Depth first search

Initial analysis of DFS shows that the complexity of this algorithm is O(|V|+|E|) where: V represents the vertices and E the edges present in a graph. Three graphs below show the complexity relative to random inputs of increasing size:

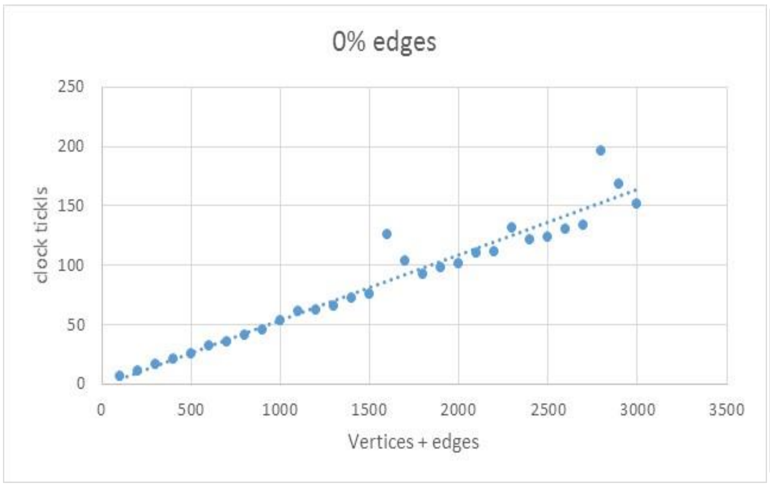

Figure D-1

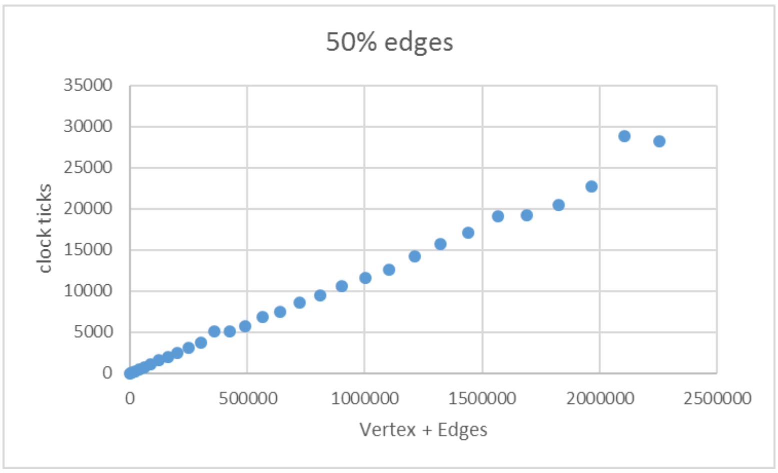

Figure D-2

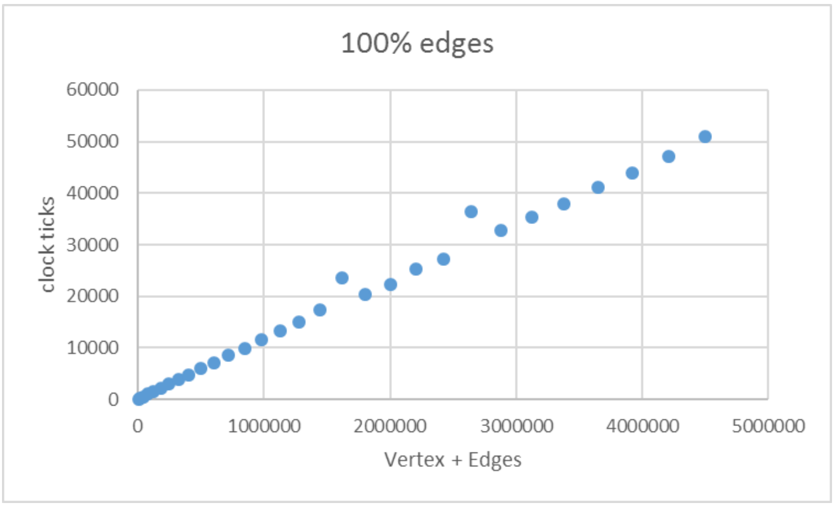

Figure D-3

All the graphs show linear growth with respect to V+E as expected. In the implementation of DFS, 2 arrays were initialized with values(false) to mark if a node was visited or not, hence using exactly Θ(|V|) and a loop that does subsequent calls to explore(v) if and only if v was not visited. This guarantees a maximum bound of |V| + |E|, since explored paths are never visited again. There are no pathological inputs that will change DFS' behaviour of linearity, as seen in Figure D-3, it grows linearly even when all possible vertices are connected. DFS is also able to spot cyclicity on the spot by identifying a back edge. If an edge points to a node/vertex that has a higher post number, or, a node that still exists in the stack, it then identifies it as a cyclic graph.

Khan sort

Initial analysis shows that Khan sort should execute in linear time, first by initializing a list of sources, that will take O(|V|), and looping while this list is not empty. In each iteration, a source is removed from the graph O(1), and for all child nodes from this source, the connecting edges are removed. There exists |E| edges, and removing edges are dealt by del(). Assuming a dense graph, a node might have an upper bound of |V|-1 edges (specifically the source). Therefore, considering the worst case for del(), the innermost loop runs for a total of |E|*(|V|-1). Hence, this implementation of Khan sort takes O(|V| + |E||V|). Note that push was used instead of insert for constant time adding, and the list was reversed afterwards O(|V|). To support this claim, assuming |E| stays constant, the graph should then run linearly with respect to |V|. From Figure K-1 and K-2, the graph grows linearly with respect to the number of vertices (with no edges) and constant edges. Figure K-3 shows how Khan grows linearly with respect to O(|V|+|V||E|).

