CS 325 Group Project 2

Robert Detjens, Srikar Valluri, Felix Brucker

Scenario

Vankin's Mile is an American solitaire game played on an $n \times n$ square grid. The player starts by placing a token on any square of the grid. Then on each turn, the player moves the token either one square to the right or one square down. The game ends when player moves the token off the edge of the board. Each square of the grid has a numerical value, which could be positive, negative, or zero. The player starts with a score of zero; whenever the token lands on a square, the player adds its value to his score. The object of the game is to score as many points as possible.

In this assignment, you describe and analyze an efficient algorithm to compute the maximum possible score for a game of Vankin's Mile, given the $n \times n$ array of values as input.

Description

Our algorithm uses a memoization apprach to achieve an $O(n^2)$ runtime. We have implemented two functions to solve this problem:

The function `get_max_score` calculates the score of traversing in either the right or down directions by one square, and returns the maximum of those two values. As we do not care about the path taken, only the score, we only need to keep track of the maximum score for each particular location.

The function `vankin_max_score` uses `get_max_score` to determine the best possible score for the given location. In order to memoise the results, an additional matrix `scores` is used to record the maximum score for each location. `scores` is populated starting from the bottom right of the matrix and traversing row by row in a reverse rasterization pattern until all squares have been visited.

Runtime Analysis

Each location in the $n \times n$ board matrix is visted once. As there are n^2 amount of squares, the runtime for this algorithm is $O(n^2)$.

Each square only needs to be visited once as the score for all previously-visited values have been recorded in `scores` and can be referred to by adjacent squares in lieu of recalculating an entire path.