**Yuan Gao z5239220 Q4**

You are in an orchard which has a quadratic shape of size 4n by 4n with equally spaced trees. You purchased apples from n2 trees which also form a square, but the owner is allowing to choose such a square anywhere in the orchard. You have a map with the number of apples on each tree. Your task is to choose such a square which contains the largest total number of apples and which runs in time O(). Note that the brute force algorithm would run in time Θ(). (20 points)

Let’s assume:



According to this graph:

The value of the element in the lower right corner

Through this formula, the left side Sum square can be created.

The sum of square:

Therefore, we only need to traverse once to find the maximum value

The Time complexity is

1. /\*the Two-dimensional vector reprsent orchard,
2. n as the length of purchased square
3. ii as the maximum i index
4. jj as the maxinum j index \*/
5. **int** getMaxApple(vector<vector<**int**>> & orchard, **int** n, **int** & ii, **int** & jj){
6. **int** max = -1;
7. **int** tmpMax = 0;
8. **for**(**int** i = 1; i < 4 \* n; ++i){
9. orchard[i][0] += orchard[i - 1][0];
10. orchard[0][i] += orchard[0][i - 1];
11. }
12. **for**(**int** i = 1; i < 4 \* n; ++i){
13. **for**(**int** j = 1; j < 4 \* n; ++j){
14. /\*The value of the element in the lower right corner\*/
15. orchard[i][j] += (orchard[i - 1][j] + orchard[i][j - 1]
16. - orchard[i - 1][j - 1]);
17. **if**(i >= n && j >= n){
18. /\*get the sum of square\*/
19. **int** tmpMax = orchard[i][j] - orchard[i - n][j]
20. - orchard[i][j - n] + orchard[i - n][j - n];
21. **if**(tmpMax > max){
22. max = tmpMax;
23. ii = i;
24. jj = j;
25. }
26. }
27. }
28. }
29. **return** max;