A framework for step-wise explaining how to solve constraint satisfaction problems

Bart Bogaerts
(Joint work with Emilio Gamba, Jens Claes, and Tias Guns)

April 20, 2021 @ Beyond Satisfiability

VUB
ARTIFICIAL INTELLIGENCE RESEARCH GROUP

BEYOND SATISFIABILITY

- “There Are No CNF Problems” (P.J. Stuckey)
- Adopt a (simple) high-level modeling language

BEYOND SATISFIABILITY

- "There Are No CNF Problems" (P.J. Stuckey)
- Adopt a (simple) high-level modeling language
- Structural information of the problem visible
- E.g., symmetry breaking

$$\forall p[Pigeon] \exists h[Hole] : In(p, h)$$
$$\forall h[Hole], p_1 p_2[Pigeon] : In(p_1, h) \land In(p_2, h) \Rightarrow p_1 = p_2.$$

BEYOND SATISFIABILITY

- "There Are No CNF Problems" (P.J. Stuckey)
- Adopt a (simple) high-level modeling language
- Structural information of the problem visible
- E.g., symmetry breaking

$$\forall p[\mathit{Pigeon}]\exists h[\mathit{Hole}] : \mathit{In}(p, h)$$
$$\forall h[\mathit{Hole}], p_1 p_2[\mathit{Pigeon}] : \mathit{In}(p_1, h) \wedge \mathit{In}(p_2, h) \Rightarrow p_1 = p_2.$$

- Choice ... fragmentation ... ASP, CP, SMT, . . .

BEYOND SATISFIABILITY: THIS TALK

- Algorithms: SAT level
- Explanation: first-order level

OUTLINE

LOGIC GRID PUZZLES

- Set of clues
- Sets of entities that need to be linked
- Each entitity is linked to exactly one entity of each other type (bijectivity)
- The links are consistent (transitivity)

LOGIC GRID PUZZLES: EXAMPLE

- The person who ordered capellini paid less than the person who chose arrabiata sauce
- The person who ordered tagliolini paid more than Angie
- The person who ordered tagliolini paid less than the person who chose marinara sauce
- Claudia did not choose puttanesca sauce
- The person who ordered rotini is either the person who paid $8 more than Damon or the person who paid $8 less than Damon
- The person who ordered capellini is either Damon or Claudia
- The person who chose arrabiata sauce is either Angie or Elisa
- The person who chose arrabiata sauce ordered farfalle

2019 HOLY GRAIL CHALLENGE: LOGIC GRID PUZZLES

- Parse puzzles and translate into CSP
- Solve CSP automatically
- Explain in a human-understandable way how to solve this puzzle

2019 HOLY GRAIL CHALLENGE: LOGIC GRID PUZZLES

- Parse puzzles and translate into CSP
- Solve CSP automatically
- Explain in a human-understandable way how to solve this puzzle

We won the challenge...

2019 HOLY GRAIL CHALLENGE: LOGIC GRID PUZZLES

- Parse puzzles and translate into CSP
- Solve CSP automatically
- Explain in a human-understandable way how to solve this puzzle

We won the challenge... out of two participants

DEMO

- Automatically generated constraint representation from natural language (no optimization of the constraints for the explanation problem)
- No modifications to the underlying solvers (we do not equip each propagator with explanation mechanisms)
- demo: `https://bartbog.github.io/zebra/pasta/`

OUTLINE

1. Introduction: Beyond Satisfiability
2. History: The Holy Grail Challenge
3. Step-Wise Explanations (ECAI 2020)
4. The OCUS Problem (Hitting Set–Based Algorithms) (Under Review)
5. Nested Explanations (AIJ 2021)
6. Conclusion

ECAI 2020 PAPER

- Formalize the step-wise explanation problem
- Propose an algorithm (agnostic of actual propagators, consistency level, etc.)
- Propose heuristics for guiding the search for explanations
- Experimentally demonstrate feasibility

PRELIMINARIES/NOTATION

- Propositional vocabulary Σ
- (partial) interpretation I: consistent set of literals over Σ
  Slightly abusing notation: set of (unit) clauses
- Propositional theory T (set of constraints over Σ)
  Slightly abusing notation: set of constraints = conjunction
- Notation $T \wedge I \models I'$

GOAL

- Given T and I, let I_{end} denote the maximal set of literals such that

$$T \wedge I \models I_{end}$$

- Explain in simple steps how to derive I_{end}
- Our focus: single steps (not optimizing entire sequence yet)

FORMALIZING EXPLANATIONS

Definition

Let I_{i-1} and I_i be partial interpretations such that $I_{i-1} \wedge T \models I_i$. We say that (E_i, S_i, N_i) explains the derivation of I_i from I_{i-1} if the following hold:

- $N_i = I_i \setminus I_{i-1}$ (i.e., N_i consists of all newly defined facts),
- $E_i \subseteq I_i$ (i.e., the explaining facts are a subset of what was previously derived),
- $S_i \subseteq T$ (i.e., a subset of the clues and implicit constraints are used), and
- $S_i \wedge E_i \models N_i$ (i.e., all newly derived information indeed follows from this explanation).

