AofC 2019 Day 12

Puzzle input is position of the 4 moons.

Puzzle challenge is to simulate their movement.

initial

each moon

has position (x, y, z) (puzzle input)

has velocity (dx, dy, dz) init (0,0,0)

for t = 0; t < 1000; t++

foreach moon

update\_velocity\_apply\_gravity

*gravity is applied on each moon with every other moon AS IT IS NOW (i.e. careful not to update position of current moon) by:*

*for each axis*

*calculate difference of the moon’s* ***position*** *against* ***other moon*** *as d*

*if d < 0 then*

*increment first moon’s velocity by 1*

*decrement second moon’s velocity by 1*

*else if d > 0*

*decrement first moon’s velocity by 1*

*increment second moon’s velocity by 1*

*else*

*no changes to velocity*

*end*

How to calculate

update\_position\_apply\_velocity

*for each moon*

*velocity is velocity + position*

*v(x) = v(x) + p(x) etc*

calc\_potential\_energy

pot = abs(x) + abs(y) + abs(z)

calc\_kinetic\_energy

kin = abs(x) + abs(y) + abs(z)

tot\_energy = pot + kin

t += 1

ANSWER

* I made mistake of misreading the apply\_velocity function and thought that it was updating the position of each moon. If this was the case then each moon’s velocity needed to be calculated from the original positions – i.e. not update the class object
* However, it was not. Only velocity changed. After this I finished it quite quickly.
* Made improvements with Swift’s code https://github.com/J-Swift/advent-of-code-2019/blob/master/day\_12/part\_2/solution.rb

PART 2.

* I had the idea of creating a string, ‘chksum’, from all the position and velocity values adding them to an array, then loop until a ‘chksum’ exists in the array
* Found Swift’s code on Megathread and changed to using a Set, which is quicker than arrays
* The first test worked, but not second. Perhaps because the data was too large, so I cheated with Swift’s code to get answer
* Running his code to see how hash works it is much smaller than my ‘chksums’.
* I do not yet understand how his hashing works, which uses this:

moon.position.send(axis), moon.velocity.send(axis)].join('\_')

send method: https://stackoverflow.com/questions/3337285/what-does-send-do-in-ruby