Algorithms Lab

Connecting Cities

Goal

- find a largest set of vertex disjoint edges

Goal

find a **largest matching**

Goal

find a **largest matching**

BGL: $O(VE) = O(n^2)$.
(hits timelimit on the second test set)

BGL solves the matching problem in general graphs

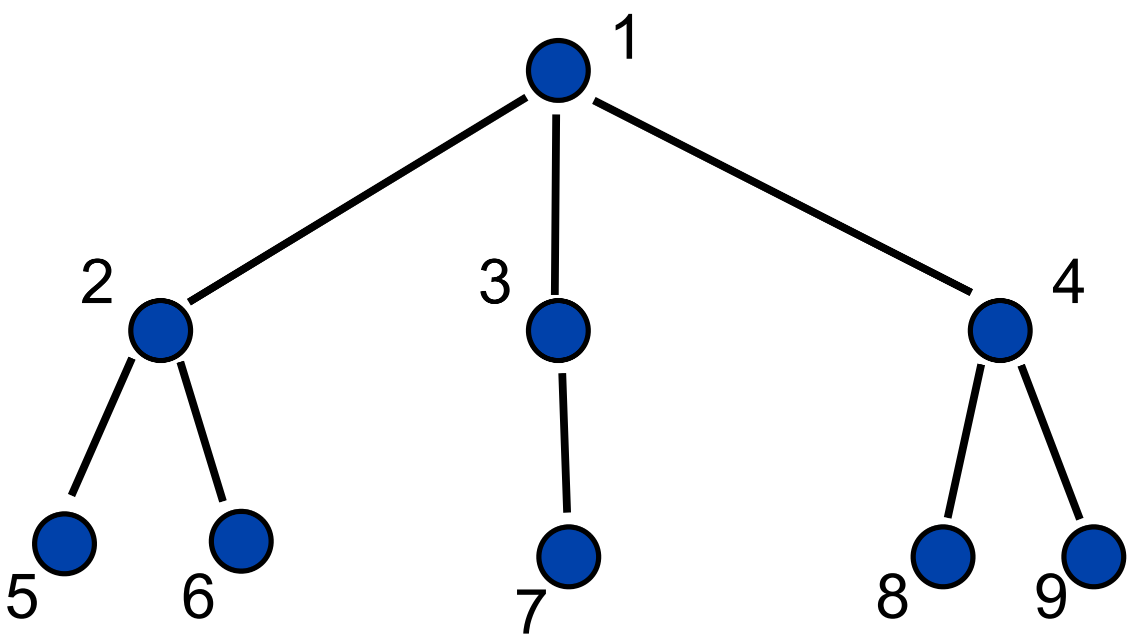
Observation:

- input graph is a **tree**

BGL solves the matching problem in general graphs

Observation:

