Defeasible ACE Rules

Martin Diller    Hannes Strass    Adam Z. Wyner

Vienna University of Technology    Leipzig University    University of Aberdeen

NTFA’2017 — 17.08.2017
Motivation

Human-aware AI:
- Can reason about information generated by humans.
  - Is usually revisable; often incomplete and inconsistent.
- Is transparent to scrutiny by (non-expert) humans.

Our setting:
- Defeasible knowledge in the form of rules.

Our goal:
- Combine advances in computational linguistics and formal argumentation to realise the goals of human-aware AI in this setting.

Added benefit:
- Connect two clearly related disciplines that remain rather disconnected in practice.
Our work:

- We extend an existing controlled natural language, ACE, with means for expressing generic generalisations ("it is usual that...").
  - A controlled natural language (CNL) is a subset of a natural language, restricted in lexicon, grammar; usually with a fixed semantics. Thus, eliminating ambiguity and reducing complexity.

- Building on tools for ACE, we develop a reasoner for defeasible rules expressed in natural language.

- We employ a novel argumentation-inspired semantics.
  - Allows for transparent reasoning with incomplete, inconsistent knowledge bases.
  - Circumvents problems in realising knowledge bases via abstract argumentation.
Background to this work

- Wyner and Strass: dARe - Using Argumentation to Explain Conclusions from a Controlled Natural Language Knowledge Base. IEA/AIE (2) 2017: 328-338.
1 Introduction
   • Motivation
   • Background to this work
   • Outline

2 Extending ACE
   • ACE
   • AceRules
   • ACE rules with generics

3 Direct-stable semantics
   • Motivation
   • Definition

4 Our prototype
   • Architecture
   • Description

5 Ongoing work

6 Conclusions
ACE: Attempto Controlled English
attempto.ifi.uzh.ch

- CNL for the English language developed at University of Zurich.
- **Vocabulary** comprises predefined function words (e.g. determiners, conjunctions, prepositions), predefined phrases (there is / are, it is false that ...), and an extendable set of content-words (nouns, verbs, adjectives, adverbs).
- **Grammar** supports (among others): quantification, negation, logical connectives, modality, active & passive voice, singular & plural, relative clauses, etc.
Semantics given in terms of **discourse representation structures (DRSes)**: account for linguistic phenomena as anaphora, tense and, more generally, presuppositions. In ACE only anaphora resolution is supported.

- DRSes are constructed dynamically (anaphora resolution).
- Complete DRSes (all co-references are resolved) have a model-theoretic semantics and can be translated to FOL.

Many tools available for ACE, including the open-source parser **APE**.

Also constructs DRSes, offers translations from DRSes to other languages (e.g. FOL, OWL, ...), and does paraphrasing.
AceRules
(Kuhn, 2007)

- ACE-based interface to formal rule systems.
- Support for logic programs under the stable and courteous semantics.
- Strict negation ("John is not a customer", "nobody knows John", ..) and negation as failure ("A customer is not provably trustworthy", “it is not provable that John has a card").
- Checks whether DRSes generated from input text by APE conform to the required rule language.
- Transforms DRSes in some cases in which the DRS does not conform syntactically, but can be made to conform ("intelligent grouping").
- Relies on external solvers for the stable semantics; native implementation of the courteous semantics.
AceRules example

Input ACE text:

John owns a car.
Bill does not own a car.
If someone does not own a car then he/she owns a house.
AceRules example (cont.)

**DRS (simplified):**

[A,B]
object(A,car)
predicate(B,own,John,A)
NOT
[C,D]
object(C,car)
predicate(D,own,Bill,C)
[E]
object(E,somebody)
NOT
[F,G]
object(F,car)
predicate(G,own,E,F)
=>
[H,I]
object(H,house)
predicate(I,own,E,H)

**FOL (with some transformations):**

\[
\begin{align*}
&[\text{object}(a, \text{car}) \land \text{predicate}(o, \text{own}, John, a)] \\
&\land \\
&[\neg \exists C (\text{object}(C, \text{car}) \land \text{predicate}(o, \text{own}, Bill, C))] \\
&\land \\
&\forall E [\text{object}(E, \text{somebody}) \land \\
&\neg \exists F (\text{object}(F, \text{car}) \land \text{predicate}(o, \text{own}, E, F))] \\
&\Rightarrow \\
&[\exists H (\text{object}(H, \text{house}) \land \text{predicate}(o, \text{own}, E, H))] 
\end{align*}
\]
AceRules example (cont.)

