THE MINIMAL SPANNING TREE (MST) AND ITS VARIANTS

- Consider a connected undirected graph G with costs c_{ij} associated with its links (x_i, x_j). Of the many spanning trees of G that may be possible, we want to find the one for which the sum of the costs is minimum.
- This problem arises where the vertices are terminals of an electric network that have to be connected together and one wants to use as short a length of wire as possible.
- If the vertices represent towns to be joined with a pipeline network, the shortest length of pipe that can be used is also given by the minimal spanning tree of the corresponding graph.
- Numerous other applications.
- Algorithms by Kruskal and Prim.

KRUSKAL’S ALGORITHM

Step 1. Start with a completely disconnected graph T of *n* vertices.

Step 2. Order the links of G in ascending order of cost.

Step 3. Starting from the top of this list, add links into T provided that this addition does not create a cycle in T.

Step 4. Repeat Step 3 until (*n* - 1) links have been added. T is then the MST of G.

- This is an example of a “greedy” algorithm. Most greedy algorithms are suboptimal, but this is the rare exception.

WORST CASE COMPLEXITY

- The computationally most expensive step is Step 2. For a graph with m links, Step 2 would require $mlog_2m$ operations to produce a complete list of links in ascending order of cost.

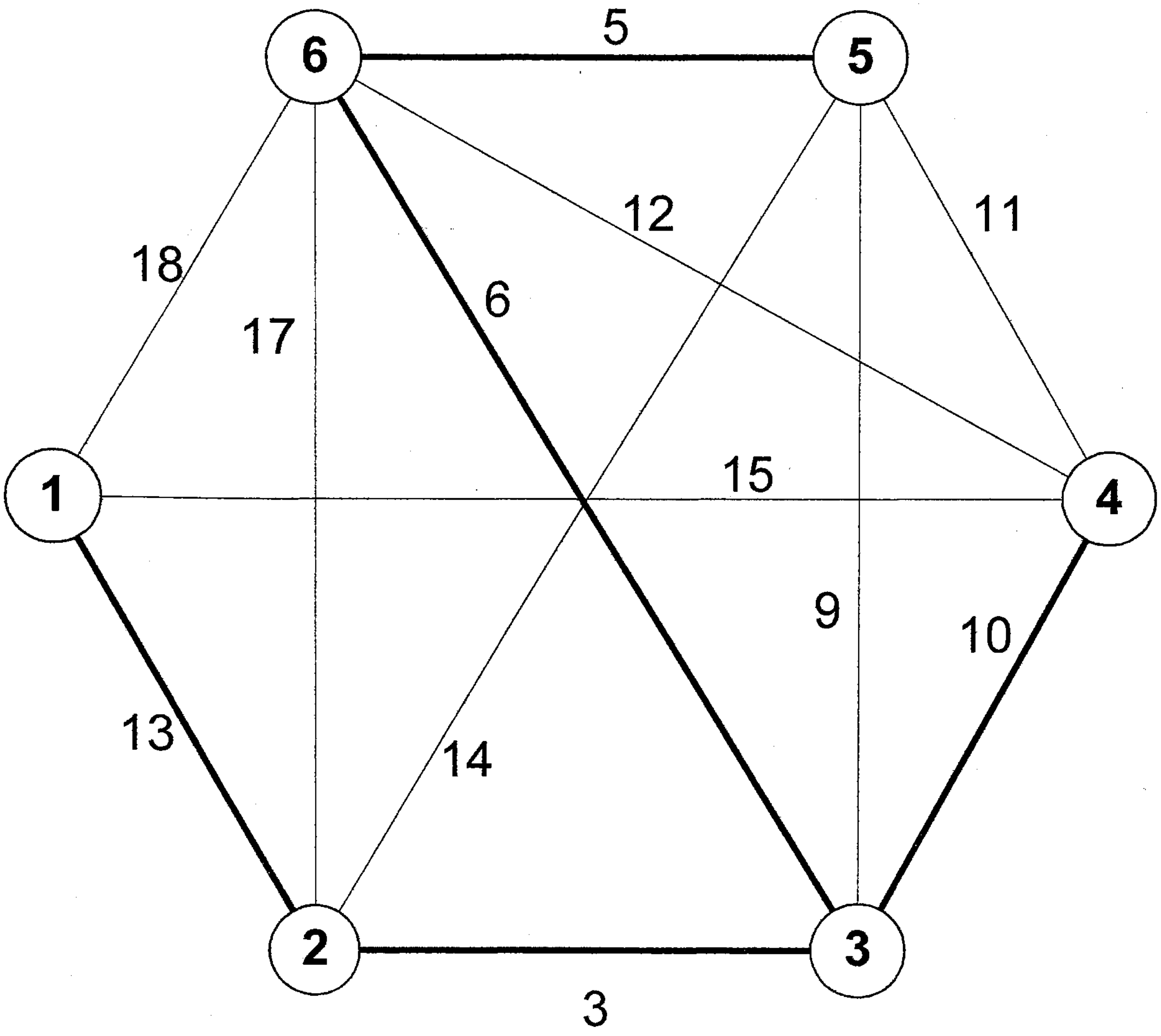
- Note, one might not want or need the complete list since it is quite likely that the n - 1 feasible links forming the MST may be found after examining only the top $r < m$ of the links in the list. This suggests that the sorting procedure used at Step 2 should be a multipass routine in which at the end of the $p^{\underline{th}}$ pass the top p links are correctly placed.

