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
Smodes

Origin  Helsinki University of Technology
People  I. Niemelä, P. Simons, and T. Syrjänen
Web    http://www.tcs.hut.fi/Software/smodels/

Input format  lparse(gringo)

Extensions  • choice rules
             • cardinality/weight constraints
             • optimization via minimize/maximize statements
Assignments

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

Example: Given $\text{atoms}(\Pi) = \{a, b, c\}$, assignment $A = \{a \mapsto T, c \mapsto F\}$ can be represented as $(\{a\}, \{c\})$ or $\{T \ a, F \ c\}$.
Assignments

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

• Let \( \Pi \) be a program and \( \text{atoms}(\Pi) \) the atoms in \( \Pi \).

• An **assignment** is a partial mapping
  \[
  A : \text{atoms}(\Pi) \rightarrow \{ T, F \}.
  \]
  Think of \( A \) as a three-valued model.

• An assignment \( A \) can be denoted as a pair \((A^T, A^F)\), where
  \[
  \begin{align*}
  A^T &= \{ v \in \text{atoms}(\Pi) \mid A(v) = T \} \\
  A^F &= \{ v \in \text{atoms}(\Pi) \mid A(v) = F \}
  \end{align*}
  \]

• We also denote an assignment \( A \) by a set of **signed objects**:
  \[
  \{ T v \mid v \in A^T \} \cup \{ F v \mid v \in A^F \}
  \]
Assignments

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

Let $\Pi$ be a program and $A$ a (partial) assignment.

smodels($\Pi$, $A$):

$A \leftarrow \text{expand}(\Pi, A)$

if $A^T \cap A^F \neq \emptyset$ then return

if $A^T \cup A^F = \text{atoms}(\Pi)$ then exit with $A^T$

$x \leftarrow \text{select}(\text{atoms}(\Pi) \setminus (A^T \cup A^F))$

smodels($\Pi$, $A \cup \{T x\}$)

smodels($\Pi$, $A \cup \{F x\}$)

Call: smodels($\Pi$, $\emptyset$)
smo\textsuperscript{e}ds \textsuperscript{al}\textsuperscript{}\textit{g}\textsuperscript{}r\textsuperscript{h}\textit{m}\textsuperscript{}\textsuperscript{}\textsuperscript{}\text{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 heuristic (\textit{select})
  • the set of remaining choices is reduced and conflicts are detected (\textit{expand})
A closer look at \textit{expand}

\begin{itemize}
  \item \texttt{expand}(\Pi, A):
    \begin{verbatim}
    repeat
      \begin{align*}
      &A \leftarrow \textit{atleast}(\Pi, A) \\
      &A' \leftarrow A \\
      &A \leftarrow A \cup \{F x \mid x \in \textit{atom}(\Pi) \setminus \textit{atmost}(\Pi, A)\}
    \end{align*}
    until \quad A = A'
    \end{verbatim}
\end{itemize}
A closer look at \textit{expand}

- \textbf{expand}(\Pi, A):
  
  repeat
  
  \begin{align*}
  A & \leftarrow \text{atleast}(\Pi, A) \\
  A' & \leftarrow A \\
  A & \leftarrow A \cup \{F_x \mid x \in \text{atom}(\Pi) \setminus \text{atmost}(\Pi, A)\}
  
  & \text{until } A = A'
  \end{align*}

- \textbf{atleast} amounts to Fitting's operation
  
  - These are atoms that must be true or false, given \( A \).
A closer look at $\text{expand}$

- **$\text{expand}(\Pi, A)$:**
  
  repeat
  
  \[
  A \leftarrow \text{atleast}(\Pi, A)
  \]
  \[
  A' \leftarrow A
  \]
  \[
  A \leftarrow A \cup \{Fx \mid x \in \text{atom}(\Pi) \setminus \text{atmost}(\Pi, A)\}
  \]
  
  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 $\Pi$.
  - Its complement contains those atoms that cannot be derived, given $A$.
  - This is called the **greatest unfounded set**
A closer look at expand

• expand(\(\Pi, A\)):
  repeat
    
    \(A \leftarrow \text{atleast}(\Pi, A)\)
    \(A' \leftarrow A\)
    
    \(A \leftarrow A \cup \{F \mid x \in \text{atom}(\Pi) \setminus \text{atmost}(\Pi, A)\}\)
  until \(A = A'\)

• \text{atleast} amounts to Fitting’s operation
  • These are atoms that \textbf{must} be true or false, given \(A\).

• \text{atmost} computes those atoms derivable in the underlying positive program of \(\Pi\).
  • Its complement contains those atoms that \textbf{cannot} be derived, given \(A\).
  • This is called the \textit{greatest unfounded set}

• expand computes the \textit{well-founded model} of \(\Pi\).
Heuristics: Lookahead

• Strengthen propagation by failed literal detection

• Given a program $\Pi$, an atom $x$, and an assignment $A$
  • if $\text{expand}(\Pi, A \cup \{T \cdot x\})$ yields a conflict, then add $F \cdot x$ to $A$
  • if $\text{expand}(\Pi, A \cup \{F \cdot x\})$ yields a conflict, then add $T \cdot x$ to $A$
  • if both yield a conflict, “backtrack”
Heuristics: Lookahead

- Strengthen propagation by failed literal detection
- Given a program $\Pi$, an atom $x$, and an assignment $A$
  - if $\text{expand}(\Pi, A \cup \{T_x\})$ yields a conflict, then add $F_x$ to $A$
  - if $\text{expand}(\Pi, A \cup \{F_x\})$ yields a conflict, then add $T_x$ 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 $\Pi$, an atom $x$, and an assignment $A$
  - if $\text{expand}(\Pi, A \cup \{T x\})$ yields a conflict, then add $F x$ to $A$
  - if $\text{expand}(\Pi, A \cup \{F x\})$ yields a conflict, then add $T x$ 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 $\text{expand}(\Pi, A \cup \{T x\})$ and
  - $x^-$ is the number of atoms that are assigned through $\text{expand}(\Pi, A \cup \{F x\})$,

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