Redundancy Rules for MaxSAT

Massimo Lauria

Sapienza Università di Roma

joint with

Ilario Bonacina

Maria Luisa Bonet

Sam Buss

SAT 2025, Glasgow

The context

Redundancy rules: *adding clause C to the database Γ*

- $C \land \Gamma$ satisfiable **iff** Γ satisfiable
- models preprocessing / inprocessing in SAT. **[JHB’12]**
- studied from proof complexity perspective
- what about **MaxSAT**? **[BMM’13, BJ’19]**

Our contribution: simple proof system for MaxSAT redundancy

Our contribution: simple proof system for MaxSAT redundancy

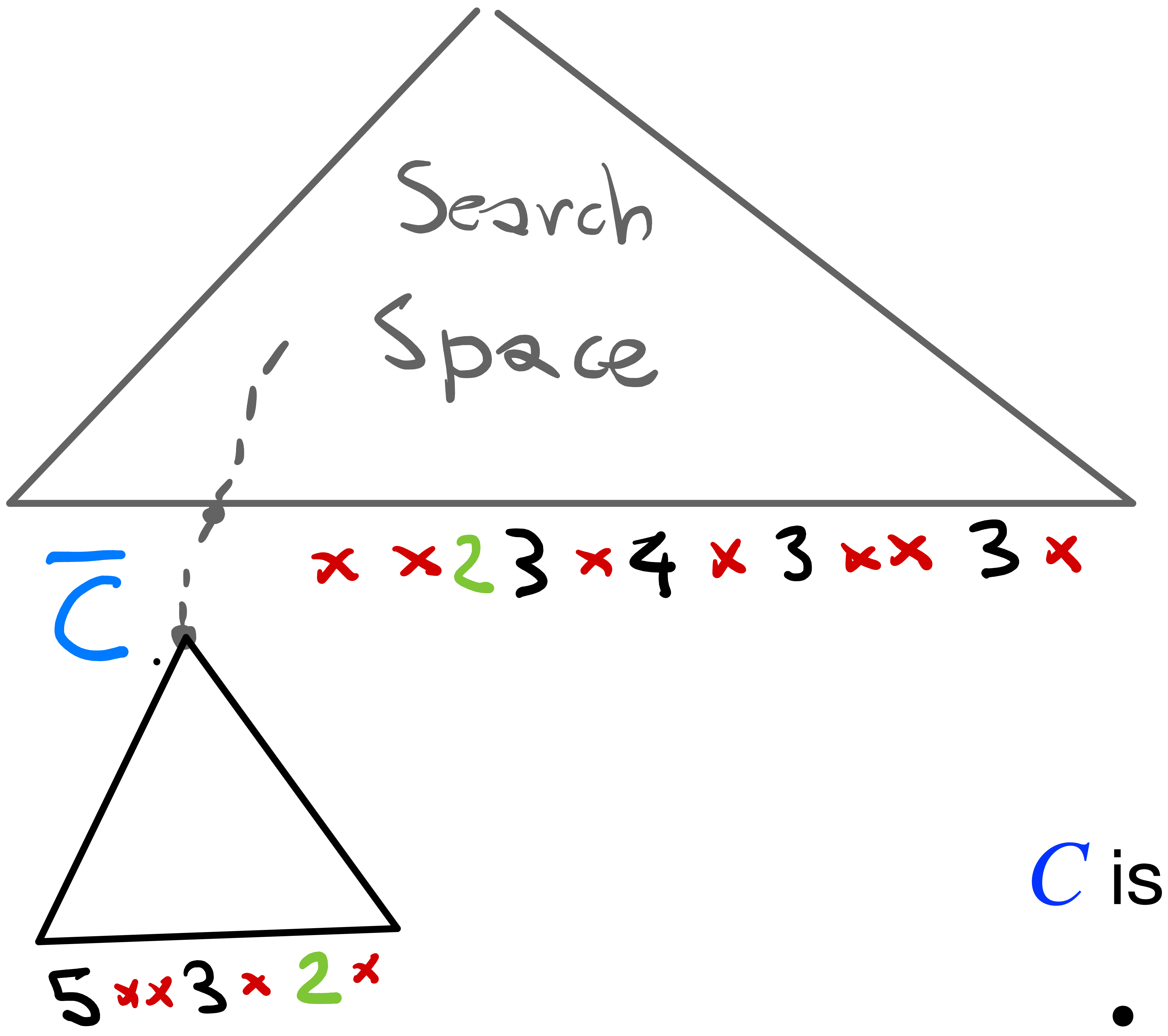
- extension of proof system RAT/LPR, SPR, PR, SR, … [see **BT’21**]
- polynomially verifiable
- proof power only depends on the redundancy rule (e.g. LPR, SPR, …)
- **GOAL:** amenable to proof complexity analysis
- **BONUS:** maybe easy to integrate with tools as `dpr-trim`

