Operációs rendszerek BSc

11. gyakorlat

Készítette:
Nemesi Gergely Tibor
Üzemmérnök-informatikus
Neptun: ILZGJC

2022.04.25

Contents

Part I
Foglalási stratégiák, IPC (szemafor)

1. feladat

Adott egy rendszer (foglalási stratégiák), melyben a következő

- Szabad területek: 30k, 35k, 15k, 25k, 75k, 45k és
- Foglalási igények: 39k, 40k, 33k, 20k, 21k állnak rendelkezésre.

A rendszerben a memória 4 kbyte-os blokkokban kerül nyilvántartásra, ennél kisebb méretű töredék igény esetén a teljes blokk lefoglalásra kerül.
Határozza meg változó méretű partíció esetén a következő algoritmusok felhasználásával:
first fit, next fit, best fit, worst fit a foglalási igényeknek megfelelő helyfoglalást – táblázatos formában (az ea. bemutatott mintafeladat alapján)!

Hasonlítsa össze, hogy a teljes szabad memóriaterület hány százaléka vész el átlagosan az egyes algoritmusok esetén!
A kapott eredményeket ábrázolja oszlop diagrammal!
Magyarázza a kapott eredményeket és hogyan lehet az eredményeket javítani!

first fit

	Memória terület - szabad terület					
Foglalási igény	30	35	15	25	75	45
39					36 (75 - 39)	
40						5 (45 - 40)
33		2 (35 - 33)				
20				5 (25 - 20)		
21	9 (30 - 31)					

Szabad területek:	30k, 35k, 15k, 25k, 75k, 45k
Foglalási igények:	39k, 40k, 33k, 20k, 21k

first fit, next fit, best fit, worst fit

next fit

	Memória terület - szabad terület					
Foglalási igény	30	35	15	25	75	45
39					36 (75 - 39)	
40						5 (45 - 40)
33		2 (35 - 33)				
20				5 (25 - 20)		
21					15 (36 - 21)	

best fit

	Memória terület - szabad terület					
Foglalási igény	30	35	15	25	75	45
39						6 (45 - 39)
40					35 (75 - 40)	
33		2 (35 - 33)				
20				5 (25 - 20)		
21	9 (30 - 31)					

worst fit

	Memória terület - szabad terület					
Foglalási igény	30	35	15	25	75	45
39					36 (75 - 39)	
40						5 (45 - 40)
33					3 (36 - 33)	
20		15 (35 - 20)				
21	9 (30 - 31)					

2. feladat

A feladat megoldásához először tanulmányozza Vadász Dénes: Operációs rendszer jegyzet, a témához kapcsolódó fejezetét (6.4)., azaz Írjon C nyelvű programokat, ahol

- kreál/azonosít szemafor készletet, benne N szemafor-t. A kezdő értéket 0-ra állítja – semset.c,
- kérdezze le és írja ki a pillanatnyi szemafor értéket – semval.c
- szüntesse meg a példácskák szemafor készletét – semkill.c
- sembuf.sem-op=1 értékkel inkrementálja a szemafort – semup.c

A futtatás eredményét is tartalmazza a jegyzőkönyv.

```c
#include <stdio.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdlib.h>
#define KEY 123456L

union semun {
    int val;                /* Value for SETVAL */
    struct semid_ds *buf;   /* Buffer for IPC_STAT, IPC_SET */
    unsigned short *array;  /* Array for GETALL, SETALL */
    struct seminfo *__buf;  /* Buffer for IPC_INFO (Linux-specific) */
};

void main() {
    union semun arg;

    int n = 5;
    int semID = semget(KEY, n, IPC_CREAT | 0666);

    if (semID == -1)
    {
        perror("Nem sikerult szemaforokat letrehozni");
        exit(-1);
    }

    arg.array = (short *)calloc(n, sizeof(int));

    if (semctl(semID, 0, SETALL, arg))
    {
        perror("Nem sikerult beallitani az erteket\n");
        exit(-1);
    }

}
```

Semset

```
/home/robo/CLionProjects/test/cmake-build-debug/test
Szemaforok tartalma:
0
0
0
0
0

Process finished with exit code 5
```

Semval

```c
#include <stdio.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdlib.h>
#define KEY 123456L

void main() {
    int n = 5;
    int semID = semget(KEY, 0, 0);
    if (semID == -1) {
        perror("Nem sikerult szemaforokat lekerdezni\n");
        exit(-1);
    }

    for (int i = 0; i < n; i++)
        semctl(semID, i, IPC_RMID);

}
```

Semkill

```c
#include <stdio.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdlib.h>
#define KEY 123456L

void main() {
    int semID = semget(KEY, 0, 0);
    if (semID == -1) {
        perror("Nem sikerult szemaforokat lekerdezni\n");
        exit(-1);
    }

    struct sembuf buffer;

    buffer.sem_num = 4;     //a 4.ik szemafort
    buffer.sem_op = 1;      //inkrementaljuk a szemaforokat
    buffer.sem_flg = 0666;  //jogok

    if (semop(semID, &buffer, 1)) {
        perror("Sikertelen\n");
        exit(-1);
    }
}
```

Semup

a. feladat

Írjon egy C nyelvű programot, melyben

- egyik processz létrehozza a szemafort (egyetlen elemi szemafort; inicializálja 1-re, vagy x-re, ha még nem létezik),
- másik processz használja a szemafort, belépési szakasz (down), a kritikus szakaszban alszik 2-3 sec-et, m pid-et kiír, kilépési szakasz (up), ezt ismételve 2x-3x (és a hallgató egyszerre indítson el 2-3 ilyen processzt),
- harmadik processzben, ha létezik a szemafor, akkor megszünteti".

Mentés: gyak11-2.c
A futtatás eredményét is tartalmazza a jegyzőkönyv.

```
/home/robo/CLionProjects/test/cmake-build-debug/test
Szam: 3
A szemafor erteke (1) : 3

Process finished with exit code 0
```

Gyak11-2

```
/home/robo/CLionProjects/test/cmake-build-debug/test
Kritikus szakasz
5
pid : 211
2
kritikus szakasz vege

Process finished with exit code 22
```

Gyak11-2-2

```
/home/robo/CLionProjects/test/cmake-build-debug/test
Torolve

Process finished with exit code 8
```

Gyak11-2-3