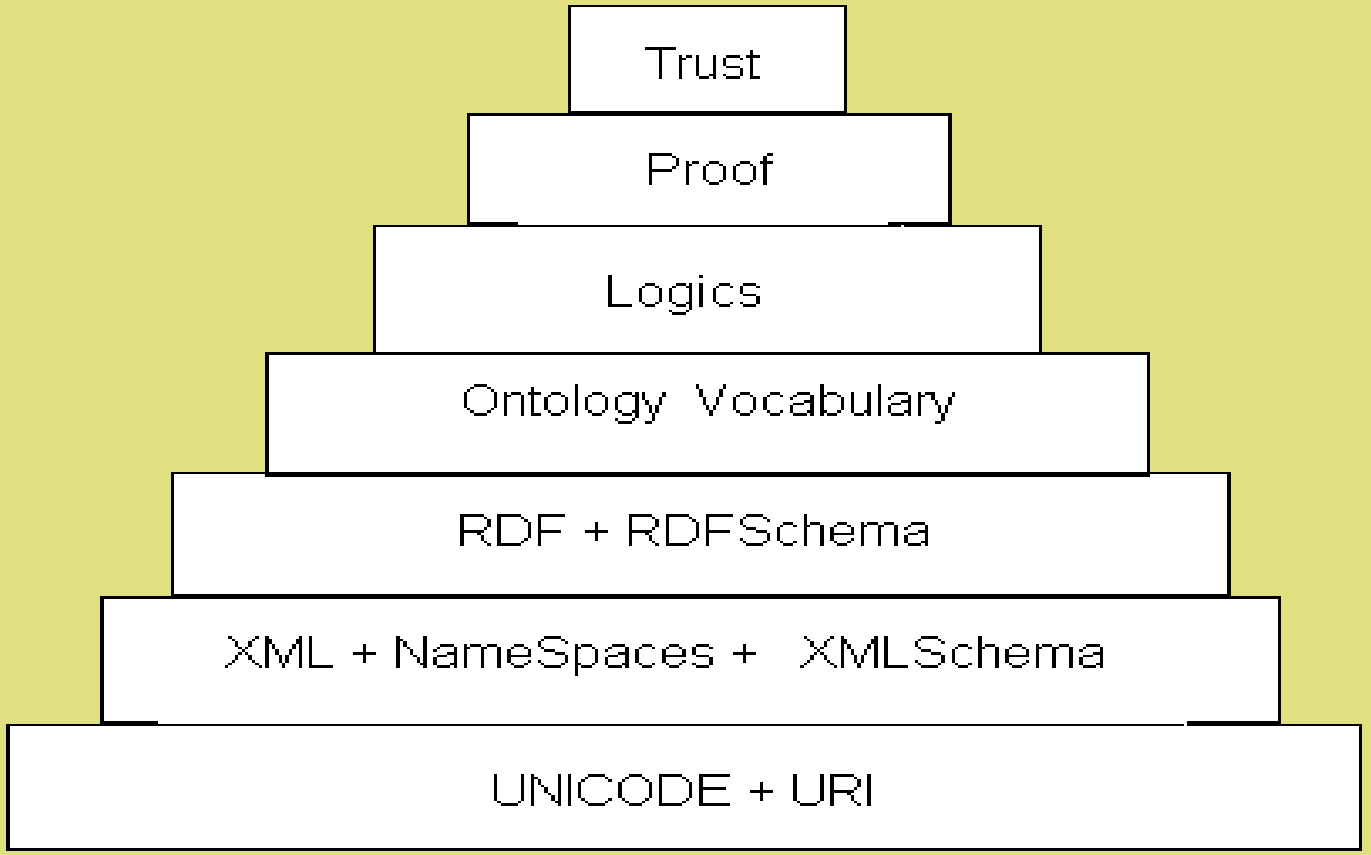
An inference program for RDF

Master thesis by G.Naudts

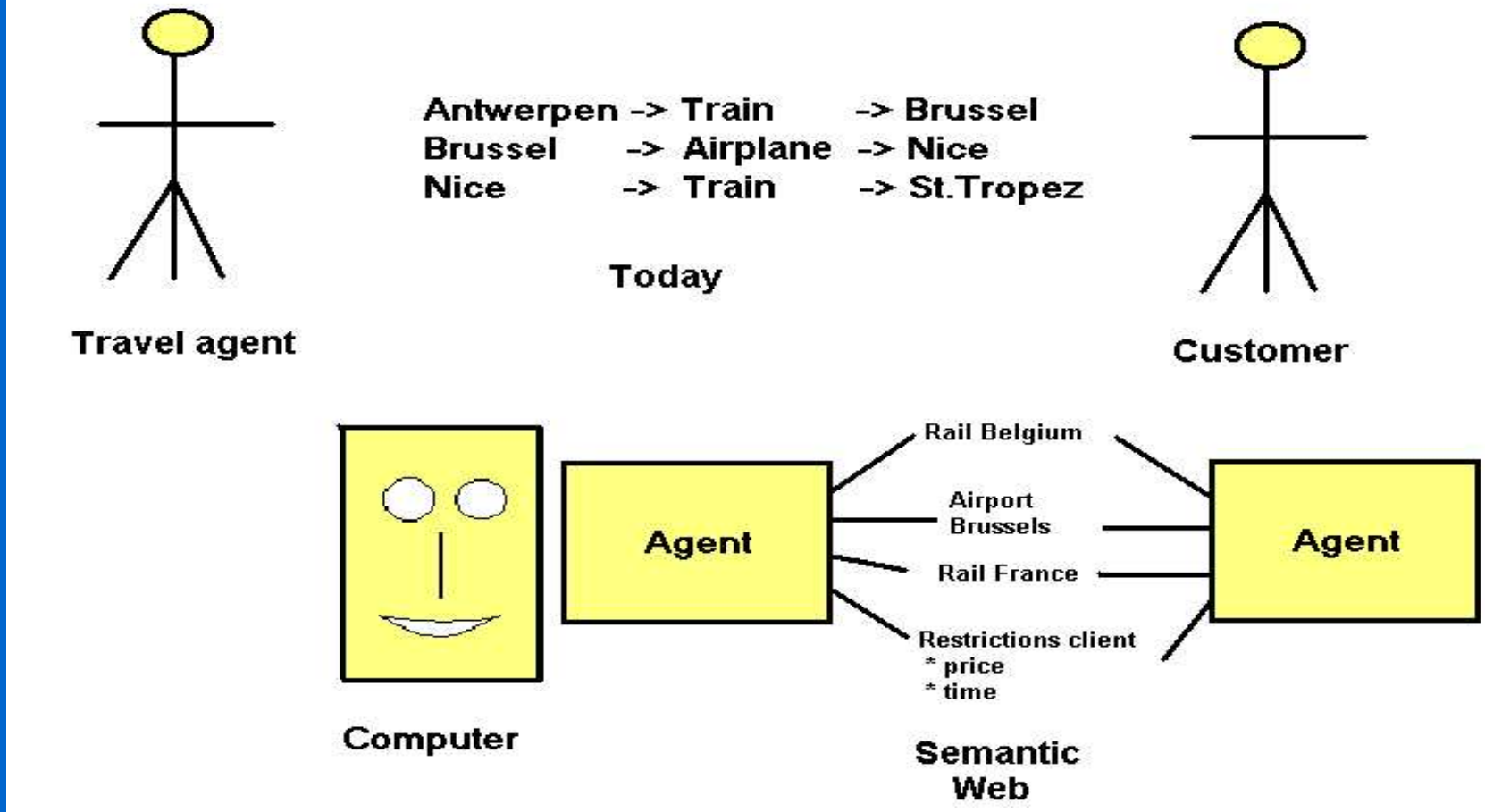
Open University of the Netherlands

Agfa

Standards


Trust
Proof
Logics
Ontology Vocabulary
RDF + RDFSchema
XML + NameSpaces + XMLSchema
UNICODE + URI

Case study


Antwerpen -> Train -> Brussel
Brussel -> Airplane -> Nice
Nice -> Train -> St.Tropez
Today
Travel agent
Customer
Rail Belgium
Airport Brussels
Rail France
Restrictions client
* price
