How to Share an NP Statement

or

Combiners for ZK Proofs

Benny Applebaum Eliran Kachlon

Tel Aviv University

Simon's workshop on Secure Computation 2025

Riddle: The Teaching Assistant Problem

Goal: Prove a Theorem in class without giving too many details (ZK)

Given 3 TAs



Soundness:	YES	NO	YES
ZK:	NO	YES	YES

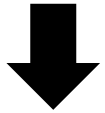
Q1: **Combine** poly-many ZK-proofs when only majority of them are sound/ZK?

- Non-interactively
- Partial solutions [GJS19,BG24,CMVX25]

Q2: How is this relevant to an MPC workshop?

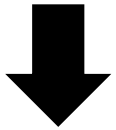
This work (TLDR)

New Funky Notion of IT-MPC



Secret-Sharing an NP-statement

- New cool notion
- Cryptographic NP-reductions
- Somewhat subtle definition



Applications: Combining ZK proofs

- Solving some open questions in ZK/MPC

How to Share a Secret [Shamir'79, Blakley'79]

t-out-of-n secret sharing:

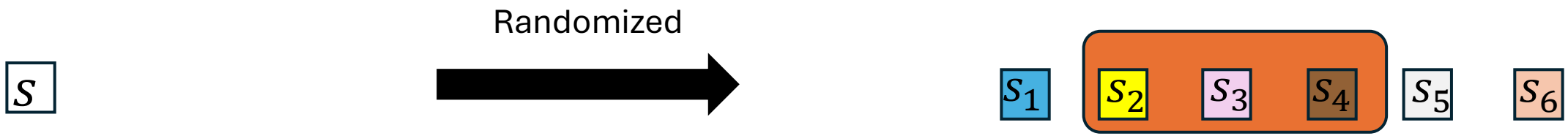
t-recovery:

t shares suffice to recover the secret

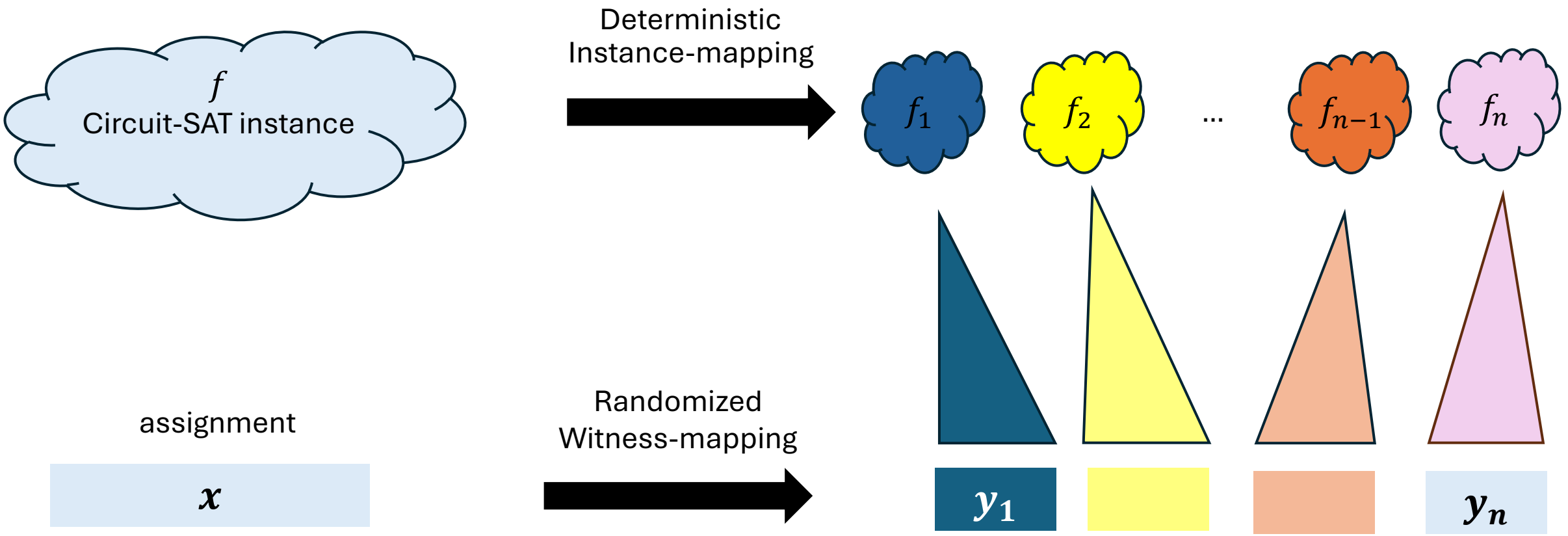
How to Share a Secret [Shamir'79, Blakley'79]

t-out-of-n secret sharing:

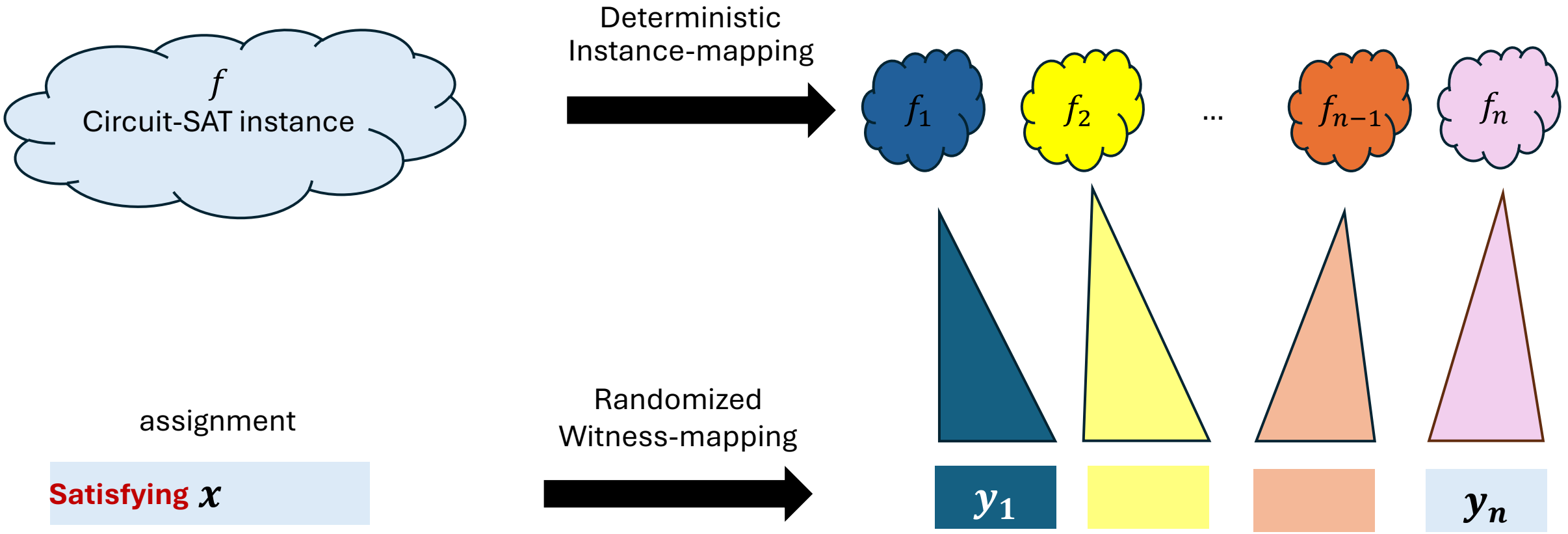
$(t-1)$-privacy:
$t-1$ shares reveal no info about the secret



How to Share an NP Statement? inspired by [Goyal, Jain, Sahai'19]

