Logica (I&E)

najaar 2017

`http://liacs.leidenuniv.nl/~vlietrvan1/logica/`

Rudy van Vliet
kamer 140 Snellius, tel. 071-527 2876
rvvliet(at)liacs(dot)nl

college 7, maandag 16 oktober 2017

1.5. Normal forms

Je moet een gat voor je laten vallen en er dan zelf inlopen.

1.5. Normal forms

Alternatives for deciding

$$\phi_1, \phi_2, \ldots, \phi_n \models \psi$$

1.5.1. Semantic equivalence, satisfiability and validity

Definition 1.40.

Let ϕ and ψ be formulas of propositional logic. We say that ϕ and ψ are *semantically equivalent* iff $\phi \models \psi$ and $\psi \models \phi$ hold.

In that case we write $\phi \equiv \psi$.

Further, we call ϕ valid if $\models \phi$ holds, i.e., if ϕ is a tautology.

A slide from lecture 4:

1.2.4 Provable equivalence

Definition 1.25.
Let ϕ and ψ be formulas of propositional logic.
We say that ϕ and ψ are *provably equivalent*,
if and only if the sequents $\phi \vdash \psi$ and $\psi \vdash \phi$ are valid;
Notation: $\phi \dashv\vdash \psi$

Example.

$$
\begin{aligned}
p \to q &\equiv \neg q \to \neg p \\
p \to q &\equiv \neg p \vee q \\
p \wedge q \to p &\equiv r \vee \neg r \\
p \wedge q \to r &\equiv p \to (q \to r)
\end{aligned}
$$

Lemma 1.41. Given formules $\phi_1, \phi_2, \ldots, \phi_n$ and ψ of propositional logic,

$$\phi_1, \phi_2, \ldots, \phi_n \models \psi$$

holds iff

$$\models \phi_1 \to (\phi_2 \to (\phi_3 \to (\ldots(\phi_n \to \psi)\ldots)))$$

Proof. . .

A slide from lecture 5:

Step 1:

If

$$\phi_1, \phi_2, \ldots, \phi_n \models \psi$$

is valid, then

Step 1:	$\models \phi_1 \to (\phi_2 \to (\phi_3 \to (\ldots(\phi_n \to \psi)\ldots)))$
Step 2:	$\vdash \phi_1 \to (\phi_2 \to (\phi_3 \to (\ldots(\phi_n \to \psi)\ldots)))$
Step 3:	$\phi_1, \phi_2, \ldots, \phi_n \vdash \psi$

Definition 1.42.
A literal L is either an atom p or the negation of an atom: $\neg p$.
A formula C is in *conjunctive normal form* (CNF)
if it is a conjunction of clauses,
where each clause D is a disjunction of literals.

Example.

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

$$(p \vee r) \wedge (\neg p \vee r) \wedge (p \vee \neg r)$$

Definition 1.42.
A literal L is either an atom p or the negation of an atom: $\neg p$.
A formula C is in *conjunctive normal form* (CNF)
if it is a conjunction of clauses,
where each clause D is a disjunction of literals.

$$\begin{array}{rcl} C & ::= & D \mid D \wedge C \\ D & ::= & L \mid L \vee D \\ L & ::= & p \mid \neg p \end{array}$$

Is

$$\models (\neg q \vee p \vee r) \wedge (\neg p \vee r) \wedge q$$

valid?

Lemma 1.43.
A disjunction of literals $L_1 \vee L_2 \vee \cdots \vee L_m$ is valid,
iff there are $1 \leq i, j \leq m$ such that L_i is $\neg L_j$.

Proof. . .

Lemma 1.43.
A disjunction of literals $L_1 \vee L_2 \vee \cdots \vee L_m$ is valid,
iff there are $1 \leq i, j \leq m$ such that L_i is $\neg L_j$.

Hence, a formula ϕ in CNF is valid, iff . . .

Definition 1.44.
Given a formula ϕ in propositional logic, we say that ϕ is satisfiable if it has a valuation in which it evaluates to T.

A slide from lecture 6a:

3.3. Consistentie

Semantisch consistent

Definitie 3.2. Een formuleverzameling $\Sigma = \{\phi_1, \ldots, \phi_n\}$ is (semantisch) consistent als Σ (minstens) een model heeft. We zeggen ook dat Σ vervulbaar is.

