Greedy algorithm

Athasit Surarerk

T H E O R E T I C A L M O D E L

March 29, 2005

Introduction

Typically, in optimization problems the algorithm needs to make a series of choices whose overall effect is to minimize the total cost, or maximize the total benefit, of some system. The greedy method consists of making the choices in sequence such that each individual choice is best according to some limited "short-term" criterion that is not too expensive to evaluate.

Optimal binary search tree

Create an optimal binary search tree

DATA	Probability
A	0.08
B	0.22
C	0.20
D	0.18
E	0.25
F	0.05
G	0.02

Optimal binary search tree

Create an optimal binary search tree

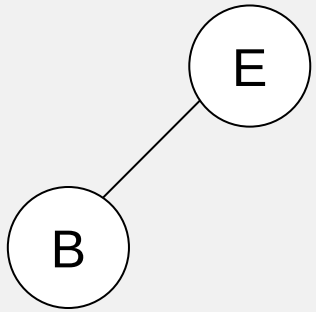
DATA	Probability
A	0.08
B	0.22
C	0.20
D	0.18
E	0.25
F	0.05
G	0.02

E

Optimal binary search tree

Create an optimal binary search tree

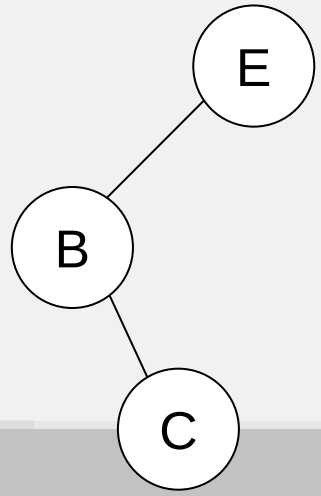
DATA	Probability
A	0.08
B	0.22
C	0.20
D	0.18
E	0.25
F	0.05
G	0.02



Optimal binary search tree

Create an optimal binary search tree

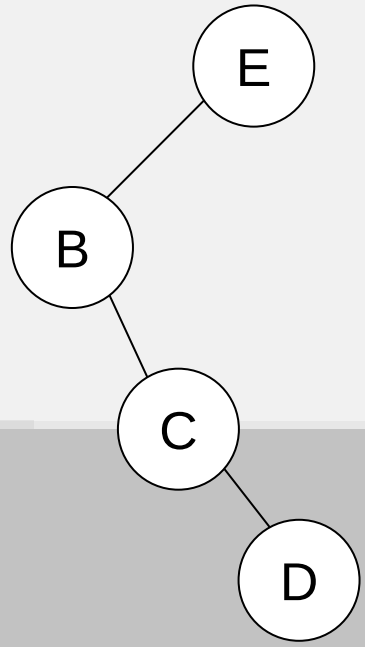
DATA	Probability
A	0.08
B	0.22
C	0.20
D	0.18
E	0.25
F	0.05
G	0.02



Optimal binary search tree

Create an optimal binary search tree

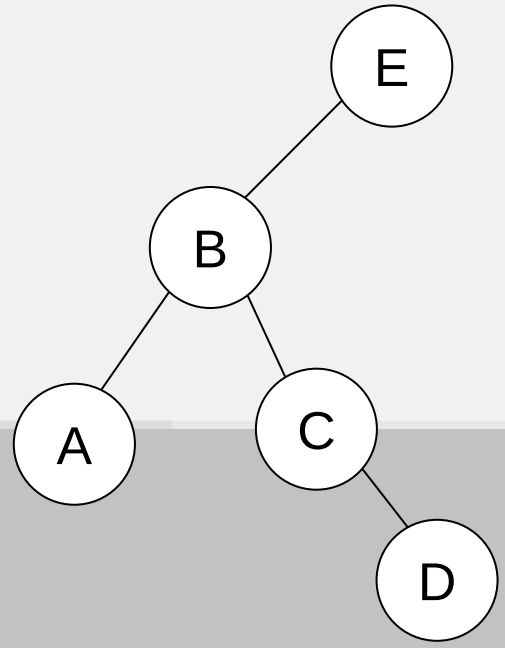
DATA	Probability
A	0.08
B	0.22
C	0.20
D	0.18
E	0.25
F	0.05
G	0.02



Optimal binary search tree

Create an optimal binary search tree

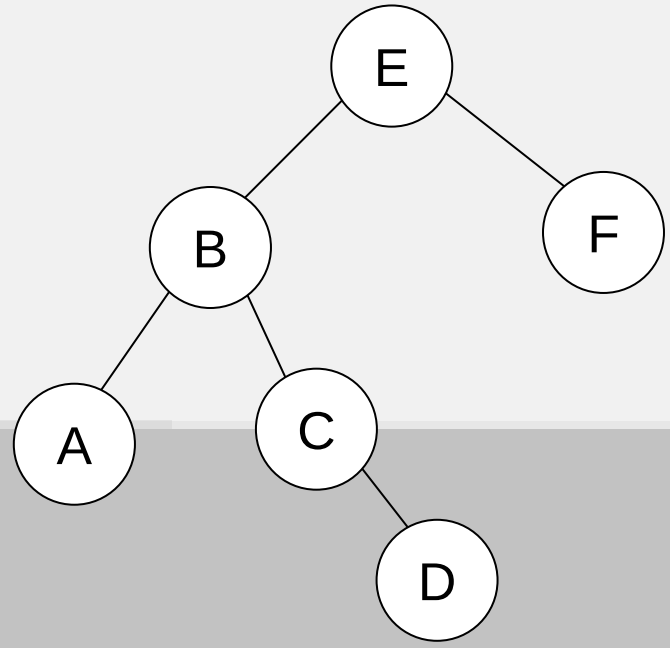
DATA	Probability
A	0.08
B	0.22
C	0.20
D	0.18
E	0.25
F	0.05
G	0.02



Optimal binary search tree

Create an optimal binary search tree

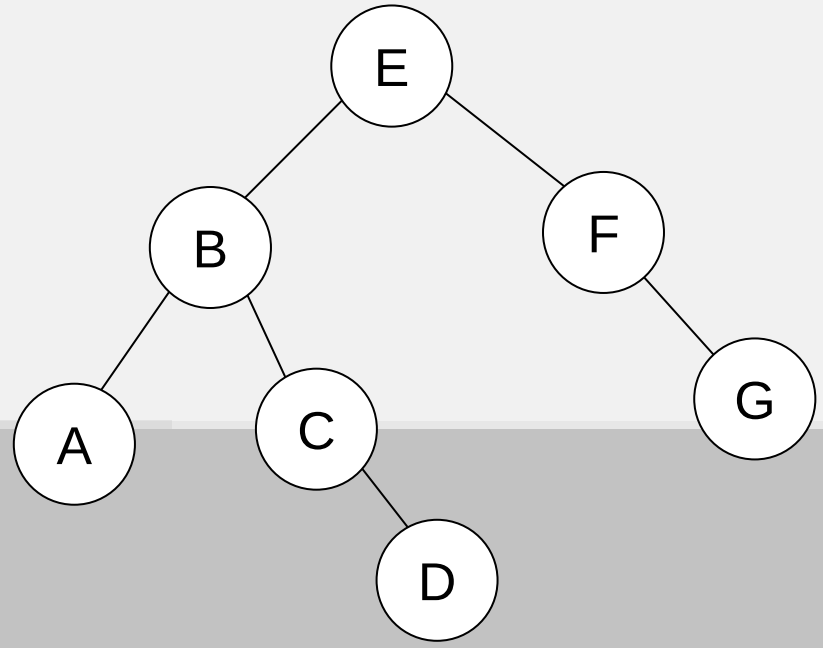
DATA	Probability
A	0.08
B	0.22
C	0.20
D	0.18
E	0.25
F	0.05
G	0.02



Optimal binary search tree

Create an optimal binary search tree

DATA	Probability
A	0.08
B	0.22
C	0.20
D	0.18
E	0.25
F	0.05
G	0.02



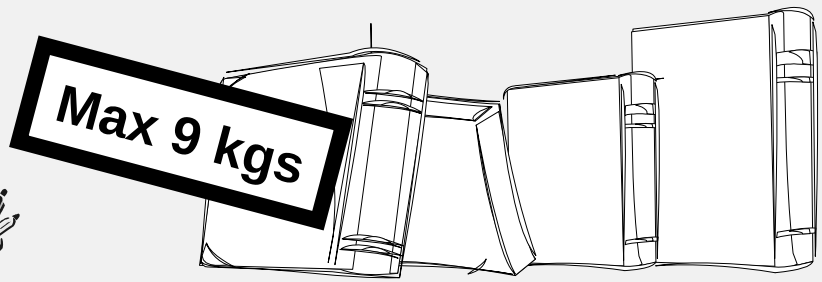
Average-time for searching **2.41**

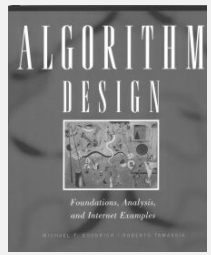
Definition

An algorithm that always takes the best immediate, or local, solution while finding an answer. Greedy algorithms find the overall, or globally, optimal solution for some optimization problem, but may find less-than-optimal solutions for some instances of other problems.

0/1 Knapsack problem

Choose items with maximum total benefit but with some limitation.





weight	**4 kgs**	**2 kgs**	**2 kgs**	**6 kgs**	**2 kgs**
value	**B200**	**B30**	**B60**	**B240**	**B800**
Value / kg	**B50**	**B15**	**B30**	**B40**	**B400**
	2	5	4	3	1

Greedy algorithm

Solution : B1060

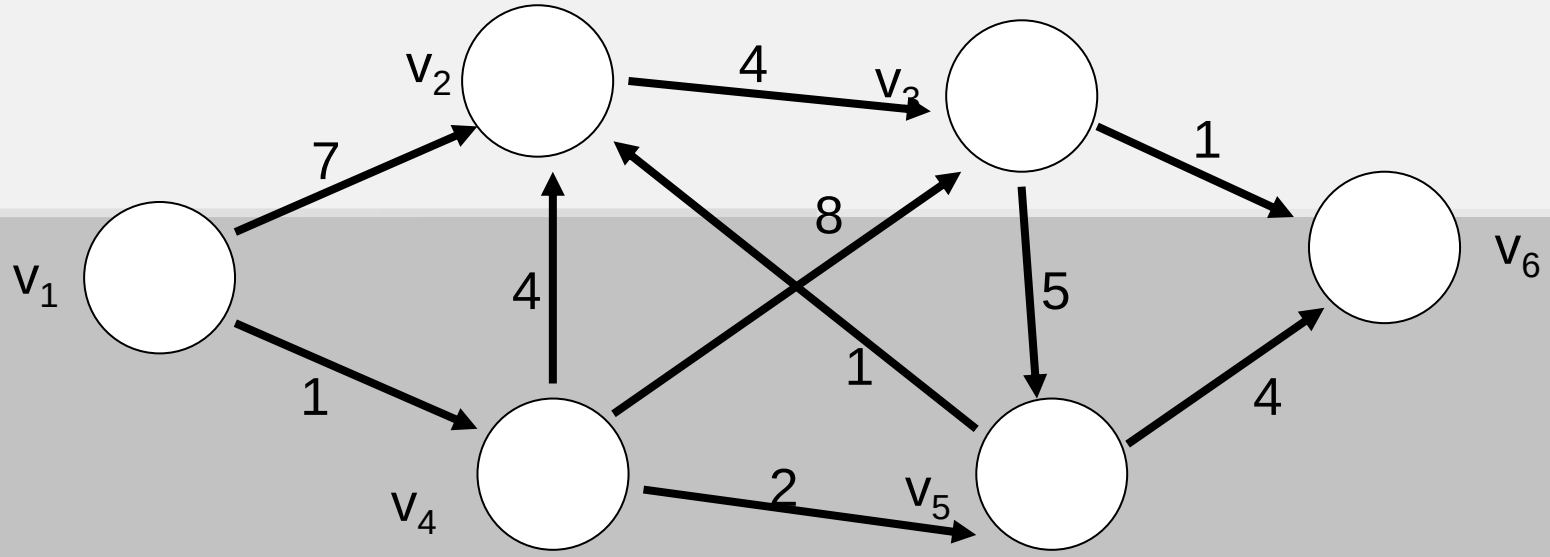
Shortest path

SINGLE-SOURCE

Ginen a nonnegative weight graph G = (V, E)

Find a shortest path from a in V to other vertices in V.

