HW1 due Tuesday 1/31

- Our First (trial) Q/A Sessions (completely optional)
- Monday 1/30/17; Room 4405 SC
- 9-10AM
- 12-1PM
- 6-7PM
- 7-8PM

We may lose the room (another facet of registration problem)
If we lose the room, look for a note on the door for a new location
Any Pending Questions?
Planning, Knowledge Representation, Logic Terms

- Operator / Action
- Precondition, Delete, Add lists
- Minimal commitment planner
- Goal / Subgoal
- Sussman Anomoly
- Situation Calculus
- Fluent

- First Order Logic / FOL / FOPC
- Predicate, Variable, Function, Constant
- Term, Literal, WFF
- Connective, Quantifier
- Syntax, Semantics
- Inference
- Inference rules, modus ponens, resolution, etc.
- Unification, MGU
- FOL Monotonicity
First-Order Logic

• Planning
• Knowledge Representation
• Symbolic Inference

• Strengths: powerful, cognitive, expressive
• Weaknesses: uncertainty
• Modern applications: scheduling, computer games, logistics, constraint / satisfiability problems
Domain Independent Planning

• Study process of planning
  Abstract; not domain dependent
• Dynamics of the world: Operators
• Problem: Initial State & Goal

Initial State -> Planner -> Solution
Goal Specification
Operator Definitions
All Reachable Situations are Defined
Given: 1) the Initial State
2) Axioms of World Change (operator definitions)

Planning: find a situation where the goal holds
This is symbolic theorem proving

Tree or Graph of states connected by actions (instantiations of operators)
Planning vs. Search
for problem solving

| Interesting action sequences | All action sequences |

Search states and actions are “inferentially opaque”
just check preconditions and generate children

Planning allows reasoning about how operators interact with state features

Supports Means – Ends analysis
Traditional Blocks World
Extremely simple!
Why?

Only support relationships change: On, Clr
A block can support at most one other block
Only one table; no left or right
The table has room for any number of blocks
Generalized block movement – no gripper
Decompose state & work on pieces

- Number of possible actions >> Number of relevant actions
- Search cannot reason about parts; no subgoals
- We want tower EFCG
- What actions *can* we do? (search)
- What actions *should* we do? (planning)
Each World State is a *Conjunction* of Properties that Hold in the World State

$S_0$

$S_{342}$
Each World State is a *Conjunction* of Properties that Hold in the World State

\[ S_0 \]

\[ \ldots \]

\[ \text{(abstract)} \]

\[ \text{(blocks world example)} \]
First order logic FOL
also known as first order predicate calculus (FOPC),
allows us to describe and reason about the world
in a computation-friendly way

On(A, C)
On(C, Tbl)
On(B, Tbl)
Blk(A)
Blk(B)
Blk(C)
Table(Tbl)
Clr(A)
Clr(B)
Clr(Tbl)
FOL as a knowledge representation language

All names (A, B, C, Tbl, On, Blk, Table) are arbitrary symbols carrying NO meaning.

Predicates: On, Blk, Table, Clr

Constants: A, B, Tbl,…

Also Variables, Functions, Connectives, Quantifiers

We will use an initial capital letter to denote predicates and constants.

On(A, C)

Each predicate (like ‘On’) denotes a relation that hold among its arguments, here the objects A and C.
First Order Logic also allows variables to denote objects in the world

Suppose we have the goal that Block B is on some other block (and not on the table) but we do not know or care which one.

We will use lower case lettersprefaced by ‘?’ to denote variables
Operators Capture World Change

• Preconditions
  – Specify if the operator may be applied
  – Possibly alternative ways to satisfy in a state

• Effects: How the state changes upon execution
  – Some properties remain true
  – Some are no longer true
  – Some become true

• An *action* is a particular way of instantiating the operator in a state
World Changes
Action must fully define resulting world state

Si

Result of Aj in Si

delete

add

persist

...
Strips-type Planning Operators

Operator = Three lists of atomic formulae (simple positive FOL statements)
Negations and disjunctions are problematic in Strips

**Precondition** list: If all hold in a state, the operator may be applied in that state
**Delete** list: Statements to be removed from the current state
**Add** list: Statements to be included in the resulting state

Effects = Delete and Add lists
Strips-type Planning Operators

• Concise because usually
  \[\text{|persist|} \gg \text{|add|} + \text{|delete|}\]
  State descriptions persist by default

