Attempto Controlled English and Its Tools

Norbert E. Fuchs

Department of Informatics & Institute of Computational Linguistics
University of Zurich

fuchs@ifi.uzh.ch
http://attempto.ifi.uzh.ch

MOLTO Extended Kick-Off Meeting

Gothenburg, 12-13 January 2012

Overview

- Attempto Controlled English (ACE)
- ACE Tools
- Some Applications of ACE
- Related Work
- Some Links

Attempto Controlled English (ACE)

- ACE is a *controlled* natural language
  - precisely defined subset of full English
  - unambiguously translatable into logic languages
- ACE is human *and* machine understandable
  - ACE seems completely natural, but is a formal language
  - ACE is a logic language with an English syntax
- ACE *combines* natural language with formal methods
  - provably easier to learn and to use than visibly formal languages
  - automated reasoning with ACE via existing tools

The Language ACE

- vocabulary
  - predefined function words (articles, conjunctions, …)
  - predefined phrases *(there is a …, it is false that …)*
  - user-defined content-words (nouns, verbs, adjectives, adverbs)
  - basic lexicon (100'000 words), optionally user-defined lexicons
  - unknown words: guessing, prefixing with word class
- construction rules
  - define admissible sentence structures
  - avoid many ambiguous or imprecise constructions
- interpretation rules
  - control logical analysis of admissible sentences
  - deterministically resolve remaining ambiguities

Example Construction Rules

- proper names
- common nouns
  - countable singular
    a customer, the customer, every customer, ...
  - countable collective plural
    2 customers, some customers, ...
  - countable distributive plural
    each of two customers, ...
  - mass
    some water, no water, all water
- modifications: adjectives, genitives, relative phrases, variables

Example Construction Rules

- verbs
  - intransitive, transitive, ditransitive, copula (*to be*)
  - third person singular & plural
  - indicative, simple present, active and passive
  - modality (*can*, *must, should, may*)
  - sentence subordination (*believe that ..., wants to ...*)
  - logical negation & negation as failure
- modifications: adverbs, prepositional phrases

John promises Mary to run. = John promises Mary that he runs.

John forces Mary to run. = John forces Mary that she runs.

Example Construction Rules

- simple declarative sentences

  A customer waits.

  A customer carefully inserts a valid card into a slot.

  There is a valid card.

- composite sentences are built with predefined constructors

  If a customer inserts a card that is valid then the automatic-teller accepts the card and displays a message.

- interrogative sentences

  Does a customer wait?

  Who inserts what?

Note: Colouring and – on later slides – {bracketing} are not part of ACE.

Example Interpretation Rules

- prepositional phrases modify the verb not the noun phrase

  A customer {enters a card with a code}.

- relative clauses modify the immediately preceding noun phrase

  A customer enters {a card that has a code}.

- surface position of a quantifier determines its relative scope

  {A customer enters {every card}}. $\exists\forall$

  {Every customer enters {a card}}. $\forall\exists$

- anaphora: most recent, most specific, accessible noun phrase

  John has a customer. John inserts his card and types a code X. Bill sees X. He inserts his own card and types the code.

