Phi ϕ Shui: Novice to Expert

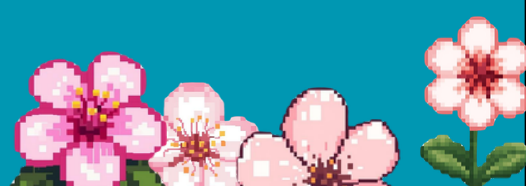
If you don't know what Feng shui then , you're in luck because I don't know much either

Quoted from Wikipedia "Chinese world, feng shui has been used to determine the orientation of buildings, dwellings, and spiritually significant structures"

For example : not placing your bed near the entrance because of the excess energy or not placing the sink(water) and stove(fire) together since the elements collide

Anyways , the problem with Feng shui is that it's not efficient and with the housing prices, we can not be sacrificing space for balancing out the chi

So , today we will be looking at the mathematical version of Feng shui that is Phi shui named after ϕ obviously (that I just made up)



Level 1: Phi Shui Novice

Sensei: "Welcome, to the digital land of xina, the origin of the art of Phi shui. Your task is to arrange these blocks inside this knapsack to maximize the value of the items without tearing the bag. Once you pass this level, you will be promoted to an Intermediate level. Good luck."

Alright, now that Sensei has assigned us our first problem, let's solve it. You may wonder how packing a bag helps us learn Phi shui, but don't worry just trust Sensei

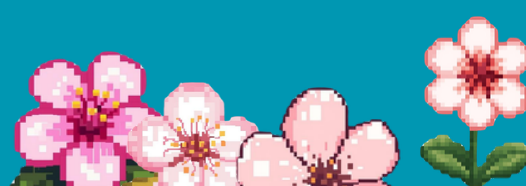
First, notice that the order inside the bag doesn't matter. Once we have chosen the optimal combination of items, they simply exist in the space. Since the blocks only have two possible states,they are either in the bag or out of the bag,we can represent this using a binary variable, x

value: 5 , weight: 5 value: 8 , weight: 12

value: 4 , weight: 5 Value: 6 , weight: 5

Weight limit is 15

*Try to find the optimal solution



For any item i:

$$x_i = \begin{cases} 1 & i \in B \\ 0 & i \notin B \end{cases}$$

This 1 and 0 binary gives us a really nice mathematical property. To find the total value of our packed bag, we just multiply each item's value by its binary variable and add them up. If the item is left behind, it gets multiplied by 0 and vanishes from the total. We use the same trick to calculate the total size or weight of the bag!

We now have our objective and our constraints:

This turns it into a bunch of equations namely:

$$\sum s_i x_i \leq c$$

$$\text{Minimize} \sum v_i x_i$$

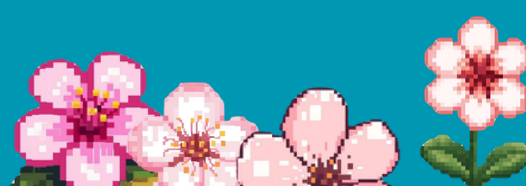
Which is an integer linear program , it just means it's solutions are integer and it has linear constraints

Just keep the idea of binary variables and adding constraints, it's gonna come in handy many times

We can solve the equation by hand but thankfully, Sensei has granted us access to digital sorcery. We can write a script in the ancient tongue of Python to solve it.

```
import pulp

prob = pulp.LpProblem("Binary_Optimization", pulp.LpMaximize)

w = pulp.LpVariable("w", cat=pulp.LpBinary)
x = pulp.LpVariable("x", cat=pulp.LpBinary)
y = pulp.LpVariable("y", cat=pulp.LpBinary)
z = pulp.LpVariable("z", cat=pulp.LpBinary)

prob += 5*x + 8*y + 4*z + 6*w

prob += 5*x + 12*y + 5*z + 5*w <= 15

prob.solve(pulp.PULP_CBC_CMD(msg=0))

print(f"Status: {pulp.LpStatus[prob.status]}")
print(f"Values: w={w.varValue}, x={x.varValue}, y={y.varValue}, z={z.varValue}")
print(f"Total Value: {pulp.value(prob.objective)}")
```

question: what if we switched 0(inside bag) and 1(outside bag)



How to solve an integer program?

Branch and Bound Technique

This is to show you how to solve an integer program by hand and develop a deeper understanding of it

The Core Concept

In a standard Linear Program (LP), variables can be decimals (like 2.5). In an Integer Linear Program (ILP), they must be whole numbers (like 2 or 3). Branch and Bound solves this by treating the ILP as a sequence of easier decimal problems.

1. The "Relaxation" (The Starting Point)
You start by ignoring the "integer only" rule. You solve the problem as if decimals are allowed. This is called the LP Relaxation.

If the result happens to be whole numbers (e.g., x=4, y=5), you're done! That's the optimal solution.

