Attempto Controlled English: Language, Tools and Applications

Reasoning in ACE

Norbert E. Fuchs, Kaarel Kaljurand
Department of Informatics & Institute of Computational Linguistics
University of Zurich
{fuchs, kalju}@ifi.unizh.ch
http://attempto.ifi.unizh.ch

December 2006
The Problem

• Is an ACE text consistent (*)?
• Given an ACE text T1 and an additional ACE text or sentence or query T2, does T1 entail T2?

(*) proof-theoretic definition of consistency: a set of first-order formulas is consistent if one cannot derive the false formula ⊥
model-theoretic definition of consistency: a set of first-order formulas is consistent if it has a model
(Non-) Entailments

• given an ACE text T and an additional ACE sentence S there are 4 possibilities

• T entails S
  – S can be deduced from T
  – usually shown by the inconsistency of $T \cup \neg S$
  – if S is a question then S can be answered on the basis of T
  – if S should be added to T then we need not bother
  – S is redundant, is not informative
(Non-) Entailments

• T entails \( \neg S \)
  – T \( \cup \) S is inconsistent
  – if S should be added to T then we must decide to delete the conflicting part of T or not to add S

• T can be split into T1 \( \cup \) T2 and T1 \( \cup \) S entails T2
  – elimination of redundancy or optimisation possible
  – computationally problematic
(Non-) Entailments

• T and S are not related by entailment
  – neither S nor ¬S follow from T
  – question S cannot be answered
  – if S should be added to T we can safely do so
Requirements for the ACE Reasoner RACE

- all input and output in ACE
- generate all proofs
- give a user-friendly justification of a proof
- allow for auxiliary first-order axioms to express background knowledge that cannot (easily) be expressed in ACE
- interface to evaluable functions and predicates
- combine theorem proving with model generation
- hide internal working from casual user
A Basis for RACE

• development of a new theorem prover for DRSs
  – DRS deduction rules (Gabbay & Reyle)
  – correct & complete
  – no efficient proof strategy

• off-the-shelf first-order theorem provers and model generators as potential basis
  – leanTAP
  – EP Tableaux
  – Otter & Mace
  – do not satisfy all requirements
A Basis for RACE

• Satchmo (Manthey & Bry 1988)
  – basically a model generator
  – can also be used a theorem prover
  – uses first-order clauses Body --> Head
  – generates minimal finite Herbrand models of clauses (if existent)
  – correct for unsatisfiability of range-restricted clauses
  – complete for unsatisfiability if used level-saturated
  – efficient Prolog core allowing ...
  – ... local extensions and modifications in Prolog
Original Satchmo

satisfiable :-
    setof(Clause, violated_instance(Clause), Clauses), !,
    satisfy_all(Clauses), satisfiable.
satisfiable.

violated_instance((B ---> H)) :-
    (B ---> H), B, \+ H.

satisfy_all([]).
satisfy_all([(_B ---> H) | RestClauses]) :-
    H, !,
    satisfy_all(RestClauses).
satisfy_all([(_B ---> H) | RestClauses]) :-
    satisfy(H),
    satisfy_all(RestClauses).

clauses: B ---> H
B conjunction of atoms or true
H disjunction of Atoms or false
no explicit negation

satisfy((A;B)) :-
    !,
    (satisfy(A) ; satisfy(B)).
satisfy(Atom) :-
    \+ Atom = false,
    assume(Atom).
assume(Atom) :-
    asserta(Atom).
assume(Atom) :-
    retract(Atom), !,
    fail.
RACE Extensions of Satchmo

• RACE extensions should preserve Satchmo's correctness, completeness and efficiency

• RACE generates all proofs
  – Satchmo stops immediately if it detects unsatisfiability
  – RACE finds all minimal unsatisfiable subsets of clauses

• RACE gives a justification of every proof
  – RACE collects indices of logical atoms used for a proof
  – RACE generates for each proof a report showing which ACE sentences were used to derive which ACE query

• input and output in ACE
Structure of RACE

\[ \text{ACE Text } T \quad \vdash \quad \text{ACE Query } Q \]

\[ \text{DRS}_T \quad \downarrow \quad \text{Clauses } C_T \quad \rightarrow \quad \text{FOL}_T \]

\[ \text{DRS}_Q \quad \downarrow \quad \text{Clauses } C_Q \quad \leftarrow \quad \text{FOL}_Q \]

\[ C_T \cup \neg C_Q \vdash \bot \]

Proof Justification
RACE Detects Entailments

Text

Every company that buys a standard machine gets a discount.
A British company buys a standard machine.
A French company buys a special machine.

Query

A company gets a discount.

RACE proved that the sentence(s)
A company gets a discount.
can be deduced from the sentence(s)
Every company that buys a standard machine gets a discount.
A British company buys a standard machine.
RACE Answers Questions

Text  
Every company that buys a standard machine gets a discount.
A British company buys a standard machine.
A French company buys a special machine.

Query  
Who buys a machine?

1. RACE proved that the query (-ies)  
   Who buys a machine?
   can be answered on the basis of the sentence(s)  
   A British company buys a standard machine.

2. RACE proved that the query (-ies)  
   Who buys a machine?
   can be answered on the basis of the sentence(s)  
   A French company buys a special machine.
RACE Detects Inconsistencies

Text  Every company that buys a standard machine gets a discount.
A British company buys a standard machine.
A French company buys a standard machine.
There is no company that gets a discount.