- Since Step 2 represents most of the work, in the worst case, the number of operations required is on the order of $n^2\log_2 n$ since $m = n(n - 1)/ 2$ for a complete graph.

PRIM'S ALGORITHM

- Let T_0 denote the tree which consists of the single vertex $\{1\}$. For $j = 1, 2, \ldots, n\text{-}1$, let T_j be obtained by adjoining to T_{j-1} an edge a_j whose length is minimal in the class of all edges with one end in T_{j-1} and one end not in T_{j-1}. Then T_{n-1} is a tree of minimal length.
- If distances are first sorted into ascending order, the procedure requires $O(n^2 \log_2 n)$ operations.
- Nijenhuis and Wilf present an improved algorithm which costs $O(n^2)$ operations in the worst case.

AN EXAMPLE



Kruskal's Algorithm:

- add arc from 2 to 3;
- add arc from 5 to 6;
- add arc from 3 to 6;
- add arc from 3 to 4;
- add arc from 1 to 2;

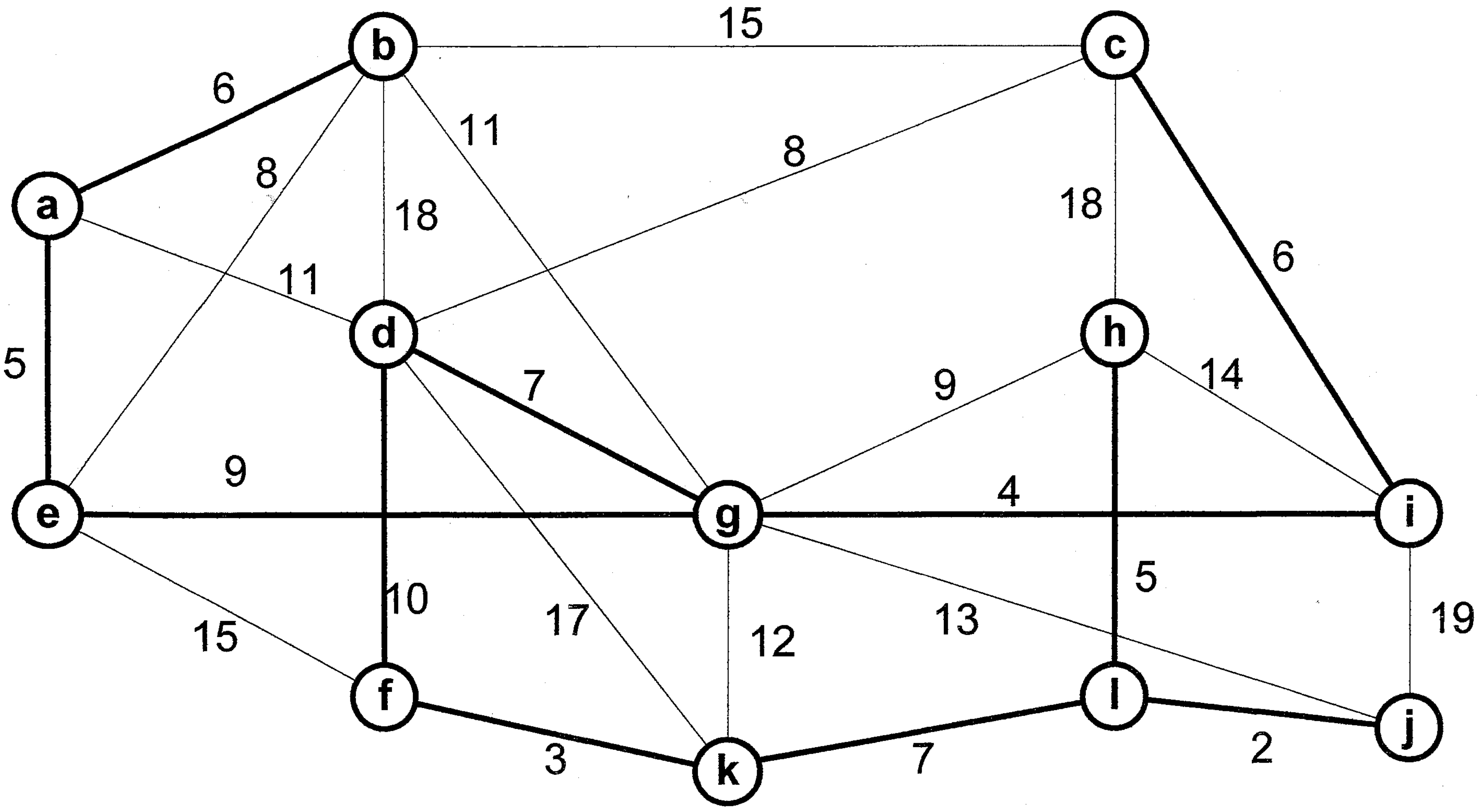
Prim's Algorithm

- add arc from 1 to 2;
- add arc from 2 to 3;
- add arc from 3 to 6;
- add arc from 6 to 5;
- add arc from 3 to 4;

