# *Baby* PIH: Parameterized Inapproximability of Min CSP


Venkatesan Guruswami

UC Berkeley

**Xuandi Ren**

UC Berkeley

Sai Sandeep

UC Berkeley

# Outline

- Background
  - Parameterized Complexity
  - Constraint Satisfaction Problem (CSP)
  - Parameterized Inapproximability Hypothesis (PIH)
- Our Result
  - Baby PIH
- Proof Overview

# Outline

- Background
  - Parameterized Complexity
  - Constraint Satisfaction Problem (CSP)
  - Parameterized Inapproximability Hypothesis (PIH)
- Our Result
  - Baby PIH
- Proof Overview

# Parameterized Complexity

How to cope with an NP-hard problem?

- Associate each instance $x$ with a parameter $k \in \mathbb{N}$
  - $k \ll |x|$
  - Measure complexity over $n = |x|$ and $k$
- **FPT** (**F**ixed-**P**arameter **T**ractable, Analogue of **P**):
  - Problems that admit $f(k) \cdot n^{O(1)}$ time algorithms for some computable function $f$

$k$-Vertex Cover

- Input:
  - $G = (V, E)$ and parameter $k$
- Output:
  - $\exists v_1, \ldots, v_k \in V$ covering all the edges?

has an $O(2^k \cdot n^{O(1)})$ enumeration algorithm

Efficient for small $k$!

$\in$ FPT

# Parameterized Complexity

How to cope with an NP-hard problem?

- Associate each instance $x$ with a parameter $k \in \mathbb{N}$
  - $k \ll |x|$
  - Measure complexity over $n = |x|$ and $k$
- **W[1]** (Analogue of **NP**): widely believed **W[1]** $\neq$ **FPT**

$k$-Clique

- Input:
  - $G = (V, E)$ and parameter $k$
- Output:
  - $\exists v_1, \ldots, v_k \in V$ forming a clique?

Unlikely to have an $f(k) \cdot n^{O(1)}$ time algorithm

No algorithm known with runtime $n^{o(k)}$

W[1] -complete

# Parameterized Approximation

Can we find a $\boldsymbol{g(k)}$-approximation in $\boldsymbol{f(k) \cdot n^{O(1)}}$ time, for some computable functions $f, g$?

E.g., Can we find a $\frac{k}{2}$-clique in a graph with a $k$-clique?

# Parameterized Approximation

Can we find a $\boldsymbol{g(k)}$-approximation in $\boldsymbol{f(k) \cdot n^{O(1)}}$ time, for some computable functions $f, g$?

- **Optimal** ratio in FPT
- Beat polytime algorithms: $2.611 + \varepsilon$ for $\boldsymbol{k}$-Median, $9 + \varepsilon$ for $\boldsymbol{k}$-Means

**Example** [Cohen-Addad, Gupta, Kumar, Lee, Li'19]:

- $\left(1 + \frac{2}{e} + \varepsilon\right)$-approximation algorithm for $\boldsymbol{k}$-Median
- $\left(1 + \frac{8}{e} + \varepsilon\right)$-approximation algorithm for $\boldsymbol{k}$-Means

with runtime $\left(\frac{k \log k}{\varepsilon^2}\right)^k \cdot n^{O(1)}$

# Parameterized Hardness of Approximation

- $\boldsymbol{k}$-SetCover
  - [Chen-Lin'18, Lin'19, Lin-Ren-Sun-Wang'23a] via threshold graph composition
  - [Karthik-Laekhanukit-Manurangsi'19] via distributed PCP framework
- $\boldsymbol{k}$-Clique
  - [Lin'21, Karthik-Khot'22, Lin-Ren-Sun-Wang'23b] via locally decodable codes
  - [Chen-Feng-Laekhanukit-Liu'23] via Sidon sets
- Max $\boldsymbol{k}$-Coverage
  - [Manurangsi'20] via $k$-wise agreement testing
- …

*Ad-hoc reductions,*

*tailored to the specific problems!*

# Parameterized Hardness of Approximation

Unified and powerful machinery for parameterized inapproximability?

*Parameterized PCP-type theorem!*

# Recall: PCP Theorem

**Constraint Satisfaction Problem**

**Input:** $\Pi = (X, \Sigma, \Phi)$
- $X$: variables
- $\Sigma$: alphabet
- $\Phi$: constraints

**Output:**
- $\exists \sigma: X \to \Sigma$ satisfying all constraints?

val($\Pi$):=max. fraction of constraints satisfied by some assignment

**$(1$ vs $\delta)$ gap CSP**

**Input:** a CSP instance $\Pi = (X, \Sigma, \Phi)$

**Goal:** distinguish $\boxed{\text{val}(\Pi)} = 1$ vs val($\Pi$)$\leq \delta$

- PCP Theorem:
  - For any constant $\Sigma$ and let $n = |X|$, there is no $n^{O(1)}$ time algorithm for (1 vs 0.9) gap CSP assuming P≠NP.

# Parameterized Inapproximability Hypothesis

**Constraint Satisfaction Problem**

`Input:` $\Pi = (X, \Sigma, \Phi)$

- $X$: variables
- $\Sigma$: alphabet
- $\Phi$: constraints

`Output:`

- $\exists \sigma: X \to \Sigma$ satisfying all constraints?

val($\Pi$):=max. fraction of constraints satisfied by some assignment

**$(1$ vs $\delta)$ gap CSP**

`Input:` a CSP instance $\Pi = (X, \Sigma, \Phi)$

`Goal:` distinguish $\boxed{\text{val}(\Pi)} = 1$ vs val$(\Pi) \leq \delta$

- Parameterized CSP:
  - $k = |X|$ and $n = |\Sigma|$, is there an $f(k) \cdot n^{O(1)}$ time algorithm?
  - Example: **Multi-colored *k*-Clique**

<u>PIH (Parameterized Inapproximability Hypothesis)</u> [Lokshtanov-Ramanujan-Saurabh-Zehavi'20]:
Let $k = |X|$ and $n = |\Sigma|$, there is no $f(k) \cdot n^{O(1)}$ time algorithm for (1 vs 0.9) gap parameterized CSP.

# Parameterized Inapproximability Hypothesis

- The analogue of PCP theorem here is W[1]≠FPT ⇒ PIH
- It was known [Dinur-Manurangsi'18] that Gap-ETH ⇒ PIH
  - **Gap-ETH**: "Constant approximating Max3SAT requires $2^{\Omega(n)}$ time"
- In a recent breakthrough [Guruswami-Lin-Ren-Sun-Wu'24], it was proven ETH ⇒ PIH
  - **ETH**: "3SAT requires $2^{\Omega(n)}$ time"

# Outline

- Background
  - Parameterized Complexity
  - Constraint Satisfaction Problem (CSP)
  - Parameterized Inapproximability Hypothesis (PIH)
- Our Result
  - Baby PIH
- Proof Overview

# List Satisfiability of CSP

2-CSP

- Input: $\Pi = (X, \Sigma, \Phi)$
- Output:
  - $\exists$ multi-assignment $\sigma: X \to 2^{\Sigma}$ list-satisfying all constraints?

List Value

max. list size of a list-satisfying multi-assignment $\sigma: X \to 2^{\Sigma}$

- A constraint on $(x, y)$ is list-satisfied iff $\exists u \in \sigma(x), v \in \sigma(y)$, s.t. $(u, v)$ satisfies this constraint.

# List Satisfiability of CSP

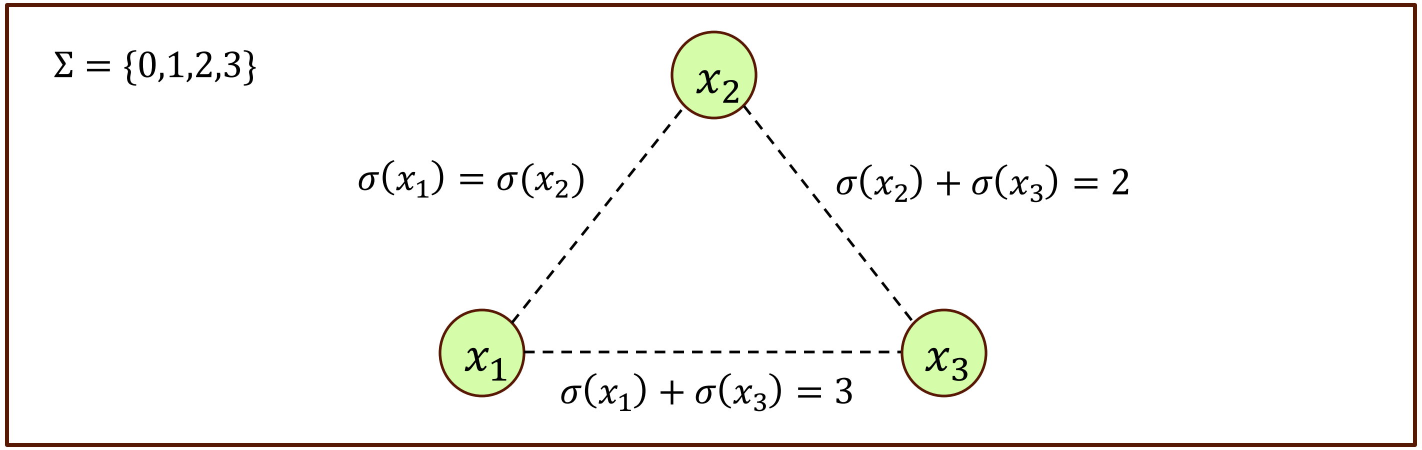
**2-CSP**

- Input: $\Pi = (X, \Sigma, \Phi)$
- Output:
  - $\exists$ multi-assignment $\sigma: X \to 2^{\Sigma}$ list-satisfying all constraints?

