

2024.10.8 5<sup>th</sup> Combinatorial Reconfiguration Workshop @ Fukuoka, Japan

# On the Complexity of Approximating Reconfiguration Problems

## Naoto Ohsaka

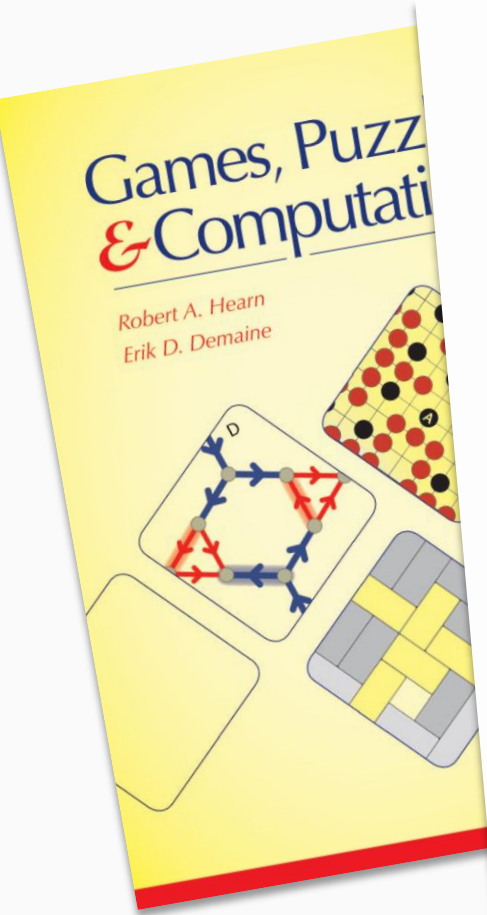
(CyberAgent Inc., Japan)

Joint work with **Shuichi Hirahara** (NII)



# Prelude

# When I started studying reconfiguration ...



## The complexity of change

Jan van den Heuvel

**Abstract**

Many combinatorial problems can be formulated as “Can I transform configuration 1 into configuration 2, if only certain transformations are allowed?”. An example of such a question is: given two  $k$ -colourings of a graph, can I transform the first  $k$ -colouring into the second one, by recolouring one vertex at a time, and always maintaining a proper  $k$ -colouring? Another example is: given two solutions of a SAT-instance, can I transform the first solution into the second one, by changing the truth value one variable at a time, and always maintaining a solution of the SAT-instance? Other examples can be found in many classical puzzles, such as the 15-Puzzle and Rubik’s Cube.

In this survey we shall give an overview of some older and some more recent work on this type of problem. The emphasis will be on the computational complexity of the problems: how hard is it to decide if a certain transformation is possible or not?

## 1 Introduction

Reconfiguration problems are combinatorial problems in which given a collection of configurations, together with some transition rule(s) that allows us to change one configuration to another, we want to know if there is a sequence of configurations that starts at a given configuration and ends at a target configuration. For example, in the so-called 15-puzzle (see Figure 1): 15 tiles are arranged in a  $4 \times 4$  grid, with one empty square; neighbouring tiles can be swapped. The normal aim is, given an initial configuration of the tiles to the position with all numbers in order (right-hand side of Figure 1). Readers of a certain age may remember Rubik’s cube, which is another example of reconfiguration problems.

More abstract kinds of reconfiguration problems abound in graph theory. For instance, suppose we are given a planar graph and two colourings of that graph. Is it possible to transform the first 4-colouring into the second one, by recolouring one vertex at a time, and never using a colour that is not in the set of colours? Taking any two different 4-colourings of the complete graph  $K_4$  shows that the answer is not always yes. But what would be the answer if we restrict to planar graphs? And whereas it is easy to see what is allowed, it is not always clear how hard it is to decide if a given 4-colouring of some planar graph can be transformed into another given 4-colouring of some planar graph by recolouring one vertex at a time?



## Introduction to Reconfiguration

Naomi Nishimura  
David R. Cheriton School of Computer Science, University of Waterloo, 200 University Avenue West, Waterloo, ON N2L 2G1, Canada; nishi@uwaterloo.ca; TEL: +1-519-888-4567

Received: 13 September 2017; accepted: 17 April 2018; Published: 19 April 2018

**Abstract** Reconfiguration is concerned with relationships among solutions to a problem where the reconfiguration of one solution to another is a sequence of steps such that each step produces an intermediate feasible solution. The solution space can be represented as a graph, where two vertices representing solutions are adjacent if one can be formed from the other by a single step. Work in the area encompasses both structural questions (Is the reconfiguration space connected?) and algorithmic ones (How can one find the shortest sequence of steps to reconfigure one solution to another?) This survey discusses techniques, results, and future directions in the area.

**Keywords:** reconfiguration problems; algorithms; complexity

## 1. Introduction

Solving puzzles, planning robot motion, and editing between strings can be viewed as reconfiguration problems. When configurations are defined as feasible solutions to a problem, the location of a robot with respect to obstacles in space, or the position of a string. When configurations are defined as feasible solutions to a problem, the location of a robot with respect to obstacles in space, or the position of a string. When configurations are defined as feasible solutions to a problem, the location of a robot with respect to obstacles in space, or the position of a string.

Reconfiguration arises in countless problems that involve movement in a space. Work in the area and the types of problems and approaches considered is distinct from other approaches to the solution space of problems, such as local search, to an instance, find a better solution that is close to the input solution (given an instance, an optimal solution, and changes to the instance, find the changed instance), and incremental problems [1] (given a yes-instance, and changes to the instance, determine if the changed instance is a yes-instance).

Algorithms 2018, 11, 52; doi:10.3390/a1104052

😊 Many nice papers & surveys are available

Prelude

# What I was interested in

On the complexity of reconfiguration problems [ISAAC 2008 & Theor. Comput. Sci. 2011]

Takehiro Ito<sup>a,\*</sup>, Erik D. Demaine<sup>b</sup>, Nicholas J.A. Harvey<sup>c</sup>, Christos H. Papadimitriou<sup>d</sup>,  
Martha Sideri<sup>e</sup>, Ryuhei Uehara<sup>f</sup>, Yushi Uno<sup>g</sup>

## 5. Open problems

There are many open problems raised by this work, and we mention some of these below:

- Can the MATCHING RECONFIGURATION problem for edge-weighted graphs be solved also in polynomial time? We conjecture that the answer is positive.
  - Is the TRAVELING SALESMAN RECONFIGURATION problem (where two tours are adjacent if they differ in two edges) PSPACE-complete?
  - Are there better approximation algorithms for the MINMAX POWER SUPPLY RECONFIGURATION problem? Lower bounds?
- Are the problems in Section 4 PSPACE-hard to approximate (not just NP-hard)?

Theme of this talk

This open problem has been **resolved**

# Outline of this talk

Part I

What is meant by "approximation"

Part II

Complexity of approximating reconf. problems

Part III

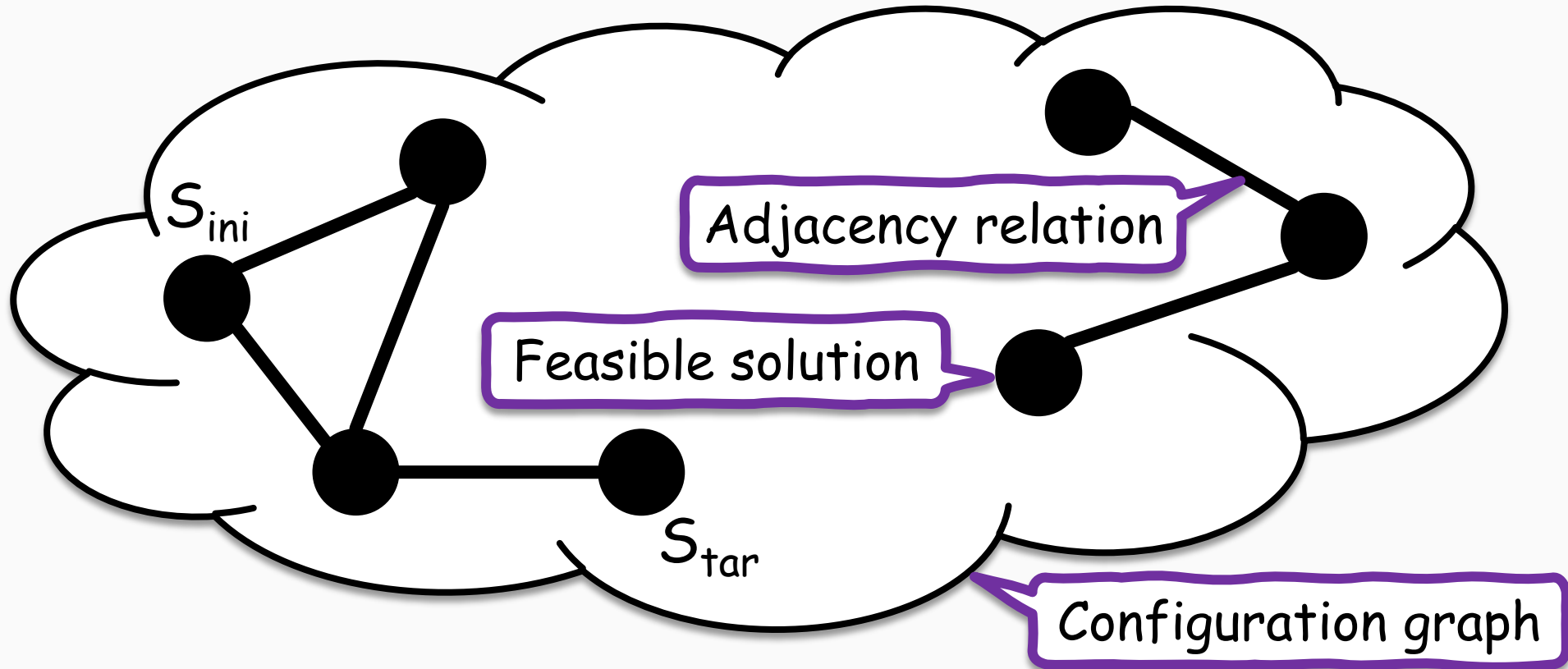
Recent progress, **Struggles** & future directions



