ASP Solvers:

The Smodels Language

ASP-solvers

- There are lots of ASP solvers available:
  smodels, *dlv*, *cmodels*, *assat*, *nomore*, $smodels_{cc}$, *clasp*, *platypus*, ...
- We'll briefly look at the original smodels implementation

Smodels

Origin	Helsinki University of Technology
People	I. Niemelä, P. Simons, and T. Syrjänen
Web	http://www.tcs.hut.fi/Software/smodels/
Input format	*lparse*(*g*ringo)
Extensions	• choice rules • cardinality/weight constraints • optimization via minimize/maximize statements

Assignments

- Let Π be a program and $atoms(\Pi)$ the atoms in Π.
- An assignment is a partial mapping
  $A : atoms(\Pi) \to \{\boldsymbol{T}, \boldsymbol{F}\}$.
  ☞ Think of A as a three-valued model.

Assignments

- Let Π be a program and $atoms(\Pi)$ the atoms in Π.
- An assignment is a partial mapping
  $$A : atoms(\Pi) \to \{\boldsymbol{T}, \boldsymbol{F}\}.$$
  ☞ Think of A as a three-valued model.
- An assignment A can be denoted as a pair $(A^{\boldsymbol{T}}, A^{\boldsymbol{F}})$, where
  - $A^{\boldsymbol{T}} = \{v \in atoms(\Pi) \mid A(v) = \boldsymbol{T}\}$
  - $A^{\boldsymbol{F}} = \{v \in atoms(\Pi) \mid A(v) = \boldsymbol{F}\}$

Assignments

- Let Π be a program and $atoms(\Pi)$ the atoms in Π.
- An assignment is a partial mapping
  $$A : atoms(\Pi) \to \{\boldsymbol{T}, \boldsymbol{F}\}.$$
  ☞ Think of A as a three-valued model.
- An assignment A can be denoted as a pair $(A^{\boldsymbol{T}}, A^{\boldsymbol{F}})$, where
  - $A^{\boldsymbol{T}} = \{v \in atoms(\Pi) \mid A(v) = \boldsymbol{T}\}$
  - $A^{\boldsymbol{F}} = \{v \in atoms(\Pi) \mid A(v) = \boldsymbol{F}\}$
- We also denote an assignment A by a set of signed objects:
  $$\{\boldsymbol{T}v \mid v \in A^{\boldsymbol{T}}\} \cup \{\boldsymbol{F}v \mid v \in A^{\boldsymbol{F}}\}$$

Assignments

- Let Π be a program and $atoms(\Pi)$ the atoms in Π.
- An assignment is a partial mapping
  $$A : atoms(\Pi) \to \{\boldsymbol{T}, \boldsymbol{F}\}.$$
  ☞ Think of A as a three-valued model.
- An assignment A can be denoted as a pair $(A^{\boldsymbol{T}}, A^{\boldsymbol{F}})$, where
  - $A^{\boldsymbol{T}} = \{v \in atoms(\Pi) \mid A(v) = \boldsymbol{T}\}$
  - $A^{\boldsymbol{F}} = \{v \in atoms(\Pi) \mid A(v) = \boldsymbol{F}\}$
- We also denote an assignment A by a set of signed objects:
  $$\{\boldsymbol{T}v \mid v \in A^{\boldsymbol{T}}\} \cup \{\boldsymbol{F}v \mid v \in A^{\boldsymbol{F}}\}$$
- Example: Given $atoms(\Pi) = \{a, b, c\}$, assignment
  $$A = \{a \mapsto \boldsymbol{T}, c \mapsto \boldsymbol{F}\}$$
  can be represented as
  $$(\{a\}, \{c\}) \qquad \text{or} \qquad \{\boldsymbol{T}a, \boldsymbol{F}c\}.$$

(Simplified) Smodels Algorithm

Let Π be a program and A a (partial) assignment.

smodels(Π, A):

$A \leftarrow$ **expand**(Π, A)
if $A^{\boldsymbol{T}} \cap A^{\boldsymbol{F}} \neq \emptyset$ **then return**
if $A^{\boldsymbol{T}} \cup A^{\boldsymbol{F}} = \mathit{atoms}(\Pi)$ **then exit with** $A^{\boldsymbol{T}}$
$x \leftarrow$ **select(**$\mathit{atoms}(\Pi) \setminus (A^{\boldsymbol{T}} \cup A^{\boldsymbol{F}})$**)**
smodels($\Pi, A \cup \{\boldsymbol{T}x\}$**)**
smodels($\Pi, A \cup \{\boldsymbol{F}x\}$**)**

