9\18\15

Combinatorial Optimization

Minimum Spanning Tree Problem

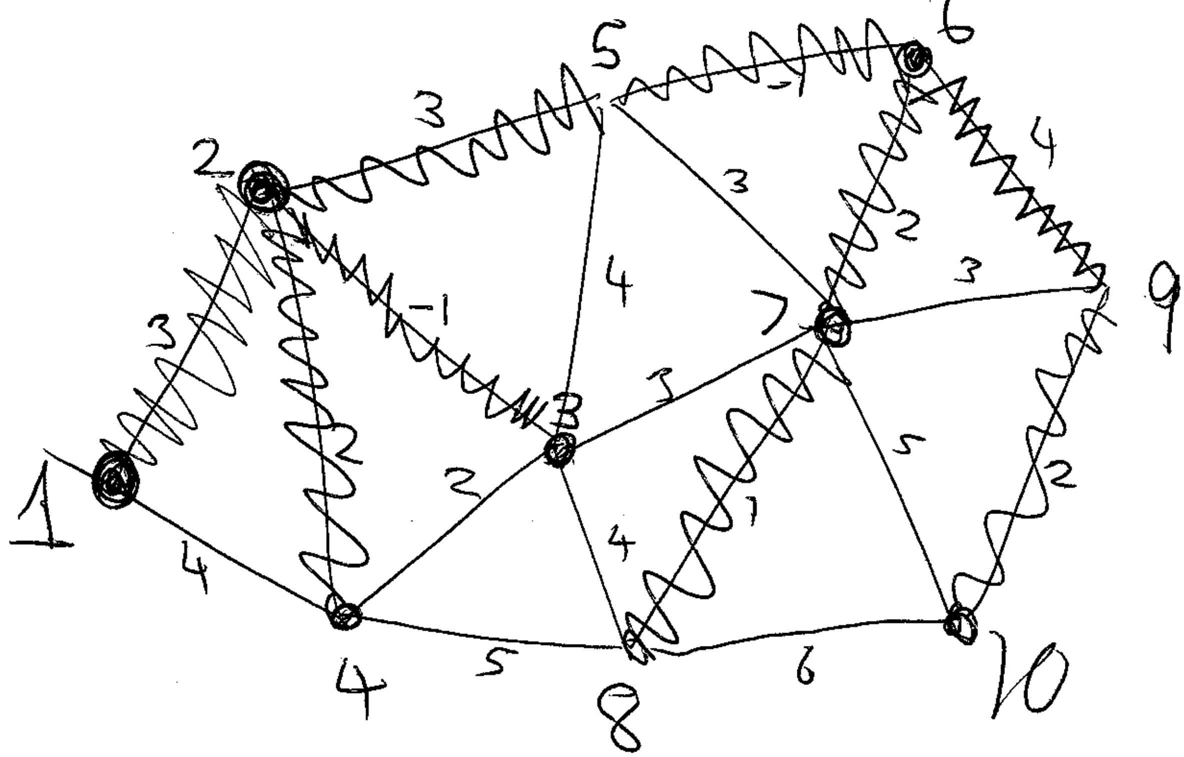
Given a graph $G = (V, E)$ and edge weights X_e; $e \in E$ find a minimum weight spanning tree;

Greedy (Kruskal) Algorithm

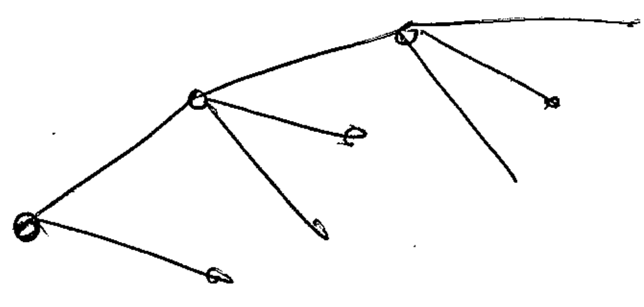
Choose $e_1, e_2, \ldots, e_{n-1}$ $\qquad n = |V|$

s.t. e_{j+1} is the cheapest edge that does not make a cycle with $e_1, \ldots, e_j$

Prim-Dijkstra Algorithm



In general, we have a tree T



To which we add the cheapest edge leaving T.

