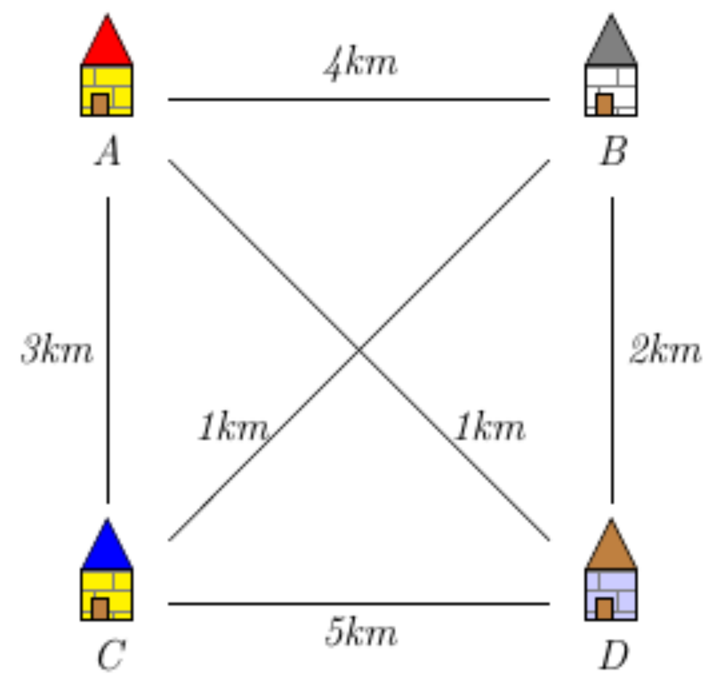
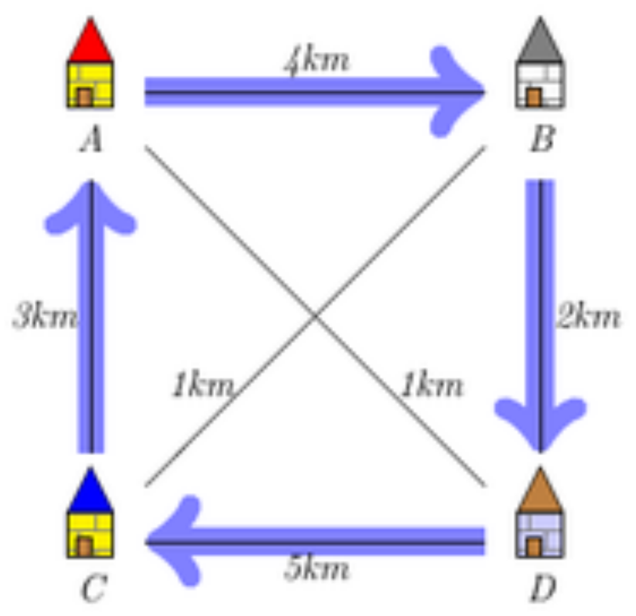
NP-PROBLEM RESEARCH BY ABDOULKADER HAIDARA

