Artificial Intelligence

Logic and Inference

MONTANA STATE UNIVERSITY

Reasoning Agents

- At the heart of our agents we find:
  - A collection of facts (stored or perceived).
  - A collection of relationships between facts (in the form of logical expressions).
  - A strategy for deriving new knowledge from existing or perceived knowledge.
- Issues needed to be addressed for our agents:
  - How do we represent the knowledge (facts, and relationships between facts)?
  - How do we reason with the knowledge?

Logic Motivation

- One of the original problem areas in AI was mathematical theorem proving: Logic Theorist, GPS.
- First complete inference procedure was computational.
- Early on many researchers realized that it was essential to have a “formal” language for talking about knowledge.
- Logic seems like the obvious language.
- Logic and logical inference are generally viewed as essential to symbolic AI.

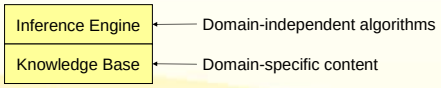
Reasoning Gone Wrong

The Quest for the Holy Grail

MONTANA STATE UNIVERSITY

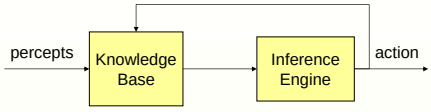
Knowledge Bases

- A knowledge base is a set of sentences in a formal language.
- Knowledge bases assume a *declarative* approach to building an agent.
  - TELL the agent what it needs to know.
  - Then the agent ASKs itself what to do—answers follow from the knowledge base.



MONTANA STATE UNIVERSITY

Knowledge-Based Agent



- Knowledge or epistemological level
  "What the agent can know about"
- Logical level
  representation of facts as sentences in a formal language
- Implementation level
  how inference is implemented on a physical representation

This is, broadly taken, the science of *knowledge representation*

This is also often called the "declarative approach."

MONTANA STATE UNIVERSITY

Simple Knowledge-Based Agent

```
function KB-Agent(percept) returns action
    static  KB,
            t = 0;

    TELL(KB,MAKE-PERCEPT-SENTENCE(percept,t));
    action ← ASK(KB,MAKE-ACTION-QUERY(t));
    TELL(KB,MAKE-PERCEPT-SENTENCE(action,t));
    t ← t + 1;
    return action;
```

- An agent must be able to:
  - Represent states, actions, etc.
  - Incorporate new percepts.
  - Update internal representations of the world.
  - Deduce hidden properties of the world.
  - Deduce appropriate actions.

MONTANA STATE UNIVERSITY

Mountains Minds

Logic

- A *language* for describing states of affairs, along with procedures that operate on sets of sentences to perform reasoning
- Formally, includes
  - Syntax—symbols of the language and how the symbols can be combined
  - Semantics—rules for interpreting symbols and sentences in the language

MONTANA STATE UNIVERSITY

Mountains Minds

Types of Logics

- Logics are characterized by what they commit to as "primitives."
- Ontological commitment: what exists? fact? objects? time? beliefs?
- Epistemological commitment: what are the possible states of knowledge?

Language	Ontological Commitment	Epistemological Commitment
Propositional Logic	facts	true/false/unknown
First-Order Logic	facts, objects, relations	true/false/unknown
Temporal Logic	facts, objects, relations,time	true/false/unknown
Probability	facts	degree of belief 0 … 1
Fuzzy logic	degree of truth	degree of belief 0 … 1

MONTANA STATE UNIVERSITY

Mountains Minds

Logic: The Landscape

- Logics consists of two basic pieces:
  - **syntax** -– what is written down; the language
  - **semantics** –- the meaning of the language; what it says about the world –- the definition of what it means to be **true**
- Two notions of determining what follows from what we know
  - $KB \models a$ → *KB entails a*; if *KB* is true, then so is *a*
  - $KB \vdash_i a$ → inference procedure *i derives a* from the sentences in *KB*

MONTANA STATE UNIVERSITY
Mountains Minds
10

Logic: The Landscape

- Entailment
  - A sentence follows logically from another sentence.
  - Denoted $\alpha \models \beta$
  - Means that sentence α entails sentence β.
- Informally, we say that the truth of β is contained in the truth of α.
- Formally,
  - ***Def:*** A *model* is a mathematical abstraction that fixes the truth or falsity of every relevant sentence.
  - ***Def:*** $\alpha \models \beta$ if and only if, in every model in which α is true, β must also be true.

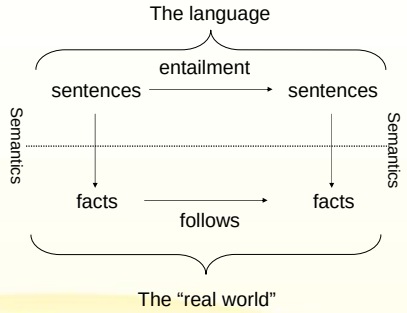
MONTANA STATE UNIVERSITY
Mountains Minds
11

Logic: The Landscape

- We can make up any way of creating new sentences from old
  - *i* is *sound* if $KB \vdash_i a$ implies that $KB \models a$
    - We only derive statements that are true given what we know
    - The record of the derivation is called a *proof*
  - *i* is *complete* if $KB \models a$ implies that $KB \vdash_i a$
    - If a fact is true, we can derive it
- Soundness is essential (and usually easy)
- Completeness is hard (and sometimes impossible!)

MONTANA STATE UNIVERSITY
Mountains Minds
12

Logic: The Landscape



MONTANA STATE UNIVERSITY

13

Mountains & Minds

Semantics

- Suppose that *S* is the set of all sentences in the language (we have to define this)
- Suppose that *I* is a function that maps every sentence of *S* to some *interpretation* of that sentence: *M* is a *model* that determines the basic truth values of the primitives:
  e.g $I[M]: S \rightarrow$ {True,False}
- We will assume all of our languages are *compositional*: that is we can recursively define the meaning of sentences in terms of their components

MONTANA STATE UNIVERSITY

14

Mountains & Minds

Truth in Logic

- **Truth**: A sentence is *true* if the state of affairs it describes is actually the case.
  - A sentence is *valid* if it is always true (also called a *tautology*)
  - A sentence is *unsatisfiable* if it is never true (also called a *contradiction*)
  - A sentence is *satisfiable* if it is possible for it to be true.

MONTANA STATE UNIVERSITY

15

Mountains & Minds

Propositional Logic

- The simplest formal logic.
- Largely based on set theory.
- Forms the foundation for most other logics.
- Will present syntax and semantics.

