max planck institut
informatik

Extending an Isabelle Formalisation of CDCL to Optimising CDCL

Mathias Fleury

joint work with Christoph Weidenbach

Matryoshka+Veridis 2019

SAARLAND UNIVERSITY
SAARBRÜCKEN GRADUATE SCHOOL OF COMPUTER SCIENCE

SIC Saarland Informatics Campus

Let’s find a model with minimal weight

10

¬ 4

20

¬ 13

Optimal partial model:

Optimal total model: ¬

How reliable is the theory?

Conference version

Branch and Bound for Boolean Optimization and
the Generation of Optimality Certificates
Javier Larrosa, Robert Nieuwenhuis, Albert Oliveras, and Enric Rodríguez-Carbonell (SAT 2009)

A literal l is *true* in I if $l \in I$, *false* in I if $\neg l \in I$, and *undefined* in I otherwise.

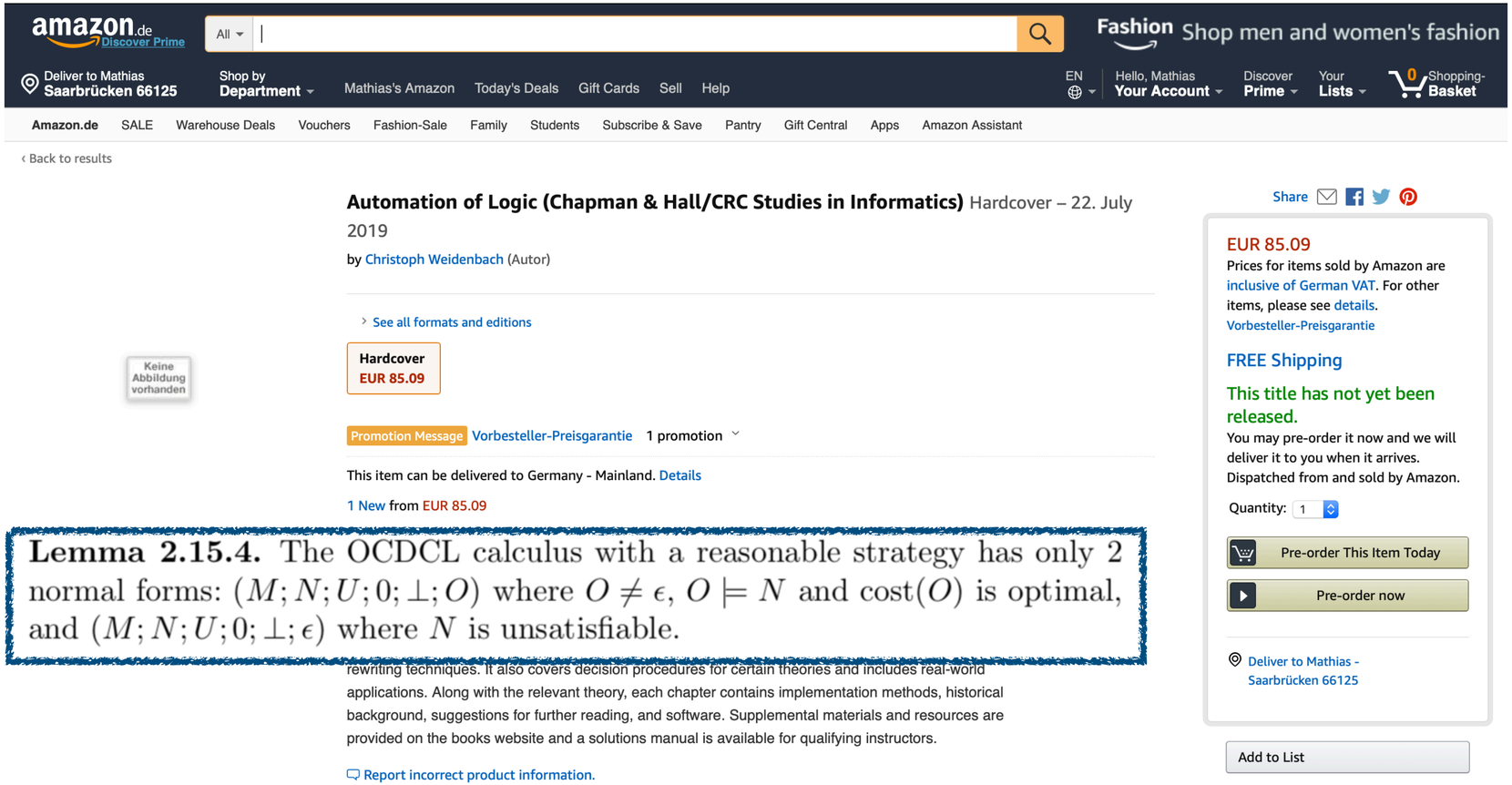
A clause set S is true in I if all its clauses are true in I. Then I is called a *model* of S, and we write $I \models S$ (and similarly if a literal or clause is true in I).