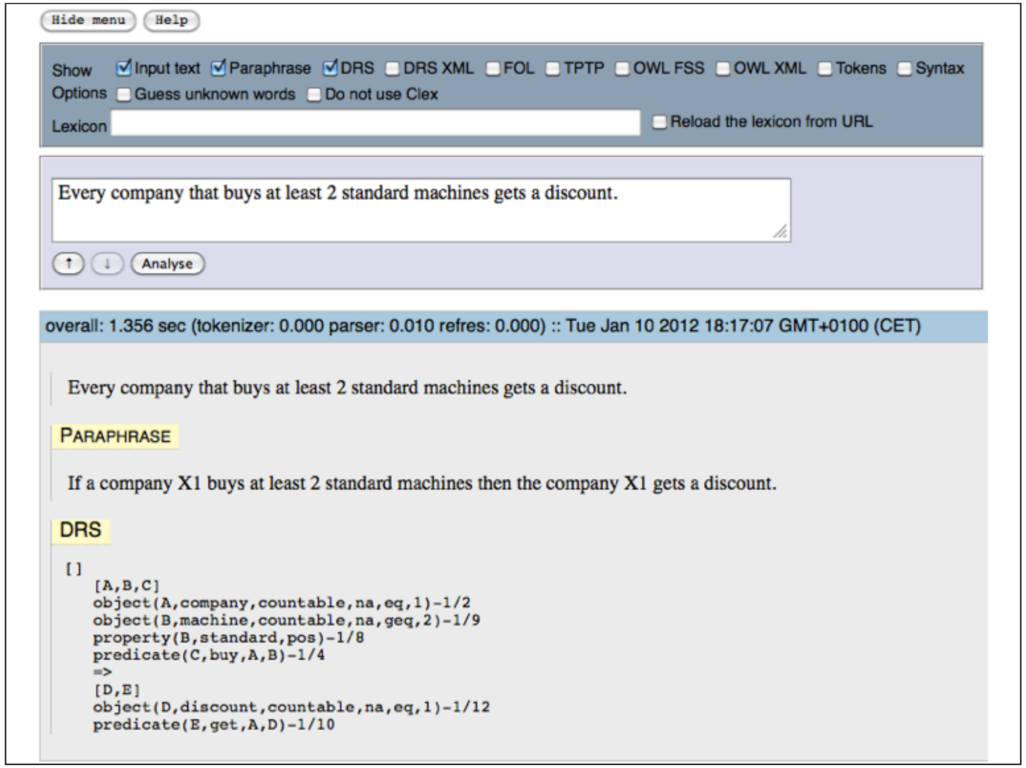
ACE Tools

- Attempto Parsing Engine (APE)
- ACE Editor
- AceWiki
- ACE Reasoner (RACE)
- ACE View Protégé Plug-in
- AceRules
- all ACE tools are fully documented and have web-interfaces
- most ACE tools are freely available

Attempto Parsing Engine (APE)

- APE
  - checks the syntax of ACE texts
  - implements construction and interpretation rules
  - Definite Clause Grammar enhanced with feature structures
  - accessible via web-service & web-interface
- for syntactically correct texts optionally outputs the analysis of the text
  - tokens & syntax tree
  - logical form (DRS and other logical languages)
  - paraphrase in ACE (generated from DRS)
- for erroneous texts
  - detects syntax errors and unknown words
  - generates error and warning messages indicating the location and the possible causes of the errors, and suggesting remedies

ACE paraphrase: Core ACE, NP ACE


Hide menu
Help
Show
Input text
Paraphrase
DRS
DRS XML
FOL
TPTP
OWL FSS
OWL XML
Tokens
Syntax
Options
Guess unknown words
Do not use Clex
Lexicon
Reload the lexicon from URL
Every company that buys at least 2 standard machines gets a discount.
Analyse
overall: 1.356 sec (tokenizer: 0.000 parser: 0.010 refres: 0.000) :: Tue Jan 10 2012 18:17:07 GMT+0100 (CET)
Every company that buys at least 2 standard machines gets a discount.
PARAPHRASE
If a company X1 buys at least 2 standard machines then the company X1 gets a discount.
DRS
[]
[A,B,C]
object(A,company,countable,na,eq,1)-1/2
object(B,machine,countable,na,geq,2)-1/9
property(B,standard,pos)-1/8
predicate(C,buy,A,B)-1/4
=>
[D,E]
object(D,discount,countable,na,eq,1)-1/12
predicate(E,get,A,D)-1/10

