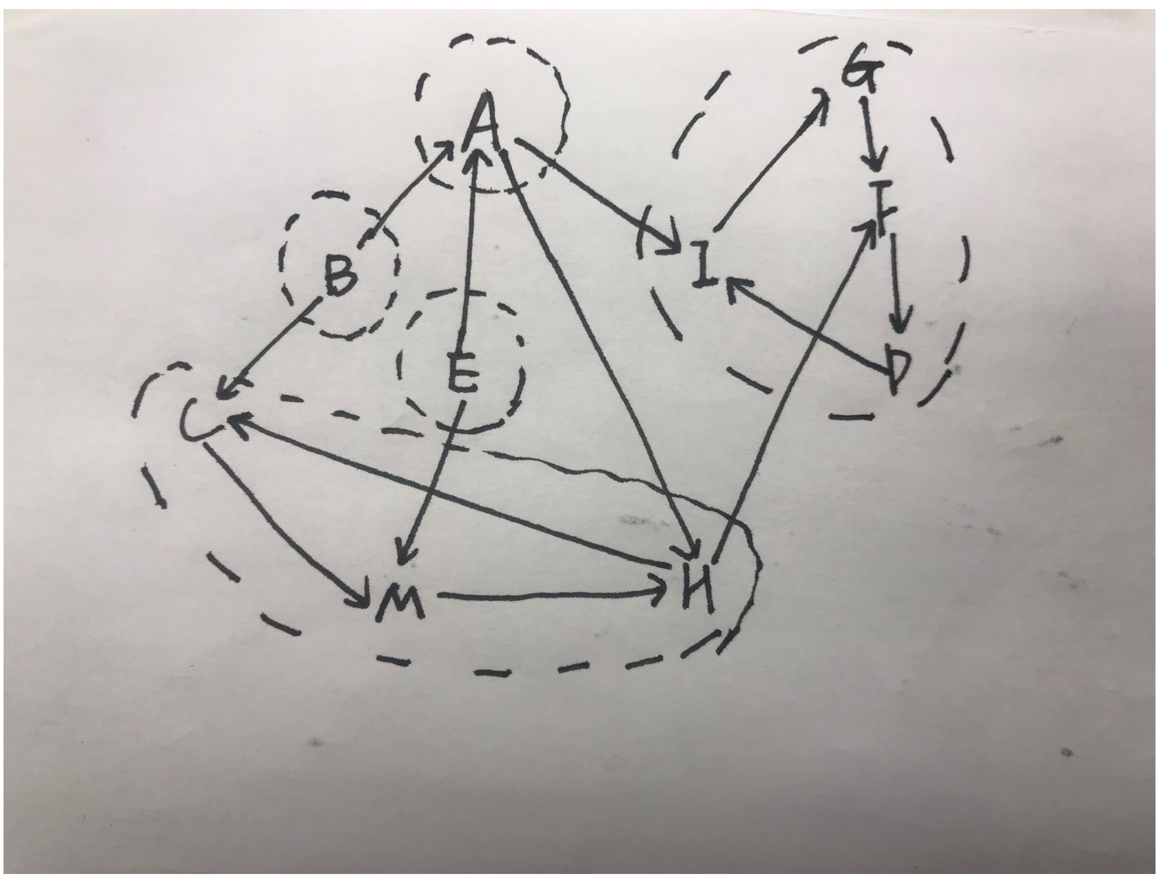