Inconsistent

Definition 1.44.
Given a formula ϕ in propositional logic, we say that ϕ is satisfiable if it has a valuation in which it evaluates to T.

Proposition 1.45.
Let ϕ be a formula of propositional logic.
Then ϕ is satisfiable iff $\neg\phi$ is not valid.

Proof. . .

Constructing CNF from truth table

Example.

p	q	r	ϕ
T	T	T	T
T	T	F	F
T	F	T	T
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	F
F	F	F	T

1.5.2. Conjunctive normal forms and validity

Transform ϕ into an equivalent formula ψ in CNF

For each input ϕ, deterministic algorithm `CNF`
(1) terminates,
(2) outputs equivalent formula ψ
(3) which is in CNF

Step 1: Eliminate implications

Use

$$\psi \to \eta \;\equiv\; \neg\psi \vee \eta$$

(recursively)

$$\texttt{IMPL\_FREE}(\; r \to (s \to (t \wedge s \to r))\;)$$

Step 2: Negation normal form

Use De Morgan:

$$\begin{aligned} \neg(\phi_1 \wedge \phi_2) &\equiv \neg\phi_1 \vee \not\phi_2 \\ \neg(\phi_1 \vee \phi_2) &\equiv \neg\phi_1 \wedge \not\phi_2 \end{aligned}$$

(recursively)

$$\texttt{NNF}(\ \neg r \vee (\neg s \vee (\neg(t \wedge s) \vee r))\ )$$

Function NNF

function $\texttt{NNF}(\phi)$
/* precondition: ϕ is implication free */
/* postcondition: $\texttt{NNF}(\phi)$ computes a NNF for ϕ */
begin function
  case
    ϕ is a literal: **return** ϕ
    ϕ is $\neg\neg\phi_1$: **return** $\texttt{NNF}(\phi_1)$
    ϕ is $\phi_1 \wedge \phi_2$: **return** $\texttt{NNF}(\phi_1) \wedge \texttt{NNF}(\phi_2)$
    ϕ is $\phi_1 \vee \phi_2$: **return** $\texttt{NNF}(\phi_1) \vee \texttt{NNF}(\phi_2)$
    ϕ is $\neg(\phi_1 \wedge \phi_2)$: **return** $\texttt{NNF}(\neg\phi_1) \vee \texttt{NNF}(\neg\phi_2)$
    ϕ is $\neg(\phi_1 \vee \phi_2)$: **return** $\texttt{NNF}(\neg\phi_1) \wedge \texttt{NNF}(\neg\phi_2)$
  end case
end function

Step 2: Negation normal form

Use De Morgan:

$$\begin{aligned} \neg(\phi_1 \wedge \phi_2) &\equiv \neg\phi_1 \vee \not\phi_2 \\ \neg(\phi_1 \vee \phi_2) &\equiv \neg\phi_1 \wedge \not\phi_2 \end{aligned}$$

(recursively)

$$\texttt{NNF}(\ \neg(\neg p \wedge q) \vee (p \wedge (\neg r \vee q))\ )$$

Step 3: CNF

$$
\begin{aligned}
\texttt{CNF}(p) &= \ldots \\
\texttt{CNF}(\neg p) &= \ldots \\
\texttt{CNF}(\phi_1 \wedge \phi_2) &= \ldots \\
\texttt{CNF}(\phi_1 \vee \phi_2) &= \ldots
\end{aligned}
$$

Step 3: CNF

$$
\begin{aligned}
\texttt{CNF}(p) &= p \\
\texttt{CNF}(\neg p) &= \neg p \\
\texttt{CNF}(\phi_1 \wedge \phi_2) &= \texttt{CNF}(\phi_1) \wedge \texttt{CNF}(\phi_2) \\
\texttt{CNF}(\phi_1 \vee \phi_2) &= \ldots
\end{aligned}
$$

Example.

$$
\begin{aligned}
\texttt{CNF}(\phi_1) &= (p \vee q \vee r) \wedge (\neg p \vee p \vee q) \\
\texttt{CNF}(\phi_2) &= (r \vee \neg q) \wedge (\neg q \vee \neg r \vee \neg p)
\end{aligned}
$$