MONTANA STATE UNIVERSITY
Mountains Minds
16

Syntax of Propositional Logic

- The constants T and F are sentences.
- A propositional symbol (e.g., *P*, *Q*, *R*) is a sentence.
- If α is a sentence and β is a sentence, then the following are sentences.
  - (α)
  - $\neg\alpha$
  - $\alpha \vee \beta$

MONTANA STATE UNIVERSITY
Mountains Minds
17

Propositional Logic Semantics

- Logical constants T and F have fixed interpretations (i.e., True and False respectively).
- If sentence α is True, then so is (α).
- If sentence α if False, then $\neg\alpha$ is True.
- If either α or β is True, then $\alpha \vee \beta$ is True.

MONTANA STATE UNIVERSITY
Mountains Minds
18

Truth Tables

α	$\neg \alpha$
T	F
F	T

α	β	$\alpha \vee \beta$
T	T	T
T	F	T
F	T	T
F	F	F

MONTANA STATE UNIVERSITY
Mountains Minds
19

More Truth Tables

α	β	$\alpha \wedge \beta$
T	T	T
T	F	F
F	T	F
F	F	F

α	β	$\alpha \Rightarrow \beta$
T	T	T
T	F	F
F	T	T
F	F	T

α	β	$\alpha \Leftrightarrow \beta$
T	T	T
T	F	F
F	T	F
F	F	T

MONTANA STATE UNIVERSITY
Mountains Minds
20

Derived Sentences

- All sentences can be derived from sentences involving the operators $\neg$ and $\vee$.
- $\alpha \wedge \beta \equiv \neg(\neg\alpha \vee \neg\beta)$
- $\alpha \Rightarrow \beta \equiv \neg\alpha \vee \beta$
- $\alpha \Leftrightarrow \beta \equiv \neg(\neg(\neg\alpha \vee \beta) \vee \neg(\neg\beta \vee \alpha))$

MONTANA STATE UNIVERSITY
Mountains Minds
21

Interpretations

- Negation (NOT): $\neg\alpha$
- Disjunction (OR): $\alpha \vee \beta$
- Conjunction (AND): $\alpha \wedge \beta$
- Implication (IF…THEN…): $\alpha \Rightarrow \beta$
- Equivalence: $\alpha \Leftrightarrow \beta$

MONTANA STATE UNIVERSITY

Mountains Minds

Proof by Truth Tables

- Given: $P_1 \vee P_2$ and $\neg P_2$
- Prove P_1

P_1	P_2	$P_1 \vee P_2$	$\neg P_2$	$(P_1 \vee P_2) \wedge \neg P_2$	Ψ	$\Psi \Rightarrow P_1$
T	T	T	F	F		**T**
T	F	T	T	T		**T**
F	T	T	F	F		**T**
F	F	F	T	F		**T**

Following setup as implication, all rows true verifies.

MONTANA STATE UNIVERSITY

Mountains Minds

Normal Forms

- Conjunctive Normal Form (CNF)
  - A conjunction of disjunctions (clauses) of literals.
  - $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$
- Disjunctive Normal Form (DNF)
  - A disjunction of conjunctions (terms) of literals.
  - $(A \wedge B) \vee (A \wedge \neg C) \vee (A \wedge \neg D) \vee (\neg B \wedge \neg C) \vee (\neg B \wedge \neg D)$
- Horn Form
  - A conjunction of Horn clauses (clauses with ≤ 1 positive literal)
  - $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$
  - Often written as a set of implications
  - $B \Rightarrow A$ and $(C \wedge D) \Rightarrow B$

MONTANA STATE UNIVERSITY

Mountains Minds

Propositional Inference

- Modus Ponens: $[\alpha \Rightarrow \beta, \alpha] \rightarrow \beta$
- Modus Tollens: $[\alpha \Rightarrow \beta, \neg\beta] \rightarrow \neg\alpha$
- AND-Elim: $[\alpha_1 \wedge \ldots \wedge \alpha_n] \rightarrow \alpha_i$
- AND-Intro: $[\alpha_1, \ldots, \alpha_n] \rightarrow [\alpha_1 \wedge \ldots \wedge \alpha_n]$
- OR-Intro: $[\alpha_i] \rightarrow [\alpha_1 \vee \ldots \vee \alpha_i \vee \ldots \vee \alpha_n]$
- Double-Negation: $[\neg\neg\alpha] \rightarrow \alpha$
- Unit Resolution: $[\alpha \vee \beta, \neg\beta] \rightarrow \alpha$
- Resolution: $[\alpha \vee \beta, \neg\beta \vee \delta] \rightarrow \alpha \vee \delta$
- Subsumption: $[(\alpha \wedge \beta) \vee \beta] \rightarrow \beta$

Example Proof (1)

- Given $(P \Rightarrow Q) \wedge (\neg P \Rightarrow Q)$
- Prove Q
- Proof:
  - $(\neg P \vee Q) \wedge (\neg\neg P \vee Q)$, Def. Implication
  - $(\neg P \vee Q) \wedge (P \vee Q)$, Double Negation
  - $((\neg P \vee Q) \wedge P) \vee ((\neg P \vee Q) \wedge Q)$, Distribution
  - $(\neg P \wedge P) \vee (Q \wedge P) \vee (\neg P \wedge Q) \vee (Q \wedge Q)$, Distribution
  - $(Q \wedge P) \vee (\neg P \wedge Q) \vee Q$, Tautology
  - $(Q \wedge P) \vee Q$, Subsumption
  - Q, Subsumption Q.E.D.

Example Proof (2)

- Given $(P \Rightarrow Q)$ and $(Q \Rightarrow R)$
- Prove $((P \Rightarrow \neg R) \Rightarrow \neg P)$
- Proof:
  - $(P \Rightarrow Q) \wedge (Q \Rightarrow R)$, AND Intro.
  - $(\neg P \vee Q) \wedge (\neg Q \vee R)$, Def. Implication
  - $(\neg P \vee R)$, Resolution
  - $(P \vee \neg P) \wedge (R \vee \neg P)$, AND Intro. w/ Tautology
  - $((P \wedge R) \vee \neg P)$, Distribution of Terms
  - $(\neg(\neg P \vee \neg R) \vee \neg P)$, DeMorgan's
  - $(\neg(P \Rightarrow \neg R) \vee \neg P)$, Def. Implication
  - $((P \Rightarrow \neg R) \Rightarrow \neg P)$, Def. Implication Q.E.D.

