Advanced OWL

John Beverley

Assistant Professor, *University at Buffalo*
Co-Director, National Center for Ontological Research
Affiliate Faculty, *Institute of Artificial Intelligence and Data Science*

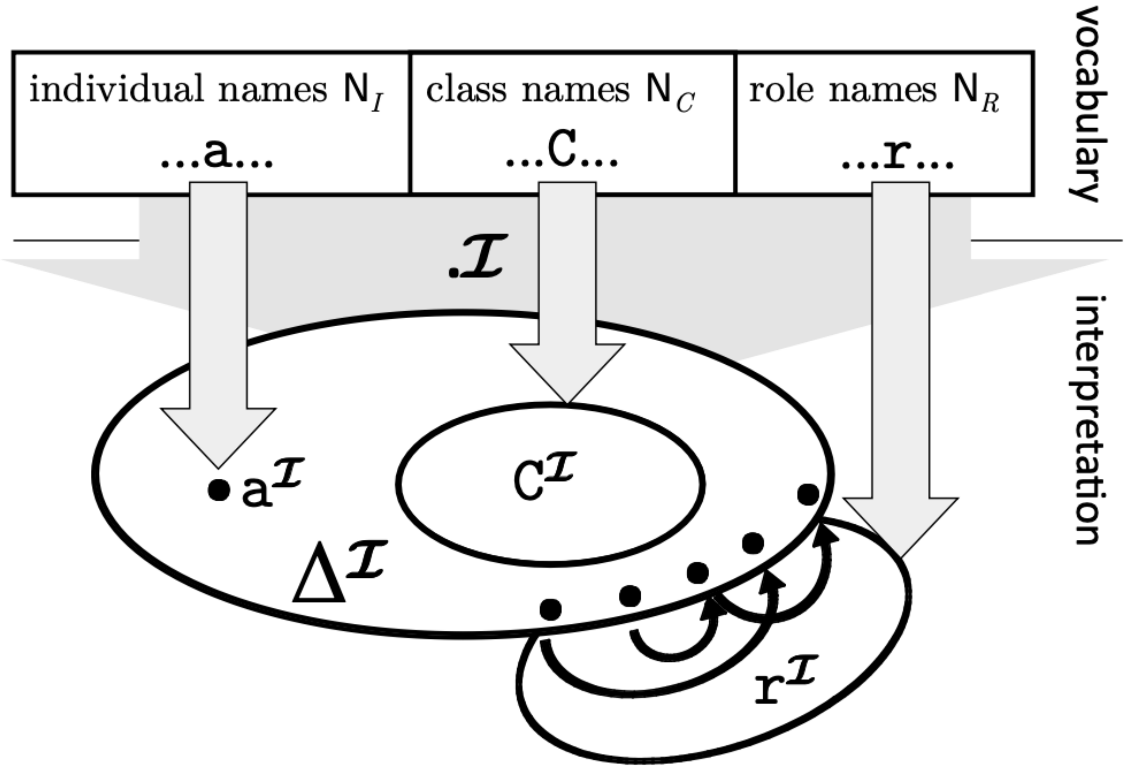
Outline

- Exercise: DL Interpretation
- Tableau Reasoning
- Exercise: Zebra Puzzle
- Role Constraint Strategies

Outline

- Exercise: DL Interpretation
- Tableau Reasoning
- Exercise: Zebra Puzzle
- Role Constraint Strategies

