Programació multifil: funcions de bloqueig

Novembre 2022

1 Introducció

Les dues darreres pràctiques de Sistemes Operatius 2 se centren en l'ús de múltiples fils per incrementar l'eficiència computacional del codi proporcionat a la pràctica anterior. Aquesta pràctica se centra en la utilització de les funcions de creació i bloqueig de fils. La darrera pràctica se centrarà en la utilització de variables condicionals per implementar el paradigma de productor-consumidor.

A les següents seccions es detallen les tasques a realitzar. Per a la implementació de la solució es demana utilitzar monitors[1]. Les funcions que es mencionen en aquest document es trobareu descrites a les dues fitxes del campus.

2 Funcionalitat a implementar

En començar a executar el vostre programa només hi haurà un fil. Anomenarem aquest fil el **fil principal**. El fil principal llegirà els codis dels aeroports, funció `read_airports`. A continuació es crida a la funció `read_airports_data`, que és la que llegeix les dades del fitxer de vols.

L'objectiu és augmentar l'eficiència del processament de les dades dels vols fent servir múltiples fils. La idea és repartir l'extracció de les dades del fitxer dels vols entre múltiples fils. Per això es faran servir **fils secundaris**, que són els fils creats pel fil principal.

A continuació es descriu l'esquema a implementar. A la secció de planificació 3 es proposa com procedir per arribar a implementar aquest esquema:

1. El fil principal crida a la funció `read_airports_data`, obre el fitxer i llegeix només la capçalera.

2. Abans de començar a processar les dades, el fil principal crearà, amb la funció *pthread_create*, $F = 2$ fils secundaris que accediran de forma concurrent al fitxer de dades per extreure'n la informació que conté. En crear els fils secundaris el fil principal passarà als fils secundaris, preferentment per argument (i no per variables globals), totes les variables necessàries que els fils secundaris necessiten per processar les dades del fitxer.

3. Els $F = 2$ fils secundaris processaran concurrentment el fitxer de dades. Per fer-ho faran servir una estratègia que està descrita a sota.

4. Mentre els fils secundaris processen el fitxer, el fil principal es quedarà esperant, amb la funció *pthread_join*, que els fils secundaris acabin la seva feina. Observar doncs que els $F = 2$

[1] A data d'inici d'aquesta pràctica possiblement encara no s'han impartit a teoria. La pràctica es pot iniciar, però, sense aquest coneixement. Veure secció 3.

fils secundaris es creen en començar a processar el fitxer, i es destrueixen un cop acaben de processar-lo.

5. Un cop hagin finalitzat tots els fils secundaris amb la seva feina, el fil principal es despertarà, tancarà el fitxer i sortirà de l'aplicació imprimint per pantalla els resultats de l'anàlisi del fitxer.

A continuació es descriu l'algorisme que els fils secundaris faran servir per processar les dades del fitxer. Cada fil secundari realitzarà el següent bucle sobre el fitxer a processar:

1. El fil llegeix un bloc de N línies **seguides** del fitxer (N podrà tenir valors de 1, 10, 100, o 10.000, o més) i les emmagatzema a una variable pròpia del fil.

2. Un cop s'ha llegit el bloc de N línies el fil passarà a extreure la informació de cada línia (del bloc) per actualitzar la informació de la matriu compartida `num_flights`.

3. Un cop el fil ha actualitzat la informació a la matriu `num_flights` torna al pas 1 fins que no hi hagi més dades a llegir. Quan no hi hagi més dades a llegir els fils sortiran del bucle i de la seva funció d'entrada (conseqüentment, els fils secundaris deixaran d'executar-se).

Atès l'esquema presentat en aquesta secció es demana respondre a les següents preguntes a l'informe a entregar

1. Quines variables haurà de passar (per argument) el fil principal al fils secundaris?

2. Els fils secundaris accedeixen a recursos (variables) compartits entre ells. Quines seran les seccions crítiques? Quines parts del codi són les que s'han de protegir? Cal protegir la lectura del fitxer? Cal protegir l'extracció de dades del bloc? Cal protegir l'actualització de la variable `num_flights`? Comenteu la vostra resposta.

3 Planificació i implementació

Per planificar-vos la feina, és important entendre bé el funcionament dels fils. Es proposen els següents punts de treball:

1. Llegiu i experimenteu amb la fitxa del campus que parla de la creació de fils (document de programació amb fils, 1a part).

