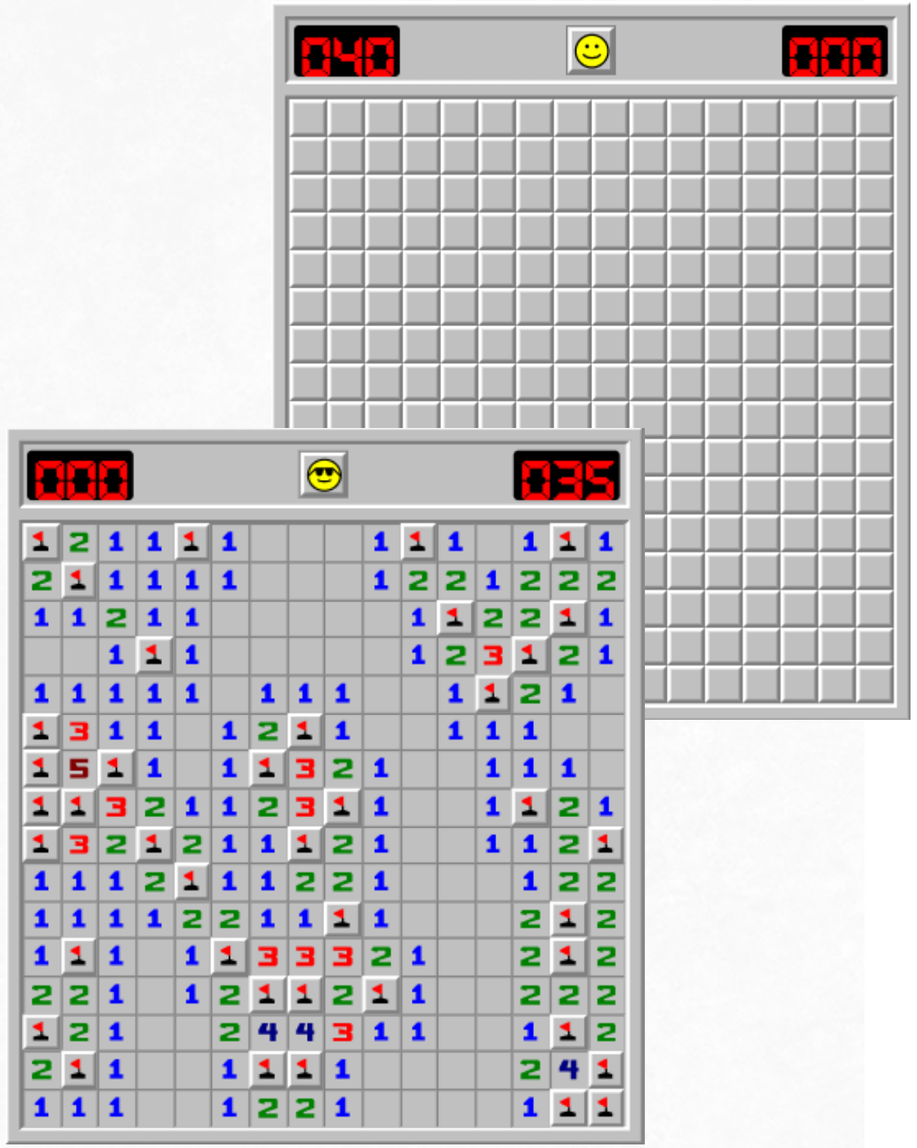
Visualization of Problem Solving with Constraint Processing: Case Study of the Minesweeper

By Chase Resio and Berthe Y. Choueiry

IN OUR GRIT, OUR GLORY™

Minesweeper and NP-Completeness

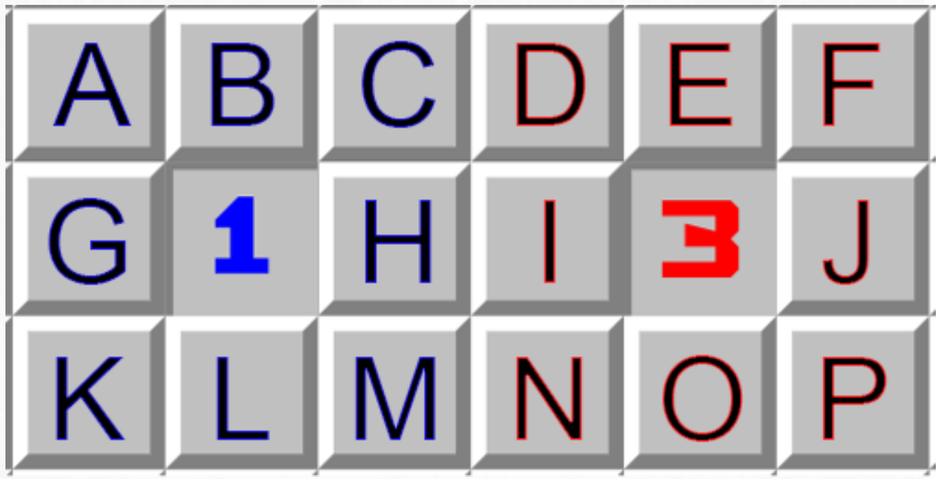
- **Minesweeper is a popular computer game from the 1990's**
- **Click on cells to reveal a mine or a number indicating how many adjacent cells contain mines**
- **Deciding whether an instance of Minesweeper consistent is NP-Complete**



- **Use Constraint Processing to model and solve Minesweeper**
- **Study Constraint Processing**
- **Learn and use React framework**
- **Solve an NP-Complete problem**

Approach

- **Minesweeper is a logic-based game**
- **Model it as a Constraint Satisfaction Problem**
- **Every cell is either safe or a mine**
- **Safe cells contain a number that constrains neighboring cells**
- **Use constraint satisfaction approaches to decide if a cell is safe**



- The 1 places a constraint on cells A,B,C,G,H,K,L,M that one of them must contain a mine and the rest are safe.
- The 3 places a constraint on cells D,E,F,I,J,N,O,P that three of them must contain a mine and the other five are safe.

