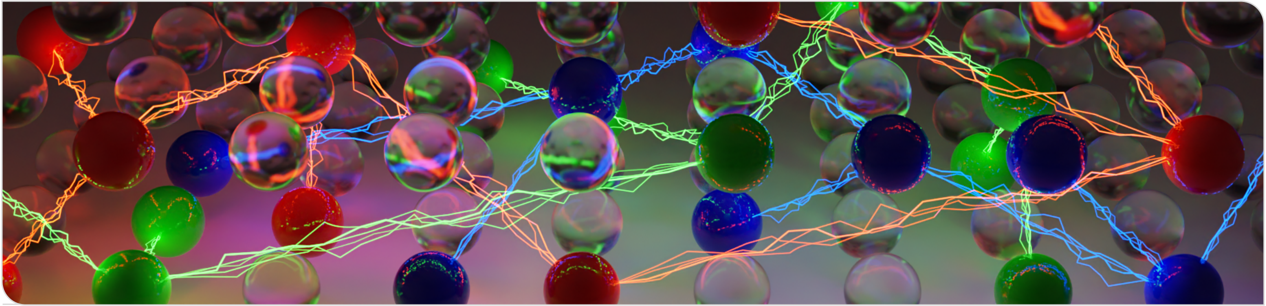
Karlsruher Institut für Technologie

Practical SAT Solving

Lecture 12

Markus Iser, Dominik Schreiber, Tomáš Balyo | July 8, 2024



Today: Preprocessing II and Proofs of Unsatisfiability

Karlsruher Institut für Technologie

Recap.

Preprocessing I (Lecture 7):

- Classic Preprocessing Techniques: Subsumption, Self-subsuming Resolution, Bounded Variable Elimination, Blocked Clause Elimination
- Relationship between Preprocessing Techniques and Gate Encodings

Today

- Preprocessing II: Propagation-based Redundancy Notions
- Proofs of Unsatisfiabilty

Preprocessing: Propagation-based Redundancy Notions

Let a formula F, and a literal x be given.

Failed Literal Probing

If $F \wedge x \vdash_{UP} \bot$, then $F \models \neg x$

$\implies$ add $\{\neg x\}$ to F

Example (Failed Literal Probing)

Let $F := \big\{\{a, b\}, \{a, \neg b\}\big\}$,

then probing $\neg a$ results in a conflict: $F \wedge \neg a \vdash_{UP} \bot$,

such that we can deduce $F \equiv F \wedge a$.

Preprocessing: Propagation-based Redundancy Notions

Let a formula F, a clause $C \in F$, and a literal $x \in C$ be given.

Asymmetric Literal Elimination (ALE)

If $F \setminus C \wedge \overline{C \setminus \{x\}} \vdash_{UP} \overline{x}$, then $F \models C \setminus \{x\}$.

$\implies$ strengthen C to $C \setminus \{x\}$

Example (Asymmetric Literal Elimination (ALE))

Let $F := \big\{\{a, b\}, \{\neg b, \neg c\}, \{a, c, d\}\big\}$,

and let $C := \{a, c, d\}$ and $x := c$,

then $\big\{\{a, b\}, \{\neg b, \neg c\}, \{\neg a\}, \{\neg d\}\big\} \vdash_{UP} \neg c$,

such that we can deduce $F \models \{a, d\}$.

F can not have a model which satisfies $\{a, c, d\}$, but not $\{a, d\}$.

Preprocessing: Propagation-based Redundancy Notions

Let a formula F, a clause $C \in F$, and a literal $x \in C$ be given.

Asymmetric Tautology Elimination (ATE)

If $F \setminus C \wedge \overline{C} \vdash_{UP} \bot$, then $F \models C$.

$\implies$ remove C from F

Example (Asymmetric Tautology Elimination (ATE))

Let $F := \{\{a, b, c\}, \{\neg b, d\}, \{a, c, d\}\}$,

and let $C := \{a, c, d\}$,

then $\{\{a, b, c\}, \{\neg b, d\}, \{\neg a\}, \{\neg c\}, \{\neg d\}\} \vdash_{UP} \bot$,

such that we can deduce $F \equiv F \setminus \{a, c, d\}$.

$\{a, c, d\}$ follows from the other clauses in F.

Preprocessing: Propagation-based Redundancy Notions

Variants and Optimizations

- **Hidden Tautology Elimination (HTE) / Hidden Literal Elimination (HLE)**
  Restricted forms of ATE/ALE which only propagate over binary clauses.
- **Distillation / Vivification**
  Interleave assignment and propagation to detect ATs / ALs early on.
- **Avoidance of Redundant Propagations**
  Sort literals and clauses in a formula to simulate a trie, and reuse propagations that share the same prefix.
  the binary implication graph and application of the *parenthesis theorem*.