MaxSAT

$$\underbrace{S_1 \wedge S_2 \wedge \ldots}_{\text{soft clauses}} \quad \bigwedge \quad \underbrace{H_1 \wedge H_2 \wedge \ldots}_{\text{hard clauses}}$$

or, equivalently

$$\underbrace{(b_1 \vee S_1) \wedge (b_2 \vee S_2) \wedge \ldots \wedge H_1 \wedge H_2 \wedge \ldots}_{\text{hard clauses}} \qquad \text{cost} := \sum_i b_i$$

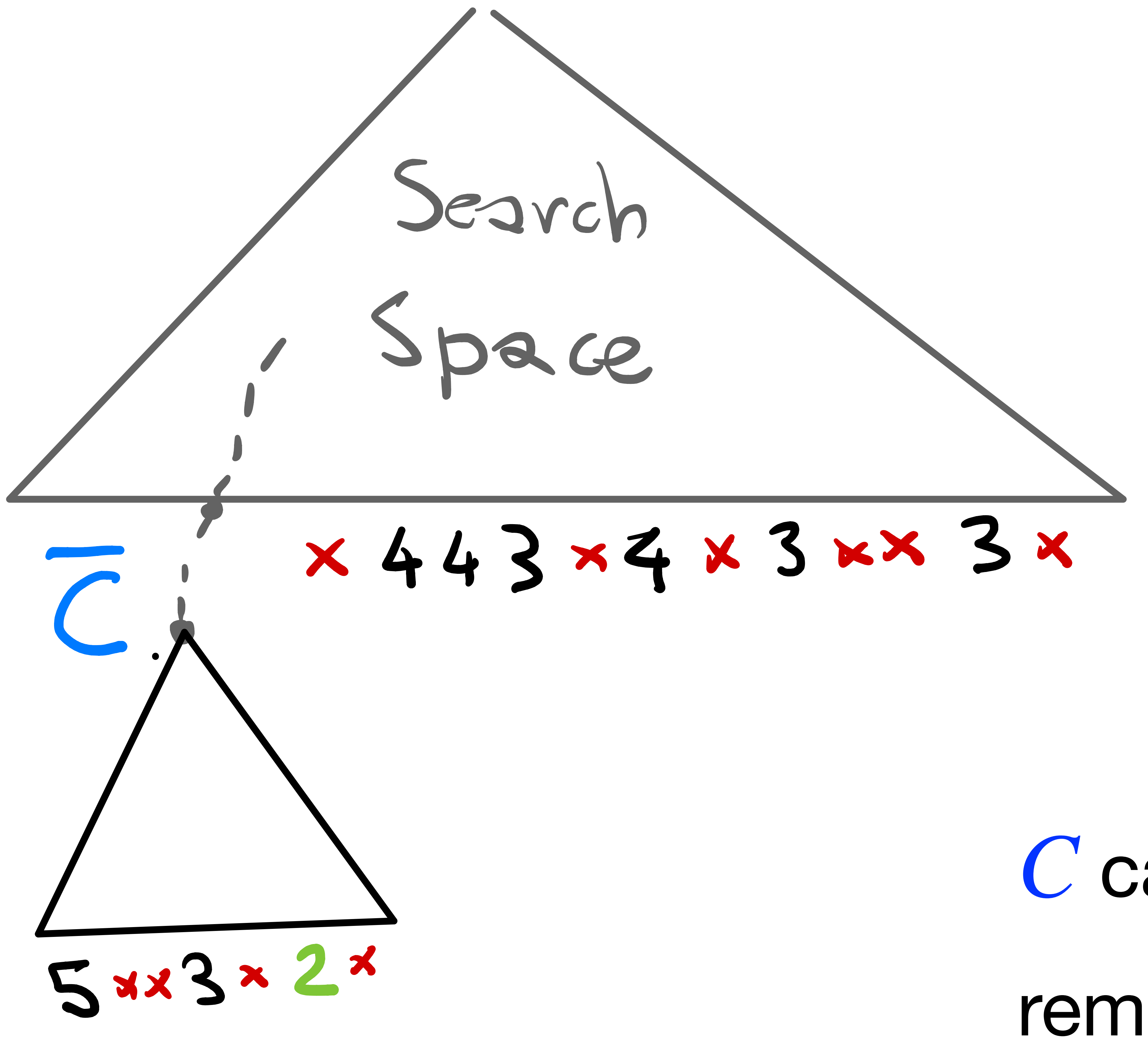


Add C to clause database

C is **redundant**

- search space reduction
- some **optimal** solutions lost
- but not **all**

