Contexts in
Answer Set Programming

Gerhard Brewka
brewka@informatik.uni-leipzig.de

Universität Leipzig
1. Motivation
2. Answer set programming (short reminder)
3. Contexts
   - The Trento school
   - Contextual default logic
   - Contextual logic programming
4. Applications
5. Conclusions
1. Motivation

- John McCarthy: *we offer no definition of context*
- viewpoints, perspectives, granularity, agents, ...
- (almost) independent units of reasoning
- locality: different languages, reasoning methods
- compatibility: information flow between contexts
- essential for handling complexity, consistency, ...

![Diagram of contexts with labels Mr. 1 and Mr. 2]
2. Answer set programming
Answer sets

- define semantics for logic programs with strict and default negation (extended LPs)
- rules of the form \((a, b_i, c_j \text{ literals})\):
  
  \[
  a \leftarrow b_1, \ldots, b_n, \text{not } c_1, \ldots, \text{not } c_m
  \]

- AS acceptable set of beliefs based on program requirements:
  - closed: all rules used to generate AS
    if all \(b_i \in \text{AS}\), no \(c_j \in \text{AS}\), then \(a \in \text{AS}\)
  - grounded: \(a\) in AS implies derivation of \(a\) from rules whose not-preconds are not in AS
To check whether $S$ is AS of $P$
- remove $S$-defeated rules ($\not\text{not } L$ in body, $L \in S$)
- remove $\not\text{not}$ literals from remaining rules
- check whether $S = \text{closure of reduced program}$
A simple test

To check whether $S$ is AS of $P$

- remove $S$-defeated rules ($\text{not } L \text{ in body, } L \in S$)
- remove $\text{not}$ literals from remaining rules
- check whether $S = \text{closure of reduced program}$

$$a \leftarrow \text{not } b$$
$$b \leftarrow \text{not } c$$
A simple test

To check whether $S$ is AS of $P$

- remove $S$-defeated rules ($\text{not}\ L$ in body, $L \in S$)
- remove $\text{not}$ literals from remaining rules
- check whether $S = \text{closure of reduced program}$

\[
\begin{align*}
    a &\leftarrow \text{not} \ b \\
    b &\leftarrow \text{not} \ c \\
\end{align*}
\]

\[
\{a\}
\]

NO, not closed
A simple test

To check whether $S$ is AS of $P$

- remove $S$-defeated rules ($\text{not } L \text{ in body}, \ L \in S$)
- remove not literals from remaining rules
- check whether $S = \text{closure of reduced program}$

\[
\begin{align*}
  a & \leftarrow \text{not } b \\
  b & \leftarrow \text{not } c
\end{align*}
\]

\{$a, b$\}

NO, not grounded
A simple test

To check whether $S$ is AS of $P$

- remove $S$-defeated rules ($\text{not } L \text{ in body, } L \in S$)
- remove $\text{not}$ literals from remaining rules
- check whether $S = \text{closure of reduced program}$

$$a \leftarrow \text{not } b$$

$$b \leftarrow \text{not } e$$

{\{b\}}

YES, grounded and closed
Example: graph coloring

Description of graph:
\[
\text{node}(v_1), \ldots, \text{node}(v_n), \text{edge}(v_i, v_j), \ldots
\]

Generate: every node needs exactly 1 color

\[
\begin{align*}
\text{col}(X, r) & \leftarrow \text{node}(X), \neg \text{col}(X, b), \neg \text{col}(X, g) \\
\text{col}(X, b) & \leftarrow \text{node}(X), \neg \text{col}(X, r), \neg \text{col}(X, g) \\
\text{col}(X, g) & \leftarrow \text{node}(X), \neg \text{col}(X, r), \neg \text{col}(X, b)
\end{align*}
\]

Test: linked nodes cannot have same color

\[
\leftarrow \text{edge}(X, Y), \text{col}(X, Z), \text{col}(Y, Z)
\]

Each answer set describes a solution!
Example: graph coloring

Description of graph:

\[ \text{node}(v_1), \ldots, \text{node}(v_n), \text{edge}(v_i, v_j), \ldots \]

Generate: every node needs exactly 1 color

\[
\begin{align*}
\text{col}(X, r) & \leftarrow \text{node}(X), \text{not col}(X, b), \text{not col}(X, g) \\
\text{col}(X, b) & \leftarrow \text{node}(X), \text{not col}(X, r), \text{not col}(X, g) \\
\text{col}(X, g) & \leftarrow \text{node}(X), \text{not col}(X, r), \text{not col}(X, b)
\end{align*}
\]

