Outils IPC

Tableaux de sémaphores - `semctl`

Cet article présente différents cas d'utilisation de la fonction `semctl` qui permet de modifier ou récupérer des informations sur les tableaux de sémaphores IPC System V.

1 Le programme création

Ce programme permet de créer un tableau de sémaphores dont la clé et le nombre de sémaphores sont passés en arguments. Si le tableau existe déjà, une erreur est affichée.

```
if((semid = semget((key_t)cle, nbSem, S_IRUSR | S_IWUSR | IPC_CREAT |
    IPC_EXCL)) == -1) {
  if(errno == EEXIST)
    fprintf(stderr, "Le tableau de sémaphores (cle=%d) existe déjà.\n",
       cle);
  else
    perror("Erreur lors de la création du tableau de sémaphores ");
  exit(EXIT_FAILURE);
}
```

2 Le programme suppression

Ce programme permet de supprimer un tableau de sémaphores dans la clé IPC est passée en argument. Dans un premier temps, il faut récupérer l'identifiant interne.

```
if((semid = semget((key_t)cle, 0, 0)) == -1) {
  perror("Erreur lors de la récupération du tableau de sémaphores ");
  exit(EXIT_FAILURE);
}
```

Le tableau peut ensuite être supprimé à l'aide de `semctl`.

```
if(semctl(semid, 0, IPC_RMID) == -1) {
  perror("Erreur lors de la suppresion du tableau de sémaphores ");
  exit(EXIT_FAILURE);
}
```

3 Le programme permettant de récupérer une valeur

Ce programme permet de récupérer soit toutes les valeurs des sémaphores d'un tableau de sémaphores, soit de récupérer une valeur d'un des sémaphores d'un tableau de sémaphores. La clé et le

numéro du sémaphore sont passés en arguments. Si le numéro n'est pas spécifié, les valeurs de tous les sémaphores sont affichées.

Dans un premier temps, l'identifiant interne est récupéré.

```
int semid;
if((semid = semget((key_t)cle, 0, 0)) == -1) {
  perror("Erreur lors de la récuperation du tableau de sémaphores ");
  exit(EXIT_FAILURE);
}
```

Grâce à `semctl`, on récupère les informations sur le tableau de sémaphores.

```
struct semid_ds sem_buf;
if(semctl(semid, 0, IPC_STAT, &sem_buf) == -1) {
  perror("Erreur lors de la récupération d'informations sur le tableau
     de sémaphores ");
  exit(EXIT_FAILURE);
}
```

Si aucun numéro de sémaphore n'est spécifié, nous récupérons les valeurs de tous les sémaphores. Pour cela, nous avons besoin d'un tableau dont la taille est récupérée dans la structure `sem_buf`.

```
unsigned short *tableau;

if((tableau = (unsigned short *)malloc(sizeof(unsigned short) * sem_buf.
    sem_nsems)) == NULL) {
  perror("Erreur lors de l'allocation mémoire ");
  exit(EXIT_FAILURE);
}

if(semctl(semid, 0, GETALL, tableau) == -1) {
  perror("Erreur lors de la récupération des valeurs du tableau de
     sémaphores ");
  exit(EXIT_FAILURE);
}
```

Si un numéro est spécifié, nous vérifions qu'il est valide.

```
if((num < 0) || (num >= sem_buf.sem_nsems)) {
  fprintf(stderr, "Numéro de sémaphore incorrect (%d), doit être dans
     [0; %ld]\n", num, sem_buf.sem_nsems - 1);
  exit(EXIT_FAILURE);
}
```

Puis nous récupérons la valeur du sémaphore.

```
if((valeur = semctl(semid, num, GETVAL)) == -1) {
  perror("Erreur lors de la récupération de la valeur du sémaphore ");
  exit(EXIT_FAILURE);
}
```

4 Le programme permettant de modifier une valeur

Ce programme permet de modifier soit une valeur d'un sémaphore, soit les valeurs de tous les sémaphores d'un tableau de sémaphores. La clé du tableau est passée en argument. Pour modifier un sémaphore, on ajoute le mot-clé `SET`, l'indice du sémaphore dans le tableau puis sa valeur. Pour modifier toutes les valeurs, on ajoute le mot-clé `ALL` puis toutes les valeurs.

Nous commençons par récupérer l'identifiant interne.

```
if((semid = semget((key_t)cle, 0, 0)) == -1) {
  perror("Erreur lors de la récupération du tableau de sémaphores ");
  exit(EXIT_FAILURE);
}
```

Puis les informations grâce à `semctl`.

```
if(semctl(semid, 0, IPC_STAT, &sem_buf) == -1) {
  perror("Erreur lors de la récupération d'info sur le tableau de
     sémaphores ");
  exit(EXIT_FAILURE);
}
```

Si l'option `SET` est spécifiée, nous vérifions que le numéro de sémaphore est correct.

```
int num = atoi(argv[3]);
if((num < 0) || (num >= sem_buf.sem_nsems)) {
  fprintf(stderr, "Numéro de sémaphore (%d) incorrect; doit être dans
     [0, %ld]\n", num, sem_buf.sem_nsems - 1);
  exit(EXIT_FAILURE);
}
```

Puis nous modifions la valeur.

```
int valeur = atoi(argv[4]);
if(semctl(semid, num, SETVAL, valeur) == -1) {
  perror("Erreur lors de la modification de la valeur ");
  exit(EXIT_FAILURE);
}
```

Si l'option `ALL` est spécifiée, nous allouons dans un premier temps le tableau :

```
unsigned short *tableau;
if((tableau = (unsigned short *)malloc(sizeof(unsigned short) * sem_buf.
    sem_nsems)) == NULL) {
  perror("Erreur␣lors␣de␣l'allocation␣mémoire␣");
  exit(EXIT_FAILURE);
}
```

Après avoir vérifié que le nombre de valeurs est correct, nous les récupérons :

```
for(i = 0; i < sem_buf.sem_nsems; i++)
  tableau[i] = atoi(argv[i + 3]);
```

Puis nous modifions les valeurs à l'aide de `semctl` :

```
if(semctl(semid, 0, SETALL, tableau) == -1) {
  perror("Erreur␣lors␣de␣la␣modification␣des␣valeurs␣");
  exit(EXIT_FAILURE);
}
```

5 Compilation et exécution

Pour compiler les programmes précédents, saisissez la commande suivante :

```
make
```

Pour tester les programmes, tapez dans un premier temps, la commande suivante dans un terminal qui permet de créer un tableau de cinq sémaphores dont la clé IPC est 1056 (il est possible de changer le nombre de sémaphores et la valeur de la clé) :

```
./creation 1056 5
```

Pour tester les différents programmes, tapez les commandes suivantes :

```
./getValeur 1056
./setValeur 1056 SET 0 4
./getValeur 1056 0
./setValeur 1056 ALL 1 2 3 4 5
./getValeur 1056
```

Pour supprimer le tableau de sémaphores, tapez la commande suivante :

```
./suppression 1056
```