

Algorithms

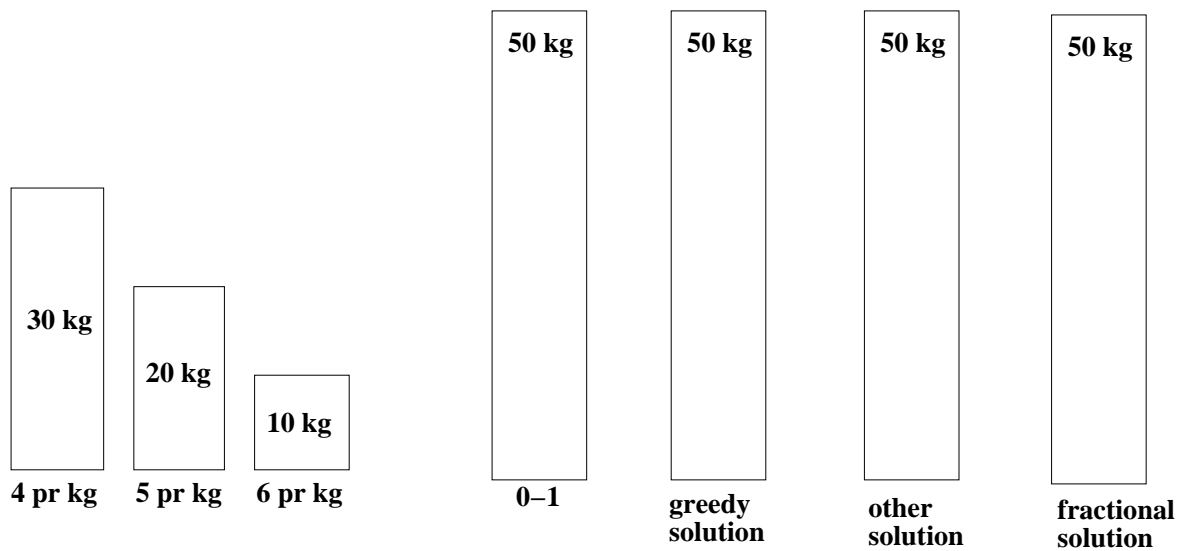
Greedy Algorithms

Overview

- Greedy Approach in General
- Job Scheduling
- Huffman Codes
- Minimum Spanning Trees

