HW2 PB6

part A, Satisfiability Intro [easy]. A boolean formula is satisfiable if there exists some variable assignment that makes the formula evaluate to true. Namely, a boolean formula is satisfiable if there is some row of the truth table that comes out true. Determining whether an arbitrary boolean formula is satisfiable is called the *Satisfiability Problem.* There is no known efficient solution to this problem, in fact, an efficient solution would earn you a million dollar prize. While this is hard problem in computer science, not all instances of the problem are hard, in fact, determining satisfiability for some types of boolean formulae is easy.

$(A \Rightarrow B) \equiv B \vee \neg A$

$(\neg B \Rightarrow \neg A) \equiv$

i. First, let's consider why this would be hard. If you knew nothing about a given boolean formula other than that it had n variables, how large is the truth table you would need to construct? Please indicate the number of columns and rows as a function of n

ii. Now consider the following 100 variable formula.

$\neg x_k \vee x_{k+1}$ particular 2CNF

x_1

$$x_1 \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3 \vee x_4) \wedge \ldots \wedge (\neg x_{99} \vee x_{100})$$

$(x_1 \Rightarrow x_2) \wedge (x_2 \Rightarrow x_3) \wedge \ldots \wedge (x_{99} \Rightarrow x_{100})$

Without constructing a truth table, how many satisfying assignments does this formula have, explain your answer.

iii Now consider an arbitrary 3-DNF formula with 100 variables and 200 clauses. 3-DNF means that the formula is in disjunctive normal form and each clause has three literals. (A literal is the instantiation of the variable in the formula, so for x, $\neg x$ or x.) An example might be something like:

$$(\neg x_1 \wedge x_3 \wedge x_{10}) \vee (\neg x_3 \wedge x_{15} \wedge \neg x_{84}) \vee (x_{17} \wedge \neg x_{37} \wedge x_{48}) \vee \ldots \vee (\neg x_{87} \wedge \neg x_{95} \wedge x_{100})$$

What is the largest size truth table needed to solve this problem. What is the maximum number of such truth tables needed to determine satisfiabilty.

HON PB2 : general 2-CNF Formula
find x bool assignment

part B: 2CNF-SAT [hard]. The 2CNF-SAT instance is a boolean CNF formula with 2 variables in each clause, "OR" inside clauses, "AND" between clauses. There are m boolean variables $x_1, x_2, ..., x_m$) and n clauses $C_1, C_2, ..., C_n$). Every variable and its negation appears in at least one clause. Such formula is given as input in format redundantly :
- for each variable there is a list of clauses containing it
- for each clause there there are 2 variables

2CNF general

For example the formula $(x_1 \vee \neg x_2) \wedge (x_2 \vee x_3) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_2 \vee \neg x_3)$ will be given as:

$m = 3, n = 4$
$x_1 : C_1$
$\neg x_1 : C_3$
$x_2 : C_2$
$\neg x_2 : C_1, C_4$
$x_3 : C_2, C_3$
$\neg x_3 : C_4$
$C_1 : x_1, \neg x_2$
$C_2 : x_2, x_2$
$C_3 : \neg x_1, x_3$
$C_4 : \neg x_2, \neg x_3$

$x_2 \Rightarrow x_1$
$\neg x_1 \Rightarrow \neg x_2$

transform each clause into 2 implications

want
emphasize : procedure (recipe)

Your task is to design a strategy that determines, for a given formula, the boolean assignments for the variables such that all clauses are satisfied, thus the formula is true (if more such assignments are possible, you only need to output one). If no such assignment is possible, output "FALSE".

As established inpart A, there are 2^m possible assignments for the variable set. So if one were to build the truth table and "brute force" search all rows/assignments until one works, it would take exponential time — not good! Instead: do trial and error, but in a smart way that only tries at most $2 * m^2$ boolean assignments.

Your strategy can be pseudocode, or you can informally describe a procedure with bullets and English statements. You can write in your procedure statements like
* $x = x_1$

* foreach C containing variable x {
- - - -
}
* C= next clause, or C = next clause containing x

* loop C through all clauses that contain x or $\neg x$

* for each $x \in C$ {
- - - -
}
* y = the other variable in clause C, other than x or $\neg x$

Lecture 9 Advanced Counting

- Binomial Th recap, binomial coef.
- PIE (n sets) proof
- Derangements: permutations with no fixed point
- Balls into bins $\quad \circ\circ|\circ\circ\circ|\circ\circ\circ$ (ex: 8 balls into 3 bins)
- Catalan number $C_n = \binom{2n}{n} - \binom{2n}{n-1}$ is answer to many counting problems.

$\boxed{\text{Binomial theorem}}$ (coef) $\Rightarrow$ Pascal $\triangle$ $x, y \in \mathbb{R}$

$n=2$

$$(x+y)^2 = 1x^2 + 2xy + y^2 = \binom{2}{0}x^2 + \binom{2}{1}xy + \binom{2}{2}y^2$$

$$(x+y)^3 = 1x^3 + 3x^2y + 3xy^2 + 1y^3 = \binom{3}{0}x^3 + \binom{3}{1}x^2y + \binom{3}{2}xy^2 + \binom{3}{3}y^3$$

$$(x+y)^4 = 1x^4 + 4x^3y + 6x^2y^2 + 4xy^3 + 1y^4$$

$$\binom{4}{2} = \frac{4!}{2!\cdot 2!} = \frac{3\cdot 4}{2}$$

$$\binom{4}{0}x^4 + \binom{4}{1}x^3y + \binom{4}{2}x^2y^2 + \binom{4}{3}xy^3 + \binom{4}{4}y^4$$

$$\binom{4}{1}x^{4-1}y^1 \qquad \binom{4}{2}x^{4-2}y^2 \qquad \binom{4}{3}x^{4-3}y^3$$

$(x+y)(x+y)(x+y)(x+y) \rightarrow$ 16 terms (incl repetion)

4 factors ( )

$\binom{4}{3}$

3 x choice

distinct terms

$\binom{4}{2}$ choose 2 x

- x^4 — 1 way : x x x x
- $\boxed{x^3y^1}$ (3 x, 1 y) xxxy xxyx xyxx yxxx 4
- $\boxed{x^2y^2}$ xxyy xyxy xyyx yxxy yyxx yxyx 6
- x^1y^3
- y^4

how many ways to choose out of n

y choice in j paren ( )

x choice in n-j paren ( )

$$(x+y)^n = \sum_{j=0}^{n} \binom{n}{j} \underbrace{x^{n-j}y^j}_{\text{choose } \overset{j}{n-j} \; \overset{\text{"y"}}{\text{"x"}}} = \sum_{j=0}^{n} \binom{n}{j} x^j y^{n-j}$$

2^n terms (with repetitions)

$\binom{n}{k}$ = # subsets of size k

$x=1 \quad y=1$

$$2^n = (1+1)^n = \sum_{j=0}^{n} \binom{n}{j} 1^{n-j} \cdot 1^j = \sum_{j=0}^{n} \binom{n}{j} = \binom{n}{0} + \binom{n}{1} + \dots + \binom{n}{n}$$

subsets $|\,| = 0$

subsets $|\,| = 1$

subset all

$x=+1, \ y=-1$

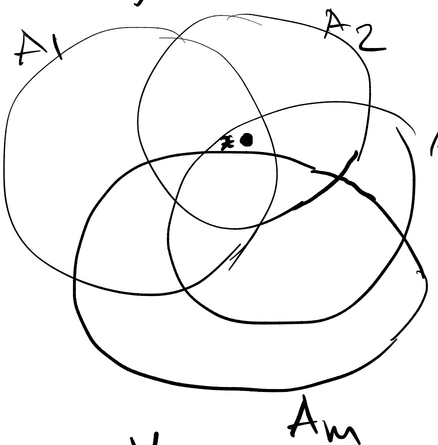
$$0 = (1-1)^n = \sum_{j=0}^{n} \binom{n}{j} \boxed{1^{n-j}} \underline{(-1)^j} = \binom{n}{0} - \binom{n}{1} + \binom{n}{2} - \binom{n}{3} + \dots (-1)^n \binom{n}{n}$$

$n=3 \qquad 1-3+3-1 = 0$

$n=4 \qquad 1-4+6-4+1 = 0$

$n=5 \qquad 1-5+10-10+5-1 = 0$

PIE general proof $|A_1 \cup A_2 \cup A_3 \cup \dots \cup A_m| =$



$= |A_1| + |A_2| \dots + |A_m|$ // |one set|

$- |A_1 \cap A_2| \dots - |A_i \cap A_j| \dots - |A_{m-1} \cap A_m|$ $-|\cap$ of $2|$

$+ |A_1 \cap A_2 \cap A_3| \dots + |A_i \cap A_j \cap A_k| + \dots + |A_{m-2} A_{m-1} A_m|$ $+ |\cap$ of $3|$

$-$

$+$

$\vdots$

$+ (-1)^{m+1} |A_1 \cap A_2 \cap \dots \cap A_m|$ $\dots$ $|\cap$ of $m|$

Select $\forall x$ in $A_1 \cup A_2 \cup \dots A_m$. It's going to part of some sets

without lose of generality assume $x \in A_1 \cap A_2 \dots \cap A_n$ $(n \leq m)$

$x \notin A_{n+1} \cup A_{n+2} \dots A_m$

<u>plan</u> $\boxed{\text{count } x}$ on RHS

$+ \binom{n}{1}$ $|A_1|$ $|A_2|$ $\dots$ $|A_n|$

$- \binom{n}{2}$ $|A_1 \cap A_2|, |A_1 \cap A_3|$ $\dots$ $|A_{n-1} \cap A_n|$

$+ \binom{n}{3}$ $|A_1 \cap A_2 \cap A_3|$ $\dots$ $|A_{n-2} \cap A_{n-1} \cap A_n|$

$\vdots$

$(-1)^{n+1}\binom{n}{n}|A_1 \cap A_2 \cap \dots A_n|$

$count(x) = \binom{n}{1} - \binom{n}{2} + \binom{n}{3} \dots + (-1)^{n+1}\binom{n}{n}$

Binomial Th: $\binom{n}{0} - \binom{n}{1} + \binom{n}{2} \dots + (-1)^n\binom{n}{n} = 0$

$1 - count(x) = \overset{1}{\binom{n}{0}} - \binom{n}{1} + \binom{n}{2} \dots - (-1)^n\binom{n}{n} = 0$

$1 - count(x) = 0 \implies count(x) = 1$ ✓

PIE application: Derangement = permutation without fix points (index sits on its own spot)

$n=5$

2 3 1 5 4   Derang...

pos 1 2 3 4 5

3 $\boxed{2}$ 4 5 1   NOT DER (pos(2)=2)

#Derangement(5) = ?

|all perm − all perm fixed points|

$A_1 = \{$permut (1 fixed)$\}$ $\boxed{1}\ \overbrace{\_\ \_\ \_\ \_}^{4!}$

$A_2 = \{$perm (2 fixed)$\}$ $\_\ 2\ \_\ \_\ \_$

$A_3 = \{$perm (3 fixed)$\}$ $\_\ \_\ 3\ \_\ \_$

$A_4 = \{$perm (4 fixed)$\}$ $\_\ \_\ \_\ 4\ \_$

$A_5 = \{$perm (5 fixed)$\}$ $\_\ \_\ \_\ \_\ 5$

$= n! - \underbrace{|A_1 \cup A_2 \cup A_3 \cup A_4 \cup A_5|}$

$= \underbrace{|A_1|}_{4!} + \underbrace{|A_2|}_{4!} \ldots$

$- \underbrace{|A_1 \cap A_2|}_{3!} \ldots$

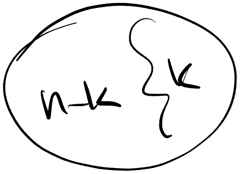
$+ \underbrace{|A_1 \cap A_2 \cap A_3|}_{2!} \ldots$

$A_1 \cap A_2 = \{1\ 2\ \overbrace{\_\ \_\ \_}^{3!}\}$

$A_1 \cap A_2 \cap A_3 = \{1\ 2\ 3\ \overbrace{\_\ \_}^{2!}\}$

exercise: $\binom{n}{k} = \binom{n}{n-k}$ choose subset of k "in"

$\Leftrightarrow$ choose n-k stay out



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

proof $\{1\ 2\ 3 \dots n\}$

$\binom{n}{k}$ = # ways to choose a subset of k out $\{1, 2, 3, \dots n\}$

= SUM RULE

case1 include last elem "n"

$\{ \underbrace{\_\ \_\ \_\ \_\ \_\ \_}_{\text{from } \{1,2,\dots n-1\}}^{k-1}\ n \}$ $\binom{n-1}{k-1}$

case2 dont inclue "n"

$\{ \underbrace{\_\ \_\ \_\ \_\ \_}_{\text{from } \{1,2,\dots n-1\}}^{k \text{ elem}} \}$ $\binom{n-1}{k}$

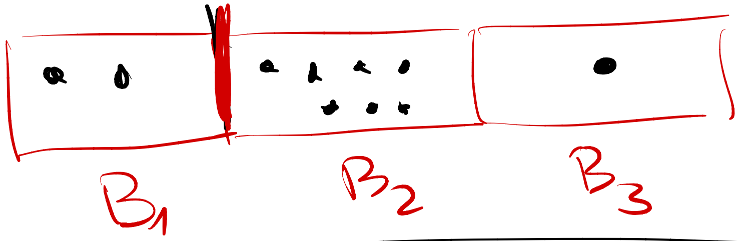
verify (exercise) with factorials.

BALLS INTO BINS (STARS AND BARS)

#ways to place n identical balls in K bins
(not distinguishable) (distinguishable)

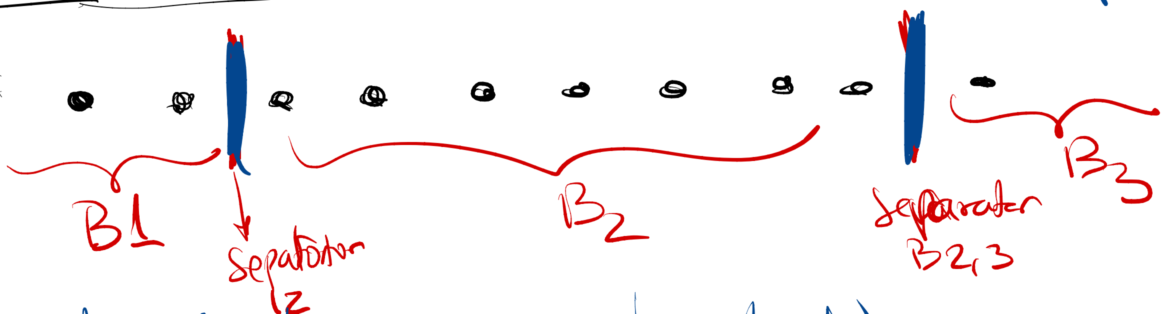
ex: $n=10$ candies (identical) distribute to $k=3$ children $c_1, c_2, \ldots c_k$

Throw balls at random



K-1 separators

Solution:



$n+k-1$ items (balls n, separators $k-1$)

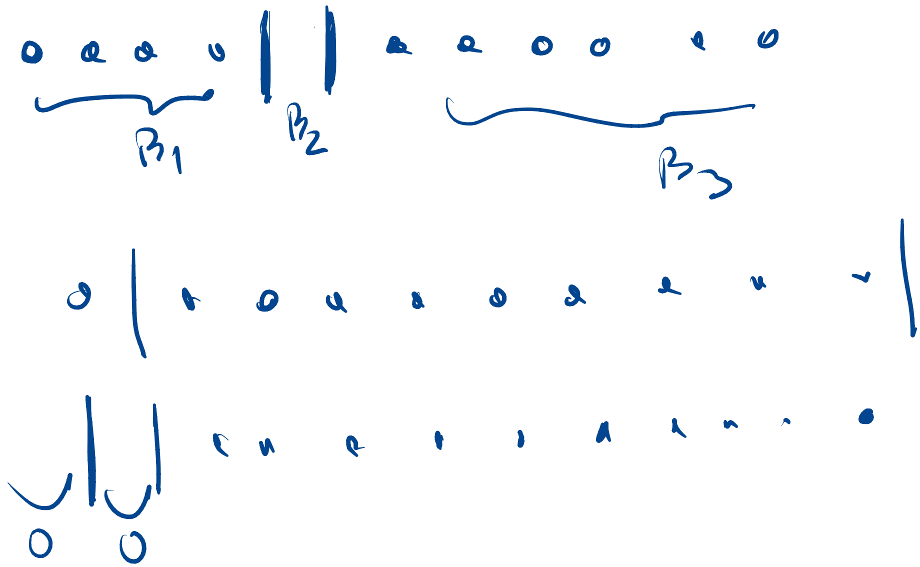
2, 7, 1 • • | • • • • • • • | • • $\Rightarrow$ #ways choose $k-1$ spots "|"

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

4, 0, 6

1, 9, 0

0, 0, 10

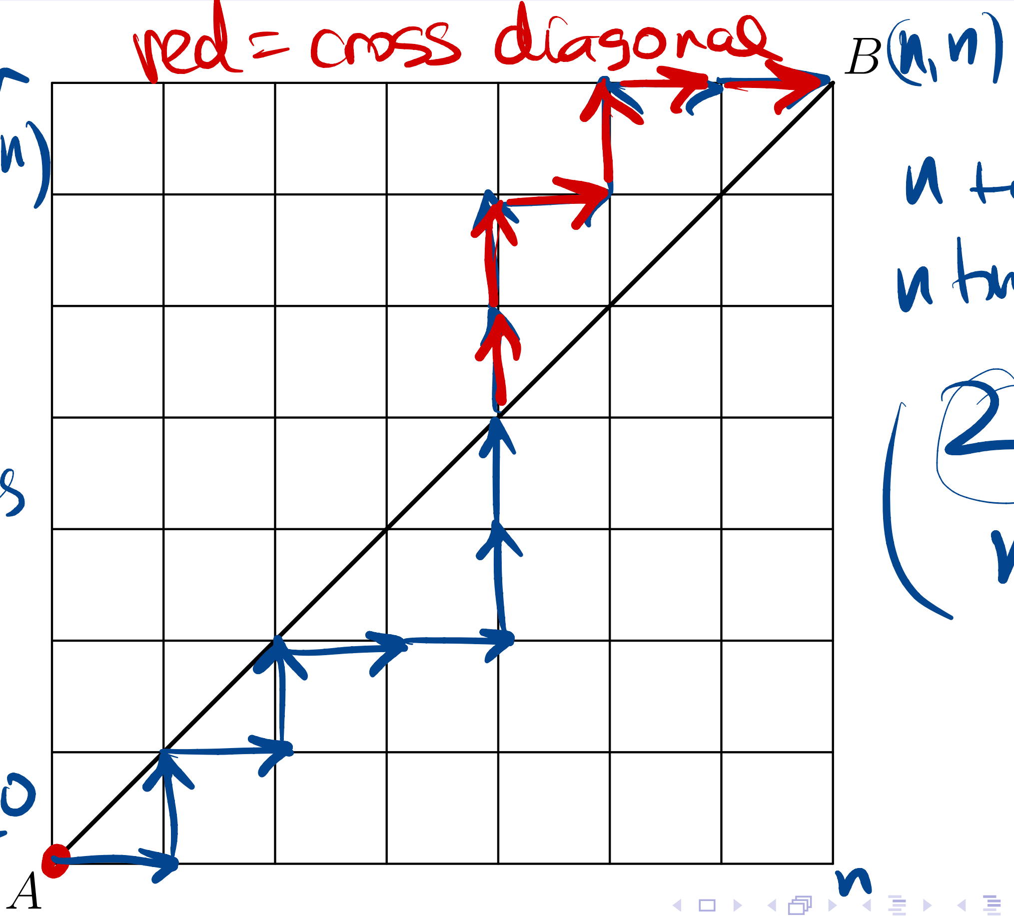


Rules for proper counting

- items are distinguishable / NOT
- REPETITIONS / NOT (REP)
- ORDER / NOT ORDER

How many shortest paths from A to B do *not* pass above the diagonal?

any path $A(0,0) \to B(n,n)$



walk moves "→" or "↑"

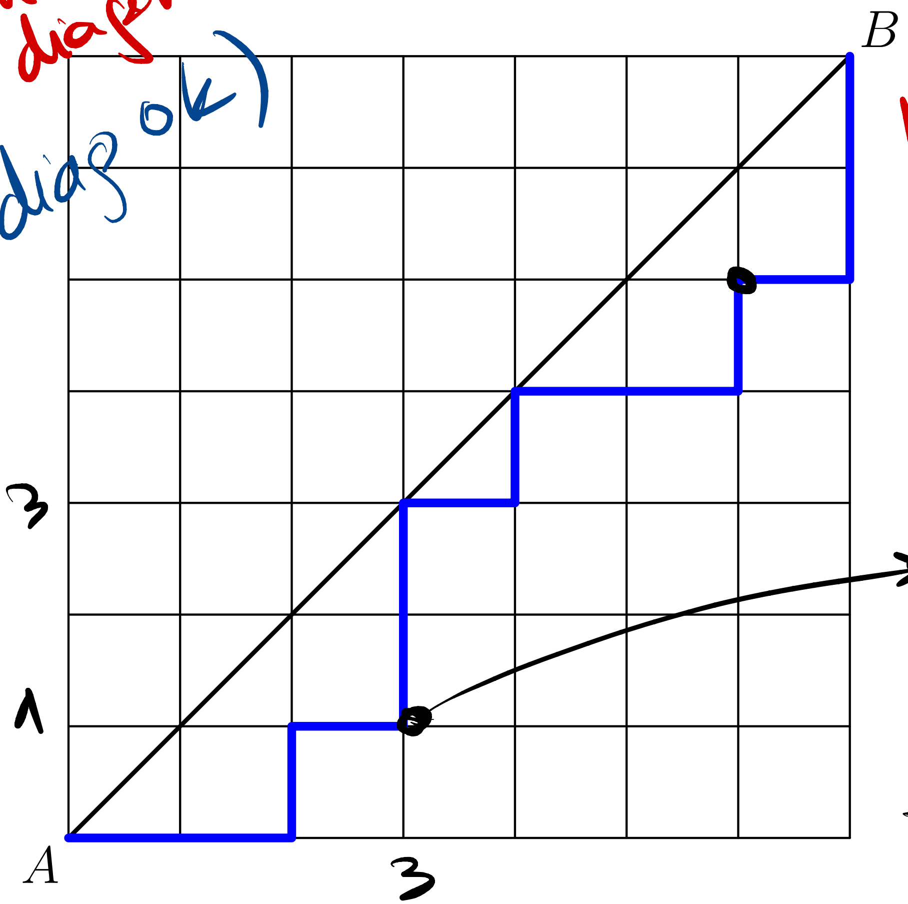
n times "→"
n times "↑"

$\binom{2n}{n}$ total moves

restriction: don't cross diagonal

(touch diag ok)

Catalan number C_n



anywhere # moves → ≥ # move ↑ (so far)

path →, →, ↑, →, ↑, ↑, →, ↑, →, →, ↑, →, ↑, ↑

$n=3$ sets of ( ) nested: ((( ))) ; (( )( )) ; (( ))( ) ; ( )(( )) ; ( )( )( )

#"(" ≥ #")" at any point in sequence

Stacks: push → at the top (LIFO)
pop → from the top

valid sequ. of stack ops : Push, Pop, push, push, pop . . .
( ) (( ) . . .

same property

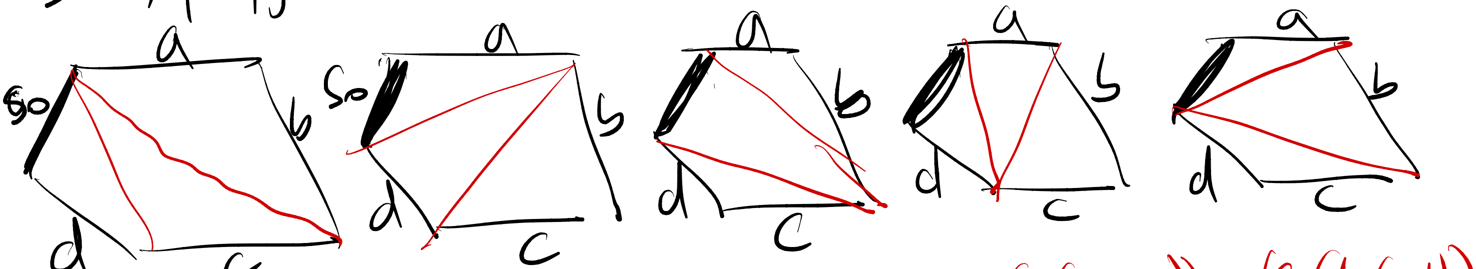
histories = # valid paths under diagonal

multiply a · b · c · d → n+1 items, divide the order

$n=3$

((a·b)·(c·d))	(((a·b)·c)·d)	(a·(b·c))·d	a·((b·c)·d))	(a·(b·(c·d)))
Keep "·" and ")": ·) · ·))	·) ·) ·)	· ·)) ·)	· ·) ·))	· · ·)))
replace "·" with "(": ( ) (( ))	( ) ( ) ( )	(( )) ( )	(( ) ( ))	((( )))

$n=3 \Rightarrow$ polygon $n+2=5$ sides

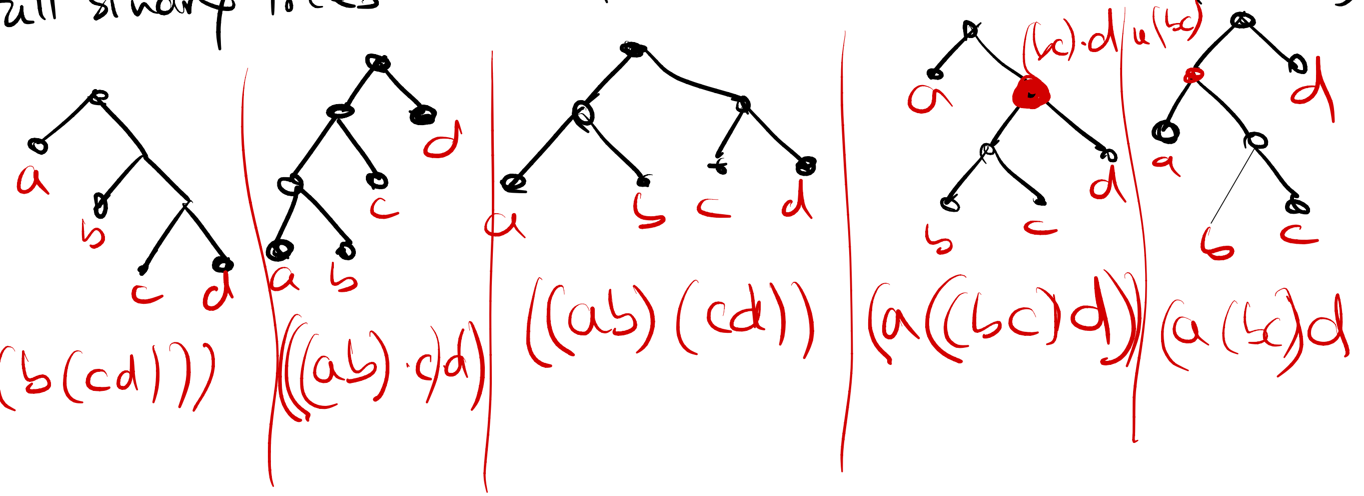


$(((ab)c)d)$ $((a(bc))d)$ $((ab)(cd))$ $(a((bc)d))$ $(a(b(cd)))$

Full binary trees – every node has 2 children (or leaf)

$n=3$

#trees $=5$

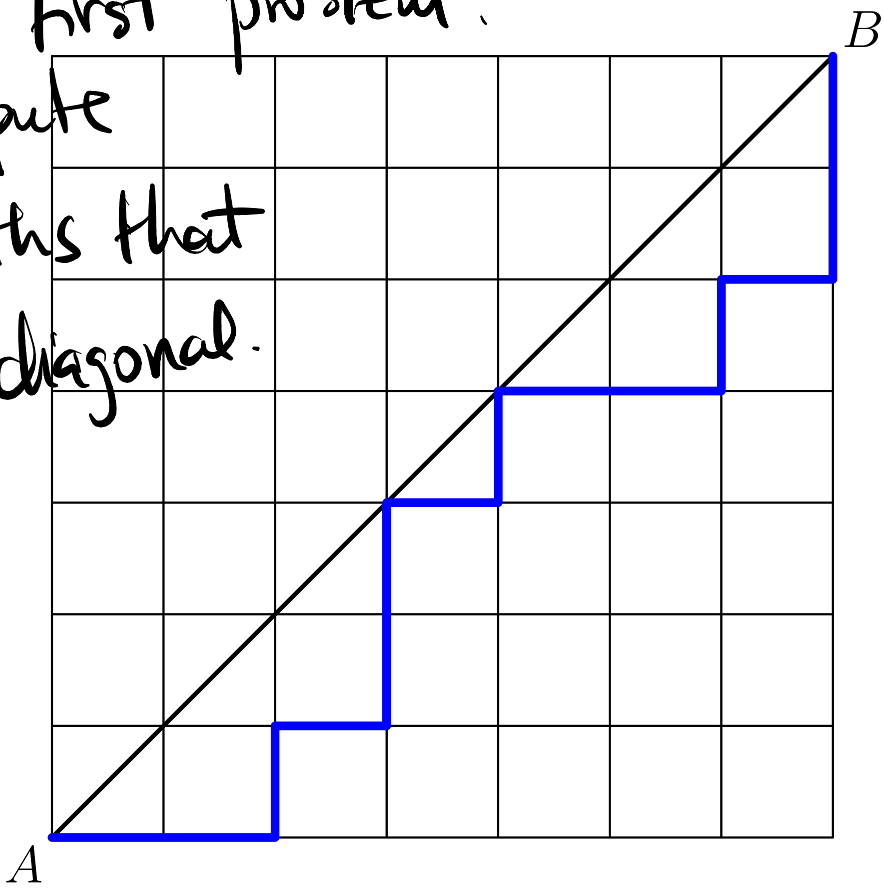


$(a(b(cd)))$ $(((ab)\cdot c)d)$ $((ab)(cd))$ $(a((bc)d))$ $(a(bc)d$

Back to first problem:

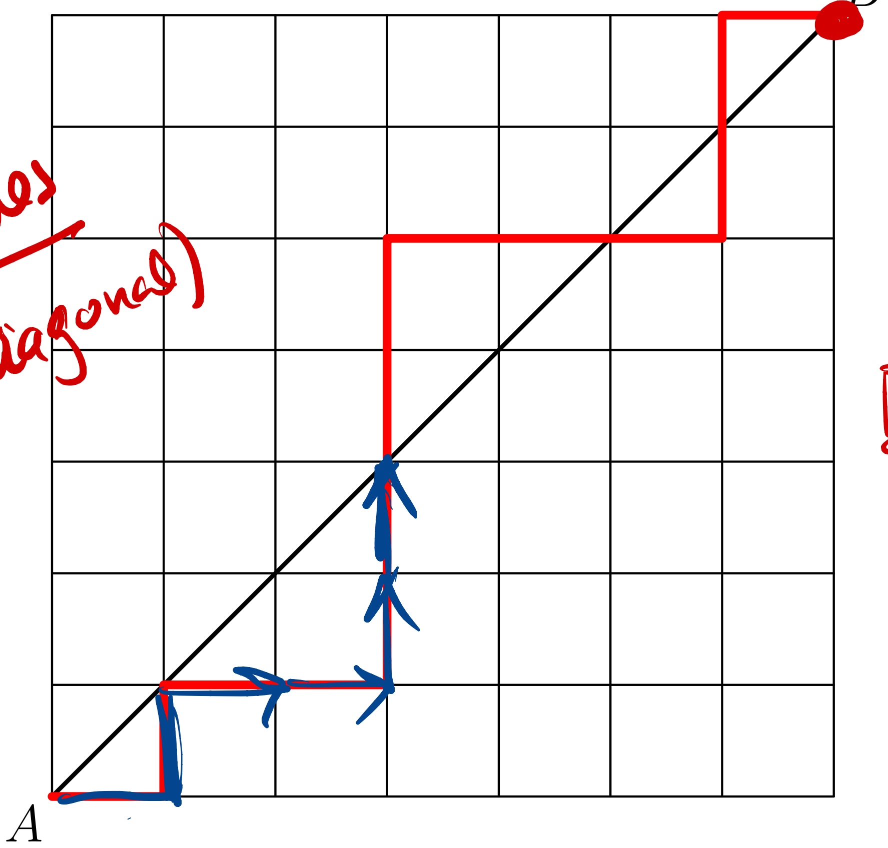
Lets compute

C_n = # paths that dont cross diagonal.



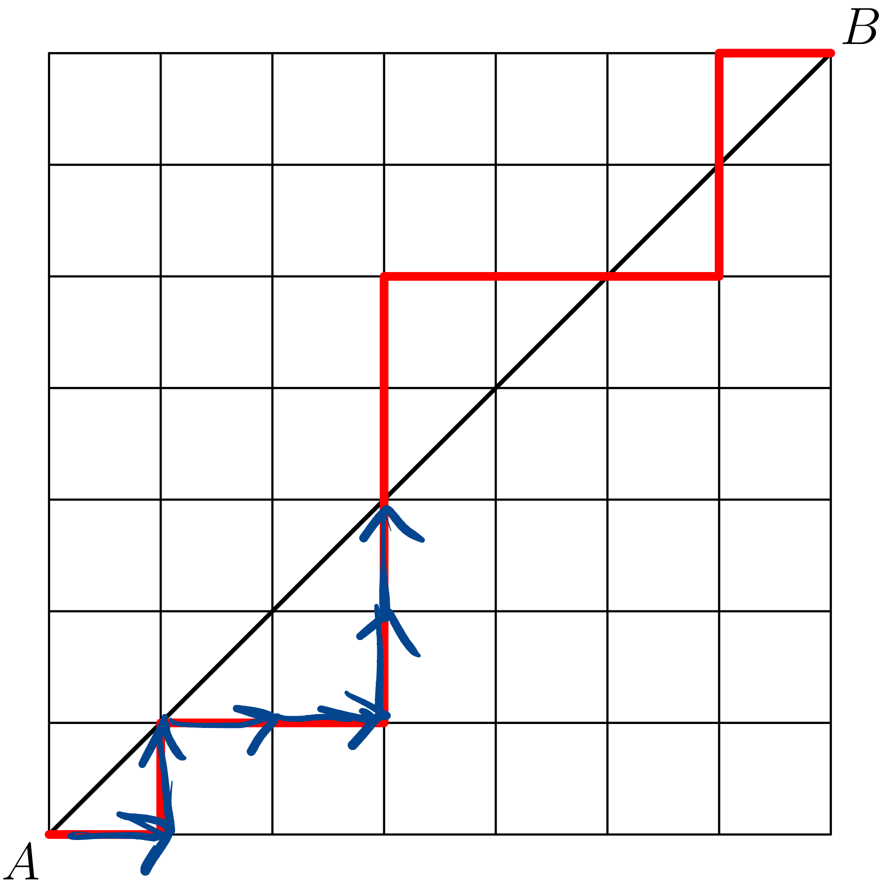
Blue path is good

paths = all paths − invalid ones
(paths cross diagonal)



all paths
$\binom{2n}{n}$

bad ones?



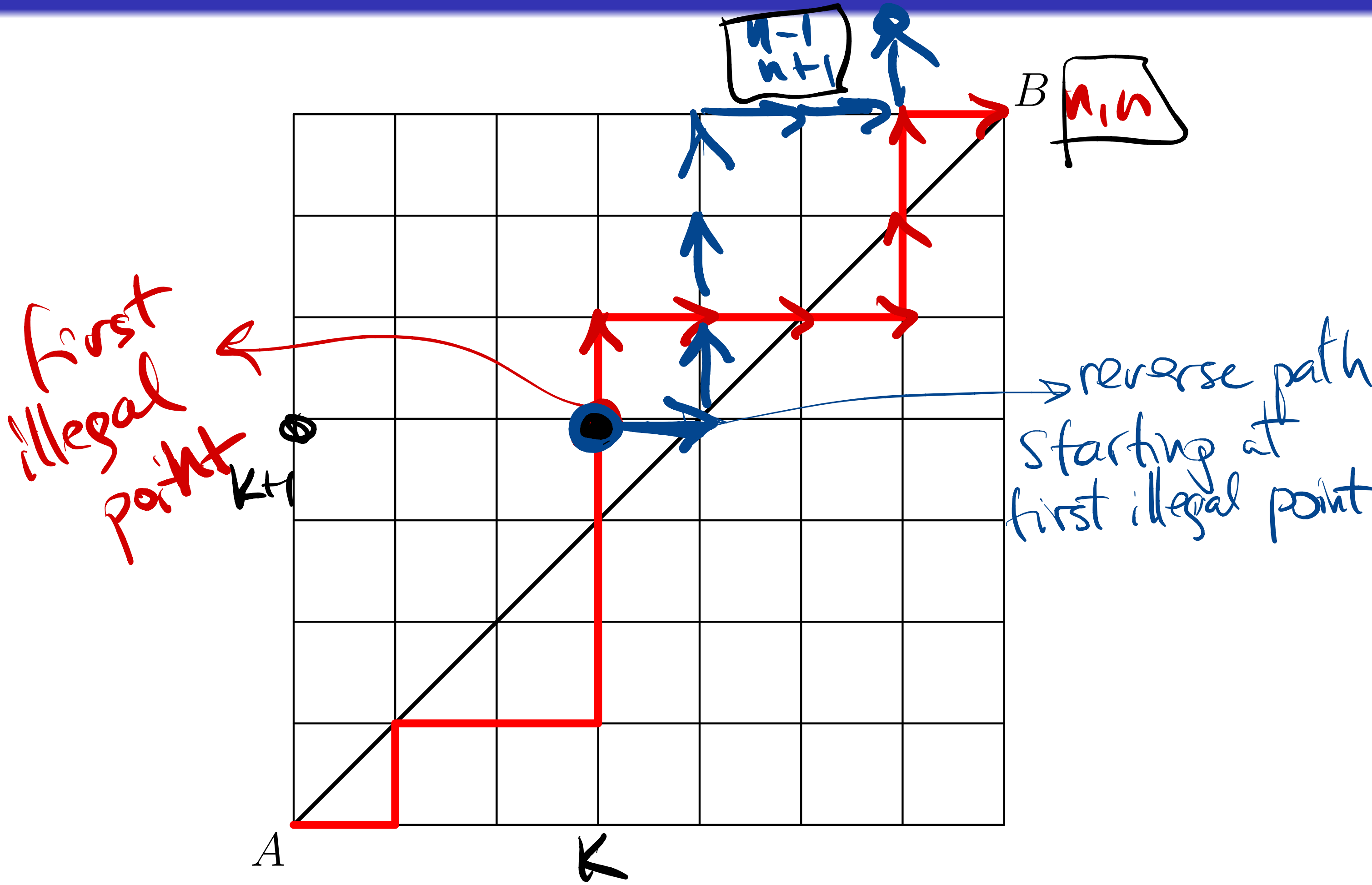
Red path is bad

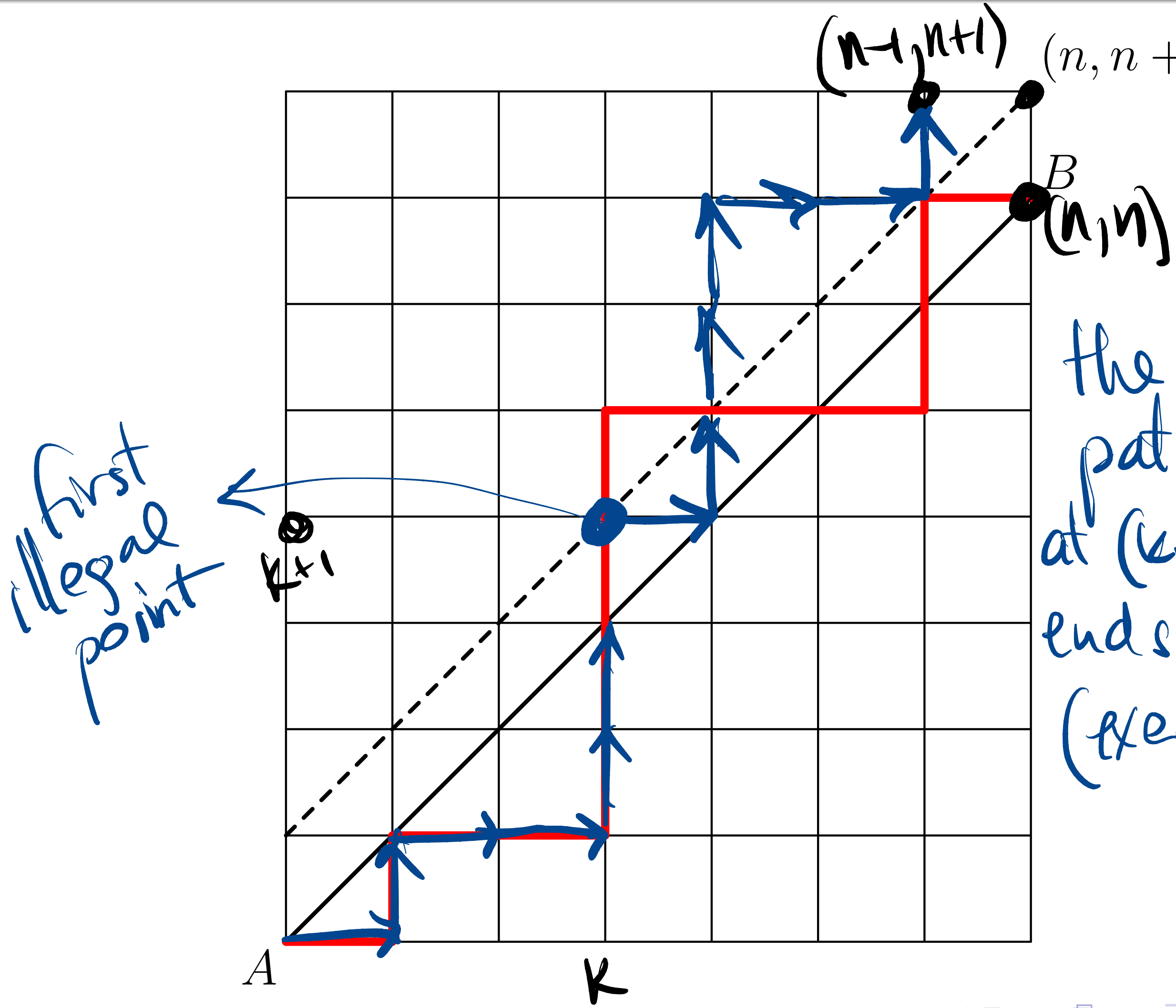
$$
\begin{aligned}
\text{Number of good paths} &= \text{Total Number of paths} - \text{Number of bad paths} \\
&= \frac{(2n)!}{n!n!} - \text{Number of bad paths}
\end{aligned}
$$

$$\begin{aligned}\text{Number of good paths} &= \text{Total Number of paths} - \text{Number of bad paths} \\ &= \frac{(2n)!}{n!n!} - \text{Number of bad paths}\end{aligned}$$

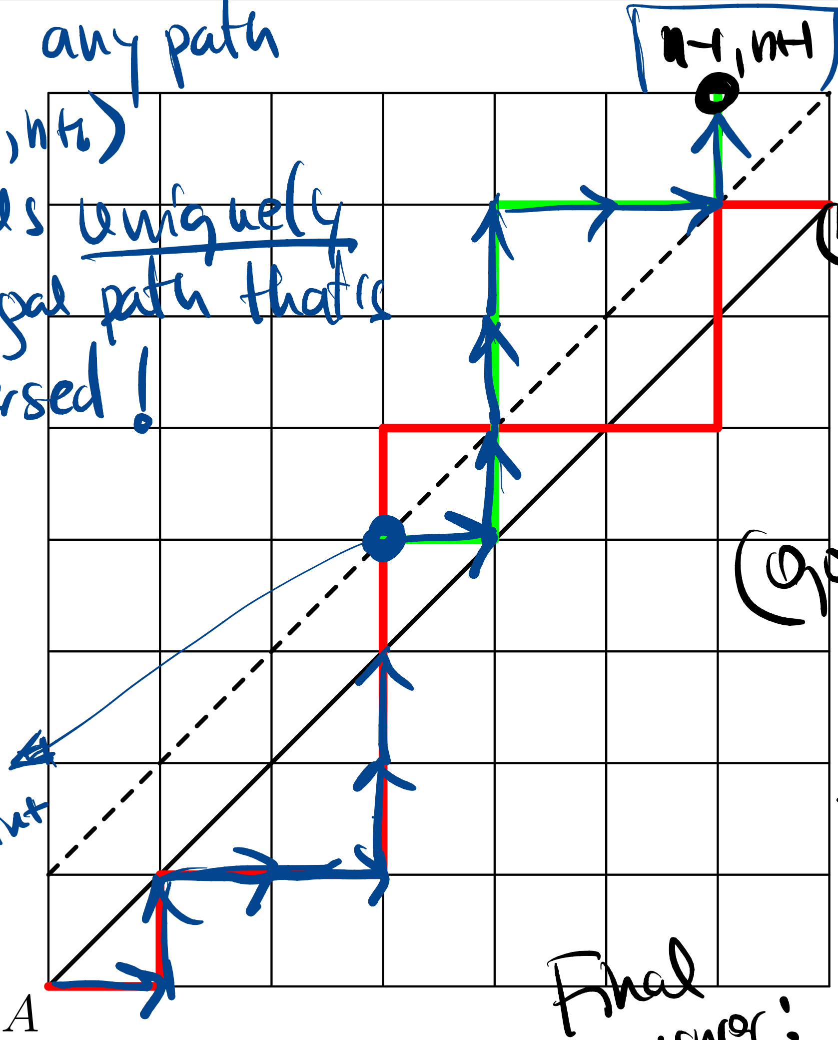
So it is sufficient to count the number of bad paths.

How to count the number of bad paths?





the reversed path starts at $(k, k+1)$ and ends at $(n-1, n+1)$ (exercise: proof) actually any path $(0,0) \to (n-1, n+1)$ corresponds uniquely to an illegal path that's seen reversed!

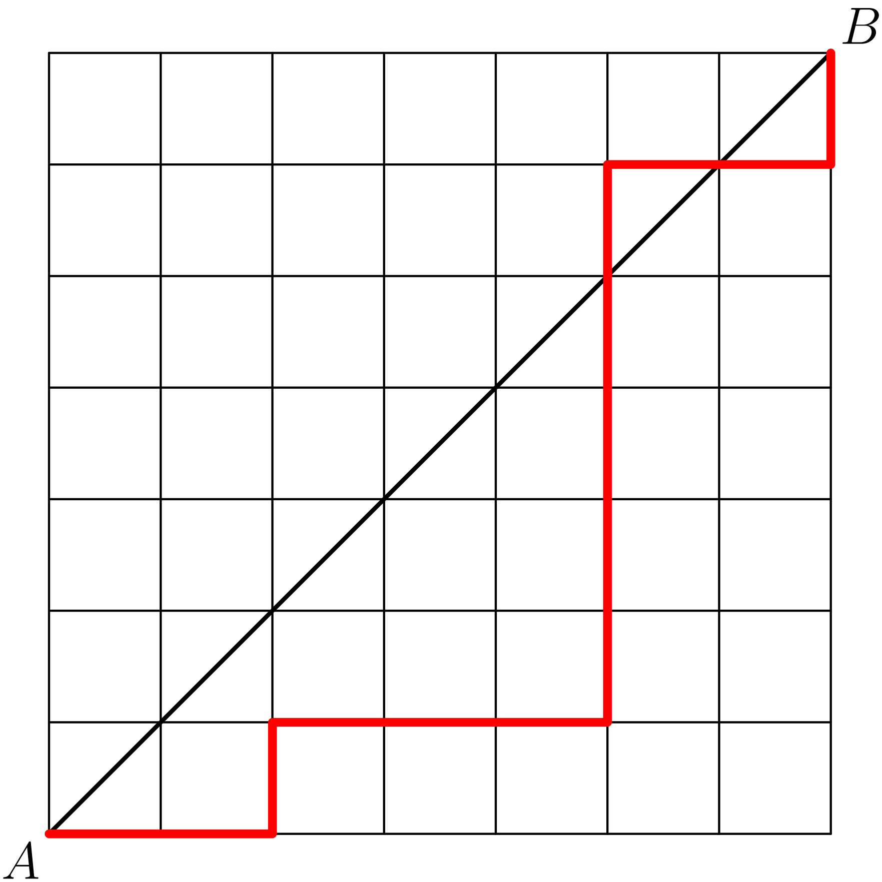


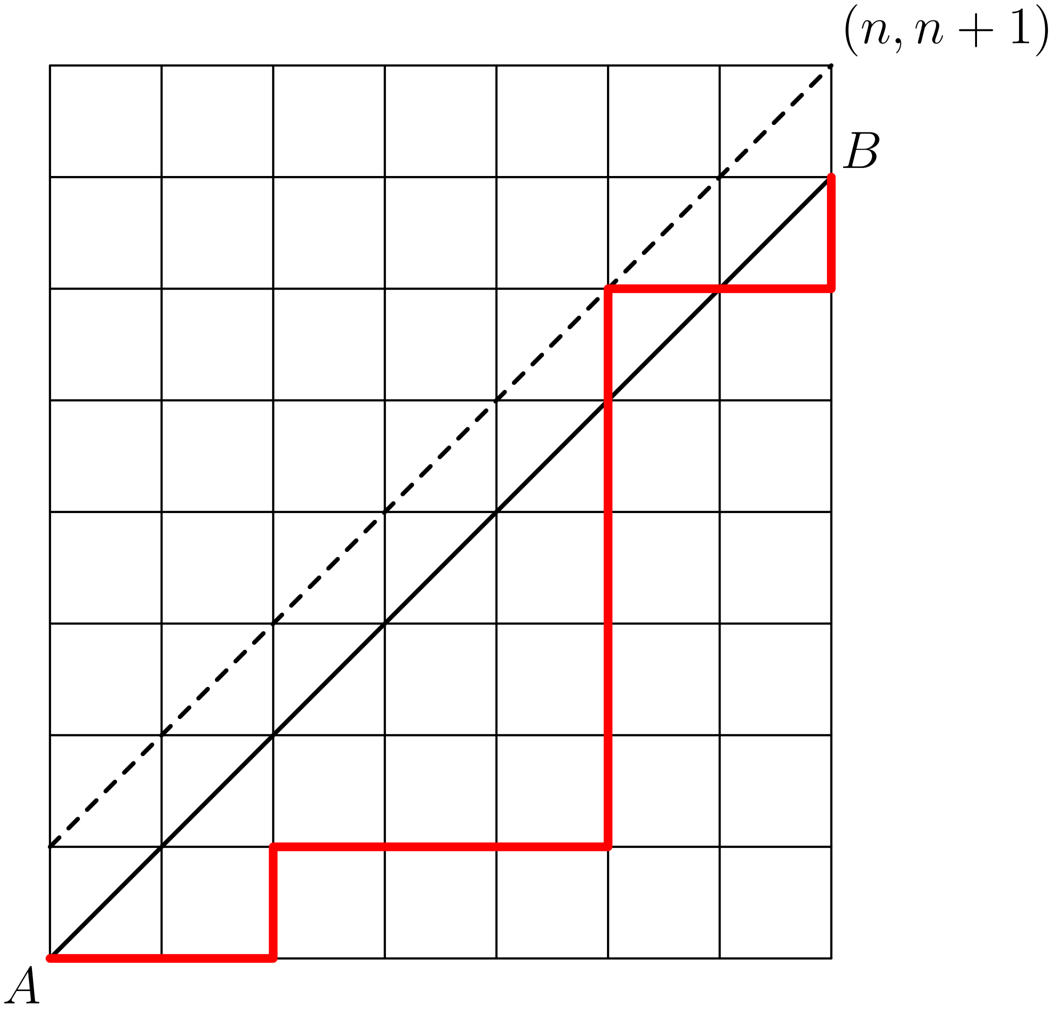
first illegal point

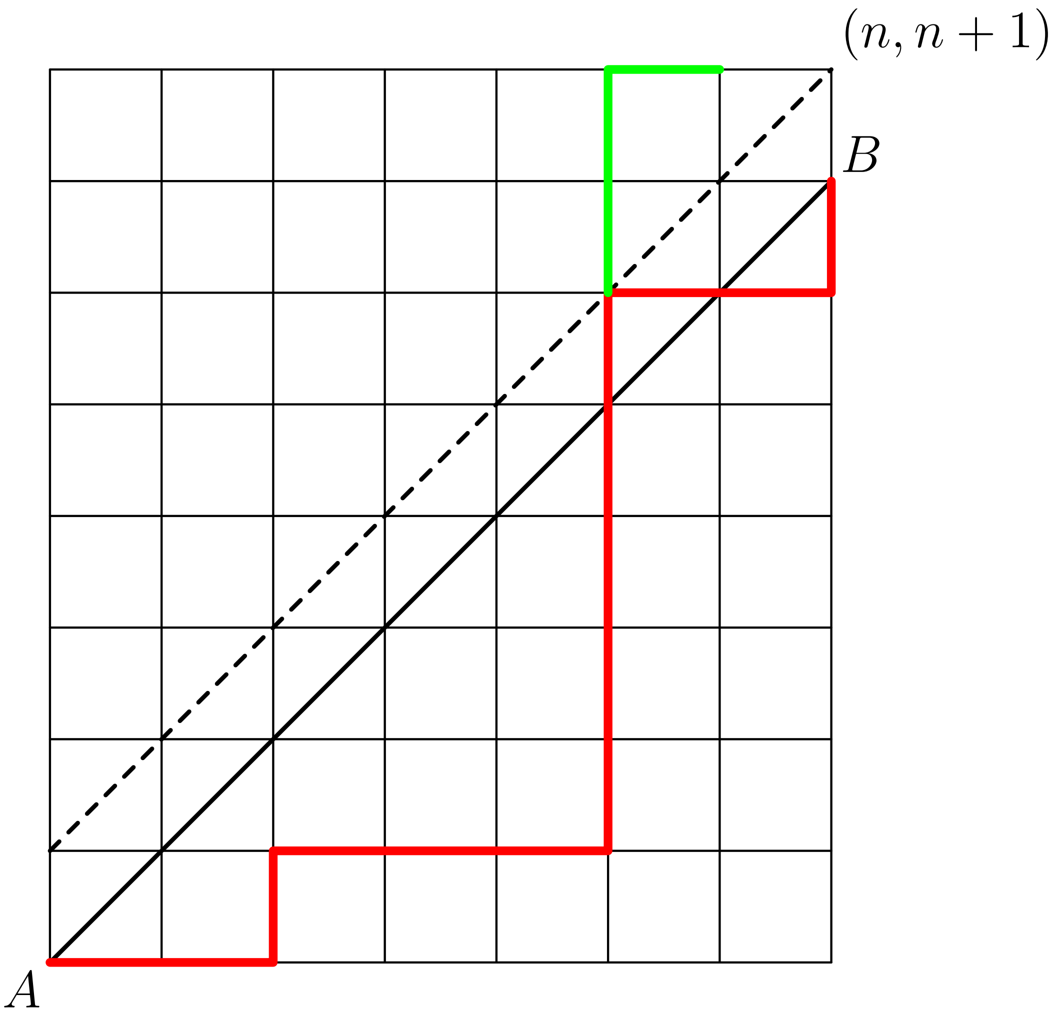
so (n,n) bad paths $\equiv$ all paths $(0,0)$ to $(n-1, n+1)$

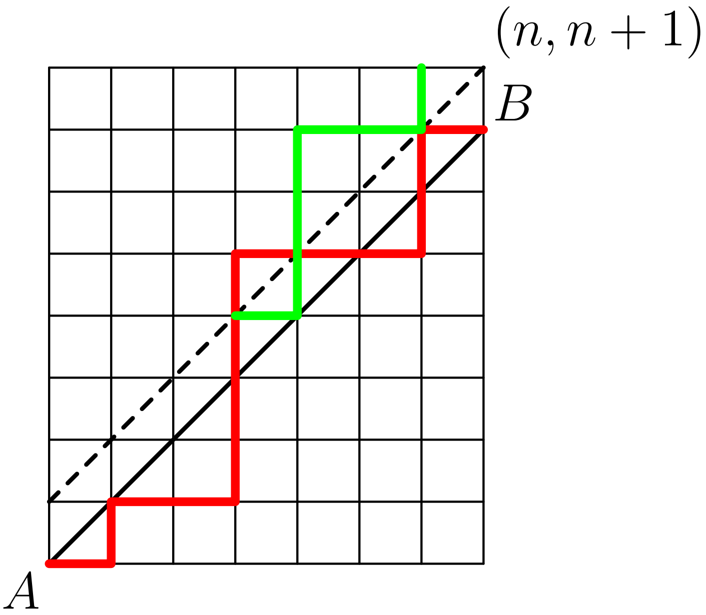
$$= \binom{2n}{n-1} = \binom{2n}{n+1}$$

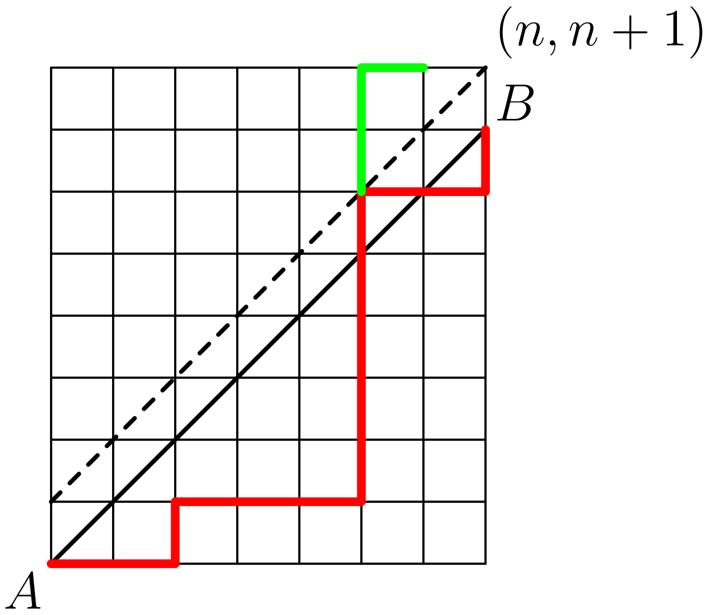
Final answer: $C_n = \binom{2n}{n} - \binom{2n}{n-1}$


B
A


$(n, n+1)$
B
A


$(n, n+1)$
B
A

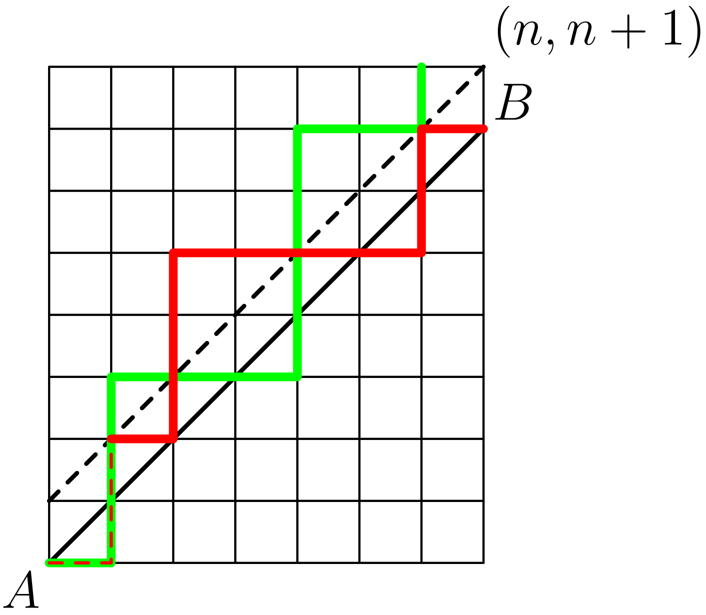




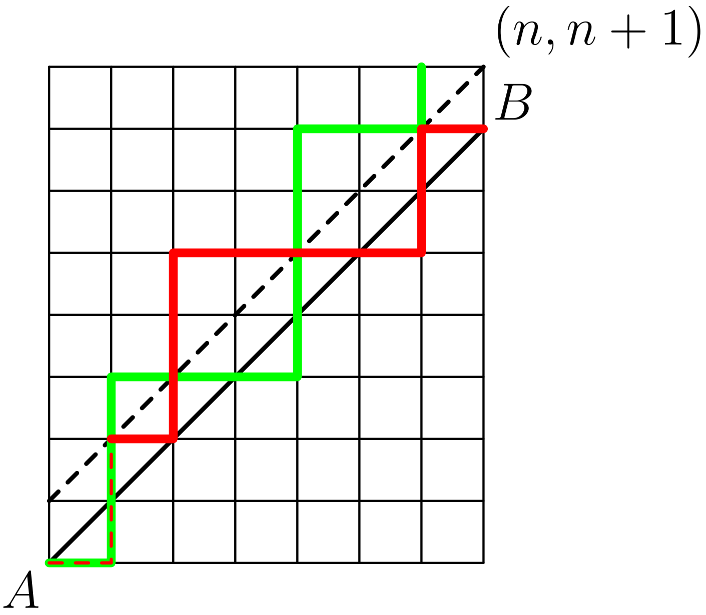
In fact, reflection turns every bad path into a path reaching $(n-1, n+1)$.

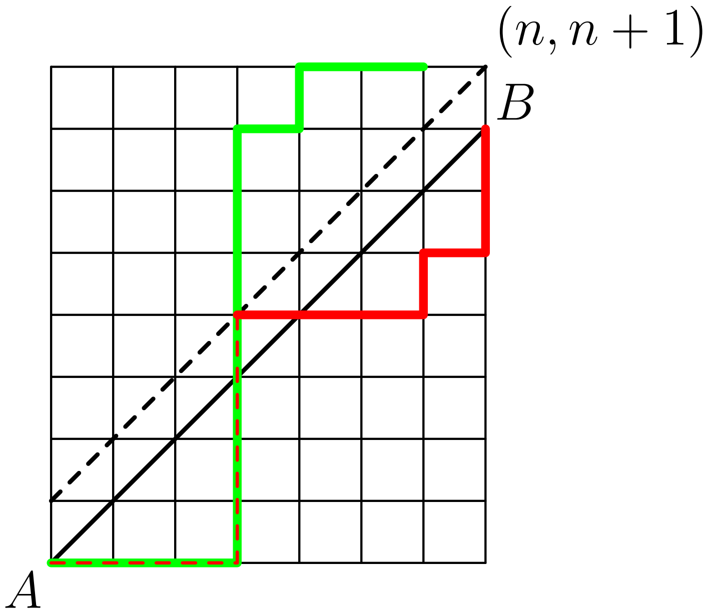
Moreover, every path reaching $(n-1, n+1)$ is obtained from a bad path.

Moreover, every path reaching $(n-1, n+1)$ is obtained from a bad path.



Moreover, every path reaching $(n-1, n+1)$ is obtained from a bad path.





Conclusion: There is a one-to-one correspondence between the set of bad paths and the set of paths reaching $(n-1, n+1)$.

Conclusion: There is a one-to-one correspondence between the set of bad paths and the set of paths reaching $(n-1, n+1)$.

Therefore the number of bad paths must equal the number of paths reaching $(n-1, n+1)$

Conclusion: There is a one-to-one correspondence between the set of bad paths and the set of paths reaching $(n-1, n+1)$.

Therefore the number of bad paths must equal the number of paths reaching $(n-1, n+1)$, which is $\frac{(2n)!}{(n-1)!(n+1)!}$.

Conclusion: There is a one-to-one correspondence between the set of bad paths and the set of paths reaching $(n-1, n+1)$.

Therefore the number of bad paths must equal the number of paths reaching $(n-1, n+1)$, which is $\frac{(2n)!}{(n-1)!(n+1)!}$.

Finally,

Number of good paths

Conclusion: There is a one-to-one correspondence between the set of bad paths and the set of paths reaching $(n-1, n+1)$.

Therefore the number of bad paths must equal the number of paths reaching $(n-1, n+1)$, which is $\frac{(2n)!}{(n-1)!(n+1)!}$.

Finally,

Number of good paths $=$ Total Number of paths $-$ Number of bad paths

Conclusion: There is a one-to-one correspondence between the set of bad paths and the set of paths reaching $(n-1, n+1)$.

Therefore the number of bad paths must equal the number of paths reaching $(n-1, n+1)$, which is $\frac{(2n)!}{(n-1)!(n+1)!}$.

Finally,

$$\begin{aligned}\text{Number of good paths} &= \text{Total Number of paths} - \text{Number of bad paths}\\ &= \frac{(2n)!}{n!n!} - \frac{(2n)!}{(n-1)!(n+1)!}\end{aligned}$$

Conclusion: There is a one-to-one correspondence between the set of bad paths and the set of paths reaching $(n-1, n+1)$.

Therefore the number of bad paths must equal the number of paths reaching $(n-1, n+1)$, which is $\frac{(2n)!}{(n-1)!(n+1)!}$.

Finally,

$$\begin{aligned}
\text{Number of good paths} &= \text{Total Number of paths} - \text{Number of bad paths} \\
&= \frac{(2n)!}{n!n!} - \frac{(2n)!}{(n-1)!(n+1)!} \\
&= \frac{1}{n+1}\binom{2n}{n}
\end{aligned}$$

Conclusion: There is a one-to-one correspondence between the set of bad paths and the set of paths reaching $(n-1, n+1)$.

Therefore the number of bad paths must equal the number of paths reaching $(n-1, n+1)$, which is $\frac{(2n)!}{(n-1)!(n+1)!}$.

Finally,

$$\begin{aligned}
\text{Number of good paths} &= \text{Total Number of paths} - \text{Number of bad paths} \\
&= \frac{(2n)!}{n!n!} - \frac{(2n)!}{(n-1)!(n+1)!} \\
&= \frac{1}{n+1}\binom{2n}{n}
\end{aligned}$$

The number $C_n = \frac{1}{n+1}\binom{2n}{n}$ is called the n^{th} Catalan number and has a lot of applications.