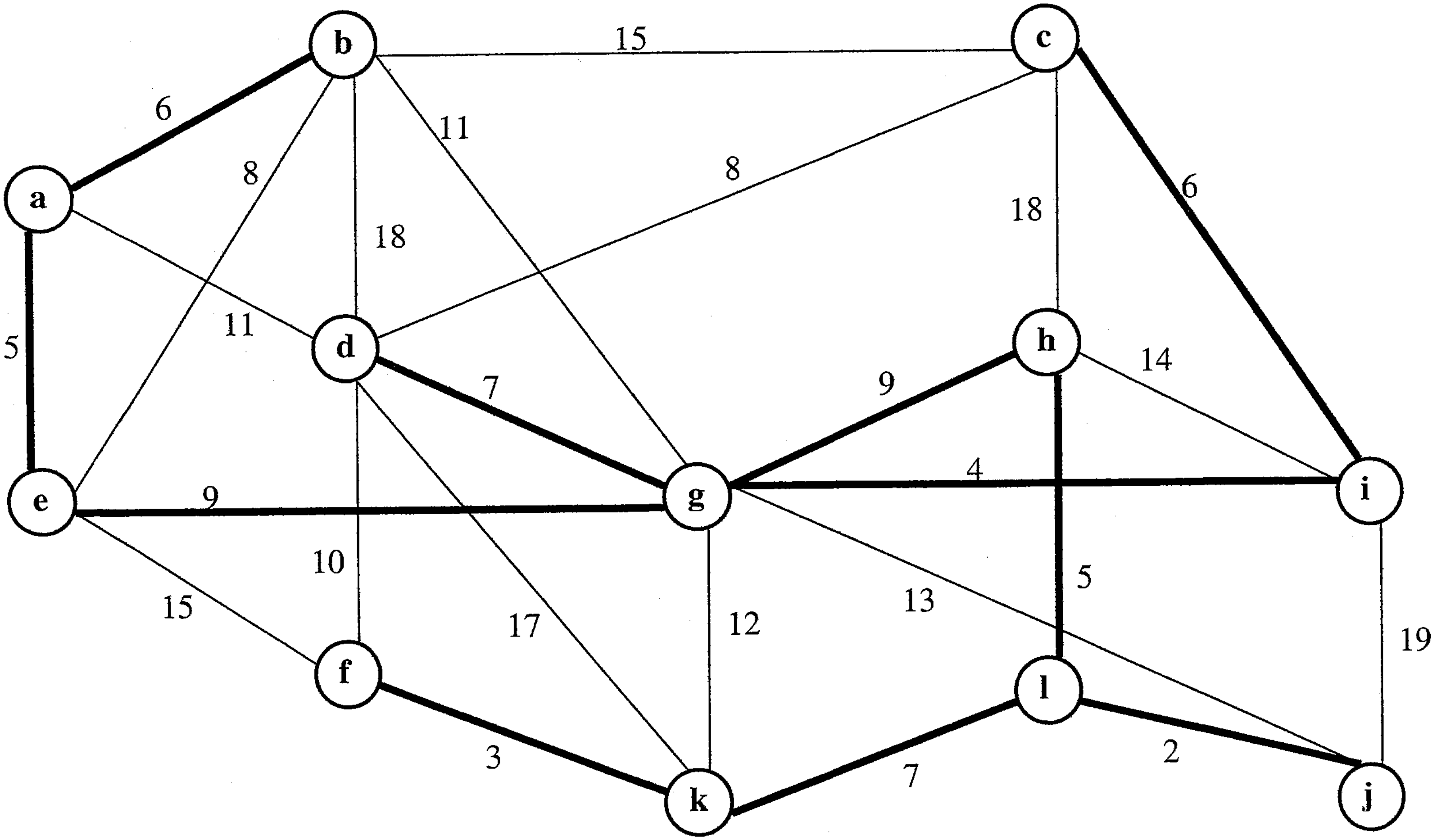
A RELATED PROBLEM

- There is an interesting relation between the MST problem and another, which sounds on the surface to be very different.
- Think of a network as being a highway map, where the number recorded beside each link is the maximum elevation encountered in traversing the link.
- Suppose someone who plans to drive from *i* to *j* dislikes high altitudes and hence wants to find a path connecting *i* and *j* that minimizes the maximum altitude.
- It is a fact that, in the undirected case, the minimal spanning tree solves the problem, and for all pairs of cities. That is, the unique path in the minimal spanning tree joining a pair of cities minimizes the path height (maximum number on the path).

