Programación Concurrente
Práctico Taller 1: Region Labeling Problem

J. Blanco, N. Wolovick

25 de septiembre de 2001

El problema 3.26 de [1] se reproduce a continuación:

> El siguiente problema de etiquetado de regiones ocurre en el procesamiento de imagenes. Se da un arreglo entero `image[1:n,1:n]`, donde cada entrada representa la intensidad del pixel. Los vecinos de cada pixel son los cuatro pixels que están arriba, abajo, a la izquierda y a la derecha. Dos pixeles pertenecen a la misma región si son vecinos y tienen el mismo valor. Luego, una región es un conjunto maximal de pixels que están conectados y tienen todos el mismo valor.
> El problema consiste en encontrar todas las regiones y asignar a todos los pixels de cada región una etiqueta única. En particular sea `label[1:n,1:n]` una segunda matriz, y asuma que inicialmente `label[i,j]= n*i + j`. El valor final de `label[i,j]` deberá ser el máximo de las etiquetas iniciales de la región a la cual el pixel `[i,j]` pertenece.
>
> Escriba un *algoritmo paralelo de grilla* que calcule los valores finales de `label`. La computación deberá terminar cuando `label` no cambie.

Un *algoritmo paralelo de grilla* típico donde la granularidad es una componente paralela por elemento de grilla, se reproduce a continuación:

```
real grid[n+1,n+1], newgrid[n+1,n+1];
bool converged = false;

process Grid[i = 1 to n, j = 1 to n] {
        while (!converged) {
                newgrid[i,j] = F(neighbors(i,j));
                barrier();
                "calc_converged"
                grid[i,j] = newgrid[i,j];
                barrier();
        }
}
```

Una barrera es una primitiva de sincronización de alto nivel, denominada también *inclusión mutua* cuya *computation proper* se muestra a continuación:

Pre: $\langle \forall i : 0 \leq i < N : \neg c.i \wedge \neg x.i \rangle$
Comp.i: $\vdots$ $c.i := true$ barrera $x.i := true$ $\vdots$
Inv: $\langle \forall i : 0 \leq i < N : x.i \Rightarrow \langle \forall j : 0 \leq j < N : c.j \rangle \rangle$

Se pide:

1. Dar la Pre, Post y el Invariante de todo el sistema.
2. Demostrar que la Post implica que el *region labeling problem* está resuelto.
3. A partir de la especificación de barrera, derivar la barrera de contador.
4. Implementar el algoritmo en algún lenguaje que soporte paralelismo (Java, C+Pthreads, SR, etc.).
5. Explique porque no resulta necesaria más atomicidad que la lectura y escritura de valores escalares. También explique porque el algoritmo funciona correctamente (aunque la detección de terminación puede complicarse) si quitamos las barreras.

Referencias

[1] Gregory Andrews, *Foundations of Multithreaded, Parallel, and Distributed Programming*, Addison Wesley, 2000

[2] Doug Lea, *Concurrent Java Programming*, Java Series - Addison Wesley, 1999

[3] Scot Oaks and Henry Wong, *Java Threads*, O'Reilly, 1998