Knowledge Representation and Learning

15. project proposals

Luciano Serafini

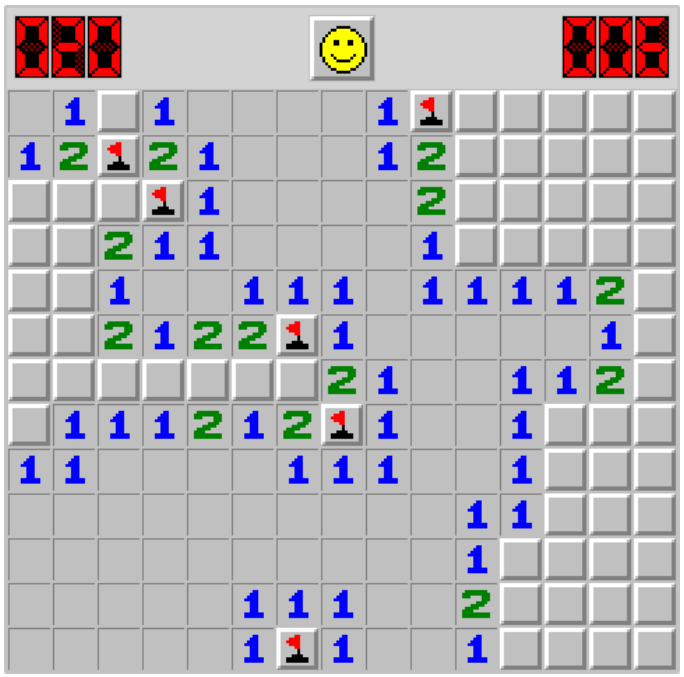
Fondazione Bruno Kessler

May 15, 2023

Minesweeper

Minesweeper is a game where mines are hidden in a grid of squares. The objective is to discover all the mines by opening the cells. At each iteration you have to open a cell if the cell contains a mine you have lost, otherwise you will discover the number of mines in the surrounding cells, which can be used to decide the next cell to be opened.

- Encode any game state (as that shown in picture) in a set of formulas S and use a SAT to decide for every cell i, j if:
- i, j is safe i.e., $S \models \neg mine_{ij}$.
- i, j is unsafe i.e., $S \models mine_{ij}$.
- If you cannot decide if i, j is safe or not (i.e., $S \not\models \neg mine_{ij}$ and $S \not\models mine_{ij}$) use (weighted) model counting to decide the move that minimize the probebility of selecting a mine.



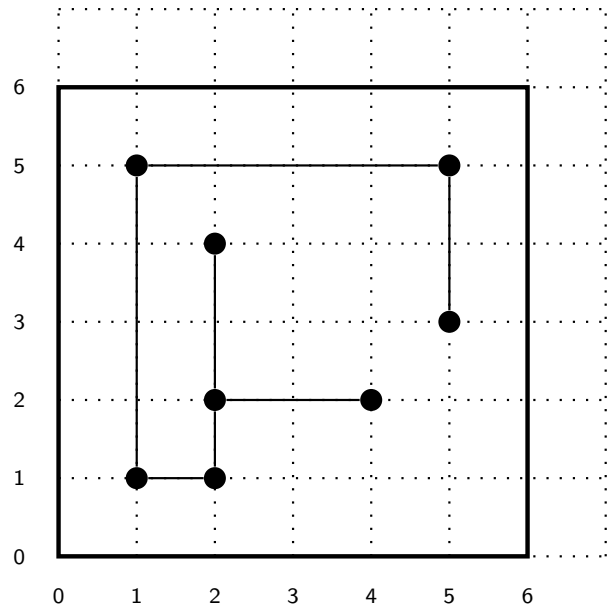
Yashi Game

An instance of the Yashi game is specifid by a $n \times n$ integer grid for some $n > 2$, on which $p > 2$ nodes are placed. A solution of the game consists in drawing horizontal and vertical segments, satisfying the following conditions:

1. No two segments cross each other.
2. The segments form a tree, i.e., they form a graph without cycles. Put differently still, for every two nodes a and b there is exactly one path between a and b.