Test: linked nodes cannot have same color

\[ f \leftarrow \text{edge}(X, Y), \text{col}(X, Z), \text{col}(Y, Z), \text{not f} \]

Each answer set describes a solution!
A useful language extension

Bounds on number of satisfied literals:

\[ L \{a_1, \ldots, a_k\} U \]

Read: at least \( L \) and at most \( U \) of the \( a_i \) must be true

Allows us to replace 3 color assignment rules with:

\[ 1 \{\text{col}(X, r), \text{col}(X, b), \text{col}(X, g)\} \leftarrow \text{node}(X) \]

or, if extension of color is \( \{r, b, g\} \):

\[ 1 \{\text{col}(X, Y) : \text{color}(Y)\} \leftarrow \text{node}(X) \]
Why LPs under AS semantics?

- simple yet expressive language
- transitive closure, nonmonotonic rules, ...

\[ \text{flies}(X) \leftarrow \text{not} \ ab(X), \text{bird}(X) \]

- simple epistemic distinctions:

\[ \text{safe} \neq \text{not} \neg \text{safe} \neq \neg \text{safe} \]

- interesting implementations: dlv, Smodels, nomore, ASSAT ...

- interesting applications: configuration, diagnosis, planning, reasoning about action, shuttle control, model checking, information integration, ...
3. Contexts
Multi-context system

\((\{T_i\}, Delta_{br})\)

each \(T_i = (L_i, \Omega_i, \Delta_i)\) formal system, \(Delta_{br}\) bridge rules using labeled formulas \(c : p\) with \(p \in L_c\).

- semantics: local models + compatibility
- information flow across contexts via bridge rules
- reasoning within/across contexts monotonic
- exception (Roelofsen/Serafini 05) has problems
Contextual default logic

Contextual default theory \(((D_1, W_1), \ldots, (D_n, W_n))\).

\((D_i, W_i)\) default theory, defaults may refer to other contexts:

\[
(c_1 : p_1), \ldots, (c_t : p_t) : (c_{t+1} : q_1), \ldots, (c_{t+k} : q_k)
\]

\(r\)

\(\Gamma(S_1, \ldots, S_n)\) minimal tuple \((S'_1, \ldots, S'_n)\) with:

1. \(W_i \subseteq S'_i\),

2. \(S'_i\) deductively closed (over \(L_i\)), and

3. \((c_1 : p_1), \ldots, (c_t : p_t) : (c_{t+1} : q_1), \ldots, (c_{t+k} : q_k)/r \in D_i\) and for all \(i, j\): \(p_i \in S'_{c_i}\) and \(-q_j \notin S_{c_{t+j}}\), then \(r \in S'_i\).

Extension fixpoint of \(\Gamma\).
Contextual logic programming

- Contextual LP system: \( C = (P_1, \ldots, P_n) \).
- \( P_i \) contextual LP: labeled literals \((c : l)\) allowed in rule bodies \((c \text{ context, } l \text{ literal in } c\text{'s language})\):

  \[
  a \leftarrow (c_1 : b_1), \ldots, \text{not} (c_j : b_j), \ldots
  \]

- Minimal context model of definite \( C \) (no not):
  smallest \((S_1, \ldots, S_n)\) with:

  1. \( a \in S_i \) if \( a \leftarrow (c_1 : b_1), \ldots, (c_k : b_k) \in P_i, b_1 \in S_{c_1}, \ldots, b_k \in S_{c_k} \),
  2. \( S_i = \text{Lit}_i \) if \( S_i \) has complementary literals.
Answer sets

\[ C = (P_1, \ldots, P_n) \text{ arbitrary system, } S = (S_1, \ldots, S_n), \ S_i \text{ literals in } P_i \text{’s language.} \]

\[ C^S \text{ obtained from } C \text{ by} \]
1. deleting rules with literal not \((c : l)\) s.t. \(l \in S_c\),
2. deleting not literals from remaining rules.

\[ C^S \text{ definite, has minimal context model } M_{C^S}. \]

\[ S \text{ answer set iff } S = M_{C^S}. \]

Also possible to define skeptical inference à la WFS
4. Applications
Car domain, \( P_1 \) abstraction of \( P_2 \), \( P_1 \) may contain

\[
\begin{align*}
\text{expensive}(X) & \leftarrow 2 : \text{price}(X, Y), Y \geq 30000 \\
\text{sportive}(X) & \leftarrow 2 : \text{speed}(X, Y), 180 < Y, \\
\text{not bad-accel}(X) & \\
\text{bad-accel}(X) & \leftarrow 2 : \text{0-to-100}(X, Y), Y > 10
\end{align*}
\]

