Backward chaining

Idea: work backwards from the query q:
to prove q by BC,
check if q is known already, or
prove by BC all premises of some rule concluding q

Avoid loops: check if new subgoal is already on the goal stack

Avoid repeated work: check if new subgoal
1) has already been proved true, or
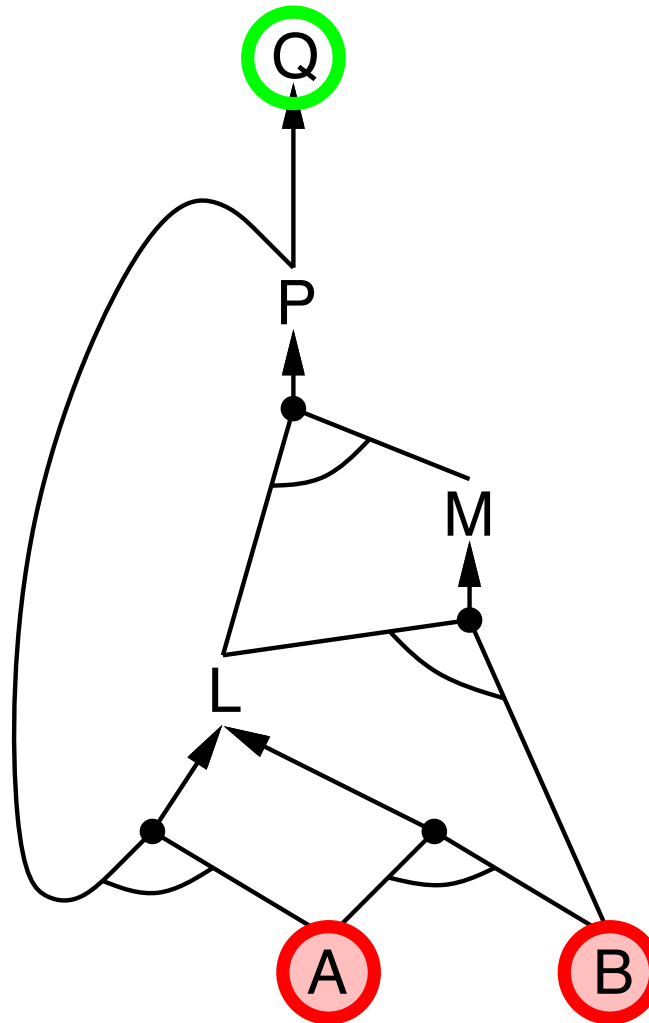2) has already failed

Forward Chaining Pros:
- sound
- complete
- linear in size of KB
- data driven
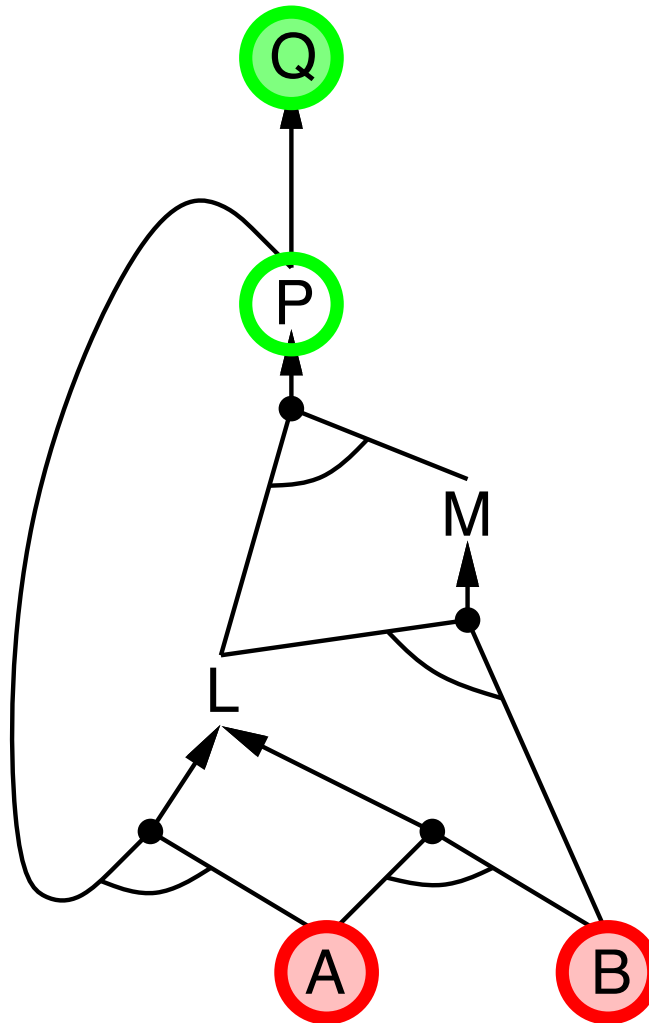- similar to (some) human reasoning

