Reference Sheet for CO140 Logic

Autumn 2016

1 Definitions

1.1 Propositional Logic

Binding Conventions (Strongest) $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ (Weakest).

Propositional Formula

1. A propositional atom is a formula.
2. $\top$ and $\bot$ are formulas.
3. If A is a formula then so is $(\neg A)$.
4. If A, B are formulas then so are $(A \wedge B), (A \vee B), (A \rightarrow B), (A \leftrightarrow B)$.

Principle Connective Connective at the root (top) of a formation tree. A formula with principle connective $\leftrightarrow$ is said to have the **logical form** $A \leftrightarrow B$.

Subformulas Correspond to the subtrees of a formation tree.

Atomic Formula of the form $\top, \bot, p$ for an atom p.

Negated Formula, Conjunction, Disjunction, Implication Formula whose logical form is $\neg A, A \wedge B, A \vee B, A \rightarrow B$ respectively.

Literal Formula that is atomic or negated-atomic.

Clause Disjunction of one or more literals.

Situation Determines whether each propositional atom is true or false.

Evaluation

1. $\top$ is true, $\bot$ is false.
2. $\neg A$ has the opposite truth value to A.
3. $A \wedge B$ is true if A and B are both true.
4. $A \vee B$ is true if one or both of A and B are true.
5. $A \rightarrow B$ is true if A is false or B is true (or both).
6. $A \leftrightarrow B$ is true if A and B have the same truth value.

Valid Argument Given formulas $A_1, A_2, \ldots, A_n, B$ an argument $A_1, A_2, \ldots, A_n \vDash B$ is valid if B is true in any situation in which $A_1, A_2, \ldots, A_n$ are all true. Here $\vDash$ denotes logical entailment.

Valid Formula A formula A is valid if it is true in every situation, i.e. $\vDash A$. A **tautology** is a valid propositional formula.

Satisfiable Formula True in at least one situation.

Equivalent Formulas True in exactly the same situations, i.e. $A \equiv B$.

Disjunctive Normal Form Formula as a disjunction of conjunctions of literals, not further simplifiable.

Conjunctive Normal Form Formula as a conjunction of disjunction of literals, not further simplifiable.

Normal Form

1. Get rid of $\rightarrow, \leftrightarrow$ using equivalences.

2. Use De Morgan laws to push negations down to atoms. Delete any double negations.

3. Rearrange using distributivity into the required normal form.

4. Use equivalences to simplify as far as possible (e.g. using absorption, idempotence, equivalences involving $\top$ and $\bot$).

Theorem Formula that can be established by a given proof system, i.e. any A such that $\vdash A$. (Note that $\vdash$ is syntactic whilst $\vDash$ is semantic - $A_1, A_2, \ldots, A_n \vDash B$ means there is a proof of B starting with $A_1, A_2, \ldots, A_n$ as givens).

Soundness Any provable formula is valid, i.e. if $A_1, A_2, \ldots, A_n \vdash B$ then $A_1, A_2, \ldots, A_n \vDash B$.

Completeness Any valid formula can be proved, i.e. if $A_1, A_2, \ldots, A_n \vDash B$ then $A_1, A_2, \ldots, A_n \vdash B$.

Consistency A formula is consistent if $\nvdash \neg A$. So a formula is consistent if and only if it is satisfiable.

1.2 Predicate Logic

Binding Conventions (Strongest) $(\neg, \forall x, \exists x), \wedge, \vee, \rightarrow, \leftrightarrow$ (Weakest).

Signature Collection of constants and relation symbols and function symbols with specified arities.

Term For a signature L:

1. Any constant in L is an L-term.

2. Any variable is an L-term.

3. For an n-ary function symbol f in L and L-terms $t_1, t_2, \ldots, t_n$, $f(t_1, t_2, \ldots, t_n)$ is an L-term.

Closed / Ground Term Does not involve a variable.

Bound Variable For a formula A and variable x, x is bound if it lies under a quantifier $\forall x$ or $\exists x$ in the formation tree of A.