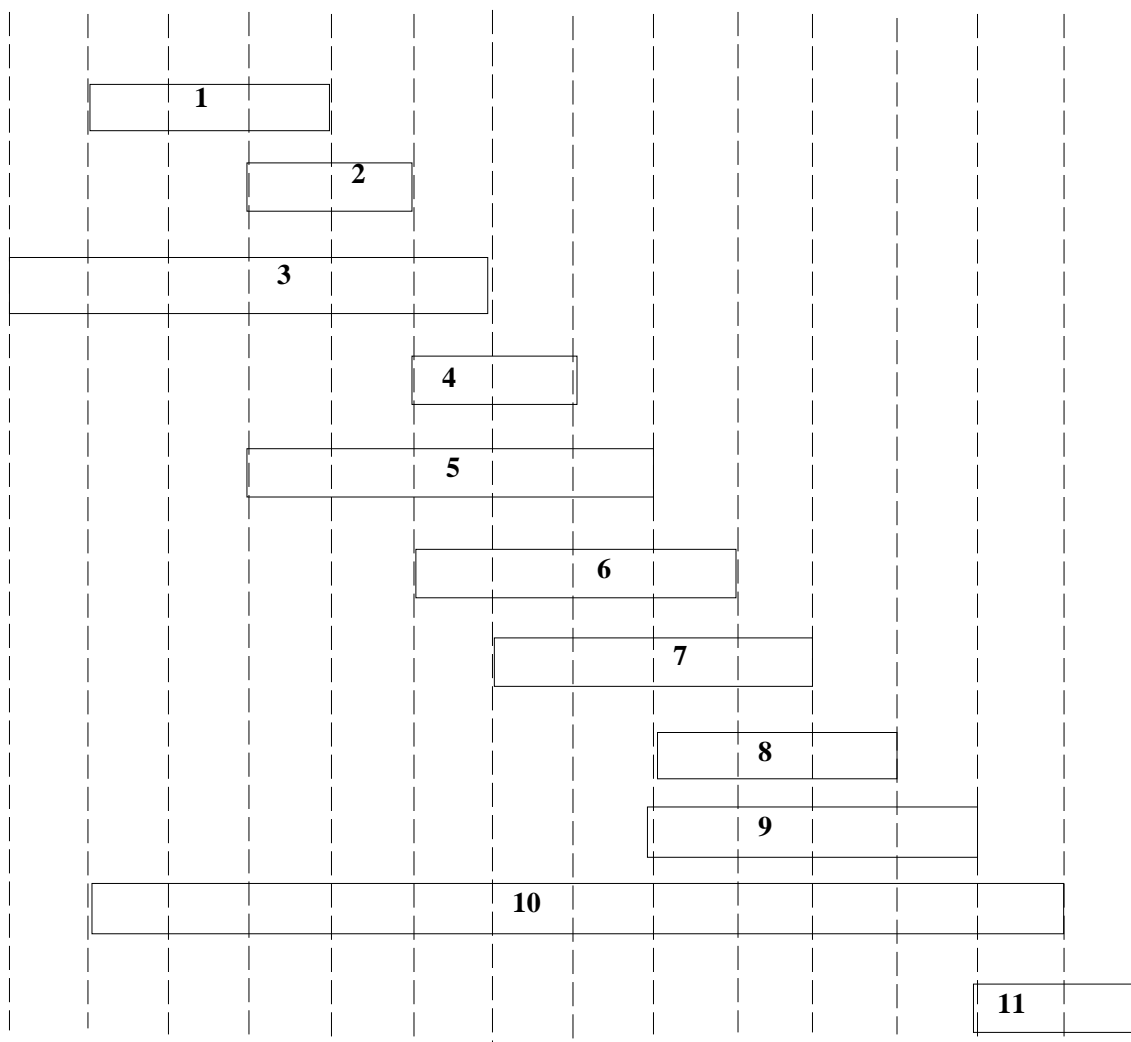
Greedy Approach in General

- Greedy-Choice Property: Optimal solution is derived by making locally optimal decisions.
- Optimal Substructure: Optimal solution contains optimal solutions to subproblems.
- Greedy vs. Dynamic
 - Shortest Path Problem.