**List Value**

max. list size of a list-satisfying multi-assignment $\sigma: X \to 2^{\Sigma}$

- A constraint on $(x, y)$ is list-satisfied iff $\exists u \in \sigma(x), v \in \sigma(y)$, s.t. $(u, v)$ satisfies this constraint.



# List Satisfiability of CSP

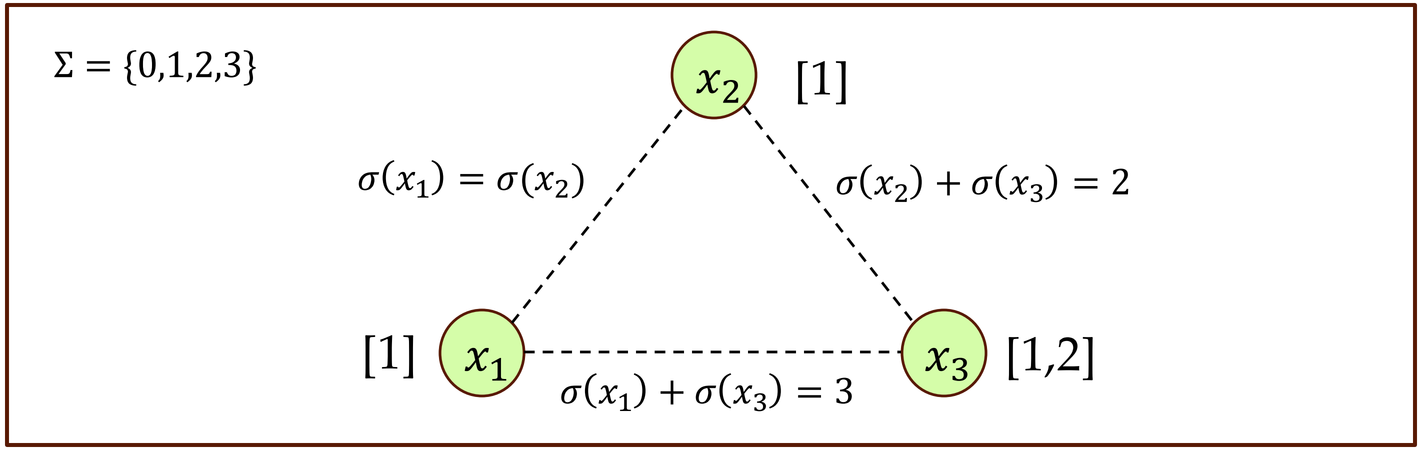
**2-CSP**

- Input: $\Pi = (X, \Sigma, \Phi)$
- Output:
  - $\exists$ multi-assignment $\sigma: X \to 2^{\Sigma}$ list-satisfying all constraints?

**List Value**

max. list size of a list-satisfying multi-assignment $\sigma: X \to 2^{\Sigma}$

- A constraint on $(x, y)$ is list-satisfied iff $\exists u \in \sigma(x), v \in \sigma(y)$, s.t. $(u, v)$ satisfies this constraint.

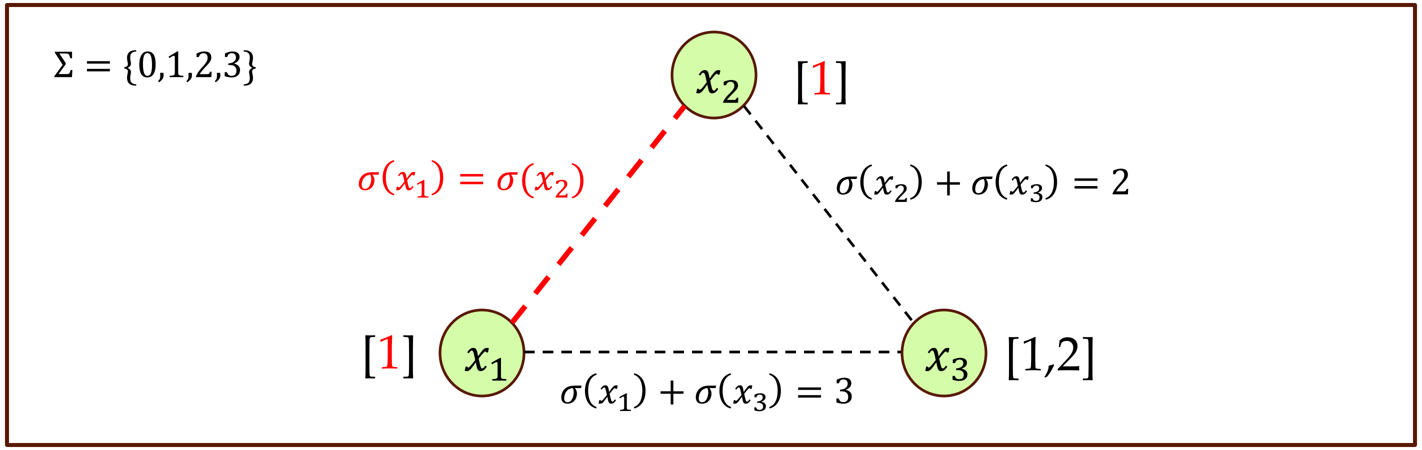


# List Satisfiability of CSP

**2-CSP**
- Input: $\Pi = (X, \Sigma, \Phi)$
- Output:
  - $\exists$ multi-assignment $\sigma: X \to 2^{\Sigma}$ list-satisfying all constraints?

**List Value**
max. list size of a list-satisfying multi-assignment $\sigma: X \to 2^{\Sigma}$

- A constraint on $(x, y)$ is list-satisfied iff $\exists u \in \sigma(x), v \in \sigma(y)$, s.t. $(u, v)$ satisfies this constraint.



# List Satisfiability of CSP

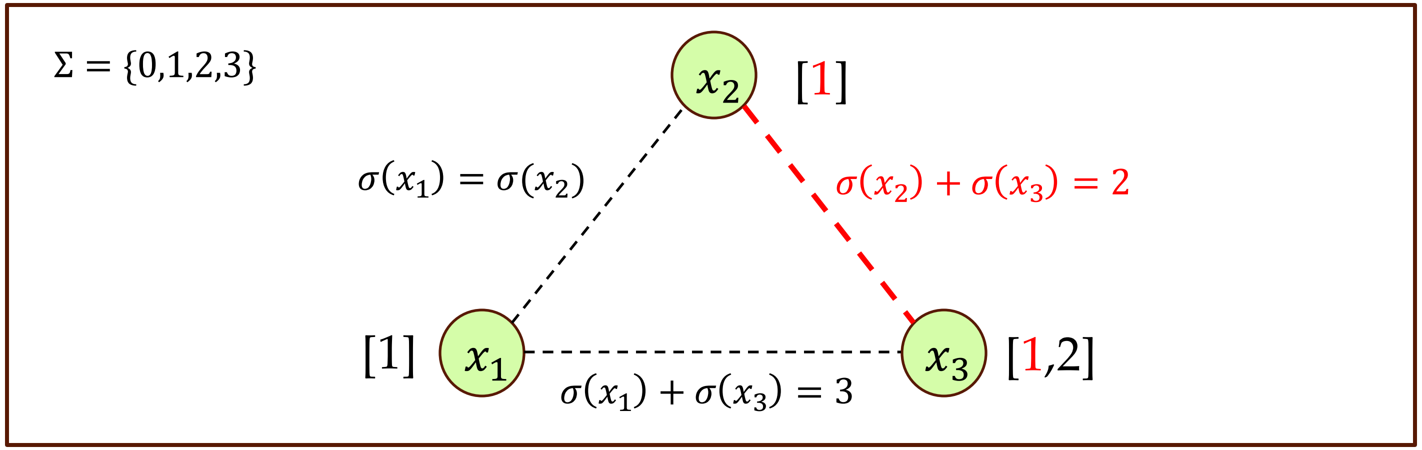
**2-CSP**

- Input: $\Pi = (X, \Sigma, \Phi)$
- Output:
  - $\exists$ multi-assignment $\sigma: X \to 2^{\Sigma}$ list-satisfying all constraints?

**List Value**

max. list size of a list-satisfying multi-assignment $\sigma: X \to 2^{\Sigma}$

- A constraint on $(x, y)$ is list-satisfied iff $\exists u \in \sigma(x), v \in \sigma(y)$, s.t. $(u, v)$ satisfies this constraint.



