PURDUE CS47100

INTRODUCTION TO AI

ANNOUNCEMENTS

- Midterm exam in a week!
  - October 20 (Thursday): 8-10pm, PHYS 112
  - Please contact DRC and reserve a testing room with them NOW if you have accessibility needs!
- Assignment 2: due by the end of Sunday (Oct 16)
  - We may turn some written questions into bonus questions depending on how much we can cover in today's class; watch out for a post on the Ed discussion forum tonight!
  - But you should make sure you understand how to solve all written questions in assignment 2, regardless of whether they are bonus questions…

RECAP: PROPOSITIONAL LOGIC & FIRST-ORDER LOGIC

- Propositional logic: Model checking
  - DPLL
  - WalkSAT
- First-order logic
  - Constants, predicates, functions, variables, connectives, equality, quantifiers

UNIVERSAL QUANTIFICATION

- ∀<variables> <sentence>
- Everyone in CS471 is smart: $\forall x\ in(x, CS471) \Rightarrow Smart(x)$
- $\forall x\ P$ is true in a model m iff P is true with x interpreted as *each* possible object in the model
- Roughly speaking, equivalent to the **conjunction** of **instantiations** of P

$$
\begin{aligned}
& In(John, CS471) \Rightarrow Smart(John) \\
\wedge\ & In(Jane, CS471) \Rightarrow Smart(Jane) \\
\wedge\ & In(CS471, CS471) \Rightarrow Smart(CS471) \\
\wedge\ & \ldots
\end{aligned}
$$

A COMMON MISTAKE TO AVOID

- Typically, $\Rightarrow$ is the main connective with $\forall$

- Common mistake: using $\wedge$ as the main connective with $\forall$
  - $\forall x \; In(x, CS471) \wedge Smart(x)$
  - means “Everyone is in CS471 and everyone is smart”

EXISTENTIAL QUANTIFICATION

- ∃<variables> <sentence>
- Someone in CS471 is smart: $\exists x\ In(x, CS471) \wedge Smart(x)$
- $\exists x\ P$ is true in a model m iff P is true with x interpreted as *some* possible object in the model
- Roughly speaking, equivalent to the **disjunction** of **instantiations** of P

$$
\begin{aligned}
& In(John, CS471) \wedge Smart(John) \\
\vee\ & In(Jane, CS471) \wedge Smart(Jane) \\
\vee\ & In(CS471, CS471) \wedge Smart(CS471) \\
\vee\ & \ldots
\end{aligned}
$$

ANOTHER COMMON MISTAKE TO AVOID

- Typically, $\wedge$ is the main connective with $\exists$
- Common mistake: using $\Rightarrow$ as the main connective with $\exists$
  - $\exists x \, In(x, CS471) \Rightarrow Smart(x)$
  - is true if there is any object that is not in CS471

PROPERTIES OF QUANTIFIERS

- $\forall x\ \forall y$ is the same as $\forall y\ \forall x$
- $\exists x\ \exists y$ is the same as $\exists y\ \exists x$
- $\exists x\ \forall y$ is **not** the same as $\forall y\ \exists x$
  - $\exists x\ \forall y$ Loves(x,y) "There is a person who loves everyone in the world"
  - $\forall y\ \exists x$ Loves(x,y) "Everyone in the world is loved by at least one person"
- **Quantifier duality**: each can be expressed using the other
  - $\forall x$ Likes(x,IceCream) $\neg\exists x\ \neg$Likes(x,IceCream)
  - $\exists x$ Likes(x,Broccoli) $\neg\forall x\ \neg$Likes(x,Broccoli)

EQUALITY

- $term_1 = term_2$ is true under a given interpretation if and only if $term_1$ and $term_2$ refer to the same object
- E.g., definition of Sibling in terms of Parent

$$\forall x,y\ Sibling(x,y) \Leftrightarrow [\neg(x = y) \land \exists m,f\ \neg(m = f) \land Parent(m,x) \land Parent(f,x) \land Parent(m,y) \land Parent(f,y)]$$

