Solve Weighing Pool Ball Puzzle

Szabolcs Seláf
selu@selu.org

December 15, 2016

1 Weighing Pool Ball Puzzle

The problem is described at https://www.mathsisfun.com/pool_balls.html with a solution. But the given solution is undetermined. Subsequent measures are depends on the result of the previous one. Ferenc Rákóczi mentioned, he already had a determined solution for this problem, but he forgot it.

1.1 The problem

You have 12 balls identical in size and appearance but 1 is an odd weight (could be either light or heavy).

You have a set of balance scales which will give 3 possible readings:

- Left = Right
- Left > Right
- Left < Right (ie Left and Right have equal weight, Left is Heavier, or Left is Lighter).

You have *only 3 chances* to weigh the balls in any combination using the scales. Find which ball is the odd one and if it's heavier or lighter than the rest.

2 Find one solution

2.1 Generate possible measures

At first I calculated the number of possible measures. We should put equal number of balls onto both arms to get a valid result, so pick even number of balls for measure, then pick half of the balls for Left arm and put remaining balls into Right arm. We can halve the numbers, because putting same set of balls to Right arm is same as to Left arm:

$$\sum_{i=1}^{6} \frac{\binom{12}{2i}\binom{2i}{i}}{2} = 36829$$

All three measures can be one of the 36829 possible ones, so the number of all possible solutions could be:

$$36829^3 = 49953943750789$$

Using information above we can generate possible measures one-by-one:

```
ids = (1..12).to_a                              # array [1, 2, .., 12] for balls
(1..6).each do |level|                          # how much balls to put on an arm
  ids.combination(level*2).each do |subset| # pick a combination for both arms
    reverse = []
    subset.combination(level).each do |left| # pick half of them for left arm
      next if reverse.include?(left)          # go to next if reversed already
      right = subset - left                   # checked
      reverse << right
```

2.2 Backtracking

At start any of the 12 balls can be heavier or lighter than others, which means we have 24 possible results. Each measure has 3 possible outcomes, 3 measures could solve $3^3 = 27$ different cases. After the first measure the remaining two could solve only $3^2 = 9$ cases and of course, the last one could solve maximum 3 cases.

The program uses this knowledge to give up, once there is a possible outcome of a measure with more different cases, than that can be eliminated by the remaining steps.

```
if state_set.map(&:case_number).max > (Scale.number_of_outcomes ** max_measures)
  return
end
```

2.3 Result

However this algorithm is almost a brute force, found the first solution within some seconds.

Number	Left arm				Right arm			
1	1	2	3	4	5	6	7	8
2	1	2	3	5	4	9	10	11
3	1	4	6	9	2	7	10	12

Table 1: Measures