Programació paral·lela

Novembre 2021

Índex

1 Introducció **1**

2 Algorisme de paral·lelització **2**

3 Implementació **2**

4 Entrega de la pràctica **4**

1 Introducció

L'objectiu d'aquesta pràctica se centra en utilitzar la programació paral·lela per executar una funció de forma més ràpida aprofitant la capacitat multiprocessador de l'ordinador. En particular, en aquesta pràctica ens centrarem en l'anomenat paral·lelisme per descomposició en les dades: es tracta d'aprofitar el fet que el conjunt original de dades es pot dividir en subconjunts més petits. Cada fil processa subconjunts diferents. Els resultats de cada fil és "combinen" per obtenir el resultat final.

El codi de partida de la pràctica és el que es va obtenir en acabar la pràctica 3: els elements de la matriu de recomanació són de tipus *char* i s'utilitza el *memset* a la funció *createEmptyRecommendationMatrix* per inicialitzar els elements d'aquesta. En cas que no es disposi d'aquest codi us el podeu baixar del campus (el codi estarà disponible un cop hagi finalitzat el termini per entregar la pràctica 3; es pot començar a treballar en aquesta pràctica sense tenir el codi esmentat).

La pràctica se centra en paral·lelitzar el bucle *for* de la funció *getRecommendedMovieForUser* que es troba al fitxer `RecommendationMatrix.c`. Aquesta és la funció que permet recomanar una pel·lícula a un usuari K fent servir la informació de les pel·lícules que han vist els altres usuaris. Observeu que el bucle *for* recorre totes les pel·lícules i mitjançant la crida a la funció *forecastRating*, amb un paràmetre de pel·lícula J, fa una predicció de la valoració que l'usuari K farà de la pel·lícula J. Observar que cadascuna d'aquestes crides són independents entre sí. Per tant, anem a aprofitar aquesta característica per implementar la funció de forma paral·lela.

A la secció 2 es descriu l'algorisme de paral·lelització i a la secció 3 es comenta breument com procedir per realitzar la implementació.

2 Algorisme de paral·lelització

Observar que la funció és altament paral·lelitzable atès que el conjunt sobre el qual es realitza el processament – totes les pel·lícules – pot dividir-se en grups de pel·lícules més petits de forma que cada fil pugui processar un (o més) grup de pel·lícules petit. Hi ha diverses formes de procedir per fer-ho. A continuació es comenta la forma amb què es proposa d'implementar en aquesta pràctica.

- El fil principal – aquell que comença a executar l'aplicació pel *main* i fa la crida a la funció *getRecommendedMovieForUser* – crea M fils secundaris (a nivell de programació és pot utilitzar, per exemple, un `#define` per tal d'indicar el nombre de fils M que es crearan). Aquests M fils secundaris són els que realitzaran el processament sobre els grups petits de pel·lícules. L'algorisme haurà d'estar implementat per a un valor M qualsevol, $M > 1$ (el valor màxim que prendrà és el nombre de processadors del qual disposa la màquina). Es recomana començar a programar l'algorisme per a un determinat nombre de fils secundaris, e.g. $M = 2$, i després ampliar-ho a un cas general.

- Per repartir la feina a realitzar entre els M fils secundaris es divideix el conjunt sobre el qual es realitza el processament en grups de mida N, $N \geq 1$ (de forma similar al valor d'M, el nombre de fils es pot implementar de forma senzilla fent servir un `#define`). A la figura 1 es mostra un exemple per a $M = 2$. Observar que l'assignació dels grups a cada fil es fa de forma dinàmica: cada fil comença agafant un grup de pel·lícules i quan hagi acabat de processar aquest grup agafa el següent grup a processar. A la figura 1 es pot observar que el fil 2 comença per agafar el primer grup de pel·lícules i el fil 1 n'agafa el segon. És possible que el fil 1 faci el processament sobre el seu grup més ràpid que el fil 2 (dependrà, entre altres coses, del temps de CPU que hagi estat assignat a cada fil). Per tant, es pot donar el cas que el fil 1 acabi abans que el fil 2. El fil 1 agafa doncs el següent grup a processar, veure figura 1. Mentre el fil 1 està processant les dades del nou grup acaba el fil 2 i aquest agafa un nou grup a processar. Aquesta assignació dinàmica finalitza un cop no hi hagi més usuaris a analitzar. Observar que el fil que processi el darrer grup d'usuaris pot tenir una mida inferior a N.

- La implementació de l'algorisme es farà fent servir l'estratègia *map-reduce*. El *map-reduce* és una estratègia de programació en què a) *Map*: els fils – els que processen les dades – emmagatzemen el resultat (parcial) de les dades que processen, b) *Reduce*: un cop els fils han acabat de processar totes les dades es recullen els resultats parcials de cada fil i es calcula el resultat final. En el context d'aquesta pràctica això es correspon a procedir fent que a) *Map*: els fils emmagatzemen localment (sense compartir-la amb cap altre fil) la pel·lícula que recomanarien a l'usuari K. Aquesta recomanació és la que obté cada fil a partir dels grups de pel·lícules que processen, b) *Reduce*: quan tots els fils han acabat de processar els grups de pel·lícules el fil principal recull els resultats parcials de cada fil (en total n'hi haurà M recomanacions) i calcula quina és la pel·lícula a recomanar a l'usuari.

3 Implementació

Es descriu a continuació la forma de procedir per realitzar la implementació. La pràctica s'ha d'implementar fent servir els POSIX threads (*pthreads*).