 - Knapsack Problem.
 - * 0-1 Knapsack Problem
 - * Fractional Knapsack Problem



Job Scheduling

- Given: n jobs with specified start and end-time.
- Maximize number of jobs that can be carried one on a single machine.

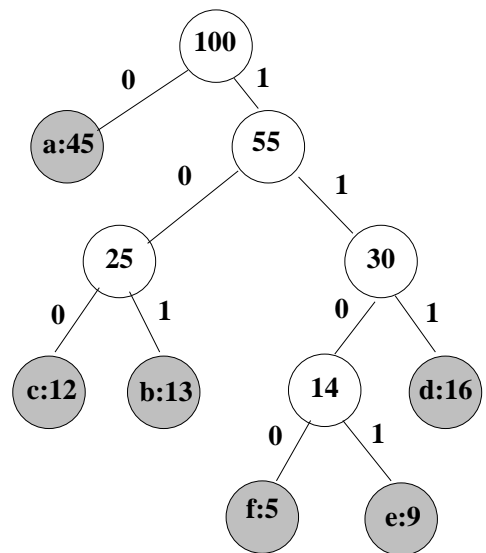
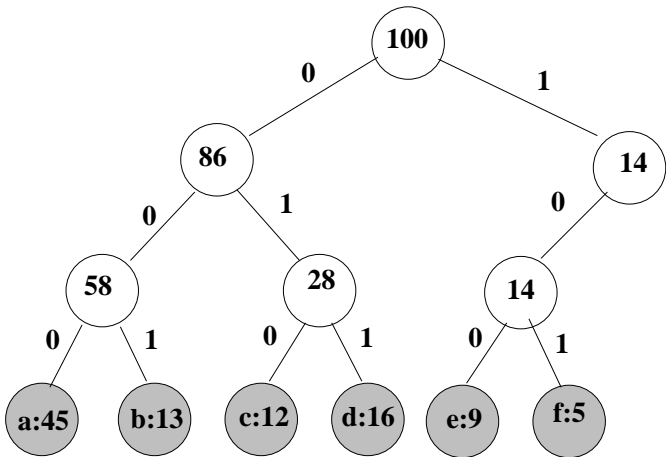


