

Chapter 2. Declarative Semantics

- **Last updated: June 12 , 2012** -

How do we know what a goal / program means?

→ Translation of Prolog to logical formulas

How do we know what a logical formula means?

→ Models of logical formulas (Declarative semantics)

→ Proofs of logical formulas (Operational semantics)

Question

Question

- What is the meaning of this program?

```
bigger(elephant, horse).
```

```
bigger(horse, donkey).
```

```
is_bigger(X, Y) :- bigger(X, Y).
```

```
is_bigger(X, Y) :- bigger(X, Z), is_bigger(Z, Y).
```

Rephrased question: Two steps

1. How does this program translate to logic formulas?
2. What is the meaning of the logic formulas?

Semantics: Translation

How do we translate a Prolog program to a formula in First Order Logic (FOL)?

→ Translation Scheme

Can any FOL formula be expressed as a Prolog Program?

→ Normalization Steps

Translation of Prolog Programs

1. A Prolog **program** is translated to a **set of formulas**, with each **clause** in the program corresponding to one **formula**:

```
{ bigger( elephant, horse ),  
  bigger( horse, donkey ),  
   $\forall x. \forall y. ( \text{bigger}(x, y) \rightarrow \text{is\_bigger}(x, y) ),$   
   $\forall x. \forall y. ( \exists z. (\text{bigger}(x, z) \wedge \text{is\_bigger}(z, y)) \rightarrow \text{is\_bigger}(x, y) )$   
}
```

2. Such a **set** is to be interpreted as the **conjunction** of all the formulas in the set:

```
bigger( elephant, horse )  $\wedge$   
bigger( horse, donkey )  $\wedge$   
 $\forall x. \forall y. ( \text{bigger}(x, y) \rightarrow \text{is\_bigger}(x, y) )$   $\wedge$   
 $\forall x. \forall y. ( \exists z. (\text{bigger}(x, z) \wedge \text{is\_bigger}(z, y)) \rightarrow \text{is\_bigger}(x, y) )$ 
```

Translation of Clauses

- Each **comma** separating subgoals becomes \wedge (**conjunction**).
- Each **$:-$** becomes \leftarrow (**implication**)
- Each **variable in the head** of a clause is bound by a \forall (**universal quantifier**)

◆ `son(X, Y) :- father(Y, X), male(X).`

◆ $\forall x. \forall y$ `son(x, y) \leftarrow father(y, x) \wedge male(x)`

- Each **variable that occurs only in the body** of a clause is bound by a \exists (**existential quantifier**)

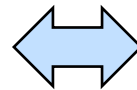
◆ `grandfather(X) :- father(X, Y), parent(Y, Z).`

◆ $\forall x.$ `(grandfather(x) \leftarrow $\exists y. \exists z.$ father(x, y) \wedge parent(y, z))`

Translating Disjunction

- Disjunction is the same as two clauses:

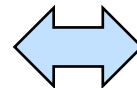
```
disjunction(X) :-  
    ( ( a(X,Y), b(Y,Z) )  
    ; ( c(X,Y), d(Y,Z) )  
    ).
```



```
disjunction(X) :-  
    a(X,Y), b(Y,Z).  
disjunction(X) :-  
    c(X,Y), d(Y,Z) .
```

- Variables with the same name in different clauses are different
- Therefore, variables with the same name in different disjunctive branches are different too!
- Good Style: Avoid accidentally equal names in disjoint branches!
 - ◆ Rename variables in each branch and use explicit unification

```
disjunction(X) :-  
    ( (X=X1, a(X1,Y1), b(Y1,Z1) )  
    ; (X=X2, c(X2,Y2), d(Y2,Z2) )  
    ).
```



```
disjunction(X1) :-  
    a(X1,Y1), b(Y1,Z1).  
disjunction(X2) :-  
    c(X2,Y2), d(Y2,Z2) .
```

Declarative Semantics – in a nutshell

Meaning of Programs (in a nutshell)

Meaning of a program

Meaning of the equivalent formula.

bigger(elephant, horse)
^
bigger(horse, donkey)

^
 $\forall x. \forall y. (\text{bigger}(x, y) \rightarrow \text{is_bigger}(x, y))$

^
 $\forall x. \forall y. (\exists z. (\text{bigger}(x, z) \wedge \text{is_bigger}(z, y)) \rightarrow \text{is_bigger}(x, y))$

Meaning of a formula

Set of logical consequences

bigger(elephant, horse)
^
bigger(horse, donkey)

^
is_bigger(elephant, horse)
^
is_bigger(horse, donkey)

^
is_bigger(elephant, donkey)

Meaning of Programs

Model =
Set of logical consequences =
What is true according to the formula

Meaning of a program

Meaning of the equivalent formula.

bigger(elephant, horse)

^

bigger(horse, donkey)

^

$\forall x. \forall y. (\text{bigger}(x, y) \rightarrow$
 $\text{is_bigger}(x, y))$

