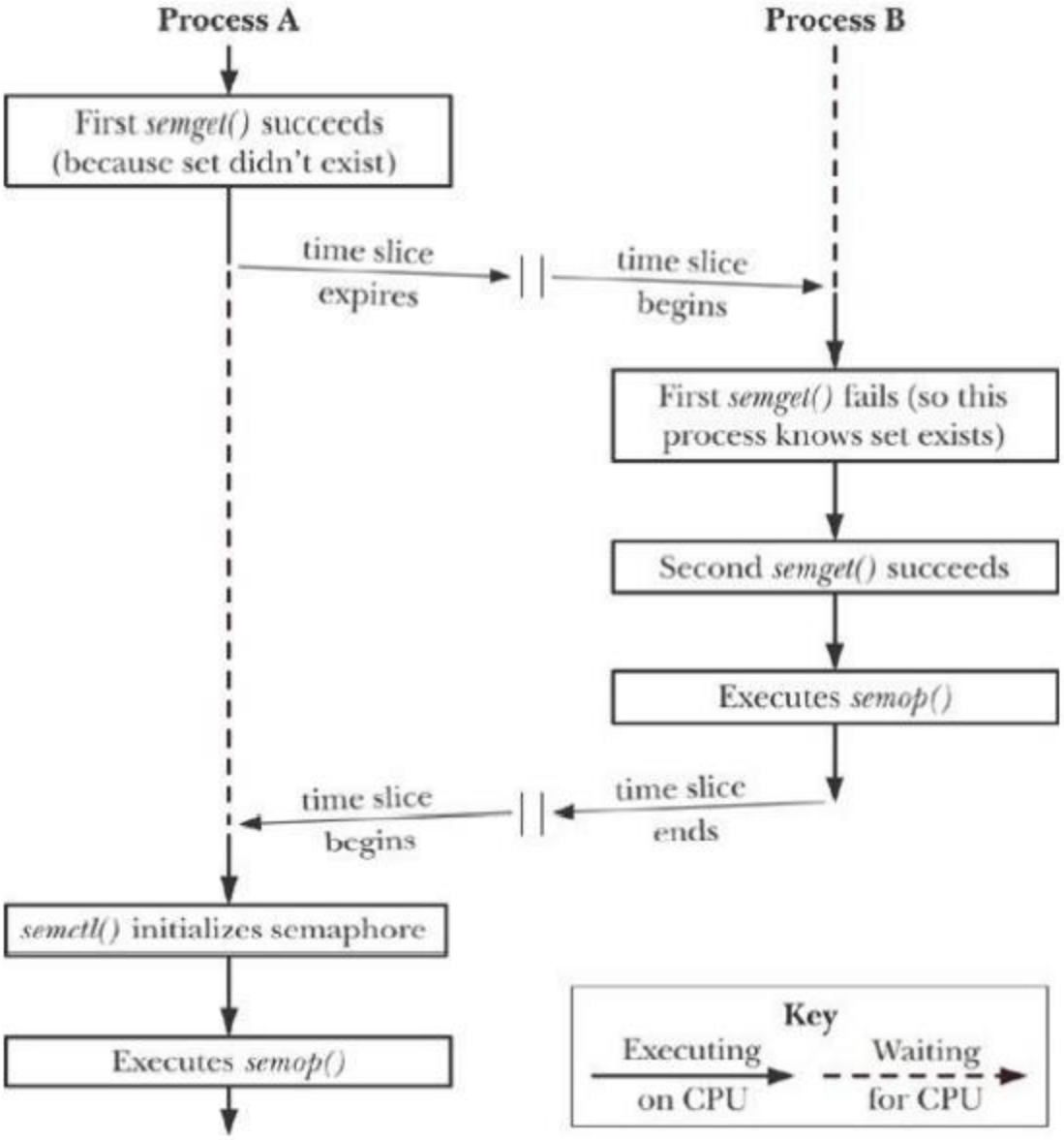
```
gcc sem_create.c -o create
./create 3
```

```
gcc sem_setall.c -o setall
./setall 0 2 1 3
```

Սեմաֆորի սկզբնավորումը 1-ից ավելի պրոցեսների դեպքում

Երբ սեմաֆորի հետ աշխատում են 1-ից ավելի պրոցեսներ, կարող է առաջանալ Նկ. 1-ում պատկերված խնդիրը, երբ առաջին պրոցեսի ընդհատումը տեղի ունենա այն պահին, երբ այն դեռ չի սկզբնավորել սեմաֆորը: Այն խնդրահարույց է 2 պատճառով.