# List Satisfiability of CSP

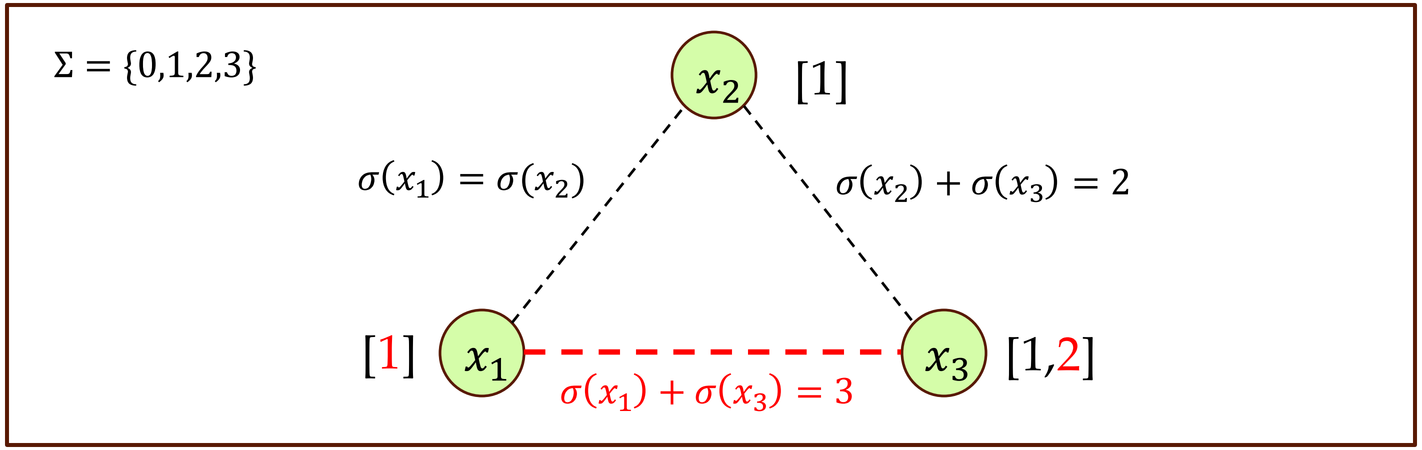
**2-CSP**

- Input: $\Pi = (X, \Sigma, \Phi)$
- Output:
  - $\exists$ multi-assignment $\sigma: X \to 2^{\Sigma}$ list-satisfying all constraints?

**List Value**

max. list size of a list-satisfying multi-assignment $\sigma: X \to 2^{\Sigma}$

- A constraint on $(x, y)$ is list-satisfied iff $\exists u \in \sigma(x), v \in \sigma(y)$, s.t. $(u, v)$ satisfies this constraint.



# List Satisfiability of CSP

**2-CSP**

- Input: $\Pi = (X, \Sigma, \Phi)$
- Output:
  - $\exists$ multi-assignment $\sigma: X \to 2^{\Sigma}$ list-satisfying all constraints?

**List Value**

max. list size of a list-satisfying multi-assignment $\sigma: X \to 2^{\Sigma}$

- We say a 2-CSP is $r$-list satisfiable iff $\exists \sigma$ with $\max_{x \in X} |\sigma(x)| \le r$ list-satisfying all constraints.

# List Satisfiability of CSP

**2-CSP**

- Input: $\Pi = (X, \Sigma, \Phi)$
- Output:
  - $\exists$ multi-assignment $\sigma: X \to 2^{\Sigma}$ list-satisfying all constraints?

**List Value**

max. list size of a list-satisfying multi-assignment $\sigma: X \to 2^{\Sigma}$

- We say a 2-CSP is $r$-list satisfiable iff $\exists \sigma$ with $\max_{x \in X} |\sigma(x)| \leq r$ list-satisfying all constraints.
  - CSP Value=1 $\Leftrightarrow$ 1-list satisfiable $\Rightarrow$ $r$-list satisfiable for $r \geq 2$
  - $r$-list satisfiable $\Rightarrow$ CSP Value $\geq 1/r^2$

# Baby PCP

**2-CSP**

- Input: $\Pi = (X, \Sigma, \Phi)$
- Output:
  - $\exists$ multi-assignment $\sigma : X \to 2^{\Sigma}$ list-satisfying all constraints?

**List Value**

max. list size of a list-satisfying multi-assignment $\sigma : X \to 2^{\Sigma}$

- We say a 2-CSP is $r$-list satisfiable iff $\exists \sigma$ with $\max_{x \in X} |\sigma(x)| \leq r$ list-satisfying all constraints.
  - CSP Value=1 $\Leftrightarrow$ 1-list satisfiable $\Rightarrow$ $r$-list satisfiable for $r \geq 2$
  - $r$-list satisfiable $\Rightarrow$ CSP Value $\geq 1/r^2$

- Baby PCP [Barto-Kozik'22]
  - For any $r > 1$, It's NP-hard to distinguish between [1-list satisfiable] and [not even $r$-list satisfiable].

# Baby PCP

**2-CSP**

- Input: $\Pi = (X, \Sigma, \Phi)$
- Output:
  - $\exists$ multi-assignment $\sigma: X \to 2^{\Sigma}$ list-satisfying all constraints?

**List Value**

max. list size of a list-satisfying multi-assignment $\sigma: X \to 2^{\Sigma}$

- We say a 2-CSP is $r$-list satisfiable iff $\exists \sigma$ with $\max_{x \in X} |\sigma(x)| \leq r$ list-satisfying all constraints.
  - CSP Value=1 $\Leftrightarrow$ 1-list satisfiable $\Rightarrow$ $r$-list satisfiable for $r \geq 2$
  - $r$-list satisfiable $\Rightarrow$ CSP Value $\geq 1/r^2$

- Baby PCP [Barto-Kozik'22]
  - For any $r > 1$, It's NP-hard to distinguish between [1-list satisfiable] and [not even $r$-list satisfiable].
- $\Leftarrow$ PCP $\Uparrow$ $\Uparrow$ (when $\varepsilon < 1/r^2$)
  - For any $\varepsilon > 0$, It's NP-hard to distinguish between [CSP Value =1] and [CSP Value <$\varepsilon$].

# Baby PCP

- Baby PCP [Barto-Kozik'22]
  - Assuming NP≠P, for any $r > 1$, distinguishing between [1-list satisfiable] and [not even $r$-list satisfiable] cannot be done in $|\Pi|^{O(1)}$ time.
  - (A combinatorial proof)
  - (Enough to prove the NP-hardness of some PCSPs (e.g., $(2+\varepsilon)$-SAT))

# Baby PIH

- Baby PCP [Barto-Kozik'22]
  - Assuming NP≠P, for any $r > 1$, distinguishing between [1-list satisfiable] and [not even $r$-list satisfiable] cannot be done in $|\Pi|^{O(1)}$ time.
  - (A combinatorial proof)
  - (Enough to prove the NP-hardness of some PCSPs (e.g., $(2+\varepsilon)$-SAT))

- Baby PIH [This work]
  - Assuming W[1]≠FPT, … cannot be done in $f(|X|) \cdot |\Sigma|^{O(1)}$ time.
  - (An itself interesting inapproximability result for list-satisfiability of CSP)
  - (A step towards PIH)
  - (Enough to get some applications of PIH?)

# Baby PIH

- Baby PCP [Barto-Kozik'22]
  - Assuming NP≠P, for any $r > 1$, distinguishing between [1-list satisfiable] and [not even $r$-list satisfiable] cannot be done in $|\Pi|^{O(1)}$ time.
  - (A combinatorial proof)
  - (Enough to prove the NP-hardness of some PCSPs (e.g., $(2+\varepsilon)$-SAT))

- Baby PIH [This work]
  - Assuming W[1]≠FPT, … cannot be done in $f(|X|) \cdot |\Sigma|^{O(1)}$ time.
  - (An itself interesting inapproximability result for list-satisfiability of CSP)
  - (A step towards PIH)
  - (Enough to get some applications of PIH?)
    - not sure…, but something stronger is enough!
    - PIH ⇒ Average Baby PIH ⇒ Baby PIH

# Outline

- Background
  - Parameterized Complexity
  - Constraint Satisfaction Problem (CSP)
  - Parameterized Inapproximability Hypothesis (PIH)
- Our Result
  - Baby PIH
- Proof Overview

# Proof Overview

- Follows from and extends [Barto-Kozik'22]'s proof of Baby PCP Theorem
- Direct Product Construction

2-CSP

$\Pi = (X, \Sigma, \Phi)$

$\longrightarrow$

$t$-wise Direct Product 2-CSP

$$\Pi^{\odot t} = \left(\binom{X}{t}, \Sigma^t, \Phi'\right)$$

*partial satisfying assignments*
for the set of variables

# Proof Overview

- Follows from and extends [Barto-Kozik'22]'s proof of Baby PCP Theorem
- Direct Product Construction

**2-CSP**
$\Pi = (X, \Sigma, \Phi)$

$\longrightarrow$

***t*-wise Direct Product 2-CSP**
$\Pi^{\odot t} = \left( \binom{X}{t}, \Sigma^t, \Phi' \right)$

*consistency checks*

# Proof Overview

- Follows from and extends [Barto-Kozik'22]'s proof of Baby PCP Theorem
- Direct Product Construction

2-CSP
$\Pi = (X, \Sigma, \Phi)$

$\longrightarrow$

$t$-wise Direct Product 2-CSP
$$\Pi^{\odot t} = \left(\binom{X}{t}, \Sigma^t, \Phi'\right)$$

*partial satisfying assignments* for the set of variables

*consistency checks*

- (Want to show):
- For any $r > 1$, there exists $t$ depending on $r$, such that for every $\Pi$,
  - (Completeness) If $\Pi$ is satisfiable, then so is $\Pi^{\odot t}$ .
  - (Soundness) If $\Pi$ is not satisfiable, then $\Pi^{\odot t}$ is not $r$-list satisfiable.