1. RACE proved that the sentence(s)
   Every company that buys a standard machine gets a discount.
   A British company buys a standard machine.
   There is no company that gets a discount.
   are inconsistent.

2. RACE proved that the sentence(s)
   Every company that buys a standard machine gets a discount.
   A French company buys a standard machine.
   There is no company that gets a discount.
   are inconsistent.
Auxiliary Axioms for RACE

• ACE can express plural sentences
  *Six Swiss companies buy a machine.* (collective)
  *Each of six Swiss companies buys a machine.* (distributive)

• first-order representation of plurals
  – additional group objects with lattice-theoretic structure
  – requires additional (domain-independent) knowledge
    • auxiliary axioms for lattice-theory, numbers, equality, …
    • evaluable functions and predicates

• flat notation allows us to integrate axioms in FOL
Proof with Auxiliary Axiom

ACE Text
Every company that buys a machine gets a discount.
Each of six Swiss companies buys a machine.

ACE Query
A company gets a discount.

DRS_T [A, …]
object(A,group,...)-2 …

DRS_Q [B, …]
object(B,atomic,...)-1 …

FOL_T
∃A(object(A,group,...) ∧ …)

FOL_Q
∃B(object(B,atomic,...) ∧ …)

FOL_{Ax}
∀X(object(X,group,...) →
∃Y(object(Y,atomic,...) ∧ part_of(Y,X))

Proof Justification
Entailment Example with Axioms

Text  

Every company that buys a machine gets a discount.  
Each of six Swiss companies buys a machine.

Query  

A company gets a discount.

RACE proved that the sentence(s)

A company gets a discount.

can be deduced from the sentence(s)

Every company that buys a machine gets a discount.  
Each of six Swiss companies buys a machine.

using the auxiliary axiom(s)

(Ax. 9): Definition of proper_part_of.  
(Ax. 10-1): Every group consists of atomic parts.  
(Ax. 22-1): Number Axiom.
# Question Answering with Axioms

| Text | Each of six Swiss companies buys a machine.  
A German company buys a special machine. |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Query</td>
<td>Who buys machines?</td>
</tr>
</tbody>
</table>

1. RACE proved that the query `-ies`  
   *Who buys machines?*  
   can be answered on the basis of the sentence(s)  
   *Six Swiss companies each buy a machine.*  
   using the FOL axiom(s)  
   (Ax. 10-2): Groups have atomic parts.  
   (Ax. 2): atomic => dom  
   (Ax. 9): Definition of proper_part_of.  
   (Ax. 11): Atoms have no proper parts.  
   (Ax. 15-1): Identity axiom for objects.  
   (Ax. 22-1): Number Axiom.  

2. RACE proved that the query `-ies`  
   *Who buys machines?*  
   can be answered on the basis of the sentence(s)  
   *A German company buys a special machine.*  
   using the FOL axiom(s)  
   (Ax. 2): atomic => dom  
   (Ax. 11): Atoms have no proper parts.  
   (Ax. 15-1): Identity axiom for objects.  
   (Ax. 22-1): Number Axiom.
Example Auxiliary Axioms

- internal representation: fol_axiom(Index, Axiom, Text)
- groups have atomic parts
  \[ \forall X (\text{object}(X, \text{group}, \ldots) \rightarrow \exists Y (\text{object}(Y, \text{atomic}, \ldots) \land \text{part_of}(Y, X))) \]
- atoms have no proper parts.
  \[ \forall X \forall Y (\text{object}(X, \text{atomic}, \ldots) \land \text{part_of}(X, Y) \rightarrow \text{is_equal}(X, Y)) \]
- identity axiom for objects
  \[ \forall X \forall Y \forall P (\text{object}(X, P) \land \text{is_equal}(X, Y) \rightarrow \text{object}(Y, P)) \]
- problems: Satchmo core offers no equality reasoning, some axioms cause inefficiency
Add Evaluable Prolog Predicates

• problems involving natural numbers …

Text

Every company that buys at least three machines gets a discount. A German company buys four machines.

Query

A company gets a discount.

• … require knowledge about natural numbers, but ...

∀X∀C(object(X,...,C,eq,4) ⇒ object(X,...,,C,geq,3))

• ... it is much better to use Prolog instead

object(A,Structure,object,Type,cardinality,count_unit,geq,NewN):-number(NewN),
object(A,Structure,object,Type,cardinality,count_unit,eq,GivenN),
number(GivenN),
NewN =< GivenN.
Evaluation of RACE

• example: Steamroller

<table>
<thead>
<tr>
<th>Representation</th>
<th>Standard</th>
<th>Attempto DRS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Satchmo (original)</td>
<td>15 ms</td>
<td>1990 ms</td>
</tr>
<tr>
<td>RACE</td>
<td>70 ms</td>
<td>375 ms</td>
</tr>
</tbody>
</table>

• open problems
  – finding *all* solutions increases search space
  – scalability, robustness to be shown
Current & Future Research

• improve question answering using Satchmo models
• extensions to support modality
• hypothetical reasoning (“What happens if …?”)
• abductive reasoning (“Under which conditions …?”)
• temporal reasoning