Description Logic Interpretations

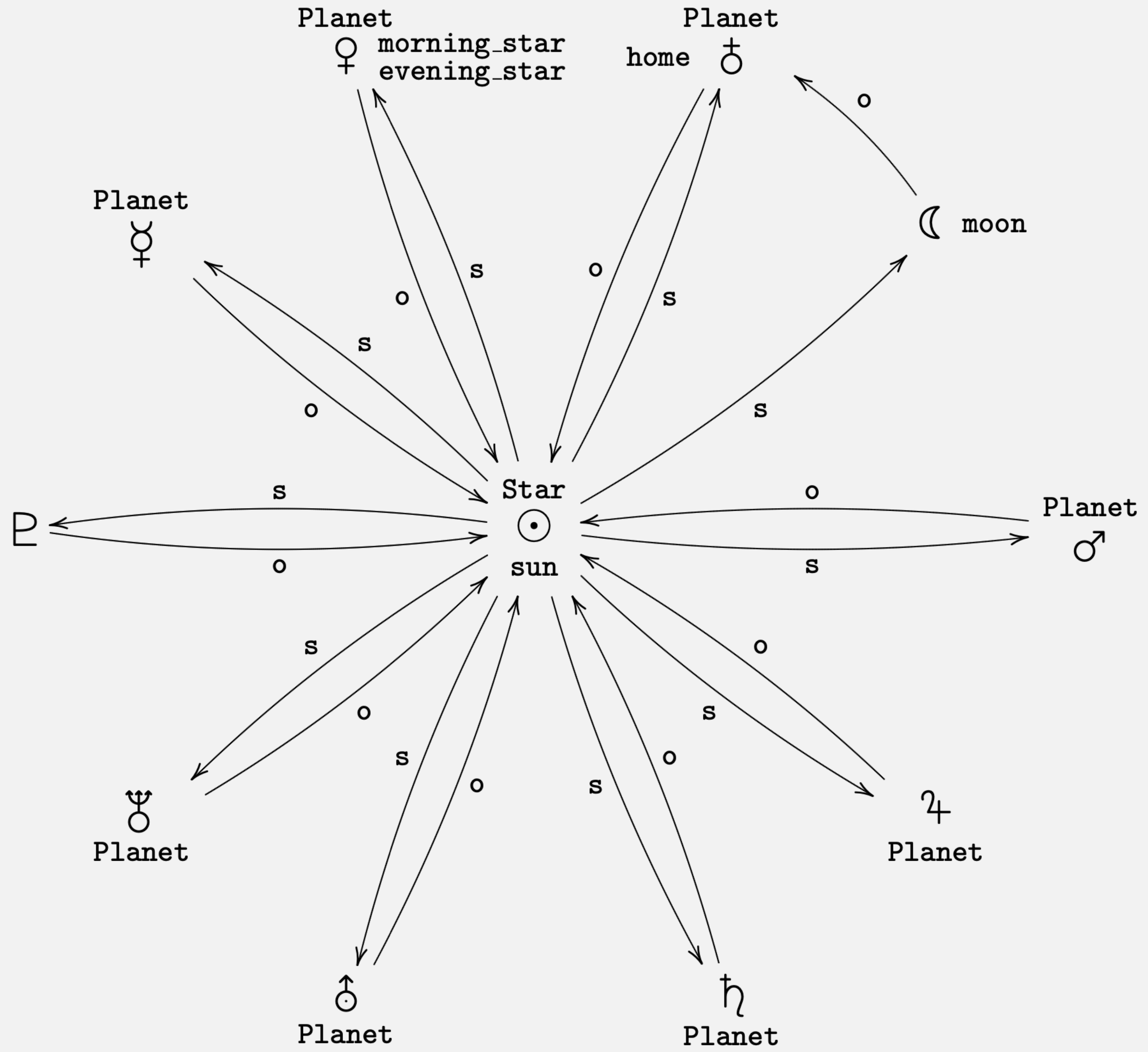


Example: Diagram a Model

- $\mathsf{N}_I = \{\mathtt{sun}, \mathtt{morning\_star}, \mathtt{evening\_star}, \mathtt{moon}, \mathtt{home}\}$.
- $\mathsf{N}_C = \{\mathtt{Planet}, \mathtt{Star}\}$.
- $\mathsf{N}_R = \{\mathtt{orbitsAround}, \mathtt{shinesOn}\}$.

We now define an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ as follows: Let our domain $\Delta^{\mathcal{I}}$ contain the following elements: $\astrosun, \mercury, \venus, \earth, \leftmoon, \mars, \jupiter, \saturn, \uranus, \neptune, \pluto$. We define the interpretation function by

$$
\begin{aligned}
\mathtt{sun}^{\mathcal{I}} &= \astrosun & \mathtt{Planet}^{\mathcal{I}} &= \{\mercury, \venus, \earth, \mars, \jupiter, \saturn, \uranus, \neptune\} \\
\mathtt{morning\_star}^{\mathcal{I}} &= \venus & \mathtt{Star}^{\mathcal{I}} &= \{\astrosun\} \\
\mathtt{evening\_star}^{\mathcal{I}} &= \venus & \mathtt{orbitsAround}^{\mathcal{I}} &= \{\langle\mercury, \astrosun\rangle, \langle\venus, \astrosun\rangle, \langle\earth, \astrosun\rangle, \langle\mars, \astrosun\rangle, \langle\jupiter, \astrosun\rangle, \\
\mathtt{moon}^{\mathcal{I}} &= \leftmoon & &\qquad \langle\saturn, \astrosun\rangle, \langle\uranus, \astrosun\rangle, \langle\neptune, \astrosun\rangle, \langle\pluto, \astrosun\rangle, \langle\leftmoon, \earth\rangle\} \\
\mathtt{home}^{\mathcal{I}} &= \earth & \mathtt{shinesOn}^{\mathcal{I}} &= \{\langle\astrosun, \mercury\rangle, \langle\astrosun, \venus\rangle, \langle\astrosun, \earth\rangle, \langle\astrosun, \leftmoon\rangle, \langle\astrosun, \mars\rangle, \\
& & &\qquad \langle\astrosun, \jupiter\rangle, \langle\astrosun, \saturn\rangle, \langle\astrosun, \uranus\rangle, \langle\astrosun, \neptune\rangle, \langle\astrosun, \pluto\rangle\}
\end{aligned}
$$


Planet
morning_star
evening_star
Planet
home
o
Planet
moon
s
o
o
s
s
o
s
Star
sun
s
o
o
Planet
s
s
o
o
s
o
s
s
o
s
o
Planet
Planet
Planet
Planet

Outline

- Exercise: DL Interpretation
- Tableau Reasoning
- Exercise: Zebra Puzzle
- Role Constraint Strategies

Styles of Automated Reasoning

- Logicians have for decades leveraged automated theorem provers and model checkers to explore logical space

- Most run by leveraging **resolution**, or **unification**, or **tableau algorithms**