Forward Chaining Cons:
- not goal directed
- may pursue action sequences
irrelevant to the goal
- in general, many action sequences
are irrelevant to the goal
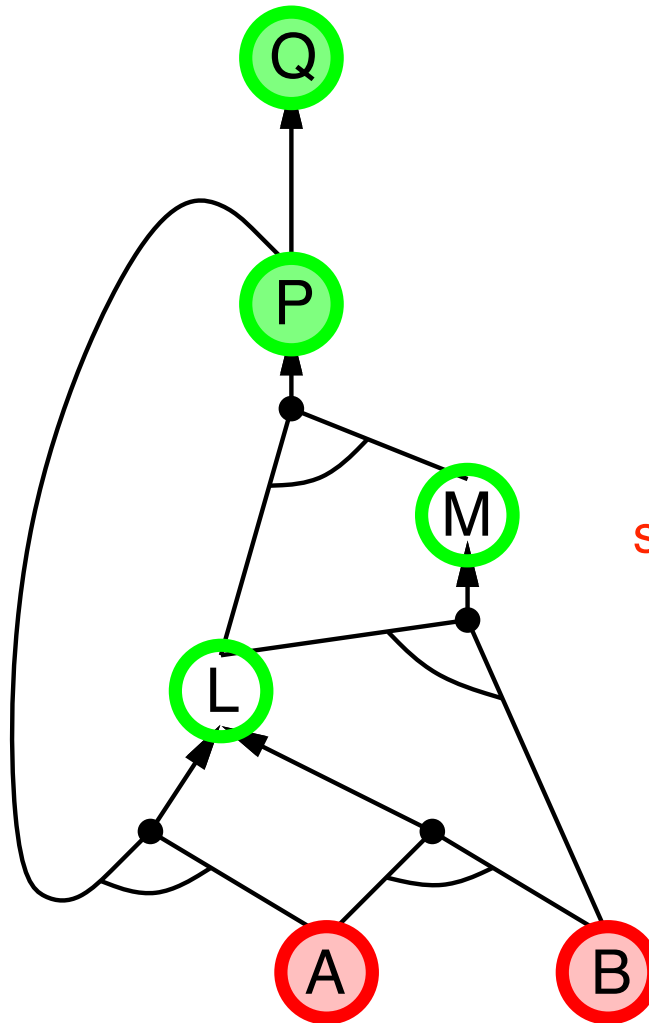
Backward chaining example

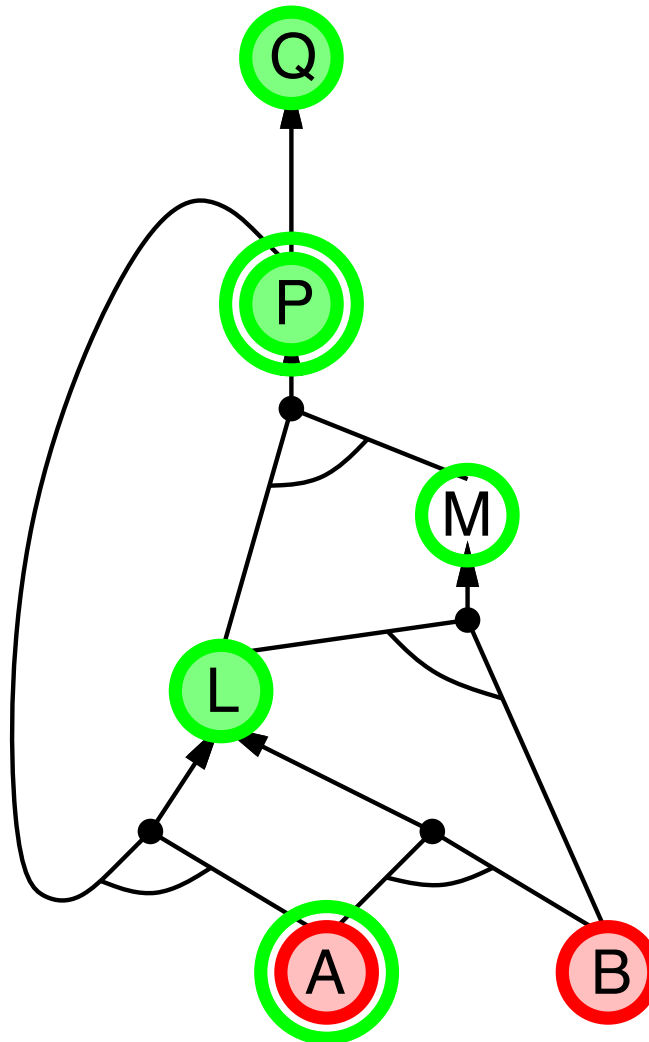Start with the goal, Q

Q

P

M

L

A   B

If we can show that
P holds, then Q
must also hold

Backward chaining example

P is implied by
conjuctions involving L
and M so we need to
show that these are true
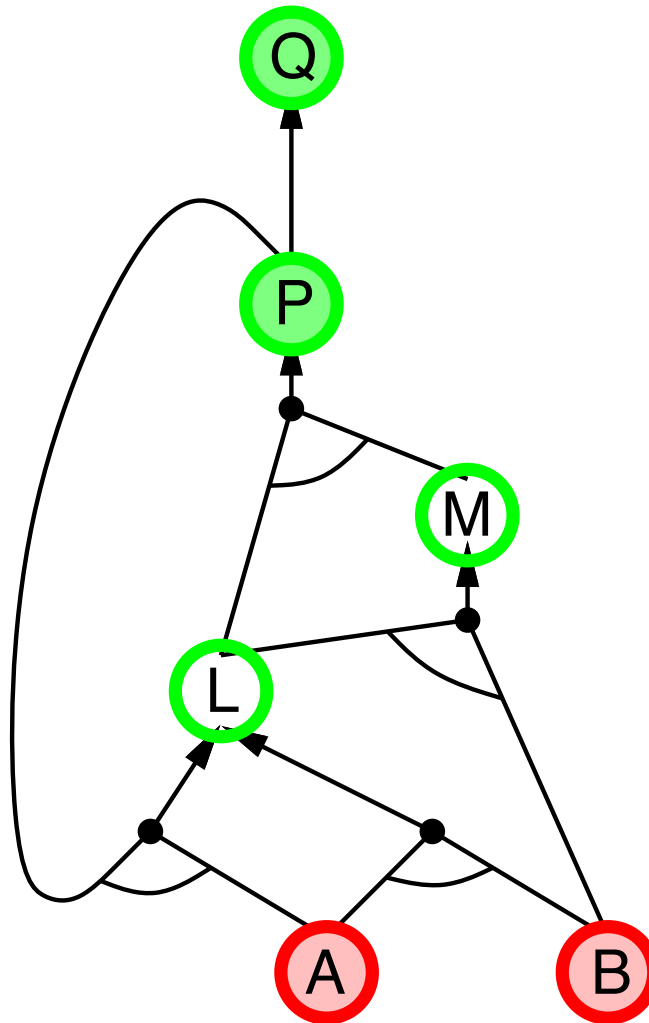
Backward chaining example

