



Practica 4

SISTEMAS OPERATIVOS 2

Pedro Pizarro Huertas

David Martin Vilar

Objetivo de la practica

El objetivo de la práctica es llevar a cabo la lectura del fichero por medio de diferentes hilos que van a hacer la lectura del fichero airports.csv y dades.csv como en las anteriores prácticas.

Pero en esta práctica tendremos que llevar a cabo todo esto de forma distinta ya que tendremos que crear x hilos que accederán al archivo y procesarán n datos.

Además, estos hilos no podrán acceder a los datos todos a la vez, podrán leer los datos uno a uno.

Para el desarrollo de esta memoria explicaré la implementación más importante que permite que esto funcione.

Desarrollo de la práctica.

Toda la práctica se lleva a cabo en el fichero ficheros-csv.c y .h.

Lo primero a destacar es la creación de los hilos:

```
for(i = 0; i < NUMTHREADS; i++){
    pthread_create(&(vt[i]), NULL, th_read_airports_data, (void *) par);
}
for(i = 0; i < NUMTHREADS; i++)
    pthread_join(vt[i], NULL);
```

Para explicar esto, nosotros definimos NUMTHREADS en ficheros-csv.h:

```
#define NUMTHREADS X
```

El primer for creará los hilos (O threads) que llevarán a cabo el código del método th_read_airports_data de forma concurrente. Además, le pasamos el parámetro par, que es una estructura que contiene el árbol y el fichero a leer.

El segundo for concluirá el trabajo de los threads creados.

Dentro del método de th_read_airports_data encontraremos:

```
while ((currentLine < NUMLINES) && (fgets(line, MAXCHAR, fp) != NULL)){
    invalid = extract_fields_airport(line,&fi);

    if (!invalid){
        list_fi[currentLine] = fi;
        currentLine++;
    }
}
```

Aquí, el thread creado lee una a una las líneas del fichero mientras no llegue al número NUMLINES (Definido también en el fichero csv.h), el cual será el máximo de líneas que leerá cada thread.

A cada línea leída añadirá los datos extraídos por el método dado `extract_fields_airports` (El cual extrae los datos necesarios de las columnas indicadas también en el fichero csv.h los cuales hacen referencia a origen, destino y delay de vuelos) a una lista inicializada anteriormente:

```
flight_information *list_fi = malloc(sizeof(flight_information)*NUMLINES);
```

La lista contiene los datos extraídos de las columnas, los cuales son los necesarios para añadir los vuelos. Pero se añadirán a nuestro árbol a continuación debido a que como esto se lleva a cabo por distintos hilos, todos no pueden acceder a la lectura a la vez, por lo tanto, esto se tendrá que limitar.

Para limitar la entrada a las lecturas utilizamos los métodos:

```
pthread_mutex_lock(&mutex);
```

y

```
pthread_mutex_unlock(&mutex);
```

Los cuales rodearán el while que extrae la información.

El parámetro pasado es un mutex inicializado como variable global

```
pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;
```

El mutex es compartido por todos los threads, por lo tanto, cuando un hilo haga lock del mutex, cuando los demás lleguen, se quedarán a la espera de que el mismo thread que hizo lock, haga unlock, una vez se haga unlock el próximo hará lock y así con todos los threads.

A continuación veremos que se pasa a insertar los datos nuevos al árbol:

```
currentLine = 0;
```

```
while ((currentLine < NUMLINES) && (strlen(list_fi[currentLine].origin) == 3)){
    insert_node_trees(
        list_fi[currentLine].origin,
        list_fi[currentLine].destination,
        list_fi[currentLine].delay, par);
    currentLine++;
}
free(list_fi);
```

En esta parte del código vemos que todos los datos recopilados antes se utilizan para añadirlos al árbol con el método “`insert_node_trees`” y al final se limpia la lista.

Esto no podía llevarse a cabo antes debido a que, si un hilo además de leer los datos tuviese que insertarlos mientras ha hecho el lock del mutex, el hilo anterior debería esperar mucho

más. De esta forma mientras este hilo añade los datos al árbol el próximo thread estará leyendo los datos, de esta forma conseguimos más eficiencia.

Pero ahora, los hilos, podrían coincidir a la hora de leer un nodo en particular, esto podría presentar un problema. Por lo que cada nodo tiene un mutex particular el cual se inicializa cuando el árbol es creado:

```
pthread_mutex_init(&(n_data->mutex), NULL);
```

Se utiliza el método anterior antes de definitivamente añadir un nodo al árbol en el método "read_airports", esto se hace por supuesto con la inicialización de un mutex en la estructura del nodo en el red-black-tree.h.