# Proof Overview

- Follows from and extends [Barto-Kozik'22]'s proof of Baby PCP Theorem
- Direct Product Construction

2-CSP
$\Pi = (X, \Sigma, \Phi)$

$\longrightarrow$

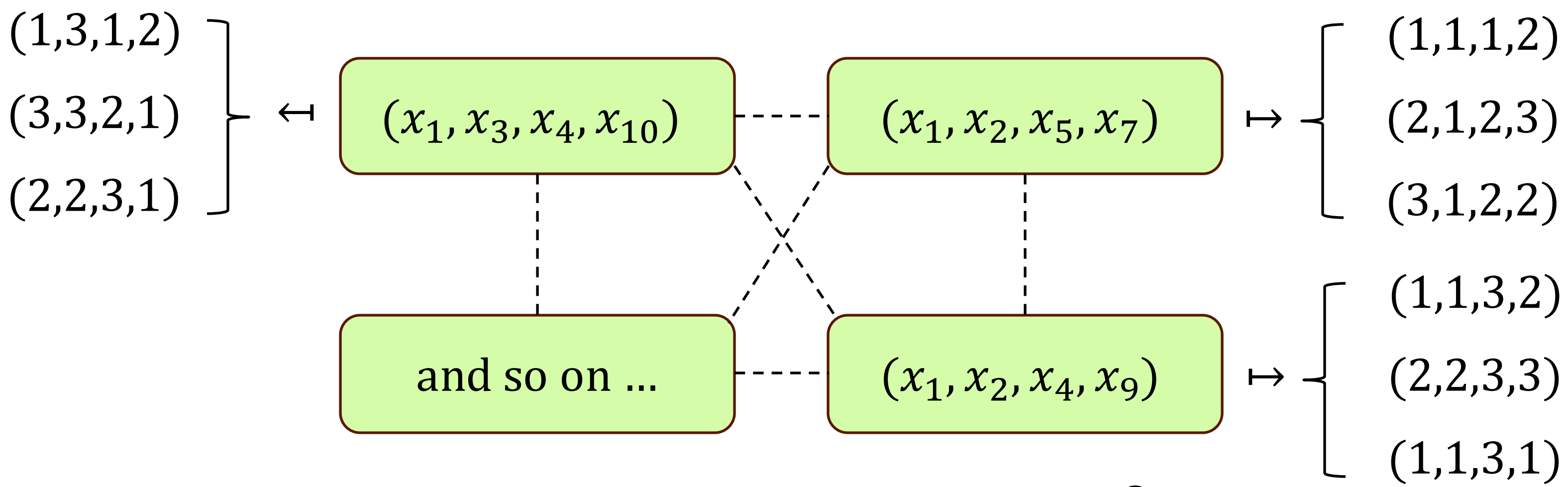
$t$-wise Direct Product 2-CSP
$\Pi^{\odot t} = \left(\binom{X}{t}, \Sigma^t, \Phi'\right)$

*partial satisfying assignments* for the set of variables

*consistency checks*

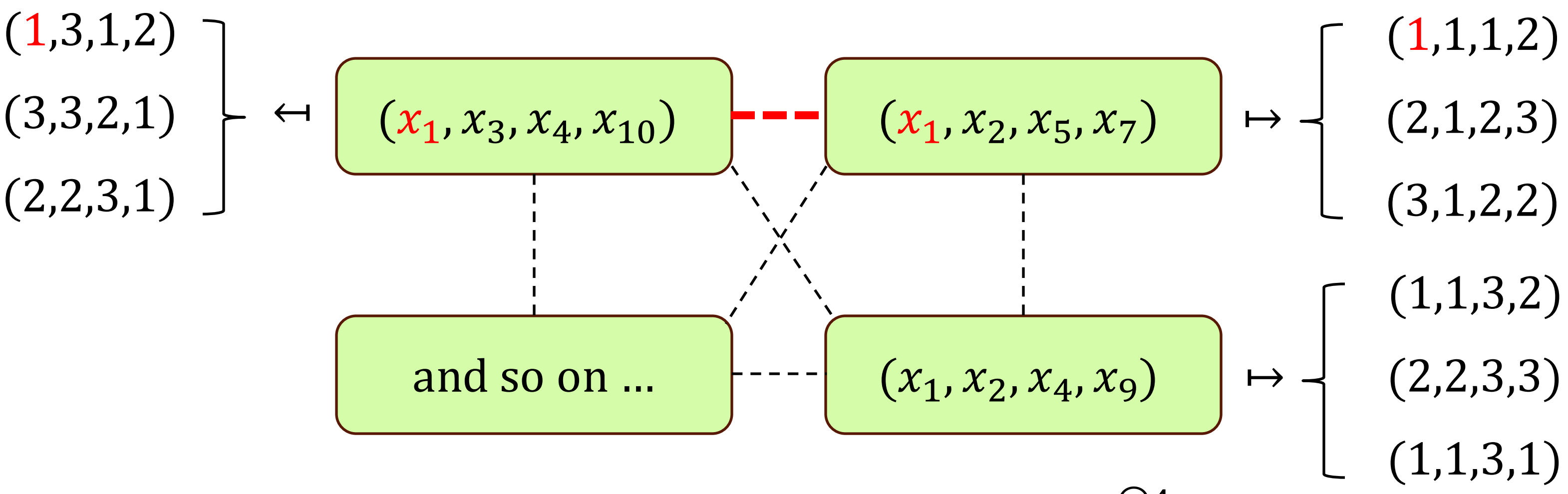
- (Want to show):
- For any $r > 1$, there exists $t$ depending on $r$, such that for every $\Pi$,
  - (Completeness) If $\Pi$ is satisfiable, then so is $\Pi^{\odot t}$ .
  - (Soundness) If $\Pi$ is not satisfiable, then $\Pi^{\odot t}$ is not $r$-list satisfiable.
- Reduction time: $n^{O_r(1)}$ where $n = |\Pi|$
  - a unified proof for both Baby PCP and Baby PIH!

# Proof Overview



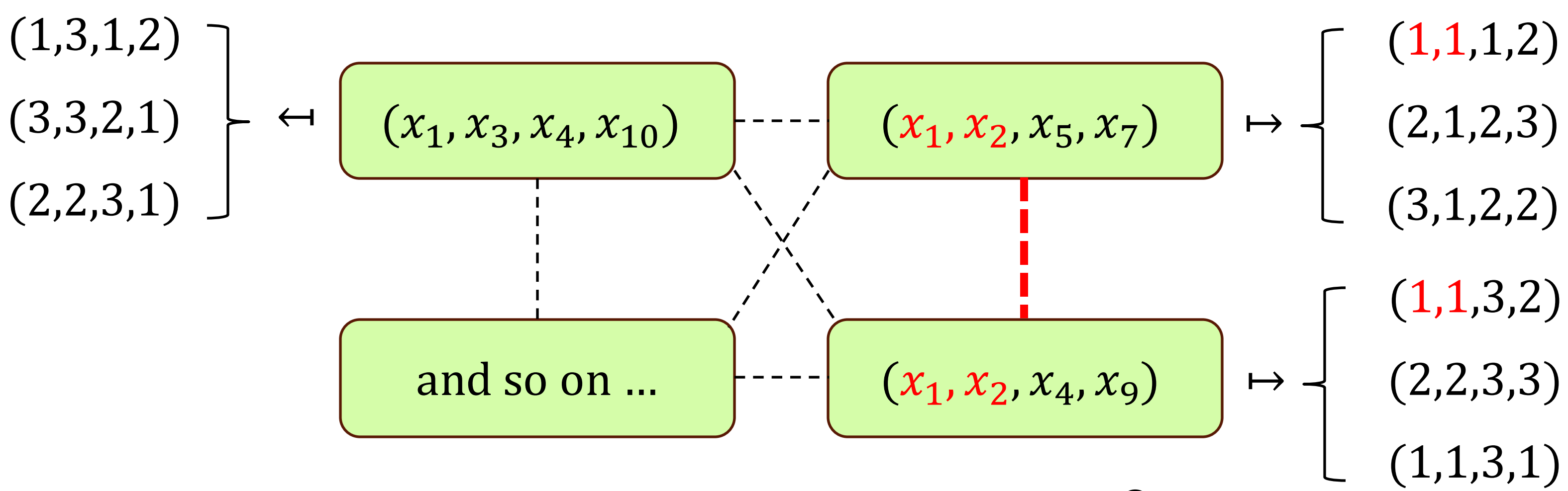

A 3-list satisfying assignment for $\Pi^{\odot 4}$

# Proof Overview



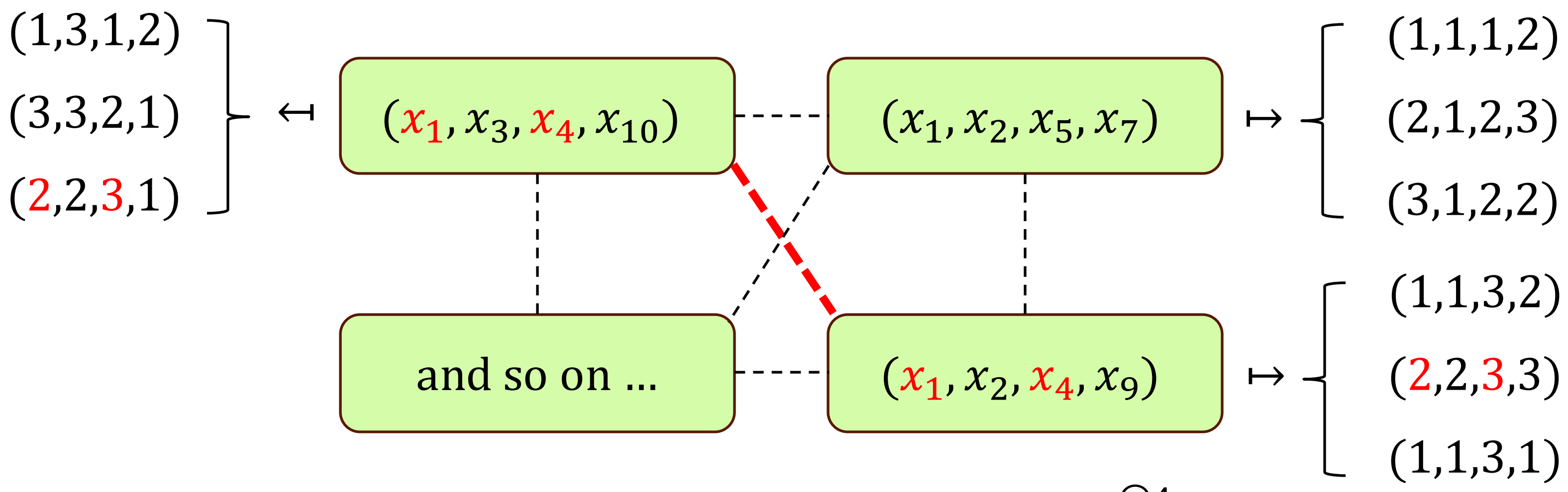

A 3-list satisfying assignment for $\Pi^{\odot 4}$

# Proof Overview




A 3-list satisfying assignment for $\Pi^{\odot 4}$

# Proof Overview




A 3-list satisfying assignment for $\Pi^{\odot 4}$

# Proof Overview

**2-CSP**
$\Pi = (X, \Sigma, \Phi)$

$\longrightarrow$

**$t$-wise Direct Product 2-CSP**
$\Pi^{\odot t} = \left( \binom{X}{t}, \Sigma^t, \Phi' \right)$

- For some sufficiently large $t = t(r)$,
  - given an $r$-list satisfying multi-assignment $\sigma$ of $\Pi^{\odot t}$,
  - want to construct an $(r-1)$-list satisfying multi-assignment $\sigma'$ of $\Pi^{\odot t'}$, for some $t' < t$.

