First-Order Logic
Semantics

Reading: Chapter 8, 9.1-9.2, ~~9.5.1-9.5.5~~

FOL Syntax and Semantics read: 8.1-8.2

FOL Knowledge Engineering read: 8.3-8.5

FOL Inference read: Chapter 9.1-9.2, ~~9.5.1-9.5.5~~

(Please read lecture topic material before and after each lecture on that topic)

Outline

- Propositional Logic is **Useful** --- but has **Limited Expressive Power**
- First Order Predicate Calculus (FOPC), or First Order Logic (FOL).
  - FOPC has greatly expanded expressive power, though still limited.
- New Ontology
  - The world consists of OBJECTS (for propositional logic, the world was facts).
  - OBJECTS have PROPERTIES and engage in RELATIONS and FUNCTIONS.
- New Syntax
  - Constants, Predicates, Functions, Properties, Quantifiers.
- New Semantics
  - Meaning of new syntax.
- Knowledge engineering in FOL
- ~~Inference in FOL~~

You will be expected to know

- FOPC syntax and semantics
  - Syntax: Sentences, predicate symbols, function symbols, constant symbols, variables, quantifiers
  - Semantics: Models, interpretations
- De Morgan's rules for quantifiers
  - connections between $\forall$ and $\exists$
- Nested quantifiers
  - Difference between "$\forall$ x $\exists$ y P(x, y)" and "$\exists$ x $\forall$ y P(x, y)"
  - $\forall$ x $\exists$ y Likes(x, y)
  - $\exists$ x $\forall$ y Likes(x, y)
- Translate simple English sentences to FOPC and back
  - $\forall$ x $\exists$ y Likes(x, y) $\Leftrightarrow$ "Everyone has someone that they like."
  - $\exists$ x $\forall$ y Likes(x, y) $\Leftrightarrow$ "There is someone who likes every person."
- Unification: Given two FOL terms containing variables
  - Find the most general unifier if one exists.
  - Else, explain why no unification is possible.
  - See figure 9.1 and surrounding text in your textbook.

Outline

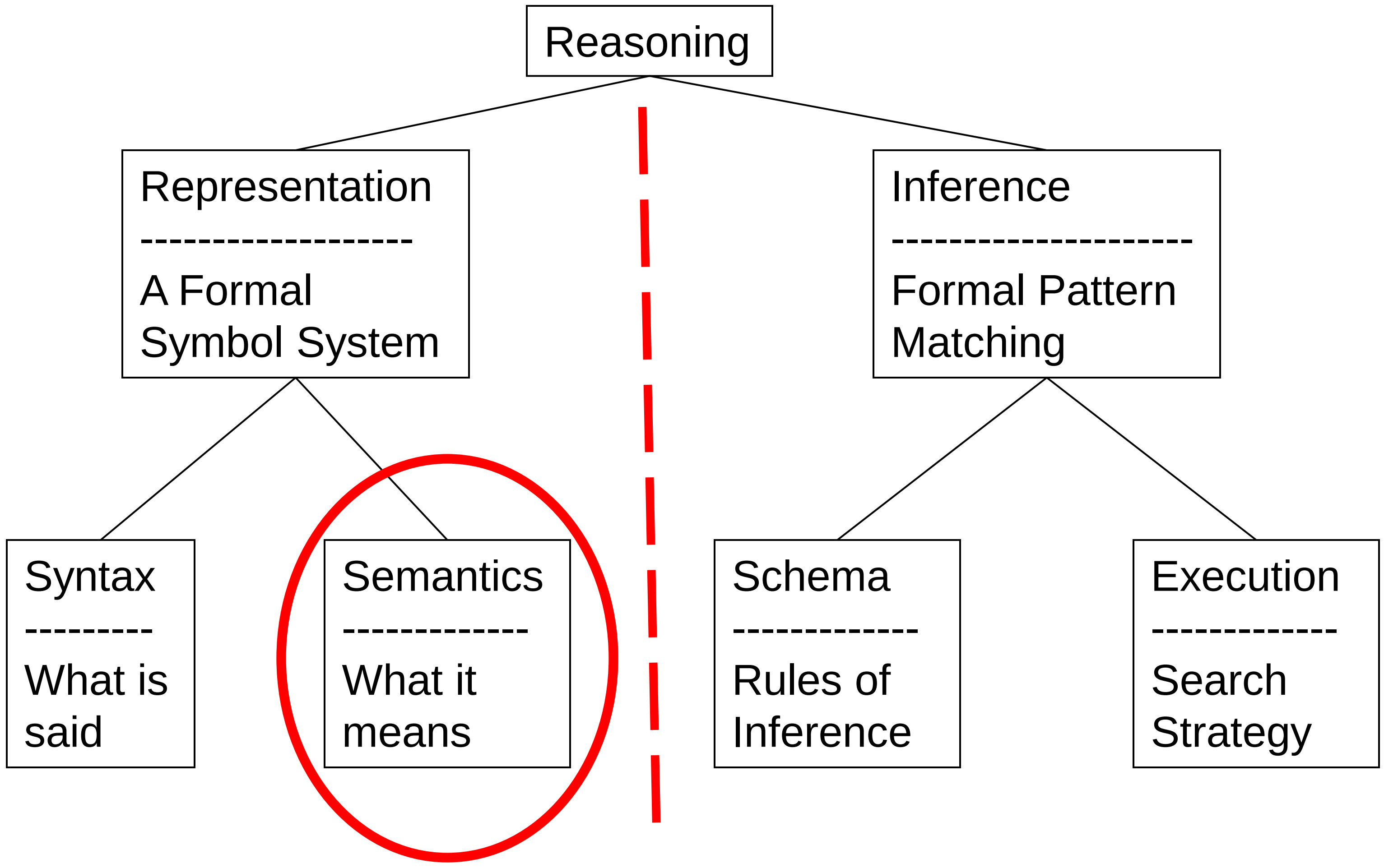
- Review: KB |= S is equivalent to |= (KB ⇒ S)
  - So what does { } |= S mean?
- Review: Follows, Entails, Derives
  - Follows: "Is it the case?"
  - Entails: "Is it true?"
  - Derives: "Is it provable?"
- Semantics of FOL (FOPC)
  - Model, Interpretation
- Unification

