Recurrence Relations

COM

DEC

Section – I

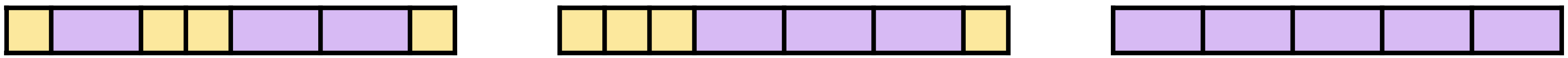
Recursion in Sequences

COM

DEC

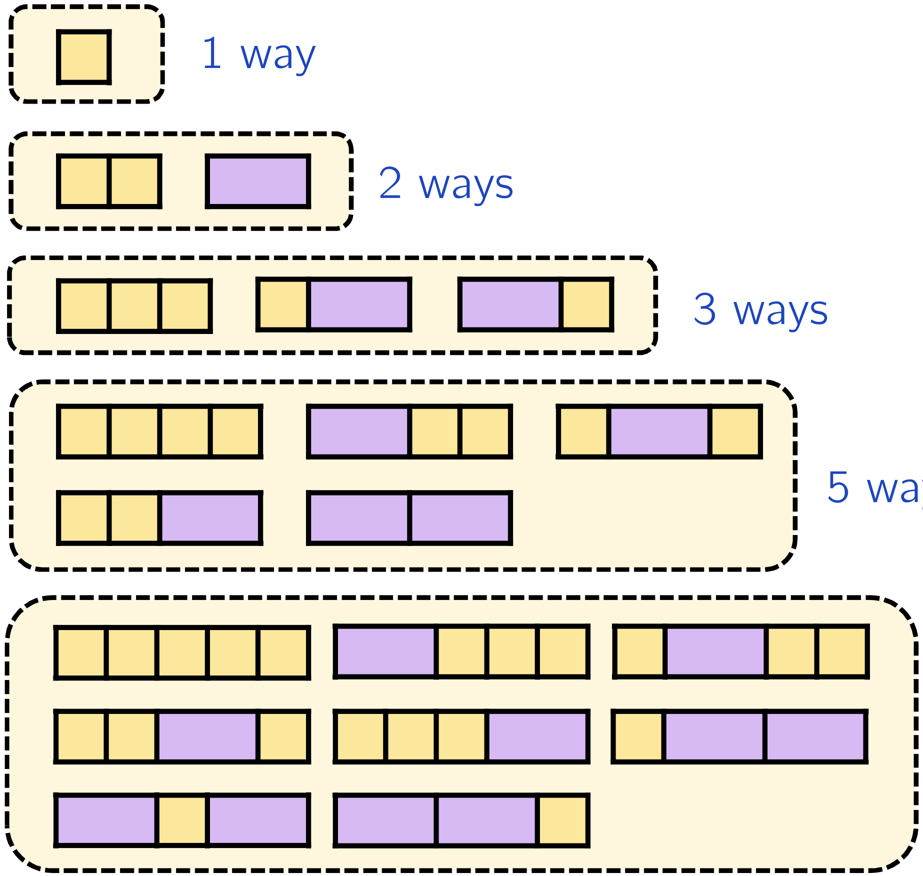
Q1: Domino Tilings

How many ways are there to tile a 1×10 rectangle using identical 1×1 tiles and identical 1×2 tiles?



Q1: Domino Tilings

Attempt: Replace 10 with n and try to find a pattern.



$n = 1$ — 1 way

$n = 2$ — 2 ways

$n = 3$ — 3 ways

$n = 4$ — 5 ways

$n = 5$ — 8 ways

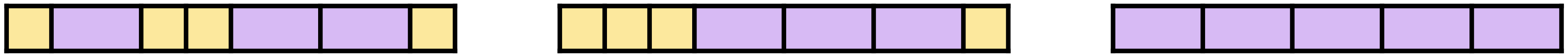
We get a familiar pattern

Next term = sum of previous 2 terms

Question: Why is this true?

Q1: Domino Tilings

How many ways are there to tile a 1×10 rectangle using identical 1×1 tiles and identical 1×2 tiles?



Solution

Let f_n be the number of ways to tile a $1 \times n$ rectangle.

Then,

$$f_n = \begin{array}{c}\text{\# of tilings}\\ \text{starting with } \square\end{array} + \begin{array}{c}\text{\# of tilings}\\ \text{starting with } \boxed{}\end{array} = f_{n-1} + f_{n-2}$$

true for all $n \geq 3$.

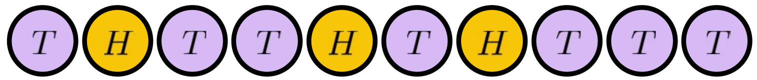
For $n = 1, 2$, we have $f_1 = 1, f_2 = 2$ by direct checking.

So, the answer follows by recursive computation: 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, …

Answer

Q2: Heads never touch

How many ways are there to arrange 10 identical coins in a row so that there are no two consecutive heads?



Q2: Heads never touch

Attempt: Again, replace 10 with n and try to find a pattern.

$n = 1$: T $\quad$ H — 2 ways

$n = 2$: TT $\quad$ TH $\quad$ HT — 3 ways

$n = 3$: TTT $\quad$ TTH $\quad$ THT $\quad$ HTT $\quad$ HTH — 5 ways

$n = 4$: $TTTT$ $\quad$ $TTTH$ $\quad$ $TTHT$ $\quad$ $THTT$ $\quad$ $HTTT$ $\quad$ $HTHT$ $\quad$ $HTTH$ $\quad$ $THTH$ — 8 ways

The same pattern emerges!

Next term = sum of previous 2 terms

Question: Why is this true?

Q2: Heads never touch

How many ways are there arrange 10 coins in a row so that there are no two consecutive heads?

Solution

Let g_n be the number of ways to arrange n coins in a row so that there are no two consecutive heads.

$g_n =$ # of arrangements starting with T + # of arrangements starting with H $= g_{n-1} + g_{n-2}$ ← true for all $n \geq 3$.

start with H ↔ start with H T

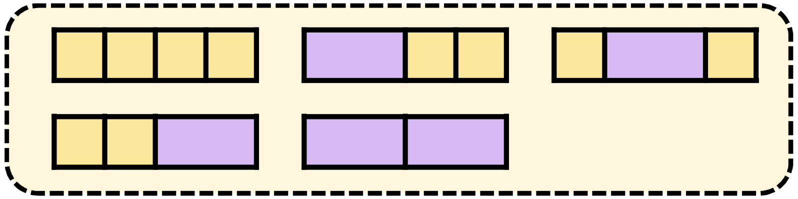
For $n = 1, 2$, we have $g_1 = 2, g_2 = 3$ by direct checking.

So, the answer follows by recursive computation: 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, …

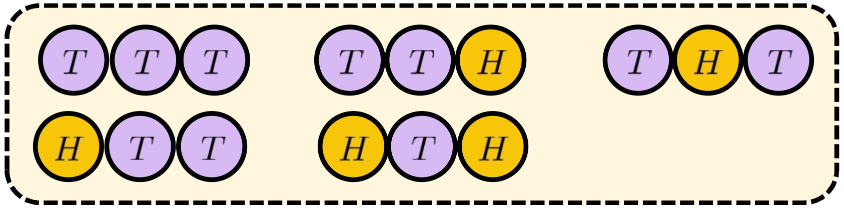
Answer

Why does $f_n = g_{n-1}$?

Looking at previous two problems, we see that $f_n = g_{n-1}$.



f_4



g_3

Boring Proof. We can prove by (strong) induction.