Example Proof (3)

- Given:
  - If the weather is warm and the sky is clear, then either we go swimming or we go boating, AND
  - It is not the case that if the sky is clear, then we go swimming.
- Prove:
  - If we do not go boating, then the weather is not warm.

MONTANA STATE UNIVERSITY

Mountains Minds

Example Proof (3)

- Convert sentence to propositional symbols.
  - W = the weather is warm
  - C = the sky is clear
  - S = we go swimming
  - B = we go boating
- Given:
  - $(W \wedge C) \Rightarrow (S \vee B)$
  - $\neg(C \Rightarrow S)$
- Prove
  - $(\neg B \Rightarrow \neg W)$

MONTANA STATE UNIVERSITY

Mountains Minds

Example Proof (3)

- Proof:
  - $\neg(W \wedge C) \vee (S \vee B)$, Def. Implication
  - $(\neg W \vee \neg C) \vee (S \vee B)$, DeMorgan's
  - $(\neg W \vee B) \vee (\neg C \vee S)$, Regrouping
  - $\neg(\neg C \vee S)$, Def. Implication ($\alpha = (\neg C \vee S)$)
  - $(\neg W \vee B)$, Resolution
  - $(W \Rightarrow B)$, Def. Implication
  - $(\neg B \Rightarrow \neg W)$, Contrapositive of Implication Q.E.D.

MONTANA STATE UNIVERSITY

Mountains Minds

Wumpus World

- $h \times h$ grid world
- Wumpus lives in one cell and eats all creatures
- Some cells have bottomless pits
- Gold exists in r cells.
- We will treat this as an agent problem and "solve" using propositional logic.

MONTANA STATE UNIVERSITY
Mountains Minds
31

Percepts

- Smell wumpus in adjacent cells
- Feel breeze from bottomless pit in adjacent cells
- See glitter in cell with gold
- If agent hits a wall, feels a bump
- Hear scream everywhere if wumpus dies

MONTANA STATE UNIVERSITY
Mountains Minds
32

Actions

- Go forward, turn right 90^o, turn left 90^o
- Pick up object in same cell
- Shoot in straight line (only one arrow)
- Climb (to leave cave from start square)
- Die (if encounter wumpus or pit)

MONTANA STATE UNIVERSITY
Mountains Minds
33

Goal

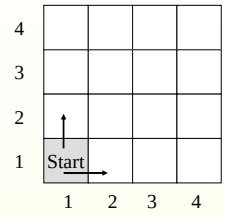
- Find the gold and exit the cave as quickly as possible.
- Gain 1000 points for climbing out of the cave with gold.
- Lose 1 point for each action taken.
- Lose 10,000 points if die.

MONTANA STATE UNIVERSITY

Mountains Minds

The Wumpus World

Stench	F
Breeze	F
Glitter	F
Bump	F
Scream	F



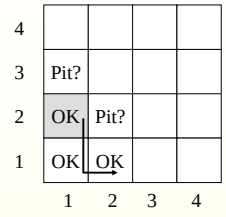
[1,2] and [2,1] are both safe.

MONTANA STATE UNIVERSITY

Mountains Minds

The Wumpus World

Stench	F
Breeze	**T**
Glitter	F
Bump	F
Scream	F



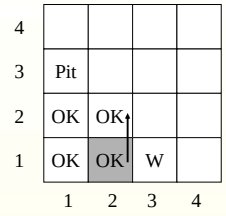
Move to [1,2] to eliminate Pit at [2,2]

MONTANA STATE UNIVERSITY

Mountains Minds

The Wumpus World

Stench	T
Breeze	F
Glitter	F
Bump	F
Scream	F



Wumpus must be in [1,1] (no), [2,2] (no), or [1,3]
Pit cannot be in [2,2] since no longer feel breeze.

Using Logic in Wumpus World

- Let $S_{i,j}$ indicate prop *stench* in cell [*i*,*j*].
- Let $B_{i,j}$ indicate prop *breeze* in cell [*i*,*j*]
- We know $\neg S_{1,1}$, $\neg S_{2,1}$, $S_{1,2}$, $\neg B_{1,1}$, $B_{2,1}$, $\neg B_{1,2}$
- We have several inference rules
  - $\neg S_{1,1} \Rightarrow \neg W_{1,1} \wedge \neg W_{1,2} \wedge \neg W_{2,1}$
  - $\neg S_{2,1} \Rightarrow \neg W_{1,1} \wedge \neg W_{2,1} \wedge \neg W_{2,2} \wedge \neg W_{3,1}$
  - $\neg S_{1,2} \Rightarrow \neg W_{1,1} \wedge \neg W_{1,2} \wedge \neg W_{2,2} \wedge \neg W_{1,3}$
  - $S_{1,2} \Rightarrow W_{1,3} \vee W_{1,2} \vee W_{2,2} \vee W_{1,1}$

Finding the Wumpus

- $\neg S_{1,1}$ yields $\neg W_{1,1} \wedge \neg W_{1,2} \wedge \neg W_{2,1}$ (MP)
- which yields $\neg W_{1,1}$, $\neg W_{1,2}$, $\neg W_{2,1}$ (AE)
- $\neg S_{2,1}$ yields $\neg W_{1,1}$, $\neg W_{2,1}$, $\neg W_{2,2}$, $\neg W_{3,1}$ (MP, AE)
- $S_{1,2}$ yields $W_{1,3} \vee W_{1,2} \vee W_{2,2} \vee W_{1,1}$ (MP)
- which yields $W_{1,3} \vee W_{1,2} \vee W_{2,2}$ (Res)
- which yields $W_{1,3} \vee W_{1,2}$ (Res)
- which yields $W_{1,3}$ (Res) — Q.E.D.

Limitations

- Propositional logic is not sufficiently powerful for general inference systems.
  - Cannot represent "quantifiers"
  - Cannot represent variables
  - Cannot represent functional information
  - Cannot represent uncertainty
- Next, we will consider a more expressive logic formalism—first-order logic.

