Everything You Always Wanted to Know About
First-Order CNF and Compactness
(And Were Not Afraid to Ask)

First-Order CNF Transformation

Basically, same procedure, same complications as for propositional logic, but we have to take care of variables and quantifiers.

Extending the Notion of a Position

$$
\begin{aligned}
\mathrm{pos}(x) &:= \{\epsilon\} \text{ if } x \in \mathcal{X} \\
\mathrm{pos}(\phi) &:= \{\epsilon\} \text{ if } \phi \in \{\top, \bot\} \\
\mathrm{pos}(\neg\phi) &:= \{\epsilon\} \cup \{1p \mid p \in \mathrm{pos}(\phi)\} \\
\mathrm{pos}(\phi \circ \psi) &:= \{\epsilon\} \cup \{1p \mid p \in \mathrm{pos}(\phi)\} \cup \{2p \mid p \in \mathrm{pos}(\psi)\} \\
\mathrm{pos}(s \approx t) &:= \{\epsilon\} \cup \{1p \mid p \in \mathrm{pos}(s)\} \cup \{2p \mid p \in \mathrm{pos}(t)\} \\
\mathrm{pos}(f(t_1, \ldots, t_n)) &:= \{\epsilon\} \cup \bigcup_{i=1}^{n} \{ip \mid p \in pos(t_i)\} \\
\mathrm{pos}(P(t_1, \ldots, t_n)) &:= \{\epsilon\} \cup \bigcup_{i=1}^{n} \{ip \mid p \in pos(t_i)\} \\
\mathrm{pos}(\forall x.\phi) &:= \{\epsilon\} \cup \{1p \mid p \in \mathrm{pos}(\phi)\} \\
\mathrm{pos}(\exists x.\phi) &:= \{\epsilon\} \cup \{1p \mid p \in \mathrm{pos}(\phi)\}
\end{aligned}
$$

Free, Bound, All Variables

The set of *all* variables occurring in a term t (formula ϕ) is denoted by $\mathsf{vars}(t)$ ($\mathsf{vars}(\phi)$) and formally defined as

$$\mathsf{vars}(t) := \{x \in \mathcal{X} \mid x = t|_p, p \in \mathsf{pos}(t)\}$$

for terms and for formulas

$$\mathsf{vars}(\phi) := \{x \in \mathcal{X} \mid x = t|_p, p \in \mathsf{pos}(\phi)\}.$$

The set of *free* variables of a formula ϕ (term t) is given by $\mathsf{fvars}(\phi, \emptyset)$ ($\mathsf{fvars}(t, \emptyset)$) and recursively defined by

$$\begin{aligned}
\mathsf{fvars}(\psi_1 \circ \psi_2, B) &:= \mathsf{fvars}(\psi_1, B) \cup \mathsf{fvars}(\psi_2, B) \\
&\quad \text{where } \circ \in \{\wedge, \vee, \rightarrow, \leftrightarrow\} \\
\mathsf{fvars}(\forall x.\psi, B) &:= \mathsf{fvars}(\psi, B \cup \{x\}) \\
\mathsf{fvars}(\exists x.\psi, B) &:= \mathsf{fvars}(\psi, B \cup \{x\}) \\
\mathsf{fvars}(\neg\psi, B) &:= \mathsf{fvars}(\psi, B) \\
\mathsf{fvars}(L, B) &:= \mathsf{vars}(L) \setminus B \quad \text{for a literal } L \\
\mathsf{fvars}(t, B) &:= \mathsf{vars}(t) \setminus B \quad \text{for a term } t
\end{aligned}$$

For $\mathsf{fvars}(\phi, \emptyset)$ I also write $\mathsf{fvars}(\phi)$.

The set of *bound* variables is defined exactly the same except for the literal (term) case: $\mathsf{bvars}(L, B) := B$ ($\mathsf{bvars}(t, B) := B$).