AN EXAMPLE

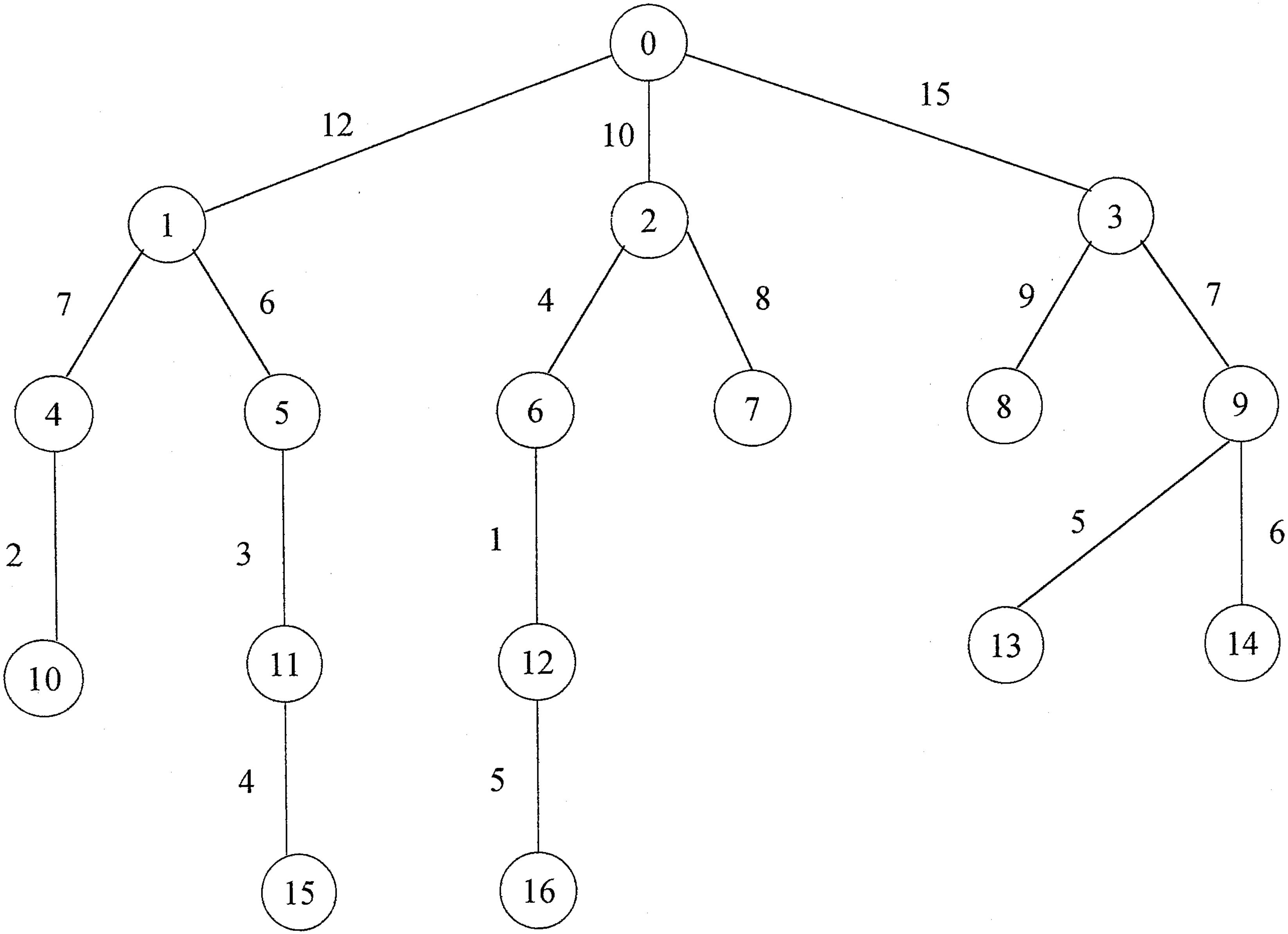


AN EXAMPLE



CAPACITATED MINIMAL SPANNING TREES

- The problem of finding a MST subject to the restriction that the number of nodes in any subtree rooted at a distinguished node not exceed a given maximum has received much attention.
- This is due to the fact that an algorithm capable of generating a MST subject to the above restriction can be used to design centralized telecommunications and data communications networks.
- A straightforward extension of this technique, where nodes are weighted by their total traffic, can be used to design such networks subject to a constraint on the total traffic on any multidrop line.
- The capacitated minimal spanning tree (CMST) problem cannot be solved optimally using a "greedy" algorithm, and no algorithm with a polynomial upper bound is currently known that solves this problem optimally.



5-8a

CMST HEURISTICS

- We must apply heuristic procedures here. Running times are on the order of seconds for problems with several hundred nodes and the quality of solution is within about 5% of the optimum.

- By making two modifications to the Kruskal procedure, one can adapt the technique to find constrained minimal spanning trees.

- First, when an arc is considered for admission to the spanning tree, the constraints (capacity restrictions) are checked and if any is violated the arc is rejected.

