2.1.5 Definition (Polarity)

The *polarity* of the subformula $\phi|_p$ of ϕ at position $p \in \mathrm{pos}(\phi)$ is inductively defined by

$$
\begin{array}{rcl}
\mathrm{pol}(\phi, \epsilon) & := & 1 \\
\mathrm{pol}(\neg\phi, 1p) & := & -\mathrm{pol}(\phi, p) \\
\mathrm{pol}(\phi_1 \circ \phi_2, ip) & := & \mathrm{pol}(\phi_i, p) \quad \text{if } \circ \in \{\wedge, \vee\}, i \in \{1,2\} \\
\mathrm{pol}(\phi_1 \rightarrow \phi_2, 1p) & := & -\mathrm{pol}(\phi_1, p) \\
\mathrm{pol}(\phi_1 \rightarrow \phi_2, 2p) & := & \mathrm{pol}(\phi_2, p) \\
\mathrm{pol}(\phi_1 \leftrightarrow \phi_2, ip) & := & 0 \quad \text{if } i \in \{1,2\}
\end{array}
$$

Valuations can be nicely represented by sets or sequences of literals that do not contain complementary literals nor duplicates.

If $\mathcal{A}$ is a (partial) valuation of domain Σ then it can be represented by the set
$\{P \mid P \in \Sigma \text{ and } \mathcal{A}(P) = 1\} \cup \{\neg P \mid P \in \Sigma \text{ and } \mathcal{A}(P) = 0\}$.

Another, equivalent representation are *Herbrand* interpretations that are sets of positive literals, where all atoms not contained in an Herbrand interpretation are false. If $\mathcal{A}$ is a total valuation of domain Σ then it corresponds to the Herbrand interpretation
$\{P \mid P \in \Sigma \text{ and } \mathcal{A}(P) = 1\}$.

2.2.4 Theorem (Deduction Theorem)

$\phi \models \psi$ iff $\models \phi \rightarrow \psi$

2.2.6 Lemma (Formula Replacement)

Let ϕ be a propositional formula containing a subformula ψ at position p, i.e., $\phi|_p = \psi$. Furthermore, assume $\models \psi \leftrightarrow \chi$. Then $\models \phi \leftrightarrow \phi[\chi]_p$.

Normal Forms

Definition (CNF, DNF)

A formula is in *conjunctive normal form (CNF)* or *clause normal form* if it is a conjunction of disjunctions of literals, or in other words, a conjunction of clauses.

A formula is in *disjunctive normal form (DNF)*, if it is a disjunction of conjunctions of literals.

Checking the validity of CNF formulas or the unsatisfiability of DNF formulas is easy:

(i) a formula in CNF is valid, if and only if each of its disjunctions contains a pair of complementary literals P and $\neg P$,

(ii) conversely, a formula in DNF is unsatisfiable, if and only if each of its conjunctions contains a pair of complementary literals P and $\neg P$

Basic CNF Transformation

ElimEquiv	$\chi[(\phi \leftrightarrow \psi)]_p \Rightarrow_{\mathrm{BCNF}} \chi[(\phi \rightarrow \psi) \land (\psi \rightarrow \phi)]_p$
ElimImp	$\chi[(\phi \rightarrow \psi)]_p \Rightarrow_{\mathrm{BCNF}} \chi[(\neg\phi \lor \psi)]_p$
PushNeg1	$\chi[\neg(\phi \lor \psi)]_p \Rightarrow_{\mathrm{BCNF}} \chi[(\neg\phi \land \neg\psi)]_p$
PushNeg2	$\chi[\neg(\phi \land \psi)]_p \Rightarrow_{\mathrm{BCNF}} \chi[(\neg\phi \lor \neg\psi)]_p$
PushNeg3	$\chi[\neg\neg\phi]_p \Rightarrow_{\mathrm{BCNF}} \chi[\phi]_p$
PushDisj	$\chi[(\phi_1 \land \phi_2) \lor \psi]_p \Rightarrow_{\mathrm{BCNF}} \chi[(\phi_1 \lor \psi) \land (\phi_2 \lor \psi)]_p$
ElimTB1	$\chi[(\phi \land \top)]_p \Rightarrow_{\mathrm{BCNF}} \chi[\phi]_p$
ElimTB2	$\chi[(\phi \land \bot)]_p \Rightarrow_{\mathrm{BCNF}} \chi[\bot]_p$
ElimTB3	$\chi[(\phi \lor \top)]_p \Rightarrow_{\mathrm{BCNF}} \chi[\top]_p$
ElimTB4	$\chi[(\phi \lor \bot)]_p \Rightarrow_{\mathrm{BCNF}} \chi[\phi]_p$
ElimTB5	$\chi[\neg\bot]_p \Rightarrow_{\mathrm{BCNF}} \chi[\top]_p$
ElimTB6	$\chi[\neg\top]_p \Rightarrow_{\mathrm{BCNF}} \chi[\bot]_p$

Basic CNF Algorithm

Algorithm: 2 bcnf(ϕ)

Input : A propositional formula ϕ.

