Frederick Wittman
W08288149

1.

a.

1. n could be small: in this case, low order terms and constants could have a significant impact on time-complexity.
2. There could be a systematic bias in the data set.
3. The application of an algorithm could demand excellent worst case performance without regard to best and average case performance.
4. There could be a factor at the machine level that reduces performance as n grows large i.e. cache performance or space in main memory.

b.

Where c is a constant and n is the number of nodes, and without any other information, it is reasonable to suppose that search in the binary search trees executes in $c * log_2 n$ time. For the tree with 1,000 elements, we have:

$$5 = c * log_2 1000$$

$$c \approx 0.5$$

For the tree with 10,000 elements, we have:

$$t = c * log_2 10,000$$

$$t = 0.5 * log_2 10,000$$

$t \approx 6.64$ seconds

c.

1. The BST search algorithm could have been implemented inefficiently.
2. It could be the case that the tree with 10,000 elements is extremely unbalanced and the tree with 1,000 elements is balanced.
3. The two tests could have taken place on different machines.

Frederick Wittman
W08288149

2.

a. Let A = (V_A , E_A) and B = (V_B , E_B), and n = $|V_A|$, $|V_B|$. Consider arbitrary orderings of V_A and V_B such that:

$V_A = \{ V_{A1} , V_{A2} , V_{A3} , \dots , V_{An} \}$

$V_B = \{ V_{B1} , V_{B2} , V_{B3} , \dots , V_{Bn} \}$

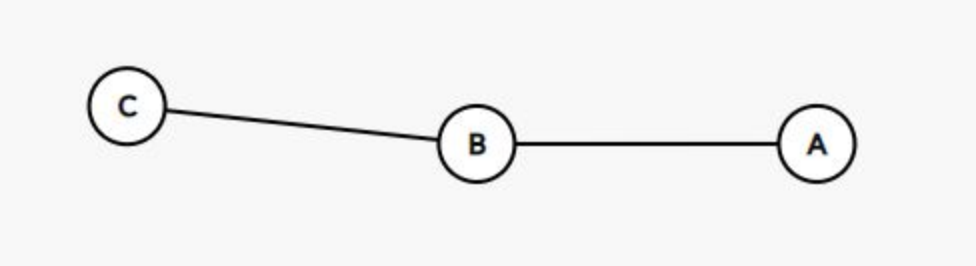
Where k is a whole number in the range [1, n], let f be the bijection given by f (V_{Ak}) = V_{Bk}. Where x and y are whole numbers in the range [1, n], where x =/= y for any particular edge, and because A is completely connected, the edges in E_A are given by (V_{Ax} , V_{Ay}). By similar reasoning, the edges in E_B are given by (V_{Bx} , V_{By}). Observe that | E_B| = | E_A|.

By this reasoning, it is clear that for each edge in E_A of form (V_{Ax} , V_{Ay}) there is exactly one edge in E_B of form (V_{Bx} , V_{By}), and vice versa. That is, (V_{Ax} , V_{Ay}) is in E_A if and only if (V_{Bx} , V_{By}) = (f(V_{Ax}) , f(V_{Ay})) is in E_B. The conclusion is that if A and B have the same number of nodes and are completely connected, they must also be isomorphic.

b.

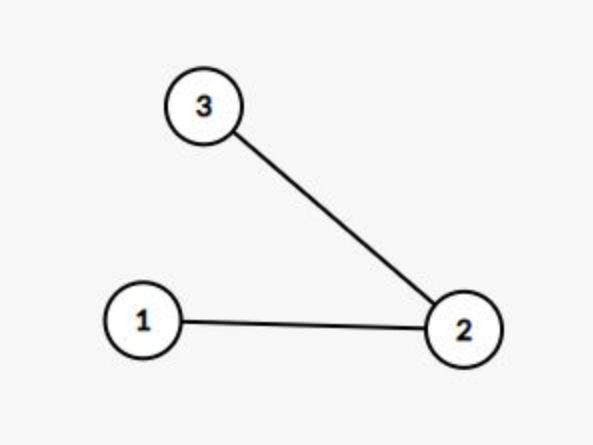
Proof by contradiction

Graph 1



Frederick Wittman
W08288149

Graph 2



Graph 1 and Graph 2 are isomorphic but not completely connected. Therefore, two graphs do not need to be completely connected to be isomorphic.

