CS262 Logic and Verification

Lecture 6: Semantic tableau

Semantic tableau and resolution

Two proof procedures for propositional logic: **semantic tableau** and **resolution**

Semantic tableau: closely connected to disjunctive normal form (DNF)

Resolution: closely connected to conjunctive normal form (CNF)

Both systems are very well suited for automation;
Assignment: implement a resolution theorem prover in Prolog

Both are refutation systems: To prove that a formula X is a tautology, we begin with $\neg X$ and produce a contradiction

In the following we first talk about tableau, then about resolution

Proof takes the form of a tree, with nodes labelled by propositional formulas.

Example:

$\neg((x \land (y \to (z \lor w))) \to (x \lor y))$

$x \land (y \to (z \lor w))$

$\neg(x \lor y)$

x

$y \to (z \lor w)$

$\neg y$ $\quad$ $z \lor w$

z $\quad$ w

Think of each branch as a conjunction of the formulas on that branch, and think of the tree as the disjunction of all of its branches (disjunction of conjunctions; DNF)

Tableau expansion

In each step, select a branch and a non-literal formula N on that branch.

If $N = \neg\top$, then extend the branch by a node labelled $\bot$ at its end.
If $N = \neg\bot$, then extend the branch by a node labelled $\top$ at its end.
If $N = \neg\neg Z$, then extend the branch by a node labelled Z at its end.
If N is an α-formula, then extend the branch by two nodes labelled α_1, α_2 (**α-expansion**) at its end.
If N is a β-formula, then add a left and right child to the final node of the branch, and label one of them β_1 and the other one β_2 (**β-expansion**).

Tableau expansion rules:

$$\frac{\neg\top}{\bot} \qquad \frac{\neg\bot}{\top} \qquad \frac{\neg\neg Z}{Z} \qquad \frac{\beta}{\beta_1 \mid \beta_2} \qquad \frac{\alpha}{\begin{array}{c}\alpha_1\\ \alpha_2\end{array}}$$

Example

$\neg((p \rightarrow (q \rightarrow r)) \rightarrow ((p \vee s) \rightarrow ((q \rightarrow r) \vee s)))$

Closed tableau

A branch of a tableau is **closed**, if both X and $\neg X$ occur on the branch for some formula X, or if $\perp$ occurs on the branch.

If x and $\neg x$ appear on a branch (where x is a variable), or if $\perp$ appears, then the branch is **atomically closed**.

A tableau is **(atomically) closed**, if every branch is (atomically) closed.

A tableau proof of X is a closed tableau for $\neg X$.

We write $\vdash_t X$ if X has a tableau proof.

Examples

Two tableau proofs of $(p \wedge (q \rightarrow (r \vee s))) \rightarrow (p \vee q)$

Tableau properties

Tableau proofs can be very short compared to truth tables: Consider $X \vee \neg X$ for some complicated formula X

Tableau method extends to first order logic (quantifiers), whereas truth tables do not

Tableau can be generalized to establish propositional consequences $S \models X$, not just tautologies $\models X$.

Tableau rules are non-deterministic: We have freedom in applying them. Different rules may produce proofs of different length.

Implementation

Reuse of formulas: How do we know when we should give up on a proof attempt? We can always try again something we have tried before, and maybe now it will work.

A tableau is **strict**, if no formula has had an expansion rule applied to it twice on the same branch.

Represent the tree as a list of lists (disjunction of conjunctions): Strictness rule allows us to remove an expanded formula from the list

With this, tableau expansion becomes identical to disjunctive normal form expansion.

Implementation

So we could run the DNF expansion algorithm, and check for closure in the very end

We do not need to expand completely before checking for closure!

By being clever about when to check for closure and when to apply which expansion rule, we can shorten a proof considerably

There is considerable scope and necessity for heuristics

Soundness and completeness

Theorem: The tableau proof system is **sound**, i.e., if X has a tableau proof, then X is a tautology.

Theorem: The tableau proof system is **complete**, i.e., if X is a tautology, then the (strict) tableau system will terminate with a proof for it.

Equivalently, $\vdash_t X$ if and only if $\models X$.

First theorem follows from the correctness proof of our DNF expansion algorithm given before: Every expansion step produces a logically equivalent formula.

Proof of second theorem: not given here; requires more advanced tools.