^

$\forall x. \forall y. (\exists z. (\text{bigger}(x, z) \wedge$
 $\text{is_bigger}(z, y)) \rightarrow$
 $\text{is_bigger}(x, y))$

Meaning of a formula

Set of logical consequences

bigger(elephant, horse)

^

bigger(horse, donkey)

^

is_bigger(elephant, horse)

^

is_bigger(horse, donkey)

^

is_bigger(elephant, donkey)

Semantics of Programs and Queries (in a nutshell)

Program

```

bigger(elephant, horse) .
bigger(horse, donkey) .

is_bigger(X, Y) :-
    bigger(X, Y) .

is_bigger(X, Y) :-
    bigger(X, Z) ,
    is_bigger(Z, Y) .
    
```

Formula

```

bigger( elephant, horse )
^
bigger( horse, donkey )
^
 $\forall x. \forall y. ( \text{is\_bigger}(x, y) \leftarrow$ 
    bigger(x, y) )
^
 $\forall x. \forall y. ( \exists z. ( \text{is\_bigger}(x, y) \leftarrow$ 
    bigger(x, z) ^
    is_bigger(z, y)))
    
```

Model

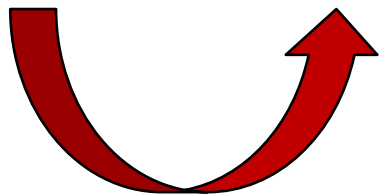
```

bigger( elephant, horse )
^
bigger( horse, donkey )
^
is_bigger(elephant, horse)
^
is_bigger(horse, donkey)
^
is_bigger(elephant, donkey)
    
```

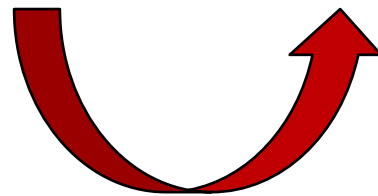
Query

```

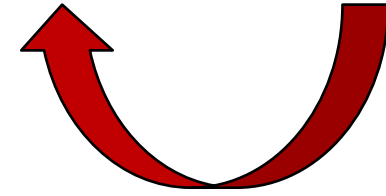
?-
bigger( elephant, X )
^
is_bigger(X, donkey)
    
```



Translation



Interpretation
(logical consequence)

