# 1 Introduction to Predicate Resolution

The resolution proof system for Predicate Logic operates, as in propositional case on sets of **clauses** and uses a **resolution rule** as the only rule of inference.

The **first goal** of this part is to define an effective **process of transformation** of any formula $A$ of a predicate language $\mathcal{L} = \mathcal{L}_{\{\neg,\cup,\cap,\Rightarrow\}}(\mathbf{P},\mathbf{F},\mathbf{C})$ into its logically equivalent set of clauses $\mathbf{C}_A$.

The **second goal** of this part is extend the definition of the propositional **resolution rule** to the case of predicate language.

**Observe** that define, as in propositional case, a **clause** as a finite set of **literals** and we define a **literal** as an **atomic** formula or a **negation** of an **atomic** formula. The **difference** with propositional resolution is in the **language**, i.e. what is a **predicate atomic** formula as opposed to **propositional atomic formula**. **Reminder:**
An **atomic formula** of a **predicate language** $\mathcal{L}(\mathbf{P},\mathbf{F},\mathbf{C})$ is any element of $\mathcal{A}^*$ (finite strings over the alphabet of $\mathcal{L}$) of the form

$$R(t_1, t_2, ..., t_n)$$

where $R \in \mathbf{P}, \#R = n$ and $t_1, t_2, ..., t_n \in \mathbf{T}$.

The set of all **atomic formulas** is denoted by $A\mathcal{F}$ and is defined as

$$A\mathcal{F} = \{R(t_1, t_2, ..., t_n) \in \mathcal{A}^* : \quad R \in \mathbf{P},\ t_1, t_2, ..., t_n \in \mathbf{T},\ n \geq 1\}$$

We use symbols $R, Q, P, ...$ with indices if necessary to **denote** the atomic formulas and we define formally the set $\mathbf{L}$ of all **literals** of $\mathcal{L}(\mathbf{P},\mathbf{F},\mathbf{C})$ as follows.

$$\mathbf{L} = \{R : \ R \in A\mathcal{F}\} \cup \{\neg R : \ R \in A\mathcal{F}\}.$$

**Reminder:**
A formula of a **predicate language** is an **open formula** if it does not contain any quantifiers, i.e. it is a formula build out of **atomic formulas** and **propositional connectives** only.

Transforming any formula $A$ of a predicate language into a set $\mathbf{C}_A$ of clauses means that we can **represent** the formula $A$ as a certain collection of atomic formulas and negations of atomic formulas, i.e. as a certain collection of **open formulas**.

In order to achieve our first **first goal** we **start** with transformation of any formula $A$ of a predicate language into an **open formula** $A^*$ of some **larger language** such that $A \equiv A^*$. The process is described in the following 2 section.

## 2 Prenex Normal Forms

We remind the following important notion.

**Term $t$ is free for $x$ in $A(x)$.** Let $A(x) \in \mathcal{F}$ and $t$ be a term, $A(t)$ be a result of substituting $t$ for all free occurrences of $x$ in $A(x)$.

We say that $t$ **is free for** $x$ **in** $A(x)$, if no occurrence of a variable in $t$ becomes a bound occurrence in $A(t)$.

In particular, if $A(x), A(x_1, x_2, ..., x_n) \in \mathcal{F}$ and $t, t_1, t_2, ..., t_n \in \mathbf{T}$, then

$$A(x/t), \quad A(x_1/t_1, x_2/t_2, ..., x_n/t_n)$$

or, more simply just

$$A(t), \quad A(t_1, t_2, ..., t_n)$$

denotes the result of replacing all occurrences of the free variables $x$, $x_1, x_2, ..., x_n$, by the terms $t, t_1, t_2, ..., t_n$, respectively, assuming that $t, t_1, t_2, ..., t_n$ are free for $x$, $x_1, x_2, ..., x_n$, respectively, in $A$.

The assumption that $t$ **is free for** $x$ **in** $A(x)$ while substituting $t$ for $x$, is important because otherwise we would distort the meaning of $A(t)$. This is illustrated by the following example.

**Example.** Let $t = y$ and $A(x)$ be

$$\exists y(x \neq y).$$

Obviously $t$ is not free for $y$ in $A$. The substitution of $t$ for $x$ produces a formula $A(t)$ of the form

$$\exists y(y \neq y),$$

which has a different meaning than $\exists y(x \neq y)$.

Here are more examples illustrating the notion: $t$ *is free for* $x$ *in* $A(x)$.

