Get our code, game play instructions, and a printable board on our project website!
questing.mobi

Monster World

An analysis of a fun game we are sick of overanalyzing.

By Ricardo Barron-Silva and Skylar Ittner

Game Setup

Start with a square board with a grid. The normal size is 10 by 10.
One player is the villain, and the other player(s) are heroes.
The villain rolls two dice and multiplies the results.
This is the number of monsters to be placed on the board.
He rolls two dice again to determine the number of treasures.
The villain player chooses locations on the grid to hide the monsters and treasures. They cannot be hidden on the top or bottom rows.
He writes the locations down on a secret piece of paper.
The villain then hides himself in the top row of the board.
The heroes each place a marker on the bottom row of the board.
Each hero rolls a die to determine their starting health points (HP).
The villain also rolls a die. He is allotted HP based on Figure 1.
If the HP of a player reaches 0, they die in real life.

Figure 1: Villain Health

d = die roll
p = number of heroes
h = villain health = $d * p * 2$ (at start of game)

Once the villain leaves his hiding place:
$h = \lceil h/2 \rceil$

Figure 2: # of possible starting layouts with given inputs

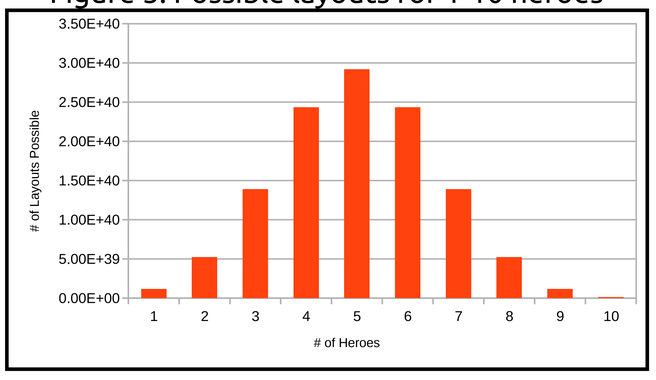
n = length of side of square board
t = number of treasures
m = number of monsters
p = number of heroes

$$\frac{(n^2-2n)!}{t!*m!*((n^2-2n)-t-m)!} * \frac{n!}{p!*(n-p)!} * \frac{n!}{1!*(n-1)!}$$

Number of possible initial layouts

For a two-player game (1 hero and 1 villain) on a 10x10 board, there are approx. 1.15861×10^{39} possible layouts.
For a game with 1 villain and anywhere from 1-10 players, there are approx. 1.18525×10^{41} layouts.

Figure 5: Possible layouts for 1-10 heroes



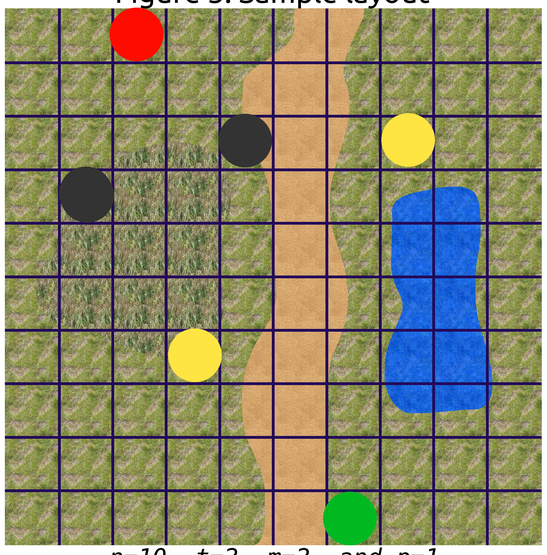
Analysis of Figure 2

Per the rules, all the heroes start on the bottom row, meaning there can be anywhere from 1 to n heroes, otherwise they wouldn't fit in the bottom row. Calculating how many ways there are to fit p players into n squares is a simple combinations formula (see second fraction).
The third fraction calculates the possibilities for the villain. The villain must be in the top row and occupies a single square out of n (the length of a row) possible squares. The formula is the same as the second fraction. Notice the player variable is replaced with the number 1, as there is only one villain.
All the possible ways to arrange the monsters and treasures on the game board is a type of permutations calculation that can be solved with some more math:

Suppose a set contains k types of objects, with n_i indistinguishable objects of type 1 for each i=1,2, … k. Then the number of distinguishable permutations of the objects is at right. We know that there are n^2 squares to place all the pieces, but since only heroes can be placed in the bottom row, there are only $n^2 - n$ places to put monsters and treasure. Also, since the villain's piece is the only piece that can be in the top row at the start of the game, that further diminishes the possibilities to $n^2 - 2n$ for where monsters or treasures can be placed. Permuting $n^2 - 2n$ objects would be over counting because these permutations do not account for rearranged monsters and treasures that yield the same setup; therefore, we divide by the number of indistinguishable object permutations. One thing to note is that these object permutations include the monsters (m!), treasures (t!), and blank spaces on the board ($((n^2 - 2n)-m-t)!$).

$$\frac{(n_1+n_2+\ldots+n_k)!}{n_1!\,n_2!\ldots n_k!}$$

Figure 3: Sample layout



n=10, t=2, m=2, and p=1

Analysis of Figure 5

Figure 5 has a "bump" because of how combinations work. When there is one player and ten places to choose from, there are ten layout choices. When more players are added, more combinations of players and empty spaces are possible. However, after five players have been added, this pattern changes. Because players are indistinguishable, when six or more are added, the player-occupied spaces might as well be blank spaces, and vice versa. This is why the chart is symmetrical and slopes back down above five players.

Analysis of Figure 4

We wrote the code in Figure 4 to automate finding all possible starting layouts. Lines 1, 2, and 3 are the inputs, comprised of the length of one side of a square board, the number of heroes, and the number of bad players. The code calculates the combinations possible for every possible number of monsters and treasures. The possible dice rolls are defined in the `dicerolls` list. The formula used corresponds to Figure 2. After the combinations are calculated for a set of dice rolls, they are multiplied together and the result is added to a running total for all combinations.

Figure 4: Calculating possible setups for all dice rolls

```
1  var boardlen = 10;
2  var players = 1;
3  var badplayers = 1;
4  var totalcomb = 0;
5  var boardsize = boardlen * boardlen;
6
7  var dicerolls = [1,2,3,4,5,6,8,9,10,12,15,16,18,20,24,25,30,36];
8
9  print("Board:     " + boardlen + "x" + boardlen + "(" + boardsize + ")");
10 print("Players:   " + players + " good, " + badplayers + " bad");
11 print("  Calculating...");
12 for (var i = 0; i < dicerolls.length; i++) {
13      m = dicerolls[i];
14      for (var j = 0; j < dicerolls.length; j++) {
15           t = dicerolls[j];
16           var comb_mt   = factorial(boardsize - 2 * boardlen) /
17                (factorial(m) * factorial(t) * factorial(boardsize - 2 * boardlen - m - t));
18           var comb_good = factorial(boardlen)/(factorial(players) * factorial(boardlen - players));
19           var comb_bad  = factorial(boardlen)/(factorial(badplayers) * factorial(boardlen - badplayers));
20           totalcomb += (comb_mt * comb_good * comb_bad);
21      }
22 }
23 print("Possible board setups: " + totalcomb);
```