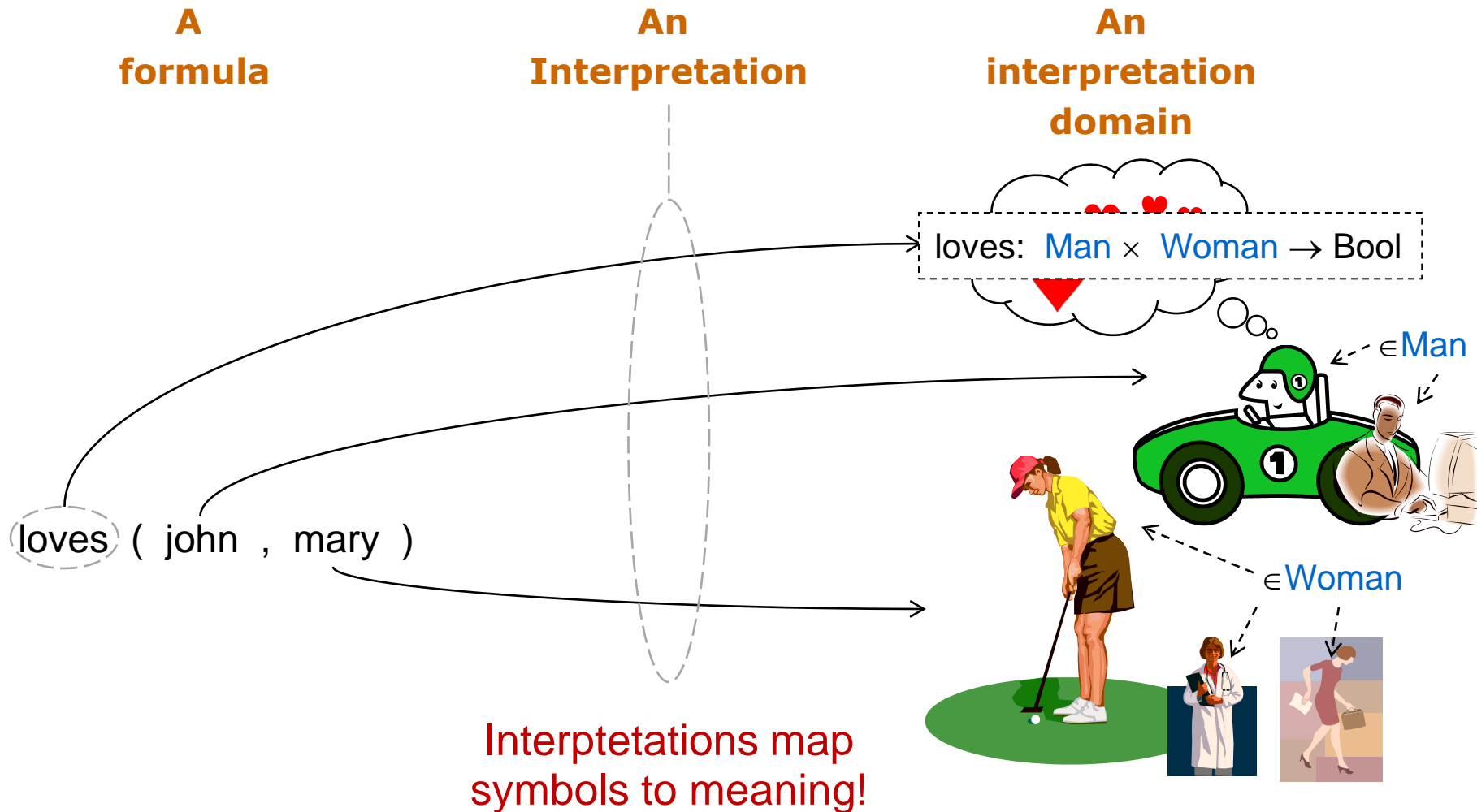


Matching

Declarative Semantics – the details

- Interpretations of formulas
 - Herbrand Interpretations
 - Herbrand Model
 - Logical Consequence