- There is an optimal scheme with the greedy choice of the first activity.
- Optimal substructure property ensures that greedy selections returns optimal solution.

Huffman Codes

	a	b	c	d	e	f
Frequency	0.45	0.13	0.12	0.16	0.9	0.5
fixed-length code	000	001	010	011	100	101
variable-length code	0	101	100	111	1101	1100

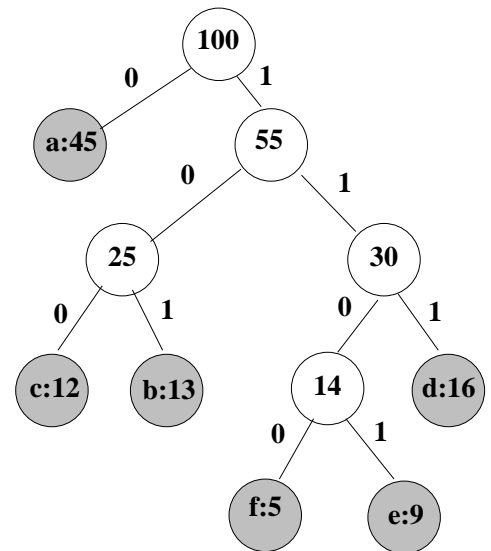
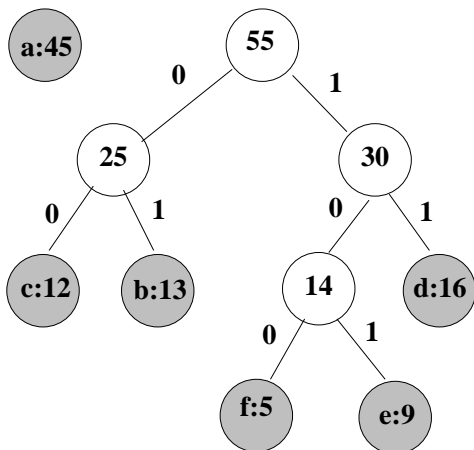
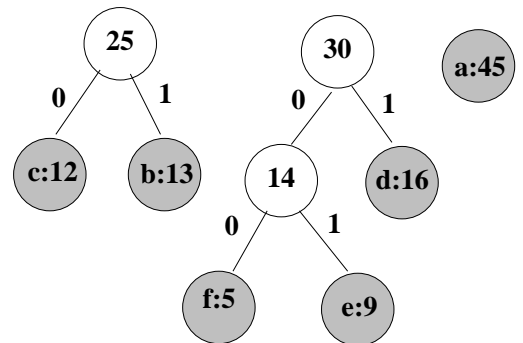
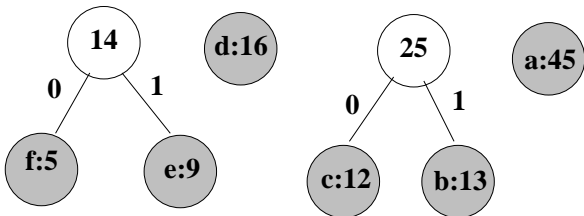
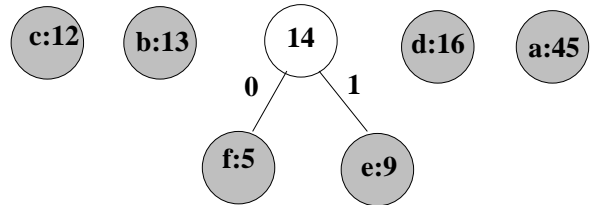
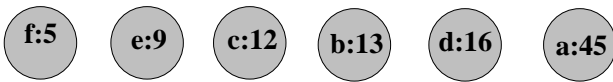
- Prefix code: No codeord is a prefix of another codeord.