Preprocessing: Scheduling of Preprocessing Techniques

At a point where one technique is unable to make further progress, another technique might be applicable and even modify the problem in a way that the first technique can make further progress.

Scheduling of Preprocessing Techniques

- **Heuristic Limits**
  Bound the number of applications of a technique.
- **Scheduling of Techniques**
  Non-trivial, benefit of techniques depends on the formula.
- **Interleaving of Techniques**
  Apply techniques in a round-robin fashion.
- **Inprocessing**
  Interleave search and preprocessing.

Autarkies

Let a formula F and a partial assignment A be given.

- A clause $C \in F$ is touched by A if it contains a variable assigned in A
- A clause $C \in F$ is satisfied by A if it contains a literal assigned to *True* by A

An autarky is a partial assignment A such that all touched clauses are satisfied.

Autarky

The partial assignment $A = \{\neg a, \neg c\}$ is an autarky for $F := \{\{\neg a, b\}, \{\neg a, c\}, \{a, \neg b, \neg c\}\}$

Autarky-based Clause Removal

All clauses touched by an autarky can be removed.

Edge Case: Pure Literals and Satisfying Assignments are Autarkies

Example Usage: Kissat analyses assignments found by sprints of local search to find autarkies.

Recap.

Preprocessing II

- Propagation-based Redundancy Notions: Failed Literal Probing, Asymmetric Literal Elimination, Asymmetric Tautology Elimination
- Autarkies: Partial assignments that satisfy all touched clauses
- Scheduling of Preprocessing Techniques

Next Up

Proofs of Unsatisfiability

Relationship with Proof Checking

Generalizations of Blocked Clauses

- **Reverse Unit Propagation (RUP)**
  A clause has the property RUP if and only if it is an Asymmetric Tautology (AT).

  In CDCL, learned clauses are RUP at the moment of their learning.

- **Resolution Asymetric Tautologies (RATs)**
  A clause C is a RAT in a formula F if it contains a literal x such that each resolvent in $C \otimes_x F_{\overline{x}}$ is an asymmetric tautology.

  Blocked Sets in particular are RATs.

Proof Checking

SAT Solvers are complex software systems, and bugs are not uncommon.

Trustworthiness of SAT Solvers

The output of a SAT solver is trustworthy because:

- For satisfiable instances, SAT solvers can output the found assignment
- For unsatisfiable instances, SAT solvers can output a proof of unsatisfiability
- Both can be checked independently from the solver by much simpler, and formally verified program

Feasibility: RAT proof checking is polynomial in the size of the proof, but the proof size is worst-case exponential in the size of the formula

Example (Applications)

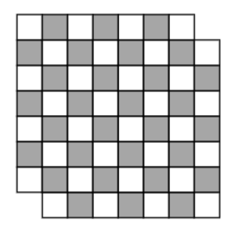
Many real-world instances are unsatisfiable, e.g., unsatisfiability of a formula witnesses . . .

- . . . the absence of certain bugs in a hardware design or software verification,
- . . . or the optimality of a certain makespan in planning
- . . .

Proof Systems: Motivating Example

Example (Mutilated Chessboards)

Let a chessboard be given with two diagonally opposite corners removed.
Is it possible to cover the remaining board with dominoes?

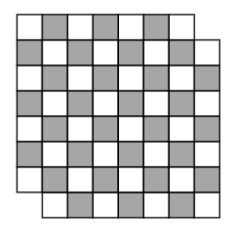


Proof Systems: Motivating Example

Example (Mutilated Chessboards)

Let a chessboard be given with two diagonally opposite corners removed.
Is it possible to cover the remaining board with dominoes?

Human: No, because each dominoe covers exactly one black and one white field, and there are two more black fields than white fields.

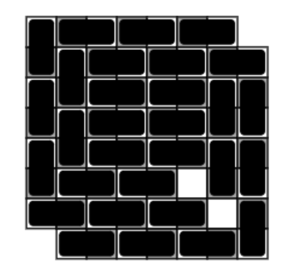


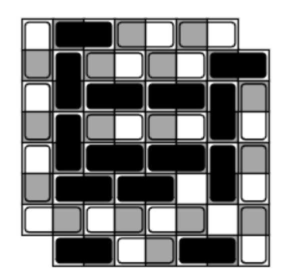
Proof Systems: Motivating Example

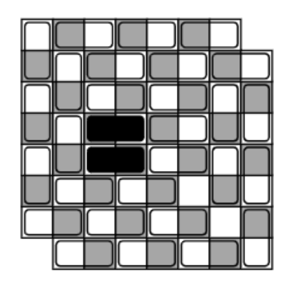
Example (Mutilated Chessboards)

Let a chessboard be given with two diagonally opposite corners removed.
Is it possible to cover the remaining board with dominoes?

