Un raccourci récursif pour CEGAR

Application au problème de satisfiabilité en logique modale K

Jean-Marie Lagniez, Daniel Le Berre,
Tiago de Lima, Valentin Montmirail

CRIL, Université d'Artois, Lens, France

Caen - 3 Juillet 2017

IJCAI-17 MELBOURNE

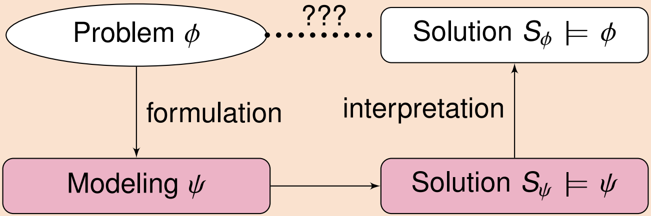
cnrs

UNIVERSITÉ D'ARTOIS

cril

Abstraction: Idea & Motivation

- Comes from: **Mathematical Modeling**

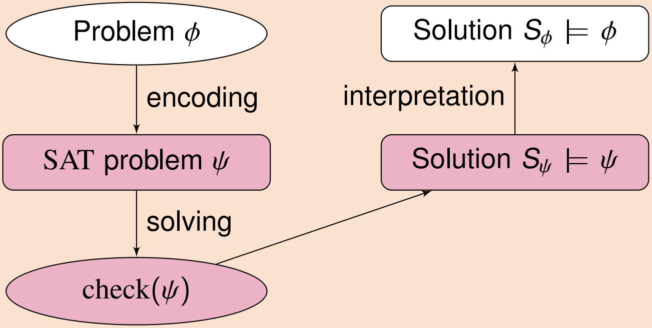


- Works for theoretical problems
- But what about practice?

Introduction: Abstraction via SAT

Modeling: Propositional Formula

- For many **NP** problems: Encoding into SAT



SAT solver

SAT solver

- Extremely efficient software
- Based on CDCL approach [SS99, MMZ+01]
- One of the current best is: Glucose [ES03, AS09] ☺
- Able to solve efficiently problems with $\approx 10^8$ clauses
- Able to give a “reason” why a formula is UNSAT [ES03]

SAT solver: One limitation

- What happen when the encoding of the problem is too big ?
- Could be solved ‘easily’ but will not because of memory...
- We need a SAT solver in a more complex procedure: CEGAR

IJCAI-17 MELBOURNE

UNIVERSITÉ D'ARTOIS

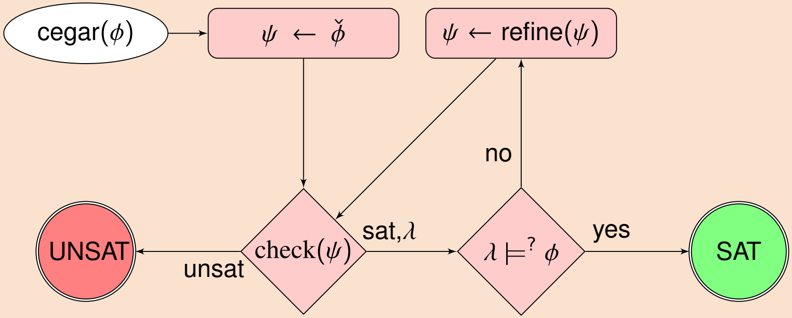
CEGAR: **C**ounter**E**xample **G**uided **A**bstraction **R**efinement

To solve a problem, consider only a part of it [CGJ+03]

- To abstract problems: hoping it will be easier to solve
- Two variants of abstraction:
  - Under-abstraction: abstraction has **more** solutions
  - Over-abstraction: abstraction has **less** solutions
- CEGAR-over: CEGAR approach using over-abstractions
- CEGAR-under: CEGAR approach using under-abstractions