Algorithms

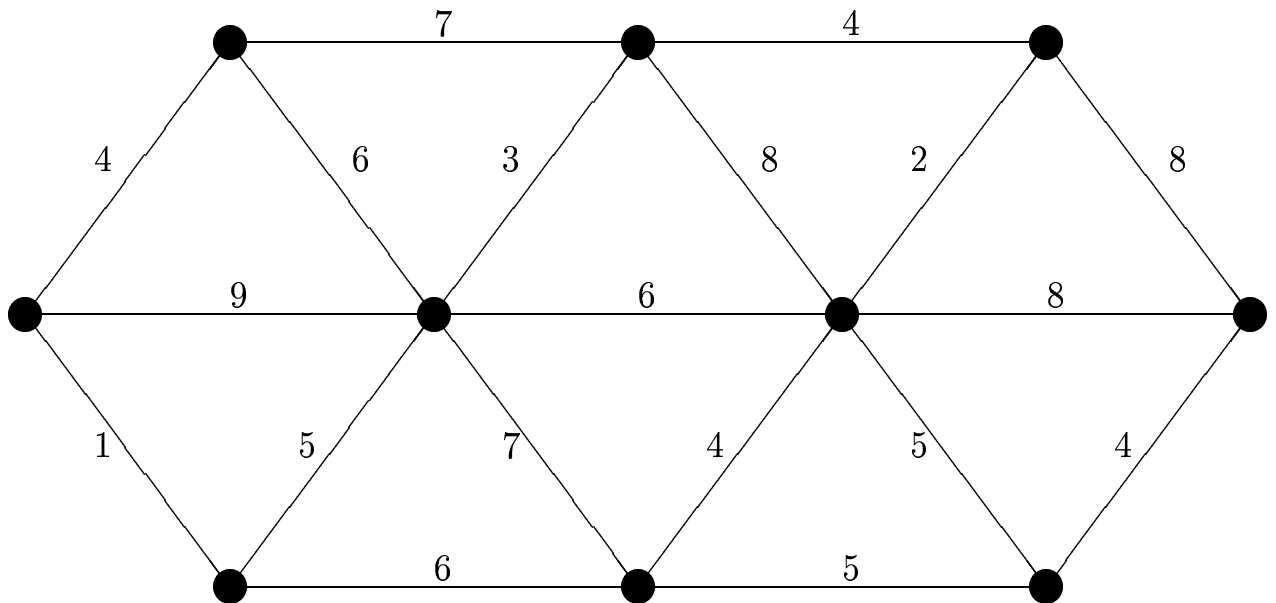
Greedy Algorithms

Huffman Codes Greedy Algorithm



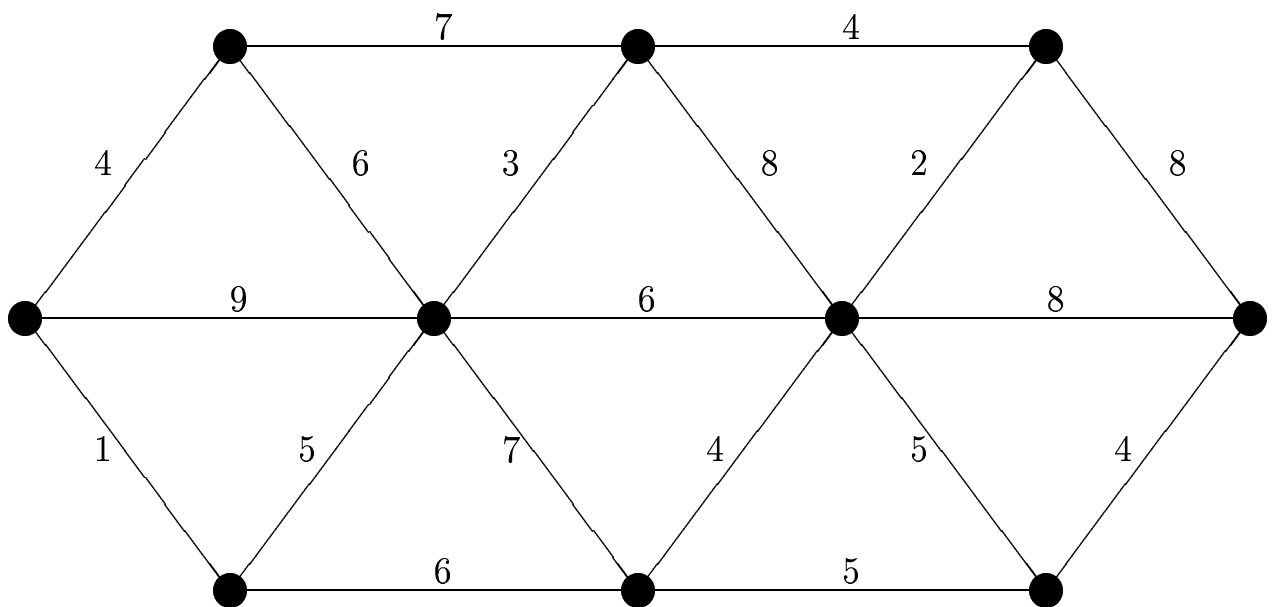
Minimum Spanning Tree - Problem Formulation

- *Given:* undirected, connected, network $G = (V, E, c)$.
- *Find:* a tree T spanning V , and such that its total cost (sum of its edge-costs) is minimized.