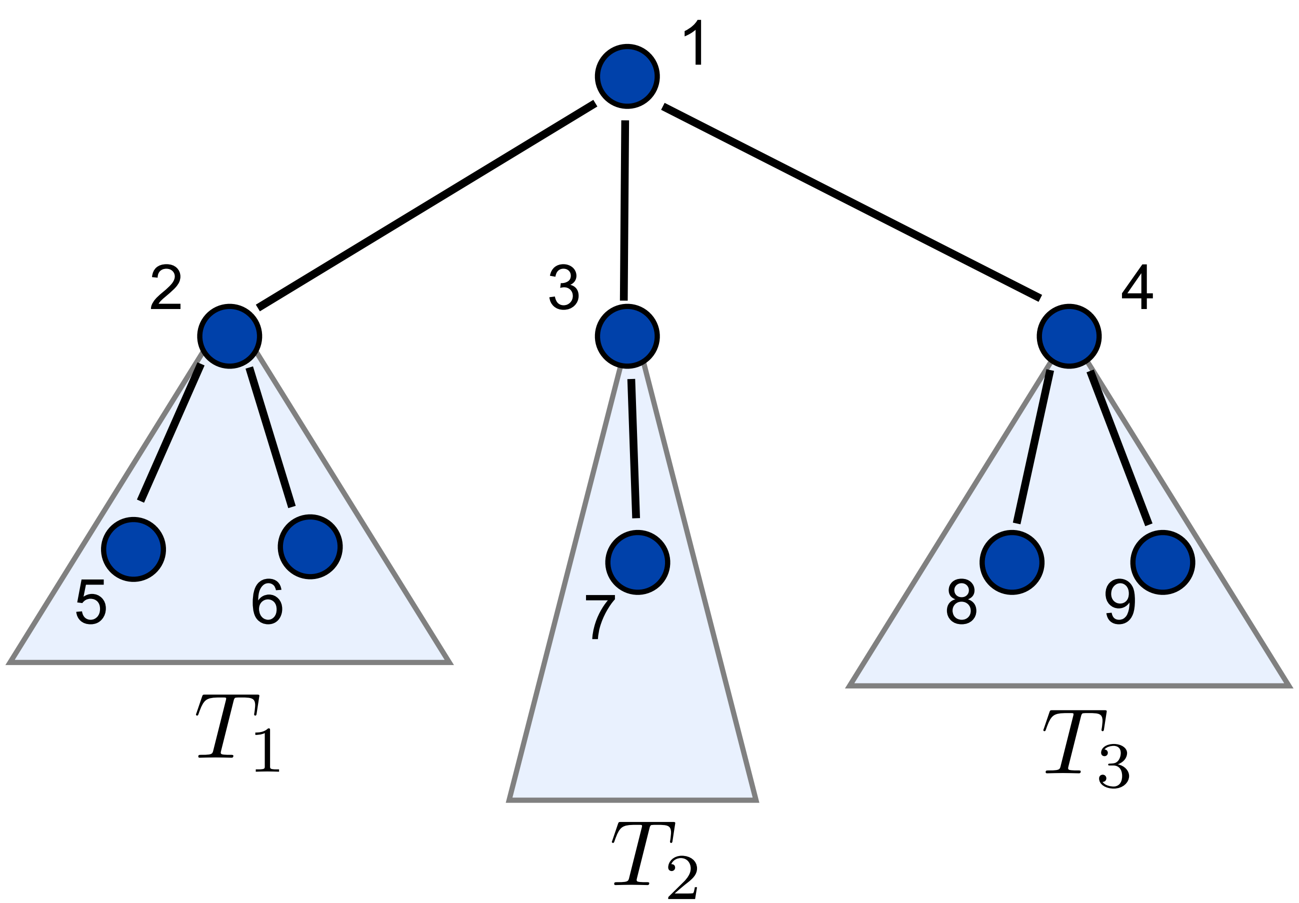
- input graph is a **tree**



BGL solves the matching problem in general graphs

Observation:

- input graph is a **tree**

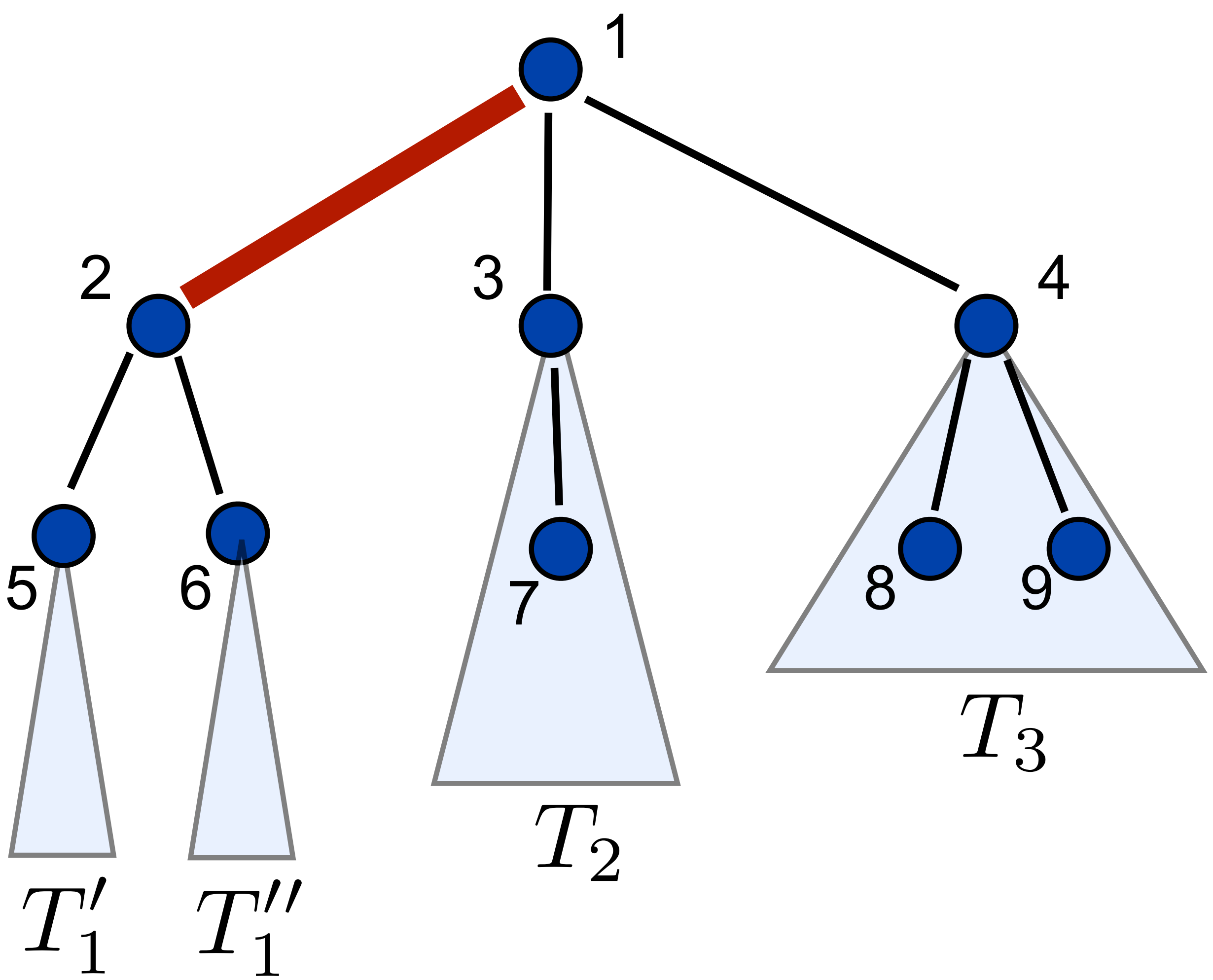


- don't take any edge incident to 1
- no edge between T_1, T_2, T_3
- max-matching in each subtree can be found **independently**!

