Generating functions

Atli FF

December 2, 2024

School of Computer Science
Reykjavík University

This module's material

- Counting
- Introductory combinatorics
- State transfer matrices
- Generating functions

Combinatorics

Combinatorics

Combinatorics is study of countable discrete structures.

Combinatorics

Combinatorics is study of countable discrete structures.

Generic enumeration problem: We are given an infinite sequence of sets $A_1, A_2, \ldots A_n, \ldots$ which contain objects satisfying a set of properties. Determine

$$a_n := |A_n|$$

for general n.

The two rules

- If you have n options, you can pick one in n ways clearly.
- If you can choose one of n options **or** one of m options then there are $n + m$ ways to do so.
- If you can choose one of n options **and** one of m options then there are nm ways to do so.
- Most things can be derived from these two rules really.

Examples

- Factorial

$$n! = 1 \cdot 2 \cdot 3 \cdots n$$

- Binomial coefficient

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Examples

- Factorial

$$n! = 1 \cdot 2 \cdot 3 \cdots n$$

- Binomial coefficient

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

  Number of ways to choose k objects from a set of n objects, ignoring order.

Binomial properties

Properties

- $$\binom{n}{k} = \binom{n}{n-k}$$
- $$\binom{n}{0} = \binom{n}{n} = 1$$
- $$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

Pascal triangle!

$$
\begin{array}{ccccccccccccccccccccc}
 & & & & & & & & & & \binom{0}{0} & & & & & & & & & & \\
 & & & & & & & & & \binom{1}{0} & & \binom{1}{1} & & & & & & & & \\
 & & & & & & & & \binom{2}{0} & & \binom{2}{1} & & \binom{2}{2} & & & & & & \\
 & & & & & & & \binom{3}{0} & & \binom{3}{1} & & \binom{3}{2} & & \binom{3}{3} & & & & & \\
 & & & & & & \binom{4}{0} & & \binom{4}{1} & & \binom{4}{2} & & \binom{4}{3} & & \binom{4}{4} & & & & \\
 & & & & & \binom{5}{0} & & \binom{5}{1} & & \binom{5}{2} & & \binom{5}{3} & & \binom{5}{4} & & \binom{5}{5} & & & \\
 & & & & \binom{6}{0} & & \binom{6}{1} & & \binom{6}{2} & & \binom{6}{3} & & \binom{6}{4} & & \binom{6}{5} & & \binom{6}{6} & & \\
 & & & \binom{7}{0} & & \binom{7}{1} & & \binom{7}{2} & & \binom{7}{3} & & \binom{7}{4} & & \binom{7}{5} & & \binom{7}{6} & & \binom{7}{7} & \\
 & & \binom{8}{0} & & \binom{8}{1} & & \binom{8}{2} & & \binom{8}{3} & & \binom{8}{4} & & \binom{8}{5} & & \binom{8}{6} & & \binom{8}{7} & & \binom{8}{8} \\
 & \binom{9}{0} & & \binom{9}{1} & & \binom{9}{2} & & \binom{9}{3} & & \binom{9}{4} & & \binom{9}{5} & & \binom{9}{6} & & \binom{9}{7} & & \binom{9}{8} & & \binom{9}{9} \\
\binom{10}{0} & & \binom{10}{1} & & \binom{10}{2} & & \binom{10}{3} & & \binom{10}{4} & & \binom{10}{5} & & \binom{10}{6} & & \binom{10}{7} & & \binom{10}{8} & & \binom{10}{9} & & \binom{10}{10}
\end{array}
$$

Other useful identities

- $$\sum_{k=0}^{n} \binom{n}{k} = 2^n$$
- $$\sum_{k=0}^{n} \binom{n}{k}^2 = \binom{2n}{n}$$
- The infamous "hockey stick sum"

$$\sum_{k=0}^{m} \binom{n+k}{k} = \binom{n+m+1}{m}$$

Example

How many rectangles can be formed on a $m \times n$ grid?

Example

How many rectangles can be formed on a $m \times n$ grid?

The rectangle is defined by covering some x-interval and some y-interval, we can choose these independently. If we have values $1, \ldots, k$ and want to choose an interval, how can we do this?

Example

How many rectangles can be formed on a $m \times n$ grid?

The rectangle is defined by covering some x-interval and some y-interval, we can choose these independently. If we have values $1, \ldots, k$ and want to choose an interval, how can we do this?

Well we want to choose 2 out of k values, without ordering since that's chosen by the values of the numbers. Thus this gives $\binom{k}{2}$ choices. Final answer is thus $\binom{n}{2}\binom{m}{2}$.

Multinomial

What if we have many objects with the same value?

Multinomial

What if we have many objects with the same value?

- Number of permutations on n objects, where n_i is the number of objects with the i-th value.(Multinomial)

$$\binom{n}{n_1, n_2, \ldots, n_k} = \frac{n!}{n_1! n_2! \cdots n_k!}$$

Multinomial

What if we have many objects with the same value?

- Number of permutations on n objects, where n_i is the number of objects with the i-th value.(Multinomial)

$$\binom{n}{n_1, n_2, \ldots, n_k} = \frac{n!}{n_1! n_2! \cdots n_k!}$$

- Number of way to choose k objects from a set of n objects with, where each value can be chosen more than once.

$$\binom{n+k-1}{k} = \frac{(n+k-1)!}{k!(n-1)!}$$

Balls and boxes

How many different ways can we divide k identical balls into n boxes?

Balls and boxes

How many different ways can we divide k identical balls into n boxes?

- Same as number of nonnegative solutions to

$$x_1 + x_2 + \ldots + x_n = k$$

Balls and boxes

How many different ways can we divide k identical balls into n boxes?

- Same as number of nonnegative solutions to

$$x_1 + x_2 + \ldots + x_n = k$$

- Let's imagine we have a bit string consisting only of 1 of length $n + k - 1$

$$\underbrace{1\,1\,1\,1\,1\,1\,1\ldots1}_{n+k-1}$$

Stars and bars

- Choose $n-1$ bits to be swapped for 0

$$1 \ldots 1 0 1 \ldots 1 0 \ldots 0 1 \ldots 1$$

Stars and bars

- Choose $n-1$ bits to be swapped for 0

$$\underbrace{1\ldots1}_{x_1}0\underbrace{1\ldots1}_{x_2}0\ldots0\underbrace{1\ldots1}_{x_n}$$

- Then total number of 1 will be k, each 1 representing an each element, and separated into n groups

Stars and bars

- Choose $n-1$ bits to be swapped for 0

$$\underbrace{1\ldots1}_{x_1}0\underbrace{1\ldots1}_{x_2}0\ldots0\underbrace{1\ldots1}_{x_n}$$

- Then total number of 1 will be k, each 1 representing an each element, and separated into n groups
- Number of ways to choose the bits to swap

$$\binom{n+k-1}{n-1} = \binom{n+k-1}{k}$$

Example

How many different lattice paths are there from $(0, 0)$ to (n, m)?

