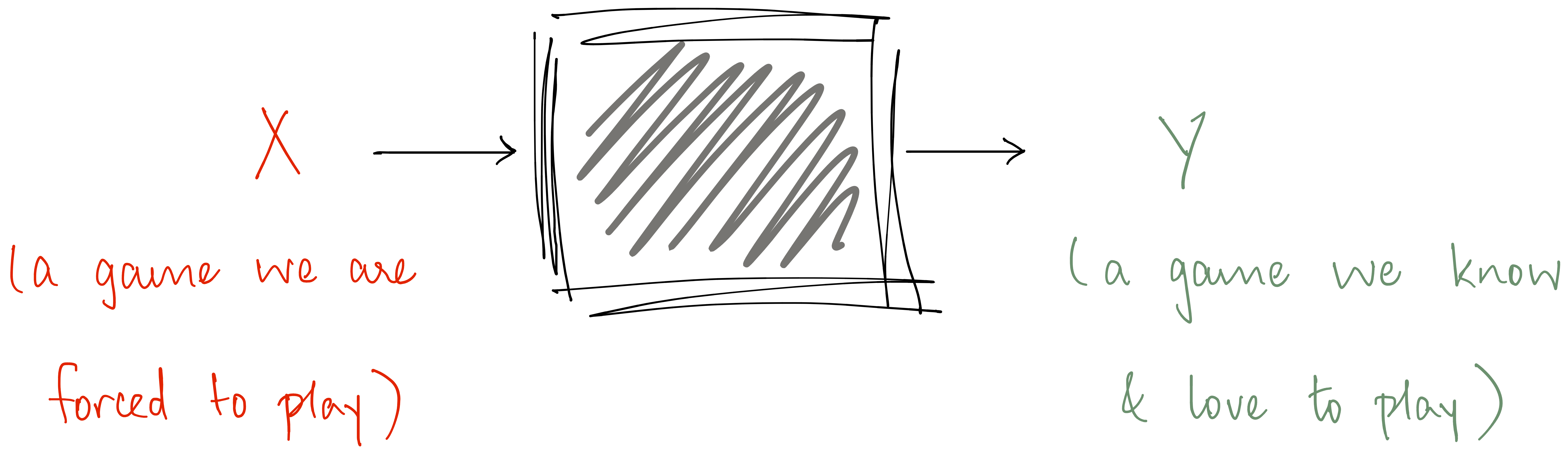
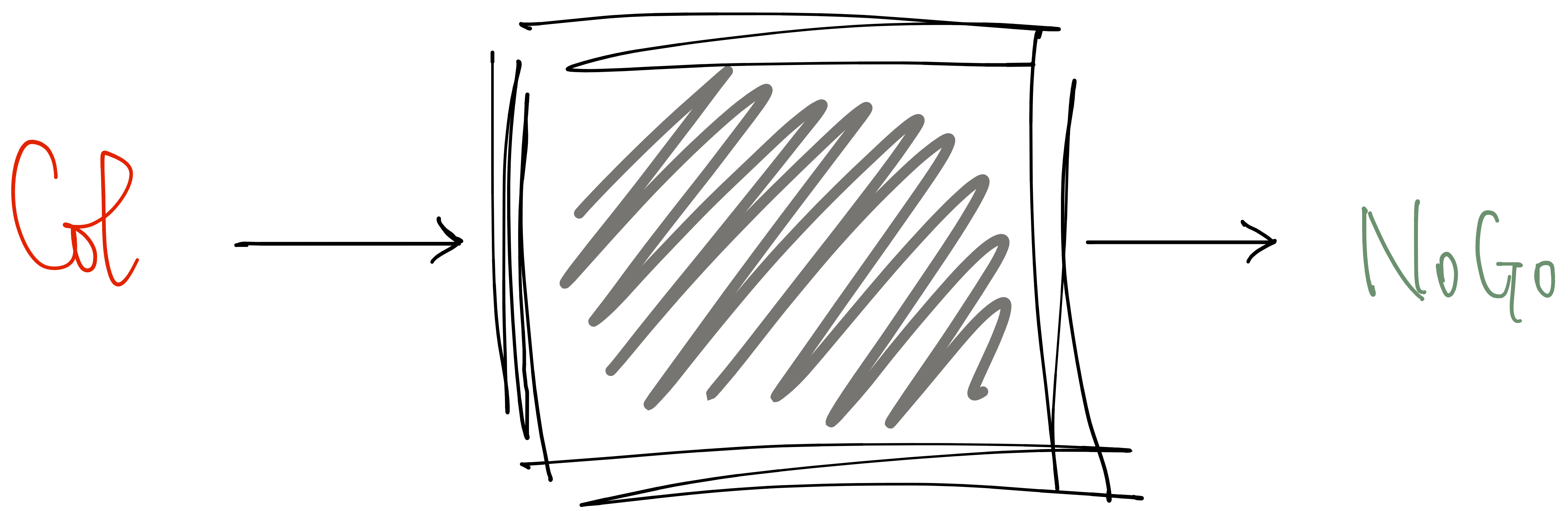
Recap

