CS207

Recurrence and Countability

By: Harsh Shah

October 2021

Contents

1 Recursive definitions

1.1 C(n,k)

It can be proved using combinatorial arguments,

$$C(n,k) = C(n-1,k-1) + C(n-1,k)$$

The above expression can be used as recursive definition of $C(n,k)$ with base cases,

$$C(n,0) = C(n,n) = 1 \quad \forall n \in N$$

1.2 Tower of Hanoi

Game: Given three pegs(say A,B and C) and n discs of sizes in succession placed in peg A with largest disc at bottom. Transfer the discs in same order to another peg without ever placing a disc over a smaller one.
Recursive algorithm: If algorithm for $n-1$ discs is known, transfer $n-1$ discs to peg B(keeping the largest disc at peg A), transfer the largest disc to peg C and then transfer $n-1$ discs in peg B to peg C.
Number of steps($M(n)$): $M(n) = 2M(n-1)+1$ and $M(1) = 1$

1.3 Catalan numbers

Catalan number $C(n)$ is defined as number of paths of reaching point (n,n) from origin with only horizontal and vertical integer steps, without encountering any point (i,j) such that $i < j$.
Recursive expression:

$$C(n) = \sum_{k=1}^{n} C(k-1)C(n-k)$$

The above expression can be derived by considering k as the smallest Z^+ such that point (k,k) is reached in the path from origin to (n,n). Then The number of such paths is $C(k-1)C(n-k)$, ($k-1$ because no path touches the diagonal before (k,k) and hence a smaller grid can be considered).

1.4 Fibonacci numbers

Recursive definition,

$$f(n) = f(n-1) + f(n-2) \quad \forall n \geq 2$$

Base case: $f(0) = 0$ and $f(1) = 1$

1.5 Ternary string counting

Find number of strings of length n(say, $M(n)$) using $\{0,1,2\}$ such that there is no substring '00' in the main string.
Recursive relation,

$$M(n) = 2M(n-1) + 2M(n-2)$$

Explanation: Consider a string of length n. Three possibilities: Ends with 1, then the remaining part gives $M(n-1)$ strings.Ends with 2, then the remaining part gives $M(n-1)$ strings. Ends with 0, then second last digit cannot be 0, hence remaining part gives $2M(n-2)$ strings.

1.6 Closed form expression using characteristic equation

Suppose, $f(n) = af(n-1) + bf(n-2) \quad \forall n \geq 2$ and $f(0) = c$, $f(1) = d$.
Then($\alpha \neq \beta$),

$$f(n) = p\alpha^n + q\beta^n$$

where α and β are roots of **characteristic equation** of the recursive relation:

$$x^2 - ax - b = 0$$

In case the roots of the above equation are equal,

$$f(n) = (p + nq)x^n$$

In any of the above cases p and q are found by given base cases.

2 Unrolling recursion using trees

A recursive relation can be represented(and even solved) using m-nary trees(trees with maximum of m children).

2.1 Tower of Hanoi

We found the recursive relation of number of moves as,

$$M(n) = 2M(n-1) + 1$$

The tree(which is binary here) can be constructed by,

- bottom-up approach: Start from base cases and construct the tree upwards to till $M(n)$ is reached
- top-down approach: Start from $M(n)$, break it into sub-trees to reach the base cases

3 Generating functions

Definition: Given $f : Z^+ \cup \{0\} \rightarrow R$,

$$G_f(X) = \sum_{i \geq 0} f(i)x^i$$

Why is it useful? It can sometimes be represented in closed form making it easier to derive closed forms of recursively defined functions.

3.1 Extended binomial theorem

For $|x| < 1$ and $a \in R$,

$$(1+x)^a = \sum_{k=0}^{\infty} C(a,k)x^k$$

where,

$$C(a,k) = \frac{a(a-1)(a-2)\ldots(a-k+1)}{k!}$$

Eg, $C(-1,k) = 1$ and $C(-2,k) = k+1$

3.2 Getting generating function

Consider Fibonacci function,

$$f(n) = f(n-1) + f(n-2) \quad \forall n \geq 2$$

Base case: $f(0) = 0$ and $f(1) = 1$
Now, multiply both sides of the equation with X^n and over all $n \geq 2$

