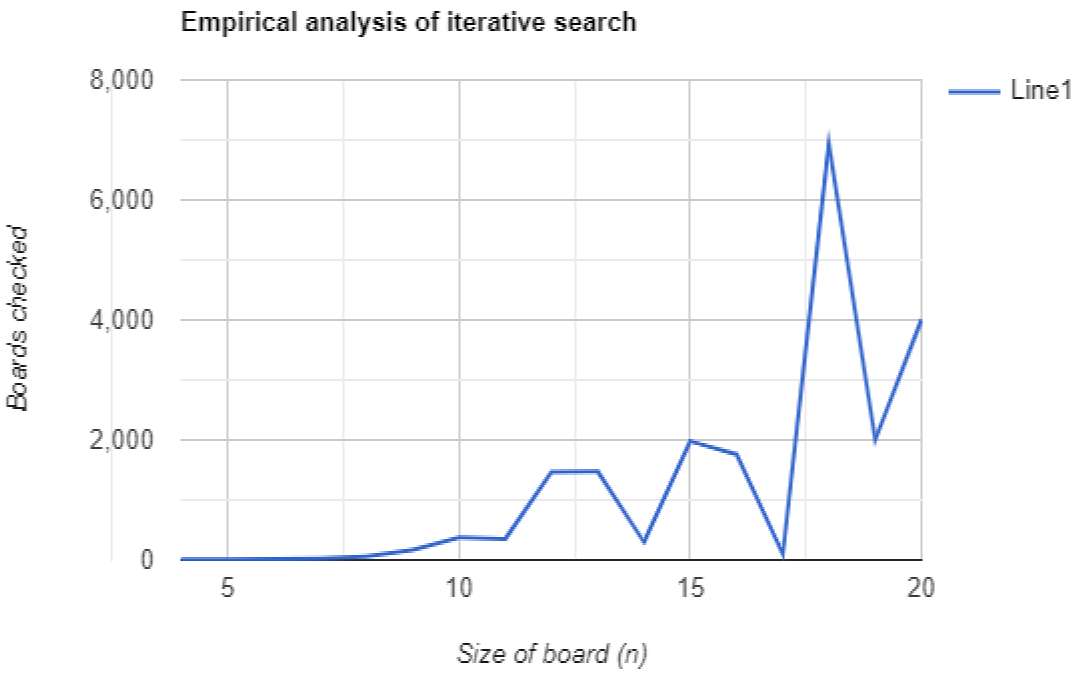
My iterative algorithm for solving the N queens problem uses “temp.attacks() < board.attacks()” (Line 131 of NQueens.java) as its basic operation. The amount of times this basic operation is checked for each n is displayed below. As you can see, it has a high variance from one value of n to the next, but generally checks more times as n grows.



Points on the above graph:
(4,5), (5,11), (6,18), (7,22), (8, 57), (9, 166), (10, 375), (11, 349), (12,1464), (13,1473), (14,299), (15,1980), (16,1760), (17,101), (18,6935), (19,2008), (20,4004)

Due to the variance in work done by this algorithm, there is no one equation to calculate the amount of work done for any given value of n, but by comparing this to the exhaustive search method, we can clearly see that even a value as high as 18 -with a high amount of work done by the computer- is not even remotely comparable to the work done while searching exhaustively. Conclusion: This method is extremely inconsistent, but leagues more efficient than the exhaustive method. Additionally, the amount of work done at each n value never changes, making it very predictable.