02286 Logic in Computer Science, Artificial Intelligence and Multi-agent Systems

Christian Gram Kalhauge s093273
Andreas Frøsig s093264
Kenneth Balsiger Andersen s093252

December 8, 2012

Part I
Important Notation

Notation

Part II
Classical Logic (45%)

1 Propositional Logic

1.1 Definition

$$\phi := p|\neg\phi|\phi \vee \phi \tag{1}$$

Where p is a proposition . A proposition is logical variable that can hold the values $\top$ or $\bot$.

Examples on propositions

- Your mom is fat!
- The Moon is made of cheese!

Rules Where p is a proposition.

$$\neg\neg p \quad \textbf{iff} \quad p \tag{2}$$

Derived Notation

$$
\begin{array}{rcl}
\phi \wedge \phi & \textbf{iff} & \neg(\neg\phi \vee \neg\phi) \\
\phi \rightarrow \phi & \textbf{iff} & \neg\phi \vee \phi \\
\phi \leftrightarrow \phi & \textbf{iff} & \phi \vee \phi
\end{array} \tag{3}
$$

NAME	IN LANGUAGE	NOTATION
negation	not A	$\neg A$
conjunction	A and B	$A \wedge B$
disjunction	A or B	$A \vee B$
implication	if A then B	$A \rightarrow B$
biconditional	B if and only if A	$A \leftrightarrow B$

Figure 1: Description of notation

Recap

1.2 Derived Definitions

Literal A literal is a propositional constant or variable or its negation.

Elementary disjunction or conjunction Is a disjunction or conjunction of one or more literals.

1.3 Forms

Every propositional formula is equivalent to a disjunctive normal form and to a conjunctive normal form

Conjunctive normal form (CNF) Is a logical propsitional formula where we conjunct one or more elementary disjunctions. And's outside and Or's inside.

Examples

- $(p \vee q) \wedge (\neg p \vee q)$

Disjunctive normal form (DNF) Same as CNF but have Or's outside and And's inside.

Examples

- $(p \wedge q) \vee (\neg p \wedge q)$

Algorithm to transform into CNF /DNF

- Eliminate all occurrences of $\leftrightarrow$ and $\rightarrow$ using the equivalence from 3.
- Transform to negation normal form by using the relevant equivalences.

1.4 Deduction Systems

2 First-order Logic

3 Semantic Tableaux for first-order logic

Quantifier rules for Semantic Tableaux

$(\forall$ T)	$\forall x A(x) : \mathrm{T}$ $\downarrow$ $A(t/x) : \mathrm{T}$ for any term t occurring on this branch and free for x in A	$(\exists$ T)*	$\exists x A(x) : \mathrm{T}$ $\downarrow$ $A(c/x) : \mathrm{T}$ for a *new* constant symbol c not yet occurring on this branch
$(\exists$ F)	$\exists x A(x) : \mathrm{F}$ $\downarrow$ $A(t/x) : \mathrm{F}$ for any term t occurring on this branch and free for x in A	$(\forall$ F)*	$\forall x A(x) : \mathrm{F}$ $\downarrow$ $A(c/x) : \mathrm{F}$ for a *new* constant symbol c not yet occurring on this branch

(*) *This rule may only be applied once for the given formula on each branch*

Here are some examples of a semantic tableaux with first-order logic [Gorb, p. 46].

Example 25 *Using Semantic Tableaux, check if $\forall x(Q \to P(x)) \vDash Q \to \forall x P(x)$, where x does not occur free in Q.*

$$
\begin{array}{c}
\forall x(Q \to P(x)) : \mathtt{T}, Q \to \forall x P(x) : \mathtt{F} \\
\downarrow \\
\forall x(Q \to P(x)) : \mathtt{T}, Q : \mathtt{T}, \forall x P(x) : \mathtt{F} \\
\downarrow \\
P(c) : \mathtt{F} \\
\downarrow \\
Q \to P(c) : \mathtt{T} \\
\swarrow \qquad \searrow \\
Q : \mathtt{F} \quad P(c) : \mathtt{T} \\
\times \qquad\quad \times
\end{array}
$$

The tableau above closes, implying that $\forall x(Q \to P(x)) \vdash_{\mathbf{ST}} Q \to \forall x P(x)$, hence $\forall x(Q \to P(x)) \vDash Q \to \forall x P(x)$ holds.

Example 26 *Using Semantic Tableaux, check if $\exists x(A(x) \to B(x)), \exists x A(x) \models \exists x B(x)$.*

$$
\begin{array}{c}
\exists x(A(x) \to B(x)) : \mathtt{T}, \exists x A(x) : \mathtt{T}, \exists x B(x) : \mathtt{F} \\
\downarrow \\
A(c_1) \to B(c_1) : \mathtt{T} \\
\downarrow \\
A(c_2) : \mathtt{T} \\
\downarrow \\
B(c_1) : \mathtt{F} \\
\downarrow \\
B(c_2) : \mathtt{F} \\
\swarrow \qquad \searrow \\
A(c_1) : \mathtt{F} \quad B(c_1) : \mathtt{T} \\
\bigcirc \qquad\qquad \times
\end{array}
$$