Web Application

Load
Report Error
About

Board Size

Beginner
Intermediate
Expert
Custom
16 rows
30 cols
99 mines

Cheats

Random
Fringe

Consistency

Unary Constraint
GAC (STR2)
2wiseConsistency (2wC)
3wiseConsistency (3wC)
4wiseConsistency (4wC)
Backbone

Backbone

Solve

Peek
Step
Loop

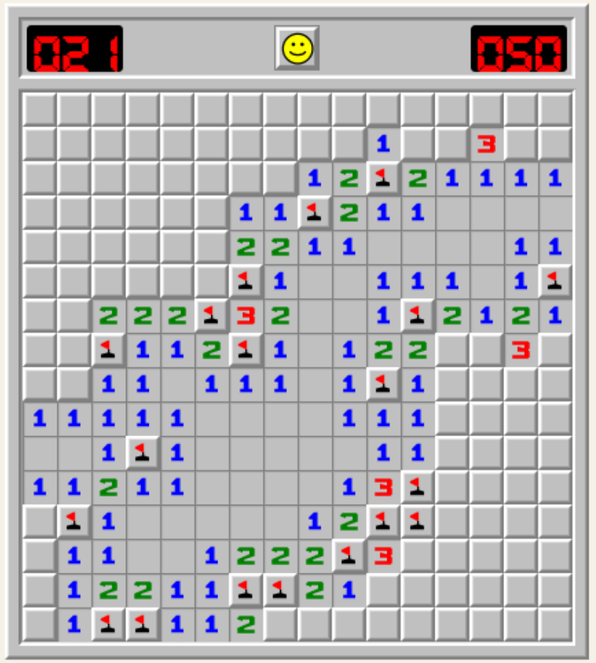
History Log

Undo
Redo

[5,8] revealed 93 cells

Reavealed 54 cells, 19[1]
Unary revealed 10 cells, 10 [1]
GAC revealed 42 additional cells, 9 [1]
2wC revealed 2 additional cells, 0 [1]
3wC revealed 0 additional cells, 0 [1]
4wC revealed 0 additional cells, 0 [1]
Backbone revealed 0 additional cells, 0[1]

Status of 36 cells revealed
Unary reveals 5 cells
GAC reveals 29 additional cells
2wC reveals 2 additional cells
3wC reveals 0 additional cells
4wC reveals 0 additional cells
Backbone reveals 0 additional cells



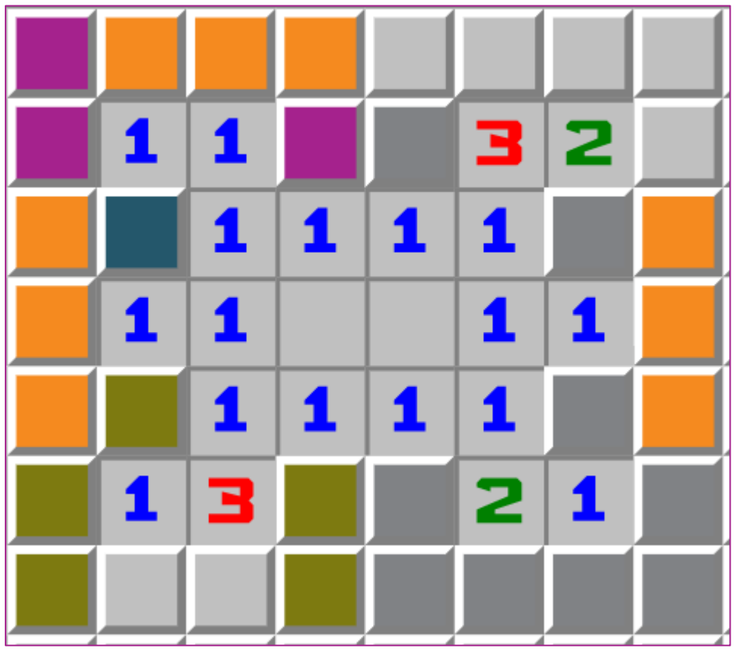
Consistency and Constraint Propagation

Six levels of consistency implemented

- **Unary ensures the consistency of each single constraint**
- **GAC ensures the consistency of each constraint and propagates to other constraints via the shared variables**
- **2wC ensures the consistency of every combination of two constraints with shared cells**

Consistency and Constraint Propagation (Cont.)

- 3wC ensures the consistency of every combination of three constraints that share cells
- 4wC ensures the consistency of every combination of four constraints that share cells
- Backbone ensures the consistency by finding the cell that has the same value in all solutions



- There is currently no way to efficiently solve all instances
- Finding one would have profound impact on computing

Future improvements

- Better mobile view
- Use number of mines to solve cells once at steady state

Acknowledgments

Developers: Kenneth Bayer, Tomo Bessho, Taylor DeMint, Joshua Snyder, and Robert Woodward

UNL UCARE

NSF REU supplements grant RI-1619344