* time
Agent
Agent
Computer
Semantic Web

Research questions

1) Haskell spec of RDF inference program

2) Soundness and completeness

3) What logics?

4) optimization

5) inconsistency

RDF / DEMO

- Database:

Regels:

child(X, Y), child(Y, Z) :> grandparent(Z, X).

grandparent(Z, X), gender(Z, female) :> grandmother(Z, X).

Feiten:

child(christine, elza).

child(wim, christine).

gender(elza, female).

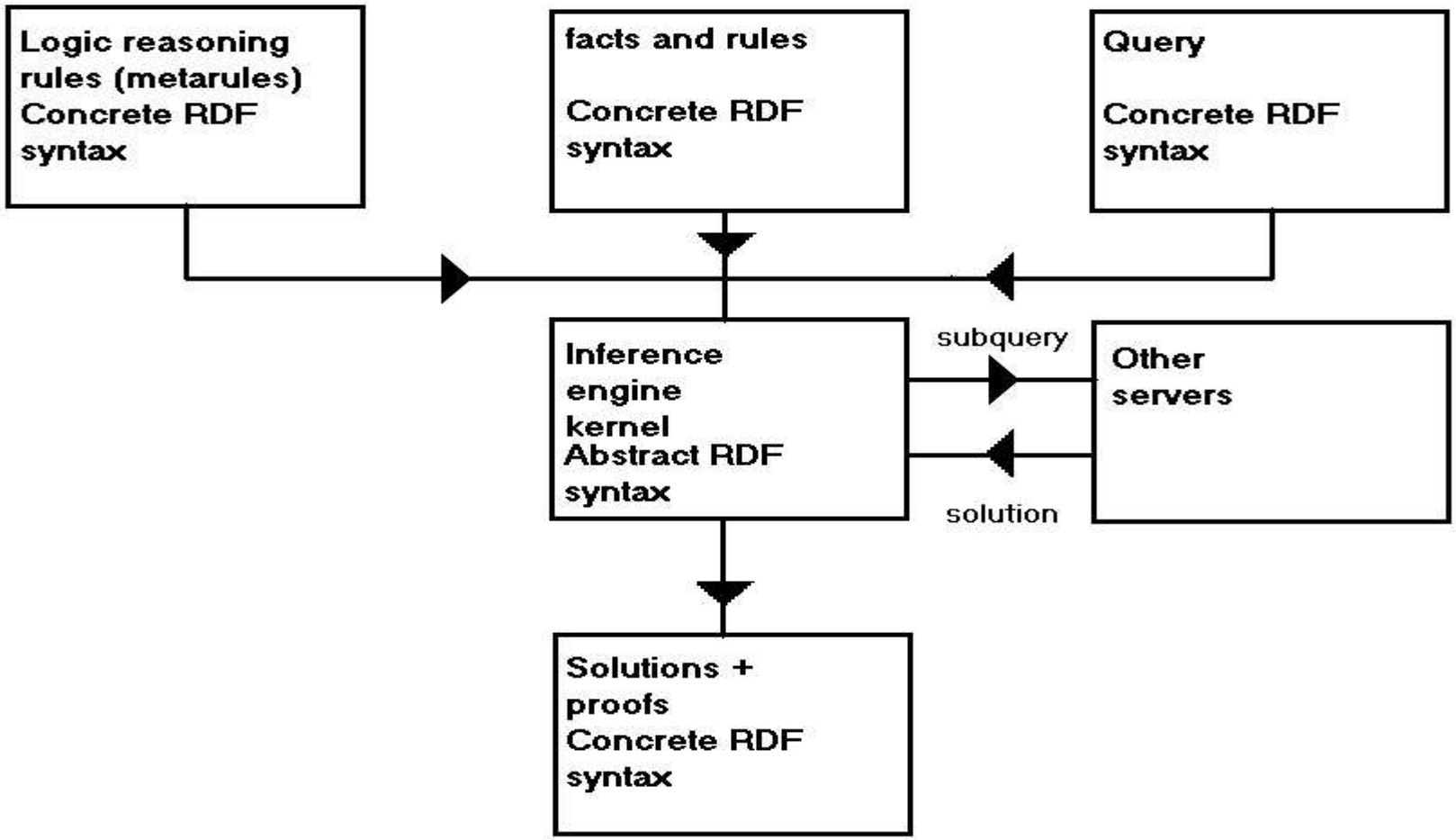
- Query:

grandmother(Z, X).