Recap

# Exact reconfiguration – a decision problem

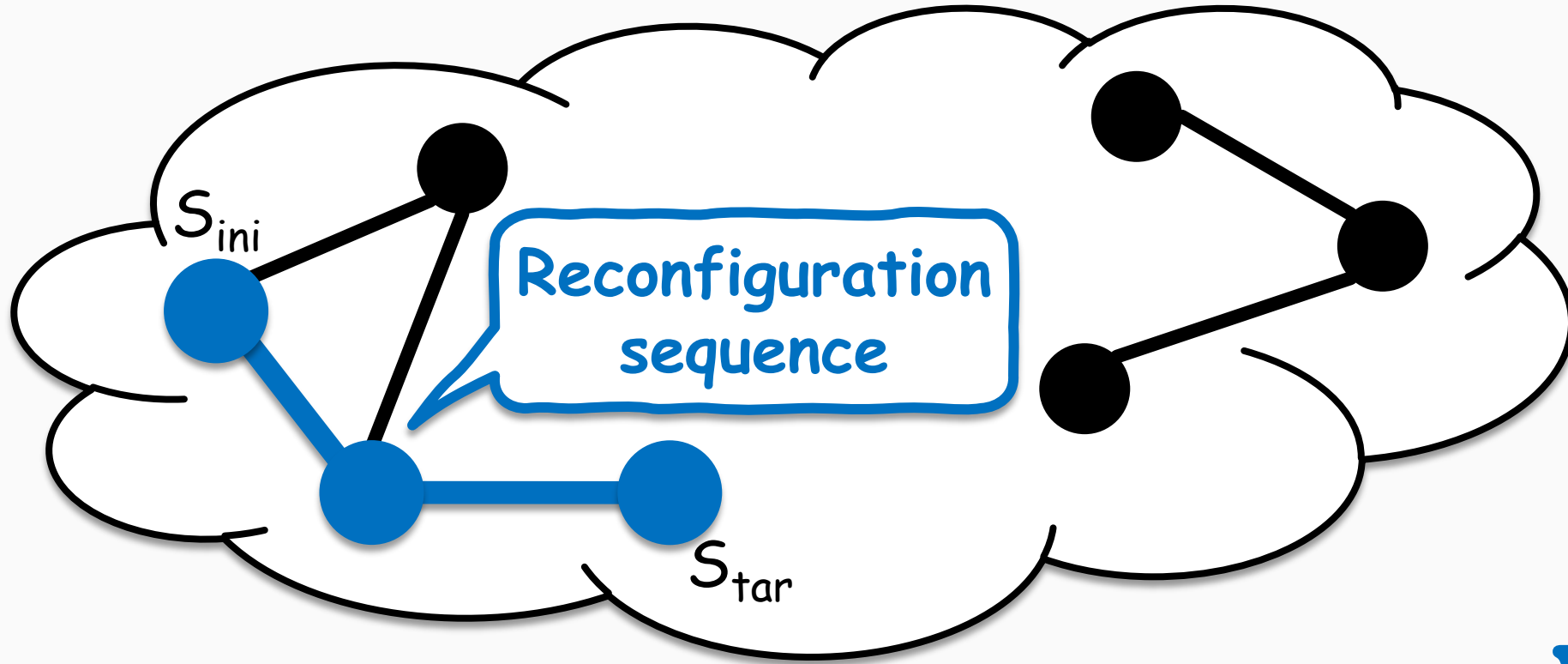
Q. Is a pair of feasible solutions reachable to each other?



Recap

# Exact reconfiguration – a decision problem

Q. Is a pair of feasible solutions reachable to each other?

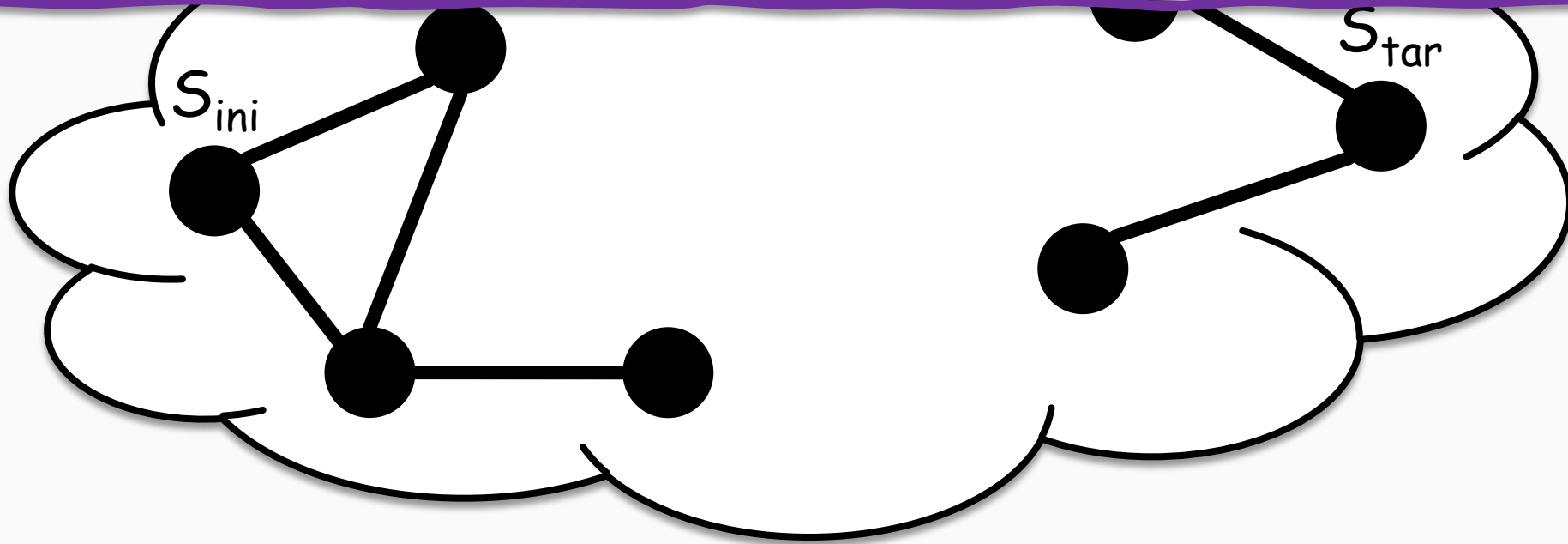


**YES**  
6

Recap

# Exact reconfiguration – a decision problem

- 🎯 Something that “looks” like a reconf. sequence even if
1. we are given a **NO** instance, or
  2. we are dealing with intractable reconf. problems