BGL solves the matching problem in general graphs

Observation:

- input graph is a **tree**



- take edge 1-2
- no edge between

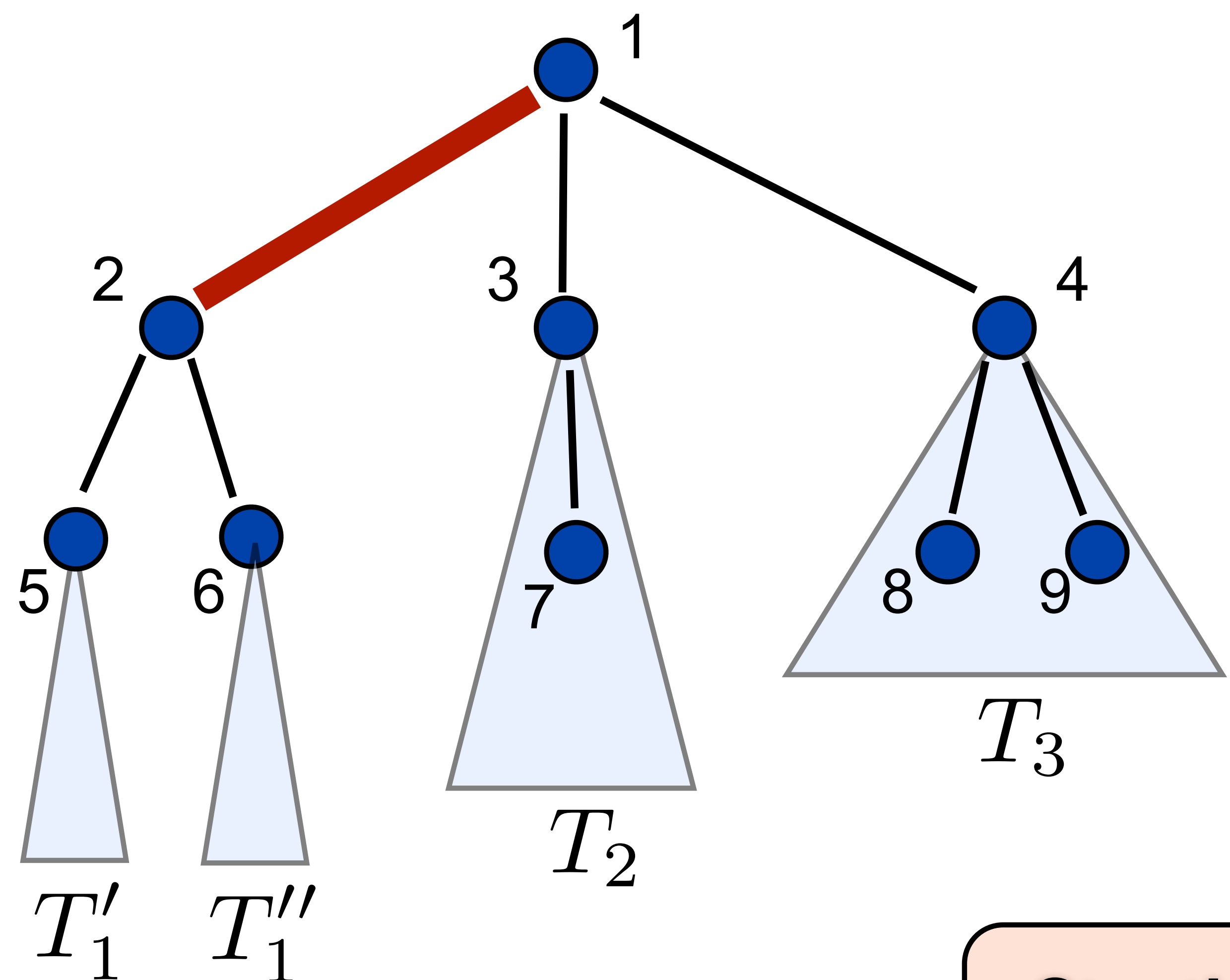
$$T_1', T_1'', T_2, T_3$$

- max-matching in each subtree can be found **independently**!

BGL solves the matching problem in general graphs

Observation:

- input graph is a **tree**



- take edge 1-2
- no edge between

$$T_1', T_1'', T_2, T_3$$

- max-matching in each subtree can be found **independently**!