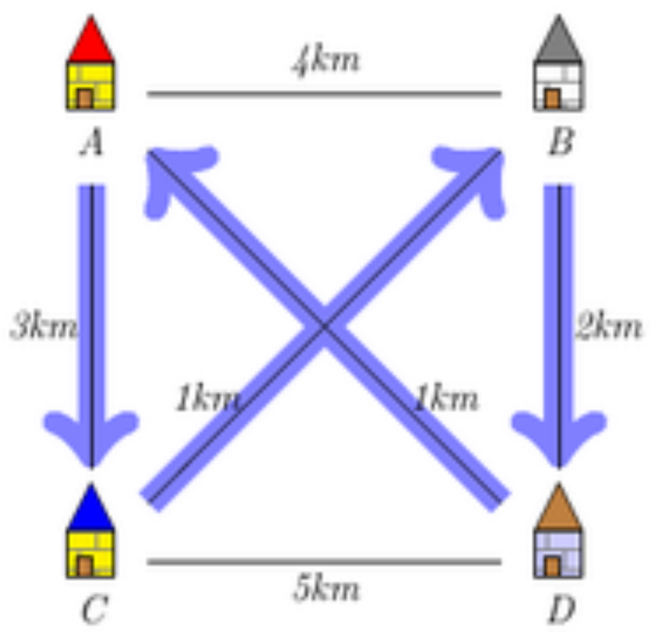
TRAVELLING SALESMAN PROBLEM



problem



not optimal path



optimal path

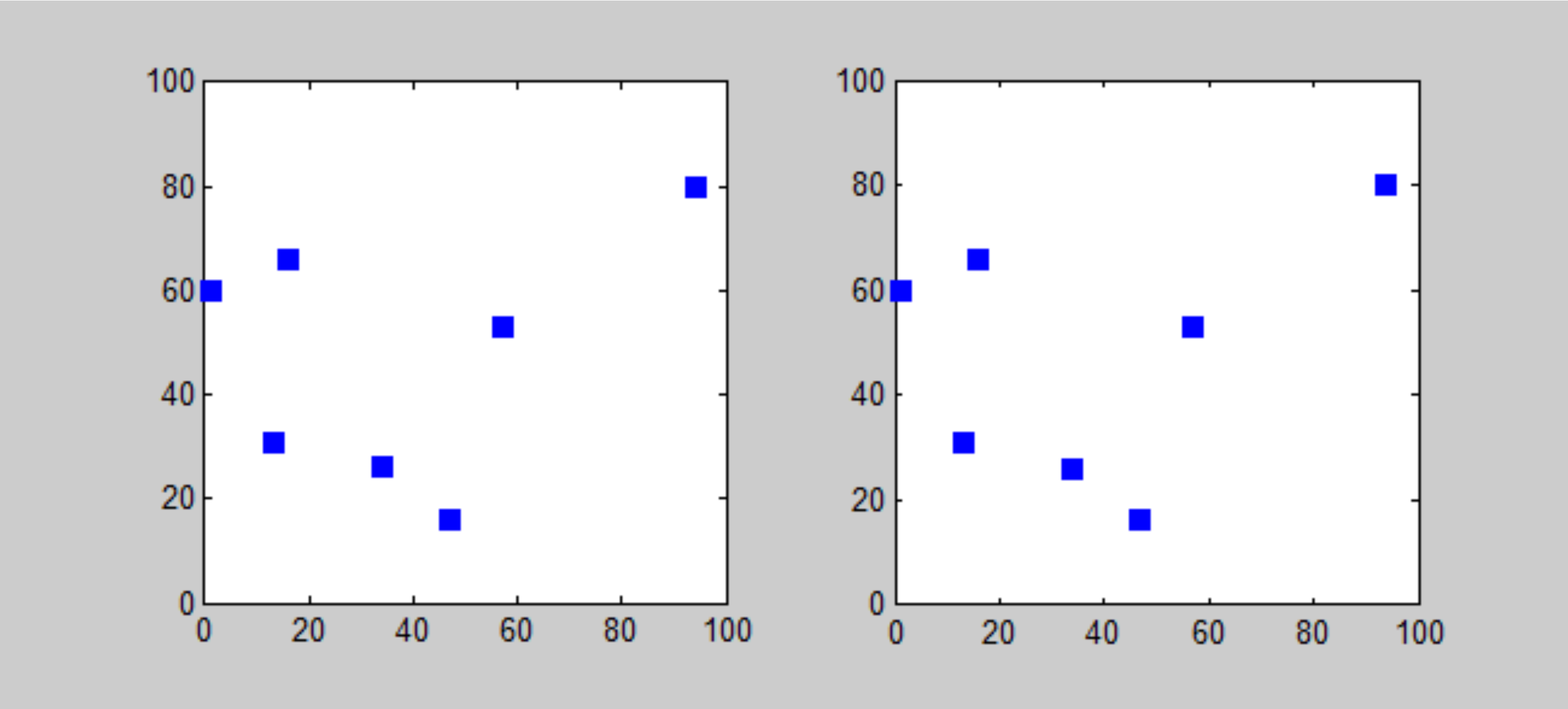
Complexity

HAM-CYCLE <= TSP

The problem is NP-complete

- **Applications**
  - School bus route
  - In logistics (deliveries, post office etc…)
  - Biology (genome sequencing for example)

BRUTE FORCE SOLUTION



7 cities using brute force

Complexities

Time: O(n!)