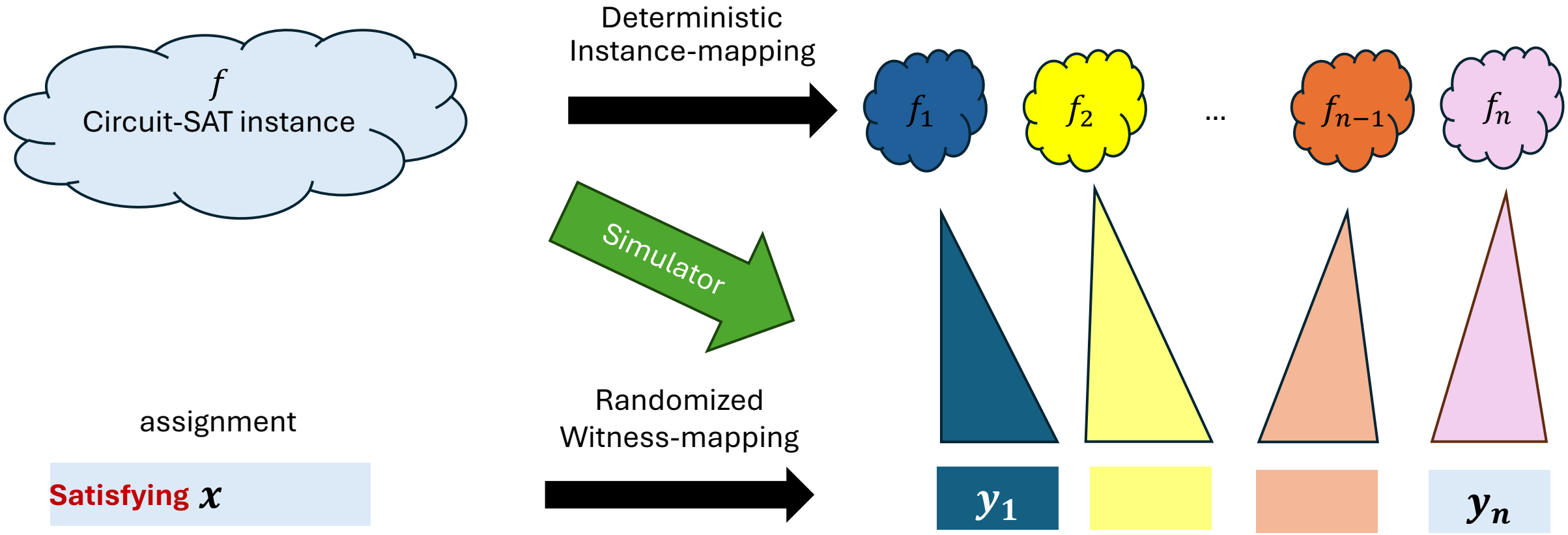


Want: **Correctness**, Privacy and Recovery



Correctness: If x satisfies f $\Rightarrow$ All assignments satisfy new circuits

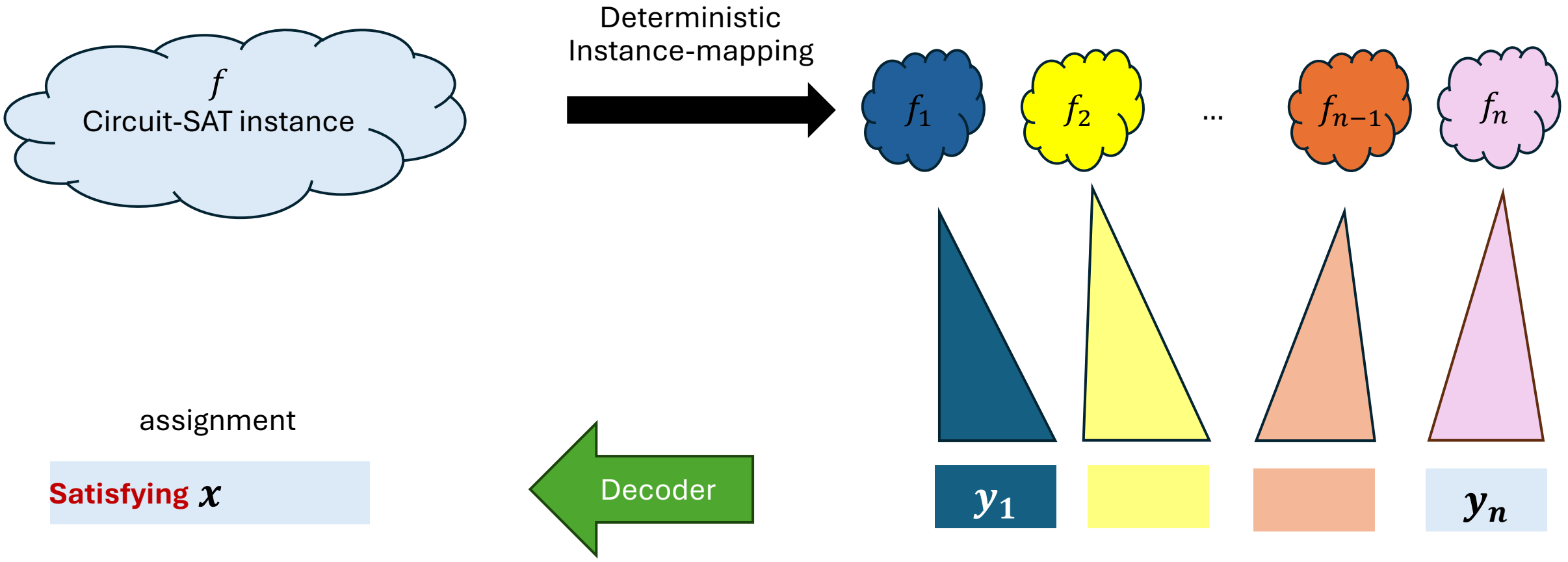
Want: Correctness, **Privacy** and Recovery



(t-1)-privacy: (t-1) subset of assignments give no information on x

- **Information-theoretically!**

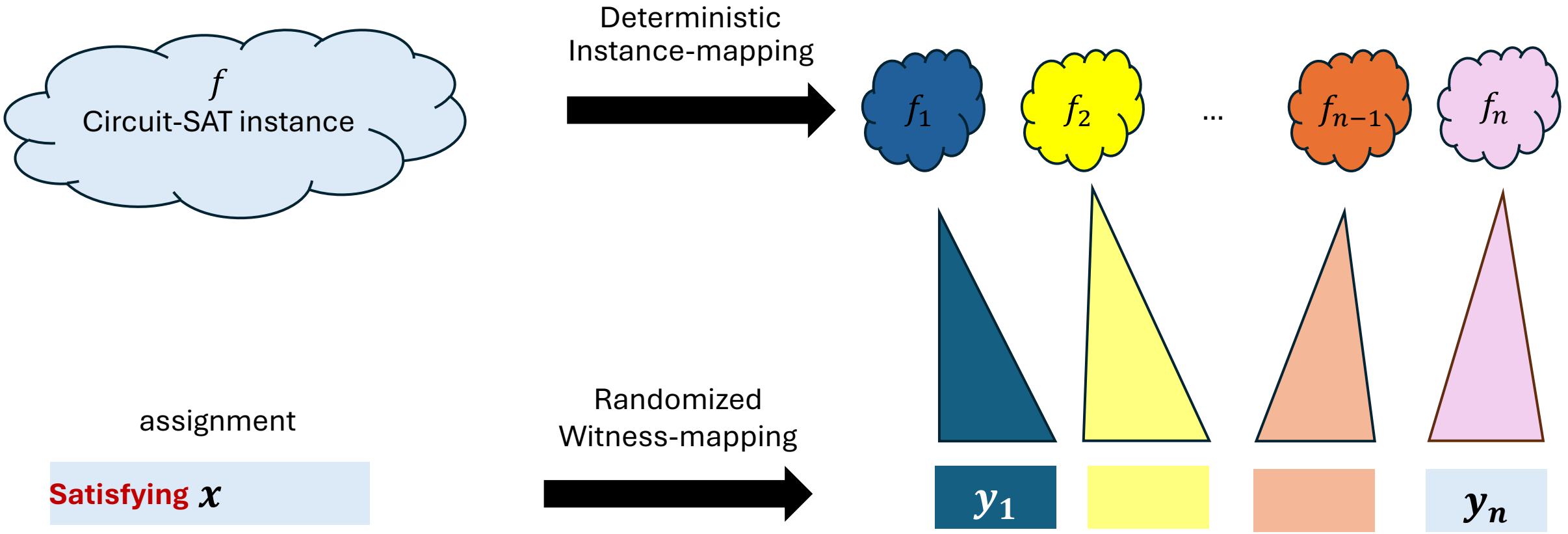
Want: Correctness, Privacy and **Recovery**



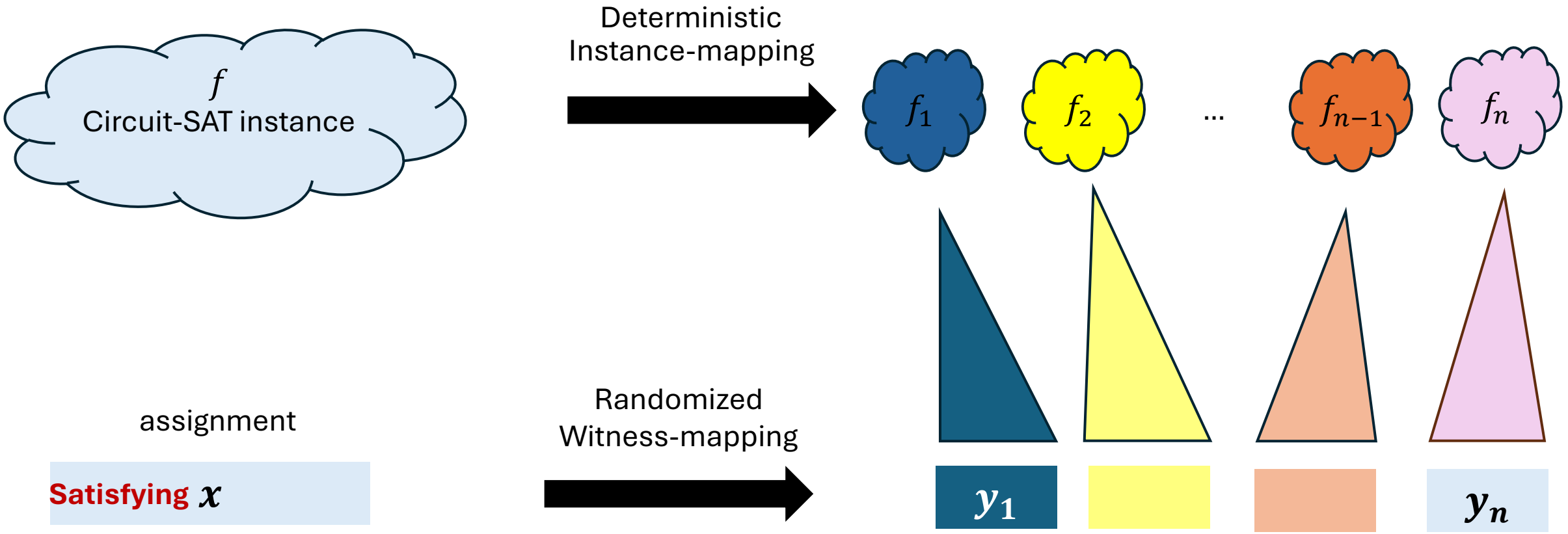
t-recovery: Given t satisfying assignments