Transformations



Recap

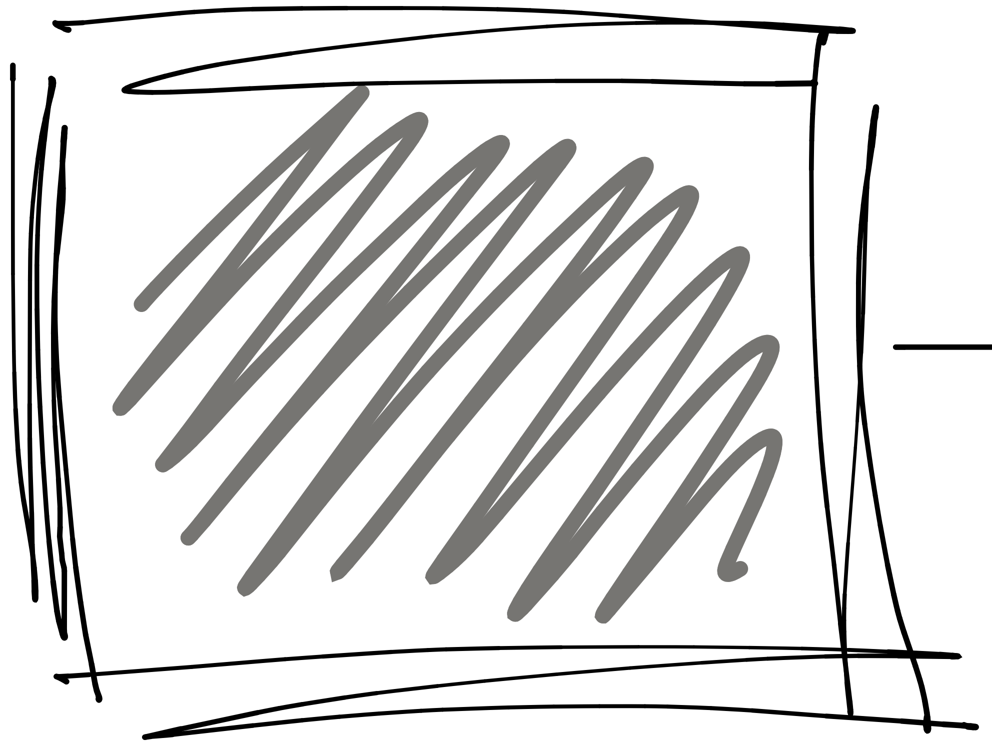
Transformations



Recap

Transformations

Boolean Formula Game $\longrightarrow$



$\longrightarrow$ Directed Geography

Complexity Theory - Detour

(source. Beyond Computation : The P v/s 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

1634733645 809 253848 443 133 883 865 090859841783 670 033092312

181110852389 333 100 104508151 212 118 167 511579

*

1900871281 664822113 126851573935 413 975 471 896 789 968 515 493

666 638 539 088 027 103 802 104 498 957 191 261 465 571

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?

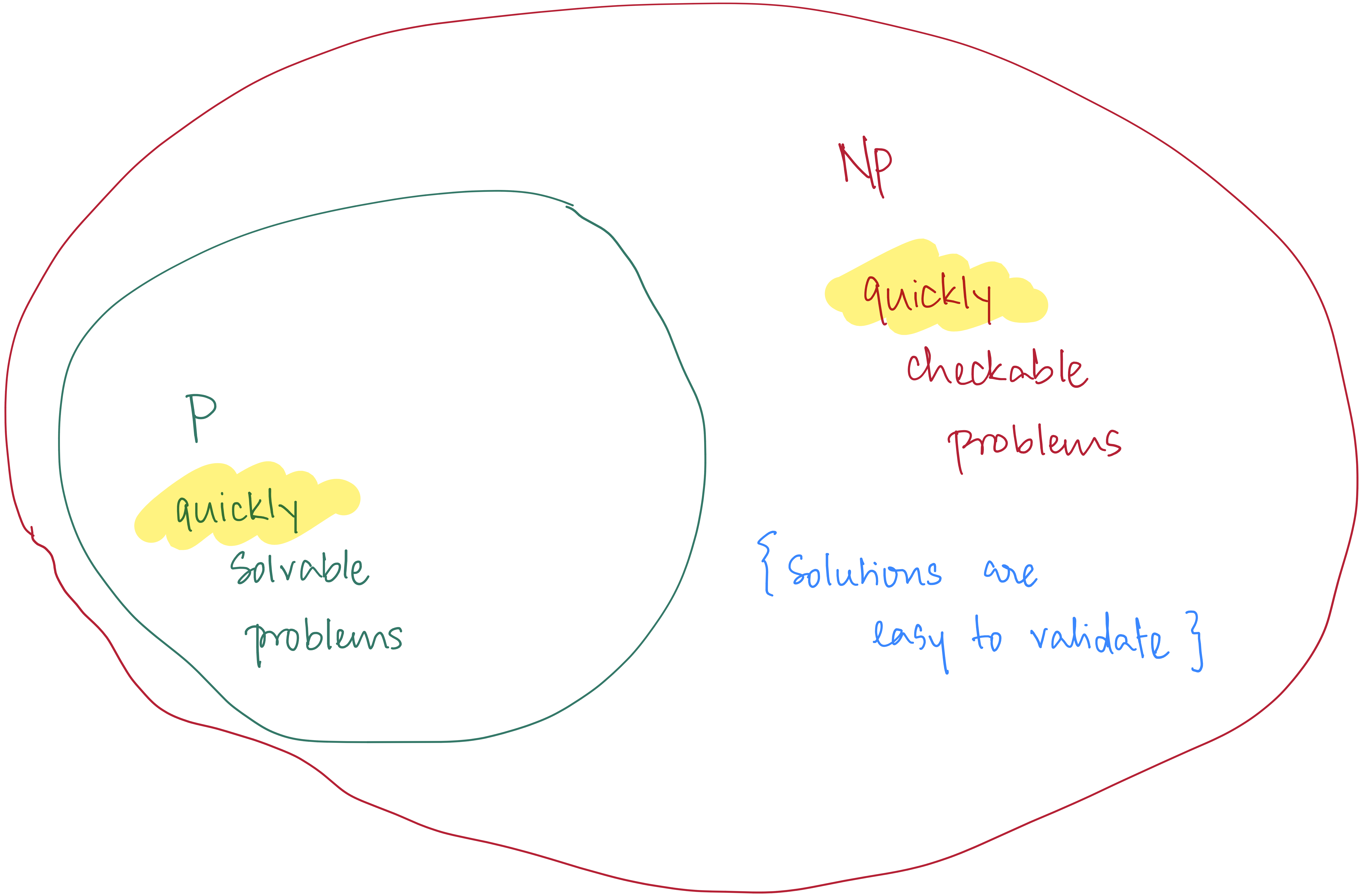
Brute-force search: the possibilities are astronomical

(even with heuristic speed-ups)

Is all this searching really necessary,

or is there a smarter shortcut?