Similarly for edges 1-3 and 1-4

Generalize previous example as a Dynamic Programming

- Let c(v) be the set of descendants of v in the tree
- Let M(v) be the size of the largest matching in the subtree rooted at v

$$M(v) = \max \begin{cases} \sum_{w \in c(v)} M(w) \\ \max_{a \in c(v)} 1 + \sum_{w \in c(v) \setminus a} M(w) + \sum_{w' \in c(a)} M(w') \end{cases}$$

Implementation details

$$M(v) = \max \begin{cases} \displaystyle\sum_{w \in c(v)} M(w) \\ \displaystyle\max_{a \in c(v)} \; 1 + \sum_{w \in c(v) \setminus a} M(w) + \sum_{w' \in c(a)} M(w') \end{cases}$$

```
matching(v) {
  S = 0;
  for all w in c(v) : S = S + M(w);
  M(v) = S;

  for all a in c(v)
     S = 1;
     for all w in c(v) \ a
       S = S + M(w);
     for all w′ in c(a)
       S = S + M(w′);
     M(v) = max(M(v), S);
}
```

Implementation details

$$M(v) = \max \begin{cases} \sum_{w \in c(v)} M(w) \\ \max_{a \in c(v)} 1 + \sum_{w \in c(v) \setminus a} M(w) + \sum_{w' \in c(a)} M(w') \end{cases}$$

```
matching(v) {
  S = 0;
  for all w in c(v) : S = S + M(w);
  M(v) = S;

  for all a in c(v)
     S = 1;
     for all w in c(v) \ a
       S = S + M(w);
     for all w' in c(a)
       S = S + M(w');
     M(v) = max(M(v), S);
}
```

If v has n-1 descendants - $\mathcal{O}(n^2)$

Implementation details

$$M(v) = \max \begin{cases} \displaystyle\sum_{w \in c(v)} M(w) \\ \displaystyle\max_{a \in c(v)} 1 + \sum_{w \in c(v) \setminus a} M(w) + \sum_{w' \in c(a)} M(w') \end{cases}$$

```
matching(v) {
  S = 0;
  for all w in c(v) : S = S + M(w);
  M(v) = S; M′(v) = S;

  for all a in c(v)
     S = 1 + M′(v) - M(a);
     for all w′ in c(a)
       S = S + M(w′);
     M(v) = max(M(v), S);
}
```

Implementation details

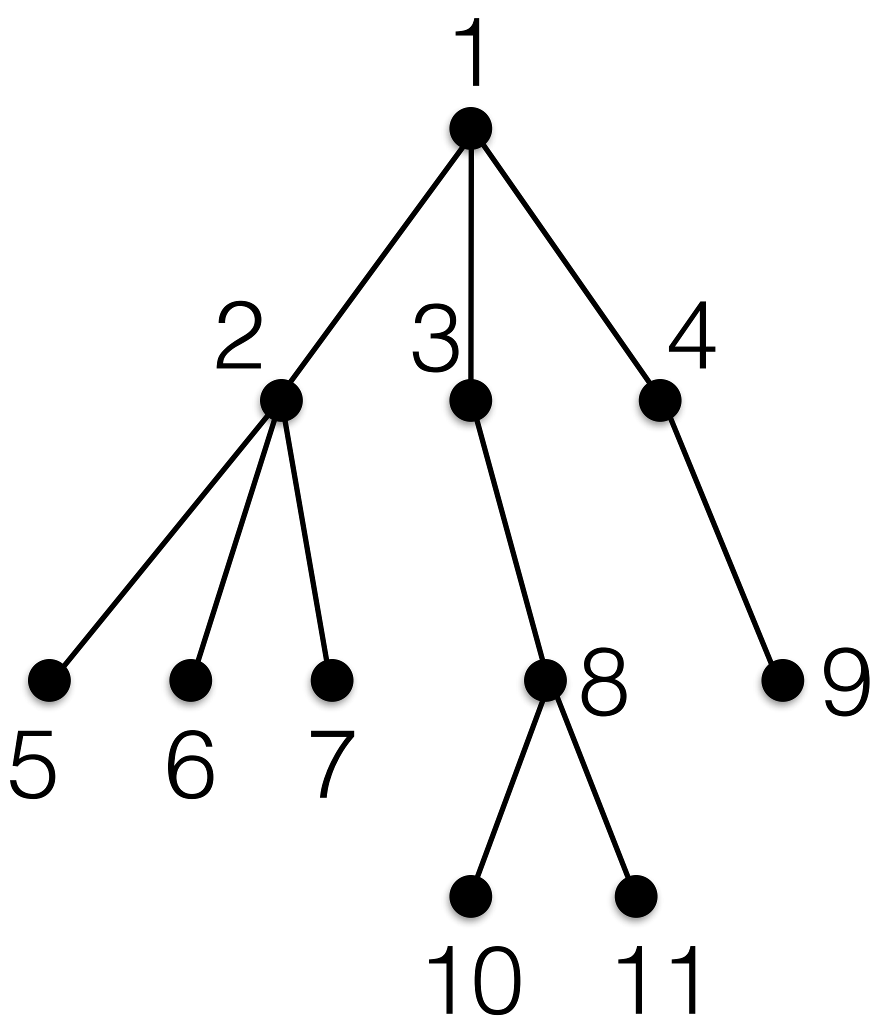
$$M(v) = \max \begin{cases} \displaystyle\sum_{w \in c(v)} M(w) \\ \displaystyle\max_{a \in c(v)} \; 1 + \sum_{w \in c(v) \setminus a} M(w) + \sum_{w' \in c(a)} M(w') \end{cases}$$

```
matching(v) {
  S = 0;
  for all w in c(v) : S = S + M(w);
  M(v) = S; M′(v) = S;

  for all a in c(v)
     S = 1 + M′(v) - M(a);
     for all w′ in c(a)
       S = S + M(w′);
     M(v) = max(M(v), S);
}
```