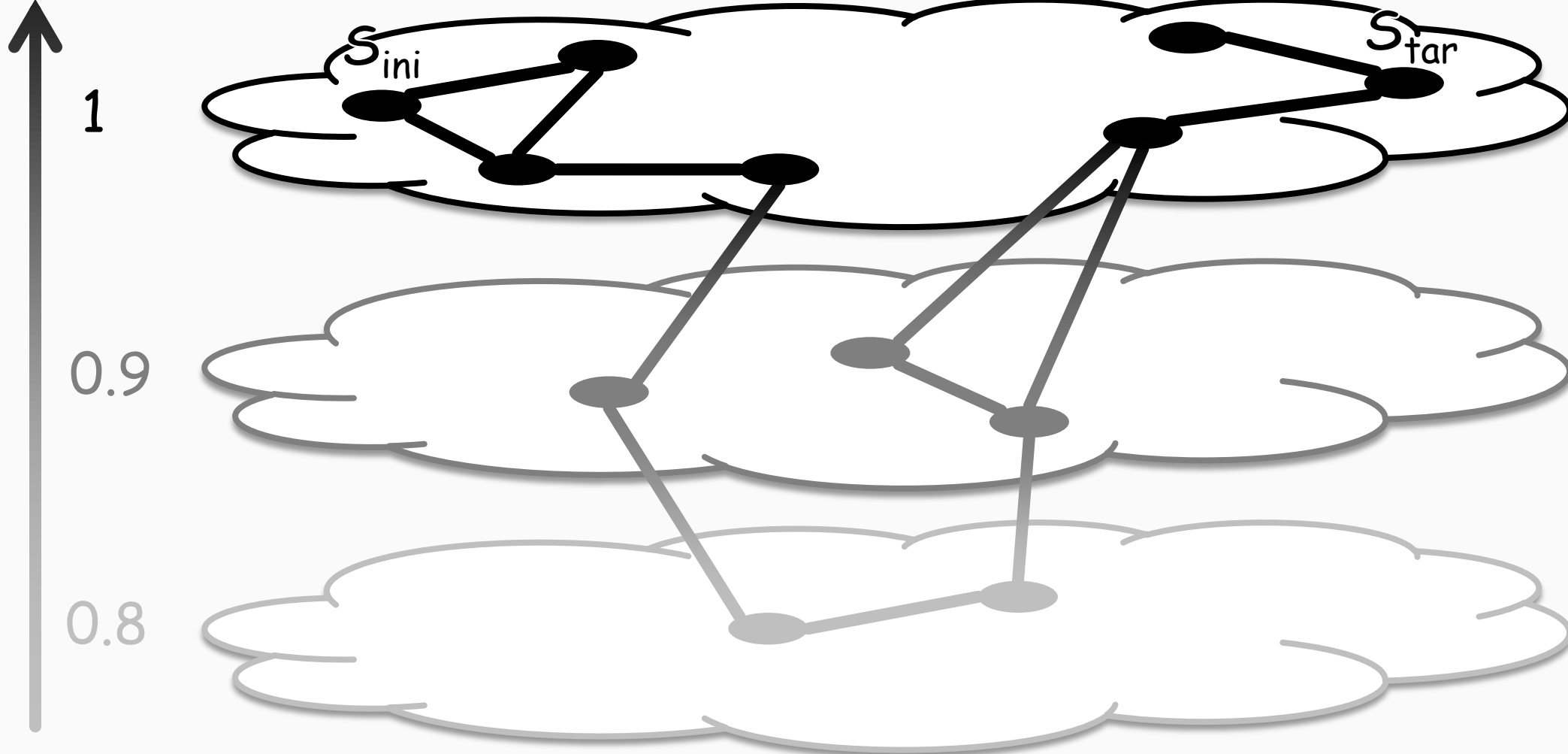
**NO**

Our focus

# Approximate reconf. – an optimization problem

Q. To what extent should the feasibility be relaxed?

feasibility



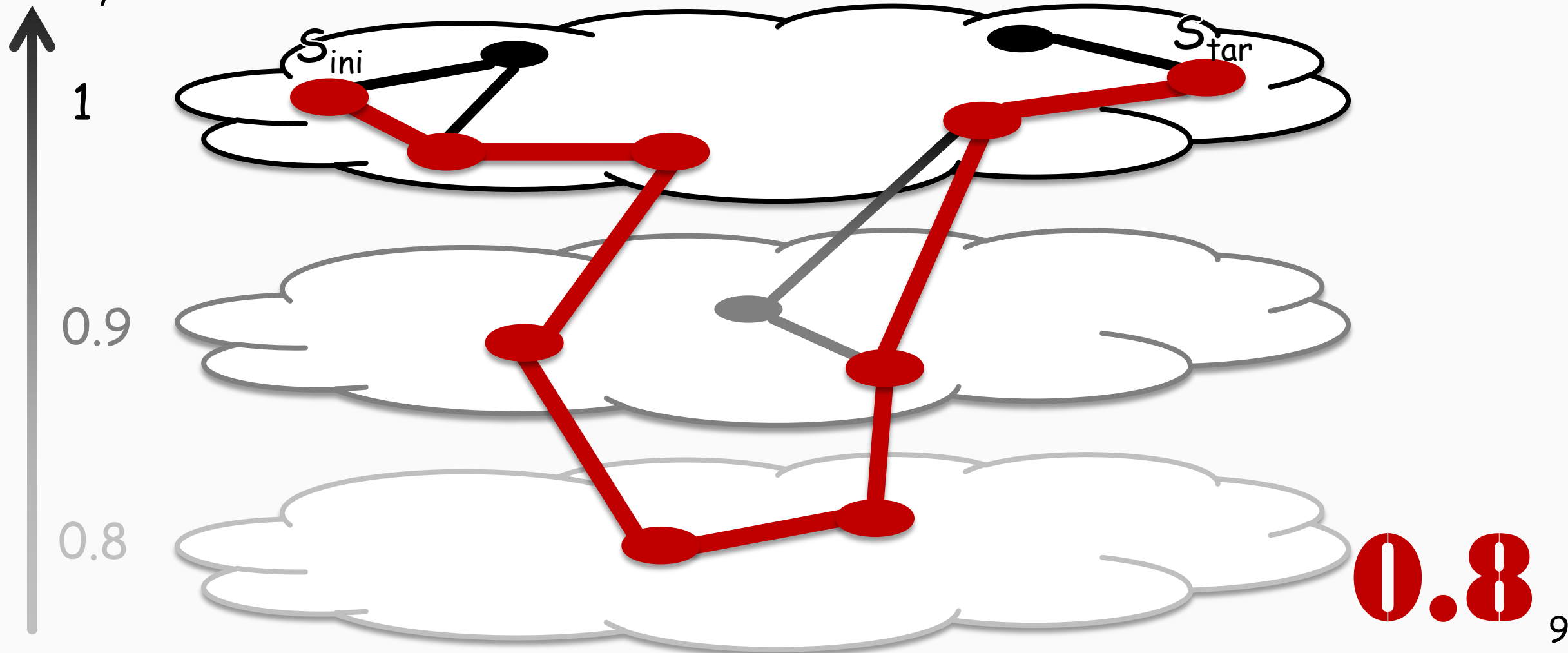


Our focus

# Approximate reconf. – an optimization problem

Q. To what extent should the feasibility be relaxed?

feasibility

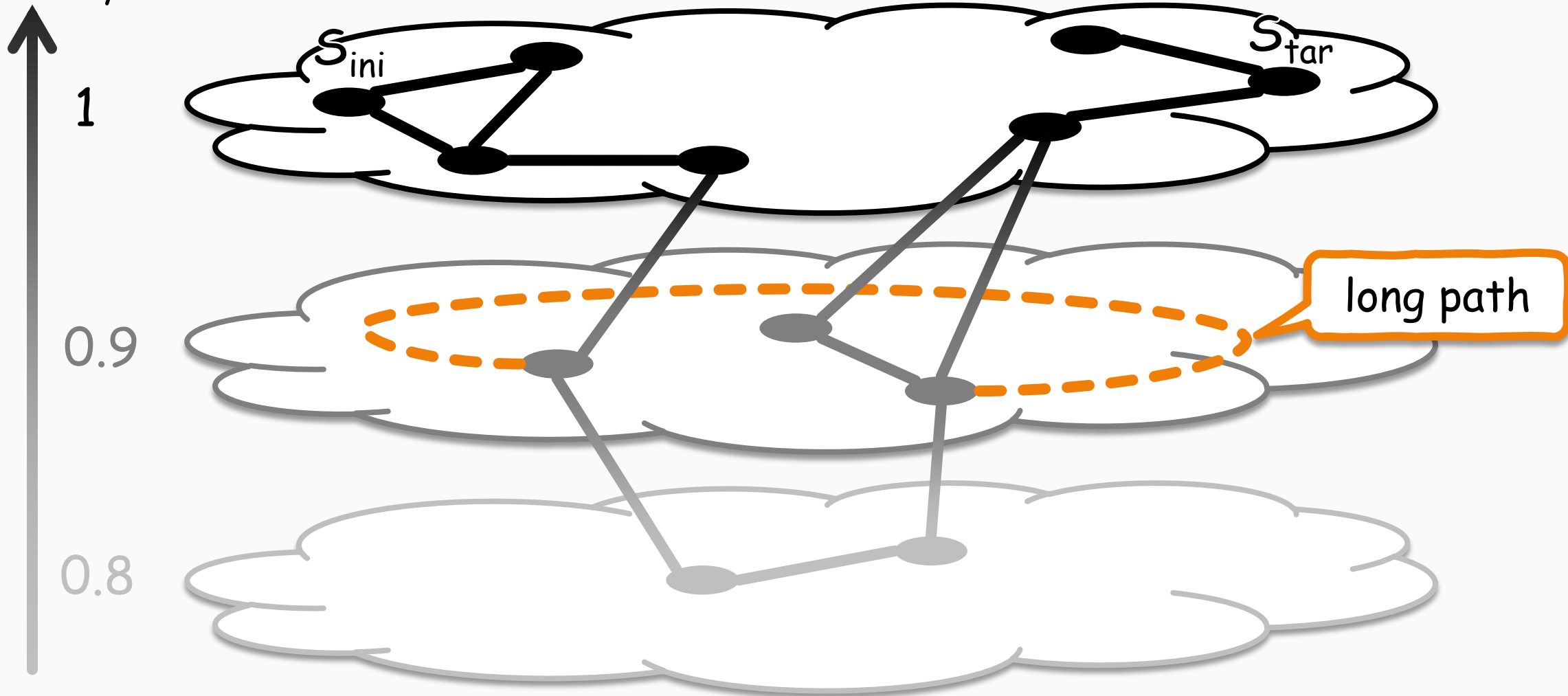


Our focus

# Approximate reconf. – an optimization problem

Q. To what extent should the feasibility be relaxed?

feasibility

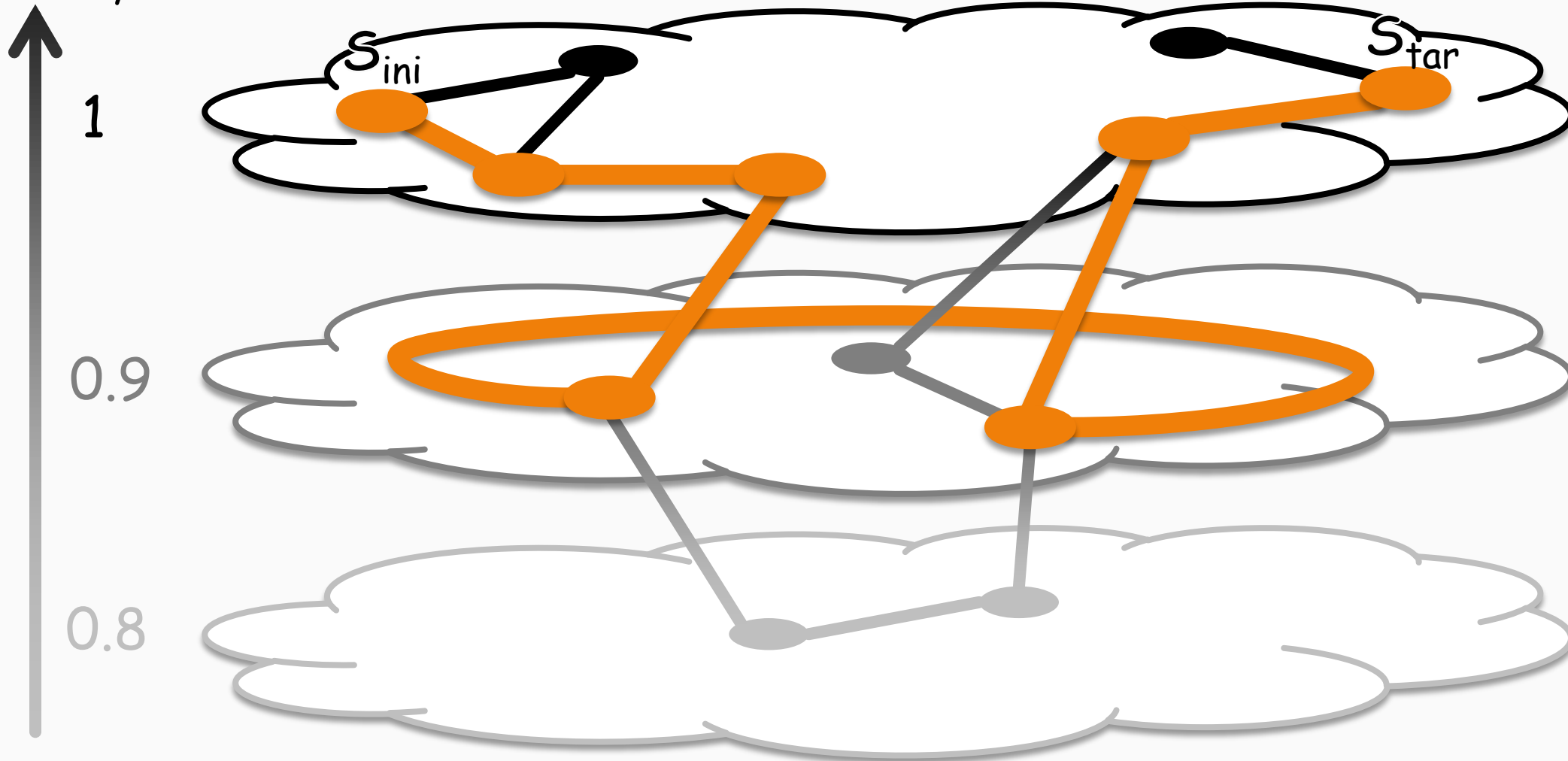


Our focus

# Approximate reconf. – an optimization problem

Q. To what extent should the feasibility be relaxed?

feasibility



0.9

## Example 1

# 3-SAT Reconfiguration

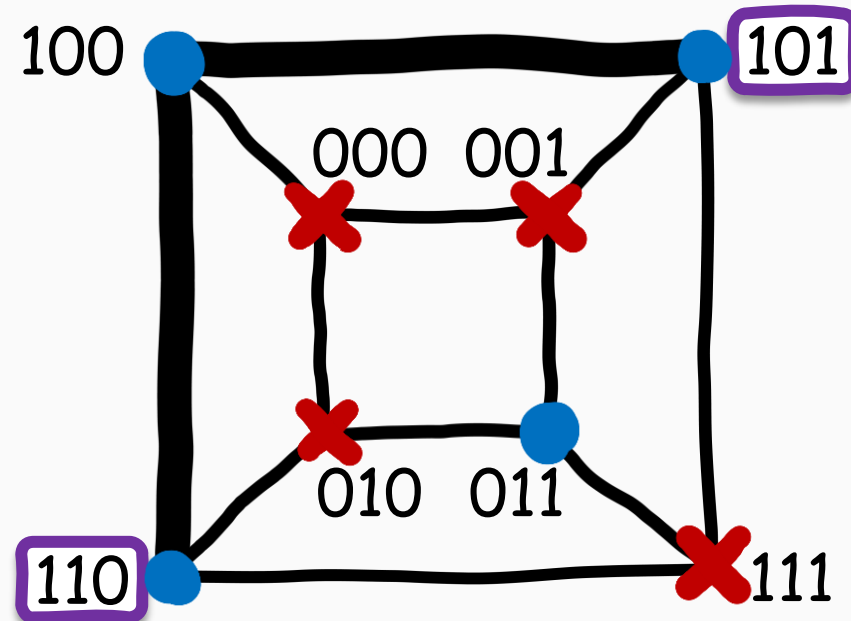
[Gopalan-Kolaitis-Maneva-Papadimitriou. SIAM J. Comput. 2009]