**Example** Let $A(x)$ be a formula

$$(\forall y P(x, y) \cap Q(x, z))$$

and $t$ be a term $f(x, z)$, i.e. $t = f(x, z)$.

None of the occurrences of the variables $x, z$ of $t$ is bound in $A(t)$, hence we say that $t = f(x, z)$ is **free for** $x$ in $(\forall y P(x, y) \cap Q(x, z))$.

Substituting $t$ on a place of $x$ in $A(x)$ we obtain a formula $A(t)$ of the form

$$(\forall y P(f(x, z), y) \cap Q(f(x, z), z)).$$

**Example** Let $A(x)$ be a formula

$$(\forall y P(x, y) \cap Q(x, z))$$

The term $t = f(y, z)$ is **not free** for $x$ in $A(x)$ because substituting $t = f(y, z)$ on a place of $x$ in $A(x)$ we obtain now a formula $A(t)$ of the form

$$(\forall y P(f(y, z), y) \cap Q(f(y, z), z))$$

which contain a bound occurrence of the variable $y$ of $t$ $(\forall y P(f(y, z), y))$. The other occurrence $(Q(f(y, z), z))$ of $y$ is free, but it is not sufficient, as for term to be free for $x$, all occurrences of its variables has to be free in $A(t)$.

Another important notion we will use here is the following notion of similarity of formulas. Intuitively, we say that $A(x)$ and $A(y)$ are similar if and only if $A(x)$ and $A(y)$ are the same except that $A(x)$ has free occurrences of $x$ in exactly those places where $A(y)$ has free occurrences of $y$.

**Example**. The formulas $\exists z(P(x, z) \Rightarrow Q(x))$ and $\exists z(P(y, z) \Rightarrow Q(y))$ are similar.

The formal definition of this notion follows.

**Definition** Let $x$ and $y$ be two different variables. We say that the formulas $A(x)$ and $A(x/y)$ are **similar** and denote it by $A(x) \sim A(x/y)$ if and only if $y$ is free for $x$ in $A(x)$ and $A(x)$ **has no** free occurrences of $y$.

**Example**
The formulas $A(x)$: $\exists z(P(x, z) \Rightarrow Q(x, y))$ and $A(x/y)$: $\exists z(P(y, z) \Rightarrow Q(y, y))$ are **not similar**; $y$ is free for $x$ in $A(x)$, but the formula $A(x/y)$ **has** a free occurrence of $y$.

**Example**. The formulas $A(x)$: $\exists z(P(x,z) \Rightarrow Q(x,y))$ and $A(x/w)$: $\exists z(P(w,z) \Rightarrow Q(w,y)$ **are similar**; $w$ is free for $x$ in $A(x)$ and the formula $A(x/w)$ has no free occurrence of $w$.
Directly from the definition we get the following.

**Lemma 2.1** *For any formula $A(x) \in \mathcal{F}$, if $A(x)$ and $A(x/y)$ are similar, i.e. $A(x) \sim A(y)$, then*

$$\forall x A(x) \equiv \forall y A(y),$$

$$\exists x A(x) \equiv \exists y A(y).$$

We prove, by the induction on the number of connectives and quantifiers in a formula $A$ the following.

**Theorem 2.1 (Replacement Theorem)** *For any formulas $A, B \in \mathcal{F}$, if $B$ is a sub-formula of $A$, if $A^*$ is the result of replacing zero or more occurrences of $B$ in $A$ by a formula $C$, and $B \equiv C$, then $A \equiv A^*$.*

Directly from the above lemma and the replacement theorem we get that the following theorem holds.

**Theorem 2.2 (Change of Bound Variables)** *For any formula $A(x), A(y), B \in \mathcal{F}$, if $A(x)$ and $A(x/y)$ are similar, i.e. $A(x) \sim A(y)$, and the formula $\forall x A(x)$ or $\exists x A(x)$ is a sub-formula of $B$, and $B^*$ is the result of replacing zero or more occurrences of $A(x)$ in $B$ by a formula $\forall y A(y)$ or $\exists y A(y)$, then $B \equiv B^*$.*

**Definition 2.1 (Naming Variables Apart)** *We say that a formula $B$ has its variables* **named apart** *if no two quantifiers in $B$ bind the same variable and no bound variable is also free.*

We can now use theorem 2.2 to prove its more general version.

