CS3383 Unit 5.2: Travelling Salesperson Problem

David Bremner

March 31, 2018

Outline

Combinatorial Search

- Travelling Salesperson problem
- Dynamic Programming for TSP
- Branch and Bound

Contents

Combinatorial Search
- Travelling Salesperson problem
- Dynamic Programming for TSP
- Branch and Bound

Travelling Salesperson Problem

TSP

Given $G = (V, E)$

Find a shortest tour that visits all nodes.

Brute Force

- $n!$ different tours

Travelling Salesperson Problem

TSP

Given $G = (V, E)$

Find a shortest tour that visits all nodes.

Brute Force

- $n!$ different tours
- Each one takes $\Theta(n)$ time to test

Travelling Salesperson Problem

TSP

Given $G = (V, E)$

Find a shortest tour that visits all nodes.

Brute Force

- $n!$ different tours
- Each one takes $\Theta(n)$ time to test
- Using Stirling's approximation for $n!$

$$n \cdot n! \in \Theta(n^{n+\frac{3}{2}} e^{-n})$$

Contents

Combinatorial Search

- Travelling Salesperson problem
- Dynamic Programming for TSP
- Branch and Bound

Subproblems for Dynamic Programming

$C(S, j)$ length of shortest path starting at 1, visiting all nodes in S and ending at j.

Recurrence

$$C(S, j) = \min_{i \in S \setminus \{ j \}} C(S \setminus \{ j \}, i) + d_{ij}$$

Dynamic Programming for TSP

$C[\{1\}, 1] \leftarrow 0$
for s = 2 to n **do**
  for $\forall$ subsets S of size s **do**
    $C[S, 1] \leftarrow \infty$
    for $j \in S \setminus \{1\}$ **do**
      $C[S, j] \leftarrow \min_{i \in S \setminus \{j\}} C[S \setminus \{j\}, i] + d_{ij}$
    end
  end
end
return $\min_j C[V, j] + d_{j1}$

Analysis

- $n \cdot 2^n$ subproblems

Analysis

- $n \cdot 2^n$ subproblems
- Each takes linear time

Analysis

- $n \cdot 2^n$ subproblems
- Each takes linear time
- Total $O(n^2 2^n)$

Analysis

- $n \cdot 2^n$ subproblems
- Each takes linear time
- Total $O(n^2 2^n)$
- Comparison with brute force (board)

Contents

Combinatorial Search
Travelling Salesperson problem
Dynamic Programming for TSP
Branch and Bound

Branch and Bound

- In general dynamic programming is too slow (not surprising since it's exact)

Branch and Bound

- In general dynamic programming is too slow (not surprising since it's exact)
- In practice people use an enhanced backtracking method called branch and bound.

Branch and Bound

- In general dynamic programming is too slow (not surprising since it's exact)
- In practice people use an enhanced backtracking method called branch and bound.

lower bounds

- Suppose we are minimizing some function $f(\cdot)$.
- We need some function lowerbound such that
  - lowerbound$(P_i) \leq f(P_i)$ for all subproblems P_i
  - lowerbound is faster to compute than f

Branch and Bound in General

```
def BranchAndBound(P_0):
    S ← { P_0 }
    best ← ∞
    while S ≠ ∅:
        (P, S) ← pop(S)
        for P_i ∈ expand(P):
            if test(P_i) = SUCCESS:
                best ← min(best, f(P_i))
            elif lowerbound(P_i) < best:
                S ← S ∪ { P_i }
    return best
```

Subproblems for B&B TSP

$[a, S, b]$ path from a to b passing through S

completed by cheapest path from b to a using $V \setminus S$.

P_0 $[a, \{ a \}, a]$

Subproblems for B&B TSP

$[a, S, b]$ path from a to b passing through S

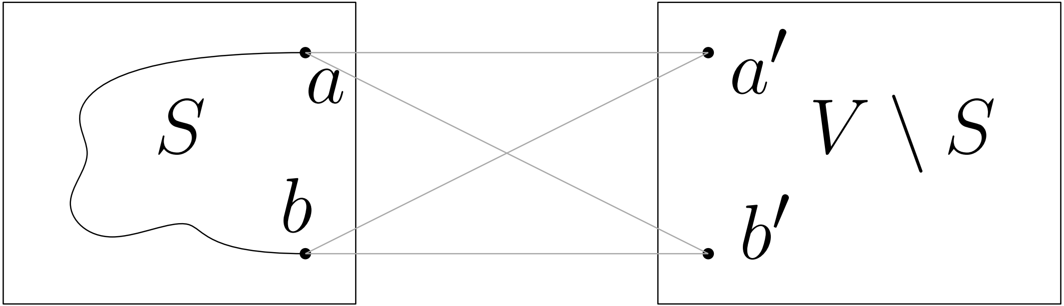
completed by cheapest path from b to a using $V \setminus S$.