- **Input:** 3-CNF formula  $\varphi$  & satisfying asgmts.  $\sigma_{ini}, \sigma_{tar}$
- **Output:**  $\vec{\sigma} = (\sigma^{(1)}=\sigma_{ini}, \dots, \sigma^{(T)}=\sigma_{tar})$  (reconf. sequence) s.t.  
 $\sigma^{(t)}$  satisfies  $\varphi$  (feasibility)  
 $\text{Ham}(\sigma^{(t)}, \sigma^{(t+1)}) = 1$  (adjacency on hypercube)

$$\varphi = (xvy) \wedge (xvz) \wedge (\bar{x}v\bar{y}v\bar{z})$$

$$\sigma_{ini}(x,y,z) = (1,1,0)$$

$$\sigma_{tar}(x,y,z) = (1,0,1)$$



**YES**

## Example 2

# 3-SAT Reconfiguration

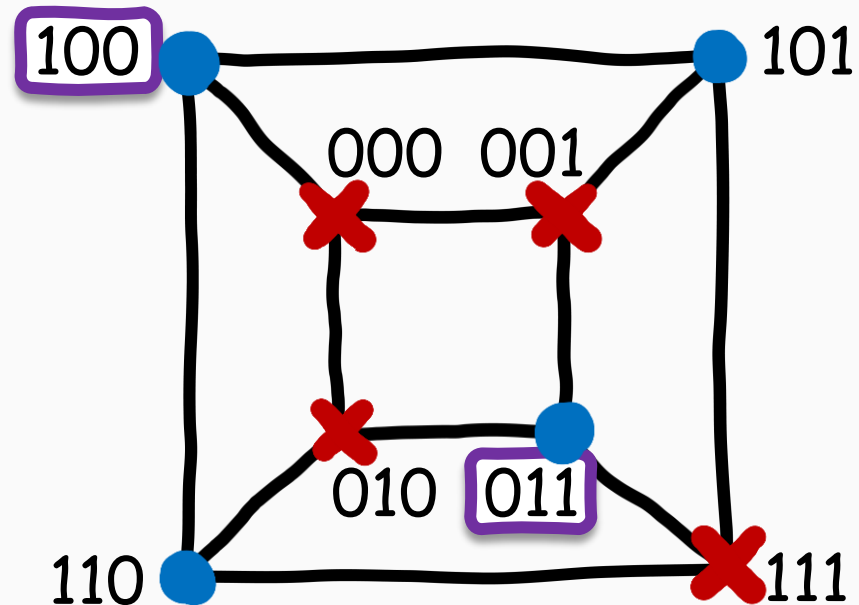
[Gopalan-Kolaitis-Maneva-Papadimitriou. SIAM J. Comput. 2009]

- **Input:** 3-CNF formula  $\varphi$  & satisfying asgmts.  $\sigma_{ini}, \sigma_{tar}$
- **Output:**  $\vec{\sigma} = (\sigma^{(1)}=\sigma_{ini}, \dots, \sigma^{(T)}=\sigma_{tar})$  (reconf. sequence) s.t.  
 $\sigma^{(t)}$  satisfies  $\varphi$  (feasibility)  
 $\text{Ham}(\sigma^{(t)}, \sigma^{(t+1)}) = 1$  (adjacency on hypercube)

$$\varphi = (xvy) \wedge (xvz) \wedge (\bar{x}v\bar{y}v\bar{z})$$

$$\sigma_{ini}(x,y,z) = (1,0,0)$$

$$\sigma_{tar}(x,y,z) = (0,1,1)$$



**NO**

## Example 3

# Maxmin 3-SAT Reconfiguration

[Ito-Demaine-Harvey-Papadimitriou-Sideri-Uehara-Uno. Theor. Comput. Sci. 2011]

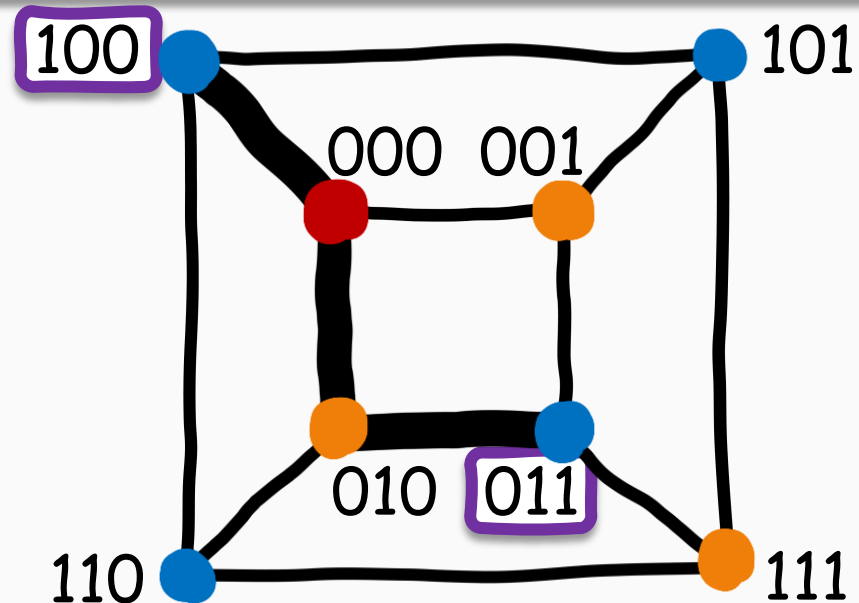
- **Input:** 3-CNF formula  $\varphi$  & satisfying asgmts.  $\sigma_{\text{ini}}, \sigma_{\text{tar}}$
- **Output:**  $\vec{\sigma} = (\sigma^{(1)}=\sigma_{\text{ini}}, \dots, \sigma^{(T)}=\sigma_{\text{tar}})$  (reconf. sequence) s.t.
  - ~~$\sigma^{(t)}$  satisfies  $\varphi$~~  (feasibility)
  - $\text{Ham}(\sigma^{(t)}, \sigma^{(t+1)}) = 1$  (adjacency on hypercube)
- **Goal:** maximize  $\text{val}_{\varphi}(\vec{\sigma}) := \min_t (\text{frac. of satisfied clauses by } \sigma^{(t)})$

$$\varphi = (xvy) \wedge (xvz) \wedge (\bar{x}v\bar{y}v\bar{z})$$

$$\sigma_{\text{ini}}(x,y,z) = (1,0,0)$$

$$\sigma_{\text{tar}}(x,y,z) = (0,1,1)$$

$$\text{val}_{\varphi}(\vec{\sigma}) = \min \left\{ 1, \frac{1}{3}, \frac{2}{3}, 1 \right\} = \frac{1}{3}$$



0.33



## Example 4

# Maxmin 3-SAT Reconfiguration

[Ito-Demaine-Harvey-Papadimitriou-Sideri-Uehara-Uno. Theor. Comput. Sci. 2011]

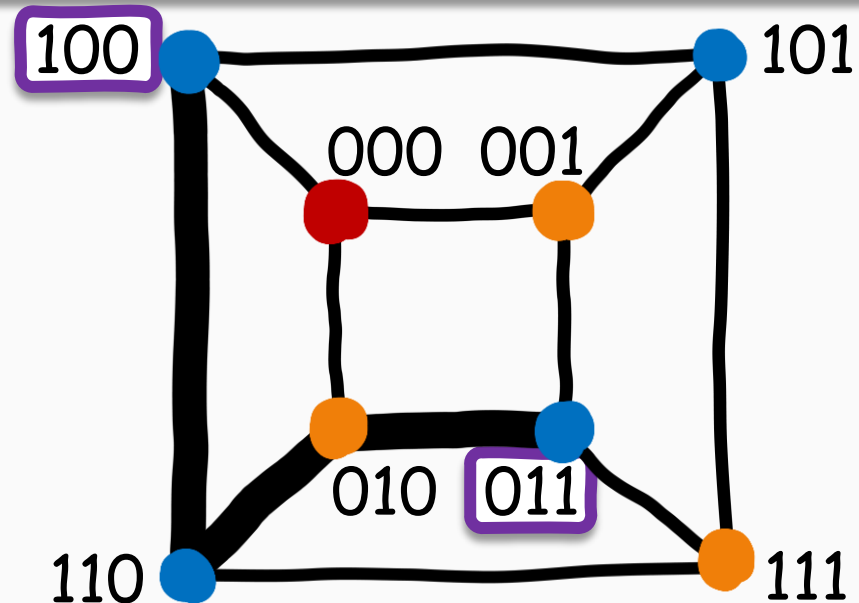
- **Input:** 3-CNF formula  $\varphi$  & satisfying asgmts.  $\sigma_{\text{ini}}, \sigma_{\text{tar}}$
- **Output:**  $\vec{\sigma} = (\sigma^{(1)}=\sigma_{\text{ini}}, \dots, \sigma^{(T)}=\sigma_{\text{tar}})$  (reconf. sequence) s.t.
  - ~~$\sigma^{(t)}$  satisfies  $\varphi$~~  (feasibility)
  - $\text{Ham}(\sigma^{(t)}, \sigma^{(t+1)}) = 1$  (adjacency on hypercube)
- **Goal:** maximize  $\text{val}_{\varphi}(\vec{\sigma}) := \min_t (\text{frac. of satisfied clauses by } \sigma^{(t)})$

$$\varphi = (xvy) \wedge (xvz) \wedge (\bar{x}v\bar{y}v\bar{z})$$

$$\sigma_{\text{ini}}(x,y,z) = (1,0,0)$$

$$\sigma_{\text{tar}}(x,y,z) = (0,1,1)$$

$$\text{val}_{\varphi}(\vec{\sigma}) = \min \{1, 1, \frac{2}{3}, 1\} = \frac{2}{3}$$