Output : A propositional formula ψ equivalent to ϕ in CNF.

whilerule *(***ElimEquiv**(ϕ)*)* **do** ;

;

whilerule *(***ElimImp**(ϕ)*)* **do** ;

;

whilerule *(***ElimTB1**(ϕ),...,**ElimTB6**(ϕ)*)* **do** ;

;

whilerule *(***PushNeg1**(ϕ),...,**PushNeg3**(ϕ)*)* **do** ;

;

whilerule *(***PushDisj**(ϕ)*)* **do** ;

;

return ϕ;

Advanced CNF Algorithm

For the formula

$$P_1 \leftrightarrow (P_2 \leftrightarrow (P_3 \leftrightarrow (\ldots(P_{n-1} \leftrightarrow P_n)\ldots)))$$

the basic CNF algorithm generates a CNF with 2^{n-1} clauses.

2.5.4 Proposition (Renaming Variables)

Let P be a propositional variable not occurring in $\psi[\phi]_p$.

1. If $\mathrm{pol}(\psi, p) = 1$, then $\psi[\phi]_p$ is satisfiable if and only if $\psi[P]_p \wedge (P \rightarrow \phi)$ is satisfiable.
2. If $\mathrm{pol}(\psi, p) = -1$, then $\psi[\phi]_p$ is satisfiable if and only if $\psi[P]_p \wedge (\phi \rightarrow P)$ is satisfiable.
3. If $\mathrm{pol}(\psi, p) = 0$, then $\psi[\phi]_p$ is satisfiable if and only if $\psi[P]_p \wedge (P \leftrightarrow \phi)$ is satisfiable.

Renaming

SimpleRenaming $\phi \Rightarrow_{\text{SimpRen}} \phi[P_1]_{p_1}[P_2]_{p_2}\dots[P_n]_{p_n} \land \operatorname{def}(\phi, p_1, P_1) \land \dots \land \operatorname{def}(\phi[P_1]_{p_1}[P_2]_{p_2}\dots[P_{n-1}]_{p_{n-1}}, p_n, P_n)$

provided $\{p_1, \dots, p_n\} \subset \operatorname{pos}(\phi)$ and for all $i, i+j$ either $p_i \parallel p_{i+j}$ or $p_i > p_{i+j}$ and the P_i are different and new to ϕ

Simple choice: choose $\{p_1, \dots, p_n\}$ to be all non-literal and non-negation positions of ϕ.

Renaming Definition

$$\text{def}(\psi, p, P) := \begin{cases} (P \rightarrow \psi|_p) & \text{if } \text{pol}(\psi, p) = 1 \\ (\psi|_p \rightarrow P) & \text{if } \text{pol}(\psi, p) = -1 \\ (P \leftrightarrow \psi|_p) & \text{if } \text{pol}(\psi, p) = 0 \end{cases}$$

Obvious Positions

A smaller set of positions from ϕ, called *obvious positions*, is still preventing the explosion and given by the rules:

(i) p is an obvious position if $\phi|_p$ is an equivalence and there is a position $q < p$ such that $\phi|_q$ is either an equivalence or disjunctive in ϕ or

(ii) pq is an obvious position if $\phi|_{pq}$ is a conjunctive formula in ϕ, $\phi|_p$ is a disjunctive formula in ϕ, $q \neq \epsilon$, and for all positions r with $p < r < pq$ the formula $\phi|_r$ is not a conjunctive formula.

A formula $\phi|_p$ is conjunctive in ϕ if $\phi|_p$ is a conjunction and $\mathrm{pol}(\phi, p) \in \{0, 1\}$ or $\phi|_p$ is a disjunction or implication and $\mathrm{pol}(\phi, p) \in \{0, -1\}$.

Analogously, a formula $\phi|_p$ is disjunctive in ϕ if $\phi|_p$ is a disjunction or implication and $\mathrm{pol}(\phi, p) \in \{0, 1\}$ or $\phi|_p$ is a conjunction and $\mathrm{pol}(\phi, p) \in \{0, -1\}$.