CMST HEURISTICS — CONTINUED

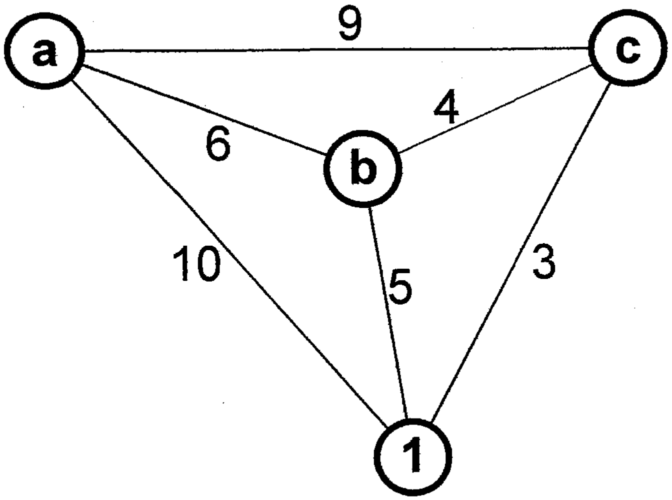
- Second, instead of considering c_{ij}, the length of arc (i,j), in ordering the arcs, one considers $t_{ij} = c_{ij} - w_i - w_j$ (there are, of course, other possibilities).

- The first modification ensures that the generated spanning tree satisfies the constraints.

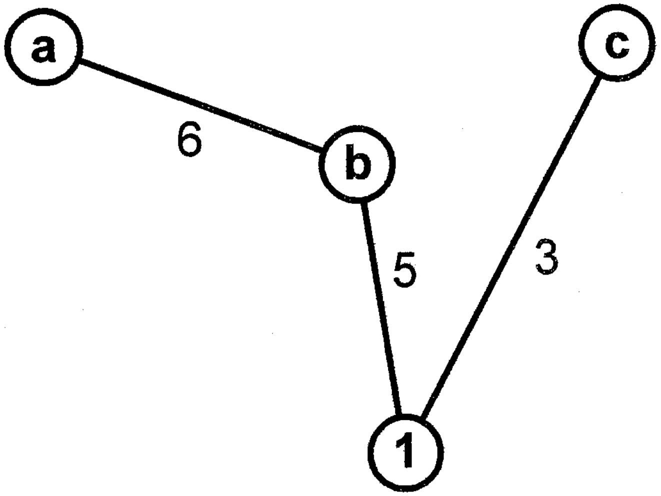
- The second modification causes the algorithm to generate spanning trees of high quality (low cost) if the node weights are chosen properly.

CMST HEURISTICS — CONTINUED

- How would you assign the node weights?
- Consider the CMST problem below as a test case. Suppose that no more than two nodes are allowed on any path to the center (node 1).
- How does your algorithm do?
- One possibility: Let $w_i = c_{1i}$.



CMST Problem

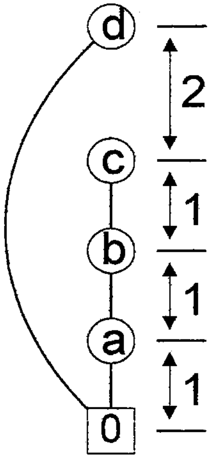


Best you can do.

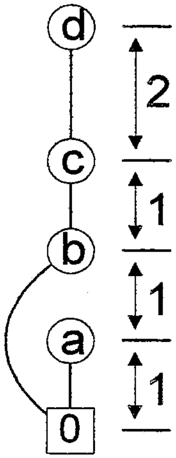
CMST HEURISTICS — CONTINUED

- Let's study the performance of 3 CMST heuristics.
  - ♦ Modified Kruskal's — ensure no subtree contains > k nodes.
  - ♦ Esau-Williams — modified Kruskal's plus tradeoff function.
  - ♦ Sharma — exploits the radial nature of the geometry.
- We consider three examples, each with a subtree capacity of 3.

Example 1.