FORMALIZING EXPLANATIONS

Definition

We call (E_i, S_i, N_i) a non-redundant explanation of the derivation of I_i from I_{i-1} if it explains this derivation and whenever $E' \subseteq E_i; S' \subseteq S_i$ while (E', S', N_i) also explains this derivation, it must be that $E_i = E', S_i = S'$.

FORMALIZING EXPLANATIONS

Definition

We call (E_i, S_i, N_i) a non-redundant explanation of the derivation of I_i from I_{i-1} if it explains this derivation and whenever $E' \subseteq E_i; S' \subseteq S_i$ while (E', S', N_i) also explains this derivation, it must be that $E_i = E', S_i = S'$.

Observation: computing non-redundant explanations of a single literal can be done using Minimal Unsat Core (MUS) extraction:

Theorem

There is a one-to-one correspondence between $\subseteq$-minimal unsatisfiable cores of $I_i \wedge T \wedge \neg\ell$ and non-redundant explanations of $I_i \cup \{\ell\}$ from I_i (given T).

FORMALIZING EXPLANATIONS

Definition

We call (E_i, S_i, N_i) a non-redundant explanation of the derivation of I_i from I_{i-1} if it explains this derivation and whenever $E' \subseteq E_i; S' \subseteq S_i$ while (E', S', N_i) also explains this derivation, it must be that $E_i = E', S_i = S'$.

Furthermore, we assume existence of a cost function $f(E_i, S_i, N_i)$ that quantifies the interpretability of a single explanation

FORMALIZING EXPLANATIONS

Definition

Given a theory T and initial partial interpretation I_0, the explanation-production problem consist of finding a non-redundent explanation sequence

$$(I_0, (\emptyset, \emptyset, \emptyset)), (I_1, (E_1, S_1, N_i)), \ldots, (I_n, (E_n, S_n, N_n))$$

such that some aggregate over the sequence $(f(E_i, S_i, N_i))_{i \leq n}$ is minimised.

MUS-BASED EXPLANATION GENERATION

Algorithm 1: $\textsc{OneStep}(T, f, I, I_{end})$

$X_{best} \leftarrow nil$;
for $\ell \in \{I_{end} \setminus I\}$ **do**
  $X \leftarrow \text{MUS}(T \wedge I \wedge \neg\ell)$;
  if $f(X) < f(X_{best})$ **then**
    $X_{best} \leftarrow X$;
  end
end
return X_{best}

MUS-BASED GENERATION NOT SUFFICIENT

- MUS guarantees non-redundancy ...
- ... does not guarantee quality

MUS-BASED GENERATION NOT SUFFICIENT

- MUS guarantees non-redundancy ...
- ... does not guarantee quality
- ECAI paper: MUS-based workaround (heuristic): do not use full T, but approximate

MUS-BASED GENERATION NOT SUFFICIENT

- MUS guarantees non-redundancy ...
- ... does not guarantee quality
- ECAI paper: MUS-based workaround (heuristic): do not use full T, but approximate
- No details in this talk.

IMPLEMENTATION (ECAI PAPER)

- Visual explanation interface
- Logic Grid puzzle cost function:
  - Single implicit axiom: very cheap
  - Single constraint + implicit: less cheap
  - Multiple constraints: very expensive

IMPLEMENTATION (ECAI PAPER)

- Visual explanation interface
- Logic Grid puzzle cost function:
  - Single implicit axiom: very cheap
  - Single constraint + implicit: less cheap
  - Multiple constraints: very expensive

  "The person who ordered capellini is either Damon or Claudia".

$$\exists p : \mathit{ordered}(p, \mathit{capellini}) \wedge (p = \mathit{Damon} \vee p = \mathit{Claudia}).$$

IMPLEMENTATION (ECAI PAPER)

- Visual explanation interface
- Logic Grid puzzle cost function:
  - Single implicit axiom: very cheap
  - Single constraint + implicit: less cheap
  - Multiple constraints: very expensive

  "The person who ordered capellini is either Damon or Claudia".

$$\exists p : ordered(p, capellini) \land (p = Damon \lor p = Claudia).$$

- Under the hood: IDP system [1]

OUTLINE

BEYOND MUS-BASED EXPLANATIONS

- MUS: $\subseteq$-minimal
- SMUS: $\#$-minimal (still not sufficient...)