DIJKSTRA



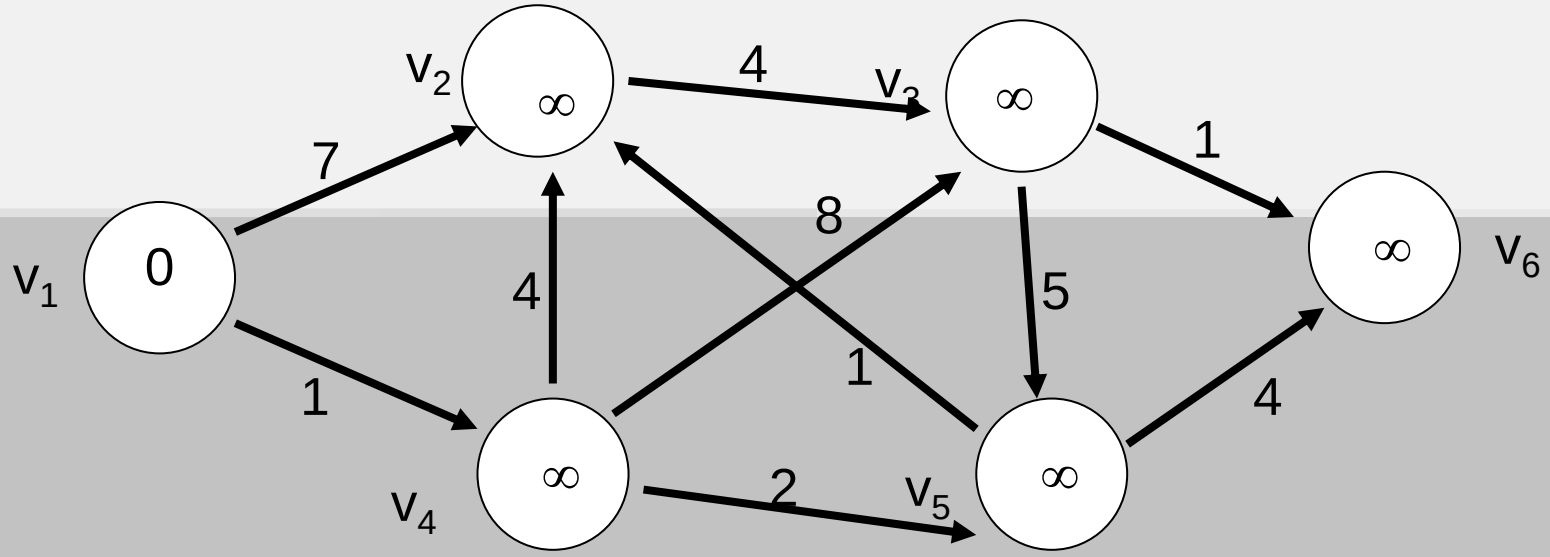
Shortest path

SINGLE-SOURCE

Ginen a nonnegative weight graph G = (V, E)

Find a shortest path from a in V to other vertices in V.

DIJKSTRA



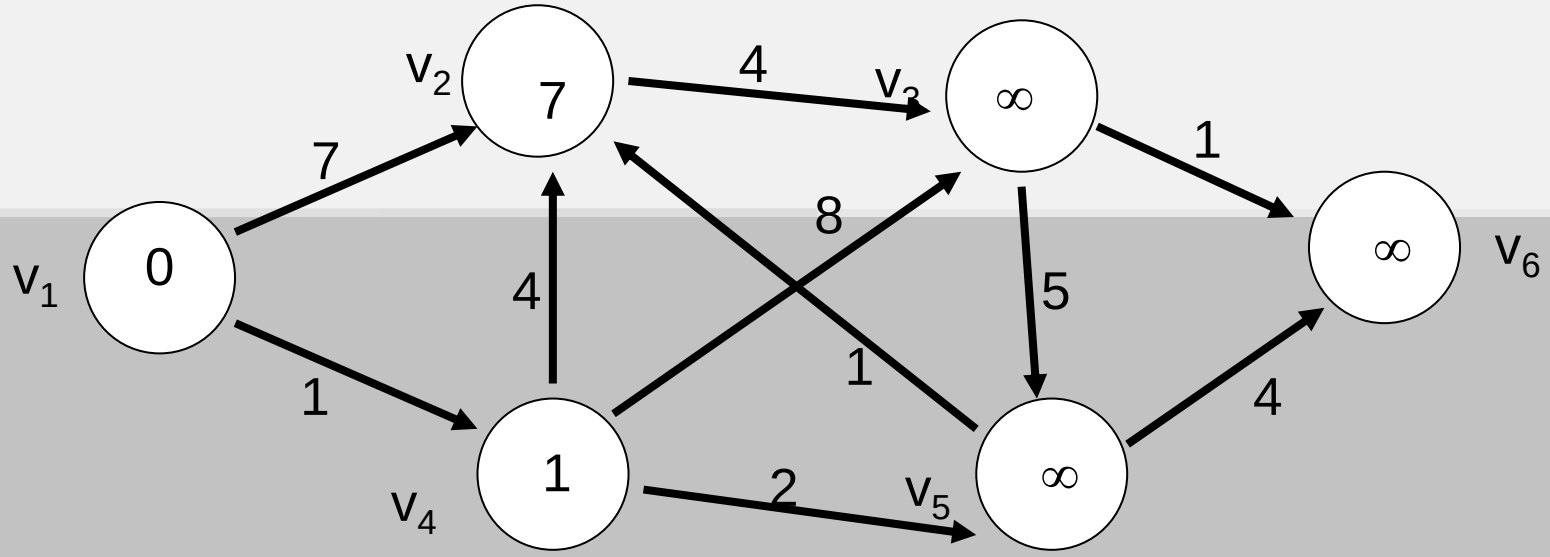
Shortest path

SINGLE-SOURCE

Ginen a nonnegative weight graph G = (V, E)

Find a shortest path from a in V to other vertices in V.

DIJKSTRA



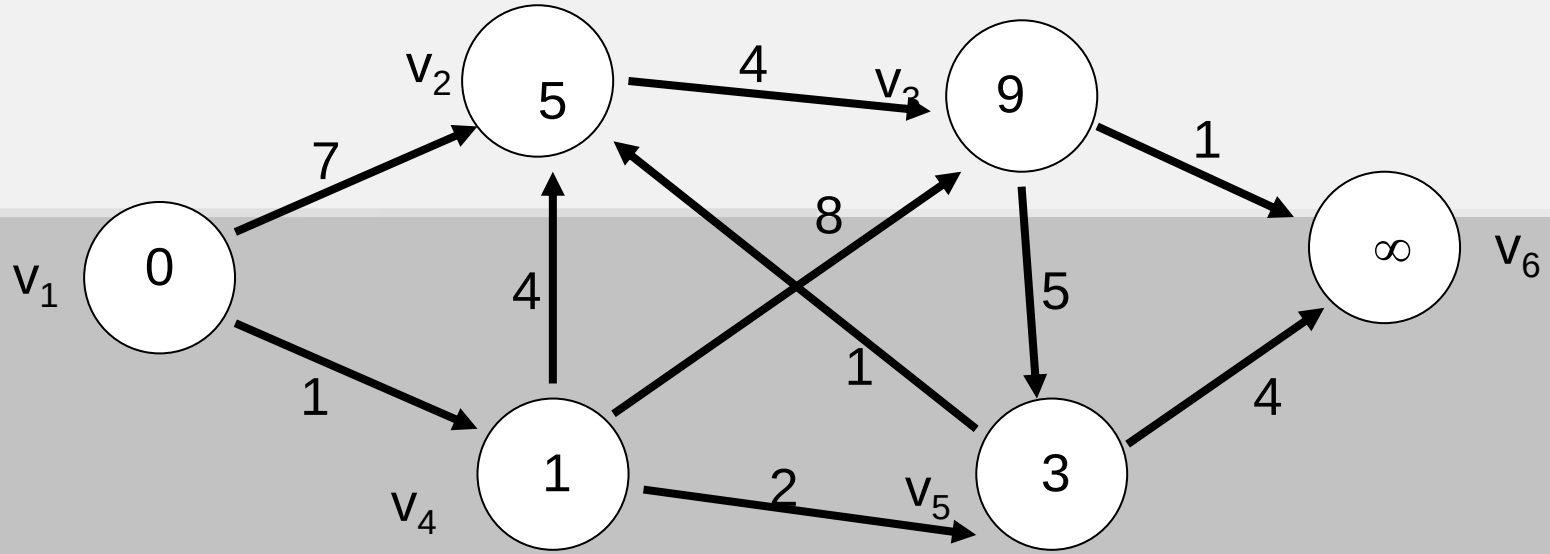
Shortest path

SINGLE-SOURCE

Ginen a nonnegative weight graph G = (V, E)

Find a shortest path from a in V to other vertices in V.

DIJKSTRA



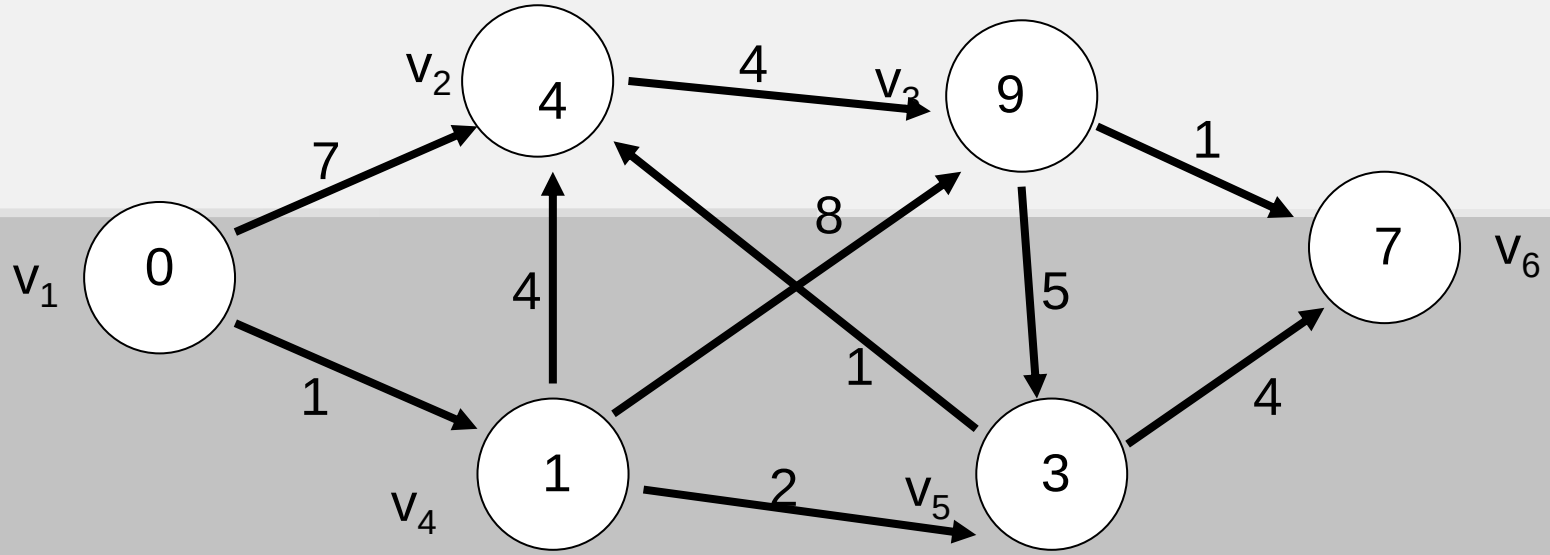
Shortest path

SINGLE-SOURCE

Ginen a nonnegative weight graph G = (V, E)

Find a shortest path from a in V to other vertices in V.