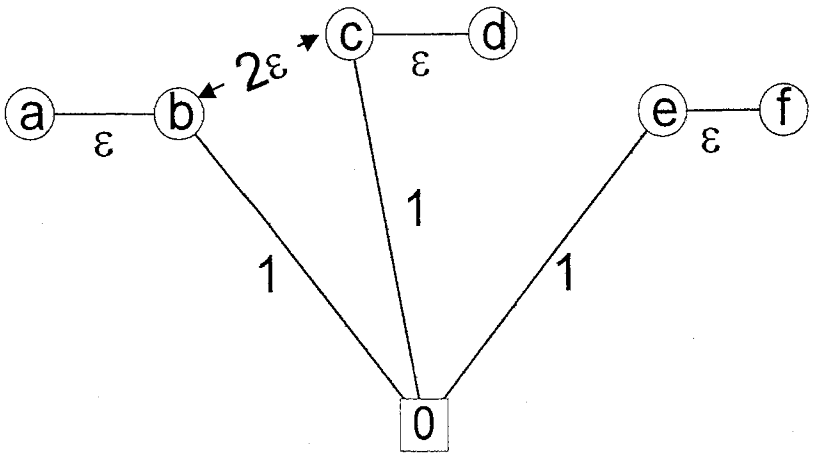
(a)
Modified Kruskal
Cost = 8



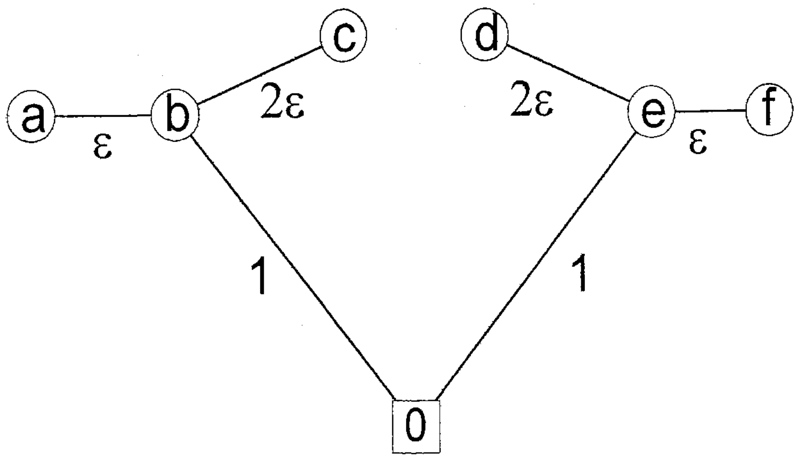
(b)
Esau-Williams (optimal)
Cost = 6

CMST HEURISTICS — CONTINUED

Example 2.



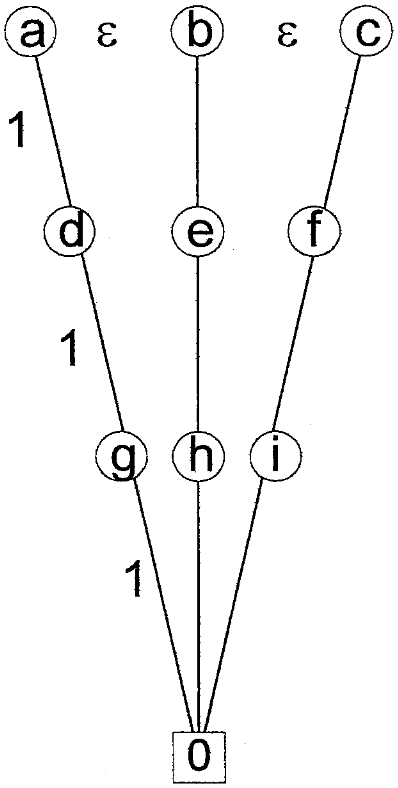
(a)
Esau-Williams
Cost = 3 + ε



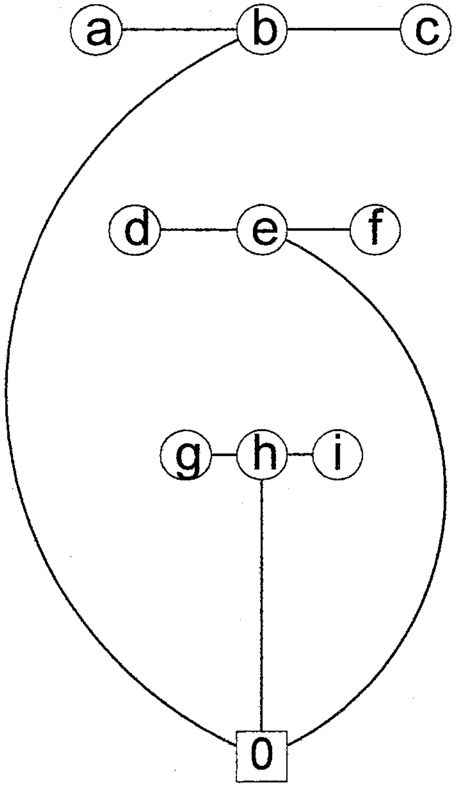
(b)
Sharma (optimal)
Cost = 2 + ε

CMST HEURISTICS — CONTINUED

Example 3.



