Proof Complexity of MaxSAT

Anil Shukla

Indian Institute of Technology Ropar

Complexity Theory Update Meeting
IMSc Chennai
January 23, 2026

Credits

The credit for my understanding on MaxSAT proof systems –

- **Sravanthi Chede** IIT Ropar, Rupnagar, India.
- **Swagato Sanyal** University of Sheffield, UK.

Talk Contents

Proof Systems

- A proof system $f : \Delta^* \to \Sigma^*$ for a language $L \in \Sigma^*$ is a polynomial-time computable function such that the range of f is L.
- For $x \in L$, if $f(w) = x$, then w is an f-proof of the fact that $x \in L$. $|w|$ is the length of the proof.

- **Soundness**: For any $x \in \Sigma^*$ and $w \in \Delta^*$, if $f(w) = x$, then $x \in L$.
- **Completeness**: For every $x \in L$, there must exists $w \in \Delta^*$ such that $f(w) = x$.

Proof Systems

Let f and g are two proof systems for a language L.

- f **simulates** g if there is a computable function A that transforms the proofs in g to proofs in f with at most a polynomial blow in size.
- If A is polynomial time computable, then f **p-simulates** g.

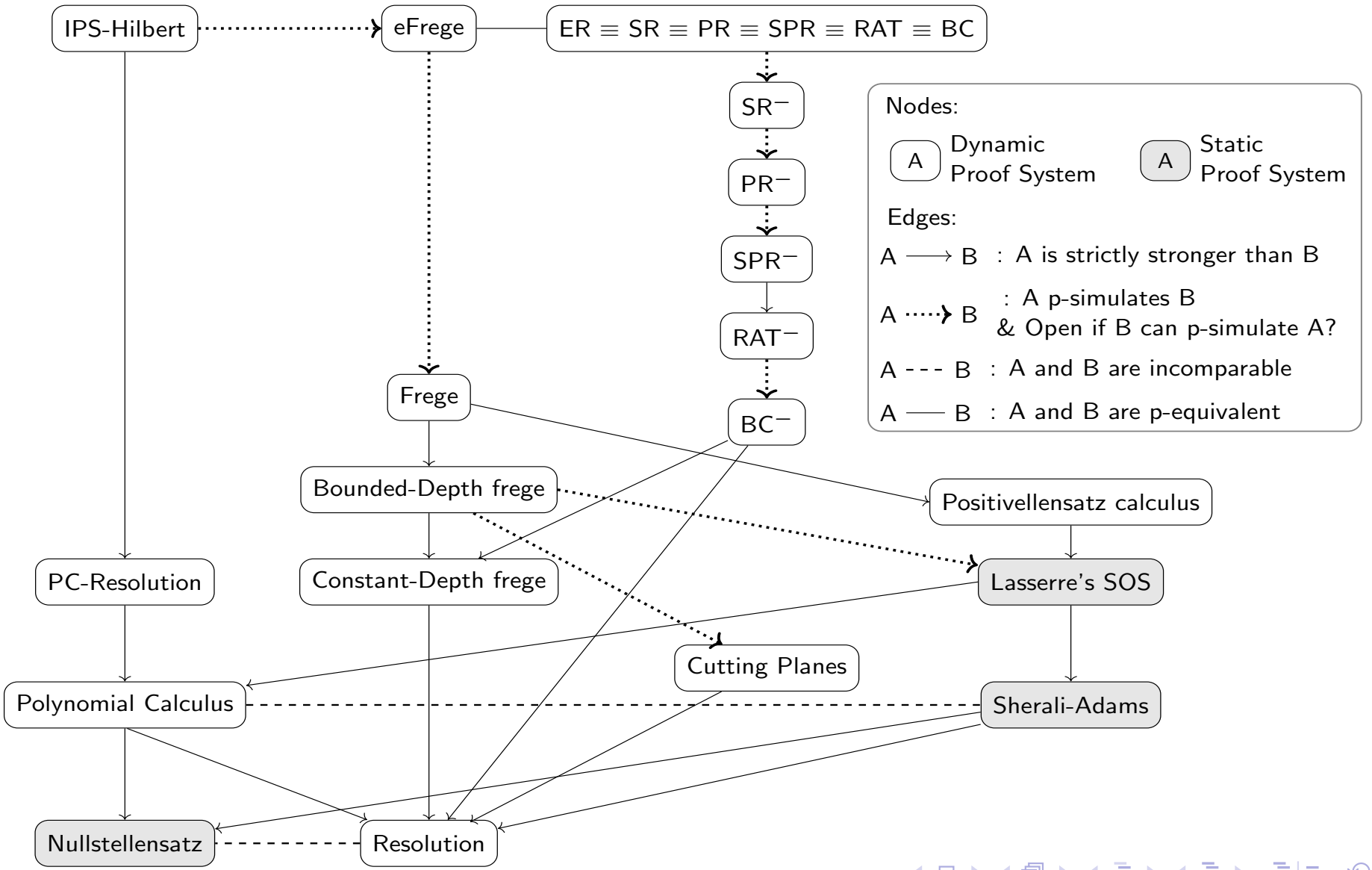
- f and g are **p-equivalent** if both can p-simulates each other.
- If f p-simulates g but g does not p-simulate f then we say that f is **strictly stronger** than g.

- f and g are **incomparable** if both cannot p-simulates each other.

Proof Systems

- Proof systems for the language $L = \text{UNSAT}$ are called propositional poof systems.
- Several propositional proof systems have been defined in literature.

Propositional proof systems



Resolution (Res) Proof System [Blake 1937, Davis and Putnam 1960, Robinson 1965]

- Resolution rule: $\frac{(C \vee x) \quad (D \vee \neg x)}{(C \vee D)}$, here C and D are any clauses.
- Let F be an unsatisfiable CNF formula.
  A Resolution proof π of F is a sequence of clauses

$$D_1, D_2, \ldots, D_k$$

such that the last clause D_k is the empty clause (i.e. $\square$)
and each D_i obeys one of the following:

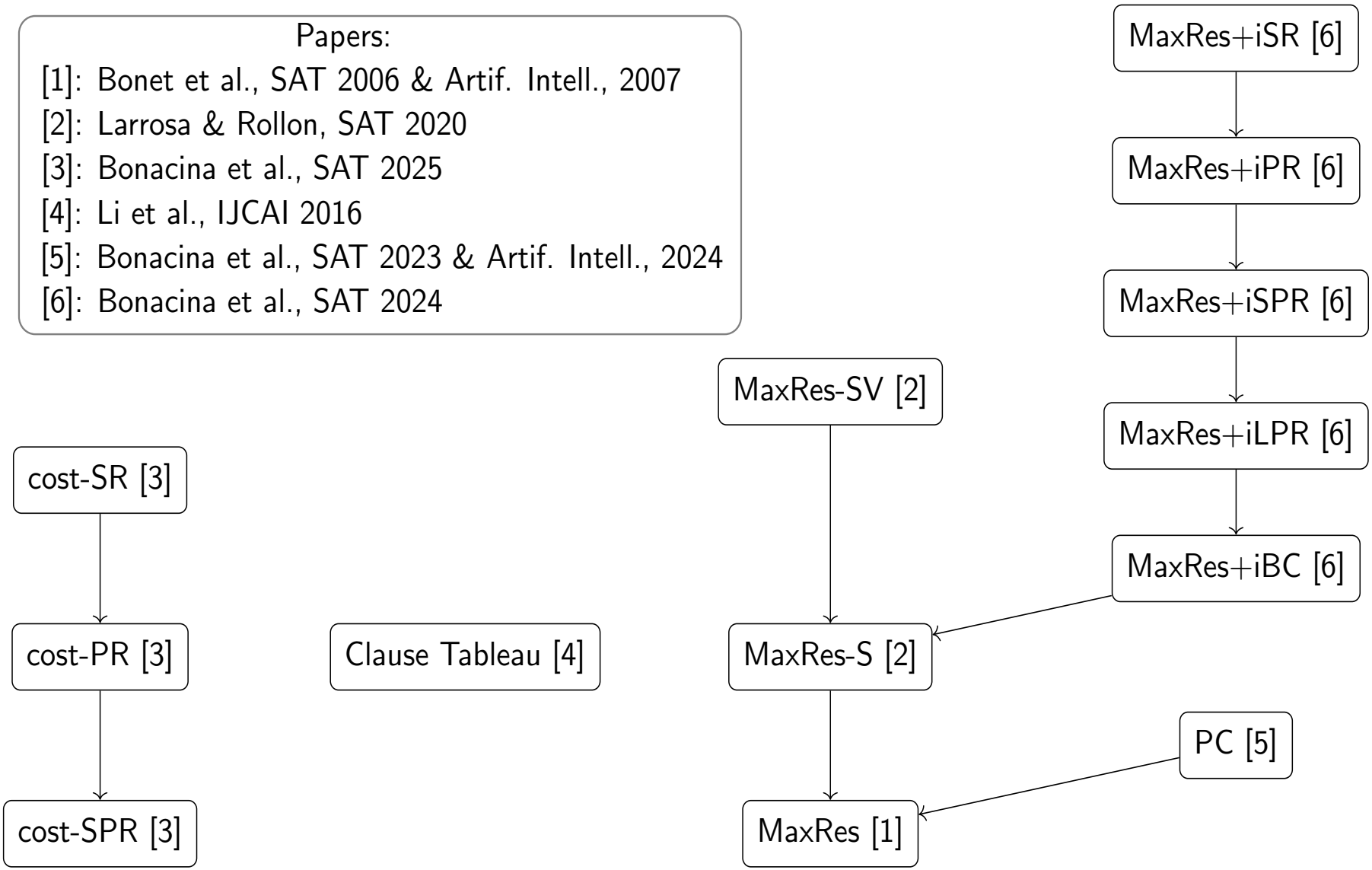
- $D_i \in F$
- D_i is derived from clauses D_k, D_j, with $j, k < i$ via the resolution rule.