$\Rightarrow$ Recover satisfying x

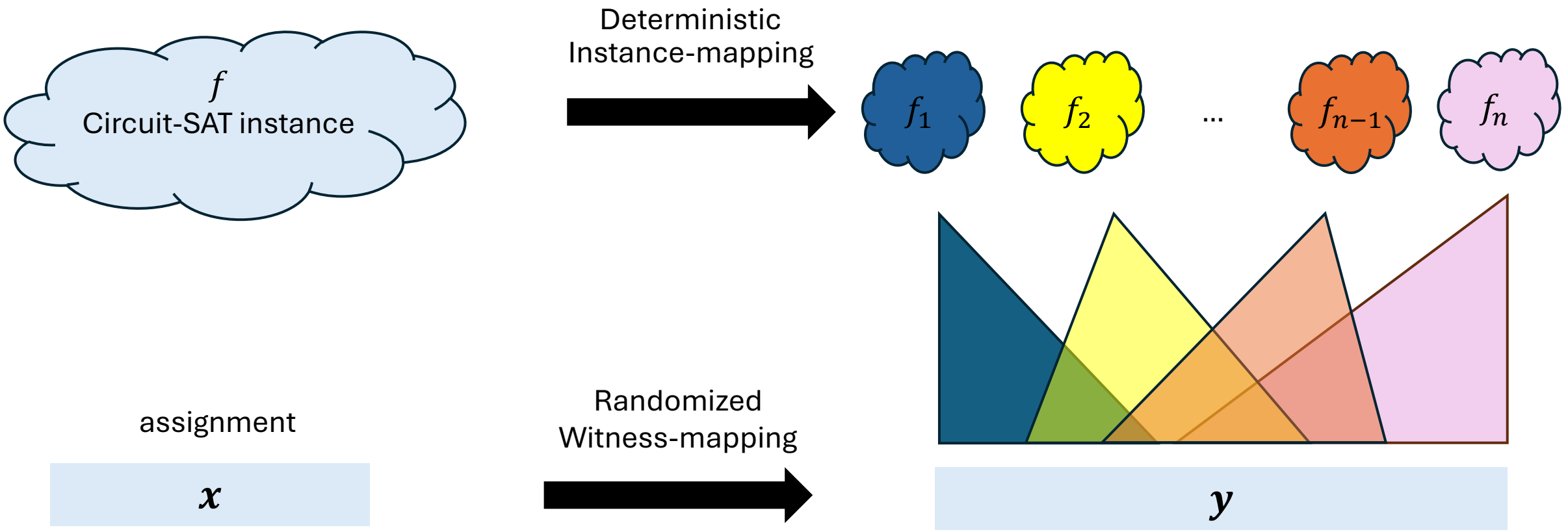
Problem:


f
Circuit-SAT instance
Deterministic
Instance-mapping
f_1
f_2
...
f_{n-1}
f_n
assignment
Satisfying x
Randomized
Witness-mapping
y_1
y_n