- Show that $f_2 = g_1$ and $f_3 = g_2$ by direct checking.
- Suppose that $f_k = g_{k-1}$ and $f_{k-1} = g_{k-2}$ for some $k \geq 3$. Then,

$$f_{k+1} = f_k + f_{k-1} = g_{k-1} + g_{k-2} = g_k$$

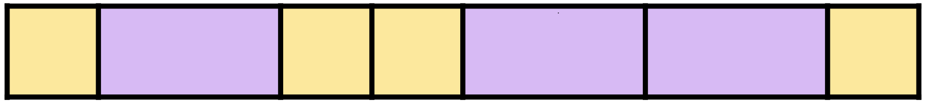
- Thus, we are done by (strong) induction!

This is too trivial that it shouldn't need a proof. But, induction is how you make it rigorous.

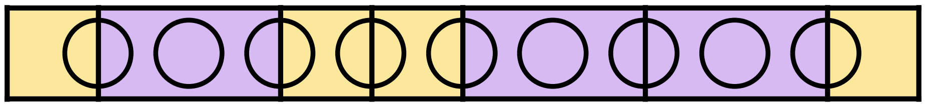
Why does $f_n = g_{n-1}$?

An Alternate (Interesting) Proof.

Idea: Construct a one-to-one correspondence between tilings and coin-arrangements!



Start with a tiling of the $1 \times n$ rectangle.



Draw a circle at each segment, giving us $n-1$ circles in total.

Change the “crossed” circles into T. Others into H. Then, no two H’s are adjacent!

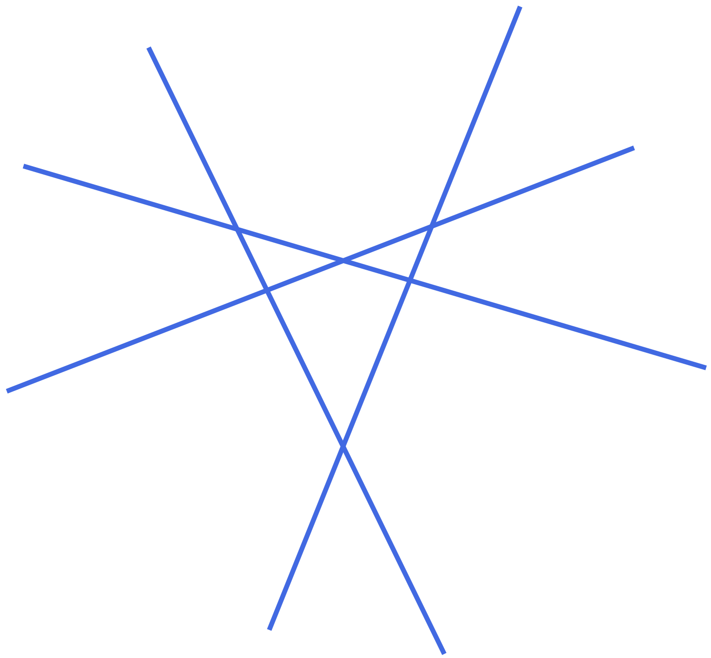
This construction can also be reversed. ← Why?

Hence, $f_n = g_{n-1}$. Bingo!

This kind of proof is called a “bijective proof”. We’ll see more in next lesson.

Q3: Slicing a Plane

Let n be a positive integer. A total of n lines are drawn on the plane so that no two are parallel and no three are concurrent. What is the number of regions do these lines divide the plane into?

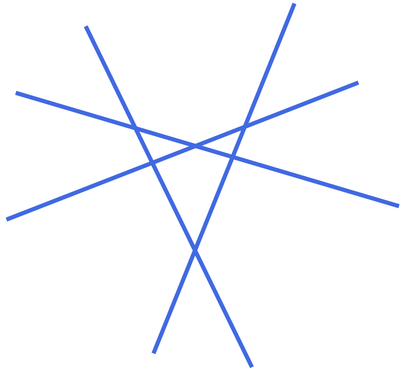


Q3: Slicing a Plane

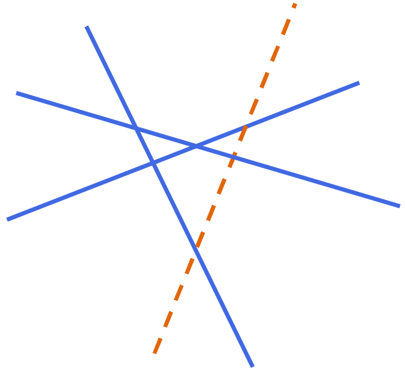
Let a_n be the number of regions when n non-parallel, non-concurrent lines are drawn.

$$a_n = a_{n-1} + \boxed{}$$

true for all $n \geq 2$.



Reason: Take a config with n lines. Redraw one of the lines. Whenever it "hits" one of the previous $n-1$ lines, a new region is created. One more region is created when the redrawing finishes.



Now, finding a_n is just algebra:

$$\begin{aligned} a_n = a_{n-1} + n = a_{n-2} + (n-1) + n = \cdots &= a_1 + 2 + 3 + \cdots + n \\ &= 1 + \frac{n(n+1)}{2}. \end{aligned}$$