Q

P

M

L

A   B

L is implied by the
conjunction A and B

Try A first

Backward chaining example

Backtrack to L and M

Backward chaining example

Q

P

M

L

A   B
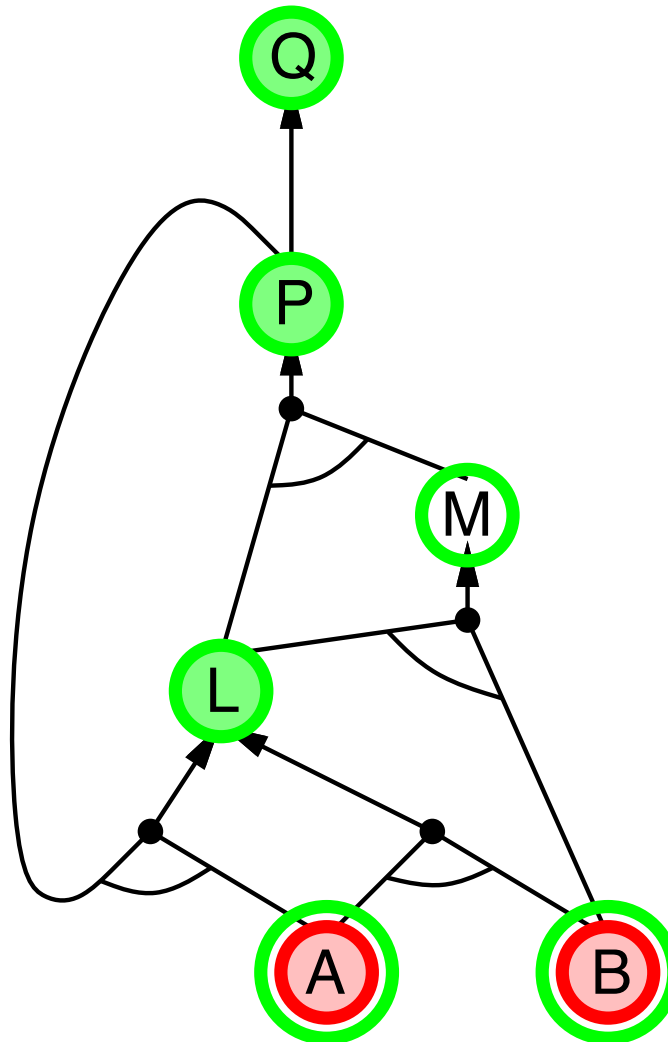
Chain L to B

We now have L
true, but still
need to
consider M
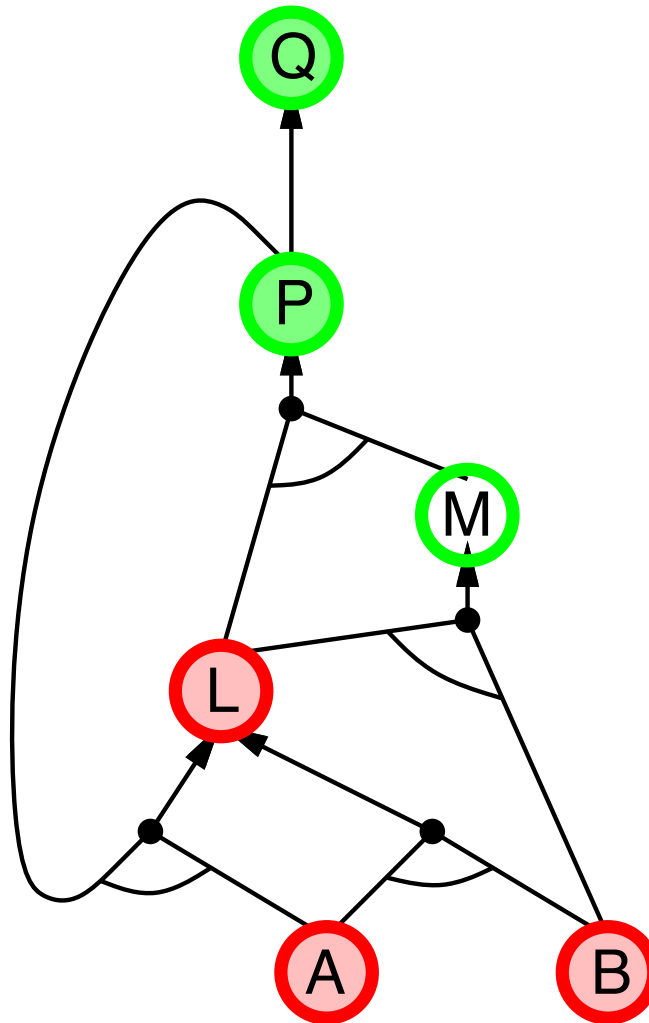
Backward chaining example
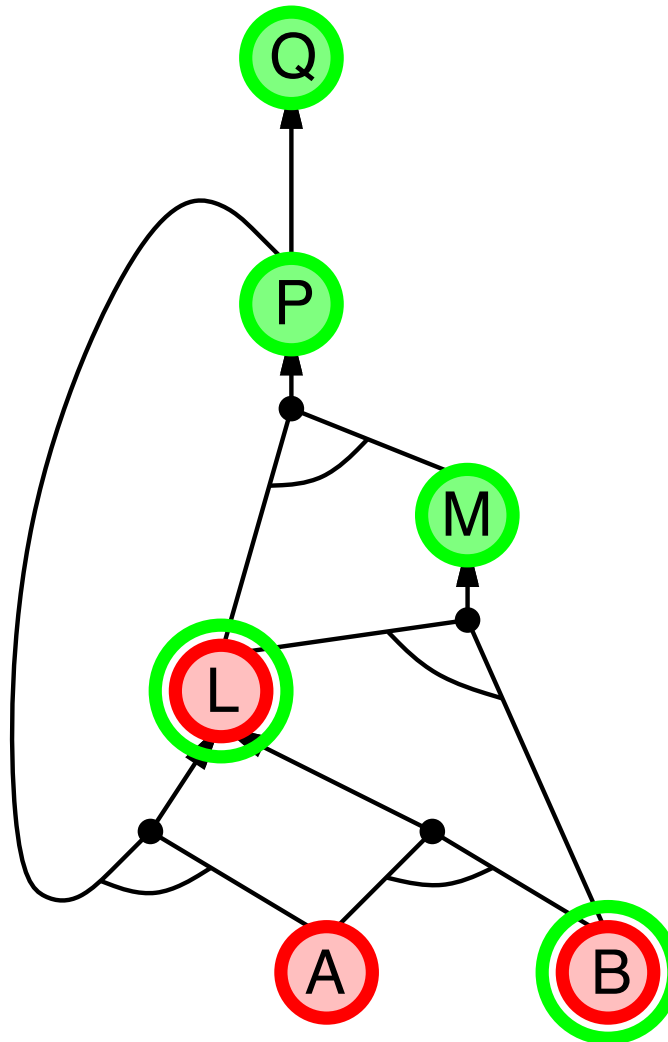
M is implied by L ∧ B