Proof by contrapositive

What we are trying to prove is the contrapositive of what we proved in 2(a). Therefore, it is also true.

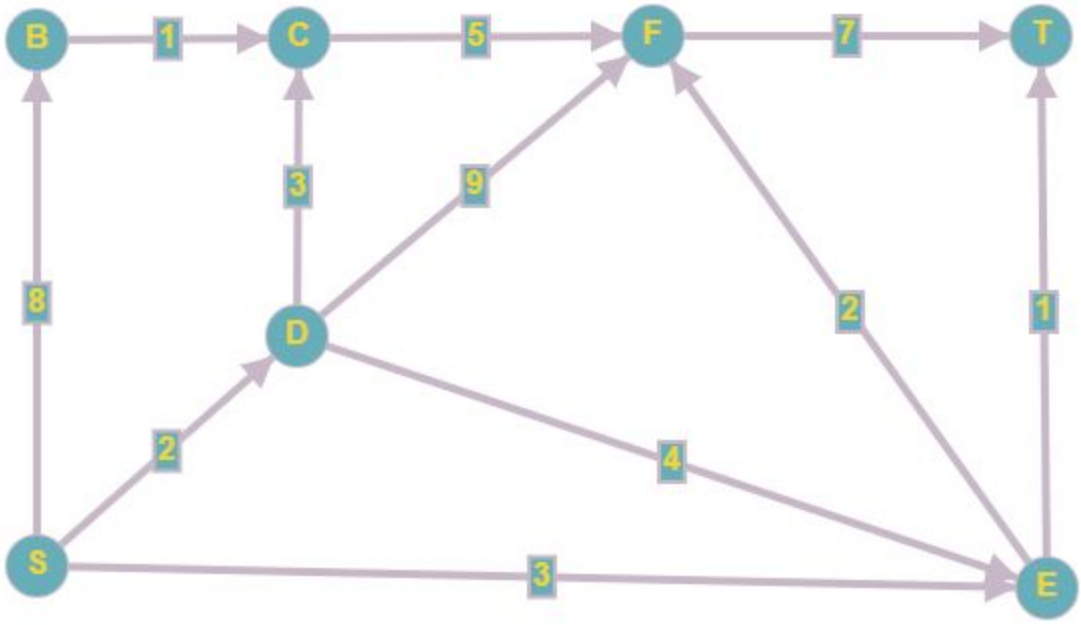
3. Please see .js file.

Frederick Wittman
W08288149

4.