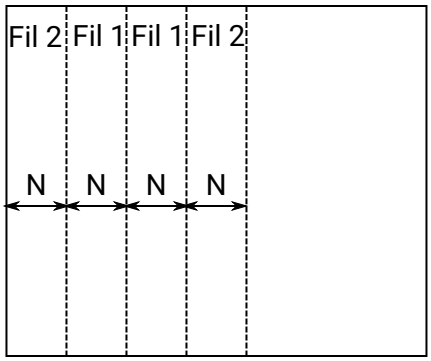

Figura 1: Repartició dinàmica del processament de les dades de la matriu de recomanació fent servir grups de N columnes.

Al campus disposeu de dos documents: un anomenat "Programació amb fils (1a part)" que se centra en les funcions de què disposem per crear fils, esperar que acabin, així com es poden passar i retornar dades a/dels fils. El segon document s'anomena "Programació amb fils (2a part)" que se centra en les funcions de què disposem per sincronitzar els fils.

Aquesta pràctica es pot implementar fent servir les funcions per crear fils i esperar que acabin (*pthread_create* i *pthread_join* respectivament) així com assegurar l'exclusió mútua entre fils mitjançant una clau (*pthread_mutex_lock* i *pthread_mutex_unlock*).

Per procedir amb la implementació de l'algorisme es recomana seguir els següents passos:

1. Comenceu amb una implementació que només crea un únic fil secundari. La idea és que el fil principal, en arribar a la funció *getRecommendedMovieForUser*, creï un únic fil secundari que realitzarà el processament sobre tot el conjunt de pel·lícules. Quan el fil secundari acabi el fil principal agafarà el resultat retornat pel fil secundari. Per implementar aquesta part es demana evitar utilitzar variables globals per passar i retornar les dades necessàries al/del fil secundari.

   Assegureu-vos, abans de començar amb la segona part, que aquesta primera part funciona correctament a partir de l'exemple de què disposeu al `Readme.txt`.

2. En el segon pas el fil principal crear M fils secundaris. Els fils secundaris processaran les dades (i.e. la matriu de recomanació) en grups de N pel·lícules. La repartició dels grups de pel·lícules entre els fils es farà de forma dinàmica tal com es descriu a la secció 3. Quina és la secció crítica? Dissenyeu-la de forma que el processament de cada grup de pel·lícules assignat a un fil es pugui fer en paral·lel amb els grups assignats a altres fils. Es demana, a més, que hi hagi un únic punt d'entrada i de sortida a la secció crítica. La clau associada a la secció crítica pot ser una variable global.

   Els fils secundaris emmagatzemaran de forma local el resultat parcial del processament – map – fins que acabin de processar tots els grups de pel·lícules. A mesura que els fils secundaris acabin, el fil principal recollirà els resultats parcials de cada fil secundari. Un cop hagin acabat tots els fils secundaris el fil principal podrà obtindrà el resultat final, i.e. la pel·lícula a recomanar a l'usuari. Farà falta una secció crítica per realitzar el reduce? De forma similar

al primer pas, per implementar aquesta part es demana evitar utilitzar variables globals per passar i retornar les dades necessàries als/dels fils secundaris.

Un cop hageu acabat amb la implementació es faran el següent experiments amb $M = 2$ fils (no fa falta provar altres valors). Per comparar els temps d'execució assegureu-vos que els codis es compilen amb les mateixes opcions de compilació, e.g. `CFLAGS=-O0 -Wall`.

1. Proveu d'executar l'aplicació amb $M = 2$ i diferents valors d'N ($N \geq 1$) com, per exemple, $N = 1$, 10, 100, 1.000, 10.000. Dóna sempre el mateix resultat?

2. Utilitzeu la comanda `time` per tal de mesurar el temps d'execució per als diferents valors d'N proposats. Recordar que cal executar aquests experiments en una màquina Linux (o Mac) nativa. Comenteu els resultats obtinguts i raoneu les diferències entre aquests. Hi ha valors d'N que sigui més òptimes, a nivell de temps d'execució, que altres valors?

4 Entrega de la pràctica

Es demana entregar un fitxer ZIP que inclogui el codi font així com les respostes a les preguntes realitzades en aquest document (i.e. "Quina és la secció crítica?", "Farà falta una secció crítica per realitzar el reduce?" i experiments amb diferents valors d'N). El nom del fitxer ha d'indicar el número del grup i els membres del grup (i.e. **P4_XX_nom1_cognom1_nom2_cognom2.zip** on XX és l'identificador de grup de parella).

En avaluar el codi font tindrà un pes d'un 80% i l'informe un 20%. L'informe a entregar ha d'estar en format `PDF` o equivalent (no s'admeten formats com `odt`, `docx`, ...).

Un informe està estructurat generalment en una introducció, treball realitzat i conclusions. A la part d'introducció es descriu breument el problema solucionat (i.e. resum del que proposa en aquesta pràctica). A la part de del treball realitzat es responen les preguntes realitzades al la pràctica i es descriuen les proves que s'han realitzat. Per respondre les preguntes podeu seguir un fil conductor que us permeti descriure el treball realitzat. També podeu respondre de forma explícita a cadascuna de les preguntes (en aquest cas indiqueu clarament a quina pregunta esteu responent en cada moment). Sigueu breus i clars en els comentaris i experiments realitzats, no cal que us esteneu en el text. Finalment, a la part de conclusions es descriuen unes conclusions tècniques de les proves realitzades. Per acabar, es poden incloure conclusions personals.

Inclogueu, preferentment en format text, les comandes que heu executat i algun comentari breu descrivint el resultat si ho creieu necessari. En cas que preferiu incloure captures de pantalla en comptes d'incloure el resultat en format text, assegureu-vos que el text de la captura es pot llegir bé (és a dir, que tingui una mida similar a la resta del text del document) i que totes les captures siguin uniformes (és a dir, que totes les captures tinguin la mateixa mida de text).

El document ha de tenir una llargada màxima de **3 pàgines** (sense incloure la portada). El document 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%.