Example

How many different lattice paths are there from $(0, 0)$ to (n, m)?

- There is 1 path to $(0, 0)$

Example

How many different lattice paths are there from $(0, 0)$ to (n, m)?

- There is 1 path to $(0, 0)$
- There is 1 path to $(1, 0)$ and $(0, 1)$

Example

How many different lattice paths are there from $(0, 0)$ to (n, m)?

- There is 1 path to $(0, 0)$
- There is 1 path to $(1, 0)$ and $(0, 1)$
- Paths to $(1, 1)$ is the sum of number of paths to $(0, 1)$ and $(1, 0)$.

Example

How many different lattice paths are there from $(0, 0)$ to (n, m)?

- There is 1 path to $(0, 0)$
- There is 1 path to $(1, 0)$ and $(0, 1)$
- Paths to $(1, 1)$ is the sum of number of paths to $(0, 1)$ and $(1, 0)$.
- Number of paths to (i, j) is the sum of the number of paths to $(i - 1, j)$ and $(i, j - 1)$.

Example

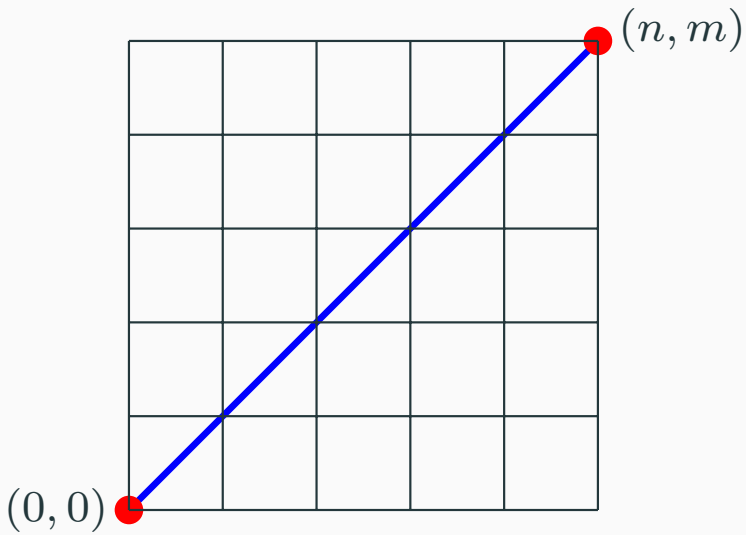
How many different lattice paths are there from $(0, 0)$ to (n, m)?

- There is 1 path to $(0, 0)$
- There is 1 path to $(1, 0)$ and $(0, 1)$
- Paths to $(1, 1)$ is the sum of number of paths to $(0, 1)$ and $(1, 0)$.
- Number of paths to (i, j) is

$$\binom{i+j}{i}$$

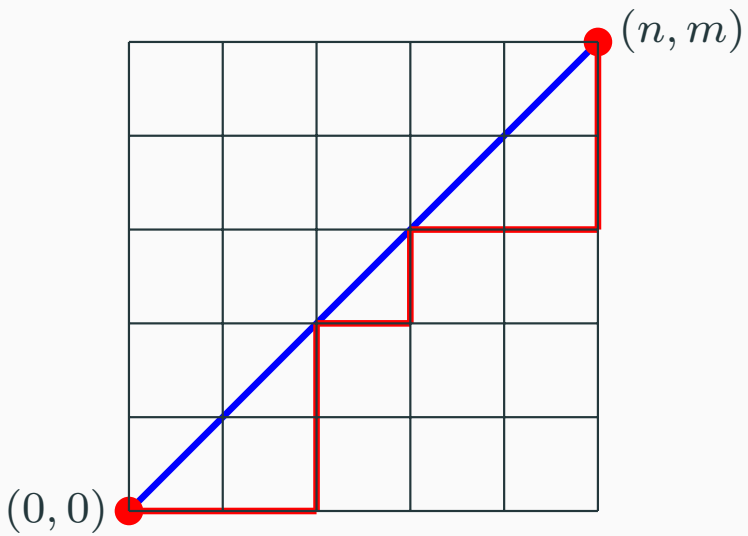
Dyck

What if we are not allowed to cross the main diagonal?



Dyck

What if we are not allowed to cross the main diagonal?

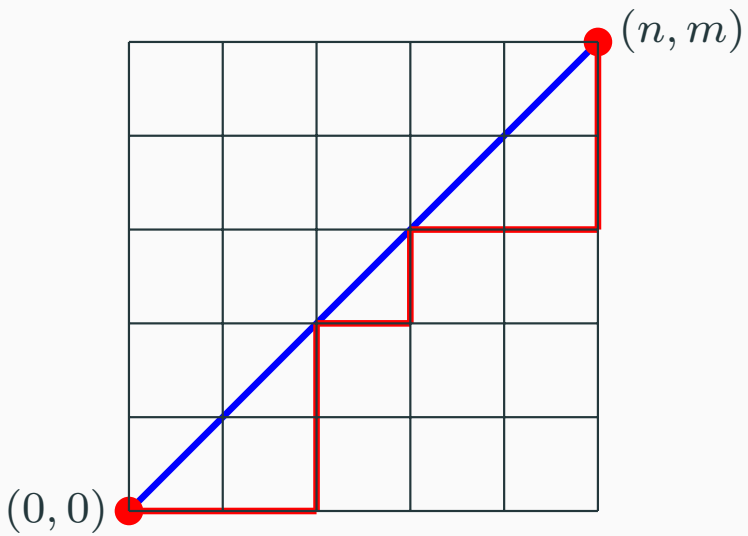


- The number of paths from $(0, 0)$ to (n, m)

$$C_n = \frac{1}{n+1}\binom{2n}{n}$$

Dyck

What if we are not allowed to cross the main diagonal?



- The number of paths from $(0, 0)$ to (n, m)

$$C_n = \frac{1}{n+1}\binom{2n}{n}$$

- C_n are known as Catalan numbers.
- Many problems involve solutions given by the Catalan numbers.

More Catalan

- Number of different ways $n+1$ factors can be completely parenthesized.

$$((ab)c)d \quad (a(bc))d \quad (ab)(cd) \quad a((bc)d) \quad a(b(cd))$$

More Catalan

- Number of different ways $n+1$ factors can be completely parenthesized.

$$((ab)c)d \quad (a(bc))d \quad (ab)(cd) \quad a((bc)d) \quad a(b(cd))$$

- Number of stack sortable permutations of length n.

More Catalan

- Number of different ways $n+1$ factors can be completely parenthesized.

$$((ab)c)d \quad (a(bc))d \quad (ab)(cd) \quad a((bc)d) \quad a(b(cd))$$

- Number of stack sortable permutations of length n.
- Number of different triangulations convex polygon with $n+2$ sides

More Catalan

- Number of different ways $n+1$ factors can be completely parenthesized.

$$((ab)c)d \quad (a(bc))d \quad (ab)(cd) \quad a((bc)d) \quad a(b(cd))$$

- Number of stack sortable permutations of length n.
- Number of different triangulations convex polygon with $n+2$ sides
- Number of full binary trees with $n+1$ leaves.

