First-Order Reasoning for Attempto Controlled English

Norbert E. Fuchs
Department of Informatics & Institute of Computational Linguistics
University of Zurich
fuchs@ifi.uzh.ch
http://attempto.ifi.uzh.ch

Second Workshop on Controlled Natural Language CNL 2010
Marettimo Island, 13-15 September 2010

Four Approaches for Reasoning in Attempto Controlled English (ACE)

- Tobias Kuhn's AceRules is a forward-chaining rule system that calculates the complete answer set from rules and facts expressed in a subset of ACE that includes logical negation and negation-as-failure.

- Automatically translate (a subset of) the ACE text into OWL/SWRL and use one of the reasoners for OWL/SWRL.

  The ACE → OWL/SWRL translation is offered by the Attempto Parsing Engine APE.

  This approach is used by Kaarel Kaljurand's ACE View and Tobias Kuhn's AceWiki.

- Automatically translate the ACE text into Sutcliffe's TPTP notation and submit it to one of the reasoners available in the TPTP system.

  The ACE → TPTP translation is offered by the Attempto Parsing Engine APE and by the TPTP web-interface.

  On the basis of this approach Nelson Dellis developed CNL-WKR that is able to answer ACE questions from external knowledge sources.

- Use the Attempto Reasoner RACE – and this is what I will talk about.

Contents

Requirements for RACE

Requirements for the Attempto Reasoner RACE

- RACE should eventually cover the first-order subset of ACE, that is all ACE constructs with the exception of *imperative sentences*, *negation-as-failure* and the modal operators *may* and *should*.
- All input and output of RACE should be in ACE or in natural language.
- RACE should give user-friendly justifications for proofs.
- RACE should hide its internal working from the casual user.
- RACE should generate all proofs.
- RACE should allow for auxiliary axioms to express background knowledge that cannot at all, or not easily, be expressed in ACE.
- There should be an interface to evaluable functions, e.g. arithmetic.
- RACE should combine theorem proving with model generation.

From the Model Generator Satchmo to RACE

Satchmo as Basis for RACE

Satchmo (Manthey & Bry 1988)

- basically a model generator, can also be used as theorem prover
- uses first-order clauses *Body* → *Head*
  - *Body* is *true* or a conjunction of logical atoms, *Head* is *false* or a disjunction of logical atoms
  - no explicit negation, instead implication to *false* ($\neg A \equiv A \rightarrow false$)
- Satchmo generates a minimal finite Herbrand model of the clauses (if one exists)
- is correct for unsatisfiability if the clauses are range-restricted
- is complete for unsatisfiability if used level-saturated
- efficient Prolog implementation allowing for
  - local extensions and modifications
  - direct calls of Prolog predicates (user-defined, built-in, library)

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).

satisfy((A;B)) :-
  !,
  (satisfy(A) ; satisfy(B)).

satisfy(Atom) :-
  \+ Atom = fail,
  assume(Atom).

assume(Atom) :-
  asserta(Atom).
assume(Atom) :-
  retract(Atom),
  !,
  fail.
```

RACE Extensions of Satchmo

RACE is based on Satchmo and contains many extensions and modifications to satisfy the requirements

- RACE's extensions should preserve Satchmo's correctness, its completeness and – as far as possible – its efficiency
- for satisfiable clauses both Satchmo and RACE generate a minimal finite Herbrand model
- for unsatisfiable clauses
  - Satchmo stops immediately if it detects unsatisfiability
  - RACE finds *all minimal unsatisfiable subsets* of the clauses
- proof justification
  - Satchmo just succeeds or fails, meaning that the clauses are satisfiable or not satisfiable
  - RACE generates for each proof a report showing which *minimal subsets* of the ACE axioms were used to deduce the ACE theorem
- input and output
  - Satchmo works on clauses given as Prolog facts and has no output
  - RACE translates ACE axioms and theorems into clauses and outputs the results in ACE

Overview of RACE

Executive Summary of RACE

- RACE performs 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
- RACE finds all proofs
- RACE is running on the dedicated Attempto server and can be accessed via a web-service or a web-interface

From ACE to Satchmo Clauses

- ACE text

  John is a man. Every man is a human. No cat is a human.

- Attempto Parsing Engine APE translates ACE text into DRS (here pretty-printed)