Reverse Unit Propogation [Goldberg and Novikov 2003]

- **Unit propagation (UP)**: Unit propagation satisfies the unit clauses of the CNF formula F by assigning their literal to true. Until you get a fix point or a conflict (x and $\neg x$ both become true for some variable x).
- Given an assignment β, $F|_\beta$ denotes the CNF formula F' without the clauses of F satisfied by β and without the literals in the clauses of F falsified by β.
- Let F be a CNF formula and C a clause. Let α be the smallest assignment that falsifies C. We say that C is implied by F through UP (denoted $F \vdash_1 C$) if UP on $F|_\alpha$ results in a conflict.
- $F \vdash_1 C$ is known as Reverse Unit Propagation.

MaxSAT proof systems

Papers:
[1]: Bonet et al., SAT 2006 & Artif. Intell., 2007
[2]: Larrosa & Rollon, SAT 2020
[3]: Bonacina et al., SAT 2025
[4]: Li et al., IJCAI 2016
[5]: Bonacina et al., SAT 2023 & Artif. Intell., 2024
[6]: Bonacina et al., SAT 2024



MaxSAT proof systems

- MaxSAT is an optimization version of SAT.
- Given a CNF formula F, MaxSAT problem asks to find an assignment of values to Boolean variables that maximizes the number of satisfied clauses in F.
- Equivalently, it asks for an assignment of values to the Boolean variables of F that minimizes the number of unsatisfied clauses in F.
- We denote the minimum number of unsatisfiable clauses in F as min-unSAT(F).
- The resolution-based MaxSAT proof system (MaxRes) was designed by Bonet, Levy, and Manyà (Artifial Intelligence 2007).
- It has one rule: the MaxSAT resolution rule.

The MaxSAT resolution rule (Bonet, Levy, and Manyà 2007)

$$(x \vee a_1 \vee a_2 \vee \cdots \vee a_s)$$

$$(\neg x \vee b_1 \vee b_2 \vee \cdots \vee b_t)$$

$$---------------------------------$$

$$a_1 \vee a_2 \vee \cdots \vee a_s \vee b_1 \vee b_2 \vee \cdots \vee b_t$$

$$x \vee a_1 \vee a_2 \vee \cdots \vee a_s \vee \neg b_1$$

$$x \vee a_1 \vee a_2 \vee \cdots \vee a_s \vee b_1 \vee \neg b_2$$

$$\ldots$$

$$x \vee a_1 \vee a_2 \vee \cdots \vee a_s \vee b_1 \vee b_2 \vee \cdots \vee b_{t-1} \vee \neg b_t$$

$$\neg x \vee b_1 \vee b_2 \vee \cdots \vee b_t \vee \neg a_1$$

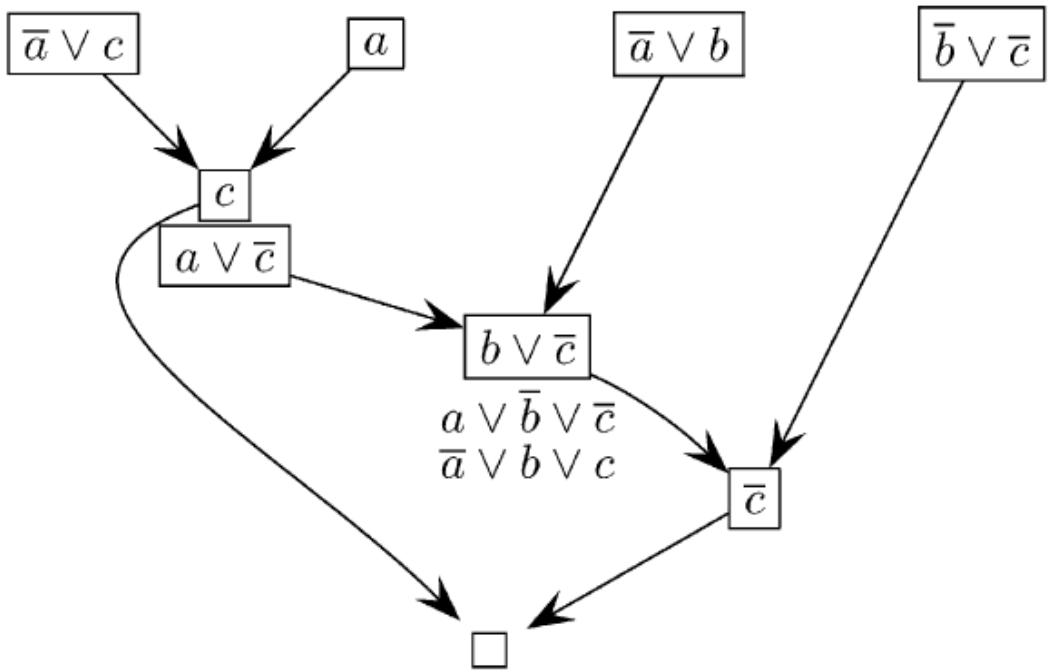
$$\neg x \vee b_1 \vee b_2 \vee \cdots \vee b_t \vee a_1 \vee \neg a_2$$

$$\ldots$$

$$\neg x \vee b_1 \vee b_2 \vee \cdots \vee b_t \vee a_1 \vee a_2 \cdots \vee a_{s-1} \vee \neg a_s$$

The MaxSAT resolution rule: Example (taken from Bonet, Levy, and Manyà 2007 [1])

$F = (a) \wedge (\overline{a} \vee b) \wedge (\overline{a} \vee c) \wedge (\overline{b} \vee \overline{c})$



$F \vdash_{MaxRes} (a \vee \overline{b} \vee \overline{c}) \wedge (\overline{a} \vee b \vee c) \wedge ()$

Min-unSAT(F)$= 1$, min-unSAT assignment: $a = b = c = 1$

The MaxSAT resolution rule

MaxSAT resolution rule is applied to multisets of clauses.

- the hypothesis of the rule are replaced by its conclusions.
- We say that the rule cuts the variable x.
- The tautological clauses from the conclusions are removed.
- Repeated literals in a clause are collapsed into one.

Let $\mathcal{C}$ and $\mathcal{D}$ are two multisets of clauses.

- $\mathcal{C} \vdash \mathcal{D}$: denotes that $\mathcal{D}$ can be obtained from $\mathcal{C}$ by applying the MaxSAT resolution rule finitely many times.
- We write $\mathcal{C} \vdash_x \mathcal{D}$ when the sequence of MaxSAT resolution rule cuts the variable x only.

Soundness

Theorem

The MaxSAT resolution rule is sound.

Proof:

- We need to show that the rule preserves the **number of unsatisfied clauses** for every truth assignment.
- Let I be any assignment.
- I cannot falsify both the hypothesis: $H_1 : (x \vee a_1 \vee \cdots \vee a_s)$ and $H_2 : (\neg x \vee b_1 \vee \cdots b_t)$.
- Let I satisfies H_1 and falsifies $H_2 \implies I(x) = 1$ & $I(b_j) = 0, \forall j \in [t]$.
  Case 1: $I(a_j) = 0, \forall j \in [s]$: I only falsifies $(a_1 \vee \cdots \vee a_s \vee b_1 \cdots \vee b_t)$ and satisfies all other.
  Case 2: I satisfies at least one literal a_j's: say $a_i = 1$ be the smallest which is set to 1 by I. Then I falsifies only $(\neg x \vee b_1 \vee \cdots b_s \vee a_1 \vee \cdots \vee a_{i-1} \vee \neg a_i)$ and satisfies remaining.