Una vez dicho todo esto, el método "insert_node_trees":

```
void insert_node_trees(char *origin, char *destination, int delay, struct parameters *par){

    int currentLine = 0;
    node_data *n_data;
    list_data *l_data;

    rb_tree *tree = par->tree;

    if (strlen(origin) >= 3){

        n_data = find_node(tree, origin);

        if (n_data) {

            pthread_mutex_lock(&(n_data->mutex));
            l_data = find_list(n_data->l, destination);

            if (l_data) {
                l_data->numero_vuelos += 1;
                l_data->retardo_total += delay;
            } else {
                l_data = malloc(sizeof(list_data));

                l_data->key = malloc(sizeof(char) * 4);
                strcpy(l_data->key, destination);

                l_data->numero_vuelos = 1;
                l_data->retardo_total = delay;

                insert_list(n_data->l, l_data);
            }
            pthread_mutex_unlock(&(n_data->mutex));
        } else {
            printf("ERROR: aeropuerto %s no encontrado en el arbol.\n", origin);
            exit(1);
        }
    }
}
```

```

    }
    currentLine++;
}
}

```

Detallar que se ha hecho el `strlen(origin)>=3` debido a que habían textos con campos vacíos en el csv que daban error.

Bien, podemos ver que en el momento que accede a añadir los datos de la lista que irá en el nodo en cuestión, justo después de encontrar el nodo origen, se hace un block del mutex del nodo, y después de insertarlo su respectivo unlock.

Esto permite que cuando un hilo entre a un nodo, si otro hilo tiene que acceder al mismo nodo, no pueda acceder hasta que estuviese introduciendo los datos acabe.

Con todo esto hemos logrado que los threads puedan convivir y concurrir sin problemas de colisiones.

Resultados de la práctica.

Resultados sin threads:

```

ziclon@ziclon-TravelMate-P256-MG: ~/Escritorio/año4/SO2/practica 3 profe
1 - Creacio de l'arbre
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informacio de l'arbre
5 - Sortir

Escull opcio: 1

Alliberant arbre.
Introdueix fitxer que conte llistat d'aeroports: aeroports.csv
Introdueix fitxer de dades: dades.csv
Tiempo para crear el arbol: 0.013855 segundos

Menu
1 - Creacio de l'arbre
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informacio de l'arbre
5 - Sortir

Escull opcio: 1

Alliberant arbre.
Introdueix fitxer que conte llistat d'aeroports: aeroports.csv
Introdueix fitxer de dades: dades.csv
Tiempo para crear el arbol: 0.010209 segundos

Menu
1 - Creacio de l'arbre
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informacio de l'arbre
5 - Sortir

Escull opcio: 

```

El tiempo oscila entre 0.01 y 0.013

Resultados con un thread leyendo 10000 líneas:

```
ziclon@ziclon-TravelMate-P256-MG: ~/Escritorio/año4/SO2/Practica 4
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informació de l'arbre
5 - Sortir

Escull opció: 1

Introdueix fitxer que conte llistat d'aeroports: aeroports.csv
Introdueix fitxer de dades: dades.csv
SOY EL HILO : -1-
Tiempo para crear el arbol: 0.010552 segundos

Menu
1 - Creacio de l'arbre
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informació de l'arbre
5 - Sortir

Escull opció: 1

Alliberant arbre.
Introdueix fitxer que conte llistat d'aeroports: aeroports.csv
Introdueix fitxer de dades: dades.csv
SOY EL HILO : -2-
Tiempo para crear el arbol: 0.012457 segundos

Menu
1 - Creacio de l'arbre
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informació de l'arbre
5 - Sortir

Escull opció: █
```

El tiempo es igual que sin threads.

Resultados con 2 threads leyendo 5000 lineas:

```
ziclon@ziclon-TravelMate-P256-MG: ~/Escritorio/año4/SO2/Practica 4
4 - Consultar informació de l'arbre
5 - Sortir

Escull opció: 1

Introdueix fitxer que conte llistat d'aeroports: aeroports.csv
Introdueix fitxer de dades: dades.csv
SOY EL HILO : -1-
SOY EL HILO : -2-
Tiempo para crear el arbol: 0.010938 segundos

Menu
1 - Creacio de l'arbre
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informació de l'arbre
5 - Sortir

Escull opció: 1

Alliberant arbre.
Introdueix fitxer que conte llistat d'aeroports: aeroports.csv
Introdueix fitxer de dades: dades.csv
SOY EL HILO : -3-
SOY EL HILO : -4-
Tiempo para crear el arbol: 0.010772 segundos

Menu
1 - Creacio de l'arbre
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informació de l'arbre
5 - Sortir

Escull opció: █
```

Igual.