```
[A, B]
predicate(B, be, named(John), A)-1/2
object(A, man, countable, na, eq, 1)-1/4
  [C]
  object(C, man, countable, na, eq, 1)-2/2
  =>
  [D, E]
  object(D, human, countable, na, eq, 1)-2/5
  predicate(E, be, C, D)-2/3
  [F]
  object(F, cat, countable, na, eq, 1)-3/2
  =>
  []
    NOT
    [G, H]
    object(G, human, countable, na, eq, 1)-3/5
    predicate(H, be, F, G)-3/3
```

From ACE to Satchmo Clauses

- standard translation of DRS into FOL statement

  exists(A, exists(B, &(object(A, man, countable, na, eq, 1)-1/4, &(predicate(B, be, named('John'), A)-1/2, &(forall(C, =>(object(C, man, countable, na, eq, 1)-2/2, exists(D, exists(E, &(object(D, human, countable, na, eq, 1)-2/5, predicate(E, be, C, D)-2/3))))), forall(F, =>(object(F, cat, countable, na, eq, 1)-3/2, -(exists(G, exists(H, &(object(G, human, countable, na, eq, 1)-3/5, predicate(H, be, F, G)-3/3))))))))))

- variant of standard translation of FOL statement into (Satchmo) clauses

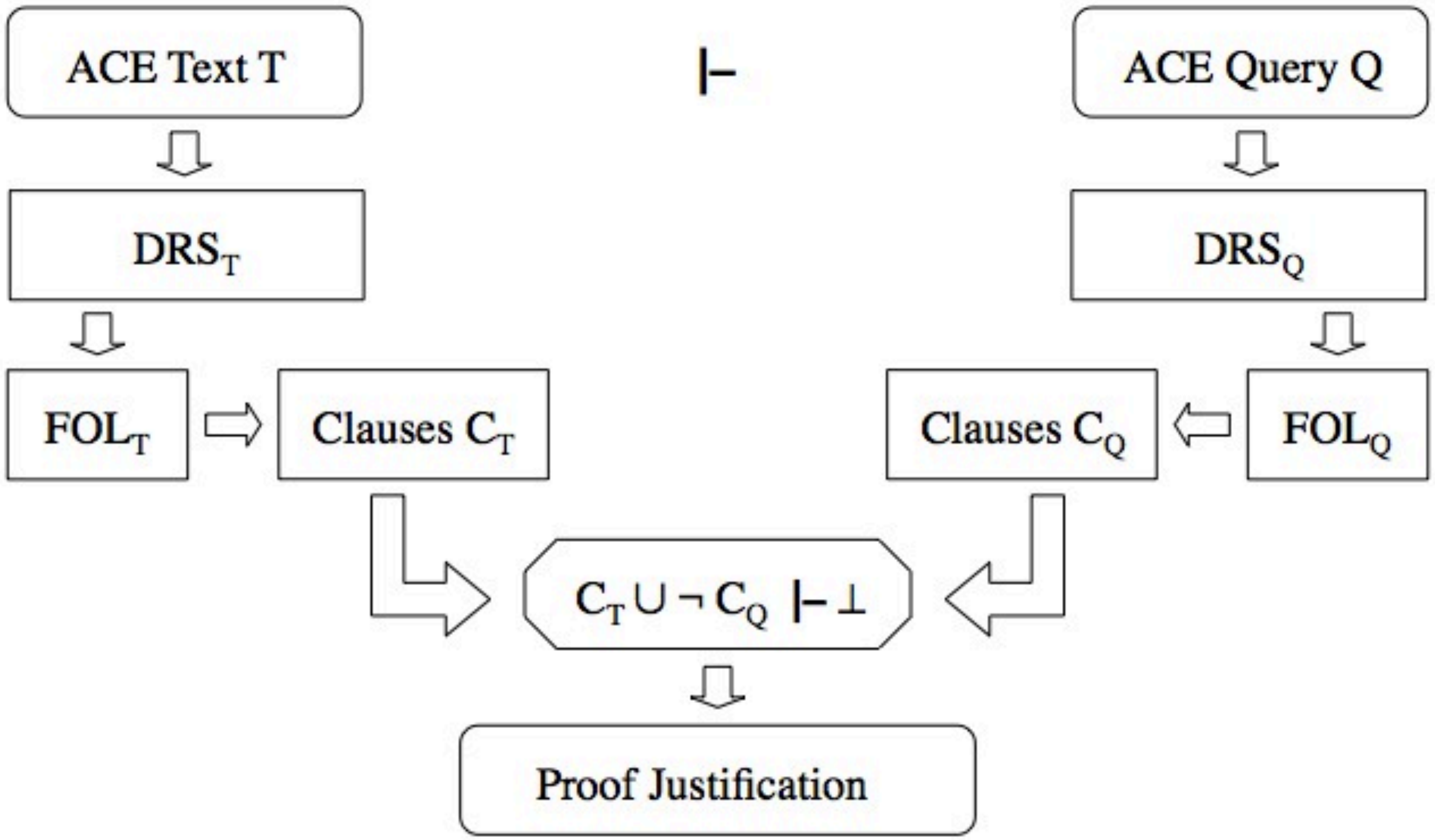
  satchmo_clause(true, predicate(sk1, sk3, be_NP, named('John'), sk2), [axiom(1)])

  satchmo_clause(true, object(sk1, sk2, man, countable, na, eq, 1), [axiom(1)])

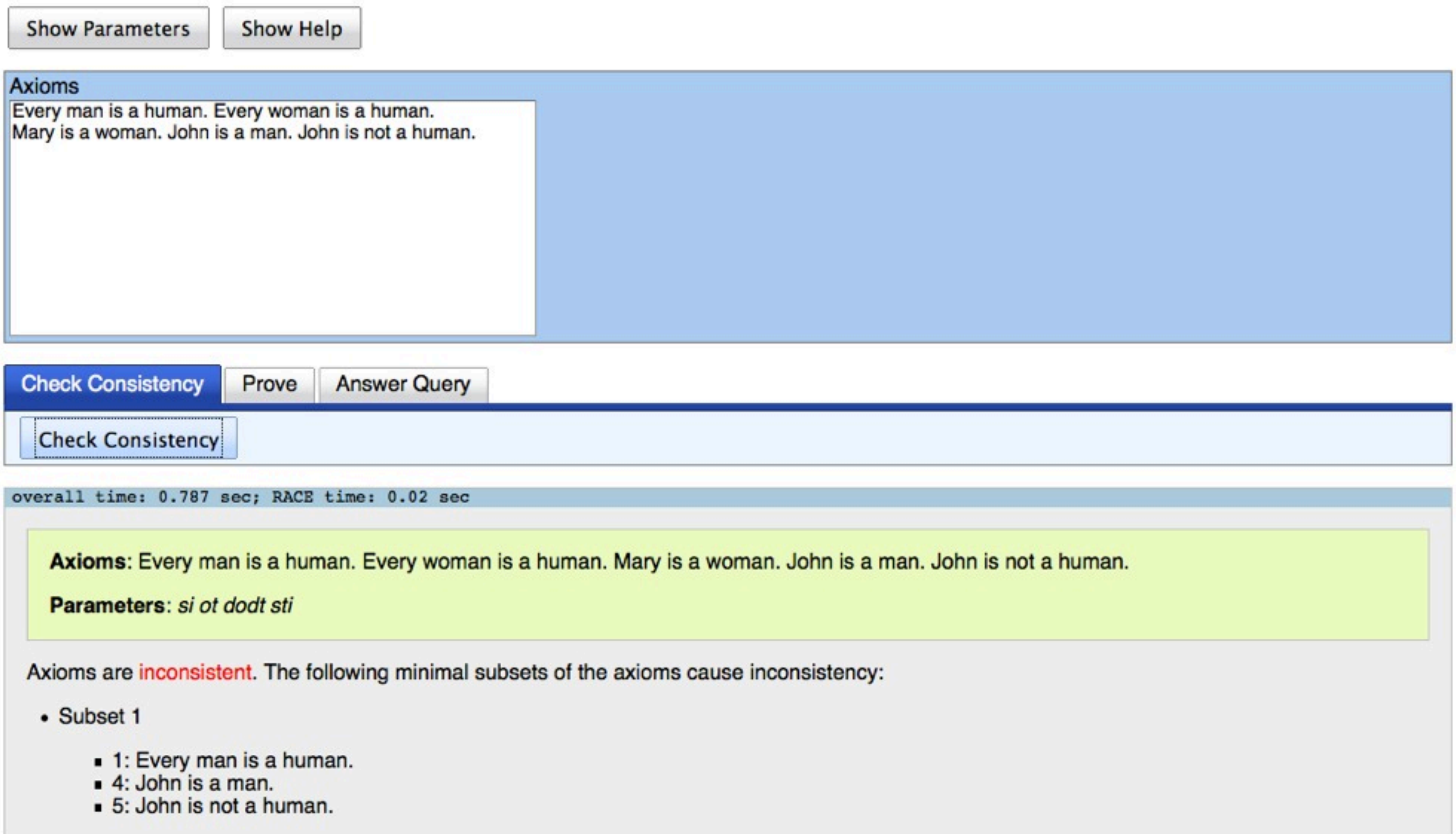
  satchmo_clause(object(sk1, A, man, countable, na, eq, 1), (object(sk1, sk4(A), human, countable, na, eq, 1), predicate(sk1, sk5(A), be_NP, A, sk4(A))), [axiom(2)])

  satchmo_clause((object(sk1, A, cat, countable, na, eq, 1), object(sk1, B, human, countable, na, eq, 1), predicate(sk1, C, be_NP, A, B)), fail, [axiom(3)])

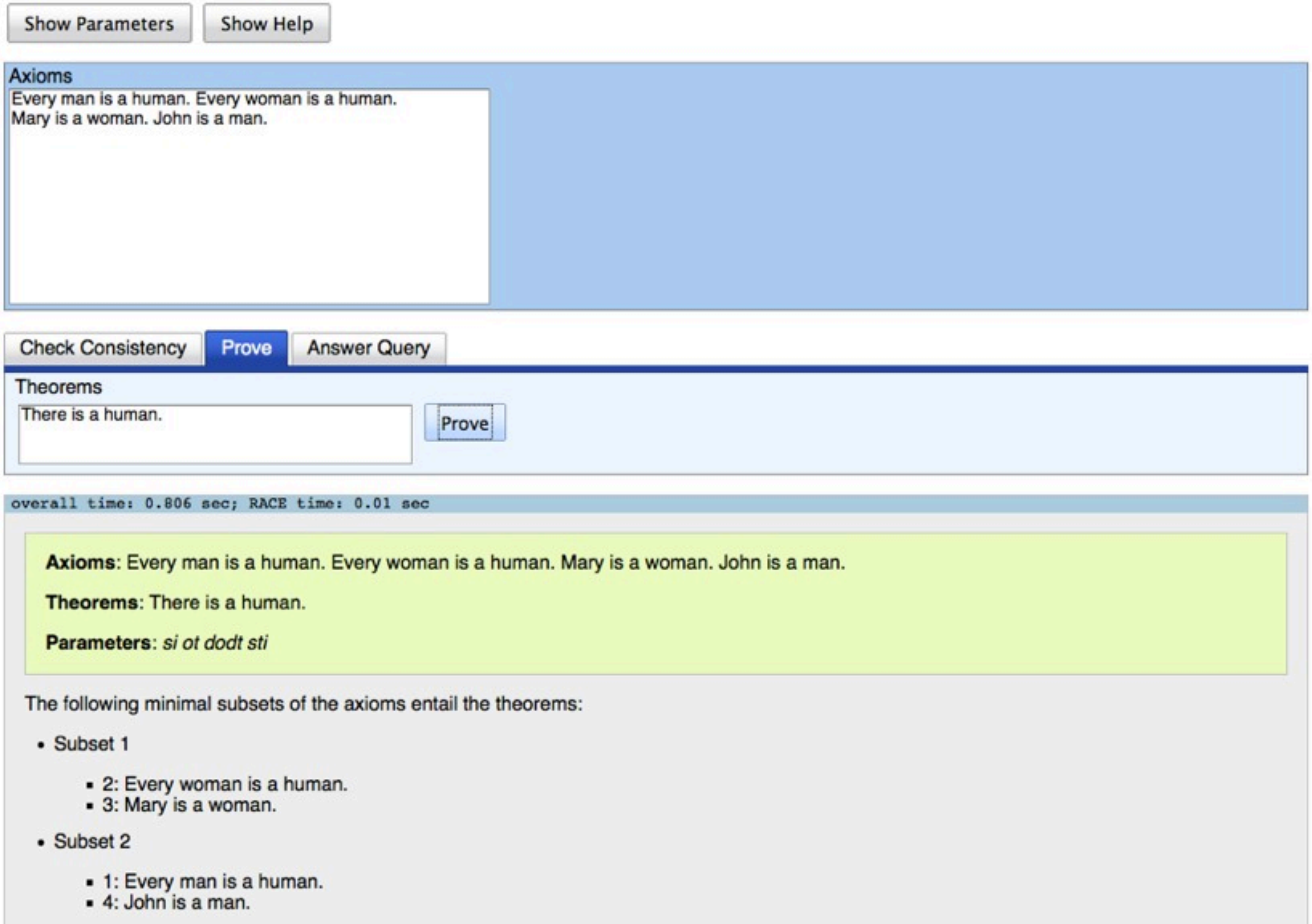
How Does RACE Work?



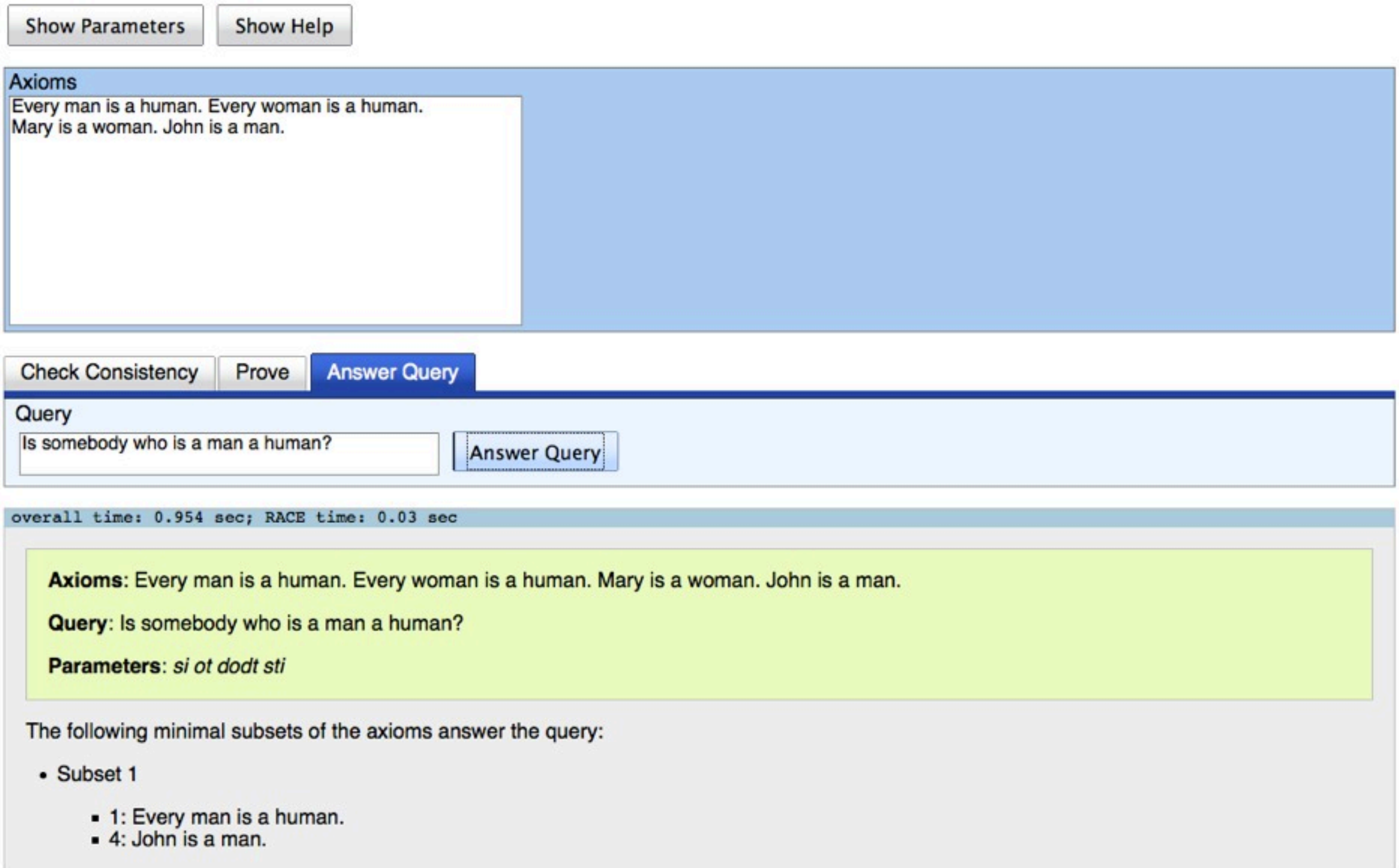
Web-Interface: Consistency Checking



Web-Interface: Theorem Proving



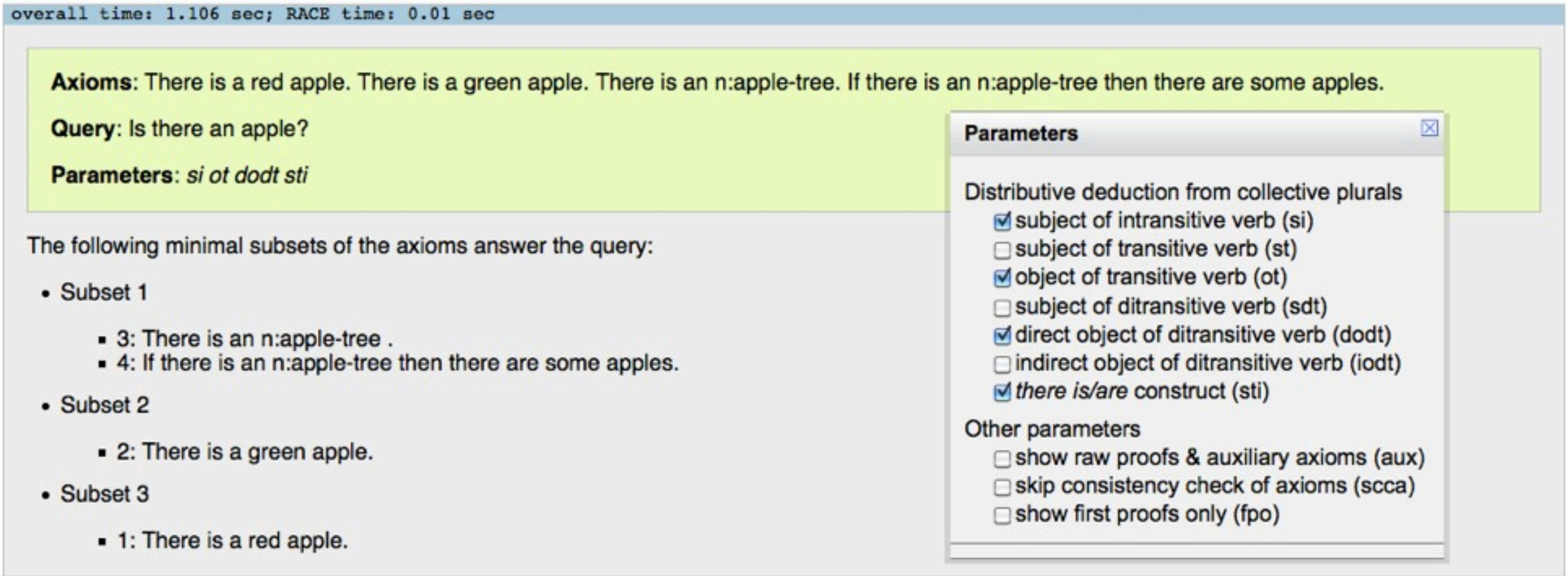
Web-Interface: Query Answering



RACE Parameters

- according to the Attempto philosophy, RACE should not require any knowledge of theorem proving in general, or RACE's internal working in particular
- nevertheless, RACE offers a number of parameters that
  - enable/disable distributive deductions from collective plurals