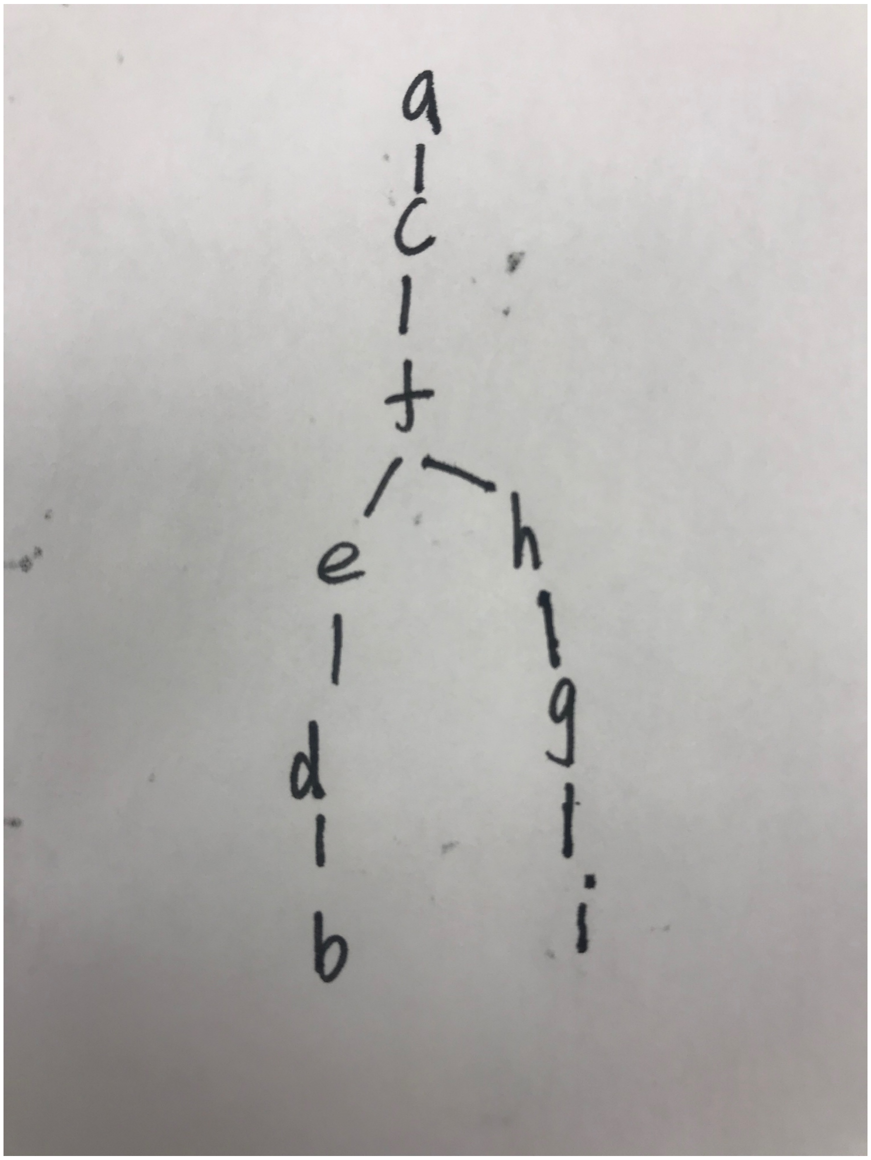
1.