You can find out more about this game from the website `http://www.sumsumpuzzle.com/yashi.htm`. Given an instance G of Yashi, develop a SAT based method to answer the following questions

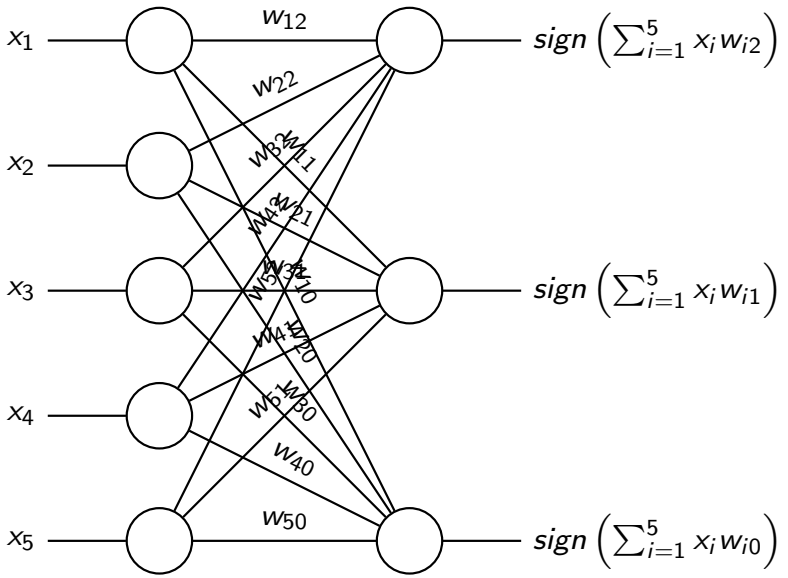
1. Decide if there is a solution for G. If there is, return one solution.
2. Decide if there is a solution for G. If there is, return a minimum-length solution.



Develop your own solution (preferred) or, alternatively, consider the SAT encoding proposed in `https://www.cs.bu.edu/faculty/kfoury/UNI-Teaching/CS512/AK_Documents/Modeling-with-PL/main.pdf`.

Fitting binarized neural networks

Binarized neural networks are nets in which both the weights and activations are binary. Their performance can be surprisingly good, and their implementation can be extremely efficient. In this project you have to show how to fit a binary neural network using SAT. Consider a one layer binary network with n neurons (n=5 in the picture).



where the unknown model parameters w_i are also binary and the function $sign(x)$ returns -1 or 1 based on the sign of x. Given a training set of data/label pairs $\{\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}\}_{i=1}^{d}$, where each $\boldsymbol{x}^{(i)} = (x_1^{(i)}, \ldots, x_n^{(i)}) \in \{-1, 1\}^n$, and $\boldsymbol{y}^{(i)} \in \{-1, 1\}^m$, one has to choose the model parameters w_{ij}, in order to maximize the correct predictions of the net.

Fitting binarized neural networks

1. Write a program that implements a fully connected layer of a binarized neural network with n elements in input and m in output.
2. Propose a binary function $f : \{-1, 1\}^n \to \{-1, 1\}^m$ and use it to generate a set of training data;
3. Train a binarized neural network composed of two fully connected layers n nodes, h intermediate and m^- output nodes.
4. Generate test data and evaluate the network on different n's.

for more details on binarized neural network see e.g. Marc Mezard and Thierry Mora. *Constraint satisfaction problems and neural networks: a statistical physics perspective*. 2008.

Weighted model counting via Knowledge compilation

- Write a method that transforms a formula in sd-DNNF form, and use this method for computing the weighted model count of the formula.
- Evaluate your approach by comparing the results of your method with the explicit computation of weighted model counting via truth table.