Pretty Printed Example DRS

Every company that buys at least 2 standard machines gets a discount.

```
[]
  [A, B, C]
  object(A, company, countable, na, eq, 1)-1/2
  object(B, machine, countable, na, geq, 2)-1/9
  property(B, standard, pos)-1/8
  predicate(C, buy, A, B)-1/4
  =>
  [D, E]
  object(D, discount, countable, na, eq, 1)-1/12
  predicate(E, get, A, D)-1/10
```

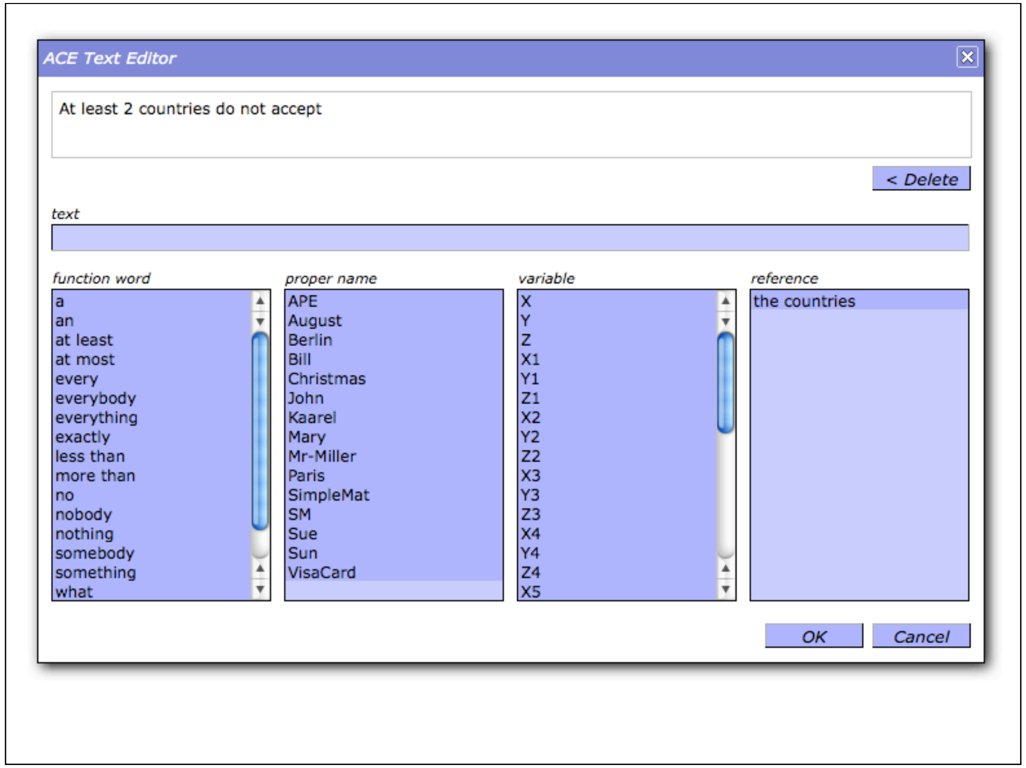
Solving the Writability Problem

- APE requires users to learn and to recall the ACE construction and interpretation rules
- ACE Editor is a predictive editor that helps users to construct syntactically and lexically correct ACE texts by just clicking on words and word classes
- ACE Editor is also used in AceWiki

ACE Editor

- solving the writability problem
  - ACE Editor is a predictive editor that helps users to construct correct ACE texts by just clicking on words and word classes
  - learning and recalling ACE construction rules is not necessary
  - ACE Editor also suggests anaphoric references
  - alternatively users can freely input ACE texts
  - prediction is controlled by a separate ACE grammar (CODECO)
  - ACE Editor submits completed text to APE for parsing

- getting the intended interpretation
  - ACE Editor relies also on APE's feedback, but additionally …
  - … provides colour-coded syntax boxes reflecting the interpretation rules


ACE Text Editor
At least 2 countries do not accept
< Delete
text
function word
a
an
at least
at most
every
everybody
everything
exactly
less than
more than
no
nobody
nothing
somebody
something
what
proper name
APE
August
Berlin
Bill
Christmas
John
Kaarel
Mary
Mr-Miller
Paris
SimpleMat
SM
Sue
Sun
VisaCard
variable
X
Y
Z
X1
Y1
Z1
X2
Y2
Z2
X3
Y3
Z3
X4
Y4
Z4
X5
reference
the countries
OK
Cancel

At least 2 countries do not accept somebody without an address.