Add C to clause database

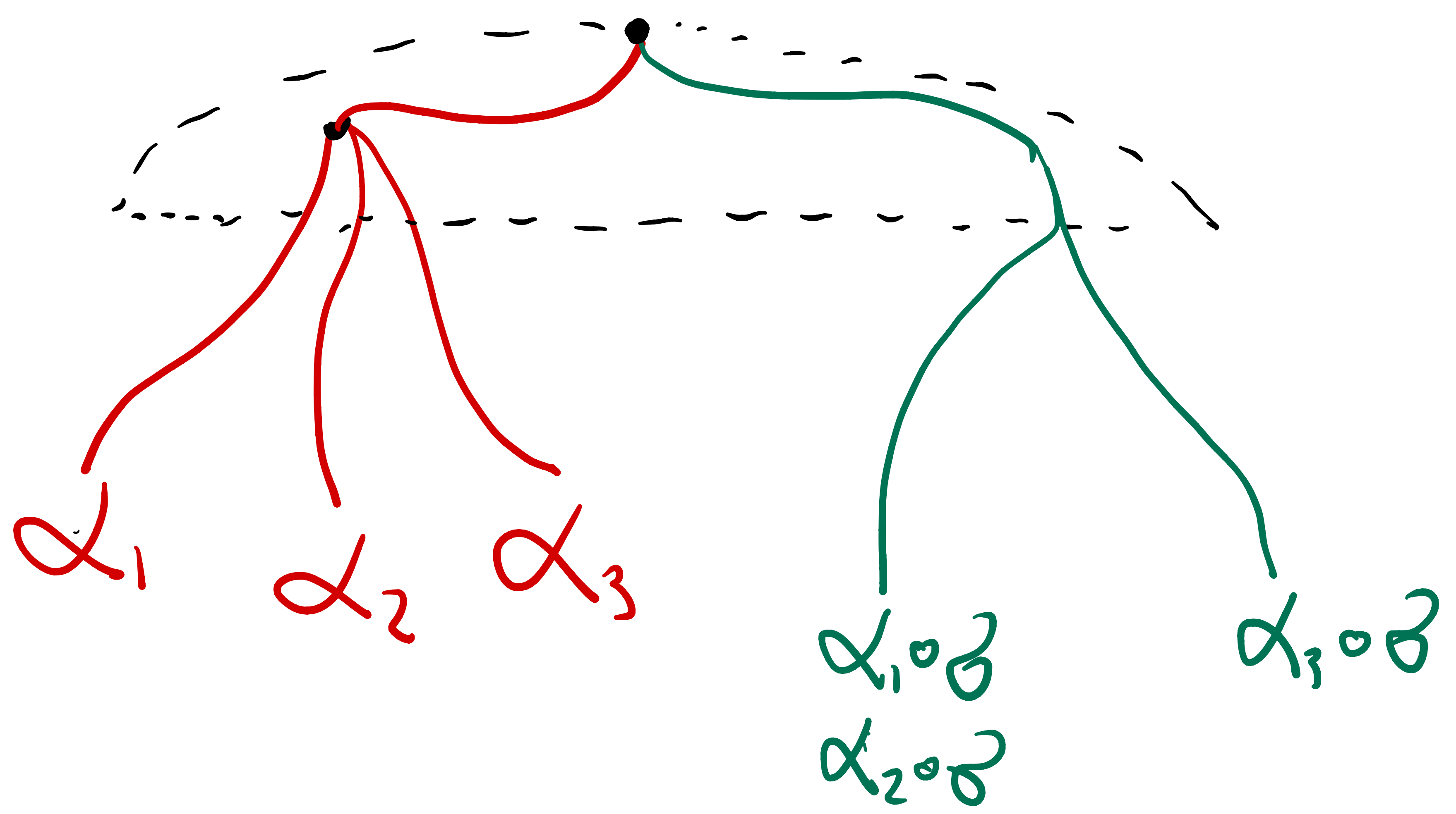


C cannot be added because it removes **all optimal solutions**

Short witness of SAT redundancy

Clause database Γ, proof that C is redundant is a **witnessing substitution** σ

if α satisfies $\Gamma \wedge \neg C$, then $\alpha \circ \sigma$ satisfies $\Gamma \wedge C$



σ uniformly fixes all problematic α

$$\equiv$$

$$\Gamma \restriction_{\neg C} \vdash_1 (\Gamma \wedge C) \restriction_{\sigma}$$

[...,BT'21, ...]