WWW aspects

- subqueries
- closed / open world
- verifiability / trust -> origin
- inconsistencies

Implementation

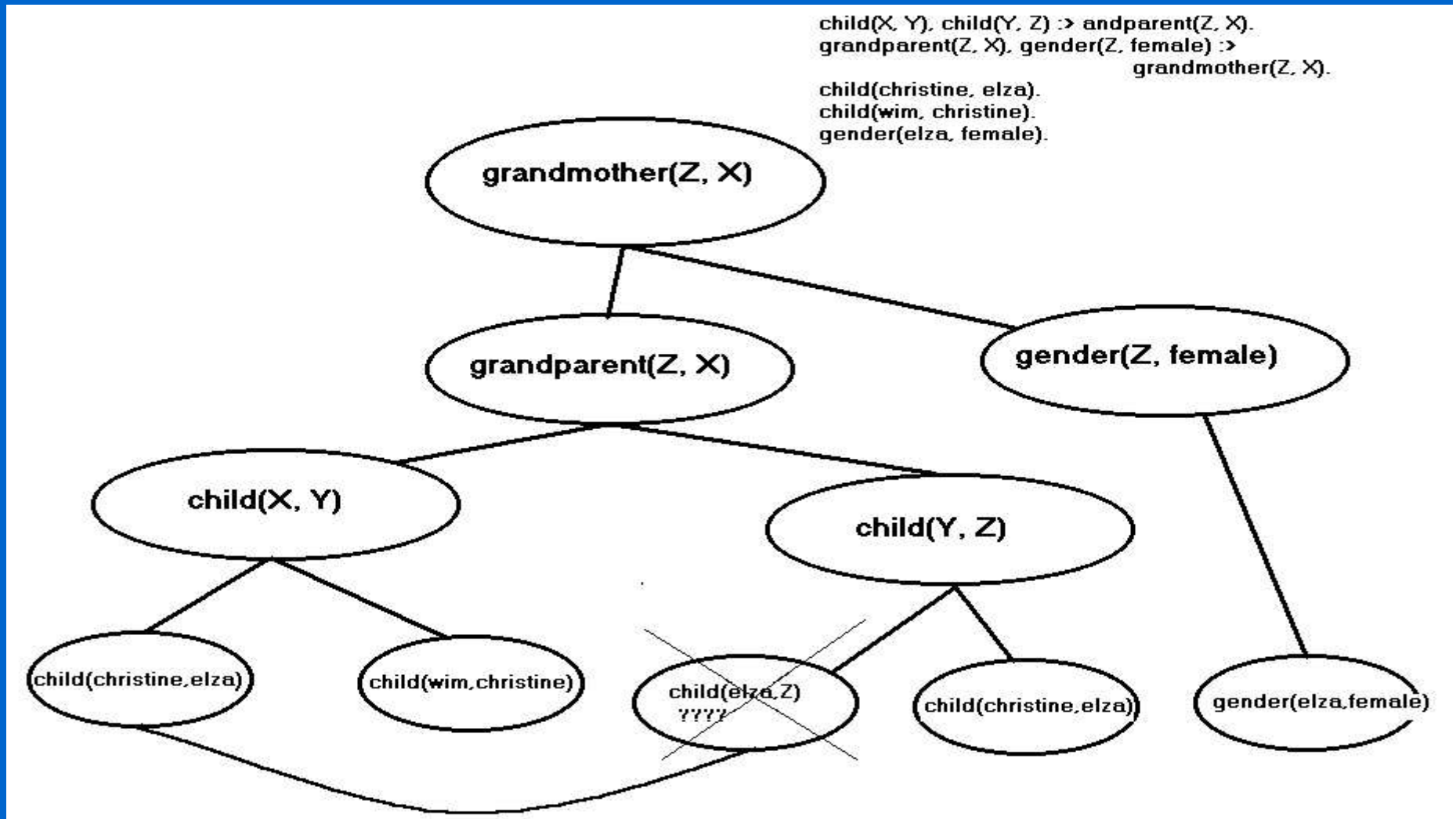


Inference engine for RDF

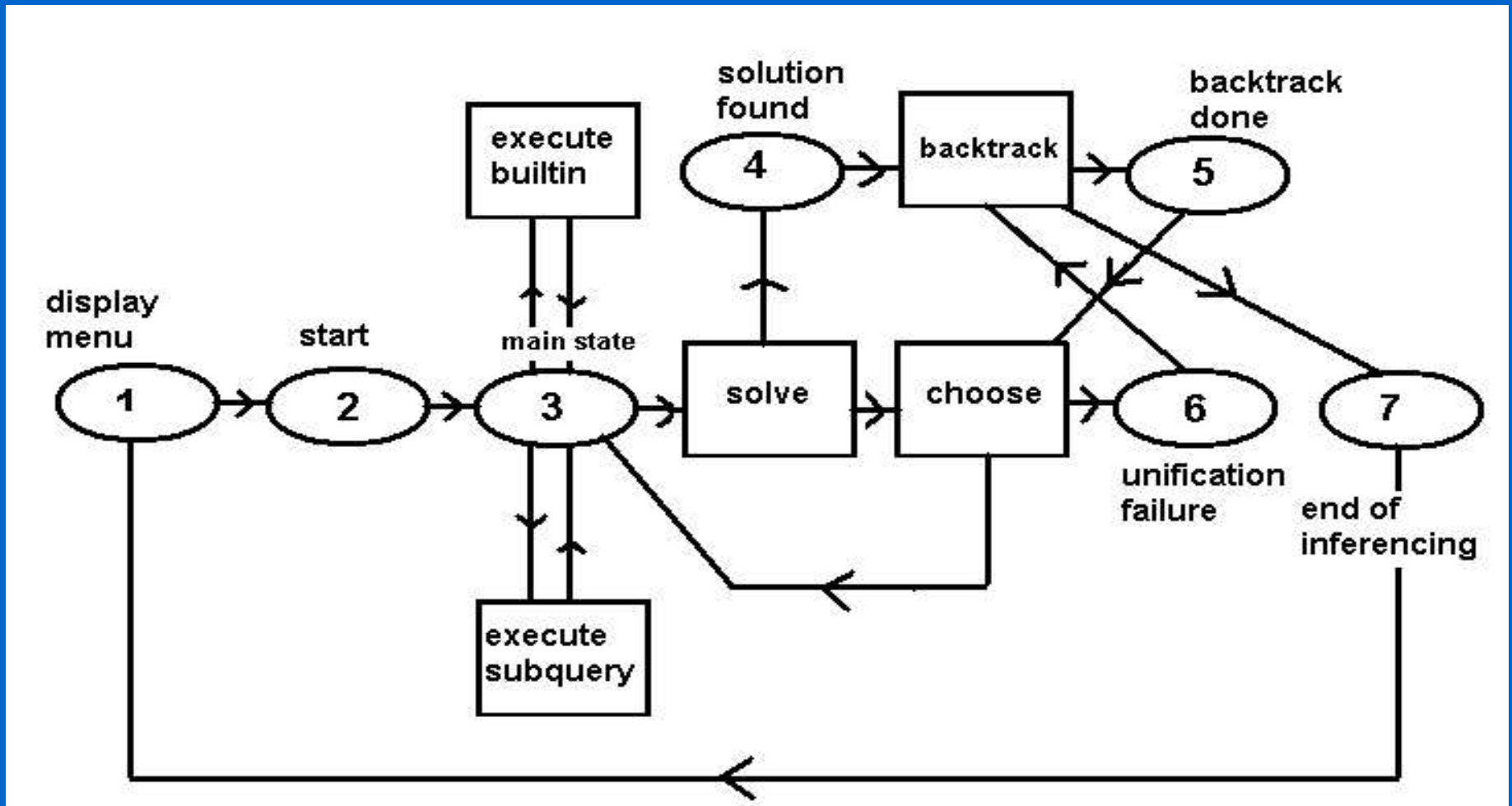
- Two versions:

