Program Verification
Revision / Background Material

Program Verifiers and Program Verification

UBC, Winter Term 2 (2021)
Alexander J. Summers

Background: Propositional Logic

- Fix an alphabet of *propositional variables* $\mathrm{p},\mathrm{q},\mathrm{r},\mathrm{p}_1,\mathrm{p}_2,\ldots$
- Define *propositional formulas* ($\neg$ binds tighter than $\wedge$, tighter than $\vee$, etc.):

  $\mathrm{A},\mathrm{B} ::= \ \mathrm{p} \ | \ \top \ | \ \bot \ | \ \neg\mathrm{A} \ | \ \mathrm{A}\wedge\mathrm{B} \ | \ \mathrm{A}\vee\mathrm{B} \ | \ \mathrm{A}\Rightarrow\mathrm{B} \ | \ \mathrm{A}\Leftrightarrow\mathrm{B}$
- A *propositional model* M is a partial map from prop. variables to *true/false*
  - Unless stated, we'll assume $\mathrm{dom}(\mathrm{M})$ includes all prop. variables in current formula
- A formula A is *satisfied by a model* M, written $\mathrm{M}\vDash\mathrm{A}$, via usual semantics:

  $\mathrm{M}\vDash\mathrm{p}$ iff $\mathrm{M}(\mathrm{p})$. $\mathrm{M}\vDash\top$ always. $\mathrm{M}\vDash\bot$ never. $\mathrm{M}\vDash\neg\mathrm{A}$ iff $\mathrm{M}\nvDash\mathrm{A}$. etc. …
- A formula A is *valid* iff for *all* models M: $\mathrm{M}\vDash\mathrm{A}$
- A formula A is *satisfiable* iff for *some* model M: $\mathrm{M}\vDash\mathrm{A}$
- A formula A is *unsatisfiable* iff not satisfiable (equivalently, $\neg\mathrm{A}$ is valid)
- A *entails* B (written $\mathrm{A}\vDash\mathrm{B}$) iff for *all* models M: if $\mathrm{M}\vDash\mathrm{A}$ then $\mathrm{M}\vDash\mathrm{B}$
- A and B *equivalent* (written $\mathrm{A}\equiv\mathrm{B}$) iff for *all* models M: $\mathrm{M}\vDash\mathrm{A}$ iff $\mathrm{M}\vDash\mathrm{B}$

Background: Conjunctive Normal Form

- A *literal* is a variable or the negation of one (p,$\neg\mathrm{p}$,…)
  - For a literal l we write $\sim\mathrm{l}$ for the negation of l, cancelling double negations
- A *clause* is a disjunction of (any finite number of) literals
  - e.g. $\mathrm{p}\vee\neg\mathrm{q}$, $\mathrm{q}\vee\mathrm{r}\vee\neg\mathrm{r}$, q are all clauses
  - the *empty clause* (0 disjuncts) is defined to be $\perp$ (why?)
  - a *unit clause* is just a single literal (exactly 1 disjunct)
  - a variable p *occurs positively in a clause* iff p is one of the clause's disjuncts
  - a variable p *occurs negatively in a clause* iff $\neg\mathrm{p}$ is one of the clause's disjuncts
- A formula A is in *conjunctive normal form (CNF)* iff
  it is a conjunction of (any finite number of) clauses
  - i.e. a conjunction of disjunctions of literals
  - an *empty conjunction* (0 disjuncts) is defined to be $\top$ (why?)
  - e.g. $(\mathrm{p}\vee\neg\mathrm{q})\wedge(\mathrm{q}\vee\mathrm{r}\vee\neg\mathrm{r})$, $(\mathrm{p}\wedge\neg\mathrm{q})$, q are in CNF ($\mathrm{p}\vee\neg\mathrm{q}\wedge\mathrm{q}\vee\mathrm{r}\vee\neg\mathrm{r}$ is not - why?)

Background: Propositional Equivalences

- Some important propositional logic equivalences (for all A, B, C):

$$\neg\neg A \equiv A \quad \text{and} \quad \neg\top \equiv \bot \quad \text{and} \quad \neg A \equiv A \Rightarrow \bot$$

$$A \land B \equiv B \land A \quad \text{and} \quad A \lor B \equiv B \lor A \quad \text{and} \quad A \Leftrightarrow B \equiv B \Leftrightarrow A$$

$$A \land \top \equiv A \quad \text{and} \quad A \land \bot \equiv \bot \quad \text{and} \quad A \lor \top \equiv \top \quad \text{and} \quad A \lor \bot \equiv A$$

$$(A \land B) \lor C \equiv (A \lor C) \land (B \lor C) \quad \text{and} \quad (A \lor B) \land C \equiv (A \land C) \lor (B \land C)$$

$$\neg(A \lor B) \equiv \neg A \land \neg B \quad \text{and} \quad \neg(A \land B) \equiv \neg A \lor \neg B$$

$$A \Rightarrow B \equiv (\neg A \lor B) \equiv \neg(A \land \neg B) \equiv \neg B \Rightarrow \neg A$$

$$A \Leftrightarrow B \equiv (A \Rightarrow B) \land (B \Rightarrow A) \equiv (A \land B) \lor (\neg A \land \neg B)$$

$$A \Rightarrow B \lor C \equiv A \land \neg B \Rightarrow C \quad \text{and} \quad A \land B \Rightarrow C \equiv A \Rightarrow \neg B \lor C$$

