



Efficiently Explaining CSPs with Unsatisfiable Subset Optimization (Extended Abstract)

Emilio Gamba, Bart Bogaerts and Tias Guns

EasyChair preprints are intended for rapid dissemination of research results and are integrated with the rest of EasyChair.

September 2, 2022




Efficiently Explaining CSPs with Unsatisfiable Subset Optimization (extended abstract)

Emilio Gamba¹  

Data Analytics Lab, Vrije Universiteit Brussel, Belgium

Bart Bogaerts   

Artificial Intelligence Lab, Vrije Universiteit Brussel, Belgium

Tias Guns   

Data Analytics Lab, Vrije Universiteit Brussel, Belgium

DTAI, KULeuven, Belgium

Building on old ideas to explain domain-specific propagations performed by constraint solvers [7, 2], we recently introduced a method that takes as input a *satisfiable* set of constraints and explains the solution-finding process in a human-understandable way [1]. Explanations in that work are sequences of simple inference steps, a.k.a an *explanation sequence*, involving as few constraints and previously-derived facts as possible. Every explanation derives at least one new fact, implied by a combination of constraints and already derived facts.

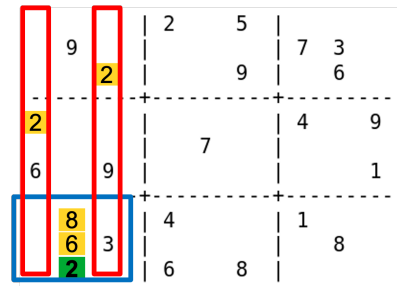
The explanation steps of [1] are heuristically optimized with respect to a given cost function that should approximate human-understandability, e.g., taking the number of constraints and facts into account, as well as an estimate of their cognitive complexity. For example, in the evaluation of explanations for logic grid puzzles, the given clues of the puzzle are considered more difficult than simpler reasoning tricks, and therefore have a higher cost.

In practice, the explanation-generation algorithms presented in our previous work rely heavily on calls for *Minimal Unsatisfiable Subsets* (MUS) [5] of a derived unsatisfiable formula, exploiting a one-to-one correspondence between so-called *non-redundant explanations* and MUSs.

► **Example 1.** (Sudoku) A traditional Constraint Satisfaction Problem (CSP) can be represented as a triple $CSP = (\mathcal{V}, \mathcal{D}, \mathcal{C})$ with variables \mathcal{V} , domain \mathcal{D} and constraint set \mathcal{C} . Let $V_{(r,c)}$, $\mathcal{D}(V_{(r,c)}) = \{1, \dots, 9\}$ be the decision variables of the 9x9 Sudoku CSP.

Let \mathcal{C} be the alldifferent constraints such that:

- all different digits on the same row r ,
 $\forall r \in 1..9 : \text{alldifferent}(\{V_{(r,c)} | c \in 1..9\})$;
- all different digits in the same column c
 $\forall c \in 1..9 : \text{alldifferent}(\{V_{(r,c)} | r \in 1..9\})$;
- all different digits in the same block
 $\forall b_r, b_c \in 0..2 : \text{alldifferent}(\{V_{(r_i,c_j)} |$
 $r_i \in 3b_r + 1 .. 3b_r + 3, c_j \in 3b_c + 1 .. 3b_c + 3\})$.



Let I contain the assigned variables at the current state of the grid (e.g. $I = \{V_{(3,3)} = 2, \dots\}$).

■ **Figure 1** Example of a non-redundant explanation for $V_{(9,2)} = 2$

Figure 1 is an example of a non-redundant explanation that explains explaining how to derive $V_{(9,2)} = 2$. The explanation is generated by computing a MUS of $\{\mathcal{C} \cup I \cup V_{(9,2)} \neq 2\}$. The MUS of the explanation depicted in Figure 1 corresponds to

$$\{\text{alldifferent}(\{V_{(r,1)} | r \in 1..9\}), \text{alldifferent}(\{V_{(r,3)} | r \in 1..9\}), V_{(3,3)} = 2, V_{(4,1)} = 2$$

$$\text{alldifferent}(\{V_{(r_i,c_j)} | r_i \in 7..9, c_j \in 1..3\}, V_{(7,2)} = 2, V_{(8,2)} = 2, V_{(9,2)} \neq 2\}$$