1) Classic structure, well tested
2) Better adapted to the semantic web

Search Tree / DEMO



Finite State Machine



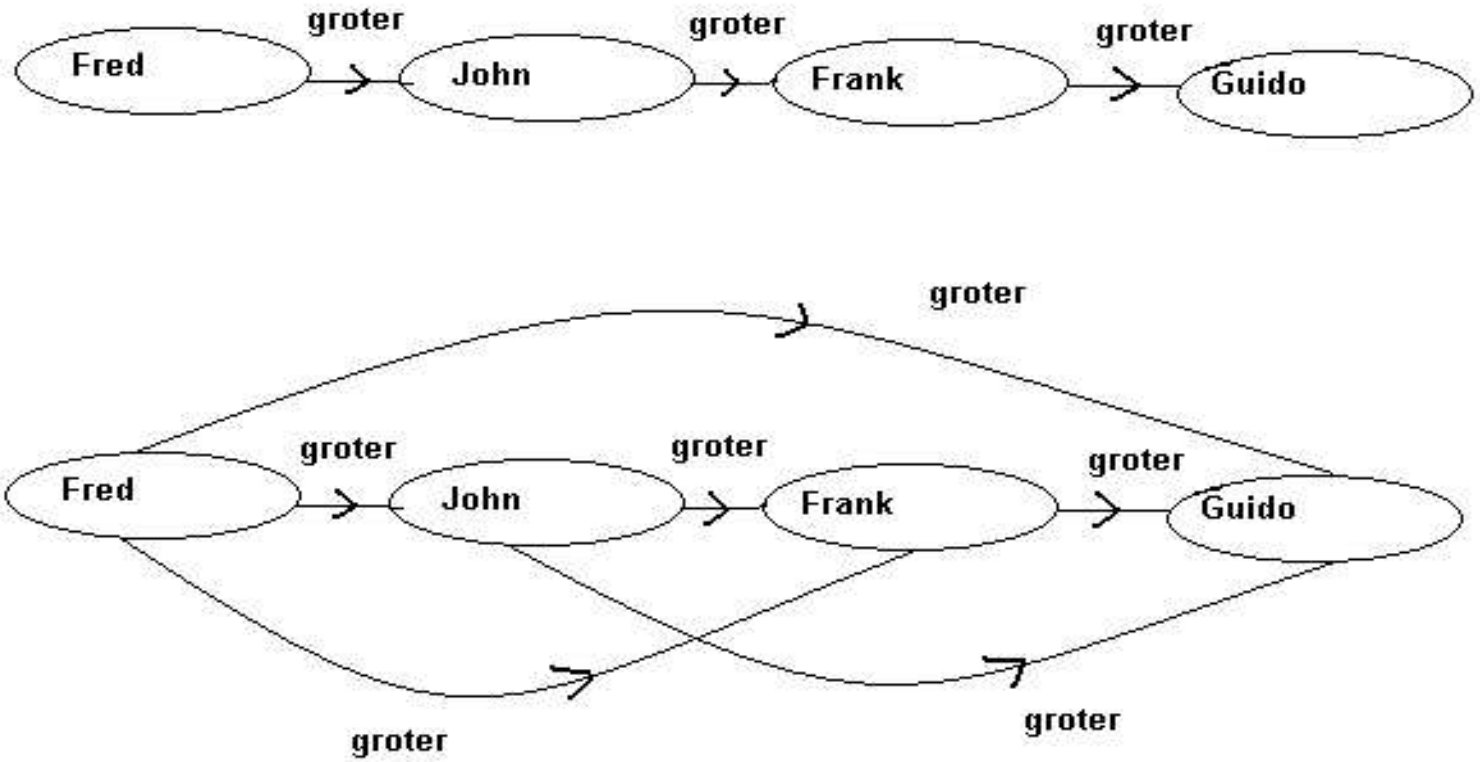
Open World Consequences

- An open set has no borders
- An (open?) set has no complement
- An element belongs to set A or set B or not known (no law of excluded middle)
- There is no universal quantifier for open sets

Research questions

1) Haskell spec of RDF inference program
2) Soundness and completeness
3) What logics?
4) optimization
5) inconsistency

Graph syntax of RDF



Graph theory

Lemmas

Resolution Lemma
Closure Lemma
Final Path Lemma
Looping Lemma I
Looping Lemma II
Infinite Lemma
Duplication Lemma
Failure Lemma
Substitution Lemma I
Substitution Lemma II
Solution Lemma I
Solution Lemma II
Solution Lemma III
Solution Lemma IV
Completeness Lemma
Infinite Looping Lemma
Monotonicity Lemma

