MA/CSSE 473 – Design and Analysis of Algorithms

Homework 04 – 75 points Updated for Summer, 2015

Most of these problems are variations on the "Tiling with Trominoes" example from Day 5 class. For another brief discussion of this example, see http://www.cut-the-knot.org/Curriculum/Geometry/Tromino.shtml
I have placed on Moodle an excerpt from the book on which my in-class discussion was based.
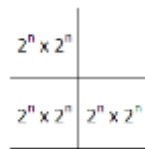
Here is pseudocode for the tiling algorithm from class. Many details are omitted, but it conveys the idea.

```python
def tileWithTrominoes(n, m):
    """n is the dimension of the deficient board to be tiled, and assumed to be a power of 2.
       m gives the location of the missing square that makes the board deficient."""
    if n == 2:
        # place the tromino so it covers the three squares
        return
    # Consider the four squares in the center of the board.
    # Three of them are in quadrants of the board that do
    # not contain m.  Place a tromino on them.  Now let
    # m1, m2, m3, and m4 be m and the three squares just covered.

    tileWithTrominoes (n/2, m1)   # assume that we have a mechanism
    tileWithTrominoes (n/2, m2)   # for translating these tilings
    tileWithTrominoes (n/2, m3)   # to the appropriate quadrants
    tileWithTrominoes (n/2, m4)   # of the original board.
```

1. (5) Write a recurrence relation for the running time of this algorithm, and use the Master Theorem to show that its solution is $\Theta(n^2)$. Thanks to Mike Jones for the following additional Master Theorem reference:
   http://www.math.dartmouth.edu/archive/m19w03/public_html/Section5-2.pdf

2. (2) Show that no 9×9 deficient rectangular board can be tiled with trominoes.

3. (8) Show that a 5×5 board with the upper-left corner square missing can be tiled with trominoes.

4. (10) Show a deficient 5×5 board that cannot be tiled with trominoes, and prove that this board cannot be tiled.

5. (5) Show that any 2i×3j board (not deficient) can be tiled with trominoes, for all positive integers i and j.
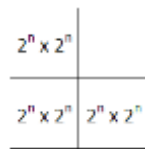
6. (10) A $2^n \times 2^n$ L-shape is a figure of the form [figure] with no missing squares, or it is a rotation of this figure by a multiple of 90°. Write a recursive algorithm that tiles any $2^n \times 2^n$ L-shape with trominoes. The recursive calls should tile smaller L-shapes. You may express it in English, pseudo code, or code from any programming language that is likely to be known by most people in this course. Feel free to use diagrams as part of your algorithm description.

7. (10) Use the preceding exercise as the basis for a different algorithm for tiling (with trominoes) any $2^n \times 2^n$ deficient rectangular board with trominoes.

8.   (10) Show that any deficient 7×7 board can be tiled with trominoes. There will be several different cases, depending on the location of the missing square. You are allowed to use the results of some of the previous problems.

9.   (15) More induction practice. A *binary tree* T is either empty, or it has a distinguished node called the *root* plus two disjoint subtrees T_L and T_R, which are both binary trees (know at the *left* and *right subtrees* of T).

   a.   (10 points) Use (strong) mathematical induction to prove the following properties of rooted binary trees: If a binary tree has leaves L_1, L_2, …, L_M whose depths (distance from the tree's root) are d_1, d_2, …, d_M, respectively, then
   $\sum_{i=1}^{M} 2^{-d_i} \leq 1$. You may want to draw a few tree shapes to convince yourself that it is true before you prove it. I suggest using as the basis for your induction the definition of binary tree as either empty or a root with two binary subtrees.

   b.   (5 points) Give a condition on the tree shape that is necessary and sufficient for replacing the "less than or equal to" sign in the statement with an "equals" sign. You do not have to supply a proof for this part.