OpenMP 1

1 The Game of Life

Conways game of life is a simple cellular automaton. Conceptually, the universe in Life consists of a 2D grid where each cell can be either populated or unpopulated. In each timestep, a grid becomes

- populated, if it has exactly three populated neighbors
- unpopulated, if it has less than two or more than three neighbors
- all other cells remain as they are

The neighborhood here includes the corners, i.e. each cell neighbors 8 other cells. We set the boundary elements to zero and never change them.

```
int conway(int N, int M, int (*current_grid)[M], int (*next_grid)[M]) {
  int total = 0;
  for (int i = 1; i < N - 1; i++) {
    for (int j = 1; j < M - 1; j++) {
      int count = 0;
      for (int ii = -1; ii <= 1; ii++)
        for (int jj = -1; jj <= 1; jj++)
          count += current_grid[i+ii][j+jj];
      if (count == 3)
        next_grid[i][j] = 1;
      else if (count == 2)
        next_grid[i][j] = current_grid[i][j];
      else
        next_grid[i][j] = 0;
      total += next_grid[i][j];
    }
  }
  return total;
}
```

The file `omp-1-conway.c` contains a corresponding program.

- Parallelize the program using the constructs you have learned about in class.
- Can you have the entire program execute in a single parallel region?
- What if N or M are small?

2 Parallel Scan

A scan is an operation that, given a sequence (a_i) computes (b_i) with

$$b_i = \sum_{j<i} a_j$$

This is called an exclusive scan, because $b_i = a_0 + \cdots + a_{i-1}$ rather than $b_i = a_0 + \cdots + a_i$.

Clearly, a simple routine such as the one below contains a dependency that makes the calculation non-trivial.

```
void scan(int N, double* a, double* b) {
  b[0] = 0;
  for (int i = 1; i < N; i++) {
    b[i] = b[i-1] + a[i-1];
  }
}
```

- Try to parallelize the program using the constructs we have learned about in class. Hint: You might want to use a barrier at some point.