ZIO Select Answers

Nebhrajani A. V.

November 27, 2020

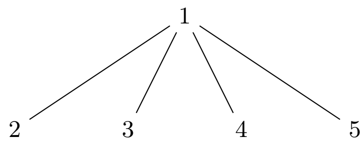
This document outlines my answers to some ZIO questions I found interesting or instructive. The solutions are written the way I solved them. In some cases, the solution is much better than the equivalent given in answer keys, in others, the answer key does the job better.

1 2010

1.1 Org-Trees

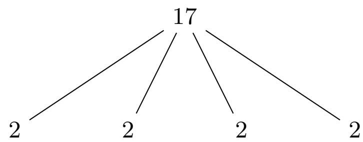
On first observation, the problem seems incredibly difficult to do because of the sheer number of combinations one can achieve in large org trees. The key insight, however, is that trees can (and should) be seen recursively bottom-up, a technique in dynamic programming.

Consider the simple tree (direction of arrows is omitted where hierarchies are obvious):

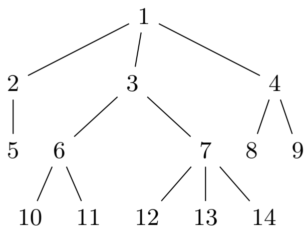


In how many ways can we delete a leaf level node? Obviously, only one way: just delete it. However, note that we always have the option of *not* deleting the node as well, giving us a total of 2 ways of dealing with leaves. Now, apply combinatorics. The total number of ways of deleting 1 (or below) will be the product of the ways of deleting its children, plus one for its own deletion.

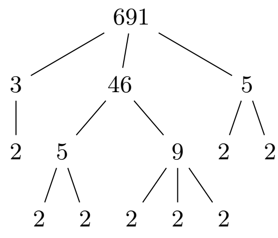
Therefore, writing number of ways of deletion into the tree:



And the problem is solved. Solving the first subproblem as an example,



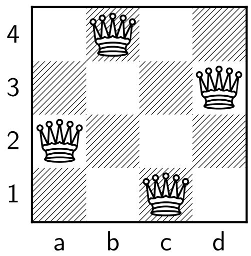
Solving bottom-up:

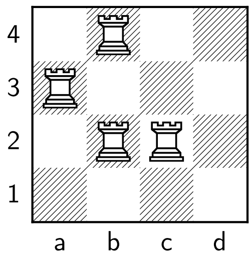


2 2011

2.1 n-Rooks Problem

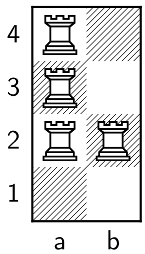
A famous problem in chess is the n-queens problem, or placing N queens on an $N \times N$ chessboard such that none of them threaten each other. Below is a solution for $N = 4$.

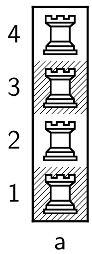




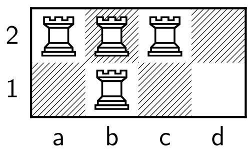
The given problem can be thought of as an n-rooks problem, where no two rooks may threaten each other. As you may have guessed, this is an easier job than queens, since we don't have to worry about diagonal threats anymore. It is therefore possible to view the given example as a 4×4 chessboard, with some rooks threatening each other.

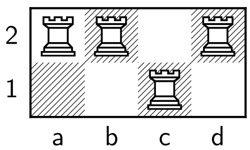
We are now left with calculating how to move these rooks so that they don't threaten each other. It is sufficient to consider this one axis at a time, in the following way. First consider threats only along the x-axis, and draw a board by collapsing empty squares. Move a minimal number of times to reduce this board to a $1 \times N$ board, then recompress. Repeat for the next axis. For the x-axis:

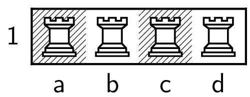




Moving Rb2 down and recompressing, we get the desired board. This took one move. (Recompressing is a visual aid and does not count as a move.) Therefore, $M_x = 1$. Now, along the y-axis:







To compress down, we must either move Rc2 to d2 and Rb1 to c1 (two moves) or move Rb1 to d1 (also two moves). So, $M_y = 2$.

$$\sum M = M_x + M_y = 1 + 2 = 3$$

Solve all subparts similarly. Be careful while doing compressions: choose a *minimum* number of moves to compress. This will always mean moving a piece to the closest empty row or column, and not one further away. This can be done by directly moving the piece or by swapping in a chain: both of which will require the same number of moves, as shown above.