4 Prenex and clausal normal forms

4.1 Negating first-order formulae. Negation normal form

Negation normal form[Gorb, p.58] means that negations only occurs in front of atomic formulae. eg:

"For every car, there is a driver who, if (s)he can start it, then (s)he can stop it."

can be formalized in a suitable language as

$$\forall x(\mathsf{Car}(x) \to \exists y(\mathsf{Driver}(y) \wedge (\mathsf{Start}(x,y) \to \mathsf{Stop}(x,y)))).$$

Now, we negate:

$$
\begin{aligned}
&\neg\forall x(\mathsf{Car}(x) \to \exists y(\mathsf{Driver}(y) \wedge (\mathsf{Start}(x,y) \to \mathsf{Stop}(x,y)))) \\
&\equiv \exists x\neg(\mathsf{Car}(x) \to \exists y(\mathsf{Driver}(y) \wedge (\mathsf{Start}(x,y) \to \mathsf{Stop}(x,y)))) \\
&\equiv \exists x(\mathsf{Car}(x) \wedge \neg\exists y(\mathsf{Driver}(y) \wedge (\mathsf{Start}(x,y) \to \mathsf{Stop}(x,y)))) \\
&\equiv \exists x(\mathsf{Car}(x) \wedge \forall y\neg(\mathsf{Driver}(y) \wedge (\mathsf{Start}(x,y) \to \mathsf{Stop}(x,y)))) \\
&\equiv \exists x(\mathsf{Car}(x) \wedge \forall y(\neg\mathsf{Driver}(y) \vee \neg(\mathsf{Start}(x,y) \to \mathsf{Stop}(x,y)))) \\
&\equiv \exists x(\mathsf{Car}(x) \wedge \forall y(\neg\mathsf{Driver}(y) \vee (\mathsf{Start}(x,y) \wedge \neg\mathsf{Stop}(x,y)))).
\end{aligned}
$$

Since $\neg A \vee B \equiv A \to B$, the last formula is equivalent to

$$\exists x(\mathsf{Car}(x) \wedge \forall y(\mathsf{Driver}(y) \to (\mathsf{Start}(x,y) \wedge \neg\mathsf{Stop}(x,y)))).$$

Thus, the negation of the sentence above is equivalent to:

"There is a car such that every driver can start it and cannot stop it."

4.2 Prenex normal forms

Prenex normal forms [Gorb, p. 59] refer to a form of the formula where all the quantifications have been moved to the outermost scope like:

$$Q_1 x_1 .. Q_n x_n A$$

where Q_n is all the quantifiers called the **prefix** and A is the formula called the **matrix**

If the formula itself is in either CNF or DNF then we refer to the formula as a *prenex CNF / PCNF* or *prenex DNF / PDNF*

Algorithm for construction these normal forms:

1. Eliminate all occurences of $\to$ and $\leftrightarrow$ as in the propositional case.

2. Import all negations inside all other logical connectives and transform the formula to negation normal form.

3. Pull all quantifiers in front and thus transform the formula into a prenex form.

   For that use the equivalences:

   (a) $\forall x P \land \forall x Q \equiv \forall x (P \land Q)$

   (b) $\exists x P \lor \exists x Q \equiv \exists (P \lor Q)$

   to pull some quantifiers outwards and, after renaming the formula *wherever necessary*.

   Then, use also the following equivalences, where x does not occur free in Q, until the formula is transformed to a prenex form:

   (c) $\forall x P \land Q \equiv Q \land \forall x P \equiv \forall x (P \land Q)$

   (d) $\forall x P \lor Q \equiv Q \lor \forall x P \equiv \forall x (P \lor Q)$

   (e) $\exists x P \lor Q \equiv Q \lor \exists x P \equiv \exists x (P \lor Q)$

   (f) $\exists x P \land Q \equiv Q \land \exists x P \equiv \exists x (P \land Q)$

4. Finally, transform the matrix in a DNF or CNF, just like a propositional formula.

Here are some examples:

Example 35 *Let* $A = \exists z(\exists x Q(x,z) \vee \exists x P(x)) \rightarrow \neg(\neg\exists x P(x) \wedge \forall x \exists z Q(z,x))$.

1. *Eliminating* $\rightarrow$: $A \equiv \neg\exists z(\exists x Q(x,z) \vee \exists x P(x)) \vee \neg(\neg\exists x P(x) \wedge \forall x \exists z Q(z,x))$
2. *Importing the negation:* $A \equiv \forall z(\neg\exists x Q(x,z) \wedge \neg\exists x P(x)) \vee (\neg\neg\exists x P(x) \vee \neg\forall x \exists z Q(z,x))$
   $\equiv \forall z(\forall x \neg Q(x,z) \wedge \forall x \neg P(x)) \vee (\exists x P(x) \vee \exists x \forall z \neg Q(z,x))$.
3. *Using the equivalences (a) and (b):* $A \equiv \forall z \forall x(\neg Q(x,z) \wedge \neg P(x)) \vee \exists x(P(x) \vee \forall z \neg Q(z,x))$.
4. *Renaming:* $A \equiv \forall z \forall x(\neg Q(x,z) \wedge \neg P(x)) \vee \exists y(P(y) \vee \forall w \neg Q(w,y))$.
5. *Using the equivalences (c)-(f) and pulling the quantifiers in front:*
   $A \equiv \forall z \forall x \exists y \forall w((\neg Q(x,z) \wedge \neg P(x)) \vee P(y) \vee \neg Q(w,y))$.
6. *The resulting formula is in a prenex DNF. For a prenex CNF we have to distribute the* $\vee$ *over* $\wedge$*:*
   $A \equiv \forall z \forall x \exists y \forall w((\neg Q(x,z) \vee P(y) \vee \neg Q(w,y)) \wedge (\neg P(x) \vee P(y) \vee \neg Q(w,y)))$.