CEGAR using under-abstractions

CEGAR-under



Example

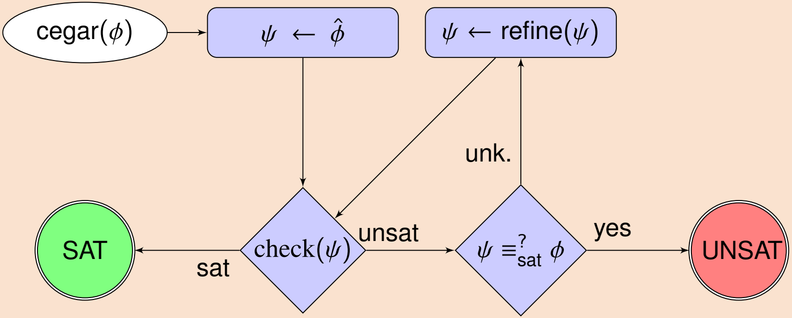
SAT problem, by increasing step by step, the number of clauses

IJCAI-17 MELBOURNE

UNIVERSITÉ D'ARTOIS

CEGAR using over-abstractions

CEGAR-over



Example

Planification problem, by increasing step by step, the horizon

CounterExample Guided Abstraction Refinement

Advantages

- If problem mainly satisfiable: CEGAR-over
- If problem mainly unsatisfiable: CEGAR-under
- Everytime check improves, CEGAR improves
- Many applications already use CEGAR

Drawbacks

- Not efficient when 50/50 chances of being SAT/UNSAT
- Not efficient when we need many refinement steps

IJCAI-17 MELBOURNE

UNIVERSITÉ D'ARTOIS

Recursive Explore and Check Abstraction Refinement

Recursive **E**xplore and **C**heck **A**bstraction **R**efinement

- Called *RECAR* [LLdLM17]
- Inspired by CEGAR [CGJ+03]
- Rely on 5 very important assumptions

RECAR Assumptions

1. Function ‘check’ is sound, complete and terminates
2. $isSAT(\hat{\phi})$ implies $isSAT(\text{refine}(\hat{\phi}))$
3. $\exists . n \in \mathbb{N}$ s.t. $\text{refine}^n(\hat{\phi}) \equiv^?_{\text{sat}} \phi$.
4. $\text{isUNSAT}(\check{\phi})$ implies $\text{isUNSAT}(\phi)$
5. $\exists n \in \mathbb{N}$ s.t. $RC(under^n(\phi), under^{n+1}(\phi))$ is false.

IJCAI-17 MELBOURNE

UNIVERSITÉ D'ARTOIS

Recursive Explore and Check Abstraction Refinement

$\exists n \in \mathbb{N}$ s.t. $RC(under^n(\phi), under^{n+1}(\phi))$ is false.

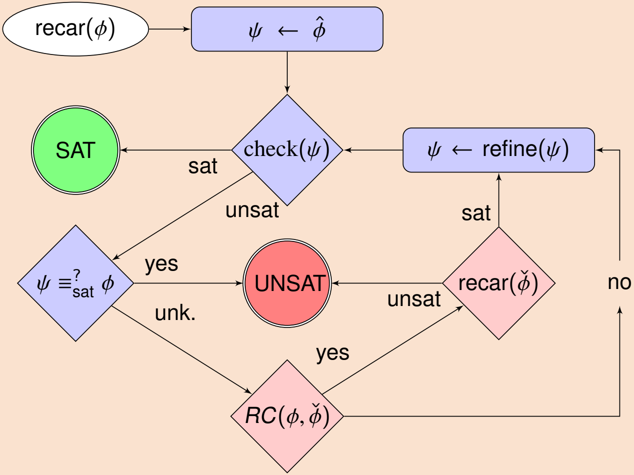
RC function

- 'true' if we can do a recursive call, 'false' otherwise
- It compares $under^i(\phi)$ and $under^{i+1}(\phi)$
- It checks if $under^{i+1}(\phi)$ will be "easier to solve" than $under^i(\phi)$

IJCAI-17 MELBOURNE

Recursive Explore and Check Abstraction Refinement

RECAR



Recursive Explore and Check Abstraction Refinement

RECAR

- 2 levels of abstractions
  - One at the Oracle level $(\text{check}(\psi))$
  - One at the Domain level (recursive call)
- Efficient even when 50/50 chance of being SAT/UNSAT
- Everytime check improves, RECAR improves
- The return of the recursive call can reduce the number of refinement
- Totally generic, can change SAT solver $\rightarrow$ FO solver?

