Paralelización con hebras: pthread y mutex en Premios MVP - UEFA Champions League 2023/24

Sistemas Operativos

19 de Abril de 2025

1. Descripción del Problema

En esta tarea de programación, se requiere implementar un programa en C que procese un archivo de datos que contiene información sobre los partidos de la UEFA Champions League 2023/24 y el jugador elegido como "Man of the Match" (MVP) en cada uno. El objetivo principal es contar cuántos premios MVP recibió cada jugador a lo largo del torneo, utilizando múltiples hebras (`pthreads`) para paralelizar el procesamiento del archivo y un `mutex` para garantizar la correcta sincronización al actualizar los contadores compartidos.

2. Requisitos

1. El programa debe aceptar dos argumentos al ejecutarse:

   a) El nombre del archivo de datos (ej. `champions_mvp_23_24.txt`).

   b) El número de hebras (N) que se utilizarán para procesar el archivo.

   Si no se proporcionan los argumentos correctos o el archivo no se encuentra, se debe informar al usuario y terminar la ejecución.

2. El archivo de entrada contendrá 125 líneas, cada una representando un partido con el siguiente formato: `Fase,EquipoLocal,EquipoVisitante,Resultado,JugadorMVP`. El programa debe ser capaz de extraer el nombre del `JugadorMVP`.

mvp_champions_23_24

```
1  Grupo MD1,AC Milan,Newcastle,0-0,Rafael Leão
2  Grupo MD1,Young Boys,Leipzig,1-3,Xavi Simons
3  Grupo MD1,Feyenoord,Celtic,2-0,Calvin Stengs
4  Grupo MD1,Lazio,Atlético de Madrid,1-1,Antoine Griezmann
5  Grupo MD1,Paris,Dortmund,2-0,Vitinha
6  Grupo MD1,Man City,Crvena zvezda,3-1,Rodri
7  Grupo MD1,Barcelona,Antwerp,5-0,João Félix
8  Grupo MD1,Shakhtar Donetsk,Porto,1-3,Galeno
9  Grupo MD1,Galatasaray,Copenhagen,2-2,Tetê
10 Grupo MD1,Real Madrid,Union Berlin,1-0,Jude Bellingham
11 Grupo MD1,Bayern,Man United,4-3,Leroy Sané
12 Grupo MD1,Sevilla,Lens,1-1,Lucas Ocampos
13 Grupo MD1,Arsenal,PSV Eindhoven,4-0,Martin Ødegaard
14 Grupo MD1,Braga,Napoli,1-2,Victor Osimhen
15 Grupo MD1,Benfica,Salzburg,0-2,Roko Šimić
16 Grupo MD1,Real Sociedad,Inter,1-1,Brais Méndez
17 Grupo MD2,Union Berlin,Braga,2-3,Bruma
```

Figura 1: Ejemplo de la estructura de las líneas en el archivo de datos `champions_mvp_23_24.txt`

3. El programa principal leerá el archivo para determinar el número total de líneas (partidos).

4. Se deben crear exactamente N hebras trabajadoras utilizando `pthread_create`.
5. El trabajo (las 125 líneas/partidos) debe distribuirse equitativamente entre las N hebras. Cada hebra será responsable de procesar un subconjunto de las líneas del archivo. Se debe manejar correctamente la división si 125 no es divisible exactamente por N a través de una aproximación.
6. Cada hebra procesará las líneas asignadas, extrayendo el nombre del `JugadorMVP` de cada una.
7. Se debe utilizar una estructura de datos compartida (accesible por todas las hebras) para almacenar el conteo de premios MVP por cada jugador (por ejemplo, una tabla hash, un array de estructuras, etc.).
8. Se debe implementar un `pthread_mutex_t` para proteger el acceso a la estructura de datos compartida. Cada vez que una hebra necesite actualizar el contador de un jugador, debe adquirir el `mutex`, realizar la actualización y luego liberarlo.
9. El proceso principal (la hebra inicial) debe esperar a que todas las N hebras trabajadoras terminen su ejecución utilizando `pthread_join`.
10. Una vez que todas las hebras hayan finalizado, el proceso principal debe crear un archivo de salida llamado `reporte_mvp.txt` y escribir en él un resumen de los resultados: una lista de cada jugador que recibió al menos un premio MVP y el número total de premios obtenidos ordenando por el número de premios. El formato puede ser similar al siguiente:

```
Jugador MVP      | Premios
-------------------------
Vinícius Júnior  | 3
Mats Hummels     | 3
Jude Bellingham  | 3
Galeno           | 3
Rasmus Falk      | 3
... (resto de jugadores) ...
```

3. Pautas Adicionales

- Asegurar una distribución eficiente y correcta de las líneas del archivo entre las hebras.
- El código debe ser claro, legible y estar bien comentado, explicando las partes clave de la lógica de paralelización y sincronización. Por ejemplo, indicar en donde se usa mutex y por qué motivo.

4. Evaluación

La tarea será evaluada en base a los siguientes criterios:

- Documentación: Variables y funciones declaradas acorde al problema, código limpio y correctamente comentado.
- Hebras y Sincronización: Creación y gestión correcta de hebras (`pthread_create`, `pthread_join`). Implementación y uso adecuado del `mutex` (`pthread_mutex_t`) para proteger el recurso compartido (contador de MVPs), evitando condiciones de carrera. Distribución correcta de la carga de trabajo entre hebras.
- Solución: El programa compila y se ejecuta correctamente. Divide correctamente el archivo de entrada. Agrega los conteos de MVP de forma precisa y maneja la concurrencia correctamente. Produciendo un archivo (`reporte_mvp.txt`) con los resultados correctos.

Plazo de Entrega

Fecha Límite de Entrega: 10 de Mayo de 2025 hasta las 23:59 hrs.