Essential points

1) Proof of soundness

2) Proof of completeness

3) Proof of solution

4) Specific rule

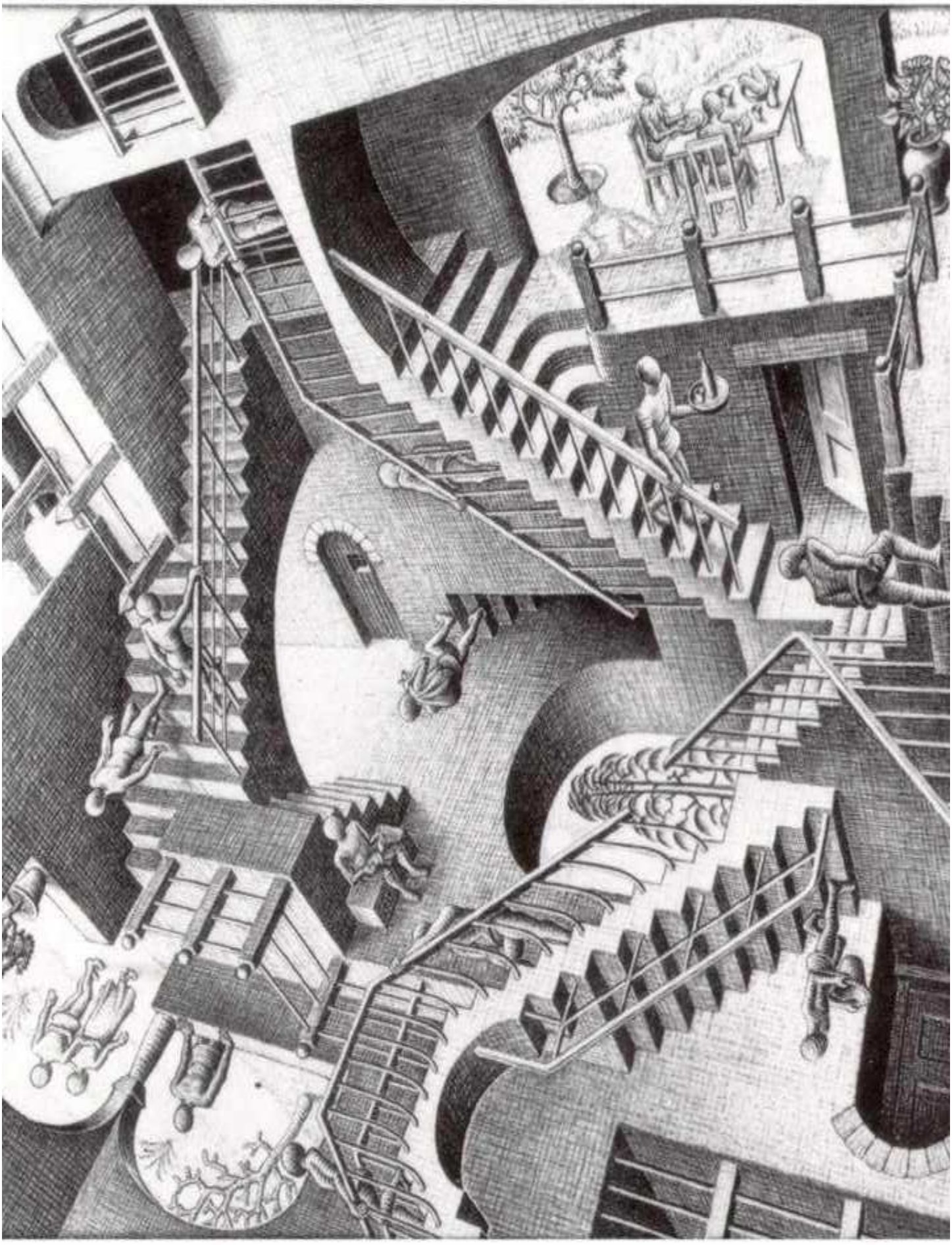
Research questions

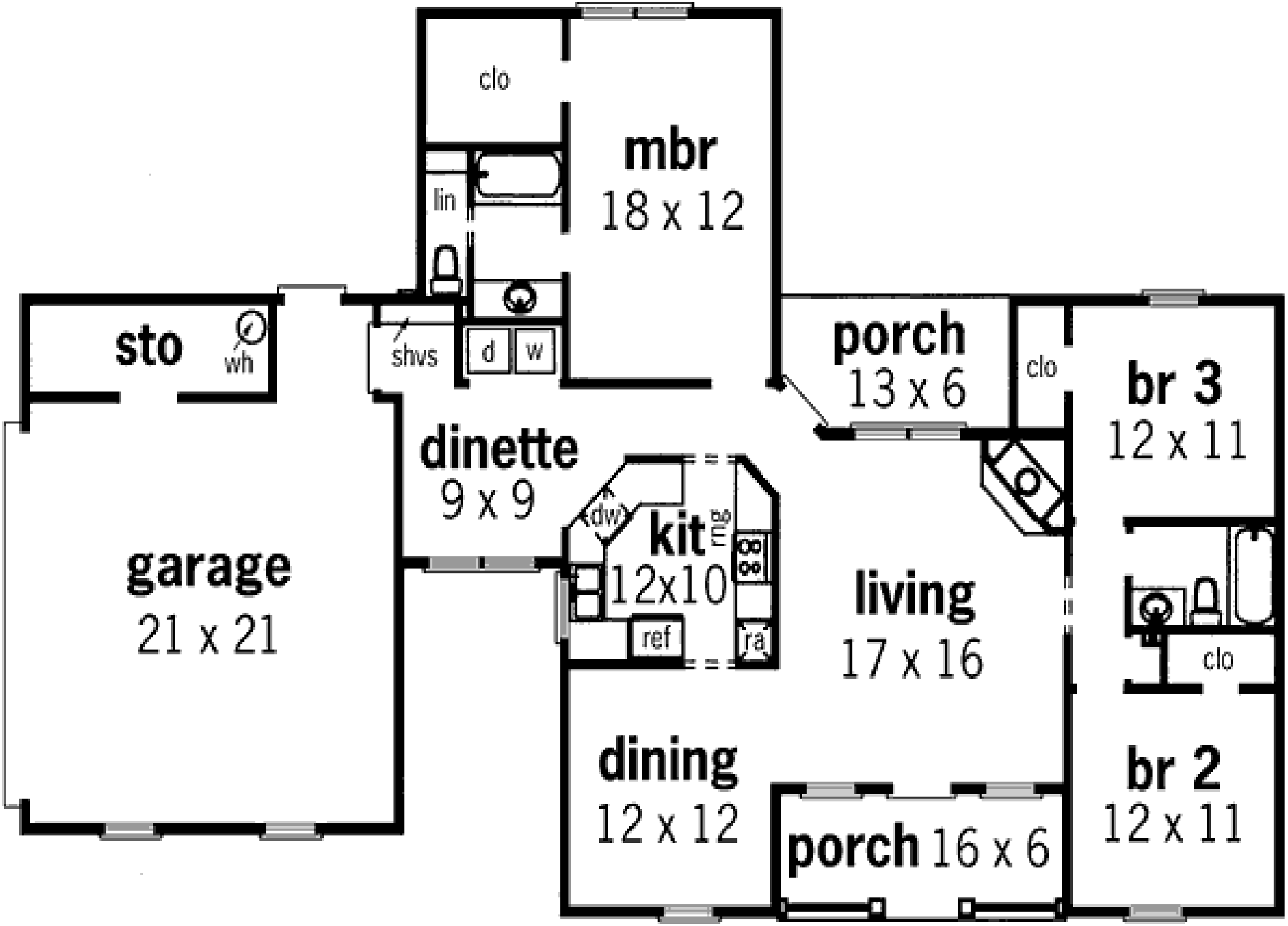
1) Haskell spec of RDF inference program
2) Soundness and completeness
3) What logics?
4) optimization
5) inconsistency