Prover9

```
============================== PROOF ==========

% -------- Comments from original proof -------
% Proof 1 at 0.00 (+ 0.03) seconds.
%  ngth of proof is 7.
%  el of proof is 3.
%  imum clause weight is 2.
%  en clauses 2.

  > Q # label(non_clause).   [assumption].
  # label(non_clause) # label(goal).   [goal].
3 -P | Q.   [clausify(1)].
4 P.   [assumption].
5 -Q.   [deny(2)].
6 Q.   [ur(3,a,4,a)].
7 $F.   [resolve(6,a,5,a)].

============================== end of proof ===
```

Practical Reasoning for Expressive Description Logics

Ian Horrocks[1], Ulrike Sattler[2], and Stephan Tobies[2]

[1] Department of Computer Science, University of Manchester[†]
[2] LuFG Theoretical Computer Science, RWTH Aachen[‡]

Abstract. Description Logics (DLs) are a family of knowledge representation formalisms mainly characterised by constructors to build complex concepts and roles from atomic ones. Expressive role constructors are important in many applications, but can be computationally problematical. We present an algorithm that decides satisfiability of the DL $\mathcal{ALC}$ extended with transitive and inverse roles, role hierarchies, and qualify-

HermiT OWL Reasoner

The New Kid on the OWL Block

Reasoning Algorithms

Davis-Putnam Algorithm – Recursively assign truth-values to variables of a proposition, if reach a contradiction then backtrack and try new assignment, terminate when assignment is found for all variables or all possibilities are exhausted

Analytic Tableau - Construct a tree where each branch represents an interpretation of a proposition, decompose proposition by extended branches with new interpretations on each, close branch if reach a contradiction, terminate when all branches closed or one full branch open

Reasoning Algorithms

Davis-Putnam Algorithm – Recursively assign truth-values to variables of a proposition, if reach a contradiction then backtrack and try new assignment, terminate when assignment is found for all variables or all possibilities are exhausted

Analytic Tableau - Construct a tree where each branch represents an interpretation of a proposition, decompose proposition by extended branches with new interpretations on each, close branch if reach a contradiction, terminate when all branches closed or one full branch open

Reasoner	Algorithm	Strengths	Weaknesses	Use Cases
HermiT	Hypertableau	Supports complex DL reasoning	High memory usage; slower on large ontologies	Highly expressive ontologies with complex DL features
Pellet	Tableau	SWRL support, datatype reasoning	Slower on large ontologies; performance can degrade	Reasoning with rules (SWRL), expressive DL ontologies
Fact++	Optimized Tableau	Fast for moderately complex ontologies	Not ideal for very large ontologies	OWL DL reasoning with moderate complexity
ELK	OWL 2 EL optimized	Extremely fast for large OWL 2 EL ontologies	Limited to OWL 2 EL; no inverse roles	Large-scale biomedical ontologies (e.g., SNOMED CT)

Tableau

The $\rightarrow_{\sqcap}$-rule

***Condition*:** $\mathcal{A}$ contains $(C_1 \sqcap C_2)(x)$, but not both $C_1(x)$ and $C_2(x)$.

***Action*:** $\mathcal{A}' := \mathcal{A} \cup \{C_1(x), C_2(x)\}$.

The $\rightarrow_{\sqcup}$-rule

***Condition*:** $\mathcal{A}$ contains $(C_1 \sqcup C_2)(x)$, but neither $C_1(x)$ nor $C_2(x)$.

***Action*:** $\mathcal{A}' := \mathcal{A} \cup \{C_1(x)\}$, $\mathcal{A}'' := \mathcal{A} \cup \{C_2(x)\}$.

The $\rightarrow_{\exists}$-rule

***Condition*:** $\mathcal{A}$ contains $(\exists r.C)(x)$, but there is no individual name z such that $C(z)$ and $r(x, z)$ are in $\mathcal{A}$.

***Action*:** $\mathcal{A}' := \mathcal{A} \cup \{C(y), r(x, y)\}$ where y is an individual name not occurring in $\mathcal{A}$.

The $\rightarrow_{\forall}$-rule

***Condition*:** $\mathcal{A}$ contains $(\forall r.C)(x)$ and $r(x, y)$, but it does not contain $C(y)$.

***Action*:** $\mathcal{A}' := \mathcal{A} \cup \{C(y)\}$.

Tableau

- **Conjunction Rule**: If given (C_1 & C_2) on branch b then add C_1 and C_2 to b
- **Disjunction Rule**: If given (C_1 v C_2) on branch b_1 then add C_1 to new branch b_1 and C_2 to new branch b_2
- **Existential Rule**: If $\forall x \exists y$ r(x,y) & Cy on branch b_1 then add r(x,a) and Ca to a new branch, as long as "a" fresh
- **Universal Rule**: $\forall x \forall y$ r(x,y) → Cy & r(x,y) on b then add Cy to b

Tableau

$\sqcap$**-rule**
- **Conjunction Rule**: If given (C_1 & C_2) on branch b then add C_1 and C_2 to b

$\sqcup$**-rule**
- **Disjunction Rule**: If given (C_1 v C_2) on branch b_1 then add C_1 to new branch b_1 and C_2 to new branch b_2

$\exists$-rule
- **Existential Rule**: If $\forall x \exists y$ r(x,y) & Cy on branch b_1 then add r(x,a) and Ca to a new branch, as long as "a" fresh