IJCAI-17 MELBOURNE

UNIVERSITÉ D'ARTOIS

RECAR: Instanciation for Modal Logic K

RECAR for Modal Logic K

- Modal Logic K is **PSPACE**-complete [Lad77, Hal95]
- What is Modal Logic K?
- How we over-approximate a formula ϕ?
- How we under-approximate a formula ϕ?
- Is it competitive against a CEGAR approach?
- Is it competitive against the state-of-the-art approaches?

IJCAI-17 MELBOURNE

UNIVERSITÉ D'ARTOIS

Preliminaries: Modal Logic

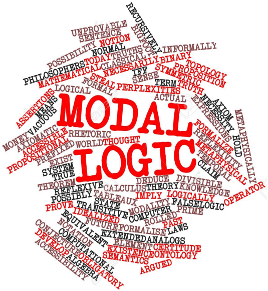
Modal Logic = Propositional Logic + $\Box$ and $\Diamond$

Modal Logic

- $\Box\phi$ means ϕ is necessarily true
- $\Diamond\phi$ means ϕ is possibly true

$$\Diamond\phi \leftrightarrow \neg\Box\neg\phi$$

$$\Box\phi \leftrightarrow \neg\Diamond\neg\phi$$



Preliminaries: Kripke Structure

- $\mathbb{P}$ finite non-empty set of propositional variables

Kripke Structure [Kri59]

$M = \langle W, R, V \rangle$ with:

- W, a non-empty set of possible worlds
- R, a binary relation on W
- V, a function that associate to each $p \in \mathbb{P}$, the set of possible worlds where p is true

Pointed Kripke Structure: $\langle \mathcal{K}, w \rangle$

- $\mathcal{K}$: Kripke Structure
- w: a possible world in W

Preliminaries: Satisfaction Relation

Definition (Satisfaction Relation)

The relation $\models$ between Kripke Structures and formulae is recursively defined as follows:

$$
\begin{array}{lll}
\langle \mathcal{K}, w \rangle \models p & \text{iff} & w \in V(p) \\
\langle \mathcal{K}, w \rangle \models \neg\phi & \text{iff} & \langle \mathcal{K}, w \rangle \not\models \phi \\
\langle \mathcal{K}, w \rangle \models \phi_1 \wedge \phi_2 & \text{iff} & \langle \mathcal{K}, w \rangle \models \phi_1 \text{ and } \langle \mathcal{K}, w \rangle \models \phi_2 \\
\langle \mathcal{K}, w \rangle \models \phi_1 \vee \phi_2 & \text{iff} & \langle \mathcal{K}, w \rangle \models \phi_1 \text{ or } \langle \mathcal{K}, w \rangle \models \phi_2 \\
\langle \mathcal{K}, w \rangle \models \Box\phi & \text{iff} & (w, w') \in R \text{ implies } \langle \mathcal{K}, w' \rangle \models \phi \\
\langle \mathcal{K}, w \rangle \models \Diamond\phi & \text{iff} & (w, w') \in R \text{ and } \langle \mathcal{K}, w' \rangle \models \phi
\end{array}
$$

$\mathcal{K}$ that satisfied a formula ϕ will be called “Kripke model of ϕ”

IJCAI-17 MELBOURNE

UNIVERSITÉ D'ARTOIS

MoSaiC

MoSaiC

- Open-Source Modal Logic K solver
- Uses Glucose as internal SAT solver
- Uses a RECAR approach