FOL (or FOPC) Ontology:

What kind of things exist in the world?

What do we need to describe and reason about?

Objects --- with their relations, functions, predicates, properties, and general rules.



Review: KB |= S means |= (KB ⇒ S)

- KB |= S is read “KB entails S.”
  - Means “S is true in every world (model) in which KB is true.”
  - Means “In the world, S follows from KB.”

- KB |= S is equivalent to |= (KB ⇒ S)
  - Means “(KB ⇒ S) is true in every world (i.e., is valid).”

- And so: { } |= S is equivalent to |= ({ } ⇒ S)

- So what does ({ } ⇒ S) mean?
  - Means “True implies S.”
  - Means “S is valid.”
  - In Horn form, means “S is a fact.” p. 256 (3^{rd} ed.; p. 281, 2^{nd} ed.)

- **Why does { } mean True here,**
  but means False in resolution proofs?

Review: (True $\Rightarrow$ S) means "S is a fact."

- By convention,
  - The null conjunct is "syntactic sugar" for True.
  - The null disjunct is "syntactic sugar" for False.
  - Each is assigned the truth value of its identity element.
    - For conjuncts, True is the identity: (A $\wedge$ True) $\equiv$ A
    - For disjuncts, False is the identity: (A $\vee$ False) $\equiv$ A

- A KB is the conjunction of all of its sentences.
  - So in the expression: { } |= S
    - We see that { } is the null conjunct and means True.
  - The expression means "S is true in every world where True is true."
    - I.e., "S is valid."
  - Better way to think of it: { } does not exclude any worlds (models).

- In Conjunctive Normal Form each clause is a disjunct.
  - So in, say, KB = { (P Q) ($\neg$Q R) () (X Y $\neg$Z) }
    - We see that () is the null disjunct and means False.

Side Trip: Functions AND, OR, and null values (Note: These are "syntactic sugar" in logic.)

function AND(*arglist*) **returns** a truth-value
  return ANDOR(*arglist,* True)

function OR(*arglist*) **returns** a truth-value
  return ANDOR(*arglist,* False)

function ANDOR(*arglist, nullvalue*) **returns** a truth-value
  /* *nullvalue* is the identity element for the caller. */
  if (*arglist* = {})
    then return *nullvalue*
  if (FIRST(*arglist*) = NOT(*nullvalue*))
    then return NOT(*nullvalue*)
  return ANDOR(REST(*arglist*), *nullvalue*)

Side Trip: We only need one logical connective.
(Note: AND, OR, NOT are "syntactic sugar" in logic.)