DIJKSTRA



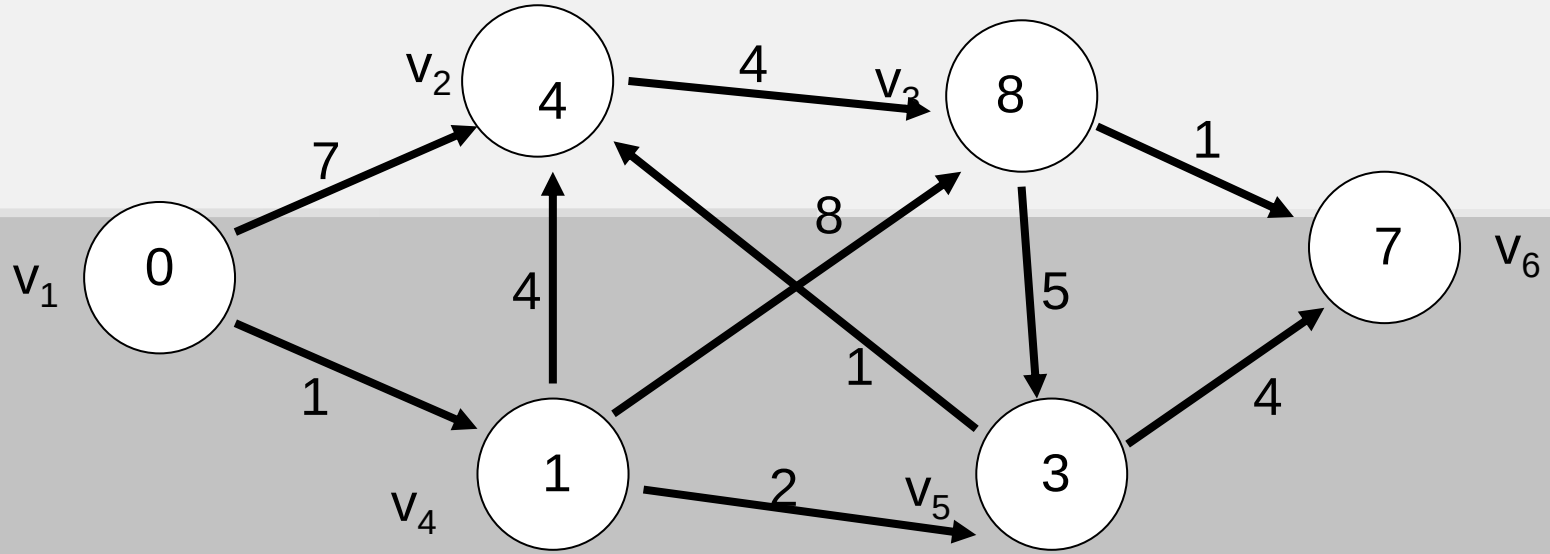
Shortest path

SINGLE-SOURCE

Ginen a nonnegative weight graph G = (V, E)

Find a shortest path from a in V to other vertices in V.

DIJKSTRA



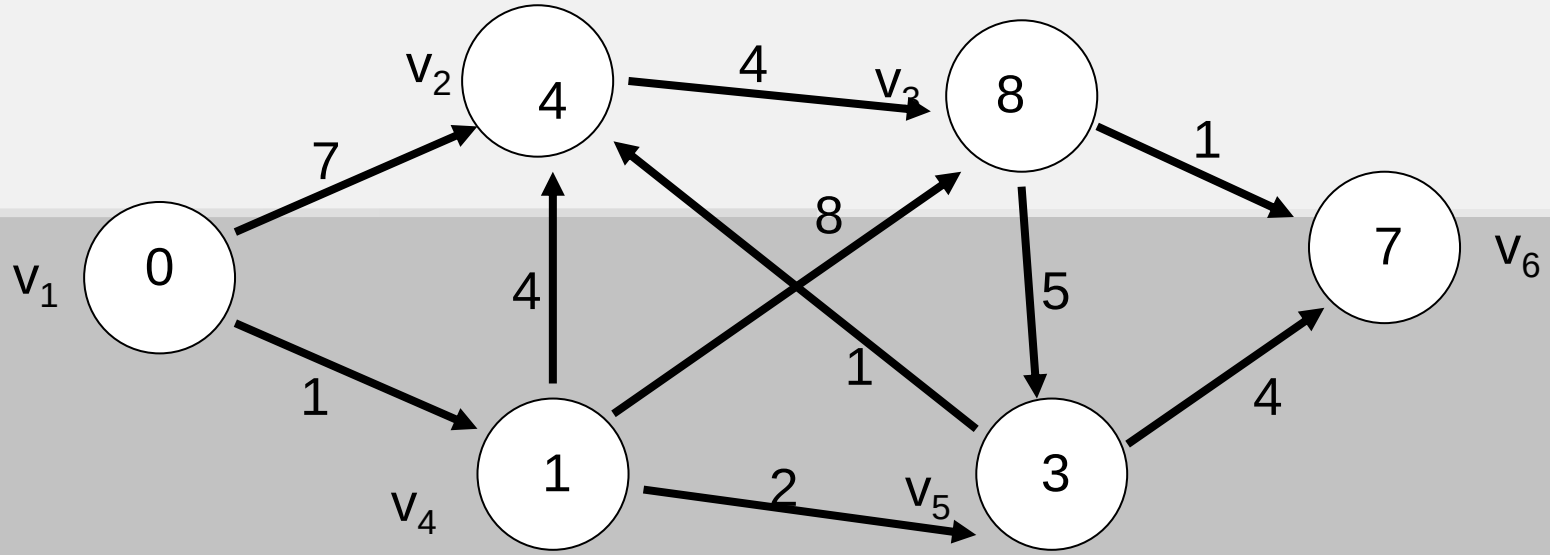
Shortest path

SINGLE-SOURCE

Ginen a nonnegative weight graph G = (V, E)

Find a shortest path from a in V to other vertices in V.

DIJKSTRA



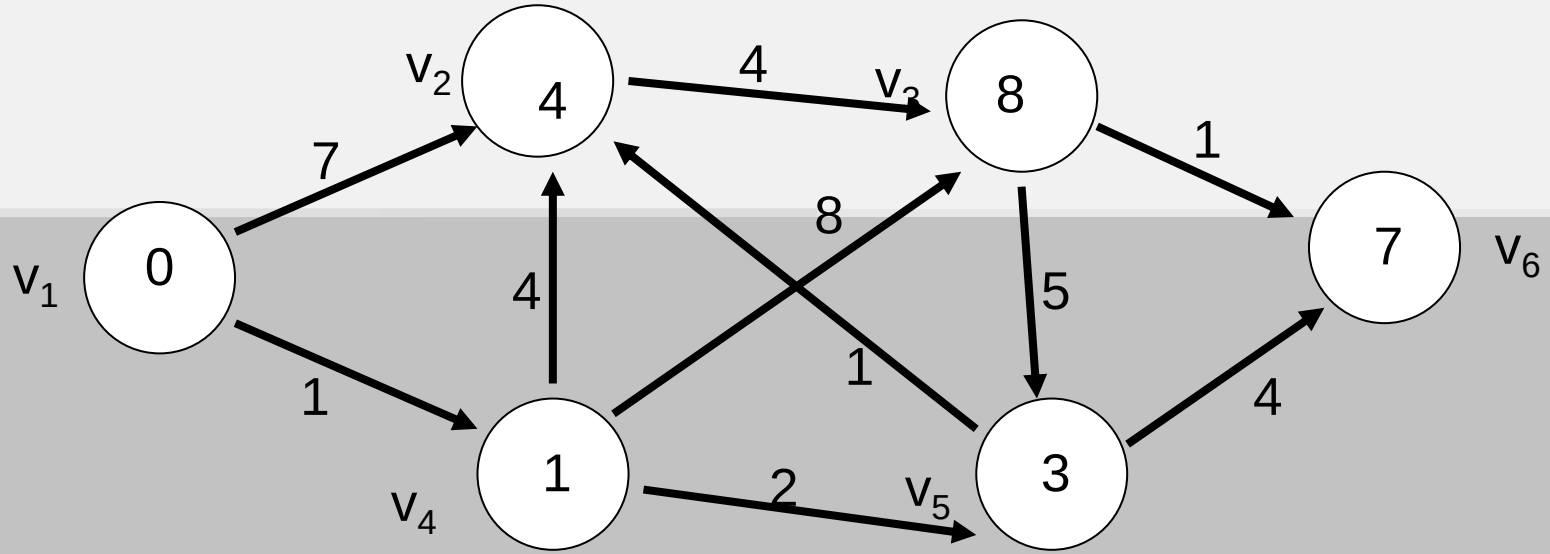
Shortest path

SINGLE-SOURCE

Ginen a nonnegative weight graph G = (V, E)

Find a shortest path from a in V to other vertices in V.

DIJKSTRA



Activity-selection

Given a set of activities { a_1, a_2, a_3, …, a_n }.

Let s_j and f_j be a started and finished time for the activity a_j.

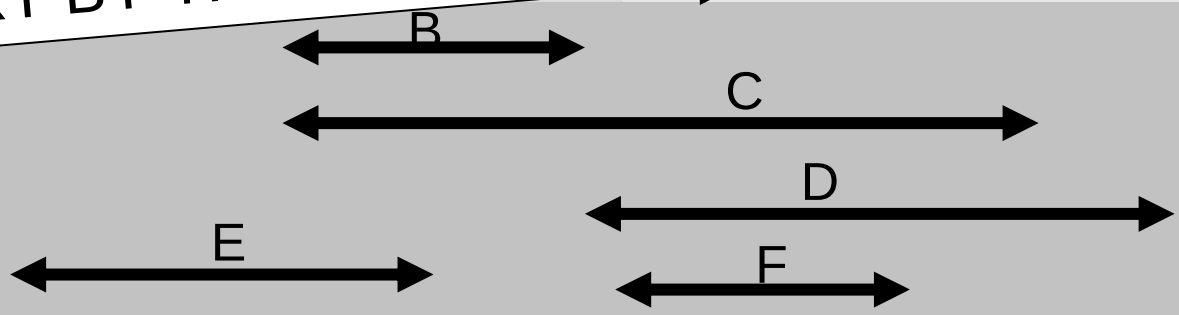
a_j	A	B	C	D	E	F
s_j	0	3	3	5	1	5
f_j	6	5	8	9	4	7

Select the maximum number of activities.

Activity-selection

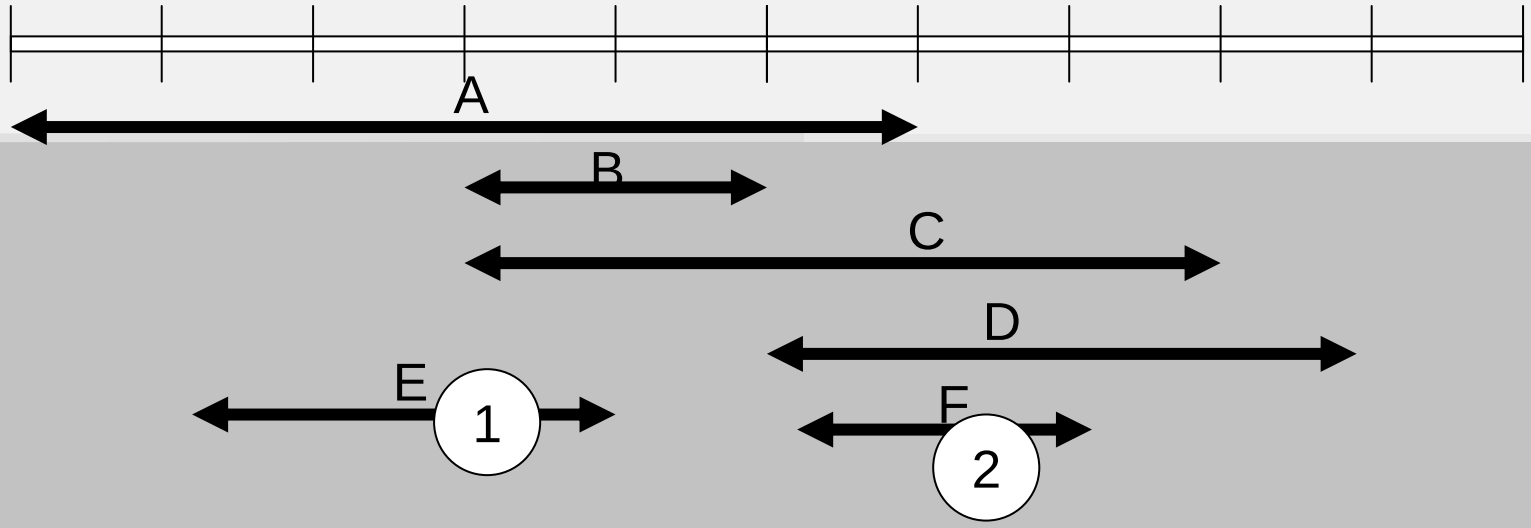
a_j	A	B	C	D	E	F
s_j	0	3	3	5	1	5
f_j	6	5	8	9	4	7

START BY THE ACTIVITY THAT FINISHS FIRSTLY.