Interpretations of Formulas



Interpretations of Formulas

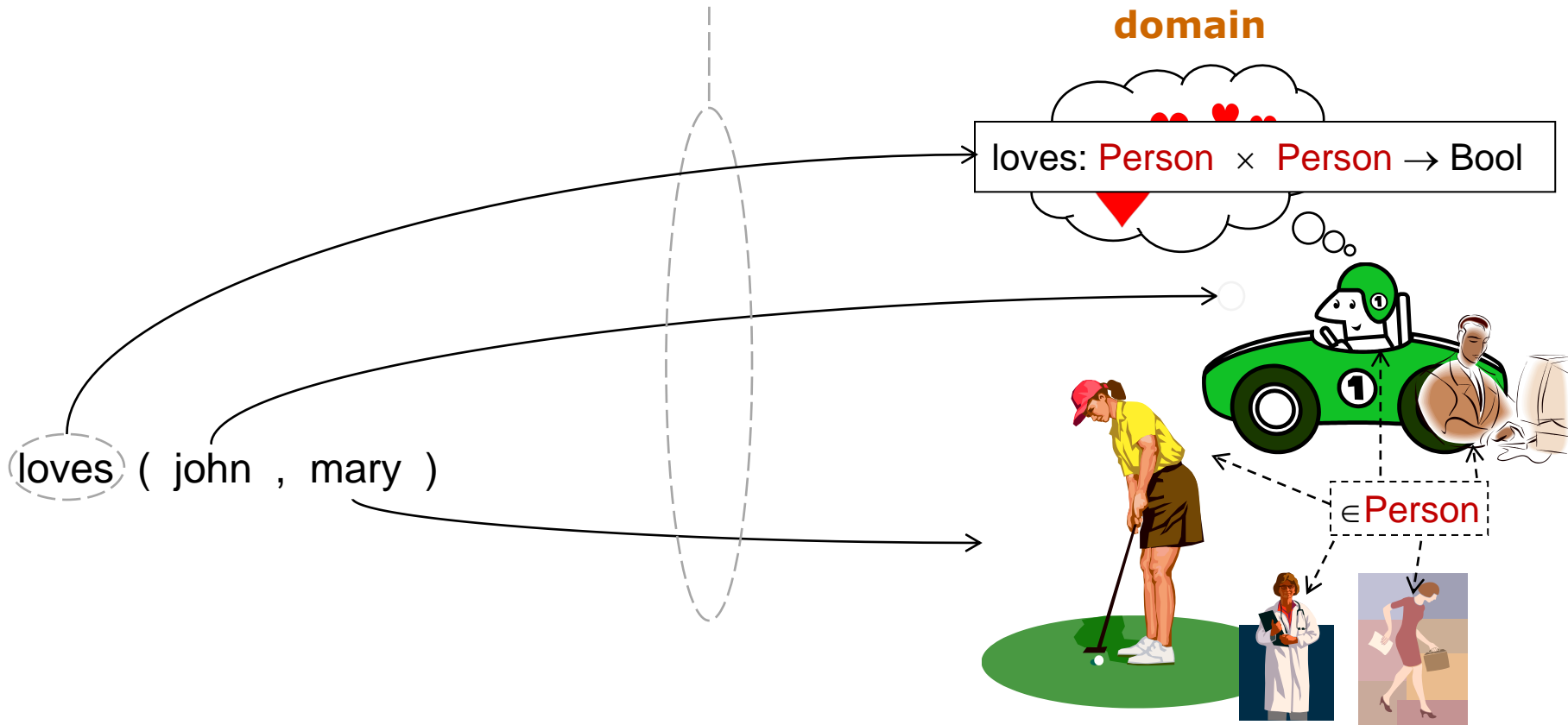
Same
formula

Slightly different
Interpretation

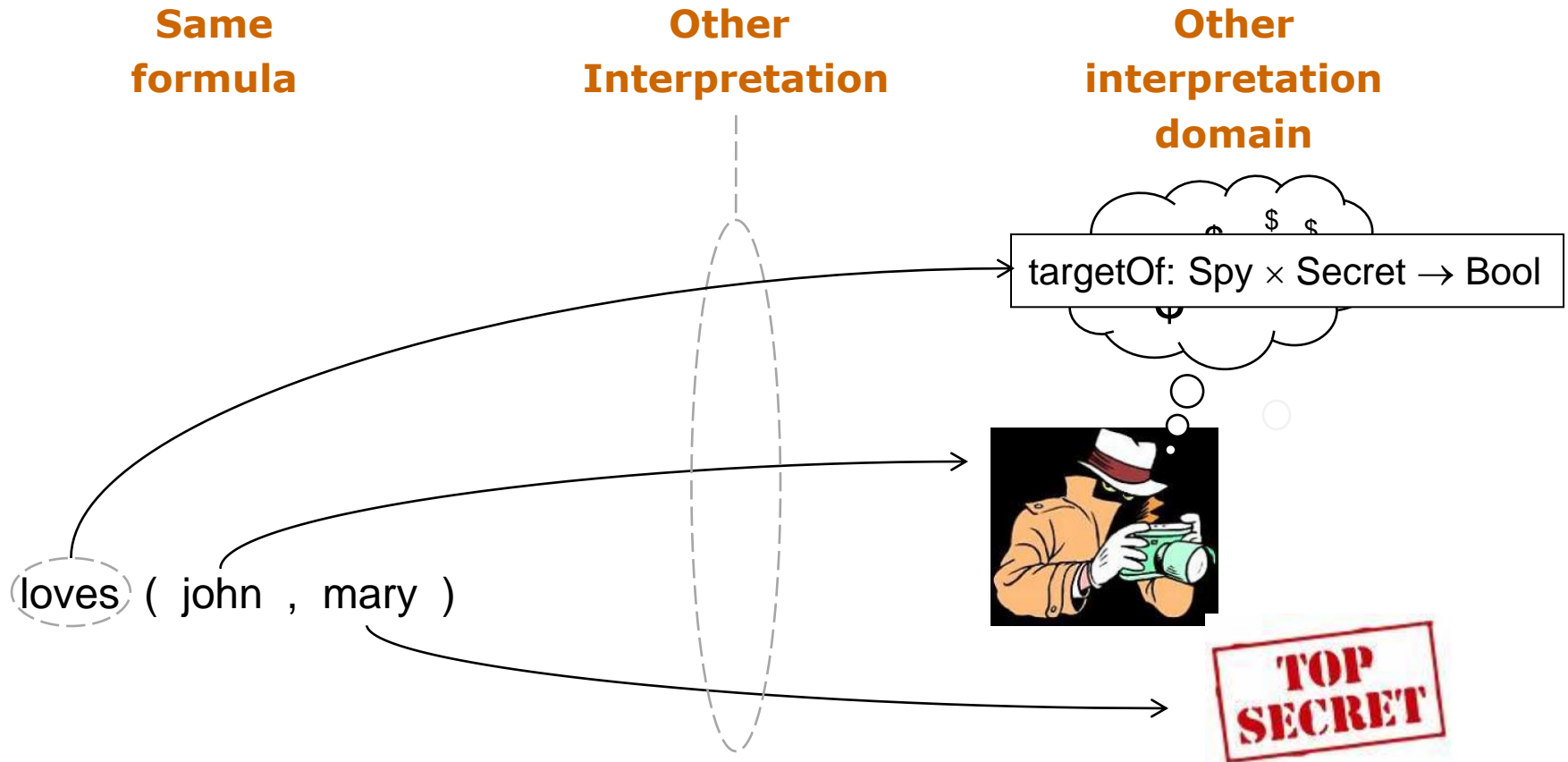
Slightly different
interpretation
domain

loves (john , mary)