First-Order Logic

- Also called Predicate Calculus
- Extend language beyond propositions
  - Constants (individuals in the world)—*KingJohn*, 2, *MSU,* …
  - Functions (map individuals to individuals)—*Sqrt*, *LeftLegOf*, …
  - Predicates (map individuals to truth values)—*Brother*, >, ==, …
  - Variables (e.g., *x*, *y*)
  - Connectives ($\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$)
  - Quantifiers (universal "$\forall$" and existential "$\exists$")
  - Equality (i.e., =)

Atomic Sentences

- Atomic sentence <=> *predicate*($term_1$,…,$term_n$) |
  $term_1 = term_2$
- *term* <=> *function*($term_1$,…,$term_n$) |
  constant | *variable*
- Examples:
  - *Brother*(*KingJohn*, *RichardTheLionheart*)
  - *GT*(*Length*(*LeftLegOf*(*Richard*)),
    Length(*LeftLegOf*(*KingJohn*)))

Complex Sentences

- Made up from atomic sentences using connectives.
  - $\neg S$, $S_1 \wedge S_2$, $S_1 \vee S_2$, $S_1 \Rightarrow S_2$, $S_1 \Leftrightarrow S_2$
- Examples:
  - *Sibling*(*KingJohn*,*Richard*) $\Rightarrow$ *Sibling*(*Richard*,*KingJohn*)
  - GT(1,2) $\vee$ LE(1,2)
  - GT(1,2) $\wedge \neg$GT(1,2)

MONTANA STATE UNIVERSITY Mountains & Minds

Truth in First-Order Logic

- Sentences are true with respect to a model and interpretation.
- Models contain objects and relations among them.
- Interpretation specifies referents for
  - *constant symbols* $\rightarrow$ objects
  - *predicate symbols* $\rightarrow$ relations
  - *function symbols* $\rightarrow$ function relations
- An atomic sentence *predicate*($term_1$,…,$term_n$) is true *iff* the objects referred to by $term_1$,…,$term_n$ are in the relation referred to by *predicate*.

MONTANA STATE UNIVERSITY Mountains & Minds

Quantifiers

- Universals are like conjunction
  - $\forall x\ P(x)$ means P holds for all values x
- Existentials are like disjunction
  - $\exists x\ P(x)$ means P holds for some value x
- Universals usually used with implication
- Existentials usually used with conjunction
- Switching like quantifiers does not change meaning.
- Switching different quantifiers does.

MONTANA STATE UNIVERSITY Mountains & Minds

Universal Quantification

- $\forall\langle variables\rangle\ \langle sentence\rangle$
- Everyone at MSU is smart.
  - $\forall x\ At(x,MSU) \Rightarrow Smart(x)$
- Remember, $\forall x\ P(x)$ is equivalent to the conjunction of *all* instantiations of $P(x)$.
  - $At(KingJohn,MSU) \Rightarrow Smart(KingJohn) \wedge At(Richard,MSU) \Rightarrow Smart(Richard) \wedge \quad At(KingJohn,MSU) \Rightarrow Smart(MSU) \wedge \ldots$
- Typically, $\Rightarrow$ is the main connective with $\forall$.
- Common mistake: using $\wedge$ as the main connective with $\forall$:
  - $\forall x\ At(x,MSU) \wedge Smart(x)$ means "everyone is at MSU and everyone is smart."
  - This is *not* the same!

MONTANA STATE UNIVERSITY

Mountains Minds

Existential Quantification

- $\exists\langle variables\rangle\ \langle sentence\rangle$
- Someone at UM is smart.
  - $\exists x\ At(x,UM) \wedge Smart(x)$
- Remember, $\exists x\ P(x)$ is equivalent to the disjunction of instantiations of $P(x)$.
  - $At(KingJohn,UM) \wedge Smart(KingJohn) \vee At(Richard,UM) \wedge Smart(Richard) \vee At(KingJohn,UM) \wedge Smart(UM) \vee \ldots$
- Typically, $\wedge$ is the main connective with $\exists$.
- Common mistake: using $\Rightarrow$ as the main connective with $\exists$:
  - $\exists x\ At(x,UM) \Rightarrow Smart(x)$ means "if there is any person at UM, then that person is smart."
  - This is *not* the same!

MONTANA STATE UNIVERSITY

Mountains Minds

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 $\forall y\ \exists x$.
  - $\exists x\ \forall y\ Loves(x,y)$ means "There is a person who loves everyone in the world."
  - $\forall y\ \exists x\ Loves(x,y)$ means "Everyone in the world is loved by at least one person."
- Duality of Quantifiers (express each as the other)
  - $\forall x\ Likes(x,IceCream) \equiv \neg\exists x\ \neg Likes(x,IceCream)$
  - $\exists x\ Likes(x,Broccoli) \equiv \neg\forall x\ \neg Likes(x,Broccoli)$

MONTANA STATE UNIVERSITY

Mountains Minds

Inference Rules in FOL

- PL inference rules apply in FOL as well.
- Universal Elim: If $\forall x\ P(x)$ is true, then $P(c)$ is true for any constant c in the domain.
- Existential Intro: If $P(c)$ is true, then $\exists x\ P(x)$ is true.
- Existential Elim: If $\exists x\ P(x)$ is true, then $P(c)$ is true for some constant c not appearing in any other sentence (Skolem constant).

Inference Rules in FOL

- Generalized Modus Ponens
  - Combines AND-Intro, Univeral-Elim, and Modus Ponens
  - Ex. From $P(c)$, $Q(c)$, and $\forall x\ (P(x) \wedge Q(x)) \Rightarrow R(x)$ derive $R(c)$.
  - Let subst(θ,α) denote substitutions resulting from applying substitution list θ to sentence α.
  - Given atomic sentences $P_1,\ldots,P_n$ and implication $(Q_1 \wedge \ldots \wedge Q_n) \Rightarrow R$ and subst(θ,P_i) = subst(θ,Q_i), derive subst(θ,R).

Inference in FOL

- Recall the Generalized Modus Ponens inference rule.
- This rule requires the ability to find substitutions allowing “facts” to match left-hand-sides of rules.
- This matching process is called “unification.”

Unification

- `unify(p,q) = θ`,
  - where `subst(θ,p)= subst(θ,q)`
  - p and q are sentences to be unified.
- Examples:
  - `unify(P(a,x),P(a,b)) = {x/b}`
  - `unify(P(a,x),P(y,b)) = {x/b,y/a}`
  - `unify(P(a,x),P(y,f(y)) = {y/a,x/f(a)}`
  - `unify(P(a,x),P(x,b)) = FAIL`

MONTANA STATE UNIVERSITY Mountains Minds 52

Unification

- To avoid last problem, standardize apart variables.
- Replace first x with x_1 and second x with x_2.
- `unify(P(a,`x_1`),P(`x_2`,b)) =` `{`x_1`/b,`x_2`/a}`
- Not everything can be unified
  - `unify(P(a,b),P(x,x)) = FAIL`
  - `unify(P(a,f(a)),P(x,b)) = FAIL`