More Catalan

- Number of different ways $n+1$ factors can be completely parenthesized.

$$((ab)c)d \quad (a(bc))d \quad (ab)(cd) \quad a((bc)d) \quad a(b(cd))$$

- Number of stack sortable permutations of length n.
- Number of different triangulations convex polygon with $n+2$ sides
- Number of full binary trees with $n+1$ leaves.
- And a lot more.

Working up to π

- If we have n options, but m of them are forbidden, we are left with $n - m$ ways to choose.
- This means if we have n options and m options, but k of them are in common between them, that's $n + m - k$ different options. This can be rewritten as $|A \cup B| = |A| + |B| - |A \cap B|$.
- We can keep going with this for larger numbers of sets, say options from A, B and C. Then to know our total number of options we have to subtract what is in common. But if we do that thrice we will have removed what is in common between all three, which we need to add back.
- In total this becomes

$$|A \cup B \cup C| = |A|+|B|+|C|-|A \cap B|-|A \cap C|-|B \cap C|+|A \cap B \cap C|$$

PIE

- By generalizing this to n sets using induction you can get an equation for the size of the union of n sets $A_1, \ldots, A_n$. This equation is usually called the principle of inclusion-exclusion, or PIE.

$$\left|\bigcup_{i=1}^{n} A_i\right| = \sum_{\substack{J \subseteq \{1,\ldots,n\} \\ J \neq \varnothing}} (-1)^{|J|-1} \left|\bigcap_{j \in J} A_j\right|$$

Permutations

- Let's look at permutations. Often it's convenient to denote them by a **bijective** function $\pi : [n] \to [n]$ where $[n] \coloneqq \{1, \ldots, n\}$.
- As mentioned before there are $n!$ permutations of $[n]$, but there's much more to it. A *fixed point* of a permutation π is an x such that $\pi(x) = x$. Things we could consider regarding this include: how many permutations have no fixed points? Such permutation sare called derangements.
- To count the number of derangements, we'll use a classic trick called counting the complement.
- We know how many permutations there are, so we'll count how many permutations have *some* fixed point and subtract that from the total.

Fixed points

- Let A_i denote all permutations of $[n]$ where i is a fixed point. Then we can permute everything but i as usual, so $|A_i| = (n-1)!$. But if we want several values to be fixed, say everything in J, then we can use the same argument to get that there are $(n - |J|)!$ such permutations. Thus $\bigcap_{j \in J} A_j$ has $(n - |J|)!$ elements.
- Thus we can now use PIE to count the size of $\bigcup_{j \in J} A_j$, which is what we want. We just have to keep in mind that the intersection of k different A_j will appear $\binom{n}{k}$ times. Thus

$$\sum_{i=1}^{n} (-1)^{i-1} \binom{n}{i} (n-i)! = n! \sum_{i=1}^{n} (-1)^{i-1} \frac{1}{i!}$$

Derangement

- Thus the number of derangements is just $n!$ minus this. We denote this by $!n$ and get

$$!n = n! \sum_{i=0}^{n} \frac{(-1)^i}{i!}$$

- A fun side effect of this is that $n!/!n$ very quickly approaches e as n grows.

A counting problem

- Let us consider an example from Kattis of a harder counting problem. We get $1 \leq n \leq 1000$ and $1 \leq c \leq 10000$ and want to find the number of permutations of n elements with c inversions. The answer should be returned modulo $10^9 + 7$.
- Where do we start?
- Can we solve the problem for n, c in terms of smaller values?

Subdivision

- Denote the answer for n, c by $f(n, c)$ and let us write our permutation as a list $\pi(1), \ldots, \pi(n)$. We can make a permutation of $n+1$ elements by adding $n+1$ to this list. If we add it at location k (and shift what's right of it) we add $n-k+1$ inversions.
- We can also do this backwards, starting with a permutation on n elements and removing n. Let us consider permutations on n lements with c inversions and the value n is t-th last in the list.
- Then if we remove n we fix $t-1$ inversions, so we end up with a permutation on $n-1$ elements with $c-t+1$ inversions. Thus for this particular t we get $f(n-1, c-t+1)$. By considering all t we get:

$$f(n, c) = \sum_{i=0}^{\min(c, n-1)} f(n-1, c-i)$$

Dynamic programming

- Now we have a recursive formula in terms of (n, c), so we can use dynamic programming! But the time complexity is $\mathcal{O}(nc^2)$ because there are nc states and each takes c time to calculate.
- We use the same trick as two days ago, we precalculate the sum so each value will only take constant time. For this we need bottom-up dynamic programming. So we define $p(n, c)$ as

$$p(n, c) = \sum_{i=0}^{c} f(n, i)$$

- Then we can update both p and f in constant time and solve the problem in $\mathcal{O}(nc)$ which is good enough!

The code

```cpp
#include <bits/stdc++.h>
using namespace std;
const int mod = 1e9 + 7;

int n, c;
int dp[1005][10005];
int pr[1005][10005];

int main() {
    cin >> n >> c;
    if(c == 0) {
        cout << "1\n";
        return 0;
    }
    for(int i = 0; i <= c; ++i) dp[0][i] = 0;
    for(int i = 0; i < n; ++i) dp[i][0] = 1;
    for(int i = 1; i < n; ++i) {
        pr[i - 1][0] = dp[i - 1][0];
        for(int j = 1; j <= c; ++j) {
            pr[i - 1][j] = (pr[i - 1][j - 1] + dp[i - 1][j]) % mod;
        }
        for(int j = 1; j <= c; ++j) {
            dp[i][j] = pr[i - 1][j];
            if(j >= i + 1) {
                dp[i][j] -= pr[i - 1][j - i - 1];
                dp[i][j] = (dp[i][j] % mod + mod) % mod;
            }
        }
    }
    cout << dp[n - 1][c] << '\n'; }
```

State transfer matrix

Linear recurrences

- Let us define a linear recurrence. We say a sequence of numbers follows a linear recurrence if they are defined using some initial values and then the subsequent values being defined by a linear equation of the form

$$a_n = \lambda_1 a_{n-1} + \lambda_2 a_{n-2} + \cdots + \lambda_k a_{n-k} + \lambda$$

  with the λ_is constant. k is said to be the degree of the recurrence relation.

- An example of this is the fibonacci numbers, where $\lambda_1 = \lambda_2 = 1$, $\lambda = 0$ and $a_1 = a_2 = 1$.