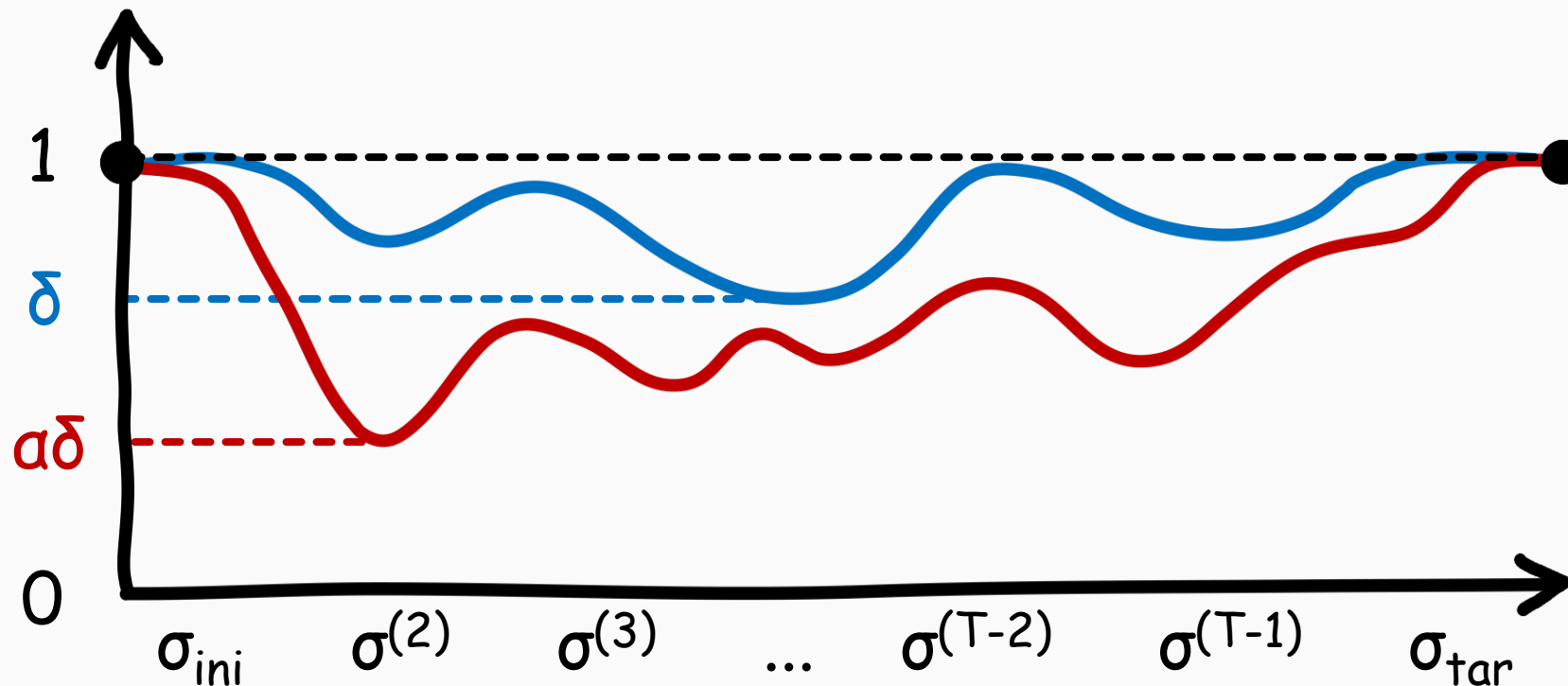


0.666

# Defining approximation algorithms

$\alpha$ -approximation algorithm  $\mathcal{A}$  for Maxmin 3-SAT Reconf.

- If there is  $\vec{\sigma}^*$  s.t.  $\text{val}_\varphi(\vec{\sigma}^*) \geq \delta$  ( $\forall \sigma^{*(t)}$  satisfies  $\geq \delta$ -frac. of clauses)
- then  $\mathcal{A}$  finds  $\vec{\sigma}$  s.t.  $\text{val}_\varphi(\vec{\sigma}) \geq \alpha\delta$  ( $\forall \sigma^{(t)}$  satisfies  $\geq \alpha\delta$ -frac. of clauses)



## Exercise 1

# 0.5-approx. alg. for Maxmin 3-SAT Reconf.

- **Input:** 3-CNF formula  $\varphi$  & satisfying asgmts.  $\sigma_{ini}, \sigma_{tar}$
- **Run:** Sample a random ordering  $\pi$  of vars s.t.  $\sigma_{ini}(x) \neq \sigma_{tar}(x)$   
Create a reconf. sequence  $\vec{\sigma} = (\sigma^{(1)}=\sigma_{ini}, \dots, \sigma^{(T)}=\sigma_{tar})$   
by flipping  $\pi(1), \pi(2), \pi(3), \dots$

Observe:

For any clause  $C$  satisfied by  $\sigma_{ini}$  &  $\sigma_{tar}$

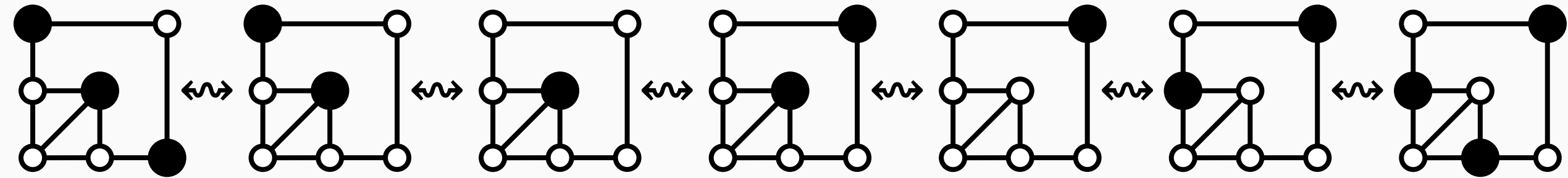
$$\Pr_{\pi}[\text{every } \sigma^{(t)} \text{ satisfies } C] \geq 0.5$$

$$\rightarrow \mathbf{E}_{\pi}[\text{val}_{\varphi}(\vec{\sigma})] \geq 0.5$$

|          |                   |                   |                   |                   |                |
|----------|-------------------|-------------------|-------------------|-------------------|----------------|
| $\pi(3)$ | 1                 | 1                 | 1 $\rightarrow$ 0 | 0                 |                |
|          | 1                 | 1                 | 1                 | 1                 |                |
| $\pi(1)$ | 0 $\rightarrow$ 1 | 1                 | 1                 | 1                 |                |
| $\pi(4)$ | 1                 | 1                 | 1                 | 1 $\rightarrow$ 0 |                |
|          | 0                 | 0                 | 0                 | 0                 |                |
| $\pi(2)$ | 1                 | 1 $\rightarrow$ 0 | 0                 | 0                 |                |
|          | $\sigma_{ini}$    | $\sigma^{(2)}$    | $\sigma^{(3)}$    | $\sigma^{(4)}$    | $\sigma_{tar}$ |

# Other approximate versions

- **Maxmin Independent Set Reconf.** under token-addition-removal model  
[Ito-Demaine-Harvey-Papadimitriou-Sideri-Uehara-Uno. Theor. Comput. Sci. 2011]  
[de Berg-Jansen-Mukherjee. Discret. Appl. Math. 2018]



- Minmax Vertex Cover Reconf. [Ito-Nooka-Zhou. IEICE Trans. Inf. Syst. 2016]
- Minmax Set Cover Reconf.  
[Ito-Demaine-Harvey-Papadimitriou-Sideri-Uehara-Uno. Theor. Comput. Sci. 2011]
- Subset Sum Reconf. [Ito-Demaine. J. Comb. Optim. 2014]
- Submodular Reconf. [O.-Matsuoka. WSDM 2022]
- Maxmin 2-CSP Reconf. [Karthik C. S.-Manurangsi. 2023] [O. 2023]

# Questions of interest

## Algorithmic side

- How well can we approximate reconfiguration problems?

## Hardness side

- How hard is it to approximate reconfiguration problems?

 **My interest**

# Outline of this talk

Part I

What is meant by "approximation"

Part II

Complexity of approximating reconf. problems

Part III

Recent progress, *Struggles* & future directions



## Exercise 2

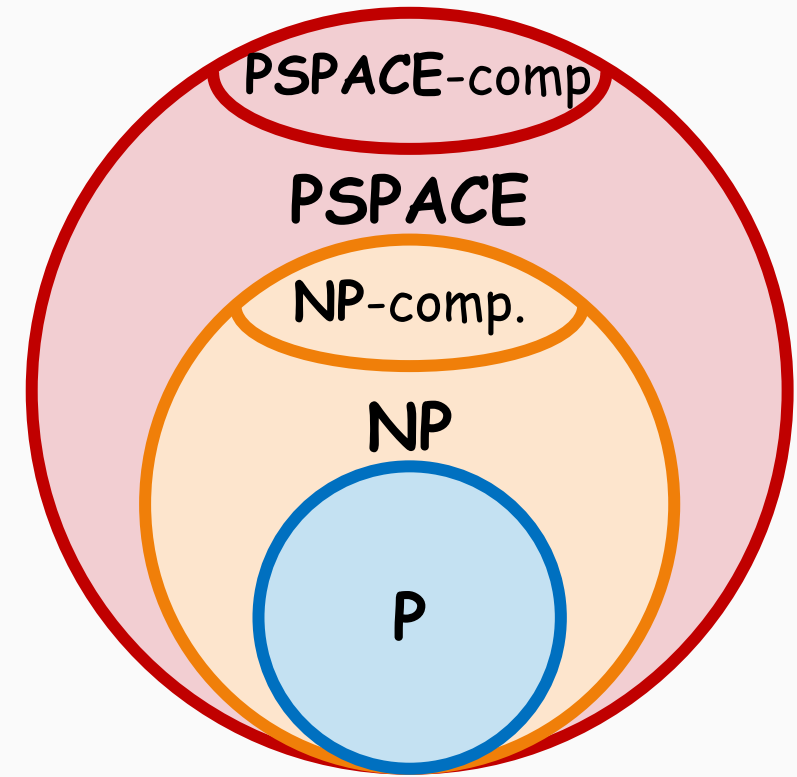
# What we already know

- Maxmin 3-SAT Reconfiguration is...