Soundness continue

- If I falsifies $H_1 : (x \vee a_1 \vee \cdots \vee a_s)$ and satisfies $H_2 : (\neg x \vee b_1 \vee \cdots b_t)$, by similar argument, I falsifies only one conclusion.
- Suppose, I satisfies both the hypothesis H_1 and H_2. Suppose, $I(x) = 1$. Then, for some j, $I(b_j) = 1$. Then I satisfies all conclusions, since b_j or x is present in all the conclusions.
- This proves that the MaxSAT resolution is sound.

Saturating multi-set of clauses

- The concept of saturation was introduced for the resolution proof system.
- Given a set of clauses and a variable, we can saturate the set of clauses by cutting the variables exhaustively, obtaining a superset of the given clauses.
- Resolution algorithm: repeat the above process for all the variables. We obtain the empty clause, whenever the original set of clauses is unsatisfiable.
- Saturating a multiset of clauses using MaxRes resolution rule is not simple: saturate a multiset of clauses with one variable, and then saturate with another variable, the resulting multiset is not saturated with both the variables.

Saturating multiset of clauses

Definition (Bonet, Levy, and Manyà 2007)

A multiset of clauses $\mathcal{C}$ is said to be saturated w.r.t. x if for every pair of clauses $C_1 = x \vee A$ and $C_2 = \neg x \vee B$ of $\mathcal{C}$, there is a literal ℓ such that ℓ is in A and $\neg\ell$ is in B.

A multiset of clauses $\mathcal{C}'$ is a saturation of $\mathcal{C}$ w.r.t. x if $\mathcal{C}'$ is saturated w.r.t. x and $\mathcal{C} \vdash_x \mathcal{C}'$. That is, $\mathcal{C}'$ can be obtained from $\mathcal{C}$ using MaxSAT resolution rule cutting x finitely many times.

Saturating multiset of clauses

Lemma (Bonet, Levy, and Manyà 2007)

For every multiset of clauses $\mathcal{C}$ and variable x, there exists a multiset $\mathcal{C}'$ such that $\mathcal{C}'$ is a saturation of $\mathcal{C}$ w.r.t. x.
Moreover, this multiset $\mathcal{C}'$ can be computed by applying the MaxSAT resolution rule to any pair of clauses $x \vee A$ and $\neg x \vee B$ with the restriction that $A \vee B$ is not a tautology, using any ordering of the literals, until we cannot apply the inference rule any longer with the restriction.

Proof.

Just apply the MaxSAT resolution rule cutting x non-deterministically, until we obtain a saturated multiset.
We only need to prove that this process terminates in finitely many inference steps.
We show this by defining a simple characteristic function for multiset of clauses. □

Saturating multiset of clauses

Definition

For every clause $C = x_1 \vee \cdots \vee x_s \vee \neg x_{s+1} \vee \cdots \vee \neg x_{s+t}$ we define its characteristic function as

$$P_C(\overrightarrow{x}) = (1 - x_1) \cdots (1 - x_s) x_{s+1} \cdots x_{s+t}$$

. For multiset of clauses $\mathcal{C} = \{C_1, \cdots, C_m\}$, define its characteristic function $P_{\mathcal{C}} = \sum_{i=1}^{n} P_{C_i}(\overrightarrow{x})$.

Clearly, for every assignment I, $P_{\mathcal{C}}(I)$ is the number of clauses of $\mathcal{C}$ falsified by I.

Proof continues

- At every step of MaxSAT resolution, we can divide the resulting multiset $\mathcal{C}_i$ into two multisets: $\mathcal{E}_i$ with all clauses that do not contain x and $\mathcal{D}_i$ with all clauses that contain the variable x.
- The MaxSAT resolution rule takes two clauses of $\mathcal{D}_i$ and replaces it with a set of clauses, where one clause, say A does not contain the variable x.
- Therefore, we obtain $\mathcal{C}_{i+1} = \mathcal{D}_{i+1} \cup \mathcal{E}_{i+1}$, where $\mathcal{E}_{i+1} = \mathcal{E}_i \cup A$.
- As A is not a tautology, P_A is not 0.
- Since, the rule is sound, $P_{\mathcal{C}_{i+1}} = P_{\mathcal{C}_i}$ and $P_{\mathcal{E}_{i+1}} = P_{\mathcal{E}_i} + P_A$.
- Therefore, we have $P_{\mathcal{D}_{i+1}} = P_{\mathcal{D}_i} - P_A$ and $P_{\mathcal{D}_{i+1}} < P_{\mathcal{D}_i}$.
- Thus the number of MaxSAT resolution step cannot be infinite.

Saturating multi-set of clauses

Lemma (Bonet, Levy, and Manyà 2007)

Let $\mathcal{C}$ be a saturated multiset of clauses w.r.t. x. Let $\mathcal{C}'$ be the subset of clauses of $\mathcal{C}$ not containing x. Then any assignment I satisfying $\mathcal{C}'$ (and not assigning x) can be extended to an assignment satisfying $\mathcal{C}$.

Proof: I is satisfying $\mathcal{C}'$, we need to give value to x such that it satisfy the entire $\mathcal{C}$. If x has a unique polarity in $\mathcal{C} \setminus \mathcal{C}'$, then the extension is trivial. If for every clause $(x \vee A)$ or $(\neg x \vee A)$, I already satisfies A, then just extend x arbitrary.
Otherwise, there is a clause $(x \vee A) \in \mathcal{C} \setminus \mathcal{C}'$ such that I falsifies A. Set $x = 1$. All clauses of the form $x \vee B$ will be satisfied. Since $\mathcal{C}$ is saturated all clauses of the form $\neg x \vee B$ also gets satisfied: since negation of some literal $\ell \in A$ must be present in B. □.

Completeness

Theorem (Bonet, Levy, and Manyà 2007)

For any multi-set of clauses $\mathcal{C}$, we have

$$\mathcal{C} \vdash \underbrace{\square, \square, \cdots, \square}_{m}, \mathcal{D}$$

where $\square$ denotes empty clause, $\mathcal{D}$ is a satisfiable multi-set of clauses, m is the minimum number of unsatisfied clauses of $\mathcal{C}$.

Completeness proof

Proof.

Let $x_1, \cdots, x_n$ be variables list of $\mathcal{C}$. We construct two sequences of multisets $\mathcal{C}_0, \cdots \mathcal{C}_n$ and $\mathcal{D}_1, \cdots, \mathcal{D}_{n+1}$ such that

- $\mathcal{C} = \mathcal{C}_0$,
- for $i = 1, \cdots, n$, $\mathcal{C}_i \cup \mathcal{D}_i$ is a saturation of $\mathcal{C}_{i-1}$ w.r.t. x_i, and
- for $i = 1, \cdots, n$, $\mathcal{C}_i$ is a multiset of clauses not containing $x_1, \cdots, x_i$, and $\mathcal{D}_i$ is a multiset of clauses containing the variable x_i.

We compute this as follows: saturate $\mathcal{C}_{i-1}$ w.r.t. x_i and then partition the resulting multi-set into subsets $\mathcal{D}_i$ containing x_i and $\mathcal{C}_i$ not containing x_i. Note: $\mathcal{C}_n$ does not contains any variables. So, $\mathcal{C}_n$ is either an empty subset, or it contains some empty clauses $\{\square, \cdots, \square\}$. $\square$

Completeness proof continue

Proof.

- We show that $\mathcal{D} = \cup_{i=1}^{n} \mathcal{D}_i$ is satisfiable (proof idea below).
- Since, MaxSAT resolution rule is sound, and $\mathcal{D}$ is satisfiable, we have $m = |\mathcal{C}_n|$ is the minimum number of unsatisfied clauses of $\mathcal{C}$.

□

Lemma

$\mathcal{D} = \cup_{i=1}^{n} \mathcal{D}_i$ is satisfiable.

