The Classes P and NP

Assignment 2

Francisco Palomo Lozano

francisco.palomo@uca.es

Computational Complexity

Department of Computer Science

Steiner trees

- In this task we will solve the Steiner tree problem (ST)
  - ST generalizes the minimum spanning tree problem (MST)
  - ST is an optimization problem
  - Its decision version is in NP
- MST is also an optimization problem
  - Prim and Kruskal are the main algorithms in this domain
  - MST $\in$ PF, as these are polynomial-time algorithms
  - Its decision version is in P

Approach

We will design a naïve algorithm for ST, just to illustrate how inefficient it can be. Although much better algorithms exist, unfortunately, no polynomial algorithm is known for this problem.

Generating combinatorial objects

Basic combinatorial objects

1. Permutations
2. Combinations

Exercise

1. Write a function to generate the "next" k-combination
2. Write a program to show the k-combinations of n, $0 \leq k \leq n$

Implementing Prim's algorithm

Prim$(p, n) \rightarrow S$

$C \leftarrow \varnothing$
for $j \leftarrow 2$ **to** n
  $C \leftarrow C \cup \{j\}$
  $c[j] \leftarrow 1$
  $d[j] \leftarrow p[1, j]$
$S \leftarrow \varnothing$
while $C \neq \varnothing$
  $k \leftarrow \text{select}(C, d)$
  $C \leftarrow C - \{k\}$
  $S \leftarrow S \cup \{\{c[k], k\}\}$
  $\text{update}(c, d, C, k, p)$

select$(C, d) \rightarrow k$

$v \leftarrow \infty$
for all $j \in C$
  if $d[j] < v$
    $v \leftarrow d[j]$
    $k \leftarrow j$

update$(c, d, C, k, p) \rightarrow (c, d)$

for all $j \in C$
  if $p[k, j] < d[j]$
    $c[j] \leftarrow k$
    $d[j] \leftarrow p[k, j]$

Computing an ST

1. Build the subgraph, G, induced by the mandatory vertices
2. Set the minimum to ∞
3. For every combination of optional vertices
   1. Extend G with the selected optional vertices and their edges
   2. Compute the minimum spanning tree, T, of the extension
   3. If the weight of T, w, is less than the minimum
      - Record w as the new minimum
4. Return the minimum

Algorithm complexity

1. Analyze the worst-case time
2. Design a parametric graph for experimentation
3. Conduct experiments for $n = 0$ to 20 optional vertices
4. Plot the experiment times versus n
5. What do you observe?