Calculating a sequence

- The trick is that using matrices we can calculate values in these sequences very fast. We can get the n-th number in $\mathcal{O}(k^3 \log(n))$ time. Since k is usually very small, almost always < 10, this is quite fast. For example for fibonacci we have $k = 2$. The trick is the following equation:

$$\begin{pmatrix} \lambda_1 & \lambda_2 & \cdots & \lambda_{k-1} & \lambda_k & \lambda \\ 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 \end{pmatrix}^n \begin{pmatrix} a_k \\ a_{k-1} \\ \vdots \\ a_2 \\ a_1 \\ 1 \end{pmatrix} = \begin{pmatrix} a_{n+k} \\ a_{n+k-1} \\ \vdots \\ a_{n+2} \\ a_{n+1} \\ 1 \end{pmatrix}$$

Linear recurrence

- So for example we can calculate the fibonacci numbers using:

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^n \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

- We omitted one row there since $\lambda = 0$. But similarly if $a_1 = 1, a_2 = 2$ and $a_n = 3a_{n-1} - a_{n-2} + 6$ the corresponding calculation would be

$$\begin{pmatrix} 3 & -1 & 6 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}^n \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix}$$

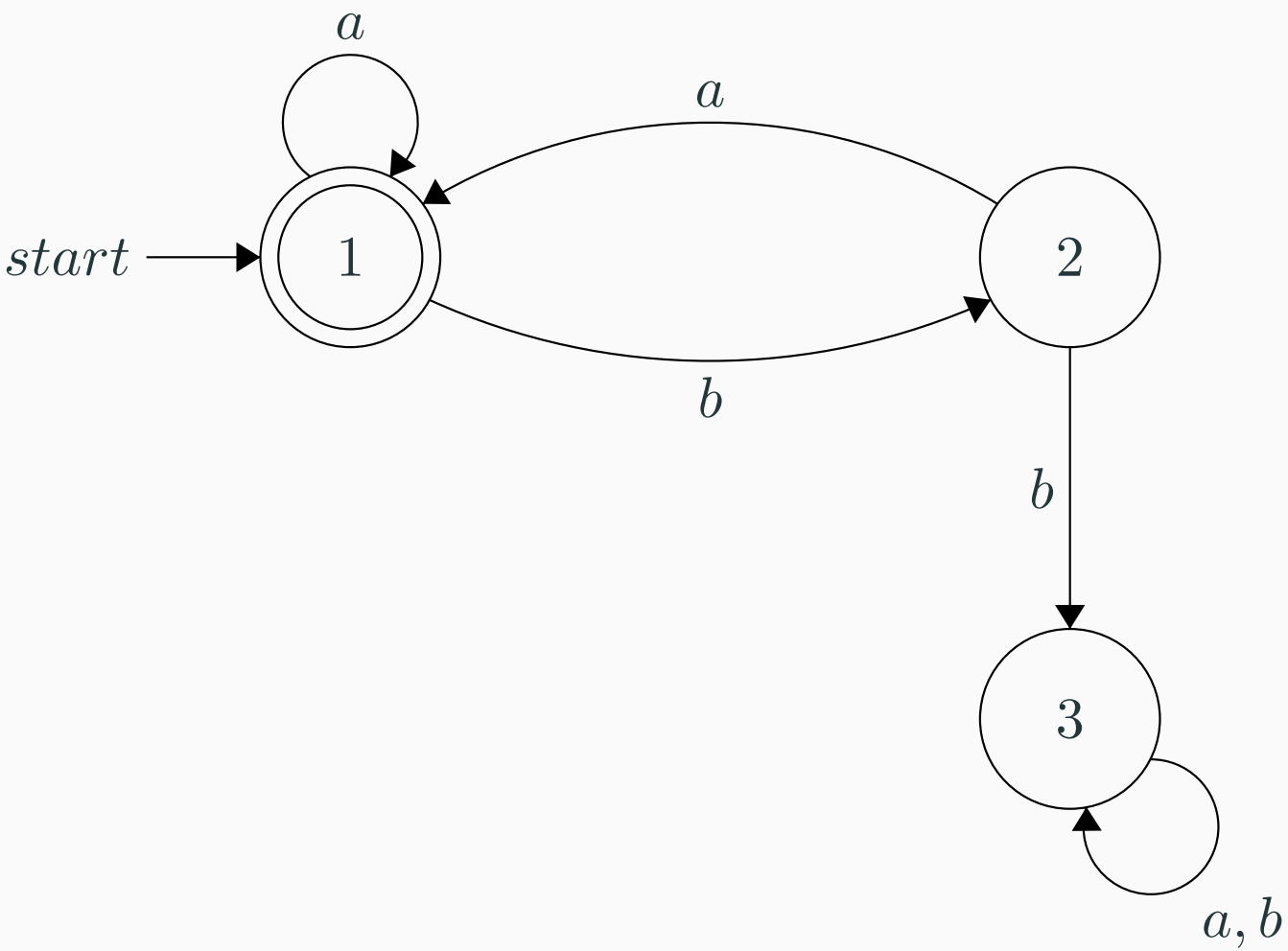
Transfer matrix

- But isn't this rather specific?
- Well, the number of words a DFA accepts of a given length follows a formula like this! We can make the matrix simply equal to the adjacency matrix of the DFA. Say we have n possible states and let A be an $n \times n$ matrix such that $A_{i,j}$ gives the number of ways to go from state i to state j. Then entry (i, j) in A^k will give the number of ways to go from state i to state j in k steps.
- You can do induction or various other things to prove this, those interested can try to work that out for themselves.

Transfer matrix

- Thus by modeling various counting tasks as moving between a discrete set of states, we can use binary exponentiation on matrices to calculate the results quickly and efficiently.
- In the simplest case we can be walking in a graph, with states being vertices and the matrix just being the adjacency matrix of the graph.
- In the homework you will both find problems like this, and less straightforward problems.
- Let us look at the example of Fibonacci words, that is to say all concatenations of a and ba.

Transfer matrix



Transfer matrix

- This gets us the following matrix:

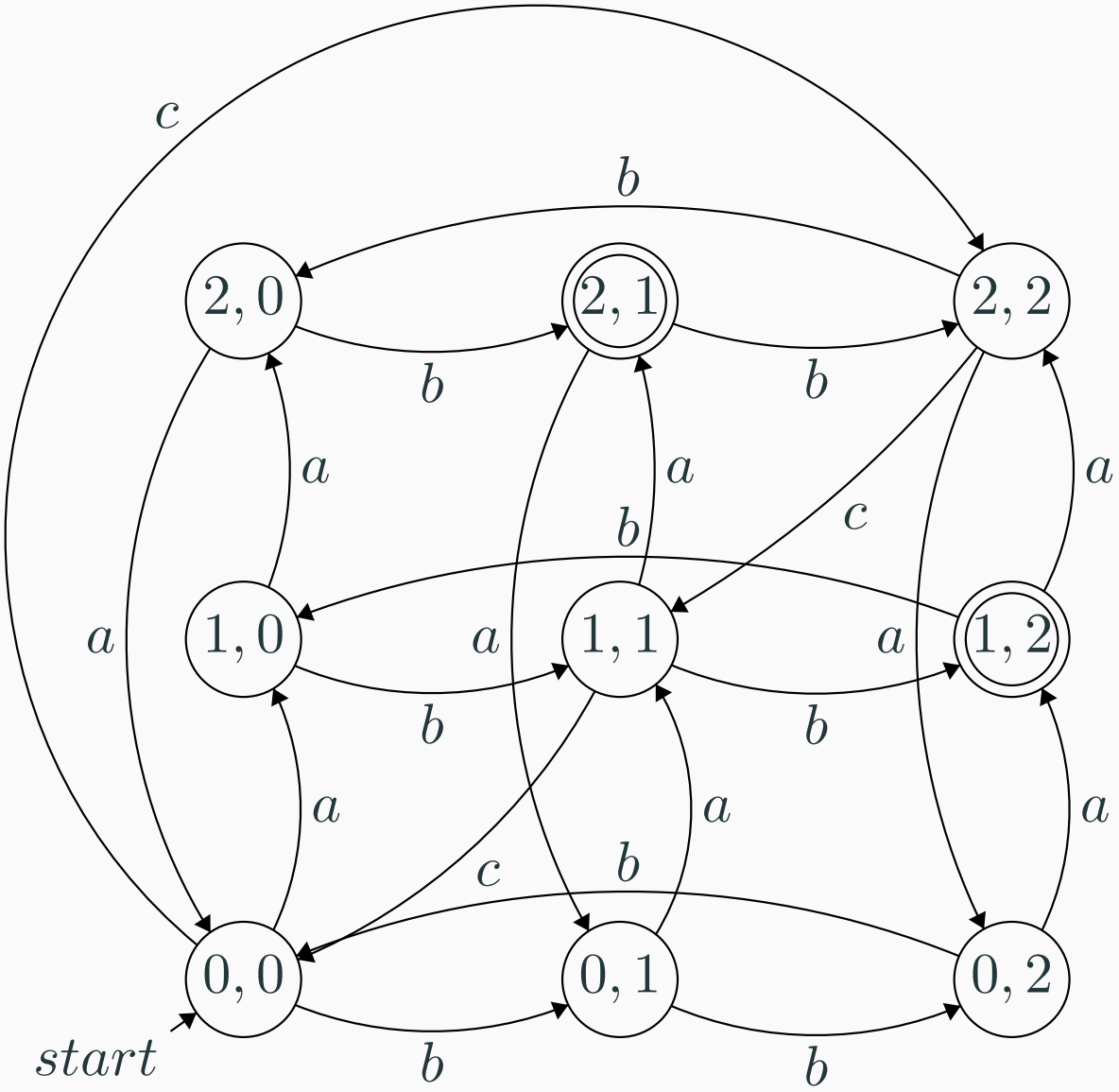
$$M = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 2 \end{pmatrix}$$

- The top left 2×2 matrix suffices, so the state transfer matrix usually won't be of minimal size, but this will work to compute the Fibonacci numbers by binary exponentiation. We need to use the initial vector $v = (1, 0, 0)$ in this case since in the base case is that after zero steps we can be in the initial state and nowhere else. We then read off the first element of $M^n v$ since the first state is the only accepting state.

Larger example

- We consider a slightly more involved example. We denote the number of a in our word by k_a and define k_b, k_c similarly. Then we want to count words such that k_a, k_b, k_c are different modulo 3.
- We make a state for each value of the pair $(k_a - k_c \pmod 3, k_b - k_c \pmod 3)$, which are 9 states. We could just do (k_a, k_b, k_c) instead, but that's 27 states which is hard to fit on a slide. Then we need this pair to be $(1, 2)$ or $(2, 1)$.

Larger example



Larger example

- From this we could then make a 9×9 matrix that lets us count the number of such words.
- But we will now see a method that allows even more analysis, not just counting.

Generating functions

A bag of numbers

- Often in combinatorics we are working with infinite sequences of numbers, for example the fibonacci numbers $0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots$
- It would be convenient if we could have just a single thing to work with instead of infinitely many numbers
- Fortunately that's possible!

A bag of numbers

- Instead of working with $a_0, a_1, a_2, \ldots$ we work with $a_0 + a_1 x + a_2 x^2 + \ldots$
- But how in the world is this simpler? Well, often this simplifies quite a lot. Like a whole lot.
- For example the fibonacci numbers result in the simple expression $\frac{x}{1-x-x^2}$!
- But why?

Fibonacci generating function

- Let $F(x) = x + 2x^2 + 3x^3 + 5x^4 + 8x^5 + 13x^6$ and so on, i.e. the fibonacci generating function
- We know that $f_n = f_{n-1} + f_{n-2}$, so $f_n x^n = x f_{n-1} x^{n-1} + x^2 f_{n-2} x^{n-2}$ for any x
- If we sum this over all $n \geq 2$ we get $F(x) - x = xF(x) + x^2 F(x)$ which can be solved to get $F(x) = \frac{x}{1-x-x^2}$
- If you try to take the Taylor expansion of $F(x)$ you will see it's exactly the Fibonacci numbers!

Generating function

- Let $a_0, a_1, a_2, \ldots$ be a sequence
- With this all in mind we define the ordinary generating function of the sequence as