SAT Solver: Let's try and learn.







Impasse Detected → Resolution → More powerful Proof-system

Proof Complexity

Proof complexity is the study of the size of proofs in different proof systems.

Relationship with SAT Solving

- Static analysis without algorithmic considerations
- Questions of automizability not addressed
- Lower bounds on proof size tell us how good a SAT solver can be in the best case
- Upper bounds on proof size tell us how good a SAT solver should be

Resolution vs. Extended Resolution

Karlsruher Institut für Technologie

Resolution

- Preserves equivalence
- CDCL is as powerful as General Resolution
- Exponential Lower Bounds: For example, proofs of unsatisfiabilty of Pigeon Hole formulas, are necessarily of exponential length in the resolution proof system (Haken 1985)

Extended Resolution (ER)

- Preserves satisfiability
- Extended Resolution (Tseitin 1966):
  incorporate extension rule $x \leftrightarrow a \wedge b$ for some a, b in formula and a new variable x
- No Lower Bounds known
- (Cook 1967) polynomial sized ER proof for PH formulas

Blocked Clauses (Kullmann 1999)

Blocked Clauses

- Preserves satisfiability
- Generalization of ER Proof systems
- Allows the addition and removal of blocked clauses
- No Lower Bounds known

Blocked Clauses (Kullmann 1999)

Blocked Clauses

- Preserves satisfiability
- Generalization of ER Proof systems
- Allows the addition and removal of blocked clauses
- No Lower Bounds known

Problem: Automizability, how to find such short proofs?

Structural Boundend Variable Addiction (SBVA) $\rightarrow$ SBVA CaDiCaL

- Select variables based on their centrality in the Variable Interaction Graph (VIG) of the formula, then add definitions over these variables in a preprocessing step
- Winner of the Main track of SAT Competition 2023
- First time in history of SAT competitions that a heuristic for ER could beat CDCL

Strong Proof-Systems without Introducing Variables

ER introduces new variables. Can we get stronger without introducing new variables?
In the following: C is redundant with respect to F means that F and $F \land C$ are equisatisfiable.

Example (Implication-based Redundancy)

Given $F := \{\{x, y, z\}, \{\neg x, y, z\}, \{x, \neg y, z\}, \{\neg x, \neg y, z\}\}$, and $G := \{\{z\}\}$,
G is at least as satisfiable as F since $F \models G$

Strong Proof-Systems without Introducing Variables

ER introduces new variables. Can we get stronger without introducing new variables?
In the following: C is redundant with respect to F means that F and $F \wedge C$ are equisatisfiable.

Example (Implication-based Redundancy)

Given $F := \{\{x, y, z\}, \{\neg x, y, z\}, \{x, \neg y, z\}, \{\neg x, \neg y, z\}\}$, and $G := \{\{z\}\}$,
G is at least as satisfiable as F since $F \models G$

Implication-based Redundancy Notion

A clause C is redundant w.r.t. formula F iff there exists an assignment ω such that $F \wedge \neg C \models (F \wedge C)|_\omega$[a]

In other words: Potential models of F falsifiying C are still models of F and C modulo an assignment ω

[a] Given an assignment α, the formula $F|_\alpha$ is the formula after removing from F all clauses that are satisfied and all literals falsified by α

Strong Proof-Systems without Introducing Variables

ER introduces new variables. Can we get stronger without introducing new variables?
In the following: C is redundant with respect to F means that F and $F \wedge C$ are equisatisfiable.

Example (Implication-based Redundancy)

Given $F := \{\{x, y, z\}, \{\neg x, y, z\}, \{x, \neg y, z\}, \{\neg x, \neg y, z\}\}$, and $G := \{\{z\}\}$,
G is at least as satisfiable as F since $F \models G$

Implication-based Redundancy Notion

A clause C is redundant w.r.t. formula F iff there exists an assignment ω such that $F \wedge \neg C \models (F \wedge C)|_\omega$ [a]

In other words: Potential models of F falsifiying C are still models of F and C modulo an assignment ω

[a] Given an assignment α, the formula $F|_\alpha$ is the formula after removing from F all clauses that are satisfied and all literals falsified by α

In Practice: Propagation Redundancy

Approximate $F \wedge \neg C \models (F \wedge C)|_\omega$ with unit propagation: $F \wedge \neg C \vdash_{UP} (F \wedge C)|_\omega$
$\rightarrow$ Efficiently Checkable Proofs: Tan et al., Verified Propagation Redundancy Checking in CakeML, TACAS 21

Theorem: Clause Redundancy via Implication

Let F be a formula, C a non-empty clause, and α the assignment blocked by C. Then, C is redundant with respect to F if and only if there exists an assignment ω such that ω satisfies C and $F|_\alpha \models F|_\omega$.