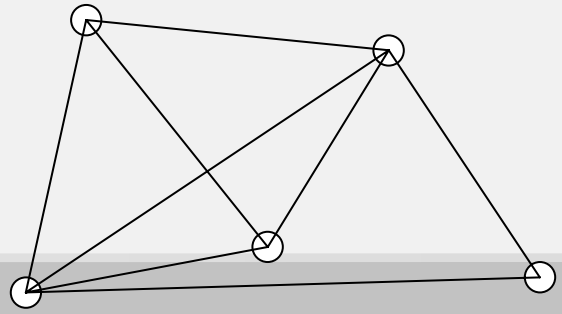
Activity-selection

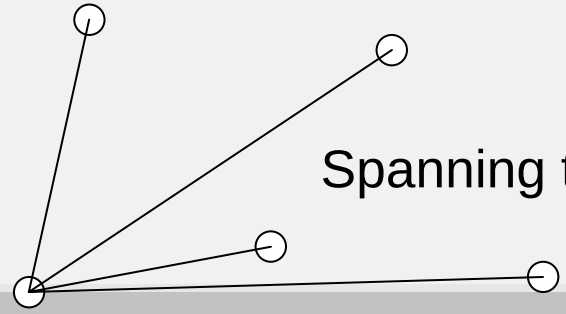
a_j	A	B	C	D	E	F
s_j	0	3	3	5	1	5
f_j	6	5	8	9	4	7



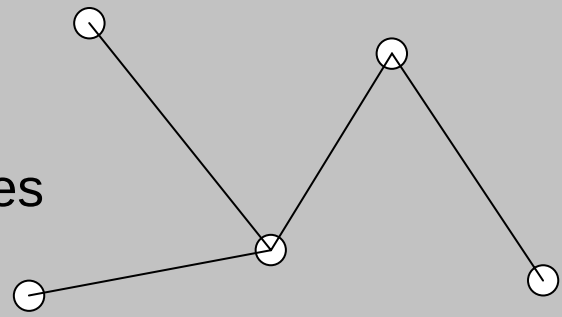
Spanning Tree

A sub-graph S of a graph G is a spanning tree if S contains all vertices of G and S is also a tree.





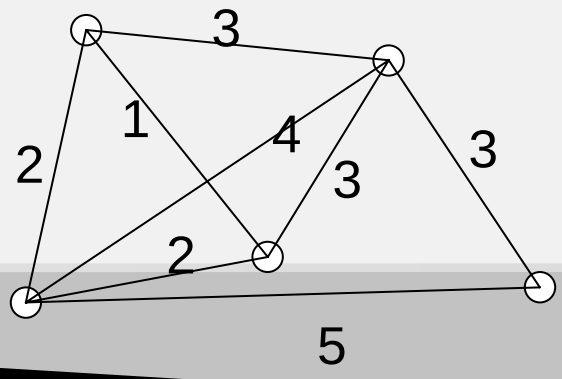
Spanning trees

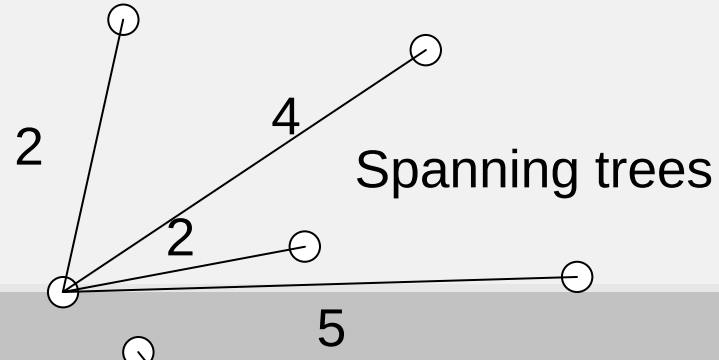


Spanning trees

Spanning Tree

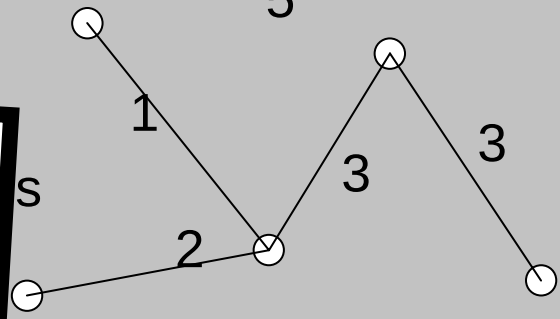
A sub-graph S of a graph G is a spanning tree if S contains all vertices of G and S is also a tree.





MINIMUM SPANNING TREE

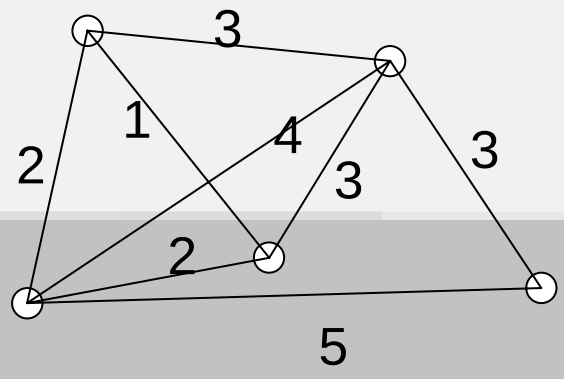
minimum weight tree



Minimum spanning tree

A spanning tree with the minimum weight.

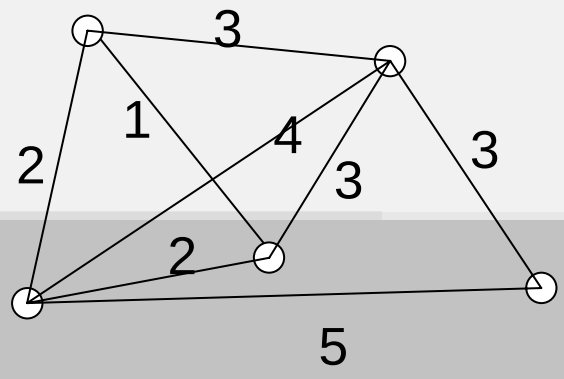
Kruskal’s algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

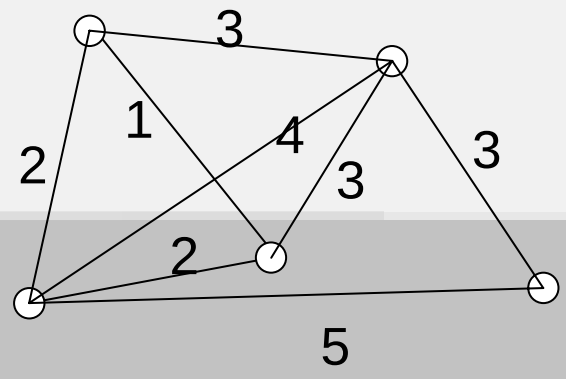
Kruskal’s algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

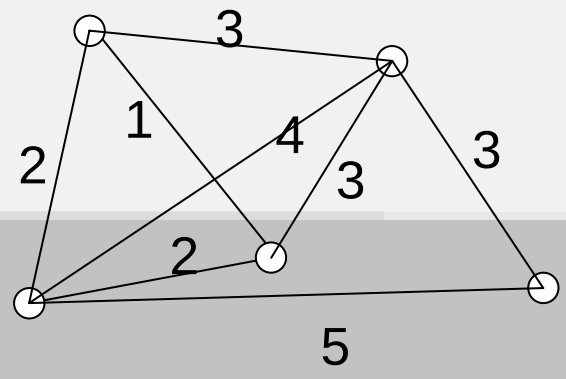
Kruskal's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

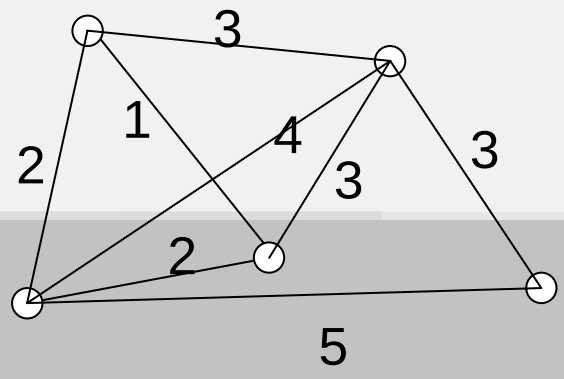
Kruskal's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

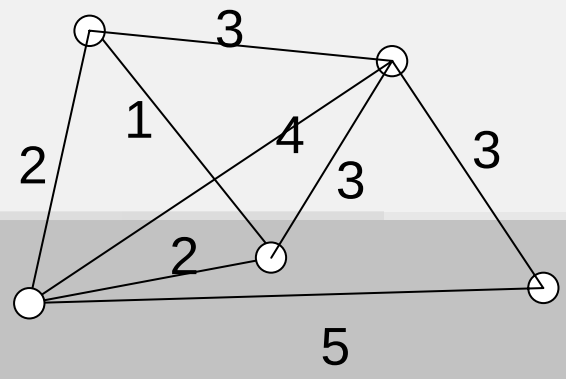
Kruskal's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

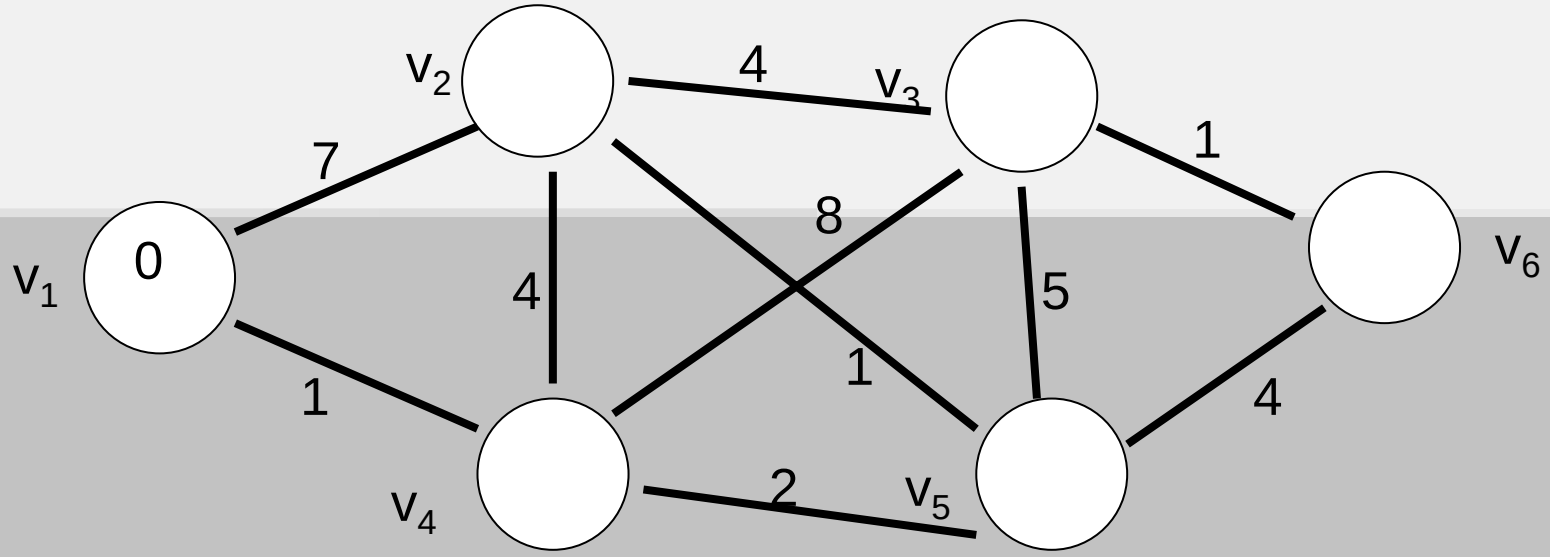
Kruskal's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

