Logic and Proof Past Paper

Proof System

Give a valid proof/model or a falsifying interpretation, a unsatisfiable proof/model or a satisfying interpretation

- y2012p6q6 (a)
  - Comparison using three methods
  - the quantifier steps leave free variables, which must be fresh;
  - otherwise, the same variables in both branches with different instantiation is wrong.
- y2007p6q9, y2008p4q5 (a)
  - satisfiable, valid or neither

Herbrand universe

- y2006p6q9 (b)
  - factoring, unification

Unification

- y2011p6q6 (a)
  - 9 cases, occur check

Decision Procedures, SMT

- y2016p6q6 (a,b)
  - Fourier-Motzkin Variable Elimination
    - Eliminate variables in succession by combining the lower / upper bound.

Modal Logic

- y2009p6q7
  - modal frame
- y2023p6q7 (b)
  - SAT, satisfy simultaneously at a particular world or proof not.
- y2006p5q9 (c)
- y2015p6q6 (b.iii)
- y2011p6q6 (b)
- y2021p6q10 (d)
  - S4 vs S5, accessibility relations $B : A \to \Box \Diamond A$

Falsifying interpretation

- y2020p6q10 (c.i)
- y2019p6q10 (b.iii)
- y2012p6q6 (b)
- y2009p6q7 (c.iii)
- y2007p5q9 (c)
  - S4

1. Sequent Calculus

Apply $(\forall, \Box r)$ and $(\exists, \diamond l)$ first

- check *not* free variables in Γ, Δ
- no substitution needed to avoid introducing free variables
- y2010p6q6 (b), y2017p6q6 (b,c), y2021p6q10 (a,b)
  - Mysterious propositional connective
- y2020p6q10 (c.ii,iii)

[Free-variable] Tableau

Negate; then convert to NNF + Skolem

- y2022p6q9 (c)
  - Valid Proof or Falsify

Sequent Calculus / Tableau

- y2010p6q6 (a)
  - Typo in question (see sols)
- y2015p6q6 (b)
- y2019p6q10 (b i,ii)
  - Free-variable Tableau

2. Clause methods

For set of formulas,

With **negation** $\neg$,

- the empty clause means that the formula is **valid theorem**.
  - Proof by contradiction
- otherwise, falsifying interpretations. (DPLL)

Without negation,

- the empty clause means that the formula is *unsatisfiable*.
- otherwise, satisfying interpretations. (DPLL)

then convert to clause form CNF, Skolem for first-order $\forall, \exists$.

2.1 Resolution

- complete for first-order logic
- Unification (simply) $x_1 \rightarrow a, [a/x_1]$
- Generate new clauses (saturation)

Set of Formulas

- y2023p6q8
- y2019p6q10 (a)
  - *negate* + Skolem
- y2005p5q9 (a)

Set of clauses

- y2022p6q10 (c)
- y2005p5q9 (b,c)

2.2 DPLL

- only works in propositional logic
- decision procedure
- instantiate variables in clauses
- y2018p6q10 (a)
  - def
- y2020p6q10 (a,b)
  - time complexity

Set of clauses

- y2010p6q6 (c)
- y2018p6q10 (b)
  - case split when no unit clause or pure literal

Set of Formulas

- y2021p6q10 (c)
  - satisfying interpretations
- y2008p3q6
  - consistent ***General clause methods***
- y2016p6q5
- y2020p6q9 (c)

Others

- y2022p6q9 (b)
  - adding a clause
- y2019p6q9 (a)
  - SAT solver, Error Analysis
- y2020p6q9 (a,b)
  - Error analysis
- y2022p6q10 (a)
  - DPLL vs Resolution

3. BDD

represents the truth table of a propositional formula by binary decisions, but is a directed graph, sharing identical subtrees.

Procedure for converting a formula to a BDD

Case split recursively on node + boolean simplification over connectives on two sub-BDDs.

- y2012p6q6 (c)
  - conjunction over two sub-BDDs
- y2014p6q5 (b)
- y2018p6q10 (c)
  - disjunction over two sub-BDDs
- y2016p6q6 (c)
- y2019p6q9 (b)
  - implication over two sub-BDDs
- y2017p6q6 (a)
  - identify logically equivalent

Comparison

- y2014p6q5 (a)
- y2022p6q9 (a)
  - DPLL vs BDD
  - BDD: implication, xor