**Theorem 2.3 (Naming Variables Apart)** *Every formula $A \in \mathcal{F}$ is logically equivalent to one in which all variables are named apart.*

We use the above theorems plus the **equational laws** for quantifiers to prove, as a next step a so called a **Prenex Form Theorem**.
In order to do so we first we define an important notion of **prenex normal form** of a formula.

**Definition 2.2 (Closure of a Formula)** *By a* **closure** *of a formula A we mean a* **closed** *formula $A'$ obtained from A prefixing in universal quantifiers all those variables that a free in A; i.e. if $A(x_1, \ldots, x_n)$ then $A' \equiv A$ is*

$$\forall x_1 \forall x_2 .... \forall x_n A(x_1, x_2, \ldots., x_n)$$

**Example**
Let A be a formula $(P(x,y) \Rightarrow \neg\exists z\ R(x,y,z))$, its **closure** $A' \equiv A$ is $\forall x \forall y (P(x,y) \Rightarrow \neg\exists z\ R(x,y,z))$.

**Definition 2.3 (Prenex Normal Form)** *Any formula A of the form*

$$Q_1 x_1 Q_2 x_2 .... Q_n x_n B$$

*where each $Q_i$ is a universal or existential quantifier, i.e. for all $1 \leq i \leq n$, $Q_i \in \{\exists, \forall\}$, $x_i \neq x_j$ for $i \neq j$, and B* **contains no quantifiers***, is said to be in* **Prenex Normal Form (PNF)***.*
*We include the case $n = 0$ when there are no quantifiers at all.*

We assume that the formula A in **PNF** is **closed**. If it is not closed we form its **closure** (definition 2.2) instead. We prove that, for every formula, we can effectively construct a formula that is its equivalent **PNF**.

**Theorem 2.4 (PNF Theorem)** *There is an effective procedure for transforming any formula $A \in \mathcal{F}$ into a logically equivalent formula $A'$ in the* **prenex normal form***.*

We define the procedure by induction on the number $k$ of occurrences of connectives and quantifiers in $A$.
Let's consider few examples.

**Exercise**
**Find** a prenex normal form **PNF** of a formula $A$: $\forall x(P(x) \Rightarrow \exists x Q(x))$.

We find **PNF** in the following steps.

**Step 1: Rename Variables Apart**

By the theorem 2.2 we can make all bound variables in $A$ different, i.e. we transform $A$ into an equivalent formula $A'$

$$\forall x(P(x) \Rightarrow \exists y Q(y)).$$

**Step 2: Pull out Quantifiers**

We apply the equational law

$$(C \Rightarrow \exists y Q(y)) \equiv \exists y \; (C \Rightarrow Q(y))$$

to the sub-formula $B : (P(x) \Rightarrow \exists y Q(y))$ of $A'$ for $C = P(x)$, as P(x) does not contain the variable y. We get its equivalent formula $B^* : \exists y(P(x) \Rightarrow Q(y))$. We substitute now $B^*$ on place of $B$ in $A'$ and get a formula $A''$:

$$\forall x \exists y (P(x) \Rightarrow Q(y))$$

such that $A'' \equiv A' \equiv A$.

$A''$ is a required prenex normal form **PNF** for $A$

**Example**
Let's now find **PNF** for the formula $A$:

$$(\exists x \forall y \; R(x, y) \Rightarrow \forall y \exists x \; R(x, y))$$

**Step 1: Rename Variables Apart**
Take a sub- formula $B(x, y)$ : $\forall y \exists x \, R(x, y)$ of $A$, get $B(x/z, y/w) : \forall z \exists w \, R(z, w)$ and replace B(x,y) by $B(x/z, y/w)$ in $A$ and get

$$(\exists x \forall y \; R(x, y) \Rightarrow \forall z \exists w \; R(z, w))$$

**Step 2: Pull out quantifiers**
We use corresponding equational laws for quantifiers to pull out first quantifiers $\forall x \exists y$ and then quantifiers $\forall z \exists w$ and get the following **PNF** for $A$

$$\exists x \forall y \exists z \forall w \; (R(x, y) \Rightarrow \; R(z, w))$$

**Observe** we can also perform **Step 2** that by pulling first the quantifiers $\forall z \exists w$ and then quantifiers $\forall x \exists y$ and obtain **another PNF** for $A$

$$\exists z \forall w \exists x \forall y \; (R(x, y) \Rightarrow \; R(z, w))$$

# 3 Skolemization

We will show now how any formula $A$ in its prenex normal form **PNF** we can transformed it into a certain **open formula** $A^*$, such that $A \equiv A^*$.
The open formula $A^*$ belongs to a richer language then the initial language to which the formula $A$ belongs. The transformation process **adds** new constants,

called **Skolem constants**, and new function symbols, called **Skolem function symbols** to the initial language.
The whole process is called the **Skolemization** of the initial language, the such build extension of the initial language is called the **Skolem extension**.