Section – II

Recursion with Pictures

COM

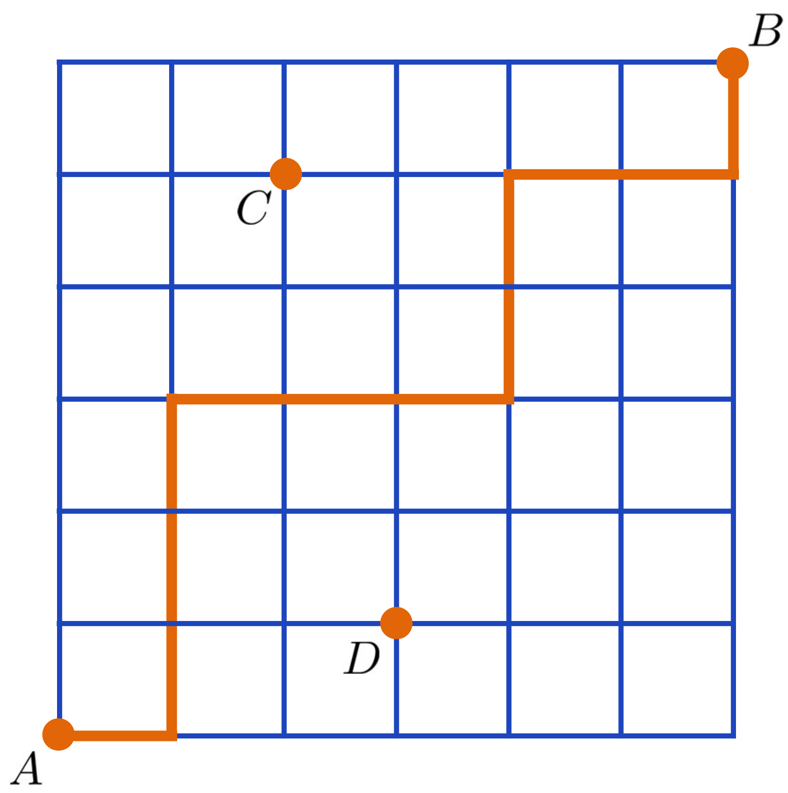
DEC

Q4: Going around the town

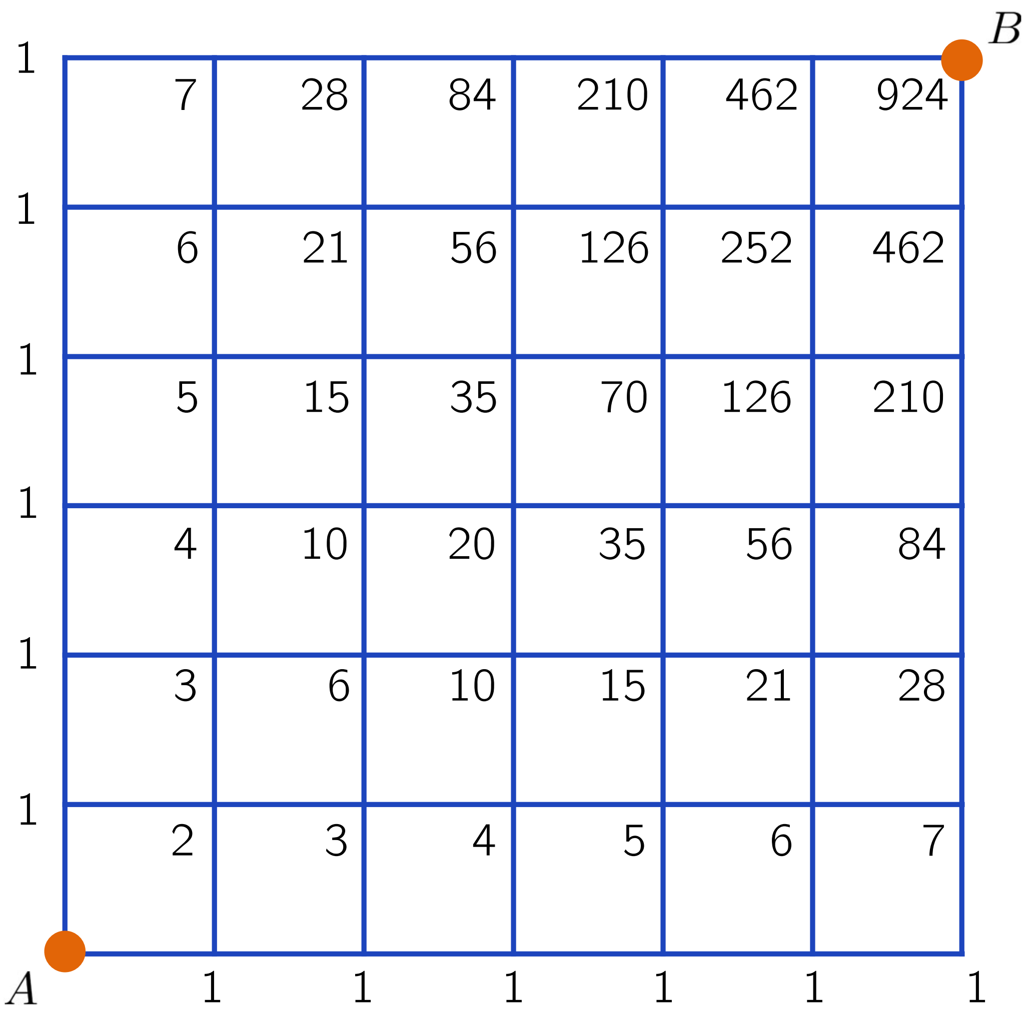
The road map of Townsville is given. Josh wants to go from A to B by going only northwards or eastwards.

(a) In how many ways can Josh go?

(b) If Josh is not allowed to go through C and D, in how many ways can he walk from A to B?

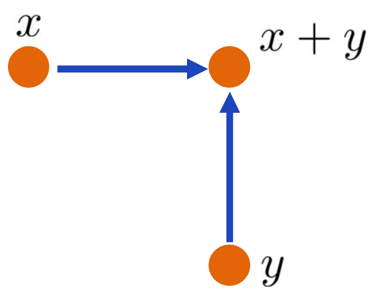


Q4: Going around the town



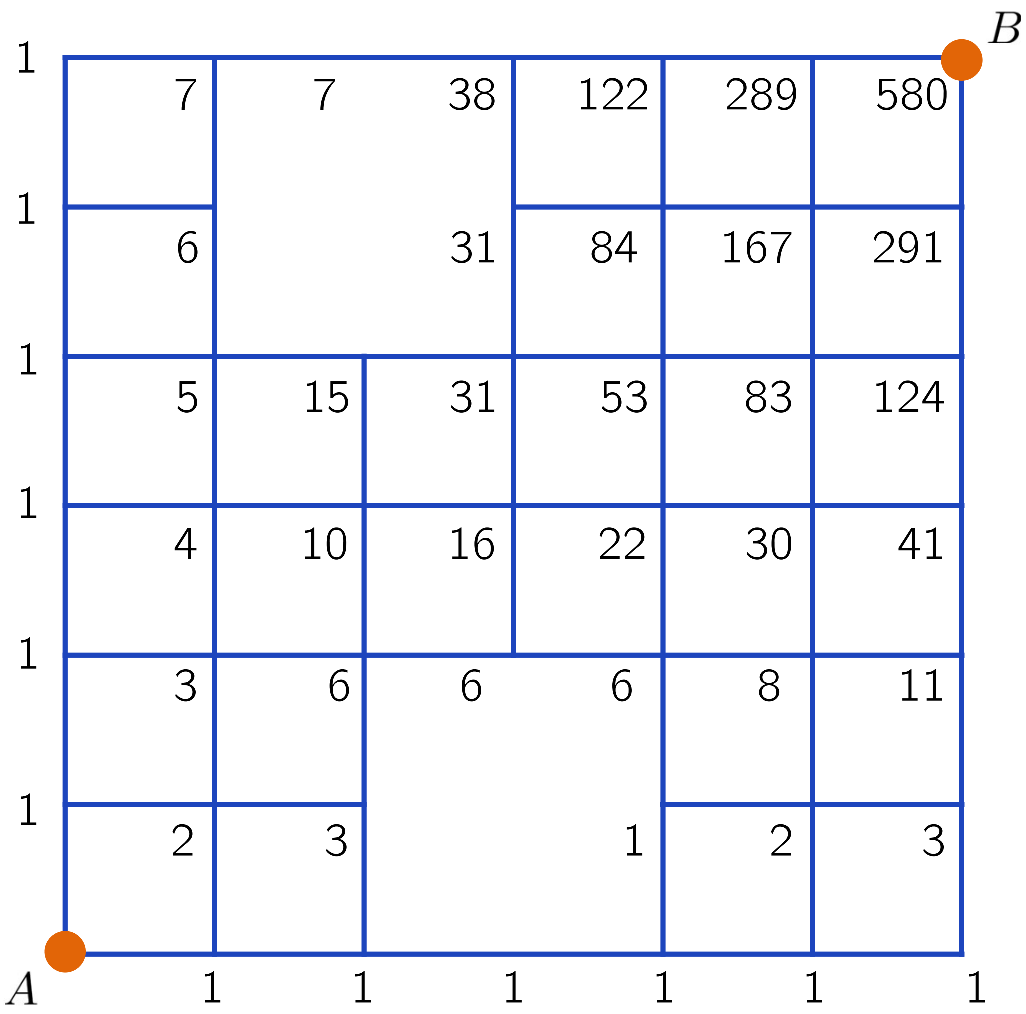
Solution (a)

To each intersection, attach the number of ways to arrive to that intersection using the given moves. Then, we have the following recurrence:



Now, we just compute until we get the value at B!

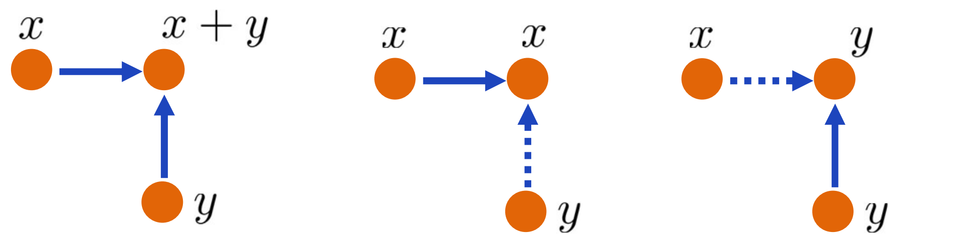
Q4: Going around the town



Solution (b)

Simply remove C and D, and redo the recurrence.