# Proof Overview

2-CSP
$\Pi = (X, \Sigma, \Phi)$

$\longrightarrow$

$t$-wise Direct Product 2-CSP
$\Pi^{\odot t} = \left(\binom{X}{t}, \Sigma^t, \Phi'\right)$

- For some sufficiently large $t = t(r)$,
  - given an $r$-list satisfying multi-assignment $\sigma$ of $\Pi^{\odot t}$,
  - want to construct an $(r-1)$-list satisfying multi-assignment $\sigma'$ of $\Pi^{\odot t'}$, for some $t' < t$.

- If we end up with the 1-list satisfiability of $\Pi^{\odot(\geq 2)}$, then we are done!

# Proof Overview

2-CSP
$\Pi = (X, \Sigma, \Phi)$

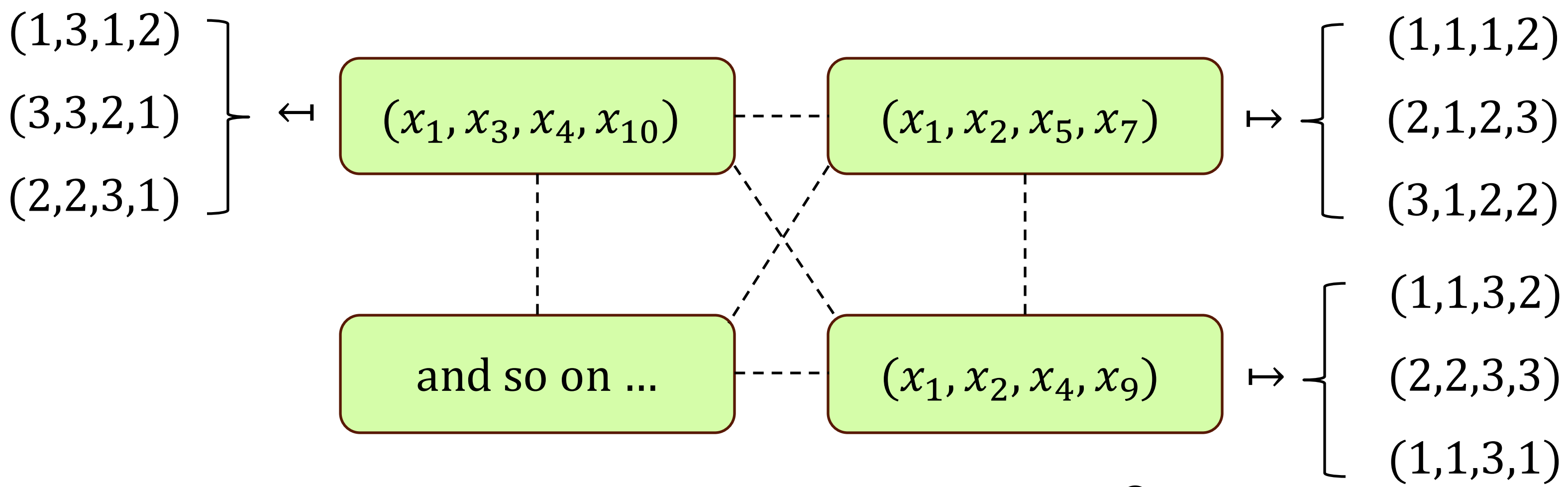
$\longrightarrow$

$t$-wise Direct Product 2-CSP
$\Pi^{\odot t} = \left( \binom{X}{t}, \Sigma^t, \Phi' \right)$

- For some sufficiently large $t = t(r)$,
  - given an $r$-list satisfying multi-assignment $\sigma$ of $\Pi^{\odot t}$,
  - want to construct an $(r-1)$-list satisfying multi-assignment $\sigma'$ of $\Pi^{\odot t'}$, for some $t' < t$.
    - for each set $S \in \binom{X}{t'}$, choose a set $T \in \binom{X}{t}$ with $S \subseteq T$
    - the list $\sigma'(S)$ is inherited from the list $\sigma(T)$ (at the hope of decreasing the list size by 1)

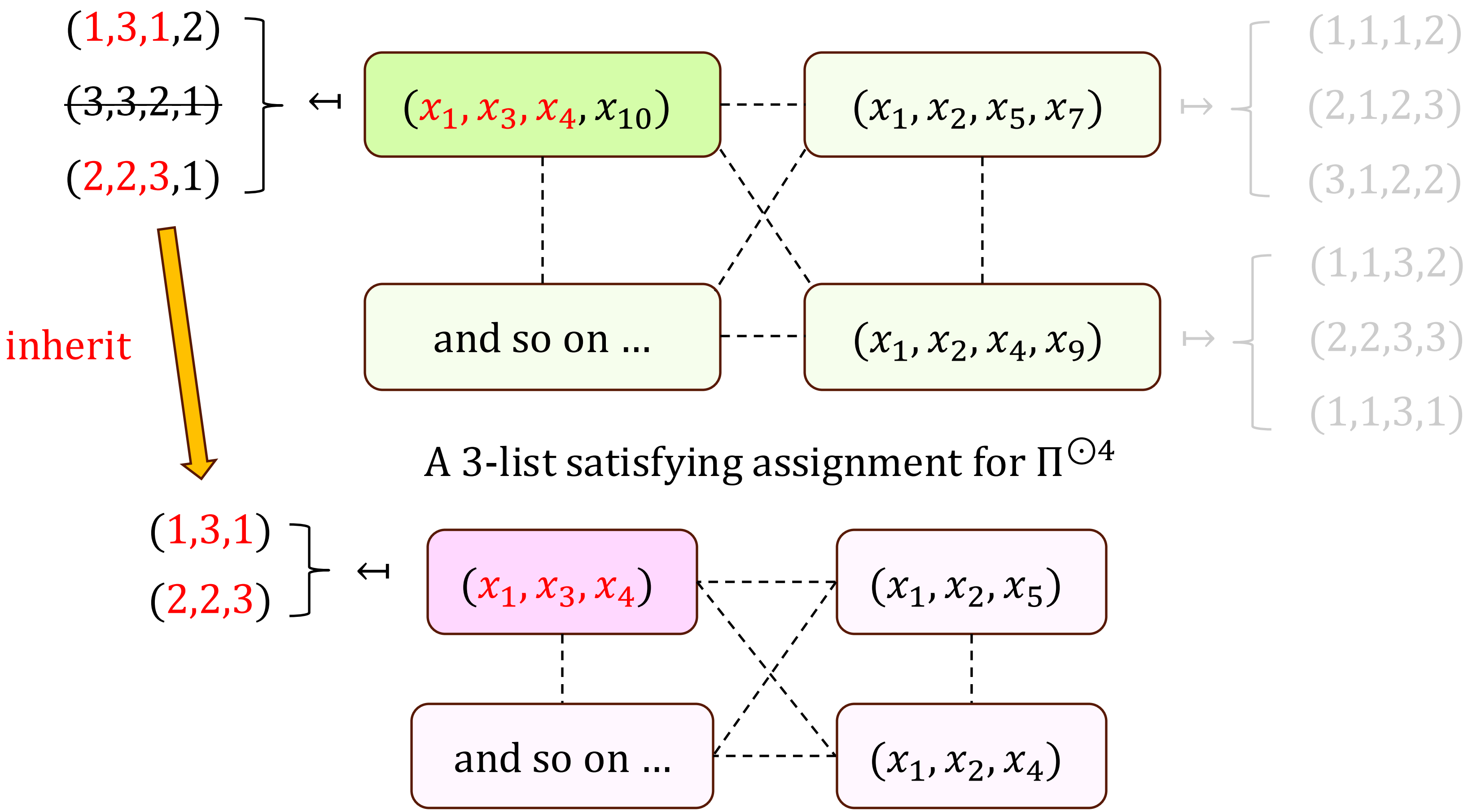
- If we end up with the 1-list satisfiability of $\Pi^{\odot(\geq 2)}$, then we are done!