"Needle in a haystack" - type problems


Np
quickly checkable problems
{ Solutions are easy to validate }
P
quickly solvable problems

S'pose 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 $a < p$,

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

Eg. $a = 2,\ p = 7$; $\quad 2^6 = 64 \equiv 1 \bmod 7$

But also: $a = 2,\ p = 4$; $\quad 2^3 = 8 \not\equiv 1 \bmod 4$ (basis for a primality test?)

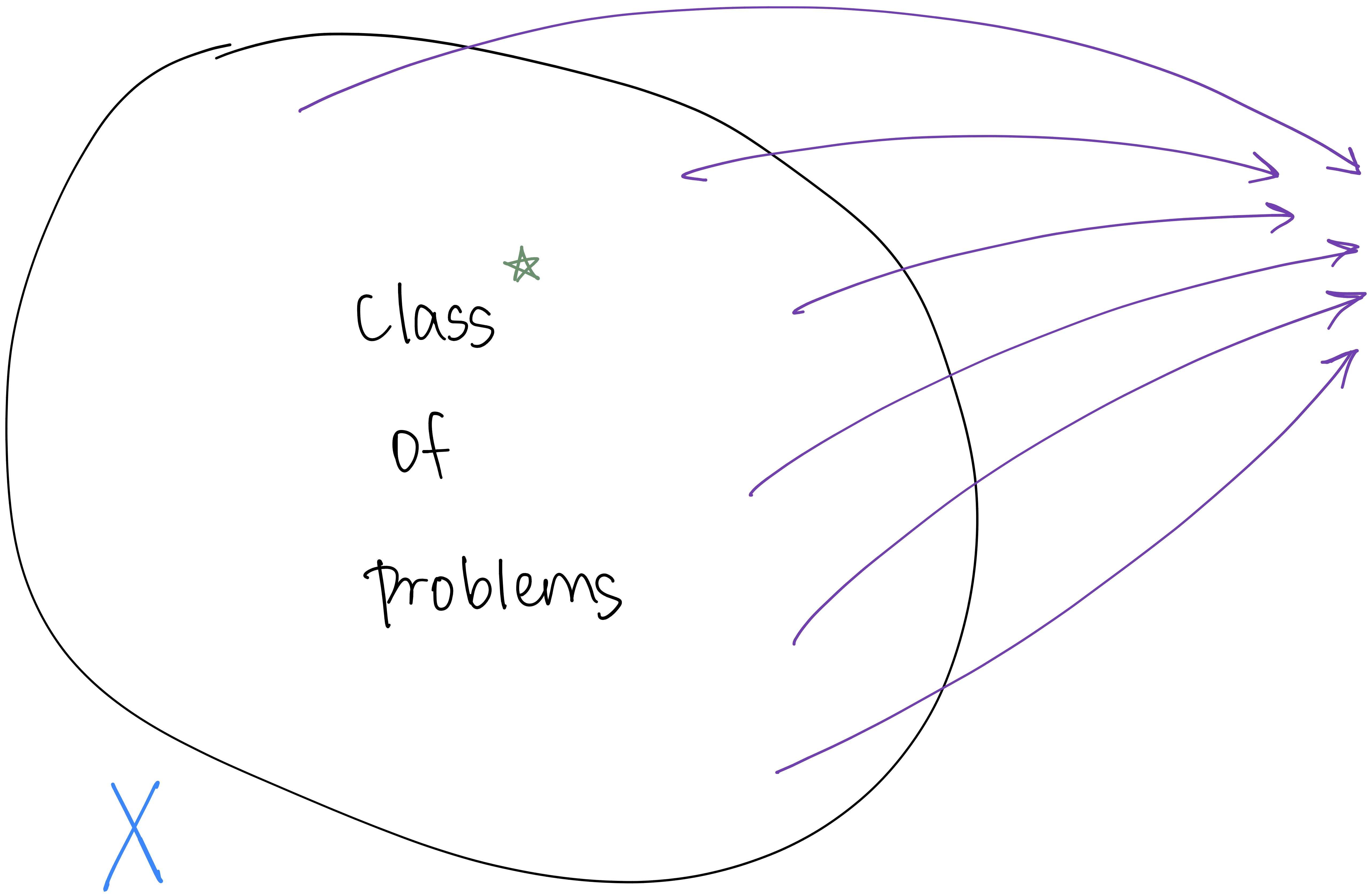
Transformations

$X \to Y$

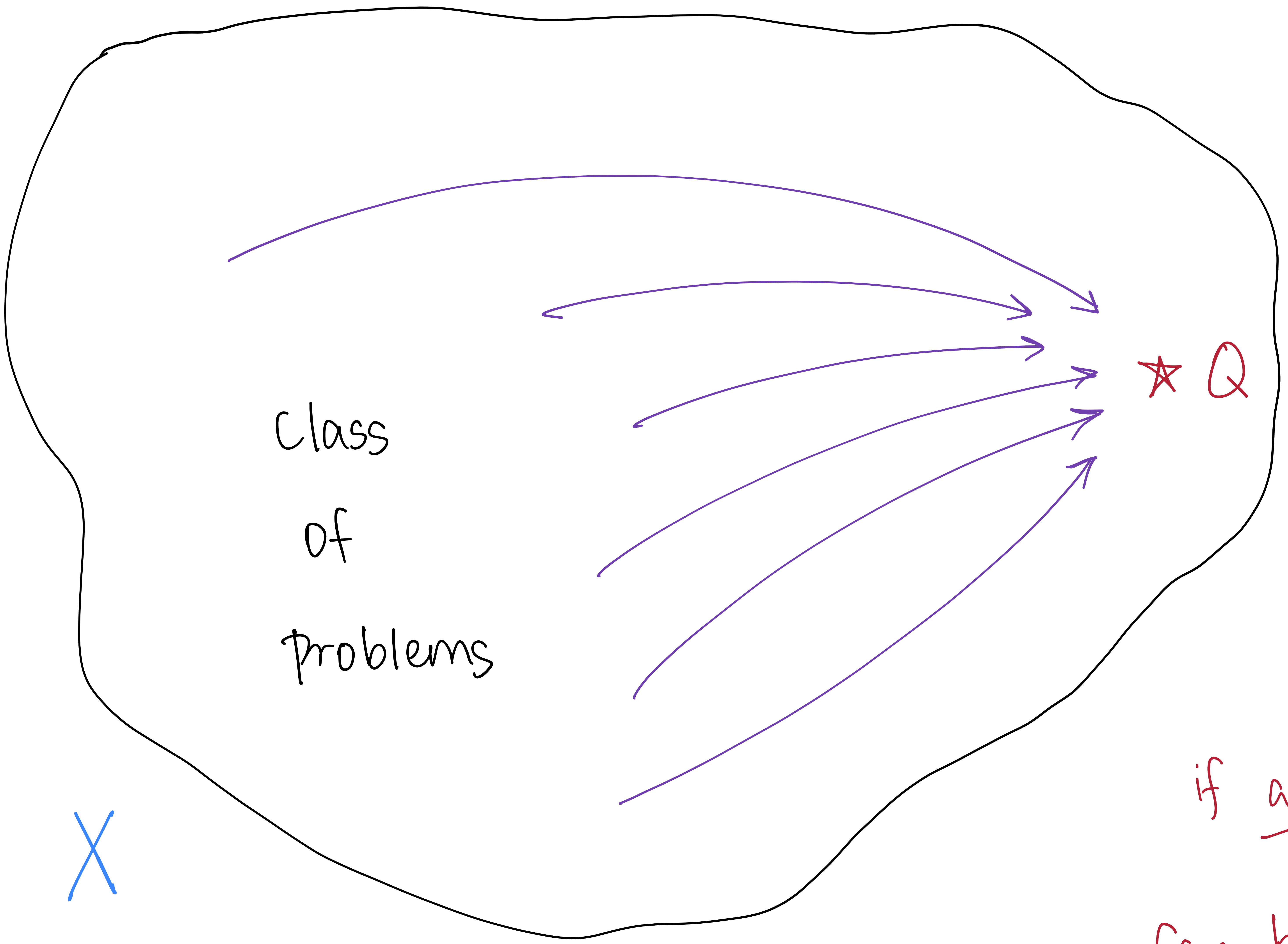
If you can solve X 'quickly'

then you can solve Y 'quickly' too!