• Order is important
  – First evaluate PC
  – Then apply Delete
  – Finally apply Add
  – All three lists are simple atomic statements

• Alternative: Situation Calculus
  – add
  – persist
    State descriptions are deleted by default
Consider an Operator that can Move A from C to B

Initial State:

Next State:

This is an action
We want an operator
Must handle all conceptually similar actions
Operator:
name & variables
preconditions
delete list
add list
Name & Vars:

Preconditions:

Delete:

Add:
Name & Vars: MOVE(\(x, ?z\))

Preconditions: Blk (\(x\)), C lr (\(x\)), On (\(x, ?y\)), C lr (\(z\)), D iff (\(x, ?z\)), D iff (\(y, ?z\))

Delete: On (\(x, ?y\)), C lr (\(z\))

Add: On (\(x, ?z\)), C lr (\(?y\))
Now we want to
Move A from C to the Table

Initial State:

Next State:

On(A, C)

On(A, Tbl)
Now we want to Move A from C to the Table

Initial State:

Next State:

MOVE(?x=A, ?z= Tbl)

PC: Blk (?x),Clr (?x), On (?x, ?y),
    Clr (?z), Diff (?x, ?z), Diff (?y, ?z)

Del: On (?x, ?y), Clr (?z)

Add: On (?x, ?z), Clr (?y)
MoveToTable Operator
Move Object to Destination

Initial State:

Next State:

On(A, Tbl)
MoveToTable(\(?x\)):

PC: Clr (\(?x\)), On (\(?x, \ ?y\)), Blk (\(?x\)),
    Tbl (\(?z\)), Blk (\(?y\))

Delete: On (\(?x, \ ?y\))

Add: On (\(?x, \ ?z\)), Clr (\(?y\))
Need a Second Move Operator

MoveToBlock (?x, ?z):

PC: Clr (?x), Clr (?z), On (?x, ?y), Blk (?x), Blk (?z), Diff (?x, ?z), Diff (?y, ?z)

Delete: On (?x, ?y), Clr (?z)
Add: On (?x, ?z), Clr (?y)

These cannot be combined into a single Move operator
The operator syntax requires three positive lists of simple FOL statements
Combining would require a conditional effect
(to avoid deleting the table being clear)
Russell & Norvig’s Operator Notation is different (USE OURS)

They combine ‘Delete & ‘Add’ into ‘Effects’

Deletes are marked by ‘\neg’
(the negation sign)

But it is NOT negation

It merely a syntactic marker to designate the positive literals to be deleted

There are semantic difficulties in using a true negation in these planning operators
Initial State

1. Check if the goal statements are satisfied in the current state
2. If not, select one or more unsatisfied statements
3. Find an operator whose ADD statements match \((UNIFY\ with\ them)\)
4. Remove (but protect) these statements; add the operator’s preconditions as subgoals
5. Keep track of alternative selections, unifications & operators, watch out for protection violations

This is a search
Use any of our search methods including possible operator cost model, heuristic function,...
Backward Chaining Plan

Initial State

MoveToBlock(B, ?x)
can add On(B, ?x)
PC: Clr(?x), Clr(B),…

MoveToTable(A, Tbl)
can add Clr(?x) { ?x=C}
PC:…

Goal

On(B, ?x)
On(?x, Tbl)
Blk(?x)
Tbl(Tbl)

I.S.

?x = C
Planning Nuances

• STRIPS is FOPC but less expressive
• Care must be taken to avoid planner unsoundness and incompleteness
  – Unsound: a plan is found that does not work
  – Incomplete: a plan exists but none can be found
• Sussman Anomoly
Planning Subtleties: the Sussman Anomaly

Initial State

Goal: On (A, B) \land On (B, C)

Try On (A, B) …

But we considered only one goal ordering…

Try On (B, C) first…

What’s the problem? What’s the fix?
Sussman Anomaly

Demonstrate unsoundness or incompleteness?
Underlying difficulty: Interacting conjunctive goals
Our naïve algorithm assumed linearity / independence / non-interaction
Elegant exposition of an underlying problem
  Don’t just want to stack blocks!
  Theory of domain independent planning!
Minimal commitment planner
Exposé and keep track of all planning decisions
Choosing an action and scheduling the action are distinct planning decisions
Our Two Strips Operators

MoveB (?x, ?z):