Free Variable Variable which is not bound (this includes variables which do not appear in A!).

Sentence Formula with no free variables. (Does not require an assignment for evaluation).

Structure For a signature L, and L-structure M:

1. Identifies a non-empty collection of objects that M 'knows about', i.e. the **domain** of M, $\text{dom}(M)$.

2. Specifies what the symbols of L mean in terms of these objects (constants specify objects in $\text{dom}(M)$ and relations specify relations between objects in $\text{dom}(M)$).

3. For an n-ary function symbol f in L, specifies which object f associates with each sequence of objects $(a_1, a_2, \ldots, a_n)$ in $\text{dom}(M)$.

For a constant c, c^M denotes the object $\text{dom}(M)$ that c names in M.

For a function f, $f^M(a_1, a_2, \ldots, a_n)$ denotes the object $\text{dom}(M)$ that $f(a_1, a_2, \ldots, a_n)$ names in M.

If a formula A is true in M, we say $M \vDash A$.

Assignment For a structure M, allocates an object in $\text{dom}(M)$ to each variable.

If a formula A is true in M under h, we say $M, h \vDash A$.

Value of Term For a signature L, an L-structure M and an assignment h, for any L-term t, the value of t in M under h is the object in $\text{dom}(M)$ allocated to t by:

1. M if t is a constant, i.e. the object t^M.

2. h if t is a variable, i.e. the object $h(t)$.

3. f^M if t is a function on terms, i.e. $f(t_1, t_2, \ldots, t_n)$ is the object $f^M(a_1, a_2, \ldots, a_n)$ where a_i is the value of t_i in M under h.

Predicate Formula For an L-structure M and an assignment h:

1. For an n-ary relation symbol in L, and L-terms $t_1, t_2, \ldots, t_n$, $R(t_1, t_2, \ldots, t_n)$ is an atomic L-formula.
   $M, h \vDash R(t_1, t_2, \ldots, t_n)$ if M says the sequence $(a_1, a_2, \ldots, a_n)$ is in the relation R, where a_i is the value of t_i in M under h.

2. For L-terms t_1, t_2, $t_1 = t_2$ is an atomic L-formula.
   $M, h \vDash t_1 = t_2$ if t_1 and t_2 have the same value in M under h.

3. $\top, \bot$ are atomic L-formulas.
   $M, h \vDash \top$ and $M, h \nvDash \bot$.

4. For L-formulas A, B, $(\neg A), (A \wedge B), (A \vee B), (A \to B), (A \leftrightarrow B)$ are L-formulas.
   $M, h \vDash A \wedge B$ if $M, h \vDash A$ and $M, h \vDash B$, etc.

5. For an L-formula A and a variable x, $(\forall x A)$ and $(\exists x A)$ are L-formulas.
   $M, h \vDash \forall x A$ if $M, g \vDash A$ for every assignment g into M with $g =_x h$ and $M, h \vDash \exists x A$ if $M, g \vDash A$ for some assignment g into M with $g =_x h$. (The notation $g =_x h$ here means g agrees with h except perhaps on x).

Some Translation Advice Note that:

1. $\forall x\,(\text{lecturer}(x) \wedge \text{human}(x))$ says everything is a lecturer and a human.

2. $\forall x\,(\text{lecturer}(x) \to \text{human}(x))$ says every lecturer is a human.

3. $\exists x\,(\text{lecturer}(x) \wedge \text{human}(x))$ says there is a lecturer that is also a human.

4. $\exists x\,(\text{lecturer}(x) \to \text{human}(x))$ says there is a non-lecturer, or there is a lecturer that is also a human.

Counting:

1. No lecturers: $\neg\exists x\,(\text{lecturer}(x))$.

2. At least one lecturer: $\exists x\,(\text{lecturer}(x))$.

3. At least two lecturers: $\exists x \exists y\,(\text{lecturer}(x) \wedge \text{lecturer}(y) \wedge x \neq y)$ or $\forall x \exists y\,(\text{lecturer}(y) \wedge y \neq x)$.