$\forall$-rule
- **Universal Rule**: $\forall x \forall y$ r(x,y) → Cy & r(x,y) on b then add Cy to b

Consistency Checking

- Is the following set of propositions **consistent**?

$$\exists y\ s(a,x)\ \&\ F(x)$$

$$s(a,b)$$

$$\forall y\ s(a,y) \rightarrow \neg F(y) \vee \neg B(y)$$

$$B(b)$$

Davis-Putnam Algorithm

- Is the following set of propositions **consistent**?

$$\exists y\ s(a,x)\ \&\ F(x)$$

$$s(a,b)$$

$$\forall y\ s(a,y) \rightarrow \neg F(y) \vee \neg B(y)$$

$$B(b)$$

Davis-Putnam Algorithm

- If so, propositions in each subset can be assigned true without contradiction

Method of Analytic Tableau

- Is the following set of propositions **consistent**?

$$\exists y\ s(a,x)\ \&\ F(x)$$

$$s(a,b)$$

$$\forall y\ s(a,y) \rightarrow \neg F(y) \vee \neg B(y)$$

$$B(b)$$

Tableau Algorithm

- If so, each can be decomposed into a tree in which one branch is open

$\exists y(s(a,x)\ \&\ F(x))$, $s(a,b)$, $\forall y\ s(a,y) \rightarrow \neg F(y) \vee \neg B(y)$, $B(b)$

∃-rule $\qquad$ $\mathbf{s(a,d)\ \&\ F(d)}$

$\exists y(s(a,x)\ \&\ F(x))$, $s(a,b)$, $\forall y\ s(a,y) \rightarrow \neg F(y) \vee \neg B(y)$, $B(b)$

∃-rule

⊓-rule

s(a,d) & F(d)

s(a,d)

F(d)

$\exists y(s(a,x)\ \&\ F(x)),\ \ s(a,b),\ \ \forall y\ s(a,y) \rightarrow \neg F(y) \vee \neg B(y),\ B(b)$

$\exists$-rule **s(a,d) & F(d)**

$\sqcap$-rule **s(a,d)**

F(d)

$\forall$-rule **$s(a,b) \rightarrow \neg F(b) \vee \neg B(b)$**

$s(a,d) \rightarrow \neg F(d) \vee \neg B(d)$

$\exists y(s(a,x) \,\&\, F(x))$, $s(a,b)$, $\forall y\, s(a,y) \rightarrow \neg F(y) \vee \neg B(y)$, $B(b)$

$\exists$-rule	$\mathbf{s(a,d) \,\&\, F(d)}$
$\sqcap$-rule	$\mathbf{s(a,d)}$
	$\mathbf{F(d)}$
$\forall$-rule	$\mathbf{s(a,b) \rightarrow \neg F(b) \vee \neg B(b)}$
	$\mathbf{s(a,d) \rightarrow \neg F(d) \vee \neg B(d)}$

∃y(s(a,x) & F(x)), s(a,b), ∀y s(a,y)→¬F(y)v¬B(y), B(b)

∃-rule	**s(a,d) & F(d)**
⊓-rule	**s(a,d)**
	F(d)
∀-rule	**s(a,b) → ¬F(b) v ¬B(b)**
	s(a,d) → ¬F(d) v ¬B(d)
	¬F(b) v ¬B(b)

$\exists y(s(a,x)\ \&\ F(x)),\ \ s(a,b),\ \ \forall y\ s(a,y) \rightarrow \neg F(y) \vee \neg B(y),\ B(b)$

$\exists$-rule	$\mathbf{s(a,d)\ \&\ F(d)}$
$\sqcap$-rule	$\mathbf{s(a,d)}$
	$\mathbf{F(d)}$
$\forall$-rule	$\mathbf{s(a,b) \rightarrow \neg F(b) \vee \neg B(b)}$
	$\mathbf{s(a,d) \rightarrow \neg F(d) \vee \neg B(d)}$
	$\mathbf{\neg F(b) \vee \neg B(b)}$
	$\mathbf{\neg F(d) \vee \neg B(d)}$

$$\exists y(s(a,x)\ \&\ F(x)),\ \ s(a,b),\ \ \forall y\ s(a,y) \rightarrow \neg F(y) \vee \neg B(y),\ B(b)$$

$\exists$-rule	**$s(a,d)\ \&\ F(d)$**
	$s(a,d)$
$\sqcap$-rule	**$F(d)$**
	$s(a,b) \rightarrow \neg F(b) \vee \neg B(b)$
$\forall$-rule	**$s(a,d) \rightarrow \neg F(d) \vee \neg B(d)$**
	$\neg F(b) \vee \neg B(b)$
	$\neg F(d) \vee \neg B(d)$
$\sqcup$-rule	**$\neg F(b)$** **$\neg B(b)$**

$\exists y(s(a,x)\ \&\ F(x)),\ \ s(a,b),\ \ \forall y\ s(a,y) \rightarrow \neg F(y) \vee \neg B(y),\ B(b)$