ACE rules (simplified):

-\text{group}(\text{pred\_mod}(\text{own}, \text{Bill}, []), \text{object}(\text{car})).
\text{group}(\text{pred\_mod}(\text{own}, A, []), \text{object}(\text{house}))
\text{<- object}(A, B, C, D, E, F), -\text{group}(\text{pred\_mod}(\text{own}, A, []), \text{object}(\text{car})).
\text{group}([\text{pred\_mod}(\text{own}, \text{John}, []), \text{object}(\text{car}])].
\text{object}(\text{Bill}).
\text{object}(\text{John}).
AceRules example (cont.)

Output:

ANSWERTEXT #1:
John owns a car.
Bill owns a house.
It is false that Bill owns a car.
AceRules example (cont.)

**Input ACE text:**
John owns a car.
The car is red.
Bill does not own a car.
If someone does not own a car then he/she owns a house.

**Output:**
ERROR: The program violates the atom-restriction.
Generics in AceRules:

John owns a car.
Bill does not own a car.
If someone does not own a car and it is not provable that he/she does not own a house then he/she owns a house.
Our treatment of generics

**Our treatment:**

John owns a car.
Bill does not own a car.
If someone does not own a car then it is usual that he/she owns a house.
A challenge for AceRules

- Variation on an example due to (Pollock, 2007).

**Input text:**

John owns a car.
Bill does not own a car.
If someone does not own a car then *it is usual that* he/she owns a house.
A challenge for AceRules

- Variation on an example due to (Pollock, 2007).

**Input text:**

John owns a car.
Bill does not own a car.
If someone does not own a car then *it is usual that* he/she owns a house.
If someone owns a house then *it is usual that* he/she is employed.
If someone owns a car then *it is usual that* he/she is employed.
A challenge for AceRules

- Variation on an example due to (Pollock, 2007).

Input text:

John owns a car.
Bill does not own a car.
If someone does not own a car then *it is usual that* he/she owns a house.
If someone owns a house then *it is usual that* he/she is employed.
If someone owns a car then *it is usual that* he/she is employed.
Paul owns a car.
If John is employed then Paul is employed.
If Bill is employed then Paul is not employed.
A challenge for AceRules (cont.)

Input text (original APE format):

John owns a car.
Bill does not own a car.
If someone does not own a car and it is not provable that he/she does
not own a house then he/she owns a house.
If someone owns a house and it is not provable that he/she is not
employed then he/she is employed.
If someone owns a car and it is not provable that he/she is not employed
then he/she is employed.
Paul owns a car.
If John is employed then Paul is employed.
If Bill is employed then Paul is not employed.
A challenge for AceRules (cont.)

Input text (original APE format):

John owns a car.
Bill does not own a car.
If someone does not own a car and it is not provable that he/she does not own a house then he/she owns a house.
If someone owns a house and it is not provable that he/she is not employed then he/she is employed.
If someone owns a car and it is not provable that he/she is not employed then he/she is employed.
Paul owns a car.
If John is employed then Paul is employed.
If Bill is employed then Paul is not employed.

No answer set under the stable semantics.
A challenge for AceRules (cont.)

Input text (original APE format):

John owns a car.
Bill does not own a car.
If someone does not own a car and it is not provable that he/she does not own a house then he/she owns a house.
If someone owns a house and it is not provable that he/she is not employed then he/she is employed.
If someone owns a car and it is not provable that he/she is not employed then he/she is employed.
Paul owns a car.
If John is employed then Paul is employed.
If Bill is employed then Paul is not employed.

One answer-set under the courteous semantics:
John is employed. Bill is employed. Paul owns a car. John owns a car.
Bill owns a house. It is false that Bill owns a car.
Our treatment of generics

Input text:

John owns a car.
Bill does not own a car.
If someone does not own a car then it is usual that he/she owns a house.
If someone owns a house then it is usual that he/she is employed.
If someone owns a car then it is usual that he/she is employed.
Paul owns a car.
If John is employed then Paul is employed.
If Bill is employed then Paul is not employed.
Our treatment of generics

**Answer text 1:**

Bill is employed.
Paul owns a car.
Bill owns a house.
John owns a car.
It is false that Paul is employed.
It is false that Bill owns a car.
Our treatment of generics

Answer text 2:

John is employed.
Paul is employed.
Paul owns a car.
Bill owns a house.
John owns a car.
It is false that Bill owns a car.
Motivation behind the direct-stable semantics

