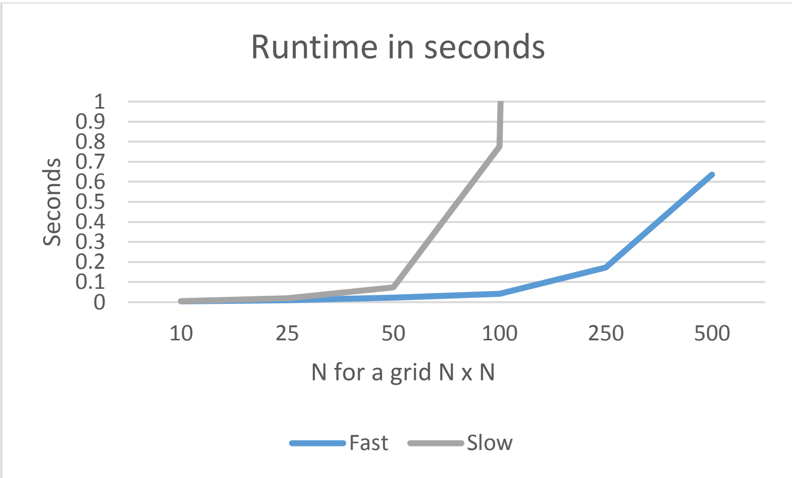
Analysis for PercolationStats.java

1. Run Time

Originally, I tested runtime in the lab machines directly. For that scenario, all times were below 240 seconds, with Slow 500 being obviously the slowest, marking roughly 230 over many attempts. The times presented here, are from running the same code remotely, therefore, somewhat larger. In a lab environment, the times should be less.

N	Fast	Slow
10	0.004	0.005
25	0.01	0.019
50	0.023	0.073
100	0.042	0.776
250	0.173	24.429
500	0.636	357.167

The following graph represents the values from the table above. Excluded are the Slow values for N=250 and 500, that were way too large by comparison, and are not truly needed in the graph in order to draw conclusions.

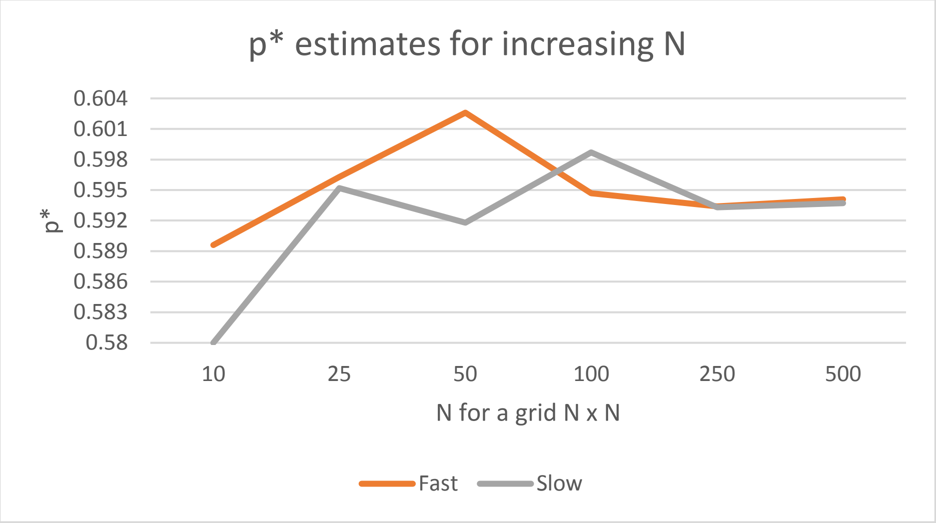


For our n by n grid, we know already that runtime doesn't increase linearly. This graphs also helps to observe that due to its increased number of operations (iterations), the Slow method (QuickFindUF) increases faster than the Fast method, for an increasing N. This difference also grows as N does, leading to the need of excluding the last two Slow results from the graph for it to be properly scaled. Noted be how the line for Slow shoots almost straight up from N=100 in order to attempt to catch up for the next N.

2. Estimate Threshold

In contrast to the runtime, the p* threshold was found to not depend on the method implemented. It depended on the value of N, with it approaching an undefined value as N grows. This means that as N grows, the value of p* varies less.

N	Fast	Slow
10	0.5896	0.58
25	0.5963	0.5952
50	0.6026	0.5918
100	0.5947	0.5987
250	0.5934	0.5933
500	0.5941	0.5937



In the handout it is stated that p* is a threshold value for an N by N grid with a sufficiently large N percolates with p cells opened. This makes it so that p* approaches a constant as N approaches infinity (grows very large). Here in the graph we can see how the difference of p* estimates for the Slow and Fast implementation is for the most part reduced as N grows. The only point that escapes this rule would be N=25, but this can easily be attributed to deviation. An important point to be made is that the difference in the implementations is not likely to be a very relevant factor in this differences of p*, instead, the cause for them is simply that N isn't large enough, so p* is more random.

As it can be observed, for N=250 and above, the difference in p* is almost negligible. Then, we can assume N>250 to be a good lower limit for the size of N to be used to estimate the threshold value.