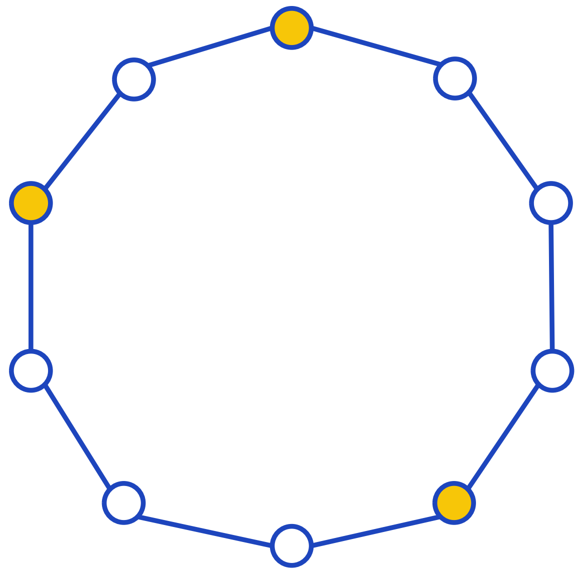
Note the following change when we don't have edges:



This problem has a nice "bijective" solution. We'll see in the next lesson!

Q5: Standing around a table

Ten people sit around a round table. When a signal is given, some people immediately stand up. In how many ways can they stand so that no two adjacent people stand next to each other?

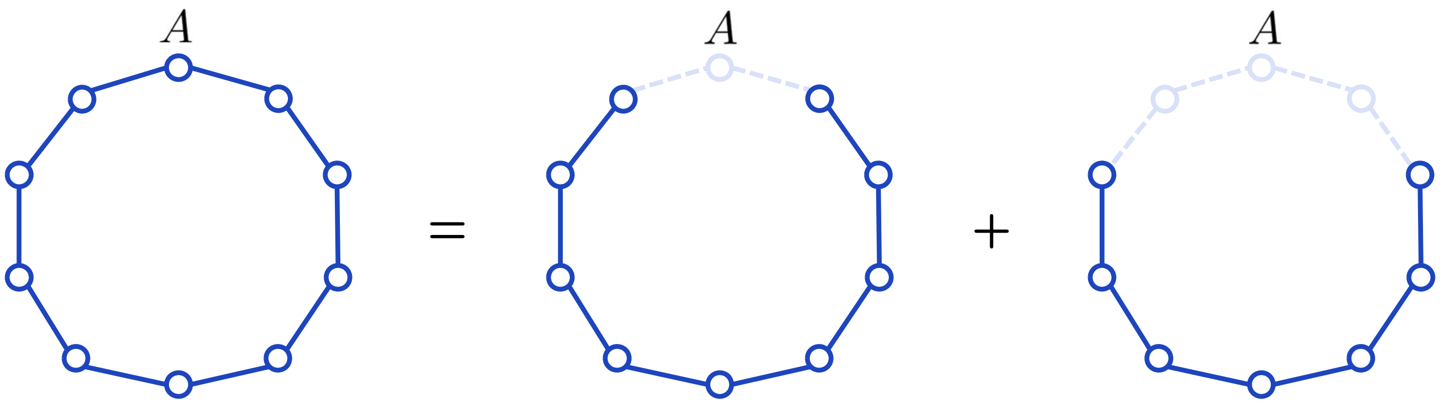


Q5: Standing around a table

Solution: Pick a person A. We have two cases: A remains seated, or A stands. So,

Answer = # ways with A sitting + # ways with A standing

Pictorially, we have this:



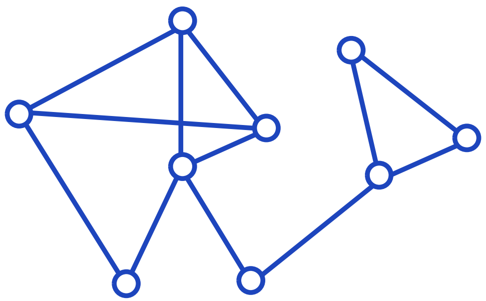
So, # ways with A sitting = # ways to arrange 9 people with no adjacent stands = g_9

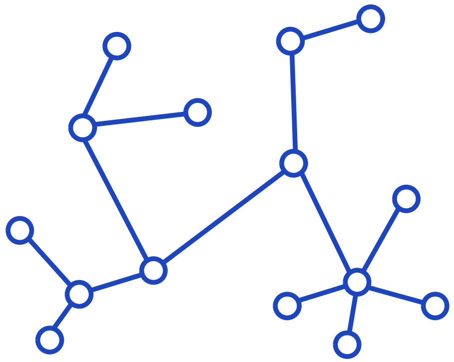
ways with A standing = # ways to arrange 7 people with no adjacent stands = g_7

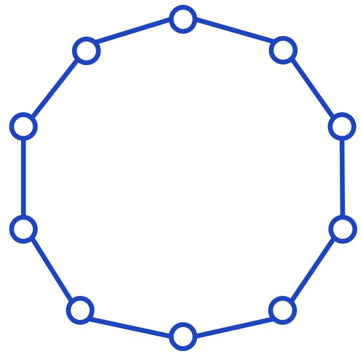
Thus, by Q2, the answer is 89 + 34 = 123.

Counting Independent Sets

Consider a social network with many users. Any two users are either friends or not friends. Then, the network can be represented by figures like this:







A (possibly empty) subset of users is called *independent* if no two of them are friends with each other.

These networks are called *graphs*. Users are called *vertices*, and friendships *edges*.

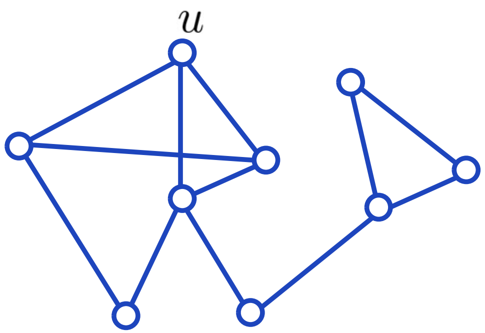
Question: How to count the number of independent subsets?

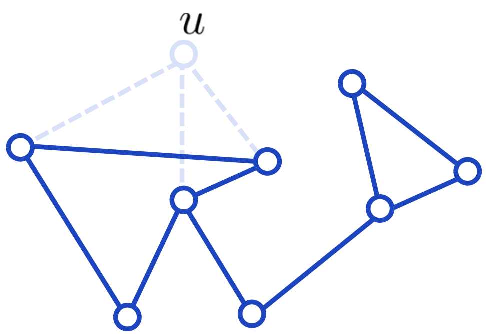
Counting Independent Sets

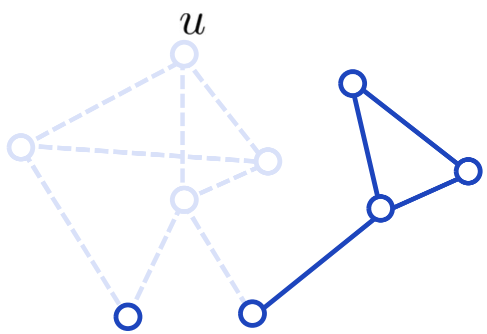
Answer: Recurrence!