Following the propositional procedure, elimintation of $\top, \bot$ and negation removal is for quantifiers as follows:

ElimTB13 $\chi[\{\forall, \exists\}x.\top]_p \Rightarrow_{\mathrm{ACNF}} \chi[\top]_p$

ElimTB14 $\chi[\{\forall, \exists\}x.\bot]_p \Rightarrow_{\mathrm{ACNF}} \chi[\bot]_p$

PushNeg4 $\chi[\neg\forall x.\phi]_p \Rightarrow_{\mathrm{ACNF}} \chi[\exists x.\neg\phi]_p$

PushNeg5 $\chi[\neg\exists x.\phi]_p \Rightarrow_{\mathrm{ACNF}} \chi[\forall x.\neg\phi]_p$

Generalizing Renaming

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

SimpleRenaming $\phi \Rightarrow_{\mathrm{ACNF}} \phi[P(\vec{x_n})]_p \wedge \mathrm{def}(\phi, p, P(\vec{x_n}))$
provided $p \in \mathsf{pos}(\phi)$ and $\mathsf{fvars}(\phi|_p) = \{x_1, \ldots, x_n\}$ and P is fresh to ϕ

Due to quantifier bindings, application of a substitution σ to a formula is more complicated.

$$\bot\sigma := \bot \qquad \top\sigma := \top$$

$$\begin{aligned}
(f(t_1,\dots,t_n))\sigma &:= f(t_1\sigma,\dots,t_n\sigma)\\
(P(t_1,\dots,t_n))\sigma &:= P(t_1\sigma,\dots,t_n\sigma)\\
(s \approx t)\sigma &:= (s\sigma \approx t\sigma)\\
(\neg\phi)\sigma &:= \neg(\phi\sigma)\\
(\phi \circ \psi)\sigma &:= \phi\sigma \circ \psi\sigma\\
&\quad\ \text{where } \circ \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}\\
(Qx.\phi)\sigma &:= Qz.(\phi\sigma[x \mapsto z])\\
&\quad\ Q \in \{\forall, \exists\}, z \text{ is a fresh variable}
\end{aligned}$$

RenVar $\quad \phi \ \Rightarrow_{\text{ACNF}} \ \phi\sigma \qquad \sigma = \{\}$

For Skolemization (next slide) mini scoping is important and I assume that explicit or implicit negations are moved inwards to the literal level.

MiniScope1 $\chi[\forall x.(\psi_1 \circ \psi_2)]_p \Rightarrow_{\text{ACNF}} \chi[(\forall x.\psi_1) \circ \psi_2]_p$
provided $\circ \in \{\wedge, \vee\}$, $x \notin \text{fvars}(\psi_2)$

MiniScope2 $\chi[\exists x.(\psi_1 \circ \psi_2)]_p \Rightarrow_{\text{ACNF}} \chi[(\exists x.\psi_1) \circ \psi_2]_p$
provided $\circ \in \{\wedge, \vee\}$, $x \notin \text{fvars}(\psi_2)$

MiniScope3 $\chi[\forall x.(\psi_1 \wedge \psi_2)]_p \Rightarrow_{\text{ACNF}} \chi[(\forall x.\psi_1) \wedge (\forall x.\psi_2)\sigma]_p$
where $\sigma = \{\}$, $x \in (\text{fvars}(\psi_1) \cap \text{fvars}(\psi_2))$

MiniScope4 $\chi[\exists x.(\psi_1 \vee \psi_2)]_p \Rightarrow_{\text{ACNF}} \chi[(\exists x.\psi_1) \vee (\exists x.\psi_2)\sigma]_p$
where $\sigma = \{\}, x \in (\text{fvars}(\psi_1) \cap \text{fvars}(\psi_2))$

For Skolemization I assume that explicit or implicit negations are moved inwards to the literal level.