  - enable/disable the display of auxiliary axioms used during a proof
  - enable/disable consistency checking the axioms for proofs and query answering
  - limit the search of the proof-tree
- default settings allow most users to ignore the parameters

Parameter "Show First Proofs Only"



overall time: 1.106 sec; RACE time: 0.01 sec

Axioms: There is a red apple. There is a green apple. There is an n:apple-tree. If there is an n:apple-tree then there are some apples.

Query: Is there an apple?

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms answer the query:

- Subset 1
  - 3: There is an n:apple-tree .
  - 4: If there is an n:apple-tree then there are some apples.
- Subset 2
  - 2: There is a green apple.
- Subset 3
  - 1: There is a red apple.

Parameters

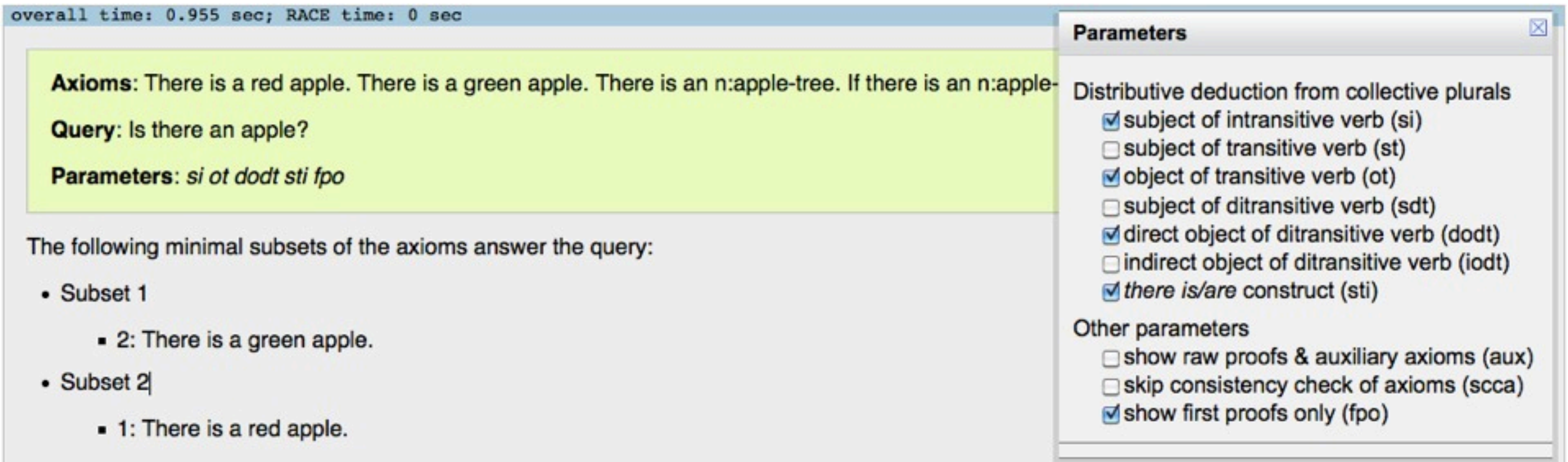
Distributive deduction from collective plurals

- [x] subject of intransitive verb (si)
- [ ] subject of transitive verb (st)
- [x] object of transitive verb (ot)
- [ ] subject of ditransitive verb (sdt)
- [x] direct object of ditransitive verb (dodt)
- [ ] indirect object of ditransitive verb (iodt)
- [x] *there is/are* construct (sti)

Other parameters

- [ ] show raw proofs & auxiliary axioms (aux)
- [ ] skip consistency check of axioms (scca)
- [ ] show first proofs only (fpo)

Parameter "Show First Proofs Only"



Auxiliary Axioms for RACE

- RACE needs *domain-independent* background knowledge that – in general – cannot be expressed in ACE
  - relations between the plural and the singular form of nouns
  - operations on natural numbers and arithmetic
  - semantics of generalised quantifiers (*at least*, *more than*, …)
  - interpretation of under-represented language elements, e.g. copula *to be*
  - ...
- access to the DRS notation allows us to express this knowledge as auxiliary axioms in the form of
  - first-order formulas
  - Prolog clauses
- auxiliary axioms can also be used to implement evaluable functions or to access external knowledge sources

Proving with Auxiliary Axioms