Both NAND and NOR are logically complete.

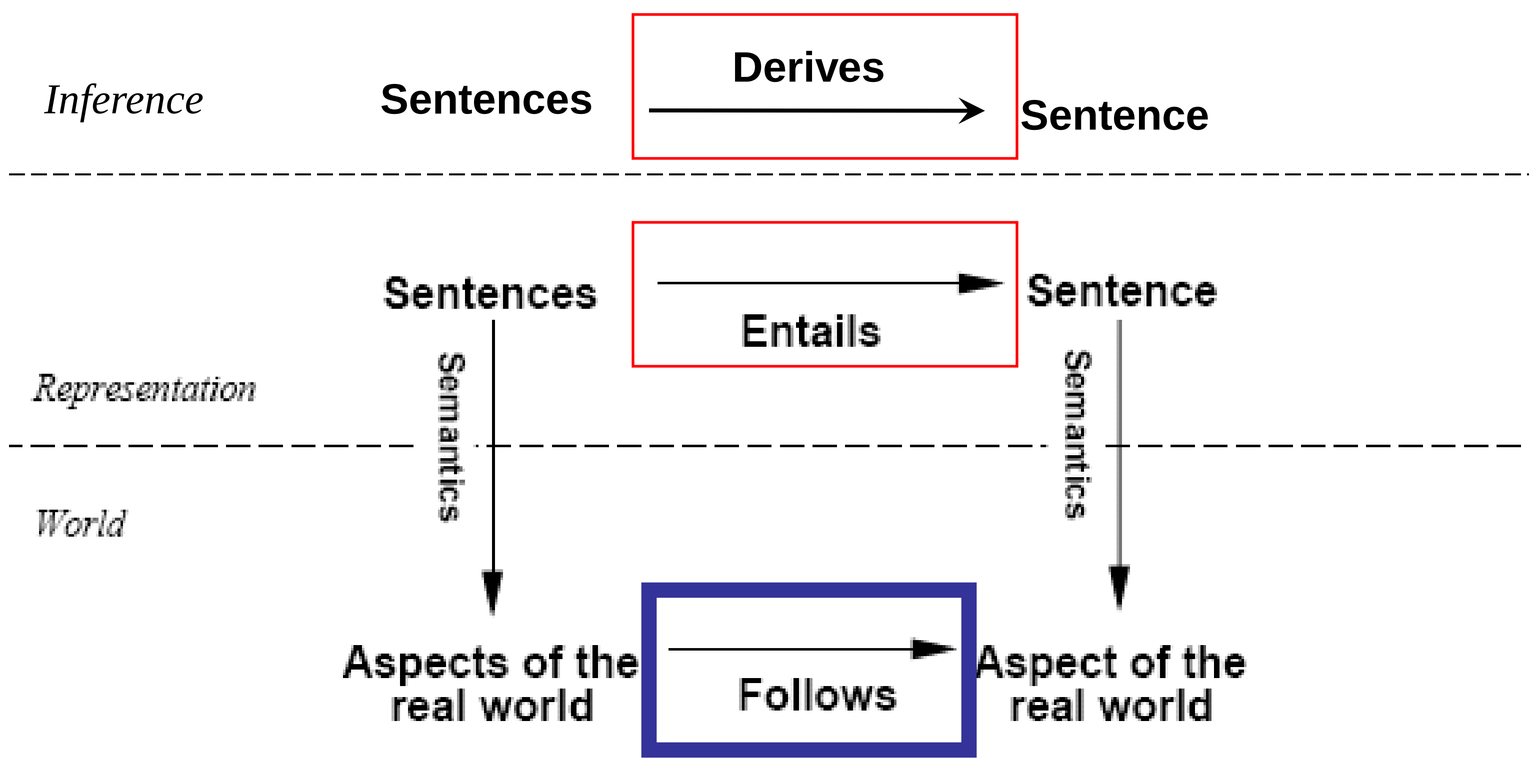
- **NAND is also called the "Sheffer stroke"**
- **NOR is also called "Pierce's arrow"**

(NOT A) = (NAND A TRUE) = (NOR A FALSE)

(AND A B) = (NAND TRUE (NAND A B))
= (NOR (NOR A FALSE) (NOR B FALSE))

(OR A B) = (NAND (NAND A TRUE) (NAND B TRUE))
=(NOR FALSE (NOR A B))

Review: Schematic for Follows, Entails, and Derives



If KB is true in the real world,

then any sentence α ***entailed*** *by KB*

and any sentence α ***derived*** *from KB*

by a sound inference procedure

is also true in the real world.