Assume now intelligent grounder instantiating context, proposition and domain variables correctly.
believe $p$ if someone does and nobody believes $\neg p$:

\[
P \leftarrow (C : P), \textbf{not} \ rej(P)
\]

\[
rej(P) \leftarrow (C : \neg P)
\]
believe $p$ if someone does and nobody believes $-p$:

\[
P \leftarrow (C : P), \textbf{not} \ rej(P)
\]
\[
rej(P) \leftarrow (C : -P)
\]

believe $p$ if someone you trust does and nobody you trust believes $-p$:

\[
P \leftarrow (C : P), \text{trusted}(C'), \textbf{not} \ rej(P)
\]
\[
rej(P) \leftarrow (C : -P), \text{trusted}(C)
\]
believe $p$ if someone does and nobody believes $\neg p$:

\[
P \leftarrow (C : P), \textbf{not rej}(P)
\]

\[
\text{rej}(P) \leftarrow (C : \neg P)
\]

believe $p$ if someone you trust does and nobody you trust believes $\neg p$:

\[
P \leftarrow (C : P), \textbf{trusted}(C'), \textbf{not rej}(P)
\]

\[
\text{rej}(P) \leftarrow (C : \neg P), \textbf{trusted}(C)
\]

\[
\text{Warning: odd loops!!}
\]

\[
P_0 \xrightarrow{t} P_1 \xrightarrow{t} P_2 \xrightarrow{t} P_3
\]

odd loop if $P_3$ mistrusts $P_2$. 
believe $p$ if majority does:

$$P \leftarrow N \{(C : P) : con(C) \} N, N > n/2$$
believe $p$ if majority does:

$$P \leftarrow N\{(C : P) : \text{con}(C)\}N, \, N > n/2$$

total preference order via numbering: $1 < 2 < 3 \ldots$

$$P \leftarrow (C : P), \textbf{not} \, \text{rej}(C, P)$$

$$\text{rej}(C, P) \leftarrow (C : P), (C' : -P), C < C'$$
believe $p$ if majority does:

$$ P \leftarrow N \{(C : P) : \text{con}(C)\} N, N > n/2 $$

total preference order via numbering: $1 < 2 < 3 \ldots$

$$ P \leftarrow (C : P), \textbf{not} \text{ rej}(C, P) $$

$$ \text{rej}(C, P) \leftarrow (C : P), (C'' : -P), C < C'' $$

partial preference order via predicate $\prec$, sceptical:

$$ P \leftarrow \text{acc}(P), \textbf{not} \text{ acc}(-P) $$

$$ \text{acc}(P) \leftarrow (C : P), \textbf{not} \text{ rej}(C, P) $$

$$ \text{rej}(C, P) \leftarrow (C : P), (C'' : -P), C \prec C'' $$

Contexts in ASP – p. 18/22
Voting

Simple majority vote:

\[ \text{votes}(X, N) \leftarrow N \{ (C : \text{best}(X)) : \text{con}(C) \} \]

\[ \text{wins}(X) \leftarrow \text{not } \neg \text{wins}(X) \]

\[ \neg \text{wins}(X) \leftarrow \text{votes}(X, N), \text{votes}(Y, M), M > N \]

Condorcet rule:

\[ \text{beats}(X, Y, N) \leftarrow N \{ (C : \text{beats}(X, Y)) : \text{con}(C) \} \]

\[ \text{wins}(X) \leftarrow \text{not } \neg \text{wins}(X) \]

\[ \neg \text{wins}(X) \leftarrow \text{beats}(X, Y, N)), \text{beats}(Y, X, M), M \geq N \]
5. Conclusions
What has been achieved?

- ASP a promising declarative paradigm
- simple yet expressive, interesting applications, interesting solvers
- showed how to add simple notion of context
- presented possible applications
- missing: current context, switching context, identification of most adequate context, ...
Future work

Integration of contexts and preferences

- use prioritized logic programs within context
- use priorities among bridge rules
- modify priorities in a context using bridge rules

More diverse systems

- combine LP under AS semantics with LP under WFS with probabilistic system with modal reasoner ...

Thanks to: F. Roelofsen, L. Serafini for cooperation you for listening