(Strass and Wyner, 2017)

Motivations behind direct-stable semantics:

- Define semantics directly on sets of strict and defeasible rules.
- Time-honored interpretation of strict rules as holding in all possible worlds, defeasible rules in all non-exceptional possible worlds.
- All the benefits of argumentation (justification, paraconsistent reasoning, ...), while avoiding explicit argument construction (potential exponential blowup of arguments!).
- Arguments can, rather, be constructed on demand for explanation.
- Rationality postulates (Caminada and Amgoud, 2007) satisfied by construction.
Defeasible Theories: propositional case

- Basis: set $\mathcal{P}$ of propositional variables
- Strict rules: $b_1, \ldots, b_m \rightarrow h$
- Defeasible rules: $b_1, \ldots, b_m \Rightarrow h$
- $b_1, \ldots, b_m, h$: literals ($p$ or $\neg p$) constructed from $\mathcal{P}$.
- A defeasible theory is a tuple $\mathcal{T} = (\mathcal{P}, S, D)$ of sets of propositional variables, strict, and defeasible rules.

- Strict rules hold in all possible worlds (consistent sets of literals).
- Defeasible rules in all non-exceptional possible worlds.
Direct Semantics: Possible Sets
Sets of consistent conclusions

**Definition (Possible Sets)**

Let $\mathcal{T} = (\mathcal{P}, S, D)$ be a defeasible theory. A set $M \subseteq \mathcal{L}_P$ of literals is a *possible set* for $\mathcal{T}$ if and only if there exists a set $D_M \subseteq D$ such that:

1. $M$ is consistent;
2. $M$ is closed under $S \cup D_M$;
3. $D_M$ is $\subseteq$-maximal with respect to items 1 and 2.

$D_M$ are the defeasible rules that hold in $M$. 
Small Example

Example

Defeasible theory $\mathcal{T} = (\{a, b\}, \emptyset, \{a \Rightarrow b, \ b \Rightarrow a\})$ has seven possible sets:

- $M_1 = \emptyset$,
- $M_2 = \{\neg a\}$,
- $M_3 = \{\neg b\}$,
- $M_4 = \{\neg a, \neg b\}$,
- $M_5 = \{a, \neg b\}$,
- $M_6 = \{\neg a, b\}$,
- $M_7 = \{a, b\}$. 
Towards Explanations and Arguments

Justifying conclusions

**Definition (Derivation)**

Let $\mathcal{T} = (\mathcal{P}, S, D)$ be a defeasible theory. A *derivation in $\mathcal{T}$* (for $z$) is a set $R \subseteq S \cup D$ of rules with a partial order $\preceq$ on $R$ such that:

1. $\preceq$ has a greatest element $(B_z, z) \in R$;
2. for each rule $(B, h) \in R$, we have: for each $y \in B$, there is a rule $(B_y, y) \in R$ with $(B_y, y) \prec (B, h)$ (where $\prec$ is the strict partial order contained in $\preceq$);
3. $R$ is $\subseteq$-minimal with respect to items 1 and 2.
Small Example

Example

Defeasible theory $\mathcal{T} = (\{a, b\}, \emptyset, \{a \Rightarrow b, \ b \Rightarrow a\})$ has no derivations. (Thus no justifiable conclusions.)

Example

Defeasible theory $\mathcal{T} = (\{a, b\}, \{\rightarrow a\}, \{a \Rightarrow b, \ b \Rightarrow a\})$ has two derivations:

- $\rightarrow a$ is a derivation for $a$
- $\rightarrow a \preceq a \Rightarrow b$ is a derivation for $b$
- $\rightarrow a \preceq a \Rightarrow b \preceq b \Rightarrow a$ is not a derivation for $a$ (since $\rightarrow a$ already is)
Direct Semantics: Stable Sets
Sets of justified conclusions

Definition (Stable Set)
Let $T = (\mathcal{P}, S, D)$ be a defeasible theory and $M \subseteq \mathcal{L}_\mathcal{P}$ be a possible set for $T$. $M$ is a stable set for $T$ iff for every $z \in M$ there is a derivation of $z$ in $(\mathcal{P}, S, D_M)$.