RECAR Assumptions: Reminder

1. Function 'check' is sound, complete and terminates
2. $isSAT(\hat{\phi})$ implies $isSAT(\text{refine}(\hat{\phi}))$
3. $\exists.n \in \mathbb{N}$ s.t. $\text{refine}^n(\hat{\phi}) \equiv^?_{\text{sat}} \phi$
4. $\text{isUNSAT}(\check{\phi})$ implies $\text{isUNSAT}(\phi)$
5. $\exists n \in \mathbb{N}$ s.t. $RC(under^n(\phi), under^{n+1}(\phi))$ is false

IJCAI-17 MELBOURNE   CNRS   UNIVERSITÉ D'ARTOIS   CRIL

MoSaiC: Over-Approximation

ϕ always in NNF and over(ϕ, i) in CNF thanks to Tseitin

$$\begin{aligned}
\text{over}(\phi, n) &= \text{over}'(\phi, 0, n) \\
\text{over}'(p_k, i, n) &= p_{k,i} \\
\text{over}'(\neg p_k, i, n) &= \neg p_{k,i} \\
\text{over}'(\Box\phi, i, n) &= \bigwedge_{j=0}^{n} (r_{i,j} \rightarrow \text{over}'(\phi, j, n)) \\
\text{over}'(\Diamond\phi, i, n) &= \bigvee_{j=0}^{n} (r_{i,j} \wedge \text{over}'(\phi, j, n))
\end{aligned}$$

- $p_{k,i}$ means p_k is true in the world w_i
- $r_{i,j}$ means that there is a relation between worlds w_i and w_j

RECAR Assumptions: Reminder

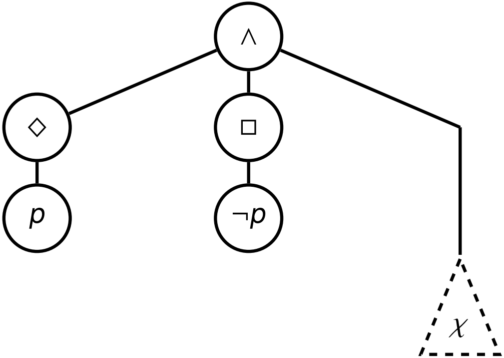
1. Function 'check' is sound, complete and terminates
2. $isSAT(\hat{\phi})$ implies $isSAT(\text{refine}(\hat{\phi}))$
3. $\exists . n \in \mathbb{N}$ s.t. $\text{refine}^n(\hat{\phi}) \equiv^?_{\text{sat}} \phi$
4. $\text{isUNSAT}(\check{\phi})$ implies $\text{isUNSAT}(\phi)$
5. $\exists n \in \mathbb{N}$ s.t. $RC(under^n(\phi), under^{n+1}(\phi))$ is false

IJCAI-17 MELBOURNE

UNIVERSITÉ D'ARTOIS

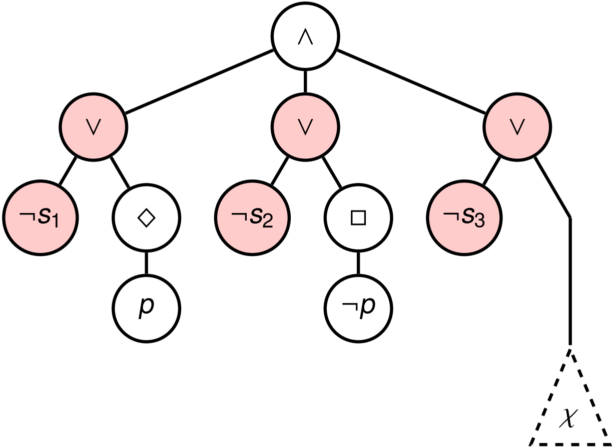
MoSaiC: Under-Approximation

Let's take an example, with χ huge but satisfiable...



