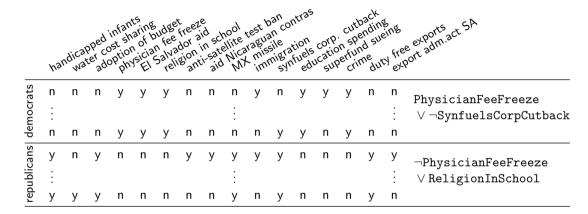
# MCP: Capturing Big Data by Satisfiability

Miki Hermann, École Polytechnique, France Gernot Salzer, TU Wien, Austria

7 July 2021

# 1984 United States Congressional Voting Records Database



#### Goal: Describe large sets of data by propositional formulas

- Extract knowledge: Characterize voting behavior of democrats vs. republicans.
- Classify new data: Given a new voting record, is it by a democrat or a republican?

# Task: find formula satisfying positive and falsifying negative samples

Given two sets of Boolean vectors (tuples) of arity k over the domain  $D=\{0,1\}^k$ , representing positive examples  $T\subseteq D$  and negative examples  $F\subseteq D$ , compute a

[ Horn | dual Horn | bijunctive | affine | general CNF ]

formula  $\varphi$ , such that

- $T \models \varphi$ ,
- for each  $f \in F$ ,  $f \not\models \varphi$ .

#### Caveats

What to do if

- $T \cap F \neq \emptyset$ ,
- $\langle T \rangle_C \cap F \neq \emptyset$  for C = Horn, dual Horn, bijunctive, affine
- $T \cup F \subseteq \{0,1\}^k$ , i.e.  $\{0,1\}^k \setminus (T \cup F) \neq \emptyset$

 $\langle T \rangle_C$  ... closure of vectors in T w.r.t. class C

### Justification

- Horn, dual Horn, bijunctive, and affine formulas are the four families of Boolean formulas, whose satisfiability problem can be decided in polynomial time.
- Horn formulas represent a theoretical background of Prolog programs.
- Horn clauses (implications of the form antecedent → consequent) represent a natural explanation pattern — easy to explain also to a non-expert in computer science or logic.
- The posed problem is an instance of PAC-learning.

# Sketch of the algorithm

Input: Positive and negative samples, T and F, with attributes over finite domains

Convert data to binary, with provisions for enumerations, ordered domains, and intervals.

For the subsets A of the attributes (enumerated by some strategy<sup>(\*)</sup>):

If the samples projected to  $\boldsymbol{A}$  can still be discriminated, then

Compute a Horn/dual Horn/...formula for  $T|_A$ .

Remove redundant literals and clauses.

Return the formula.

Otherwise return "Unsolvable"

Output: Small Horn/dual Horn/... formula that satisfies the positive samples and falsifies the negative ones (in binary form)

(\*) Enumeration strategies: 'begin', 'end', 'lowcard', 'highcard', 'random', 'nosect'

## Choices for Computing the Closure

large: The satisfying assignments of the formula are the *largest* closure of the positive samples not intersecting the negative samples.

exact: The satisfying assignments of the formula are the *smallest* closure of the positive samples not intersecting the negative samples.

## Learning Horn Formulas

- For each  $f \in F$ , determine if  $f \in \langle T \rangle_{\text{Horn}}$  efficiently, without computing the Horn closure.
- Compute the minimal section of  $\langle T \rangle_{\rm Horn}$  and F.
- Compute the Horn formula according to the chosen direction and strategy on the (approximate) minimal section of  $\langle T \rangle_{\rm Horn}$  and F.
- Different algorithms for strategies:
  - large: D. Angluin, M. Frazier, and L. Pitt. Learning conjunctions of Horn clauses. *Machine Learning*, 9(2-3):147–164, 1992.
  - exact: J.-J. Hébrard and B. Zanuttini.

    An efficient algorithm for Horn description.

    Information Processing Letters, 88(4):177–182, 2003.

## Learning Dual Horn Formulas

#### Easy procedure:

- lacktriangle Swap the polarity of the bit vectors in T and F, producing T' and F', respectively.
- ② Compute the Horn formula  $\varphi'$  for T' and F'.
- **3** Swap the polarity of literals in  $\varphi'$ , producing the dual Horn formula  $\varphi$ .

# Learning Bijunctive Formulas

#### Problems:

- There is no known possibility to determine if  $f \in \langle T \rangle_{\text{bijunctive}}$  for each  $f \in F$  without computing the bijunctive closure  $\langle T \rangle_{\text{bijunctive}}$  of T.
- The bijunctive closure  $\langle T \rangle_{\rm bijunctive}$  of T can be (and usually is) time and space consuming.

#### Solution:

- Computes the section using an intersection test,
- Followed by application of the Baker-Pixley Theorem (projection on two coordinates), which implicitly guarantees the bijunctive closure.

## Learning General CNF Formulas

advantage: We get a propositional formula in any case, provided that  $T \cap F = \emptyset$ .

drawback: The produced formula is usually very big.

### Different approaches for strategies:

large: For each false element  $f \in F$  produce the unique clause  $c_f$  which falsifies f. The resulting formula  $\varphi$  is the conjunction of all falsification clauses  $c_f$ .

exact: Algorithm producing a CNF formula in time  $O(|T| \, k^2)$ , where k is the arity/length of tuples in T, using a Boolean restriction of a larger algorithm presented in

A. Gil, M. Hermann, G. Salzer, and B. Zanuttini.

Efficient algorithms for constraint description problems over finite totally ordered domains.

SIAM Journal on Computing, 38(3):922-945, 2008.

### Implementation

- 7000 lines of C++ code
- use of standard library for vectors, deques, ...
- Critical part of the software: computation of the minimal section optimization.
- Three types of parallelization
  - MPI Message Passing Interface
  - POSIX threads
  - hybrid combination of MPI and POSIX threads

### Future extensions

- Browser-compatible front-end
- Generalization to finitely-valued logic to avoid binarization:
  - A. Gil, M. Hermann, G. Salzer, and B. Zanuttini. Efficient algorithms for constraint description problems over finite totally ordered domains. *SIAM Journal on Computing*, 38(3):922–945, 2008.

# Availability and Applications

- http://github.com/miki-hermann/mcp
- Tested on examples from the UCI Machine Learning Repository http://archive.ics.uci.edu/ml/

Try your own eamples in MCP

Thanks for watching