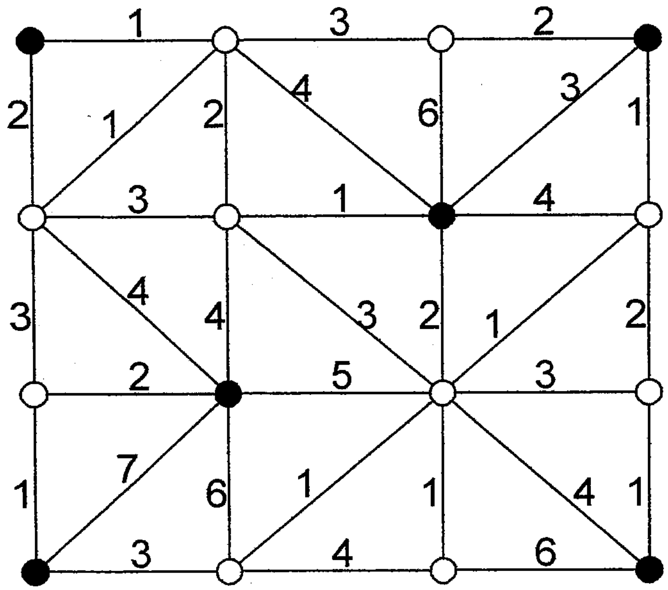
(a)
Sharma
Cost = 9



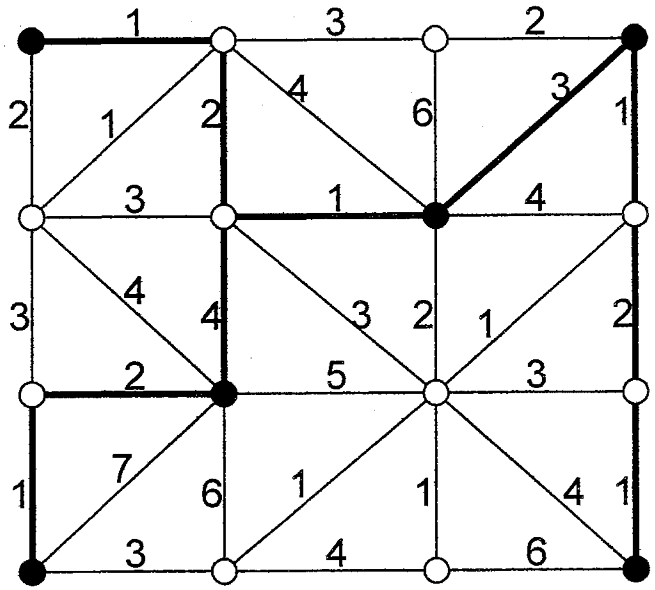
(b)
Esau-Williams
Cost = 6 + ε

THE STEINER PROBLEM IN NETWORKS

- A problem closely related to the MST, but much more difficult, is known as the Steiner Problem in Networks.
- In this problem, the shortest tree T is required which spans a specified subset $p \subset N$ of the vertices of G. Some of the other vertices may be spanned in order to minimize the length of T.



Original network

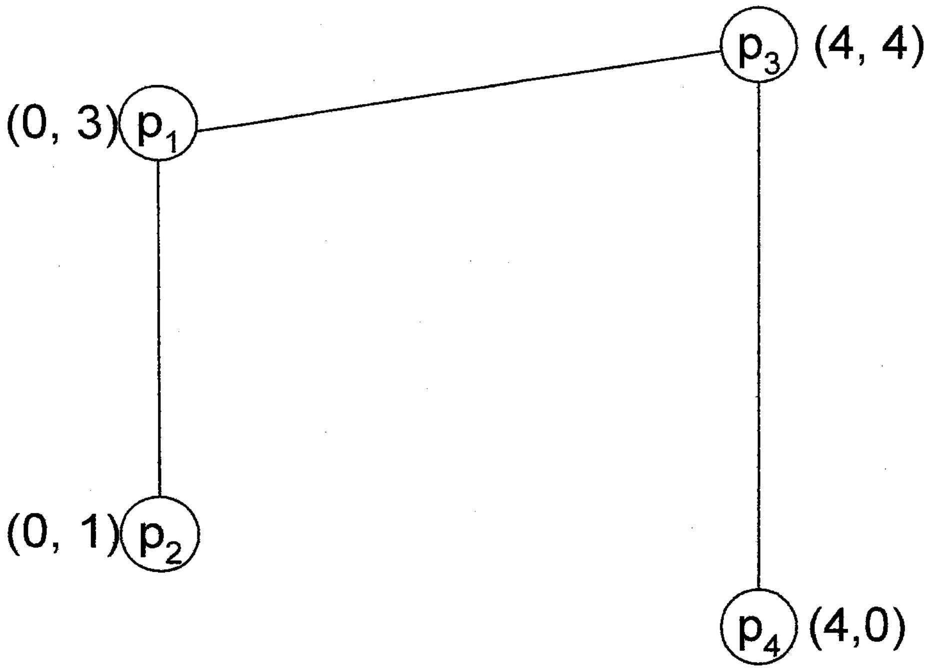


Possible solution

THE EUCLIDEAN STEINER PROBLEM

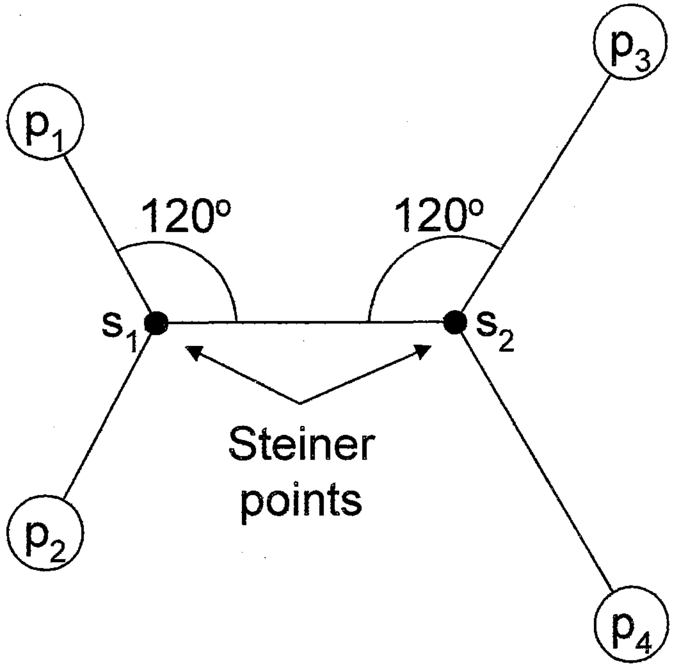
- The Euclidean Steiner Problem is a difficult geometric problem where a set *p* of points on a Euclidean plane are to be connected by lines so that the total length of lines drawn is a minimum.
- When points other than the set *p* of points can be introduced on the plane (Steiner points), then the total length may be less than the length of the MST.
- As many Steiner points as necessary could be added anywhere in the plane, in order to produce the shortest tree spanning the specified set p of points.
- The resulting shortest tree is then called a shortest Steiner tree.

