Homework 4 – Fork-Join Pattern & Pipeline Pattern

Due Date: Tuesday, 5/12

Fork-Join Pattern

1. Fundamentally, what does the Fork-Join pattern do for parallel computing? That is, what do the fork and join operations allow us to do?

2. Flood Fill is an algorithm that attempts to convert the color of cells in a grid (think of a "bucket fill" tool in paint programs). Below is a recursive parallel algorithm for a Flood Fill. cell is the current cell of the grid that we are examining. targetColor is the color we’re looking to change. newColor is the color that will replace targetColor

```plaintext
FloodFill(cell, targetColor, newColor)
1 | if targetColor == newColor then // just in case...
2 |     return
3 | if the color of cell ≠ targetColor then
4 |     return
5 | else
6     set color of cell to newColor
7     fork( FloodFill(one step west of cell, targetColor, newColor) )
8     fork( FloodFill(one step east of cell, targetColor, newColor) )
9     fork( FloodFill(one step north of cell, targetColor, newColor) )
10    FloodFill(one step south of cell, targetColor, newColor)
11     join()
12    return
```

(a) Using Figure 1 below, and the FloodFill() function above, draw the DAG of the forks and joins for FloodFill(21, white, blue). Be sure to show all work (i.e. which forks are doing what cells). You may include a new image if you’d like, but it is not necessary.

(b) What is the total amount of concurrency available in this operation?

(c) Are there any data races in this problem? Explain.

3. What is parallel slack, and why is it important to consider in fork-join problems?

Pipeline Pattern

1. What happens to available parallelism when the number of data items in a pipeline increases, but the number of serial tasks in a data item remains constant?

2. Describe the benefits and negatives of stage-bound workers, item-bound workers, and hybrid workers in a pipeline implementation. Can you think of scenarios where you would prefer one over the others?
Figure 1: Flood Fill Image