Worst case for CEGAR using our 'over' function

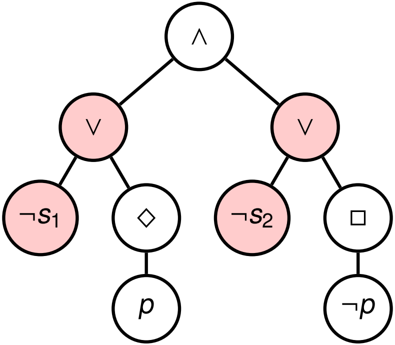
MoSaiC: Under-Approximation



Modern SAT solvers returns 'the reason' why a formula with n worlds is unsatisfiable ($core = \{s_1, s_2\}$)

MoSaiC: Under-Approximation

We want to cut what is not part of the 'unsatisfiability' ($s_i \notin core$)



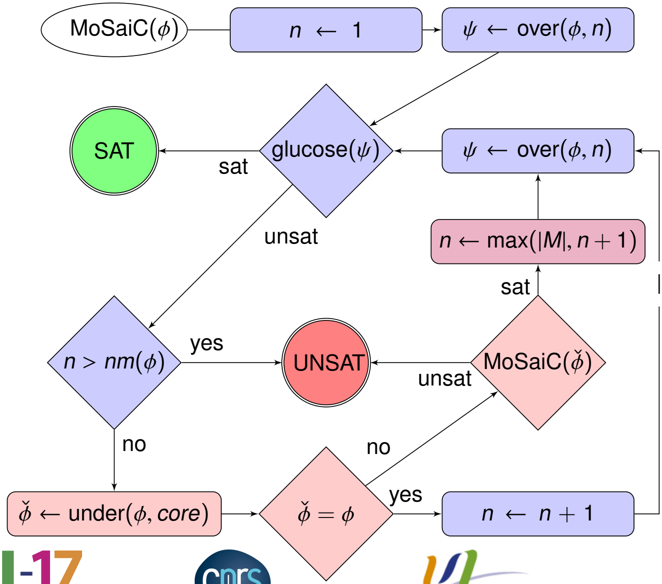
We just create $\check{\phi}$ smaller than ϕ and easier to solve.
The function *RC* from RECAR just says here: did we cut something ?

RECAR Assumptions: Reminder

1. Function 'check' is sound, complete and terminates
2. $isSAT(\hat{\phi})$ implies $isSAT(\text{refine}(\hat{\phi}))$
3. $\exists.n \in \mathbb{N}$ s.t. $\text{refine}^n(\hat{\phi}) \equiv^?_{\text{sat}} \phi$
4. $\text{isUNSAT}(\check{\phi})$ implies $\text{isUNSAT}(\phi)$
5. $\exists n \in \mathbb{N}$ s.t. $RC(under^n(\phi), under^{n+1}(\phi))$ is false

