

Sokoban With ASP-Based Hint System

Hannan Khalil, Viktor Tiger, Emil Anderö, Axel Lindh

December 17, 2025

1 Introduction and Motivation

In this project we focus on the domain of puzzle games, specifically the classic game Sokoban originating from Japan. Sokoban is a grid puzzle game where the player moves boxes to designated goal tiles while navigating around walls and other obstacles on the map. A key characteristic of the game is that boxes can only be pushed and never pulled, which means that incorrect decisions can easily lead to unsolvable situations.

This domain is particularly interesting for Answer Set Programming (ASP) because Sokoban can naturally be modeled as a planning and constraint problem. The rules of the game are strict and well-defined and the concept of reaching a solved configuration can be expressed declaratively. The game requires reasoning several steps ahead which makes it suitable for logic-based reasoning techniques.

The scope of this project is to build a text-based Sokoban game and integrate ASP as a solver that can reason about the current game state after every player-move. ASP is used to determine whether a puzzle is solvable and to generate valid sequences of actions and by showing the player the first move in a sequence we can use that as hints. The project does not focus on performance optimization or large-scale evaluation. Instead, the weight is on demonstrating how symbolic reasoning with ASP can be applied to an interactive puzzle domain.

- *Briefly describe your chosen domain (e.g., healthcare, education, smart cities).*
- *Why is this domain interesting or useful for ontology[1]/KG/ASP modeling?*
- *Clearly define the scope of your project (what you cover, what you do not).*

Note:

- In this template, all italic text should be removed and replaced with your own text (which should not be italic); the italic text is just a placeholder letting you know what to write there.
- If you use Figures or Tables, please make sure to give each one a caption and a figure/table number and refer to them from the main text!
- References should be provided where applicable.

2 Methodology

Content depends on your chosen track, but structure is similar

Ontology + KG parts (Tracks 1 + 3)

- *Describe ontology design: main classes, object/data properties, restrictions.*
- *Include a diagram or screenshot of your class hierarchy.*
- *Explain how you populated the KG (data source, manual vs. automated, number of individuals/triples).*

ASP part (Tracks 2 + 3)

- *Describe the ASP encoding: main predicates, rules, constraints.*
- *Explain how the ASP part interacts with the ontology/KG (if applicable).*
- *Explain why ASP is suited for this reasoning.*

Hybrid part (Track 1)

- *Include a simple architecture diagram (e.g., $\text{query} \rightarrow \text{KG} \rightarrow \text{fallback to LLM}$).*
- *Explain how the ontology/KG is used to enhance LLM responses.*

2.1 Evaluation

Describe how you evaluated your system (according to the evaluation suggestions given in the project description), and present the evaluation results.

2.2 Discussion and Conclusion

- *Discuss the strengths and limitations of your approach.*
- *What did you learn about combining symbolic and data-driven AI?*

References

- [1] Guarino, Nicola, Daniel Oberle, and Steffen Staab. "What is an ontology?." *Handbook on ontologies* (2009): 1-17.