Ground Transformation

The *ground transformation* $\mathbf{DB}_{\mathbf{G}}$ of $\mathbf{DB}$ is defined as follows:

- for each clause $\mathcal{C}$ in $\mathbf{DB}$

  for each grounding substitution θ from the variables of $\mathcal{C}$ to constants in $\mathbf{L}_{\mathbf{DB}}$

  - add clause $\mathcal{C}\theta$ to $\mathbf{DB}_{\mathbf{G}}$

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

Note that $\mathbf{DB}_{\mathbf{G}}$ 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 $\mathbf{DB}_{\mathbf{G}}$ in definitions and never in actuality transform a $\mathbf{DB}$ into $\mathbf{DB}_{\mathbf{G}}$.

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 folowwing 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 $\mathbf{not}(q)$ 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.

- A. van Gelder, K. Ross, & J. Schlipf. Unfounded Sets and Well-Founded Semantics for General Logic Programs. *Proceedings of the 7th Symposium on Principles of Database Systems (PODS).* pp. 221–230, 1988.

Horn Transformation
for the Stable Model Semantics

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

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

$$a\,\langle\vec{x}\,\rangle \;\leftarrow\; b\,\langle\vec{y}\,\rangle_1, \ldots, b\,\langle\vec{y}\,\rangle_m, \mathsf{not}\ d\,\langle\vec{z}\,\rangle_1, \ldots, \mathsf{not}\ d\,\langle\vec{z}\,\rangle_n.$$

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 $\mathsf{horn}\,(\mathbf{DB}, I)$

Stable Model Semantics

The interpretation I is a *stable model* of $\mathbf{DB}$ *iff*

$$I = \mathrm{M}_{\mathbf{DB}^{I}_{\mathbf{G}}}$$

Here, M stands for the minimum model.

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

- M. Gelfond & V. Lifschitz. The Stable Model Semantics for Logic Programming. *Proceedings of the 5th International Conference and Symposium on Logic Programming.* Eds R.A. Kowalski and K.A. Bowen. pp. 1070-1080, August 1988.

Well-Supported Models
Equivalent to Stable Model Semantics

A model $I \subseteq \mathbf{HB}_{\mathrm{DB}}$ is *well supported* with respect to $\mathbf{DB}$ *iff* there exists a well founded partial order '$>/2$' on $I \times I$ such that, for each atom $p\ \langle \vec{c}\,\rangle \in I$, there exists a rule $\mathcal{C}$ for p in $\mathbf{DB}$,

$$\mathcal{C}: \quad p\ \langle \vec{x}\,\rangle \leftarrow b\ \langle \vec{y}\,\rangle_1, \ldots, b\ \langle \vec{y}\,\rangle_m, \textsf{not}\ d\ \langle \vec{z}\,\rangle_1, \ldots, \textsf{not}\ d\ \langle \vec{z}\,\rangle_n.$$

and a grounding substitution θ from the variables of $\mathcal{C}$ to constants in $\mathbf{L}_{\mathrm{DB}}$ 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. $d\ \langle \vec{z}\,\rangle_1\theta, \ldots, d\ \langle \vec{z}\,\rangle_n\theta \notin I$, and
3. $p\ \langle \vec{c}\,\rangle > b\ \langle \vec{y}\,\rangle_i\theta$, for every $i \in \{1, \ldots, m\}$.

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

- F. Fages. A new fixpoint semantics for general logic programs compared with the well-founded and the stable model semantics. *New Generation Computing*, 9:425–443, 1991.

Well-Founded Semantics

Advantages and Disadvantages

Advantages

- There is always exactly *one* well-founded partial model.
- For datalog$\neg$, 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 **not** as if it were '$\neg$'); vice-versa is not true, though.

Disadvantages

- There are datalog$\neg$ databases with *no* stable models.
- There are datalog$\neg$ databases with *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 Locally Stratified Datalog¬
Semantics?

For any locally stratified Datalog¬ 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.