If you get decimals (e.g., x=2.7), you move to the next step.

2. Branching (Splitting the Problem)
Since x=2.7 isn't allowed, you know the real answer must be either x <= 2 or x >= 3. You split the problem into two new "sub-problems" (branches):

Branch A: The original problem + the constraint (x <= 2).

Branch B: The original problem + the constraint (x >= 3).

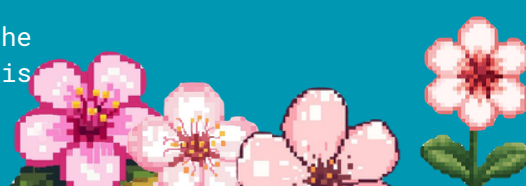
3. Bounding (The Logic Filter)
As you solve these branches, you keep track of the best "whole number" solution you've found so far. This is your Lower Bound (if you're maximizing).

If a branch gives you an LP solution of 100, but you already found a whole-number solution that gives you 110, you kill that branch.

Because a decimal solution is always "better" or equal to the integer version of that same branch, if the decimal version is already worse than your current best, the integer versio
n will be even worse.

solve :
maximize $5x+4y$

$x+y \leq 5$
$10x+6y \leq 45$
$x, y \geq 0$ and both are integers



Level 2: Phi Shui Intermediate

Sensei: "Good job, my student. Now you must use this knowledge to solve the 2D grid version of Phi shui.Youre given items along with their utility , you need to maximize utility"

Now we have an actual, spatial version of Phi shui on our hands. Instead of merely deciding whether an item should be in the room, we also have to decide exactly where it goes on a 2D grid.

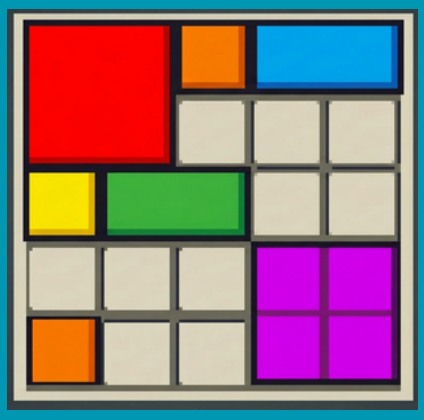
We have vastly more possibilities now. But using our mantra of adding binary variables, we can tackle this. Let's just add more variables for the placement!

We choose an "anchor" coordinate for each block,namely, its bottom-left corner.
So, if item 1 was placed with its anchor on coordinate (a,b) the specific variable

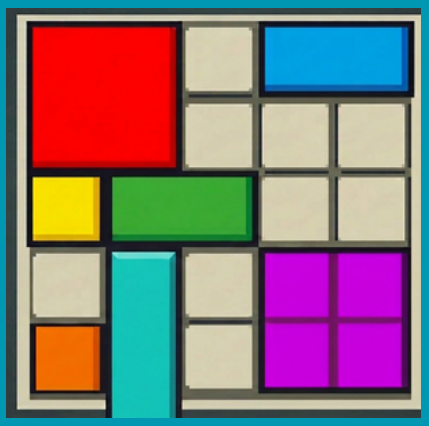
$$x_1(a,b) = 1$$

All other placement variables for item 1 would be 0
This creates a new problem: if we place an anchor too close to the wall, the block will clip right through the room's outer bounds. We could also accidentally generate "duplicates" by assigning the same item to multiple coordinates.How can we avoid these errors?

valid arrangement



Invalid due to duplication



Invalid due to going off grid

*question:What would happen if you chose top-right as the anchor instead?

If you think carefully, clipping is easy to fix. If an anchor placement would cause the item to clip through a wall, that is an invalid placement. We just ditch that variable entirely. We would never place the block like that, so the variable doesn't need to exist in our math.

$$x_1(2,1) = 1$$

For the duplicates, we add a strict constraint: an item can only exist in the room exactly once (or zero times).

$$\forall i, \sum_{a,b} x_i(a,b) \leq 1$$

With this, we avoid the cloning problem. However, we have an even bigger problem: overlap. In Phi shui, you cannot place a stove inside a sofa.

How can we prevent overlap? By adding more constraints! Instead of locking the item and checking its coordinates, we lock the coordinate and check the items. For every single 1×1 square on our grid, we state that the sum of all item-parts occupying that specific square must be less than or equal to 1.