$$G_f(X) - f(0) - f(1)X = X(G_f(X) - f(0)) + X^2 G_f(X)$$

Further simplification and substitution of base values yields

$$G_f(X) = \frac{X}{1 - X - X^2}$$

In general, if

$$f(n) = af(n-1) + bf(n-2) \quad \forall n \geq 2$$

Base case: $f(0) = c$ and $f(1) = d$, then

$$G_f(X) = \frac{c + (d - ac)X}{1 - aX - bX^2}$$

Now, if we want to get generating function of $g(k) = \sum_{i=0}^{k} f(i)$
Use recursive definition of $g(k)$ as,

$$g(k) = g(k-1) + f(k)$$

Base case: $g(0) = f(0)$
Again multiplying by X^k and summing up, we get

$$G_g(X) = \frac{G_f(X)}{1 - X}$$

3.3 Applications of generating functions

Example: Find $G_g(X)$ where $g(k) = \sum_{i=0}^{k}(i+1)^2$
Firstly we require a closed form expression of $G_f(X)$ where $f(k) = (k+1)^2$.
Consider the expression

$$\frac{1}{1-x} = \sum_{i=0}^{\infty} x^i$$

Now, differentiate the above equation, then multiply by X, then again differentiate to get G_g as

$$G_g(X) = \frac{1 + X}{(1 - x)^3}$$

Then use $G_g(X) = \frac{G_f(X)}{1-X} = \frac{1+X}{(1-X)^4}$

3.4 Unlabelled balls into labelled bins

Let $f(n)$ = number of ways placing n unlabelled balls into d labelled bins
Generating function for f,

$$G_f(X) = (1 + X + X^2 + X^3 \ldots)^d = \frac{1}{(1 - X)^d}$$

Extended binomial theorem can be used to get $f(n)$

3.5 Proof for closed form expression

The generating function can be used to the closed form expression of $f(n)$ described earlier($f(n) = p\alpha^n + q\beta^n$ or $f(n) = (p + nq)x^n$) using

$$G_f(X) = \frac{c + (d - ac)X}{1 - aX - bX^2}$$

Split the RHS using partial fractions and use extended binomial theorem to reach the result.

3.6 Closed form for Catalan numbers

Recursive definition

$$C(n) = \sum_{k=1}^{n} C(k-1)C(n-k)$$

Consider the below manipulations

$$C(n)X^n = X^n \sum_{k=1}^{n} C(k-1)C(n-k) = X \sum_{k=1}^{n} [C(k-1)X^{k-1}][C(n-k)X^{n-k}]$$

Therefore,

$$\text{coefficient of } X^n \text{ in } G_C(X) = \text{coefficient of } X^n \text{ in } [\sum_{k=1}^{\infty} C(k-1)X^{k-1}][\sum_{k=n}^{-\infty} C(n-k)X^{n-k}]$$

The above equation holds for all $n \geq 1$. Therefore including the term $C(0)X^0 = 1$ we get,

$$G_C(X) = 1 + XG_C(X)G_C(X)$$

Now, solve quadratic in $G_C(X)$ use the base case to eliminate the larger root and use extended binomial theorem on $C(1/2, k)$ to get

$$C(n) = \frac{C(2k, k)}{k + 1}$$

The $C(.,.)$ on RHS is the combinatorics function.

4 Asymptotic analysis

4.1 Big O notation

$O(f(n))$ denotes set of all functions $g(n)$ such that

$$\exists c, n' > 0 \text{ such that } \forall n > n' \quad 0 \leq g(n) \leq f(n)$$

Often $g(n) \in O(f(n))$ is wrongly represented as $g(n) = O(f(n))$.
Important examples:

- $|log(n!) - log(n^n)| = O(n)$
- $|log(n!) - log((n/e)^n)| = O(log(n))$
- $|log(n!) - log((n/e)^n) + 0.5log(n)| = O(1)$

The above upper bounds can be proved using **Stirling's** approximation for factorial.