More general algorithm

Suppose we have selected

$F_k = \{e_1, e_2, \dots, e_k\}$ and $T_1, T_2, \dots, T_{n-k}$

are components induced by F_k

Choose <u>any</u> T_i and add the

cheapest edge leaving T_i — e_{k+1}

Repeat.

Kruskal: choose i so that cheapest leaving T_i is cheapest overall

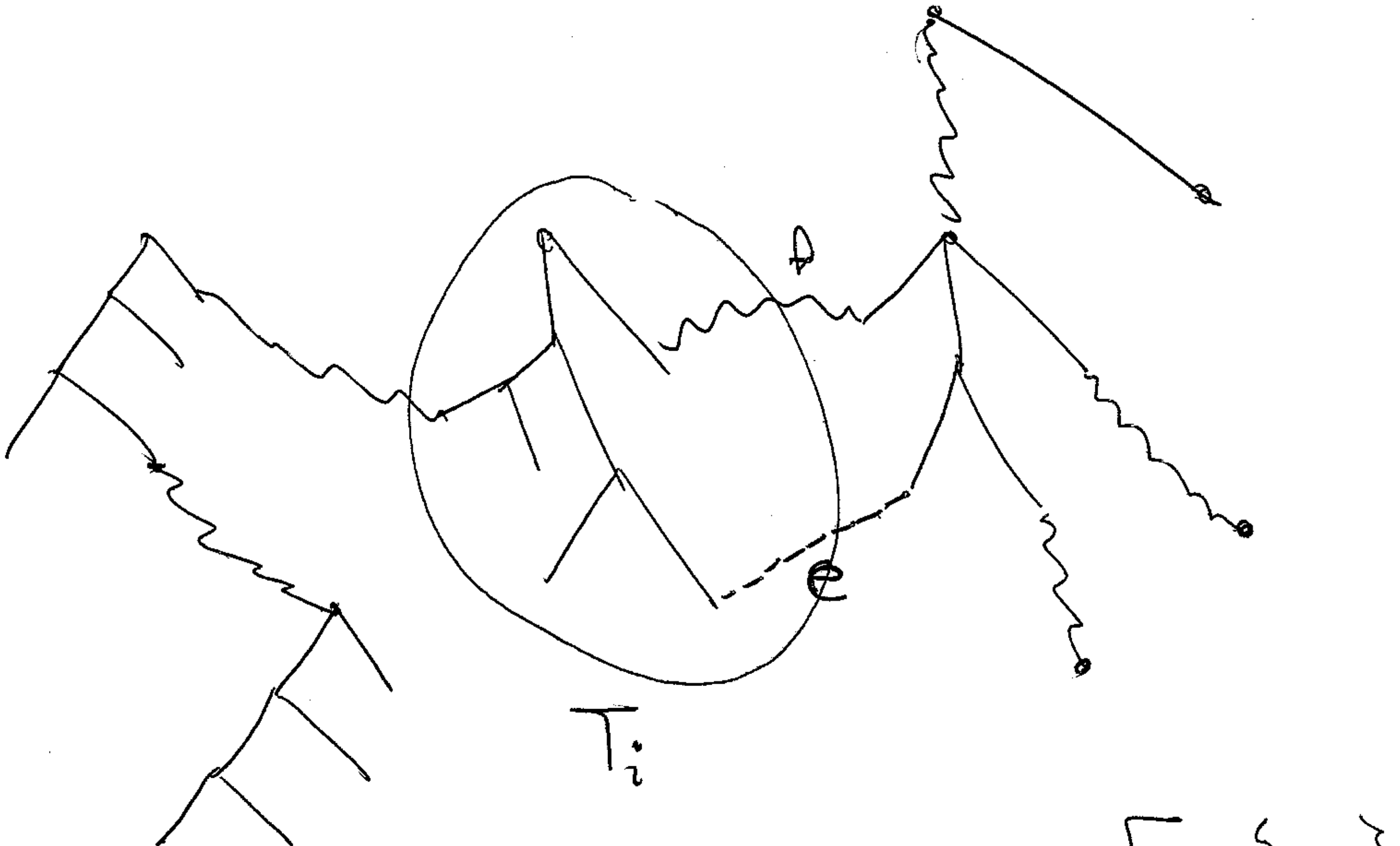
Prim: Always choose T_1

Claim At all times $\exists$ a minimum length tree that contains all of the edges chosen so far $\Rightarrow$ algorithm is correct.