BEYOND MUS-BASED EXPLANATIONS

- MUS: $\subseteq$-minimal
- SMUS: $\#$-minimal (still not sufficient...)
- New problem OUS

THE OUS PROBLEM

Definition

Let T be a formula, $f : 2^T \to \mathbb{N}$ a cost function. We call $\mathcal{S} \subseteq T$ an OUS of T (with respect to f) if

- $\mathcal{S}$ is unsatisfiable,
- all other unsatisfiable $\mathcal{S}' \subseteq T$ satisfy $f(\mathcal{S}') \geq f(\mathcal{S})$.

THE OUS PROBLEM

Definition

Let T be a formula, $f : 2^T \to \mathbb{N}$ a cost function. We call $\mathcal{S} \subseteq T$ an OUS of T (with respect to f) if

- $\mathcal{S}$ is unsatisfiable,
- all other unsatisfiable $\mathcal{S}' \subseteq T$ satisfy $f(\mathcal{S}') \geq f(\mathcal{S})$.

Q: How to compute OUSs?

OUS-BASED EXPLANATION GENERATION

Algorithm 2: ONESTEP(T, f, I, I_{end})

```
X_best ← nil;
for ℓ ∈ {I_end \ I} do
    X ← OUS(T ∧ I ∧ ¬ℓ);
    if f(X) < f(X_best) then
        X_best ← X;
    end
end
return X_best
```

BEYOND OUS-BASED EXPLANATIONS

- The different iterations (for loop line 2)... very similar
- Can we exploit this?

BEYOND OUS-BASED EXPLANATIONS

- The different iterations (for loop line 2)... very similar
- Can we exploit this?
- Essentially, the task at hand is: find a single unsatisfiable subset of

$$T \land I \land \bigvee_{\ell \in I_{end} \setminus I} \neg \ell$$

that:

  - Is optimal w.r.t. f
  - Contains exactly one literal $\neg \ell$ with $\ell \in I_{end} \setminus I$ (example!)

THE OCUS PROBLEM

Definition

Let T be a formula, $f : 2^T \rightarrow \mathbb{N}$ a cost function and p a predicate $p : 2^T \rightarrow \{\mathbf{t}, \mathbf{f}\}$. We call $\mathcal{S} \subseteq T$ an OCUS of T (with respect to f and p) if

- $\mathcal{S}$ is unsatisfiable,
- $p(\mathcal{S})$ is true
- all other unsatisfiable $\mathcal{S}' \subseteq T$ with $p(\mathcal{S}') = \mathbf{t}$ satisfy $f(\mathcal{S}') \geq f(\mathcal{S})$.

OCUS-BASED EXPLANATION GENERATION

Algorithm 3: EXPLAIN-ONE-STEP-OCUS(T, f, I, I_{end})

$p \leftarrow$ exactly one of $\overline{I_{end} \setminus I}$

return OCUS$(T \wedge I \wedge \overline{I_{end} \setminus I}, f, p)$

HOW TO FIND OCUSS?

- Hitting set–based algorithms: used for MaxSAT and SMUS

Theorem

A set $\mathcal{S} \subseteq \boldsymbol{T}$ is a MCS of $\boldsymbol{T}$ iff it is a minimal hitting set of MUS**s**$(\boldsymbol{T})$. *A set $\mathcal{S} \subseteq \boldsymbol{T}$ is a MUS of $\boldsymbol{T}$ iff it is a minimal hitting set of* MCS**s**$(\boldsymbol{T})$.

HOW TO FIND OCUSS?

- Hitting set–based algorithms: used for MaxSAT and SMUS

Theorem

A set $\mathcal{S} \subseteq T$ is a MCS of T iff it is a minimal hitting set of $\textsc{mus}\mathbf{s}(T)$. *A set $\mathcal{S} \subseteq T$ is a MUS of T iff it is a minimal hitting set of* $\textsc{mcs}\mathbf{s}(T)$.