Logic

- Constructive logic

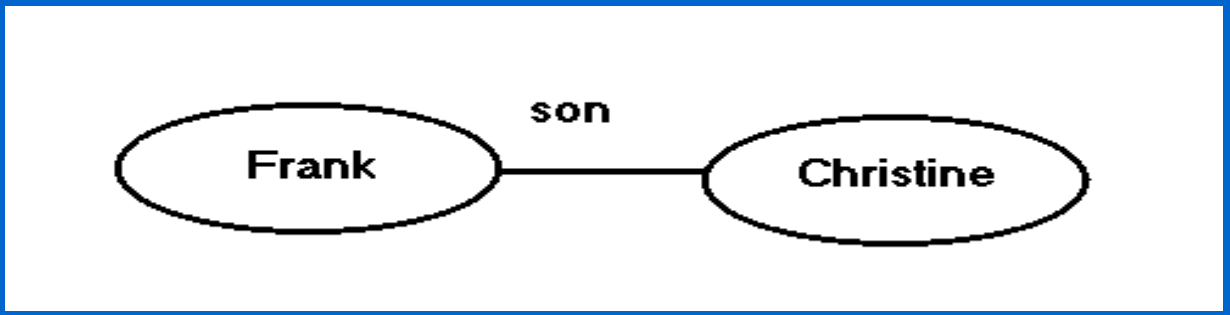
  cfr. Escher - architect
- Each triple in a proof must exist
- Verifiability




clo
mbr
18 x 12
lin
sto
wh
shvs
d
w
porch
13 x 6
clo
br 3
12 x 11
dinette
9 x 9
dw
kit
12x10
mg
ref
ra
garage
21 x 21
living
17 x 16
clo
dining
12 x 12
br 2
12 x 11
porch 16 x 6

RDF Graph

- URI:http://www.daml.org/2001/01/gedcom/gedcom#
- gd:sun(def:Frank,def:Christine).



Research questions

1) Haskell spec of RDF inference program
2) Soundness and completeness
3) What logics?
4) optimization
5) inconsistency

Optimization / DEMO

- Question: grandmother(Z, X)
- Solution: fact -> rule -> …. -> rule

-> solution = grandmother(elza,wim)

- Specific rule:

  t1, t2, t3, .., tn --> grandmother(Z,X)

- child(X,Y), child(Y,Z), gender(Z, female):>

grandmother(Z,X).

- Other

Research questions

1) Haskell spec of RDF inference program
2) Soundness and completeness
3) What logics?
4) optimization
5) inconsistency

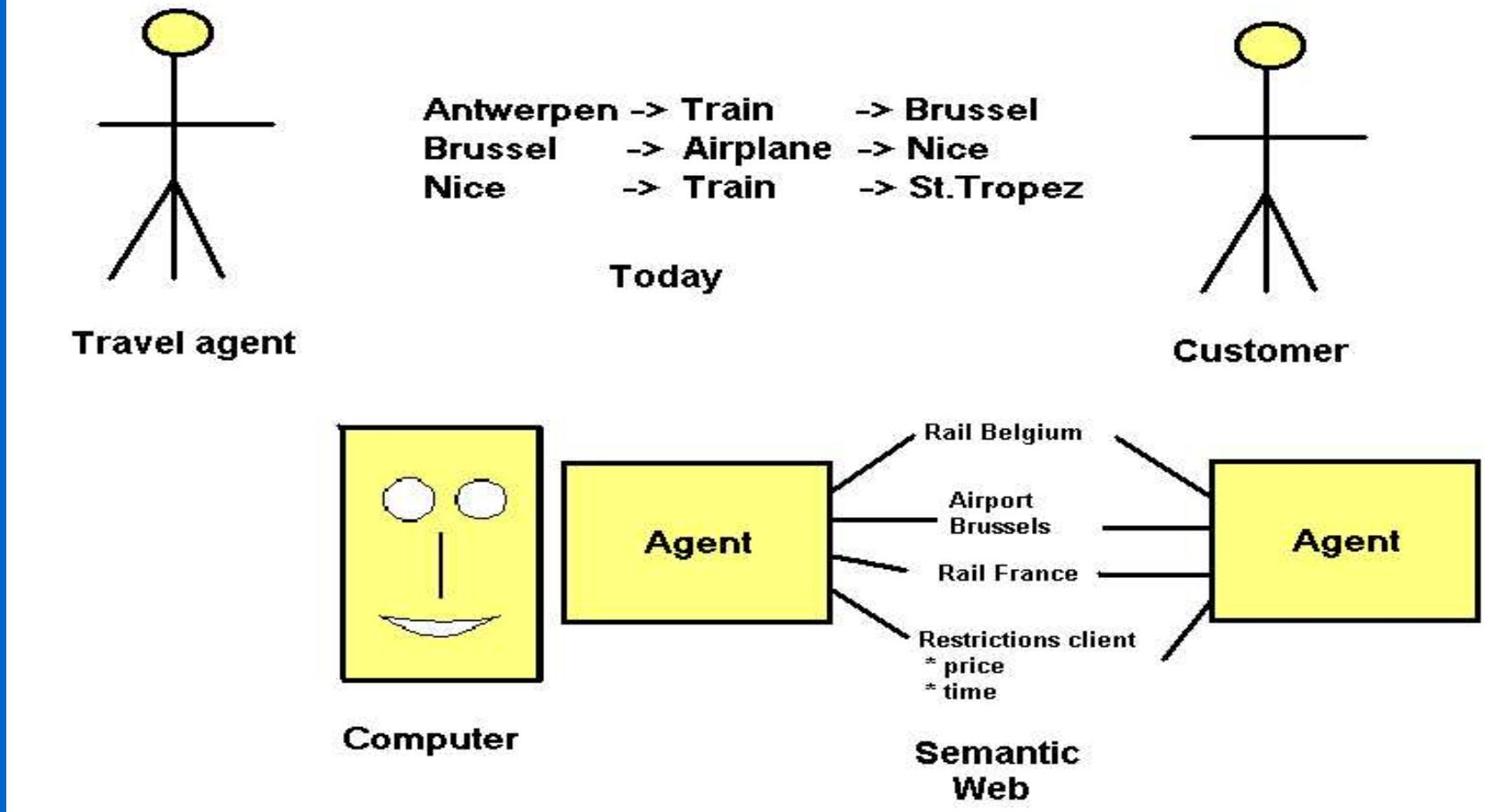
Inconsistencies

- Importance of selected logic
- No Ex Contradictione Quodlibet
- Trust system
- Wait and see