	Discovery time	Finishing time
A	1	16
B	17	18
C	11	14
D	4	9
E	19	20
F	3	10
G	6	7
H	2	15
I	5	8
M	12	13

There are five strongly connected components:{E},{B},{A},{H,M,C},{F,G,I,D}



2.The edges which are added into A (c,f),(g,h),(a,c),(b,d),(e,d),(f,h),(e,f),(g,i).



3.

Analysis:

KRUSKAL(G,w)
A = ∅
for each vertex $v \in G.V(v)$
 MAKE-SET (v)

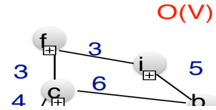
sort the edges of G.E into nondecreasing order by weight w
for each (u,v) taken from the sorted list
 if FIND-SET(u) ≠ FIND-SET(v)
  A = A ∪ {(u,v)}
  UNION(u, v)
return A

Initialize X: O(1)

First **for** loop: IVI MAKE-SET

Sort E: O(E lg E)

Second **for** loop:
O(E) FIND-SET and
O(V) UNION



Because the running time of MAKE_SET is O(log|V|), FIND_SET is O(log|V|), UNION is O(log|V|),

first for loop:O(|V|log|V|)

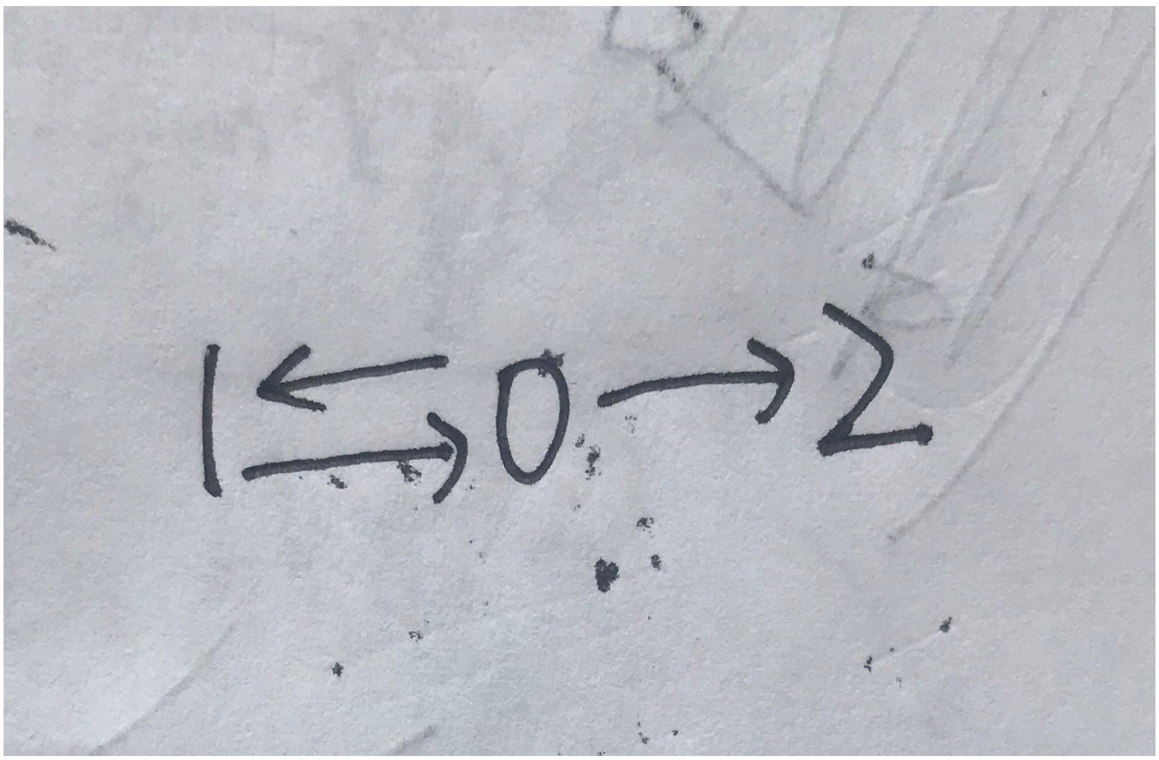
sort E: O(|E|log|E|)

second for loop: O(E log|V|)+O(|V|log|V|)

Therefore, the running time of the algorithm is O(|V|log|V|+|E|log|E|)

4.The simpler algorithm does not always produce correct results.

Consider a counter-example like:



The first DFS beginning with 0 will result in the following finishing time order:1,2,0, the second execution of DFS will show all vertices are in the same strongly connected component because 1 is reachable to both 0 and 2, however, this graph definitely has two strongly connect component which are {0,1},{2}. So the simpler algorithm is wrong.

5.For each ship, according to the distance between it and the safe polities for it, we make a min_heap, whose running time is O(n), and the we take the minimum from the heap in O(1).