Substitution Redundancy (SR) $\sigma : \{\text{variables}\} \longrightarrow \{\text{literals}\} \cup \{0,1\}$

Propagation Red. (PR) σ is a partial assignment

Set Propagation Red. (SPR) σ only sets variables in C

Literal Propagation Red. (LPR, RAT) σ only sets one variable in C

*in this work: no new variables and no deletions

cost-SR / cost-PR / cost-SPR / cost-LPR

A new clause C can be added to Γ when

cost-SR / cost-PR / cost-SPR / cost-LPR

A new clause C can be added to Γ when

- there is a witnessing substitution σ so that

$$\Gamma \restriction_{\neg C} \vdash_1 (\Gamma \wedge C) \restriction_{\sigma} \qquad \text{(sat-redundancy)}$$

cost-SR / cost-PR / cost-SPR / cost-LPR

A new clause C can be added to Γ when

- there is a witnessing substitution σ so that

$$\Gamma \restriction_{\neg C} \vdash_1 (\Gamma \wedge C) \restriction_{\sigma} \qquad \text{(sat-redundancy)}$$

- whenever α falsifies C, $\mathrm{cost}(\alpha \circ \sigma) \leq \mathrm{cost}(\alpha)$ (cost)

cost-SR / cost-PR / cost-SPR / cost-LPR

A new clause C can be added to Γ when

- there is a witnessing substitution σ so that

$$\Gamma \restriction_{\neg C} \vdash_1 (\Gamma \wedge C) \restriction_{\sigma}$$ (sat-redundancy)

- whenever α falsifies C, $\mathrm{cost}(\alpha \circ \sigma) \leq \mathrm{cost}(\alpha)$ (cost)

- **equivalently**, $\left(\Sigma_i b_i - \Sigma_i \sigma(b_i)\right)\Big|_{\neg C}$ must be non-negative **(EASY!)**

cost-SR / cost-PR / cost-SPR / cost-LPR

A new clause C can be added to Γ when

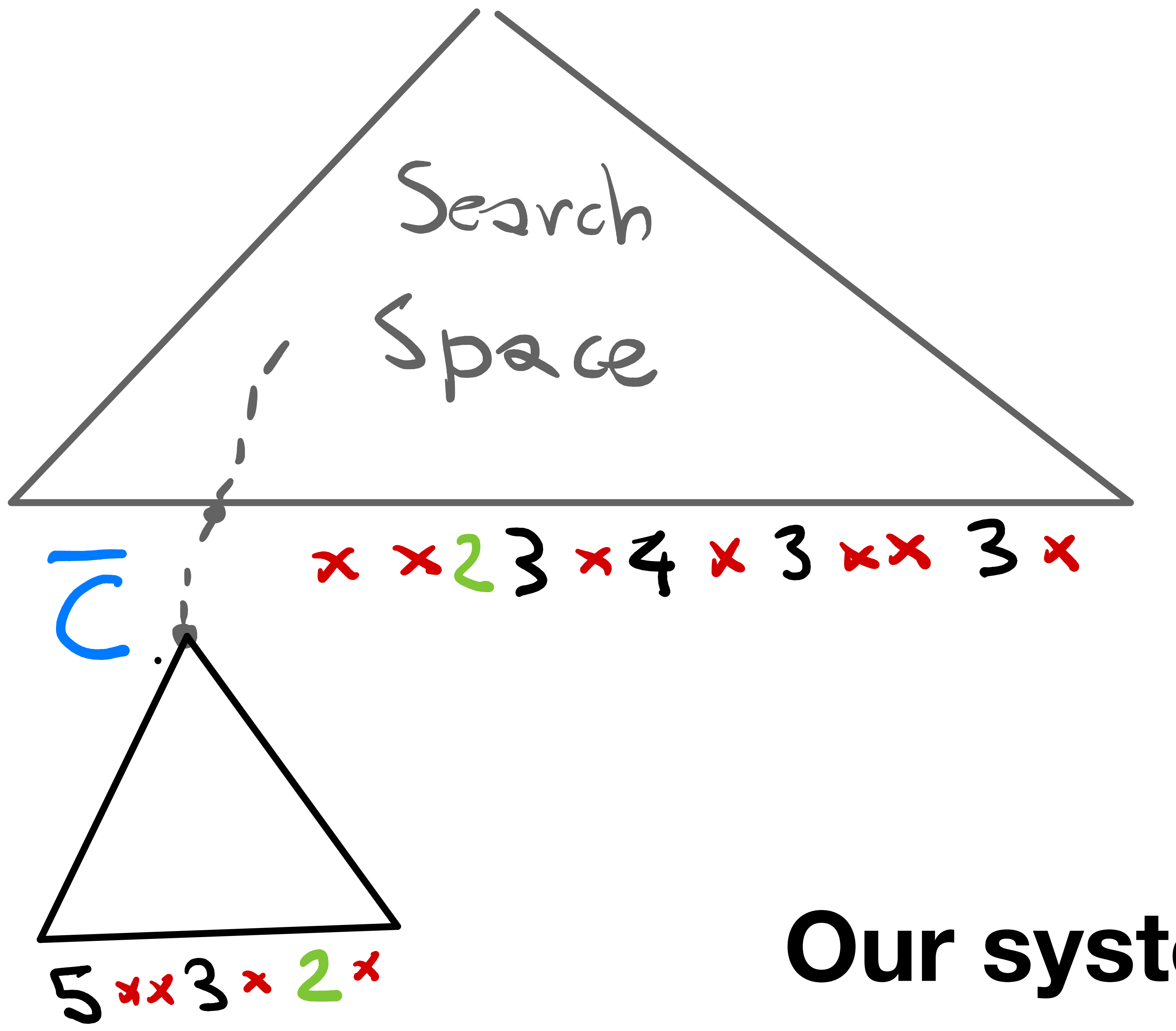
- there is a witnessing substitution σ so that