Space: O(n)

```
This solution becomes unfeasible for even 20 cities.
If a computing a path takes 10^-6second,
for 25 cities the running time exceeds the age of universe(14 billion years).
```

DYNAMIC PROGRAMMING SOLUTION(HELD-KARP ALGORITHM)

$1, 2, \ldots, n$ cities

begins by calculating, for each set of cities $S \subseteq \{2, \ldots, n\}$ and every city $e \neq 1$ not contained in S

the shortest one-way path from to that passes through every city in in some order.

Example

$g(\emptyset, e)$ = d(1, e) $g(\{2\}, 3)$ is the length of $1 \to 2 \to 3$

Once S contains three or more cities, the number of paths through S rises quickly, but only a few such paths need to be examined to find the shortest.

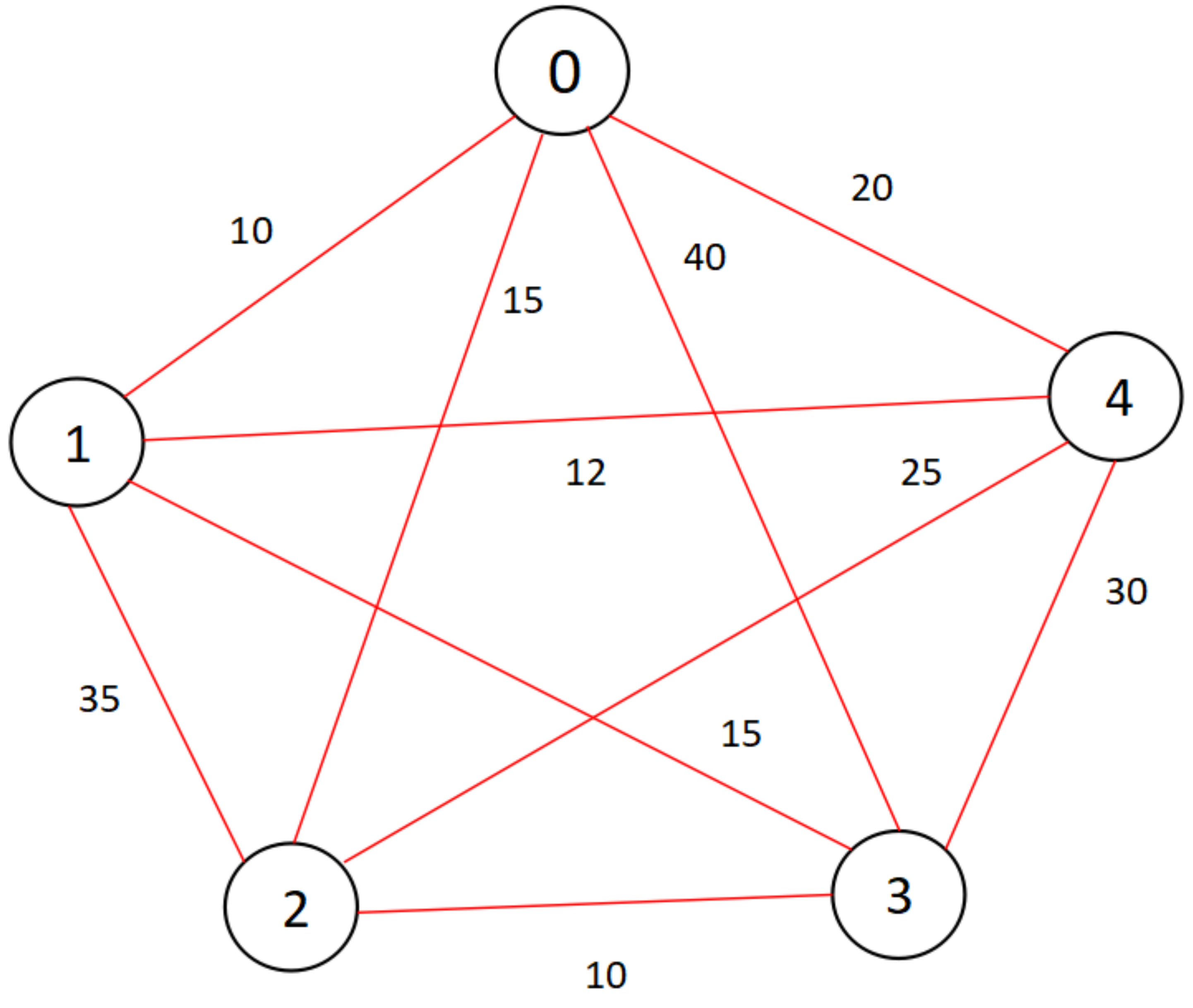
$1 \to 2 \to 3 \to 4$ < $1 \to 3 \to 2 \to 4$ → $1 \to 2 \to 3 \to 4 \to 5$ < $1 \to 3 \to 2 \to 4 \to 5$

