Knowledge Representation and Learning

Theorem Proving and Model Building

Luciano Serafini

Fondazione Bruno Kessler

May 19, 2022

First Order Theorem proving

- A first-order theorem prover is a computer program that proves the validity/unsatisfiability of formulas in first-order logic.
- Since validity in FOL is only semi-decidable, first-order theorem provers are not guaranteed to terminate
- Despite this limitation, many automated theorem provers exist and are useful: Vampire, SPASS, Prover9, ...
- The basis underlying all theorem provers today is the principle of first-order resolution
- To show that a formula is valid, they attempt to derive the empty clause by repeated application of first order resolution to the CNF conversion of the negation of the formula.

First order Model building

- A model builder attempts to build a first order model for a set of formulas and therefore it shows that the set of formulas are satisfiable.
- it is often used in parallel with a theorem prover to build counter-examples of some not yet proved theorem. E.g., a model for $\Gamma \cup \{\neg\phi\}$ proving that ϕ is not a logical consequence of Γ.
- The result of a model builder is a finite set Δ and an interpretation function $\mathcal{I}$ for the first order symbols (constants, funcitons, and predicates) that appear in the set of formulas.
- There are sets of formulas which are satisfiable only by infinite models (i.e., models in which the domain of interpretation is infinite). In this case the model builder does not provide any answer.

The `nltk.inference` module

- The `nltk.inference` module provies interfaces and base classes for theorem provers and model builders.
- There are currently three theorem provers included with NLTK: `Prover9`, `TableauProver`, and `ResolutionProver`. The first is an off-the-shelf prover, while the other two are written in Python and included in the nltk.inference package.
- There is currently a single model builder, which makes use of the external "Mace4" package.

The ProverCommand

- A ProverCommand is a stateful holder for a theorem prover. The state includes:
  - the theorem prover instance;
  - a goal,
  - a list of assumptions,
  - the result of the proof,
  - and a string version of the entire proof.
- there are three ProverCommand implementations: Prover9Command, TableauProverCommand, and ResolutionProverCommand.

The `MaceCommand`

- A `ModelBuilderCommand` is a stateful holder for a model builder. The state includes:
  - the model builder instance;
  - a goal (the formula for which we have to build a counterexample)
  - a list of assumptions,
  - the model;
  - and a string version of the model.
- there is only one ModelBuilderCommand which is `MaceCommand`

Prover9's Proof Method

- The primary mode of inference used by Prover9 is resolution. It repeatedly makes resolution inferences with the aim of detecting inconsistency
- Prover9 will first do some preprocessing on the input file to convert it into the form it uses for inferencing.
  1. First it negates the formula given as a goal
  2. It then translates all formulae into clausal form.
  3. In some cases it will do some further pre-processing (not described here)
- Then it will compute inferences and by default write these standard output.
- If it detects an inconsistency it will stop and print out a proof consisting of the sequence of resolution rules that generated the inconsistency.
- It will also print out various statistics associated with the proof.

Prover9's Proof Method

- Mace4 performs the same preprocessing then Prover9
- When it is called with a set of assumptions Γ and a goal ϕ it tries to "disprove" $\Gamma \models \phi$. by building a first order model for $\Gamma \cup \{\neg\phi\}$.
- It proceeds incrementally on the size of the domain, starting from domain with two elements.
- it is possible to provide an upperbound on the number of elements of the domain.

Mace4 output format (model)

The output of mace4 is the description of a model (with finite domain) and it contains the following information:

- the number n of elements of the domain. The domain is assumed to be $\{0, 1, 2, \ldots n-1\}$
- for every m-aary function f, a tensor F of rank m, i.e., with m dimensions, where for every $0 \leq i_1, \ldots, i_m \leq n-1$, $F_{i_1,\ldots,i_m} \in \{0, \ldots, n-1\}$
  - constants (0-ary functions) $\rightarrow$ scalar in $\{0, \ldots, n-1\}$
  - unary functions $\rightarrow$ vectors in $\{0, \ldots, n-1\}^n$
  - biary functions $\rightarrow$ matrices in $\{0, \ldots, n-1\}^{n\times n}$
- For every m-ary predicate p, a tensor P of rank m, with values in $\{0, 1\}$
  - 0-ary predicates, i.e., propositional variables $\rightarrow$ a value in $\{0, 1\}$
  - unary predicates $\rightarrow$ n vectors with values in $\{0, 1\}$
  - binary predicates $n \times n$ matrixes with values in $\{0, 1\}$.

Prover9 and Mace4 in parallel

- Given a set of assumptions Γ and a goal ϕ if is possible to run in parallel Prover9 and Mace4, which tries to prove and disprove respectively the fact that $\Gamma \models \phi$.