Tiempo con 4 threads leyendo 2500 lineas:

```
ziclon@ziclon-TravelMate-P256-MG: ~/Escritorio/año4/SO2/Practica 4
Introdueix fitxer que conte llistat d'aeroports: aeroports.csv
Introdueix fitxer de dades: dades.csv
SOY EL HILO : -1-
SOY EL HILO : -2-
SOY EL HILO : -3-
SOY EL HILO : -4-
Tiempo para crear el arbol: 0.010217 segundos

Menu
1 - Creacio de l'arbre
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informacio de l'arbre
5 - Sortir

Escull opcio: 1

Alliberant arbre.
Introdueix fitxer que conte llistat d'aeroports: aeroports.csv
Introdueix fitxer de dades: dades.csv
SOY EL HILO : -5-
SOY EL HILO : -6-
SOY EL HILO : -7-
SOY EL HILO : -8-
Tiempo para crear el arbol: 0.010710 segundos

Menu
1 - Creacio de l'arbre
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informacio de l'arbre
5 - Sortir

Escull opcio: █
```

Igual

10 hilos 1000 lineas:

```
ziclon@ziclon-TravelMate-P256-MG: ~/Escritorio/año4/SO2/Practica 4
SOY EL HILO : -19-
SOY EL HILO : -20-
Tiempo para crear el arbol: 0.011665 segundos

Menu
1 - Creacio de l'arbre
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informacio de l'arbre
5 - Sortir

Escull opcio: 1

Alliberant arbre.
Introdueix fitxer que conte llistat d'aeroports: aeroports.csv
Introdueix fitxer de dades: dades.csv
SOY EL HILO : -21-
SOY EL HILO : -22-
SOY EL HILO : -23-
SOY EL HILO : -24-
SOY EL HILO : -25-
SOY EL HILO : -26-
SOY EL HILO : -27-
SOY EL HILO : -28-
SOY EL HILO : -29-
SOY EL HILO : -30-
Tiempo para crear el arbol: 0.009145 segundos

Menu
1 - Creacio de l'arbre
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informacio de l'arbre
5 - Sortir

Escull opcio: █
```

Podemos ver que sean los hilos que sean no van a modificar el resultado en cuestión de tiempo.

Si vemos los datos finales del árbol podemos ver que son correctos ya que son iguales que las lecturas sin threads.

LAS:

```
ziclon@ziclon-TravelMate-P256-MG: ~/Escritorio/año4/SO2/Practica 4
Introdueix aeroport per cercar retard o polsa enter per saber l'aeroport amb mes destins: LAS
Media de retards para LAS
Retardos para el aeropuerto: LAS
TUS -- 43.273 minutos
TUL -- 56.250 minutos
TPA -- 27.750 minutos
STL -- 34.818 minutos
SNA -- 16.261 minutos
SMF -- 32.552 minutos
SLC -- 34.000 minutos
SJC -- 41.719 minutos
SFO -- 80.125 minutos
SEA -- -1.800 minutos
SDF -- 52.750 minutos
SAT -- 14.533 minutos
SAN -- 28.333 minutos
RNO -- 28.429 minutos
RDU -- 22.000 minutos
PVD -- 43.250 minutos
PIT -- 23.875 minutos
PHX -- 18.576 minutos
PHL -- 22.714 minutos
PDX -- 17.667 minutos
ORF -- 52.667 minutos
ONT -- 18.763 minutos
OMA -- 39.917 minutos
OKC -- 17.000 minutos
OAK -- 30.327 minutos
MSY -- 14.125 minutos
MHT -- 12.000 minutos
MDW -- 23.391 minutos
MCO -- 61.500 minutos
MCI -- 23.261 minutos
MAF -- 50.500 minutos
LIT -- 3.250 minutos
LBB -- 30.500 minutos
LAX -- 44.623 minutos
ISP -- 54.000 minutos
IND -- 88.875 minutos
IAD -- 13.250 minutos
```

ABQ:

```
ziclon@ziclon-TravelMate-P256-MG: ~/Escritorio/año4/SO2/Practica 4
5 - Sortir
Escull opcio: 4
Introdueix aeroport per cercar retard o polsa enter per saber l'aeroport amb mes destins: ABQ
Media de retards para ABQ
Retardos para el aeropuerto: ABQ
TUS -- 13.500 minutos
TPA -- 6.333 minutos
STL -- -4.667 minutos
SLC -- 9.833 minutos
SEA -- -3.667 minutos
SAN -- 27.333 minutos
PHX -- 11.600 minutos
PDX -- 26.333 minutos
OAK -- 33.500 minutos
MDW -- -9.667 minutos
MCO -- 1.333 minutos
MCI -- 18.833 minutos
MAF -- 21.000 minutos
LBB -- 25.000 minutos
LAX -- 25.083 minutos
LAS -- 22.471 minutos
HOU -- 26.125 minutos
ELP -- 18.429 minutos
DEN -- 20.222 minutos
DAL -- 17.960 minutos
BWI -- -13.333 minutos
AMA -- 53.667 minutos
Menu
1 - Creacio de l'arbre
2 - Emmagatzemar arbre a disc
3 - Llegir arbre de disc
4 - Consultar informacio de l'arbre
5 - Sortir
Escull opcio: 
```