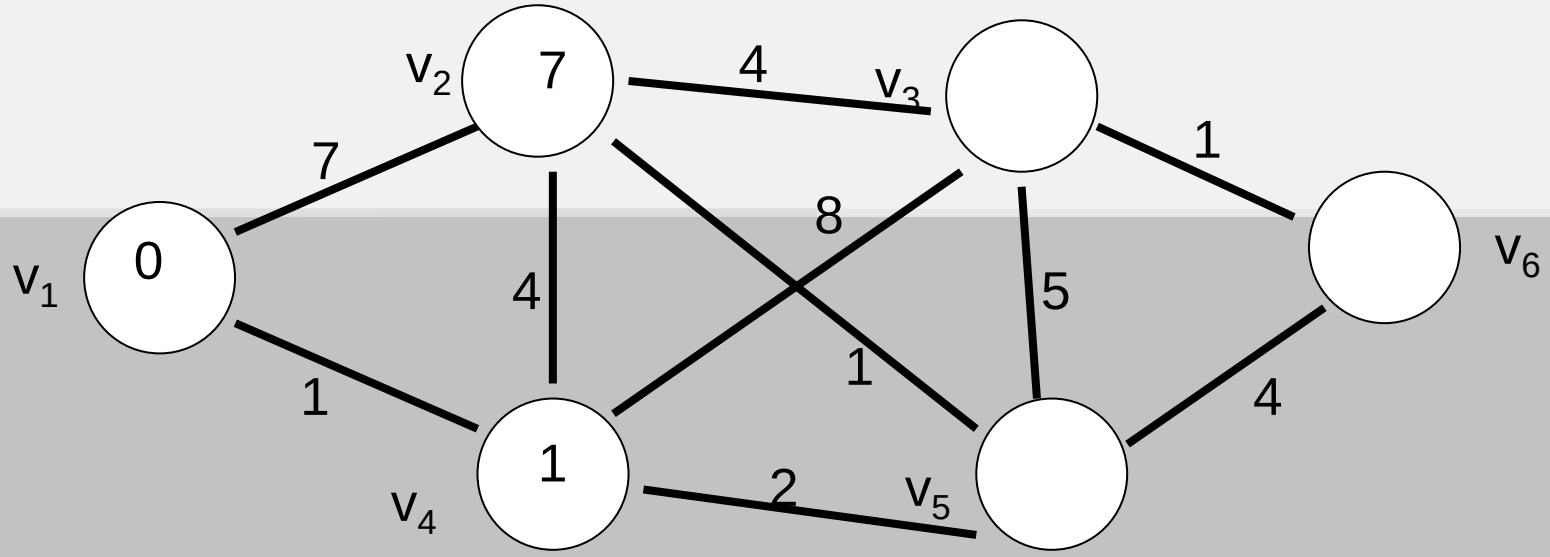
Prim's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

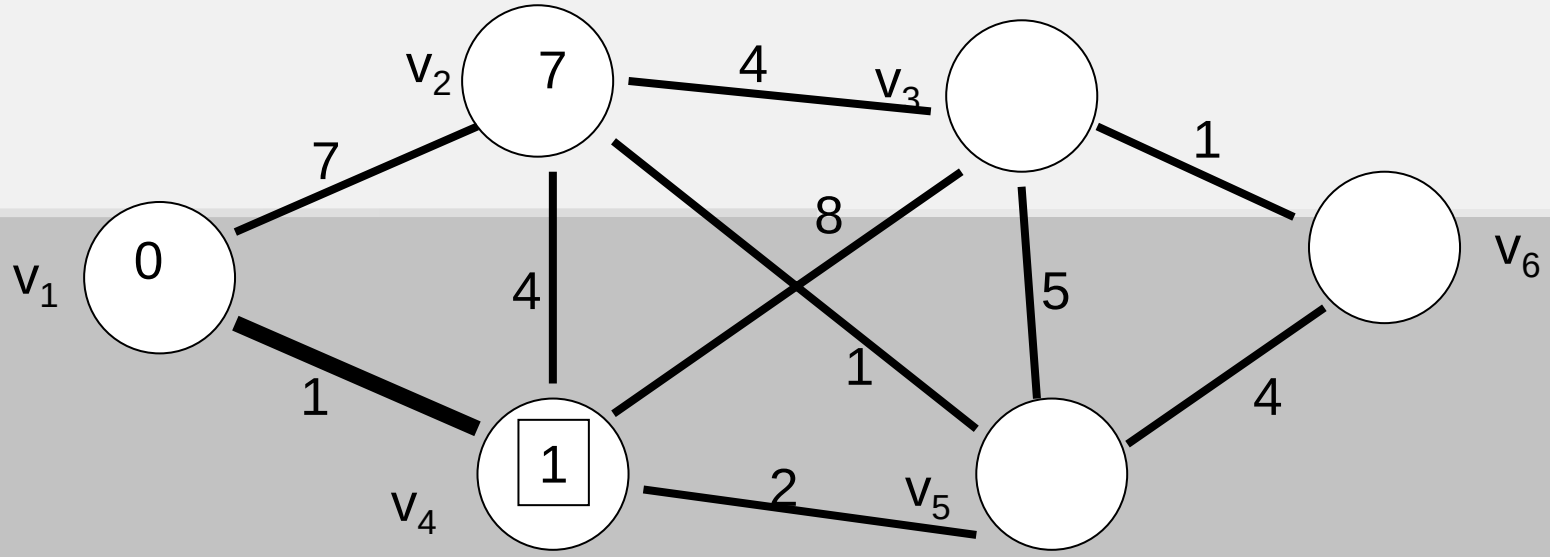
Prim's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

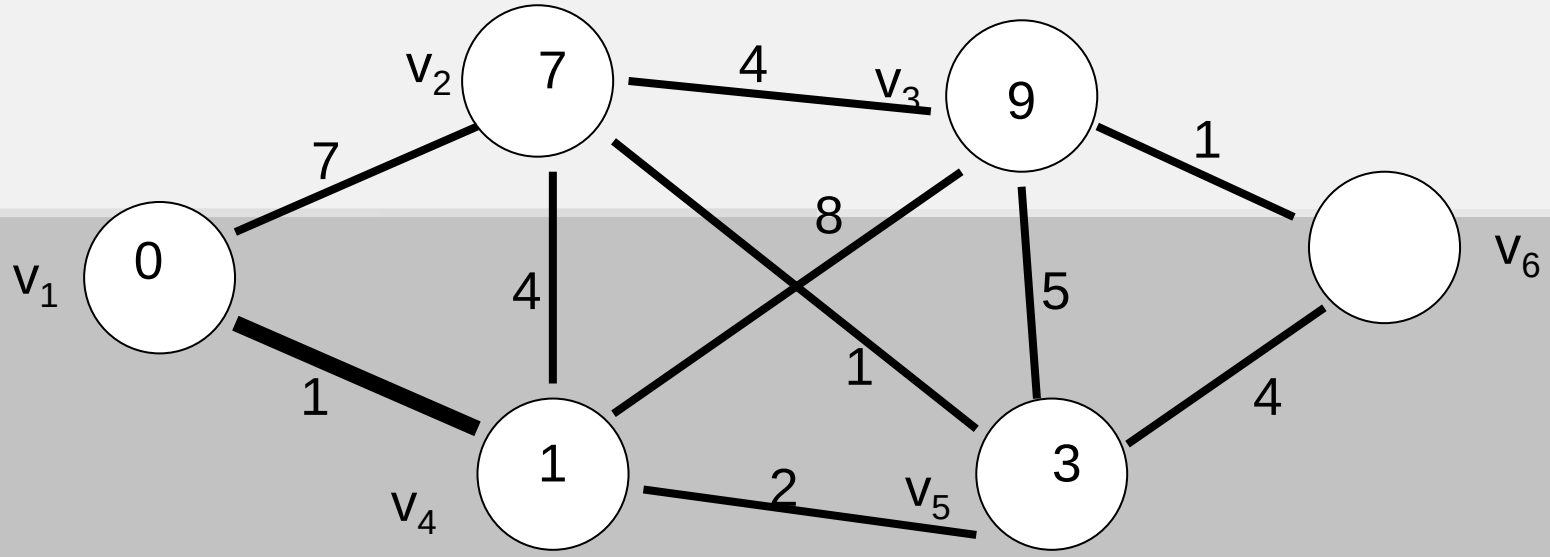
Prim's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

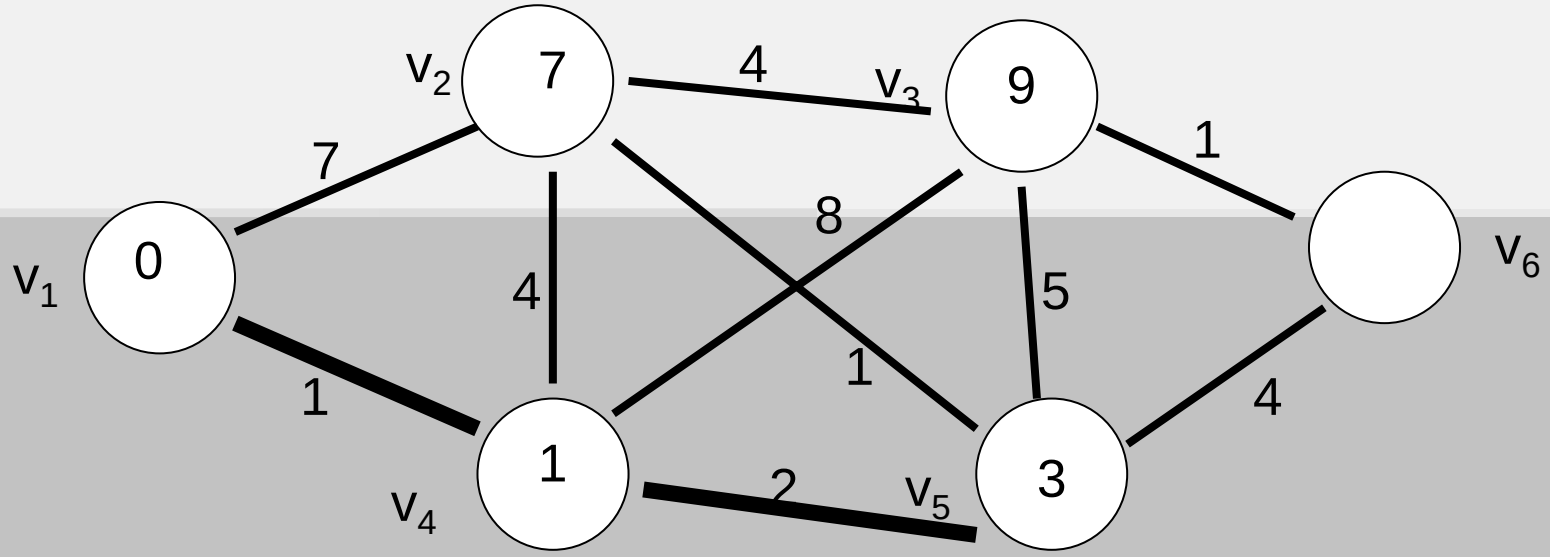
Prim's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