MONTANA STATE UNIVERSITY Mountains Minds 53

Unification Algorithm

function UNIFY(*x*, *y*, *θ*) **returns** a substitution making *x* and *y* identical
inputs: *x*, a variable, constant, list, or compound
y, a variable, constant, list, or compound
θ, the substitution built up so far (optional, defaults to empty)

if *θ* = *failure* **then return** *failure*
else if *x* = *y* **then return** *θ*
else if VARIABLE?(*x*) **then return** UNIFY-VAR(*x*, *y*, *θ*)
else if VARIABLE?(*y*) **then return** UNIFY-VAR(*y*, *x*, *θ*)
else if COMPOUND?(*x*) **and** COMPOUND?(*y*) **then**
return UNIFY(ARGS[*x*],ARGS[*y*],UNIFY(OP[*x*],OP[*y*],*θ*))
else if LIST?(*x*) **and** LIST?(*y*) **then**
return UNIFY(REST[*x*],REST[*y*],UNIFY(FIRST[*x*],FIRST[*y*],*θ*))
else return *failure*

MONTANA STATE UNIVERSITY Mountains Minds 54

Unification Algorithm

function UNIFY-VAR(*var*, *x*, θ) **returns** a substitution
  inputs: *var*, a variable
    x, any expression
    θ, the substitution built up so far

  if {*var* / *val*} $\in \theta$ **then return** UNIFY(*val*, *x*, θ)
  else if {*x* / *val*} $\in \theta$ **then return** UNIFY(*var*, *val*, θ)
  else if OCCUR-CHECK?(*var*, *x*) **then return** *failure*
  else return θ + {*var* / *x*}

MONTANA STATE UNIVERSITY
Mountains & Minds
55

Sample Proof

- Given:
  - Bob is a buffalo.
  - Pat is a pig.
  - Buffaloes outrun pigs.
- Prove:
  - Bob outruns Pat.

MONTANA STATE UNIVERSITY
Mountains & Minds
56

Sample Proof

- Convert premises to FOL.
  - *Buffalo*(*Bob*)
  - *Pig*(*Pat*)
  - $\forall x,y$ *Buffalo*(x) $\wedge$ *Pig*(y) $\Rightarrow$ *Faster*(x,y)
- Convert goal to FOL.
  - *Faster*(*Bob*,*Pat*)

MONTANA STATE UNIVERSITY
Mountains & Minds
57

Sample Proof

- Proceed with proof.
  - *Buffalo*(*Bob*) $\wedge$ *Pig*(*Pat*), AND Intro.
  - (*Buffalo*(*Bob*) $\wedge$ *Pig*(*Pat*)) $\Rightarrow$ *Faster*(*Bob*,*Pat*), Universal Elimination {*x*/*Bob*, *y*/*Pat*}
  - *Faster*(*Bob*,*Pat*), Modus Ponens Q.E.D.

MONTANA STATE UNIVERSITY

Mountains Minds

Logical Inference

- Given a set of axioms (i.e., a set of sentences assumed to be true).
- Generate a set of theorems (i.e., a set of sentences inferred to be true from the axioms.
- Two approaches:
  - Forward Chaining
  - Backward Chaining

MONTANA STATE UNIVERSITY

Mountains Minds

Forward Chaining

- From known facts, infer new facts by matching facts to l.h.s. of rules and inferring r.h.s.
- This approach makes use of Modus Ponens.
- Inference process continues by "chaining" through rules until desired conclusions are reached.

MONTANA STATE UNIVERSITY

Mountains Minds

Forward Chaining

function FOL-FC-ASK(KB,α) **returns** a substitution or *false*
  inputs: KB, the knowledge base
    α, the query
  local variables: *new*, new inferred sentences

  repeat until *new* is empty
    new ← { }
    for each sentence r **in** KB **do**
      $(p_1 \wedge \ldots \wedge p_n \Rightarrow q) \leftarrow$ STANDARDIZE-APART(r)
      for each θ such that SUBST(θ, $p_1' \wedge \ldots \wedge p_n'$)
          for some $p_1', \ldots, p_n'$ in KB
        $q' \leftarrow$ SUBST(θ,q)
        if q' is not a renaming of some sentence in KB or *new* **then do**
          add q' to *new*
          $\phi \leftarrow$ UNIFY(q',α)
          if ϕ is not *fail* **then return** ϕ
    add *new* to KB
  return *false*

MONTANA STATE UNIVERSITY
Mountains Minds
61

Example

- Add facts in turn, firing rules as appropriate.
  1. *Buffalo*(*x*) ∧ *Pig*(*y*) ⇒ *Faster*(*x*,*y*)
  2. *Pig*(*y*) ∧ *Slug*(*z*) ⇒ *Faster*(*y*,*z*)
  3. *Faster*(*x*,*y*) ∧ *Faster*(*y*,*z*) ⇒ *Faster*(*x*,*z*)
  4. *Buffalo*(*Bob*)
  5. *Pig*(*Pat*)
     → 6. *Faster*(*Bob*,*Pat*) [1 {*x*/*Bob*,*y*/*Pat*},4,5]
  7. *Slug*(*Steve*)
     → 8. *Faster*(*Pat*,*Steve*) [2 {*y*/*Pat*,*z*/*Steve*},5,7]
        → 9. *Faster*(*Bob*,*Steve*) [3,6,8]

MONTANA STATE UNIVERSITY
Mountains Minds
62

Backward Chaining

- Start with goal conclusion and state as hypothesis (i.e., something assumed to be true).
- Match goal to r.h.s of rules and take l.h.s. as new sub-goal.
- Chain back through rules until known facts are found.

MONTANA STATE UNIVERSITY
Mountains Minds
63

Backward Chaining

function FOL-BC-ASK(*KB*, *goals*,θ) **returns** a set of substitutions
  inputs: *KB*, the knowledge base
    goals, a list of conjuncts forming a query (θ already applied)
    θ, the current substitution, initial empty
  local variables: *answers*, a set of substitutions, initially empty

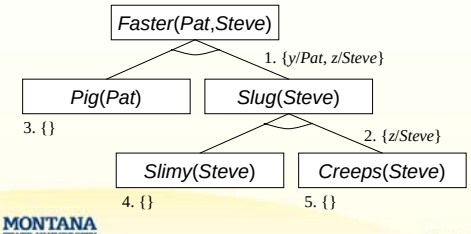
  if *goals* is empty **then return** {θ}
  $q' \leftarrow$ SUBST(θ,FIRST(*goals*))
  for each sentence *r* **in** *KB* where
    STANDARDIZE-APART(*r*) = $(p_i \wedge \ldots \wedge p_n \Rightarrow q)$ and
    $\theta' \leftarrow$ UNIFY(q, q') succeeds
    new_goals $\leftarrow [p_1,\ldots,p_n|\text{REST}(goals)]$
    answers $\leftarrow$ FOL-BC-ASK(*KB*, *new_goals*, COMPOSE(θ',θ))
      $\cup$ *answers*
  return *answers*