Schematic Example: Follows, Entails, and Derives

Inference

"Mary is Sue's sister and Amy is Sue's daughter."

"An aunt is a sister of a parent."

Derives →

Is it provable?

"Mary is Amy's aunt."

Representation

"Mary is Sue's sister and Amy is Sue's daughter."

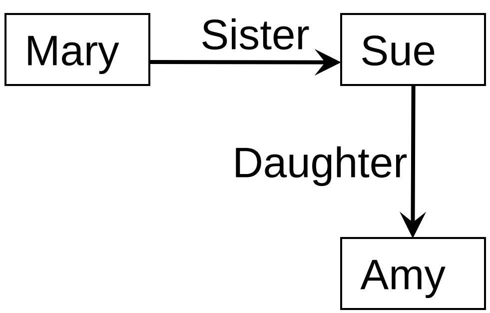
"An aunt is a sister of a parent."

Entails →

Is it true?

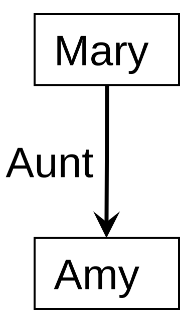
"Mary is Amy's aunt."

World



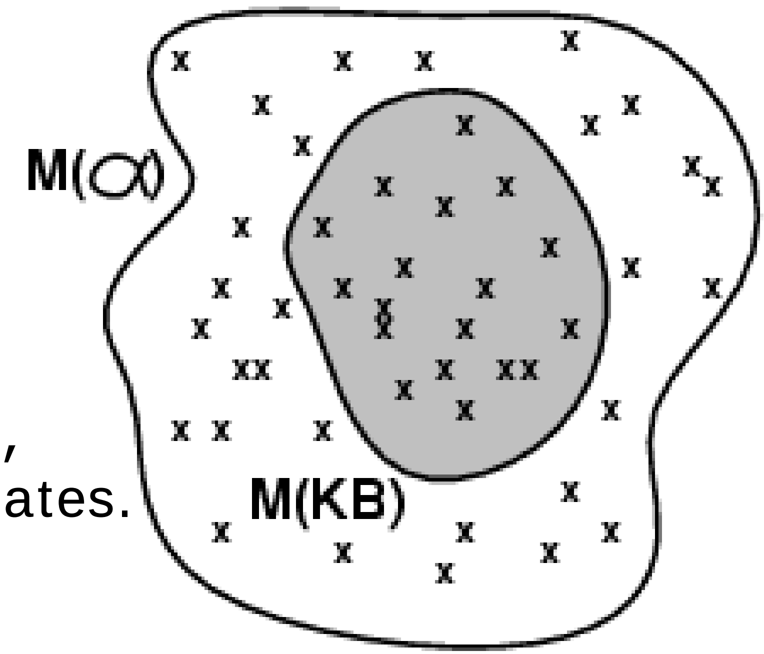
Follows →

Is it the case?



Review: Models (and in FOL, Interpretations)

- Models are formal worlds in which truth can be evaluated
- We say *m* is a model of a sentence α if α is true in *m*
- *M(α)* is the set of all models of α
- Then KB $\models$ α iff $M(KB) \subseteq M(\alpha)$
  - E.g. *KB*, = "Mary is Sue's sister and Amy is Sue's daughter."
  - α = "Mary is Amy's aunt."



- Think of KB and α as constraints, and of models m as possible states.
- M(KB) are the solutions to KB and M(α) the solutions to α.
- Then, KB $\models$ α, i.e., $\models$ (KB $\Rightarrow$ a) , when all solutions to KB are also solutions to α.

