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

Bart Bogaerts
(Joint work with Emilio Gamba, Jens Claes, and Tias Guns)
September 13, 2020 @ XLoKR20
Franz: "How tailored is this to CSP?"

▶ "CSP": any formalism that allows searching for models in the context of a *finite* domain (Model Expansion for FO, propositional logic, ASP, ...)

▶ Our step-wise ideas could also be applied to generate "simple first-order proofs" (to be explored)
Franz: “How tailored is this to CSP?”

- “CSP”: any formalism that allows searching for models in the context of a finite domain (Model Expansion for FO, propositional logic, ASP, …)
- Our step-wise ideas could also be applied to generate “simple first-order proofs” (to be explored)
CONCLUSION

- Overview of a (relatively young) research project
  ⇒ 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
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

More generic paper at ECAI 2020 [1]
Journal version and follow-up conference paper under review.
WHAT WE WORKED ON ALREADY

- 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
- (unpublished) Nested explanations (conceptual extension)
- (unpublished) Incremental OMUS algorithms (efficiency bottleneck)
LOGIC GRID PUZZLES

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

Definition

Let $I_{i-1}$ and $I_i$ be partial interpretations such that $I_{i-1} \land T \models I_i$. 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
- $S_i \cup E_i \models N_i$ (i.e., all newly derived information indeed follows from this explanation).
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 \(E_i = E'\), \(S_i = S'\).
PROBLEM

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 \(I_i \land T \land \neg l\) and non-redundant explanations of \(I_i \cup \{l\}\) from \(I_i\).
PROBLEM

Definition

We call \((E_i, S_i, N_i)\) a non-redundant explanation of the derivation of \(l_i\) from \(l_{i-1}\) 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.
**Problem**

**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 a predefined aggregate over the sequence $(f(E_i, S_i, N_i))_{i \leq n}$ is minimised.
ALGORITHM

▶ Greedy algorithm (max aggregate)
  ▶ At each step, for each solution literal, find a MUS *
  ▶ Pick the cheapest (cost-wise)
  ▶ (some caching)
▶ Under the hood: IDP system [3]
▶ * single MUS call does not suffice
▶ * Pruning based on optimistic approximation of cost
▶ * no guarantee of optimality
▶ * inefficient!
Algorithm 2: SingleStepExplain\((T, f, I)\)

1. \(\text{BestVal} \leftarrow \infty\);
2. \(\text{for } l \text{ such that } T \land I \models l \text{ and } l \notin I \text{ do}\)
3. \(\quad X \leftarrow \text{OMUS}(T \land I \land \neg l, f)\);
4. \(\quad \text{if } f(X) < \text{BestVal} \text{ then}\)
5. \(\quad \quad \text{BestVal} \leftarrow f(X)\);
6. \(\quad \quad T_{\text{best}} \leftarrow T \cap X\);
7. \(\quad \quad I_{\text{best}} \leftarrow I \cap X\);
8. \(\quad \quad l_{\text{best}} \leftarrow l\);
9. \(\quad \text{end}\)
10. \(\text{end}\)
11. \(\text{return } (T_{\text{best}}, I_{\text{best}}, l_{\text{best}})\)
Visual explanation interface

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”.

$$\exists p : \text{ordered}(p, \text{capellini}) \land (p = \text{Damon} \lor p = \text{Claudia}).$$
USE CASES

▶ Teach humans how to solve a certain problem
▶ Quantify problem difficulty
▶ “Help” button
▶ Interactive configuration/planning/scheduling
NEXT STEPS: NESTED EXPLANATION

- Idea: explanations at different levels of abstraction
- Explain hardest steps of the sequence
- Counterfactual reasoning/proof by contradiction
- See demo https://bartbog.github.io/zebra/pasta/
NEXT STEPS: OMUS COMPUTATION

- Algorithms to compute Optimal MUSs
- Based on hitting-set duality
- Combining existing SMUS (#-minimal) [6, 5] algorithms and MAXSAT [2] algorithms
- Incremental OMUS computation
- Constrained OMUS computation
- No experimental results yet
Algorithm 2: **SingleStepExplain**\((T, f, I)\)

1. \(\text{BestVal} \leftarrow \infty;\)
2. \textbf{for} \(l\) such that \(T \land I \models l\) and \(l \notin I\) \textbf{do}
   
   3. \(X \leftarrow \text{OMUS}(T \land I \land \neg l, f);\)
   
   4. \textbf{if} \(f(X) < \text{BestVal}\) \textbf{then}
      
      5. \(\text{BestVal} \leftarrow f(X);\)
      
      6. \(T_{\text{best}} \leftarrow T \cap X;\)
      
      7. \(I_{\text{best}} \leftarrow I \cap X;\)
      
      8. \(l_{\text{best}} \leftarrow l;\)
   
   9. \textbf{end}

10. \textbf{end}

11. \textbf{return} \((T_{\text{best}}, I_{\text{best}}, l_{\text{best}})\)
MORE 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 [4]
- Scaling up (approximate algorithms; decomposition of explanation search)
- Incremental algorithms over different “why” queries
REFERENCES