$\exists$-rule	**$s(a,d)\ \&\ F(d)$**	
$\sqcap$-rule	**$s(a,d)$**	
	$F(d)$	
$\forall$-rule	**$s(a,b) \rightarrow \neg F(b) \vee \neg B(b)$**	
	$s(a,d) \rightarrow \neg F(d) \vee \neg B(d)$	
	$\neg F(b) \vee \neg B(b)$	
	$\neg F(d) \vee \neg B(d)$	
$\sqcup$-rule	**$\neg F(b)$**	**$\neg B(b)$**
		✕

∃y(s(a,x) & F(x)), s(a,b), ∀y s(a,y)→¬F(y)v¬B(y), B(b)

∃-rule	**s(a,d) & F(d)**	
⊓-rule	**s(a,d)**	
	F(d)	
∀-rule	**s(a,b) → ¬F(b) v ¬B(b)**	
	s(a,d) → ¬F(d) v ¬B(d)	
	¬F(b) v ¬B(b)	
	¬F(d) v ¬B(d)	
⊔-rule	**¬F(b)**	**¬B(b)**
⊔-rule	**¬F(d)** **¬B(d)**	

$\exists y(s(a,x) \,\&\, F(x))$, $s(a,b)$, $\forall y\; s(a,y) \rightarrow \neg F(y) \vee \neg B(y)$, $B(b)$

∃-rule **s(a,d) & F(d)**

⊓-rule **s(a,d)**

F(d)

∀-rule **s(a,b) → ¬F(b) v ¬B(b)**

s(a,d) → ¬F(d) v ¬B(d)

¬F(b) v ¬B(b)

¬F(d) v ¬B(d)

⊔-rule **¬F(b)** | **¬B(b)**

⊔-rule **¬F(d)** | **¬B(d)**

$\exists y(s(a,x)\ \&\ F(x)),\ \ s(a,b),\ \ \forall y\ s(a,y) \rightarrow \neg F(y) v \neg B(y),\ B(b)$

∃-rule	**s(a,d) & F(d)**	
⊓-rule	**s(a,d)** **F(d)**	
∀-rule	**s(a,b) → ¬F(b) v ¬B(b)** **s(a,d) → ¬F(d) v ¬B(d)** **¬F(b) v ¬B(b)** **¬F(d) v ¬B(d)**	
⊔-rule	**¬F(b)**	**¬B(b)**
⊔-rule	**¬F(d)** **¬B(d)**	

Outline

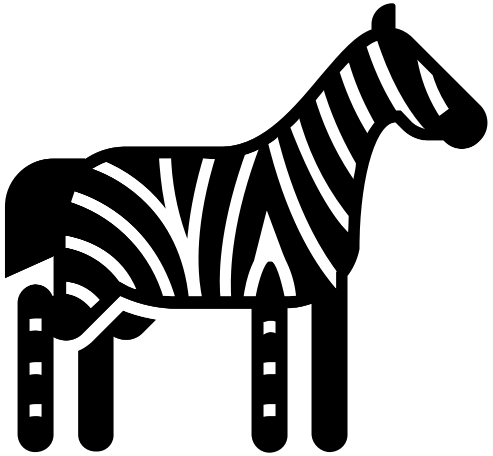
- Exercise: DL Interpretation
- Tableau Reasoning
- Exercise: Zebra Puzzle
- Role Constraint Strategies

Zebra Puzzle

1. There are five houses.
2. The Englishman lives in the red house.
3. The Spaniard owns the dog.
4. Coffee is drunk in the green house.
5. The Ukrainian drinks tea.
6. The green house is immediately to the right of the ivory house.
7. The Old Gold smoker owns snails.
8. Kools are smoked in the yellow house.
9. Milk is drunk in the middle house.
10. The Norwegian lives in the first house.
11. The man who smokes Chesterfields lives in the house next to the man with the fox.
12. Kools are smoked in a house next to the house where the horse is kept.
13. The Lucky Strike smoker drinks orange juice.
14. The Japanese man smokes Parliaments.
15. The Norwegian lives next to the blue house.

Zebra Puzzle

1. There are five houses.
2. The Englishman lives in the red house.
3. The Spaniard owns the dog.
4. Coffee is drunk in the green house.
5. The Ukrainian drinks tea.
6. The green house is immediately to the right of the ivory house.
7. The Old Gold smoker owns snails.
8. Kools are smoked in the yellow house.
9. Milk is drunk in the middle house.
10. The Norwegian lives in the first house.
11. The man who smokes Chesterfields lives in the house next to the man with the fox.
12. Kools are smoked in a house next to the house where the horse is kept.
13. The Lucky Strike smoker drinks orange juice.
14. The Japanese man smokes Parliaments.
15. The Norwegian lives next to the blue house.



WHO OWNS THE ZEBRA?

Zebra Puzzle

1. There are five houses.
2. The Englishman lives in the red house.
3. The Spaniard owns the dog.
4. Coffee is drunk in the green house.
5. The Ukrainian drinks tea.
6. The green house is immediately to the right of the ivory house.
7. The Old Gold smoker owns snails.
8. Kools are smoked in the yellow house.
9. Milk is drunk in the middle house.
10. The Norwegian lives in the first house.
11. The man who smokes Chesterfields lives in the house next to the man with the fox.
12. Kools are smoked in a house next to the house where the horse is kept.
13. The Lucky Strike smoker drinks orange juice.
14. The Japanese man smokes Parliaments.
15. The Norwegian lives next to the blue house.