MONTANA STATE UNIVERSITY

Mountains Minds

Example

- 1. $Pig(x) \wedge Slug(y) \Rightarrow Faster(x,y)$
- 2. $Slimy(z) \wedge Creeps(z) \Rightarrow Slug(z)$
- 3. $Pig(Pat)$ 4. $Slimy(Steve)$ 5. $Creeps(Steve)$



MONTANA STATE UNIVERSITY

Mountains Minds

Resolution

- An application of a special form of Modus Ponens.
- Can also be considered a form of backward chaining.
- Provides a method for automated theorem proving.
- Assumes all sentences are in conjunctive normal form (a.k.a. clause form)
  - $(p_1 \vee q_1 \vee r_1) \wedge (p_2 \vee q_2 \vee r_2) \wedge (p_3 \vee q_3 \vee r_3)$

MONTANA STATE UNIVERSITY

Mountains Minds

Converting to Clause Form

- Eliminate implications
- Reduce scope of negation symbols
- Standardize variables
- Eliminate existential quantifiers
- Move universal quantifiers left
- Rewrite in conjunctive normal form
- Eliminate universal quantifiers
- Separate into clauses
- Rename variables

MONTANA STATE UNIVERSITY
Mountains & Minds
67

A Detailed Example

- Convert the following to clause form:
  - $\forall x\,[P(x) \Rightarrow [\forall y\,[P(y) \Rightarrow P(f(x,y))] \wedge \neg\forall y\,[Q(x,y) \Rightarrow P(y)]]]$
  - where P and Q are predicates, x and y are variables, and f is a function

MONTANA STATE UNIVERSITY
Mountains & Minds
68

Step 1

- Eliminate implications
- Recall $x \Rightarrow y$ is the same as $\neg x \vee y$
- Conversion yields
  - $\forall x\,[\neg P(x) \vee [\forall y\,[\neg P(y) \vee P(f(x,y))] \wedge \neg\forall y\,[\neg Q(x,y) \vee P(y)]]]$

MONTANA STATE UNIVERSITY
Mountains & Minds
69

Step 2

- Reduce scope of negation symbols
- DeMorgan's laws
  - $\neg(X \wedge Y) \equiv \neg X \vee \neg Y$
  - $\neg(X \vee Y) \equiv \neg X \wedge \neg Y$
- Double negation: $\neg(\neg X) \equiv X$
- Duality of quantifiers
  - $\neg\forall x\, P(x) \equiv \exists x\, [\neg P(x)]$
  - $\neg\exists x\, P(x) \equiv \forall x\, [\neg P(x)]$

MONTANA STATE UNIVERSITY

Mountains Minds

Step 2 (cont.)

- Only part of sentence affected:
  - $\neg\forall y\, [\neg Q(x,y) \vee P(y)]$
- Apply duality of quantifiers:
  - $\exists y\, [\neg[\neg Q(x,y) \vee P(y)]]$
- Apply DeMorgan's:
  - $\exists y\, [\neg[\neg Q(x,y)] \wedge \neg P(y)]$
- Apply double negation:
  - $\exists y\, [Q(x,y) \wedge \neg P(y)]$

MONTANA STATE UNIVERSITY

Mountains Minds

Step 3

- Standardize variables
- Results in renaming variables associated with each quantifier to be unique.
- Conversion yields
  - $\forall x\, [\neg P(x) \vee [\forall y\, [\neg P(y) \vee P(f(x,y))] \wedge \exists w\, [Q(x,w) \wedge \neg P(w)]]]$
- Be careful of scope of quantifiers!!

MONTANA STATE UNIVERSITY

Mountains Minds

Step 4

- Eliminate existential quantifiers
- Skolem constants
  - Replace existentially quantified variables by a unique constant.
- Skolem functions
  - If existentially quantified variable is within scope of a universal quantifier too, replace variables with a function of universally quantified variable.

MONTANA STATE UNIVERSITY
Mountains Minds
73

Step 4 (cont.)

- Conversion yields
  - $\forall x\ [\neg P(x) \vee [\forall y\ [\neg P(y) \vee P(f(x,y))] \wedge [Q(x,g(x)) \wedge \neg P(g(x))]]]$
- Note the Skolemization did not include $\forall y$ since the brackets limit the scope of this quantifier.

MONTANA STATE UNIVERSITY
Mountains Minds
74

Step 5

- Move all univeral quantifiers left.
- This is called *prenex* form.
- This is allowed because all variables are unique to each universal quantifier.
- Conversion yields
  - $\forall x\ \forall y\ [\neg P(x) \vee [\neg P(y) \vee P(f(x,y))] \wedge [Q(x,g(x)) \wedge \neg P(g(x))]]$

MONTANA STATE UNIVERSITY
Mountains Minds
75

Step 6

- Rewrite in Conjunctive Normal Form (CNF)
- Uses distributive law twice:
  - $A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$
- Conversion yields
  - $\forall x\ \forall y\ [[\neg P(x) \vee \neg P(y) \vee P(f(x,y))] \wedge [\neg P(x) \vee Q(x,g(x))] \wedge [\neg P(x) \vee \neg P(g(x))]]$

MONTANA STATE UNIVERSITY Mountains Minds

Step 7

- Eliminate universal quantifiers
- From here on, all variables are *assumed* to be universally quantified.
- Conversion yields
  - $[[\neg P(x) \vee \neg P(y) \vee P(f(x,y))] \wedge [\neg P(x) \vee Q(x,g(x))] \wedge [\neg P(x) \vee \neg P(g(x))]]$

MONTANA STATE UNIVERSITY Mountains Minds

Step 8

- Separate into clauses
- This amounts to AND-elimination
- Conversion yields
  - $\neg P(x) \vee \neg P(y) \vee P(f(x,y))$
  - $\neg P(x) \vee Q(x,g(x))$
  - $\neg P(x) \vee \neg P(g(x))$

MONTANA STATE UNIVERSITY Mountains Minds

Step 9

- Rename variables
- No variable symbol should appear in more than one clause
- Conversion yields
  - $\neg P(x_1) \vee \neg P(y) \vee P(f(x_1,y))$
  - $\neg P(x_2) \vee Q(x_2,g(x_2))$
  - $\neg P(x_3) \vee \neg P(g(x_3))$

MONTANA STATE UNIVERSITY
Mountains Minds
79

Resolution Theorem Proving

- Proof by contradiction
- Unification used to match terms in clauses
- Procedure:
  - Negate the theorem to be proven
  - Add to axiom set
  - Convert all axioms to clause form
  - Resolve clauses until either the empty clause is produced or no resolvable clauses remain
  - Empty clause proves theorem (contradiction)

MONTANA STATE UNIVERSITY
Mountains Minds
80

Detailed Example

- Given
  - Whoever can read is literate.
  - Dolphins are not literate.
  - Some dolphins are intelligent.
- Prove
  - Some who are intelligent cannot read.