PC: Clr (?x), Clr (?z), On (?x, ?y), Blk (?x),
    Blk (?z), Diff (?x, ?z), Diff (?y, ?z)

DEL: On (?x, ?y), Clr (?z),
ADD: On (?x, ?z), Clr (?y)

MoveT (?x):

PC: Clr (?x), On (?x, ?y), Blk (?x), Tbl (?z), Blk(?y)

DEL: On (?x, ?y)
ADD: On (?x, ?z), Clr (?y)
A minimal-commitment planner can produce this without backing up
M1 must be scheduled after M3 and M2 after M1 due to protected relations
• Logic Lect 1 ends here
Announcement

• Homework 2 is available on the website
• Due 2/7 (2/5 for extra credit)
Is it possible to have the slides on the website match the slides in lecture?

- The final version of the slides appears by noon the day following the lecture.
- They can be made always to match but only if we forego preliminary slides.
A minimal-commitment planner can produce this without backing up  
M1 must be scheduled after M3 and M2 after M1 due to protected relations
This is Classical Planning

<table>
<thead>
<tr>
<th>Applications</th>
<th>Other forms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Military logistics</td>
<td>Scheduling</td>
</tr>
<tr>
<td>Transportation routing</td>
<td>Path Planning</td>
</tr>
<tr>
<td>Pickup / Delivery scheduling</td>
<td>Multi-Agent Planning</td>
</tr>
<tr>
<td>NASA Satellite Access, Hubble, Deep Space,...</td>
<td>Planning with/for Sensing</td>
</tr>
<tr>
<td>Multiobjective combinatorial bidding</td>
<td>Temporal Planning</td>
</tr>
<tr>
<td>Computer games</td>
<td>Situated Action</td>
</tr>
<tr>
<td>[many others]</td>
<td>RL (next after Logic)</td>
</tr>
<tr>
<td></td>
<td>[many others]</td>
</tr>
</tbody>
</table>
First Order Logic

- FOL aka FOPC (First Order Predicate Calculus)
- STRIPS is FOL but less expressive due to the requirement that the lists are a conjunction of simple positive statements
- General Knowledge Representation Language
- Symbolic Inference
Inference

• Coming to *realize explicitly* something that already holds
  – recall search
  – also later w/ statistical inference

• Given some axioms capturing our knowledge about the world

• Using a formal inference mechanism

• Conclude other statements of interest that must hold in the world
Why not English for Knowledge Representation and Inference?

Apples are delicious things
Delicious things are edible
Therefore...

“I’ve eaten apples…yes! they are delicious and edible”

“Hold on, I’ve eaten apples. They are delicious, but they give me bad indigestion; they are not edible.”

Philosophical / AI problem of “grounding”
Symbolic Logic offers a solution
Symbolic Inference

A \implies B
B \implies C

Therefore:

A \implies C

…

But what does “B” mean / stand for?

But truth without “understanding” is therefore possible.
Objects are denoted by symbols:

Andy17 denotes

object constant symbol
constant symbol
object constant

For us, different object constants implies different objects and vice versa.

The symbol / object association is arbitrary:
Car54 denotes

Andy Smith
Age 23
Height 5’9”
...
Predicates / Relations are also Denoted by Symbols

Married(Andy17, Car54)

A particular relationship exists between the individuals

Predicates are n-ary

Meaning of a predicate is a (possibly infinite) set of n-tuples:
{(Joe23, Jill6), (Liz13, Fred972), ...(Andy17, Car54)...}

here we used the symbols but really its their denotations

This is the semantics (meaning) of the predicate

The statement is truth-valuable; it claims (Andy17, Car54) is a member of the set Married
Functions are Denoted by Symbols

Father-of(Andy17)

Function Symbol

Another way of denoting an individual i.e., John3; the expression is NOT truth-valuable

Functions are n-ary

Meaning of a function is a (possibly infinite) set of n+1 tuples: 
{(Andy17, John3), (Liz13, John3), ...(Joe23, Fred972) ...}

These are NOT algorithmic functions; we do not write them

All we have to do is name them – the rest is taken care of for us
Variables - another type of symbol

- **First Order** (also higher order, not zeroth / propositional)
- Stand only for individuals in the universe of discourse
- Not functions or relations (that is higher order)
- Ours will always be “bound” (cf “free”)

“within the scope of a quantifier”
(NB: NOT a programming notion)