$$\Gamma \restriction_{\neg C} \vdash_1 (\Gamma \wedge C) \restriction_\sigma$$ (sat-redundancy)

- whenever α falsifies C, $\mathrm{cost}(\alpha \circ \sigma) \leq \mathrm{cost}(\alpha)$ (cost)

- **equivalently**, $\left(\Sigma_i b_i - \Sigma_i \sigma(b_i)\right)\Big|_{\neg C}$ must be non-negative **(EASY!)**

A proof that $\mathrm{cost}(F) \geq k$ is a derivation of unit clauses $b_{i_1}, b_{i_2}, \ldots, b_{i_k}$ from F



A deal with the devil…

Our system: check $\mathrm{cost}(\alpha \circ \sigma) \leq \mathrm{cost}(\alpha)$ whenever α falsifies C

Actual redundancy: check $\mathrm{cost}(\alpha \circ \sigma) \leq \mathrm{cost}(\alpha)$ whenever α falsifies C **and satisfies Γ**

Other observations…

Other observations…

Symmetry breaking is often beneficial in solving, but here is **a necessity**.

E.g. $b_1 \vee b_2 \quad b_3 \vee b_4 \quad b_5 \vee b_6 \quad \dots$

Mitigation: disjoint sets of b_is minimum HS

Other observations…

Symmetry breaking is often beneficial in solving, but here is **a necessity**.

E.g. $b_1 \vee b_2 \quad b_3 \vee b_4 \quad b_5 \vee b_6 \quad \ldots$

Mitigation: disjoint sets of b_is minimum HS

Cost requirement can be checked in **parallel**, and separately from redundancy

Other observations…

Symmetry breaking is often beneficial in solving, but here is **a necessity**.

E.g. $b_1 \lor b_2 \quad b_3 \lor b_4 \quad b_5 \lor b_6 \quad \ldots$

Mitigation: disjoint sets of b_is minimum HS

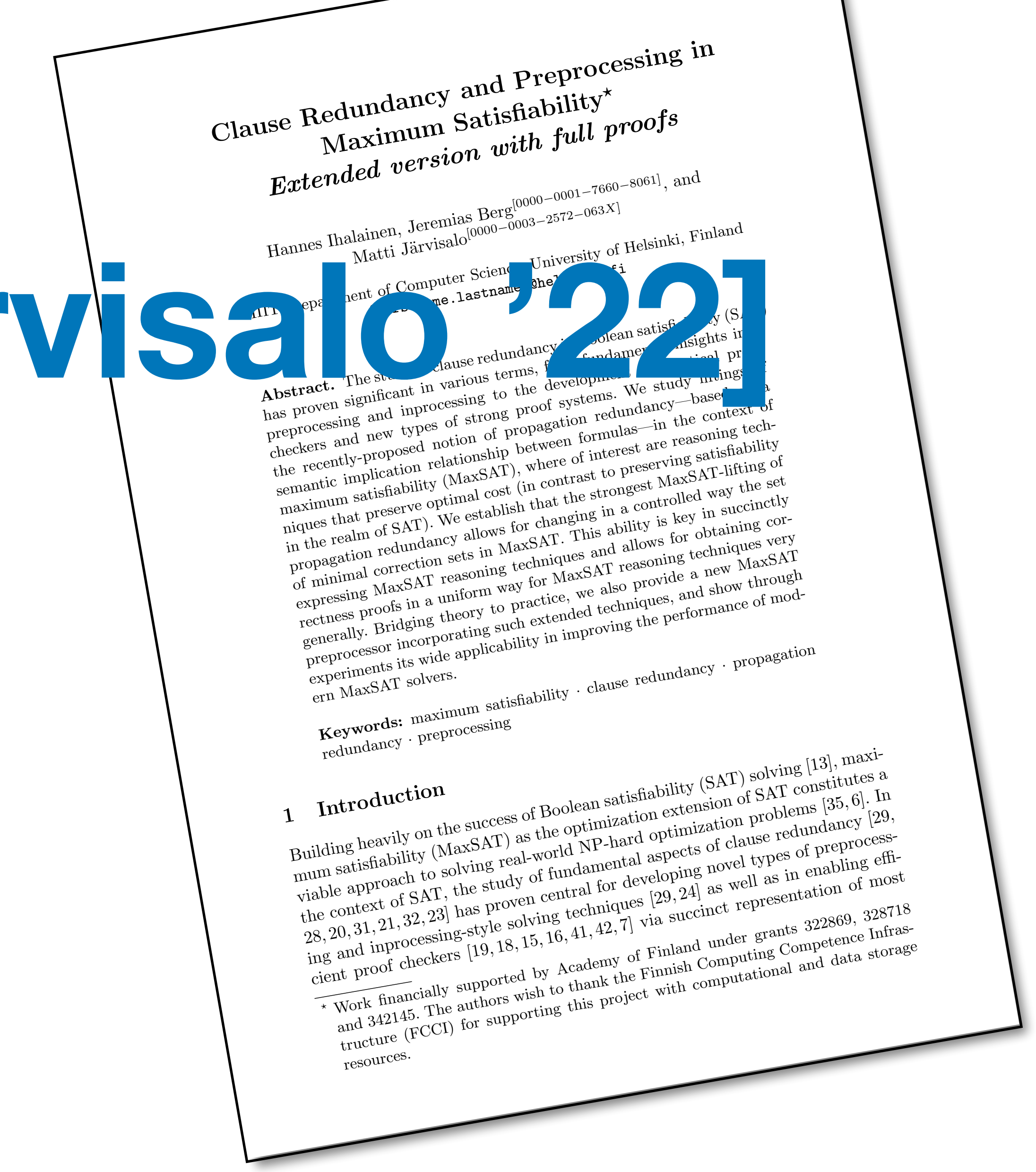
Cost requirement can be checked in **parallel**, and separately from redundancy