```
overall time: 2.003 sec; RACE time: 0.01 sec
```

Axioms: Mary sees some cats and some dogs.

Theorems: Mary sees a cat.

Parameters: *si ot dodt sti aux*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: Mary sees some cats and some dogs.

```
▪ Prolog Axiom cd2: at least M objects |- M, M-1, ..., 1 objects
▪ Prolog Axiom npc3: Deduction from NP conjunction as object of transitive verb.
```

Parameters

Distributive deduction from collective plurals

- [x] subject of intransitive verb (si)
- [ ] subject of transitive verb (st)
- [x] object of transitive verb (ot)
- [ ] subject of ditransitive verb (sdt)
- [x] direct object of ditransitive verb (dodt)
- [ ] indirect object of ditransitive verb (iodt)
- [x] *there is/are* construct (sti)

Other parameters

- [x] show raw proofs & auxiliary axioms (aux)
- [ ] skip consistency check of axioms (scca)
- [ ] show first proofs only (fpo)

Why? Why Not?

As we have seen, RACE answers the question "Why?" – but it also answers the question "Why Not?".

- if ACE theorems/queries can be deduced from ACE axioms then RACE answers the question "why?"
  - RACE lists the ACE axioms needed to deduce the ACE theorems/queries
  - optionally, RACE lists the auxiliary axioms used in the deduction
- if ACE theorems/queries cannot be deduced from ACE axioms then RACE answers the question "why not?"
  - RACE lists the words or constructs of the ACE theorem/query that could not be proved
  - list is generated from the set difference {model of the theorem/query} – {model of the axioms}

Why Not?

missing words