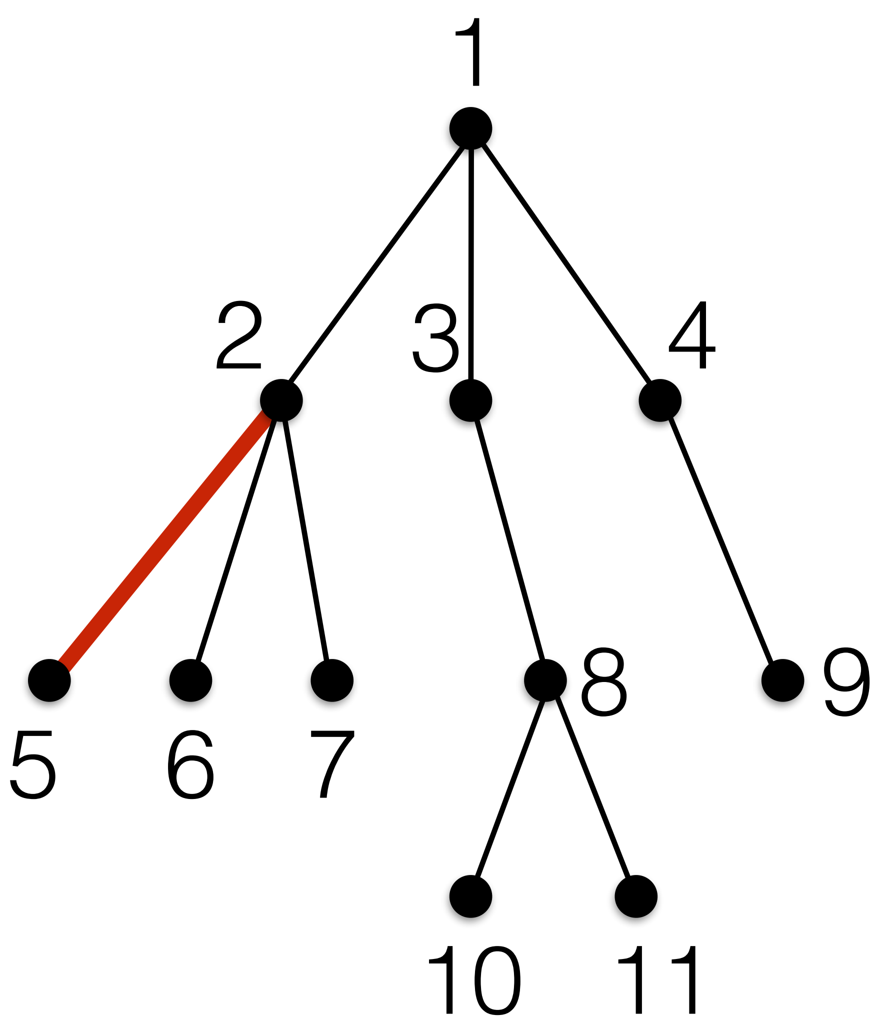
Exercise: show that the running time is linear!