This is a minimum viable system that highlights redundancy in MaxSAT

Comparison with **some** other approaches

CPR

[Ihalainen, Berg, Järvisalo '22]



Clause Redundancy and Preprocessing in Maximum Satisfiability*
Extended version with full proofs

Hannes Ihalainen, Jeremias Berg[0000-0001-7660-8061], and Matti Järvisalo[0000-0003-2572-063X]

HIIT, Department of Computer Science, University of Helsinki, Finland
[illegible]

Abstract. The [illegible] clause redundancy in Boolean satisfiability (SAT) has proven significant in various terms, from fundamental insights into preprocessing and inprocessing to the development of [illegible] checkers and new types of strong proof systems. We study liftings of the recently-proposed notion of propagation redundancy—based on a semantic implication relationship between formulas—in the context of maximum satisfiability (MaxSAT), where of interest are reasoning techniques that preserve optimal cost (in contrast to preserving satisfiability in the realm of SAT). We establish that the strongest MaxSAT-lifting of propagation redundancy allows for changing in a controlled way the set of minimal correction sets in MaxSAT. This ability is key in succinctly expressing MaxSAT reasoning techniques and allows for obtaining correctness proofs in a uniform way for MaxSAT reasoning techniques very generally. Bridging theory to practice, we also provide a new MaxSAT preprocessor incorporating such extended techniques, and show through experiments its wide applicability in improving the performance of modern MaxSAT solvers.

Keywords: maximum satisfiability · clause redundancy · propagation redundancy · preprocessing