EXAMPLES

- Every gardener likes the sun: $\forall x\ Gardener(x) \Rightarrow Likes(x,Sun)$
- You can fool some of the people all of the time: $\exists x\ (Person(x) \wedge \forall t\ (Time(t) \Rightarrow CanFool(x,t)))$
- You can fool all of the people some of the time: $\forall x\ (Person(x) \Rightarrow \exists t\ (Time(t) \wedge CanFool(x,t)))$
- All purple mushrooms are poisonous: $\forall x\ (Mushroom(x) \wedge Purple(x)) \Rightarrow Poisonous(x)$
- No purple mushroom is poisonous:
  $\neg \exists x\ Purple(x) \wedge Mushroom(x) \wedge Poisonous(x)$
  $\forall x\ (Mushroom(x) \wedge Purple(x)) \Rightarrow \neg Poisonous(x)$
- There are exactly two mushrooms:
  $\exists x\ \exists y\ Mushroom(x) \wedge Mushroom(y) \wedge \neg(x=y) \wedge (\forall z\ Mushroom(z) \Rightarrow ((x=z) \vee (y=z)))$

INFERENCE IN FIRST ORDER LOGIC

- How can we use our knowledge of Propositional Logic to solve problems in First Order Logic?
- We can simply convert a problem from FOL to PL, and run resolution on it.

WHAT SYMBOLS ARE PRESENT IN SENTENCES IN FIRST ORDER LOGIC BUT NOT IN PROPOSITIONAL LOGIC?

$\forall \quad \exists$

HOW TO CONVERT?

- Alice
- Bob
- Eve
- John

- AI
- LinearAlgebra
- Probability

$\forall x$ Studies(x, AI) $\Rightarrow$ Awesome(x)

$\exists y$ InClass(y,AI) $\wedge$ WillAttend(y,GradSchool)

$\forall x\ \exists y$ InClass(x,y) $\wedge$ Studies(x,y)

HOW TO CONVERT?

- Alice
- Bob
- Eve
- John

- AI
- LinearAlgebra
- Probability

$\forall x$ Studies(x, AI) $\Rightarrow$ Awesome(x)

- Studies(Alice,AI) $\Rightarrow$ Awesome(Alice)
- Studies(Bob,AI) $\Rightarrow$ Awesome(Bob)
- Studies(Eve,AI) $\Rightarrow$ Awesome(Eve)
- Studies(John,AI) $\Rightarrow$ Awesome(John)
- Studies(AI,AI) $\Rightarrow$ Awesome(AI)
- Studies(LinearAlgebra,AI) $\Rightarrow$ Awesome(LinearAlgebra)
- Studies(Probability,AI) $\Rightarrow$ Awesome(Probability)

UNIVERSAL INSTANTIATION (UI)

- Every instantiation of a universally quantified sentence is entailed by it:

$$\frac{\forall\, v\; \alpha}{\mathrm{SUBST}(\{v, g\}, \alpha)}$$

for any variable v and *ground* term g (term without variables)

  - Example, given the sentence: $\forall$x King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)
  - Then entailed sentences include:
    King(John) $\wedge$ Greedy(John) $\Rightarrow$ Evil(John)
    King(Richard) $\wedge$ Greedy(Richard) $\Rightarrow$ Evil(Richard)
    King(Father(John)) $\wedge$ Greedy(Father(John)) $\Rightarrow$ Evil(Father(John))
    ……

HOW TO CONVERT?

- Alice
- Bob
- Eve
- John

- AI
- LinearAlgebra
- Probability

$\exists$y InClass(y,AI) ∧ WillAttend(y,GradSchool)

- (InClass(Alice,AI) ∧ WillAttend(Alice,GradSchool))

  ∨ (InClass(Bob,AI) ∧ WillAttend(Bob,GradSchool))

  ∨ (InClass(Eve,AI) ∧ WillAttend(Eve,GradSchool))

  ∨ (InClass(John,AI) ∧ WillAttend(John,GradSchool))

  ∨ (InClass(AI,AI) ∧ WillAttend(AI,GradSchool))

  ∨ (InClass(LinearAlgebra,AI) ∧ WillAttend(LinearAlgebra,GradSchool))

  ∨ (InClass(Probability,AI) ∧ WillAttend(Probability,GradSchool))
- **InClass(C,AI) ∧ WillAttend(C,GradSchool)**

Skolem Constant

EXISTENTIAL INSTANTIATION (EI)

- For any sentence α with variable v:

$$\frac{\exists\, v\ \alpha}{\mathrm{SUBST}(\{v, k\}, \alpha)}$$

  for *constant* term k that does **not** appear elsewhere in the knowledge base

- E.g., $\exists x$ Crown(x) ∧ OnHead(x,John) yields:
  Crown(C_1) ∧ OnHead(C_1,John)
  provided C_1 is a new constant symbol (a **Skolem constant**)

- Like saying "We know that there is some x for which this is true, so let's just call it C_1"

HOW TO CONVERT?