Make sure that you're comfortable with understanding and using these

For propositional logic lecture notes, see e.g. CPSC 121 course materials, or e.g. https://www.doc.ic.ac.uk/~imh/teaching/140_logic/140.pdf (p. 53-58)

Background: Sorted First-Order Logic - Syntax

- Fix a set of *sorts* (types), T_1, T_2, … (typically includes Bool)
- For each sort, fix an alphabet of *variables* $\mathrm{x},\mathrm{y},\mathrm{z},\mathrm{x}_1$,…
- Fix a set of *function symbols* $\mathrm{f},\mathrm{g},\mathrm{h},\mathrm{f}_1$,… each with a *function signature*
  - a *function signature* defines an *arity* (≥ 0), sort for each argument, return sort
  - nullary functions are also referred to as *constant symbols*
- Then we can define *first-order terms* $\mathrm{t},\mathrm{s} ::= \mathrm{x} \mid \mathrm{f}(\mathrm{t}_1,\mathrm{t}_2,\ldots)$
  - we assume all terms to be type-correct (*well-sorted*) and to respect arities
- A *signature* is a set of sorts and function symbols (over those sorts)
- For a given signature, *first-order assertions* A are defined by $\mathrm{A},\mathrm{B} ::= \mathrm{t} \mid \neg\mathrm{A} \mid \mathrm{A}\wedge\mathrm{B} \mid \mathrm{A}\vee\mathrm{B} \mid \mathrm{A}\Rightarrow\mathrm{B} \mid \mathrm{A}\Leftrightarrow\mathrm{B} \mid \forall\mathrm{x}{:}\mathrm{T}.\mathrm{A} \mid \exists\mathrm{x}{:}\mathrm{T}.\mathrm{A}$
  - Here, t must be a term of (interpreted) sort Bool (assertions also Bool terms)
  - We write $\mathrm{FV}(\mathrm{A})$ for set of variables occurring free (not bound by $\forall$/$\exists$) in A

Background: Sorted First-Order Logic - Semantics

- A *first-order model* M is a (partial) map mapping:
  - sorts to *non-empty sets of values*: the *interpretation of the sort in* M
  - variables to *elements* (i.e. values) of the interpretations of their sorts
  - function symbols to (total) *mathematical functions* with appropriate arity, domain/range according to interpretations of function arguments/return sort
- The *value of a term* t *in a model* M, written $\ulcorner \mathrm{t} \lrcorner_{\mathrm{M}}$, is defined by:
  $\ulcorner \mathrm{x} \lrcorner_{\mathrm{M}} = \mathrm{M}(\mathrm{x})$ and $\lceil \mathrm{f}(\mathrm{t}_1,\mathrm{t}_2,\ldots) \rfloor_{\mathrm{M}} = \mathrm{M}(\mathrm{f})(\lceil \mathrm{t}_1 \rfloor_{\mathrm{M}},\lceil \mathrm{t}_2 \rfloor_{\mathrm{M}},\ldots)$
- A formula A is *satisfied by a model* M, written $\mathrm{M} \vDash \mathrm{A}$, defined by:

$\mathrm{M} \vDash \mathrm{t}$ iff $\ulcorner \mathrm{t} \lrcorner_{\mathrm{M}} =$true. $\mathrm{M} \vDash \neg \mathrm{A}$ iff $\mathrm{M} \nvDash \mathrm{A}$ …(usual propositional cases)

$\mathrm{M} \vDash \forall \mathrm{x}:\mathrm{T}.\mathrm{A}$ iff for all values $\mathrm{v} \in \mathrm{M}(\mathrm{T})$, $\mathrm{M}[\mathrm{x} \mapsto \mathrm{v}] \vDash \mathrm{A}$

$\mathrm{M} \vDash \exists \mathrm{x}:\mathrm{T}.\mathrm{A}$ iff for some value $\mathrm{v} \in \mathrm{M}(\mathrm{T})$, $\mathrm{M}[\mathrm{x} \mapsto \mathrm{v}] \vDash \mathrm{A}$

- here $\mathrm{M}[\mathrm{x} \mapsto \mathrm{v}]$ denotes a new model mapping x to v and otherwise unchanged

Background: First-Order Logic Equivalences

- Note: we will sometimes omit sorts from quantifiers when irrelevant

$$\forall x_1.\forall x_2.\ A \ \equiv\ \forall x_2.\forall x_1.\ A$$

$$\exists x_1.\exists x_2.\ A \ \equiv\ \exists x_2.\exists x_1.\ A$$

$\neg\forall x.A \ \equiv\ \exists x.\neg A$ and $\neg\exists x.A \ \equiv\ \forall x.\neg A$

$$\forall x.(A\wedge B) \ \equiv\ (\forall x.A)\wedge(\forall x.B)$$

$$\exists x.(A\vee B) \ \equiv\ (\exists x.A)\vee(\exists x.B)$$

If $x\notin FV(A)$ then $\exists x.A \ \equiv\ A \ \equiv \forall x.A$ and
$\forall x.(A \vee B) \ \equiv\ A\vee(\forall x.B)$ and $\exists x.(A\wedge B) \ \equiv\ A\wedge(\exists x.B)$

Make sure that you're comfortable with understanding and using these

For lecture notes on first-order logic, see CPSC 121 , or e.g.
https://www.doc.ic.ac.uk/~imh/teaching/140_logic/140.pdf (p. 199-203)