~~NP-comp.~~

**PSPACE-comp.**



[Gopalan-Kolaitis-Maneva-Papadimitriou.  
SIAM J. Comput. 2009]

⚠ Approximate versions are (at least) harder than decision problems

Our focus

# 😜 Three possible worlds(?)

- **0.999-approx.** of Maxmin 3-SAT Reconfiguration is...



NP-comp.

PSPACE-comp.

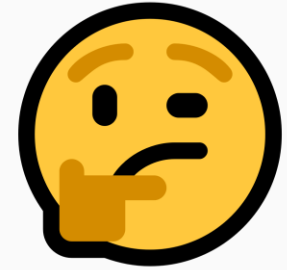
We only know NP-hardness  
(until recently)

[Ito-Demaine-Harvey-Papadimitriou-  
Sideri-Uehara-Uno. ISAAC 2008 &  
Theor. Comput. Sci. 2011]

- Are the problems in Section 4 PSPACE-hard to approximate (not just NP-hard)?

# So why we need **PSPACE**-completeness?

It doesn't matter whether **NP**-hard or **PSPACE**-hard.



algorithm designer

- 1. **PSPACE**-completeness is tight
- 2. No efficient algorithm under  $P \neq PSPACE$
- 3. No short reconf. sequence under  $NP \neq PSPACE$

## Significance of PSPACE-completeness

# 1. PSPACE-completeness is tight

 Reconfiguration problems of

Satisfiability, Independent Set, Coloring, Vertex Cover, Dominating Set, Clique,

Shortest Path, Hamiltonian Cycle, Feedback Vertex Set, Steiner Tree,

Vertex Separator, Odd Cycle Transversal, Induced Forest, L(2,1)-Labeling,

Integer Linear System, Target Set, Set Cover, Subset Sum, H-word

are PSPACE-complete

PSPACE-comp. of approx. implies...

 Solving **approximately** is as hard as solving **exactly**

# Significance of PSPACE-completeness

## 2. No efficient algorithm under $P \neq PSPACE$

| Proposition                  | RW's Estimated Likelihood |
|------------------------------|---------------------------|
| TRUE                         | 100%                      |
| $EXP^{NP} \neq BPP$          | 99%                       |
| $NEXP \not\subseteq P/poly$  | 97%                       |
| $L \neq NP$                  | 95%                       |
| $NP \not\subseteq SIZE(n^k)$ | 93%                       |
| $BPP \subseteq SUBEXP$       | 90%                       |
| $P \neq PSPACE$              | 90%                       |
| $P \neq NP$                  | 80%                       |
| ETH                          | 70%                       |

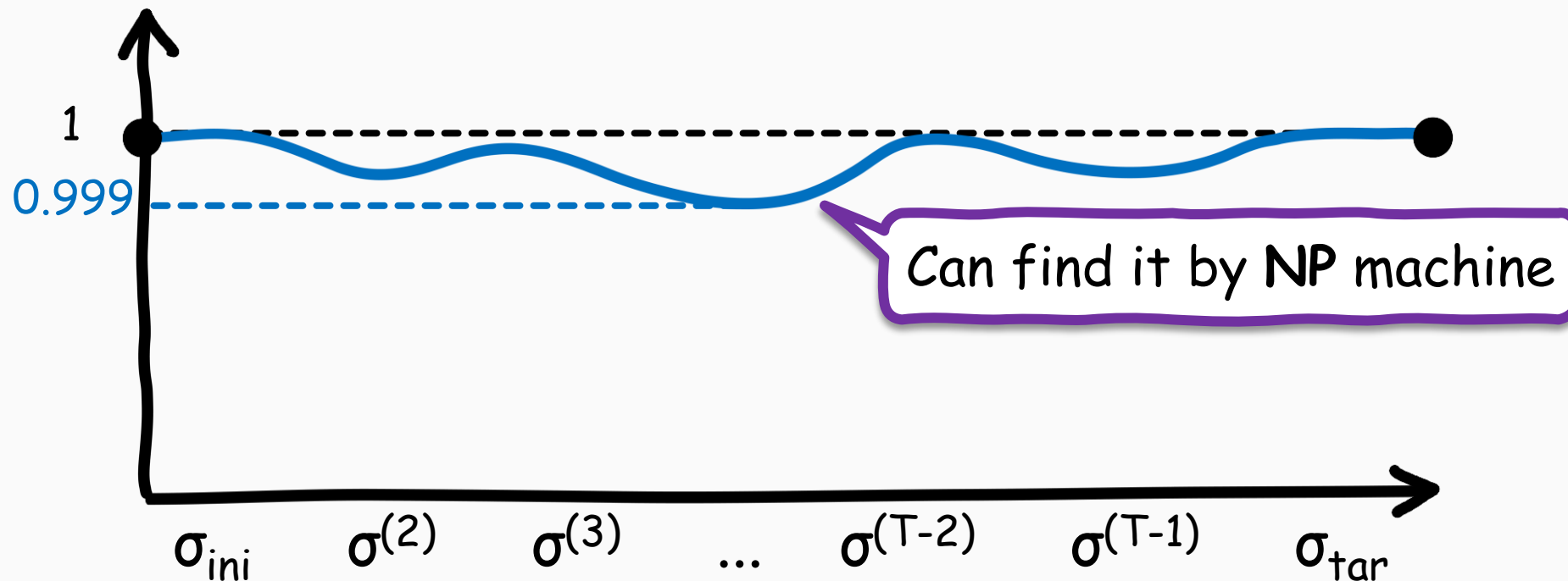
[Ryan Williams. "Some estimated likelihoods for computational complexity". 2019]

 Become **10%** more confident

Significance of PSPACE-completeness

### 3. No short reconf. seq. under $NP \neq PSPACE$

Suppose "there is a 0.999-approx. reconf. sequence of poly-length"



⚠ Diameter of 3-SAT Reconf. can be  $2^{\Omega(n)}$   
[Gopalan-Kolaitis-Maneva-Papadimitriou. SIAM J. Comput. 2009]

😊 Complexity results imply (some) **structural** properties



# Formulating hardness of approximation

## Gap[1 vs. $1-\epsilon$ ] 3-SAT Reconfiguration

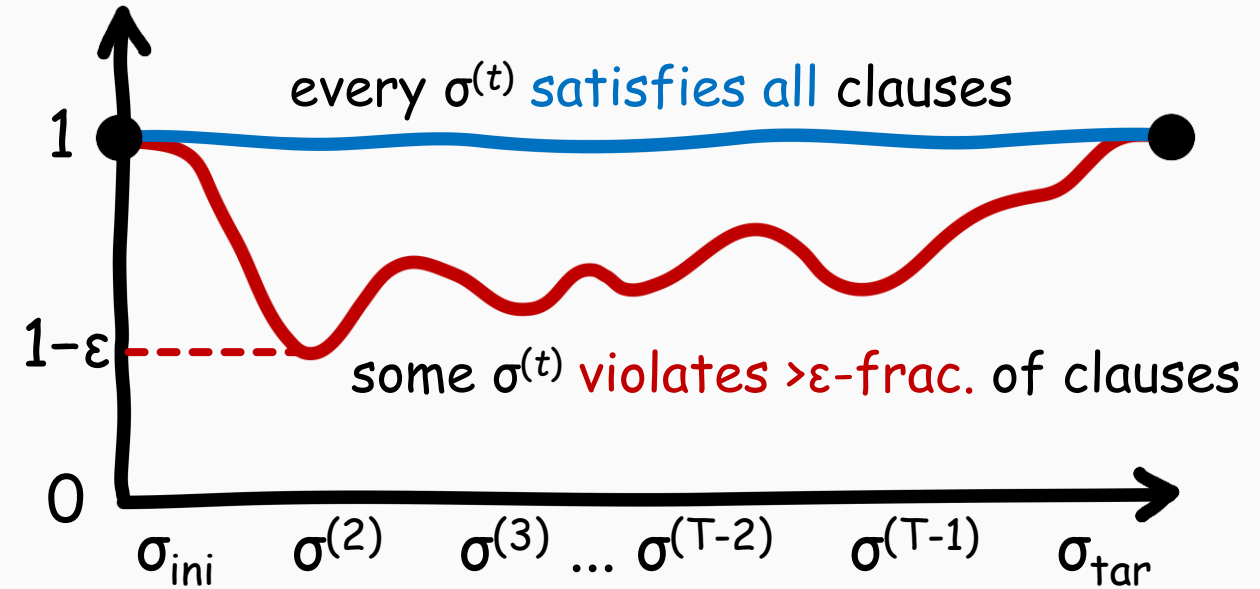
• **Input:**  $\varphi$  & satisfying  $\sigma_{\text{ini}}, \sigma_{\text{tar}}$

• **Goal:** Distinguish between

(Completeness)  $\exists \vec{\sigma} \text{ val}_{\varphi}(\vec{\sigma}) = 1$

(Soundness)  $\forall \vec{\sigma} \text{ val}_{\varphi}(\vec{\sigma}) < 1-\epsilon$

$\text{val}_{\varphi}(\vec{\sigma}) := \min_t (\text{frac. of satisfied clauses by } \sigma^{(t)})$



Gap[1 vs. 1] 3-SAT Reconf. is PSPACE-comp.

Gap[1 vs. 0.5] 3-SAT Reconf. is P

Gap[1 vs. 0.999] 3-SAT Reconf. is  $\mathcal{C}$ -hard

Studying gap problems is enough

$\Rightarrow$  0.999-approx. of Maxmin 3-SAT Reconf. is  $\mathcal{C}$ -hard

# Known hardness-of-approx. results by 2022

**NP-hard**

Probabilistically  
Checkable Proof theorem  
[ALMSS. J. ACM 1998]  
[AS. J. ACM 1998]

Max Ind. Set  
[Håstad. Acta Math. 1999]

Max SAT  
[Håstad. J. ACM 2001]

Maxmin Ind. Set Reconf.  
[IDHPSUU. TCS 2011]

Maxmin SAT Reconf.  
[IDHPSUU. TCS 2011]

**PSPACE-complete**

What's going on?