L was shown to be true
earlier in BC

B is a fact

So M is true

Backward chaining example

Q

P

M

L

A   B
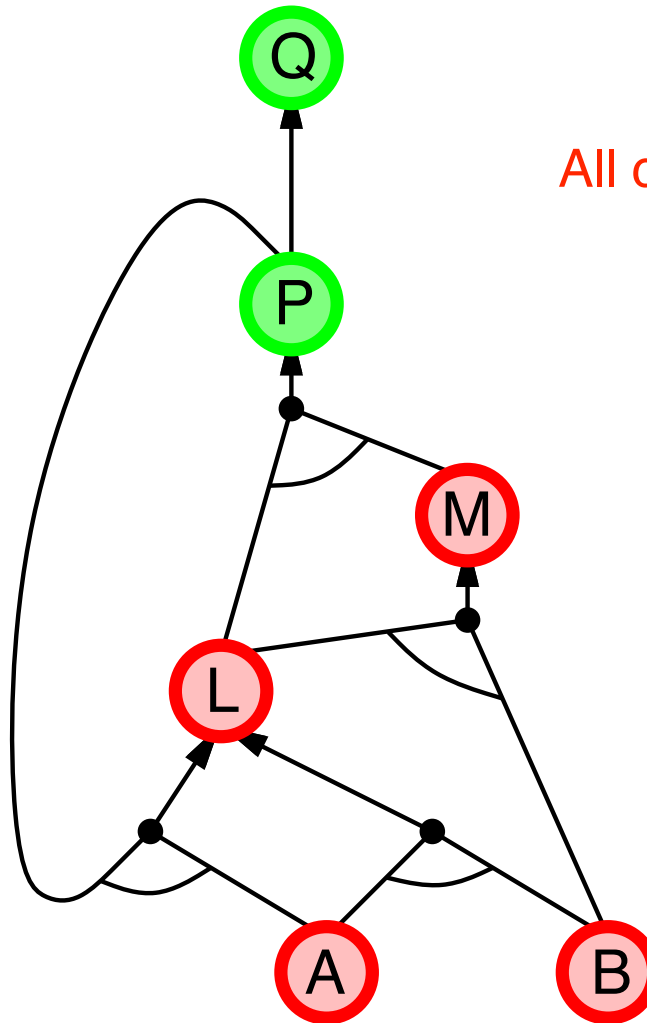
All conjuctions in prmises with
implication P are true

So P must be true
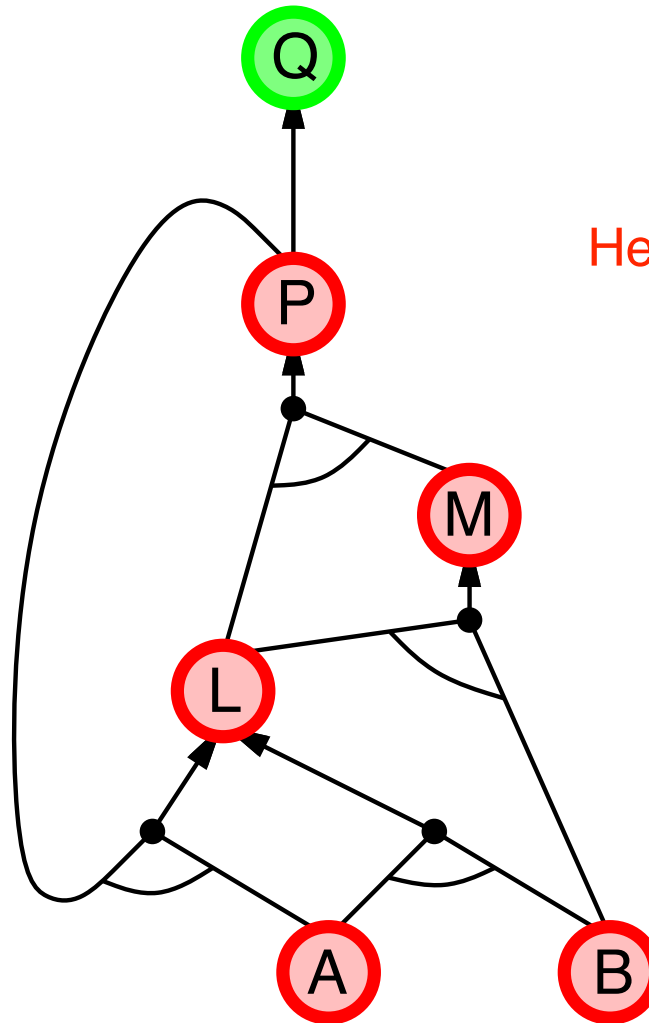
Hence the goal must be true