**Elimination of Quantifiers**

Given a formula A be in its **Prenex Normal Forma PNF**

$$Q_1 x_1 Q_2 x_2 \dots Q_n x_n B(x_1, x_2, \dots x_n)$$

where each $Q_i$ is a universal or existential quantifier, i.e. for all $1 \leq i \leq n$, $Q_i \in \{\exists, \forall\}$, $x_i \neq x_j$ for $i \neq j$, and $B(x_1, x_2, \dots x_n)$ **contains no quantifiers**.

We describe now a procedure of **elimination of all quantifiers** from the formula A and hence transforming into a logically equivalent **open formula** $A^*$. We assume that A is **closed**. If it is not closed we form its **closure** instead. We considerer 3 cases.
**Case 1**
All quantifiers $Q_i$ for $1 \leq i \leq n$ are **universal**, i.e. formula A is

$$\forall x_1 \forall x_2 \dots \forall x_n B(x_1, x_2, \dots, x_n)$$

We replace the formula A by the **open formula** $A^*$:

$$B(x_1, x_2, \dots., x_n).$$

**Case 2**
All quantifiers $Q_i$ for $1 \leq i \leq n$ are **existential**, i.e. formula A is

$$\exists x_1 \exists x_2 .... \exists x_n B(x_1, x_2, \dots x_n)$$

We replace the formula A by the **open formula** $A^*$:

$$B(c_1, c_2, \dots., c_n)$$

where $c_1, c_2, \dots., c_n$ and **new** individual constants, all different, **added** to our original language $\mathcal{L}$. We call such constants added to the language **Skolem constants**

**Case 3**
The quantifiers are **mixed** . We assume that A is **closed**. If it is not closed we form its **closure** instead. We eliminate quantifiers one by one and step by step

depending on first, and consecutive quantifiers.

Given a **closed PNF** formula A

$$Q_1x_1Q_2x_2 \dots Q_nx_nB(x_1, x_2, \dots x_n)$$

**Step 1** Elimination of $Q_1x_1$
We have two possibilities for the first quantifier $Q_1x_1$, namely **P1** $Q_1x_1$ is **universal** or **P2** $Q_1x_1$ is **existential**.

Consider **P1**
First quantifier in A is universal, i. e. A is

$$\forall x_1Q_2x_2 \dots Q_nx_nB(x_1, x_2, \dots x_n)$$

We **replace** A by a formula $A_1$ :

$$Q_2x_2 \dots Q_nx_nB(x_1, x_2, \dots x_n)$$

We have **eliminated** the quantifier $Q_1$ in this case.

Consider **P2**
First quantifier in A is **existential**, i. e. A is

$$\exists x_1Q_2x_2 \dots Q_nx_nB(x_1, x_2, \dots x_n)$$

We **replace** A by a formula $A_1$ :

$$Q_2x_2 \dots Q_nx_nB(b_1, x_2, \dots x_n)$$

where $b_1$ is a **new** constant symbol **added** to our original language $\mathcal{L}$. We call such constant symbol added to the language **Skolem constant** symbol. We have **eliminated** the quantifier $Q_1$ in this case. We have covered all cases and this **ends** the **Step 1**.

**Step 2** Elimination of $Q_2x_2$.

Consider now the **PNF** formula $A_1$ from **Step1- P1**

$$Q_2x_2 \dots Q_nx_nB(x_1, x_2, \dots x_n)$$

Remark that the formula $A_1$ might not be closed.

We have again two possibilities for elimination of the quantifier $Q_2x_2$, namely **P1** $Q_2x_2$ is **universal** or **P2** $Q_2x_2$ is **existential**.

Consider **P1**
First quantifier in $A_1$ is **universal**, i.e. $A_1$ is

$$\forall x_2 Q_3x_3 \ldots Q_nx_n B(x_1, x_2, x_3, \ldots x_n)$$

We replace $A_1$ by the following $A_2$

$$Q_3x_3 \ldots Q_nx_n B(x_1, x_2, x_3, \ldots x_n)$$

We have **eliminated** the quantifier $Q_2$ in this case.