Proof Idea: Let $\mathcal{E}_i = \mathcal{D}_i \cup \cdots \cup \mathcal{D}_n$, for $i \in [n]$ and $\mathcal{E}_{n+1}$ is $\emptyset$.
Let I_{n+1} is the empty assignment which satisfies $\mathcal{E}_{n+1} = \emptyset$.
Using previous lemma, we can construct I_i by extending I_{i+1} with an assignment for x_i such that I_i satisfies $\mathcal{E}_i$. This is easy as $\mathcal{E}_i$ is saturated w.r.t. x_i.

An algorithm for MaxSAT (Bonet, Levy, and Manyà 2007)

Data: $\mathcal{C}$

/* the input $\mathcal{C}$ is a multi-set of clauses */

$\mathcal{C}_0 = \mathcal{C}$;

for $i = 1$ *to* n **do**

  $\mathcal{C} = \text{saturation}(\mathcal{C}_{i-1}, x_i)$;

  $(\mathcal{C}_i, \mathcal{D}_i) = \text{partition}(\mathcal{C}, x_i)$;

end

$m = |\mathcal{C}_n|$;

$I = \emptyset$;

for $i = n$ **downto** 1 **do**

  $I = I \cup [x_i \rightarrow \text{extention}(x_i, I, \mathcal{D}_i)]$;

end

output: m, I

Analysis of the MaxSAT algorithm

Theorem

For any multiset $\mathcal{C}$ with k clauses on n variables, we can deduce $\mathcal{C} \vdash \square, \square, \cdots, \mathcal{D}$, where $\mathcal{D}$ is satisfiable, in less than $n \cdot k \cdot 2^n$ inference steps. Moreover, the search of this proof can also be done in time $O(m2^n)$.

Redundancy rules

- Redundancy rules were introduced in SAT solving to allow the introduction of clauses that preserves satisfiability even though they are not logical equivalent.
- Redundant clauses formalize the notion of reasoning "without loss of generality".
- It reduces the space of solutions without killing it entirely.
- Blocked clauses (BC) and RAT (Resolution Asymmetric Tautologies) are the first redundancy rules studied in the literature.
- Researchers have tried to extend these redundancy rules for MaxSAT problem as well.
- We see two such results in this talk.

MaxSAT Resolution with Inclusion Redundancy (Bonacina, Bonet, and Lauria 2024)

- In the SAT problem, for a CNF formula F, we say that a clause C is redundant if F is satisfiable if and only if $F \cup C$ is satisfiable.
- Similar for the MaxSAT problem, we say that for a multiset of clauses S, a clause C is redundant if min-unSAT(S) = min-unSAT$(S \cup C)$. Note: we can add any number of C in S without changing the answer of the min-unSAT problem.
- We need the following definitions.

Substitution

Definition

A substitution σ for a set of variables X is a function so that $\sigma(x)$ is either $0, 1$ or some literal defined in X (a literal is either a variable x or $\neg x$). For convenience, we extend this to $\sigma(0) = 0, \sigma(1) = 1$, and $\sigma(\neg x) = \neg(\sigma(x))$, for any variable $x \in X$.

The composition of two substitutions σ, τ is the substitution $\sigma \circ \tau$, where $\sigma \circ \tau(x) = \sigma(\tau(x))$ for $x \in X$.

For example, consider 3 variables x_1, x_2, x_3, substituitions could be $\sigma := \{x_1 = \neg x_2, x_2 = 0, x_3 = x_3\}$ and $\tau := \{x_1 = x_3, x_2 = x_1, x_3 = \neg x_3\}$.

The composition of $\sigma \circ \tau := \{x_1 = x_3, x_2 = \neg x_2, x_3 = \neg x_3\}$

Assignments

- A substitution σ is an assignment when $\sigma(x) \in \{0, 1, x\}$ for any variable $x \in X$.
- A domain of any assignment σ is $\mathrm{dom}(x) = \sigma^{-1}(\{0, 1\})$
- σ is a total assignment over X if X is its domain.
  That is, σ maps all variables in X to Boolean values.

Restrictions

- Let $C = \vee_i \ell_i$ and σ a substitution. Then,
$$C|_\sigma = \sigma(C) = \vee_i \sigma(\ell_i)$$
- $D \vee 0 = D, D \vee 1 = 1, D \vee \ell \vee \ell = D \vee \ell$.
- If $C|_\sigma = 1$ or $C|_\sigma$ is a tautology, then σ satisfies C (denoted $\sigma \models C$). For example, if $C = (x_1 \vee x_2 \vee \neg x_3)$ and $\sigma := \{x_1 = \neg x_2, x_2 = x_2, x_3 = \neg x_3\}$. Then, $C|_\alpha = (\neg x_2 \vee x_2 \vee x_3)$ is a tautology.

- The restriction of a multiset of clauses Γ by a substitution σ is
$$\Gamma|_\sigma = \{C|_\sigma \mid C \in F, C|_\sigma \neq 1\}$$
- We say that σ satisfies Γ ($\sigma \models \Gamma$) if for all $C \in \Gamma$, $\sigma \models C$.

- We say that $\Gamma \models C$ if for every substitution σ, if $\sigma \models \Gamma$ then $\sigma \models C$.

Inclusion Substitution Redundant, iSR

Definition (Bonacina, Bonet, Lauria 2024)

A clause C is Inclusion Substitution Redundant (iSR) w.r.t a multiset of clauses S if there exists a substitution σ such that

$$(S \cup \{C\})|_\sigma \subseteq S|_{\overline{C}}$$

$\overline{C}$ denotes the least assignment falsifying C.
For example, if $C = (x \vee \neg y \vee z)$ then $\overline{C} = \{x = 0, y = 1, z = 0\}$.

Inclusion Substitution Redundant, iSR

In the UNSAT domain: A Clause C is SR w.r.t. a CNF S if there exists a substituition σ such that $\sigma \models C$ and $(S)|_{\overline{C}} \vdash_1 S|_\sigma$

Definition (Bonacina, Bonet, Lauria 2024)

A clause C is Inclusion Substitution Redundant (iSR) w.r.t a multiset of clauses S if there exists a substitution σ such that

$$(S \cup \{C\})|_\sigma \subseteq S|_{\overline{C}}$$

$\overline{C}$ denotes the least assignment falsifying C.
For example, if $C = (x \vee \neg y \vee z)$ then $\overline{C} = \{x = 0, y = 1, z = 0\}$.

iSR rule is sound

A clause C is Inclusion Substitution Redundant (iSR) w.r.t a multiset of clauses S if there exists a substitution σ such that $(S \cup \{C\})|_\sigma \subseteq S|_{\overline{C}}$

Lemma

If a clause C is iSR w.r.t. a multiset of clauses S, then C is redundant w.r.t. S. That is, then min-unSAT(S) =min-unSAT($S \cup \{C\}$).

Proof:

- Clearly min-unSAT(S) $\leq$min-unSAT($S \cup \{C\}$).
- If C is iSR, we show that min-unSAT($S \cup \{C\}$) $\leq$ min-unSAT(S).
- Let min-unSAT(S) $= k$ and β is a total assignment such that $|S|_\beta| = |\{\square, \cdots, \square\}| = k$. If $\beta \vDash C$, then we are done.
- So, now suppose that $C|_\beta = \square$. Clearly β extends $\overline{C}$.

iSR rule is sound (contd.)

A clause C is Inclusion Substitution Redundant (iSR) w.r.t a multiset of clauses S if there exists a substitution σ such that $(S \cup \{C\})|_\sigma \subseteq S|_{\overline{C}}$

Proof continues:

- As C is an iSR, there exists a substitution σ such that $(S \cup \{C\})|_\sigma \subseteq S|_{\overline{C}}$.
- Therefore, $(S \cup \{C\})|_{\beta \circ \sigma} \subseteq S|_{\beta \circ \overline{C}} = S|_\beta$.
- And we have $|(S \cup \{C\})|_{\beta \circ \sigma}| \leq |S|_\beta| = k$.
- We have a total assignment $\beta \circ \sigma$ which produces k empty clause when restricted on $S \cup C$. min-unSAT is minimum over all.
- Therefore, min-unSAT$(S \cup C) \leq k$.

MaxSAT-Resolution + iSR

Definition (Bonacina, Bonet, Lauria 2024)

A sequence of multisets $(S_i)_{i\in[m]}$ is a derivation of (S_m) from (S_1) in MaxSAT-Resolution + iSR if for each $i \in [m]$ either S_{i+1} is derived using the MaxSAT resolution rule from one of the already derived multisets $(S_i), \cdots (S_1)$ or

- $S_{i+1} = S_i \cup \{C\}$, where C is iSR w.r.t. S_i;
- $S_{i+1} = S_i \setminus \{C\}$ where C is iSR w.r.t. $S_i \setminus \{C\}$.