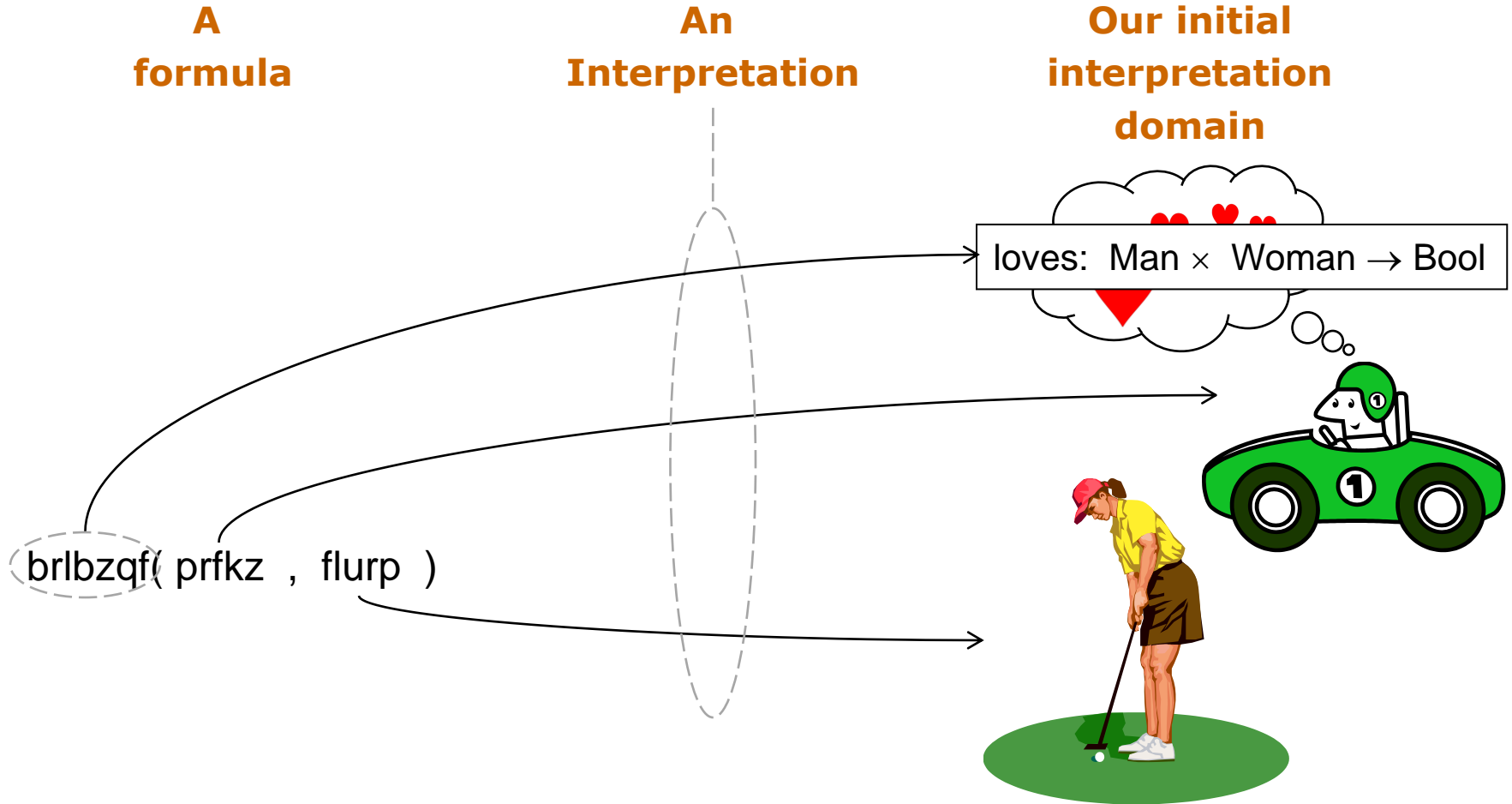
loves: Person \times Person \rightarrow Bool



Interpretations of Formulas




Interpretations of Formulas



What does this tell us?

Observations

- Formulas only have a meaning with respect to an interpretation
- An interpretation maps formulas to elements of an interpretation domain
 - ◆ **constants** to constant in the domain
 - ⇒ “john” to 
 - ◆ **function symbols** to functions on the domain
 - ⇒ function symbols do not occur in our example
 - ◆ **predicate symbols** to predicates on the domain
 - ⇒ “loves/2” to “**targetOf: Spy × Secret → Bool**”
 - ◆ **formulas** to truth values
 - ⇒ “loves(john,mary)” to “**true**”

What does this tell us?