- Alice
- Bob
- Eve
- John

- AI
- LinearAlgebra
- Probability

$\forall x\ \exists y\ InClass(x,y) \land Studies(x,y)$

- [(InClass(Alice,Alice) ∧ Studies(Alice,Alice))

  ∨ (InClass(Alice,Bob) ∧ Studies(Alice,Bob)

  ∨ (InClass(Alice,Eve) ∧ Studies(Alice,Eve)

  ∨ (InClass(Alice,John) ∧ Studies(Alice,John)

  ∨ (InClass(Alice,AI) ∧ Studies(Alice,AI)

  ∨ (InClass(Alice,LinearAlgebra) ∧ Studies(Alice,LinearAlgebra)

  ∨ (InClass(Alice,Probability) ∧ Studies(Alice,Probability)]

  ∧ [(InClass(Bob,Alice) ∧ Studies(Bob,Alice)) ∨ …

HOW TO CONVERT?

- Alice
- Bob
- Eve
- John

- AI
- LinearAlgebra
- Probability

∀x ∃y InClass(x,y) ∧ Studies(x,y)

- [(InClass(Alice,Alice) ∧ Studies(Alice,Alice))

  ∨ (InClass(Alice,Bob) ∧ Studies(Alice,Bob)

  ∨ (InClass(Alice,Eve) ∧ Studies(Alice,Eve)

  ∨ (InClass(Alice,John) ∧ Studies(Alice,John)

  ∨ (InClass(Alice,AI) ∧ Studies(Alice,AI)

  ∨ (InClass(Alice,LinearAlgebra) ∧ Studies(Alice,LinearAlgebra)

  ∨ (InClass(Alice,Probability) ∧ Studies(Alice,Probability)]

  ∧ [(InClass(Bob,Alice) ∧ Studies(Bob,Alice)) ∨ …

∀x InClass(x,C) ∧ Studies(x,C)

Skolem Constant

HOW TO CONVERT?

- Alice
- Bob
- Eve
- John

- AI
- LinearAlgebra
- Probability

$\forall x\ \exists y\ InClass(x,y) \land Studies(x,y)$

- [(InClass(Alice,Alice) ∧ Studies(Alice,Alice))

  ∨ (InClass(Alice,Bob) ∧ Studies(Alice,Bob)

  ∨ (InClass(Alice,Eve) ∧ Studies(Alice,Eve)

  ∨ (InClass(Alice,John) ∧ Studies(Alice,John)

  ∨ (InClass(Alice,AI) ∧ Studies(Alice,AI)

  ∨ (InClass(Alice,LinearAlgebra) ∧ Studies(Alice,LinearAlgebra)

  ∨ (InClass(Alice,Probability) ∧ Studies(Alice,Probability)]

  ∧ [(InClass(Bob,Alice) ∧ Studies(Bob,Alice)) ∨ …

$$\forall x\ InClass(x,C(x)) \land Studies(x,C(x))$$

Skolem Function

REDUCTION TO PROPOSITIONAL INFERENCE

- Given a KB containing:
  $\forall x$ King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)
  King(John)
  Greedy(John)
  Brother(Richard,John)

- Instantiate the universal sentence in **all possible** ways:
  King(John) $\wedge$ Greedy(John) $\Rightarrow$ Evil(John)
  King(Richard) $\wedge$ Greedy(Richard) $\Rightarrow$ Evil(Richard)
  King(John)
  Greedy(John)
  Brother(Richard,John)