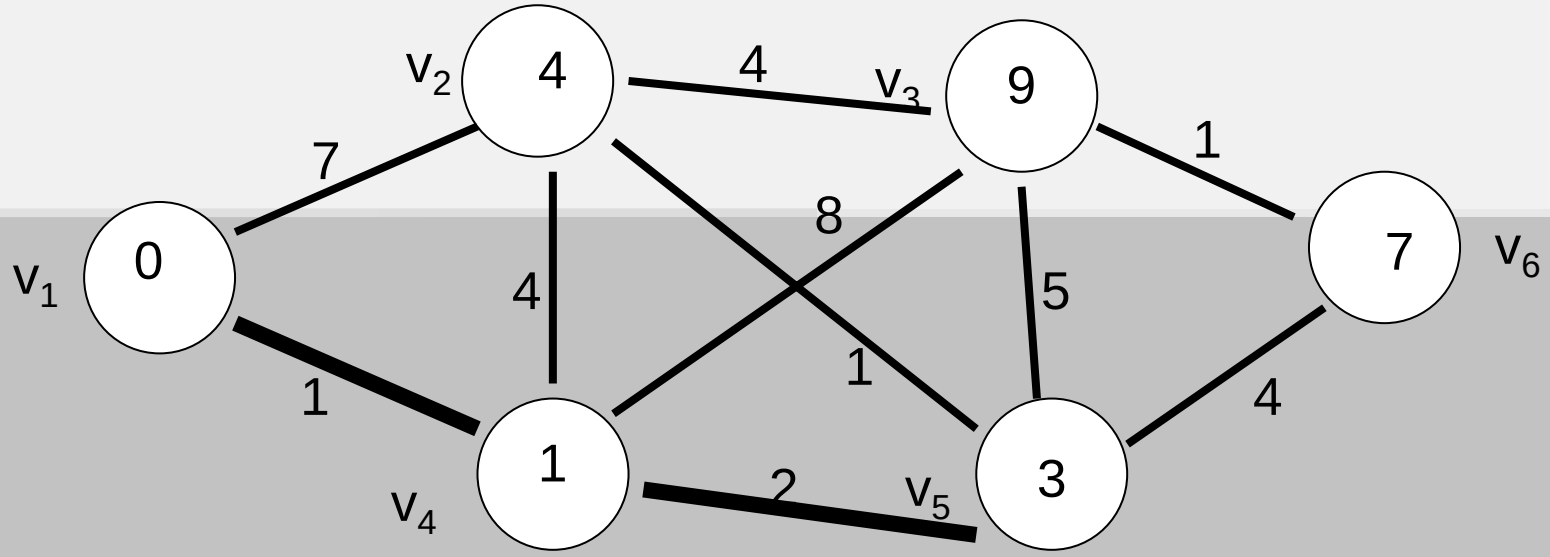
Prim's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

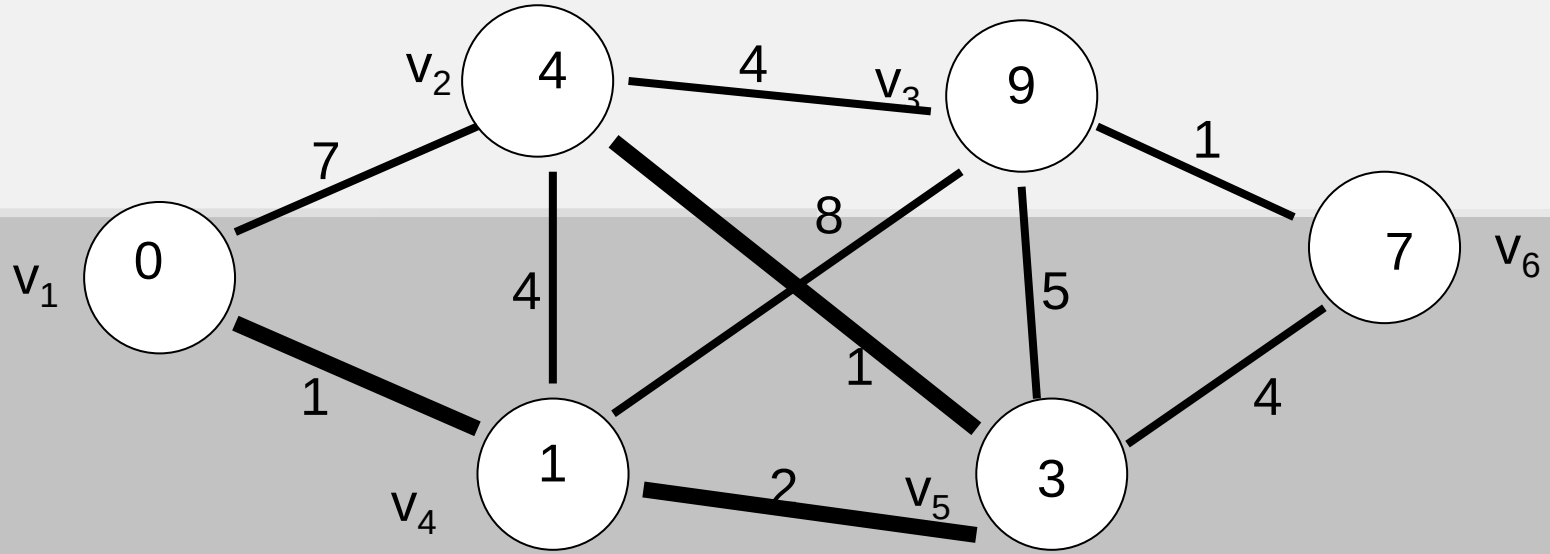
Prim's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

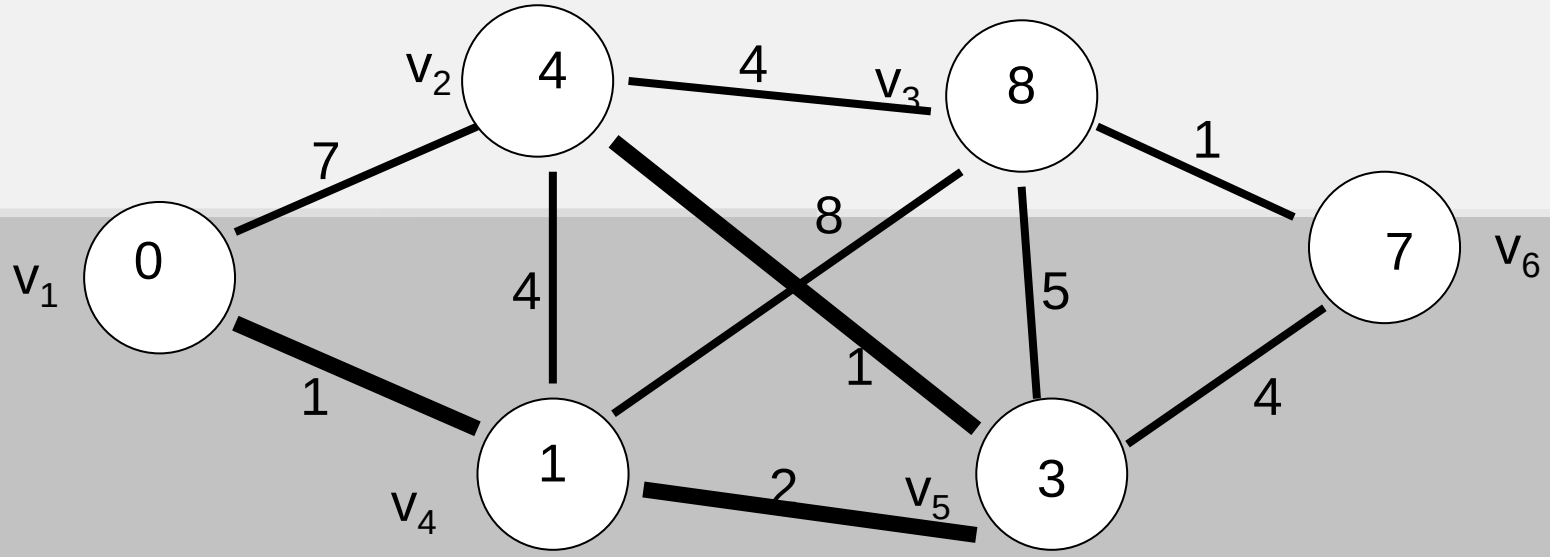
Prim's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

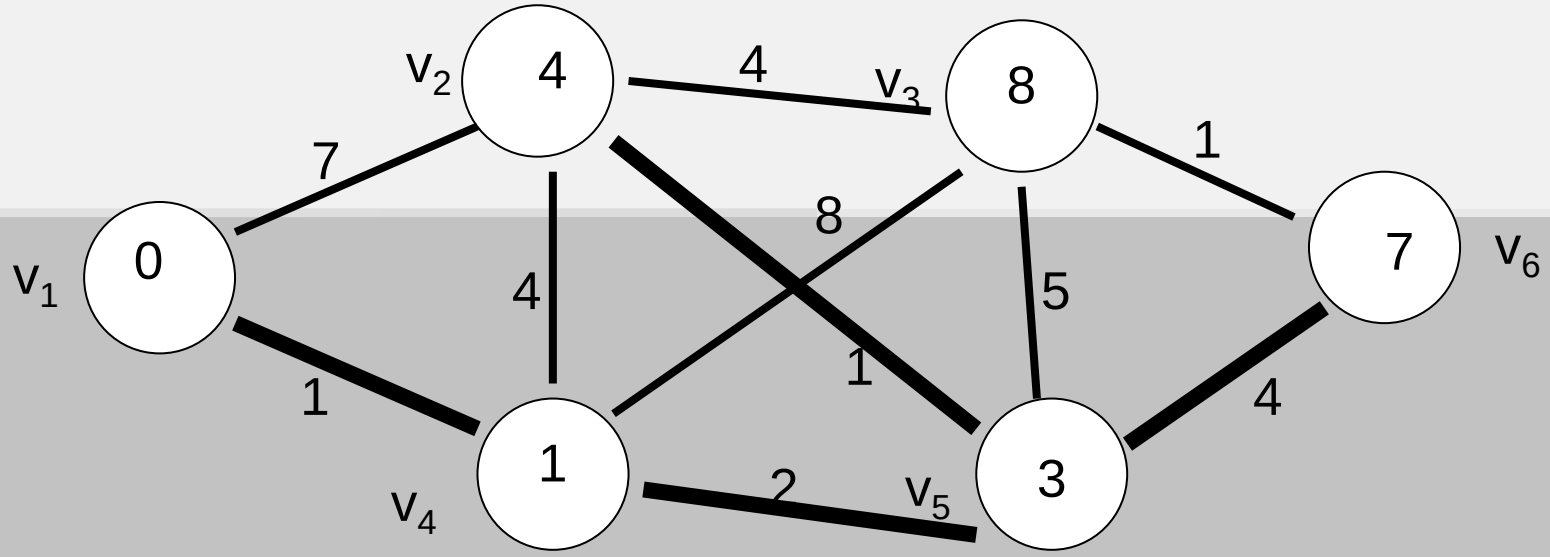
Prim's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