Pick any user u. Then, we have the following recurrence:

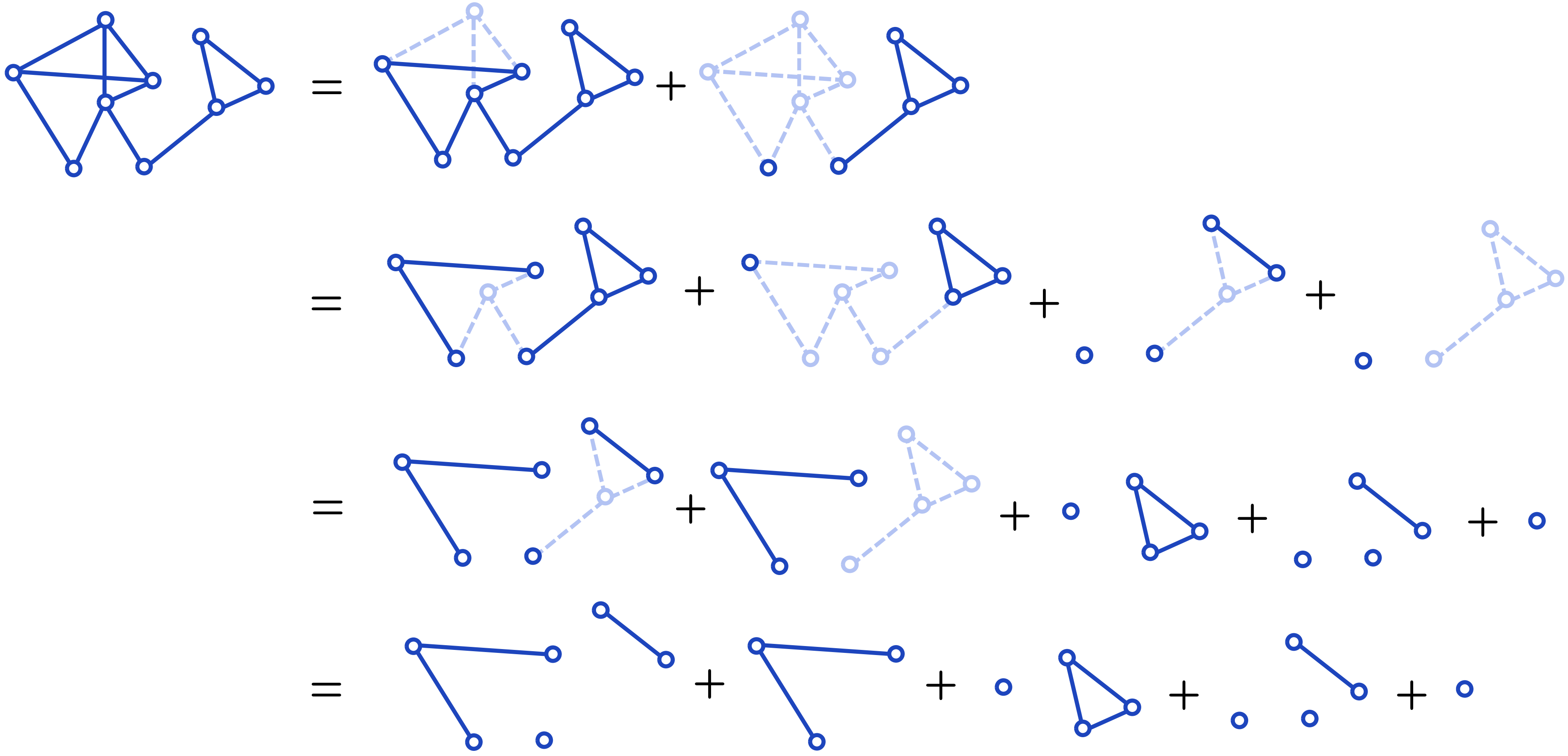
$$\#\text{ of independent sets} = \begin{array}{c}\#\text{ of independent sets}\\ \text{with } u \text{ removed}\end{array} + \begin{array}{c}\#\text{ of independent sets with } u\\ \text{and its friends removed}\end{array}$$







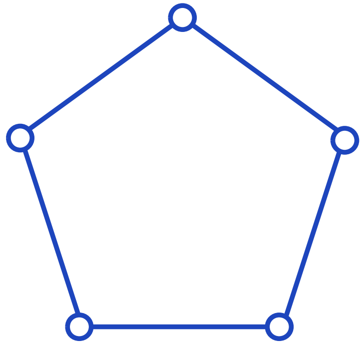
Counting Independent Sets



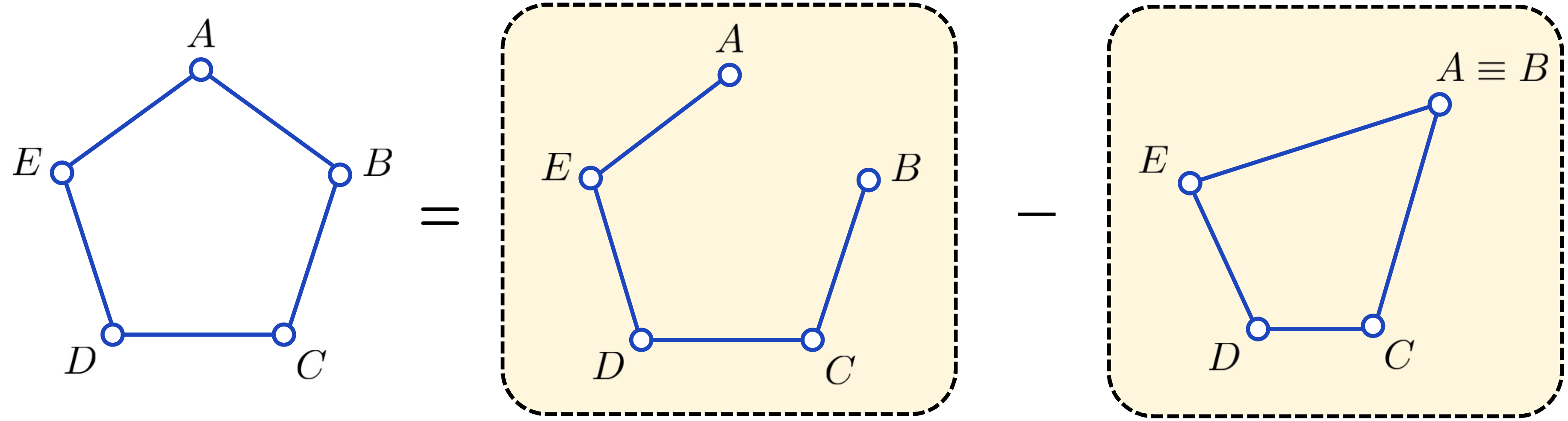
$$= 5 \cdot 3 \cdot 2 + 5 + 2 \cdot 4 + 2 \cdot 2 \cdot 3 + 2$$

Q6: Colouring a pentagon

Let $k \geq 3$ be a positive integer. In how many ways can we paint the vertices of a regular pentagon so that adjacent vertices receive different colours?



Q6: Colouring a pentagon



Colourings which allow A and B to have same colour.