Consider **P2**
First quantifier in $A_1$ is **existential**, i.e. $A_1$ is

$$\exists x_2 Q_3x_3 \ldots Q_nx_n B(x_1, x_2, x_3, \ldots x_n)$$

Observe that now the variable $x_1$ is a **free** variable in $B(x_1, x_2, x_3, \ldots x_n)$ and hence in $A_1$.
We replace $A_1$ by the following $A_2$

$$Q_3x_3 \ldots Q_nx_n B(x_1, f(x_1), x_3, \ldots x_n)$$

where $f$ is a **new** one argument functional symbol **added** to our original language $\mathcal{L}$. We call such functional symbols added to the original language **Skolem** functional symbols.
We have **eliminated** the quantifier $Q_2$ in this case.

Consider now the **PNF** formula $A_1$ from **Step1 - P2**

$$Q_2x_2Q_3x_3 \ldots Q_nx_n B(b_1, x_2, \ldots x_n)$$

Again we have two cases.

Consider **P1**
First quantifier in $A_1$ is **universal**, i.e. $A_1$ is

$$\forall x_2 Q_3x_3 \ldots Q_nx_n B(b_1, x_2, x_3, \ldots x_n)$$

We replace $A_1$ by the following $A_2$

$$Q_3x_3 \ldots Q_nx_n B(b_1, x_2, x_3, \ldots x_n)$$

We have **eliminated the quantifier** $Q_2$ in this case.
Consider **P2**
First quantifier in $A_1$ is **existential**, i.e. $A_1$ is

$$\exists x_2 Q_3 x_3 \ldots Q_n x_n B(b_1, x_2, x_3, \ldots x_n)$$

We replace $A_1$ by $A_2$

$$Q_3 x_3 \ldots Q_n x_n B(b_1, b_2, x_3, \ldots x_n)$$

where $b_2 \neq b_1$ is a new Skolem constant symbol **added** to our original language $\mathcal{L}$.
We have **eliminated** the quantifier $Q_2$ in this case. We have covered all cases and this **ends** the **Step 2**. **Step 3** Elimination of $Q_3 x_3$

Let's now consider, as an **example** formula $A_2$ from **Step 2; P1** i.e. the formula

$$Q_3 x_3 \ldots Q_n x_n B(x_1, x_2, x_3, \ldots x_n)$$

We have again 2 choices to consider, but will describe only the following.

**P2** First quantifier in $A_2$ is **existential**, i. e. $A_2$ is

$$\exists x_2 Q_4 x_4 \ldots Q_n x_n B(x_1, x_2, x_3, x_4, \ldots x_n)$$

Observe that now the variables $x_1, x_2$ are **free** variables in $B(x_1, x_2, x_3, \ldots x_n)$ and hence in $A_2$.

We replace $A_2$ by the following $A_3$

$$Q_4 x_3 \ldots Q_n x_n B(x_1, x_2, g(x_1, x_2), x_4 \ldots x_n)$$

where $g$ is a **new** two argument functional symbol **added** to our original language $\mathcal{L}$.
We have **eliminated** the quantifier $Q_3$ in this case.

**Step i**
At each **Step i**, for $1 \leq i \leq n$), we build a **binary tree** of possibilities **P1**$Q_i x_i$ is **universal** or **P2** $Q_i x_i$ is **existential** and as result we obtain a formula $A_i$ with one less quantifier. The elimination process builds a sequence of formulas

$$A,\ A_1,\ A_2,\ \ldots,\ A_n = A^*$$

where the formula A belongs to our original language

$$\mathcal{L} = \mathcal{L}_{\{\neg, \cup, \cap, \Rightarrow\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}),$$

the formula $A^*$ belongs to its **Skolem extension**.
The **Skolem extension** $S\mathcal{L}$ is obtained from $\mathcal{L}$in the quantifiers elimination process and

$$S\mathcal{L} = \mathcal{L}_{\{\neg,\cup,\cap,\Rightarrow\}}(\mathbf{P}, \mathbf{F} \cup S\mathbf{F},\ \mathbf{C} \cup S\mathbf{C})$$

**Observe** that in the elimination process an **universal quantifier** introduces free variables in the formula $B(x_1, x_2, \ldots x_n)$. The **elimination** of an existential quantifier that follows universal quantifiers introduces a **new** functional symbol with number of arguments equal the number of universal quantifiers preceding it.
The resulting **open** formula $A^*$ logically equivalent to the PNF formula A.