★ typically defined in terms of resources required to solve them



★ Q ⇝ a specific problem is said to be X - HARD if *all* problems in X can be transformed to Q.



$\star$ Q $\rightsquigarrow$ a specific problem is said to be X - COMPLETE if all problems in X can be transformed to Q & $Q \in X$.

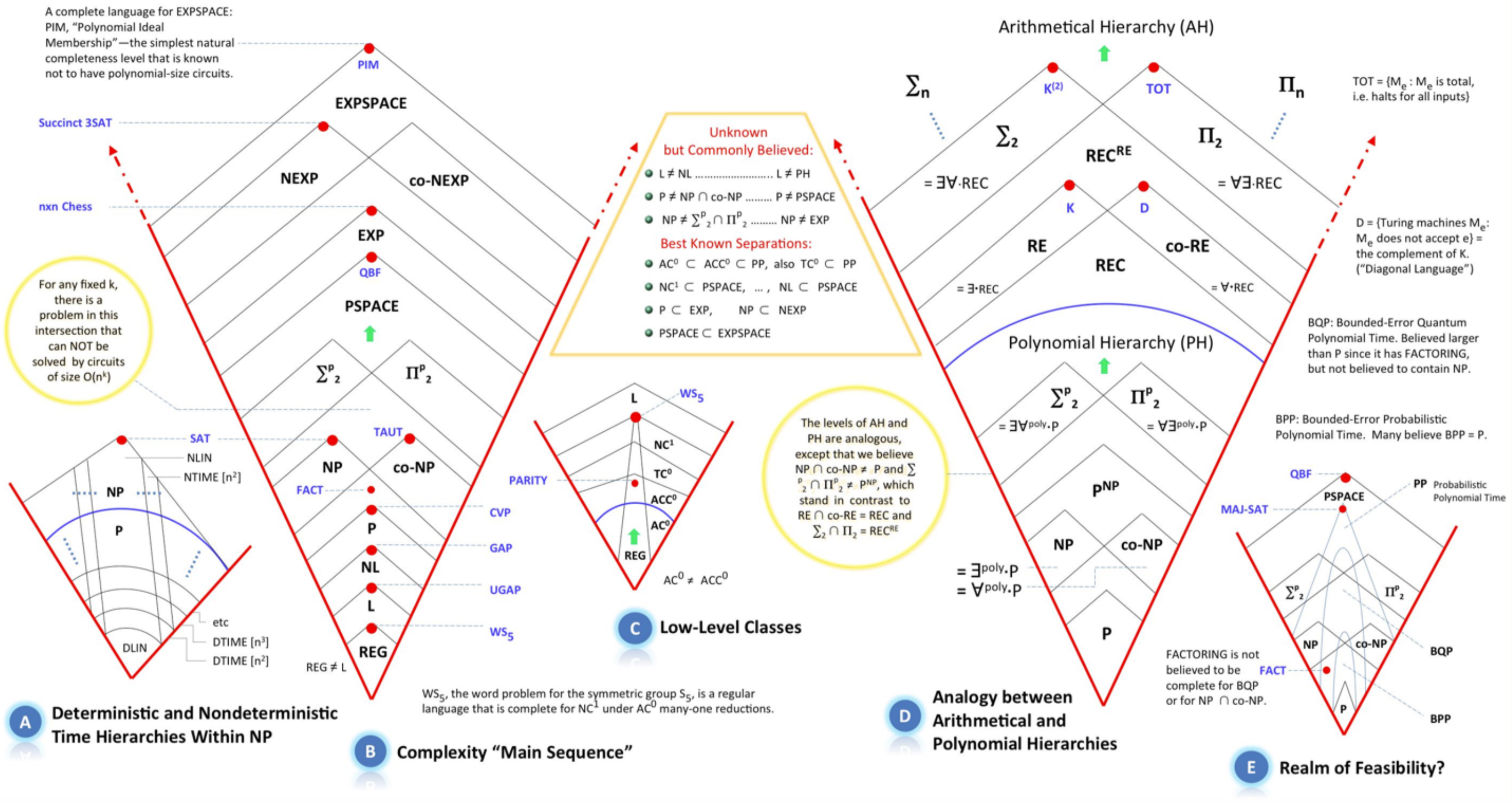
University at Buffalo
The State University of New York

cse@buffalo

LANDSCAPE OF COMPUTATIONAL COMPLEXITY

Spring 2008

State University of New York at Buffalo
Department of Computer Science & Engineering
Mustafa M. Faramawi, MBA   Dr. Kenneth W. Regan



Claim. Hanabi is NP-complete.