# Proof Overview

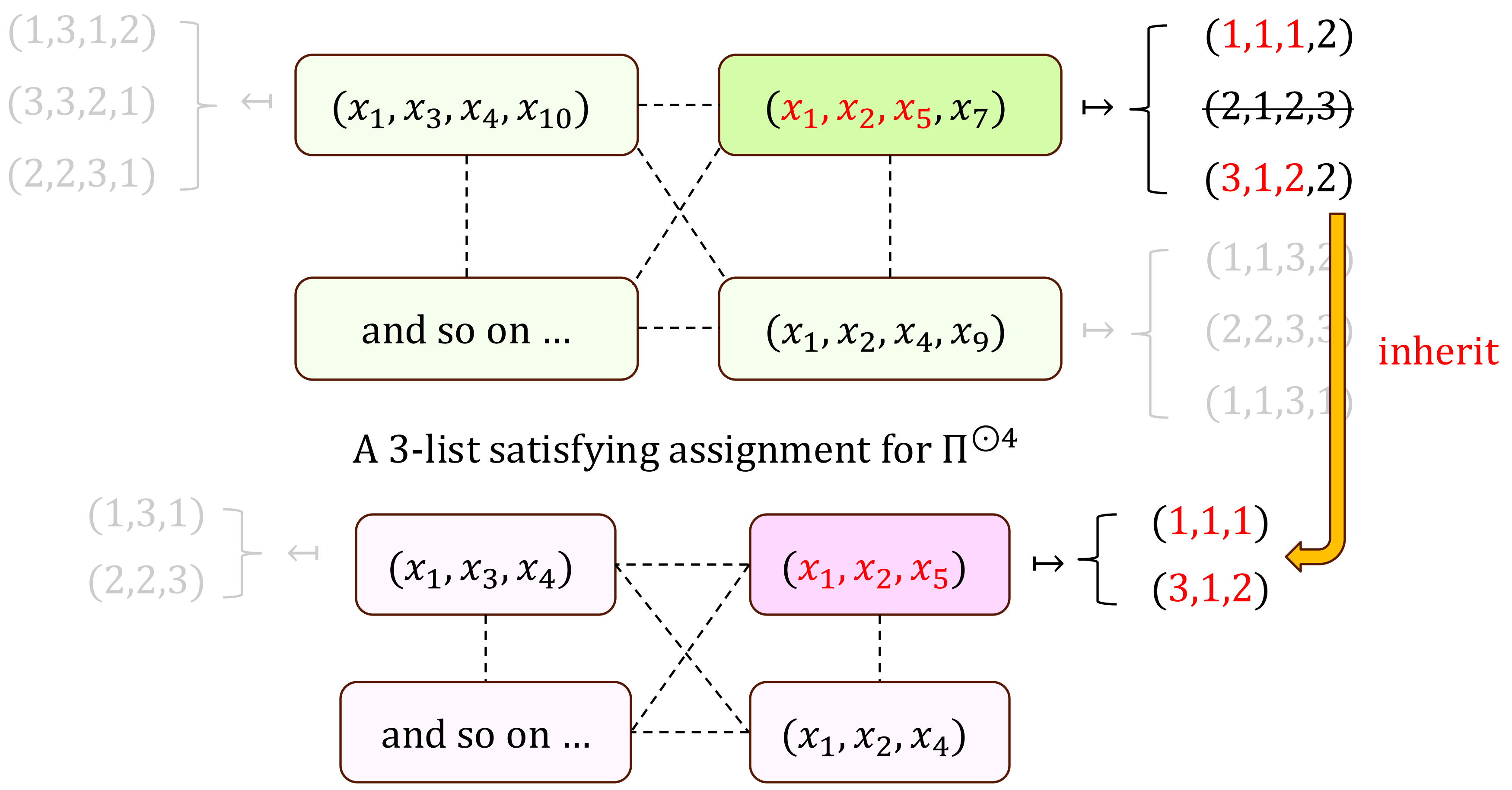



A 3-list satisfying assignment for $\Pi^{\odot 4}$

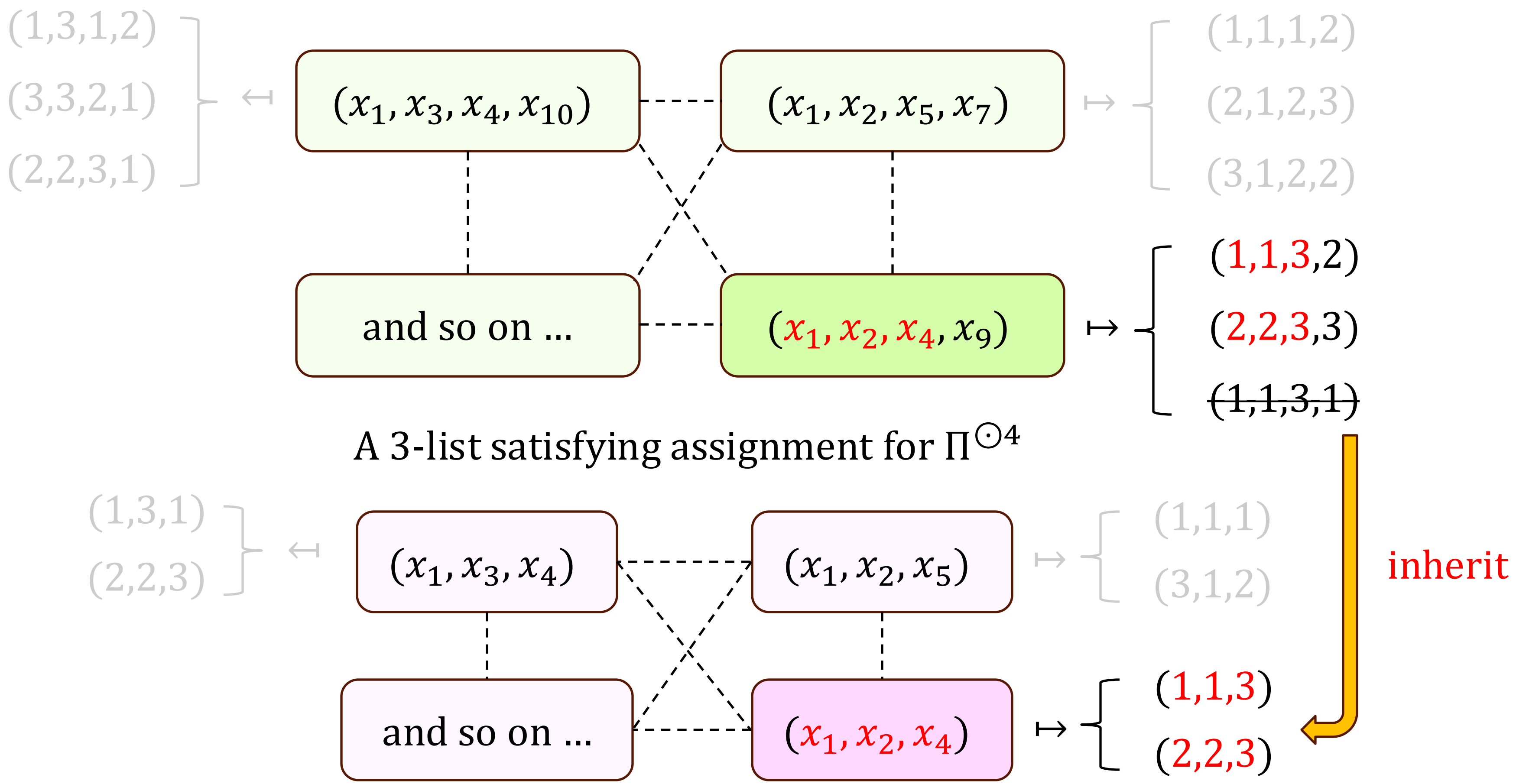
# Proof Overview



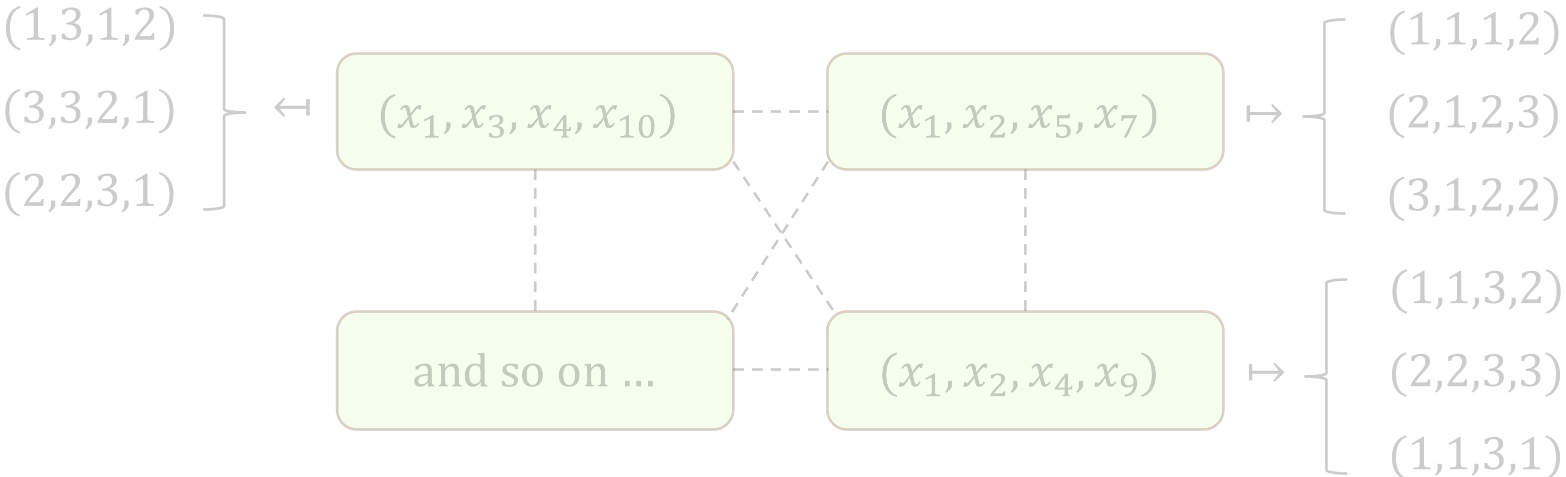
# Proof Overview



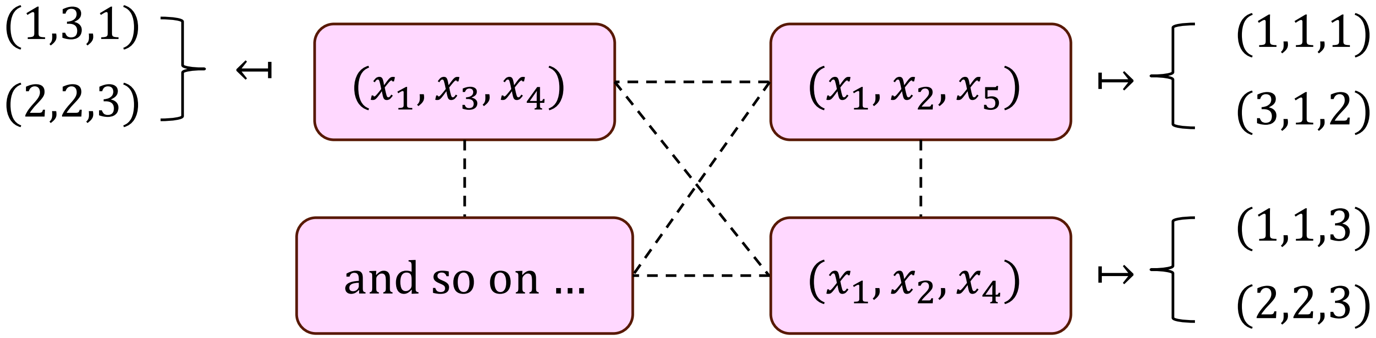
# Proof Overview



# Proof Overview




A 3-list satisfying assignment for $\Pi^{\odot 4}$



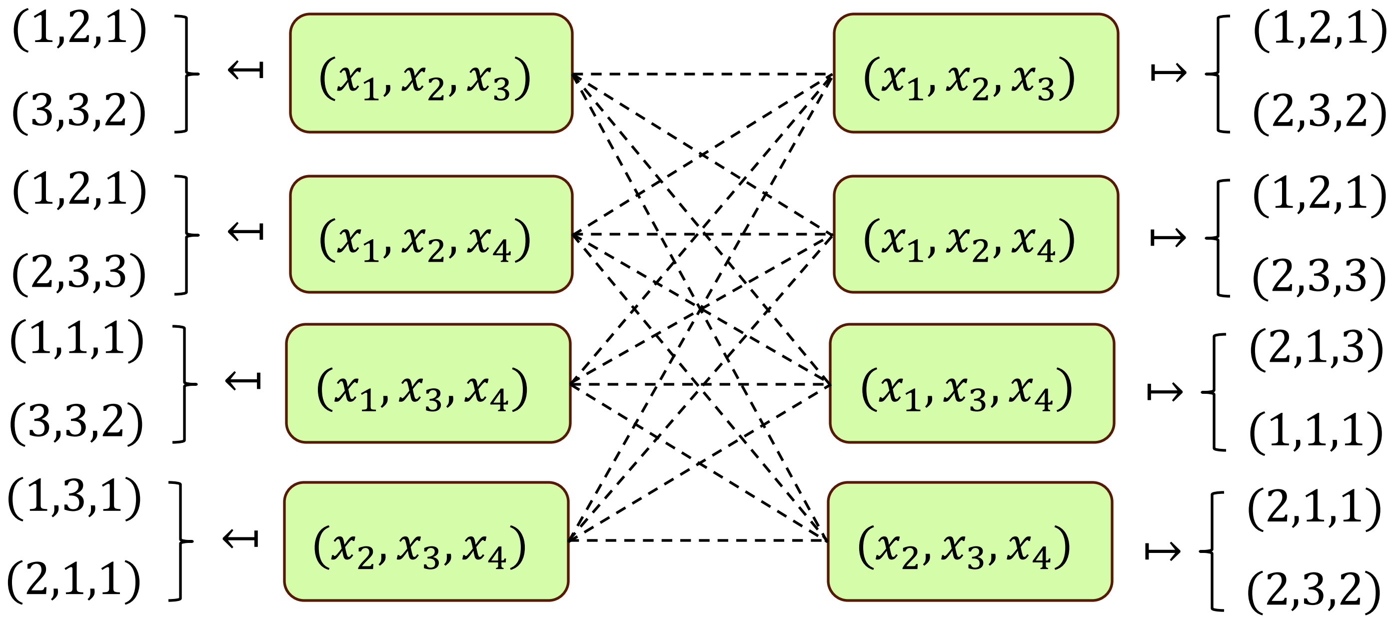

A 2-list satisfying assignment for $\Pi^{\odot 3}$?

# Proof Overview

- How can we discard one assignment safely?