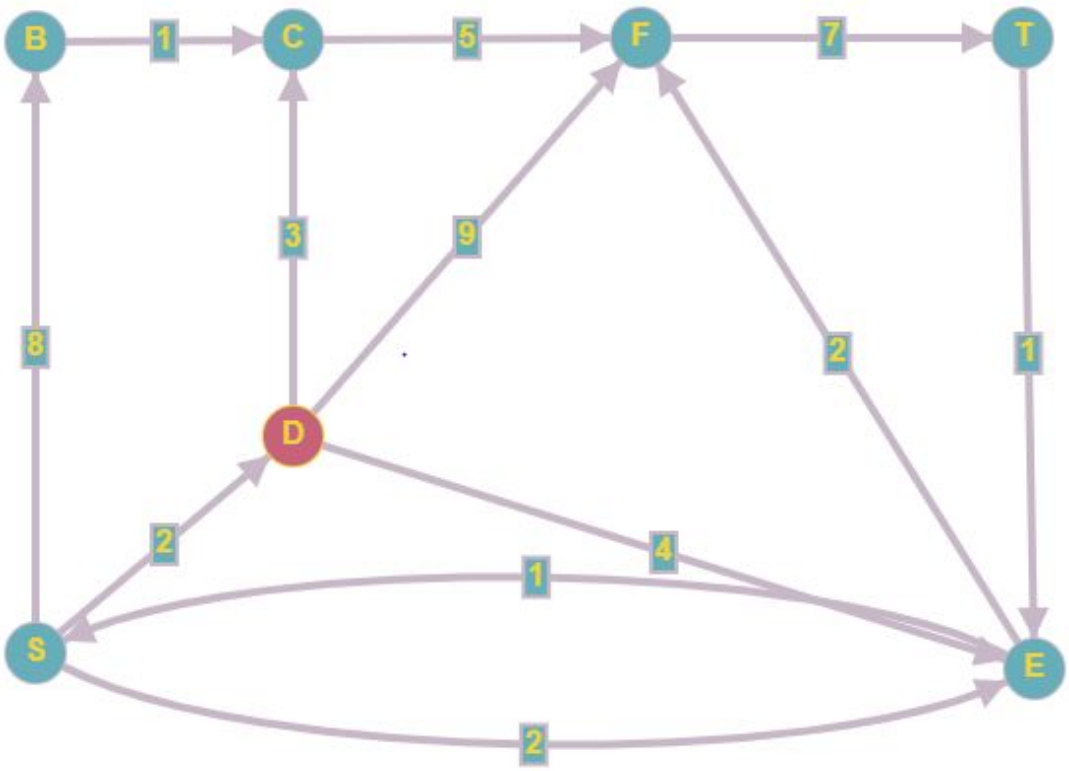
Graphing software:
https://csacademy.com/app/graph_editor/
https://graphonline.ru/en/