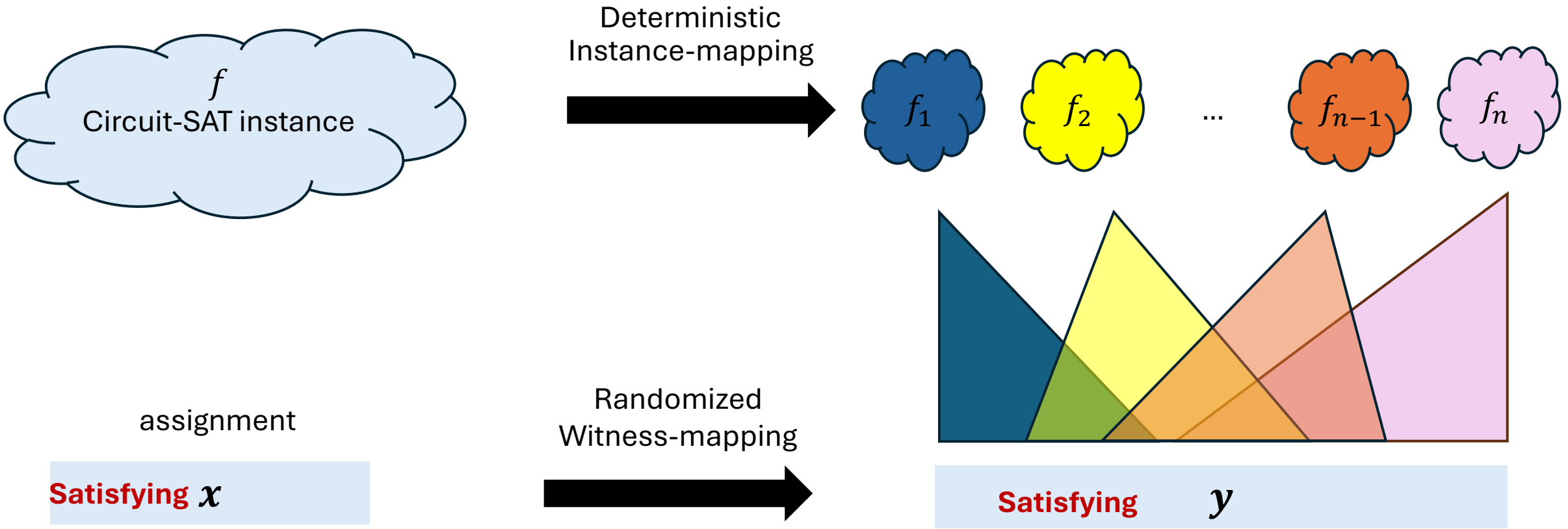
Problem: Implies P = NP



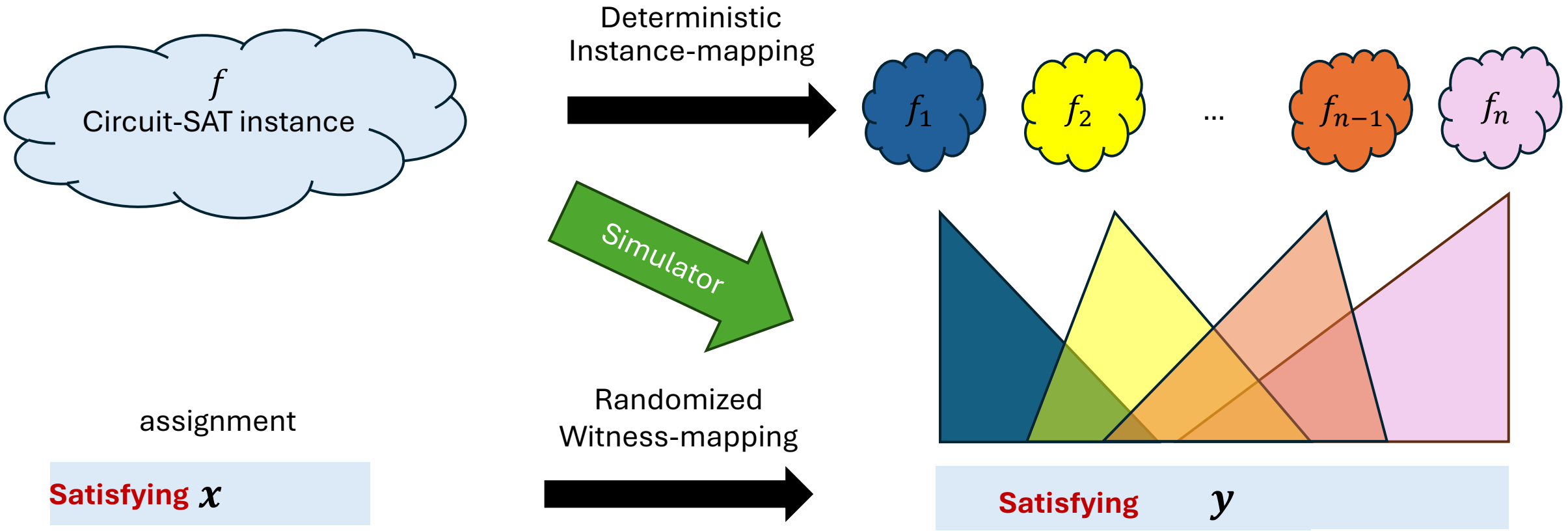
Solution: Overlapping Variables



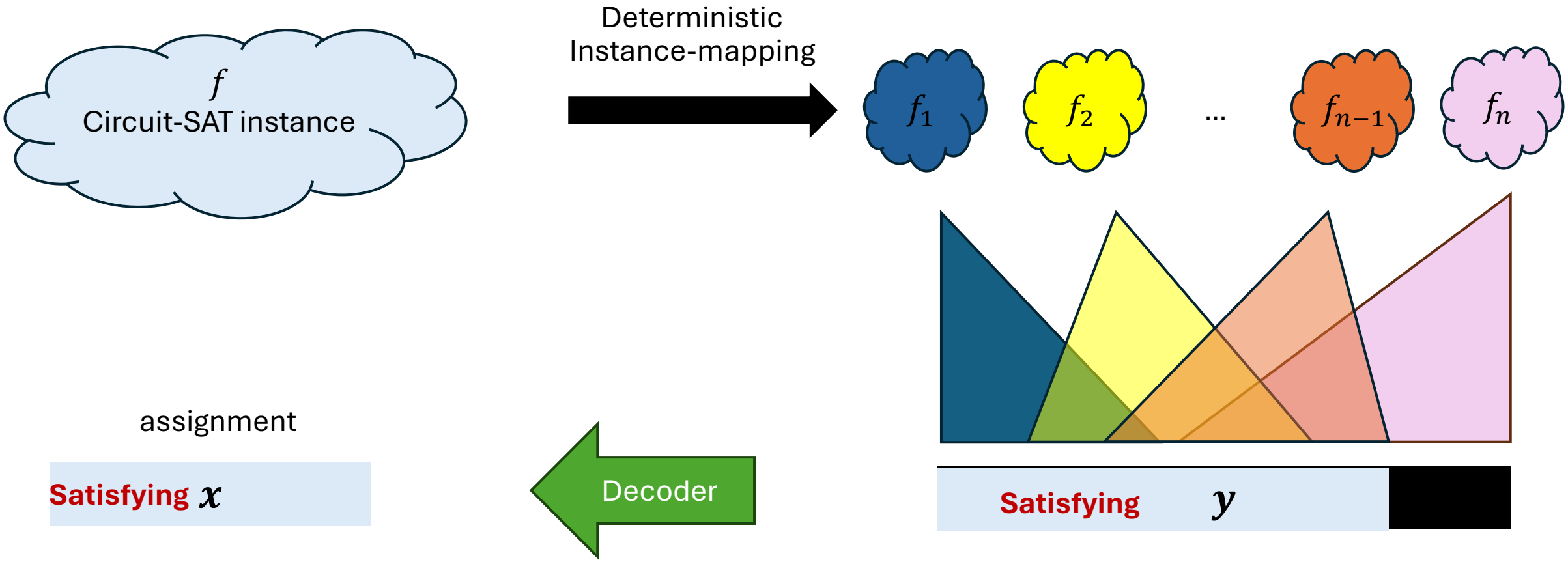
Want: **Correctness**, Privacy and Recovery



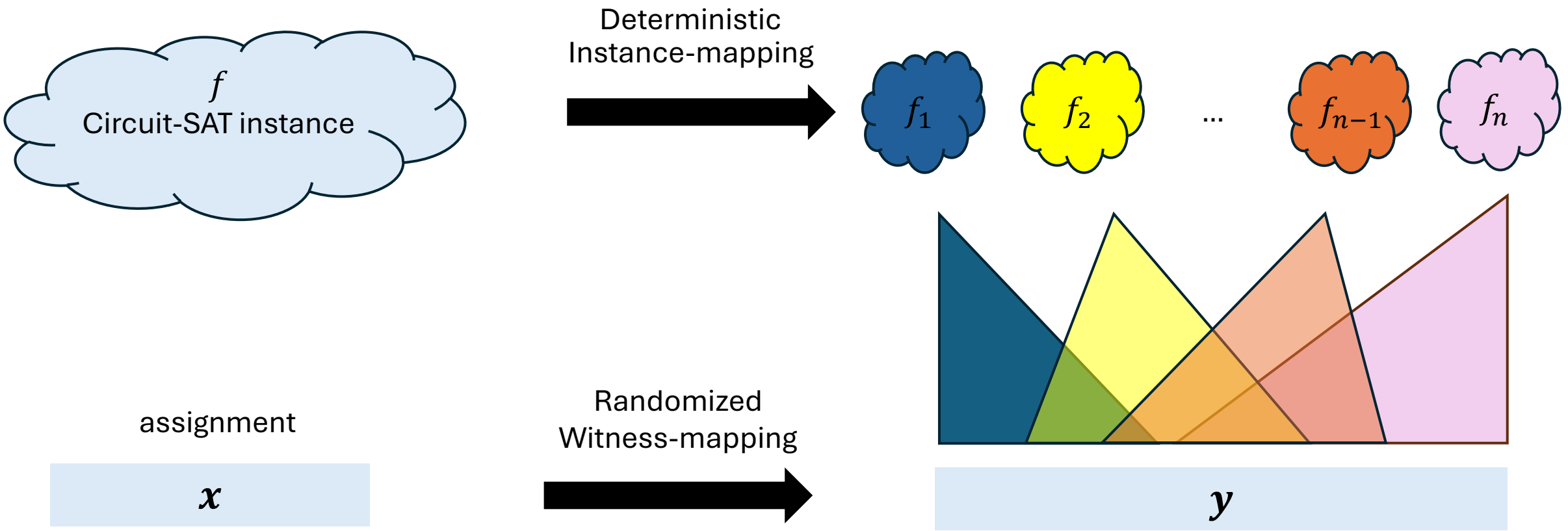
Want: Correctness, **Privacy** and Recovery



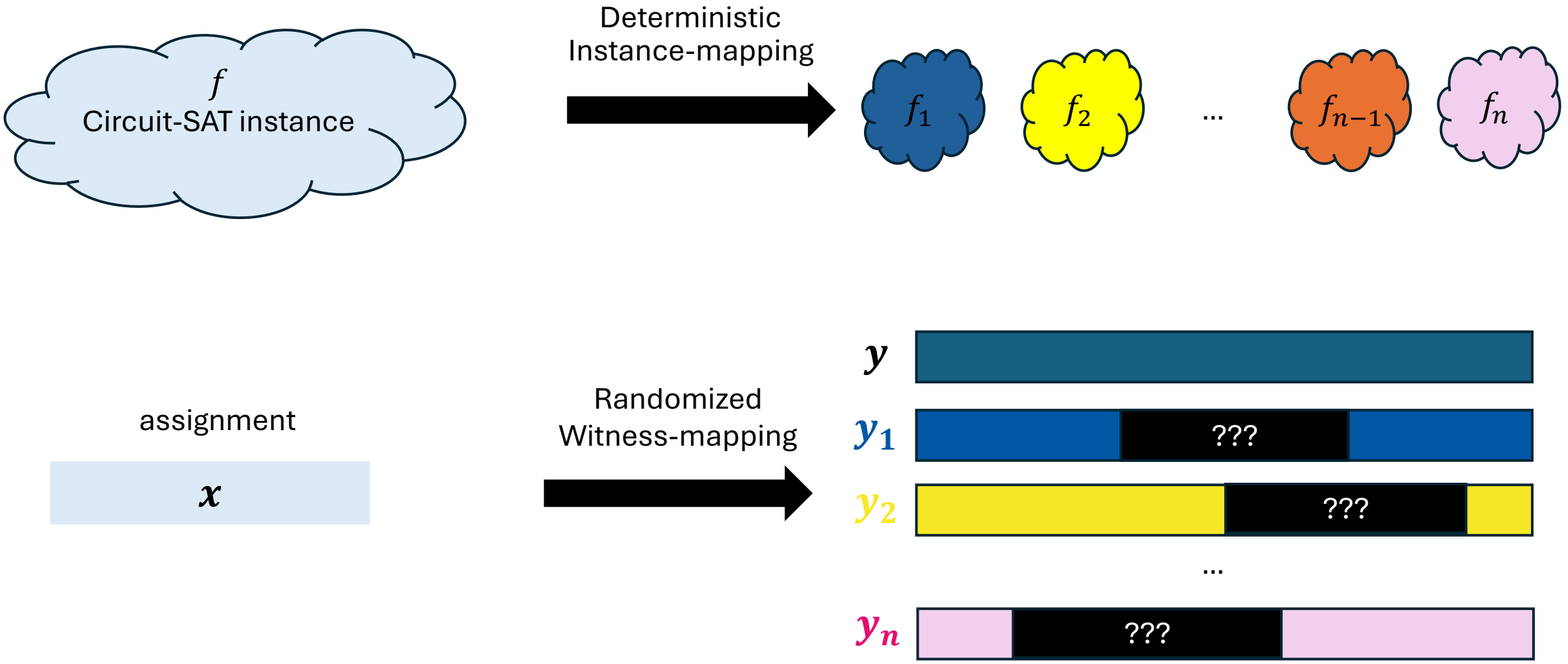
Want: Correctness, Privacy and **Recovery**



