Matris Çarpımı

Ege Üniversitesi Bilgisayar Mühendisliği
İşletim Sistemleri 2
Ödev 1

Umut BENZER 05-06-7670 http://www.ubenzer.com/

0.2
28 Nisan 2010

Sorunun Cevabı

Pipe, mesaj kuyruğu, fifo, paylaşılan bellek gibi hiç bir süreçler arası iletişim mekanizması kullanmayacak olsanız ne gibi bir problemle karşılaşırdınız ve bu problemi nasıl çözerdiniz?

Bu yöntemlerin hiçbirini kullanmasaydım, birbiriyle ilişkili processler iletişim kuramayacaktı. Ancak bazen, birbirine paralel işleyebilecek işler için çocuk süreçler yaratılır ve bilgilerin toplanması için onlardan geri bilgi almak gerekebilir.

Bu sorunu çözmek için vfork ile çocuk süreç yaratılabilir. Çocuk süreçlerin hepsi aynı veri bölgesini paylaşacağından iletişim kurulmuş olur. Öte yandan hangi sürecin ne zaman çalışacağı bilinemediğinden, bu yöntemle iletişim kurmak çok zor olacaktır. Bir değişkene bir süreç bilgi yazabilir, sonra tekrar yazabilir, eskisi arada kaynayabilir ve bunun gibi birçok sorun olabilir.

Başka bir yöntem iki süreçten birinin yazıp diğerinin okuduğu bir dosya olabilir. Birer süreç bir geçici dosyaya mesajlarını yazar. Diğeri baştan başlayarak mesajları okur. Pipe yöntemini, dosyalarla simüle etmiş gibi oluruz.

Kaynak Kod

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>

#define LINE 255
int main() {

    /* Acilis yazilari BASLANGICI */
    printf("Umut BENZER\n");
    printf("05-06-7670\n");
    printf("Ege Universitesi Bilgisayar Muhendisligi 3. Sinif\n");
    printf("http://www.ubenzer.com\n");
    printf("Threads\n\n");
    /* Acilis yazilari SONU */

    int m;
    int n;
    int k;

    int pid;
    pid = fork();

    if (pid == 0) {
        /* B süreci */
        FILE *fp;
        if((fp=fopen("input.txt","r")) == NULL) {
            printf ("Dosya acilamadi.");
            exit(-1);
        }

        fscanf(fp,"%d %d %d",&m, &n, &k);

        int matrisA[m][n];
        int matrisB[n][k];
        int matrisC[m][k];

        int i;
        int j;

        for(i=0;i<m;i++) {
             for(j=0;j<k;j++) {
                matrisC[i][j] = 0;
            }
        }

        /* A MATRISINI OKU */
        for(i=0;i<m;i++) {
            for(j=0;j<n;j++) {
                fscanf(fp,"%d",&matrisA[i][j]);
            }
        }

        /* B MATRISINI OKU */
        for(i=0;i<n;i++) {
            for(j=0;j<k;j++) {
                fscanf(fp,"%d",&matrisB[i][j]);
            }
        }

        fclose(fp);

        /* EKRANA MATRISLERI YAZDIR */
        printf("Matris A:\n");
        for(i=0;i<m;i++) {
            for(j=0;j<n;j++) {
                printf("%d\t", matrisA[i][j]);
            }
            printf("\n");
```

```
        }

        printf("\nMatris B:\n");
        for(i=0;i<n;i++) {
            for(j=0;j<k;j++) {
                printf("%d\t", matrisB[i][j]);
            }
            printf("\n");
        }

        int pidler[m];
        int pipeler[i][2];
        for(i=0;i<m;i++) {

            if(pipe(pipeler[i]) < 0) fprintf(stderr, "Program coker. %d\n", errno);
            pidler[i] = fork();
            if (pidler[i] == 0) {
                close(pipeler[i][0]); /* OKuma ucunu kapat */
                int ic1,ic2;
                int temp = 0;
                for(ic1=0; ic1<k; ic1++) {
                    for(ic2=0; ic2<n; ic2++)  {
                        temp += matrisA[i][ic2] * matrisB[ic2][ic1];
                    }

                    /* integer to char array dönü&#351;ümü */
                    char array[LINE] = {0};
                    sprintf(array, "%d\n", temp);
                    write(pipeler[i][1],array,LINE);
                    temp = 0;
                }

                exit(0);

            } else if(pidler[i] > 0) {
                close(pipeler[i][1]); /* Yazma ucunu kapat. */

            } else {
                fprintf(stderr, "Fork yapamadim. %d\n", errno);
                exit(EXIT_FAILURE);
            }
        }
        for(i=0;i<m;i++) {
            wait(pidler[i]);
        }

        char veribuf[LINE];
        for(i=0;i<m;i++) {
            for(j=0;j<n;j++) {
                read(pipeler[i][0],veribuf,LINE);
                matrisC[i][j] = atoi(veribuf);
            }
        }

        printf("\nMatris C:\n");
        for(i=0;i<m;i++) {
            for(j=0;j<k;j++) {
                printf("%d\t", matrisC[i][j]);
            }
            printf("\n");
        }
        exit(0);

    } else if (pid > 0) {
        wait();
    } else {
        fprintf(stderr, "Fork yapamadim. %d\n", errno);
        exit(EXIT_FAILURE);
    }
}
```