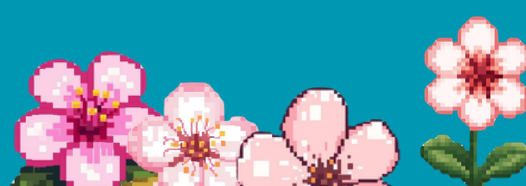
$$\forall (a,b), \sum_{i} x_i(a,b) \leq 1$$

We have successfully built a 2D Integer Program!

$$\forall (a,b), \sum_{i} x_i(a,b) \leq 1$$

$$\forall i, \sum_{a,b} x_i(a,b) \leq 1$$

$$\text{Maximize} \sum v_i z_i(r,c)$$



How to solve an integer program ?
pt 2

Genetic algorithms

In the 2d case , the number of variables skyrocket,
For a 10×10 board with 10 blocks , the number of variables skyrocket to 1000

That's why it's actually useful to use a genetic algorithm which gives an approximate solution but way quicker

1. Encoding
Every solution is a string of bits. Each bit represents a variable: 1 means "Include this item/Activate this switch," and 0 means "Exclude/Deactivate."

Creature A: [1, 0, 1, 1, 0]

Creature B: [0, 1, 1, 0, 0]

2. Fitness
You calculate a score for each creature based on your goal (e.g., maximizing profit).

The Constraint Check: If the sum of weights for all "1" bits exceeds your limit, you give that creature a massive penalty. This effectively "kills" the solution so it doesn't pass its genes to the next generation.

3. Selection
You choose parents based on their fitness. A common method is Tournament Selection: you grab a few random creatures, and the one with the highest fitness wins the right to breed. This ensures the "strongest" bits stay in the gene pool.

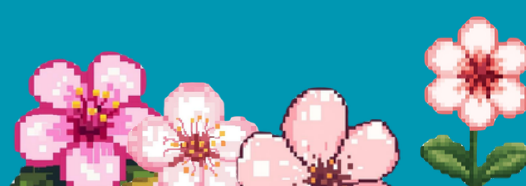
4. Crossover
You combine two parents to make a child. You pick a "slice point" and swap the tails.

Parent X: [1, 0 | 1, 1, 0]

Parent Y: [0, 1 | 1, 0, 0]

Result (Child): [1, 0, 1, 0, 0]

5. Mutation (The Random Flip)
To keep the population from getting stuck, you occasionally flip a random bit (change a 0 to a 1, or vice versa). This is the only way the algorithm "discovers" new combinations that weren't
in the original parents.Repeat this process over and over while keeping the best performing one in the population and eventually you will get a good answer



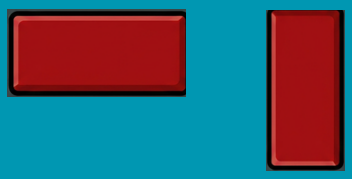
Level 3: Phi Shui Expert

Sensei: "Impressive. But real life is rarely static. You must now allow the furniture to rotate, and recognize that some pieces are meant to be joined."

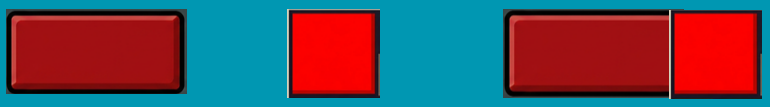
This level introduces some real-world complexities that we can solve using our established toolkit.Our first problem is allowing a 90° rotation.

Mathematically, rotating an item is exactly the same as introducing a brand-new, pre-rotated piece of furniture into our catalog. Let's call the rotated version of item i' and the original item i . Then we can again add some constraints and it's actually not that much different

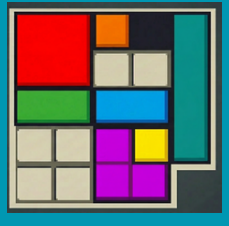
$$\forall i, i', \sum (x_i(a,b) + x_{i'}(a,b)) \leq 1$$



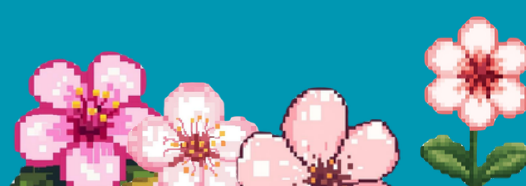
What if we are rewarded more for joining two specific items, kind of like puzzle pieces? We use the exact same technique: define a new, composite item that represents both pieces joined together, and add a constraint so that if the joined version is placed, the individual versions cannot be.



And what if the room isn't a perfect rectangle? Well, that just changes our original clipping idea! We simply remove any anchor variables that would force a block into the non-rectangular boundary.



Wow, that was actually easy because we already built the right tools. We're done, right?



Level 4: Phi Shui God
Sensei: "You have mastered the grid. Now, break it. The world is not a chessboard. You must allow rotations of any angle, in continuous space."

In this level, we allow rotations of 13°,47.2° or any arbitrary angle.