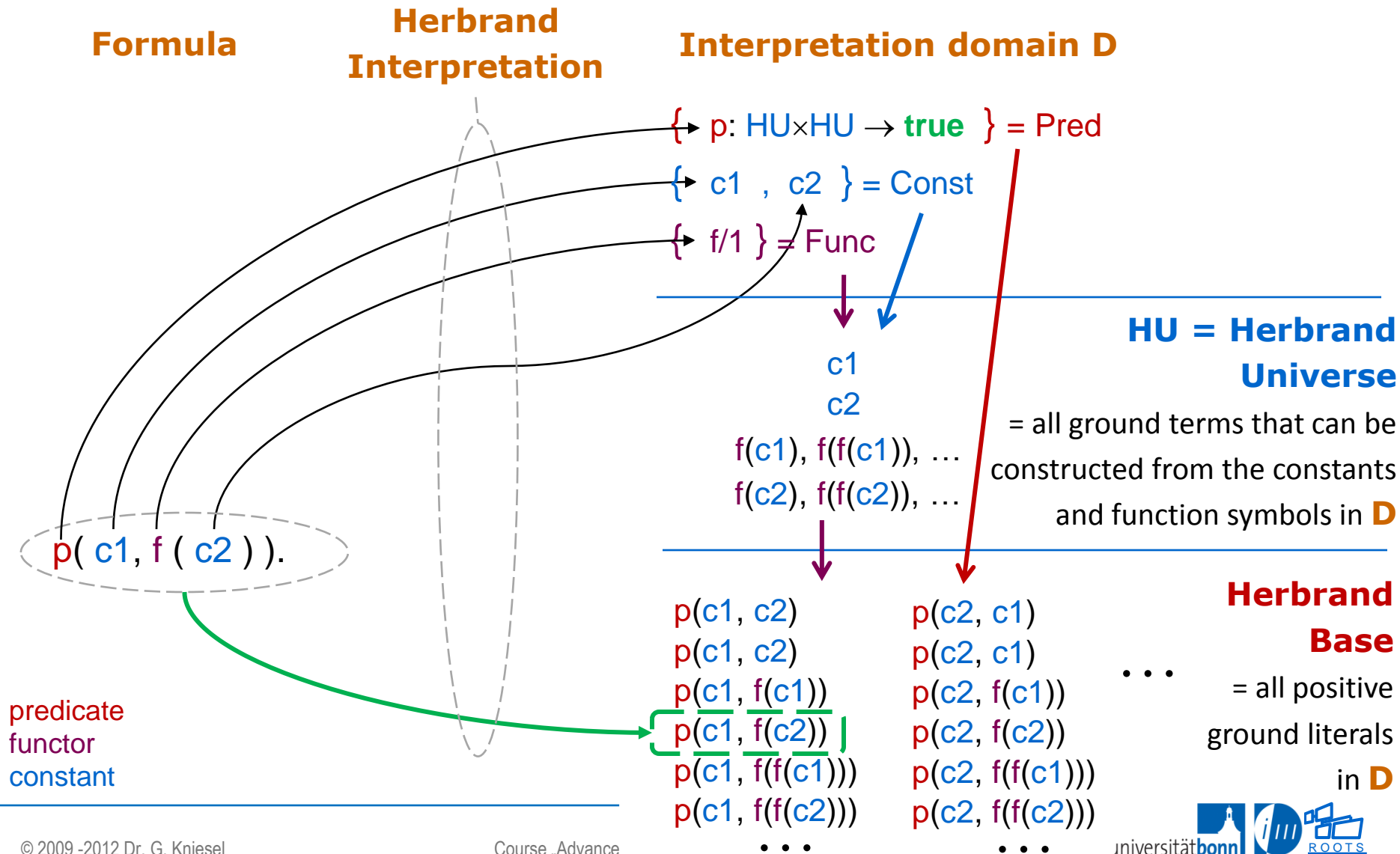
Dilemma

- Too many possible interpretations!
- Which interpretation to use for proving truth?