- B պրոցեսը իրականացնում է semop() կանչը չսկզբնավորված սեմաֆորի վրա,
- A պրոցեսի semctl() կանչը վերագրում է B պրոցեսի կողմից կատարված գործողությունները:



Նկ. 1 Սեմաֆորի ստեղծումը և սկզբնավորումը

Խնդրի լուծումը հիմնվում է **semid_ds** ստրուկտուրայի **sem_otime** դաշտի վրա: Երբ սեմաֆորների հավաքածուն ստեղծվում է, **sem_otime** դաշտը սկզբնավորվում է 0 արժեքով և փոխում է արժեքը միայն **semop()** կանչի ժամանակ:

Մրցակցային իրավիճակը հանգուցալուծելու նպատակով, կարող ենք հավաքածուն ստեղծելիս ավելացնել semop() կանչ, որը չի փոխի սեմաֆորի արժեքը, բայց ավտոմատ կերպով կթարմացնի sem_otime դաշտի արժեքը:

semid_ds ստրուկտուրան.

```
struct semid_ds {
        struct ipc_perm sem_perm; /* Ownership and permissions */
        time_t sem_otime; /* Time of last semop() */
        time_t sem_ctime; /* Time of last change */
        unsigned long sem_nsems; /* Number of semaphores in set */
};
```

Խնդրի լուծումը ներկայացված է sem_good_init.c ծրագրում: Այն կատարելու համար անհրաժեշտ քայլերն են.

```
gcc sem_good_init.c -o init
./init && ./init
```

Սեմաֆորի չեղարկման արժեքներ (SEM_UNDO)

Դիցուք, սեմաֆորի արժեքը փոփոխելուց հետո (օրինակ՝ այն դարձնելով 0) պրոցեսն ավարտվում է: Լռելյայն կերպով սեմաֆորի արժեքը մնում է անփոփոխ: Սա կարող է խնդիրներ առաջացնել սեմաֆորն օգտագործող մյուս պրոցեսների համար, որոնք սպասում են սեմաֆորի արժեքի փոփոխությանը (0-ից մեծ արժեք ընդունելուն):

Այսպիսի խնդիրներից խուսափելու նպատակով սեմաֆորի արժեքը semop() կանչով փոխելիս կիրառվում է **SEM_UNDO** դրոշակը: Կիրառումը ներկայացված է sem_op_undo.c ծրագրում, որը որպես հրամանային տողի արգումենտ ընդունում է սեմաֆորնորի հավաքածուի id-ն: Այն կատարելու համար անհրաժեշտ է իրականացնել հետևյալ քայլերը.

```
gcc sem_op_undo.c -o undo
./undo 0
```

Առաջադրանքներ

1. Ստեղծել նոր սեմաֆորների հավաքածու, որը պարունակում է 4 սեմաֆոր:
2. Սկզբնավորել դրանք հետևյալ արժեքներով. 3, 2, 1, 0
3. Կատարել sem_good_init ծրագիրը 2 պրոցեսներով՝ զուգահեռաբար: Բացատրել 2 պրոցեսների կողմից ցուցադրված հաղորդագրությունները:
4. Կատարել sem_op_undo ծրագիրը: Ցուցադրել սեմաֆորների հավաքածուի մասին տեղեկատվություն, և բացատրել SEM_UNDO դրոշակի ազդեցությունը: Կատարել նույն գործողությունները՝ SEM_UNDO դրոշակը հեռացնելուց հետո: