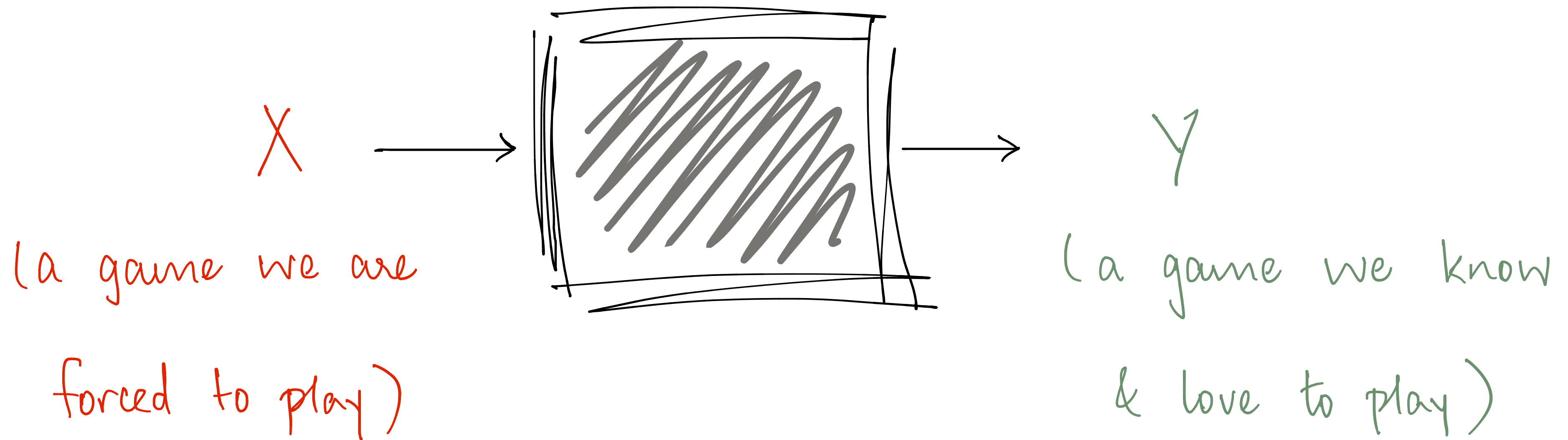


Recap

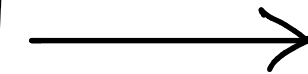
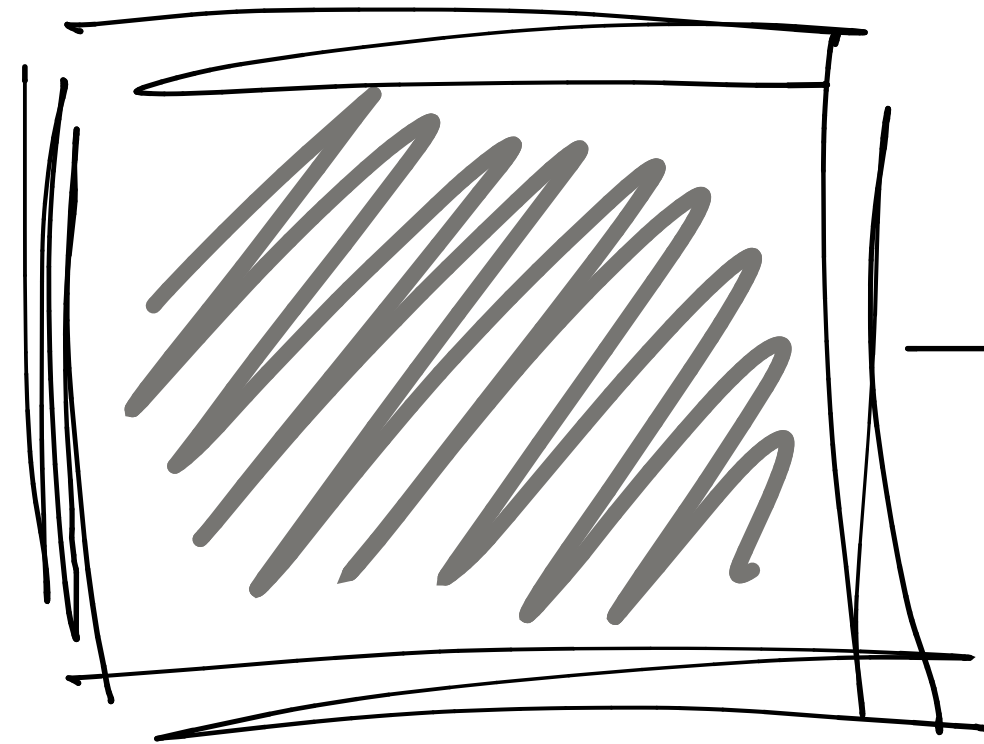
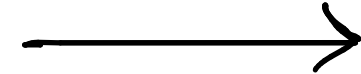
Transformations



~~Recap~~

Transformations

Col

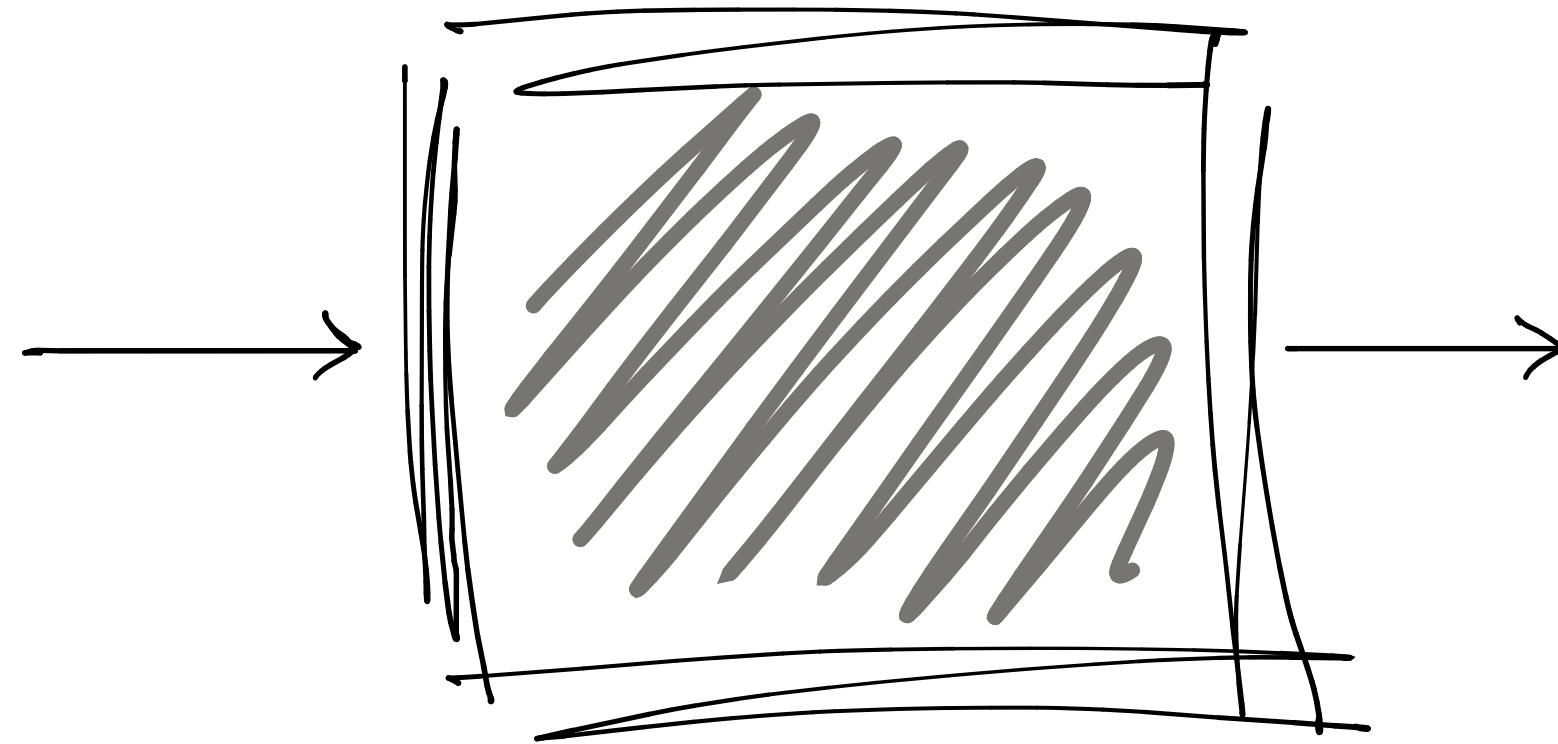


NoGo

~~Recap~~

Transformations

Boolean
Formula
Game



Directed
Geography

Complexity Theory - Detour

(source. Beyond Computation: The P vs NP problem
by Michael Sipser)

Computers can store a ton of information

↳ perform a lot of computation blazing-fast

↳ They can solve some problems really fast

1634733645809253848443133883865090859841783670033092312

181110852389333100104508151212118167511579

*

1900871281664822113126851573935413975471896789968515493

666638539088027103802104498957191261465571

Computers can store a ton of information

↳ perform a lot of computation blazing-fast

↳ They can solve some problems really fast

3 107 418 240 490 043 721 350 750 035 888 567 930 037 346
022 842 727 545 720 161 948 823 206 440 518 081 504 556
346 829 671 723 286 782 437 916 272 838 033 415 471 073
108 501 919 548 529 007 337 724 822 783 525 742 386 454
014 691 736 602 477 652 346 609

Computers can store a ton of information

↳ perform a lot of computation blazing-fast

But some problems take a little longer ☹️

Computers can store a ton of information

↳ perform a lot of computation blazing-fast

But some problems take a little longer 😞

Factor

3 107 418 240 490 043 721 350 750 035 888 567 930 037 346
022 842 727 545 720 161 948 823 206 440 518 081 504 556
346 829 671 723 286 782 437 916 272 838 033 415 471 073
108 501 919 548 529 007 337 724 822 783 525 742 386 454
014 691 736 602 477 652 346 609

Computers can store a ton of information

↳ perform a lot of computation blazing-fast

But some problems take a little longer ☹️

Factor

3 107 418 240 490 043 721 350 750 035 888 567 930 037 346
022 842 727 545 720 161 948 823 206 440 518 081 504 556
346 829 671 723 286 782 437 916 272 838 033 415 471 073
108 501 919 548 529 007 337 724 822 783 525 742 386 454
014 691 736 602 477 652 346 609

RSA Challenge

Why is factoring 'harder' than multiplication?

Why is factoring 'harder' than multiplication?

Brute-force search: the possibilities are astronomical
(even with heuristic speed-ups)

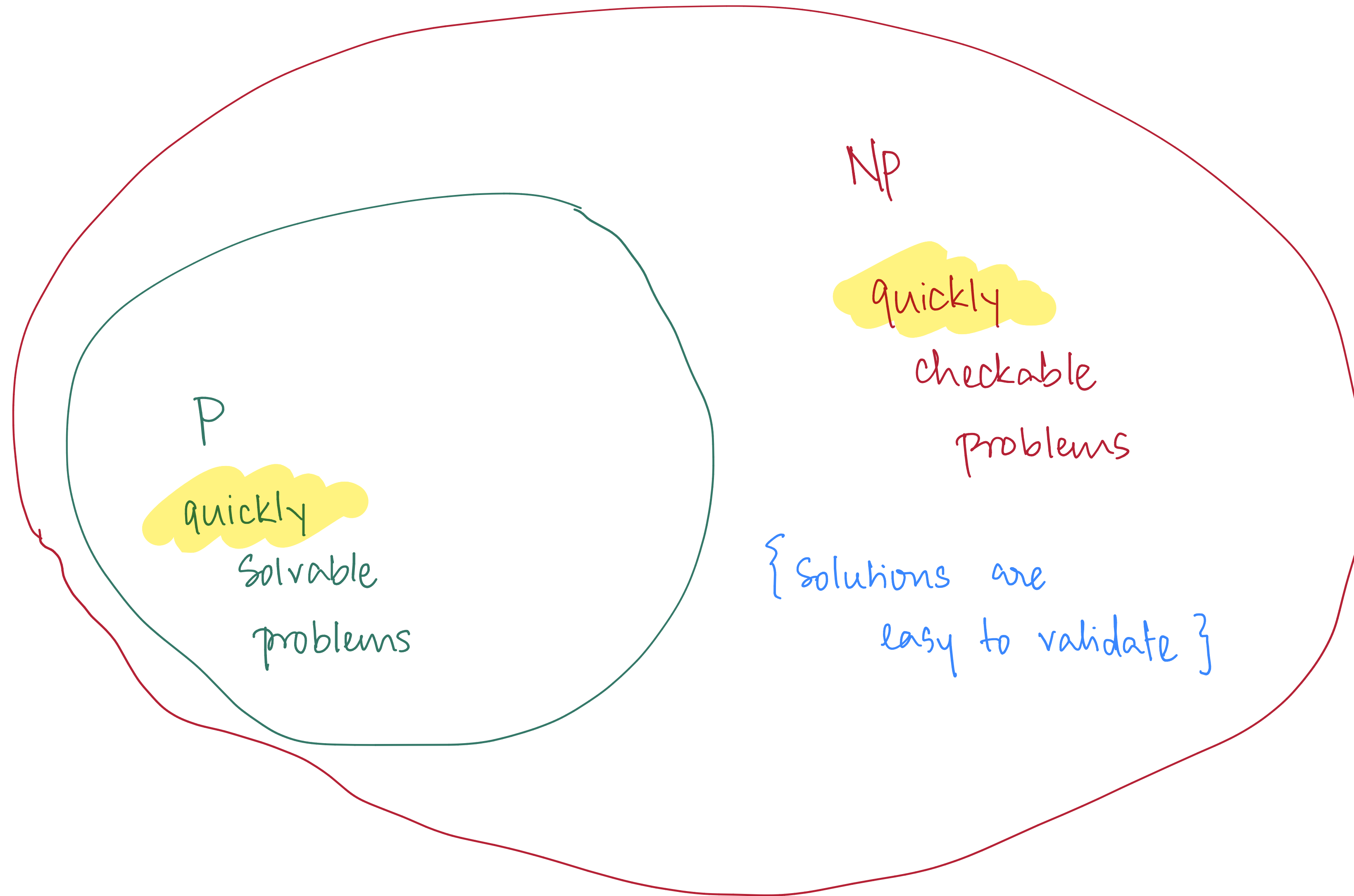
Why is factoring 'harder' than multiplication?

Brute-force search: the possibilities are astronomical
(even with heuristic speed-ups)

Is all this searching really necessary,
or is there a smarter shortcut?

"Needle in a haystack" - type problems





Suppose you have a machine that
can tell if a mathematical statement
has a proof of length n .

$\varphi(n) \rightsquigarrow$ The time needed by such a machine

How fast does $\varphi(n)$ grow
for an optimal machine?

A Strange Way to Test Primality

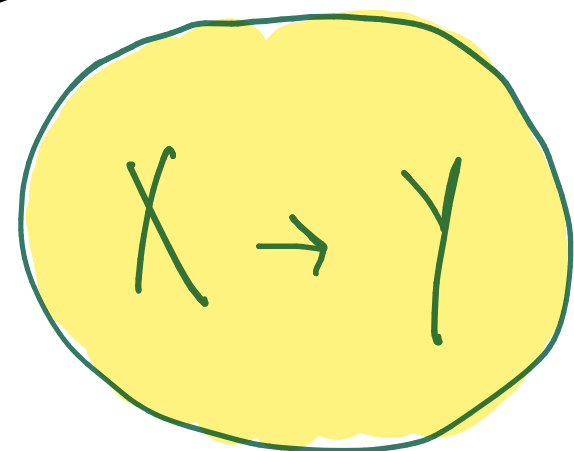
For a prime p & a natural number $a < p$,

$$a^{p-1} = 1 \pmod{p}$$

Eg. $a=2, p=7$; $2^6 = 64 = 1 \pmod{7}$

But also: $a=2, p=4$; $2^3 = 8 \neq 1 \pmod{4}$ (basis for a primality test?)

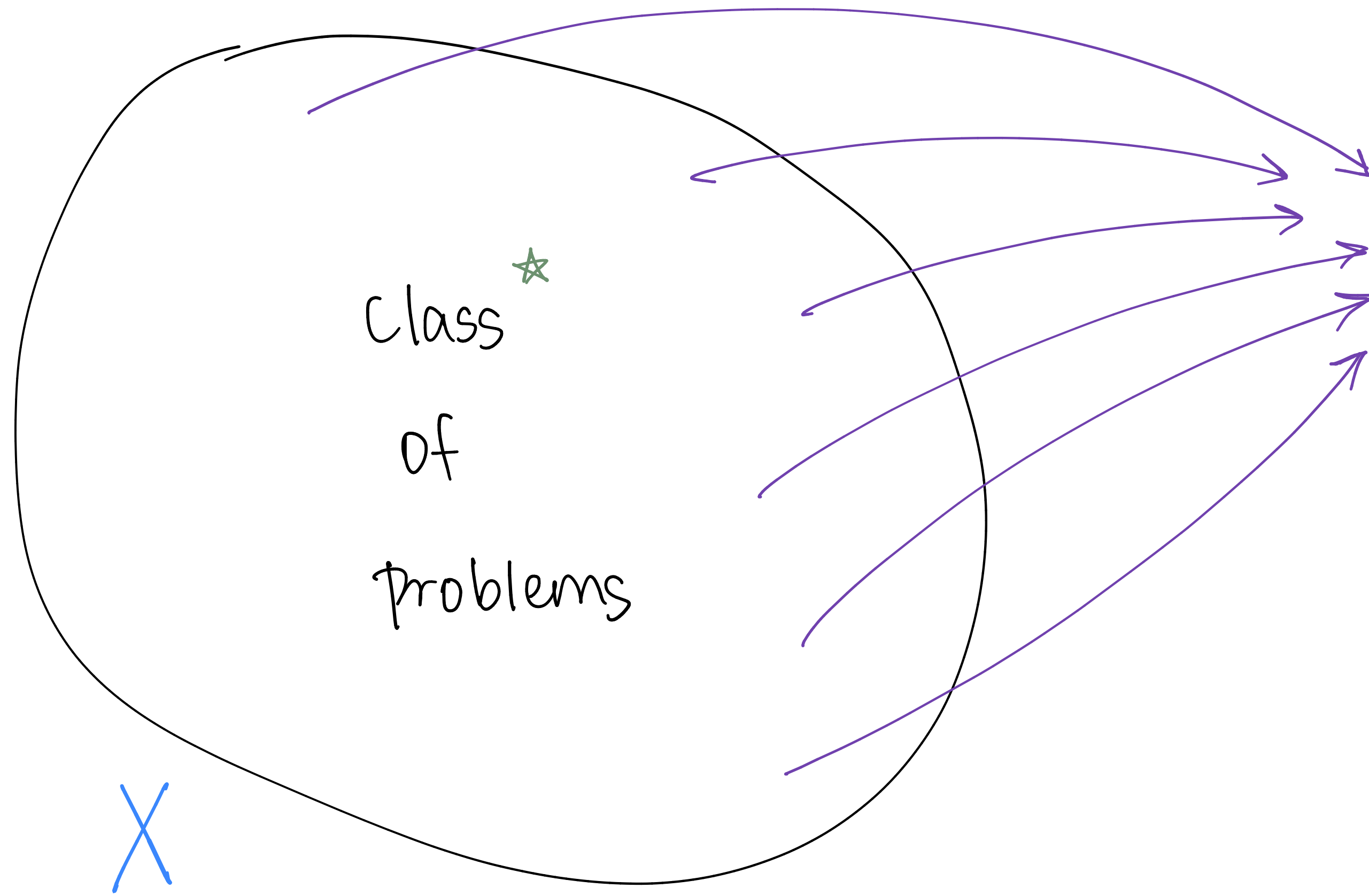
Transformations


$$X \rightarrow Y$$

If you can solve X 'quickly'

then you can solve Y 'quickly' too!

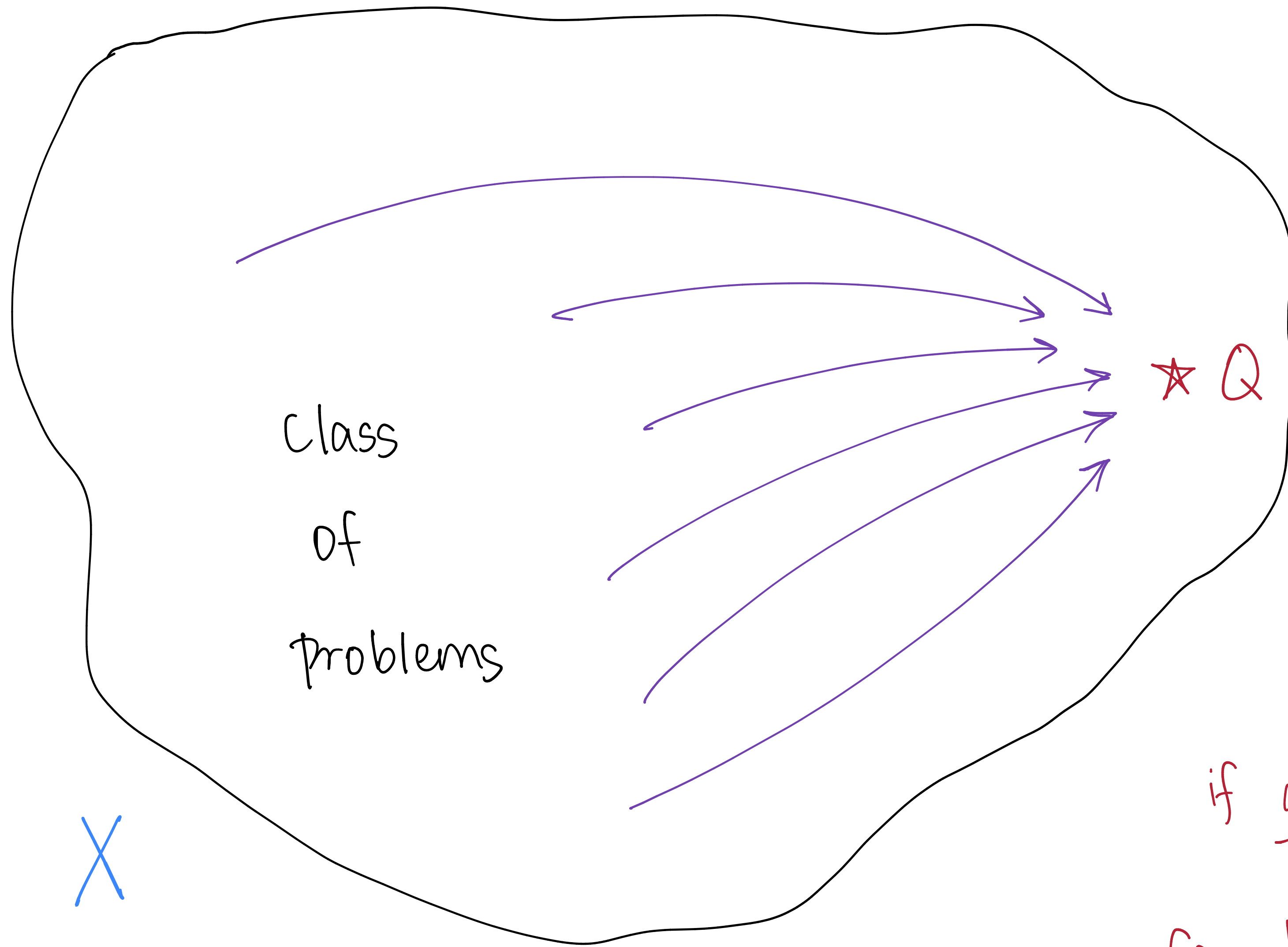
★ typically defined in terms of resources required to solve them



★ $Q \rightsquigarrow$ a specific problem is said to be

X - HARD

if all problems in X can be transformed to Q .



↪ a specific problem is said to be

X - COMPLETE

if all problems in X can be transformed to Q & $Q \in X$.

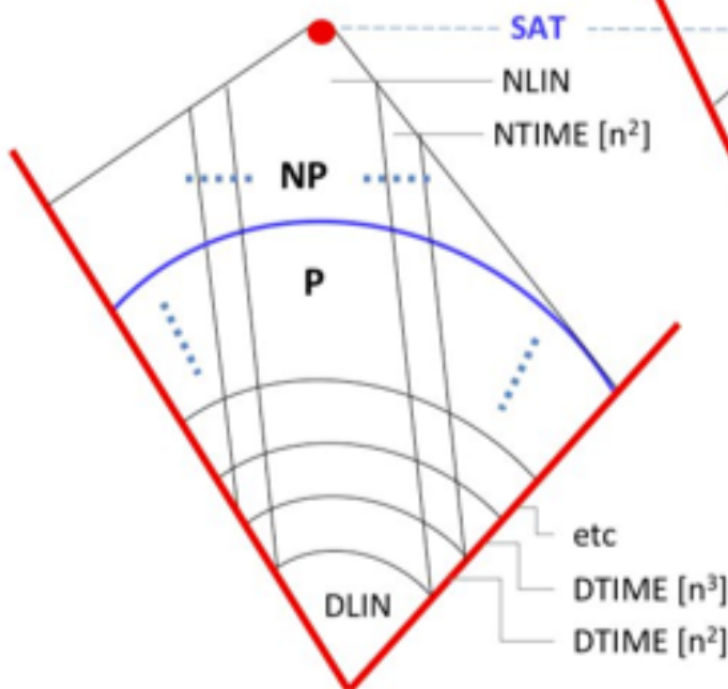
A complete language for EXPSPACE:
PIM, "Polynomial Ideal
Membership"—the simplest natural
completeness level that is known
not to have polynomial-size circuits.

Succinct 3SAT

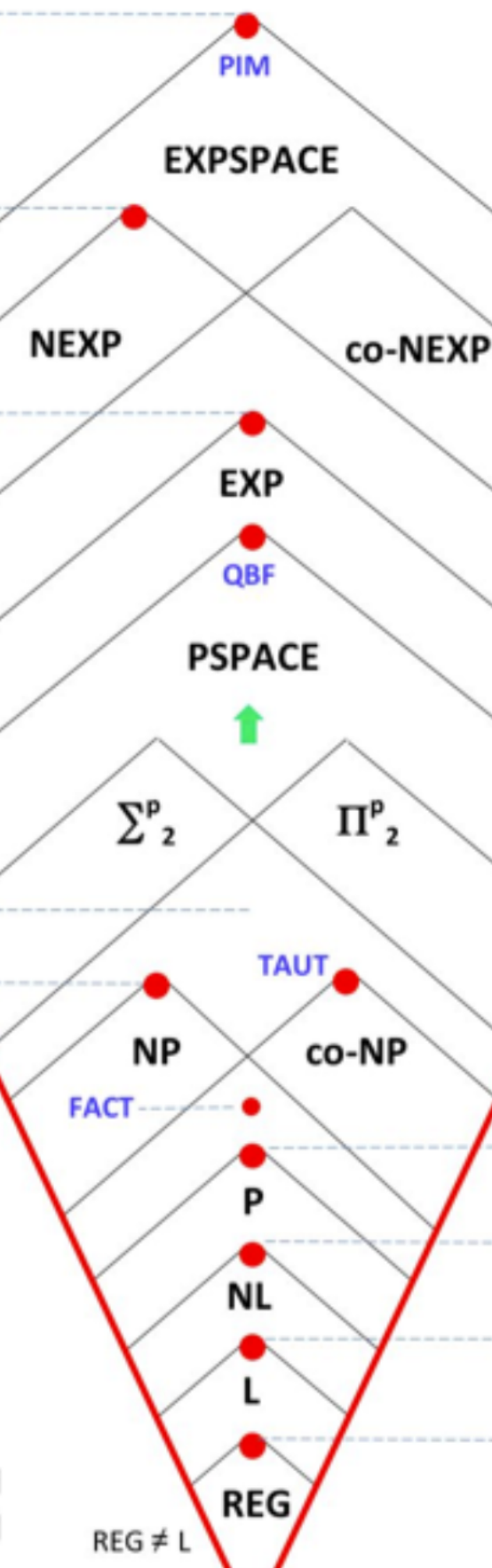
$n \times n$ Chess

For any fixed k ,
there is a
problem in this
intersection that
can NOT be
solved by circuits
of size $O(n^k)$

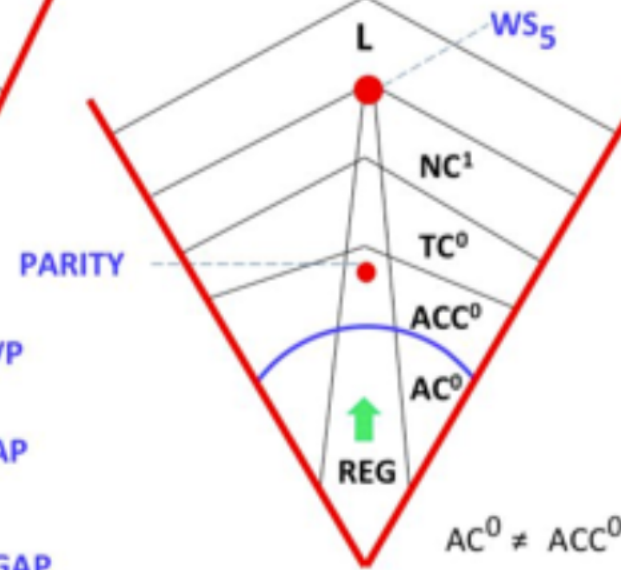
SAT
NLIN
NTIME [n^2]



A Deterministic and Nondeterministic Time Hierarchies Within NP



B Complexity "Main Sequence"



C Low-Level Classes

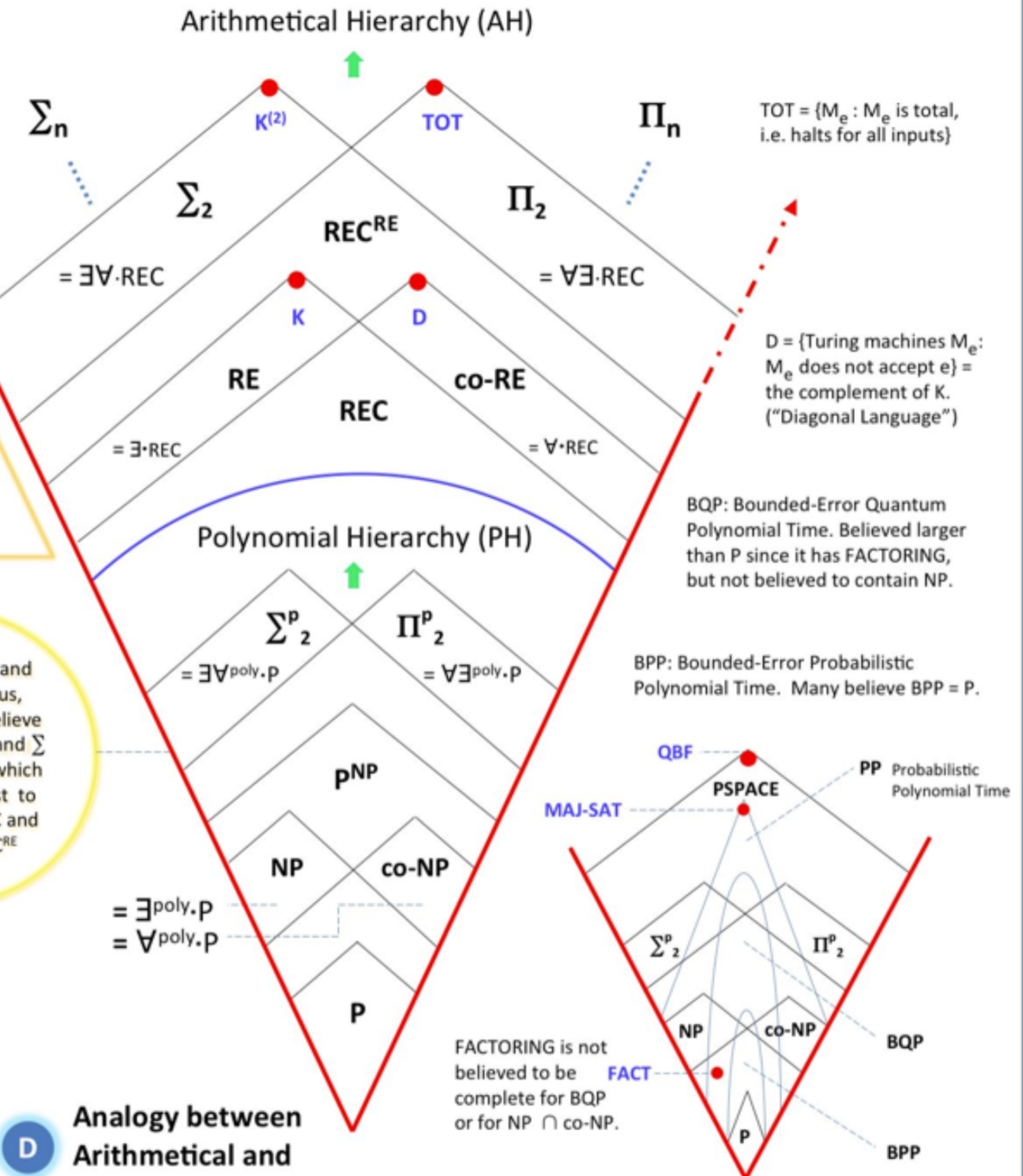
WS_5 , the word problem for the symmetric group S_5 , is a regular language that is complete for NC^1 under AC^0 many-one reductions.

Unknown but Commonly Believed:

- $L \neq NL \dots \dots \dots L \neq PH$
 - $P \neq NP \cap co-NP \dots \dots P \neq PSPACE$
 - $NP \neq \Sigma_2^P \cap \Pi_2^P \dots \dots NP \neq EXP$
- Best Known Separations:
- $AC^0 \subset ACC^0 \subset PP$, also $TC^0 \subset PP$
 - $NC^1 \subset PSPACE, \dots, NL \subset PSPACE$
 - $P \subset EXP, NP \subset NEXP$
 - $PSPACE \subset EXPSPACE$

The levels of AH and PH are analogous, except that we believe $NP \cap co-NP \neq P$ and $\Sigma_2^P \cap \Pi_2^P \neq P^{NP}$, which stand in contrast to $RE \cap co-RE = REC$ and $\Sigma_2 \cap \Pi_2 = REC^{RE}$

D Analogy between Arithmetical and Polynomial Hierarchies



E Realm of Feasibility?

FACTORING is not believed to be complete for BQP or for $NP \cap co-NP$.

$TOT = \{M_e : M_e \text{ is total, i.e. halts for all inputs}\}$

$D = \{\text{Turing machines } M_e : M_e \text{ does not accept } e\} = \text{the complement of } K. \text{ ("Diagonal Language")}$

BQP: Bounded-Error Quantum Polynomial Time. Believed larger than P since it has FACTORING, but not believed to contain NP.

BPP: Bounded-Error Probabilistic Polynomial Time. Many believe BPP = P.

PP Probabilistic Polynomial Time

BQP

BPP

Claim. Hanabi is NP-complete.

Recall.

$v = \# \text{ values}$

$h = \text{storage threshold}$

$C = \# \text{ colors}$

$r = \# \text{ repeats (upper bound)}$

i/p \rightsquigarrow a stream of N cards

(moves: keep, discard, or play)

o/p: Decide if full stacks can be formed for all colors.

3SAT

formula φ

n variables

x_1, x_2, \dots, x_n

m clauses

$\dots (x_1 \text{ OR } \bar{x}_7 \text{ OR } x_8) \dots$

C_i

Goal. Π_φ is

playable iff

φ is satisfiable

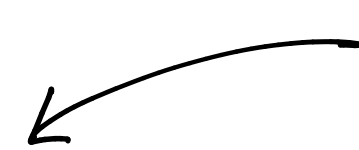
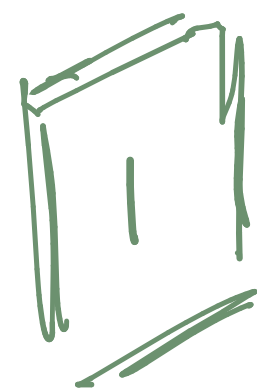
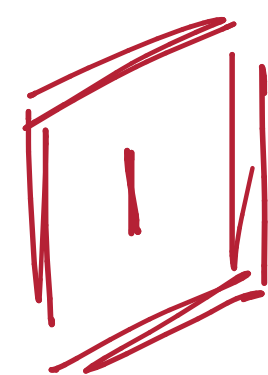


seq. of cards Π_φ

Warm-up \rightsquigarrow how do we get a play sequence
to correspond to an assignment?

Warm-up \rightsquigarrow how do we get a play sequence
to correspond to an assignment?

Attempt #1



play this?

set x_i to True



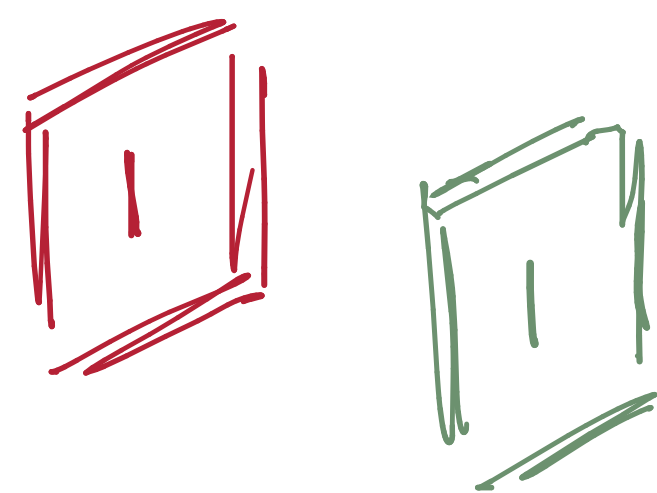
play this?

set x_i to False

Warm-up \rightsquigarrow how do we get a play sequence

to correspond to an assignment?

Attempt #1



play this?
set x_i to True

play this?
set x_i to False

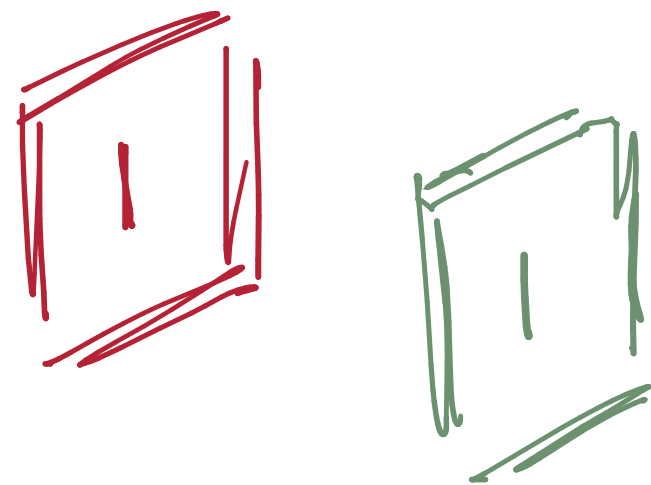
Idea:

1 red & 1 green card
of value 1, 2, 3, ..., n

Warm-up \rightsquigarrow how do we get a play sequence

to correspond to an assignment?

Attempt #1



play this?
set x_i to True

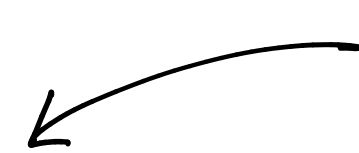
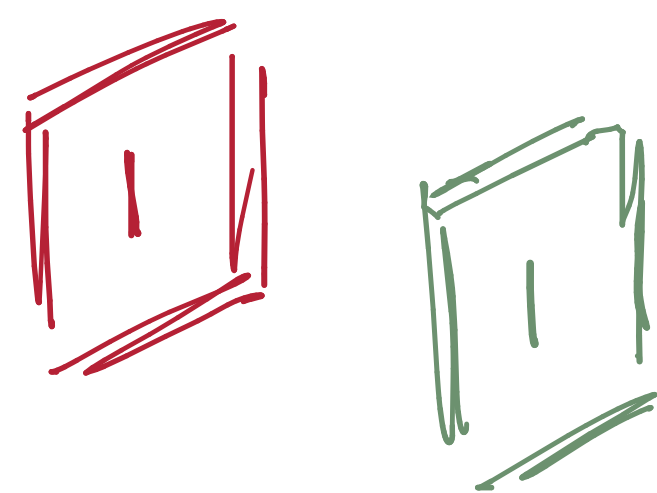
play this?
set x_i to False

1 2 3 ... n
1 2 3 ... n

Warm-up \rightsquigarrow how do we get a play sequence

to correspond to an assignment?

Attempt #1



play this?

set x_i to True



play this?

set x_i to False

1 2 3 ... n

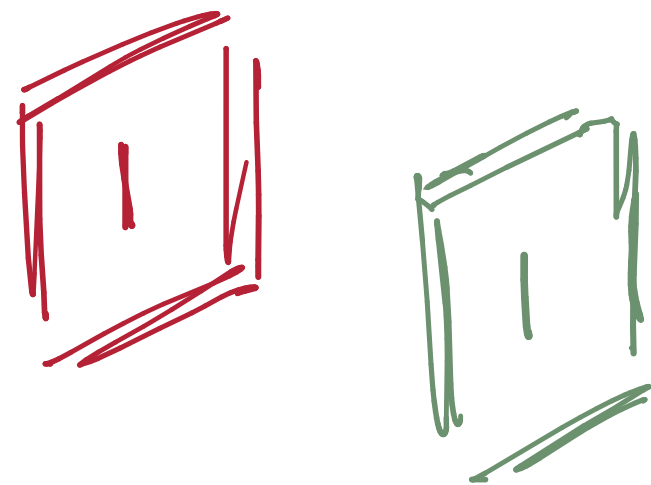
1 2 3 ... n

What's the problem with this?

Warm-up \rightsquigarrow how do we get a play sequence

to correspond to an assignment?

Attempt #1



play this?
set x_i to True

play this?
set x_i to False

1 2 3 ... n
1 2 3 ... n

What's the problem with this?

► Can't force a hold!

Warm-up \rightsquigarrow how do we get a play sequence
to correspond to an assignment?

Attempt #2

Associate a different color with every variable

Warm-up \rightsquigarrow how do we get a play sequence
to correspond to an assignment?

Attempt #2

Associate a different color with every variable

Enforce some kind of blocking mechanism

can do this OR that
but NOT both
and NOT neither.

Warm-up \rightsquigarrow how do we get a play sequence
to correspond to an assignment?

Attempt #2

Associate a different color with every variable

$X_1 \rightsquigarrow$ colors are RED₁ and GREEN₁



Warm-up \rightsquigarrow how do we get a play sequence
to correspond to an assignment?

Attempt #2

Associate a different color with every variable

$X_1 \rightsquigarrow$ colors are RED₁ and GREEN₁



Warm-up \rightsquigarrow how do we get a play sequence
to correspond to an assignment?

Attempt #2

Associate a different color with every variable

$X_1 \rightsquigarrow$ colors are RED₁ and GREEN₁



Warm-up \rightsquigarrow how do we get a play sequence
to correspond to an assignment?

Attempt #2

Associate a different color with every variable

$X_1 \rightsquigarrow$ colors are RED₁ and GREEN₁

have to hold
these to play
the seq 1-2-3



Warm-up \rightsquigarrow how do we get a play sequence
to correspond to an assignment?

Attempt #2

Associate a different color with every variable

$X_1 \rightsquigarrow$ colors are RED₁ and GREEN₁

have to hold
these to play
the seq 1-2-3



if $h=1$
we are forced
to make a choice

Warm-up \rightsquigarrow how do we get a play sequence

to correspond to an assignment?

Attempt #2

We can either play : 1 2 3 1 (discard 2)

or : 1 2 3 1 (discard 2)

have to hold
these to play
the seq 1-2-3



if $h=1$
we are forced
to make a choice

Useful Hack: Suppose we

set $h=2$ for our game

(for whatever reason)

Useful Hack: Suppose we

set $h=2$ for our game

(for whatever reason)

but for some part of the sequence, we want to force $h=1$

(for whatever reason)

Useful Hack: Suppose we

set $h=2$ for our game

(for whatever reason)

but for some part of the sequence,

we want to force $h=1$

(for whatever reason)

how do we do this?

Useful Hack: Suppose we set $h=2$ for our game

(for whatever reason)

but for some part of the sequence, we want to force $h=1$

(for whatever reason)

how do we do this?



Useful Hack: Suppose we set $h=2$ for our game

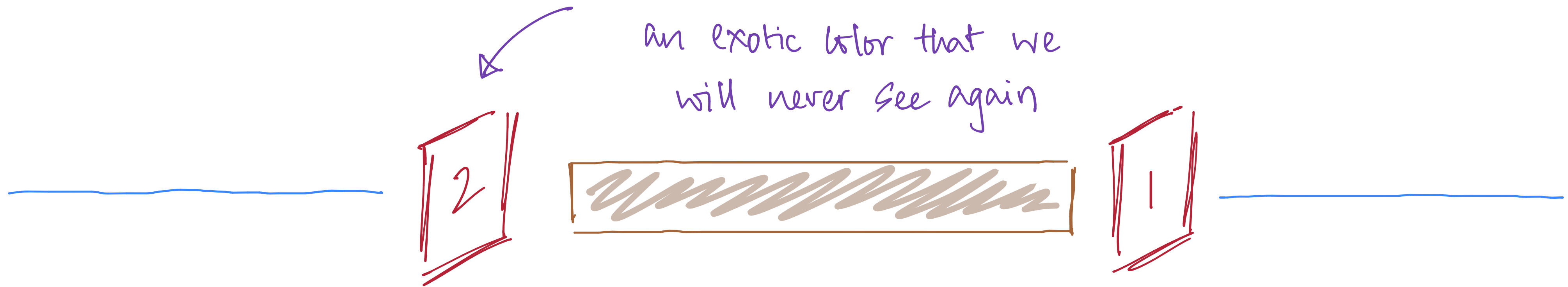
(for whatever reason)

but for some part of the sequence, we want to force $h=1$

(for whatever reason)

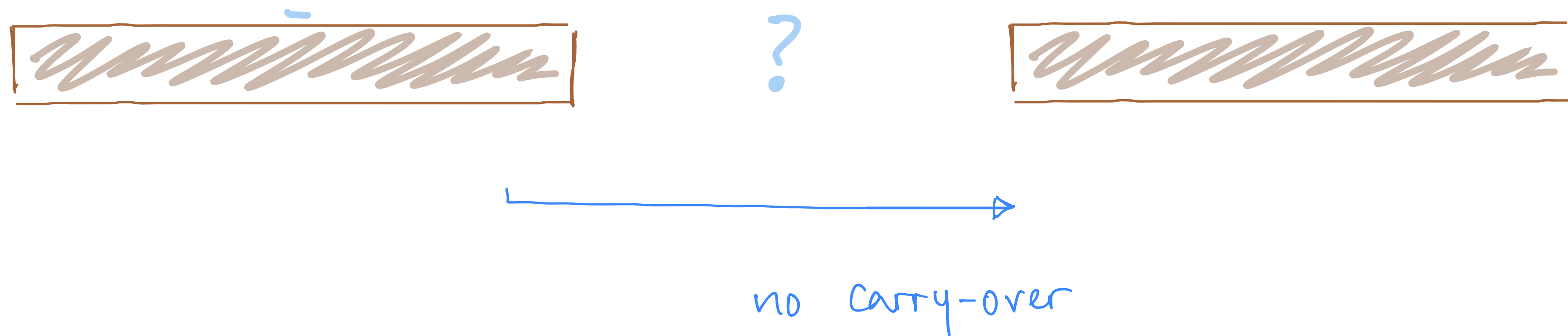
how do we do this? forced hold via

an exotic color that we will never see again

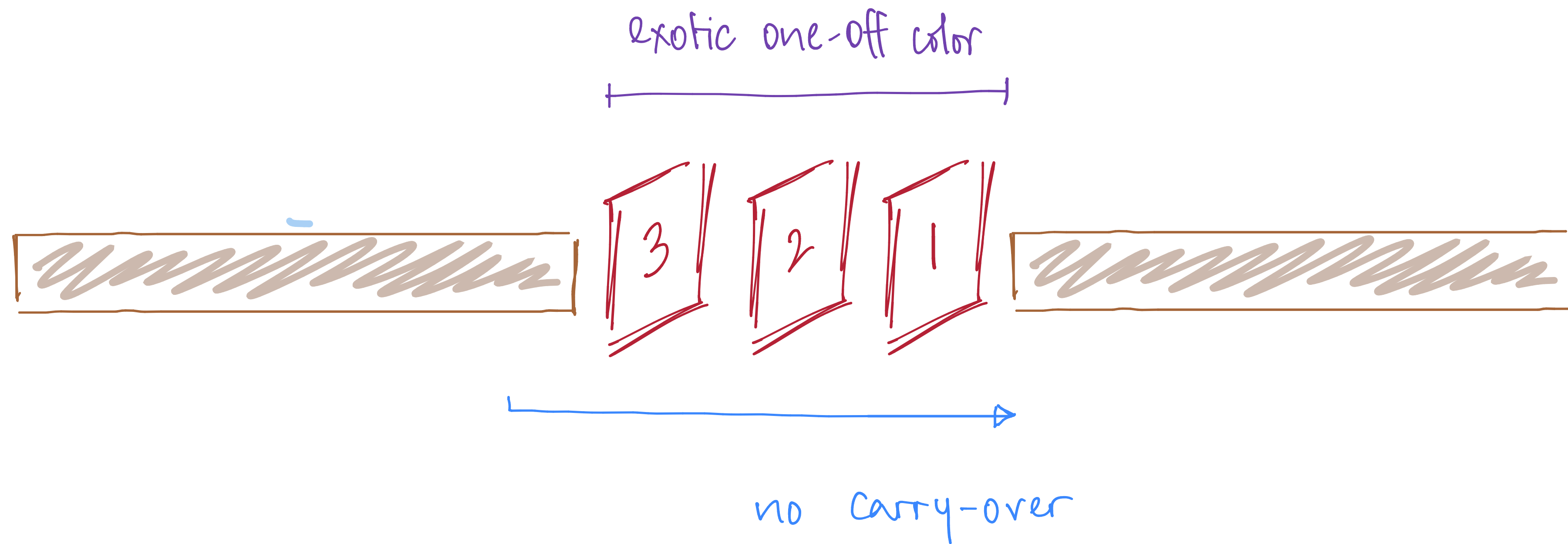


Useful hack: Suppose we want to force that
no cards from one part of our sequence
are carried over to the next part

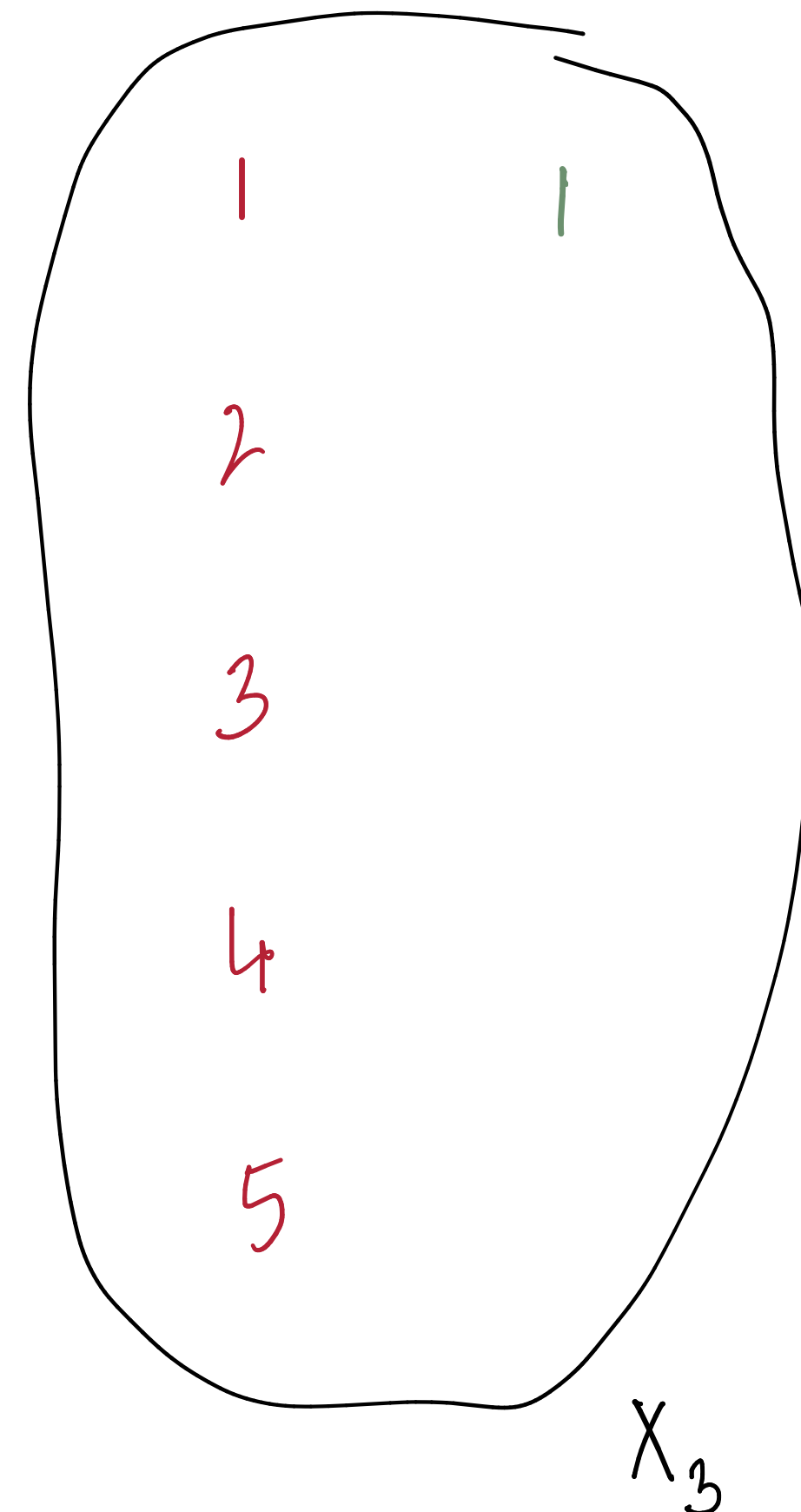
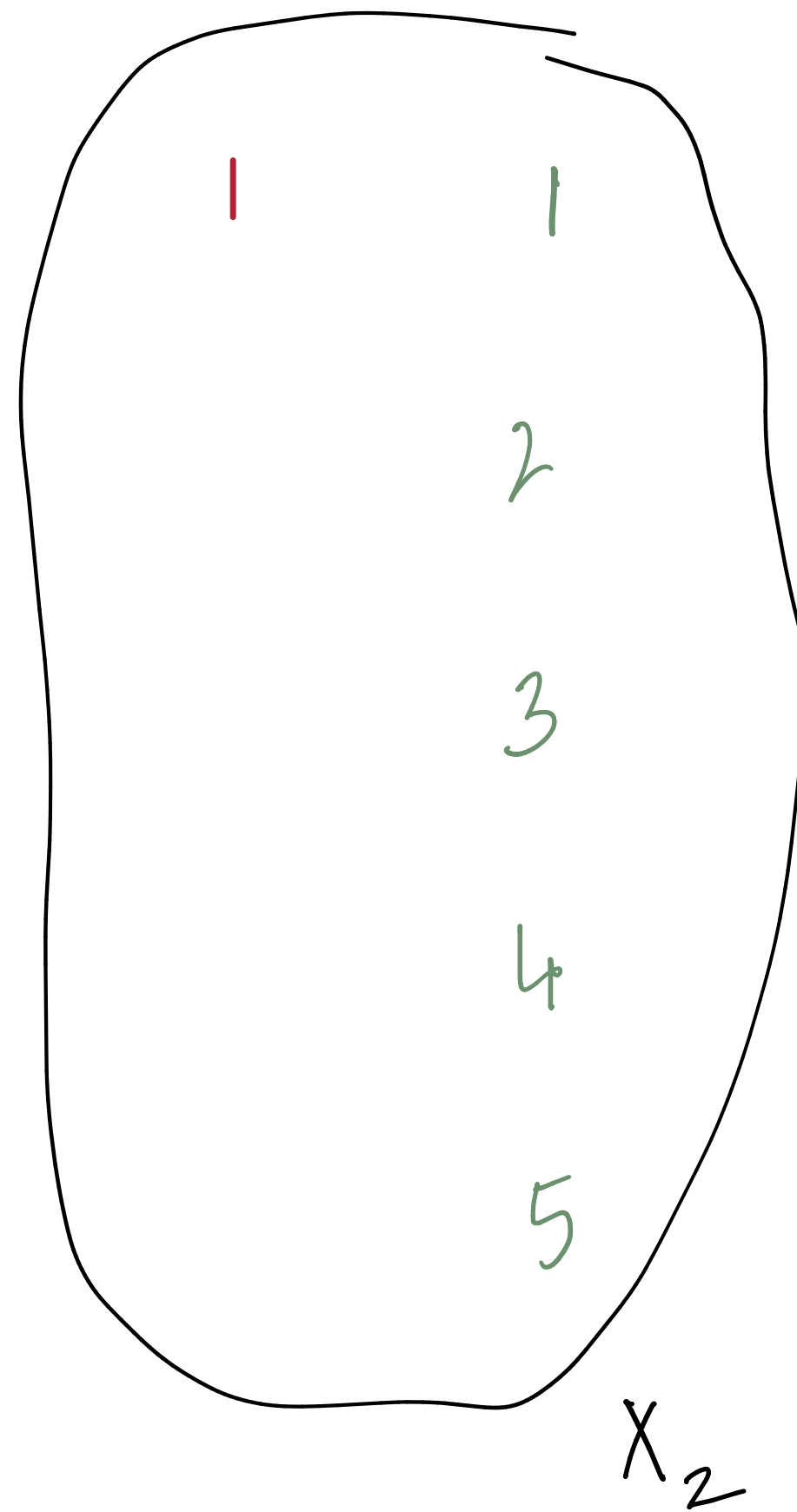
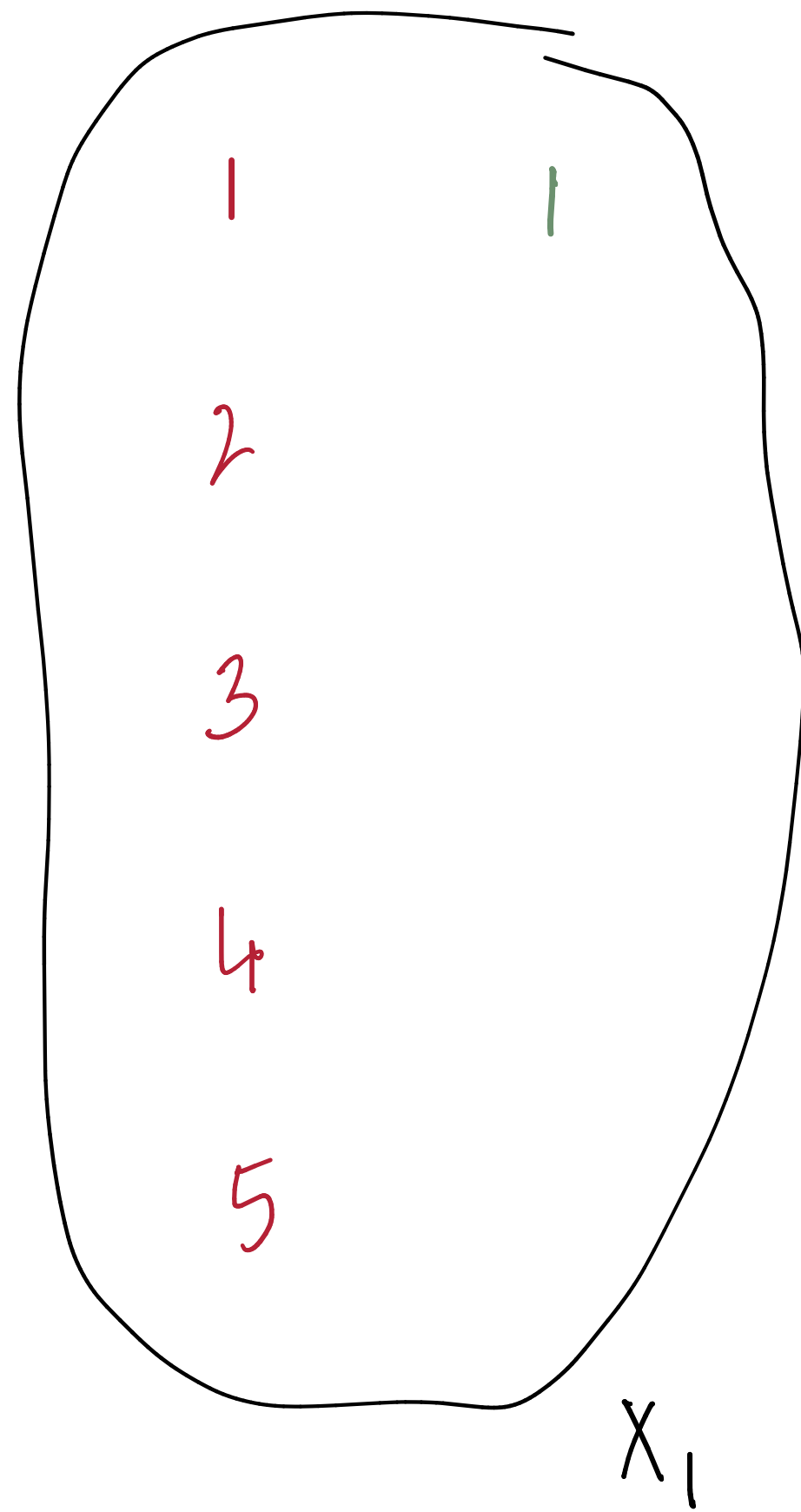
Useful hack: Suppose we want to force that
no cards from one part of our sequence
are carried over to the next part



Useful hack: Suppose we want to force that
no cards from one part of our sequence
are carried over to the next part



Setup so far:



What about the clauses?