Soundness: Since both the MaxSAT resolution rule and iSR are sound, we know that min-unSAT(S_1) =min-unSAT(S_m). If (S_m) contains $\square$ with multiplicity at least k, the derivation certifies that min-unSAT$(S_1) \geq k$.

Completeness: Since MaxSAT resolution proof system is complete.

Redundancy rules for MaxSAT Bonacina, Bonet, Buss, and Lauria 2025

- MaxSAT-Resolution + iSR proves min-unSAT(F) $\geq k$.
- Bonacina, Bonet, Buss, and Lauria SAT-2025 introduced different redundancy proof systems (cost-SR) for MaxSAT.
- These cost-SR systems prove min-unSAT(F) $= k$.
- These proof systems work with hard clauses and blocking variables.

Soft and hard clauses

- **Hard clauses**: clauses which must be satisfied.
- **Soft clauses**: clauses which can be falsified.
- $F = H \wedge S$ where H, S are multi-sets of hard clauses and soft clauses, respectively.

- The generalized MaxSAT problem asks to find the maximum number of clauses in S that can be simultaneously satisfied by an assignment that satisfies all clauses in H.

- The problem is not defined if H is unsatisfiable.
- H may be set or multi-set: Not relevant.
- Multiplicity of clauses in S is relevant.

MaxSAT with blocking variables

- Without loss of generality, we can assume that all soft clauses in a MaxSAT instance are unit clauses.

Definition

Let $F = H \wedge S$, with $S = \{C_1 \wedge \cdots \wedge C_m\}$. The blocking variable formulation of F is $F' = H' \wedge S'$ where,

- $H' = H \wedge (C_1 \vee b_1) \wedge \cdots \wedge (C_m \vee b_m)$,
- $S' = \overline{b_1} \wedge \cdots \wedge \overline{b_m}$,

$b_1, \cdots, b_m$ are new variables (called blocking variables) not appearing in F. We say that Γ is a MaxSAT instance encoded with blocking variables, when it is given as a set of hard clauses of the form as in H' above. The soft clauses then are implicit.

MaxSAT instance with blocking variables

Lemma

Let $F = H \wedge S$ be a MaxSAT instance and $F' = H' \wedge S'$ be the blocking variable formulation of F.

- *Then, any assignment satisfies H and falsifies k clauses in S can be extended to an assignment that satisfies H' and sets k blocking variables to true.*
- *Vice versa, any assignment that satisfies H' and sets k blocking variables to true, satisfies H too and falsifies at most k clauses in S.*

Recall: $F = H \wedge S$, where $S = \{C_1 \wedge \cdots \wedge C_m\}$.
$H' = H \wedge (C_1 \vee b_1) \wedge \cdots (C_m \vee b_m)$
$S' = \overline{b_1} \wedge \cdots \overline{b_m}$.

Equivalent problem to MaxSAT

- Finding min-unSAT($H \wedge S$) is reduced to the problem of finding a satisfying assignment of H' that sets the least number of blocking variables to true.
- In the rest of this talk, Γ denotes the MaxSAT instance encoding with blocking variables (i.e., $H' \wedge S'$) of $F = H \wedge S$.
  Note: Γ is given as a set of hard clauses H' only. (S' is implicit).
- Given a total assignment α for Γ, we define

$$\text{cost}(\alpha) = \sum_{i=1}^{m} \alpha(b_i) \quad \text{and} \quad \text{cost}(\Gamma) = \min_{\alpha:\alpha \models \Gamma} \text{cost}(\alpha)$$

- Clearly, min-unSAT($H \wedge S$) $\equiv$ cost(Γ).

Cost substitution redundant (cost-SR)

Definition (redundant clause)

A clause C is redundant w.r.t. a MaxSAT instance Γ when

$$\text{cost}(\Gamma) = \text{cost}(\Gamma \cup \{C\}). \tag{1}$$

The condition in equation (1) is not polynomial-time checkable. The goal is to consider efficiently certifiable notion of redundancy.

Definition (Bonacina, Bonet, Buss, Lauria 2025)

A clause C is cost substitution redundant (cost-SR) w.r.t. Γ if there exists a substitution σ such that

- $\Gamma|_{\overline{C}} \vdash_1 (\Gamma \cup \{C\})|_\sigma$ (redundancy)
- for all total assignment $\tau \supseteq \overline{C}, \text{cost}(\tau \circ \sigma) \leq \text{cost}(\tau)$ (cost)

When the substitution σ above is partial assignment with the same domain as $\overline{C}$, we call the rule as cost-SPR (subset propagation redundant) rule.

A clause C is cost substitution redundant (cost-SR) w.r.t. Γ if there exists a substitution σ such that

- $\Gamma|_{\overline{C}} \vdash_1 (\Gamma \cup \{C\})|_\sigma$ (redundancy)
- for all total assignment $\tau \supseteq \overline{C}, \text{cost}(\tau \circ \sigma) \leq \text{cost}(\tau)$ (cost)

Lemma

If C is cost-SR w.r.t. Γ, then C is redundant w.r.t. Γ.

Proof:

Clearly, $\text{cost}(\Gamma) \leq \text{cost}(\Gamma \cup \{C\})$. We show that $\text{cost}(\Gamma) \geq \text{cost}(\Gamma \cup \{C\})$.
Let $\text{cost}(\Gamma) = k$.
Let α is an optimal total assignment that satisfies Γ and sets to true exactly k blocking variables. If α satisfies C, then we are done. Since, then $\alpha \vDash \Gamma \cup \{C\}$ and $\text{cost}(\alpha) = k$.
Otherwise, α extends $\overline{C}$ and, by assumption, there is a substitution σ such that $\text{cost}(\alpha \circ \sigma) \leq \text{cost}(\alpha) = k$.
If $\alpha \circ \sigma \vDash \Gamma \cup \{C\}$, then we are done: $\text{cost}(\Gamma \cup \{C\}) \leq k = \text{cost}(\Gamma)$.

A clause C is cost substitution redundant (cost-SR) w.r.t. Γ if there exists a substitution σ such that

- $\Gamma|_{\overline{C}} \vdash_1 (\Gamma \cup \{C\})|_\sigma$ (redundancy)
- for all total assignment $\tau \supseteq \overline{C}, \text{cost}(\tau \circ \sigma) \leq \text{cost}(\tau)$ (cost)

We only need to show that $\alpha \circ \sigma \models \Gamma \cup \{C\}$.
By assumption, we have

$$\Gamma|_{\overline{C}} \vdash_1 (\Gamma \cup \{C\})|_\sigma$$

Since, $\alpha \models \Gamma$ and extends $\overline{C}$. We have $\alpha \models \Gamma|_{\overline{C}}$.
From the above, the following must hold: $\alpha \models (\Gamma \cup \{C\})|_\sigma$. Equivalently, $\alpha \circ \sigma \models \Gamma \cup \{C\}$. □

Lemma (cost-SR is polynomial time verifiable)

Let Γ be a MaxSAT instance, C a clause and σ a substitution. There is a polynomial time algorithm to decide whether C is cost-SR w.r.t. Γ, given the substitution σ.

cost-SR calculus

Definition (Bonacina, Bonet, Buss, Lauria 2025)

The cost-SR calculus is a proof system for MaxSAT. A derivation of a clause C from a MaxSAT instance Γ (encoded with blocking variable) is a sequence of clauses $D_1, D_2, \cdots, D_t$, where $C \in \Gamma \cup \{D_i\}_{i \in [t]}$, and each D_i is either

- already in Γ, or
- is deduced using earlier clauses in the sequence using the resolution rule, or
- D_i is a cost-SR w.r.t. $\Gamma \cup \{D_1, \cdots, D_{i-1}\}$ with $\text{var}(D_i) \subseteq \mathit{var}(\Gamma)$.

The length of such a derivation is t.

cost-SR calculus

- If we wish to certify that $\text{cost}(\Gamma) \geq s$ using cost-SR calculus, then we can achieve this by deriving s distinct unit clauses of the form $\{b_{i_1}, \cdots, b_{i_s}\}$.
- If the goal is to certify that $\text{cost}(\Gamma) = s$, then we can achive this by proving a cost-SR derivation of the unit clauses $\{b_{i_1}, \cdots, b_{i_s}\}$ along with the derivations of the unit clauses $\{\neg b_j \ : \ j \notin \{i_1, \cdots, i_s\}\}$.