4. At least three lecturers: similar to above, e.g. $\forall x \forall y \exists z\,(\text{lecturer}(z) \wedge z \neq x \wedge z \neq y)$.

5. At most one lecturer: $\neg\exists x \exists y\,(\text{lecturer}(x) \wedge \text{lecturer}(y) \wedge z \neq y)$ or $\forall x \forall y\,(\text{lecturer}(x) \wedge \text{lecturer}(y) \to x = y)$ or $\exists x \forall y\,(\text{lecturer}(y) \to y = x)$.

6. Exactly one lecturer: at least one lecturer$\wedge$at most one lecturer or $\exists x \forall y\,(\text{lecturer}(y) \leftrightarrow y = x)$.

Remember:

1. Always consider the vacuous case when using $\forall$.

2. The order of quantifiers is very important!

Valid Argument For a signature L, and L-formulas $A_1, A_2, \ldots, A_n, B$, the argument $A_1, A_2, \ldots, A_n \vDash B$ is valid if for any L-structure M and assignment h into M, if $M, h \vDash B$ given $M, h \vDash A_1, M, h \vDash A_2, \ldots, M, h \vDash A_n$.

Valid Formula The L-formula A is valid if for for all L-structures M and assignment h into M, $M, h \vDash A$. We say $\vDash A$.

Satisfiable Formula The L-formula A is satisfiable if for for some L-structure M and assignment h into M, $M, h \vDash A$.

Equivalent Formulas The L-formulas A and B are equivalent if for for every L-structure M and assignment h into M, $M, h \vDash A$ if and only if $M, h \vDash B$.

1.3 Many-Sorted Predicate Logic

Term Redefined such that:

1. Each variable and constant comes with a sort s. We indicate this as $x : \mathrm{s}$ and $c : \mathrm{s}$.

2. Each n-ary function symbol f comes with a template $f : (\mathrm{s}_1, \mathrm{s}_2, \ldots, \mathrm{s}_n) \to \mathrm{s}$.

Formula Redefined such that:

1. Each n-ary relation symbol R comes with a template $R(\mathrm{s}_1, \mathrm{s}_2, \ldots, \mathrm{s}_n)$.

2. $t_1 = t_2$ is a formula if t_1, t_2 have the same sort.

It is polite to indicate the sort of a variable in $\forall, \exists$, e.g. $\forall x : \text{lecturer} \exists y : \text{Sun}\,(\text{bought}_{lecturer,Sun}(x, y))$.

1.4 Formal Specification of Programs

Pre-condition Formula $A(x_1, x_2, \ldots, x_n)$ such that any arguments $(a_1, a_2, \ldots, a_n)$ satisfy the pre-condition iff $A(a_1, a_2, \ldots, a_n)$ is true. If there is no restrictions on arguments beyond type information, we write 'none' or $\top$.

Post-condition Formula expressing intended value of a funtion in terms of arguments.

Lists We can define a signature suitable for lists of type [Nat]:
Constants:

- $\underline{0}, \underline{1}, \cdots$: Nat

Relations:

- $<, \leq, >, \geq$: (Nat,Nat)

Functions:

- $+, -, \times$: (Nat,Nat)$\rightarrow$Nat
- [] : [Nat]
- cons(:) : (Nat,[Nat])$\rightarrow$[Nat]
- ++ : ([Nat],[Nat])$\rightarrow$[Nat]
- head : [Nat]$\rightarrow$Nat
- tail : [Nat]$\rightarrow$[Nat]
- # : [Nat]$\rightarrow$Nat
- !! : ([Nat],Nat)$\rightarrow$Nat

Pre-conditions for functions on lists are usually $\top$ or sometimes involve checking the list is non-empty ($\#xs > 0$).
Post-conditions for functions on lists often involve:

1. Checking a property of a list and its length compared to the given list (e.g. $\#xs = \#ys \wedge \forall i : \text{Nat}\,(i < \#xs \rightarrow P\,(ys!!i))$ where $ys = f(x)$).
2. Checking a property of an item and its presence in the given list (e.g. $\exists i : \text{Nat}\,(i < \#xs \wedge xs!!i = y) \wedge P\,(y)$ where $y = f(x)$).