Call: **smodels(**$\Pi, \emptyset$**)**

smodels algorithm

- Backtracking search, building a binary search tree
- Choices on unassigned atoms
- The search space is pruned by
  - making one choice at a time by appeal to a heuristics (*select*)
  - the set of remaining choices is reduced and conflicts are detected (*expand*)

A closer look at *expand*

- expand(Π, A):

  repeat

  $$A \leftarrow \mathit{atleast}(\Pi, A)$$
  $$A' \leftarrow A$$
  $$A \leftarrow A \cup \{\boldsymbol{F}x \mid x \in \mathit{atom}(\Pi) \setminus \mathit{atmost}(\Pi, A)\}$$

  until $A = A'$

A closer look at *expand*

- expand(Π, A):

  repeat

  $$A \leftarrow \mathit{atleast}(\Pi, A)$$
  $$A' \leftarrow A$$
  $$A \leftarrow A \cup \{\boldsymbol{F}x \mid x \in \mathit{atom}(\Pi) \setminus \mathit{atmost}(\Pi, A)\}$$

  until $A = A'$

- atleast amounts to Fitting's operation
  - These are atoms that must be true or false, given A.

A closer look at *expand*

- expand(Π, A):

  repeat

  $$A \leftarrow \mathit{atleast}(\Pi, A)$$
  $$A' \leftarrow A$$
  $$A \leftarrow A \cup \{\boldsymbol{F}x \mid x \in \mathit{atom}(\Pi) \setminus \mathit{atmost}(\Pi, A)\}$$

  until $A = A'$

- atleast amounts to Fitting's operation
  - These are atoms that must be true or false, given A.
- atmost computes those atoms derivable in the underlying positive program of Π.
  - Its complement contains those atoms that cannot be derived, given A.
  - This is called the *greatest unfounded set*

A closer look at *expand*

- expand(Π, A):

  repeat

  $$A \leftarrow \mathit{atleast}(\Pi, A)$$
  $$A' \leftarrow A$$
  $$A \leftarrow A \cup \{\boldsymbol{F}x \mid x \in \mathit{atom}(\Pi) \setminus \mathit{atmost}(\Pi, A)\}$$

  until $A = A'$

- atleast amounts to Fitting's operation
  - These are atoms that must be true or false, given A.
- atmost computes those atoms derivable in the underlying positive program of Π.
  - Its complement contains those atoms that cannot be derived, given A.
  - This is called the *greatest unfounded set*
- expand computes the *well-founded model* of Π.

Heuristics: Lookahead

- Strengthen propagation by failed literal detection
- Given a program Π, an atom x, and an assignment A
  - if expand$(\Pi, A \cup \{\boldsymbol{T}x\})$ yields a conflict, then add $\boldsymbol{F}x$ to A
  - if expand$(\Pi, A \cup \{\boldsymbol{F}x\})$ yields a conflict, then add $\boldsymbol{T}x$ to A
  - if both yield a conflict, "backtrack"

Heuristics: Lookahead

- Strengthen propagation by failed literal detection
- Given a program Π, an atom x, and an assignment A
  - if expand$(\Pi, A \cup \{\boldsymbol{T}x\})$ yields a conflict, then add $\boldsymbol{F}x$ to A
  - if expand$(\Pi, A \cup \{\boldsymbol{F}x\})$ yields a conflict, then add $\boldsymbol{T}x$ to A
  - if both yield a conflict, "backtrack"

- Lookahead is also used for selecting the next unassigned atom x

Heuristics: Lookahead

- Strengthen propagation by failed literal detection
- Given a program Π, an atom x, and an assignment A
  - if expand$(\Pi, A \cup \{\boldsymbol{T}x\})$ yields a conflict, then add $\boldsymbol{F}x$ to A
  - if expand$(\Pi, A \cup \{\boldsymbol{F}x\})$ yields a conflict, then add $\boldsymbol{T}x$ to A
  - if both yield a conflict, "backtrack"

- Lookahead is also used for selecting the next unassigned atom x
- That is, given that
  - x^+ is the number of atoms that are assigned through expand$(\Pi, A \cup \{\boldsymbol{T}x\})$ and
  - x^- is the number of atoms that are assigned through expand$(\Pi, A \cup \{\boldsymbol{F}x\})$,

  select an atom x with a maximal $min(x^+, x^-)$