2.2 Minimum cost spanning trees

Spanning trees have a number of applications:

• network design (communication, electrical, ...)
• compact memory storage (DNA)
• diffusion of secret messages
• ...

2.2.1 Problem and Prim's algorithm

Example

Design a communication network so as to connect $n$ cities (offices) at minimum total cost.

Model: Graph $G = (N, E)$ with $n = |N|$, $m = |E|$ and a cost function $c : E \to c_e \in \mathbb{R}$, with $e = [v, w] \in E$
Properties:

1) Each pair of cities must communicate \( \Rightarrow \) connected subgraph containing all the nodes

2) Minimum total cost \( \Rightarrow \) subgraph with no cycles

Problem

Given an undirected graph \( G = (N, E) \) and a cost function, find a spanning tree of minimum total cost

\[
\min_{T \in X} \sum_{e \in T} C_e
\]

where \( X \) is the set of all spanning trees of \( G \)
Theorem

A. Cayley (1889)

A complete graph with \( n \) nodes has \( n^{n-2} \) spanning trees, for \( n \geq 1 \).

Esempio

\( n = 3 \)

Recall: a tree with \( n \) nodes has \( n - 1 \) edges
Example Prim's algorithm

Some feasible solutions:

$S = \{1\}$

$T = \emptyset$

$S = \{1, 2\}$

$T = \{\{1, 2\}\}$
\[ S = \{1, 2\} \]

\[ S = \{1, 2, 5\} \]

\[ S = \{1, 2, 3, 5\} \]

\[ S = N \]

\[ \text{costo: 9} \]
Prim's algorithm

Connected graph $G = (N, E)$ with a cost function

Set of edges $T \subseteq E$ such that $G_T = (N, T)$ is a spanning tree of $G$

BEGIN

T := $\emptyset$; S := {1};

WHILE $|T| < n-1$ DO /* a tree with $n$ nodes has $n-1$ edges */

find $[v,h] \in \delta(S)$ of min cost, with $v \in S$ e $h \in N \setminus S$;

T := T $\cup$ {[v,h]};

S := S $\cup$ {h};

END-WHILE

END
Prim algorithm is a *greedy algorithm* : at each step a min cost edge is selected among those in the cut induced by the current set of nodes $S$ without reconsidering previous decisions.
2.2.2 $O(n^2)$ implementation

Data structure:

• $k =$ number of edges selected so far

• Subset $T \subseteq E$ of selected edges

• Subset $S \subseteq N$ of nodes incident to the selected edges

....
• $C[j] = \min \{ c_{ij} : i \in S \} \quad \forall j \notin S$; \quad \text{se} \ [i,j] \notin E, \ c_{ij} = +\infty

\begin{align*}
\text{pred}[j] &= \begin{cases} 
\text{“predecessore” di } j \text{ nell’albero minimo,} & \forall j \in S \\
\min \{ c_{ij} : i \in S \} & \forall j \notin S
\end{cases} 
\end{align*}
$\text{cut } \delta(S) = \{[1,3], [1,4], [1,5], [2,3], [2,5]\}$

$S = \{1, 2\}$

$T = \{[1,2]\}$

$\text{pred}[2] := 1$

$C[3] := c_{23} = 3$ (since $c_{23} < c_{13}$)

$\text{pred}[3] := 2$

$C[5] := c_{15} = 2$ (since $c_{15} = c_{25}$)

$\text{pred}[5] := 1$

$C[4] := c_{14} = 6$ (since $[2,4]$ does not exist)

$\text{pred}[4] := 1$


$h := 5; \ \text{pred}[h] := 1; \ S := S \cup \{5\}; \ T := T \cup \{[1,5]\}$

etc…
\(O(n^2)\) version of Prim's algorithm

BEGIN

\[ T := \emptyset; \ S := \{1\}; \quad /* \text{initialization} */ \]