1 **Introduction**

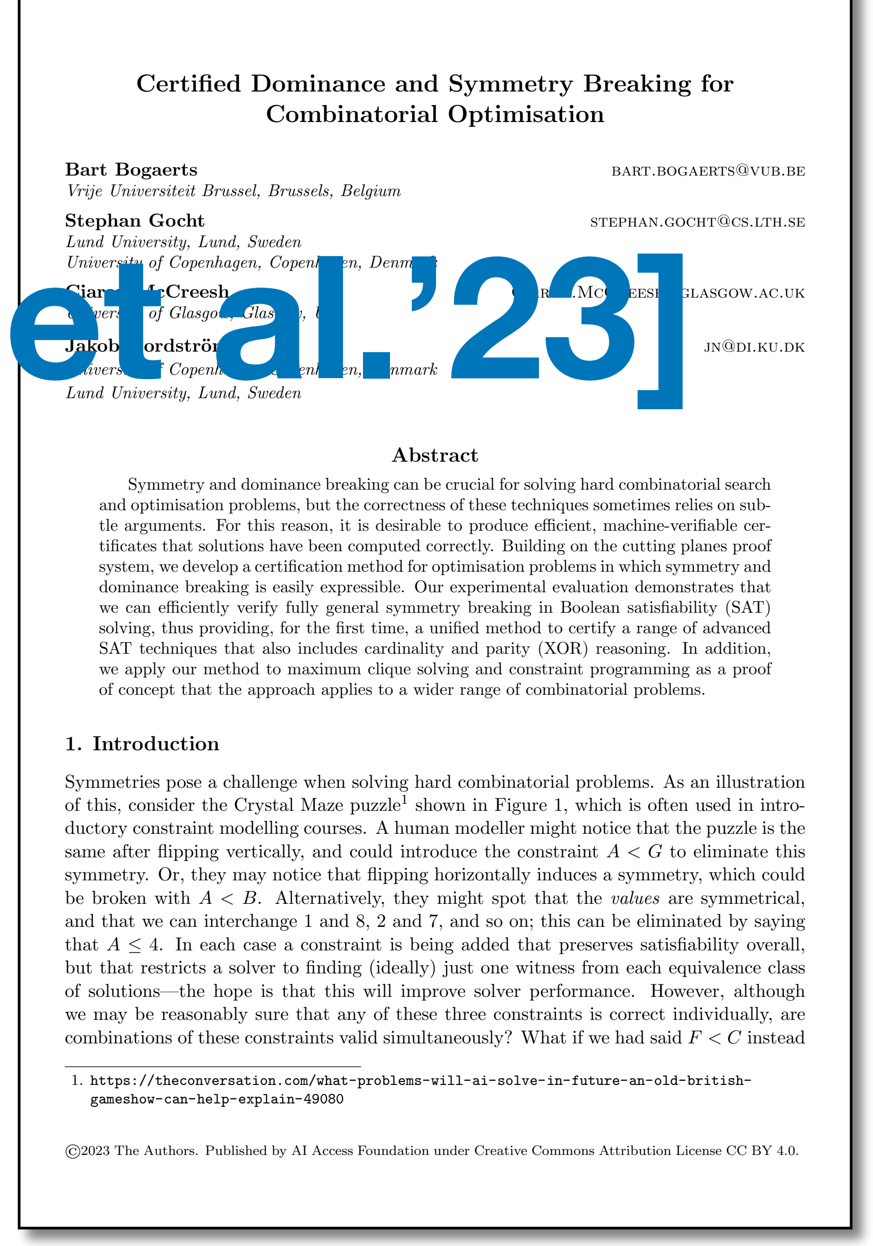
Building heavily on the success of Boolean satisfiability (SAT) solving [13], maximum satisfiability (MaxSAT) as the optimization extension of SAT constitutes a viable approach to solving real-world NP-hard optimization problems [35, 6]. In the context of SAT, the study of fundamental aspects of clause redundancy [29, 28, 20, 31, 21, 32, 23] has proven central for developing novel types of preprocessing and inprocessing-style solving techniques [29, 24] as well as in enabling efficient proof checkers [19, 18, 15, 16, 41, 42, 7] via succinct representation of most

* Work financially supported by Academy of Finland under grants 322869, 328718 and 342145. The authors wish to thank the Finnish Computing Competence Infrastructure (FCCI) for supporting this project with computational and data storage resources.

- cost condition: $\mathrm{cost}(F \wedge C, \sigma) = \mathrm{cost}(F, \sigma) \leq \mathrm{cost}(F, \neg C)$
- not polynomially checkable, requires a SAT call (i.e. additional proof)
- they introduce poly-checkable subsystems CSPR, CLPR

veriPB

[Bogaerts et al.'23]



Certified Dominance and Symmetry Breaking for Combinatorial Optimisation

Bart Bogaerts BART.BOGAERTS@VUB.BE
Vrije Universiteit Brussel, Brussels, Belgium

Stephan Gocht STEPHAN.GOCHT@CS.LTH.SE
Lund University, Lund, Sweden
University of Copenhagen, Copenhagen, Denmark

Ciaran McCreesh [illegible]GLASGOW.AC.UK
University of Glasgow, Glasgow, [illegible]

Jakob Nordström JN@DI.KU.DK
University of Copenhagen, Copenhagen, Denmark
Lund University, Lund, Sweden

Abstract

Symmetry and dominance breaking can be crucial for solving hard combinatorial search and optimisation problems, but the correctness of these techniques sometimes relies on subtle arguments. For this reason, it is desirable to produce efficient, machine-verifiable certificates that solutions have been computed correctly. Building on the cutting planes proof system, we develop a certification method for optimisation problems in which symmetry and dominance breaking is easily expressible. Our experimental evaluation demonstrates that we can efficiently verify fully general symmetry breaking in Boolean satisfiability (SAT) solving, thus providing, for the first time, a unified method to certify a range of advanced SAT techniques that also includes cardinality and parity (XOR) reasoning. In addition, we apply our method to maximum clique solving and constraint programming as a proof of concept that the approach applies to a wider range of combinatorial problems.