Greedy solution



Repeat: take an edge which contains a leaf and remove its endpoints

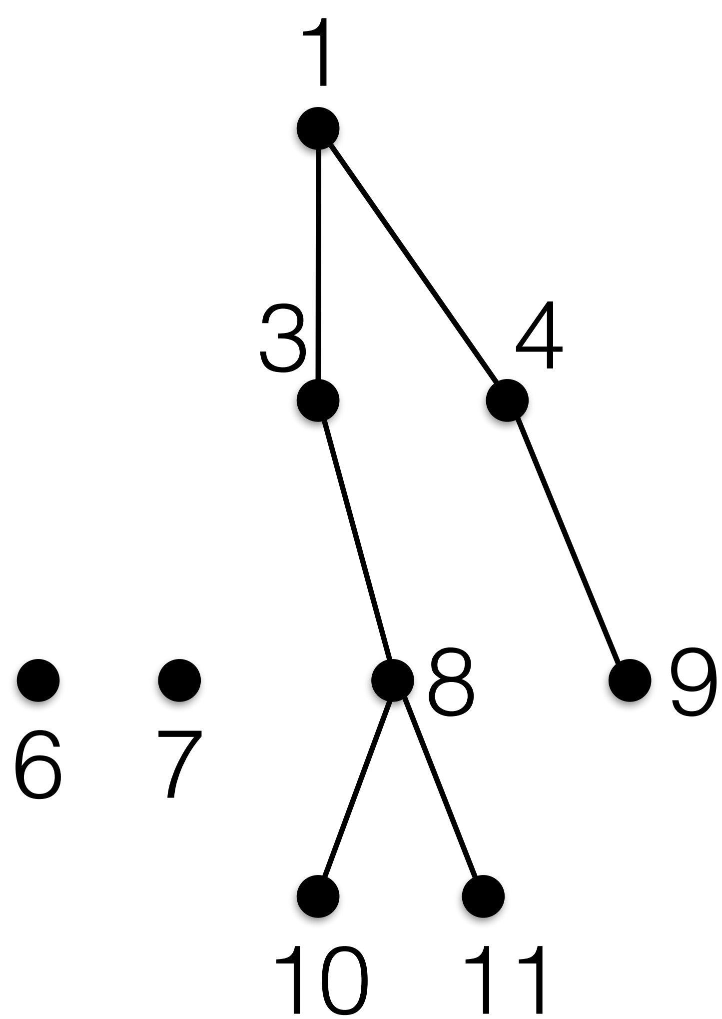
Greedy solution



Repeat: take an edge which contains a leaf and remove its endpoints

{2,5}

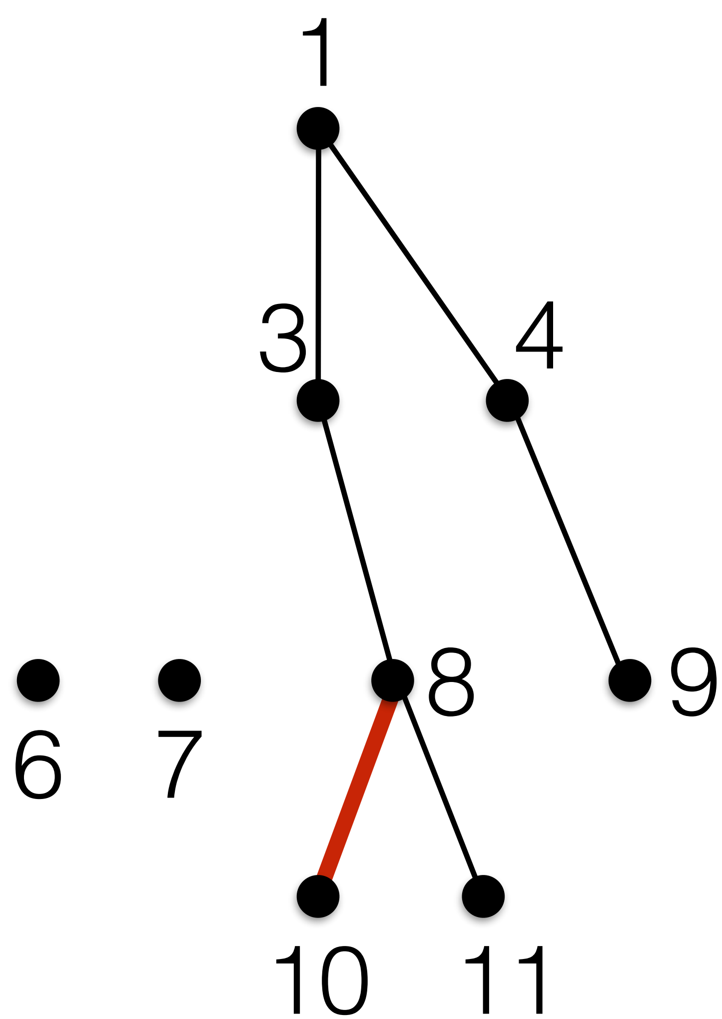
Greedy solution



Repeat: take an edge which contains a leaf and remove its endpoints

{2,5}

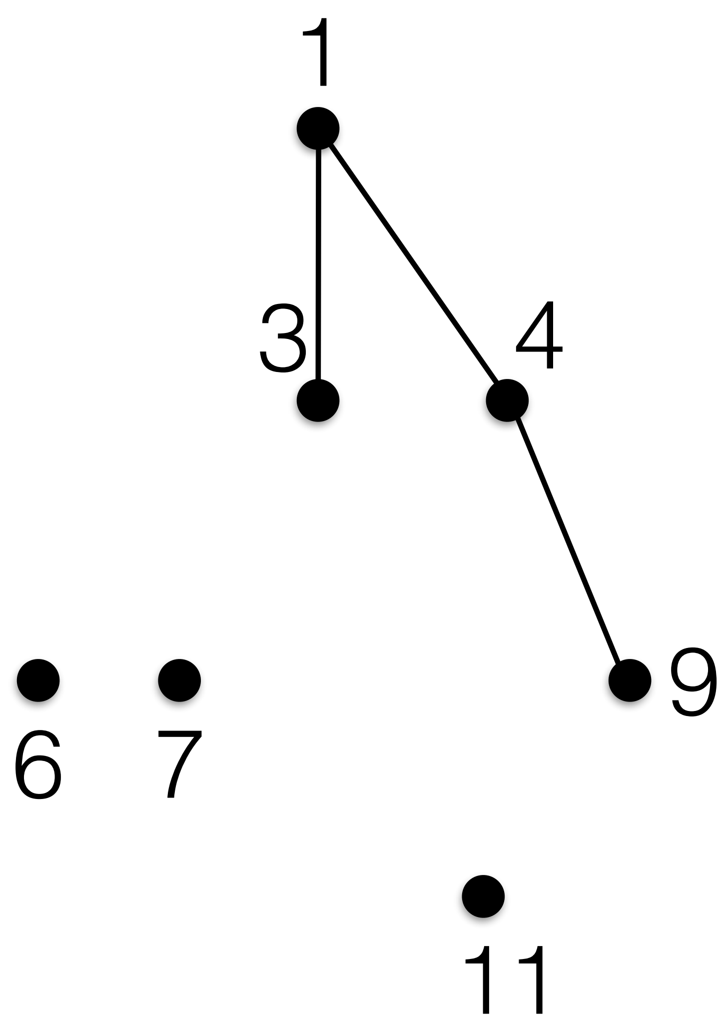
Greedy solution



Repeat: take an edge which contains a leaf and remove its endpoints