FOR \( j := 2 \ \text{TO} \ n \ \text{DO} \quad /* \forall \ \text{nodes} \ j \notin S \ */ \]
\[
T := T \cup \{[1, j]\}; \quad \text{if} [1,j] \notin E, \ c_{1j} = +\infty \\
C[j] := c_{ij}; \quad \text{pred}[j] := 1; \]

END-FOR

FOR \( k := 1 \ \text{TO} \ n-1 \ \text{DO} \quad /* \text{select} n - 1 \ \text{edges of the tree} */ \]
\[
\text{min} := +\infty; \\
\text{min} := C[j]; \quad \text{h} := j; \quad \text{END-IF} \\
S := S \cup \{h\}; \quad T := T \cup \{[\text{pred}[h], h]\}; \quad /* \text{extend} \ S \ \text{and} \ T */ \\
\text{END-FOR} \\
\text{END-FOR} \\
\text{END}
\]
Example

\[ S = \{1\} \]
\[ T = \emptyset \]
\[ C = (+\infty, 1, 4, 6, 2) \]
\[ \text{pred} = (1, 1, 1, 1, 1, 1) \]

\[ S = \{1, 2\} \]
\[ T = \{[1,2]\} \]
\[ C = (+\infty, 1, 3, 6, 2) \]
\[ \text{pred} = (1, 1, 2, 1, 1) \]

\[ S = \{1, 2, 5\} \]
\[ T = \{[1,2], [1,5]\} \]
\[ C = (+\infty, 1, 2, 4, 2) \]
\[ \text{pred} = (1, 1, 5, 5, 1) \]

etc...
A minimum spanning tree consists of the $n-1$ edges $[\text{pred}[j], j]$ con $j = 2, ..., n$.

**Example:** Since $\text{pred} = (1, 1, 5, 5, 1)$ a spanning tree consists of the edges: $[1,2], [5,3], [5,4]$ and $[1,5]$

![Minimum spanning tree diagram]

**cost:** 9
BEGIN
<initialization>
FOR j:=2 TO n DO
   (...)
END-FOR
FOR k:=1 TO n-1 DO
   FOR j:=2 TO n DO
      (...)
   END-FOR
END-FOR
END

1. Initialization requires $O(n)$
2. They are executed $n - 1$ times in the external cycle
3. The two internal FOR cycles require $O(n)$ each

Overall complexity: $O(n^2)$
For sparse graphs, where $m << n(n-1)/2$, a more sophisticated data structure leads to an $O(m \log n)$ complexity.
Prim's algorithm is exact, i.e., it is guaranteed to yield a minimum spanning tree of $G$, regardless of the choice of the first node and of the minimum cost edge selected in case of ties in $\delta(S)$.

Few optimization problems admit exact greedy algorithms!

We will show that each selected edge belongs to a minimum spanning tree.
Cost-decreasing edges

Given a spanning tree $T$, an edge $e \not\in T$ is cost decreasing if when it is added to $T$ it creates a cycle $C \subseteq T \cup \{e\}$ and there exists an edge $f \in C \setminus \{e\}$ such that $c_e < c_f$

$$c(T \cup \{e\} \setminus \{f\}) < c(T) = \sum_{e' \in T} c_{e'}$$
Property of optimum trees

If a spanning tree $T^*$ is of minimum total cost, no cost-decreasing edge exist.

$$c_e \geq c_f \text{ for each } f \in C \setminus \{e\}$$

Otherwise we could decrease the cost of $T^*$ by exchanging the cost-decreasing edge $e$ with any $f$ of $C$ with $c_e < c_f$!
Proposition

Given $S \subseteq N$ and a minimum cost edge $e = [v, h] \in \delta(S)$, then there exists a minimum spanning tree containing $e$.