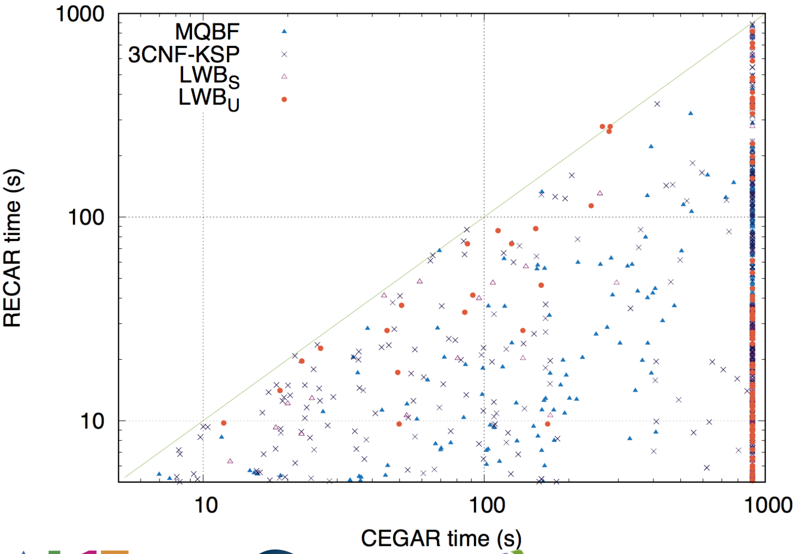
IJCAI-17 MELBOURNE

MoSaiC: RECAR for Modal Logic K

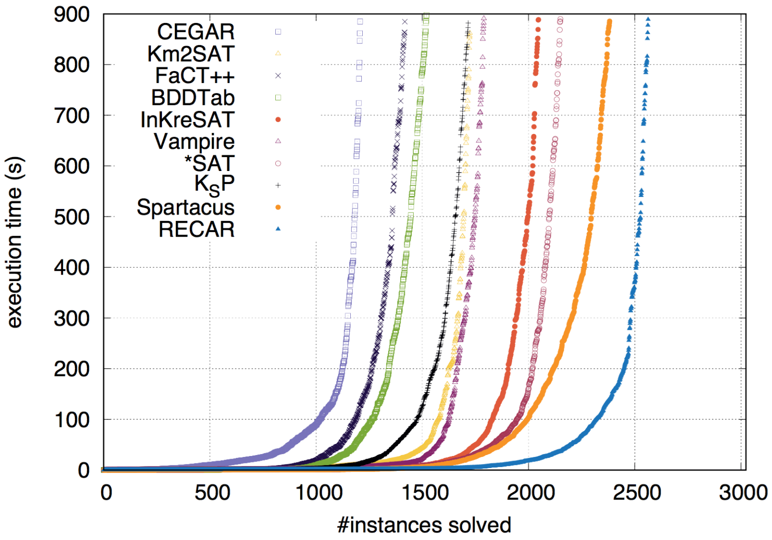


IJCAI-17 MELBOURNE

MoSaiC: RECAR for Modal Logic K



MoSaiC: RECAR for Modal Logic K



IJCAI-17 MELBOURNE

cnrs

UNIVERSITÉ D'ARTOIS

cril

Conclusion

Perspective: Other modal logics

NP
K5
K45
KB45
KD5
KD45
KT5

PSPACE
K
KT
KT4
KB
KD4
KD
K4
KDB
KBT

IJCAI-17 MELBOURNE

UNIVERSITÉ D'ARTOIS

Un raccourci récursif pour CEGAR

Application au problème de satisfiabilité en logique modale K

Jean-Marie Lagniez, Daniel Le Berre,
Tiago de Lima, Valentin Montmirail

CRIL, Université d'Artois, Lens, France

Caen - 3 Juillet 2017

IJCAI-17 MELBOURNE

UNIVERSITÉ D'ARTOIS

Bibliographie I

Gilles Audemard and Laurent Simon.
Predicting learnt clauses quality in modern SAT solvers.
In *Proc. of IJCAI'09*, pages 399–404, 2009.

Edmund M. Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith.
Counterexample-guided abstraction refinement for symbolic model checking.
Journal of the ACM, 50(5):752–794, 2003.

Niklas Eén and Niklas Sörensson.
An extensible sat-solver.
In *Proc. of SAT'03*, pages 502–518, 2003.

Bibliographie II

Joseph Y. Halpern.
The Effect of Bounding the Number of Primitive Propositions and the Depth of Nesting on the Complexity of Modal Logic.
Artificial Intelligence, 75(2):361–372, 1995.

Saul Kripke.
A completeness theorem in modal logic.
J. Symb. Log., 24(1):1–14, 1959.

Richard E. Ladner.
The Computational Complexity of Provability in Systems of Modal Propositional Logic.
SIAM J. Comput., 6(3):467–480, 1977.

IJCAI-17 MELBOURNE

UNIVERSITÉ D'ARTOIS

Bibliographie III

Jean-Marie Lagniez, Daniel Le Berre, Tiago de Lima, and Valentin Montmirail.
A Recursive Shortcut for CEGAR: Application To The Modal Logic K Satisfiability Problem.
In *Proc. of IJCAI'17*, 2017.

Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik.
Chaff: Engineering an efficient SAT solver.
In *Proc. of DAC'01*, pages 530–535, 2001.

João P. Marques Silva and Karem A. Sakallah.
GRASP: A search algorithm for propositional satisfiability.
IEEE Transactions on Computers, 48(5):506–521, 1999.