### Exercise 3

# Gap-preserving reduction from Max 3-SAT to Maxmin 5-SAT Reconf.

[Ito-Demaine-Harvey-Papadimitriou-Sideri-Uehara-Uno. Theor. Comput. Sci. 2011]

Gap[1 vs.  $1-\epsilon$ ] 3-SAT  $\varphi$   
n variables :  $x_1, \dots, x_n$   
m clauses :  $C_1, \dots, C_m$



Gap[1 vs.  $1-\frac{\epsilon}{2}$ ] 5-SAT Reconf.  $(\psi, \sigma_{ini}, \sigma_{tar})$   
n+2 vars. :  $x_1, \dots, x_n, y, z$   
2m clauses :  $C_1 \vee y \vee \bar{z}, \dots, C_m \vee y \vee \bar{z}$   
 $C_1 \vee \bar{y} \vee z, \dots, C_m \vee \bar{y} \vee z$   
asgmts. :  $\sigma_{ini} := 0^{n+2}$  &  $\sigma_{tar} := 1^{n+2}$

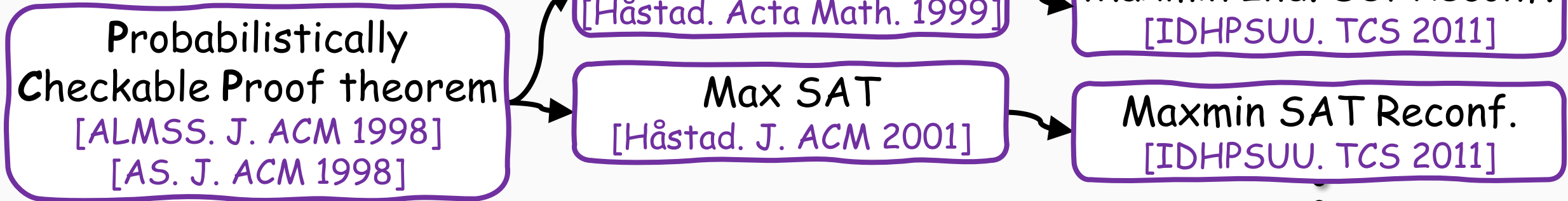
(Completeness)  $\exists \sigma$  satisfies all clauses of  $\varphi \implies \exists \vec{\sigma} \text{ val}_{\psi}(\vec{\sigma}) = 1$

(Soundness)  $\forall \sigma$  violates  $>\epsilon$ -frac. clauses of  $\varphi \implies \forall \vec{\sigma} \text{ val}_{\psi}(\vec{\sigma}) < 1 - \frac{\epsilon}{2}$

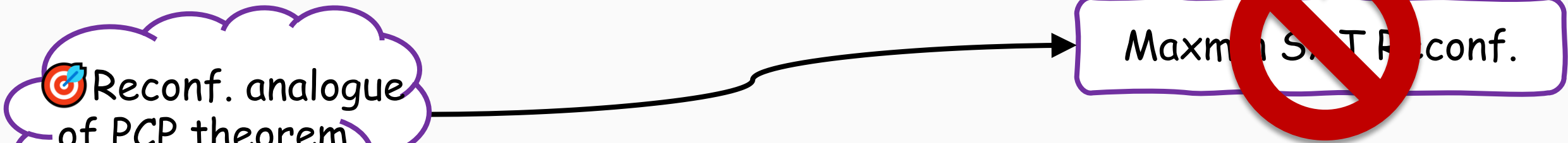
⚠  $\vec{\sigma}$  must "touch"  $y \neq z \rightarrow$  Half of clauses look like:  $C_1 \wedge C_2 \wedge \dots \wedge C_m$

# Toward PSPACE-comp. of approx...

## NP-hard



## PSPACE-complete



Need a theory **beyond** the PCP theorem for NP

# Reconf. analogue of the PCP theorem circa 2023

[O. STACS 2023]

## Reconfiguration Inapproximability Hypothesis

(formal statement omitted)

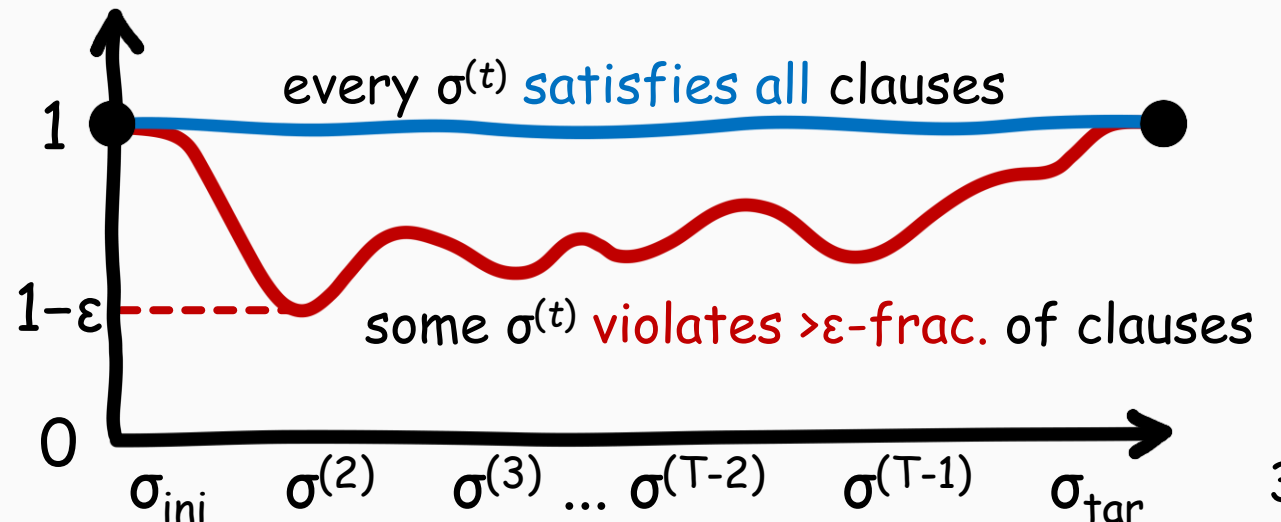
↕ equivalent

" $\exists \varepsilon > 0$  Gap[1 vs.  $1-\varepsilon$ ] 3-SAT Reconf. is PSPACE-complete"

🔄 **Goal:** Distinguish between

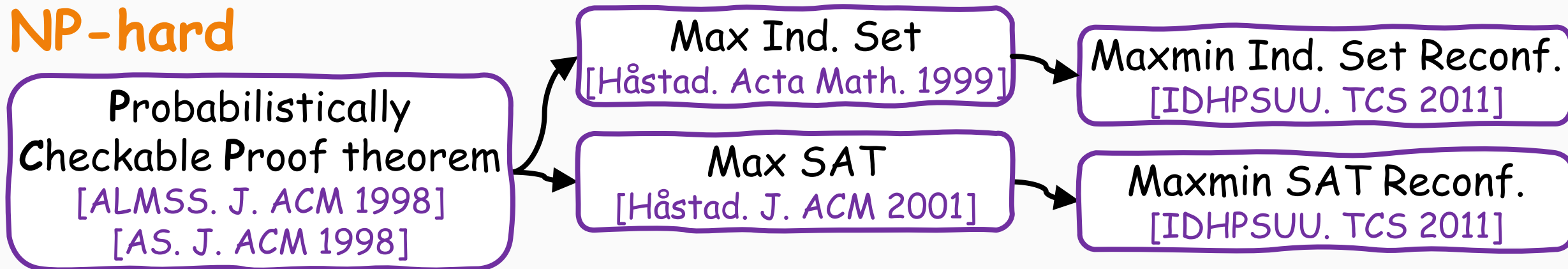
(Completeness)  $\exists \vec{\sigma} \text{ val}_{\varphi}(\vec{\sigma}) = 1$

(Soundness)  $\forall \vec{\sigma} \text{ val}_{\varphi}(\vec{\sigma}) < 1-\varepsilon$

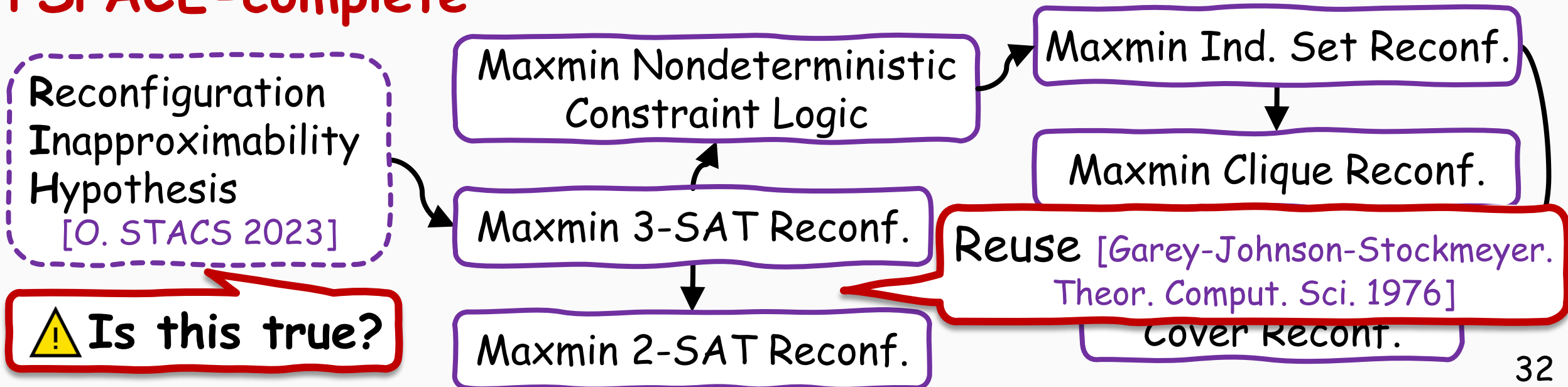


# Settling the open problem **conditional** on RIH

## NP-hard



## PSPACE-complete



# Our main result

[Hirahara-O. STOC 2024] [Karthik C. S.-Manurangsi. 2023]

## Probabilistically Checkable Reconfiguration Proof **theorem**

**def** PCP-like characterization of PSPACE

For any language  $L$  in PSPACE

$\exists$  a verifier  $\mathcal{V}$  with  $O(\log n)$  randomness &  $O(1)$  query complexity

$\exists$  poly-time alg.  $\pi_{\text{ini}}$  &  $\pi_{\text{tar}}$  s.t. for every input  $x \in \{0,1\}^*$

$x \in L \implies \exists (\pi^{(1)}=\pi_{\text{ini}}(x), \dots, \pi^{(T)}=\pi_{\text{tar}}(x)) \quad \forall t \Pr[\mathcal{V}(x) \text{ accepts } \pi^{(t)}] = 1$

$x \notin L \implies \forall (\pi^{(1)}=\pi_{\text{ini}}(x), \dots, \pi^{(T)}=\pi_{\text{tar}}(x)) \quad \exists t \Pr[\mathcal{V}(x) \text{ accepts } \pi^{(t)}] < \frac{1}{2}$



# The open problem resolved **unconditionally**

[Hirahara-O. STOC 2024] [Karthik C. S.-Manurangsi. 2023]

## Probabilistically Checkable Reconfiguration Proof **theorem**

↕ equivalent

$\exists \varepsilon > 0$  Gap[1 vs.  $1-\varepsilon$ ] 3-SAT Reconf. is **PSPACE**-complete

- Are the problems in Section 4 PSPACE-hard to approximate (not just NP-hard)?