{2,5},{10,8}

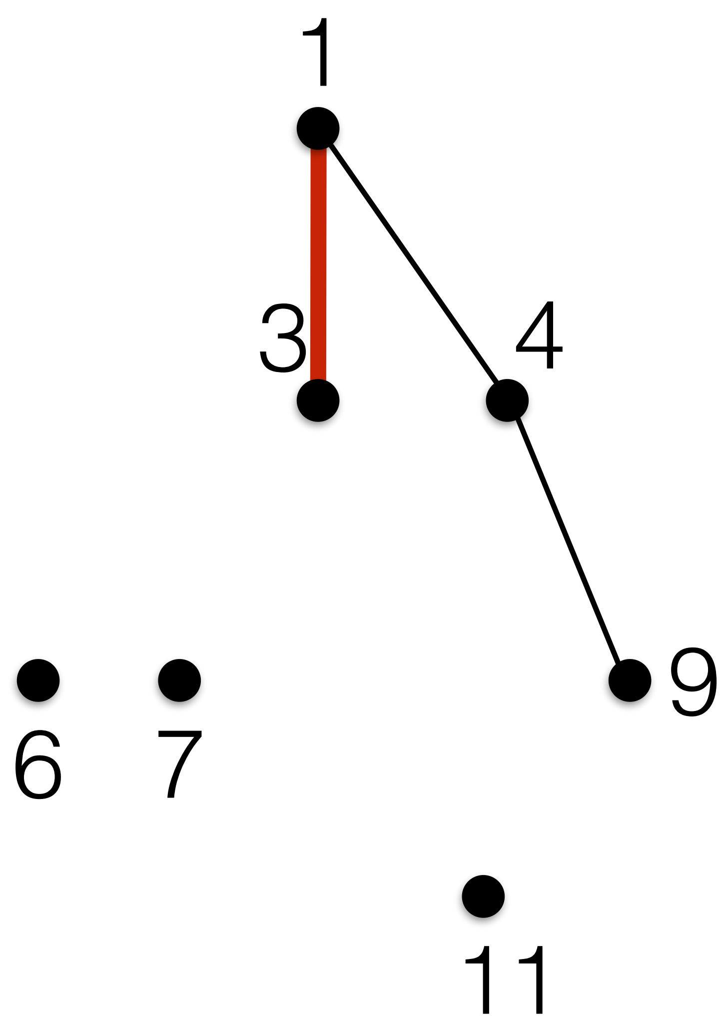
Greedy solution



Repeat: take an edge which contains a leaf and remove its endpoints

{2,5},{10,8}

Greedy solution

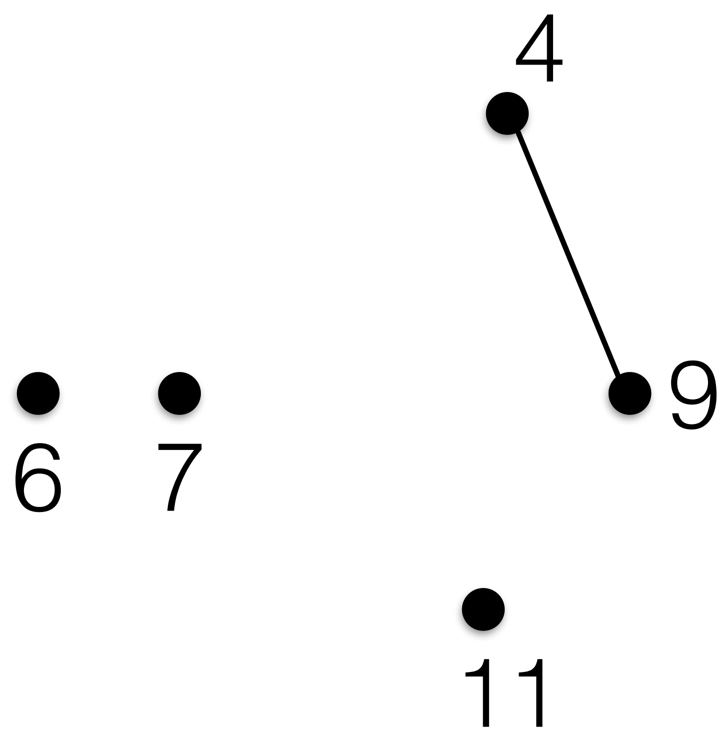


Repeat: take an edge which contains a leaf and remove its endpoints

{2,5},{10,8},{1,3}

Greedy solution

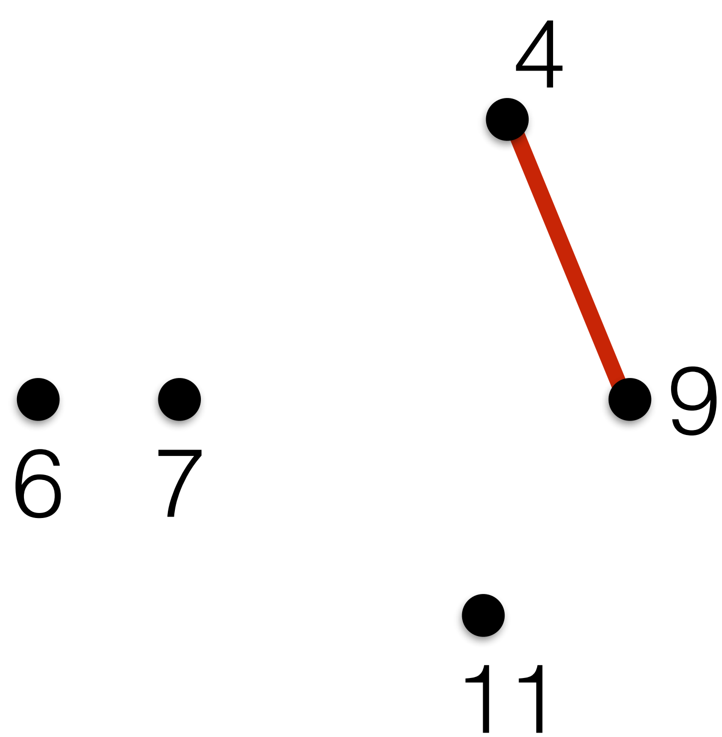
Repeat: take an edge which contains a leaf and remove its endpoints



{2,5},{10,8},{1,3}

Greedy solution

