Sistemes Operatius II

Informe de la Pràctica 4

Realitzat per Joaquim Yuste i Marcos Plaza

Índex

Introducció

En aquest informe trobareu les respostes a les preguntes plantejades en els diferents apartats de l'enunciat de la pràctica.

Resposta a les preguntes de l'enunciat

Podem veure que a l'apartat 2.4 (Creació de l'arbre fent servir múltiples fills) se'ns pregunta sobre si té sentit utilitzar més processos per a llegir la base de dades que el nombre de processadors que té el nostre ordinador. Podem veure que això no és així, ja que un dels avantatges de la programació concurrent és poder executar diverses tasques a la vegada. Cada processador té un fil d'execució i si volem executar més nombre de processos que de fils d'execució, la programació concurrent deixa de tenir sentit, ja que haurem d'anar executant, deixant d'un procés en estat preparat o bloquejat i executant un altre, i així successivament.

A continuació anomenarem els diferents avantatges o problemes per a les quatre solucions proposades a l'enunciat, per a evitar les *race conditions*[1]. Abans de començar a fer l'anàlisi de cadascun, val a dir que tant per aquesta solució com per totes les següents, existeixen una sèrie de precondicions a tenir en compte. Els recursos compartits per a cada procés, és a dir, tant l'arbre com els noms dels fitxers a processar (el fitxer de la base de dades) han d'estar dins un fitxer anònim, ja que aquest mètode és un dels més eficients en la comunicació entre processos amb relació pare-fill. A més tots els punts de vista, impliquen que hem de deixar passar d'un en un a cada procés a la regió crítica tot portant el control de quin fitxer s'està llegint i anar amb precaució i no agafar el mateix fitxer dues vegades.

a) fer una secció crítica en què estigui inclòs agafar el fitxer a processar, llegir i extreure les paraules, i actualitzar les dades dels nodes,

Per a aquesta solució hem de processar el fitxer, extreure totes les paraules i actualitzar el nombre de coincidències per a cada paraula del fitxer. Tot i que podem dir que és una solució correcta, hauríem de mantenir un fitxer anònim compartit per a alguna estructura de dades o variable que dugués el control dels fitxers llegits i els que no com hem dit amb anterioritat. A més estem ampliant molt la regió crítica (hem de realitzar moltes operacions per a cada procés) i no fem possible que altres processos tractin un fitxer diferent alhora, per tant la concurrència no es realitzaria correctament.

[1] Quan diversos fils (o processos) poden accedir per lectura o escriptura a zones de memòria compartida i no es controla l'ordre d'execució. Els resultats són impredictibles.

b) fer una secció crítica que, en extreure una paraula, inclogui comprovar si la paraula és a l'arbre i actualitzar el comptador si és necessari,

En aquest punt, estem reduint molt més la regió crítica. Quan executem el sem_wait() hem de veure de quina paraula es tracta en entrar a la secció i si la paraula és a l'arbre, actualitzar el número de coincidències per al node en particular abans de tornar a habilitar el semàfor. En aquesta ocasió, també és necessari que agafar el fitxer correcte per a cada procés, però, veiem que estem realitzant menys operacions per a cada procés amb un resultat igual de correcte.

c) fer una secció crítica que, únicament en cas que calgui actualitzar un comptador, bloquegi (tot) l'arbre i actualitzi el comptador corresponent,

Podríem dir que aquesta solució és la més efectiva, ja que estem reduint la secció crítica al màxim. Aquí només estem gestionant l'operació atòmica que ens interessa de debò. Estem dient a la cua dels processos a executar, que han d'incrementar només el nombre de coincidències. Com sempre hem de tenir cura amb quin fitxer llegeix cada procés.

d) fer una secció crítica que bloquegi cada node de forma independent cada cop que calgui actualitzar el comptador.

En aquesta ocasió fa falta configurar la lectura de cada node de manera que ell mateix bloquegi la lectura a altres processos, la qual cosa implica mantenir un semàfor per a cada node.

Solució implementada a la nostra pràctica

La nostra solució és més semblant a la proposta amb la lletra **c)**, ja que no fem la comprovació de la paraula prèviament, i anem a buscar si existeix el node per tot seguit agafar el seu num_times i incrementar-lo. Cal dir que fem servir dos semàfors; un per a incrementar la variable de l'identificador del fitxer i un altre per a sumar una unitat el número de coincidències de cada paraula.

A més a més, hem afegit un identificador de fitxer que ens servirà per trobar el següent arxiu a llegir per un procés. A l'hora d'incrementar aquest identificador fem servir semàfors, ja que tots els processos accedeixen a aquest valor.

Problemes trobats

Els semàfors no funcionaven correctament, això era degut al fet que la clau que feien servir no era accessible per a tots els processos, per solucionar-ho hem mapat a memòria aquesta clau possibilitant així una correcta lectura inter-procés.