Recall.

v = # values

c = # colors

h = storage threshold

r = # 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 $x_1, x_2, \ldots, x_n$

m clauses $\cdots \underbrace{(x_1 \text{ OR } \overline{x_7} \text{ OR } x_8)}_{C_i} \cdots$

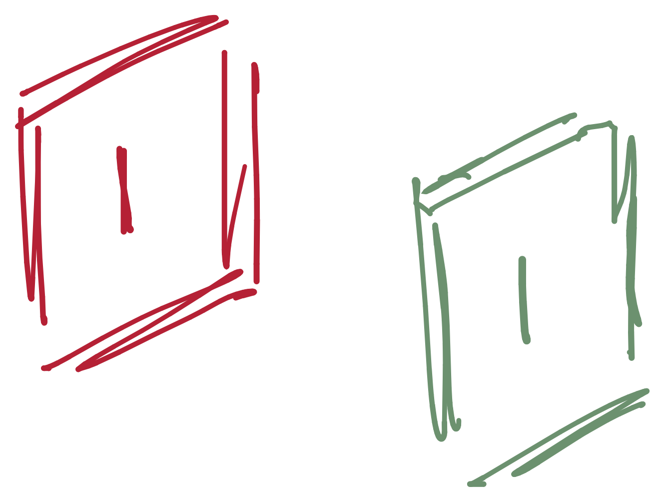
Goal. Π_φ is playable iff φ is satisfiable

seq. of cards Π_φ

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

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

Attempt #1



play this?

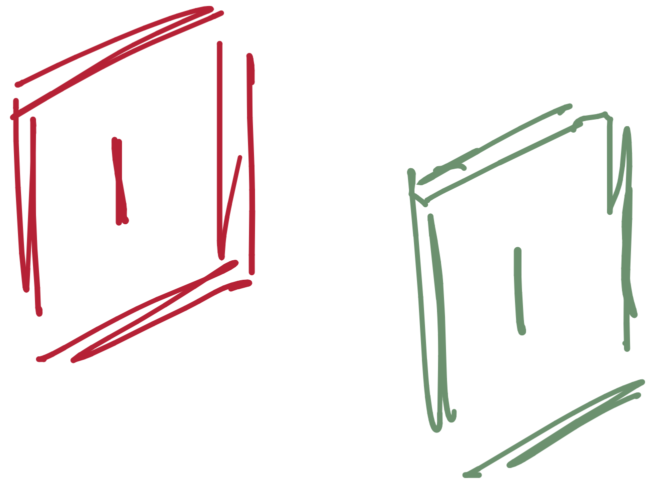
set x_1 to True

play this?

set x_1 to False

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

Attempt #1



play this?

set x_1 to True

play this?

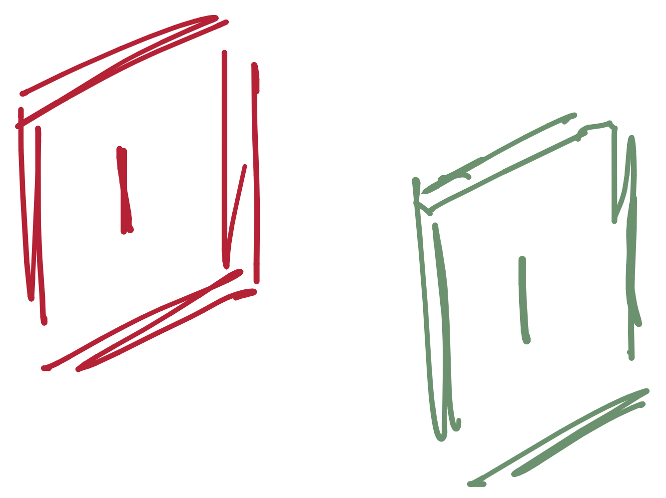
set x_1 to False

Idea:

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

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

Attempt #1



play this?
set x_1 to True

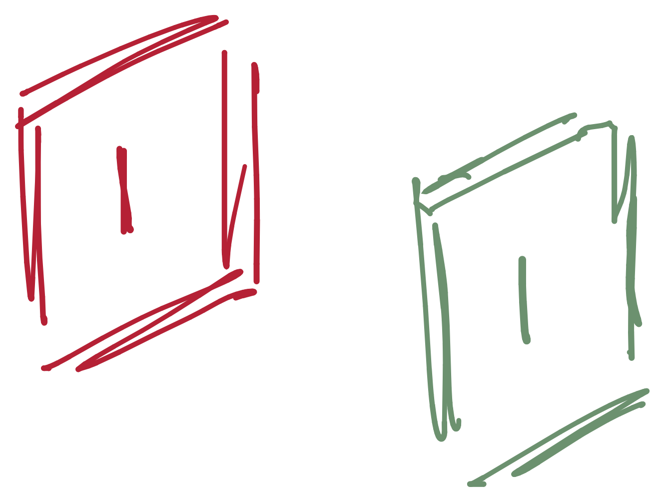
play this?
set x_1 to False

1 2 3 ... n

1 2 3 ... n

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

Attempt #1



play this? set x_1 to True

play this? set x_1 to False

1 2 3 . . . n

1 2 3 . . . n

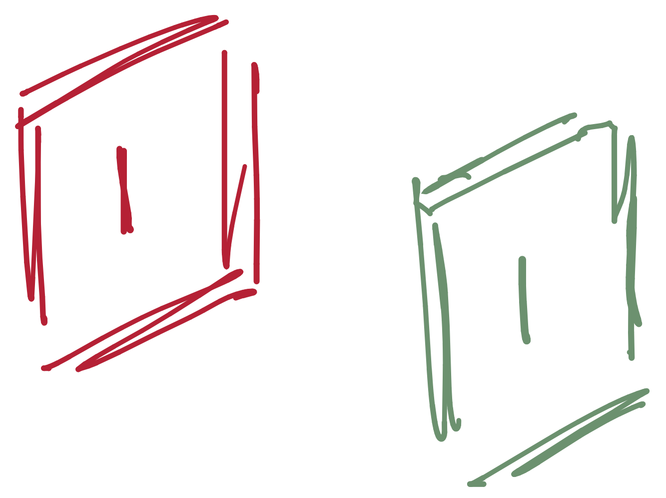
What's the problem with this?