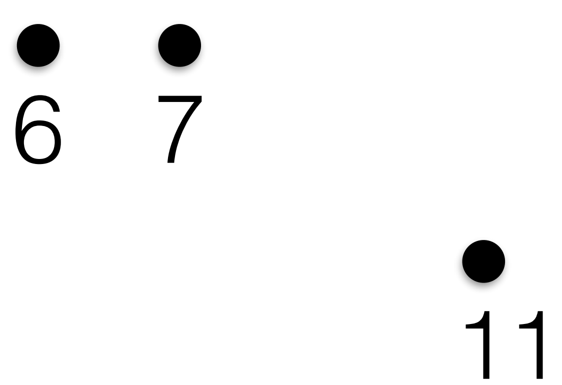
Repeat: take an edge which contains a leaf and remove its endpoints



{2,5},{10,8},{1,3},{4,9}

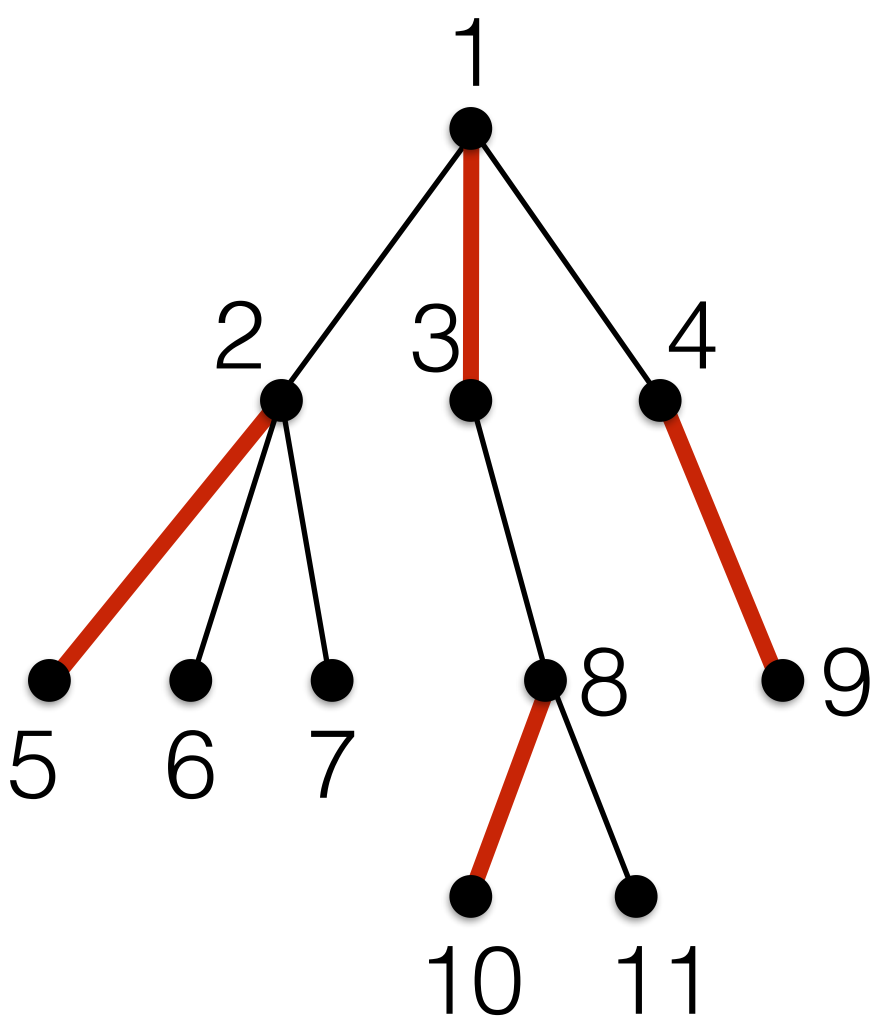
Greedy solution

Repeat: take an edge which contains a leaf and remove its endpoints



{2,5},{10,8},{1,3},{4,9}

Greedy solution



Repeat: take an edge which contains a leaf and remove its endpoints

Correctness can be proven using the exchange argument (see slides from Week 2)

{2,5},{10,8},{1,3},{4,9}