2. Modifiqueu el vostre codi perquè el fil principal comenci per crear només $F = 1$ fil secundari un cop ha llegit la capçalera del fitxer. En aquest cas, atès que només hi ha un fil secundari, no es produiran condicions de carrera (race conditions). L'objectiu amb aquest punt és assegurar que el processament de dades del fitxer es realitza correctament si només es fa servir un únic fil secundari.

3. Llegiu i experimenteu amb les les funcions de bloqueig que s'expliquen també a la següent fitxa penjada al campus. Tingueu en compte que per a la realització d'aquesta pràctica no cal utilitzar variables condicionals. Les variables condicionals es faran servir a la següent pràctica.

4. Modifiqueu el codi per fer servir $F = 2$ fils secundaris i introduïu al codi les funcions de bloqueig *pthread_mutex_lock* i *pthread_mutex_unlock* necessàries per tal d'assegurar exclusió mútua.

5. Assegureu-vos que els resultats estadístics que obteniu en processar el fitxer amb múltiples fils són els mateixos que els obtinguts executar el codi amb el codi de la pràctica 4 fent servir els fitxers `2007.csv` i `2008.csv`.

4 Entrega

Es demana **entregar un fitxer ZIP** que inclogui el codi i un informe associat a la feina feta. El nom del fitxer ha d'indicar el número del grup, seguit del nom i cognom dels integrants del grup (e.g. **P4_Grup07_nom1_cognom1_nom2_cognom2.zip**). L'informe a entregar ha d'estar en **format PDF** o equivalent (no s'admeten formats com `odt`, `docx`, ...). Indiqueu les respostes a les preguntes de l'enunciat. El codi font s'inclourà dins de la carpeta `src`. Aquí hi ha els detalls:

- La carpeta `src` contindrà el codi font de la pràctica. S'hi han d'incloure tots els fitxers necessaris per compilar i generar l'executable. El codi ha de compilar sota Linux amb la instrucció `make`. No us oblideu doncs d'incloure el fitxer *Makefile* corresponent. És convenient que proveu el codi compilat amb opcions d'optimització: no feu servir l'opció `-g` en compilar sinó que feu servir l'opció -O.

- L'informe (**màxim 4 pàgines**, en format PDF, sense incloure la portada) en què es respongui a les preguntes plantejades a la secció 2, així com el experiments de temps d'execució que s'han fet amb els fitxers `2007.csv` o `2008.csv` (no pas el `fitxer_petit.csv`). En fer les proves amb aquests fitxers es recomana no apuntar el primer o segon resultat que es pugui obtenir atès el primer cop que s'executi el codi el fitxer es quedarà, parcialment, a la memòria RAM associada a la màquina. És per això que es recomana repetir múltiples vegades el mateix experiment per veure el temps d'execució associat a un determinat experiment.

  Es proposa fer experiments analitzant el temps d'execució per a l'aplicació sense fils secundaris (el codi de la pràctica anterior) així com l'aplicació amb $F = 2$ fils. Podeu fer proves amb més de $F = 2$ fils secundaris i comentar si el rendiment augmenta indicant la raó.

  També es proposa analitzar el temps d'execució per a diferents valors d'N (veure secció 2). Podeu provar valors d'N(N pot tenir valors de 1, 10, 100, o 10.000, o més). Feu totes les proves en el mateix ordinador; els resultats obtinguts dependran molt del tipus de disc (magnètic o SSD) així com el tipus de processador i altres característiques de l'ordinador.

  A l'informe no s'han d'explicar les funcions o variables utilitzades. En tot cas, es pot incloure, si es creu necessari, un esquema de la solució implementada.

Els pesos de la pràctica són: 70% per al codi font i 30% per a l'informe entregat. Respecte el codi, es demana que funcions estiguin comentades, codi modular i net, bon estil de programació (per a les seccions crítiques només es permet que n'hi hagi un punt d'entrada i un punt de sortida), que el programa funcioni correctament, tota la memòria ha de ser alliberada i sense accessos invàlids a memòria. El document ha de tenir una **llargada màxima de 4 pàgines** (sense incloure la portada). L'informe s'avaluarà amb els següents pesos: proves realitzades i comentaris associats, un 60%; escriptura sense faltes d'ortografia i/o expressió, un 20%; paginació del document feta de forma neta i uniforme, un 20%.