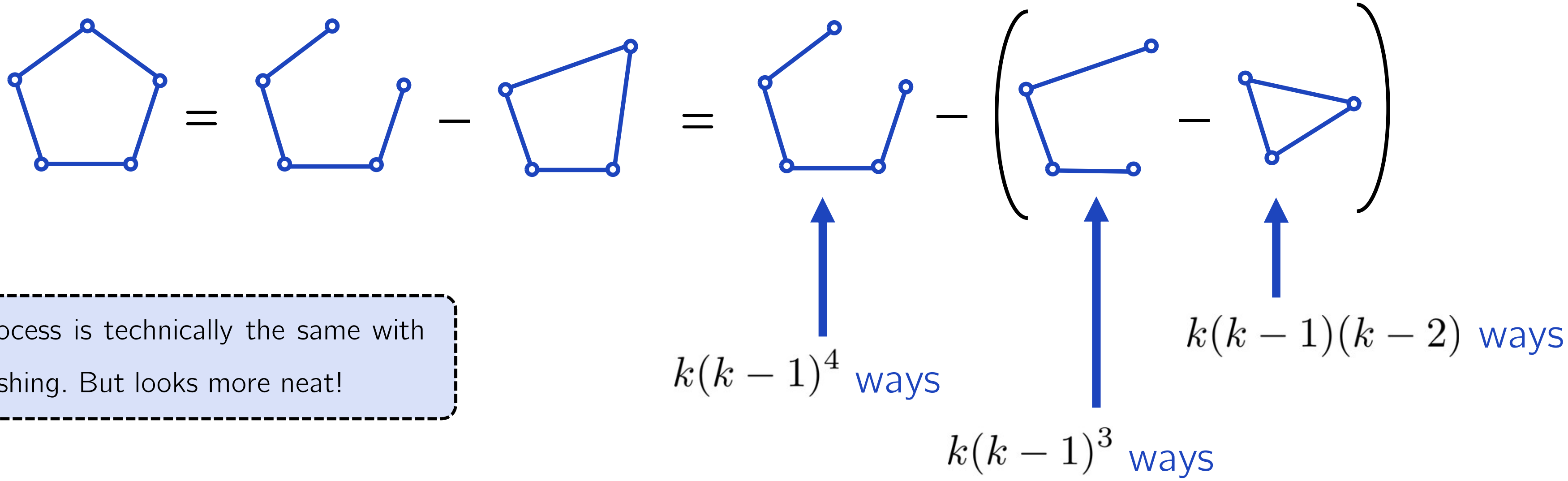
$k(k-1)^4$ ways

Colourings in which A and B have the same colour.

Use the same trick again!!

Q6: Colouring a pentagon

Solution



This process is technically the same with case-bashing. But looks more neat!

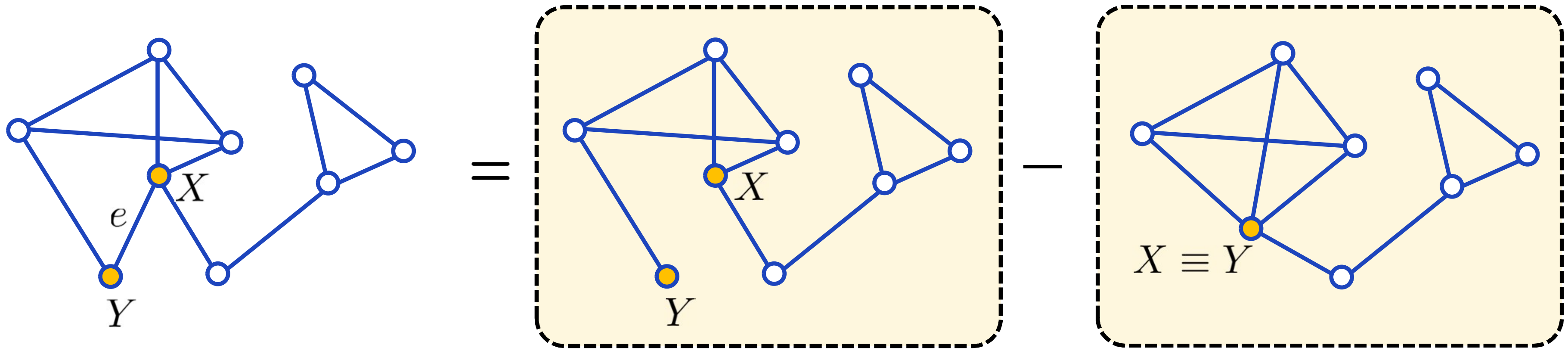
Therefore, the answer is $k(k-1)^4 - k(k-1)^3 + k(k-1)(k-2) = k(k-1)(k-2)(k^2-2k+2)$

Exercise: Compute the number of ways to colour the vertices of an n-gon with k colours.

Chromatic recurrence

The same process can be used to compute the number of proper colourings of any graph (network).

Fix any edge e with ends X and Y.



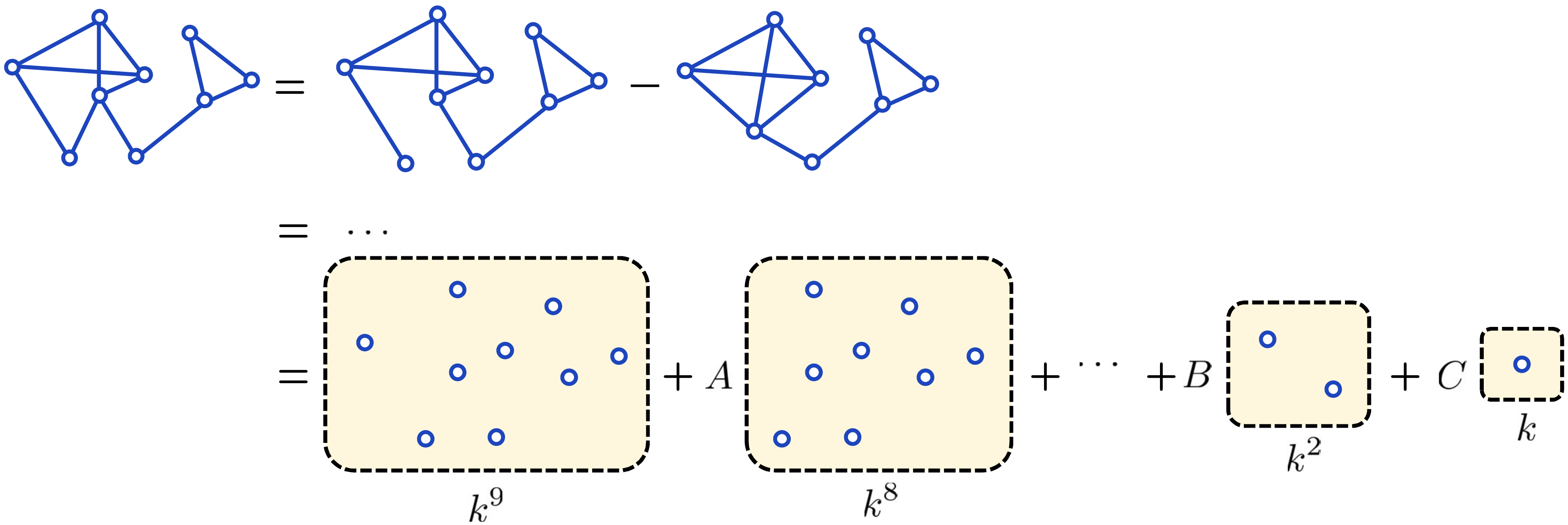
Colourings which allow X and Y to have same colour.

Colourings in which X and Y have the same colour.

Question: Why is this a recurrence? Which quantity decreases? ← Number of edges decreases

Chromatic recurrence

Therefore, as we recurse more and more, we will be left with graphs with no edges in the end.

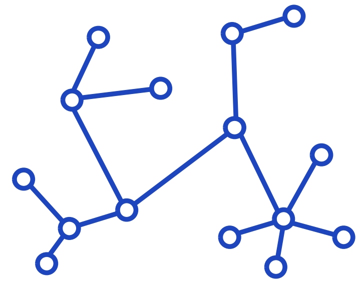


So, number of ways to colour a graph in k colours is always a polynomial of k.