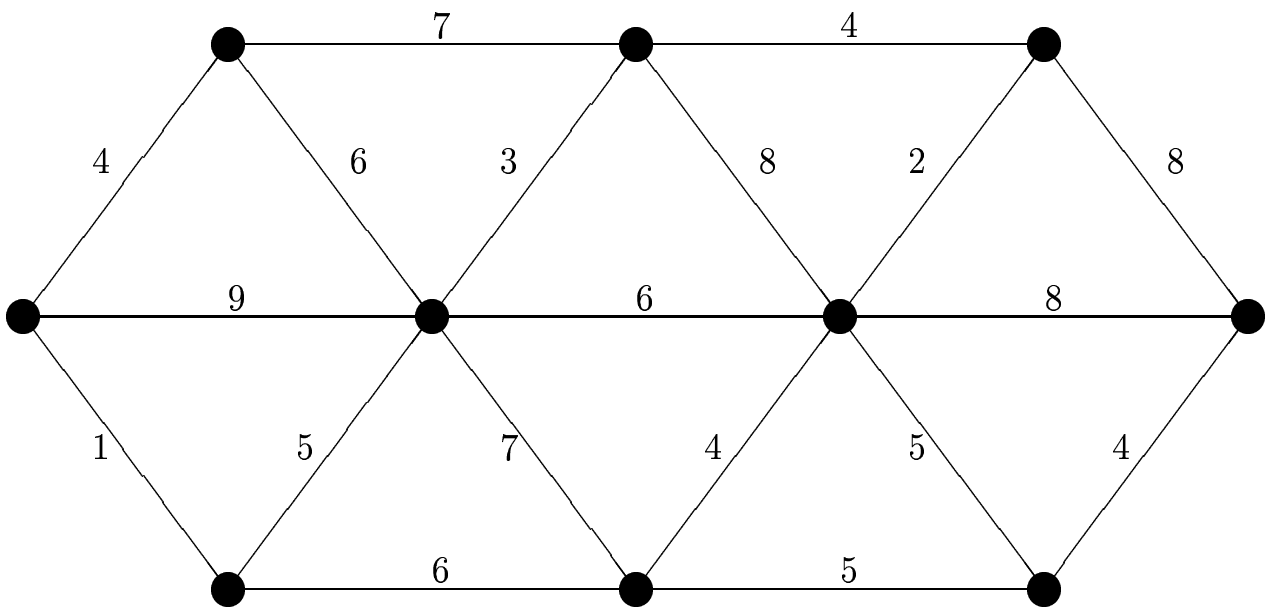
Kruskal's Algorithm

- **Initialize:** F : forest consisting of isolated vertices of G . E_s : set of sorted edges.
- **Terminate:** If F is connected then terminate.
- **Update:** Scan E_s and delete edges with both end-vertices in the same tree. Let e denote the first edge with end-vertices in different trees. Join the trees containing the end-vertices of e . Delete e from E_s . Go to the **Termination**.



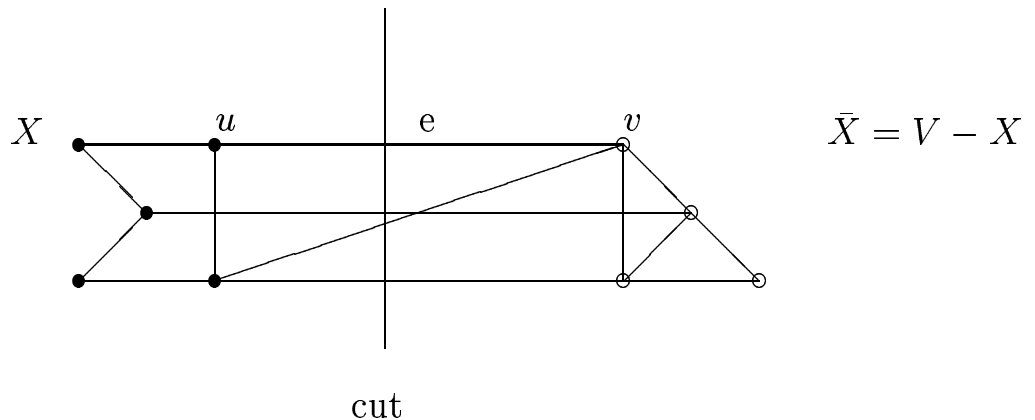
Prim's Algorithm

- **Initialize:** F : a forest consisting of isolated vertices of G . T : arbitrarily chosen tree in the forest.
- **Terminate:** If F is connected then terminate.
- **Update:** Select an isolated vertex v closest to T . Add the shortest edge between v and T to F . Go to the **Termination**.