```
overall time: 1.953 sec; RACE time: 0 sec
```

Axioms: Mary sees a kitten on a farm.

Theorems: Mary assumes that the kitten is owned by the farmer.

Parameters: *si ot dodt sti*

Theorems do not follow from axioms.

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

- transitive verb: own
- countable common noun: (a/the) farmer
- transitive verb: assume
- proper name: Mary

Why Not?

missing constructs

```
overall time: 1.954 sec; RACE time: 0.01 sec
```

Axioms: John sees Mary.

Theorems: John can see Mary and himself.

Parameters: *si ot dodt sti*

Theorems do not follow from axioms.

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

- conjunctive noun phrase
- modal operator or sentence subordination

All Things Considered

Data Base Query

Given the ACE sentences

John has a red apple.
John has a green apple.
John has a yellow apple.

show that

John has an apple.

```
overall time: 1.128 sec; RACE time: 0.01 sec
```

Axioms: John has a red apple. John has a green apple. John has a yellow apple.

Theorems: John has an apple.

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 3: John has a yellow apple.
- Subset 2
  - 2: John has a green apple.
- Subset 3
  - 1: John has a red apple.

Modus Ponens

Given the ACE sentences

Every man is a human.
John is a man.

show that

John is a human.

```
overall time: 0.855 sec; RACE time: 0 sec
```

Axioms: Every man is a human. John is a man.

Theorems: John is a human.

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: Every man is a human.
  - 2: John is a man.

More Modus Ponens
Lewis Carroll's Grocer Puzzle

Given the ACE sentences

Every honest and industrious person is healthy.
No grocer is healthy.
Every industrious grocer is honest.
Every cyclist is industrious.
Every unhealthy cyclist is dishonest.
No healthy person is unhealthy.
No honest person is dishonest.
Every grocer is a person.
Every cyclist is a person.

show that

No grocer is a cyclist.

Invoking RACE

```
overall time: 2.752 sec; RACE time: 0.65 sec
```

Axioms: Every honest and industrious person is healthy. No grocer is healthy. Every industrious grocer is honest. Every cyclist is industrious. Every unhealthy cyclist is dishonest. No healthy person is unhealthy. No honest person is dishonest. Every grocer is a person. Every cyclist is a person.

Theorems: No grocer is a cyclist.

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: Every honest and industrious person is healthy.
  - 2: No grocer is healthy.
  - 3: Every industrious grocer is honest.
  - 4: Every cyclist is industrious.
  - 8: Every grocer is a person.

Variations of Query Answering

yes/*no*-queries ask for the existence or non-existence of a specified situation

```
overall time: 1.794 sec; RACE time: 0.01 sec
```

Axioms: John sleeps. Mary sleeps silently.

Query: Does somebody sleep?

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms answer the query:

- Subset 1
  - 2: Mary sleeps silently.
- Subset 2
  - 1: John sleeps.

Variations of Query Answering

wh-queries (*who*, *whose*, *what*, *which*) ask for noun phrases

`overall time: 1.953 sec; RACE time: 0.01 sec`

Axioms: John sleeps. Mary does not sleep.

Query: Who does not sleep?

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms answer the query:

- Subset 1
  - 2: Mary does not sleep.

`overall time: 2.004 sec; RACE time: 0.02 sec`

Axioms: John's dog barks. A dog of Mary barks.

Query: Whose dog barks?

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms answer the query:

- Subset 1
  - 2: A dog of Mary barks.
- Subset 2
  - 1: John ' s dog barks.

Variations of Query Answering

how-queries ask for adverbs and prepositional phrases