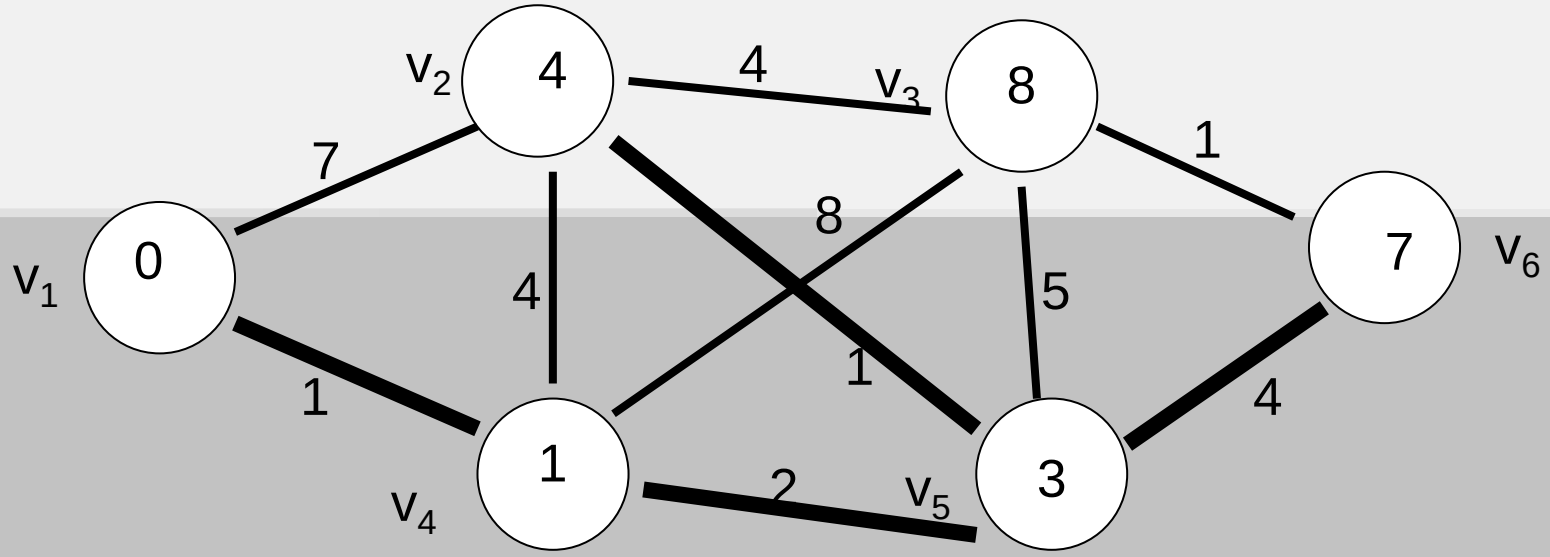
Prim's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

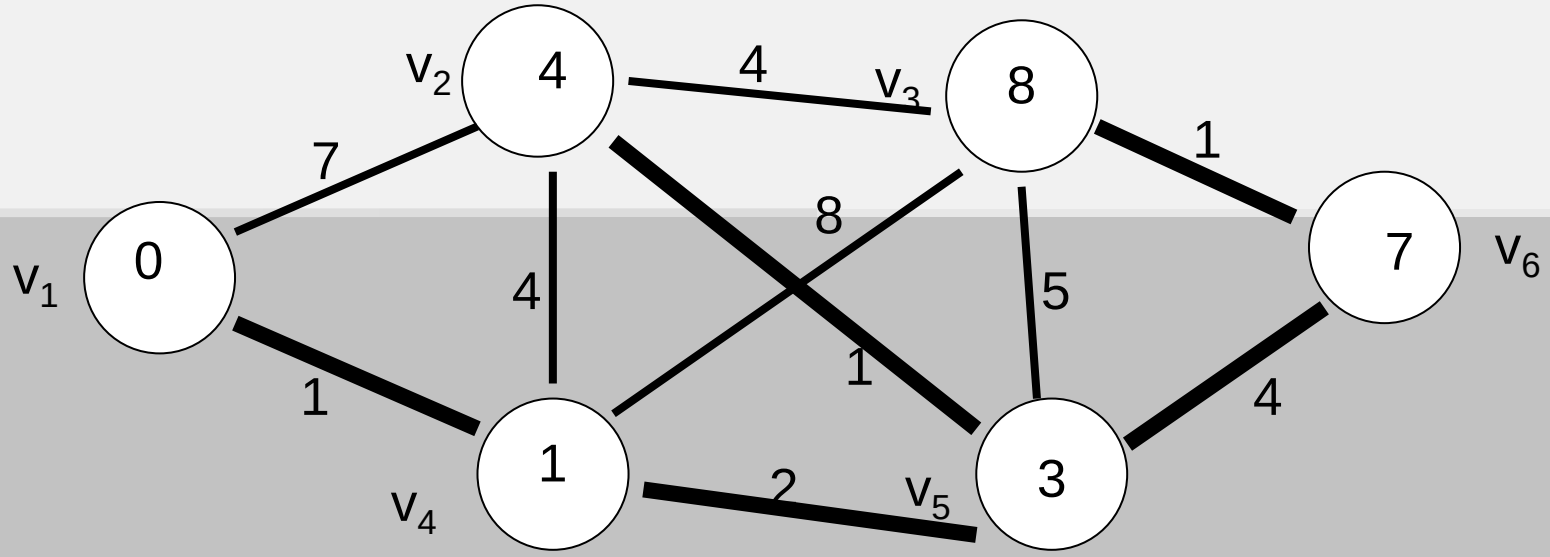
Prim's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

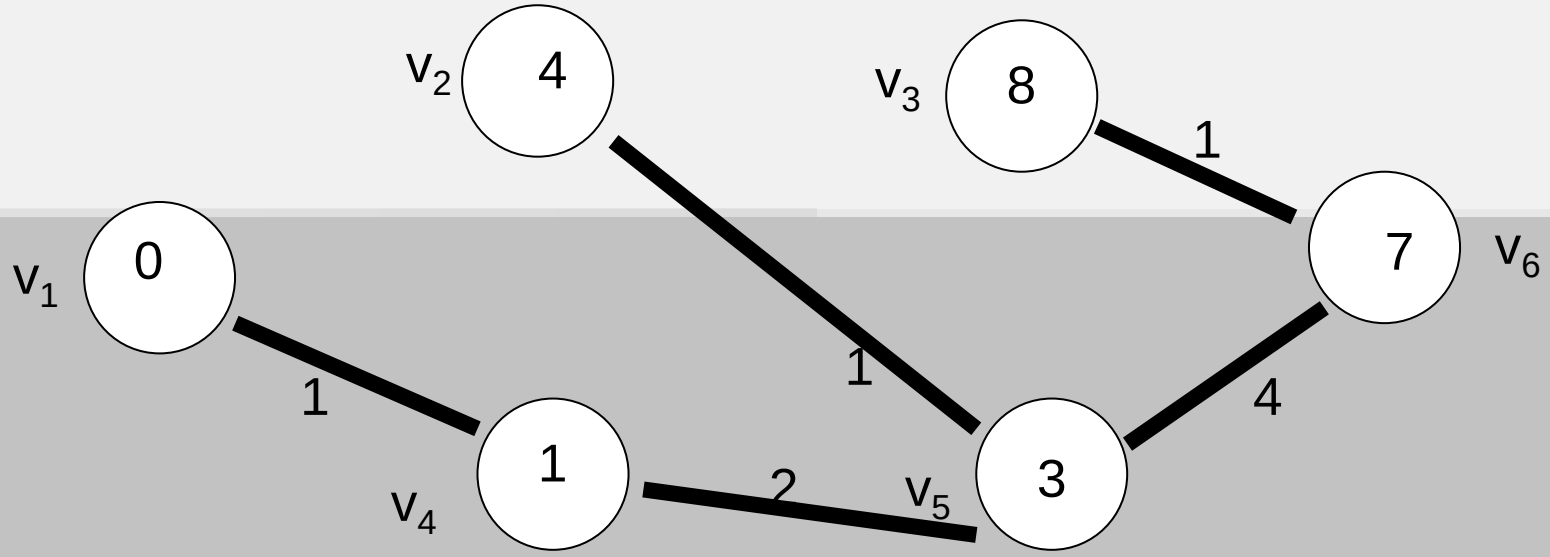
Prim's algorithm



Minimum spanning tree

A spanning tree with the minimum weight.

Prim's algorithm



Huffman code

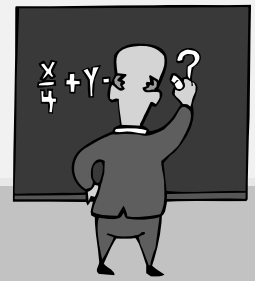
Find an algorithm for encoding these characters.

Characters	frequency
A	40
B	14
C	15
D	8
E	2
F	21
G	18

Problems

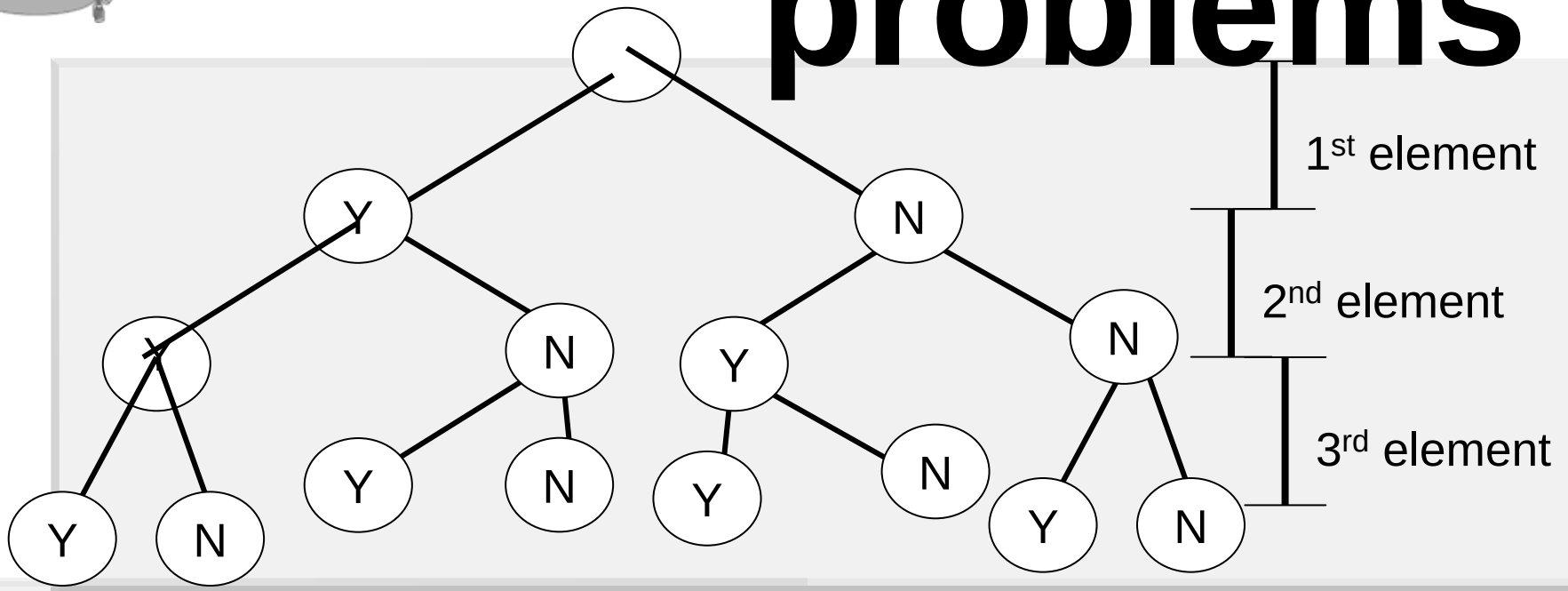
Three groups of complexity problems

- **Subset problems (2^n)**
- **Permutation problems ($n!$)**
- **Partition problems ($n!$)**