Paraphrase

```
There are at least 2 countries X1.
It is false that the countries X1 accept somebody without an address.
```

Sentences

```
['At least 2 countries do not accept somebody without an address.']
```

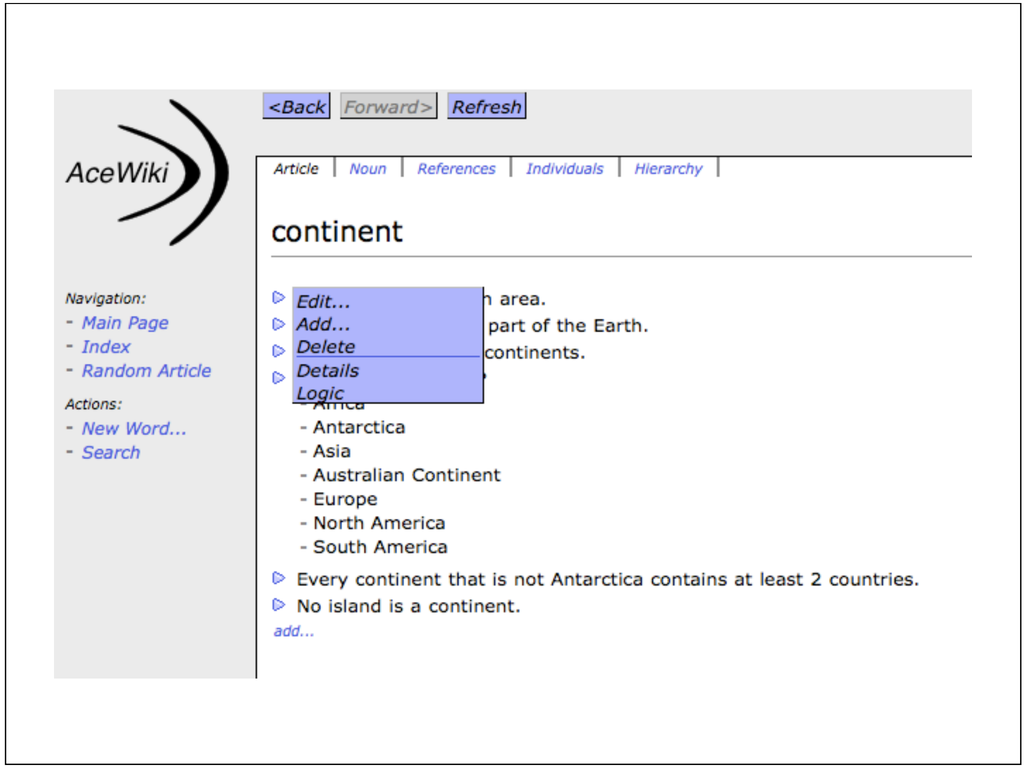
Syntax Boxes

at least 2 countries | do not accept | somebody | without | an address | .

AceWiki

- shortcomings of many existing semantic wikis
  - hard to understand for people who are not familiar with formal languages
  - relatively inexpressive (mostly subject-predicate-object structures)
- AceWiki offers an alternative
  - uses ACE to express wiki articles
  - articles are formal but still readable by people
  - ACE covers a large part of FOL and is highly expressive
  - collaborative ontology management in ACE
- Kaarel's talk tomorrow will present the latest developments of AceWiki


