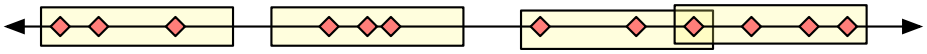
CS 473: Undergraduate Algorithms, Spring 2009
Homework 3½

Practice only

---

1. After graduating from UIUC, you are hired by a mobile phone company to plan the placement of new cell towers along a long, straight, nearly-deserted highway out west. Each cell tower can transmit the same fixed distance from its location. Federal law requires that any building along the highway must be within the broadcast range of at least one tower. On the other hand, your company wants to build as few towers as possble. Given the locations of the buildings, where should you build the towers?

   More formally, suppose you are given a set $X = \{x_1, x_2, \ldots, x_n\}$ of points on the real number line. Describe an algorithm to compute the minimum number of intervals of length 1 that can cover all the points in X. For full credit, your algorithm should run in $O(n \log n)$ time.



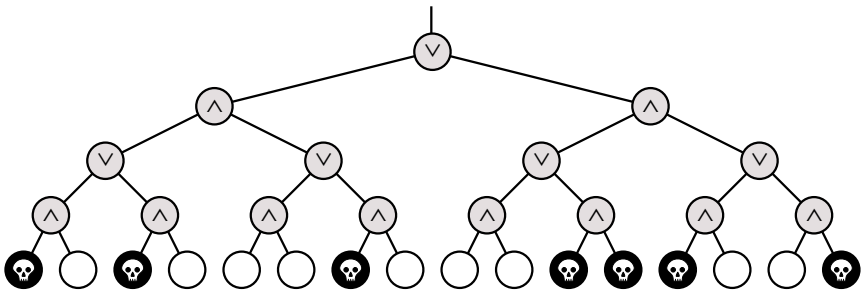
A set of points that can be covered by four unit intervals.

2. (a) The *left spine* of a binary tree is a path starting at the root and following only left-child pointers down to a leaf. What is the expected number of nodes in the left spine of an n-node treap?

   (b) What is the expected number of leaves in an n-node treap? *[Hint: What is the probability that in an n-node treap, the node with kth smallest search key is a leaf?]*

   (c) Prove that the expected number of proper descendants of any node in a treap is exactly equal to the expected depth of that node.

3. Death knocks on your door one cold blustery morning and challenges you to a game. Death knows that you are an algorithms student, so instead of the traditional game of chess, Death presents you with a complete binary tree with 4^n leaves, each colored either black or white. There is a token at the root of the tree. To play the game, you and Death will take turns moving the token from its current node to one of its children. The game will end after $2n$ moves, when the token lands on a leaf. If the final leaf is black, you die; if it's white, you will live forever. You move first, so Death gets the last turn.



You can decide whether it's worth playing or not as follows. Imagine that the nodes at even levels (where it's your turn) are OR gates, the nodes at odd levels (where it's Death's turn) are AND gates. Each gate gets its input from its children and passes its output to its parent. White and black stand for TRUE and FALSE. If the output at the top of the tree is TRUE, then you can win and live forever! If the output at the top of the tree is FALSE, you should challenge Death to a game of Twister instead.

(a) Describe and analyze a deterministic algorithm to determine whether or not you can win. *[Hint: This is easy!]*

(b) Unfortunately, Death won't give you enough time to look at every node in the tree. Describe a *randomized* algorithm that determines whether you can win in $O(3^n)$ expected time. *[Hint: Consider the case $n = 1$.]*

*(c) Describe a randomized algorithm that determines whether you can win in $O(c^n)$ expected time, for some constant $c < 3$. *[Hint: You may not need to change your algorithm from part (b) at all!]*