REPRESENT THE PUZZLE IN PROTEGE

Zebra Puzzle

1. There are five houses.
2. The Englishman lives in the red house.
3. The Spaniard owns the dog.
4. Coffee is drunk in the green house.
5. The Ukrainian drinks tea.
6. The green house is immediately to the right of the ivory house.
7. The Old Gold smoker owns snails.
8. Kools are smoked in the yellow house.
9. Milk is drunk in the middle house.
10. The Norwegian lives in the first house.
11. The man who smokes Chesterfields lives in the house next to the man with the fox.
12. Kools are smoked in a house next to the house where the horse is kept.
13. The Lucky Strike smoker drinks orange juice.
14. The Japanese man smokes Parliaments.
15. The Norwegian lives next to the blue house.

REPRESENT THE PUZZLE IN PROTEGE

IF DONE CORRECTLY, THE REASONER WILL INFER WHO OWNS THE ZEBRA

Zebra Puzzle

1. **Houses = {red} ⊔ {green} ⊔ {ivory} ⊔ {yellow} ⊔ {blue}**
2. The Englishman lives in the red house.
3. The Spaniard owns the dog.
4. Coffee is drunk in the green house.
5. The Ukrainian drinks tea.
6. The green house is immediately to the right of the ivory house.
7. The Old Gold smoker owns snails.
8. Kools are smoked in the yellow house.
9. Milk is drunk in the middle house.
10. The Norwegian lives in the first house.
11. The man who smokes Chesterfields lives in the house next to the man with the fox.
12. Kools are smoked in a house next to the house where the horse is kept.
13. The Lucky Strike smoker drinks orange juice.
14. The Japanese man smokes Parliaments.
15. The Norwegian lives next to the blue house.

HINT…

Outline

- Exercise: DL Interpretation
- Tableau Reasoning
- Exercise: Zebra Puzzle
- Role Constraint Strategies

Strategy

- Suppose you are checking combinations of role constraints on object properties in OWL2 with the direct semantics

- For example, you may be exploring what pairs can be expected in the set defining an object property R that is both **transitive** and **symmetric**

  - R is **transitive** just in case: if $<x,y> \in R$ & $<y,z> \in R$ then $<x,z> \in R$
  - R is **symmetric** just in case: if $<x,y> \in R$ then $<y,x> \in R$

Strategy

- You might proceed as follows:

Strategy

- You might proceed as follows:

  1. Suppose there is some <a,b> and <b,c> in R

Strategy

- You might proceed as follows:

1. Suppose there is some <a,b> and <b,c> in R
2. By Transitivity, <a,c> is in R

Strategy

- You might proceed as follows:

1. Suppose there is some <a,b> and <b,c> in R
2. By Transitivity, <a,c> is in R
3. By Symmetry, <b,a>, <c,b>, and <c,a> are in R

Strategy

- You might proceed as follows:

1. Suppose there is some <a,b> and <b,c> in R
2. By Transitivity, <a,c> is in R
3. By Symmetry, <b,a>, <c,b>, and <c,a> are in R
4. By Transitivity, <a,a>, <b,b>, and <c,c> are in R

Strategy

- You might proceed as follows:

1. Suppose there is some <a,b> and <b,c> in R
2. By Transitivity, <a,c> is in R
3. By Symmetry, <b,a>, <c,b>, and <c,a> are in R
4. By Transitivity, <a,a>, <b,b>, and <c,c> are in R

If <a,b> & <b,a> then <a,a>
If <b,a> & <a,b> then <b,b>
If <c,a> & <a,c> then <c,c>

Strategy

- You might proceed as follows:

  1. Suppose there is some <a,b> and <b,c> in R
  2. By Transitivity, <a,c> is in R
  3. By Symmetry, <b,a>, <c,b>, and <c,a> are in R
  4. By Transitivity, <a,a>, <b,b>, and <c,c> are in R

R={<a,b>, <b,c>, <a,c>, <b,a>, <c,b>, <c,a>, <a,a>, <b,b>, <c,c>}

Strategy

- You might proceed as follows:

1. Suppose there is some <a,b> and <b,c> in R
2. By Transitivity, <a,c> is in R
3. By Symmetry, <b,a>, <c,b>, and <c,a> are in R
4. By Transitivity, <a,a>, <b,b>, and <c,c> are in R

R={<a,b>, <b,c>, <a,c>, <b,a>, <c,b>, <c,a>, <a,a>, <b,b>, <c,c>}

Have you explored all the possible cases?