MONTANA STATE UNIVERSITY
Mountains Minds
81

Convert to FOL

- Given
  - $\forall x\ [Read(x) \Rightarrow Literate(x)]$
  - $\forall x\ [Dolphin(x) \Rightarrow \neg Literate(x)]$
  - $\exists x\ [Dolphin(x) \wedge Intelligent(x)]$
- Prove
  - $\exists x\ [Intelligent(x) \wedge \neg Read(x)]$

MONTANA STATE UNIVERSITY

Mountains Minds

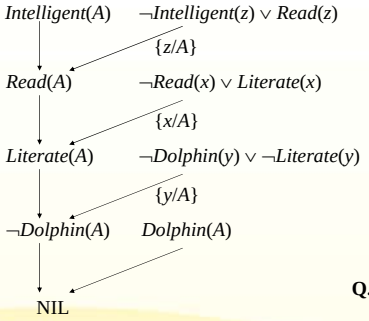
Convert to Clause Form

- Axioms
  - $\neg Read(x) \vee Literate(x)$
  - $\neg Dolphin(y) \vee \neg Literate(y)$
  - $Dolphin(A)$ {A is a Skolem constant)
  - $Intelligent(A)$
- Negated Theorem
  - $\neg\exists x\ [Intelligent(x) \wedge \neg Read(x)]$ which yields in clause form
  - $\neg Intelligent(z) \vee Read(z)$

MONTANA STATE UNIVERSITY

Mountains Minds

Resolution Proof Tree



Q.E.D.

MONTANA STATE UNIVERSITY

Mountains Minds

Resolution Search Strategies

- Unit Preference
  - Always try to resolve with single literals
- Set of support
  - Start with negated query and only resolve against descendents of that query
- Input Resolution
  - Every resolution combines an input sentence (KB or query) with some other sentence
  - Linear resolution is a slight generalization
- Subsumption
  - Only keep the most general set of sentences around

MONTANA STATE UNIVERSITY

Mountains Minds

Why is Resolution Complete

- Completeness says, "if it is true, we can derive it."
- Any set of sentences can be put into an equivalent clausal form
- Assume S is in clausal form and unsatisfiable
- Some set S' of ground clauses generated from S is unsatisfiable
- Resolution can find the contradiction in S'
- By a "lifting lemma," we can take this ground proof and put back the variables to get the refutation of the original sentences

MONTANA STATE UNIVERSITY

Mountains Minds

Translating English to FOL

- Every gardener likes the sun.
  - $\forall x$ gardener(x) $\Rightarrow$ likes(x,Sun)
- You can fool some of the people all of the time.
  - $\exists x \; \forall t$ (person(x) $\wedge$ time(t)) $\Rightarrow$ can-fool(x,t)
- You can fool all of the people some of the time.
  - $\forall x \; \exists t$ (person(x) $\wedge$ time(t)) $\Rightarrow$ can-fool(x,t)
- No purple mushrooms are poisonous.
  - $\neg\exists x$ purple(x) $\wedge$ mushroom(x) $\wedge$ poisonous(x)
  - $\forall x$ purple(x) $\wedge$ mushroom(x) $\Rightarrow \neg$poisonous(x)

MONTANA STATE UNIVERSITY

Mountains Minds

Translating English to FOL

- There are exactly two purple mushrooms.
  - $\exists x\ \exists y$ mushroom(x) $\wedge$ purple(x) $\wedge$ mushroom(y) $\wedge$ purple(y) $\wedge \neg(x=y) \wedge \forall z$ mushroom (z) $\wedge$ purple(z) $\Rightarrow (x=z) \vee (y=z)$
- Deb is not tall.
  - $\neg$tall(Deb)
- X is above Y if X is on top of Y or else there is a pile of one or more objects on top of one another starting with X and ending with Y.
  - $\forall x\ \forall y$ above(x,y) $\Leftrightarrow$ (on(x,y) $\vee\ \exists z$ (on(x,z) $\wedge$ above(z,y)))

MONTANA STATE UNIVERSITY Mountains & Minds

88

Equality

- $term_1 = term_2$ is true under a given interpretation *iff* $term_1$ and $term_2$ refer to the same object.
- Example:
  - 2 = 2 is valid
  - $\forall x$ *Mult*(*Sqrt*(x),*Sqrt*(x)) = x is satisfiable
- Definition of *Sibling*
  - $\forall x,y$ *Sibling*(x,y) $\Leftrightarrow$ [$\neg(x = y) \wedge \exists m,f\ \neg(m = f) \wedge$ *Parent*(m,x) $\wedge$ *Parent*(f,x) $\wedge$ *Parent*(m,y) $\wedge$ *Parent*(f,y)]

MONTANA STATE UNIVERSITY Mountains & Minds

89

Axioms

- Axioms in FOL are logical expressions capturing the basic facts about a domain.
- Axioms are assumed to be true.
- An *independent* axiom is one that cannot be derived from other axioms in the domain.
- Logical expressions that follow from the axioms are called *theorems*.
- An axiom of the form, $\forall x,y\ P(x,y) \Leftrightarrow \ldots$ is often called a *definition* of P since it defines exactly for what objects P does and does not hold.

MONTANA STATE UNIVERSITY Mountains & Minds

90

Defining Axioms

- We will define several axioms covering the domain of set theory.
- Assume the following:
  - *EmptySet* is a constant.
  - *Member* and *Subset* are predicates.
  - *Intersection*, *Union*, and *Adjoin* are functions.
    - (Note: *Adjoin* is a function that adds one element to a set.)
  - *Set* is a predicate that is true only of sets.

