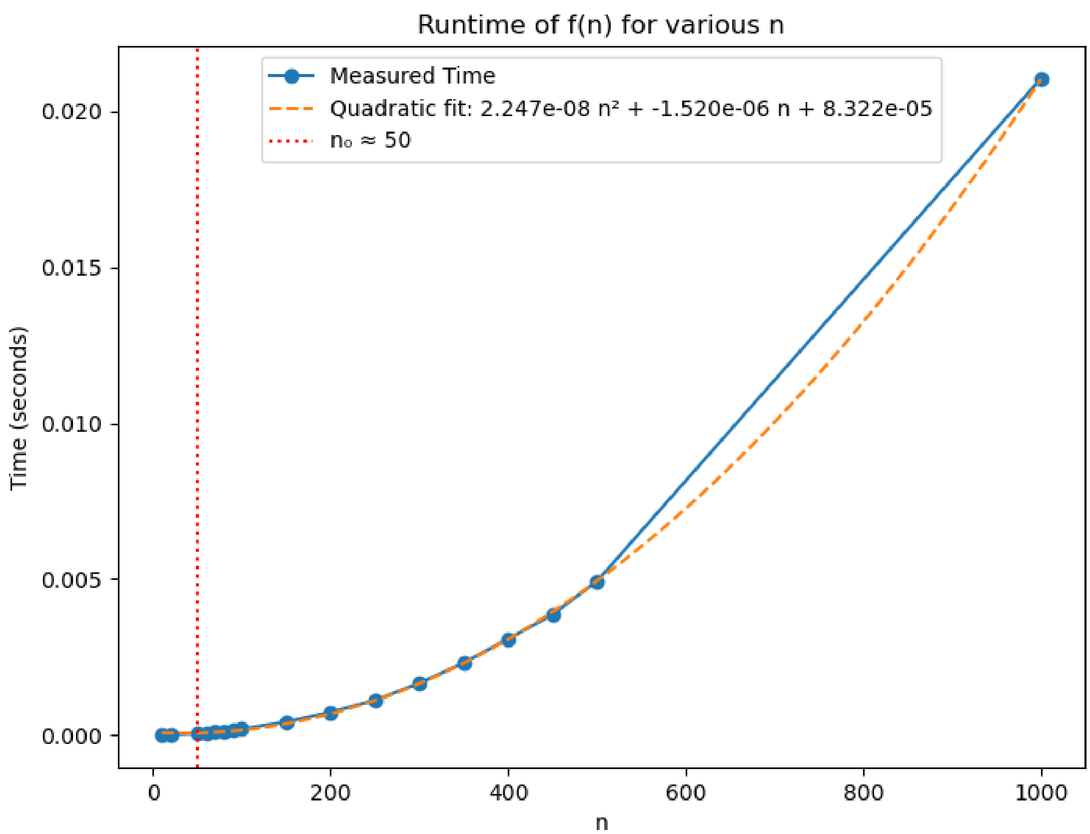
1) The runtime of the algorithm is given by

$$T(n) = \sum_{i=1}^{n} \sum_{j=1}^{n} 1 = n^2$$

Hence
Big Theta: $\Theta(n^2)$
Big Omega: $\Omega(n^2)$
Big O: $O(n^2)$

2) The polynomial graph is provided below:

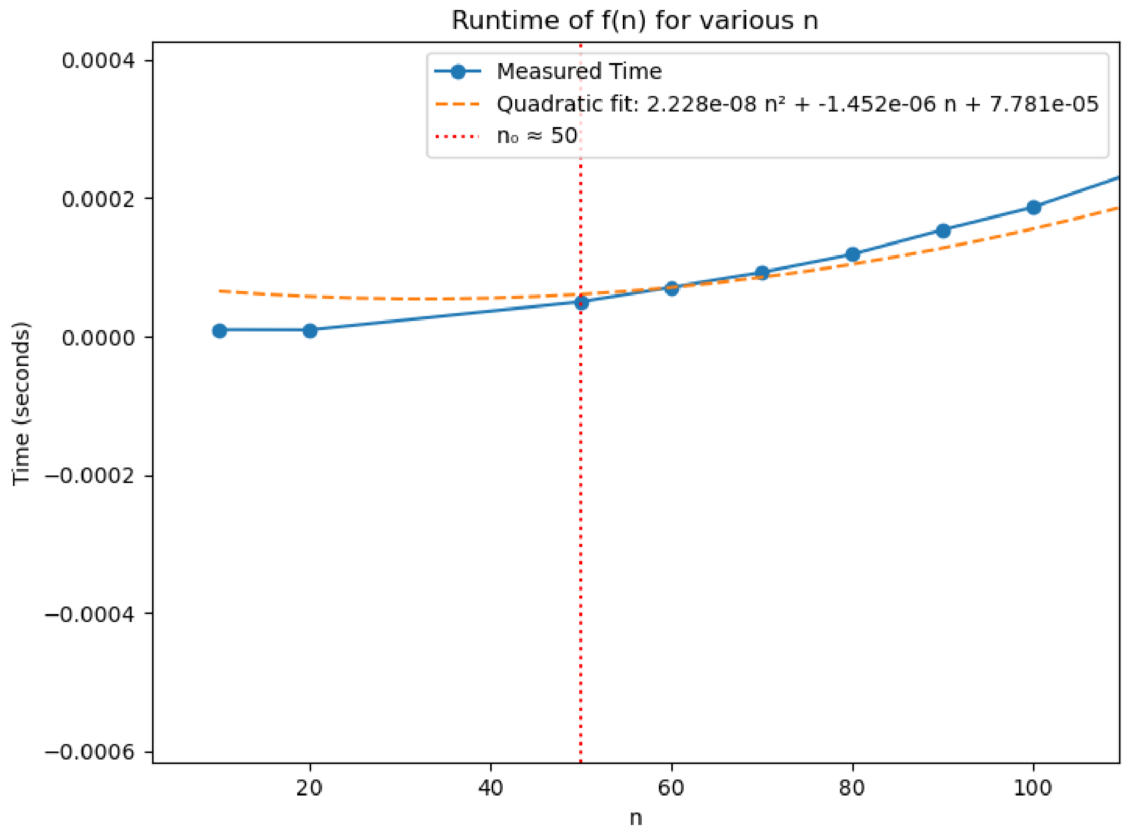


3) Since the n^2 term dominates for large n, we can state that there exist positive constants c_1 and c_2 such that, for all $n \geq n_0$ (for some threshold n_0):

$$c_1 n^2 \leq T(n) \leq c_2 n^2$$

Hence
Big-O : $O(n^2)$ (upper bound)
Big Omega: $\Omega(n^2)$ (lower bound)
Big Theta: $\Theta(n^2)$ (tight bound)

4) For $n < 50$, the timings deviate noticeably from the quadratic curve For $n \geq 50$, the data aligns well with the quadratic polynomial, indicating that the n^2 term dominates and the overhead is negligible.



4) In the modified function, the inner loop now performs **two** constant-time operations (one for updating *x* and one for updating *y*) instead of just one. This effectively increases the constant factor of the runtime.
5) It will not affect the asymptotic analysis from Part #1. Even though the actual runtime increases by a constant factor, the asymptotic behavior remains the same.