Journal version

A Framework for Certified Boolean Branch-and-Bound Optimization
Javier Larrosa, Robert Nieuwenhuis, Albert Oliveras, and Enric Rodríguez-Carbonell (JAR 2011)

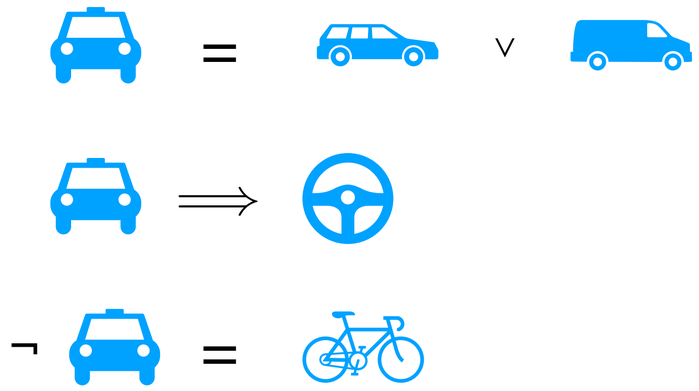
literals of a clause C are false in I. A clause set S is true in I if all its clauses are true in I; if I is also total, then I is called a *total model* of S, and we write $I \models S$.

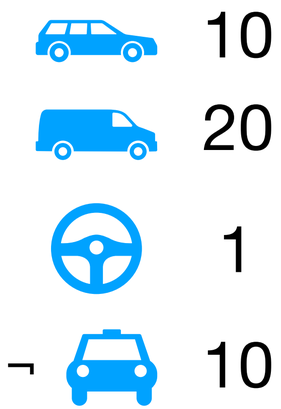
How reliable is the theory?



max planck institut informatik

Let’s optimise our problem

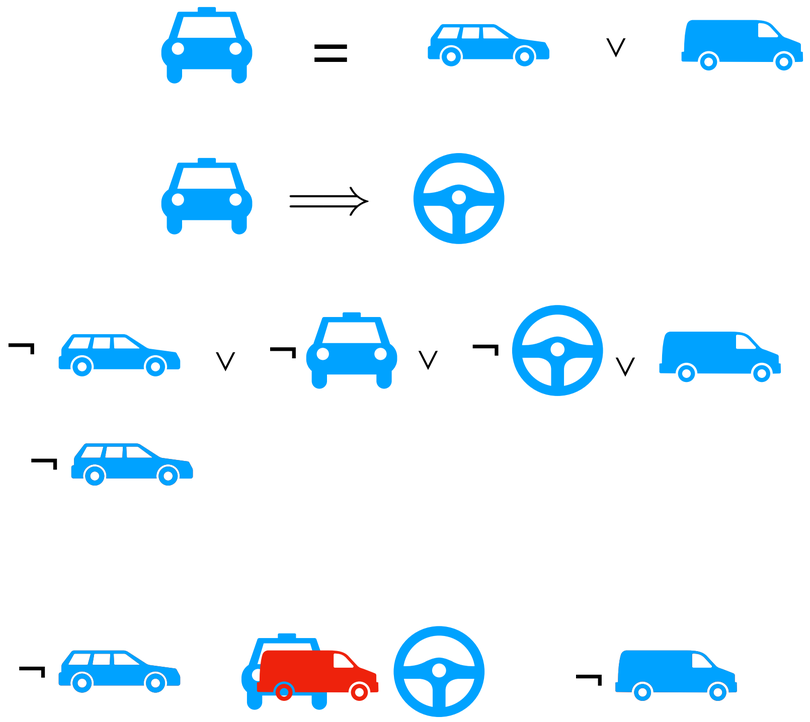


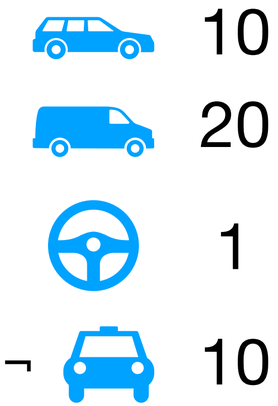


Let’s optimise our problem

OCDCL = CDCL + identify better models
+ conflicts based on weights

Let’s optimise our problem





Optimal model 11

How lazy do you like your formalisation?