Q

P
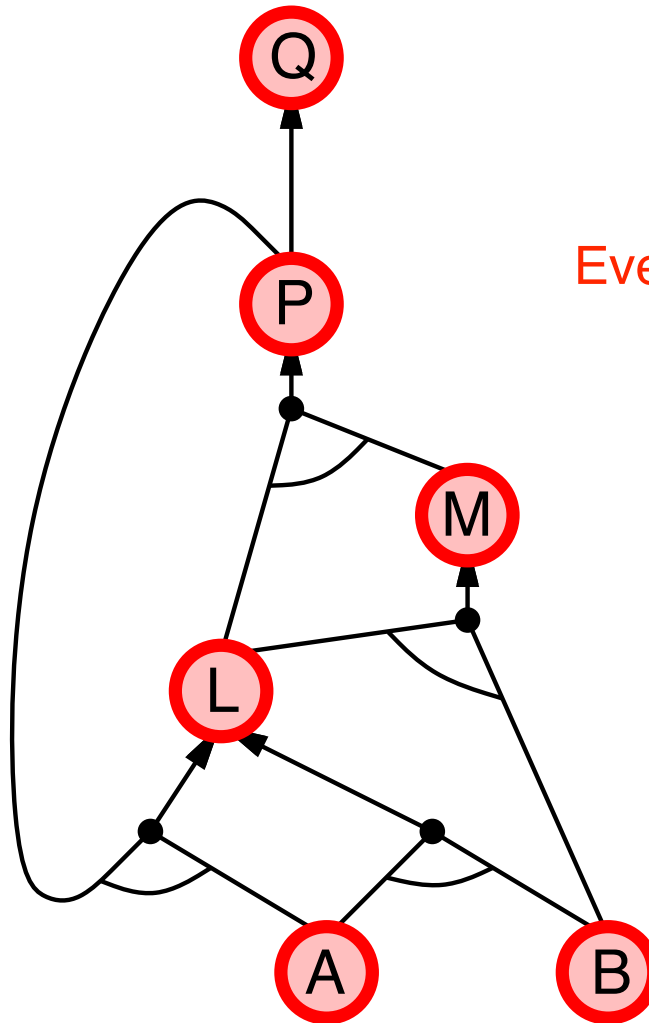
Everything in this example is now a known fact

M

L

A   B

Forward vs. backward chaining

FC is data-driven, cf. automatic, unconscious processing,
   e.g., object recognition, routine decisions

May do lots of work that is irrelevant to the goal

BC is goal-driven, appropriate for problem-solving,
   e.g., Where are my keys? How do I get into a PhD program?

Complexity of BC can be **much less** than linear in size of KB

Resolution

Conjunctive Normal Form (CNF—universal)
conjunction of $\underbrace{\text{disjunctions of literals}}_{\text{clauses}}$

E.g., $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

Resolution inference rule (for CNF): complete for propositional logic

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n}$$

where ℓ_i and m_j are complementary literals. E.g.,

$$\frac{P_{1,3} \vee P_{2,2}, \qquad \neg P_{2,2}}{P_{1,3}}$$

Resolution is sound and complete for propositional logic
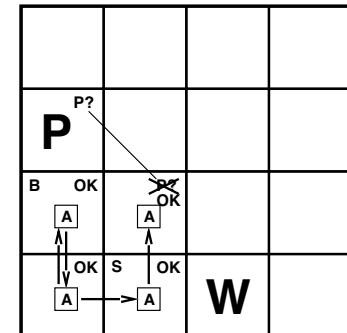
Conversion to CNF

$B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$

1. Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)$.

   $(B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$

2. Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \lor \beta$.

   $(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land (\neg(P_{1,2} \lor P_{2,1}) \lor B_{1,1})$

3. Move $\neg$ inwards using de Morgan's rules and double-negation:

   $(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land ((\neg P_{1,2} \land \neg P_{2,1}) \lor B_{1,1})$

4. Apply distributivity law ($\lor$ over $\land$) and flatten:

   $(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land (\neg P_{1,2} \lor B_{1,1}) \land (\neg P_{2,1} \lor B_{1,1})$

SAT is the class of problems: decide if a CNF sentence is satisfiable

This is the canonical NP-complete problem

Resolution algorithm

Proof by contradiction, i.e., show $KB \wedge \neg\alpha$ unsatisfiable

---

function PL-RESOLUTION(KB, α) **returns** *true* or *false*
    inputs: KB, the knowledge base, a sentence in propositional logic
             α, the query, a sentence in propositional logic

    clauses ← the set of clauses in the CNF representation of $KB \wedge \neg\alpha$
    new ← { }
    loop do
        for each C_i, C_j **in** *clauses* **do**
            resolvents ← PL-RESOLVE(C_i, C_j)
            if *resolvents* contains the empty clause **then return** *true*
            new ← *new* ∪ *resolvents*
        if *new* ⊆ *clauses* **then return** *false*
        clauses ← *clauses* ∪ *new*