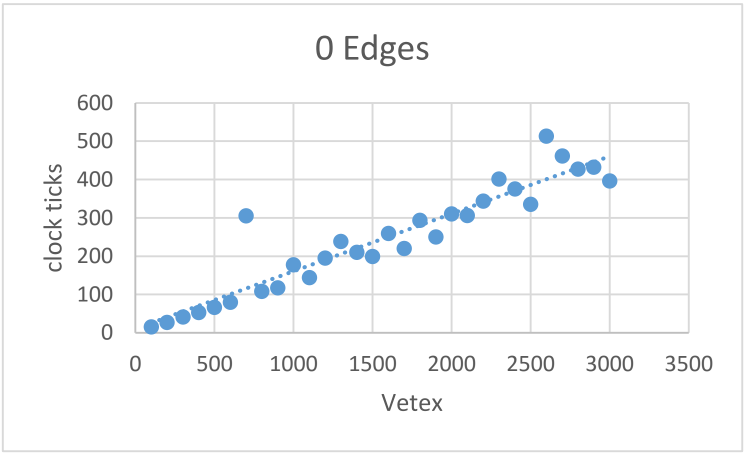

Figure K-1

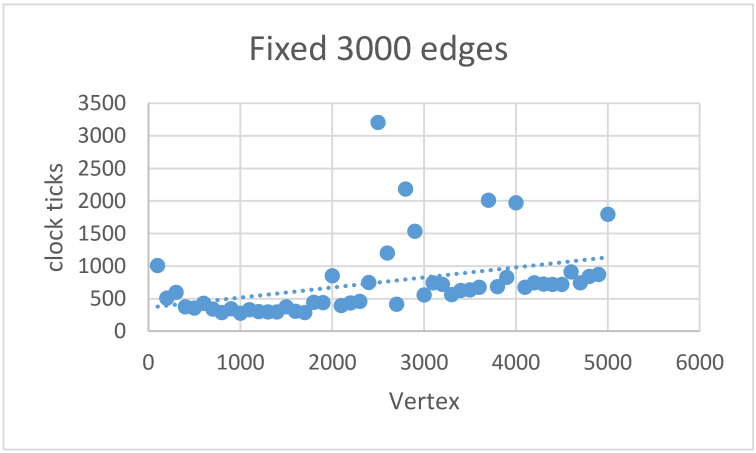

Figure K-2

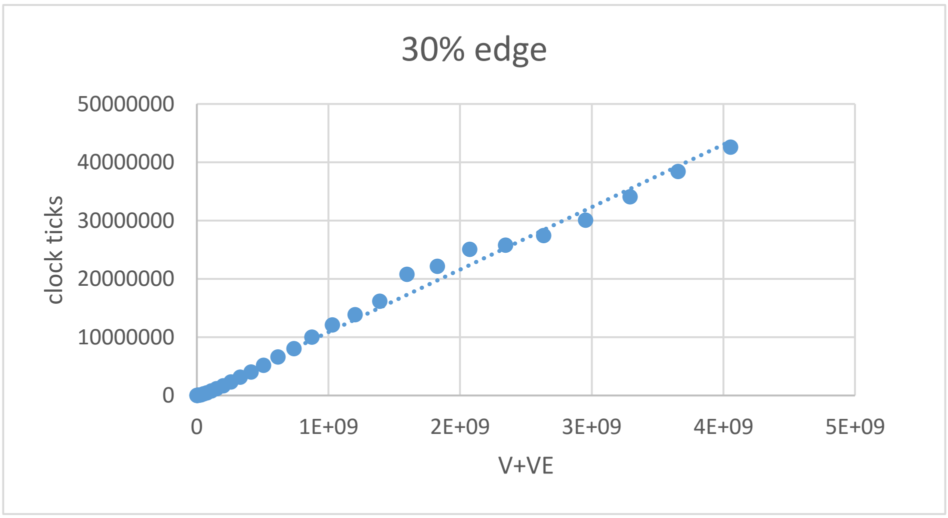

Figure K-3

Therefore, if V or E are not constants, Khan's algorithm becomes polynomial