- We extended this to OCUS:

HITTING SET–BASED OCUS

Algorithm 4: OCUS(T, f, p)

```
H ← ∅
while true do
    S ← COST-OPTIMAL-HITTINGSET(H, f, p)
    if ¬SAT(S) then
        return S
    end
    S ← GROW(S, T)
    H ← H ∪ {T \ S}
end
```

CORRECTNESS

Theorem

Let $\mathcal{H}$ be a set of correction subsets of $\boldsymbol{T}$. If $\mathcal{S}$ is a hitting set of $\mathcal{H}$ that is $\boldsymbol{f}$-optimal among the hitting sets of $\mathcal{H}$ satisfying a predicate $\mathbf{p}$, and $\mathcal{S}$ is unsatisfiable, then $\mathcal{S}$ is an OCUS of $\boldsymbol{T}$.
If $\mathcal{H}$ has no hitting sets satisfying $\mathbf{p}$, then $\boldsymbol{T}$ has no OCUSs.

TWO FURTHER IDEAS

- Incrementality: re-use previous computations in future calls
- Grow: Develop implementations of “grow” tailored for explanations

INCREMENTALITY

Algorithm 5: OCUS(T, f, p)

$\mathcal{H} \leftarrow \ldots$
while true **do**
    $\mathcal{S} \leftarrow$ COST-OPTIMAL-HITTINGSET$(\mathcal{H}, f, p)$
    if $\neg$SAT$(\mathcal{S})$ **then**
        return $\mathcal{S}$
    end
    $\mathcal{S} \leftarrow$ GROW$(\mathcal{S}, T)$
    $\mathcal{H} \leftarrow \mathcal{H} \cup \{T \setminus \mathcal{S}\}$
end

DIFFERENT GROW STRATEGIES

When calling OCUS, the theory consists of

1. The original theory (constraints) T
2. The current interpretation I_{end}
3. The negation of literals in I_{end} $\overline{I_{end}}$

- What to take into account for GROW?
- What about the cost function?

EXPERIMENTS

Implementation building on pysat + cpMpy

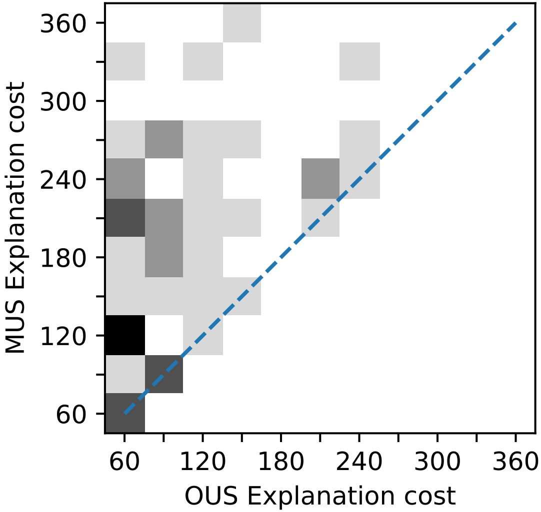
Q1 What is the effect of requiring optimality of the generated MUSs on the **quality** of the generated explanations?

Q2 Which **domain-specific GROW methods** perform best?

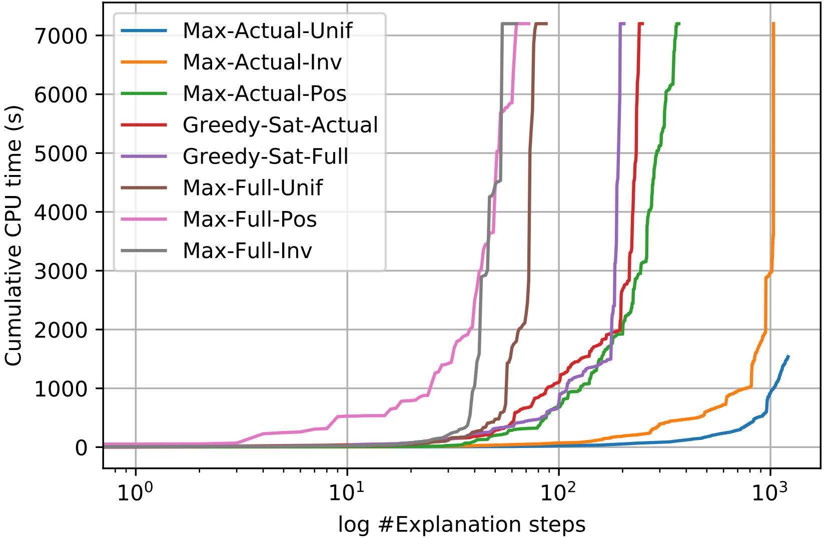
Q3 What is the effect of the use of **contrainedness** on the time required to compute an explanation sequence?

Q4 Does **re-use** of computed satisfiable subsets improve efficiency?