P_0 $[a, \{ a \}, a]$

Expand

$$\text{expand}([a, S, b]) = \{ [a, S \cup \{ x \}, x] \mid x \in V \setminus S \}$$

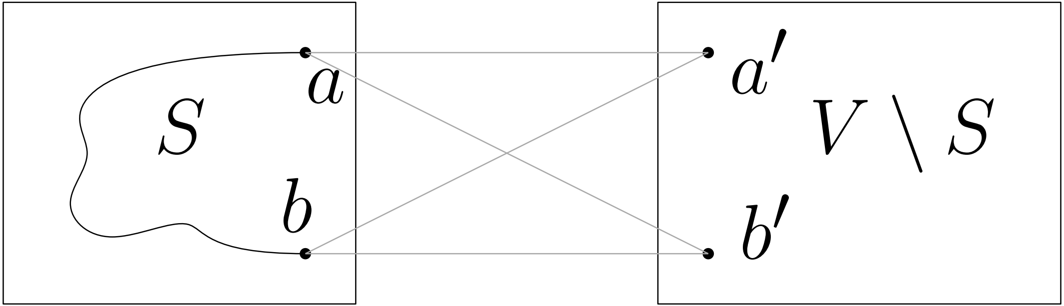
Lower bounds from MST



We need to connect

- a to some $a' \in V \setminus S$

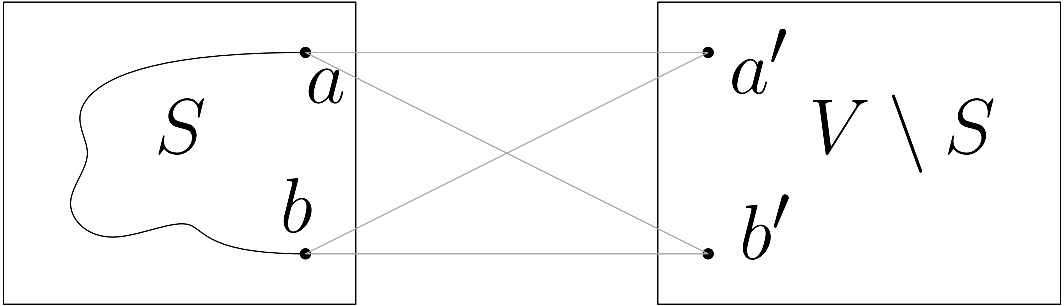
Lower bounds from MST



We need to connect

- a to some $a' \in V \setminus S$
- b to some $b' \in V \setminus S$

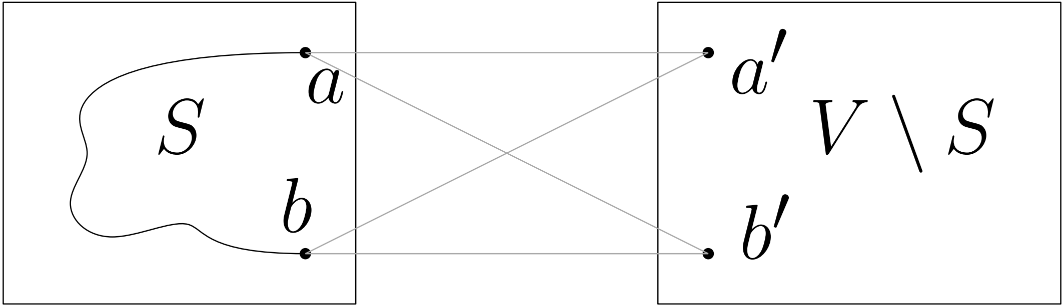
Lower bounds from MST



We need to connect

- a to some $a' \in V \setminus S$
- b to some $b' \in V \setminus S$
- a' to b' using all nodes of $V \setminus S$.

Lower bounds from MST



We need to connect

- a to some $a' \in V \setminus S$
- b to some $b' \in V \setminus S$
- a' to b' using all nodes of $V \setminus S$.
- The last one is a (very special) spanning tree of $V \setminus S$.