1. Introduction

Symmetries pose a challenge when solving hard combinatorial problems. As an illustration of this, consider the Crystal Maze puzzle[1] shown in Figure 1, which is often used in introductory constraint modelling courses. A human modeller might notice that the puzzle is the same after flipping vertically, and could introduce the constraint $A < G$ to eliminate this symmetry. Or, they may notice that flipping horizontally induces a symmetry, which could be broken with $A < B$. Alternatively, they might spot that the *values* are symmetrical, and that we can interchange 1 and 8, 2 and 7, and so on; this can be eliminated by saying that $A \leq 4$. In each case a constraint is being added that preserves satisfiability overall, but that restricts a solver to finding (ideally) just one witness from each equivalence class of solutions—the hope is that this will improve solver performance. However, although we may be reasonably sure that any of these three constraints is correct individually, are combinations of these constraints valid simultaneously? What if we had said $F < C$ instead

1. https://theconversation.com/what-problems-will-ai-solve-in-future-an-old-british-gameshow-can-help-explain-49080

- base language is **cutting planes**
- very expressive: redundancy, dominance, extension variables,…
- redundancy of C is expressible in `veriPB` itself, hence can be certified by a `veriPB` proof.
- `veriPB` easily simulates cost-SR

$\supseteq$

[Bonacina, Bonet, Lauria '24]

$$\Gamma \restriction_{\neg C} \vdash_1 (\Gamma \wedge C) \restriction_\sigma$$

- redundancy condition via multiset inclusion $\Gamma \restriction_{\neg C} \supseteq (\Gamma \wedge C) \restriction_\sigma$
- rule applies directly to soft clauses
- preserves # of falsified soft clauses
- can be integrated with MaxSAT resolution


MaxSAT Resolution with Inclusion Redundancy

Ilario Bonacina
Universitat Politècnica de Catalunya, Barcelona, Spain

Maria Luisa Bonet
Universitat Politècnica de Catalunya, Barcelona, Spain

Massimo Lauria
Sapienza Università di Roma, Italy

Some results about these systems

cost-SR is sound: only proves true cost bounds.

cost-SPR is complete:
(proof sketch) use an optimal assignment as witness σ, to block every other assignment α, with redundant clause $\neg\alpha$.

cost-LPR is incomplete

Upper bound

F is minimally unsat, with short refutation in PR

$$\longrightarrow$$

cost-PR has a short proof that $\mathrm{cost}(F) \geq 1$

Upper bound*

cost-SR has a short proof that $\mathrm{cost}(\mathrm{PHP}_n^m) \geq m - n$

*for refutation, system SPR is sufficient **[BT'21]**

The requirement of unit clauses $b_{i_1}, b_{i_2}, \ldots, b_{i_k}$ to prove $\mathrm{cost}(F) \geq k$ seems **rigid**

✗

b1	b2	b3	b4	b5
0	1	1	0	0

b1	b2	b3	b4	b5
1	0	0	1	0

✗

b1	b2	b3	b4	b5
0	0	0	1	1

The requirement of unit clauses $b_{i_1}, b_{i_2}, \ldots, b_{i_k}$ to prove $\mathrm{cost}(F) \geq k$ seems **rigid**

✗

b1	b2	b3	b4	b5
0	1	1	0	0

b1	b2	b3	b4	b5
1	0	0	1	0

✗

b1	b2	b3	b4	b5
0	0	0	1	1

$\mathrm{flip}(C, \sigma)$:= max HammingDistance(α, $\alpha \circ \sigma$), for α that falsifies C

Thm. Assuming any two optimal assignments of F have distance $\geq d$, and no b_i is determined in optimal assignments. Even proving $\mathrm{cost}(F) \geq 1$ requires a redundant C with witness σ and $\mathrm{flip}(C, \sigma) \geq d$.

To cut distant solutions σ must "fix" many variables

Corollary. There is a formula family F_n

with $O(n)$ variables, $O(n)$ clauses and $\mathrm{cost}(F_n) = \Omega(n)$

where, in order to prove $\mathrm{cost}(F_n) \geq 1$, any cost-SR proof derives a clause C

with $\mathrm{flip}(C, \sigma) = \Omega(n)$, where σ is its witnessing substitution.

Corollary. cost-LPR/cost-RAT is incomplete, since it can flip at most one variable

Corollary. cost-SPR can only flip variables in C, hence some C must be of large width

Summary

- A proof system for **understanding** redundancy in MaxSAT
- Potentially simpler to analyze, i.e. good for theory

Open Problems

- Our cost condition is easy to check, but too restrictive
- awkward to express $\mathrm{cost}(F) \geq k$ with $b_1, b_2, \ldots$
- cost-SR vs MaxSAT resolution
- lower bound for cost-SPR (could be easier than SPR)

Thank you!