THE EUCLIDEAN STEINER PROBLEM — CONTINUED



(a) Minimal spanning tree

Length = 10.123



(b) Shortest Steiner tree

Length = 9.196

THE RECTILINEAR (TAXICAB) STEINER PROBLEM

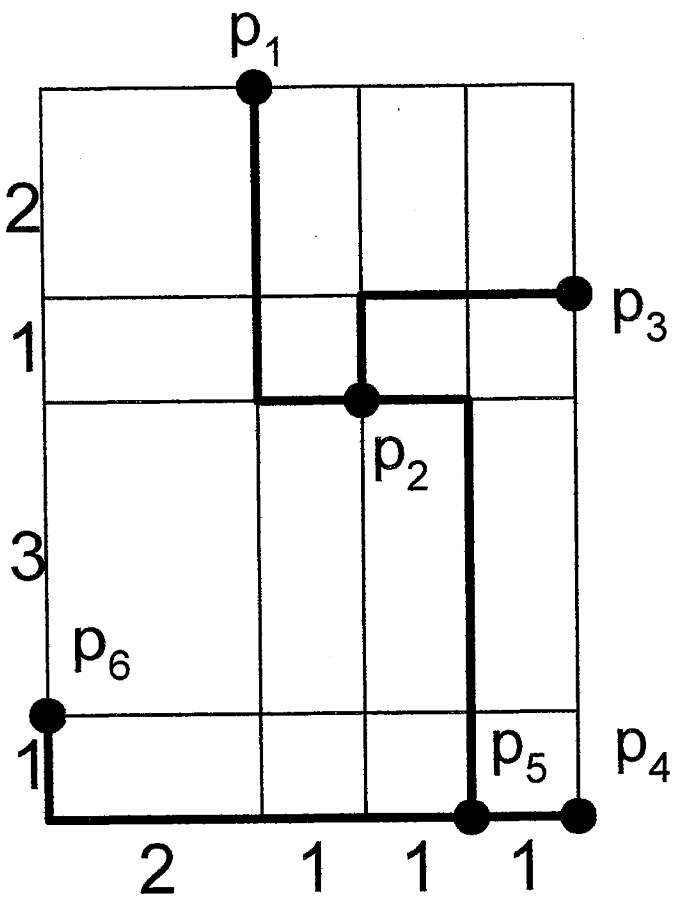
- This problem is very similar to the Steiner Problem in Networks. The distance between two points, however, is given by the taxicab distance.

- From the complete graph, a MST can be computed easily.

- If the intersections of N-S and E-W grid lines are considered as possible Steiner point locations, then Steiner trees can be constructed.

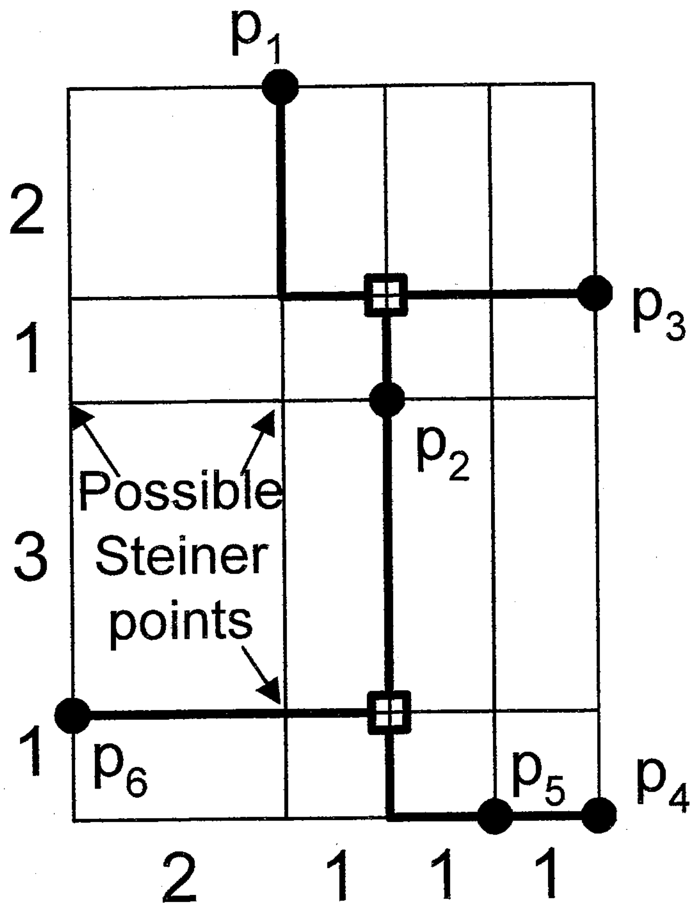
- The Shortest Steiner tree is, again, hard to find.

THE RECTILINEAR (TAXICAB) STEINER PROBLEM — CONTINUED



(a) Minimal spanning tree

Length = 18



(b) Shortest Steiner tree

Length = 15

⊞ = Steiner points