2 Checking Validity

We can use:

1. Truth tables - but not for predicate logic
2. Direct argument
3. Equivalences
4. Proof systems - e.g. natural deduction

2.1 Direct Argument

Propositional Logic

1. Take an arbitrary situation.
2. Prove that the formula is true in this situation. (Often this will require the law of excluded middle - argument by cases).

Predicate Logic To show the argument $A_1, A_2, \ldots, A_n \vDash B$ is valid:

1. Consider any M such that $M \vDash A_1, M \vDash A_2, \ldots, M \vDash A_n$.
2. Show $M \vDash B$, e.g.:
   (a) $M \vDash \forall x\,(B\,(x))$: Consider an arbitrary object a in $\text{dom}\,(M)$. Show $M \vDash B\,(a)$.
   (b) $M \vDash \exists x\,(B\,(x))$: Consider any object b in $\text{dom}\,(M)$. Show $M \vDash B\,(b)$.

2.2 Equivalences

When using equivalences, you *must* justify every step by stating the equivalence you used. Remember you can work from either direction. Note that distributivity is often very helpful when applied backwards.
Be especially careful with $\wedge$ and $\vee$!

$\neg$

1. $\neg\top \equiv \bot$
2. $\neg\bot \equiv \top$
3. $\neg\neg A \equiv A$
4. $\neg(A \wedge B) \equiv \neg A \vee \neg B$ (De Morgan)
5. $\neg(A \vee B) \equiv \neg A \wedge \neg B$ (De Morgan)

$\wedge$

1. $A \wedge B \equiv B \wedge A$ (Commutativity)
2. $A \wedge A \equiv A$ (Idempotence)
3. $A \wedge \top \equiv A$
4. $\bot \wedge A \equiv \neg A \wedge A \equiv \bot$
5. $(A \wedge B) \wedge C \equiv A \wedge (B \wedge C)$ (Associativity)
6. $A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$ (Distributivity)
7. $A \wedge (A \vee B) \equiv A$ (Absorption)

$\vee$

1. $A \vee B \equiv B \vee A$ (Commutativity)
2. $A \vee A \equiv A$ (Idempotence)
3. $\top \vee A \equiv \neg A \vee A \equiv \top$
4. $A \vee \bot \equiv A$
5. $(A \vee B) \vee C \equiv A \vee (B \vee C)$ (Associativity)
6. $A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$ (Distributivity)
7. $A \vee (A \wedge B) \equiv A$ (Absorption)

$\to$

1. $A \to A \equiv \top$
2. $\top \to A \equiv A$
3. $A \to \top \equiv \top$
4. $\bot \to A \equiv \top$
5. $A \to \bot \equiv \neg A$
6. $A \to B \equiv \neg A \vee B \equiv \neg (A \wedge \neg B)$
7. $\neg(A \to B) \equiv A \wedge \neg B$

$\leftrightarrow$

1. $A \leftrightarrow B \equiv (A \to B) \wedge (B \to A) \equiv (A \wedge B) \vee (\neg A \wedge \neg B) \equiv \neg A \leftrightarrow \neg B$
2. $\neg (A \leftrightarrow B) \equiv A \leftrightarrow \neg B \equiv \neg A \leftrightarrow B \equiv (A \wedge \neg B) \vee (\neg A \wedge B)$

$\forall, \exists$

1. $\forall x \forall y A \equiv \forall y \forall x A$
2. $\exists x \exists y A \equiv \exists y \exists x A$
3. $\neg \forall x A \equiv \exists x \neg A$
4. $\neg \exists x A \equiv \forall x \neg A$
5. $\forall x (A \wedge B) \equiv \forall x A \wedge \forall x B$
6. $\exists x (A \vee B) \equiv \exists x A \vee \exists x B$

For $\boldsymbol{A}$ in which $\boldsymbol{x}$ does not Occur Free:

1. $A \equiv \forall x A \equiv \exists x A$
2. $\exists x (A \wedge B) \equiv A \wedge \exists x B$
3. $\forall x (A \vee B) \equiv A \vee \forall x B$
4. $\exists x (A \to B) \equiv A \to \exists x B$
5. $\forall x (A \to B) \equiv A \to \forall x B$
6. $\exists x (B \to A) \equiv \forall x B \to A^*$
7. $\forall x (B \to A) \equiv \exists x B \to A^*$

* Watch out for these two cases!