  - the one that is never used to meet any consistency constraints!

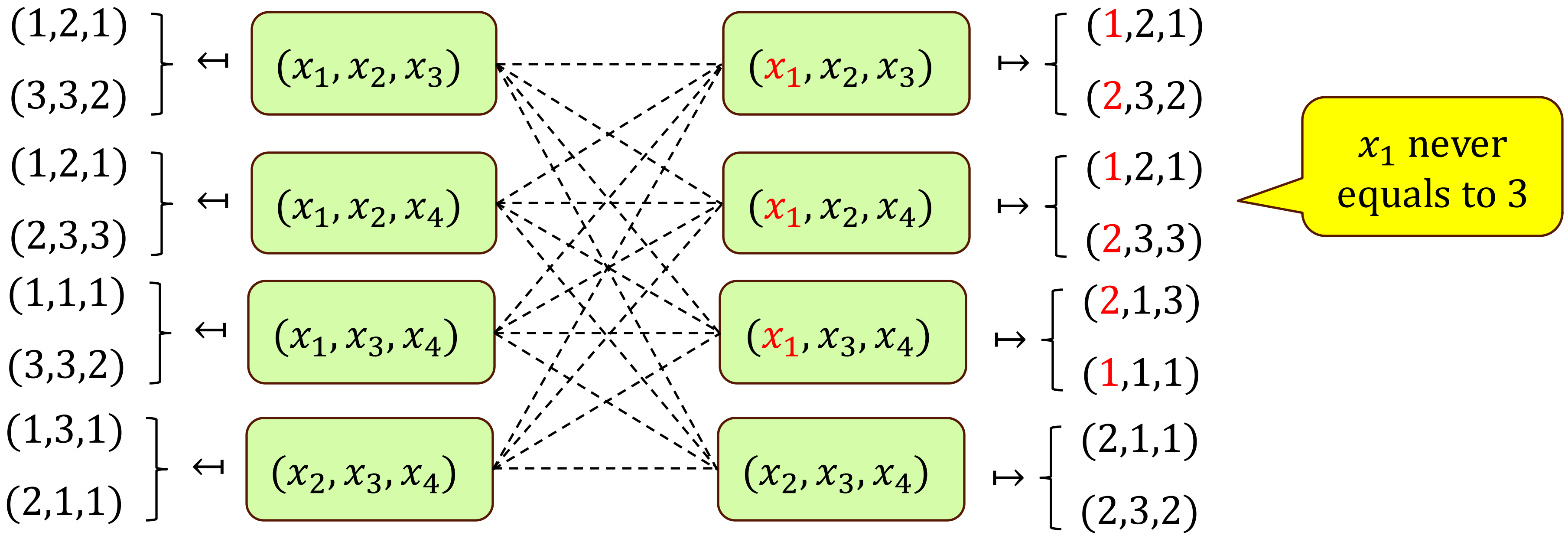
# Proof Overview

- How can we discard one assignment safely?
  - the one that is never used to meet any consistency constraints!
- Suppose we have the following *bipartite* direct product instance:



# Proof Overview

- How can we discard one assignment safely?
  - the one that is never used to meet any consistency constraints!
- Suppose we have the following *bipartite* direct product instance:

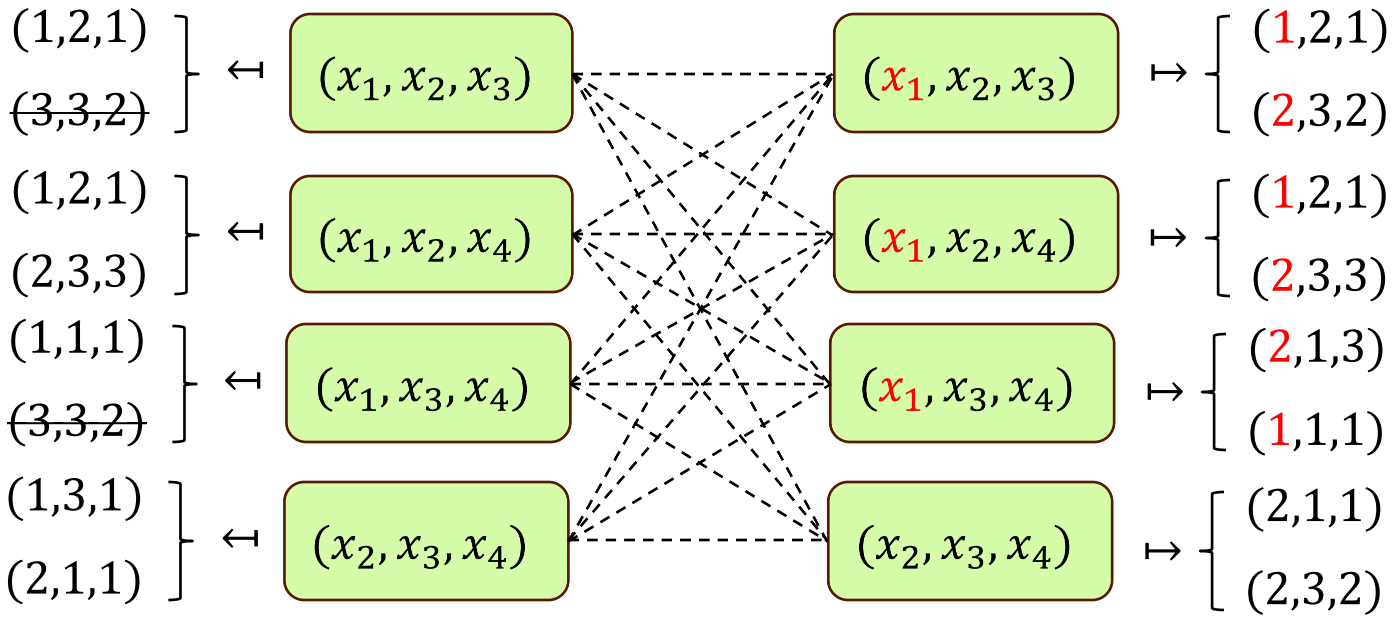


# Proof Overview

- How can we discard one assignment safely?
  - the one that is never used to meet any consistency constraints!

- Suppose we have the following *bipartite* direct product instance:

can safely remove the assignment with $x_1 = 3$



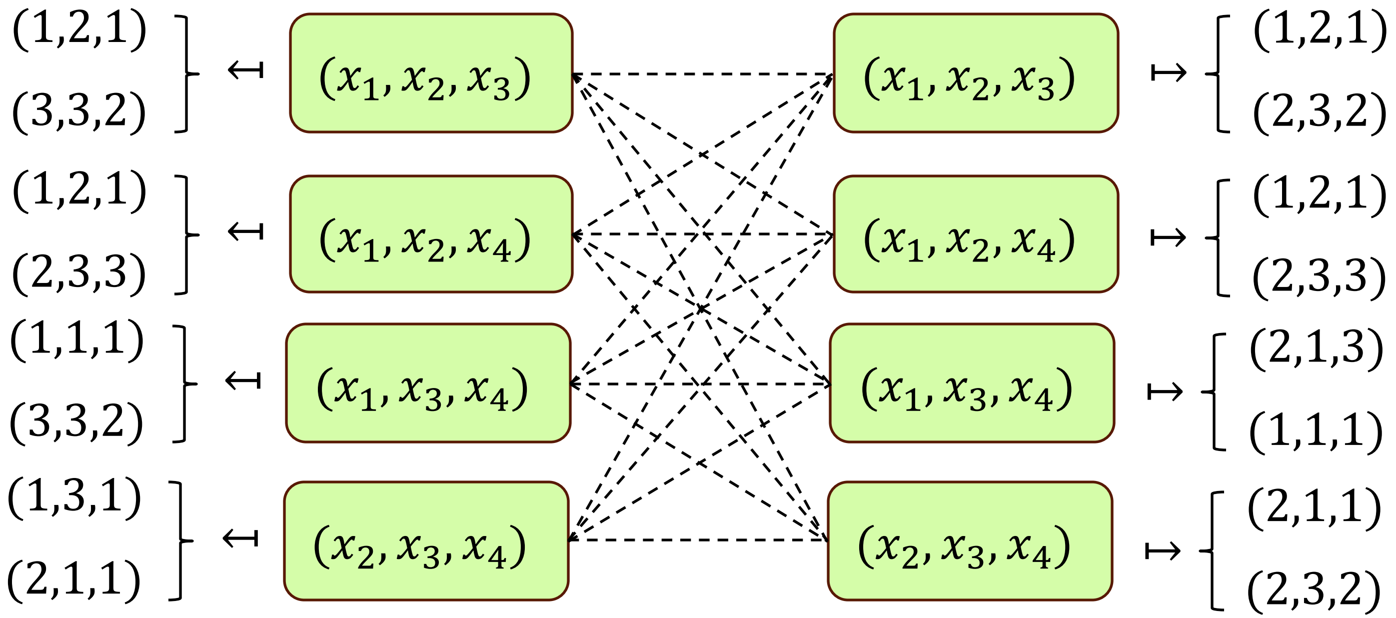

$x_1$ never equals to 3

# Proof Overview

- How can we discard one assignment safely?
  - the one that is never used to meet any consistency constraints!

- Suppose we have the following *bipartite* direct product instance:

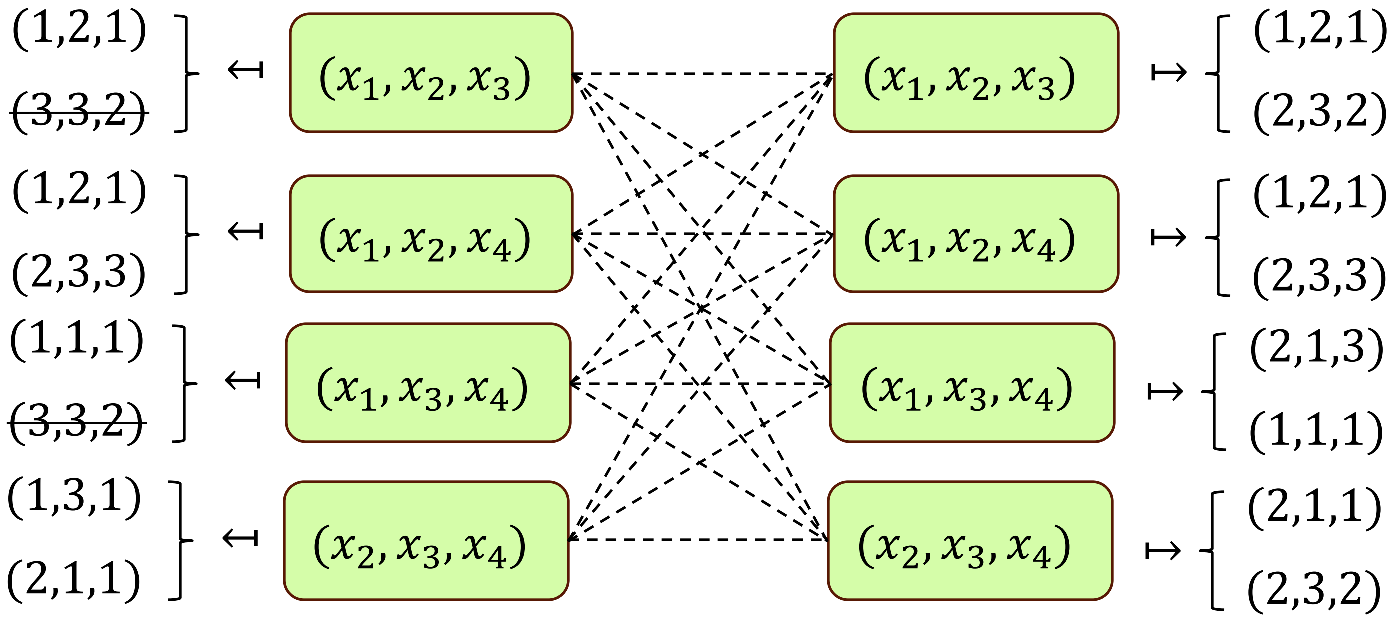
for each $S$, inherit from such a $T$ that $3 \in \sigma(T)_{|x_1}$




for each $S$, inherit from such a $T$ that $3 \notin \sigma(T)_{|x_1}$

# Proof Overview

- How can we discard one assignment safely?
  - the one that is never used to meet any consistency constraints!
- Suppose we have the following *bipartite* direct product instance:




for each $S$, inherit from such a $T$ that $3 \in \sigma(T)_{|x_1}$

by discarding this assignment, list size is decreased by 1

for each $S$, inherit from such a $T$ that $3 \notin \sigma(T)_{|x_1}$

# Proof Overview

- Bipartite $(r, 1)$-case

$$\uparrow$$

- Bipartite $(r, q)$-case

$$\uparrow$$

- Non-bipartite $r$-case

# Proof Overview

- Bipartite $(r, 1)$-case
- Bipartite $(r, q)$-case
- Non-bipartite $r$-case

# Takeaway

- Parameterized Inapproximability Hypothesis – parameterized analog of PCP
- Baby PIH – inapproximability of the list-satisfiability of (parameterized) 2CSP
  - W[1]-hard to distinguish between [1-list satisfiable] and [not even $r$-list satisfiable]
  - Proof Idea: induction on the list size

# Takeaway

- Parameterized Inapproximability Hypothesis – parameterized analog of PCP
- Baby PIH – inapproximability of the list-satisfiability of (parameterized) 2CSP
  - W[1]-hard to distinguish between [1-list satisfiable] and [not even $r$-list satisfiable]
  - Proof Idea: induction on the list size

- Average Baby PIH?
  - $\Rightarrow$ constant inapproximability of $k$-ExactCover

# Takeaway

- Parameterized Inapproximability Hypothesis – parameterized analog of PCP
- Baby PIH – inapproximability of the list-satisfiability of (parameterized) 2CSP
  - W[1]-hard to distinguish between [1-list satisfiable] and [not even $r$-list satisfiable]
  - Proof Idea: induction on the list size
- Average Baby PIH?
  - $\Rightarrow$ constant inapproximability of $k$-ExactCover
- Thanks!