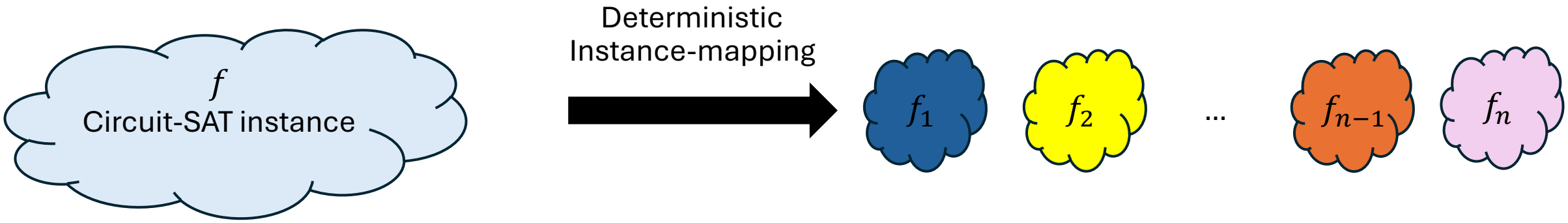
Relaxation: Partial Assignments

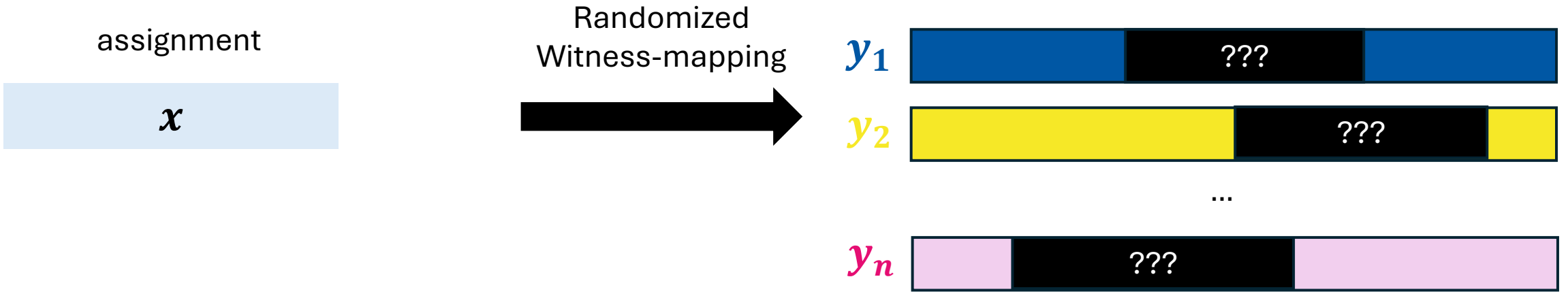


Relaxation: Partial Assignments



Relaxation: Partial Assignments

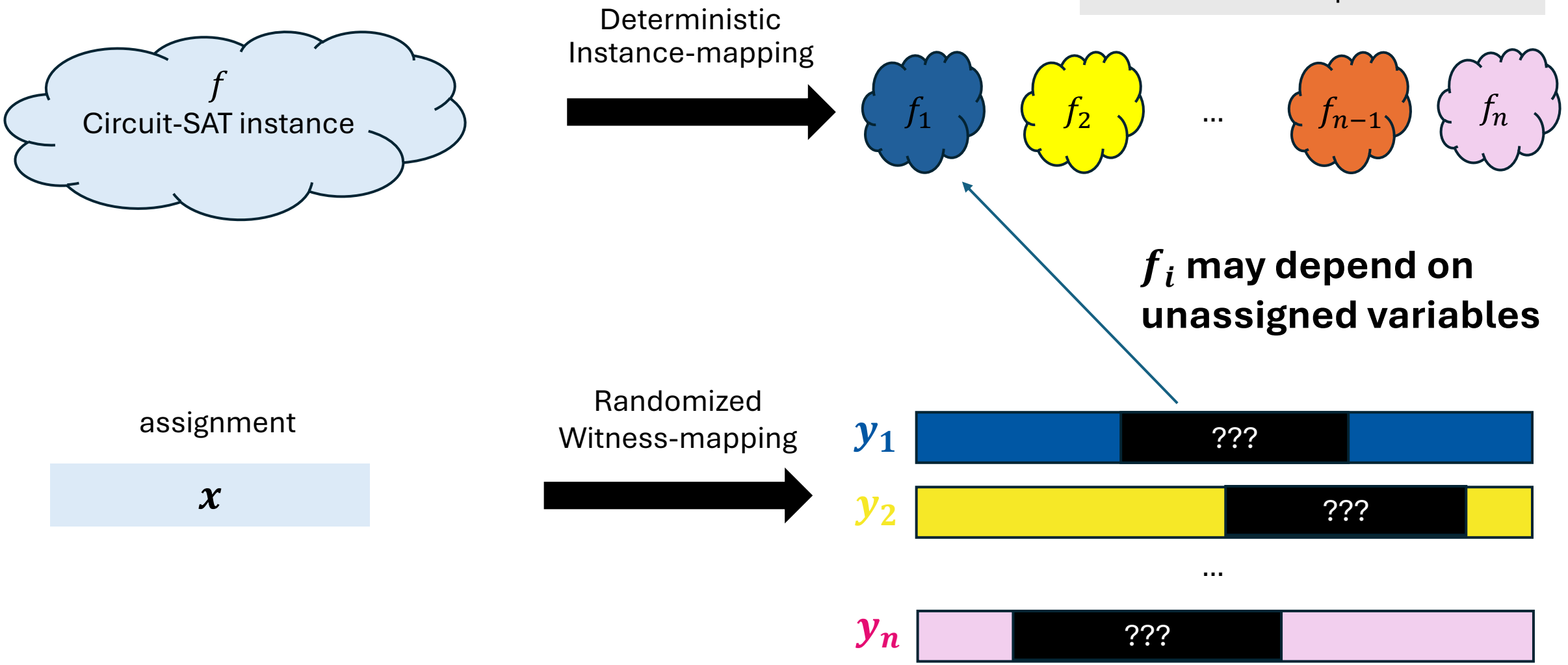




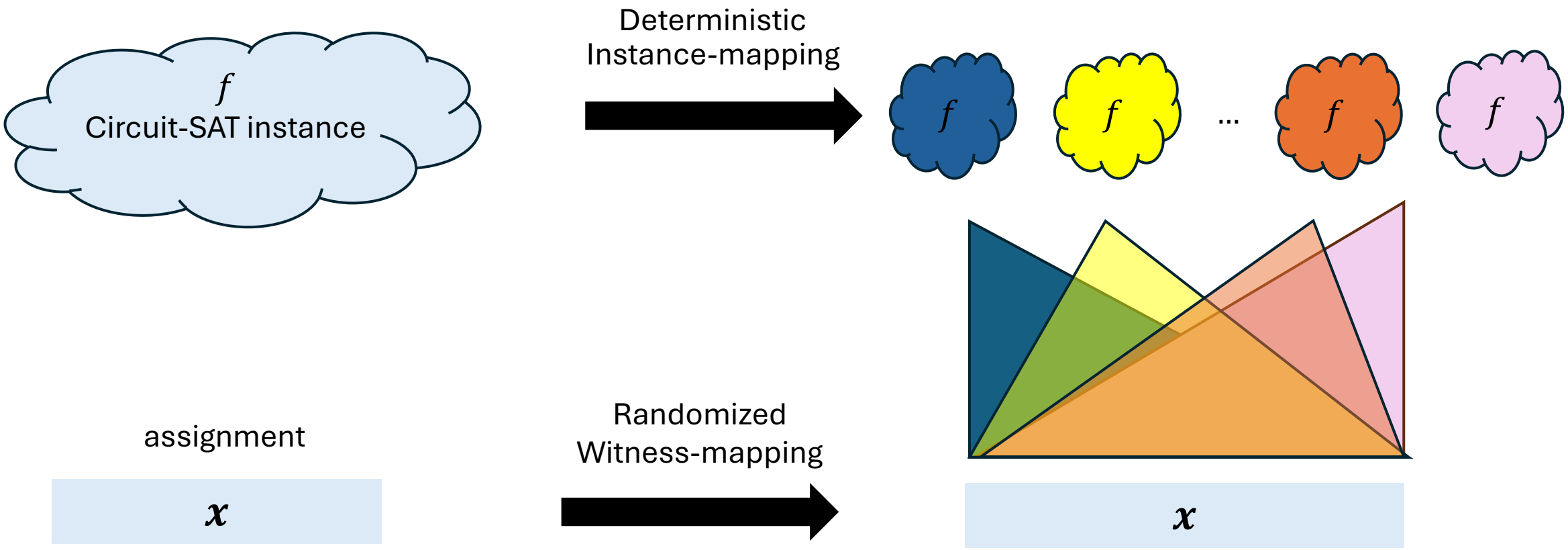
Relaxation: Partial Assignments

Evaluating partial assignments:
- $1 \vee ? = 1$
- $0 \wedge ? = 0$
- Otherwise: result is ?