Christoph's view:

OCDCL_W = CDCL + improve + conflict rules

copy-paste of proofs

My first idea:

OCDCL = CDCL + improve +
{-M. cost M ≥ min_cost}

reuse CDCL proofs

How lazy do you like your formalisation?

My first idea:

OCDCL = CDCL + improve +
{-M. cost M ≥ min_cost}

reuse CDCL proofs

My second idea:

$CDCL_{bnb}$ = CDCL + improve +
$\mathcal{T}$(min_cost)

reuse CDCL proofs

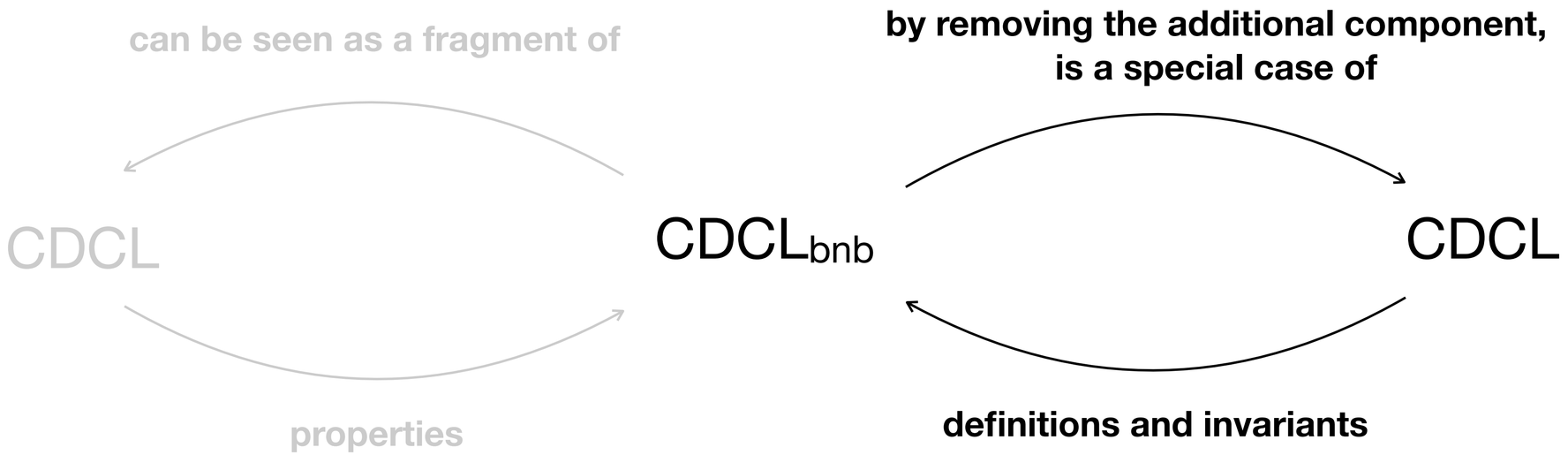
How lazy do you like your formalisation?

CDCL_{bnb} = CDCL + improve +
$\mathcal{T}$(min_cost)

OCDCL = CDCL_{bnb} where
$\mathcal{T}$(min_cost) = {-M. cost M ≥ min_cost}

OCDCL_{W} = OCDCL + restrictions

Reuse!



Reuse!

Propagate rule

in Isabelle

```
C ∨ L ∈ N ⟹ M ⊨as ¬C ⟹ undefined_lit M L ⟹
   (M, N) ⇒CDCL (L # M, N)
```