- Defeasible Theories with (First-Order) Variables: semantics via grounding
Properties of Stable Sets

Stable Set Semantics

- satisfies the rationality postulates of Caminada and Amgoud (2007): direct and indirect consistency, closure.
- is as expressive as propositional logic
- computational complexity:
  - stable set verification is coNP-complete
  - stable set existence is $\Sigma_2^P$-complete
  - credulous reasoning is $\Sigma_2^P$-complete
  - skeptical reasoning is $\Pi_2^P$-complete
Architecture

of our approach

input: CNL text

parse

DRS

translate

defeasible theory

compute

possible/stable sets

output: verbalization

verbalize

Martin Diller, Hannes Strass, Adam Z. Wyner

Defeasible Ace Rules
Currently we have an experimental adaptation of AceRules for our purposes.

- i.e. supports defeasible rules using "It is usual that ..." in a rule.
- [www.dbai.tuwien.ac.at/proj/adf/dAceRules/](http://www.dbai.tuwien.ac.at/proj/adf/dAceRules/)

Interleaves calls to AceRules (and APE) parser, answer set programming (ASP) encodings of direct stable semantics of (and ASP solver), and APE paraphrasing for verbalisation of results.

- Tracks and processes defeasible rules externally.

In (Diller, Wyner, Strass, 2017): extended example of the use of our approach in the context of AceWiki (Kuhn, 2009).

- [attempto.ifi.uzh.ch/acewiki](http://attempto.ifi.uzh.ch/acewiki)

Ongoing work: develop an implementation that does not rely on AceRules.
Problems with AceRules grouping 1

**Input text:**
Bill owns a house.
Bill does not own a car.
If Bill owns a house then he owns an expensive car.
Problems with AceRules grouping 1 (cont.)

**Answer text (AceRules):**
- There is a car X1.
- Bill owns a house.
- Bill owns the car X1.
- The car X1 is expensive.
- It is false that Bill owns a car.
Problems with AceRules grouping 2

Input ACE text:
John owns a car.
The car is red.
Bill does not own a car.
If someone does not own a car then he/she owns a house.

Output:
ERROR: The program violates the atom-restriction.
Problems with AceRules grouping 2 (cont.)

%Extras 1
person(bill).
person(john).
object(a).

%John owns a car.
car(a).
owns(john,a).

%The car is red.
red(a).

%Bill does not own a car.
-owns(bill,X):-car(X).
-car(X):-owns(bill,X).
Problems with AceRules grouping 2 (cont.)

%If someone does not own a car then he/she owns a house.
eap(X):-aon(X).

%Verifies -own(X,Y) \(\lor\) -car(Y)
vaon(X,Y):- ~owns(X,Y),car(Y).
vaon(X,Y):- owns(X,Y),~car(Y).
vaon(X,Y):- ~owns(X,Y),~car(Y).
%Credulous variant:
%vaon(X,Y):- not owns(X,Y),car(Y),person(X).
%...

%For some object Y, -owns(X,Y) \(\lor\) -car(Y) is not verified.
-aon(X) :- not vaon(X,Y), person(X),object(Y).

%-owns(X,Y) \(\lor\) -car(Y) is verified for every object Y.
aon(X) :- not ~aon(X), person(X).
%If someone does not own a car then he/she owns a house.
eap(X):-aon(X).

...

%There is a house that X owns.
house(house(X)):-eap(X).
owns(X,house(X)):-eap(X).

%Extras 2:
object(house(X)):-house(house(X)).
Problems with AceRules grouping 2 (cont.)

**Answerset:**

person(bill) person(john)
object(a) car(a) owns(john,a) red(a)
-owns(bill,a)
-aon(john)
vaon(bill,a)
aon(bill)
eap(bill)
owns(bill,house(bill)) -car(house(bill))
house(house(bill)) object(house(bill))
vaon(bill,house(bill))
Conclusions

Current work:
- We have an approach and prototype for argumentation-inspired reasoning on defeasible ACE rule knowledge bases.

Ongoing work:
- Improve implementation.
  - Turn off grouping / improve grouping ...
- Also have support for justifications.
Future work / speculation:

- Alternative to grouping: target a more expressive rule language.
  - Direct-stable semantics needs to be generalised.

- Generic generalizations without explicit linguistic markers.
  - Lions have manes. Bill walks to work at 9:00 ...

- Generic generalizations as the default, strict rules as the exception?

- Inferring what is defeasible / what not from the knowledge base (similar to anaphora resolution in DRSes)?

- Defeasible rules beyond generic generalizations?
  - Abduction, inferences on the basis of expert opinion..., argument schemes...

- ...
The End