Circuit is sat if output is 1

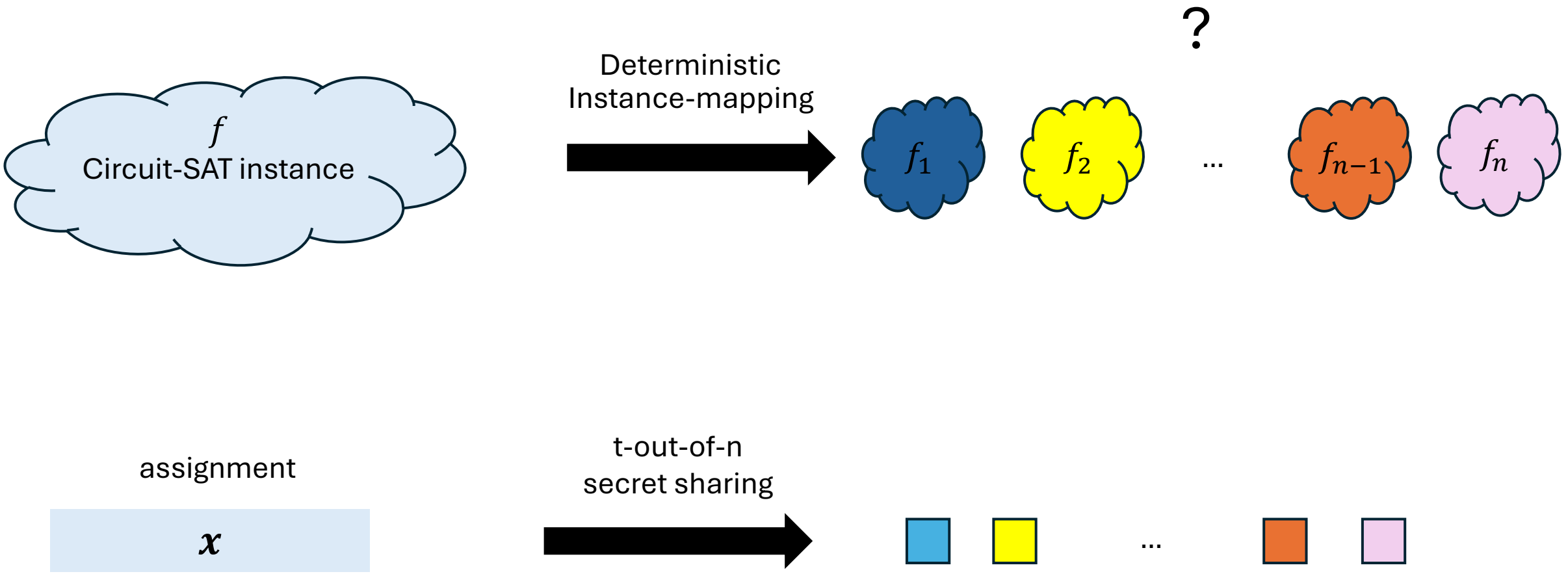


Naïve Attempt 1: Repetition



Correctness and Recovery hold but no Privacy !

Naïve Attempt 2: Secret-Share x



Privacy holds but Impossible to define f_i (unless P=NP)

NPSS vs. ZK-PCP [KPT97]

Can be abstracted using the same framework

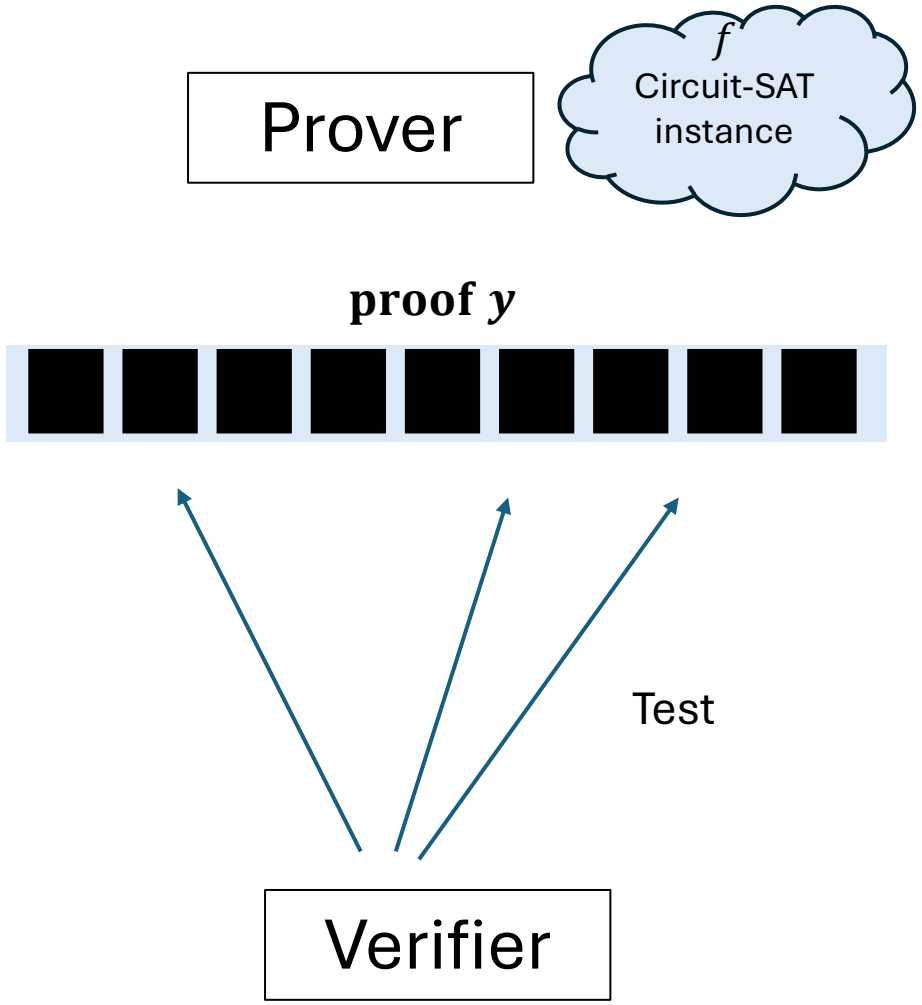
But optimizes different parameters

ZK-PCP:

- Optimize communication (small alphabet, small q)
- Deal with malicious verifier maximize number of queries q'

t-out-of-n NPSS: Verifier sample t-1 random f_i 's as a test

- Perfect completeness
- Perfect "honest-verifier" ZK
- Soundness error $1/\binom{n}{t-1}$
- Large communication – q is very large
- Optimal ZK-vs-Soundness tradeoff



ZK-PCP:
δ-Soundness for q queries
ZK for q' queries

Unrelated Notions

Secret-sharing for NP access structure [Komargodski, Naor, Yogev'14]

- Share data
- NP problem defines authorized sets

NPSS

- Share NP-statement (and witness)
- authorized sets: all sets of size t (threshold access structure)

Main Result: t-out-of-n NPSS

Thm. For every $t \leq n$ there is a t-out-of-n NPSS

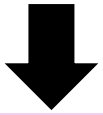
- Perfect correctness, privacy and recovery
- Running time $poly(n, |f|)$

Previously [GJS19]:

- Computational NPSS
- n-out-of-n NPSS
- $n/3$-privacy $2n/3$-soundness

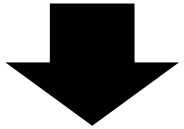
Main Result: t-out-of-n NPSS

New Notion of MPC



Thm. For every $t \leq n$ there is a t-out-of-n NPSS

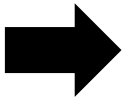
- Perfect correctness, privacy and recovery
- Running time $poly(n, |f|)$



Applications: "strong" ZK combiners

Multi-string NIZK

Multi-Verifier Offline/Online ZK



General MPC
- honest-majority
- active GOD security
- 3 rounds (optimal)

from **One-Way Function**

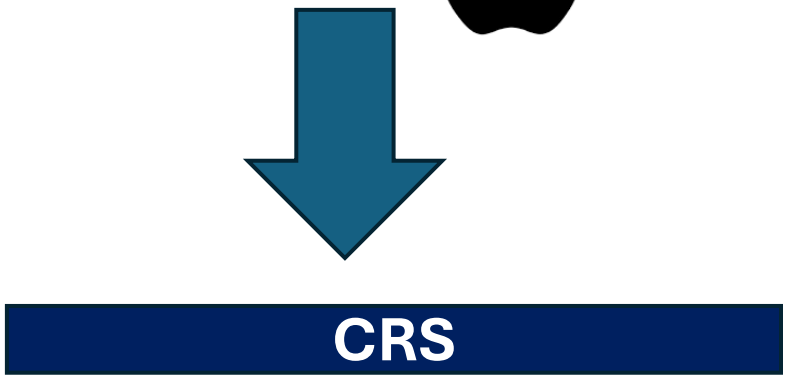
Previously:
- TFHE+CRS+NIZK [GLS15]
- LWE [BJMS20]
- ZAPs+PKE [ACGJ18]
- sub-exp. 1-1 OWF [AKP22]

few parties

Applications:
Multi-String NIZK

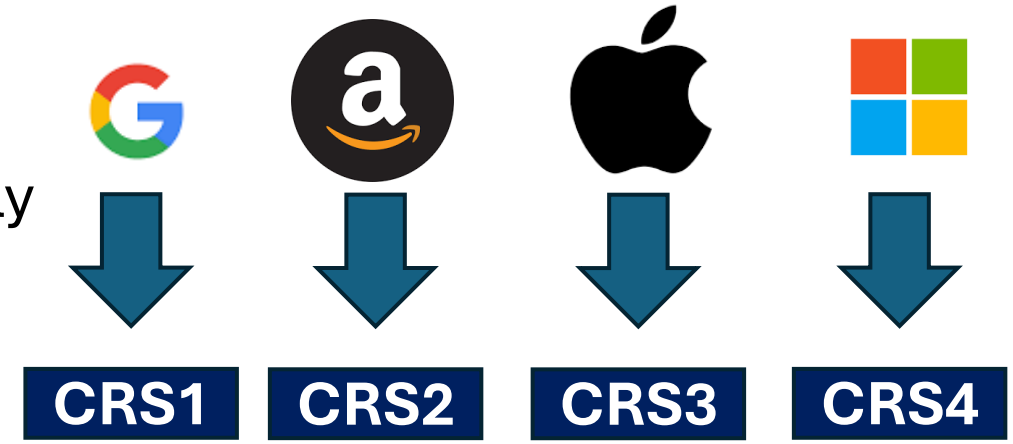
Non-Interactive ZK [BFM'88]

- BAD CRS can violate soundness/ZK
- Standard Sol: Distribute trust
  - Secure multiparty computation
- Problem: Expensive
  - Interactive…
  - High communication and computation
    especially for structured CRS



Multi-String NIZK [Groth, Ostrovsky'14]

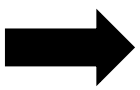
- Split trust in non-interactive way
  - Every entity generates a CRS independently
  - Security if honest majority of entities



NIZK over uniform CRS

+

One-Way Functions



Multi-String NIZK

Prover  Verifier

OPEN: What about general NIZK with structured CRS?