4.3 Skolemization

Skolemization[Gorb, p. 60] is a procedure used to eliminate the existential quantifiers in a first-order formula in a prenex form by a uniform replacement of all occurrences of existentially quantified individual variables with terms headed by new functional symbols, called *Skolem functions*.

Skolem functions take as arguments all variables (if any) which are bound by universal quantifiers in the scope of which the given existential quantifier sits. In particular, existentially quan- tified variables not in the scope of any universal quantifiers are replaced by constant symbols, called *Skolem constants*.

Examples of skolemization

Example 36

1. *The result of Skolemization of the formula*

$$\exists x \forall y \forall z (P(x,y) \rightarrow Q(x,z))$$

is

$$\forall y \forall z (P(c,y) \rightarrow Q(c,z)),$$

where c is a new constant symbol, called ***Skolem constant****.*

2. *More generally, the result of Skolemization of the formula*

$$\exists x_1 \cdots \exists x_k \forall y_1 \cdots \forall y_n A(x_1, \ldots, x_k, y_1, \ldots, y_n)$$

is

$$\forall y_1 \cdots \forall y_n A(c_1, \ldots, c_k, y_1, \ldots, y_n),$$

where $c_1, \ldots, c_k$ are new Skolem constants.

Note that the resulting formula is not equivalent *to the original one, but is* equally satisfiable with it.

3. *The result of Skolemization of the formula*

$$\exists x \forall y \exists z (P(x,y) \rightarrow Q(x,z))$$

is

$$\forall y (P(c,y) \rightarrow Q(c, f(y))),$$

where c is a new Skolem constant and f is a new unary function, called ***Skolem function****.*

4. *More generally, the result of Skolemization of the formula*

$$\forall y \exists x_1 \cdots \exists x_k \forall y_1 \cdots \forall y_n A(y, x_1, \ldots, x_k, y_1, \ldots, y_n)$$

is

$$\forall y \forall y_1 \cdots \forall y_n A(y, f_1(y), \ldots, f_k(y), y_1, \ldots, y_n),$$

where $f_1, \ldots, f_k$ are new Skolem functions.

5. *The result of Skolemization of the formula*

$$\forall x \exists y \forall z \exists u A(x, y, z, u)$$

is the formula

$$\forall x \forall z A(x, f(x), z, g(x,z)),$$

where f is a new unary Skolem function and g is a new binary Skolem function.

4.4 Clausal form

Clausal forms [Gorb, p. 62] is a set of *literals* (representing their disjunction).

All clauses are implicitly assumed to be universally quantified.

The algorithm for transforming any formula into clausal form

- Transform A into prenex CNF
- Skolemize all existential quantifiers.
- Remove all universalt quantifiers.
- Wrute the *matrix* (which is in CNF) as a set of clauses.

Example 39 *Consider the formula*

$$A = \exists z(\exists x Q(x,z) \vee \exists x P(x)) \to \neg(\neg\exists x P(x) \wedge \forall x \exists z Q(z,x)).$$

from example 35.

1. *Transforming A to a prenex CNF:*

   $$A \equiv \forall z \forall x \exists y \forall w((\neg Q(x,z) \vee P(y) \vee \neg Q(w,y)) \wedge (\neg P(x) \vee P(y) \vee \neg Q(w,y))).$$

2. *Skolemize all existential quantifiers:*
   $\mathrm{Skolem}(A) =$

   $$\forall z \forall x \forall w((\neg Q(x,z) \vee P(f(z,x)) \vee \neg Q(w,f(z,x))) \wedge (\neg P(x) \vee P(f(z,x)) \vee \neg Q(w,f(z,x)))).$$

3. *Remove all universal quantifiers and write the matrix as a set of clauses:*

   $$\mathrm{clausal}(A) = \{\neg Q(x,z), P(f(z,x)), \neg Q(w,f(z,x))\}, \{\neg P(x), P(f(z,x)), \neg Q(w,f(z,x))\}.$$

Part III
Non-Classical Logic (55%)

Part IV
Index

Index

References

[Gora] Valentin Goranko, *Lecture notes, part i: Classical propositional logic*.

[Gorb] ———, *Lecture notes, part ii: Classical propositional logic*.

[Gorc] ———, *Temporal logics of computations*.

[Pac] Eric Pacuit, *Notes on modal logic*.