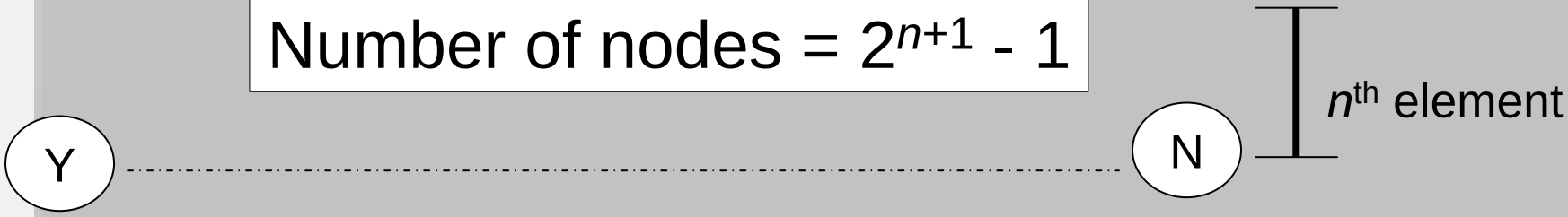


State-space graph / tree

Subset problems



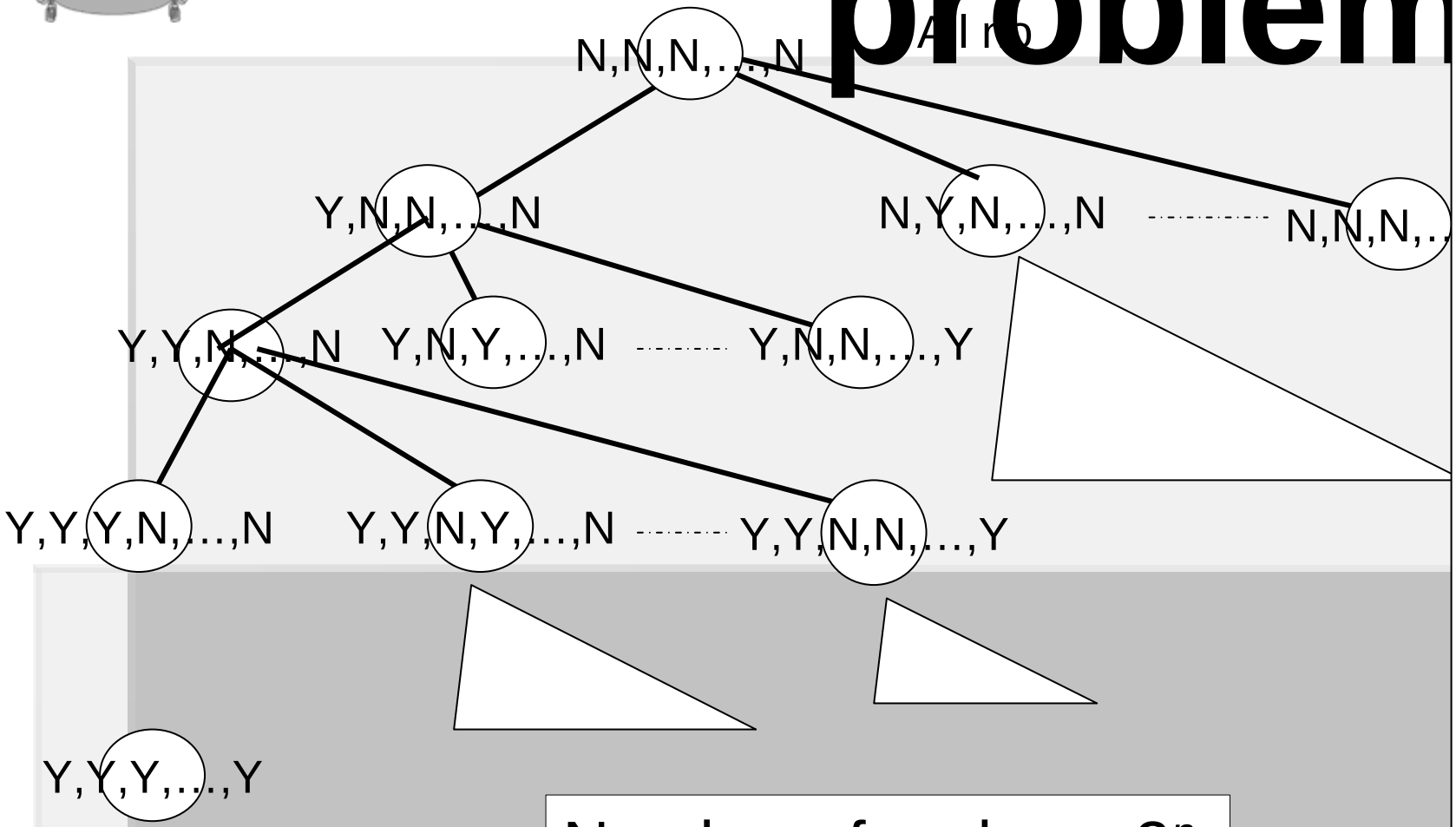
Number of nodes = $2^{n+1} - 1$



All yes

All no

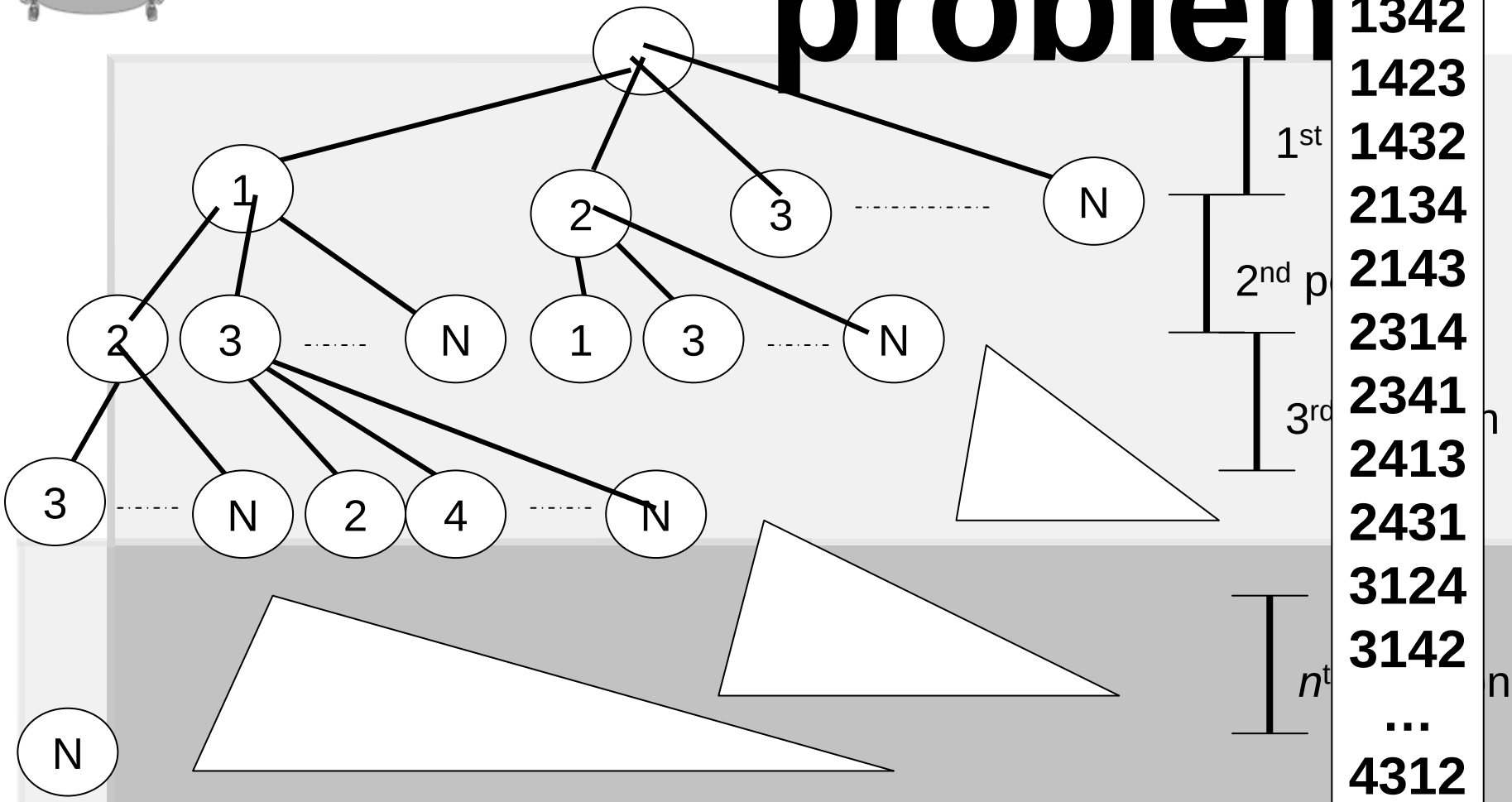
Subset problem



0000
1000
1100
1110
1111
1101
1010
1011
1001
0100
0110
0111
0101
0010
0011
0001

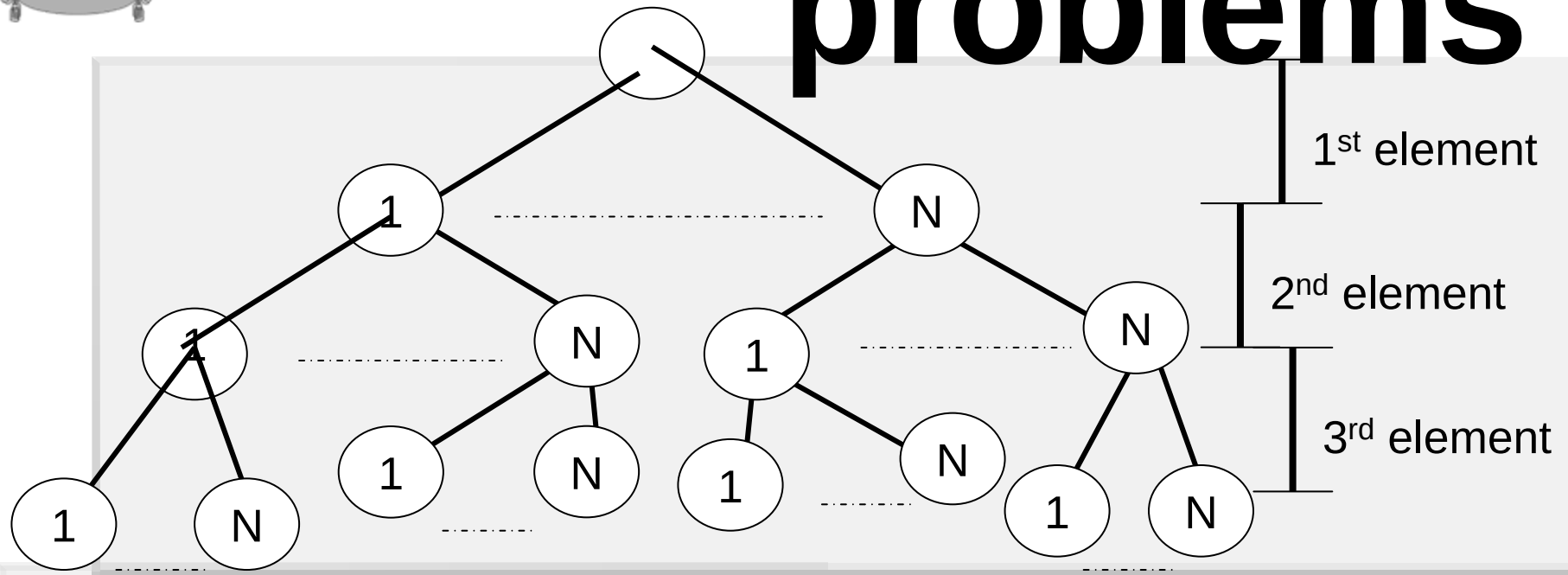
Number of nodes = 2^n

Permutatio
problen



1234
1243
1324
1342
1423
1432
2134
2143
2314
2341
2413
2431
3124
3142
...
4312
4321

Partition problems



Number of nodes = n^n

All 1

All n

Traversal

Three possible ways

- Depth-first technique
- Breadth-first technique
- Best-first technique

Back tracking

Technique

- Depth-first technique
- Keep track and return back when it cannot be branched.

Branch & bound

	1	2	3	4
A	10	7	13	15
B	12	5	16	12
C	14	9	14	20
D	11	7	14	13