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

Attempt #1

1 2 3 . . . n

1 2 3 . . . n



play this?

set x_1 to True

play this?

set x_1 to False

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

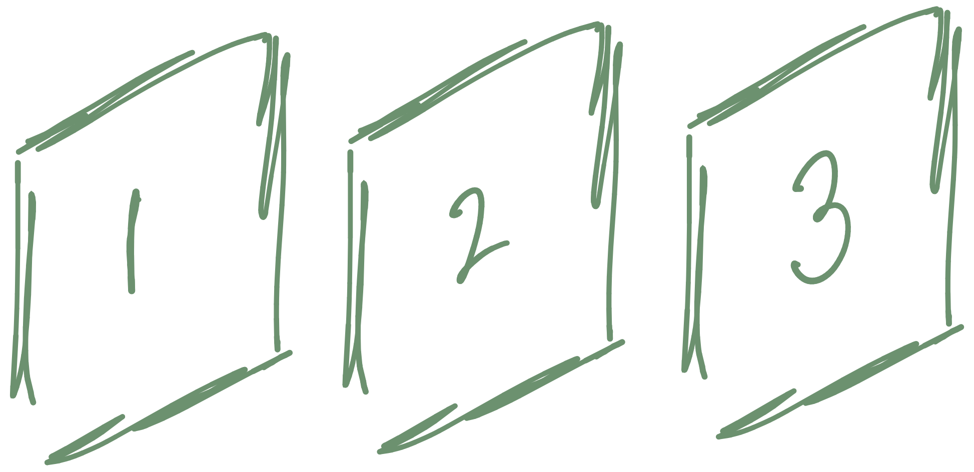
Can do this OR that
but NOT both
and NOT neither.

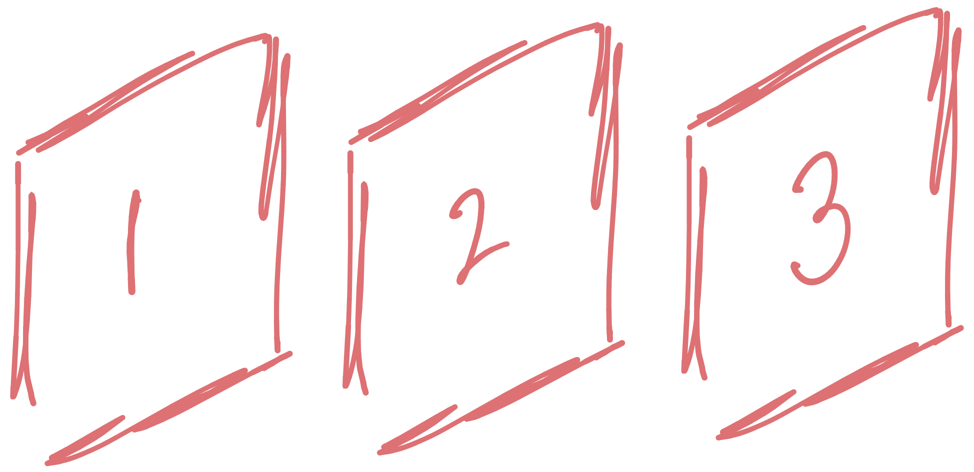
Warm-up $\rightsquigarrow$ how do we get a play sequence to correspond to an assignment ?

Attempt #2

Associate a different color with every variable

$x_1 \rightsquigarrow$ colors are RED_1 and GREEN_1



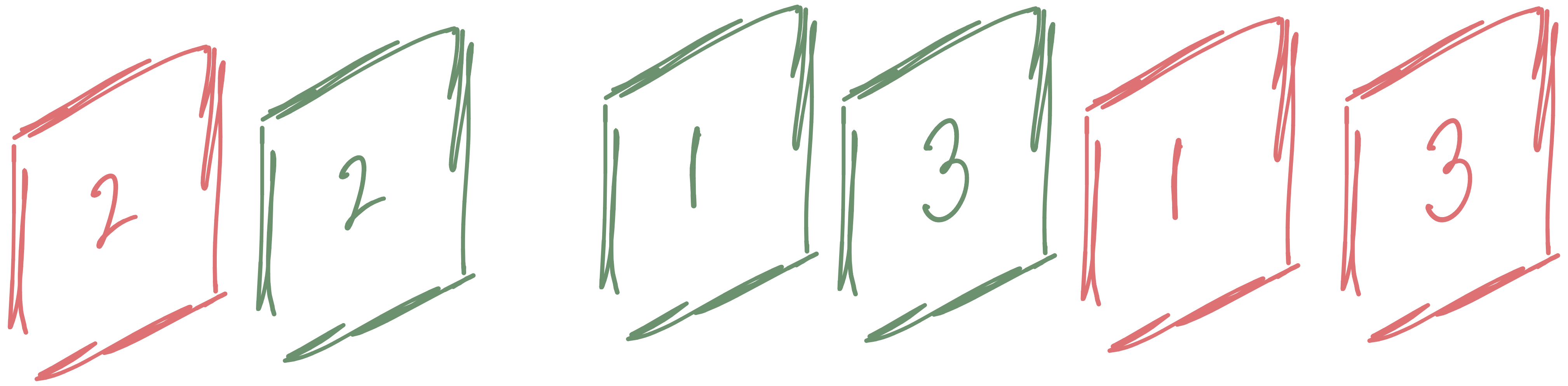


Warm-up $\rightsquigarrow$ how do we get a play sequence to correspond to an assignment?