Main Observation

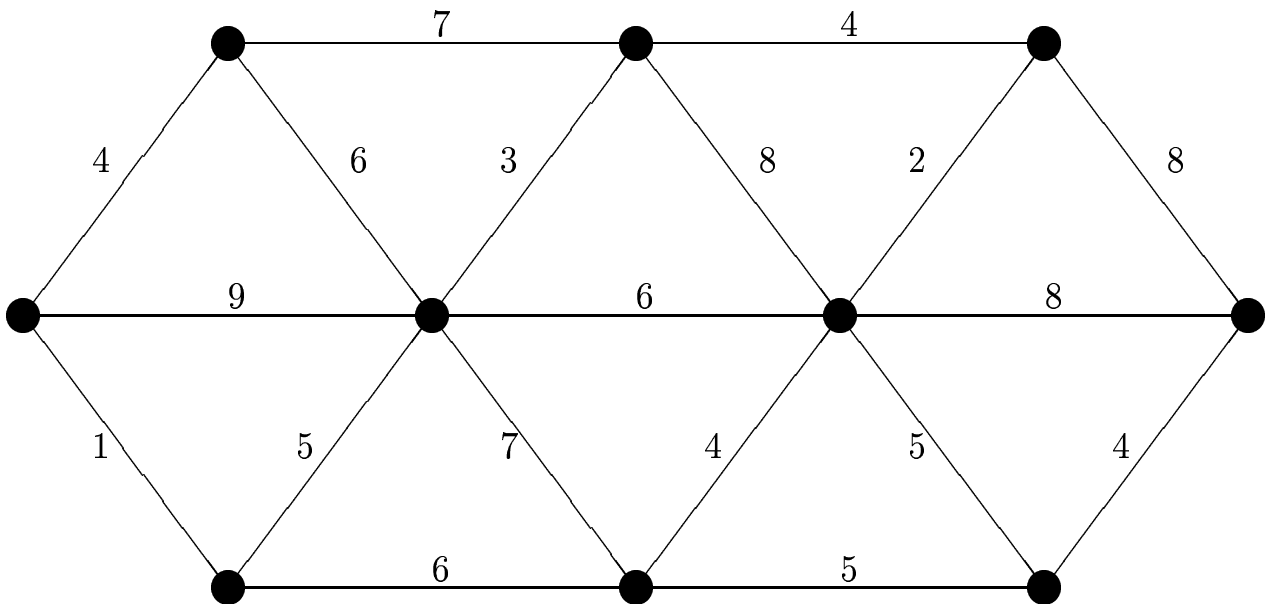
- $C = \{X, \bar{X}\}$ is a *cut* in G : a partition of the vertex set in two parts X and \bar{X} . An edge *crosses* the cut if it has one endpoint in X and the other endpoint in \bar{X}
- e is a minimal cost edge across C .
- at least one MST of G contains e .



- Suppose that e is not in any MST of G .
- Let T be one of the MSTs of G .
- There is a path from u to v in T .
- Let f be the first edge on this path crossing C .
- $T \setminus f \cup e$ is a tree spanning all vertices and $c(T) \geq c(T \setminus f \cup e)$.
- Hence, $T \setminus f \cup e$ is an MST, a contradiction.

Blue Rule

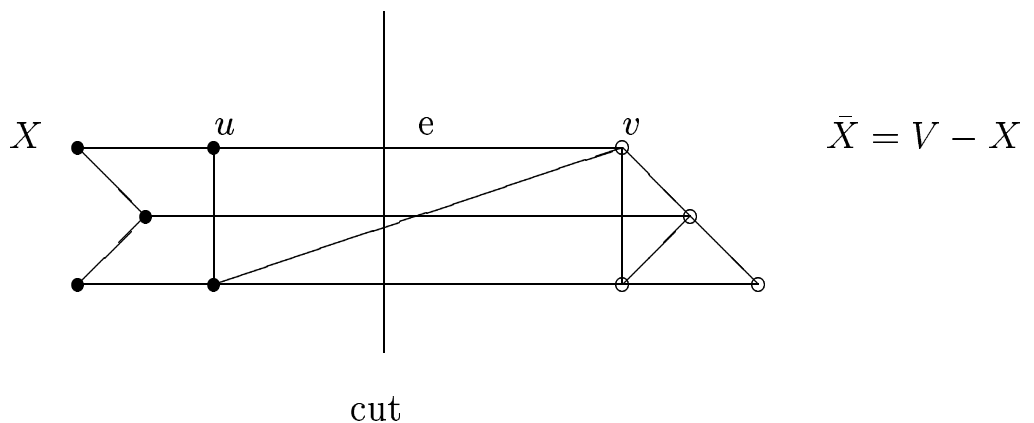
- Select a cut C with no blue edges.
- Among all edges in C , select one of minimum cost and colour it blue.
- Repeat as long as possible.



Blue Rule - Correctness

Every MST can be obtained by repeated applications of the blue rule. Let T be one of (possibly several) MSTs. Keep colouring edges of T as long as the blue rule is not violated.