Semantics: Worlds

- **The** world **consists of** objects **that have** properties**.**
  - **There are** relations **and** functions **between these objects**
  - **Objects in the world, individuals:** people, houses, numbers, colors, baseball games, wars, centuries
    - Clock A, John, 7, the-house in the corner, Tel-Aviv, Ball43
  - **Functions** on individuals:
    - father-of, best friend, third inning of, one more than
  - **Relations**:
    - brother-of, bigger than, inside, part-of, has color, occurred after
  - **Properties (a relation of arity 1)**:
    - red, round, bogus, prime, multistoried, beautiful

Semantics: Interpretation

- An **interpretation** of a sentence (wff) is an assignment that maps
  - Object constant symbols to objects in the world,
  - n-ary function symbols to n-ary functions in the world,
  - n-ary relation symbols to n-ary relations in the world
- Given an interpretation, an atomic sentence has the value "true" if it denotes a relation that holds for those individuals denoted in the terms. Otherwise it has the value "false."
  - Example: Kinship world:
    - Symbols = Ann, Bill, Sue, Married, Parent, Child, Sibling, …
  - World consists of individuals in relations:
    - Married(Ann,Bill) is false, Parent(Bill,Sue) is true, …

Truth in first-order logic

- Sentences are true with respect to a model and an interpretation
- Model contains objects (domain elements) and relations among them
- Interpretation specifies referents for

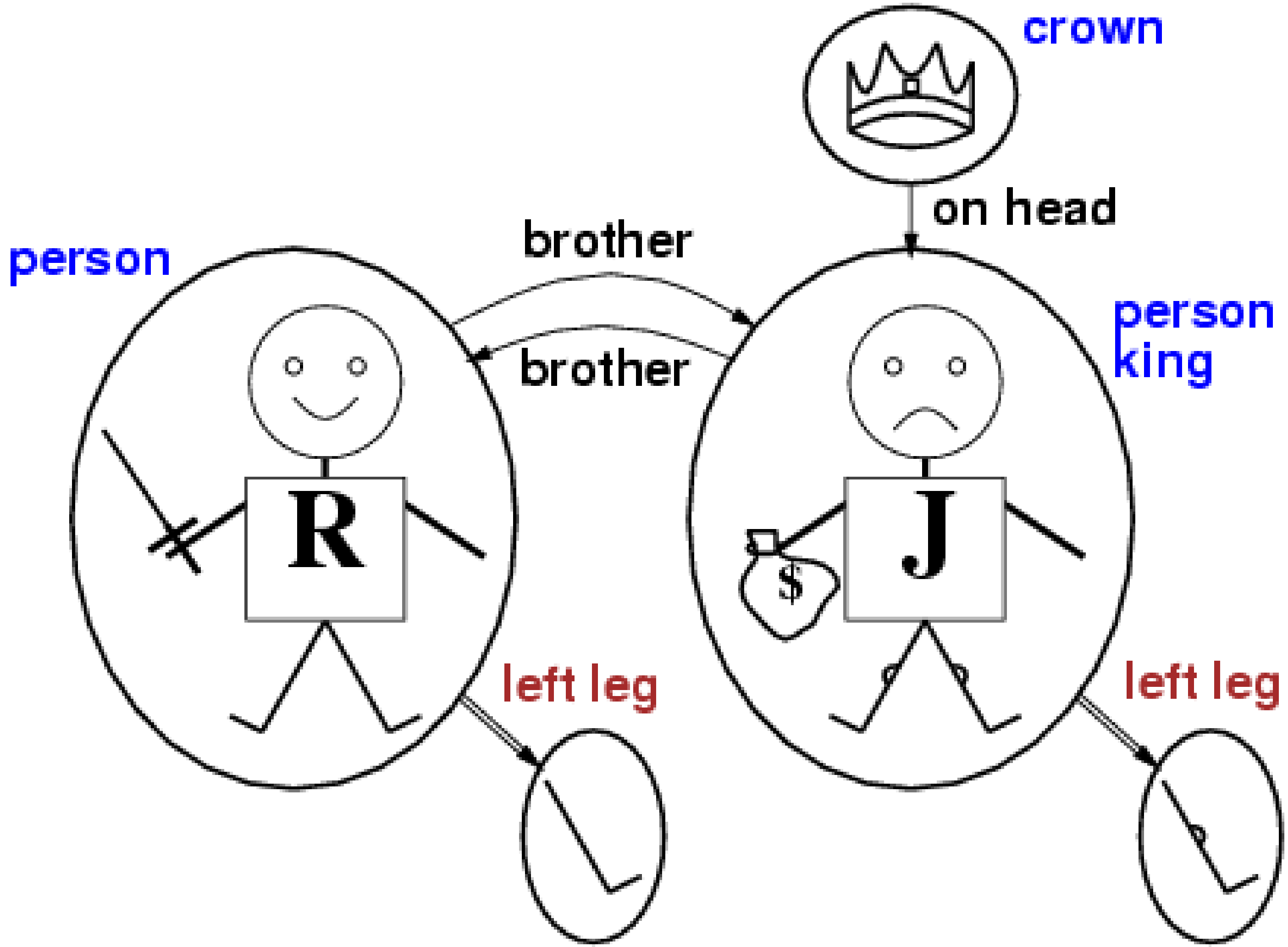
constant symbols	→	objects
predicate symbols	→	relations
function symbols	→	functional relations