Attempt #2

Associate a different color with every variable

$x_1 \rightsquigarrow$ colors are RED_1 and GREEN_1

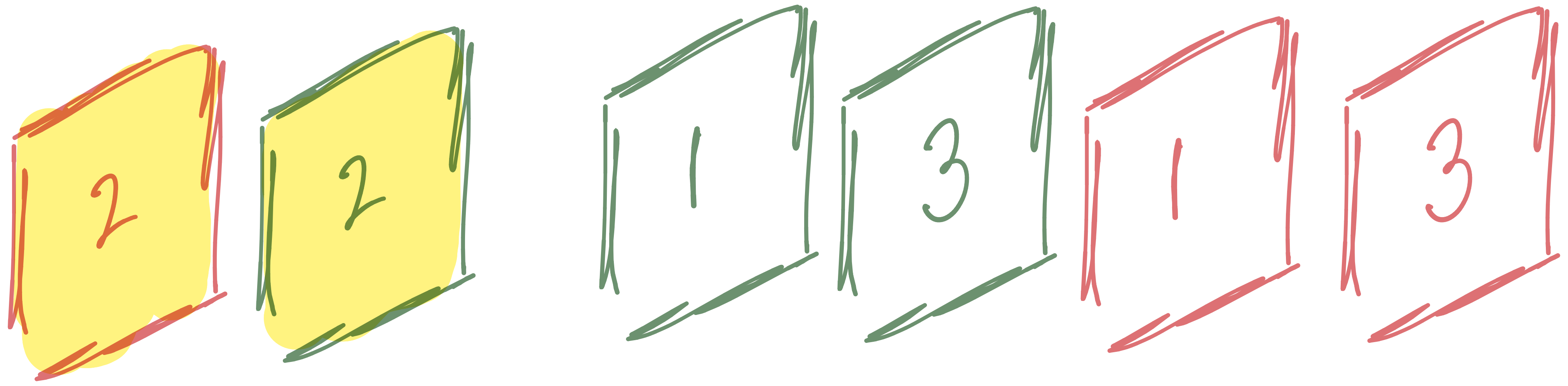


Warm-up $\rightsquigarrow$ how do we get a play sequence to correspond to an assignment?

Attempt #2

Associate a different color with every variable

$x_1 \rightsquigarrow$ colors are RED_1 and GREEN_1



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 RED_1 and $GREEN_1$

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 RED_1 and GREEN_1

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

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

Warm-up 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 $n=1$ 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

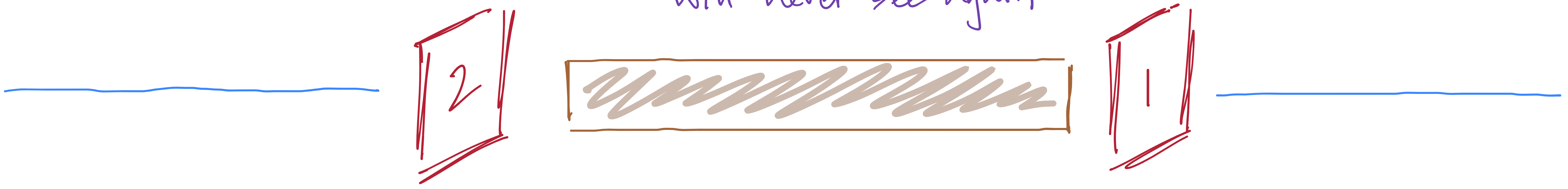
(for whatever reason)

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

(for whatever reason)

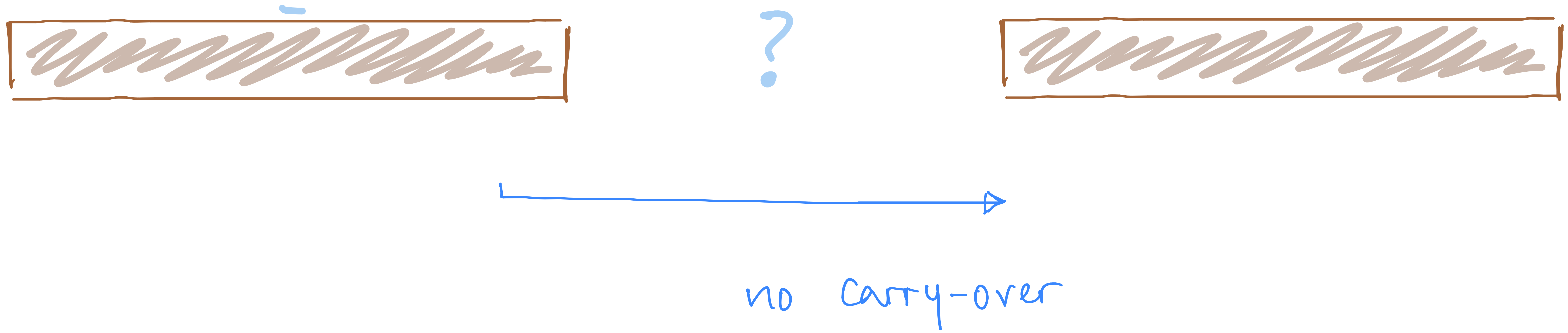
how do we do this?