**Example 1**
Let A be a **PNF** formula

$$\forall y_1 \exists y_2 \forall y_3 \exists y_4\ B(y_1, y_2, y_3, y_4, y_4)$$

We eliminate $\forall y_1$ and get a formula $A_1$

$$\exists y_2 \forall y_3 \exists y_4\ B(y_1, y_2, y_3, y_4)$$

We eliminate $\exists y_2$ by replacing $y_2$ by $h(y_1)$ where $h$ is a **new** one argument functional symbol **added** to our original language $\mathcal{L}$.
We get a formula $A_2$

$$\forall y_3 \exists y_4\ B(y_1, h(y_1), y_3, y_4)$$

We eliminate $\forall y_3$ and get a formula $A_3$

$$\exists y_4\ B(y_1, h(y_1), y_3, y_4)$$

We eliminate $\exists y_4$ by replacing $y_4$ by $f(y_1, y_3)$, where $f$ is a **new** two argument functional symbol **added** to our original language $\mathcal{L}$.
We get a formula $A_4$ that is our resulting **open** formula $A^*$

$$B(y_1, h(y_1), y_3, f(y_1, y_3))$$

**Example 2**
Let now A be a **PNF** formula

$$\exists y_1 \forall y_2 \forall y_3 \exists y_4 \exists y_5 \forall y_6\ B(y_1, y_2, y_3, y_4, y_4, y_5, y_6)$$

We eliminate $\exists y_1$ and get a formula $A_1$

$$\forall y_2 \forall y_3 \exists y_4 \exists y_5 \forall y_6\ B(b_1, y_2, y_3, y_4, y_4, y_5, y_6)$$

where $b_1$ is a **new** constant symbol **added** to our original language $\mathcal{L}$.
We eliminate $\forall y_2, forally_3$ and get a formulas $A_2, A_3$; here is the formula $A_3$

$$\exists y_4 \exists y_5 \forall y_6 \ B(b_1, y_2, y_3, y_4, y_4, y_5, y_6)$$

We eliminate $\exists y_4$ and get a formula $A_4$

$$\exists y_5 \forall y_6 \ B(b_1, y_2, y_3, g(y_2, y_3), y_5, y_6)$$

where $g$ is a **new** two argument functional symbol **added** to our original language $\mathcal{L}$.
We eliminate $\exists y_5$ and get a formula $A_5$

$$\forall y_6 \ B(b_1, y_2, y_3, g(y_2, y_3), h(y_2, y_3), y_6)$$

where $h$ is a **new** two argument functional symbol **added** to our original language $\mathcal{L}$.
We eliminate $\forall y_6$ and get a formula $A_6$ that is the resulting **open** formula $A^*$

$$B(b_1, y_2, y_3, g(y_2, y_3), h(y_2, y_3), y_6)$$

# 4 Clausual Form of Formulas

# 5 Unification

Unification is the process of determining whether **two** atomic formulas, i.e. **two** positive literals can be made identical by appropriate substitution for their variables. Unification is an essential part of resolution.
Intuitively, a **substitution** is is a set of associations between variables and terms in which **1.** each variable is associated with at most one term, and **2.** no variable with an associated term occurs within any of the associated terms.
For example, the following is a well defined substitution

$$\{x/c, \ y/f(b), \ z/w\}$$

and the following is not a substitution

$$\{x/g(y), \ y/f(x)\}$$

as the variable x which is associated with term $g(y)$, occurs in the term $f(x)$ associated with y; the variable y occurs in term $g(y)$ associated with variable x.

Given two atomic formulas (positive literals) $P_1 = P(x, y, z)$ and $P_2 = P(c, f(b), w)$. The substitution $\{x/c,\ y/f(b),\ z/w\}$ **unifies** $P_1$ and $P_2$, as when applied to $P_1$ produces $P_2$, i.e.

$$P_1 = P(x, y, z)\{x/c,\ y/f(b),\ z/w\} = P(x, y, z)\{x/c,\ y/f(b),\ z/w\} = P(c, f(b), w) = P_2$$

We often speak of terms associated with variables in a substitution as *bindings* for those variables; the substitution itself is called a *binding list*; the variables with bindings are said to be *bound*.

We apply substitution to literals in the clauses and produce new clauses. For example,