<Back
Forward>
Refresh
AceWiki
Article
Noun
References
Individuals
Hierarchy
continent
Every continent is an area.
Every continent is a part of the Earth.
There are exactly 7 continents.
What is a continent?
- Africa
- Antarctica
- Asia
- Australian Continent
- Europe
- North America
- South America
Every continent that is not Antarctica contains at least 2 countries.
No island is a continent.
add...
Navigation:
- Main Page
- Index
- Random Article
Actions:
- New Word...
- Search



<Back
Forward>
Refresh
AceWiki
Article
Noun
References
Individuals
Hierarchy
continent
Navigation:
- Main Page
- Index
- Random Article
Actions:
- New Word...
- Search
Edit...
Add...
Delete
Details
Logic
area.
part of the Earth.
continents.
- Antarctica
- Asia
- Australian Continent
- Europe
- North America
- South America
Every continent that is not Antarctica contains at least 2 countries.
No island is a continent.
add...

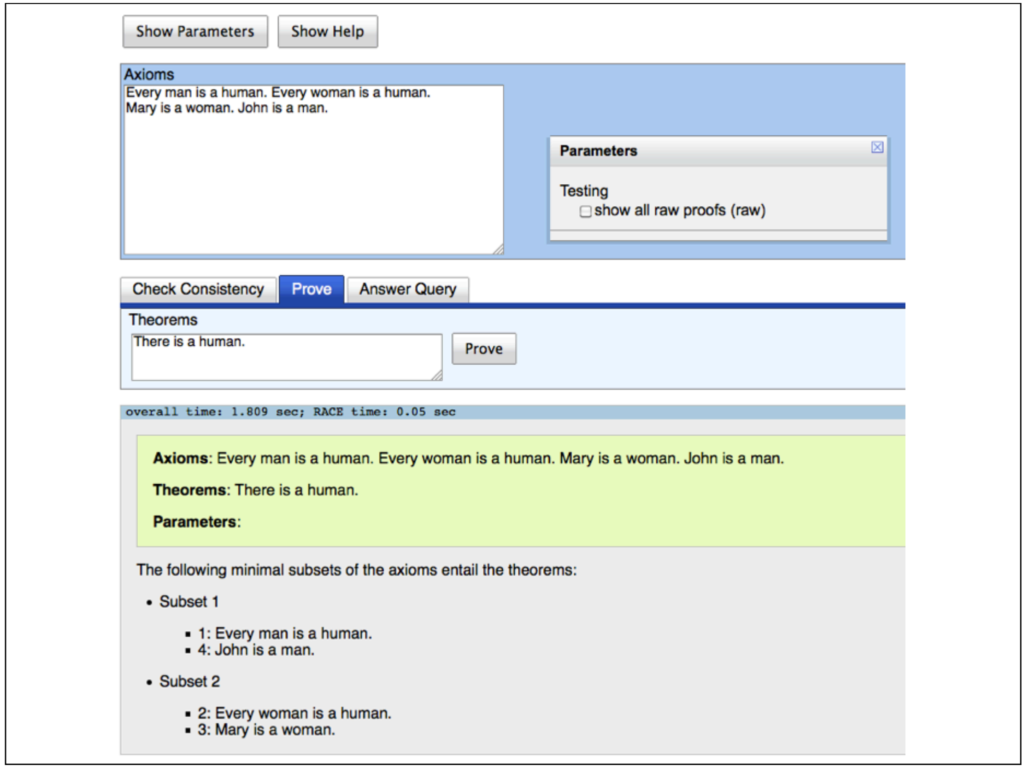
<Back
Forward>
Refresh
AceW
Navigatio
- Main
- Index
- Rand
Actions:
- New
- Searc
Sentence Editor
Every continent is an area
< Delete
text
function word
.
and
or
that
new variable
X
Y
Z
X1
Y1
Z1
X2
Y2
Z2
X3
Y3
Z3
OK
Cancel

AceWiki Reasoning

- AceWiki currently uses the OWL reasoner FaCT++
- AceWiki marks sentences that make a text inconsistent
- ACE sentences that cannot be translated to OWL do not take part in reasoning
- AceWiki can answer questions
- AceWiki can infer class membership and hierarchies