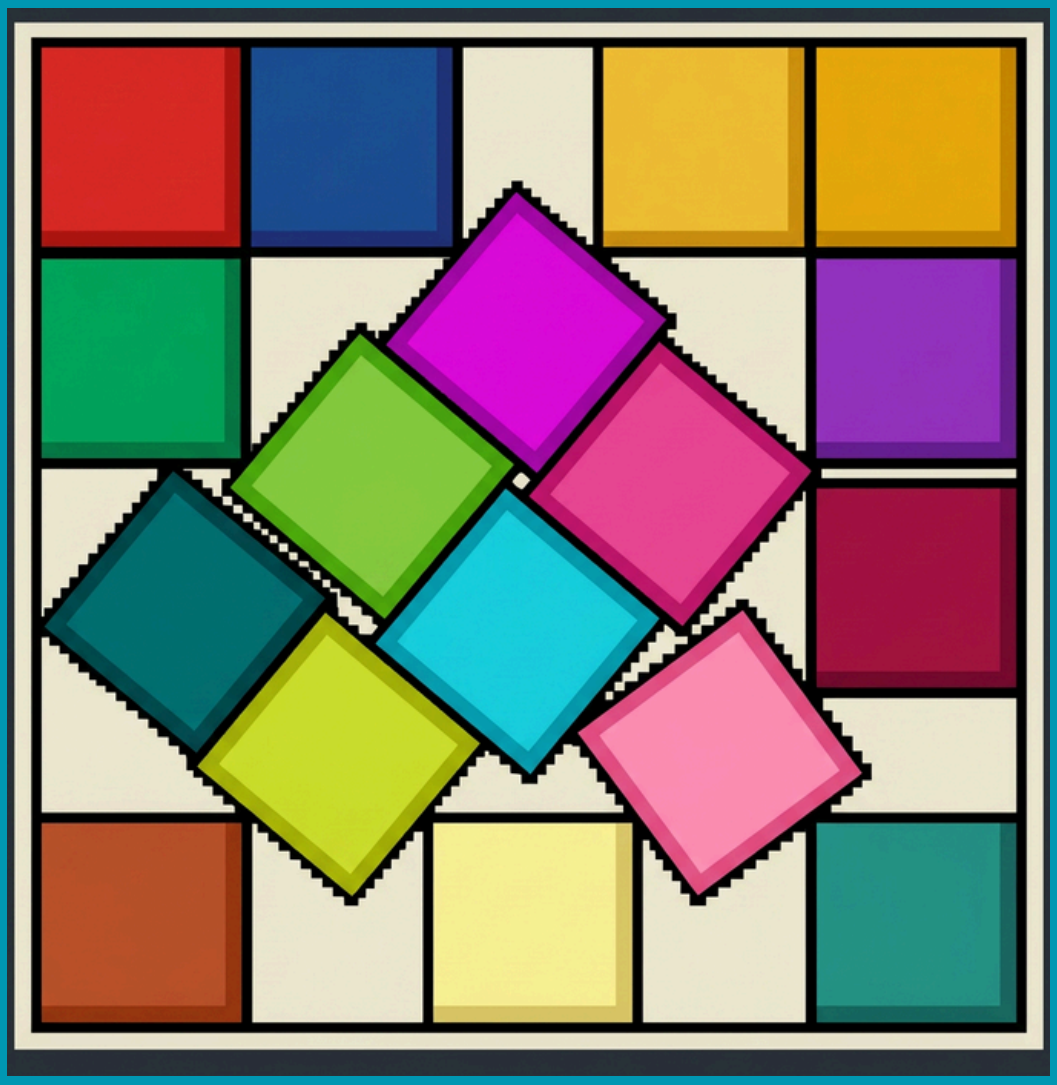
This entirely shatters our previous techniques. Our math relied on integers—discrete grids, strict boundaries, and a finite, fixed number of binary variables. When you introduce continuous angles and floating-point coordinates, the number of possible placements becomes infinite.

Everything is thrown out of the window. We move from Integer Linear Programming into the terrifying realm of non-linear, non-convex optimization.

When you ask a computer to find the absolute optimal Phi shui arrangement in continuous space, you don't get neat, straight lines. Because mathematical perfection doesn't care about human aesthetics, the optimal arrangements often look incredibly ugly.

Items are wedged at bizarre, decimal-point angles to shave off millimeters of wasted space. It is a chaotic, mathematically flawless mess that a human interior designer would never come up with.

And perhaps that is the ultimate lesson of Phi Shui. We can use mathematics to seek perfect efficiency, modeling our world with 1s and 0s to pack our metaphorical knapsacks. But when we push those equations to their absolute limits, stripping away the neat grids we invent for our own comfort, we find that true mathematical perfection is wonderfully, and beautifully chaotic.



Optimal arrangement for 17 unit squares and side length 4.675

Sensei : "Good job my student, you have finally seen the true mathematical beauty you have reached the true final level

Most of these types of problems are solved using a bunch of techniques namely genetic algorithms which you already saw and simulated annealing which basically simulates what would happen if you put all the items in a block and shook them vigorously and slowly reduce the magnitude to obtain a good solution