**Important quantifiers**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \exists )</td>
<td>existential</td>
<td>“there exists”</td>
</tr>
<tr>
<td>( \forall )</td>
<td>universal</td>
<td>“for all”</td>
</tr>
</tbody>
</table>
In the COMPUTER

Object constant
Variable
Function expression
Predicate symbol

In the WORLD

Individuals
Properties
Relations

Denotation / Meaning

All Symbols

All Real Things
Logical Connectives

\neg \text{ negation} \quad \text{“not”}

\wedge \text{ conjunction} \quad \text{“and”}

\vee \text{ disjunction} \quad \text{“or”}

\Rightarrow \text{ implication} \quad \text{“implies”}

\Leftrightarrow \text{ equivalence} \quad \text{“if and only if”}

A \Rightarrow B \text{ means precisely } \neg A \vee B

\Leftrightarrow \text{ is just } \Rightarrow \text{ both directions}
A *term* denotes an individual in the universe of discourse
variable
object constant
function expression

A *function expression* is an n-ary function symbol with n
terms as arguments

An *atom* (also atomic sentence, atomic Well-Formed
Formula) is an n-ary predicate symbol with n terms as
arguments

A *literal* is an atom or a negated atom
Well Formed Formulas (WFFs)

Atoms are Well-Formed Formulas (WFFs)

If \( \Theta \) and \( \Phi \) are WFFs then so are

\[
\forall x \, \Theta \quad \exists x \, \Theta \quad \neg \Theta
\]

\[
\Theta \land \Phi \quad \Theta \lor \Phi
\]

\[
\Theta \Rightarrow \Phi \quad \Theta \Leftrightarrow \Phi
\]

Quantifiers \( \forall \) (for all) and \( \exists \) (there exists)

Logical implication \( \Theta \Rightarrow \Phi \) is precisely \( \neg \Theta \lor \Phi \)

(\textit{not} English implication!)

\( \Theta \Leftrightarrow \Phi \) is precisely \( (\Theta \Rightarrow \Phi) \land (\Phi \Rightarrow \Theta) \)

\( \lor \) “or” is inclusive
English / WFF Examples

Some student is named “John”

\[ \exists x \ [ \text{Student}(x) \land \text{Name}(x, \text{“John”})] \]

Every student owns a computer

\[ \forall x \ [ \text{Student}(x) \Rightarrow \exists y \ (\text{Computer}(y) \land \text{Owns}(x,y))] \]

Scope of y

Scope of x

\[ \exists y \ [ \text{Computer}(y) \land \forall x \ (\text{Student}(x) \Rightarrow \text{Owns}(x,y))] \]

WFFs have different meanings

The English statement is ambiguous
More Examples

Birds fly.

Some birds fly.

Room 1320 DCL is empty.

Some Ford is better than any Buick.

Someone on the basketball team is taller than anyone on the football team.
“Birds Fly”

\[\forall x \ [\text{Bird}(x) \implies \text{Flies}(x)]\]

\[\forall x \ [\text{B}(x) \implies \text{F}(x)]\]

where \(B\) means “is a bird” and \(F\) means “can fly”

We can also think about the meaning as

“There are no birds that cannot fly”

\[\neg \exists x \ [\text{Bird}(x) \land \neg \text{Flies}(x)]\]

These are equivalent: the two predicate calculus sentences have the same meaning although they look quite different. (unambiguous but not canonical)
“Birds Fly”

\[ \forall x [\text{Bird}(x) \Rightarrow \text{Flies}(x)] \text{ or } \forall x [\text{B}(x) \Rightarrow \text{F}(x)] \]

where B means “is a bird” and F means “can fly”

Equivalently: \[ \forall x [\neg \text{Bird}(x) \lor \text{Flies}(x)] \]

We can also think about the meaning as

“There are no birds that cannot fly”

\[ \neg \exists x [\text{Bird}(x) \land \neg \text{Flies}(x)] \]

These are equivalent: the three predicate calculus sentences have the same meaning although they look quite different. (unambiguous but not canonical)
Some birds fly.
\[ \exists x \ [\text{Bird}(x) \land \text{Flies}(x)] \]
Note: in logic “some” traditionally means “at least one”

