Solving Hitori: Applying Answer Set Programming to Hitori

Answer Set Programming (ASP)

```
p(1).
p(2).
q(X) :- p(X),
    ↪not q(X - 1).
```

Grounding →

$$p_1$$
$$p_2$$
$$q_1 \leftarrow \neg q_0$$
$$q_2 \leftarrow \neg q_1$$

high level-language — propositional logic

- grounder keeps applying rules
- solver finds answer set to propositional logic

No circular reasoning
$\{p, q\}$ is not a valid answer set for $p \leftarrow q.\ q \leftarrow p.$

Hitori

Unsolved Hitori puzzle

2	2	2	3
3	1	4	2
4	4	1	4
4	4	3	1

Solution

■ 2	2	■ 2	3
3	1	4	2
4	■ 4	1	■ 4
■ 4	4	3	1

Mark tiles black to satisfy following rules:

- **Uniqueness**: no duplicate numbers visible in row/column
- **Adjacency**: no two black tiles are adjacent
- **Connectivity**: all white tiles must be connected

Hitori is **NP-complete**

Research Questions

1. What is the impact of redundant constraints on performance?
2. What properties of a puzzle impact solving time?
3. How do different paradigms vary in runtime?

Model

Direct translation of first 2 rules.
Connectivity:

- Every white tile must be connected
- A white tile is connected if it's adjacent to a white tile
- Top left corner is considered connected from the start

note: modelling connectivity this way only works because ASP prevents circular reasoning.

Experiments

Custom generator for generating puzzles

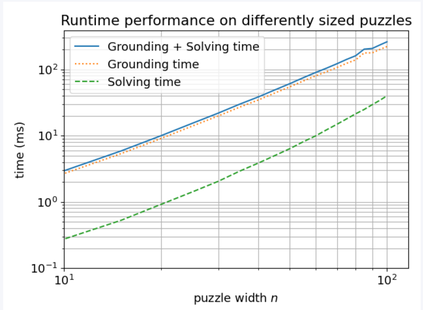
Results

RQ1 Redundant constraints

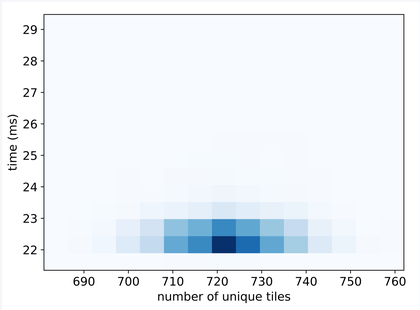
- Most redundant constraints **worsen** solving time
- Many reduce number of conflicts

Results (cont.)

RQ2 Influence of puzzle properties on runtime

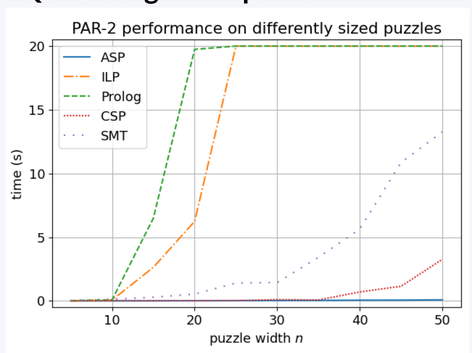


- Grounding time dominates
- 250 ms for 100-by-100 puzzle



- Little variance in solving time
- No or little impact by puzzle properties

RQ3 Paradigm comparison



- ASP scales best
- ASP is the only paradigm that never exceeds the timeout for $n = 50$.

Limitations

- generator might be biased → answers not representative for all puzzles

Sappho de Nooij (s.denooij@student.tudelft.nl)

Research Project CSE3000

Supervisor: Dr. Anna L.D. Latour