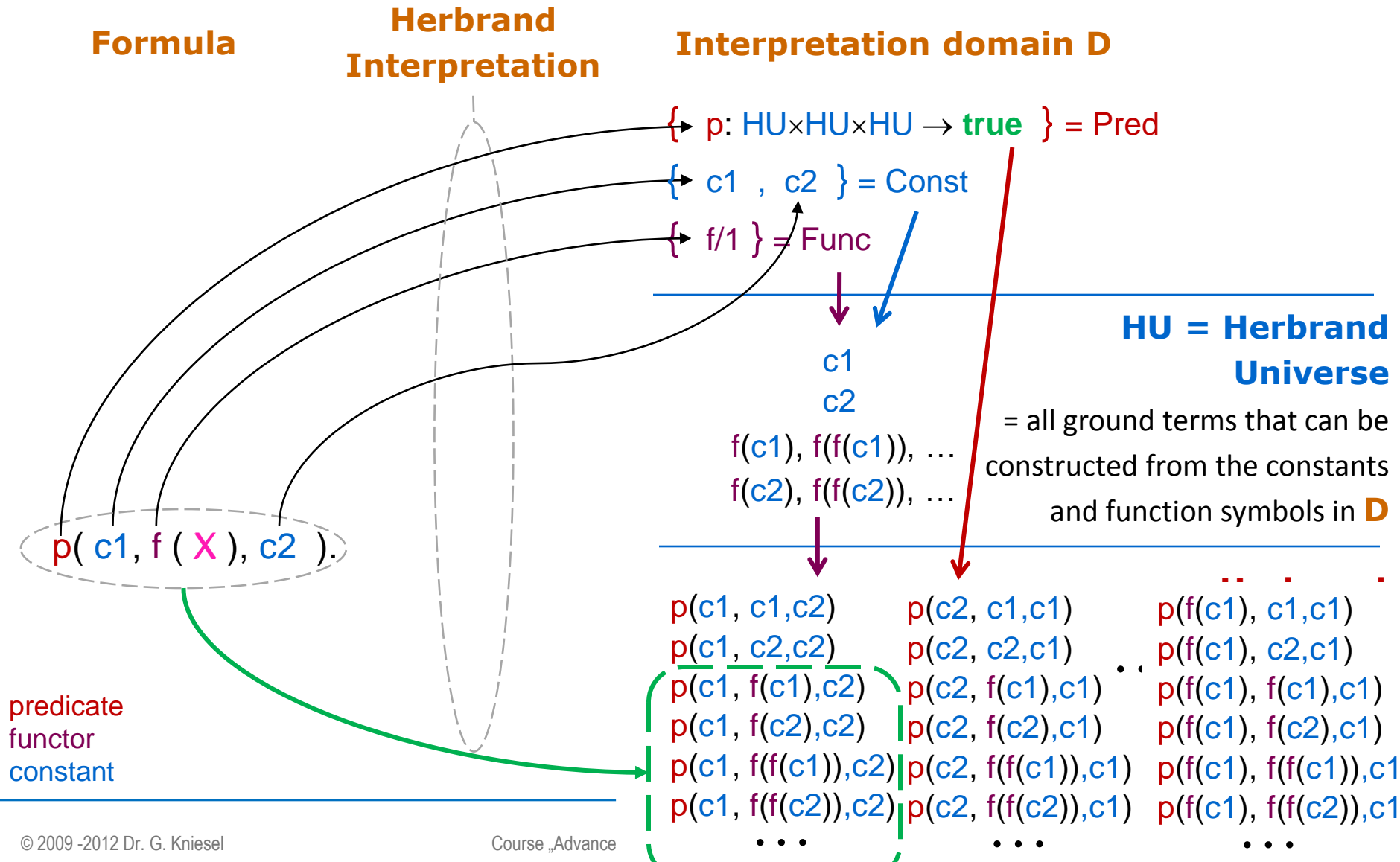
Solution

- For universally quantified formulas there is a “standard” interpretation, the “Herbrand interpretation”, which has two nice properties:
 - ◆ If any interpretation satisfies a given set of clauses S then there is a Herbrand interpretation that satisfies them
 - ⇒ It suffices to check satisfiability for the Herbrand interpretation!
 - ◆ If S is unsatisfiable then there is a finite unsatisfiable set of ground instances from the Herbrand base defined by S .
 - ⇒ Unsatisfiability can be checked finitely

Herbrand Interpretations



Herbrand Interpretations of Formulas with Variables



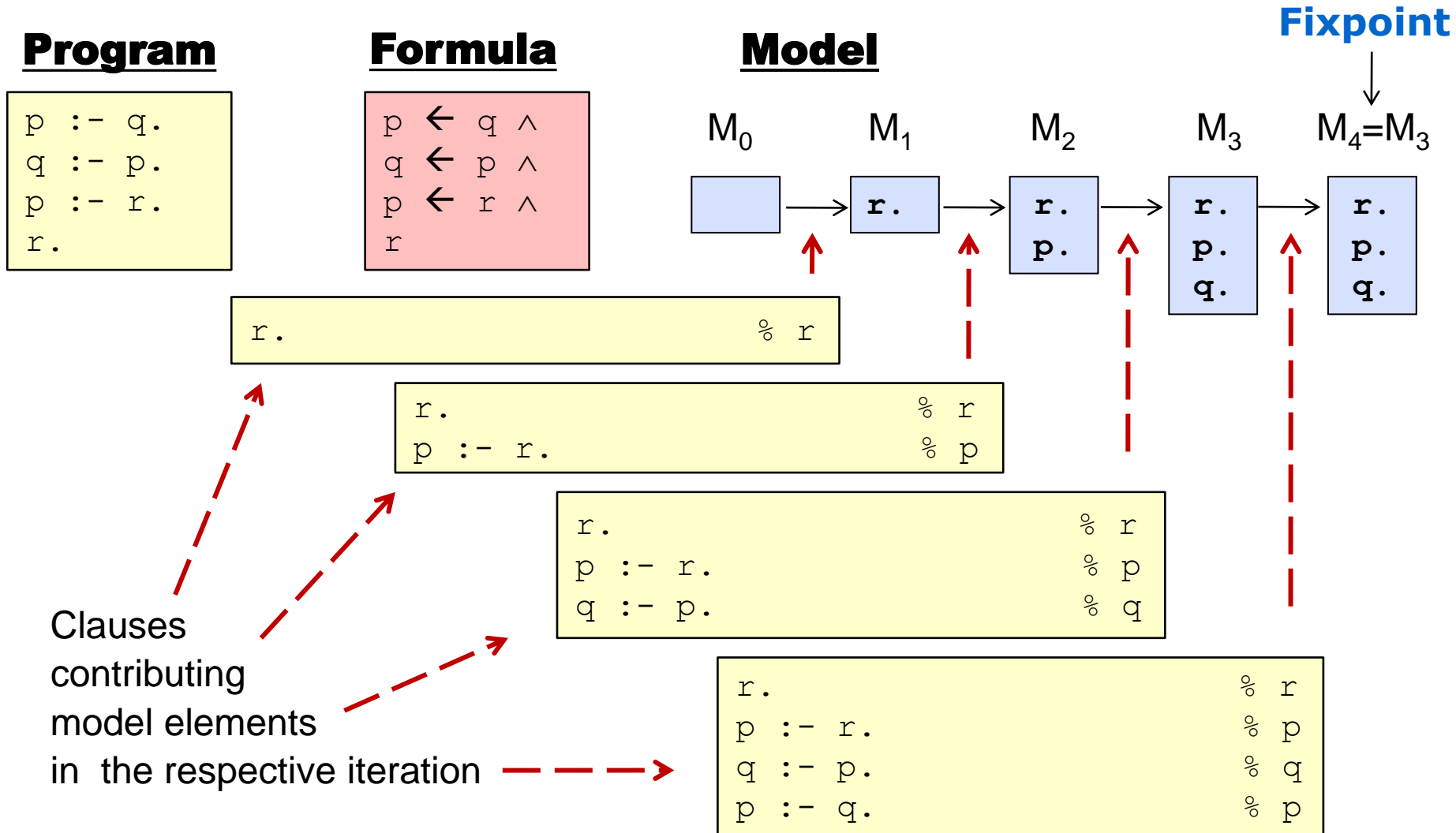
Herbrand Models (1)