Proof

By induction on k

$k = 0$. $E_0 = \emptyset$



$F_k = \{-\}$

By induction we can add edge to F_k to give a minimum length tree T_k^* additional

Now let e = minimum length edge leaving T_i

Case 1 : $e \in T_k^*$. $T_{k+1}^* = T_k^*$

Case 2 : $e \notin T_k^*$: $\exists f \in T_k^*$ in a cycle with e . $X_e \leq X_f$
f leaves T_i

$$T^*_{k+1} = T^*_k + e - f$$

$$\text{length}(T^*_{k+1}) \leq \text{length}(T^*_k)$$

$$=$$

Shortest Paths

Given a digraph $G = (V, E)$ with lengths on arcs, find shortest paths between vertices.

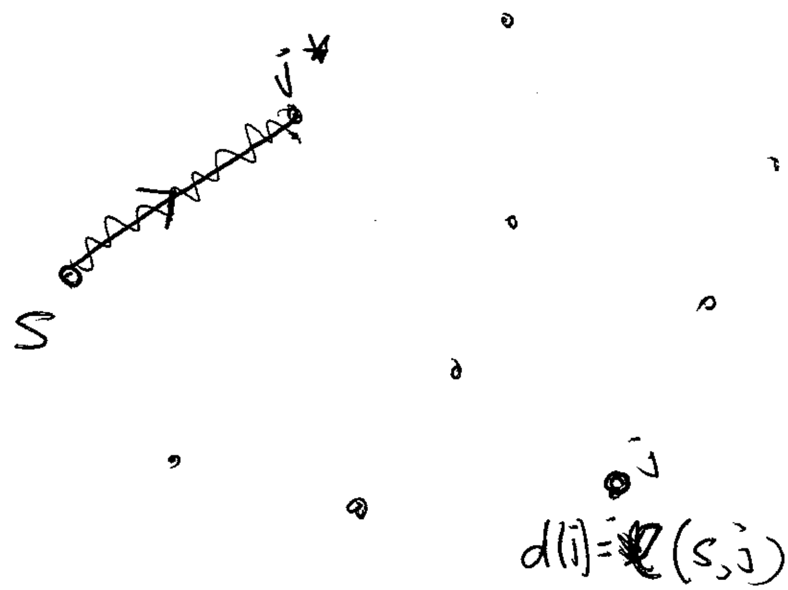
Let $\ell(e)$ = length of edge (arc) e.

~~Goal~~

For a path P: $\ell(P) = \sum_{e \in P} \ell(e)$.

Case 1: $\ell(e) \geq 0, \forall e$.

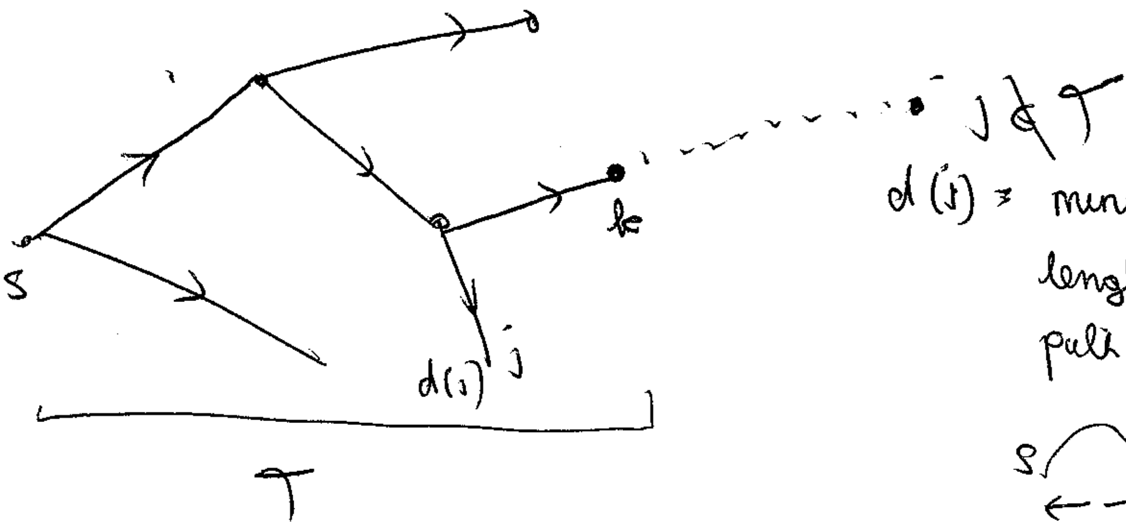
Find a shortest path from vertex s to all other vertices.



