Knowledge base for the wumpus world

“Perception”
\[ \forall b, g, t \ \text{Percept}([\text{Smell}, b, g], t) \Rightarrow \text{Smelt}(t) \]
\[ \forall s, b, t \ \text{Percept}([s, b, \text{Glitter}], t) \Rightarrow \text{AtGold}(t) \]

Reflex: \[ \forall t \ \text{AtGold}(t) \Rightarrow \text{Action(Grab, t)} \]

Reflex with internal state: do we have the gold already?
\[ \forall t \ \text{AtGold}(t) \land \neg \text{Holding(} \text{Gold, t}) \Rightarrow \text{Action(Grab, t)} \]

\[ \text{Holding(} \text{Gold, t}) \text{ cannot be observed} \]
\[ \Rightarrow \text{keeping track of change is essential} \]
Deducing hidden properties

Properties of locations:
\[ \forall x, t \; \text{At}(\text{Agent}, x, t) \land \text{Smelt}(t) \Rightarrow \text{Smelly}(x) \]
\[ \forall x, t \; \text{At}(\text{Agent}, x, t) \land \text{Breeze}(t) \Rightarrow \text{Breezy}(x) \]

Squares are breezy near a pit:

**Diagnostic** rule—infer cause from effect
\[ \forall y \; \text{Breezy}(y) \Rightarrow \exists x \; \text{Pit}(x) \land \text{Adjacent}(x, y) \]

**Causal** rule—infer effect from cause
\[ \forall x, y \; \text{Pit}(x) \land \text{Adjacent}(x, y) \Rightarrow \text{Breezy}(y) \]

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:
\[ \forall y \; \text{Breezy}(y) \Leftrightarrow [\exists x \; \text{Pit}(x) \land \text{Adjacent}(x, y)] \]
Keeping track of change

Facts hold in situations, rather than eternally
E.g., \(\text{Holding}(\text{Gold, Now})\) rather than just \(\text{Holding}(\text{Gold})\)

Situation calculus is one way to represent change in FOL:
- Adds a situation argument to each non-eternal predicate
  E.g., \(\text{Now}\) in \(\text{Holding}(\text{Gold, Now})\) denotes a situation

Situations are connected by the \textit{Result} function
\(\text{Result}(a, s)\) is the situation that results from doing \(a\) in \(s\)
Describing actions I

“Effect” axiom—describe changes due to action
∀s AtGold(s) ⇒ Holding(Gold, Result(Grab, s))

“Frame” axiom—describe non-changes due to action
∀s HaveArrow(s) ⇒ HaveArrow(Result(Grab, s))

Frame problem: find an elegant way to handle non-change
(a) representation—avoid frame axioms
(b) inference—avoid repeated “copy-overs” 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, . . .
Describing actions II

Successor-state axioms solve the representational frame problem

Each axiom is “about” a **predicate** (not an action per se):

\[
P \text{ true afterwards } \Leftrightarrow [\text{an action made } P \text{ true} \\\ \vee \\ \text{P true already and no action made } P \text{ false}]
\]

For holding the gold:

\[
\forall a, s \ \text{Holding}(\text{Gold, Result}(a, s)) \Leftrightarrow
[\ (a = \text{Grab} \land \text{AtGold}(s)) \\ \vee \ (\text{Holding}(\text{Gold, s}) \land a \neq \text{Release} )]
\]
Making plans

Initial condition in KB:

\[
\begin{align*}
  \text{At}(Agent, [1, 1], S_0) \\
  \text{At}(Gold, [1, 2], S_0)
\end{align*}
\]

Query: \( \text{Ask}(KB, \exists s \ \text{Holding}(Gold, s)) \)

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 KB
Making plans: A better way

Represent plans as action sequences \([a_1, a_2, \ldots, a_n]\)

\(PlanResult(p, s)\) is the result of executing \(p\) in \(s\)

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:
\[
\begin{align*}
\forall s \; PlanResult([], s) &= s \\
\forall a, p, s \; PlanResult([a|p], s) &= PlanResult(p, Result(a, s))
\end{align*}
\]

Planning systems are special-purpose reasoners designed to do this type of inference more efficiently than a general-purpose reasoner
Summary

First-order logic:
  – objects and relations are semantic primitives
  – syntax: constants, functions, predicates, equality, quantifiers

Increased expressive power: sufficient to define wumpus world

Situation calculus:
  – conventions for describing actions and change in FOL
  – can formulate planning as inference on a situation calculus KB