Some properties:

- If $f(n) = O(T(n))$ and $g(n) = O(T(n))$, then $f(n) + g(n) = O(T(n))$
- If $f(n) = O(g(n))$ and $g(n) = O(T(n))$, then $f(n) = O(T(n))$

4.2 Master Theorem

Given: $T(n) = aT(n/b) + c \cdot n^d$, where $a \geq 1, b > 1, c > 0, d \geq 0$(For simplicity assume $n = b^k$)
If

- $a = b^d$, then $T(n) = O(n^d log(n))$
- $a > b^d$, then $T(n) = O(n^{log_b(a)})$
- $a < b^d$, then $T(n) = O(n^d)$

Can be proved by unrolling recursion using trees or algebraically.

4.3 Θ notation

Definition(again $\Theta(.)$ represents a set but = notation is wrongly used):
If $O(f(n)) = g(n)$ and $O(g(n)) = f(n)$, then $f(n) = \Theta(g(n))$
Property:If $f(n) = \Theta(T(n))$ and $g(n) = \Theta(T(n))$, then $f(n) + g(n) = \Theta(T(n))$

4.4 Asymptotically equal($\simeq$)

$f(n) \simeq g(n)$ iff, $Lim_{n\to\infty}(\frac{f(n)}{g(n)}) = 1$
If $\exists c > 0$ s.t. $f(n) \simeq c \cdot g(n)$, then $f(n) = \Theta(g(n))$
Note that inverse of above statement does not hold, $\Theta(.)$ does not require limits to exist.

4.5 Asymptotically much smaller ($<<$)

$f(n) << g(n)$ iff, $Lim_{n\to\infty}\frac{f(n)}{g(n)} = 0$

5 Countability

Given two sets A and B, $|A| = |B|$, iff there exists a bijection from A to B.
A set S is **countably infinite** iff $|S| = |N|$
A set S is said to be **countable** iff it is finite or countably infinite.
Important results:

- N^2 is countable: Make diagonal slashes **or**
  One-One function from $N \to N^2 : f(k) = (k, 1)$
  One-One function from $N^2 \to N : g(a, b) = 2^a 3^b$ (using unique factorization)
  Then use **CSB** theorem.
- Z^2 is countable: Bijective from $Z \to N$ exists, hence from $Z^2 \to N^2$. Also we proved there exists a bijection from $N^2 \to N$.

If A and B are countable sets, then $A \times B$ is also countable.
Now, define
$|A| \leq |B|$ iff there exists one-to-one function from A to B.

5.1 CSB theorem

There exists a bijective function from A to B iff there exists a one-to-one function from A to B and from B to A.
Proof: Forward implication is easy. For reverse implication,
Given two injective functions, $f : A \to B$ and $g : B \to A$
Consider a bipartite graph $G = (A, B, f \cup g)$
The chains can be partitioned(atmost) into 3 types:

1. starting from A
2. starting from B
3. Cyclic or doubly infinite

We aim to create a **perfect matching** from A to B and hence a bijection $h(.)$
For A nodes in chains of type 1, let $h(a) = f(a)$
For rest of the A nodes, let $h(a) = g^{-1}(a)$
It is required to use $f(.)$ for type 1 chains, else the starting node will be left out.

5.2 Examples

- Alphabet strings of any finite size are countably infinite
- Set of all infinitely long binary strings(say T) has same cardinality as R (Map a binary string $[a_1, a_2 \dots]$ to decimal representation $0.a_1 a_2 \dots$, and map any real number to [0,1] using appropriate function and then represent the number in binary)
- $|R^2| = |R|$: A bijective mapping from T to T^2 will be sufficient to prove the claim. This can be achieved by forming a binary string from two given binary strings by alternatively choosing a bit from each string.

Definition: A set S is **uncountable** if it is infinite but not countably infinite.

5.3 Cantor's diagonal slash

claim: R is uncountable
Proof: We know, $|R| = |T| = |P(N)|$ ($P(.)$ is power set)
We now prove that $|P(N)| \neq |N|$ Consider a table with binary entries such that $T_{ij} = 1$ iff $j \in f(i)$.
Construct a set $S = \{j|T_{jj} = 0\}$(that is all the diagonal elements). This set cannot belong to any of $f(i)$ (because the i^{th} bit in $f(i)$ is toggled)
In a similar way(not using table representation, though), it can be proved that
There exists no onto function $f : A \rightarrow P(A)$ for any set A.