forced hold via an exotic color that we will never see again

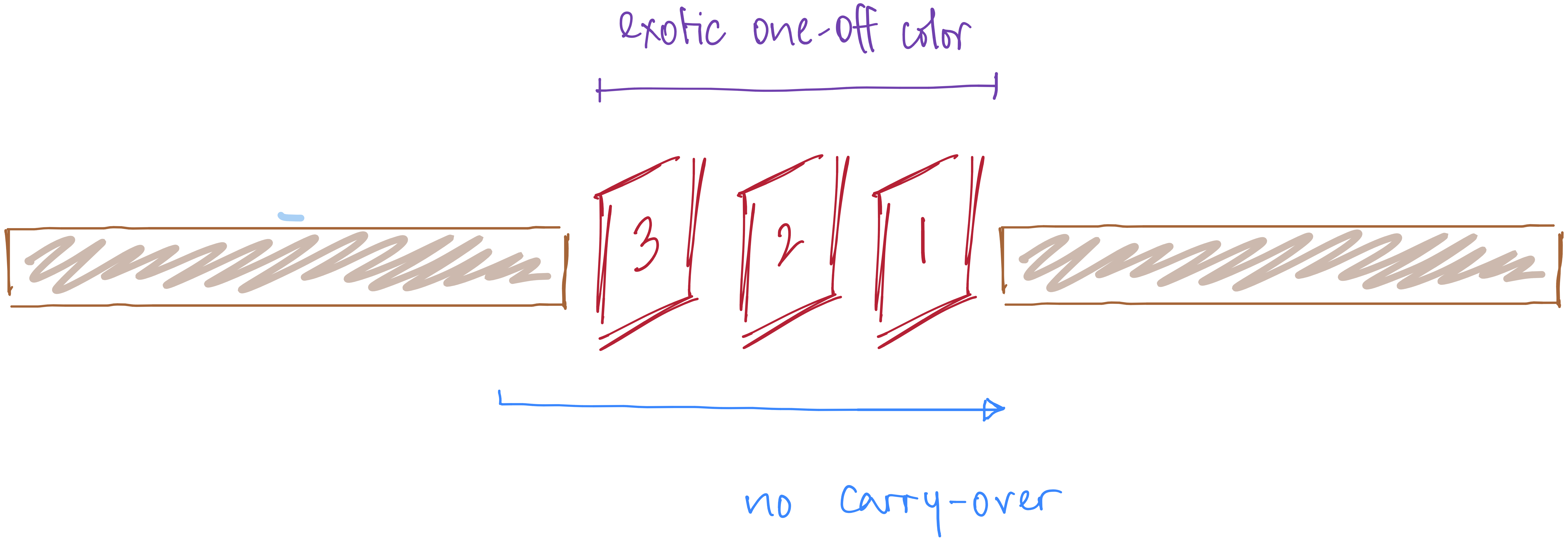


Useful Hack: Suppose we want to force that no cards from one part of our sequence are carried over to the next part

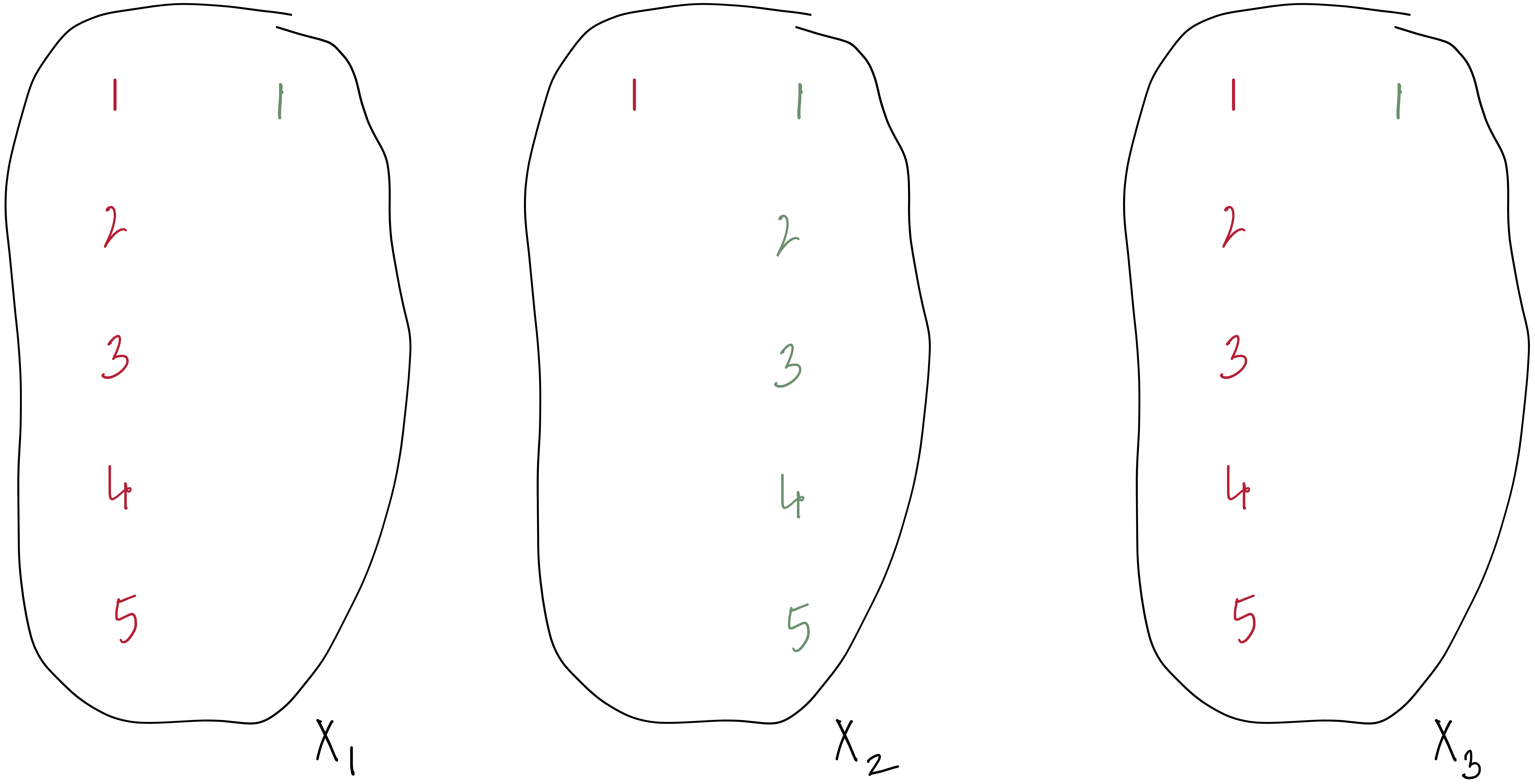
Useful Hack: Suppose we want to force that no cards from one part of our sequence are carried over to the next part



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?