Question: This recurrence takes too many steps. How many? ← about 2^{12}

Chromatic recurrence

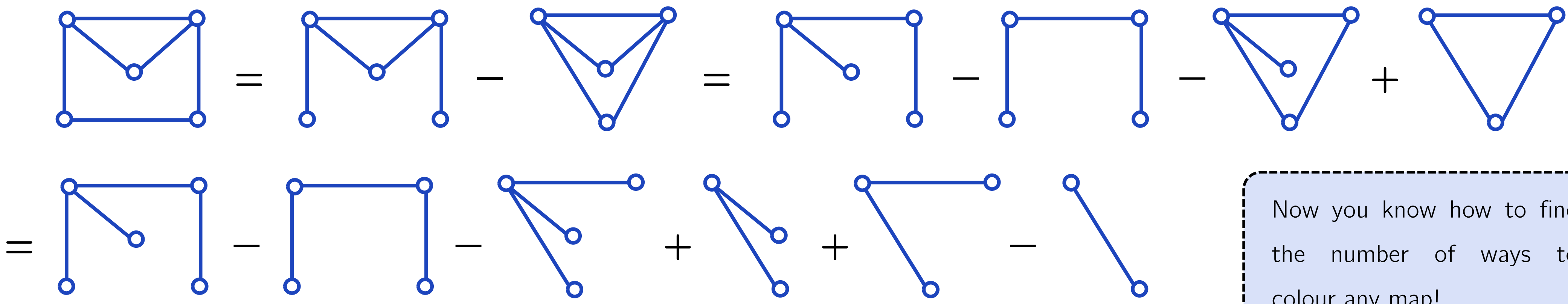
In practice, we recurse until there are no cycles. At this point, computation is easy.



Question: Find the number of ways to colour this graph with k colours.

Answer: $k(k-1)^{14}$

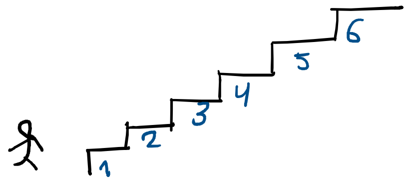
As an example, we can compute the number of colourings of an "envelope" as follows:



$$= k(k-1)^4 - 2k(k-1)^3 + 2k(k-1)^2 - k(k-1)$$

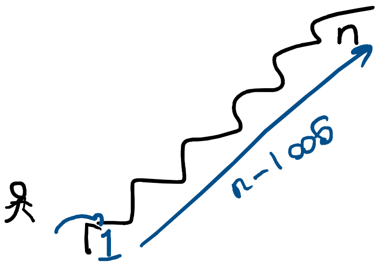
Now you know how to find the number of ways to colour any map!

3. Jumpy wants to walk up a flight of 6 stairs. In each step, he can climb either 1, 2 or 3 stairs. In how many different ways can he climb the flight of stairs?



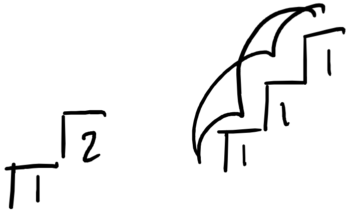
လှေကား: ထစ် n ထစ် ရှိရင် တက်နိုင်သောနည်း: $= a_n$.

a_n ရဲ့ recurrence relation ကဘာလဲ။



$a_n =$ ပထမခြေလှမ်း: က 1 step ဖြစ်ခဲ့ရင် ကျန် $+$ 〃 2 steps ဖြစ်ရင် ကျန် $+$ 〃 3 steps ဖြစ်ရင် ကျန်

$$\boxed{a_n = a_{n-1} + a_{n-2} + a_{n-3}} \leftarrow (n \geq 4)$$



a_1	a_2	a_3	a_4	a_5	a_6
1	2	4	7	13	$\boxed{24}$

Section – III

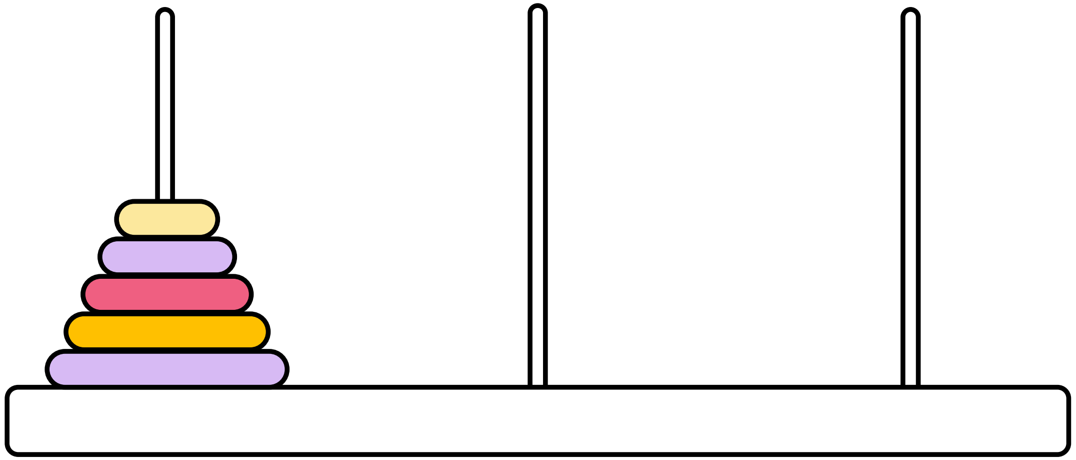
Recursive Algorithms

COM

DEC

Q7: Hanoi Tower

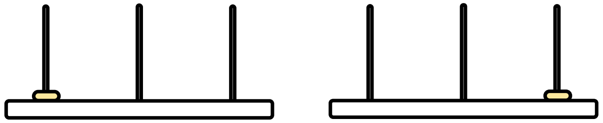
In the game *Hanoi Tower*, you are given three pegs, and n discs of increasing sizes are stacked through one of the pegs. Sizes of the discs decrease from top to bottom. In a *move*, you may take a disc from the top of a peg place it on a peg as long as you are not stacking a larger disc on top of a smaller one. You want to move all the discs to a peg that is different from the starting one. What is the minimum number of moves needed to do this?



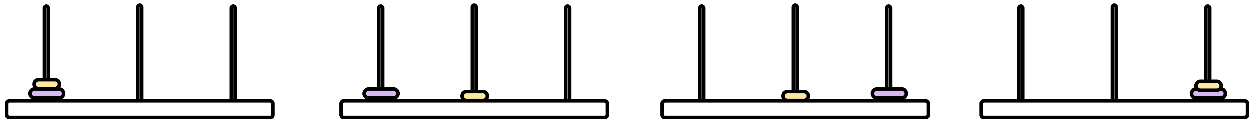
Q7: Hanoi Tower

Let's try for small values of n.