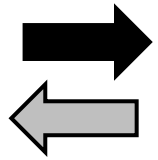
- Captures many useful settings

Our Results: MS-NIZK

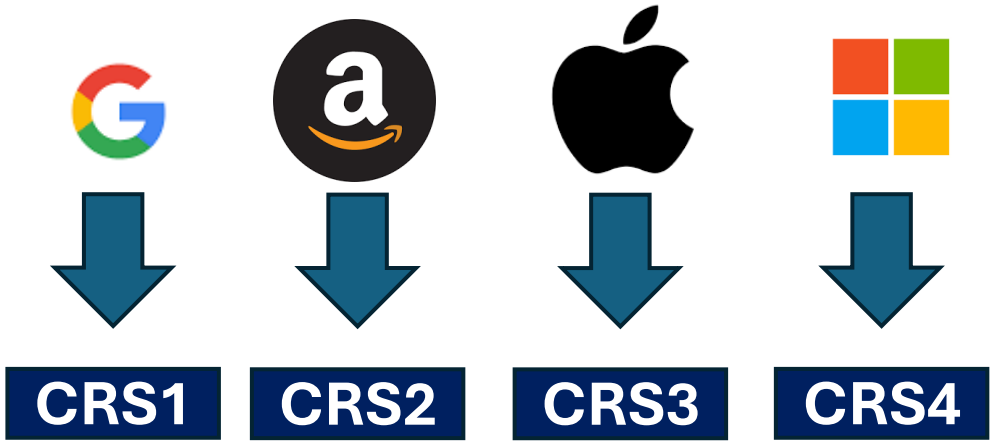
NIZK over CRS

+

One-Way Functions



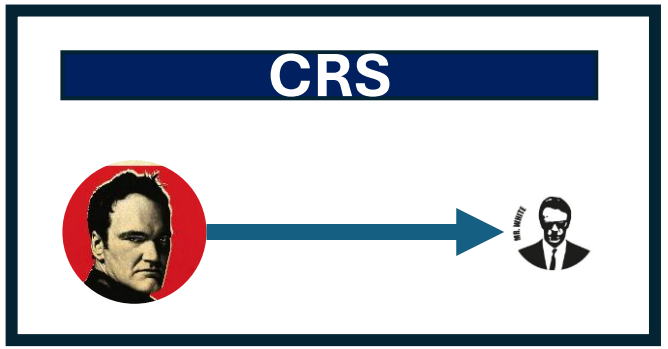
Multi-String NIZK



- Preserve special properties:
  - Stat. soundness
  - Offline/online Simulation
  - PoK + stat. ZK (Requires CRH)
  - Succinctness (Requires CRH)
  - ⋮

Prover → Verifier

Our Construction



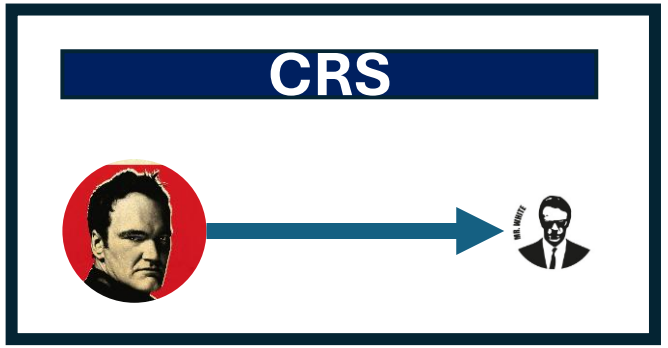
Prover

$f, \boldsymbol{x}$

Verifier

f

Our Construction



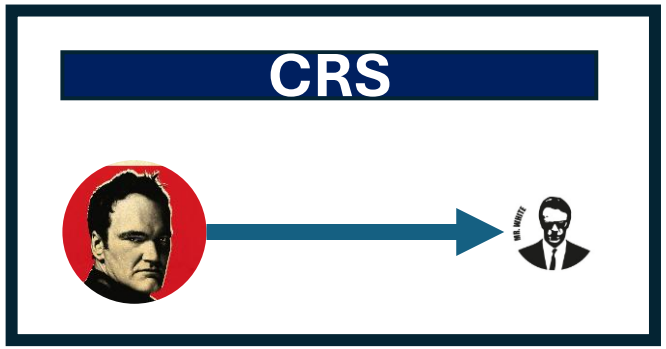
Prover

$f, \boldsymbol{x}$

Verifier

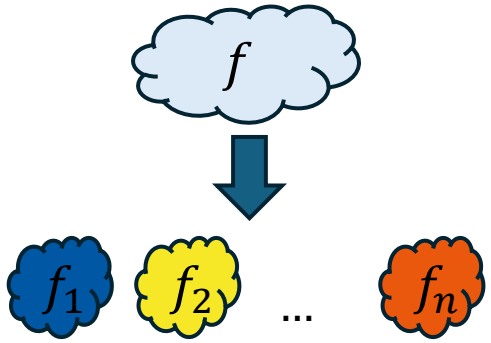
f

Our Construction



$\left\lfloor \frac{n+1}{2} \right\rfloor$-out-of-$n$ NPSS

Privacy against any minority
Recovery for any majority



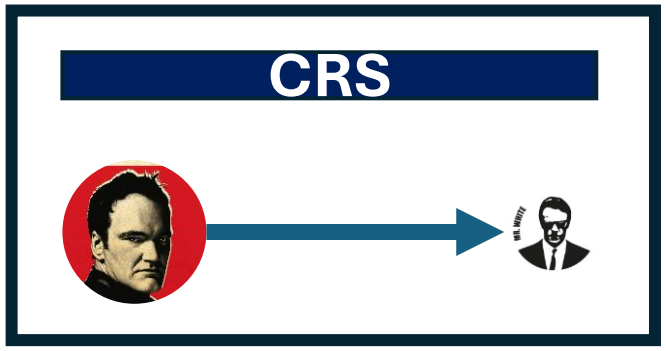
Prover

$f, \boldsymbol{x}$

Verifier

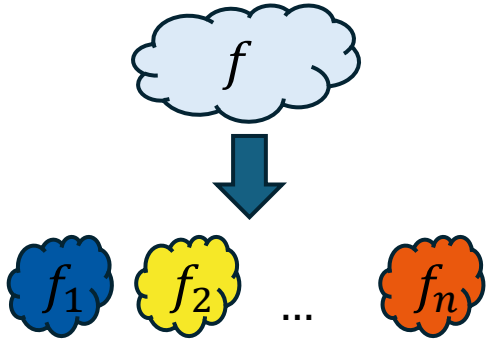
f

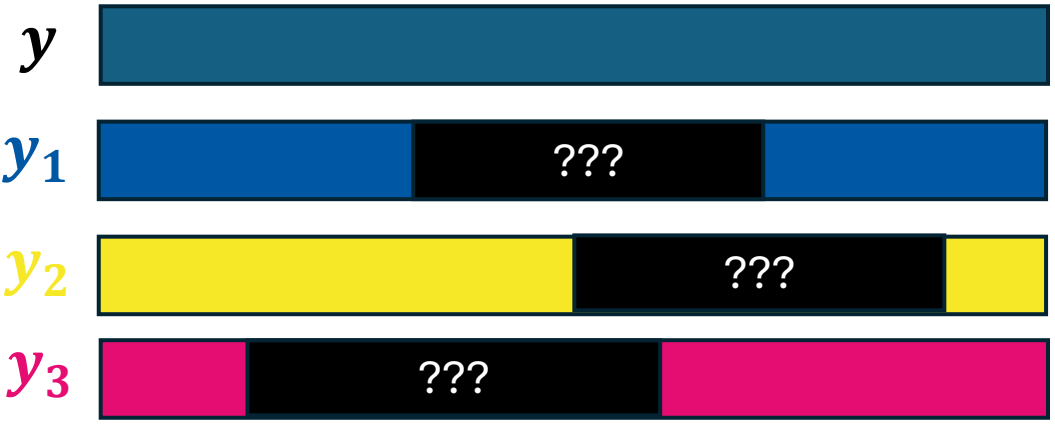
Our Construction



$\left\lfloor \frac{n+1}{2} \right\rfloor$-out-of-$n$ NPSS

Privacy against any minority
Recovery for any majority





Prover

$\vec{C}$: Commit to every bit of y

$\pi_1, \ldots, \pi_n$

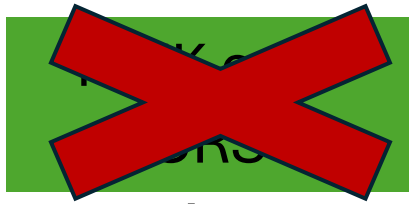
Verifier

f

f, x

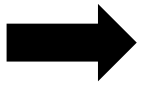
π_i Uses CRS_i to prove:
$\exists y_i$ that satisfies f_i and is consistent with $\vec{C}$