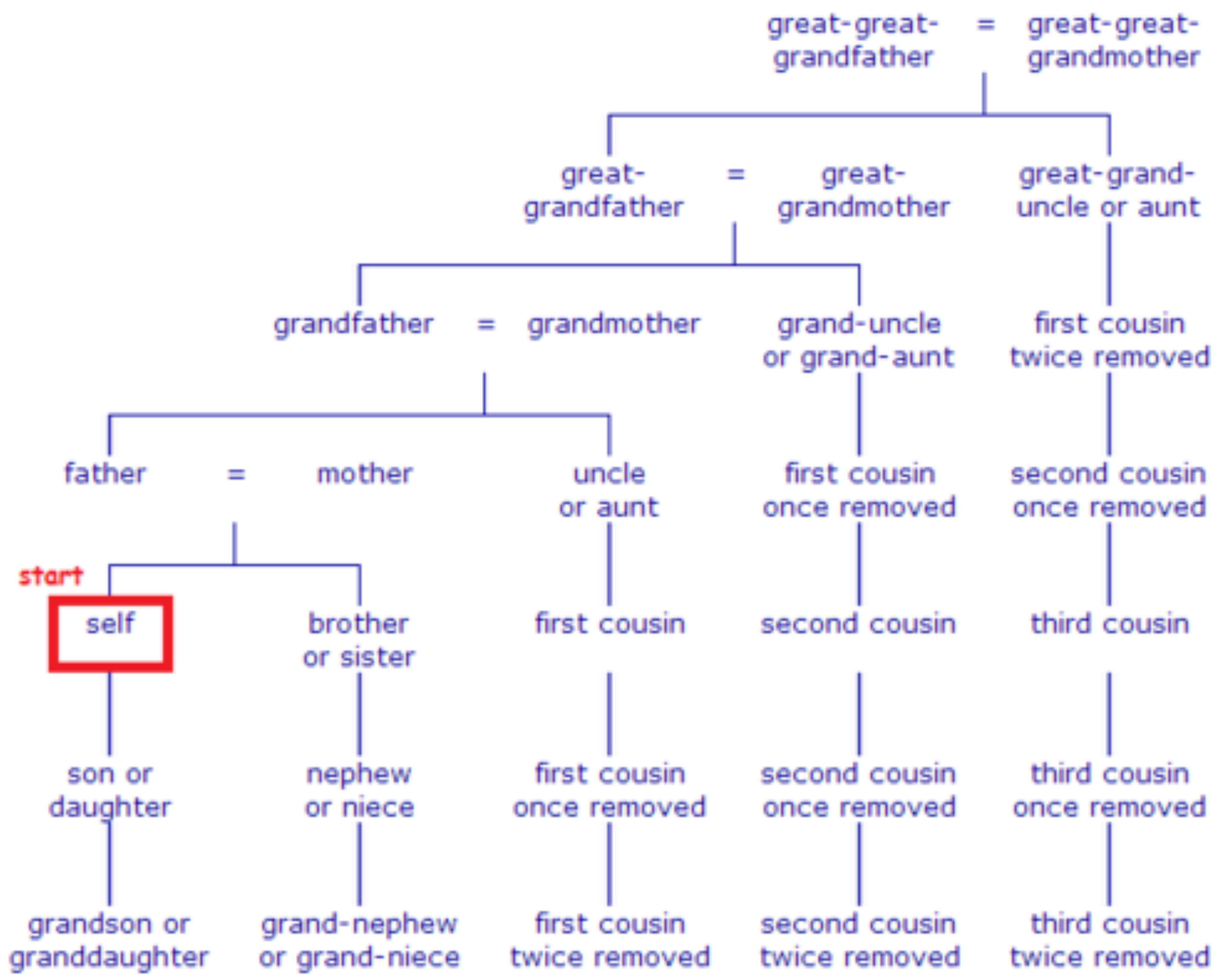
- The new KB is **propositionalized** with proposition symbols:
  - King(John), Greedy(John), Evil(John), King(Richard), etc.