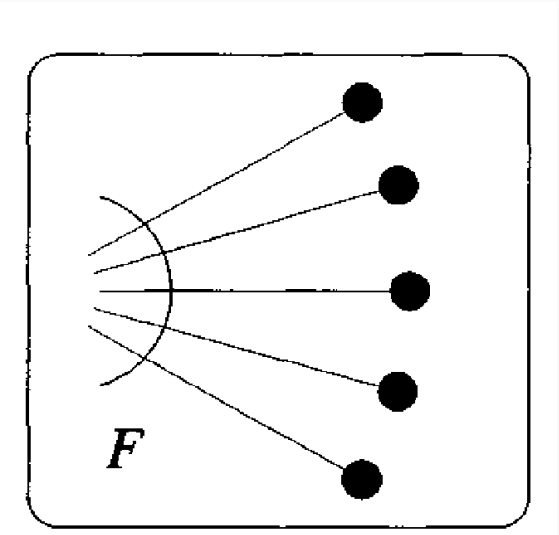
$$A(x) = \sum_{k=0}^{\infty} a_k x^k$$

- We similarly define the exponential generating function as

$$A(x) = \sum_{k=0}^{\infty} a_k \frac{x^k}{k!}$$

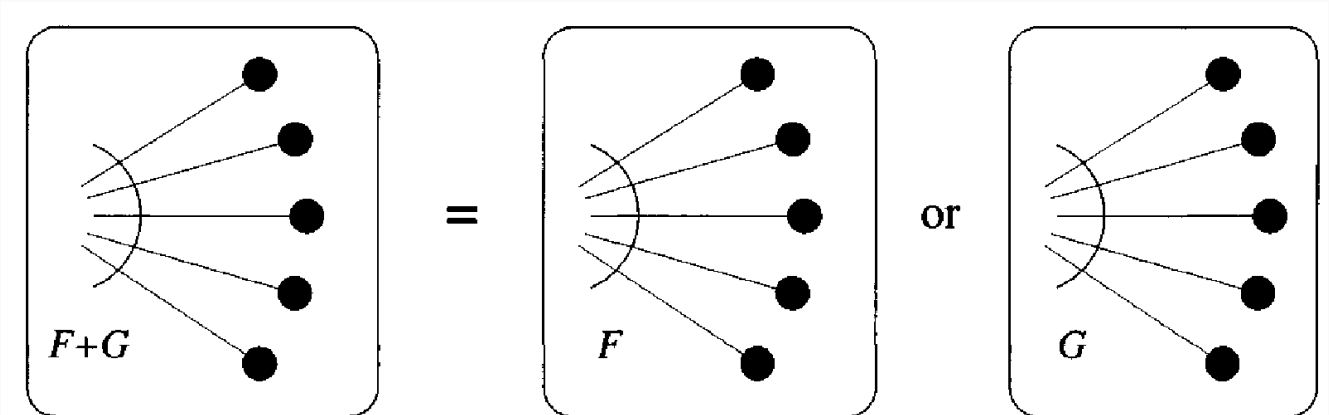
Generating function

- Suppose we have some $a_0, a_1, a_2, \ldots$ sequence and its generating function $a_0 + a_1 x + a_2^2 x^2 + \ldots$
- Then we have some kind of structure where there are a_0 ways of making that structure of size 0, a_1 ways to make a structure of size a_1 and so on
- When we work with ordinary generating functions we are considering unlabeled elements and exponential generating functions consider labeled elements, more on that later



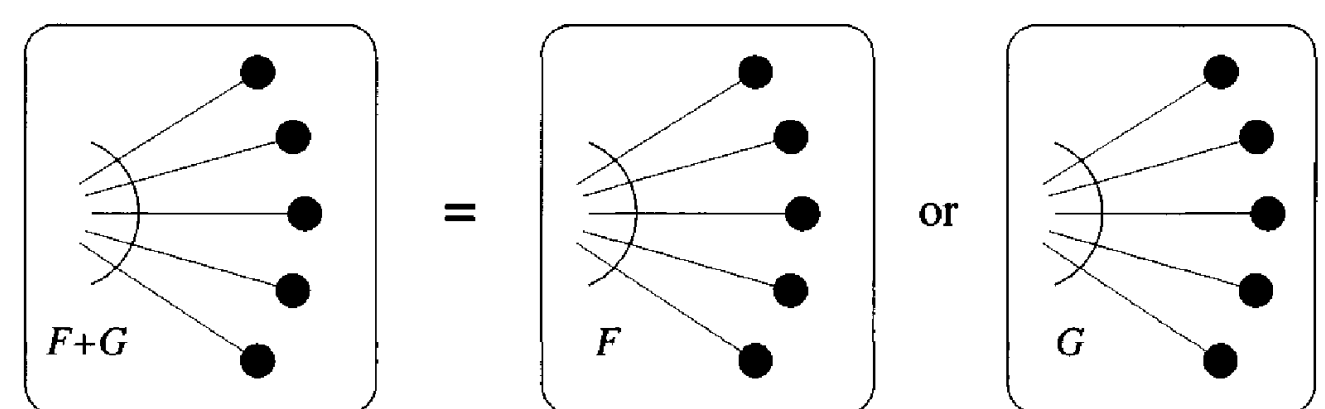
Operations on generating functions, sum

- Suppose you have to generating functions F, G counting different structures
- Can we somehow combine F, G to count new things? What does $F + G$ count?
- If we have $f_n + g_n$ then this means we can construct f on n items or g on n items
- So if f counts words with equally many 0s and 1s and g counts words with twice as many 0s as 1s, then $f + g$ would count words with equally many 0s and 1s or twice as many 0s as 1s



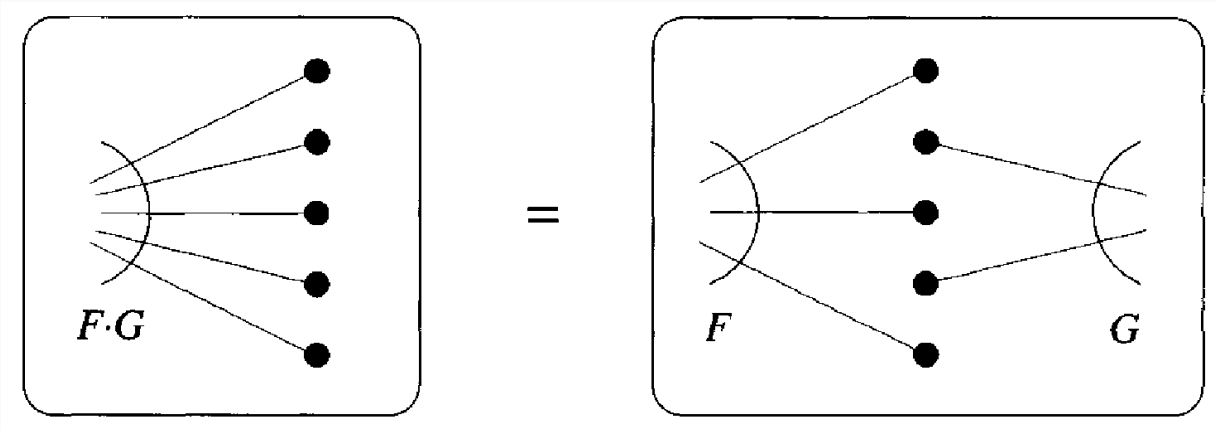
Operations on generating functions, sum

- Let $E(x)$ be the generating function of sets, so $e_k = 1$ for all k. Then $E(x) = \frac{1}{1-x}$.
- Let $E_o(x)$ be the generating function of odd sets, so $e_k = 0$ if k is even and $e_k = 1$ otherwise. Similarly let $E_e(x)$ be the generating function of even sets.
- Then $E = E_e + E_o$, which in ordinary generating functions means $\frac{1}{1-x} = \frac{1}{1-x^2} + \frac{x}{1-x^2}$
- In exponential generating functions it means $e^x = \cosh(x) + \sinh(x)$



Operations on generating functions, product

- What does $F \cdot G$ count?
- Then we have $\sum_{k=0}^{n} f_k g_{n-k}$ structures of size n
- This is exactly the number of ways to split n items into two groups, one of size k with an f-structure on it and the other with size $n-k$ with a g-structure on it
- This way we can make the generating function of subsets $\mathcal{P}$ by multiplying the generating function for sets E with itself, so $\mathcal{P}(x) = E(x) \cdot E(x) = \frac{1}{(1-x)^2}$



Operations on generating functions, product

- Let's consider an example using what we've seen so far.
- How many binary search trees are there? Well, to express what we want I will define $1(x) = 1$ and $X(x) = x$, so 1 denotes something that can only be empty, and X denotes something that can only be of size one.
- A binary search tree is either empty or contain a root. If it contains a root it then contains two sub-binary search trees as well. If we let $B(x)$ be the generating function of binary search trees, this means $B = 1 + X \cdot B \cdot B$.
- I won't go over the details here, but you can even solve this equation directly to get $B(x) = \frac{1-\sqrt{1-4x}}{2x}$ which solves to $B_k = \frac{1}{k+1}\binom{2k}{k}$.

