

This assignment asks you to implement Kruskal's algorithm for determining a MST of a given connected undirected network  $G$  with  $n$  nodes and  $m$  edges. Use a rooted tree approach with *union by rank* and *path halving*; implement a  $d$ -heap (on edge indices) to carry out the min operation. Your  $d$ -heap will need to be able to carry out (at least) the operations of `makeheap`, `findmin`, `deletemin`.

**Input** to your program is an edge list representation of  $G$ : a sequence of records containing the undirected edges and their real costs. No ordering of the input edges  $(i, j)$  or their costs  $c_{ij}$  is to be assumed. You can however suppose that the nodes of  $G$  are consecutively numbered  $1, 2, \dots, n$ . If needed by your code, the values  $n$  and  $m$  can first be input.

**Output** should include the edges of the MST (and their individual costs) as well as the cost of the MST.

- (1) Run your algorithm on the sample problems (to be supplied) and provide detailed output, as described above.
- (2) Investigate the *empirical* complexity for  $d = 2, 3, 4$  using the sample problems. Compare the CPU times. You may want to put the real work of the algorithm into a loop and run it many times to get an average CPU time. (In MATLAB, timing can be done using `tic` and `toc`.)
- (3) Determine the *time* and *space* complexities of your algorithm, in terms of relevant network parameters. Try to be space economical.
- (4) From a theoretical point of view, which value of  $d$  should be selected to reduce the (worst case) time complexity associated with the `deletemin` operations? Does this agree with your empirical findings in (2)? Discuss.

#### Things to Keep in Mind:

- make your code as general and modular as possible
- it's nice to have the main routine "self-documenting"
- adequately document your code
- avoid excess temporary variables, excessive copying, and unnecessary special cases
- timing of codes should exclude input and output; it should however include building the heap and running Kruskal's algorithm

Provide adequately documented code as well as a clear English language description of your data structures, what your code is doing, and important program variables.