Room 1320 DCL is empty. [taken to mean empty of people]
Really Bad: \[ P \]
Poor: \[ \text{Empty}(\text{Room DCL1320}) \]
Better: \[ \forall x \ [\text{Person}(x) \implies \]
\[ \text{Different}(\text{Location-of}(x), \text{DCL1320})] \]
Still Better: \[ \forall x \forall y \ [(\text{Person}(x) \land \text{Location}(y) \land \text{At}(x,y)) \]
\[ \implies \text{Different}(y, \text{DCL1320})] \]
Completely Wrong: (why?)
\[ \forall x \ [\text{Person}(x) \implies \text{At}(x, \neg \text{DCL1320})] \]

NOTE: functions (like Location-of) are partial...
Some Ford is better than any Buick.

$$\exists x \ [\text{Ford}(x) \land \forall y \ [\text{Buick}(y) \Rightarrow \text{Better}(x,y)]]$$

Better($x,y$) means “$x$ is better than $y$”

Someone on the basketball team is taller than anyone on the football team.

$$\exists x \ [\text{Member}(x,\text{BBallTeam}) \land$$
$$\forall y \ \forall z \ \forall w \ [(\text{Height}(x,z) \land \text{Member}(y,\text{FBallTeam}) \land \text{Height}(y,w))$$
$$\Rightarrow \text{Greater}(z,w)]]$$

Greater($x,y$) means “$x$ is larger than $y$”
Planning Operators

• Planning operators are FOL statements with
  – Special syntax
  – Additional constraints for efficiency
• Using normal syntax & without constraints it becomes *Situation Calculus*
• A “situation” is just a world state
• “Moving a block” states a constraint that holds between two situations separated by a particular action
Recall one of our Move operators

MoveToBlock (?x, ?z):

PC: Clr (?x), Clr (?z), On (?x, ?y), Blk (?x), Blk (?z), Diff (?x, ?z), Diff (?y, ?z)

Delete: On (?x, ?y), Clr (?z)
Add: On (?x, ?z), Clr (?y)
Situation Calculus Move Operator

Move(x,y) has the form:

\[ \forall x \ \forall y \ \forall z \ \forall s [\Theta \Rightarrow \Psi] \]

If \( \Theta \) holds in the current situation: s
- x is on y
- z is clear
- x is a block
- x is clear
- ...

Then \( \Psi \) will hold in the next situation: s after Move
- x is on z
- y is clear
- ...

Preconditions

Effects
“On” and other Fluents

• Relations that can change: On, CLr
• But not Blk or Table
• FOL is *monotonic*
• Instead of On(A, B) we need On(A, B, S)
  – On(A, B) says A is always on B
  – On(A, B, S) says A is always on B in situation S
• The *Result* function
  – “s” is a situation and “a” is an action
  – Result(a, s) denotes the situation resulting from the action
• Might say On(A, Tbl, Result(Move(A, Tbl), s))
  New Situation
the Move operator

\( \text{Move}(x, y) \)

\[ \forall x \forall y \forall z \forall s [ \]

\( (\text{Clr}(x, s) \land \text{Clr}(z, s) \land \text{On}(x, y, s) \land \text{Blk}(x) \land \text{Diff}(x, z) \land \text{Diff}(y, z)) \]

\[ \implies \]

\( (\text{On}(x, z, \text{Result}(\text{Move}(x, y), s)) \land \text{Clr}(y, \text{Result}(\text{Move}(x, y), s)) \land \text{Clr}(x, \text{Result}(\text{Move}(x, y), s)) \land \text{Table}(z) \implies \text{Clr}(z, \text{Result}(\text{Move}(x, y), s)) ) ] \]

\text{Conditional Effects are OK}

Only Partial. Also need what does \textit{not} change
Logic Lect 2 ends here
HW2 due Tuesday 2/7

• Optional Q/A Sessions
  Monday 2/6; Room 3124 SC (different room)
  • 9-10AM
  • 12-1PM
  • 6-7PM
  • 7-8PM

A better room for us, but again we’re not supposed to use it for class. If we lose the room, look for a note on the door for a new location
Announcements

• Remember your iClickers next time
• Reinforcement Learning next
• An relevant faculty candidate talk
• “New Results in Statistical Reinforcement Learning”
• Thursday, February 9, @ 10 a.m.
  Location: 2405 Siebel Center
• A week or so early; probably a little advanced...
• But a student’s reach should exceed his/her grasp
Please Please

• I have had additional complaints about noise during lectures
• We are packed into a big room
• It has hard walls and little noise abatement
• Avoid talking
• Avoid typing
• If you must, then do so as quietly as possible
• Please be considerate of your classmates
First Order Logic