Iteration 0



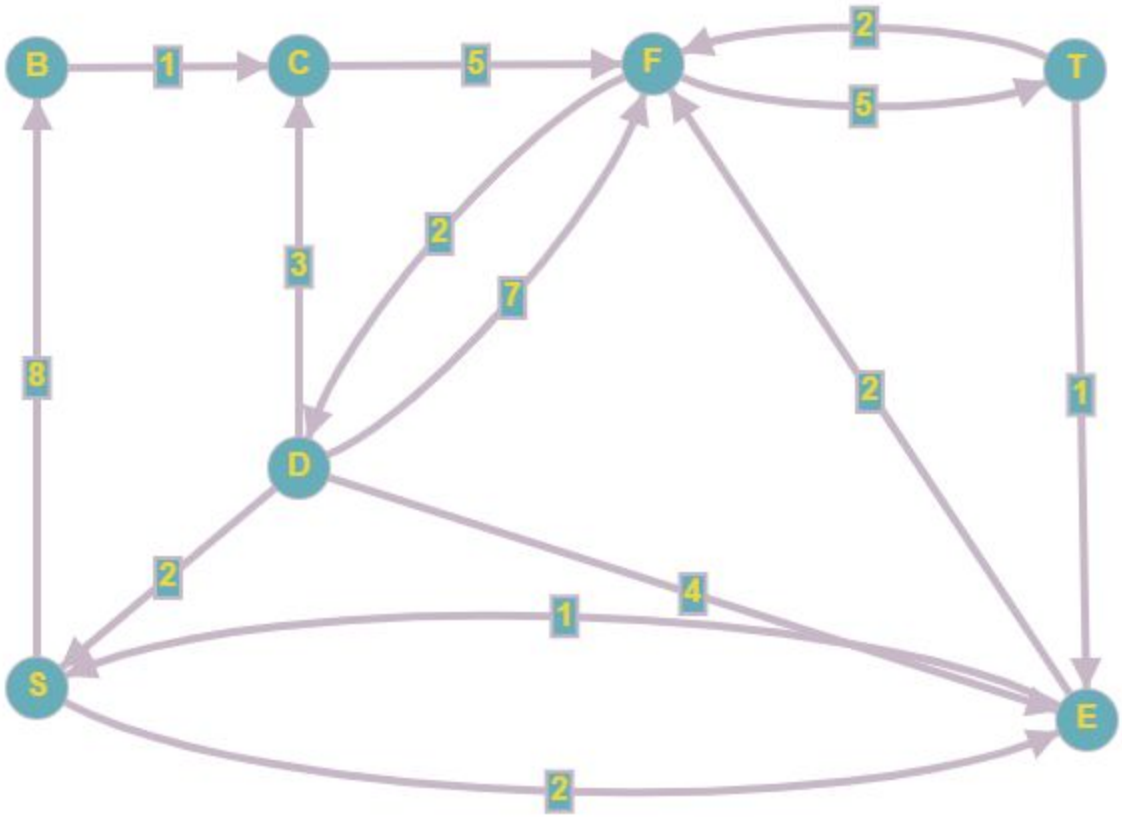
Frederick Wittman
W08288149

Iteration 1



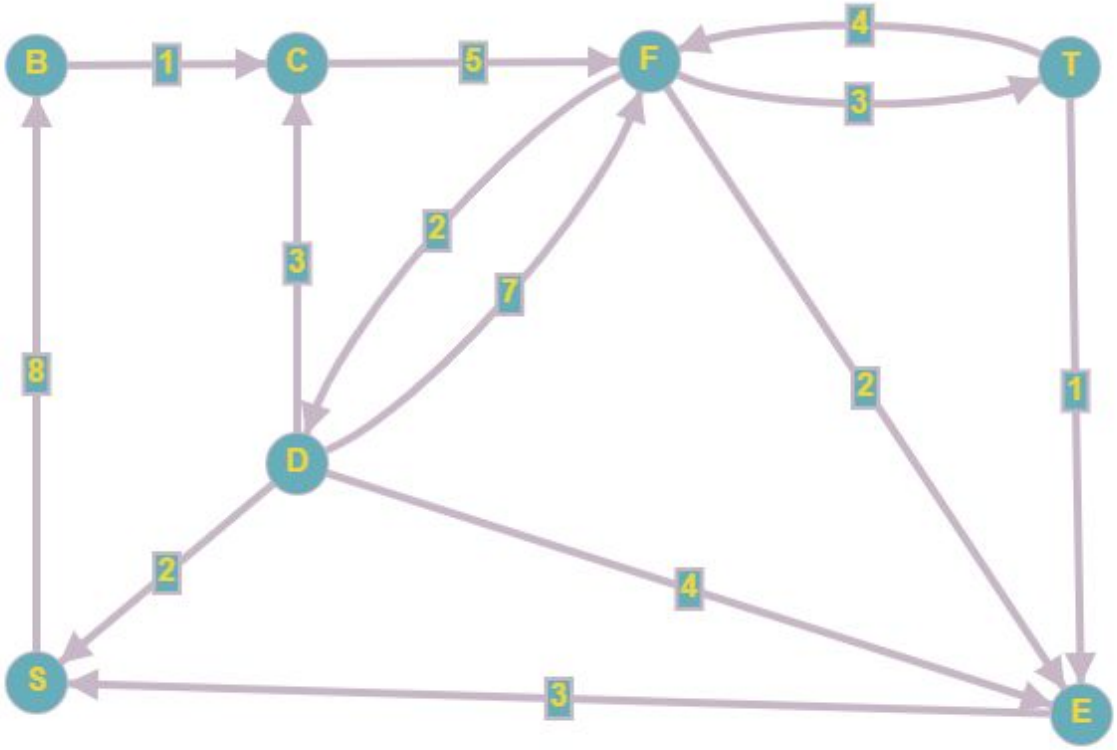
Frederick Wittman
W08288149

Iteration 2



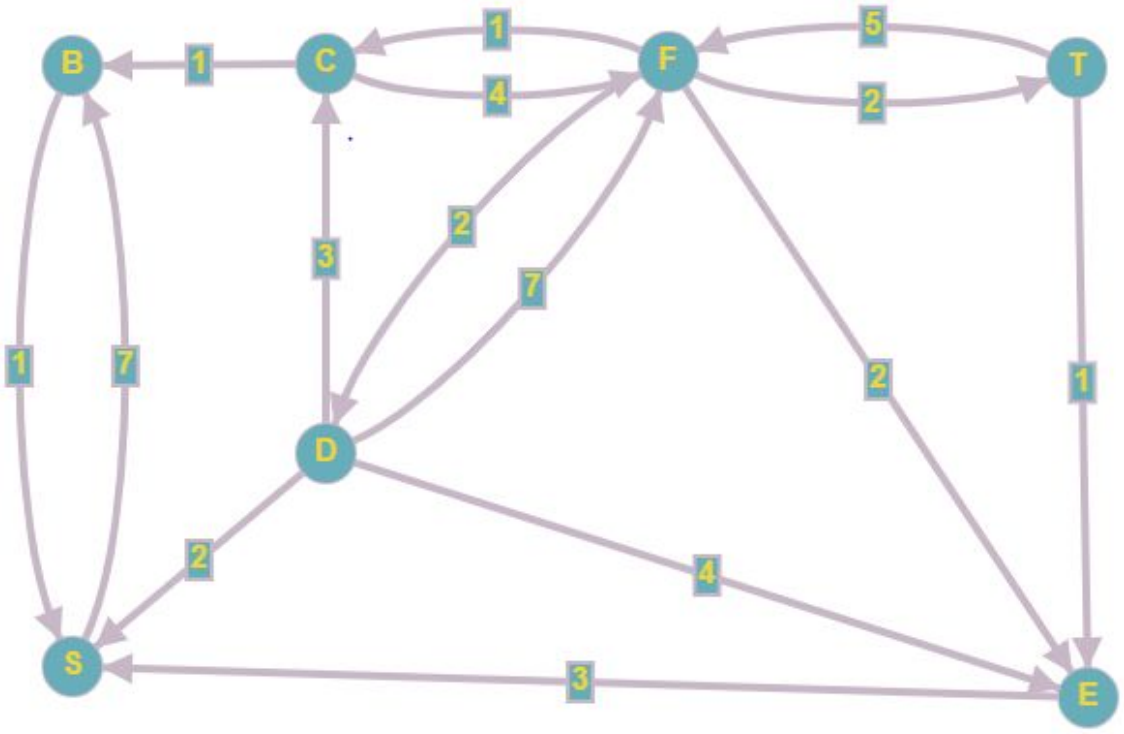
Frederick Wittman