Soundness of cost-SR calculus

Theorem (Bonacina, Bonet, Buss, Lauria 2025)

Let Γ be a MaxSAT instance encoded with blocking variables. If there is a cost-SR proof of k distinct blocking variables, then cost$(\Gamma) \geq k$. If there is a cost-SR proof of k distinct blocking variables $\{b_{i_1}, \cdots, b_{i_k}\}$ and all the unit clauses $\neg b_j$ for $j \notin \{i_1, \cdots, i_k\}$, then cost$(\Gamma) = k$.

Proof:

- Let $\{b_1, \cdots, b_m\}$ are the blocking variables of Γ.
- Let Γ' contains all the clauses in Γ plus all the clauses derived by the cost-SR calculus.
- In particular, Γ' contains all unit clauses $\{b_{i_1}, \cdots, b_{i_k}\}$.
- Clearly, cost$(\Gamma') \geq k$.
- Since, cost-SR calculus is sound, we have cost$(\Gamma) =$ cost$(\Gamma') \geq k$. In addition, if Γ' also contains $\neg b'_j s$ then cost$(\Gamma') =$ cost$(\Gamma) = k$. □

Completeness of cost-SR calculus

In fact we show the completeness of cost-SPR calculus. Recall, in cost-SPR rule, the substitution σ must be a partial assignment with the same domain as $\overline{C}$.

Theorem

Let Γ be a MaxSAT instance encoded with blocking variables, of $cost(\Gamma) = k$. There is a cost-SPR derivation of the unit clauses $b_{i_1}, \cdots, b_{i_k}$ for some disctinct k blocking variables and all the $\neg b_j$ for $\jmath \notin \{i_1, \cdots, i_k\}$.

Proof:

- Let $\{b_1, \cdots, b_m\}$ be the blocking variables of Γ.
- Let α_{opt} be an optimal total assignment, that is $\alpha_{\text{opt}} \vDash \Gamma, \text{cost}(\alpha_{\text{opt}}) = k$, and for every total assignment β such that $\beta \vDash \Gamma, \text{cost}(\beta) \geq k$.
- Without loss of generality α_{opt} sets $b_1, \cdots, b_k$ to 1 and the remaining $b_j's$ to 0.

Completeness proof continues

Proof continue:

- For any assignment γ, let $\overline{\gamma}$ be the largest clause falsified by γ.
- For example, if $\gamma = x \leftarrow 0, y \leftarrow 1, z \leftarrow 0$, then $\overline{\gamma} = (x \vee \neg y \vee z)$.
- Consider all total assignments γ, such that $\gamma \models \Gamma$ and are different from α_{opt}.
- Let
  $\Sigma = \{\overline{\gamma} : \gamma \text{ is total assignment}, \gamma \models \Gamma, \text{and}, \gamma \text{ is different from } \alpha_{\mathsf{opt}}\}$.
- We plan to derive Σ from Γ via cost-SPR rule. We do this by adding the clauses of Σ one by one using the cost-SPR rule (see Lemma below: the witnessing assignment for the cost-SPR is α_{opt}).

Completeness proof continues

Proof continues:

- After we have added all the clauses $\overline{\gamma}$ to Γ the **only** total assignment satisfying $\Gamma \cup \Sigma$ is α_{opt}.
- Since resolution is complete, it is capable of deriving all the literal of α_{opt}: $\Gamma \cup \Sigma$ and $(b_1) \wedge \cdots \wedge (b_k) \wedge (\neg b_{k+1}) \wedge \cdots \wedge (\neg b_m)$ are logically equivalent.
- So, we have a cost-SPR derivations of $b_i, i \in [k]$ and $\neg b_j, j \notin [k]$.
- Only we need to prove the claim.

Completeness proof continues

Lemma

Let γ be a total assignment satisfying Γ and different from α_{opt}. $\overline{\gamma}$ is the largest clause falsified by γ. Σ be the set of all clauses $\overline{\gamma}$. Then any clause $\overline{\gamma} \in \Sigma$ and any $\Sigma' \subseteq \Sigma$, the clause $\overline{\gamma}$ is a cost-SPR w.r.t. $\Gamma \cup \Sigma'$

Proof.

Here C is $\overline{\gamma}$, and so $\overline{C}$ is γ. We need to provide a total assignment σ with same domain as γ which satisfies the redundancy and the cost rules. Let α_{opt} be the required σ. We need to prove the following statements:

- $(\Gamma \cup \Sigma')|_\gamma \left|\frac{}{1}\right. (\Gamma \cup \Sigma \cup \overline{\gamma})|_{\alpha_{\text{opt}}}$.
- for all total assignment $\tau \supseteq \gamma$ $\text{cost}(\tau \circ \alpha_{\text{opt}}) \leq \text{cost}(\tau)$).

First condition holds: $(\Gamma \cup \Sigma \cup \overline{\gamma})|_{\alpha_{\text{opt}}}$ is True.
Second condition holds: α_{opt} is total and optimal. □

Short cost-SR proofs of the pigeonhole principle

- Let $m > n \geq 1$. The pigeonhole principle from m pigeons and n holes, with blocking variables (bPHP_n^m) has the following formulation:
- The totality clause: $(p_{i,j} \vee b_i)$ for $\imath \in [m]$.
- The injective clause: $(\overline{p}_{i,j} \vee \overline{p}_{k,j} \vee b_{i,k,j})$ for $1 \leq i < k \leq m$ and $j \in [n]$.

Theorem (Bonacina, Bonet, Buss, Lauria 2025)

cost-SR proves cost$(bPHP_n^m) = m - n$ *in polynomial size.*

Polynomial Calculus for MaxSAT (Bonacina, Bonet, Levi 2023)

- Let $\mathcal{F}_q$ be a finite field with q elements.
- Let X be a set of variables.
- Let $\mathcal{F}_q[X]$ denotes the ring of multivariate polynomials with coefficients in $\mathcal{F}_q$ and variables in X.
- Let $f \in \mathcal{F}_q[X]$ and $\alpha : X \to \mathcal{F}_q$ is an assignment. We say that $f(\alpha)$ is the evaluation of f in α.
- If $f(\alpha) = 0$, we say that α satisfies the polynomial f.
- The polynomial 1 represent the unsatisfiable polynomial.
- The concept of hard and soft clauses can be extended to polynomials as well.

Polynomial Calculus for MaxSAT

- The set of polynomials $H \subset \mathcal{F}_q[X]$ are hard polynomials which must be satisfied.
- The set of soft polynomials

$$F = \{[f_1, w], \cdots, [f_m, w_m]$$

  where $f_i \in \mathcal{F}_q[X]$ and $w_i \in \mathbb{N}$.
- We are interested in assignments α that minimize the weight of falsified soft polynomials in F, and satisfy all the polynomials in H.
- We denote the calculus for this problem as $(wPC_{\mathcal{F}_q,\mathbb{N}})$.
- In this talk, we only discuss the $(wPC_{\mathcal{F}_2,\mathbb{N}})$ calculus

$\mathbb{N}$-weighted Polynomial calculus $wPC_{\mathcal{F}_2,\mathbb{N}}$

- The initial instance consists of multi-sets of weighted polynomials, that is, pairs $[f, w]$ with $f \in \mathcal{F}_2[X]$ and $w \in \mathbb{N}$, and a set of hard polynomials H.
- We need an assignment α that satisfy all of H and minimize the weights of falsified soft polynomials in F.
- Before presenting the calculus $wPC_{\mathcal{F}_2,\mathbb{N}}$, we need some definitions.

Definition (H-compatible assignment)

Let $H \subset \mathcal{F}_2[X]$. An assignment $\alpha : X \to \mathcal{F}_2$ is H-compatible if for every $h \in H, h(\alpha) = 0$.

Definitions

For each assignment $\alpha : X \to \mathcal{F}_2$, we measure how close it is to satisfying all the polynomials in F. We do this by defining the cost:

Definition

The cost of an assignment $\alpha : X \to \mathcal{F}_2$ on F is

$$\text{cost}(\alpha, F) = \sum_{i \in [m]} w_i \chi_i(\alpha),$$

where $\chi_i(\alpha) = 1$ if $f_i(\alpha) \neq 0$ and 0 otherwise.

We need to find the minimum cost for any H-compatible assignment α.

Definition

$$\text{cost}_H(F) = \min_{\alpha H\text{-compatible}} \text{cost}(\alpha, F).$$

If F is satisfiable using a H-compatible assignment, then $\text{cost}_H(F) = 0$.

Encoding maxcut problem as weighted Polynomial equations

Definition

Maxcut of a graph $G = (V, E)$**:** The largest possible number of cross-edges ($\in E$) you can get by partitioning the V into two separate sets.