Research questions

1) Haskell spec of RDF inference program
2) Soundness and completeness
3) What logics?
4) optimization
5) inconsistency

Case study


Antwerpen -> Train -> Brussel
Brussel -> Airplane -> Nice
Nice -> Train -> St.Tropez
Today
Travel agent
Customer
Rail Belgium
Airport Brussels
Rail France
Restrictions client
* price
* time
Agent
Agent
Computer
Semantic Web

DEMO

Questions?

DEMO 1

start

Enter command (? for help):

?

h : help

? : help

q : exit

s : perform a single inference step

g : go; stop working interactively

m : menu

e : execute menu item

qe : enter a query

p : print the graphs

DEMO 2

m&&&&
You entered:m&&&&&
Please, try again.
m
Type the command e.g. "e pro11."
pro11 authentication example
pro21 flight simulation
pro31 subClassOf example
pro41 demo example
pro51 gedcom

Enter "e item".

DEMO 3

Enter "e item".
e pro41

RDFProlog version 2.0
Reading files ...
Enter command (? for help):
s
step:
goal:
grandmother(_1?Z,_1?X)./10
goallist:
grandmother(_1?Z,_1?X)./10

substitution:
[]

history:

DEMO 4

s

step:

goal:

grandparent(1$_$_2?Z,1$_$_2?X)./2

goallist:

grandparent(1$_$_2?Z,1$_$_2?X)./2

gender(1$_$_2?Z,female)./2

substitution:

[(_1?Z,1$_$_2?Z)(_1?X,1$_$_2?X)]

history:

(grandparent(1$_$_2?Z,1$_$_2?X),gender(1$_$_2?Z,female) :>
grandmother(1$_$_2?Z,1$_$_2?X)./2,grandmother(_1?Z,_1?X)./10)

DEMO 5

s

step:

goal:

child(2$_$_1?X,2$_$_1?Y)./2

goallist:

child(2$_$_1?X,2$_$_1?Y)./2

child(2$_$_1?Y,2$_$_1?Z)./2

gender(1$_$_2?Z,female)./2

substitution:

[(_1?Z,1$_$_2?Z)(_1?X,1$_$_2?X)(1$_$_2?Z,2$_$_1?Z)(1$_$_2?X,2$_$_1?X)]

history:

(child(2$_$_1?X,2$_$_1?Y),child(2$_$_1?Y,2$_$_1?Z) :>
grandparent(2$_$_1?Z,2$_$_1?X)./2,grandparent(1$_$_2?Z,1$_$_2?X)./2)

(grandparent(1$_$_2?Z,1$_$_2?X),gender(1$_$_2?Z,female) :>
grandmother(1$_$_2?Z,1$_$_2?X)./2,grandmother(_1?Z,_1?X)./10)

DEMO 6

s

step:

goal:

/0

goallist:

/0

child(2$_$_1?Y,2$_$_1?Z)./2

gender(1$_$_2?Z,female)./2

substitution:

[(_1?Z,1$_$_2?Z)(_1?X,1$_$_2?X)(1$_$_2?Z,2$_$_1?Z)(1$_$_2?X,2$_$_1?X)(2$_$_1?X,christine)(2$_$_1?Y,elza)]

history:

(child(christine,elza)./2,child(2$_$_1?X,2$_$_1?Y)./2)

(child(2$_$_1?X,2$_$_1?Y),child(2$_$_1?Y,2$_$_1?Z) :>
grandparent(2$_$_1?Z,2$_$_1?X)./2,grandparent(1$_$_2?Z,1$_$_2?X)./2)

(grandparent(1$_$_2?Z,1$_$_2?X),gender(1$_$_2?Z,female) :>
grandmother(1$_$_2?Z,1$_$_2?X)./2,grandmother(_1?Z,_1?X)./10)

DEMO 7

s

step:

goal:

child(2$_$_1?Y,2$_$_1?Z)./2

goallist:

child(2$_$_1?Y,2$_$_1?Z)./2

gender(1$_$_2?Z,female)./2

substitution:

[(_1?Z,1$_$_2?Z)(_1?X,1$_$_2?X)(1$_$_2?Z,2$_$_1?Z)(1$_$_2?X,2$_$_1?X)(2$_$_1?X,christine)(2$_$_1?Y,elza)]

history:

(child(christine,elza)./2,child(2$_$_1?X,2$_$_1?Y)./2)

(child(2$_$_1?X,2$_$_1?Y),child(2$_$_1?Y,2$_$_1?Z) :>
grandparent(2$_$_1?Z,2$_$_1?X)./2,grandparent(1$_$_2?Z,1$_$_2?X)./2)

(grandparent(1$_$_2?Z,1$_$_2?X),gender(1$_$_2?Z,female) :>
grandmother(1$_$_2?Z,1$_$_2?X)./2,grandmother(_1?Z,_1?X)./10)

DEMO 8

s

**** No unification; path aborted. ****

s

**** Backtrack done. ****

s

DEMO 9

s

step:

goal:

/0

goallist:

/0

child(2$_$_1?Y,2$_$_1?Z)./2

gender(1$_$_2?Z,female)./2

substitution:

[(_1?Z,1$_$_2?Z)(_1?X,1$_$_2?X)(1$_$_2?Z,2$_$_1?Z)(1$_$_2?X,2$_$_1?X)(2$_$_1?X,wim)(2$_$_
1?Y,christine)]

history:

(child(wim,christine)./2,child(2$_$_1?X,2$_$_1?Y)./2)