```
overall time: 1.854 sec; RACE time: 0.01 sec
```

Axioms: Mary sleeps silently. John sleeps with some heavy dreams.

Query: How does somebody sleep?

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms answer the query:

- Subset 1
  - 2: John sleeps with some heavy dreams.
- Subset 2
  - 1: Mary sleeps silently.

Variations of Query Answering

where- and *when*-queries ask for adverbs and prepositional phrases

overall time: 1.904 sec; RACE time: 0.01 sec

Axioms: Mary sleeps silently. John sleeps in a bedroom.

Query: Where does somebody sleep?

Parameters: *si ot dodt sti*

There is 1 message.

Importance	Type	Sentence	Problem	Description/Suggestion
warning	query answering		The query word "where" was interpreted as the query word "how".	

The following minimal subsets of the axioms answer the query:

- Subset 1
  - 2: John sleeps in a bedroom.
- Subset 2
  - 1: Mary sleeps silently.

since ACE does not use thematic roles – that would allow RACE to distinguish location, time, manner, instrument etc. – the *where*- and *when*-queries are interpreted as the less specific *how*-queries

Deductions from Plurals

Given the ACE sentence

John has three apples.

show that

John has one apple.

The deduction seems obvious to us humans – but RACE doesn't do it.

`overall time: 0.705 sec; RACE time: 0 sec`

Axioms: John has three apples.

Theorems: John has one apple.

Parameters: *si dodt sti*

Theorems do not follow from axioms.

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

- countable common noun: (a/the) apple

To have RACE perform this deduction information needs to be provided that links the plural *apples* of the axiom to the singular *apple* of the theorem.

Deductions from Plurals
Adding the Missing Information as an ACE Axiom

Given the ACE sentence

John has three apples.
If John has three apples then he has one apple.

show that

John has one apple.

Now RACE does the deduction.

```
overall time: 0.754 sec; RACE time: 0 sec
```

Axioms: John has three apples. If John has three apples then he has one apple.

Theorems: John has one apple.

Parameters: *si dodt sti*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: John has three apples.
  - 2: If John has three apples then he has one apple.

The approach works for this case, but obviously cannot be generalised.

Deductions from Plurals Auxiliary Axioms

To relate singulars and collective plurals of countable nouns we can make use of their logical representations that are introduced by the translation of an ACE text into a DRS.

- while normally nouns like *apple* are represented logically as predicates – for instance *apple(A)* – in our DRS notation nouns are represented as constant arguments of the predefined predicate *object/6*
  - *an apple* is represented as *object(A, apple, countable, na, eq, 1)*
  - *three apples* is represented as *object(B, apple, countable, na, eq, 3)*
- singulars and collective plurals of countable nouns get the same representation with appropriate cardinalities
- this representation allows us to logically express the relation between collective plurals and singulars of all countable nouns – expressed by the variable *Noun* – as the auxiliary axiom

  $\forall$ *A,Noun,N (object(A, Noun, countable, na, eq, N)* $\wedge$ *N>1* $\rightarrow$ *object(A, Noun, countable, na, eq, 1))*
- or more generally

  $\forall$ *A,Noun,N1,N2 (object(A, Noun, countable, na, eq, N1)* $\wedge$ *N1>N2* $\rightarrow$ *object(A, Noun, countable, na, eq, N2))*

Deductions from Plurals
Auxiliary Axioms

Having an auxiliary axiom relating singulars and collective plurals of nouns, we must distinguish when to use the axiom and when not to use it.

- Clearly we want: *John has three apples.* |– *John has one apple.*
- Probably we do not want: *Five men lift a piano.* |– *A man lifts a piano.*

RACE's parameters control when a deduction from a collective plural is enabled or disabled. Default settings conform to the most common use.

- parameter si "distributive deduction from a collective plural actings as the subject of an intransitive verb" is per default enabled (*Three cats sleep.* |– *Two cats sleep.*)
- parameter si "distributive deduction from a collective plural actings as the subject of an transitive verb" is per default disabled (*Five men lift a piano.* |/– *A man lifts a piano.*)
- parameter ot "distributive deduction from a collective plural actings as the object of an transitive verb" is per default enabled (*John has three apples.* |– *John has one apple.*)
- ...

More on Auxiliary Axioms

- auxiliary axioms can be formulated for other cases involving nouns – e.g. generalised quantifiers and distributive plurals – and for other word classes like the copula *to be*
- much of the deductive power of RACE stems from auxiliary axioms which allows us to state

  axioms |- theorems

  if and only if

  {model of theorems} ⊆ *{model of axioms* ∪ *additional model derived via auxiliary axioms}*
- RACE uses auxiliary axioms expressed as first-order formulas that are translated into Satchmo clauses, or as Prolog clauses that are called directly
- auxiliary axioms expressed as Prolog clauses can also implement evaluable functions or access external knowledge sources
- currently there are about 70 auxiliary axioms, and the number is growing
- auxiliary axioms also have their problems
  - there can be deductions that only differ in the auxiliary axioms being used – which wastes runtime
  - auxiliary axioms can interact to produce incorrect deductions; RACE mutually excludes them

Deductions from Plurals Using Auxiliary Axioms

Given the ACE sentence

John has three apples.

show that

John has one apple.

RACE performs the deduction using the auxiliary axiom cd1 that is expressed as a Prolog clause.

```
overall time: 0.703 sec; RACE time: 0 sec
```

Axioms: John has three apples.

Theorems: John has one apple.

Parameters: *si ot dodt sti aux*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: John has three apples.
  - `Prolog Axiom cd1: M objects |- M-1, ..., 1 objects`

Deductions from Generalised Quantifiers

Given the ACE sentence

John has at least three apples.

show that

John has more than 2 apples.

RACE performs the deduction using the auxiliary axiom cd8.