Resolution example

$$KB = (B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})) \land \neg B_{1,1} \quad \alpha = \neg P_{1,2}$$
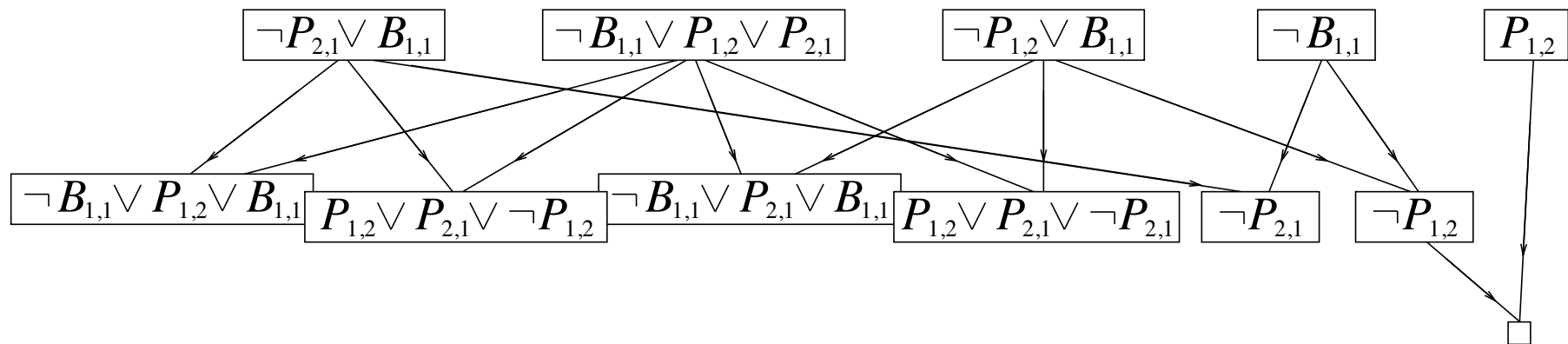
PL-RESOLVE means eliminate a variable if it and its negation appear in the combined clauses

In this example we reach an empty clause at the first pass - in general we take the new clauses and carry on for as many steps as needed

Modern SAT solvers extend this basic idea with concepts such as intelligent backtracking and component analysis to solve huge SAT instances quickly

Summary

Logical agents apply inference to a knowledge base
to derive new information and make decisions

Basic concepts of logic:
- syntax: formal structure of sentences
- semantics: truth of sentences wrt models
- entailment: necessary truth of one sentence given another
- inference: deriving sentences from other sentences
- soundess: derivations produce only entailed sentences
- completeness: derivations can produce all entailed sentences

Wumpus world requires the ability to represent partial and negated information, reason by cases, etc.

Forward, backward chaining are linear-time, complete for Horn clauses
Resolution is complete for propositional logic

Propositional logic lacks expressive power

FIRST-ORDER LOGIC

CHAPTER 8

Outline

◇ Why FOL?

◇ Syntax and semantics of FOL

◇ Fun with sentences

◇ Wumpus world in FOL

Pros and cons of propositional logic

☺ Propositional logic is **declarative**: pieces of syntax correspond to facts

☺ Propositional logic allows partial/disjunctive/negated information
(unlike most data structures and databases)

☺ Propositional logic is **compositional**:
meaning of $B_{1,1} \land P_{1,2}$ is derived from meaning of $B_{1,1}$ and of $P_{1,2}$

☺ Meaning in propositional logic is **context-independent**
(unlike natural language, where meaning depends on context)

☹ Propositional logic has very limited expressive power
(unlike natural language)
E.g., cannot say "pits cause breezes in adjacent squares"
except by writing one sentence for each square

First-order logic

Whereas propositional logic assumes world contains **facts**,
first-order logic (like natural language) assumes the world contains

- Objects: people, houses, numbers, theories, Ronald McDonald, colors, baseball games, wars, centuries . . .

- Relations: red, round, bogus, prime, multistoried . . .,
  brother of, bigger than, inside, part of, has color, occurred after, owns, comes between, . . .

- Functions: father of, best friend, third inning of, one more than, end of
  . . .

Logics in general

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, times	true/false/unknown
Probability theory	facts	degree of belief
Fuzzy logic	facts + degree of truth	known interval value

Structure of the nature of reality I Potentiality of human knowledge

Syntax of FOL: Basic elements

Constants	$KingJohn,\ 2,\ UCB, \ldots$
Predicates	$Brother,\ >, \ldots$
Functions	$Sqrt,\ LeftLegOf, \ldots$
Variables	$x,\ y,\ a,\ b, \ldots$
Connectives	$\wedge\ \vee\ \neg\ \Rightarrow\ \Leftrightarrow$
Equality	$=$
Quantifiers	$\forall\ \exists$

For us, a predicate is a function that returns true or false and hence defines a relation

The equality symbol (=) is a logical constant and every model of the logic interprets it as the identity relation. FOL without equality exists, but is beyond the scope of this module

Atomic sentences

Atomic sentence $=$ $predicate(term_1, \ldots, term_n)$
or $term_1 = term_2$

Term $=$ $function(term_1, \ldots, term_n)$
or $constant$ or $variable$

E.g., $Brother(KingJohn, RichardTheLionheart)$
$> (Length(LeftLegOf(Richard)), Length(LeftLegOf(KingJohn)))$

Complex sentences

Complex sentences are made from atomic sentences using connectives

$$\neg S, \quad S_1 \wedge S_2, \quad S_1 \vee S_2, \quad S_1 \Rightarrow S_2, \quad S_1 \Leftrightarrow S_2$$

E.g. $Sibling(KingJohn, Richard) \Rightarrow Sibling(Richard, KingJohn)$
$>(1,2) \vee \leq(1,2)$
$>(1,2) \wedge \neg>(1,2)$

Truth in first-order logic

Sentences are true with respect to a model and an interpretation

Model contains ≥ 1 objects (domain elements) and relations among them

Interpretation specifies referents for
    constant symbols $\rightarrow$ objects
    predicate symbols $\rightarrow$ relations
    function symbols $\rightarrow$ 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 $predicate$