$=$

1. $t = t \equiv \top$
2. $t = u \equiv u = t$
3. **Leibniz Principle** If A is a formula in which x occurs free and y does not occur and B is the formula obtained from A by replacing one or more free occurences of x by y, then $x = y \to (A \leftrightarrow B) \equiv \top$.

Renaming Bound Variables The formula in which all bound occurrences of a variable and the respective quantifiers are changed to a new variable is equivalent to the original formula.

2.3 Natural Deduction

When using natural deduction, remember line numbers and reasoning are required for *every* step. You should take time to check your answers.

$\wedge$-Introduction

1	A	
2	B	
3	$A \wedge B$	$\wedge I\,(1,2)$

$\wedge$-Elimination

1	$A \wedge B$	
2	A	$\wedge E\,(1)$
3	B	$\wedge E\,(1)$

$\vee$-Introduction

1	A	
2	$A \vee B$	$\wedge I\,(1)$
3	$B \vee A$	$\wedge I\,(1)$

$\vee$-Elimination

1	$A \vee B$				
2	A	ass	4	B	ass
3	C		5	C	
6	C	$\vee E\,(1,2,3,4,5)$			

$\rightarrow$-Introduction

1	A	ass
2	B	
3	$A \rightarrow B$	$\rightarrow I\,(1,2)$

$\rightarrow$-Elimination

1	$A \rightarrow B$	
2	A	
3	B	$\rightarrow E\,(1,2)$

$\leftrightarrow$-Introduction

1	$A \rightarrow B$	
2	$B \rightarrow A$	
3	$A \leftrightarrow B$	$\leftrightarrow I\,(1,2)$

$\leftrightarrow$-Elimination

1	$A \leftrightarrow B$			1	$A \leftrightarrow B$	
2	A		or	2	B	
3	B	$\leftrightarrow E\,(1,2)$		3	A	$\leftrightarrow E\,(1,2)$

$\neg$-Introduction

1	A	ass
2	$\perp$	
3	$\neg A$	$\neg I\,(1,2)$

$\neg$-Elimination / $\perp$-Introduction

1	A	
2	$\neg A$	
3	$\perp$	$\neg E\,(1,2)$ or $\perp I\,(1,2)$

$\perp$-Elimination

1	$\perp$	
2	A	$\perp E\,(1)$

$\neg\neg$-Elimination

1	$\neg\neg A$	
2	A	$\neg\neg E\,(1)$

Excluded Middle 1 $A \vee \neg A$ lemma

Proof by Contradiction

1	$\neg A$	ass
2	$\perp$	
3	A	$PC\,(1,2)$

$\exists$-Introduction

1	$A\,(t/x)$	
2	$\exists x A$	$\exists I\,(1)$

$\exists$-Elimination

1	$\exists x A$	
2	$A\,(c/x)$	ass
3	B	
4	B	$\exists E\,(1,2,3)$

$\forall$-Introduction

2	c	$\forall I$ const
3	$A\,(c/x)$	
4	$\forall x A$	$\forall I\,(1,2)$

$\forall$-Elimination

1	$\forall x A$	
2	$A\,(t/x)$	$\forall E\,(1)$

∀ →-Elimination	1	$\forall x\,(A\,(x) \rightarrow B\,(x))$	
	2	$A\,(t/x)$	
	3	$B\,(t/x)$	$\forall \rightarrow E\,(1,2)$

Reflexivity	1	$t = t$	refl

Substitution	1	$A\,(t/x)$	
	2	$t = u$	
	3	$A\,(u/x)$	sub$(1,2)$

Symmetry	1	$c = d$	
	2	$d = c$	sym(1)