- The Interpretation Domain (D) of a program P consists of three sets:
 - ◆ $Const$ contains all constants occurring in P
 - ◆ $Func$ contains all function symbols occurring in P
 - ◆ $Pred$ contains a predicate $p: \underbrace{HU \times \dots \times HU}_{\text{arity } n} \rightarrow true$
for each predicate symbol p of arity n occurring in the program P
- The Herbrand Universe (HU) of a program P is the set of all ground terms that can be constructed from the function symbols and constants in P
- The Herbrand Base of a program P is the set of all / positive ground literals that can be constructed by applying the predicate symbols in P to arguments from the Herbrand Universe of P = not negated

Herbrand Models (2)

- A **Herbrand Interpretation** maps each formula in P to the elements of the Herbrand Base that are its **logical consequences**
 - ◆ Each ground fact is mapped to true.
 - ◆ Each possible ground instantiation of a non-ground fact is mapped to true.
 - ◆ Each instantiation of the head literal of a rule that is a logical consequence of the rule body is mapped to true
- The **Herbrand Model** of a program P is the **subset of the Herbrand Base of P that is true** according to the Herbrand Interpretation.
 - ◆ It is the **set of all logical consequences** of the program
- The **Herbrand Model** can be **constructed by fixpoint iteration**:
 - ◆ Initialize the model with the ground instantiations of facts in P
 - ◆ Add all new facts that follow from the intermediate model and P
 - ◆ ... until the model does not change anymore (= fixpoint is reached)

Constructing Models by Fixpoint Iteration



Model-based Semantics → Algorithm

Model-based semantics

- Herbrand interpretations and Herbrand models
- Basic step = “Entailment” (Logical consequence)
- A formula is true if it is a logical consequence of the program

Algorithm = Logic + Control

- Logic = Clauses
- Control =
 - ◆ Bottom-up fixpoint iteration to build the model
 - ◆ Matching of queries to the model

Program

```
bigger(elephant, horse) .  
bigger(horse, donkey) .  
...
```

Formula

```
bigger( elephant, horse )  
^  
bigger( horse, donkey )  
^  
...
```

Model

```
bigger( elephant, horse )  
^  
bigger( horse, donkey )  
^  
...
```

Query

```
?-  
bigger( elephant, X )  
^  
is_bigger(X, donkey)
```



Translation



Interpretation
(logical consequence)



Matching

Examples

- All Prolog programs shown so far are declarative

Homework / Exam Preparation

- Build the Herbrand Universe, Basis and Model of
 - ◆ is_grandfather/2 from page 13
 - ◆ binary_tree/2 from page 26
- How do the two predicates differ with respect to their universe, basis and model?

Declarative Semantics Assessed

Pro

- Simple
 - ◆ Easy to understand
- Thorough formal foundation
 - ◆ implication (entailment)

Perfect for understanding the meaning of a program

Contra

- Inefficient
 - ◆ Need to build the whole model in the worst case
- Inapplicable to infinite models
 - ◆ Never terminates if the query is not true in the model

Bad as the basis of a practical interpreter implementation

Chapter Summary

- Translation to logic
 - ◆ From clauses to formulas
- Declarative / Model-based Semantics
 - ◆ Herbrand Universe
 - ◆ Herbrand Interpretation
 - ◆ Herbrand Model
- Operational interpretation
 - ◆ Model construction by fix-point iteration
- Assessment
 - ◆ Strength
 - ◆ Weaknesses