General $S = \{s_1, \ldots, s_k\}$ $g(S, e) = \min_{1 \leq i \leq k} g(S_i, s_i) + d(s_i, e)$

The algorithm finishes when $g(\{2, \ldots, i-1, i+1, \ldots, n\}, i)$ is known for every integer $2 \leq i \leq n$

Complexities
Time: O(n^2*2^n)
Space: O(n*2^n)

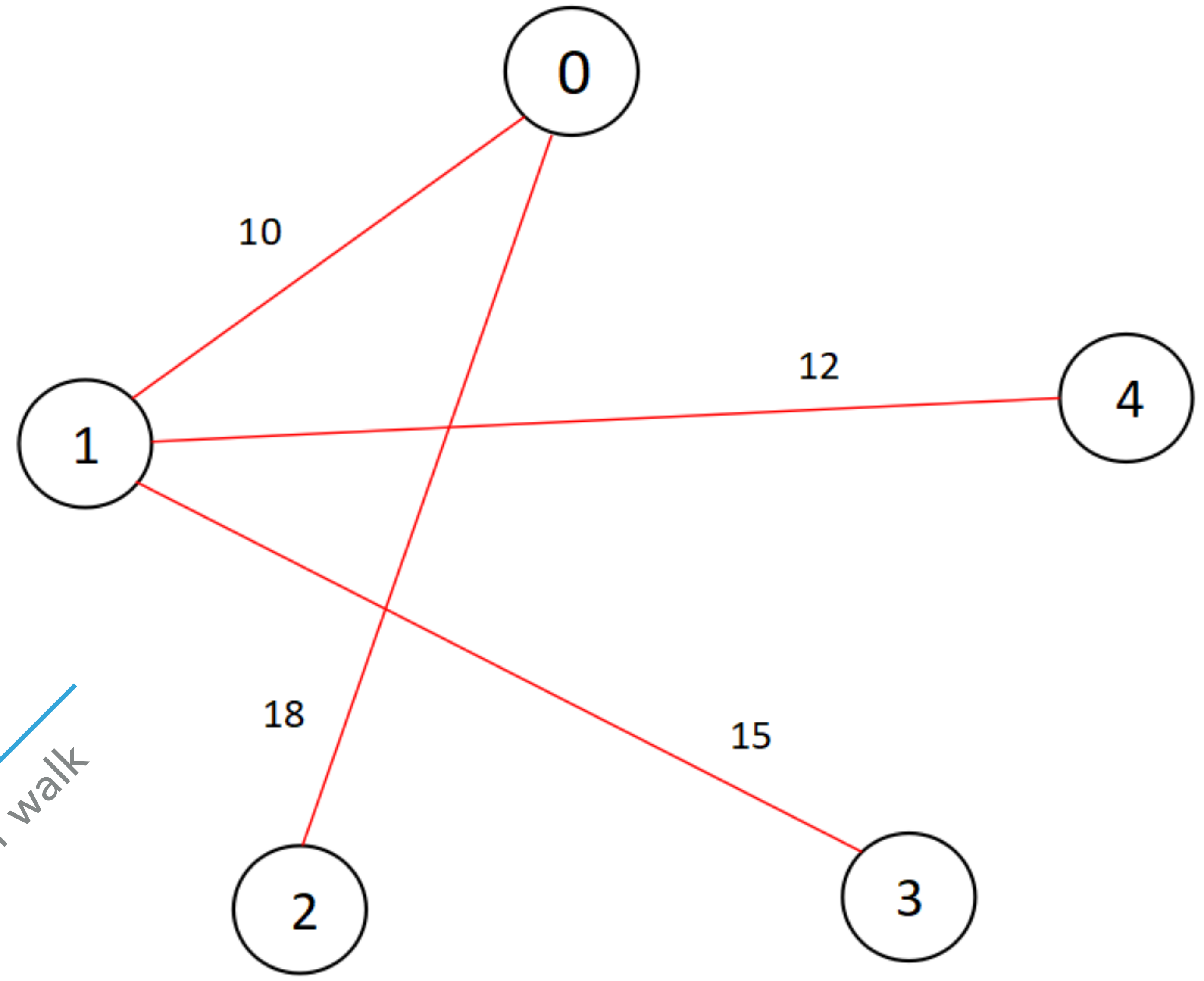
