**Knowledge Representation and Reasoning**

**Knowledge-based agents**

Can accept new tasks in the form of explicitly described goals

* being told or learning new knowledge about the envirnoment
* adapt to changes in the envirnoment by updating the relevant knowledge

**Knowledge Base**

* a set of “sentences”, each representing some assertion about the world
* expressed in a “knowledge representation language”
* initial content: “background knowledge”

Adding new sentences to the knowledge base (assertions) : **TELL**

Querying what is known: **ASK**

Interference: deriving new sentences from existing ones

* when asking a question of the knowledge base, the answer should “follow” from what has been told to the knowledge base (previous assertions)

**Knowledge-based Agent Program**

* TELL the KB what it perceives
* ASK the KB what action to perform
  + reasoning about the current state of the world, outcomes of possible actions, etc
* TELL the KB which action was performed in the world

**Knowledge vs Implementation Level**

A knowledge-based agent can be described at the “knowledge level”

* we need only to specificy what the agent knows and what its goals are
* example:
  + an automated taxi has the goal of taxing passenger from Porto to gaia and might know that it must cross one of the beautiful bridges of the Douro river
  + we expect it to cross a bridge because it knows this will achieve its goal
* “declarative” approach to system building: TELLing the agent what it needs to know

Implementation level: data structures inside the KB and algorithms that work on them

* procedural approach: encode behaviours directly as program code

**Logic**

Representing the sentences in the KB

* syntax: specifies the sentences that are well formed
  + ex. “x+y = 4”, not “x4y +=”
* semantics: assigns meaning to sentences, determining their truthfulness in respect to each possible world, or model
  + ex. “x+y = 4” is true in a world in which both x and y are 2, but false in a world where they are both 1

Sentence a is true in a model m

* m satisfies a, or m is a model of a

M(a): the set of all models of a

**Entailment**

Entailment: 𝜶 ⊨ B

* 𝜶 entails B (or B follows logically from 𝜶)
* 𝜶 ⊨ if and only if M(𝜶) ⊆ M(B)
  + 𝜶 is a stronger assertion than B

Adding knowledge to a KB:

* KB ⊨ 𝜶

**Logical Interence**

Entailment can be applied to derive conclusions: “logical inference”

Algorithm i that can derive a from KB

* KB ⊢i a1

Properties of intererence algorithms:

* “soundness” (or trith preserving): derive only entailed sentences
* completeness: derive any sentence that is entailed

If KB is true in the real world, then any sentence a derived from KB by a sound inference procedure is also true in the real world

**Correspondence**

The inference procedure:

* operates on the syntatic representations (sentences) but corresponds to the real-world relationship
* constructs new sentences from existing ones
* to be sound, should entail only sentences representing facts that follow form the facts represented by the KB

**Propositional Logic: Syntax**

* Symbols:
  + logical constants True and False
  + propositional symbols such as P and Q
  + logical connectives: ∧ ∨ ⇒ ⇔ ¬
  + parentheses “(“ and “)”
* Sentences are sequences of symbols, such that:
  + True, False, P or Q are sentences by themselves (atomic sentences)
  + complex sentences are constructed from simples sentences, using parenthesis and logical connectives:
    - ∧ (and). A sentence whose main connective is ∧ is valled a conjunction
    - ∨ (or). A sentence whose main connective is ∨ is called a disjunction
    - ⇒ (implies). A sentence in the form (P ∧ Q ⇒ R) is called an implication
    - ⇔ (if and only if). A sentence in the form (𝑃 ∧ 𝑄 ⇔ 𝑄 ∧ 𝑃) is an equivalence
    - ¬ (not). A sentence in the form ¬P is called a negation of P
  + operator precedence: ¬ ∧ ∨ ⇒ ⇔

**Propositional Logic: Semantics**

* True represents a true fact; False represents a false fact
* The meaning of a complex sentence is derived from the meaning of its parts by a process of decomposition
  + 𝑃 ∨ 𝑄 ∧ ¬S: first determing the meaning of (P ∨ Q) and of ¬S, then combine the two using the definition of ∧
* the truth value of every other proposition symbol must be specified directly in the model
  + 𝑚1 = {𝑃1,2 = 𝑓𝑎𝑙𝑠𝑒, 𝑃2,2 = 𝑓𝑎𝑙𝑠𝑒, 𝑃3,1 = 𝑡𝑟𝑢e}

**Logical Equivalence**

Two sentences A and B are logically equivalent if they are true in the same set of models: M(A) = M(B)

In other words: A = B if A ⊨ B and B ⊨ A

**Validity and Satifiability**

A sentence is valid it it is true in all models

* tautology: a necessarily true sentence
* deduction theorem: A ⊨ B if (A ⇒ B) is valid

A sentence is satifieable if it is true in some model

* KB = (R1, R2, R3, R4, R5) is satisfiable because it is true in three models

A is valid if ¬A is unsatisfiable

A is satisfiable if ¬A is not valid

A ⊨ B if the sentence (A ∧ ¬B) is unsatisfiable

* principle of the proof by contradictionq

**Inference Rules**

Truth tables can be used t