- Let $X = \{x_v | x_v \in V\}$, to partition these vertices into two sets, assign each x_v to either 0 or 1.
- For cross-edges $(u, v) \in E$, $x_u = 0$ & $x_v = 1$ or vice-versa.

Consider the following equations

$$F := \{[x_u + x_v + 1, 1] : (u, v) \in E\}$$
$$cost(F) = \text{max-cut}(G)$$

Note: In $\mathbb{F}_2$: $1 + 1 = 0$, so $1 + 1 + 1 = 1 \neq 0$, so such equations are not satisfied.

CNFs encoding as polynomials

- The set of polynomials may come by encoding the CNFs.
- We encode CNFs using twin variables.
- We call twin variables the set of variables $X = \{x_1, \cdots, x_n, x'_1, \cdots, x'_n\}$.
- The meaning of $x'_i = 1 - x_i$.
- For every clause $C = \{x_i \ : \ i \in I\} \cup \{\neg x_j \ : \ j \in J\}$, we associate a monomial

$$M(C) = \prod_{i \in I} x'_i \prod_{j \in J} x_j$$

in the twin variables X.

CNFs encoding as polynomials

- A set of clauses $\{C_1, \cdots, C_m\}$ can be encoded as

$$\{M(C_1), \cdots, M(C_m)\} \cup \{x_i^2 - x_i, x_i + x_i' - 1 \ : \ i \in [n]\}$$

- Any assignment $\alpha : \{x_1, \cdots, x_n\} \to \{0, 1\}$ can be extended to an assignment $\beta : X \to \{0, 1\}$, where for each ı $\in [n]$, $\beta(x_i) = \alpha(x_i)$ and $\beta(x_i') = 1 - \alpha(x_i)$.
- Then α satisfies a CNF formula if and only if β satisfies the polynomial encoding of F (that is, β is a common solution of the polynomials).

Recall: $C = \{x_i \ : \ i \in I\} \cup \{\neg x_j \ : \ j \in J\}$
$M(C) = \prod_{i \in I} x_i' \prod_{j \in J} x_j$

Polynomial calculus for MaxSAT: $wPC_{\mathcal{F}_2,\mathbb{N}}$

- Recall, $\mathcal{F}_2$ is a finite field with two elements $\{0, 1\}$.
- For each element $a \in \mathcal{F}_2$, we have $a^2 = a, 2 \cdot a = a + a = 0$.
- An inference rule is sound, if for every assignment α, the cost of set of hypothesis on α equals the cost of the conclusions on α.
- There are two types of inference rules:
  1) Structural rules: the Fold, Unfold and the H-simplification,
  2) proper inference rules: Sum and Split.

Fold-unfold

Fold-unfold Let F, G be two multi-sets of weighted polynomials, we say that F and G are fold-unfold equivalent ($F \approx G$), if one can be derived by another using the following inference rules

$$\text{Fold: } \frac{[f, u] \quad [f, w]}{f, u + w}$$

$$\text{0-Fold} \frac{[f, 0]}{}$$

$$\text{Unfold: } \frac{[f, u + w]}{[f, u] \quad [f, w]}$$

$$\text{0-Unfold: } \frac{}{[f, 0]}$$

where, $f \in \mathcal{F}_2$ and $u, w \in \mathbb{Z}$.

H-equivalence

H-equivalence Two function $f, g \in \mathcal{F}_2[X]$ are H-equivalence, if for every H-compatible assignment α, $f(\alpha) = g(\alpha)$. This can be seen with the following inference rule:

$$H\text{-equivalence}\frac{[f, w]}{[g, w]}$$

where $f, g \in \mathcal{F}_2[X]$ and $w \in \mathbb{Z}$ are such that for every H-compatible assignment $\alpha : X \to \mathcal{F}_2$, $f(\alpha) = g(\alpha)$.

- Note: Checking whether two polynomials f and g are H-equivalent might be problematic. It depends on H.
- When polynomials come from CNFs, the H-equivalence can be checked efficiently.

Split

Split We also have the following inference rule in $wPC_{\mathcal{F}_2,\mathbb{N}}$ and $wPC_{\mathcal{F}_2,\mathbb{Z}}$:

$$\text{Split}\frac{[f, w]}{[fg, w] \quad [f(g+1), w]}$$

for all $f, g \in \mathcal{F}_2[X]$ and $w \in \mathbb{Z}$.

- Split rule is sound: if $f(\alpha) = 0$, then both the conclusions $f(\alpha).g(\alpha)$ and $f(\alpha).(g(\alpha)+1)$ are 0.
  If $f(\alpha) = 1$, then exactly one of the conclusions is 1: if $f(\alpha).g(\alpha) = 1$, then $f(\alpha).(g(\alpha)+1) = 0$ (since $1+1=0$)

Sum rule

Sum $wPC_{\mathcal{F}_2,\mathbb{N}}$ and $wPC_{\mathcal{F}_2,\mathbb{Z}}$ also contains the following rule

$$\text{Sum}\frac{[f,w] \quad [g,w]}{[f+g,w] \quad [fg,2w]}$$

where $f, g \in \mathcal{F}_2[X]$ and $w \in \mathbb{Z}$.

- Sum rule is sound: similar to split, one can argue that the sum rule is sound. We briefly explain why $2w$ appears in the conclusion.
- It comes when both $f(\alpha) = 1$ and $g(\alpha) = 1$. In this case, $f(\alpha) + g(\alpha) = 0$ and $f(\alpha).g(\alpha) = 1$. Therefore, the conclusion fg should contain the sum of weights of both the hypothesis, which is two.

$wPC_{\mathcal{F}_2,\mathbb{N}}$ and $wPC_{\mathcal{F}_2,\mathbb{Z}}$ calculus

Definition (Bonacina, Bonet, Levi 2023)

Given a multi-set of weighted polynomials F and a set of hard constraints H, a $wPC_{\mathcal{F}_2,\mathbb{Z}}$ derivation of a weighted polynomial $[f, w]$ from F and H is a sequence of multi-sets $L_0, \cdots, L_\ell$ such that

- $L_0 = F$,
- $[f, w] \in L_\ell$ and all the other weighted polynomials $[f', w'] \in L_\ell$ have $w' \in \mathbb{N}$ (they all have non-negative weights), and
- for each $i > 0$ either $L_i \approx L_{i-1}$ (fold-unfold equivalent) or L_i is the result of an application of the split/sum/H-equivalence rule on L_{i-1}.

If all weights belong to $\mathbb{N}$, we call the system as $wPC_{\mathcal{F}_2,\mathbb{N}}$. The size of a $wPC_{\mathcal{F}_2,\mathbb{N}}/wPC_{\mathcal{F}_2,\mathbb{Z}}$ derivation $L_0, \cdots, L_\ell$ is the total number of occurrences of symbols in $L_0, \cdots, L_\ell$.

Soundness of $wPC_{\mathcal{F}_2,\mathbb{Z}}$

Theorem

Given $F = \{[f_1, w_1], \cdots, [f_m, w_m]\}$ where $f_i \in \mathcal{F}_2[X]$ and a set of polynomials $H \subseteq \mathcal{F}_2[X]$, if there is a $wPC_{\mathcal{F}_2,\mathbb{Z}}$ derivation of $[1, w]$ from F and H, then $cost_H(F) \geq w$.

Proof.

Let $L_0, \cdots, L_s$ be a $wPC_{\mathcal{F}_2,\mathbb{Z}}$ derivation of $[1, w]$.
Clearly, $cost_H(L_S) \geq w$: since $[1, w] \in L_s$ and weights of all other polynomials in L_s are non-negative.
Since all rules are sound, $cost_H(F) \geq w$. □

Completeness of $wPC_{\mathcal{F}_2,\mathbb{N}}$

Theorem (Bonacina, Bonet, Levi 2023)

Given F a set of weighted polynomials over $\mathcal{F}_2[X]$, there is a $wPC_{\mathcal{F}_2,\mathbb{N}}$ derivation of $[1, cost_H(F)]$ from F and hard constraints H.

- The completeness proof uses the concept of saturation from the MaxSAT resolution proof system.
- Let us define saturation first.

x-saturated set

Definition

We say a polynomial f depends on a variable x if for every polynomial g not containing x (and also x' in case of twin variables), $f \not\equiv g$.