APPROXIMATION SOLUTION USING MINIMUM SPANNING TREE



Initial graph

Step 1

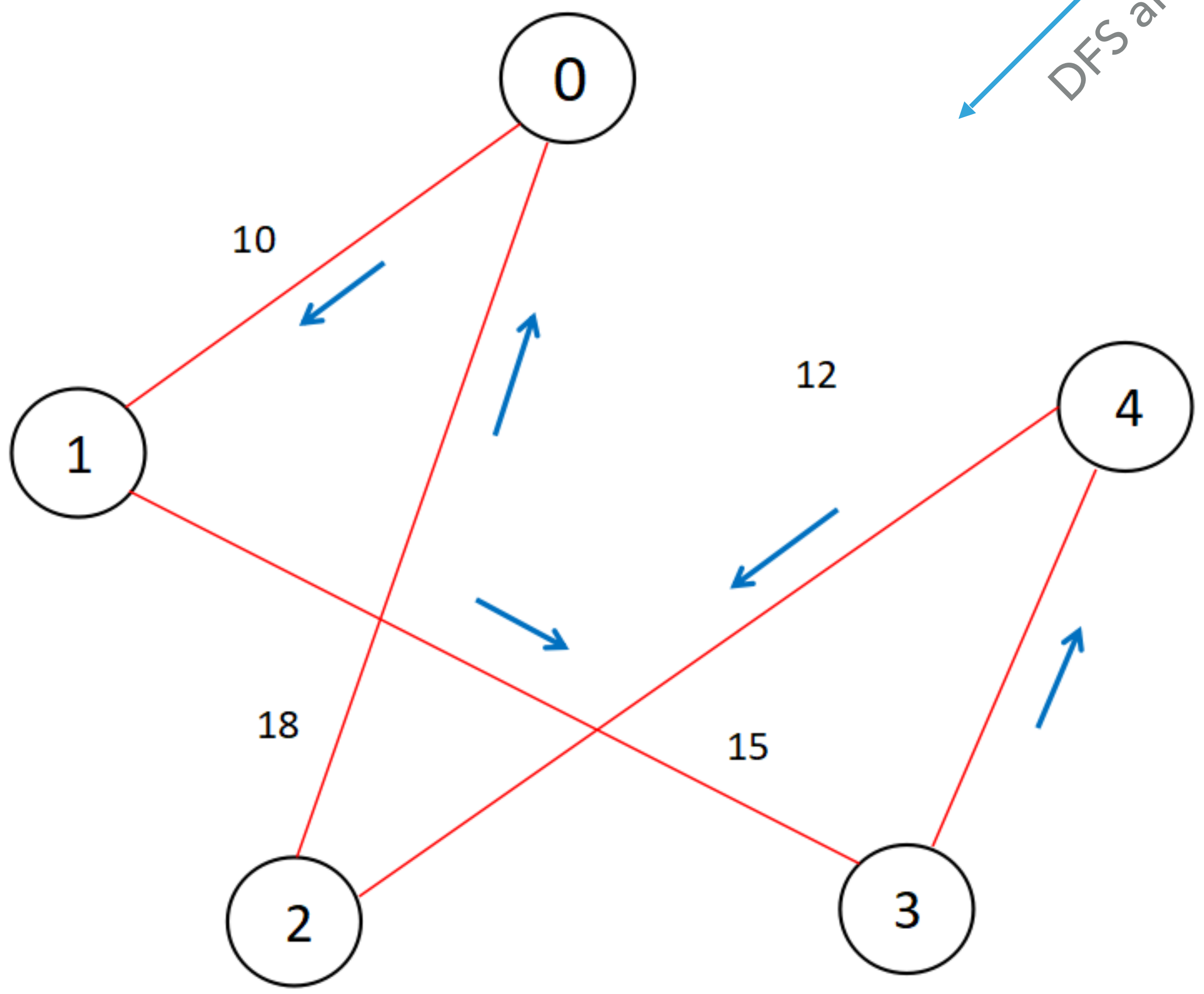
Minimum spanning tree



Minimum spanning tree: Prim's Algorithm

Step 2, 3

DFS and preorder walk



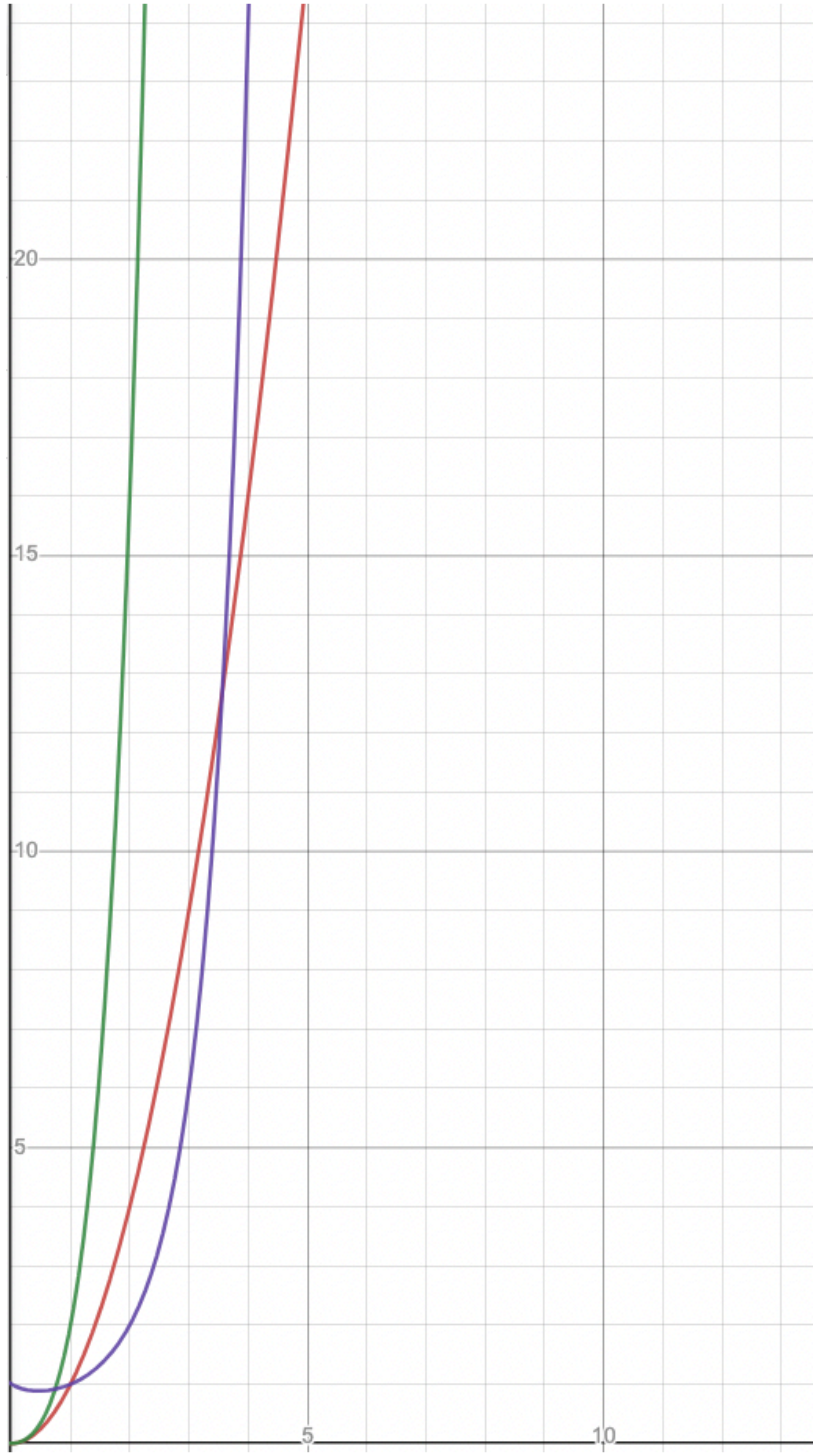
Nodes connected

Complexities

Time: O(V*V) which is the time complexity for prim's algorithm

Space: O(V*V)

SOLUTION COMPARISONS - EXECUTION TIME



Approximation solution

Brute force solution

Dynamic programming solution

Example: for n = 12

Brute force: 479001600 operations

Dynamic programming: 589824 operations

Approximation: 144 operations