Attempto Reasoner (RACE)

- RACE performs first-order deductions on ACE texts
- basic proof procedure: if an ACE text (= set of sentences) is inconsistent then RACE identifies all minimal inconsistent subsets
- variants of the basic proof procedure allow RACE to
  - prove that one ACE text (axioms) entails another ACE text (theorems)
  - answer ACE queries on the basis of an ACE text
- RACE provides a proof justification in ACE and full English
- RACE finds all proofs
- RACE uses domain-independent auxiliary axioms to reason about plurals, natural numbers, equality etc.


Show Parameters
Show Help
Axioms
Every man is a human. Every woman is a human.
Mary is a woman. John is a man.
Parameters
Testing
show all raw proofs (raw)
Check Consistency
Prove
Answer Query
Theorems
There is a human.
Prove
overall time: 1.809 sec; RACE time: 0.05 sec
Axioms: Every man is a human. Every woman is a human. Mary is a woman. John is a man.
Theorems: There is a human.
Parameters:
The following minimal subsets of the axioms entail the theorems:
Subset 1
1: Every man is a human.
4: John is a man.
Subset 2
2: Every woman is a human.
3: Mary is a woman.

Why? Why Not?

- Why?
  - for a succeeding proof RACE answers the question "why?" by listing the axioms needed for the proof
- Why Not?
  - for a failing proof RACE answers the question "why not?" by listing those parts of the theorem or query that could not be proved

```
overall time: 1.455 sec; RACE time: 0 sec
```

Axioms: Mary sees a cat on a farm.

Theorems: Mary believes that a farmer owns a cat.

Parameters:

Theorems do not follow from axioms.

The following parts of the theorems/query could not be proved:

- transitive verb: own
- countable common noun: (at least 1) farmer
- modal operator or sentence subordination
- transitive verb: believe

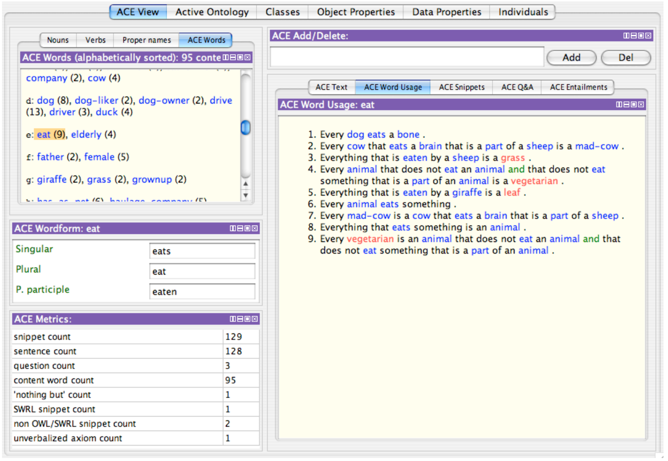
ACE View

- idea: use ACE as an alternative syntax for OWL and related languages
- ACE View
  - ontology and rule editor
  - uses ACE for the user interface
  - creates, views and edits OWL 2 ontologies and SWRL rulesets
- implemented as plug-in for the Protégé ontology editor

ACE View

- ontology and rule editing using ACE
- input can be actually full English, but only OWL/SWRL-compatible sentences participate in reasoning
- "semantic feedback" in ACE
  - entailments
  - entailment explanation
  - query-answering
- implementation
  - integrates translators ACE→OWL/SWRL and OWL→ACE
  - implemented as a plug-in for Protégé 4 …
  - … which makes it easy to switch between the "ACE View" and the traditional "Protégé view"