¹ Corresponding author



However, the algorithm developed in [1] has two main weaknesses. First, it provides *no guarantees on the quality* of the produced explanations due to internally relying on the computation of \subseteq -minimal unsatisfiable subsets, which are often suboptimal with respect to the given cost function. Second, it suffers from *performance problems*: the lack of optimality is partly overcome by calling a MUS algorithm on increasingly larger subsets of constraints for each candidate fact to explain. However, using multiple MUS calls per literal in each iteration quickly triggers efficiency problems, causing the explanation generation process to take several hours, even for simple puzzles that are designed to be solvable by humans. We observe that in the explanation setting, many of the individual calls for MUSs can actually be replaced by a single call that searches for an optimal unsatisfiable subset among subsets satisfying certain structural constraints. In this talk, we present the following contributions to tackle the limitations discussed above:

- We develop an algorithm that computes (cost-) **Optimal Constrained** Unsatisfiable Subsets (OCUSs) based on the well-known hitting-set duality that is also used for computing cardinality-minimal MUSs [4, 6].
- We develop techniques for **optimizing** the OCUS algorithms further, exploiting domain-specific information coming from the fact that we are in the *explanation-generation context*. Such optimizations include
 - (1) the development of methods for **information re-use** between consecutive OCUS calls;
 - (2) an explanation-specific version of the OCUS algorithm.
- Finally, we extensively evaluate the different extensions of the base OCUS algorithm on a large number of CSP instances including the puzzles from [1, 3], as well as, generated Sudoku instances of varying difficulty.

The algorithms we present rely on the implicit hitting set duality between Minimum Correction Subsets and Minimal Unsatisfiable Subsets. However, the main bottleneck of this approach is having to repeatedly compute optimal hitting sets, checking its satisfiability by calling the SAT solver and computing a correction subset.

Based on the runtime for explaining the puzzles in our benchmark data set we notice that most of the time is spent in computing hitting sets when calling the OCUS algorithm. This is mainly because the hitting set solver needs to consider an increasingly large collection of sets-to-hit, potentially combining an exponential number of literals and clauses.

To compensate, we develop a correction subset enumeration method that balances the trade-off between efficiency and quality of generated subsets. Results show that, within a given timeout window, our correction subset enumeration method is able to explain more instances, compared to a naive approach, by shifting the computation away from the hitting set solver.

For efficiently generating the whole *explanation sequence*, we introduce incrementality, which allows the re-use of computed information, specifically satisfiable subsets that remain valid from one explanation call to another. Our results show that adding incrementality improves the speed of generating a full explanation sequence and require less sets-to-hit to be computed.

To conclude, with the observed impact of different methods for enumerating correction subsets, an open question remains whether we can quantify precisely, and in a generic way, what a good or even the *best* set-to-hit is in a hitting set approach.

Finally, the concept of (incremental) OCUS is not limited to explanations of satisfaction problems and we are keen to explore other applications too. One potential avenue for future work is to ask how these explanation generating methods map to a constraint optimization setting where branching or searching is required for solving the problem at hand.

References

- 1 Bart Bogaerts, Emilio Gamba, Jens Claes, and Tias Guns. Step-wise explanations of constraint satisfaction problems. In *Proceedings of ECAI*, pages 640–647, 2020.
- 2 Eugene C Freuder, Chavalit Likitvivanavong, and Richard J Wallace. Explanation and implication for configuration problems. In *IJCAI 2001 workshop on configuration*, pages 31–37, 2001.
- 3 Jörg Hoffmann and Daniele Magazzeni. Explainable ai planning (xaip): Overview and the case of contrastive explanation. In *Reasoning Web. Explainable Artificial Intelligence*, pages 277–282. 2019.
- 4 Alexey Ignatiev, Alessandro Previti, Mark Liffiton, and Joao Marques-Silva. Smallest MUS extraction with minimal hitting set dualization. In *Proceedings of CP*, 2015.
- 5 Joao Marques-Silva. Minimal unsatisfiability: Models, algorithms and applications. In *2010 40th IEEE International Symposium on Multiple-Valued Logic*, 2010.
- 6 Paul Saikko, Johannes Peter Wallner, and Matti Järvisalo. Implicit hitting set algorithms for reasoning beyond NP. In *Proceedings of KR*, pages 104–113, 2016.
- 7 Mohammed H Sqalli and Eugene C Freuder. Inference-based constraint satisfaction supports explanation. In *AAAI/IAAI, Vol. 1*, pages 318–325, 1996.