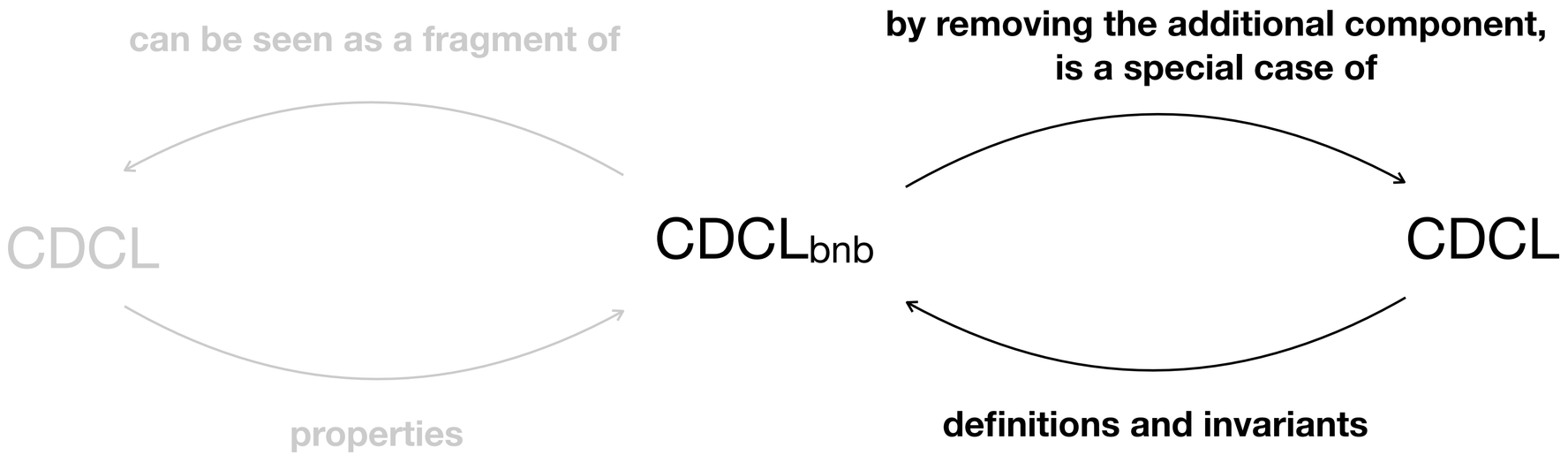
obtained for free, thanks to abstraction over the state!
also invariants and theorems can be reused

Propagate rule

in Isabelle

```
C ∨ L ∈ N ⟹ M ⊨as ¬C ⟹ undefined_lit M L ⟹
   (M, N, O) ⇒CDCLbnb (L # M, N, O)
```

Reuse!



Translate to reuse

Propagate rule

in Isabelle

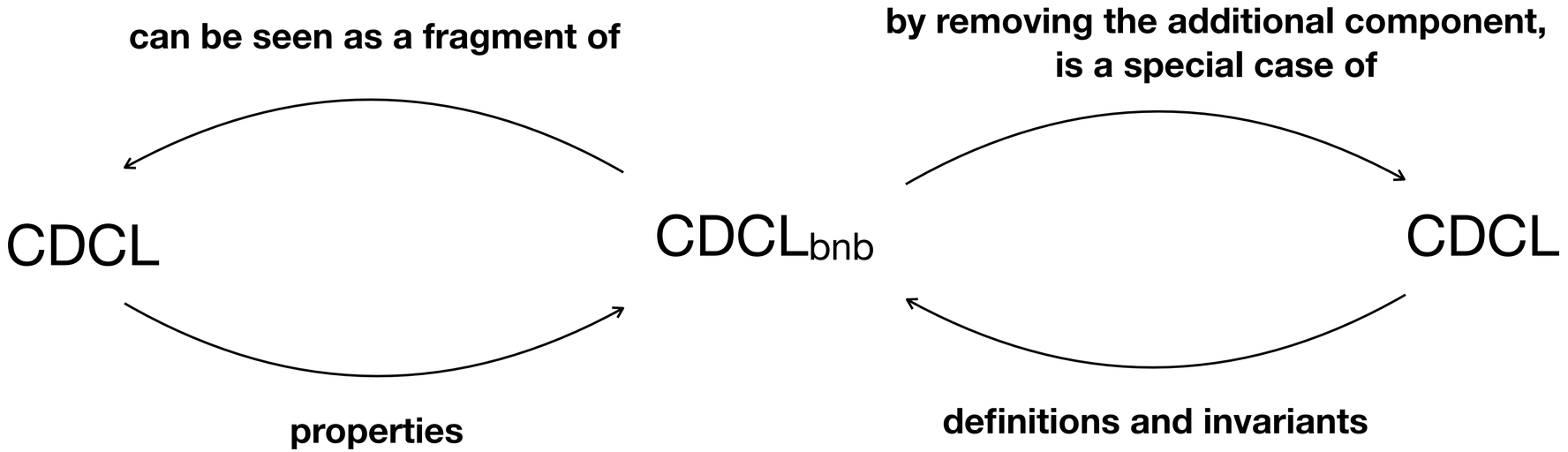
```
C ∨ L ∈ N ⟹ M ⊨as ¬C ⟹ undefined_lit M L ⟹
   (M, N, 0) ⇒CDCLbnb (L # M, N, 0)
```

Propagate rule

in Isabelle

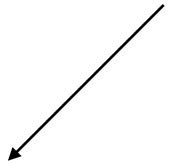
```
C ∨ L ∈ N+𝒯(min_cost) ⟹ M ⊨as ¬C ⟹
undefined_lit M L ⟹
  (M, N + 𝒯(min_cost), 0) ⇒CDCL
     (L # M, N + 𝒯(min_cost), 0)
```

Reuse!