- An atomic sentence *predicate*$(term_1, \ldots, term_n)$ is true
  iff the objects referred to by $term_1, \ldots, term_n$
  are in the relation referred to by *predicate*

Semantics: Models

- **An interpretation** satisfies **a wff (sentence) if the wff has the value "true" under the interpretation.**
- **Model: A domain and an interpretation that satisfies a wff is a** model **of that wff**
- **Validity: Any wff that has the value "true" under all interpretations is** valid
- **Any wff that does not have a model is inconsistent or unsatisfiable**
- **If a wff w has a value true under all the models of a set of sentences KB then KB logically entails w**

Models for FOL: Example



Unification

- Recall: Subst(θ, p) = result of substituting θ into sentence p

- Unify algorithm: takes 2 sentences p and q and returns a unifier if one exists

  Unify(p,q) = θ   where Subst(θ, p) = Subst(θ, q)

- Example:
  p = Knows(John,x)
  q = Knows(John, Jane)

  Unify(p,q) = {x/Jane}

Unification examples

- simple example: query = Knows(John,x), i.e., who does John know?

p	q	θ
Knows(John,x)	Knows(John,Jane)	{x/Jane}
Knows(John,x)	Knows(y,OJ)	{x/OJ,y/John}
Knows(John,x)	Knows(y,Mother(y))	{y/John,x/Mother(John)}
Knows(John,x)	Knows(x,OJ)	{fail}

- Last unification fails: only because x can't take values John and OJ at the same time
  - But we know that if John knows x, and everyone (x) knows OJ, we should be able to infer that John knows OJ

- Problem is due to use of same variable x in both sentences

- Simple solution: Standardizing apart eliminates overlap of variables, e.g., Knows(z,OJ)

Unification

- To unify *Knows(John,x)* and *Knows(y,z)*,

  θ = {y/John, x/z } or θ = {y/John, x/John, z/John}

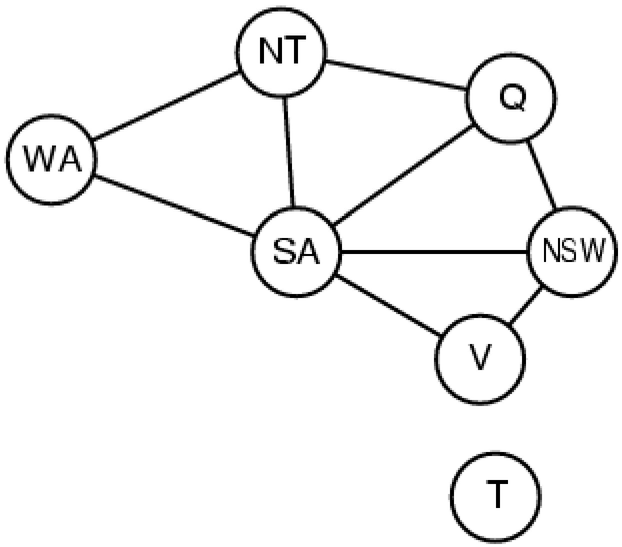
- The first unifier is more general than the second.

- There is a single most general unifier (MGU) that is unique up to renaming of variables.

  MGU = { y/John, x/z }

- General algorithm in Figure 9.1 in the text

Hard matching example



Diff(wa,nt) $\wedge$ *Diff(wa,sa)* $\wedge$ *Diff(nt,q)* $\wedge$
Diff(nt,sa) $\wedge$ *Diff(q,nsw)* $\wedge$ *Diff(q,sa)* $\wedge$
Diff(nsw,v) $\wedge$ *Diff(nsw,sa)* $\wedge$ *Diff(v,sa)* $\Rightarrow$
Colorable()

Diff(Red,Blue) *Diff (Red,Green)*
Diff(Green,Red) *Diff(Green,Blue)*
Diff(Blue,Red) *Diff(Blue,Green)*