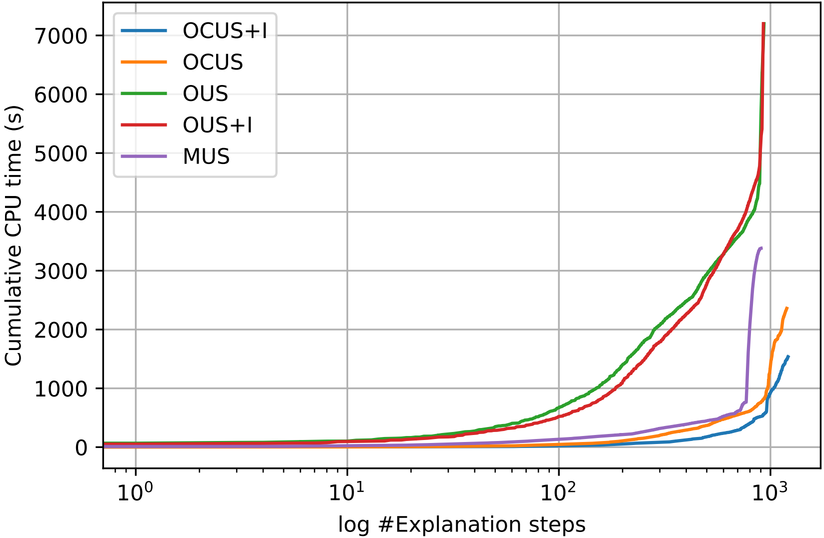
EXPERIMENTS: SOLUTION QUALITY



EXPERIMENTS: GROW STRATEGIES



EXPERIMENTS: PERFORMANCE



OUTLINE

1. Introduction: Beyond Satisfiability
2. History: The Holy Grail Challenge
3. Step-Wise Explanations (ECAI 2020)
4. The OCUS Problem (Hitting Set–Based Algorithms) (Under Review)
5. Nested Explanations (AIJ 2021)
6. Conclusion

OBSERVATION

- Some steps still quite difficult.
- Idea: explanations at different levels of abstraction
- Explain hardest steps of the sequence

EXAMPLE

← Prev | HELP | Next →

	capellini	farfalle	tagliolini	rotini	4	8	12	16	angie	damon	claudia	elisa
the_other_type1		-										
arrabiata_sauce	-	✓	-	-	-					-	-	
marinara_sauce		-	-		-							
puttanesca_sauce		-									-	
angie	-		-				-	-				
damon		-		-				-				
claudia		-										
elisa	-											
4		-	-									
8												
12												
16	-		-									

NESTED EXPLANATIONS

- Idea: explanations at different levels of abstraction
- Counterfactual reasoning/proof by contradiction
- See demo `https://bartbog.github.io/zebra/pasta/`

NESTED EXPLANATIONS

- Idea: explanations at different levels of abstraction
- Counterfactual reasoning/proof by contradiction
- See demo `https://bartbog.github.io/zebra/pasta/`
- For which steps? Hardest step of the nested sequence simpler than the step to explain

OUTLINE

1. Introduction: Beyond Satisfiability
2. History: The Holy Grail Challenge
3. Step-Wise Explanations (ECAI 2020)
4. The OCUS Problem (Hitting Set–Based Algorithms) (Under Review)
5. Nested Explanations (AIJ 2021)
6. Conclusion

CONCLUSION

- Overview of a (relatively young) research project
  $\Rightarrow$ Lots of open questions!
- Goal:Provide human-understandable explanations of inferences made by a constraint solver
- Our proposal: split in small comprehensible steps
- Explain them at different levels of detail (abstraction)
- Triggers novel algorithmic needs
- Demonstrated on logic grid puzzles

USE CASES

- Teach humans how to solve a certain problem
- Quantify problem difficulty
- “Help” button
- Interactive configuration/planning/scheduling

FUTURE WORK

- Learning the optimization function (from humans) – Learning the level of abstraction
- Explaining optimization (different types of “why” queries); close relation to Explainable AI Planning [2]
- Scaling up (approximate algorithms; decomposition of explanation search)
- Incremental algorithms over different “why” queries

REFERENCES

[1] Broes De Cat, Bart Bogaerts, Maurice Bruynooghe, Gerda Janssens, and Marc Denecker. Predicate logic as a modelling language: The IDP system. *CoRR*, abs/1401.6312v2, 2016.

[2] Maria Fox, Derek Long, and Daniele Magazzeni. Explainable planning. *arXiv preprint arXiv:1709.10256*, 2017.