Ground Transformation

The ground transformation $\text{DB}_g$ of $\text{DB}$ is defined as follows:

- for each clause $\mathcal{C}$ in $\text{DB}$
  - for each grounding substitution $\theta$ from the variables of $\mathcal{C}$ to constants in $L_{\text{DB}}$
    - add clause $\mathcal{C}\theta$ to $\text{DB}_g$

Call $\mathcal{C}\theta$ a ground rule.

Note that $\text{DB}_g$ may be infinite, because an infinite number of constants exist in the domain of discourse as we have defined it. This does not pose a problem, as we only use $\text{DB}_g$ in definitions and never in actuality transform a $\text{DB}$ into $\text{DB}_g$. 
Unfounded Sets
for the Well-founded Semantics

Let a program $\mathcal{P}$, its associated Herbrand base $\mathbf{HB}_\mathcal{P}$, and a partial interpretation $I$ be given. We say $\mathcal{A} \subseteq I$ is an unfounded set of $\mathcal{P}$ with respect to $I$ if each atom $p \in \mathcal{A}$ satisfies the following condition: For each ground rule $r$ of $\mathcal{P}$ whose head is $p$, (at least) one of the following holds:

1. Some positive subgoal $q$ or negative subgoal $\text{not}(q)$ of the body occurs in $\neg I$ (i.e., is consistent with $I$);
2. Some positive subgoal of the body occurs in $\mathcal{A}$.

A literal that makes 1 or 2 true is called a witness of unusability for rule $r$ (with respect to $I$).

There is a greatest unfounded set with respect to $I$.

Horn Transformation
for the Stable Model Semantics

The **Horn transformation** \( \text{horn}(\text{DB}, I) \) of ground \( \text{DB} \) with respect to interpretation \( I \) is defined as follows:

- for each clause \( \mathcal{C} \) in \( \text{DB} \) (which is ground since \( \text{DB} \) is)
  - let \( \mathcal{C} \) be represented by

    \[
    a \langle \vec{x} \rangle \leftarrow b \langle \vec{y} \rangle_1, \ldots, b \langle \vec{y} \rangle_m, \text{not } d \langle \vec{z} \rangle_1, \ldots, \text{not } d \langle \vec{z} \rangle_n.
    \]

    if \( \{ d \langle \vec{z} \rangle_1, \ldots, d \langle \vec{z} \rangle_n \} \cap I \neq \emptyset \) then
    * do nothing (discard the clause)
  else
    * add the clause

    \[
    a \langle \vec{x} \rangle \leftarrow b \langle \vec{y} \rangle_1, \ldots, b \langle \vec{y} \rangle_m.
    \]

    to \( \text{horn}(\text{DB}, I) \)
Stable Model Semantics

The interpretation $I$ is a stable model of $DB$ iff

$$I = M_{DB}$$

Here, M stands for the minimum model.

Let us denote the set of stable models of $DB$ by $S_{DB}$. We call a database $DB$ stable iff $DB$ has at least one stable model; that is, $S_{DB}$ is non-empty.

Well-Supported Models
Equivalent to Stable Model Semantics

A model \( I \subseteq \mathbf{HB}_{\mathbf{DB}} \) is well supported with respect to \( \mathbf{DB} \) iff there exists a well founded partial order ‘\( \succ /2 \)’ on \( I \times I \) such that, for each atom \( p \langle \vec{c} \rangle \in I \), there exists a rule \( C \) for \( p \) in \( \mathbf{DB} \),

\[
C: \quad p \langle \vec{x} \rangle \leftarrow b \langle \vec{y} \rangle_1, \ldots, b \langle \vec{y} \rangle_m, \textbf{not} \ d \langle \vec{z} \rangle_1, \ldots, \textbf{not} \ d \langle \vec{z} \rangle_n.
\]

and a grounding substitution \( \theta \) from the variables of \( C \) to constants in \( \mathbf{L}_{\mathbf{DB}} \) such that \( p \langle \vec{c} \rangle = p \langle \vec{x} \rangle \theta \) and

1. \( b \langle \vec{y} \rangle_1 \theta, \ldots, b \langle \vec{y} \rangle_m \theta \in I \),
2. \( d \langle \vec{z} \rangle_1 \theta, \ldots, d \langle \vec{z} \rangle_n \theta \not\in I \), and
3. \( p \langle \vec{c} \rangle \succ b \langle \vec{y} \rangle_i \theta \), for every \( i \in \{1, \ldots, m\} \).

\( I \in \mathcal{S}_{\mathbf{DB}} \) (\( I \) is a stable model of \( \mathbf{DB} \)) iff \( I \) is a well supported model of \( \mathbf{DB} \).

Well-Founded Semantics
Advantages and Disadvantages

Advantages

• There is always exactly one well-founded partial model.
• For datalog¬, polynomial (in the size of the database!) algorithms are known.

Disadvantages

• Intuitively seems weak to some.
  – E.g., Cannot reason by case in the negative.
Stable Model Semantics
Advantages and Disadvantages

Advantages

• Intuitively more satisfying to some.
  – Does reason over case in the negative.
  – Each stable model is a minimal model of the database (treating \textbf{not} as if it were ‘\neg’); vice-versa is not true, though.

Disadvantages

• There are datalog-$\neg$ databases with \textit{no} stable models.
• There are datalog-$\neg$ databases with \textit{more than one} stable model.
  (Bothers some people.)
• For datalog-$\neg$, it is exponential (in the size of the database!) in worst-case to compute.
• It is not “stable”. Huh?!
  – Add a rule or delete a rule, and the database may cease to have any stable models.
For any locally stratified Datalog\(\neg\) database, there is *exactly one* stable model, and its well-founded model is *complete*. Also

- the perfect model,
- the stable model, and
- the well-founded model

are all equivalent.