(child(2$_$_1?X,2$_$_1?Y),child(2$_$_1?Y,2$_$_1?Z) :>
grandparent(2$_$_1?Z,2$_$_1?X)./2,grandparent(1$_$_2?Z,1$_$_2?X)./2)

(grandparent(1$_$_2?Z,1$_$_2?X),gender(1$_$_2?Z,female) :>
grandmother(1$_$_2?Z,1$_$_2?X)./2,grandmother(_1?Z,_1?X)./10)

DEMO 10

s

step:

goal:

child(2$_$_1?Y,2$_$_1?Z)./2

goallist:

child(2$_$_1?Y,2$_$_1?Z)./2

gender(1$_$_2?Z,female)./2

substitution:

[(_1?Z,1$_$_2?Z)(_1?X,1$_$_2?X)(1$_$_2?Z,2$_$_1?Z)(1$_$_2?X,2$_$_1?X)(2$_$_1?X,wim)(2$_$_1?Y,christine)]

history:

(child(wim,christine)./2,child(2$_$_1?X,2$_$_1?Y)./2)

(child(2$_$_1?X,2$_$_1?Y),child(2$_$_1?Y,2$_$_1?Z) :>
grandparent(2$_$_1?Z,2$_$_1?X)./2,grandparent(1$_$_2?Z,1$_$_2?X)./2)

(grandparent(1$_$_2?Z,1$_$_2?X),gender(1$_$_2?Z,female) :>
grandmother(1$_$_2?Z,1$_$_2?X)./2,grandmother(_1?Z,_1?X)./10)

DEMO 11

s

step:

goal:

/0

goallist:

/0

gender(1$_$_2?Z,female)./2

substitution:

[(_1?Z,1$_$_2?Z)(_1?X,1$_$_2?X)(1$_$_2?Z,2$_$_1?Z)(1$_$_2?X,2$_$_1?X)(2$_$_1?X,wim)(2$_$_1?Y,christine)(2$_$_1?Z,elza)]

history:

(child(christine,elza)./2,child(christine,2$_$_1?Z)./2)

(child(wim,christine)./2,child(2$_$_1?X,2$_$_1?Y)./2)

(child(2$_$_1?X,2$_$_1?Y),child(2$_$_1?Y,2$_$_1?Z) :>
grandparent(2$_$_1?Z,2$_$_1?X)./2,grandparent(1$_$_2?Z,1$_$_2?X)./2)

(grandparent(1$_$_2?Z,1$_$_2?X),gender(1$_$_2?Z,female) :>
grandmother(1$_$_2?Z,1$_$_2?X)./2,grandmother(_1?Z,_1?X)./10)

DEMO 12

s

step:

goal:

gender(1$_$_2?Z,female)./2

goallist:

gender(1$_$_2?Z,female)./2

substitution:

[(_1?Z,1$_$_2?Z)(_1?X,1$_$_2?X)(1$_$_2?Z,2$_$_1?Z)(1$_$_2?X,2$_$_1?X)(2$_$_1?X,wim)(2$_$_1?Y,christine)(2$_$_1?Z,elza)]

history:

(child(christine,elza)./2,child(christine,2$_$_1?Z)./2)

(child(wim,christine)./2,child(2$_$_1?X,2$_$_1?Y)./2)

(child(2$_$_1?X,2$_$_1?Y),child(2$_$_1?Y,2$_$_1?Z) :>
grandparent(2$_$_1?Z,2$_$_1?X)./2,grandparent(1$_$_2?Z,1$_$_2?X)./2)

(grandparent(1$_$_2?Z,1$_$_2?X),gender(1$_$_2?Z,female) :>
grandmother(1$_$_2?Z,1$_$_2?X)./2,grandmother(_1?Z,_1?X)./10)

DEMO 13

s

step:

goal:

/0

goallist:

/0

substitution:

[(_1?Z,1$_$_2?Z)(_1?X,1$_$_2?X)(1$_$_2?Z,2$_$_1?Z)(1$_$_2?X,2$_$_1?X)(2$_$_1?X,wim)(2$_$_
1?Y,christine)(2$_$_1?Z,elza)]

history:

(gender(elza,female)./2,gender(elza,female)./2)

(child(christine,elza)./2,child(christine,2$_$_1?Z)./2)

(child(wim,christine)./2,child(2$_$_1?X,2$_$_1?Y)./2)

(child(2$_$_1?X,2$_$_1?Y),child(2$_$_1?Y,2$_$_1?Z) :>
grandparent(2$_$_1?Z,2$_$_1?X)./2,grandparent(1$_$_2?Z,1$_$_2?X)./2)

(grandparent(1$_$_2?Z,1$_$_2?X),gender(1$_$_2?Z,female) :>
grandmother(1$_$_2?Z,1$_$_2?X)./2,grandmother(_1?Z,_1?X)./10)

DEMO 14

solution:

Solution:

Substitution:

[(_1?Z,1$_$_2?Z)(_1?X,1$_$_2?X)(1$_$_2?Z,2$_$_1?Z)(1$_$_2?X,2$_$_1?X)(2$_$_1?X,wim)(2$_$_1?Y,christine)(2$_$_1?Z,elza)]

Proof:

(gender(elza,female)./2,gender(elza,female)./2)

(child(christine,elza)./2,child(christine,elza)./2)

(child(wim,christine)./2,child(wim,christine)./2)

(child(wim,christine),child(christine,elza) :> grandparent(elza,wim)./2,grandparent(elza,wim)./2)

(grandparent(elza,wim),gender(elza,female) :> grandmother(elza,wim)./2,grandmother(elza,wim)./10)

General rule:

"gender(X2,female),child(X1,X2),child(X3,X1) :> grandmother(X2,X3)."

Variable numbering 1

Enter "e item".
e pro41

RDFProlog version 2.0
Reading files ...
Enter command (? for help):
s
step:
goal:
grandmother(_1?Z,_1?X)./10
goallist:
grandmother(_1?Z,_1?X)./10

substitution:
[]

history:

Variable numbering 2

s

step:

goal:

grandparent(1$_$_2?Z,1$_$_2?X)./2

goallist:

grandparent(1$_$_2?Z,1$_$_2?X)./2

gender(1$_$_2?Z,female)./2

substitution:

[(_1?Z,1$_$_2?Z)(_1?X,1$_$_2?X)]

history:

(grandparent(1$_$_2?Z,1$_$_2?X),gender(1$_$_2?Z,female) :>
grandmother(1$_$_2?Z,1$_$_2?X)./2,grandmother(_1?Z,_1?X)./10)