ACE View: Various Views

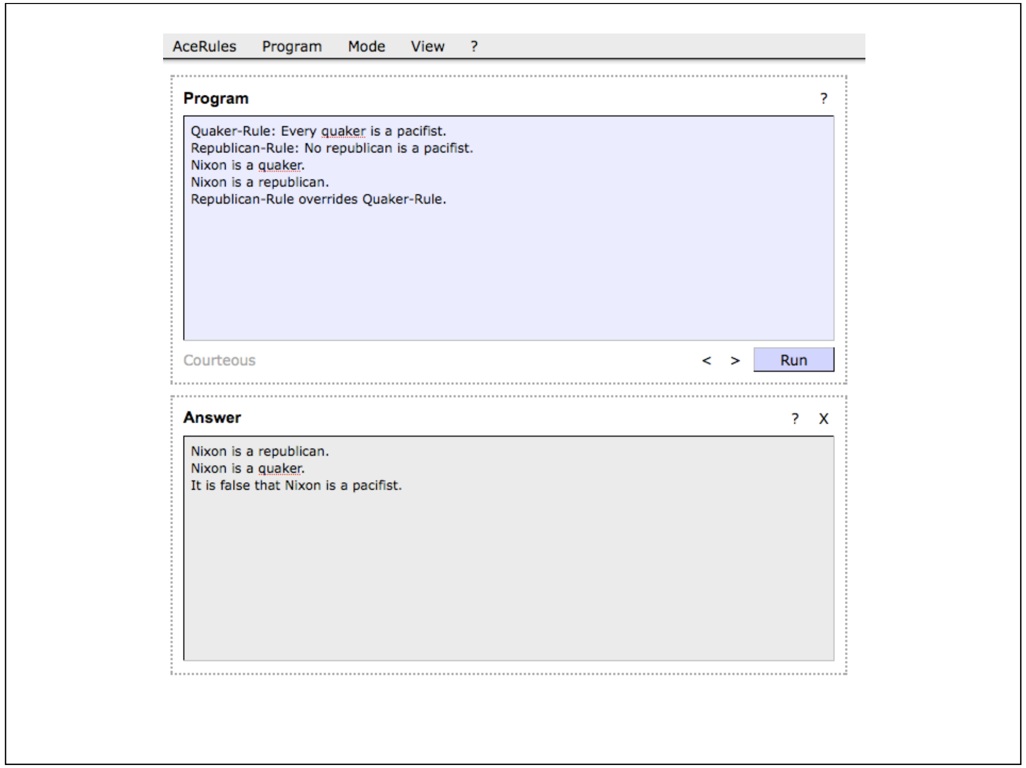


AceRules

- domain specialists that are supposed to create and/or validate rules are often not familiar with formal languages
- verbalisation of the rules in natural language becomes necessary
- translation of rules into NL (and backwards) is complicated and a potential source of errors
- AceRules offers an alternative
  - expresses rules in ACE
  - rules expressed in ACE are formal and still readable by humans

AceRules Interpreter

- AceRules uses forward-reasoning
- semantics of rules is exchangeable
- currently supported semantics
  - courteous logic programming
  - stable models
  - stable models with strong negation


AceRules
Program
Mode
View
?
Program
Quaker-Rule: Every quaker is a pacifist.
Republican-Rule: No republican is a pacifist.
Nixon is a quaker.
Nixon is a republican.
Republican-Rule overrides Quaker-Rule.
Courteous
Run
Answer
Nixon is a republican.
Nixon is a quaker.
It is false that Nixon is a pacifist.

Some Applications of ACE

- *specifications*: automated teller, Kemmerer's library data base, Schubert's Steamroller, data base integrity constraints, Kowalski's subway regulations
- *natural language interfaces*: model generator EP Tableaux (Munich), FLUX agent control (Dresden), MIT's process query language (Zurich), signal processing (Weizman)
- *medicine*: doctors' reports (Uppsala), clinical practice guidelines (Yale)
- *rules*: AceRules, policy rules (EU REWERSE), grammar rules (Jones)
- *translation into and partially from semantic web languages*: OWL, SWRL, RuleML (New Brunswick), R2ML (Cottbus)
- *semantic web tools*: ACE View plug-in for Protégé, AceWiki
- *documentation*: annotations of web-pages in controlled natural language (Macquarie)