- If all edges of T have been coloured, there is nothing to prove.
- e : one of the edges in T that cannot be coloured by the blue rule.
 - $T - e$ falls into two parts, and vertices in these two parts form a cut.
 - No edge across the cut can be blue; only edges of T have been coloured blue so far.
 - Since e cannot be coloured blue, it is not of minimum cost among cut-edges.
 - This contradicts the assumption that T is an MST.



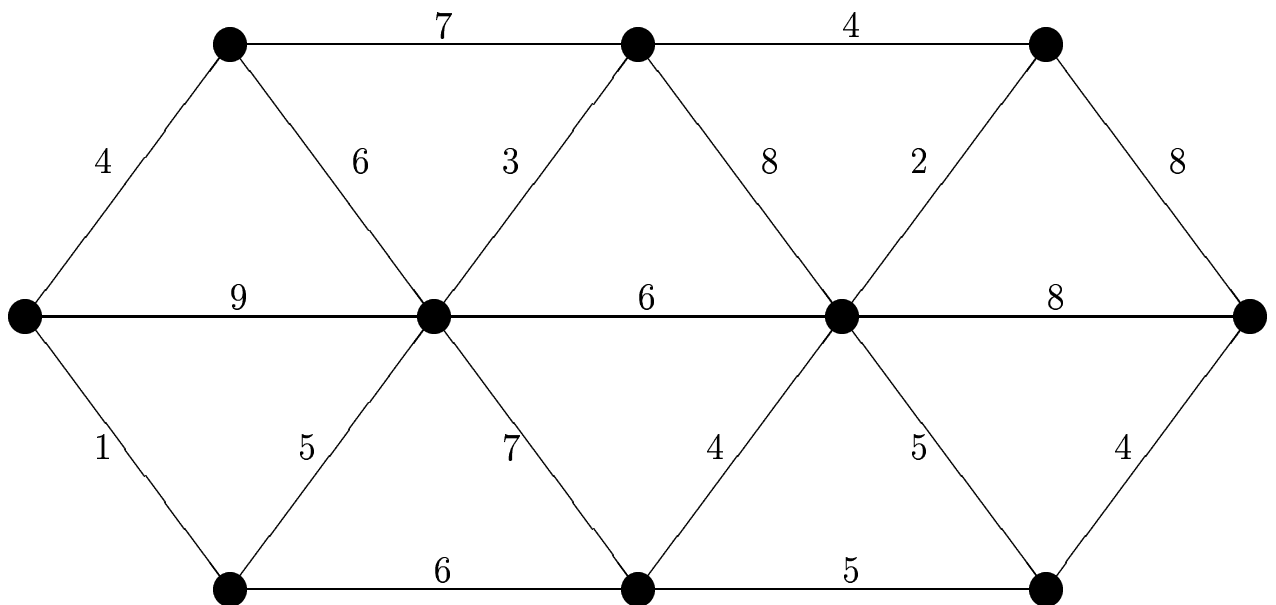
Blue Rule - Correctness

Only MSTs can be obtained by repeated applications of the blue rule.

- Blue rule generates a tree.
- Suppose that at some iteration the blue rule selects an edge e such that e together with previously coloured edges is in no MST.
- The edge set across the cut from which e was selected must contain at least one uncoloured edge e' in some MST T containing all edges chosen before e .
- $c(e') = c(e)$.
- $T + e - e'$ is an MST. It contains e as well as all edges coloured blue before e , a contradiction.

Red Rule

- Select a cycle with no red edge.
 - Among all edges in this cycle, select one of maximum cost and colour it red.
 - Repeat as long as possible.
-
- Every MST can be obtained by repeated applications of the red rule.
 - Only MSTs can be obtained by repeated applications of the red rule.



Blue-Red Rule

- Either select a cut with no blue edge, and colour its minimum cost uncoloured edge blue,
- Or select a cycle with no red edge and colour its maximum cost uncoloured edge red.
- Repeat as long as possible.
- Every MST can be obtained by repeated applications of the red rule.
- Only MSTs can be obtained by repeated applications of the red rule.
- All implementations are based on the blue-colouring.