```
overall time: 2.056 sec; RACE time: 0 sec
```

Axioms: John has at least three apples.

Theorems: John has more than 2 apples.

Parameters: *si ot dodt sti aux*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: John has at least three apples.
  - `Prolog Axiom cd8: at least M objects |- more than 1, 2, ..., M-1 objects`

Deductions from Conjunctive Plurals

Given the ACE sentence

John and Mary wait.

show that

Mary waits.

The deduction seems obvious – but checking the respective DRSs one realises that RACE cannot perform it directly, nor with the auxiliary axioms for collective plurals introduced so far.

 # John and Mary wait.
[A, B]
has_part(B, named(John))
has_part(B, named(Mary))
object(B, na, countable, na, eq, 2)
predicate(A, wait, B)

 # Mary waits.
[A]
predicate(A, wait, named(Mary))

Another auxiliary axiom is needed.

Deductions from Conjunctive Plurals

Given the ACE sentence

John and Mary wait.

show that

Mary waits.

RACE performs the deduction using the auxiliary axiom npc1.

overall time: 1.756 sec; RACE time: 0.01 sec

Axioms: John and Mary wait.

Theorems: Mary waits.

Parameters: *si ot dodt sti aux*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: John and Mary wait.
  - Prolog Axiom npc1: Deduction from NP conjunction as subject of intransitive verb.

Aggregation
Counting Cats

just adding 2 and 3 cats

`overall time: 1.802 sec; RACE time: 0.01 sec`

Axioms: John has 2 red cats. John has 3 black cats.

Theorems: John has 5 cats.

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: John has 2 red cats.
  - 2: John has 3 black cats.

Note added after the presentation:
The representation tacitly assumes that the classes "red cat" and "black cat" are distinct.
Some participants of CNL 2010 pointed out that this should better be made explicit, e.g. by "No red cat is a black cat.".

Aggregation
Counting Cats and Dogs

adding elements of the super-class *animal*

`overall time: 1.86 sec; RACE time: 0.06 sec`

Axioms: John has a cat. John has a dog. Every cat is an animal. Every dog is an animal.

Theorems: John has 2 animals.

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: John has a cat.
  - 2: John has a dog.
  - 3: Every cat is an animal.
  - 4: Every dog is an animal.

Note added after the presentation:
The representation on this and the following two slides tacitly assumes that the classes "cat" and "dog" are distinct.
Some participants of CNL 2010 pointed out that this should better be made explicit, e.g. by "No cat is a dog.".

Aggregation
Counting Cats and Dogs

`overall time: 2.254 sec; RACE time: 0.06 sec`

Axioms: John has 2 cats. John has 3 dogs. Every cat is an animal. Every dog is an animal.

Theorems: John has 5 animals.

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: John has 2 cats.
  - 2: John has 3 dogs.
  - 3: Every cat is an animal.
  - 4: Every dog is an animal.

`overall time: 2.304 sec; RACE time: 0.1 sec`

Axioms: John has a cat X. John has a dog Y. John has an animal that is not X and that is not Y. Every cat is an animal. Every dog is an animal.

Theorems: John has at least 3 animals.

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: John has a cat X.
  - 2: John has a dog Y.
  - 3: John has an animal that is not X and that is not Y.
  - 4: Every cat is an animal.
  - 5: Every dog is an animal.

Aggregation & Conjunctive Plurals
Counting Dogs

Given the ACE sentence

Mary has a peaceful dog and a dog that bites the postman.

show that

Mary has two dogs.

RACE performs the deduction using the auxiliary axioms npc3 and agg1.

```
overall time: 2.076 sec; RACE time: 0.01 sec
```

Axioms: Mary has a peaceful dog and a dog that bites the postman.

Theorems: Mary has two dogs.

Parameters: *si ot dodt sti aux*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: Mary has a peaceful dog and a dog that bites the postman.
  - `Prolog Axiom agg1: aggregation`
- Subset 2
  - 1: Mary has a peaceful dog and a dog that bites the postman.
  - `Prolog Axiom agg1: aggregation`
  - `Prolog Axiom npc3: Deduction from NP conjunction as object of transitive verb.`

Modality

ACE provides modal auxiliaries for *possibility* and *necessity*, as well as *sentence subordination* ...

A man can wait. A customer must wait. John believes that a customer waits.

... which extend the DRS language by the three operators *diamond* <>, *box* [] and *colon* :

```
[A, B, C, D]
object(A, man, countable, na, eq, 1)-1/2
  <>
  [E]
  predicate(E, wait, A)-1/4
object(B, customer, countable, na, eq, 1)-2/2
  []
  [F]
  predicate(F, wait, B)-2/4
predicate(C, believe, named(John), D)-3/2
  D:
  [G, H]
  object(G, customer, countable, na, eq, 1)-3/5
  predicate(H, wait, G)-3/6
```

Deductions with Modality

Given the ACE sentence

John waits.

show that

John can wait.

How can we prove that?

One possibility is to use one of the axioms of propositional modal logic, in this case

$A \rightarrow <> A$

but this approach doesn't work for deriving

John can wait.

from

John waits patiently in the hall.

that requires a more powerful logic.

Extended Standard Translation to Possible World Semantics

Modal logic can be mapped to first-order predicate logic via the so-called *standard translation*

- standard translation adds to each logical atom an argument that stands for a "possible world"
- possible worlds are connected by a binary accessibility relation
- RACE assumes the accessibility relation to be *reflexive*, *symmetric* and *transitive* (= equivalence relation)
- standard translation DRS → FOL with Johan Bos' extension for *sentence subordination* (cf. rule 9)

1. $(w, \langle\{x_1 \ldots x_n\}, \{\gamma_1 \ldots \gamma_m\}\rangle)^{fo} \stackrel{\text{def}}{=} \exists x_1 \cdots \exists x_n((w, \gamma_1)^{fo} \wedge \cdots \wedge (w, \gamma_m)^{fo})$;
2. $(w, R(x_1, \ldots, x_n))^{fo} \stackrel{\text{def}}{=} R(w, x_1, \ldots, x_n)$;
3. $(w, x_1 = x_2)^{fo} \stackrel{\text{def}}{=} x_1 = x_2$;
4. $(w, \neg B)^{fo} \stackrel{\text{def}}{=} \neg(w, B)^{fo}$;
5. $(w, B_1 \vee B_2)^{fo} \stackrel{\text{def}}{=} (w, B_1)^{fo} \vee (w, B_2)^{fo}$;
6. $(w, \langle\{x_1 \ldots x_n\}, \{\gamma_1 \ldots \gamma_m\}\rangle \Rightarrow B)^{fo} \stackrel{\text{def}}{=}$
   $\forall x_1 \cdots \forall x_n(((w, \gamma_1)^{fo} \wedge \cdots \wedge (w, \gamma_m)^{fo}) \rightarrow (w, B)^{fo})$;
7. $(w, \Diamond B)^{fo} \stackrel{\text{def}}{=} \exists v(R(w, v) \wedge (v, B)^{fo})$;
8. $(w, \Box B)^{fo} \stackrel{\text{def}}{=} \forall v(R(w, v) \rightarrow (v, B)^{fo})$;
9. $(w, v : B)^{fo} \stackrel{\text{def}}{=} (R(w, v) \wedge (v, B)^{fo})$.

Modality: Example Deductions

A I– B and modality axiom B I– <> B (if B then it is possible that B)

`overall time: 2.042 sec; RACE time: 0.01 sec`

Axioms: John waits patiently in the hall.

Theorems: John can wait.

Parameters: *si ot dodt sti aux*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
    - 1: John waits patiently in the hall.
    - `Prolog Axiom pw1: Accessibility relation is reflexive.`

Modality: Example Deductions

modality axiom [] A |– ¬ <> ¬ A (if A is necessary then it is not possible that not A)

`overall time: 1.754 sec; RACE time: 0.01 sec`

Axioms: John must wait.

Theorems: It is not possible that John cannot wait.

Parameters: *si ot dodt sti aux*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: John must wait.
  - `Prolog Axiom pw1: Accessibility relation is reflexive.`
- Subset 2
  - 1: John must wait.
  - `Prolog Axiom pw1: Accessibility relation is reflexive.`
  - `Prolog Axiom pw3: Accessibility relation is symmetric.`

Sentence Subordination: Example Deductions

John promises that he waits. = John promises to wait.