W08288149

Iteration 3



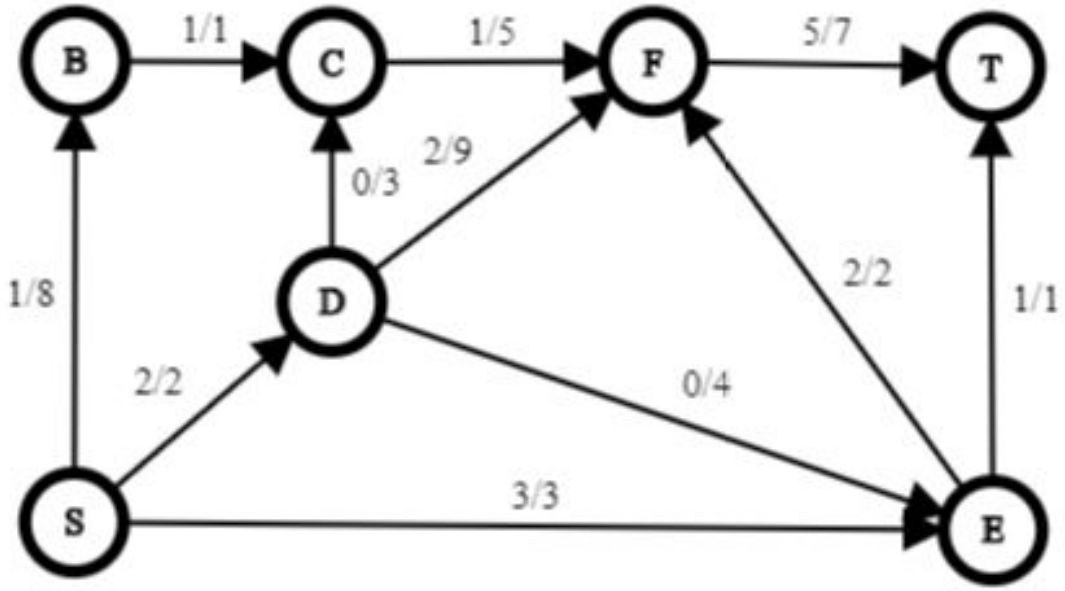
Frederick Wittman
W08288149

Iteration 4



Frederick Wittman
W08288149

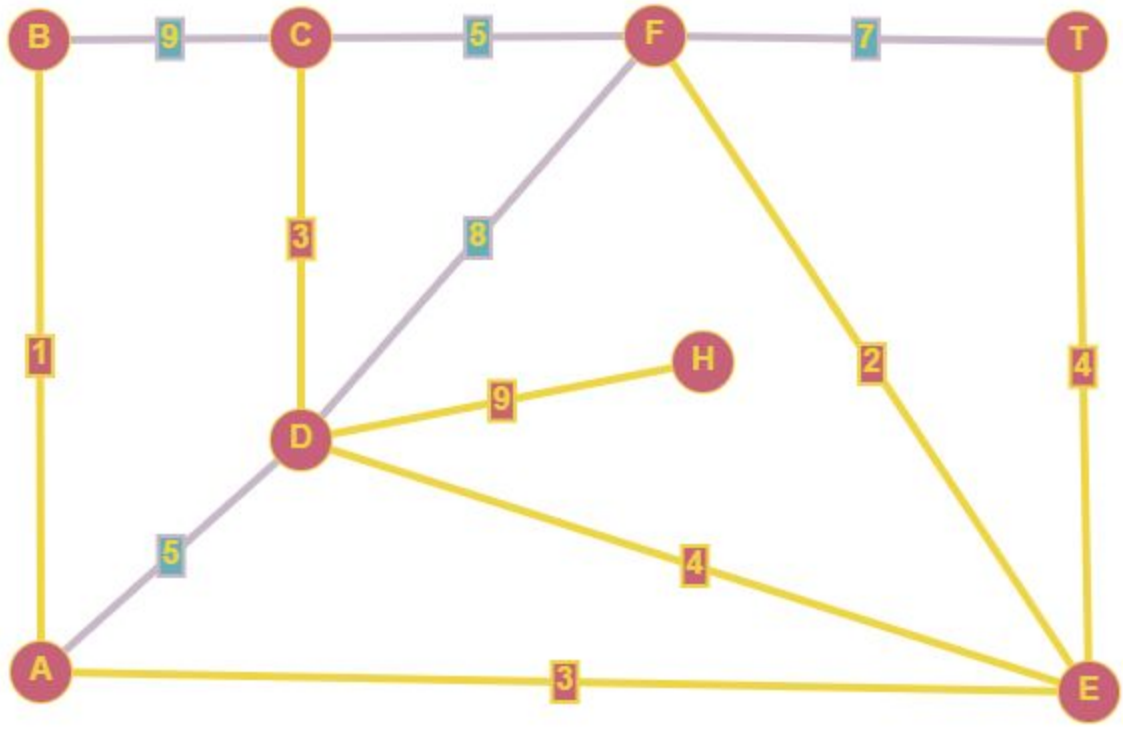
Original Graph



A total of 6 units of flow can be pushed from S to T.

Frederick Wittman
W08288149

5.



The minimum spanning tree is highlighted in yellow. See https://graphonline.ru/en/. The nodes could have been added in the order:

1. A, B
2. E, F
3. C, D
4. edge (A E)
5. edge (E D)
6. T
7. H