Probabilistic reasoning via WMC

- Implement probabilistic inference in bayesian netwoirk using weighted model counting as explained in class.
- Use some dataset availeble in the python package `bnlearn` to test your implementation

Planning with (Max)Sat

Planning domain

- Let $\mathcal{P} = \{p_1, \ldots, p_n\}$ a set of propositional variable
- Any set of propositional variables $s \subset \mathcal{P}$ is a state
- An action $a = (pre(a), eff^+(a), eff^-(a))$
  - $pre(a)$ is a formula in $\mathcal{P}$, the precondition of a
  - $eff^+(a) \subseteq \mathcal{P}$ the positive effects of a
  - $eff^-(a) \subseteq \mathcal{P}$ the negative effects of a
- if $s \models pre(a)$, then $a(s) = s \cup eff^+(a) \setminus eff^-(a)$

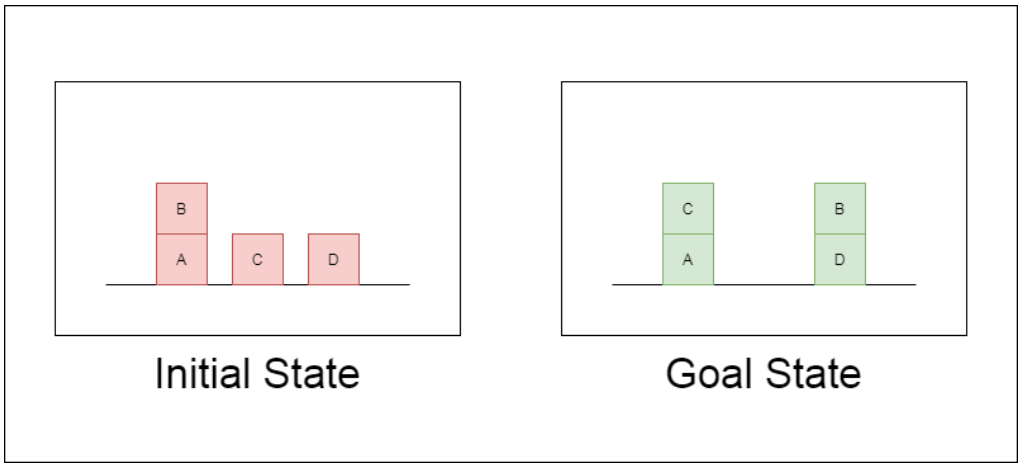
Planning problem

Given a set of actions A, an initial state s_0 and a goal g, which is a formula in $\mathcal{P}$, find a plan, i.e., a sequence of actions $a_1, a_2, \ldots, a_k$ such that

$$a_k(a_{k-1} \ldots a_2(a_1(s)), \ldots) \models g \tag{1}$$

Planning with (Max)Sat

- codify the problem of finding a plan of length less then or equal to k for reaching the goal g from an initial state s_0 as a SAT problem.
- Create a planning domain to solve the planning problem shown in the following picture:



- you can move blocks only if they don't have other blocks on top,
- you can mode them either on top of other blocks or on the table.
- find the smallest k for which a plan exists.

Grounding First Order Logic

Grounding

The grounding of a first order formula Φ that contains no constant anf function symbols on a domain with n elements is recursively defined as follows:

$$Ground(\forall x\, \phi, A) = \bigwedge_{a \in A} Ground(\phi[s/a], A)$$

$$Ground(\exists x\, \phi, A) = \bigvee_{a \in A} Ground(\phi[x/a], A)$$

$$Ground(\phi \circ \phi, A) = Ground(\phi, A) \circ Ground(\phi, A)$$

$$Ground(\neg \phi, A) = \neg Ground(\phi, A)$$

Grounding First Order Logic

- Implement a system that ground first order sentence Φ on a finite domain $\{a_1, \ldots, a_n\}$;
- Use a SAT solver to check satisfiability of the grouned formula;
- If the grounding of Φ is satisfiable, extract from the truth assignment a first order interpretation on the domain $\{a_1, \ldots, a_n\}$ that satisfies Φ.

Reference

 Mezard, Marc and Thierry Mora. *Constraint satisfaction problems and neural networks: a statistical physics perspective*. 2008.