(Heule et al., 2019, Strong Extension-Free Proof Systems)

Proof "only if"

Assume F and $F \wedge C$ are equisatisfiable. Show that there exists an ω satisfying C and $F|_\alpha \models F|_\omega$.

If $F|_\alpha$ is unsatisfiable, then the semantic implication trivially holds.

Assume now that $F|_\alpha$ is satisfiable, implying that F is satisfiable.
Since F and $F \wedge C$ are equisatisfiable, there exists an assignment ω that satisfies both F and C.
Thus, since ω satisfies F, it holds that $F|_\omega = \emptyset$ and so $F|_\alpha \models F|_\omega$.

Theorem: Clause Redundancy via Implication

Let F be a formula, C a non-empty clause, and α the assignment blocked by C. Then, C is redundant with respect to F if and only if there exists an assignment ω such that ω satisfies C and $F|_\alpha \models F|_\omega$.

(Heule et al., 2019, Strong Extension-Free Proof Systems)

Proof "if"

Assume there exists an assignment ω satisfying C and $F|_\alpha \models F|_\omega$. Show that F and $F \wedge C$ are equisatisfiable.

Let γ be a (total) assignment that satisfies F and falsifies C. Then, we can turn γ into a satisfying assignment γ' for $F \wedge C$ as follows:
As γ falsifies C, it coincides with α on vars (C). Therefore, since γ satisfies F, it must satisfy $F|_\alpha$ and since $F|_\alpha \models F|_\omega$, it must also satisfy $F|_\omega$. Now, consider the following assignment γ' which clearly satisfies C:

$$\gamma'(x) = \begin{cases} \omega(x) & \text{if } x \in \text{vars}(\omega) \\ \gamma(x) & \text{otherwise} \end{cases}$$

Since γ satisfies $F|_\omega$, and $\text{vars}(F|_\omega) \subseteq \text{vars}(\gamma) \setminus \text{vars}(\omega)$, γ' satisfies F. Hence, γ' satisfies $F \wedge C$.

Pruning Branches with Conditional Autarkies

Theorem (Monien and Speckenmeyer 1985): Let α be an autarky of F. Then, F and $F|_\alpha$ are equisatisfiable.

Conditional Autarkies

An assignment $\alpha = \alpha_{con} \cup \alpha_{aut}$ is a conditional autarky of F if α_{aut} is an autarky of $F|_{\alpha_{con}}$

Then F and $F \wedge (\alpha_{con} \rightarrow \alpha_{aut})$ are equisatisfiable.

Example (Pruning Branches with a Conditional Autarky)

Let $F := \{\{x, y\}, \{x, \neg y\}, \{\neg y, \neg z\}\}$, and let $\alpha_{con} = \{x\}$ and $\alpha_{aut} = \{\neg y\}$.

Then $F|_{\alpha_{con}} = \{\neg y, \neg z\}$ and $\alpha_{aut} = \{\neg y\}$ is an autarky of $F|_{\alpha_{con}}$,
such that $\alpha = \{x, \neg y\}$ is a conditional autarky of F.

We can thus learn the clause $\{\neg x, \neg y\}$.

Satisfaction Driven Clause Learning (SDCL)

Idea: Also learn clauses if no conflict is detected, but a positive reduct is satisfiable.

Positive Reduct

Let $\alpha := \neg C$, the positive reduct $p(F, \alpha)$ is the formula that contains C and all clauses of F satisfied by α.

A satisfying assignment ω of the positive reduct $p(F, \alpha)$ is a conditional autarky of F.

Positive reducts are typically very easy to solve.

Key Idea of SDCL: While solving a formula, check the positive reducts of current assignments α for satisfiability, and if so, prune the branch α.

Heule et al., 2017, PRuning Through Satisfaction

Last slide

Karlsruher Institut für Technologie

Recap.

- Preprocessing II: Propagation-based Redundancy Notions, Autarkies, Scheduling of Preprocessing Techniques
- Proof Systems: Resolution, Extended Resolution, Blocked Clauses, Implication-based Redundancy
- Pruning Branches with Conditional Autarkies
- Satisfaction Driven Clause Learning (SDCL)