with respect to the size of the graph, V+E (Appendix K-4A). Khan's algorithm is also slow to detect cyclicity. It can only determine if a graph is cyclic after the loop ends.

Appendix

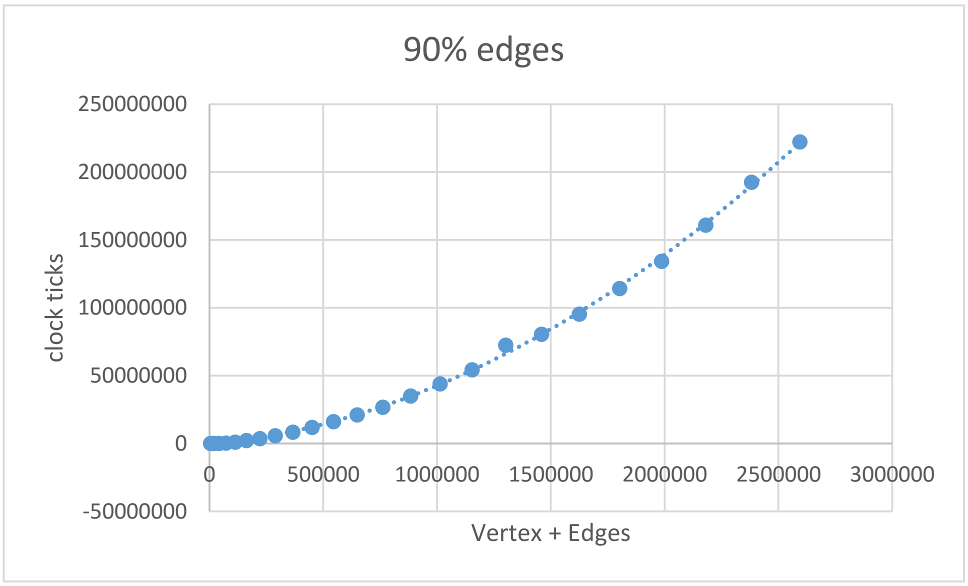

Figure K-4A