Skolemization $\quad \chi[\exists x.\phi]_p \;\Rightarrow_{\text{ACNF}}\; \chi[\phi\{x \mapsto f(y_1, \ldots, y_n)\}]_p$
provided there is no position q, $q < p$ with $\chi|_q = \exists z.\psi$, $\text{fvars}(\exists x.\phi) = \{y_1, \ldots, y_n\}$, $f : \text{sort}(y_1) \times \ldots \times \text{sort}(y_n) \to \text{sort}(x)$ is a fresh function symbol

Finally universal quantifiers are dropped

RemForall $\chi[\forall x.\psi]_p \Rightarrow_{\mathrm{ACNF}} \chi[\psi]_p$

and then the actual CNF is then done by distributivity, exactly as it is done in propositional logic.

Algorithm 2: acnf(ϕ)

Input : A first-order formula ϕ.
Output : A formula ψ in CNF satisfiability preserving to ϕ.

```
whilerule (ElimTB1(ϕ),...,ElimTB14(ϕ)) do ;
RenVar(ϕ);
SimpleRenaming(ϕ) on obvious positions;
whilerule (ElimEquiv1(ϕ),ElimEquiv2(ϕ)) do ;
whilerule (ElimImp(ϕ)) do ;
whilerule (PushNeg1(ϕ),...,PushNeg5(ϕ)) do ;
whilerule (MiniScope1(ϕ),...,MiniScope4(ϕ)) do ;
whilerule (Skolemization(ϕ)) do ;
whilerule (RemForall(ϕ)) do ;
whilerule (PushDisj(ϕ)) do ;
return ϕ
```

Superposition Saturation Formally

Definition (Inferences & Redundancy)

$$\mathrm{Red}(N) := \{C \mid N^{\prec C} \models C\}$$
$$\mathrm{Sup}(N) := N \cup \{C \mid N \Rightarrow_{SUP} N \cup \{C\}\}$$

Definition (Saturation)

N is called *saturated up to redundancy* if

$$\mathrm{Sup}(N \setminus \mathrm{Red}(N)) \subseteq N \cup \mathrm{Red}(N)$$

Theorem (Superposition Completeness)

Let N be saturated up to redundancy, then $N \models \bot$ iff $\bot \in N$.

Proof.

Follows from Theorem 3.13.9. $\square$

Computing Saturated Sets

Definition (Run)

A *run* of the superposition calculus is a sequence $N_0 \vdash N_1 \vdash N_2 \vdash \ldots$ such that

1. if $C \in (N_{i+1} \setminus N_i)$ then $N_i \Rightarrow_{SUP} N_i \cup \{C\}$
2. if $C \in (N_i \setminus N_{i+1})$ then $C \in \mathrm{Red}(N_i)$

For a run define

$$N^\infty = \bigcup_{i \geq 0} N_i \qquad \text{and} \qquad N^* = \bigcup_{i \geq 0} \bigcap_{j \geq i} N_j$$

where N^* is called the set of *persistent clauses* of the run.

Definition (Fair Run)

A run is called *fair*, if for every $C \in \mathrm{Sup}(N^* \setminus \mathrm{Red}(N^*))$ there is some i with $C \in (N_i \cup \mathrm{Red}(N_i))$.

Theorem (Dynamic Completeness)

Let N^* be the limit of a fair run $N_0 \vdash N_1 \vdash \ldots$. Then N_0 is satisfiable iff $\bot \notin N^*$.

Proof.

$\Rightarrow$: Obvious because $N_0 \models N^*$.
$\Leftarrow$: By fairness, N^* is saturated up to redundancy. If $\bot \notin N^*$ then $(\mathrm{grd}(\Sigma, N^*))_{\mathcal{I}} \models N^*$. For every clause $C \in N_0$ either $C \in N^*$ or $C \in \mathrm{Red}(N^*)$, therefore $(\mathrm{grd}(\Sigma, N^*))_{\mathcal{I}} \models N_0$. $\square$

Now if $\bot \in N^*$ then $\bot \in N_i$ for some minimal i and therefore there is a subset $N' \subseteq N_0$ with $N' \models \bot$.

Practically, fairness can be guaranteed by always considering minimal clauses of with respect to a well-founded ordering $\lhd$ such that for any clause $C \in N$ there are only finitely many clauses D with $D \lhd C$. Counting the number of symbols in a clause together with $<$ is an example for such an ordering.