Models for FOL: Example
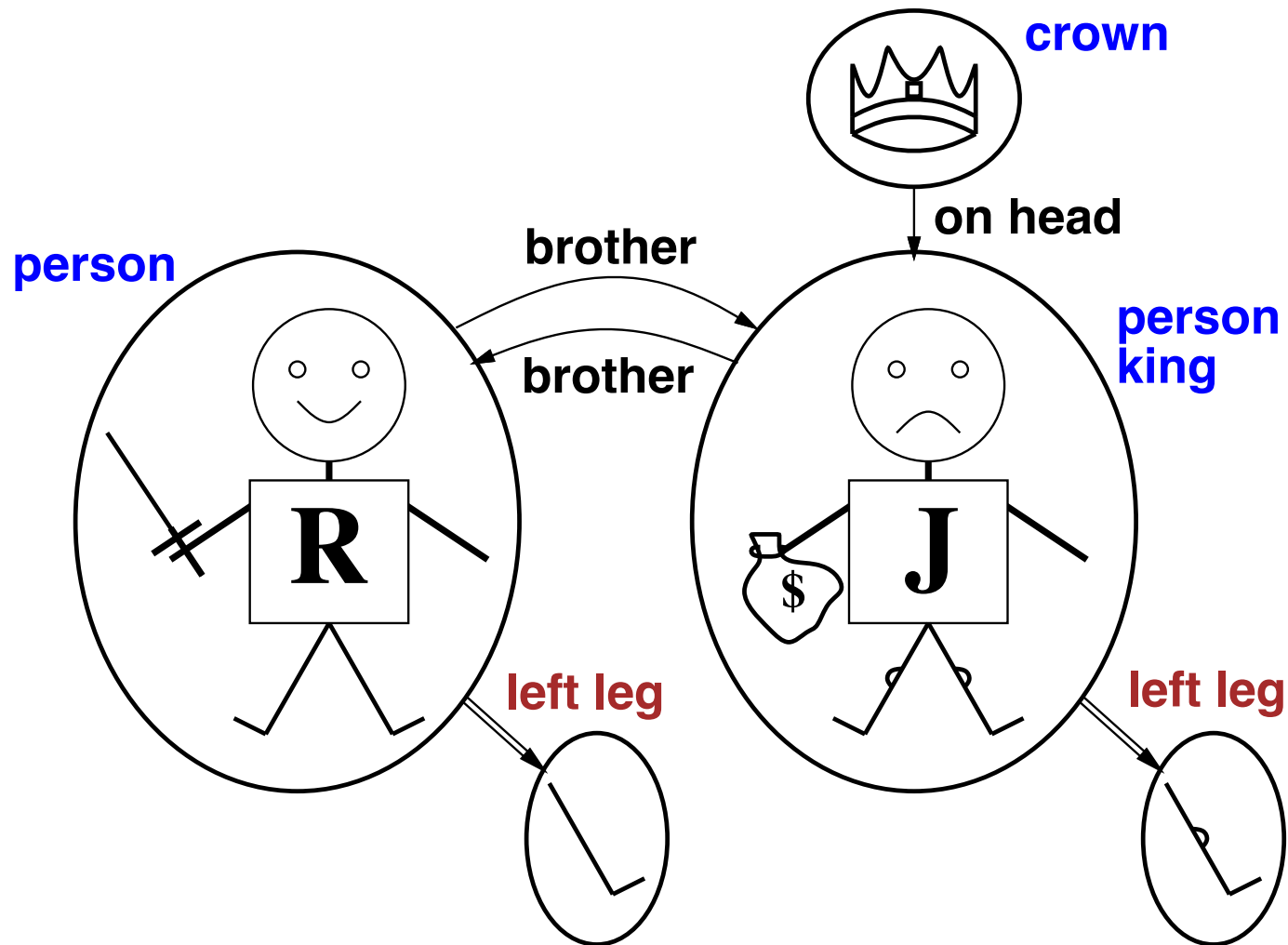
Truth example

Consider the interpretation in which
$Richard \rightarrow$ Richard the Lionheart
$John \rightarrow$ the evil King John
$Brother \rightarrow$ the brotherhood relation

Under this interpretation, $Brother(Richard, John)$ is true
just in case Richard the Lionheart and the evil King John
are in the brotherhood relation in the model

Models for FOL: Lots!

Entailment in propositional logic can be computed by enumerating models

We **can** enumerate the FOL models for a given KB vocabulary:

For each number of domain elements n from 1 to ∞
    For each k-ary predicate P_k in the vocabulary
        For each possible k-ary relation on n objects
            For each constant symbol C in the vocabulary
                For each choice of referent for C from n objects . . .

Computing entailment by enumerating FOL models is not easy!

Universal quantification

$\forall \langle variables \rangle \ \langle sentence \rangle$

Everyone at Berkeley is smart:
$\forall x \ \ At(x, Berkeley) \implies Smart(x)$

$\forall x \ \ P$ is true in a model m iff P is true with x being **each** possible object in the model

Roughly speaking, equivalent to the conjunction of instantiations of P

$$(At(KingJohn, Berkeley) \implies Smart(KingJohn))$$
$$\wedge \ (At(Richard, Berkeley) \implies Smart(Richard))$$
$$\wedge \ (At(Berkeley, Berkeley) \implies Smart(Berkeley))$$
$$\wedge \ \dots$$

A common mistake to avoid

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

Common mistake: using $\wedge$ as the main connective with $\forall$:

$$\forall\, x \quad At(x, Berkeley) \wedge Smart(x)$$

means "Everyone is at Berkeley and everyone is smart"

Existential quantification

$\exists \langle variables \rangle \; \langle sentence \rangle$

Someone at Stanford is smart:
$\exists x \; At(x, Stanford) \wedge Smart(x)$

$\exists x \; P$ is true in a model m iff P is true with x being
some possible object in the model

Roughly speaking, equivalent to the disjunction of instantiations of P

$$(At(KingJohn, Stanford) \wedge Smart(KingJohn))$$
$$\vee \; (At(Richard, Stanford) \wedge Smart(Richard))$$
$$\vee \; (At(Stanford, Stanford) \wedge Smart(Stanford))$$
$$\vee \; \ldots$$

Another common mistake to avoid

Typically, $\wedge$ is the main connective with $\exists$

Common mistake: using $\Rightarrow$ as the main connective with $\exists$:

$$\exists x \ At(x, Stanford) \Rightarrow Smart(x)$$

is true if there is anyone who is not at Stanford!