Polarity Dependent Equivalence Elimination

ElimEquiv1 $\chi[(\phi \leftrightarrow \psi)]_p \Rightarrow_{\text{ACNF}} \chi[(\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)]_p$
provided $\text{pol}(\chi, p) \in \{0, 1\}$

ElimEquiv2 $\chi[(\phi \leftrightarrow \psi)]_p \Rightarrow_{\text{ACNF}} \chi[(\phi \wedge \psi) \vee (\neg\phi \wedge \neg\psi)]_p$
provided $\text{pol}(\chi, p) = -1$

Extra $\top, \bot$ Elimination Rules

ElimTB7	$\chi[\phi \rightarrow \bot]_p$	$\Rightarrow_{\text{ACNF}}$	$\chi[\neg\phi]_p$
ElimTB8	$\chi[\bot \rightarrow \phi]_p$	$\Rightarrow_{\text{ACNF}}$	$\chi[\top]_p$
ElimTB9	$\chi[\phi \rightarrow \top]_p$	$\Rightarrow_{\text{ACNF}}$	$\chi[\top]_p$
ElimTB10	$\chi[\top \rightarrow \phi]_p$	$\Rightarrow_{\text{ACNF}}$	$\chi[\phi]_p$
ElimTB11	$\chi[\phi \leftrightarrow \bot]_p$	$\Rightarrow_{\text{ACNF}}$	$\chi[\neg\phi]_p$
ElimTB12	$\chi[\phi \leftrightarrow \top]_p$	$\Rightarrow_{\text{ACNF}}$	$\chi[\phi]_p$

where the two rules ElimTB11, ElimTB12 for equivalences are applied with respect to commutativity of $\leftrightarrow$.

Advanced CNF Algorithm

Algorithm: 3 acnf(ϕ)

Input : A formula ϕ.

Output : A formula ψ in CNF satisfiability preserving to ϕ.

```
whilerule (ElimTB1(ϕ),...,ElimTB12(ϕ)) do ;
;
SimpleRenaming(ϕ) on obvious positions;
whilerule (ElimEquiv1(ϕ),ElimEquiv2(ϕ)) do ;
;
whilerule (ElimImp(ϕ)) do ;
;
whilerule (PushNeg1(ϕ),...,PushNeg3(ϕ)) do ;
;
whilerule (PushDisj(ϕ)) do ;
;
return ϕ;
```

Propositional Resolution

The propositional resolution calculus operates on a set of clauses and tests unsatisfiability.

Recall that for clauses I switch between the notation as a disjunction, e.g., $P \lor Q \lor P \lor \neg R$, and the multiset notation, e.g., $\{P, Q, P, \neg R\}$. This makes no difference as we consider $\lor$ in the context of clauses always modulo AC. Note that $\bot$, the empty disjunction, corresponds to $\emptyset$, the empty multiset. Clauses are typically denoted by letters C, D, possibly with subscript.

Resolution Inference Rules

Resolution $(N \uplus \{C_1 \vee P, C_2 \vee \neg P\}) \Rightarrow_{\mathrm{RES}}$
$(N \cup \{C_1 \vee P, C_2 \vee \neg P\} \cup \{C_1 \vee C_2\})$

Factoring $(N \uplus \{C \vee L \vee L\}) \Rightarrow_{\mathrm{RES}}$
$(N \cup \{C \vee L \vee L\} \cup \{C \vee L\})$

2.6.1 Theorem (Soundness & Completeness)

The resolution calculus is sound and complete:

$$N \text{ is unsatisfiable iff } N \Rightarrow^{*}_{\mathrm{RES}} N' \text{ and } \perp \in N' \text{ for some } N'$$

Resolution Reduction Rules

Subsumption $(N \uplus \{C_1, C_2\}) \Rightarrow_{\text{RES}} (N \cup \{C_1\})$
provided $C_1 \subset C_2$

Tautology Deletion $(N \uplus \{C \lor P \lor \neg P\}) \Rightarrow_{\text{RES}} (N)$

Condensation $(N \uplus \{C_1 \lor L \lor L\}) \Rightarrow_{\text{RES}} (N \cup \{C_1 \lor L\})$

Subsumption Resolution $(N \uplus \{C_1 \lor L, C_2 \lor \text{comp}(L)\}) \Rightarrow_{\text{RES}}$
$(N \cup \{C_1 \lor L, C_2\})$
where $C_1 \subseteq C_2$

2.6.6 Theorem (Resolution Termination)

If reduction rules are preferred over inference rules and no inference rule is applied twice to the same clause(s), then $\Rightarrow^{+}_{\mathrm{RES}}$ is well-founded.

The Overall Picture

Application System $+$ Problem
System Algorithm $+$ Implementation
Algorithm Calculus $+$ Strategy
Calculus Logic $+$ States $+$ Rules
Logic Syntax $+$ Semantics