Reuse in practise!

ignore the additional component



$CDCL_{bnb} = CDCL + improve + \mathcal{T}(min\_cost)$

Inherited:

Definitions (for free)

Reuse in practise!

Inherited:

Termination (for free) Definitions (for free)

Reuse in practise!

$CDCL_{bnb}$ does not know anything about what is optimised!

Inherited:

Termination (for free) Definitions (for free)

Ends with an unsat set (nearly for free)

Why does it work?

OCDCL = CDCL_{bnb} where
$\mathcal{T}$(min_cost) = {-M. cost M ≥ min_cost}

Lemma

If I is a total model of N with cost < min_cost,
then I is a model of N+$\mathcal{T}$(min_cost)

Why does it work?

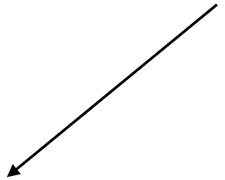
Lemma

If I is a total model of N with cost < min_cost, then I is a model of N+$\mathcal{T}$(min_cost)

Fails for partial models!

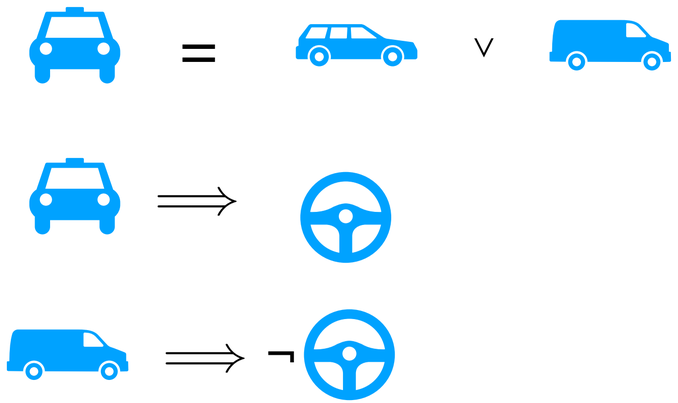
How lazy do you like your formalisation?

make sure that the rules on paper and in Isabelle are the same



$\mathrm{OCDCL}_{\mathrm{W}}$ = OCDCL + restrictions

Another application: Dead features



Can every option be true?

How lazy do you like your formalisation?

Christoph's view:

CDCLcm$_W$ = CDCL + improve + conflict rules

copy-paste of proofs

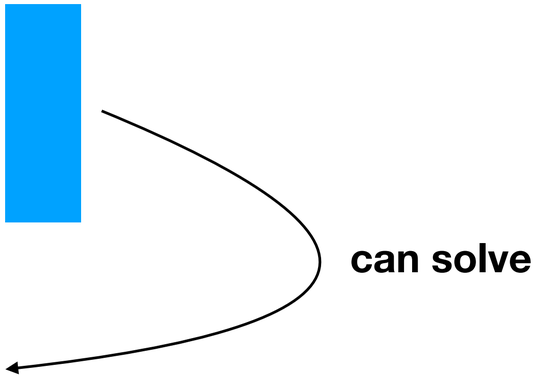
My idea:

CDCLcm = CDCL$_{bnb}$ where

$\mathcal{T}$(models_founds) = {-M. there is a model with more trues in models_founds}

How lazy do you like your formalisation?

	Lines of codes (kloc)
CDCL_{bnb}	2,0
OCDCL	1,8
OCDCL_{W}	1,0
Partial Encoding	1,2
MaxSAT	0,4



Conclusion

Concrete outcome

- CDCL with branch and bound

OCDCL = CDCL_{bnb} where
$\mathcal{T}$(min_cost) = {-M. cost M ≥ min_cost}

OCDCL = CDCL_{bnb} where
$\mathcal{T}$(min_cost) = {-D. {M. cost M ≥ min_cost} ⊨ D}

Future work

- CDCL($\mathcal{T}$)

Conclusion: How about CDCL($\mathcal{T}$)?

But isn't CDCL($\mathcal{T}$) exactly:

CDCL$_{\text{bnb}}$ where
$\mathcal{T}$ = {clauses entailed theory}

Not exactly, because the wrong conflict clause (negation of the trail) is used

Translate to reuse