See this by rewriting to CNF:

Not(At(x,Stanford)) $\vee$ Smart(x)

Properties of quantifiers

$\forall x \; \forall y$ is the same as $\forall y \; \forall x$ (<u>why</u>??)

$\exists x \; \exists y$ is the same as $\exists y \; \exists x$ (<u>why</u>??)

Roughly speaking, because the order in conjunctions and disjunctions doesn't matter

$\exists x \; \forall y$ 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

$\forall x \; Likes(x, IceCream)$ $\neg \exists x \; \neg Likes(x, IceCream)$

$\exists x \; Likes(x, Broccoli)$ $\neg \forall x \; \neg Likes(x, Broccoli)$

Fun with sentences

Brothers are siblings

Fun with sentences

Brothers are siblings

$$\forall\, x, y \;\; Brother(x, y) \;\Rightarrow\; Sibling(x, y).$$

"Sibling" is symmetric

Fun with sentences

Brothers are siblings

$\forall\, x, y \ \ Brother(x, y) \ \Rightarrow \ Sibling(x, y).$

"Sibling" is symmetric

$\forall\, x, y \ \ Sibling(x, y) \ \Leftrightarrow \ Sibling(y, x).$

One's mother is one's female parent

Fun with sentences

Brothers are siblings

$\forall x, y \; Brother(x, y) \; \Rightarrow \; Sibling(x, y)$.

"Sibling" is symmetric

$\forall x, y \; Sibling(x, y) \; \Leftrightarrow \; Sibling(y, x)$.

One's mother is one's female parent

$\forall x, y \; Mother(x, y) \; \Leftrightarrow \; (Female(x) \wedge Parent(x, y))$.

A first cousin is a child of a parent's sibling

Fun with sentences

Brothers are siblings

$\forall x, y \ \ Brother(x, y) \ \Rightarrow \ Sibling(x, y)$.

"Sibling" is symmetric

$\forall x, y \ \ Sibling(x, y) \ \Leftrightarrow \ Sibling(y, x)$.

One's mother is one's female parent

$\forall x, y \ \ Mother(x, y) \ \Leftrightarrow \ (Female(x) \wedge Parent(x, y))$.

A first cousin is a child of a parent's sibling

$\forall x, y \ \ FirstCousin(x, y) \ \Leftrightarrow \ \exists p, ps \ \ Parent(p, x) \wedge Sibling(ps, p) \wedge Parent(ps, y)$

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., $1 = 2$ and $\forall x \ \times(Sqrt(x), Sqrt(x)) = x$ are satisfiable
     $2 = 2$ is valid

E.g., definition of (full) $Sibling$ in terms of $Parent$:
   $\forall x, y \ \ Sibling(x, y) \Leftrightarrow [\neg(x = y) \land \exists m, f \ \neg(m = f) \land$
       $Parent(m, x) \land Parent(f, x) \land Parent(m, y) \land Parent(f, y)]$

We can also use the equality symbol to signify that two
terms refer to the same object:

Father(John) = Henry

Interacting with FOL KBs

Suppose a wumpus-world agent is using an FOL KB
and perceives a smell and a breeze (but no glitter) at $t = 5$:

$Tell(KB, Percept([Smell, Breeze, None], 5))$
$Ask(KB, \exists a \; Action(a, 5))$

I.e., does KB entail any particular actions at $t = 5$?

Answer: $Yes, \; \{a/Shoot\}$ $\leftarrow$ substitution (binding list)

Action a can only be
one of a few things.
We substitute a
specific action for a

Given a sentence S and a substitution σ,
$S\sigma$ denotes the result of plugging σ into S; e.g.,
$S = Smarter(x, y)$
$\sigma = \{x/Hillary, y/Bill\}$
$S\sigma = Smarter(Hillary, Bill)$

$Ask(KB, S)$ returns some/all σ such that $KB \models S\sigma$

Knowledge base for the wumpus world

"Perception"
$\forall b, g, t \; Percept([Smell, b, g], t) \Rightarrow Smelt(t)$
$\forall s, b, t \; Percept([s, b, Glitter], t) \Rightarrow AtGold(t)$

Reflex: $\forall t \; AtGold(t) \Rightarrow Action(Grab, t)$

Reflex with internal state: do we have the gold already?
$\forall t \; AtGold(t) \land \neg Holding(Gold, t) \Rightarrow Action(Grab, t)$

$Holding(Gold, t)$ cannot be observed
$\quad\quad \Rightarrow$ keeping track of change is essential

We can have At(Agent,s,t) as a predicate

Fix the Wumpus with forall t At(Wumpus,[3,1],t)

Quantification over time means we need only one
successor-state rule for each predicate, instead of
a different copy for each t

FOL is more expressive and more concise

Deducing hidden properties

Properties of locations:
$$\forall\, x, t \quad At(Agent, x, t) \wedge Smelt(t) \;\Rightarrow\; Smelly(x)$$
$$\forall\, x, t \quad At(Agent, x, t) \wedge Breeze(t) \;\Rightarrow\; Breezy(x)$$

Squares are breezy near a pit:

Diagnostic rule—infer cause from effect
$$\forall\, y \quad Breezy(y) \;\Rightarrow\; \exists\, x \quad Pit(x) \wedge Adjacent(x, y)$$

Causal rule—infer effect from cause
$$\forall\, x, y \quad Pit(x) \wedge Adjacent(x, y) \;\Rightarrow\; Breezy(y)$$

Neither of these is complete—e.g., the causal rule doesn't say whether squares far away from pits can be breezy

Definition for the $Breezy$ predicate:
$$\forall\, y \quad Breezy(y) \;\Leftrightarrow\; [\exists\, x \quad Pit(x) \wedge Adjacent(x, y)]$$

Exercises: 8.8, 8.10, 8.11