My strategy was to divide the board into n squares, where n is the closest perfect square to the number of threads. Each thread then sums the total number of ants for each square. Once all of the sums have been completed, I sort the boxes by number of ants and then start guessing each tile within each of the boxes, starting with the boxes with the highest number of ants.

One issue I encountered while implementing this strategy was that retrieving the number of ants for a given cell was atomically updating the query variable, meaning that it would block other threads which were trying to also retrieve the number of ants for a given cell.
I couldn’t find an easy way around this, so I removed the updating of query within the function for retrieving the number of ants. To be clear, my algorithm always makes n^2 queries where n is the length of the grid. I felt that this was an ok change since it doesn’t change the functionality of the method for retrieving the number of ants at a given cell.

The same thing is true for the guess function, it atomically blocks to update the guess variable. I also removed the updating of the guess variable. Instead I keep track of the number of guesses by locally updating the number of guesses made by each thread and then printing this out at the end.

My code can be run with:

```
module load intel

export OMP_NUM_THREADS=49

icpc -o anthill.exe anthill.cpp -qopenmp

./anthill.exe <length> <x> <y> <steps>
```