```
overall time: 2.053 sec; RACE time: 0.01 sec
```

Axioms: John a:honestly promises to wait for Mary.

Theorems: John promises to wait.

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: John a:honestly promises to wait for Mary.

Sentence Subordination & Modality

combining

- sentence subordination
- modality axiom A → <> A
- modality axiom [] A → <> A

```
overall time: 1.703 sec; RACE time: 0.01 sec
```

Axioms: Mary believes that John must wait.

Theorems: It is possible that Mary believes that John can wait.

Parameters: *si ot dodt sti aux*

The following minimal subsets of the axioms entail the theorems:

- Subset 1
  - 1: Mary believes that John must wait.
  - `Prolog Axiom pw1: Accessibility relation is reflexive.`
- Subset 2
  - 1: Mary believes that John must wait.
  - `Prolog Axiom pw1: Accessibility relation is reflexive.`
  - `Prolog Axiom pw3: Accessibility relation is symmetric.`

Decidability, Efficiency and Other Aspects

Decidability of ACE

- ACE has been developed for high expressivity
- complexity and decidability considerations have been of minor concern
- undecidable English fragment (Pratt-Hartman & Third 2006)
  - Cop + Rel + TV + GA
  - Cop = singular, existentially/universally quantified nouns, predicative adjectives, copula with/without negation, Rel = relative clauses, TV = transitive verbs with/without negation, GA = reflexive/non-reflexive pronouns as anaphors, resolution by co-indexing anaphors and antecedent noun phrases
- ACE is not decidable since already its FOL subset is larger than this English fragment
- some decidable subsets of ACE
  - Cop + Rel + TV + DTV (DTV = ditransitive verbs with/without negation)
  - translation ACE ⇔ OWL defines a decidable subset of ACE
  - do not allow for explicit attributive adjectives, adverbs and prepositional phrases

Efficiency Considerations

- ACE is undecidable
  - RACE may not terminate
  - RACE uses a time-out limit calculated on the number of Satchmo clauses
- RACE uses forward-chaining of Satchmo clauses – i.e. eager evaluation – whose worst-case time complexity is $O(n^2)$ where n is the number of clauses
- reduction of the number of clauses that participate in forward-chaining
  - simplifying the DRS representation can reduce the number of clauses
  - clause compaction generates a smaller number of more complex clauses
    (clause head is a disjunction of conjunctions of logical atoms, clause body is a conjunction of disjunctions of logical atoms)
  - eliminate after the first round of forward-reasoning the clauses with the body *true* that cannot be fired again
  - using Prolog auxiliary axioms instead of FOL auxiliary axioms – that are translated into clauses – reduces the number of clauses and replaces eager evaluation by lazy evaluation for these axioms

Further Efficiency Considerations

- other means
  - complement splitting – given a disjunction $(A \vee B)$, one investigates $(A \wedge \neg B)$, respectively $(\neg A \wedge B)$ – though complement splitting is not guaranteed to increase the efficiency in each case
  - intelligent search for clauses that could be fired in the next round of forward chaining
  - Rete algorithm
  - good Prolog programming practices, specifically optimal clause indexing
- when proving theorems and answering questions with large sets of axioms users may want to set the parameters that
  - switch off the consistency check of axioms (RACE parameter scca)
  - limit the search of the proof-tree (RACE parameter fpo)

Lewis Carroll's Grocer Puzzle Runtime Comparisons

Lewis Carroll's Grocer Puzzle: influence of the parameters *fpo* and *scca* on the runtime

```
overall time: 2.754 sec; RACE time: 0.65 sec
```

Axioms: Every honest and industrious person is healthy. No grocer is healthy. Every industrious grocer is honest. Every cyclist is industrious. Every unhealthy cyclist is dishonest. No healthy person is unhealthy. No honest person is dishonest. Every grocer is a person. Every cyclist is a person.

Theorems: No grocer is a cyclist.

Parameters: *si ot dodt sti*

	Attempto Server	MacBook Pro
no parameter	650 ms	70 ms
fpo	480 ms	50 ms
scca	440 ms	41 ms
fpo & scca	280 ms	30 ms

Preventing Looping

- here is an excerpt from an ontology written by Kaarel Kaljurand

  ... Flossie is a mad cow. Every mad cow is a cow that eats the brain of a sheep. Every cow that eats the brain of a sheep is a mad cow. ...

- since ACE does not offer a specific construct for equality, Karel expresses the equality of the two classes

  A = mad cow and *B = cow that eats the brain of a sheep*

  by the implications

  $A \rightarrow B$

  $B \rightarrow A$

  which seems innocuous enough

- however, together with the fact

  A

  forward-reasoning in RACE leads to an infinite loop A, B, A, B, A, ...

- RACE contains a flexible loop detector that stops the loop after a predefined number of times

Reasoning Despite Loops

consistency checking

`overall time: 0.752 sec; RACE time: 0.01 sec`

Axioms: Flossie is a mad cow. Every mad cow is a cow that eats the brain of a sheep. Every cow that eats the brain of a sheep is a mad cow.

Parameters: *si ot dodt sti*

Axioms are consistent.

query answering

`overall time: 0.772 sec; RACE time: 0.01 sec`

Axioms: Flossie is a mad cow. Every mad cow is a cow that eats the brain of a sheep. Every cow that eats the brain of a sheep is a mad cow.

Query: Does a cow eat the brain of a sheep?

Parameters: *si ot dodt sti*

The following minimal subsets of the axioms answer the query:

- Subset 1
    - 1: Flossie is a mad cow.
    - 2: Every mad cow is a cow that eats the brain of a sheep.

Other Theoretical Aspects of RACE

RACE uses

- the unique name assumption, i.e. different names refer to different entities unless explicitly stated otherwise (*John is Harry.*)
- the open world assumption, i.e. missing knowledge of the truth of an ACE sentence A, respectively a failed proof of A, is not interpreted as $\neg$ A

Outlook

Conclusions and Further Research

So far, so good – but many things remain to be done, for example

- mathematical operations: arithmetic, sets, lists, strings
- answering of wh-queries by showing substitutions (e.g. "who = John")
- hypothetical reasoning (“What happens if …?”)
- abductive reasoning (“Under which conditions happens ...?”)
- temporal reasoning, perhaps with event calculus
- executable specifications
- problems suggested by RACE users (legal language, contracts, business rules, ...)
- larger examples, possibly using external knowledge sources
- ...