Related Work

- web-sites
  - http://sites.google.com/site/controllednaturallanguage/
  - http://en.wikipedia.org/wiki/Controlled_natural_language
- small selection of other controlled natural languages
  - Bernardi, Calvanese, Thorne (Bozen-Bolzano): Lite Natural Language
  - Pulman et al. (Oxford): Computer Processable Controlled Language
  - Schwitter (Macquarie): Processable English (PENG)
  - Sowa (VivoMind): Common Logic Controlled English
  - Hart et al. (Ordnance Survey): Rabbit
  - Clark et al. (Boeing): Computer-Processable Language (CPL)
  - ...

CPL: Parsing is performed using SAPIR, a mature, bottom-up, broad coverage chart parser. Six times as many error rules as parsing rules.

Attempto Website

- dedicated Attempto server
- news feed
- mailing list
- documentation
- publications
- talks, courses, screen-casts
- demos of ACE tools
- download of tools (GNU Lesser General Public Licence), but we are now moving to GitHub
- workshop CNL 2012

CNL
2012

Third Workshop on Controlled Natural Language (CNL 2012)

29–31 August 2012
Zurich, Switzerland

More information at attempto.ifi.uzh.ch/site/cnl2012

Appendix

Extended DRS

- Kamp & Reyle: DRS can be translated into standard FOL
- extended DRS covers all of ACE
  - declarative sentence: derived DRS can be translated into FOL
  - interrogative sentence: in DRS represented as label & declarative sentence with query conditions; can be translated into FOL
  - modality (*can*, *must*) and sentence subordination: in DRS represented as label & declarative sentence; can be translated into FOL using possible-world semantics (Bos)
  - modality (*should*, *may*) and negation as failure: in DRS represented as label & declarative sentence; cannot be translated into FOL
- in the sequel I will focus on the 'FOL subset' of ACE

Properties of DRS Representation

Only Predefined Relation Symbols

Every company that buys at least 2 standard machines gets a discount.

```
[]
  [A, B, C]
  object(A, company, countable, na, eq, 1)-1/2
  object(B, machine, countable, na, geq, 2)-1/9
  property(B, standard, pos)-1/8
  predicate(C, buy, A, B)-1/4
  =>
  [D, E]
  object(D, discount, countable, na, eq, 1)-1/12
  predicate(E, get, A, D)-1/10
```

Properties of DRS Representation

Predicates as Arguments

Every company that buys at least 2 standard machines gets a discount.

```
[]
  [A, B, C]
  object(A, company, countable, na, eq, 1)-1/2
  object(B, machine, countable, na, geq, 2)-1/9
  property(B, standard, pos)-1/8
  predicate(C, buy, A, B)-1/4
  =>
  [D, E]
  object(D, discount, countable, na, eq, 1)-1/12
  predicate(E, get, A, D)-1/10
```

Properties of DRS Representation

Quantity Information

Every company that buys at least 2 standard machines gets a discount.

```
[]
  [A, B, C]
  object(A, company, countable, na, eq, 1)-1/2
  object(B, machine, countable, na, geq, 2)-1/9
  property(B, standard, pos)-1/8
  predicate(C, buy, A, B)-1/4
  =>
  [D, E]
  object(D, discount, countable, na, eq, 1)-1/12
  predicate(E, get, A, D)-1/10
```

Properties of DRS Representation

Token Indices

Every company that buys at least 2 standard machines gets a discount.

```
[]
  [A, B, C]
  object(A, company, countable, na, eq, 1)-1/2
  object(B, machine, countable, na, geq, 2)-1/9
  property(B, standard, pos)-1/8
  predicate(C, buy, A, B)-1/4
  =>
  [D, E]
  object(D, discount, countable, na, eq, 1)-1/12
  predicate(E, get, A, D)-1/10
```