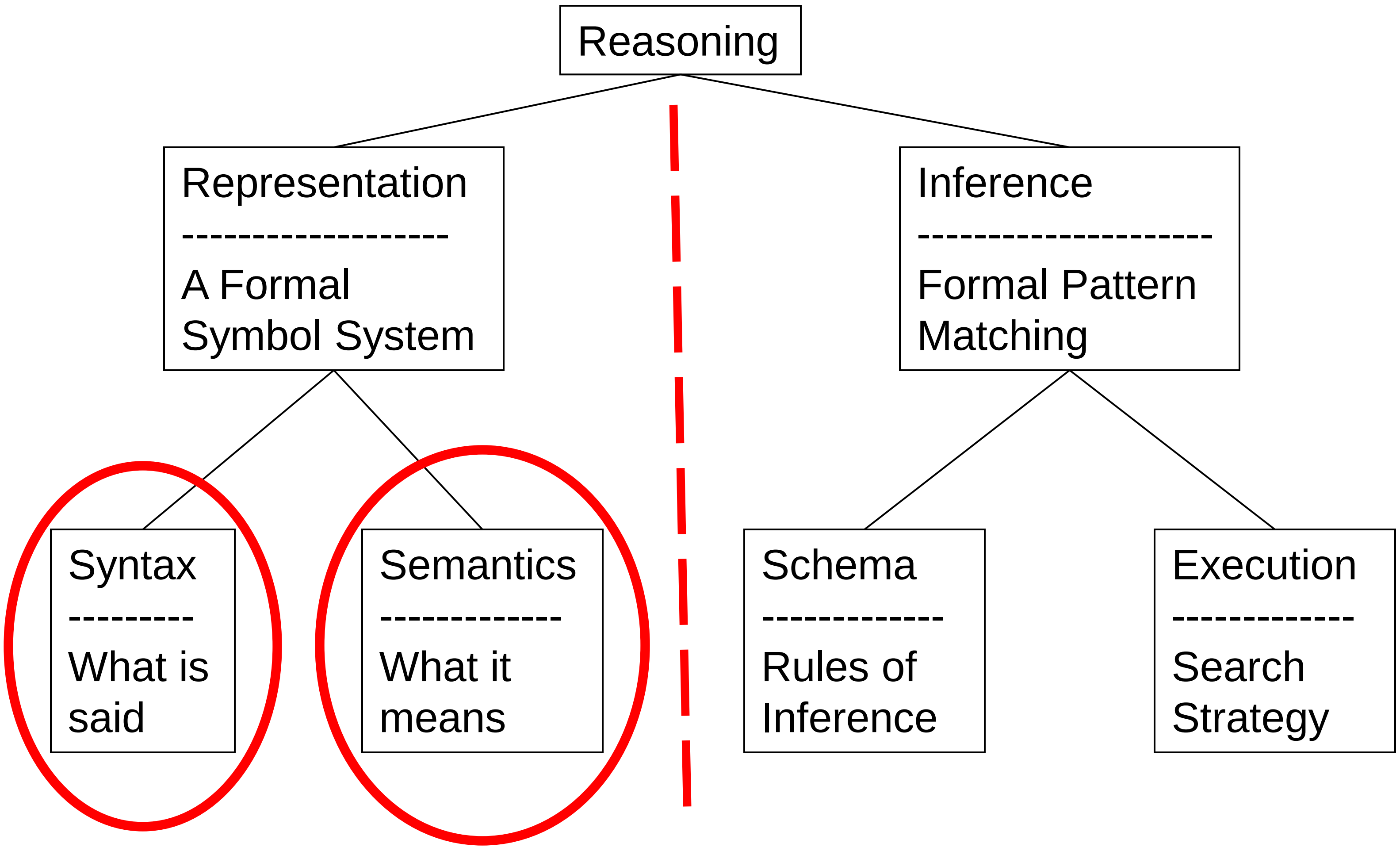
- To unify the grounded propositions with premises of the implication you need to solve a CSP!
- *Colorable*() is inferred iff the CSP has a solution
- CSPs include 3SAT as a special case, hence matching is NP-hard

FOL (or FOPC) Ontology:

What kind of things exist in the world?

What do we need to describe and reason about?

Objects --- with their relations, functions, predicates, properties, and general rules.



Summary

- First-order logic:
  - Much more expressive than propositional logic
  - Allows objects and relations as semantic primitives
  - Universal and existential quantifiers
- Syntax: constants, functions, predicates, equality, quantifiers
- Nested quantifiers
- Translate simple English sentences to FOPC and back
- Unification