Step 3: CNF

$$
\begin{aligned}
\texttt{CNF}(p) &= p \\
\texttt{CNF}(\neg p) &= \neg p \\
\texttt{CNF}(\phi_1 \wedge \phi_2) &= \texttt{CNF}(\phi_1) \wedge \texttt{CNF}(\phi_2) \\
\texttt{CNF}(\phi_1 \vee \phi_2) &= \texttt{DISTR}(\texttt{CNF}(\phi_1), \texttt{CNF}(\phi_2))
\end{aligned}
$$

Step 3: CNF

Use distributivity:

$$\begin{aligned}(\eta_{11} \wedge \eta_{12}) \vee \eta_2 &\equiv (\eta_{11} \vee \eta_2) \wedge (\eta_{12} \vee \eta_2) \\ \eta_1 \vee (\eta_{21} \wedge \eta_{22}) &\equiv (\eta_1 \vee \eta_{21}) \wedge (\eta_1 \vee \eta_{22})\end{aligned}$$

(recursively)

Use distributivity:

$$\begin{aligned}(\eta_{11} \wedge \eta_{12}) \vee \eta_2 &\equiv (\eta_{11} \vee \eta_2) \wedge (\eta_{12} \vee \eta_2) \\ \eta_1 \vee (\eta_{21} \wedge \eta_{22}) &\equiv (\eta_1 \vee \eta_{21}) \wedge (\eta_1 \vee \eta_{22})\end{aligned}$$

(recursively)

function DISTR(η_1, η_2)
/* precondition: η_1 and η_2 are in CNF */
/* postcondition: DISTR(η_1, η_2) computes a CNF for $\eta_1 \vee \eta_2$ */
begin function
  case
    η_1 is $\eta_{11} \wedge \eta_{12}$: **return** DISTR($\eta_{11}, \eta_2$) $\wedge$ DISTR(η_{12}, η_2)
    η_2 is $\eta_{21} \wedge \eta_{22}$: **return** DISTR($\eta_1, \eta_{21}$) $\wedge$ DISTR(η_1, η_{22})
    otherwise (= no conjunctions): **return** $\eta_1 \vee \eta_2$
  end case
end function

function CNF(ϕ)
/* precondition: ϕ implication free and in NNF */
/* postcondition: CNF(ϕ) computes an equivalent CNF for ϕ */
begin function
  case
    ϕ is a literal: **return** ϕ
    ϕ is $\phi_1 \wedge \phi_2$: **return** $\texttt{CNF}(\phi_1) \wedge \texttt{CNF}(\phi_2)$
    ϕ is $\phi_1 \vee \phi_2$: **return** $\texttt{DISTR}(\texttt{CNF}(\phi_1), \texttt{CNF}(\phi_2))$
  end case
end function

$$\texttt{CNF}(\ (p \vee \neg q) \vee (p \wedge (\neg r \vee q))\ )$$

1.5.3. Horn clauses and satisfiability

Example.

$(p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

1.5.3. Horn clauses and satisfiability

Example.

$(p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

$(p_2 \wedge p_3 \wedge p_5 \to p_{13}) \wedge (\top \to p_5) \wedge (p_5 \wedge p_{11} \to \bot)$

Not Horn formulas

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

$(p_2 \wedge p_3 \wedge p_5 \rightarrow p_{13} \wedge p_{27}) \wedge (\top \rightarrow p_5) \wedge (p_5 \wedge p_{11} \vee \bot)$

From CNF to Horn formula

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

Definition 1.46. A *Horn formula* is a formula ϕ of propositional logic that can be generated from H in this grammar:

$$\begin{aligned} H &::= C \mid C \wedge H \\ C &::= A \to P \\ A &::= P \mid P \wedge A \\ P &::= p \mid \bot \mid \top \end{aligned}$$

Deciding satisfiability for Horn formulas

function HORN(ϕ)
/* precondition: ϕ is a Horn formula */
/* postcondition: HORN(ϕ) decides the satisfiability for ϕ */
begin function
  mark all occurrences of $\top$ in ϕ
  while there is a conjunct $P_1 \wedge P_2 \wedge \cdots P_{k_i} \rightarrow P'$ of ϕ
    such that all P_j are marked but P' is not **do**
      mark P'
  end while

  if $\bot$ is marked
  then return 'unsatisfiable'
  else return 'satisfiable'
end function

Exercise 1.5: 15.

Apply algorithm HORN to each of these Horn formulas:

(a)

$$(p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$$