• Strips-like planning operators are specialized FOL statements

• A general translation procedure shows this

MoveToBlock (?x, ?z):

PC: Clr (?x), Clr (?z), On (?x, ?y), Blk (?x), Blk (?z), Diff (?x, ?z), Diff (?y, ?z)

Delete: On (?x, ?y), Clr (?z)
Add: On (?x, ?z), Clr (?y)
Situation Calculus Move Operator

Move(x,y) has the form:

$$\forall x \forall y \forall z \forall s [\Theta \Rightarrow \Psi]$$

If $\Theta$ holds in the current situation: $s$
- x is on y
- z is clear
- x is a block
- x is clear
- ...

Then $\Psi$ will hold in the next situation: $s$ after Move
- x is on z
- y is clear
- ...

Preconditions

Effects
“On” and other Fluents

• Relations that can change: On, CLr
• But not Blk or Table
• FOL is monotonic
• Instead of On(A, B) we need On(A, B, S)
  – On(A, B) says A is always on B
  – On(A, B, S) says A is always on B in situation S
• The Result function
  – “s” is a situation and “a” is an action
  – Result(a, s) denotes the situation resulting from the action
• Might say On(A, Tbl, Result(Move(A, Tbl), s))
  New Situation
the Move operator

Only Partial

Move(x, y)

∀x ∀y ∀z ∀s [

(Cl r (x, s) ∧ Cl r (z, s) ∧ O n (x, y, s)
∧ B l k (x) ∧ D iff(x, z) ∧ D iff(y, z))

⇒

(On (x, z, Result (Move (x, y), s)) ∧
C l r (y, Result (Move (x, y), s)) ∧
C l r (x, Result (Move (x, y), s)) ∧
Table (z) ⇒
    Cl r (z, Result (Move (x, y), s)) ) ]

Conditional Effects and Negations are OK in FOL
Why Only Partial?
Action must fully define resulting world state

Si
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   

Result of Aj in Si
   
   
   
   
   
   
   
   
   
   
   
   
   

{ add }  
   
   
   
   
   
   
   
   
   
   
   
   
   

{ persist }  
   
   
   
   
   
   
   
   
   
   
   

{ delete }  
   
   
   
   
   
   
   
   
   
   
   

...
Why Only Partial?

- FOL needs a way deriving everything that we want it to believe
- There is no default as in Strips operators (recall there statements persist by default)
- Strips operators specify
  - Add
  - Delete
- FOL operators specify
  - Add
  - Persist
- We have the *change axiom*
- *Frame axioms* specifies things that persist (remain unchanged by an action)
Unification

• Performing inference requires matching with variables

• For example, operators in planning:
  – matching preconditions to a state
  – matching add items to a subgoal
    – On(y, B, Result(Move(u,v), s1)) with On(A, x, s2)

• But also general FOL inference

• This is done by unification
Unification

Inference requires conditional matching

All men are mortal
Socrates is a man

\[ \forall x \ (\text{Man}(x) \Rightarrow \text{Mortal}(x)) \]
\[ \text{Man}(\text{Socrates}) \]

\[ \text{Socrates is mortal} \]
\[ \text{Mortal}(\text{Socrates}) \]

\text{Man}(x) \text{ unifies with Man}(\text{Socrates})

Matching ‘x’ to Socrates in the antecedent
specializes the consequent to Mortal(Socrates)

Two expressions UNIFY if they can be made identical by
constraining universally quantified variables to be other terms

A unifier keeps track of the constraints: \( \{x = \text{Socrates}\} \)
A *unifier* (also *substitution*, sometimes: *binding list*) is a set of pairings of variables with terms:

\[ \{ v_1 = e_1, v_2 = e_2, v_3 = e_3, \ldots v_n = e_n \} \]

variables: \( v_1, v_2, v_3 \ldots \)

terms: \( e_1, e_2, e_3 \ldots \)

1. each variable is paired at most once
2. a variable’s pairing term may not contain the variable directly or indirectly

\{x = \text{Socrates}\}  \quad \text{(No “?” because \( \forall \) marked x as a variable)}

* Do not confuse with bound / free variables!!!
Are These Acceptable Unifiers?

