

## 1 Supporting descendent searches

In Lecture 9, I gave pseudocode and analysis for cost-based ancestor search. In this problem, you should show how to support the following operations in  $O(\log n)$  amortized time per operation.

- $\text{DESCENDENTFINDCOST}(u, \alpha)$ : find the farthest (or nearest) ancestor of  $u$  having cost at most  $\alpha$ , or *null* if there is no such descendent.
- $\text{DESCENDENTFINDMIN}(u)$ : find the minimum-cost descendent of  $u$  in the conceptual tree.

The dynamic tree implementation to use is the one with dashed nodes. Each node should be assigned a label  $\delta$  min. Describe the invariant to be satisfied by  $\delta$  min. Give pseudocode for the above operations, and give an argument for why each operation should take amortized  $O(\log n)$  time. (That is, the virtual cost should be  $O \log n$ .)

## 2 Edge separators

Your goal in this problem is a data structure that represents a forest of maximum degree three so as to support the following operations, each in amortized  $O(\log n)$  time:

- $\text{REMOVE}(e)$ : remove an edge  $e$  from the forest.
- $\text{ADD}(e)$ : add an edge to the forest.
- $\text{SEPARATOR}(e)$ : given an edge  $e$  in the forest, return an edge  $\hat{e}$  in the tree  $T$  containing  $e$  such that each connected component of  $T - \hat{e}$  has at most  $2|E(T)|/3$  edges.