

Enumerative Combinatoric Algorithms

Contents

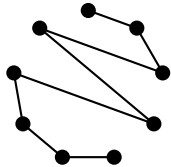
1	Enumerating vs. Counting	2
1.1	Example 1: Spanning Paths	2
1.2	Example 2: Blokus	3
1.2.1	Fingerprints	5
1.3	Example Spanning Trees on Ladders	6
2	Inclusion – Exclusion Principle	7
2.1	Example: Multipliers	7
2.2	Example: Relatively Prime Numbers	7
2.3	Example: Number of Words	8
3	The Pigeon Hole Principle	9
3.1	Example: People in Vienna	9
3.2	Example: Division of odd numbers	9
3.3	Example: Lossless Data Compression	10
4	Reverse Search	10
4.1	Example: Triangulations	10
4.2	Basic Idea	11
4.3	Example: Poker Chips Stacking	11
4.4	Example: Tic Tac Toe	12
4.5	Example: Connect Four (4 Gewinnt)	12
4.6	Algorithm	12
4.7	Example: Triangulations	12

1 Enumerating vs. Counting

1.1 Example 1: Spanning Paths

Given: n points in convex position

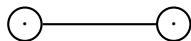
Question: How many straight line planar (crossing-free) spanning paths (all points are used) are there?



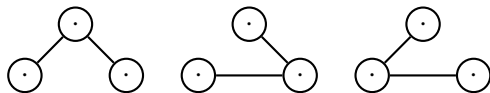
- $n = 1$



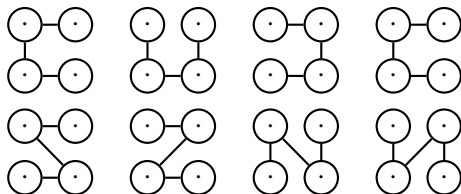
- $n = 2$



- $n = 3$: 4 paths



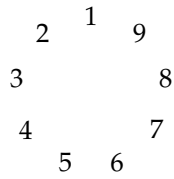
- $n = 4$: 8 paths



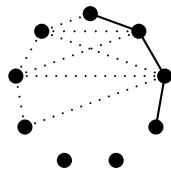
- $n = 5$:

- more possibilities \rightarrow we go towards counting rather than enumerating
- look at possible types of paths, in CW and CCW direction for each of the five starting points. Divide by 2 for identical paths obtained with different starting points (since paths have 2 ends)

Counting



- # of paths starting at $\circ = \#P_\circ$
- $\#P_n = \frac{n \cdot \#p_\circ}{2} = n \cdot \frac{2^{n-1}}{2} = n \cdot 2^{n-2}$



- set of not yet used points must not be split
- continue: always two options, 1 for the last

1.2 Example 2: Blokus

Definition

- consists of unit squares connected via edges (not just connected through vertices)
- n -polyomino has n unit squares
- Animal: does not contain holes
- free: rotation/mirroring is not taken into account
- fixed: rotated/mirrored polyominoes are regarded as being distinct

Question: How many n -polyominoes are there?

- $n = 1$:



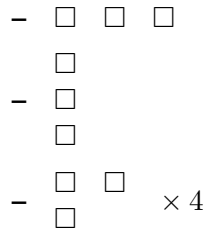
1

- $n = 2$:



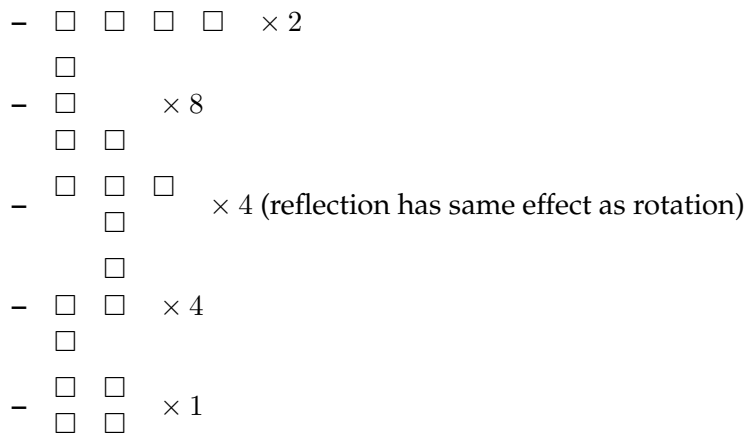
fixed: 2, free: 1

- $n = 3$:



fixed: 6, free: 2

- $n = 4$:



fixed: 19, free: 5

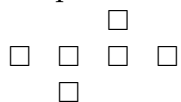
Enumerate/Construct all polyominoes

- construct the $(n + 1)$ -ominoes from the n -ominoes. This works because we can show that from each $(n + 1)$ -ominoes you can remove a square and get a valid n -omino (idea: construct an arbitrary spanning tree, remove leaf)
- find and remove duplicates
- how many $(n + 1)$ -ominoes are generated?

maximum number: $2n + 2$ or $4n - 2|E|$

Why?

\square : 4 possibilities to add a square (at each edge)



$$4n - 2|E| \rightarrow 4n - 2(n - 1) = 2(n + 2)$$

How to find duplicates? compare every pair $\rightarrow O(k^2n)$

1.2.1 Fingerprints

- maps an object (here: polyomino) to a *unique* representation, independent of translation, rotation, reflection
- sort the fingerprints (insertion sort)
- binary search

One possibility for polyominoes: use vector representations on a coordinate system

Examples:

- $\begin{array}{c} 3 \\ 2 \\ 1 \end{array} \begin{array}{ccc} & & \square \\ & & \square \\ \square & \square & \square \end{array}$ has vector representation $\langle (1, 1), (2, 1), (3, 1), (3, 2), (3, 3) \rangle$
 $\begin{array}{ccc} & & \\ & & \\ 1 & 2 & 3 \end{array}$
- $\begin{array}{c} 3 \\ 2 \\ 1 \end{array} \begin{array}{ccc} \square & & \\ \square & & \\ \square & \square & \square \end{array}$ has vector representation $\langle (1, 1), (2, 1), (3, 1), (1, 2), (1, 3) \rangle$
 $\begin{array}{ccc} & & \\ & & \\ 1 & 2 & 3 \end{array}$

This is lexicographically smaller than the first one

- $\begin{array}{c} 3 \\ 2 \\ 1 \end{array} \begin{array}{ccc} \square & & \\ \square & \square & \\ \square & \square & \square \end{array}$ $\begin{array}{c} 3 \\ 2 \\ 1 \end{array} \begin{array}{ccc} & \square & \square \\ \square & \square & \\ \square & & \end{array}$ $\langle (1, 1), (1, 2), (2, 2), (2, 3), (3, 3) \rangle$
 $\begin{array}{ccc} & & \\ & & \\ 1 & 2 & 3 \end{array}$ $\begin{array}{ccc} & & \\ & & \\ 1 & 2 & 3 \end{array}$
- $\begin{array}{c} 3 \\ 2 \\ 1 \end{array} \begin{array}{ccc} \square & \square & \square \\ \square & & \square \\ \square & & \square \end{array}$ $\begin{array}{c} 3 \\ 2 \\ 1 \end{array} \begin{array}{ccc} \square & \square & \\ \square & & \\ \square & \square & \square \end{array}$ $\langle (1, 1), (3, 1), (1, 2), (1, 3), (2, 3) \rangle$
 $\begin{array}{ccc} & & \\ & & \\ 1 & 2 & 3 \end{array}$ $\begin{array}{ccc} & & \\ & & \\ 1 & 2 & 3 \end{array}$

Generally:

1. generate all 8 transformations (4 rotations, 2 reflections)
2. take the lexicographically smallest one $\rightarrow O(1) \cdot O(n)$ for an n -omino

$f'(n)$ fingerprints generated in total $\rightarrow f'(n) (O(n) + \log(f'(n) \cdot O(n)))$

$$\lim_{n \rightarrow \infty} \frac{f(n+1)}{f(n)} = \lambda, 3.98 < \lambda < 4.65$$

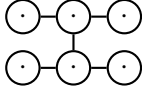
$f(n)$ is known up to $n = 56$.

1.3 Example Spanning Trees on Ladders

with n rungs

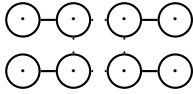
Question: Number of spanning trees on a ladder with n rungs

- $n = 1$



= 1

- $n = 2$



= 4

- $n = 3$



= 15

Answer: Counting spanning trees on a ladder (with n rungs) \Leftrightarrow counting spanning trees on a $2n$ grid

$n \rightarrow n + 1$

1. The 2 rightmost vertices for n share a connected component $\rightarrow A(n)$
2. The 2 rightmost vertices for n are in different connected components $\rightarrow B(n)$

Consider case 1

$$A(n+1) = 3A(n) + 1B(n)$$

Consider case 2

$$A(1) = \frac{1}{2\sqrt{3}} \left((2 + \sqrt{3})^1 - (2 - \sqrt{3})^1 \right) = \frac{2\sqrt{3}}{2\sqrt{3}}$$

$$A(2) = \dots = \frac{8\sqrt{3}}{2\sqrt{3}} = 4$$

$$\begin{aligned} A(n) &= 4A(n-1) - A(n-2) \\ &= \frac{4}{2\sqrt{3}} \left((2 + \sqrt{3})^{n-1} - (2 - \sqrt{3})^{n-1} \right) - \frac{1}{2\sqrt{3}} \left((2 + \sqrt{3})^{n-2} - (2 - \sqrt{3})^{n-2} \right) \end{aligned}$$

2 Inclusion – Exclusion Principle

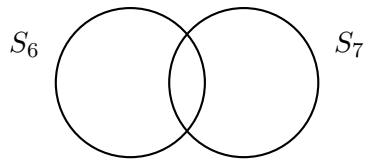
2.1 Example: Multipliers

$$S = \{1, \dots, 100\}$$

Question: How many numbers in S are multipliers of 6 or 7?

$$|S_6| = \lfloor \frac{100}{6} \rfloor = 16, |S_7| = 14$$

$$|S_{6,7}| \neq |S_6| + |S_7|:$$



$$|S_{6,7}| = |S_6| + |S_7| - |S_{6 \& 7}| = 16 + 14 - 2 = 28$$

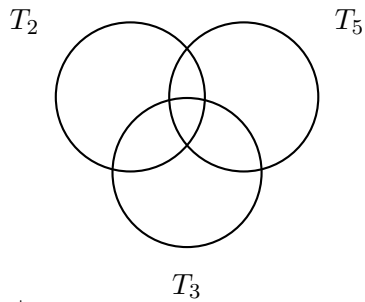
2.2 Example: Relatively Prime Numbers

$$S = \{1, \dots, 180\}$$

Question: How many numbers in S are relatively prime to 180?

Idea: Count all numbers *not* relatively prime to 180 $\rightarrow T$.

$$|T| = ?$$



$$|T_2| = 90$$

$$|T_3| = 60$$

$$|T_5| = 36$$

$$|T| = |T_2| + |T_3| + |T_5| - (|T_2 \cap T_3| + |T_2 \cap T_5| + |T_3 \cap T_5|) + |T_2 \cap T_3 \cap T_5|$$

$$|T| = 90 + 60 + 36 - (30 + 18 + 12) + 6 = 132$$

$$|S| - |T| = 48$$

In general:

$$\mathcal{A} = \{A_i\}_{i=1}^n, I = \{1, \dots, n\}$$

$$|_{i \in I} A_i| = \sum_{i=1}^n \left((-1)^{i-1} \sum_{J \subseteq I, |J|=i} \left| \bigcap_{j \in J} A_j \right| \right)$$

Proof: $x \in A_1 \cup \dots \cup A_n$, x is contained in m sets A_i .

$$|A_1 \cup \dots \cup A_n| = \sum (\cdot) - \sum (\cdot \cap \cdot) + \sum (\cdot \cap \cdot \cap \cdot) - \sum (\cdot \cap \cdot \cap \cdot \cap \cdot) \dots$$

1. for (single) sets *not* containing $x \rightarrow +0$
 2. if in any $A_i \cap \dots \cap A_j$ at least one of the sets does *not* contain $x \rightarrow \pm 0$
 3. \rightarrow only consider intersections of sets where all sets contain x
- $\binom{m}{k}$ intersections of k sets, all sets containing $x \rightarrow \pm 1$

$$\Rightarrow + \binom{m}{1} - \binom{m}{2} + \dots + (-1)^{m-1} \binom{m}{m} = 1$$

$$\binom{m}{0} + \binom{m}{1} - \binom{m}{2} + \dots + (-1)^m \binom{m}{m} = 0$$

Pascal's triangle



2.3 Example: Number of Words

Given: Language L with alphabet $\Sigma = \{A, B, C\}$ and every word of L consists of 10 characters.

Question: How many words of L contain each character at least once?

$$|G| = |L| - |X| = 3^{10} - (3 \cdot 2^{10} - 3 + 0) = 55980$$

$$|S_A| = |S_B| = |S_C| = 2^{10}$$

$$|S_A \cap S_B| = |S_A \cap S_C| = |S_B \cap S_C| = 1$$

$$|S_A \cap S_B \cap S_C| = 0$$

$$|X| = |S_A| + |S_B| + |S_C| - |S_A \cap S_B| - |S_A \cap S_C| - |S_B \cap S_C| + |S_A \cap S_B \cap S_C|$$

3 The Pigeon Hole Principle

If n elements are distributed to k sets, then there exists at least one set containing at least $\lceil \frac{n}{k} \rceil$ elements.

3.1 Example: People in Vienna

Claim: In Vienna there exist at least two persons with the exact same number of hairs on their head.

- human: at most $\approx 500,000$ hairs
- Vienna: $\geq 1.7 \cdot 10^6$ citizens

Idea: Take 500,000 people. If 2 of them have the same amount of hair, then we are done. If not, add one more person, whose amount of hair must then already be present.

3.2 Example: Division of odd numbers

Given: $1, 3, 7, 15, 31, \dots; a_i := 2^i - 1, i \geq 1$ and $q > 0$ arbitrary odd number.

Claim: At least one of the a_i 's is (integer) divisible by q .

Proof: Consider all $a_i, 1 \leq i \leq q$

- If one of these a_i is divisible by q , then we are done
- otherwise: $a_i = d_i \cdot q + r_i, 0 < r_i < q; q \geq n > m > 0$

$\Rightarrow (q-1)$ remainders $\xrightarrow{\text{p.h.p.}} \exists r_m, r_n$ such that $r_m = r_n$

Then $(a_n - a_m) = (d_n - d_m)q + \underbrace{(r_n - r_m)}_0$

$$(2^n - 1) - (2^m - 1) = 2^n - 2^m = 2^m \underbrace{(2^{n-m} - 1)}_{=a_{n-m}}.$$

2^m is even and thus not divisible by q .

In general: if A_1, \dots, A_k are finite (pairwise disjoint) sets and $|A_1 \cup \dots \cup A_k| > kr$, then $\exists i$ such that $|A_i| > r$.

Proof: Assume for the sake of contradiction that all $|A_i| \leq r$

$$k \cdot r \geq \sum_{i=1}^k |A_i| \geq |A_1 \cup \dots \cup A_k| > k \cdot r$$

3.3 Example: Lossless Data Compression

... cannot guarantee compression for all input data!

- file \rightarrow string of bits
- assume that each file is transformed into a *distinct* file that is not larger
- let F be the file with the least number of bits (M) that compresses $\rightarrow N$ bits
- $N < M : 2^N + 1$ – cannot be all distinct

4 Reverse Search

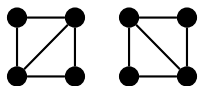
- Avis, Fukuda 1992
- enumerating data/objects/elements with almost no structure
- algorithmic
- count all objects *exactly* once, no overcounting

4.1 Example: Triangulations

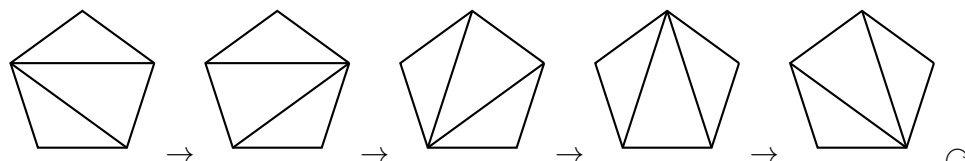
Given: set S of labelled points in convex position

Question: # of triangulations on S .

- $n = 4$



- $n = 5$



Flip on triangulations: 4 points, remove the diagonal and add the other diagonal

4.2 Basic Idea

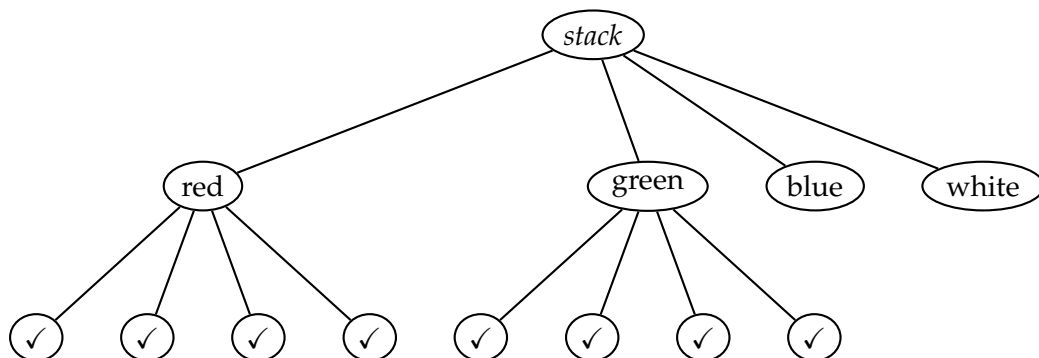
- abstract graph $G(V, E)$
- vertices V : elements to be enumerated, $|V|$ is very large
- edges $e \in E : e = v_1, v_2; v_1, v_2 \in V$
- $\Rightarrow G$ is an undirected graph
- G must be connected
- task: enumerate all vertices of G
- idea: breadth-first or depth-first search: $O(|V| + |E|)$ which is ok, but too much space necessary

4.3 Example: Poker Chips Stacking

Given: four colours (red, green, blue, white) of poker chips

Task: build stacks of height ≤ 30 such that at most 3 chips of equal colour appear directly after another

- v_\emptyset : empty stack
- neighbours:
 - 'successors': add a chip on top of the stack
 - 'predecessors': remove the topmost chip from the stack
- depth-first search



- store current stack.
Nodes that have already been visited can be deduced from the stack composition.

Requirements:

- unique root
- successors and predecessors in arbitrary but fixed order
neighbour relation $\gamma(v, k) \in \Gamma(v)$ (k th neighbour)
- a unique predecessor function $f(v), f : v \rightarrow v$ in $G(V, E)$
 1. $\exists v_0 \in V : f(v_0) = v_0$: root
 2. $\forall v \in V \setminus \{v_0\} : f(v) \in \Gamma(v)$: exactly one root
 3. $\forall v \in V \setminus \{v_0\} \forall x \in \mathbb{N} : f^x(v) \neq v$: cycle-free ($f^x = f(f \dots f(v) \dots)$) x times
 \rightarrow reaching the root in finite time $f^{|V|}(v) = v_0$

4.4 Example: Tic Tac Toe

Given:

Task: Enumerate all valid game positions (not considering symmetries)

- V : game positions
- labelled board cells
- neighbours $\gamma(v, k)$: legally add k -th mark or remove one
- predecessor function: empty board as root, otherwise remove highest legal mark
- when iterating over the possible positions, check each node's predecessor to see if it has been visited before

4.5 Example: Connect Four (4 Gewinnt)

⋮	⋮	⋮	⋮	⋮	⋮
			X		
			O		

too complex for reverse search, NP-hard

4.6 Algorithm

Complexity Analysis

4.7 Example: Triangulations