Heuristics

- For n individuals there are "n factorial" non-reflexive combinations

  n factorial = n! = n*(n-1)*(n-2)*…*1

- If you grant reflexive pairs, there are **n!+n** combinations

- Suppose you assert a, b, and c are in the domain

Heuristics

a b c

1

<a,b>	<b,a>	<a,a>
<a,c>	<c,a>	<b,b>
<b,c>	<c,b>	<c,c>

Heuristics

a b c

2

<a,b>	<b,a>	<a,a>
<a,c>	<c,a>	<b,b>
<b,c>	<c,b>	<c,c>

Heuristics

a b c

3

<a,b>	<b,a>	<a,a>
<a,c>	<c,a>	<b,b>
<b,c>	<c,b>	<c,c>

Heuristics

4

a b c

<a,b>	**<b,a>**	**<a,a>**
<a,c>	**<c,a>**	**<b,b>**
<b,c>	**<c,b>**	**<c,c>**

Heuristics

5

a b c

<a,b>	**<b,a>**	**<a,a>**
<a,c>	**<c,a>**	**<b,b>**
<b,c>	**<c,b>**	**<c,c>**

Heuristics

a b c

6

<a,b> <b,a> <a,a>

<a,c> <c,a> <b,b>

<b,c> <c,b> <c,c>

Heuristics

9

a b c

<a,b>	<b,a>	<a,a>
<a,c>	<c,a>	<b,b>
<b,c>	<c,b>	<c,c>

Heuristics

- For n individuals there are "n factorial" non-reflexive combinations

n factorial = n! = n*(n-1)*(n-2)*…*1

- If you grant reflexive pairs, there are **n!+n** combinations

- Suppose you assert a, b, and c are in the domain

n=3

n! = 6

n!+n=9

Heuristics

- So, when you are exploring possible individual pairs under a given object property
- Save yourself time and first identify the number of possible pairs
- Helpful:
  - **So you know when to stop looking for pairs**
  - **As a way to identify pairs to test**

R is Transitive and Symmetric

R={<a,b>, <b,c>, <a,c>, <b,a>, <c,b>, <c,a>, <a,a>, <b,b>, <c,c>}

- Observe that any object property R* is reflexive **just in case** for every x in the domain, <x,x>∈R*

R is Transitive and Symmetric

R={<a,b>, <b,c>, <a,c>, <b,a>, <c,b>, <c,a>, <a,a>, <b,b>, <c,c>}

- Observe that any object property R* is reflexive **just in case** for every x in the domain, <x,x>∈R*

- Domain = {a, b, c}

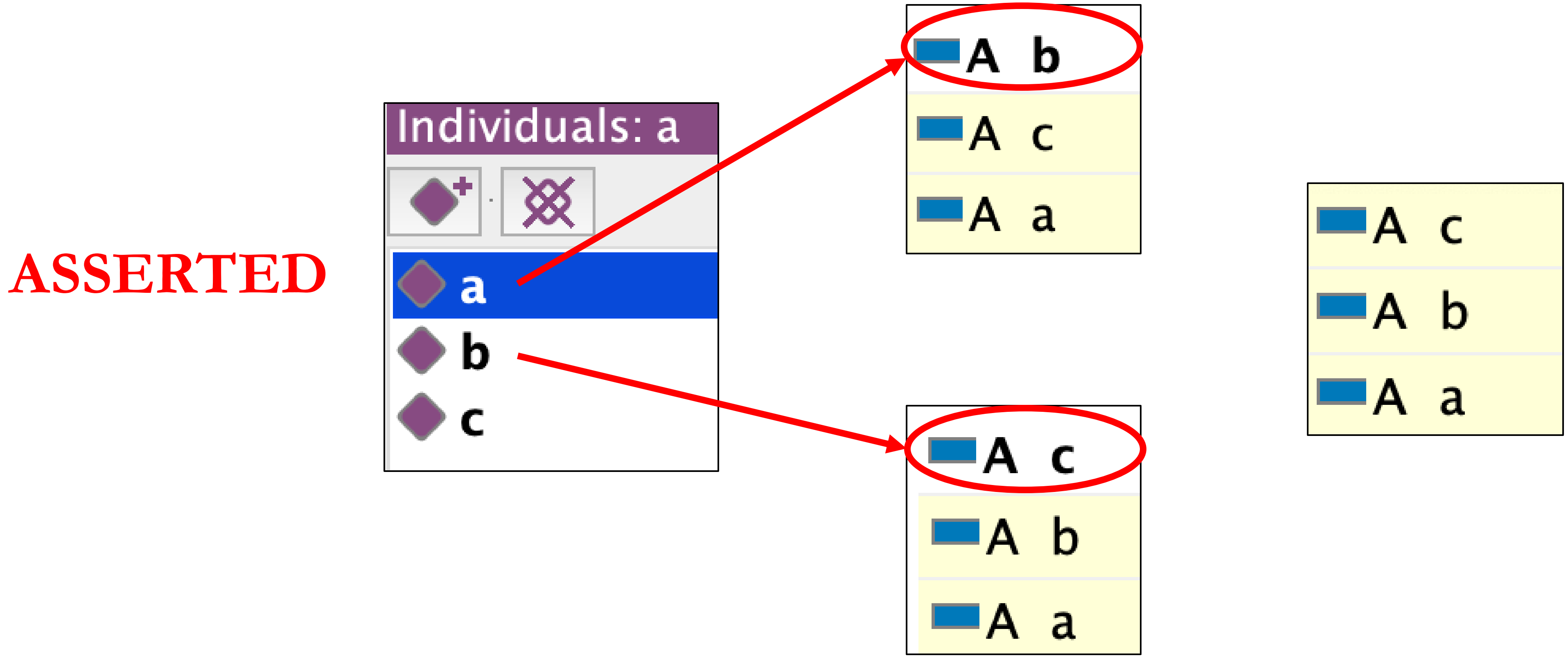
R is Transitive and Symmetric

R={<a,b>, <b,c>, <a,c>, <b,a>, <c,b>, <c,a>, <a,a>, <b,b>, <c,c>}

- Observe that any object property R* is reflexive **just in case** for every x in the domain, <x,x>∈R*
- Domain = {a, b, c}
- R is **reflexive**