Dim. By contradiction: Let $T^* \subseteq E$ be a min spanning tree with $e \notin T^*$

Adding $e \Rightarrow$ cycle $C$

Let $f \in \delta(S) \cap C$

If $c_e = c_f$ then $T^* \cup \{e\} \setminus \{f\}$ is optimum since same cost of $T^*$

If $c_e < c_f$, $e$ is a cost-decreasing edge and hence $T^*$ is not minimum!
2.2.4 **Kruskal algorithm**

**input**

\[ G = (N, E) \text{ and a cost function} \]

**output**

Subset of edges \( T^* \subseteq E \) s.t. \( G_{T^*} = (N, T^*) \) is a spanning tree of \( G \)

**Idea:** sort the edges in order of non-decreasing cost and select the edges which do not create cycles.
Esempio

[1,2], [2,5], [1,5], [3,5], [2,3], [4,5], ...

\[1 \quad 2 \quad 2 \quad 2 \quad 3\]

\[\text{costo: } 9\]
Kruskal's algorithm

BEGIN
sort the edges of G in order of non-decreasing cost;
T* := ∅;
WHILE |T*| < (n-1) DO
    select an edge e ∈ E of minimum cost;
    E := E \ {e};
    IF T* ∪ {e} is acyclic THEN T* := T* ∪ {e};
END-WHILE
END
Complexity

• Order \( m \) edges: \( O(m \log m) \)

\[
\log m < \log n^2 = 2\log n \quad \Rightarrow \quad O(m \log n)
\]

• Check that an edge creates a cycle: in constant time by checking that its two nodes belong to two different connected components (c.c.)

Each update of the node labels for the c.c.: \( O(n) \)

\[\Rightarrow \text{overall complexity: } O(m \log n + n^2)\]

N.B. Complexity can be reduced by using a more sophisticated data structure
A tree $T$ is **minimum** if and only if **no cost-decreasing edges exist.**

($\Rightarrow$)

If a cost-decreasing edge exists, $T$ is not optimum (property of minimum trees)

($\Leftarrow$)

If **no cost-decreasing edges exist**, then $T$ is optimum

By exchanging edges we can transform the optimum $T^*$ found by Prim algorithm into $T$ without modifying the total cost, hence $T$ is optimum.
Kruskal's algorithm is exact

Each edge $e \notin T^* \ (T^*$ is the resulting spanning tree) has been discarded because it would have created a cycle.

Moreover $c_e \geq \ldots$ of all edges of that cycle, since the edges are considered in order of non-decreasing cost.

$\Rightarrow$ the resulting spanning tree satisfies the optimality condition and hence is of minimum total cost.

no cost-decreasing edge exists
Optimality test

The optimality condition allows us to verify whether a given spanning tree $G_T$ is optimum:

$G = (N, E)$

$G_T = (N, T)$

$c(T) = 9$

It suffices to check that each $e \in E \setminus T$ is not a cost-decreasing edge.
2.2.5 **Indirect application**

Optimal message passing:

Given a communication network $G = (N, E)$ whose edges $[i, j] \in E$ correspond to the pairs of “nodes” that can directly communicate.

Let $p_{ij}$, $0 \leq p_{ij} \leq 1$, be the probability a secret message is intercepted along edge $[i, j] \in E$

How to pass a secret message to all the “nodes” of $G$ while minimizing the probability of interception?
Minimize the probability of interception (along an edge) ⇐

Maximize the probability of non-interception

\[
\max \prod_{[i,j] \in T} (1 - p_{ij})
\]

\(T\) is spanning tree

- Diffusion to all nodes ⇒ connected
- acyclic to avoid redundancy and a higher probability of interception
Applicando una funzione monotona crescente, ad esempio log(.), non cambiano le soluzioni ottime (solo il valore)

\[ \max \log( \prod_{[i,j] \in T} (1 - p_{ij})) \equiv \max \sum_{[i,j] \in T} \log(1 - p_{ij}) \]

Si adattano in modo ovvio gli algoritmi di Prim e Kruskal.