Extension



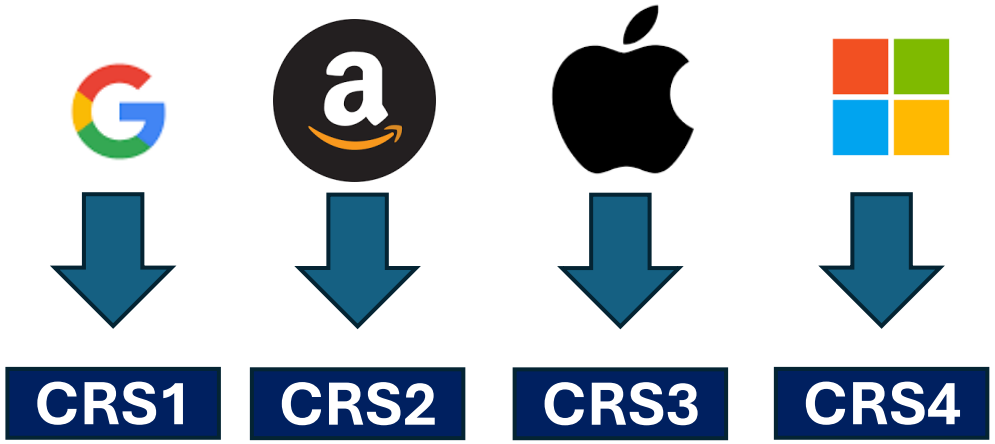
+

One-Way Functions



Multi-String NIZK

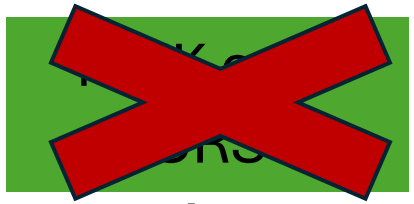
Designated Prover



Private message

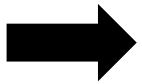
Prover → Verifier

Extension



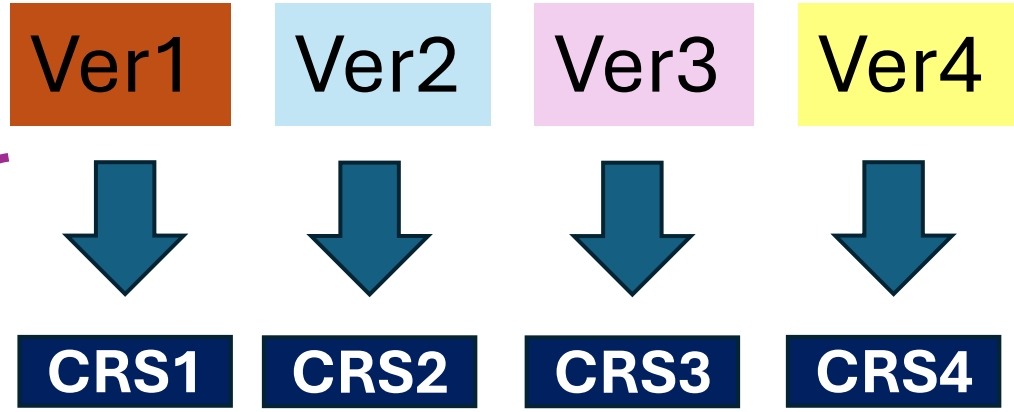
+

One-Way Functions



Multi-String NIZK

Designated Prover



Private message

Prover

2-round Multi-Verifier Zero-Knowledge proofs

- Previously known under NIZK
- Get everlasting security (for NC1 under CRH)
- Useful primitive for honest-majority MPC
- Yields 3-round honest-majority active MPC [AKP22]

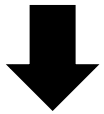
Constructing NPSS

New Notion of (t,n)-MPC

- Perfect passive correctness
- Perfect Passive privacy against **unauthorized set (size<t)**
- Perfect Active correctness (w/abort) against n-t+1 corrupted parties
- Adversary can't "fool" **honest set of size>t**

Construction in OT-hybrid model

based on player virtualization and monotone formula for Threshold



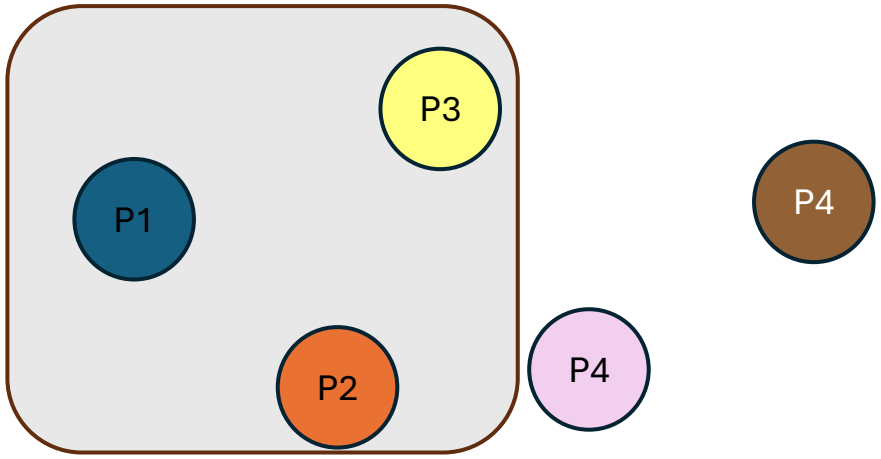
Variant of MPC-in-the-Head [IKOS'09]

- OT channel => partial assignments

(t,n)-NPSS

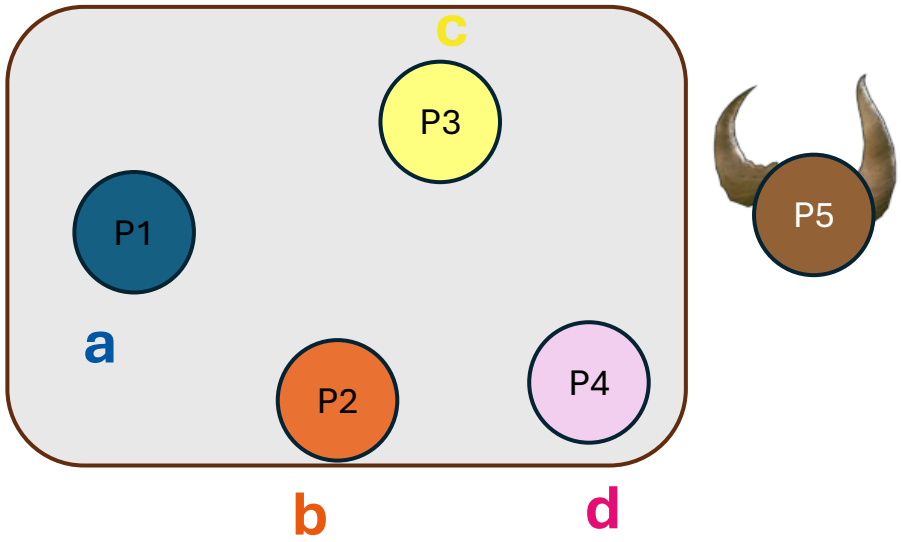
Example: 4-out-of-5 MPC for f

- In an honest execution, 3 passive parties learn nothing



Example: 4-out-of-5 MPC for f

- In an honest execution, 3 passive parties learn nothing
- 4 honest parties can recover the output/abort
  - 1 active party can't violate correctness
  - If **f(a,b,c,d, *)=0, the output will never be 1**



Note 1:
Existing protocols fail to achieve this notion even for small values of t

Note 2:
Security is non-monotone in t

Summary

New Notion of IT-MPC

- Threshold behavior for privacy/correctness
- Virtualization is a powerful tool!

Secret-Sharing an NP-statement

- Several interesting open questions
- Other applications?

Applications: Combining ZK proofs

Multi-string NIZK

Multi-Verifier Offline/Online ZK

3-round Honest-Majority Active MPC from **OWF**

"Give me an abstraction good enough, and I shall move the world…"

Follow-up Work https://eprint.iacr.org/2025/995

New Notion of IT-MPC

+Leakage-Resilient

Also, new results about standard leakage-resilient MPC

Secret-Sharing an NP-statement

+Leakage-Resilient

Thank You!

Applications: Combining ZK proofs

NIZK-Amplification

Resolving open questions from
[Goyal, Jain, Sahai'19,Bitansky, Geier'24]