REDUCTION TO PROPOSITIONAL INFERENCE

- Every FOL KB can be propositionalized so as to preserve entailment
- That is, a ground sentence is entailed by new KB iff entailed by original KB
- Thus, we can propositionalize KB and query, apply resolution, and return result
- However, what other aspect of first-order logic differentiates it from propositional logic?
  - *Functions!*
  - *E.g., Father(John), LeftLeg(Richard), Successor(4)*

FOL KB CAN BE INFINITE IN PL

- Father(John) = Bob
- But what about?
  Father(Father(Father(Father(John))))
- Can go arbitrarily deep



https://metrouk2.files.wordpress.com/2014/10/family.png

PROBLEMS WITH PROPOSITIONALIZATION (1)

- With function symbols, there are infinitely many ground terms
- Theorem – If a sentence α is entailed by an FOL KB, it is entailed by a **finite** subset of the propositionalized KB *(Herbrand 1930)*
- Idea – For n = 0 to ∞ do
  - Create a propositional KB by instantiating with depth n terms
  - See if α is entailed by this KB
- Works if α is entailed, loops if α is not entailed!
- Theorem – Entailment for FOL is **semidecidable** (algorithms exist that say yes to every entailed sentence, but no algorithm exists that also says no to every nonentailed sentence) *(Turing 1936, Church 1936)*

PROBLEMS WITH PROPOSITIONALIZATION (2)

- Propositionalization seems to generate lots of irrelevant sentences.
- For example, from the KB
  $\forall x\ King(x) \wedge Greedy(x) \Rightarrow Evil(x)$
  King(John)
  $\forall y\ Greedy(y)$
  Brother(Richard,John)
- It seems obvious that Evil(John), but propositionalization produces lots of facts such as Greedy(Richard) that are irrelevant
- With *p* k-ary predicates and n constants, there are pn^k instantiations.
- With function symbols, it gets much worse

SOLUTION

- More efficient inference methods that work directly with variables
  - Unification
  - Resolution
  - Generalized Modus Ponens
  - Forward chaining and backward chaining

LIFTED INFERENCE

- Find substitutions that make different logical expressions look identical
- Example:
  - $\forall x$ King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)
  - King(John)
  - $\forall y$ Greedy(y)
- Find some x such that the premise of a rule is identical to sentence already in the KB, then we can assert the conclusion
  - {x/John, y/John} then we can conclude Evil(John)

UNIFICATION

- Two sentences α,β can be **unified** with substitution θ if SUBST(θ, α)=SUBST(θ, β)

α	β	θ
Knows(John,x)	Knows(John,Jane)	
Knows(John,x)	Knows(y,Steve)	
Knows(John,x)	Knows(y,Mother(y))	
Knows(John,x)	Knows(x,Steve)	

UNIFICATION

- Two sentences α,β can be **unified** with substitution θ if SUBST(θ, α)=SUBST(θ, β)

α	β	θ
Knows(John,x)	Knows(John,Jane)	**{x/Jane}**
Knows(John,x)	Knows(y,Steve)	
Knows(John,x)	Knows(y,Mother(y))	
Knows(John,x)	Knows(x,Steve)	

UNIFICATION

- Two sentences α,β can be **unified** with substitution θ if SUBST(θ, α)=SUBST(θ, β)

α	β	θ
Knows(John,x)	Knows(John,Jane)	**{x/Jane}**
Knows(John,x)	Knows(y,Steve)	**{x/Steve,y/John}**
Knows(John,x)	Knows(y,Mother(y))	
Knows(John,x)	Knows(x,Steve)	

UNIFICATION

- Two sentences α,β can be **unified** with substitution θ if SUBST(θ, α)=SUBST(θ, β)

α	β	θ
Knows(John,x)	Knows(John,Jane)	**{x/Jane}**
Knows(John,x)	Knows(y,Steve)	**{x/Steve,y/John}**
Knows(John,x)	Knows(y,Mother(y))	**{y/John,x/Mother(John)}**
Knows(John,x)	Knows(x,Steve)	

UNIFICATION

- Two sentences α,β can be **unified** with substitution θ if SUBST(θ, α)=SUBST(θ, β)

α	β	θ
Knows(John,x)	Knows(John,Jane)	**{x/Jane}**
Knows(John,x)	Knows(y,Steve)	**{x/Steve,y/John}**
Knows(John,x)	Knows(y,Mother(y))	**{y/John,x/Mother(John)}**
Knows(John,x)	Knows(x,Steve)	**{fail}**

- Standardizing apart eliminates overlap of variables, e.g., Knows(z,Steve)

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.

- For every unifiable pair of expressions, there exists a single **most general unifier** (MGU) that is unique up to renaming of variables.
  - MGU = { y/John, x/z }