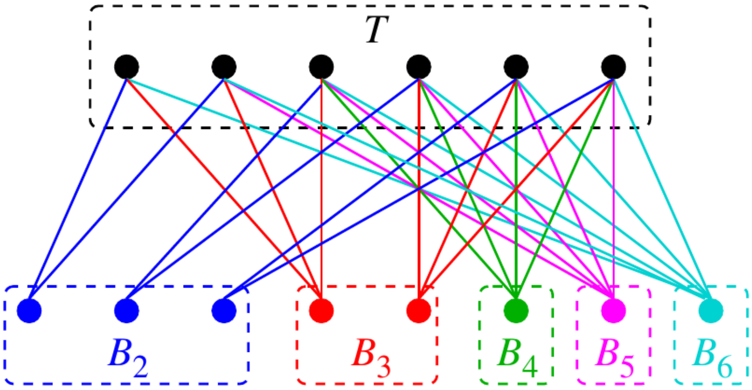
A logarithmic approximation algorithm for MIN-VERTEX-COVER

```
Initialize U = ∅.
while (E is not empty) {
   Find a vertex u ∈ V of largest (remaining) degree.
   Add u to U.
   Delete from E all the (remaining) edges with u as one endpoint.
}
Return U.
```

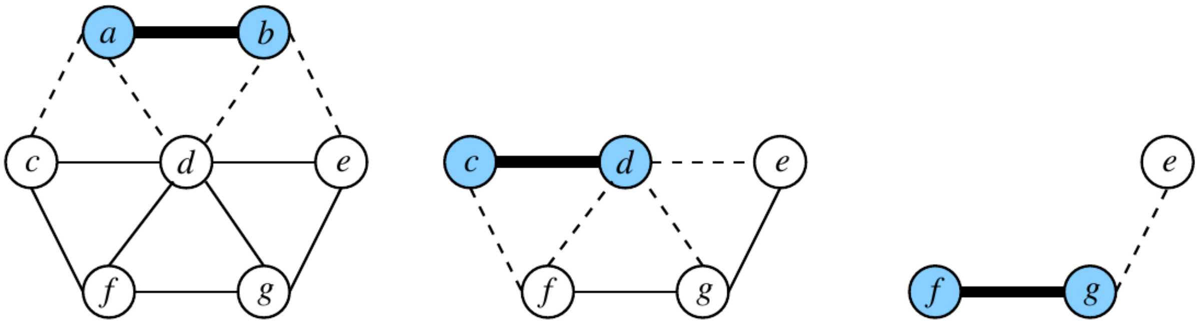
Tightness of the approximation ratio

The logarithmic approximation factor for the greedy vertex cover algorithm is optimal



A 2-approximation algorithm for MIN-VERTEX-COVER

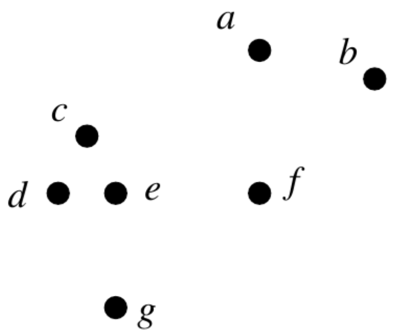
Initialize $U = \emptyset$.
while (E is not empty) {
    Pick any edge $e = (u, v)$ from E.
    Add u and v to U.
    Remove u and v from V.
    Remove from E all edges incident on u or v.
}
Return U.



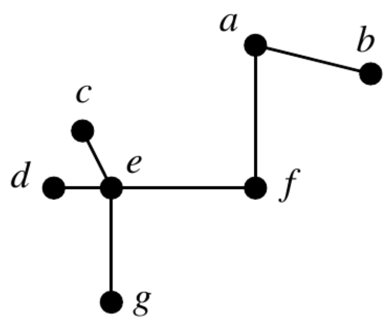
$U = \{a, b\}$ $\quad$ $U = \{a, b, c, d\}$ $\quad$ $U = \{a, b, c, d, f, g\}$

A 2-approximation algorithm for ETSP

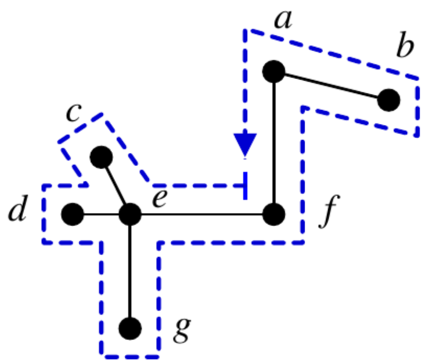
```
Compute a minimum spanning tree T of G under the given cost function.
Choose an arbitrary vertex u0 of T.
Treat T as a tree rooted at u0.
Impose an arbitrary ordering on the children of each node.
Make a pre-order traversal of T (starting at the root u0).
Suppose that the traversal returns the list u0,u1,u2,...,un-1 of visited nodes.
Return the Hamiltonian cycle Z = (u0,u1,u2,...,un-1,u0).
```



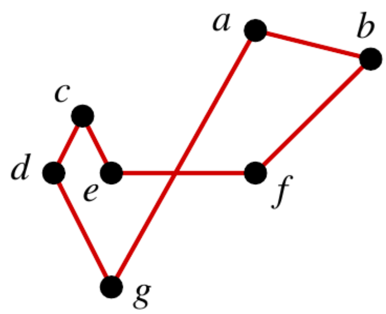
(a) Location of the cities



(b) Computation of an MST



(c) Preorder traversal of MST



(d) The TSP cycle