Definition (x-saturated set)

Let $x \in X$ and S be a set of weighted polynomials. The set S is x-saturated if every H-compatible assignment $\alpha : X \to \mathcal{F}_2$ can be modified in x to a H-compatible assignment satisfying all weighted polynomials in S that depends on x.

As in MaxSAT resolution proof system, there is an algorithm for saturation.

Completeness of $wPC_{\mathcal{F}_2,\mathbb{N}}$

Lemma

In the context of H the Boolean axiom, for every set of weighted polynomials S and every variable x, there is a $wPC_{\mathcal{F}_2,\mathbb{N}}$ derivation of a set of polynomials S' which is x-saturated.

By iterating the saturation on all the variables one by one, the completeness is straightforward.

Lemma

Let $H \subset \mathcal{F}_2[X]$. If for every set of weighted polynomials S and for every variable x, there is a $wPC_{\mathcal{F}_2,\mathbb{N}}$ derivation of a set of polynomials S' which is x-saturated, then for every set of weighted polynomials F over $\mathcal{F}_2[X]$ there exists a $wPC_{\mathcal{F}_2,\mathbb{N}}$-derivation of $[1, cost_H(F)]$ from F and the hard constraints H.

Conclusion and Open Problems

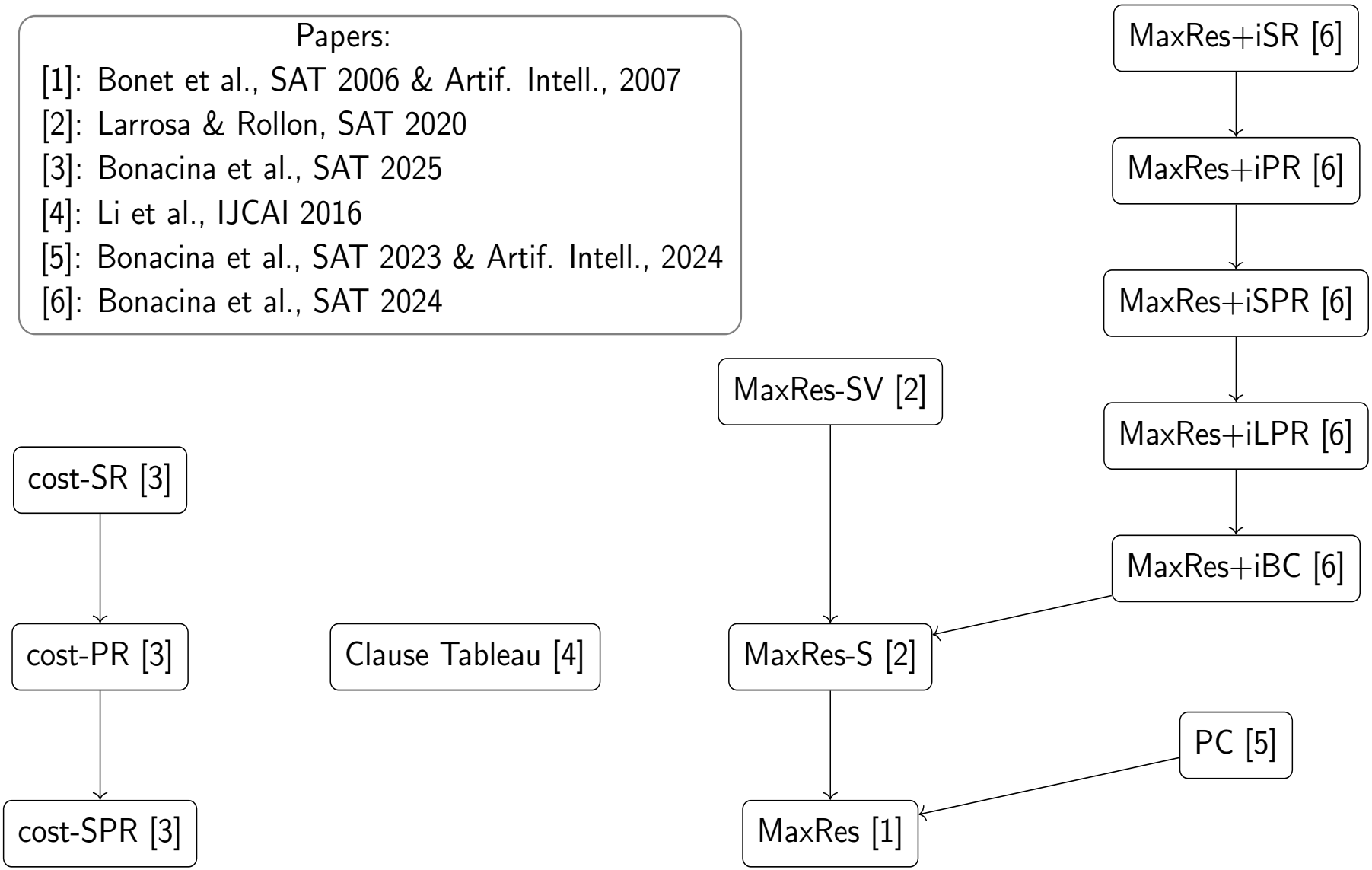
- In this talk, we have discussed four MaxSAT proof systems.

Open Problems:

- Most of the simulation results among the discussed proof systems are not known.
- Does cost-SR p-simulate MaxSAT resolution proof system?
- MaxSAT-Resolution + iSR vs MaxSAT-Resolution + cost-SR?
- How to extend other propositional proof systems for MaxSAT?

MaxSAT proof systems

Papers:
[1]: Bonet et al., SAT 2006 & Artif. Intell., 2007
[2]: Larrosa & Rollon, SAT 2020
[3]: Bonacina et al., SAT 2025
[4]: Li et al., IJCAI 2016
[5]: Bonacina et al., SAT 2023 & Artif. Intell., 2024
[6]: Bonacina et al., SAT 2024



Thank you.

Bibliography I

[1] Maria Luisa Bonet, Jordi Levy, and Felip Manyà. "Resolution for Max-SAT". In: *Artif. Intell.* 171.8-9 (2007), pp. 606–618.

[2] Javier Larrosa and Emma Rollon. "Towards a Better Understanding of (Partial Weighted) MaxSAT Proof Systems". In: *SAT 2020*. Vol. 12178. Lecture Notes in Computer Science. Springer, 2020, pp. 218–232.

[3] Ilario Bonacina et al. "Redundancy rules for MaxSAT". In: *28th International Conference on Theory and Applications of Satisfiability Testing, SAT 2025*. 2024.

[4] Chu Min Li, Felip Manyà, and Joan Ramon Soler. "A Clause Tableau Calculus for MaxSAT". In: *IJCAI 2016*. Ed. by Subbarao Kambhampati. IJCAI/AAAI Press, 2016, pp. 766–772.

Bibliography II

[5] Ilario Bonacina, Maria Luisa Bonet, and Jordi Levy. "Polynomial Calculus for MaxSAT". In: *SAT 2023*. Vol. 271. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023, 5:1–5:17.

[6] Ilario Bonacina, Maria Luisa Bonet, and Massimo Lauria. "MaxSAT Resolution with Inclusion Redundancy". In: *SAT 2024*. Vol. 305. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024, 7:1–7:15.

Cost-SR example

A clause C is cost substitution redundant (cost-SR) w.r.t. Γ if there exists a substitution σ such that

- $\Gamma|_{\overline{C}} \vdash_1 (\Gamma \cup \{C\})|_\sigma$ (redundancy)
- for all total assignment $\tau \supseteq \overline{C}, \text{cost}(\tau \circ \sigma) \leq \text{cost}(\tau)$ (cost)

Let $F = (x) \wedge (\overline{x}) \implies H' := (x \vee b_1) \wedge (\overline{x} \vee b_2)$

$C_1 = (b_1 \vee \overline{b_2} \vee \overline{x})$ is cost-SR w.r.t. H' with $\sigma = \{x = 0, b_2 = 0, b_1 = 1\}$.

- $\overline{C} = \{b_1 = 0, b_2 = 1, x = 1\}$
- the redundancy rule is trivially true with tautologies on both sides
- for the cost rule, $\tau = \overline{C}$ and both $\text{cost}(\alpha) = \text{cost}(\tau) = 1$.

By resolutions, derive $(b_1 \vee b_2), (b_1 \vee \overline{b_2})$ and resolve again for (b_1)

$C_2 = (\overline{b_1} \vee \overline{b_2})$ is cost-SR with $\sigma = \{x = 0, b_1 = 1, b_2 = 0\}$.
Resolve to derive $(\overline{b_2})$. Therefore, Min-unSAT(F)= 1