Dominoes Sequence Algorithm

Problem and Solution Description:

Modeling the domino dispositions and using graph theory, we found that problems can be described as:

1. Detect the number of sources in the non-heavy DAG;
2. Discover the greatest path present in dag not heavy.

The solution to the first problem is trivial since we only need to verify which vertices have an input degree of zero.

The second problem can be solved by using dynamic programming as follows:

The distance from a vertex S to vertex B will be the maximum distances from vertex S to predecessors b plus 1 to include the new traveled edge, i.e., $dist(B) \ = max_{(A,\,B)\in E}\{dist(A) \ + \ 1\}$.

The distances from the sources are 0. To prevent the algorithm from being recursive, the vertices are traversed in topological order, which ensures that when you reach vertex B, the distances to its parents have already been previously calculated.

Sources:
https://www.mathcs.emory.edu/~cheung/Courses/171/Syllabus/11-Graph/Docs/longest-path-in-dag.pdf
https://en.wikipedia.org/wiki/Topological_sorting#Kahn's_algorithm

Theoretical Analysis

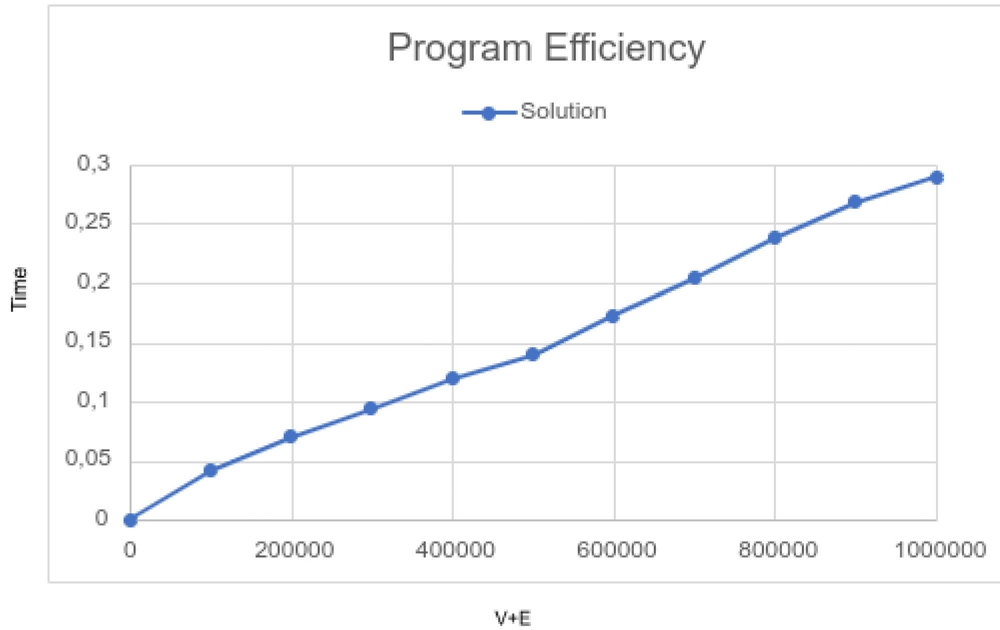
Steps:

- Reading input data: Simple file reading with cycles depending linearly on E. Thus O(E);
- Topological ordering: Initialization of the input degrees of vertices and stack depending linearly on V. Graph traversal depending linearly on V+E. Thus, O(V) + O(V+E) = O(V+E);
- Application of the longest path algorithm: Traversing the vertices in topological order depending linearly on V+E. Thus O(V+E);

Solution's Global Complexity: O(E) + O(V+E) + O(V+E) = O(V+E)

Experimental Analysis of Results

For each pair (V, E) measured the execution time, and used the results' arithmetic mean to plot the following graph:



We conclude, based on the analysis of the graph, that the execution time of the solution presented increases linearly with V+E as predicted in the theoretical analysis of the solution.