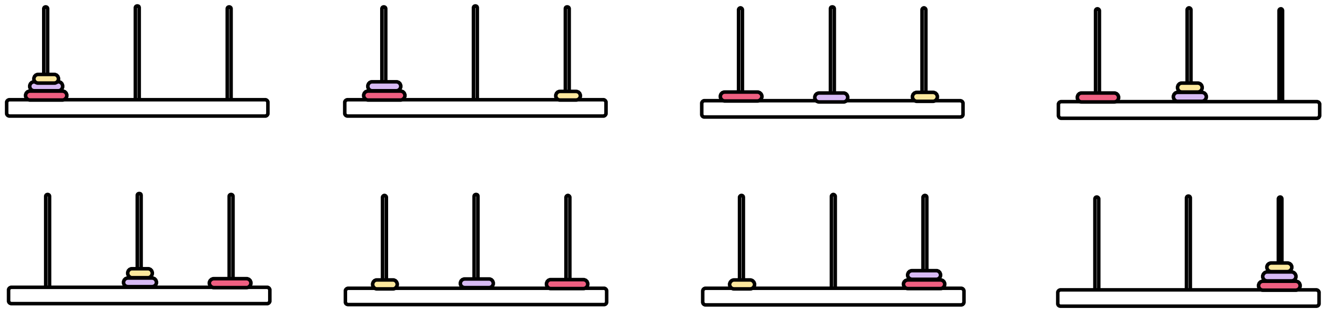
For $n = 1$, min. number of moves = 1



For $n = 2$, min. number of moves = 3



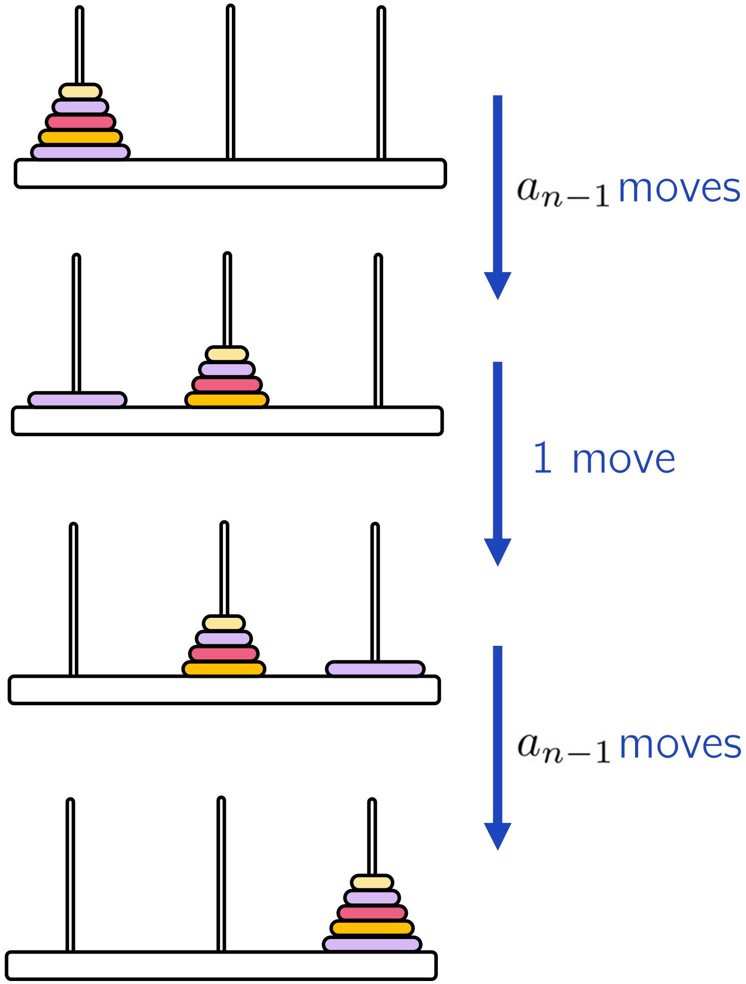
For $n = 3$, min. number of moves = 7



This is too early to say.

But... the answer = $2^n - 1$?

Q7: Hanoi Tower



Solution

Let a_n be the minimum number of moves needed.

- To finish the game, we need to move the largest disc onto the last peg.
- So, we need to move the first $n-1$ discs onto the middle peg.
- Then, move the largest disc to the last peg.
- Then, put the $n-1$ discs from middle onto last peg.

All of these needs to be done. So, we have

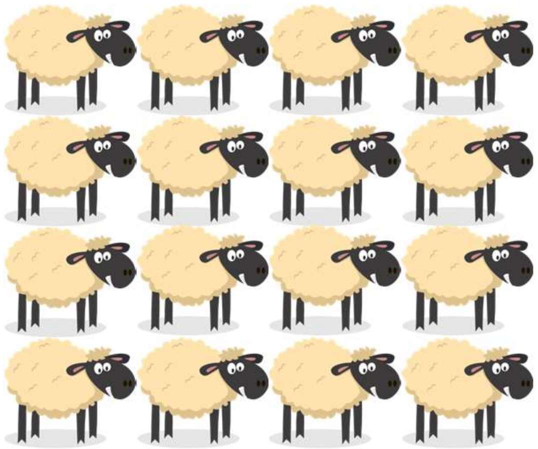
$$a_n = 2a_{n-1} + 1$$

true for all $n \geq 2$.

Solving the recurrence with $a_1 = 1$ gives $a_n = 2^n - 1$.

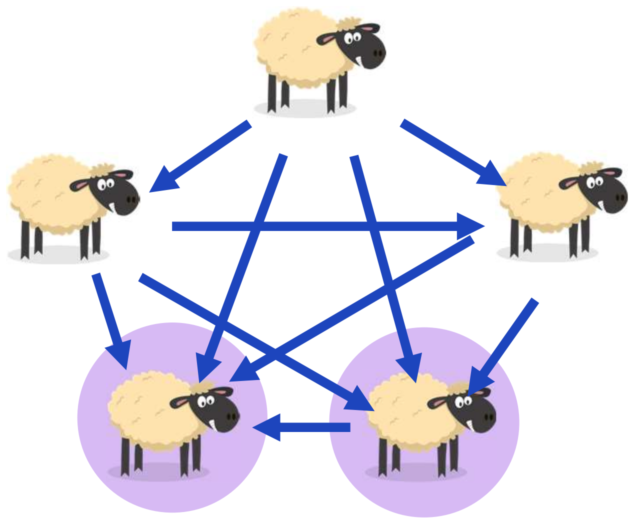
Q8: Sheep Sort

There are sixteen sheep with distinct weights, and you have a sheep-scale. You can put two sheep on the sheep-scale at a time, and the sheep-scale tells you which sheep is heavier. We would like to arrange the sheep in increasing order by weights. Can we do this by using the sheep-scale less than 50 times?



Q8: Sheep Sort

The most obvious way: Compare any two of the sheep. Then, find the heaviest sheep, then the 2nd heaviest sheep, etc.



Question: How many times do we have to use the sheep scale for this? ← $\binom{16}{2} = 120$ Too many!

Q8: Sheep Sort

Let a_n be the minimum number of times that we need to use the scale for 2^n sheep.

Obviously $a_1 = 1$. Let's try our best for 4 sheep.

- Break the sheep into 2 herds of equal size.
- We know how to arrange each herd.
- Now, with one comparison, we can find the lightest sheep!
- With another comparison, we can find the 2nd lightest sheep.
- With another comparison, we can find the 3rd lightest sheep.

We used the scale 5 times in total. Therefore, $a_2 \leq 5$.

We only have $\leq$ and not $=$!

We only showed that 5 moves sufficie.

We haven't showed that 5 moves are necessary!

Q8: Sheep Sort

Similarly, we can do the same for 8 sheep.

- Break the sheep into 2 herds of equal size.
- We know how to arrange each herd. This takes $2 \times 5 = 10$ moves.
- Now, with one comparison, we can find the lightest sheep!
- With another comparison, we can find the 2nd lightest sheep.
- ...
- With another comparison, we can find the heaviest sheep.

We used the scale $2 \times 5 + 7 = 17$ times in total. Therefore, $a_3 \leq 17$.

Question: What if we did the same procedure for 16 sheep? ← $2 \times 17 + 15 = 49$ moves!

Q8: Sheep Sort

Solution

Let a_n be the minimum number of times needed to use the sheep-scale for 2^n sheep. Then, we may follow the following procedure to sort 2^n sheep:

- Break the sheep into two herds of 2^{n-1} sheep each.
- Sort the sheeps in these herds using a_{n-1} moves each.
- Then, we can find the lightest sheep by using the scale once.
- Then, we can find the 2nd lightest sheep by using the scale one more time.
- …
- Lastly, we can find second heaviest and the heaviest sheep by using the scale one more time.

In total, we used $2a_{n-1} + (2^n - 1)$ moves. Hence, we have $a_n \leq 2a_{n-1} + (2^n - 1)$. Therefore,

$$a_4 \leq 2a_3 + 15 \leq 2(2a_2 + 7) + 15 \leq 2(2(2a_1 + 3) + 7) + 15 = 49.$$

Summary of Main Ideas

Main Philosophy: If you can make an object smaller in some way, we are done by induction!

Recursion in Sequences: Let a_n be the number of things that you are interested in. Main idea is to find the relation between a_n and previous terms. To do so, we may

- take an object of interest and divide it into many cases,
- find the smaller version of the object inside the object of interest.

Recursion with Pictures: Same idea with sequences. The resulting pictures should be "smaller" than the original picture in some sense.

To solve recurrences, always try substituting backwards!