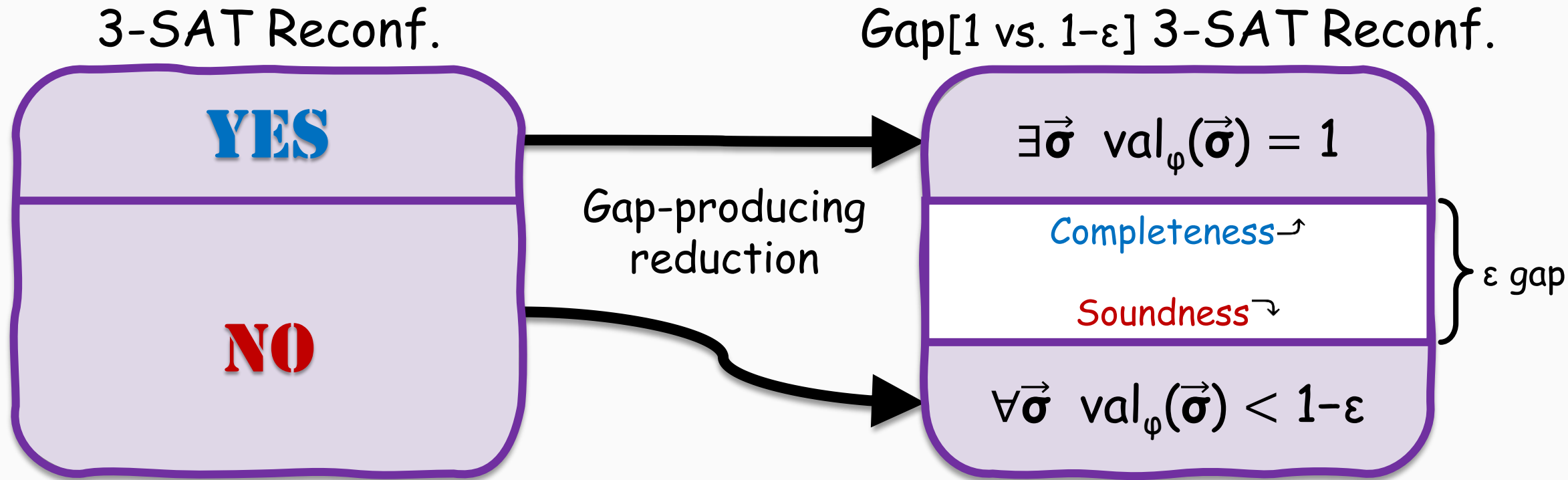
**YES!**





# How to prove the PCRP theorem?

[Hirahara-O. STOC 2024] [Karthik C. S.-Manurangsi. 2023]



- Our luck: **PCP of proximity** (a.k.a. assignment testers)

[Ben-Sasson, Goldreich, Harsha, Sudan, Vadhan. SIAM J. Comput. 2006]

[Dinur-Reingold. SIAM J. Comput. 2006]

So the story ends...?



 Optimal **PSPACE**-completeness of approx.

# Outline of this talk

Part I

What is meant by "approximation"

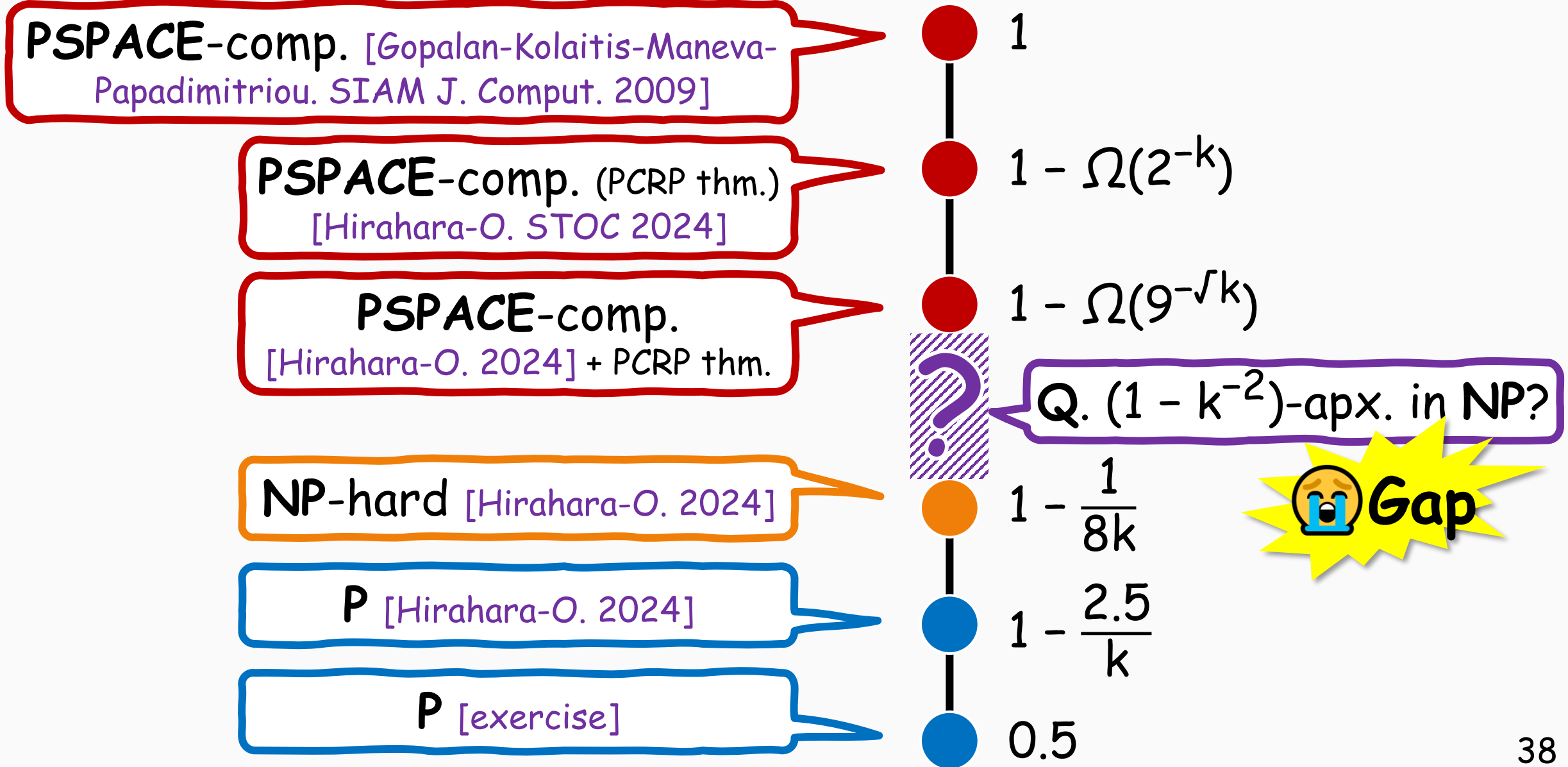
Part II

Complexity of approximating reconf. problems

Part III

Recent progress, **Struggles** & future directions

# Current status of Maxmin k-SAT Reconf.




# Review of recent progress

| problem                  | polynomial time                           | PSPACE-complete                                   |
|--------------------------|---|---|
| Maxmin k-SAT Reconf.     | $1 - \frac{2.5}{k}$<br>[Hirahara-O. 2024] | $1 - \Omega(9^{-\sqrt{k}})$<br>[Hirahara-O. 2024] |
| Minmax Set Cover Reconf. | 2 [IDHPSUU. Theor. Comput. Sci. 2011]     | $2 - o(1)$<br>[Hirahara-O. ICALP 2024]            |
| Maxmin Ind. Set Reconf.  | $n^{-1}$                                  | $n^{-0.001}$<br>[Hirahara-O. STOC 2024]           |
| Maxmin 2-CSP Reconf.     | 0.5 [Karthik C. S.-Manurangsi. 2023]      | 0.9942<br>[O. SODA 2024]                          |
| Maxmin k-Cut Reconf.     | $1 - \frac{2}{k}$<br>[Hirahara-O. 2024]   | $1 - \Omega(\frac{1}{k})$<br>[Hirahara-O. 2024]   |



# to transfer PCP tools to the reconfiguration world

| existing PCP tools  | techniques in reconf. world  |
|---|--|
| FGLSS reduction [Feige-Goldwasser-Lovász-Safra-Szegedy. J. ACM 1996]    | Alphabet squaring [Hirahara-O. ICALP 2024]   |
| Degree reduction [Papadimitriou-Yannakakis. J. Comput. Syst. Sci. 1991] | Alphabet squaring [O. STACS 2023]  |
| Gap amplification [Dinur. J. ACM 2007]                                  | Alphabet squaring [O. SODA 2024]   |
| Alphabet reduction [Dinur. J. ACM 2007]                                 | Reconfigurability of Hadamard codes [O. ICALP 2024]  |
| Parallel repetition theorem [Raz. SIAM J. Comput. 1998]                 |  Applied to Max k-SAT [Håstad. J. ACM 2001] |
| Long code test [Bellare-Goldreich-Sudan. SIAM J. Comput. 1998]          |  |

# Some future directions

## 1. Source problems in P

- Shortest Path Reconf. is **PSPACE**-complete [Bonsma. Theor. Comput. Sci. 2013]
- **PSPACE**-complete to approximate as well?

## 2. Puzzles

- Study approximability of Sliding Block Puzzle (?)  
[Hearn-Demaine. Theor. Comput. Sci. 2005]
-  Hardness of approx. for planar Nondeterministic Constraint Logic

Logspace analogue of XP

## 3. Parameterized inapproximability

- $k$ -Clique Reconf. &  $k$ -Dominating Set Reconf. are "**XL**-complete"  
[Bodlaender-Groenland-Nederlof-Swennenhuis. FOCS 2021]  
[Bodlaender-Groenland-Swennenhuis. IPEC 2021]
- **XL**-complete to approximate?

# Conclusion & Takeaway

Resolved 4<sup>th</sup> open problem of

[Ito-Demaine-Harvey-Papadimitriou-Sideri-Uehara-Uno. Theor. Comput. Sci. 2011]

Now ready to study

hardness of approx. & approx. algorithms

for reconfiguration problems

**THANK YOU!**