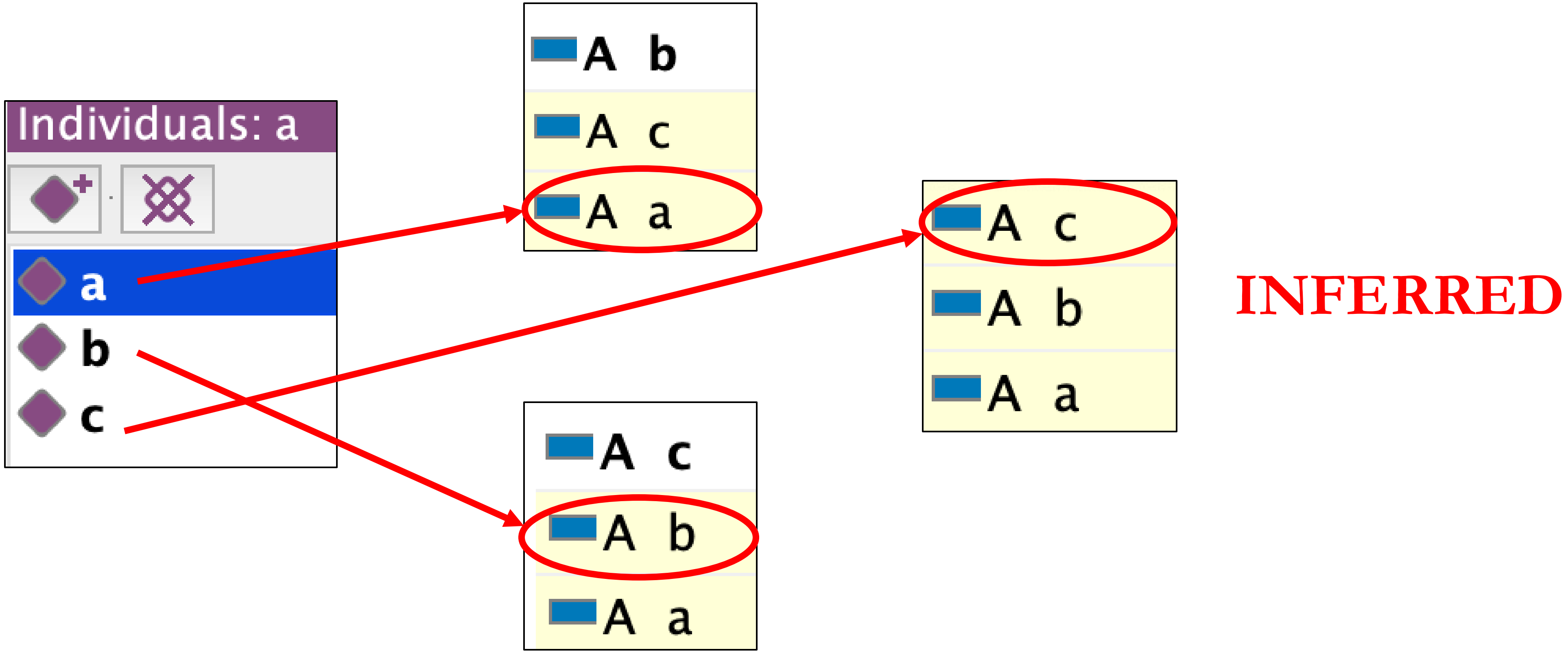
Transitivity & Symmetry Entail Reflexivity

- Which is to say that if A is transitive and symmetric then it's reflexive



Transitivity & Symmetry Entail Reflexivity

- Which is to say that if A is transitive and symmetric then it's reflexive



Symmetry

- Suppose B is a subproperty of A

- If B is symmetric and $<a,b> \in B$ and $<b,c> \in B$, then:

$A = \{<a,b>, <b,c>, <b,a>, <c,b>\}$

$B = \{<a,b>, <b,c>, <b,a>, <c,b>\}$

- Hence, **if B is symmetric then A is symmetric**

Irreflexivity

- Suppose B is a subproperty of A
- B being irreflexive with <a,a>∈A is **consistent**
- A being irreflexive with <a,a>∈B is **inconsistent**

Irreflexivity does not populate up but populates down

Asymmetry & Symmetry

- Suppose B is a subproperty of A
- B being **a**symmetric and A being symmetric is **consistent**
- B being symmetric and A being **a**symmetric is **inconsistent**

Asymmetry does not populate upward

Asymmetry

- Suppose B is a subproperty of A and A is **a**symmetric

1. SUPPOSE: <x,y> in A and <y, x> in B
2. If <y, x> in B then <y, x> in A
3. Hence, <y, x> in A
4. If <x, y> in A then by asymmetry <y, x> is not in A
5. !

Asymmetry populates downward

	Populates Up?	Populates Down?
Symmetry	YES	NO
Asymmetry	NO	YES
Transitivity	YES	NO
Reflexivity	YES	YES
Irreflexivity	NO	YES