Operations on generating functions, product

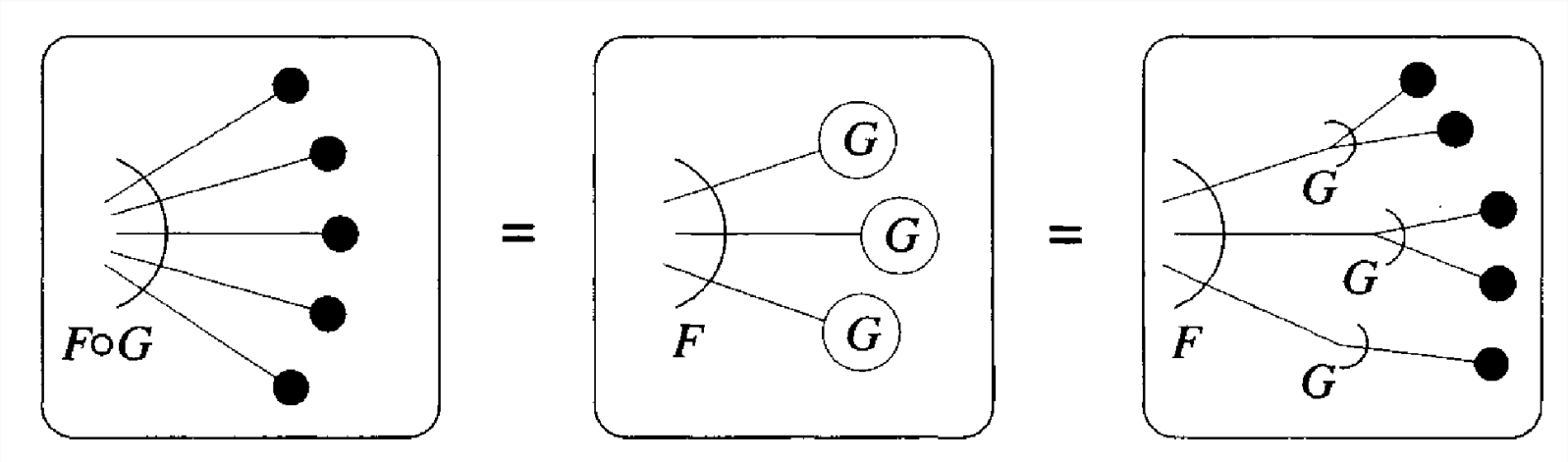
- There's even more we can do with just sum and product
- Consider the number of ways of writing a number as a sum of $3, 5$ or 7 (just one copy of each number)
- We can get this by multiplying out $(1 + x^3)(1 + x^5)(1 + x^7)$ since for each parenthesis we choose whether to use the number or not, and we add the exponents up
- We can expand upon this idea

Operations on generating functions, product

- If we want to allow the sum to use 3 many times we can then pick 0, 3, 6, 9 and so on, so if we want the earlier problem but with repetition we get
- $(1+x^3+x^6+x^9+\dots)(1+x^5+x^{10}+\dots)(1+x^7+x^{14}+\dots)$
- This can be rewritten as $\frac{1}{1-x^3}\frac{1}{1-x^5}\frac{1}{1-x^7}$
- But then how can we do it if we want the number of ways to write n as the sum of any positive integers? Well, we just multiply it all
- So we get the generating function of integer partitions, $I_p(X) = \prod_{k=1}^{\infty} \frac{1}{1-x^k}$

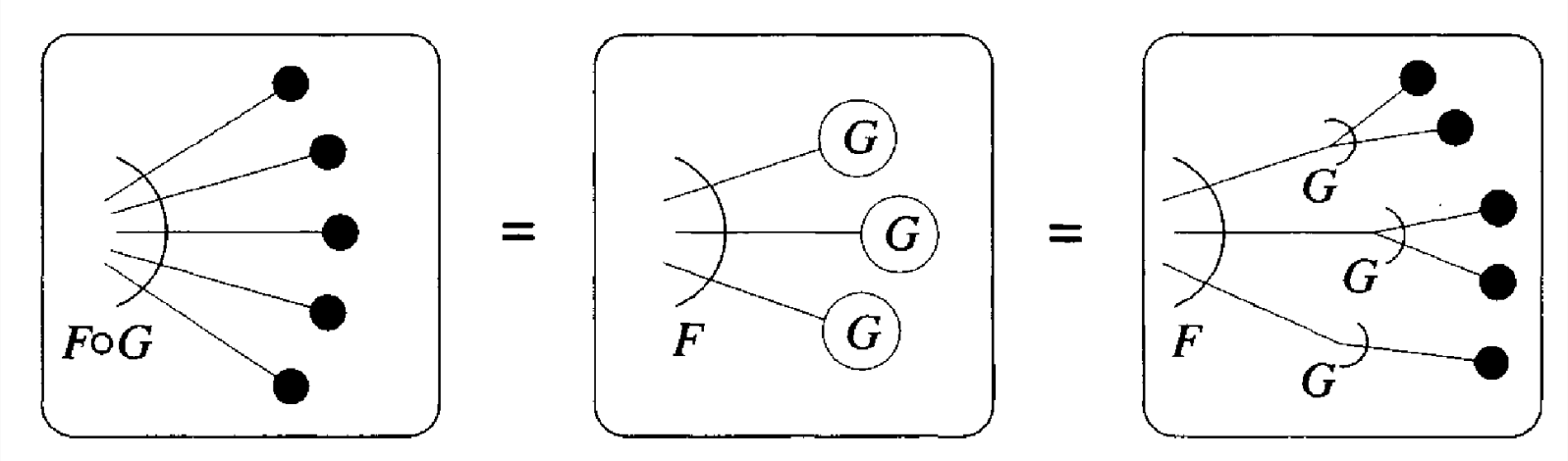
Operations on generating functions, composition

- Two more, what does $F \circ G$ count?
- This time we have an F-structure of G-structures
- A very common thing to compose with is E, sets. For example let T be the generating function of a rooted tree. A tree is a root along with a set of subtrees which are rooted. Thus $T(x) = xE(T(x))$, as an example.