MONTANA STATE UNIVERSITY

Mountains Minds

Defining Axioms

- "The only sets are the empty set and those made by adjoining something to a set."
  - $\forall s\ Set(s) \Leftrightarrow (s = EmptySet) \vee (\exists x, s_2\ Set(s_2) \wedge s = Adjoin(x, s_2))$
- "The empty set has no elements adjoined to it (i.e., the empty set cannot be decomposed)."
  - $\neg\exists x, s\ Adjoin(x, s) = EmptySet$
- "Adjoining an element already in the set has no effect."
  - $\forall x, s\ Member(x, s) \Leftrightarrow s = Adjoin(x, s)$
- "The only members of a set are the elements adjoined to it."
  - $\forall x, s\ Member(x, s) \Leftrightarrow \exists y, s_2\ (s = Adjoin(y, s_2) \wedge (x = y \vee Member(x, s_2)))$

MONTANA STATE UNIVERSITY

Mountains Minds

Defining Axioms

- "A set is a subset of another *iff* all of the first set's members are members of the second set."
  - $\forall s_1, s_2\ Subset(s_1, s_2) \Leftrightarrow (\forall x\ Member(x, s_1) \Rightarrow Member(x, s_2))$
- "Two sets are equal *iff* each is a subset of the other."
  - $\forall s_1, s_2\ s_1 = s_2 \Leftrightarrow (Subset(s_1, s_2) \wedge Subset(s_2, s_1))$
- "An object is a member of the intersection of two sets *iff* it is a member of each of the sets."
  - $\forall x, s_1, s_2\ Member(x, Intersection(s_1, s_2)) \Leftrightarrow Member(x, s_1) \wedge Member(x, s_2)$
- "An object is a member of the union of two sets *iff* it is a member of either set."
  - $\forall x, s_1, s_2\ Member(x, Union(s_1, s_2)) \Leftrightarrow Member(x, s_1) \vee Member(x, s_2)$

MONTANA STATE UNIVERSITY

Mountains Minds

Generic Logic-Based Agents

- Generic agent:
  - Records current percept as *fact*
  - Infers best action using *inference*
  - Records *fact* that it will perform best action
  - Performs the action as *function*
- Predicates must be indexed by time.
- Percepts are tested using predicates at appropriate time.

Reflex Agents

- Maps percepts to actions
  - s = stench, b = breeze g = glitter, u = bump, c = scream, and t = time.
  - $\forall s,b,u,c,t\ percept([s,b,Glitter,u,c],t) \Rightarrow action(Grab,t)$
- Requires a rule for every possible state.
- Can reduce rules by introducing rules for perception, but this adds internal state.
- Reflex agent must add states for having gold to decide when to leave.
- Cannot avoid infinite loops — no history!

Model-Based Agents

- Develop internal model of world as you go (internal state).
  - Record whether or not have gold.
  - Record whether or not a cell was previously visited.
- *Situation Calculus* can be used to represent changing world.

Keeping Track of Change

- Facts hold in *situations* rather than eternally.
  - E.g., *Holding*(*Gold*,*Now*) rather than just *Holding*(*Gold*)
- The *situation calculus* is one way to represent change in first-order logic.
  - Adds a situation argument to each "non-eternal" predicate, such as *Now* in *Holding* above.
- Situations are connected by a *Result* function.
  - *Result*(*a*,*s*) is the situation that results from performing action *a* in situation *s*.

Situation Calculus

- Situations are states at particular time
- Actions map situations to situations.
  - $result(walk\text{-}forward, s_i) = s_j$
- Describe actions in FOL in terms of effects.
  - $\forall s\ at(blackboard, s) \Rightarrow at(table, result(walk\text{-}forward, s))$
- Must state all things that do *not* change.
  - $\forall s\ wearing(gr\text{-}shirt, s) \Rightarrow wearing(gr\text{-}shirt, result(walk\text{-}forward, s))$

Describing Actions

- An "effect axiom" describes changes due to action.
  - $\forall s\ AtGold(s) \Rightarrow Holding(Gold, Result(Grab, s))$
- A "frame axiom" describes non-changes due to actions.
  - $\forall s\ HaveArrow(s) \Rightarrow HaveArrow(Result(Grab, s))$

The Yale Shooter Problem

- This is due to Hanks & McDermott (1986).
- Assume the following:
  - Living creatures remaining alive.
  - Loaded guns remain loaded.
- Assume the following sequence of events.
  - Fred is alive.
  - A gun is loaded.
  - The gun is fired at Fred.
- Is Fred alive or dead?
- Is the gun loaded or unloaded?

The Frame Problem

- Need to find an elegant way to handle non-change.
- Representation should avoid frame axioms (too many of them).
- Inference should avoid repeated "copy over" to keep track of state.
- Qualification Problem: True descriptions of real actions require endless caveats—what if gold is slippery, or nailed down, or …?
- Ramification Problem: Real actions have many secondary consequences—what about the dust on the gold, wear and tear on gloves, …?

Successor State Axioms

- Successor state axioms solve the "representational" frame problem (i.e., the problem of specifying large numbers of frame axioms).
- Each axiom is "about" a predicate (not an action per se).
  - *P* true afterward $\Leftrightarrow$ [an action made *P* true $\vee$ *P* is true already and no action made *P* false]
- For holding the gold,
  - $\forall a,s\ Holding(Gold,Result(a,s)) \Leftrightarrow [((a = Grab) \wedge AtGold(s)) \vee (Holding(Gold,s) \wedge a \neq Release)]$

Making Plans

- Initial Condition in KB:
  - $At(Agent,[1,1],S_0)$
  - $At(Gold,[1,2],S_0)$
- Query: ASK(*KB*, $\exists s$ $Holding(Gold,s)$)
  - I.e., in what situation will I be holding the gold?
- Answer: $\{s/Result(Grab,Result(Forward,S_0))\}$
  - I.e., go forward and then grab the gold.
- This assumes that the agent is interested in plans starting at S_0 and that S_0 is the only situation described in the knowledge base.

MONTANA STATE UNIVERSITY

Mountains Minds

Making Better Plans

- Represent plans as action sequences $[a_1, a_2, \ldots, a_n]$
- $PlanResult(p,s)$ is the result of executing plan p in situation s.
- Then the query, ASK(*KB*, $\exists p$ $Holding(Gold,PlanResult(p,S_0))$) has the solution, $\{p/[Forward, Grab]\}$
- Definition of *PlanResult* in terms of *Result*:
  - $\forall s$ $PlanResult([],s) = s$
  - $\forall a,p,s$ $PlanResult([a|p],s) = PlanResult(p,Result(a,s))$
- Planning systems are special purpose reasoners designed to do this type of inference more efficiently than a general purpose reasoner.

MONTANA STATE UNIVERSITY

Mountains Minds