Choose j^* to minimize $d(j)$

Add arc (s, j)

General stage:

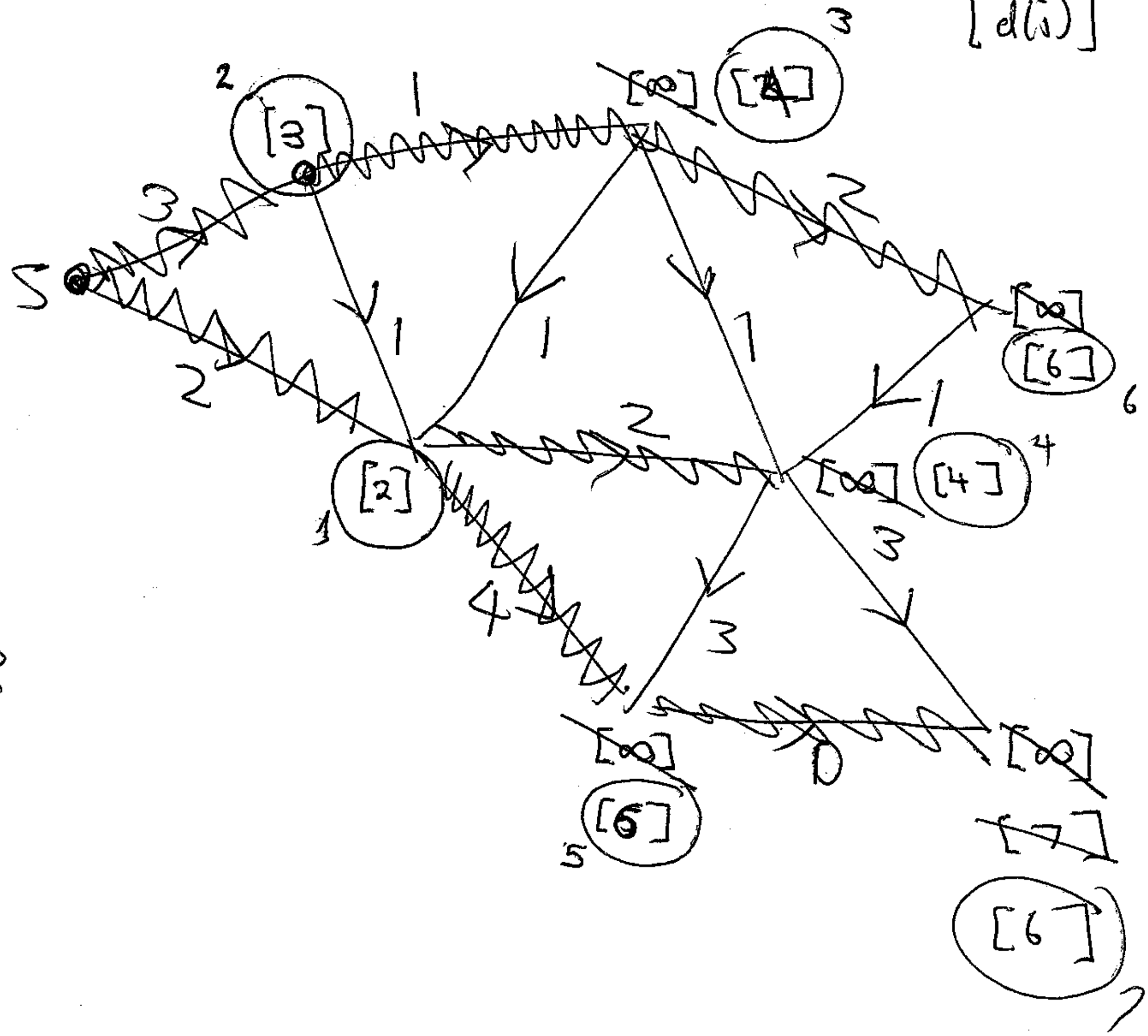


$d(j)$ = minimum length of a path that goes

Tree paths are shortest paths

$j \in T$ $d(j)$ = length of shortest path

Now choose $j^* \notin T$ to minimize $d(j)$ and add arc to tree



$T = \{s\}$

$T = \{s, 1\}$

⋮