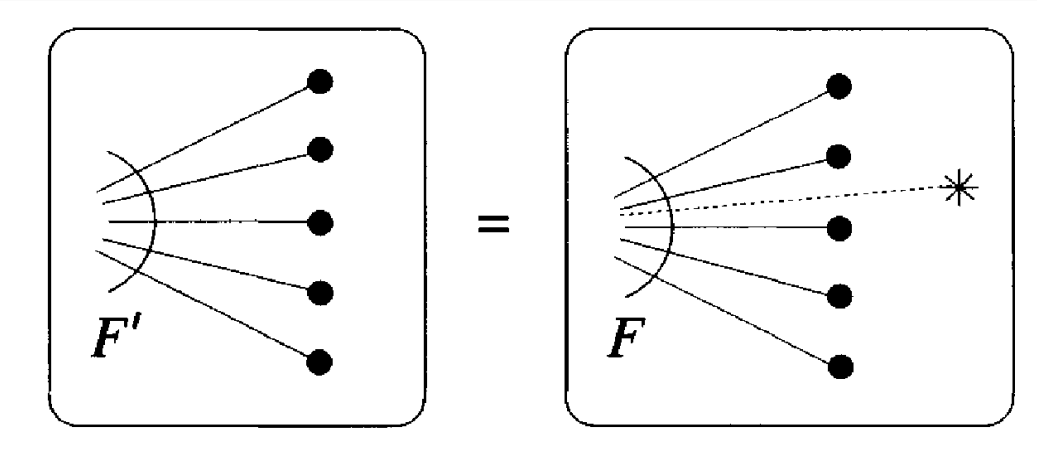
Operations on generating functions, composition

- Let $E_+ = E - 1$, so the generating function of non-empty sets.
- What is a partition? Well it's a way to split a set into non–empty sets, so a set of non-empty sets. So if P is the generating function of Partitions we immediately get $P(x) = E(E_+(x))$. Thus $P(x)$ has the exponential generating function $e^{e^x - 1}$.



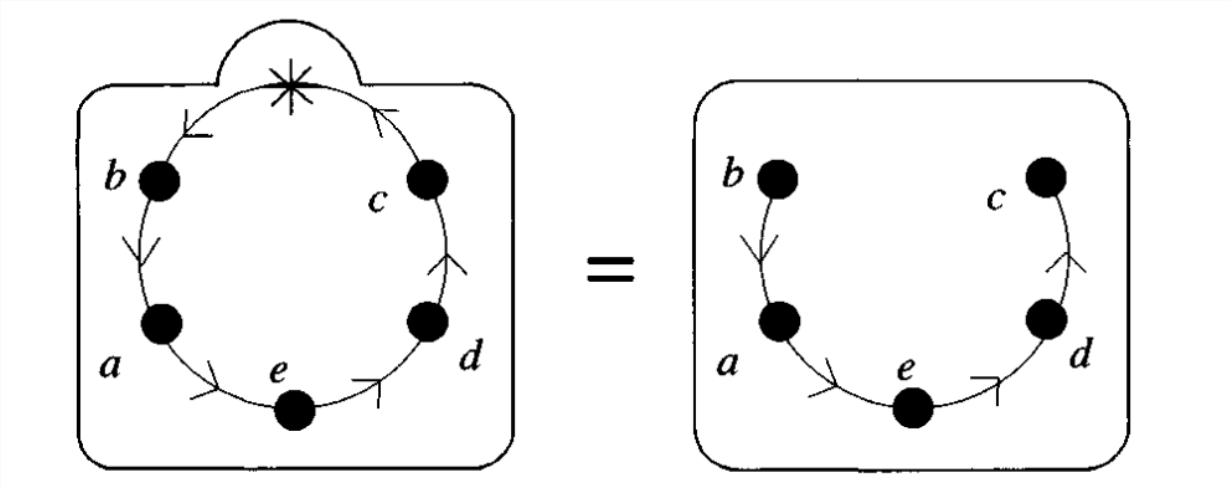
Operations on generating functions, derivation

- The last operation we will look at is a derivative
- For a generating function F, the generating function F' will represent an F structure with one extra special elemenet $*$
- This sometimes allows us to relate different structures in a convenient way



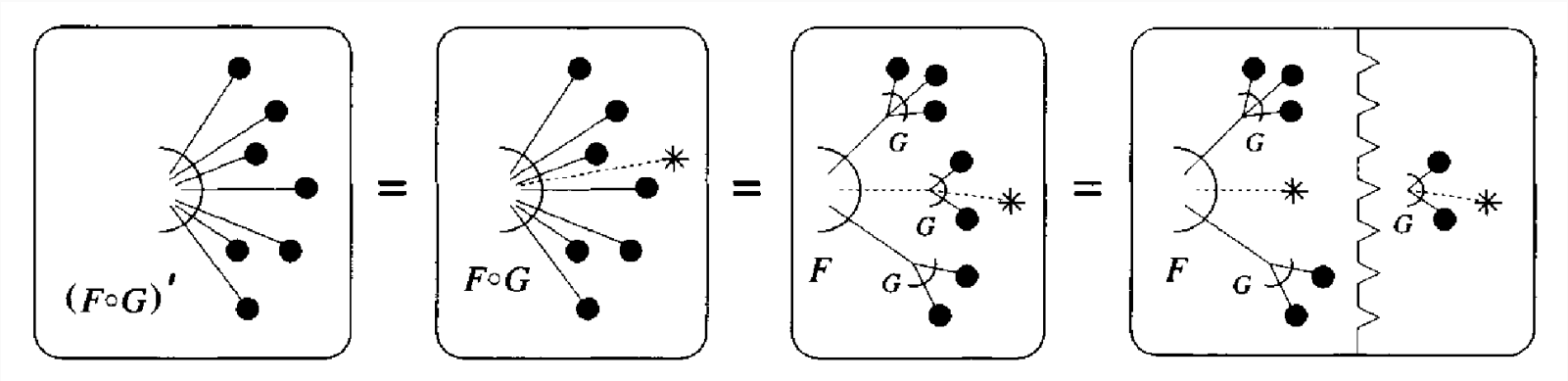
Operations on generating functions, derivation

- For example there is one way to arrange n elements into a row if they are all the same, so the linear order generating function is $L(x) = \frac{1}{1-x}$
- Then we consider the generating function of cycles C, what happens when we consider C'?
- Well then the rest of the elements make a linear order since $*$ breaks up the cycle, so $C' = L$, so $C(x) = \log \frac{1}{1-x}$



Operations on generating functions, derivation

- We can even consider the chain rule $(F \circ G)' = (F' \circ G) \cdot G'$



Note

- This material is tricky to be sure!
- There are some homework problems about state transfer matrices on Kattis, and some about generating functions to turn in as a PDF on Canvas
- Please, do make use of the fact that we are here for hours and hours per day and are willing and able to help you with the assignments
- They are designed with the fact that you have access to us in mind