\{
\{x = y, z = F(x)\} \quad \text{YES}
\{x = y, z = F(y), x = A\} \quad \text{NO}
\{x = y, z = F(y), y = A\} \quad \text{YES}
\{x = y, y = F(z), z = G(x)\} \quad \text{NO}
\}
Do these Unify?

\[ P(x, y, z) \quad \quad P(w, w, Fred) \]

\{x=Fred, y=Fred, z=Fred, w=Fred\}

Yields: \( P(Fred, Fred, Fred) \)

Yes.

A set (of any number) of expressions unifies iff there exists a single unifier such that become character for character identical after applying the unifier to each expression
**Most General Unifier MGU**

The MGU imposes the fewest constraints, specifying the *weakest* conditions for matching

MGU is unique assuming

- order is not important
- variable names are not important
  (alphabetic variants)

Applying a unifier to an expression yields a *unification instance*.

Applying the MGU to an expression yields a *most general unification instance*.

Hint: A variable with a paired expression in the unifier WILL NEVER APPEAR in a unification instance
What is the MGU?

\[ P(x, y, z) \quad P(w, w, Fred) \]

\{x=w, y=w, z=Fred\}

Yields \( P(w, w, Fred) \)

Equivalently, \{x=u, y=u, w=u, z=Fred\}

Yields the alphabetic variant \( P(u, u, Fred) \)
What is the MGU?

<table>
<thead>
<tr>
<th>M(Ann, x, Bob)</th>
<th>M(Ann, x, Bob)</th>
</tr>
</thead>
<tbody>
<tr>
<td>M(Ann, x, Bob)</td>
<td>M(y, x, Chuck)</td>
</tr>
<tr>
<td>M(Ann, x, Bob)</td>
<td>M(y, x, Father-of(Chuck))</td>
</tr>
<tr>
<td>P(w, w, Fred)</td>
<td>P(x, y, y)</td>
</tr>
<tr>
<td>Q(r, r)</td>
<td>Q(x, F(x))</td>
</tr>
<tr>
<td>R(G(x, Bob), y, y)</td>
<td>R(z, G(Fred, w), z)</td>
</tr>
</tbody>
</table>
Inference Rules

Deriving Theorems from Axioms

Modus Ponens

A well-known rule of inference
There are many others

\[ \Theta \Rightarrow \Psi \]

Are there entailed WFFs that Modus Ponens cannot derive?

Yes

Modus Ponens is incomplete
Inference Rules

Deriving Theorems from Axioms

A different but related inference rule

\[ \neg \Theta \lor \Psi \]

[\Theta]

[\Psi]
Deduction

Putting it together

All men are mortal
Socrates is a man

∀x (Man(x) ⇒ Mortal(x))
Man(Socrates)

Mortal(Socrates)

by modus ponens

Θ ⇒ Ψ

Θ

Ψ
Deduction

Putting it together

All men are mortal \( \forall x (\text{Man}(x) \Rightarrow \text{Mortal}(x)) \)
Socrates is a man \( \text{Man}(\text{Socrates}) \)
George is a man \( \text{Man}(\text{George}) \)

How many mortals can we infer?
Deduction

Putting it together

All men are mortal
Socrates is a man
George is a man
All men have fathers
All fathers are men

\( \forall x (\text{Man}(x) \Rightarrow \text{Mortal}(x)) \)
\( \text{Man}(\text{Socrates}) \)
\( \text{Man}(\text{George}) \)
\( \forall x (\text{Man}(x) \Rightarrow \text{Father}(\text{F-of}(x))) \)
\( \forall x (\text{Father}(x) \Rightarrow \text{Man}(x)) \)

Now how many mortals do we have?

A small set of axioms can lead to an infinite world
More Inference Rules
AND, OR / Elimination, Introduction

\[ \top \land \top \rightarrow \top \]
\[ \top \lor \top \rightarrow \top \]

Double Negation: \[ \top \Rightarrow \top \]
Resolution

$\alpha \lor \beta \quad \neg \alpha \Rightarrow \beta$

$\neg \beta \lor \gamma \quad \beta \Rightarrow \gamma$

$\alpha \lor \gamma \quad \neg \alpha \Rightarrow \gamma$
FOL Difficulties

• Applies best to satisfiability problems
• Representing uncertainty is awkward
  – variable confidence
  – need measurable quantities
• Dealing with Ignorance is awkward
  – non-monotonic logic
  – “can you start your car?”
• Qualification problem
  – real world needs an infinite number of qualifiers
  – “Birds Fly”
Logic Lecture 3 Ends Here