Logical Agents

- Objectives
  - Inference and entailment
  - Sound and complete inference algorithms
  - Inference by model checking
  - Inference by proof
  - Resolution
  - Forward and backward chaining
- Reference
  - Russel/Norvig: Chapter 7

Knowledge-Based Agents

- When env is partially observable, how can agent find out things that are important but unobservable?
  - The role of knowledge
- How can knowledge on task env be maintained?
- A knowledge base (KB) contains a set of sentences formatted according to a representation language.
  - Each sentence represents an assertion about the env.
- How does agent access KB?
  1) Add new sentences by Tell.
  2) Query what agent needs to know by Ask.

Inference

A. Ask is not a simple database retrieval.
   - Answer to a query may not be a sentence added earlier.
B. Answering a query often requires inference that derives new sentences from existing ones.
C. An inference algorithm is associated with KB.
   - Answer to a query may be added to KB.
D. Fundamental requirement of inference is that an answer should follow from sentences in KB.

Ex Wumpus World

1) A wumpus lives in a cave who eats anyone in its room.
2) Pits trap explorer, except wumpus.
3) Agent has one arrow and can shoot wumpus.
4) Rooms next to wumpus are smelly.
5) Rooms next to pit are breezy.
6) Wall bumps.
7) One room has a heap of gold.
8) Gold glitters.
9) Dying wumpus screams.
10) Room (1,1) is safe.
Exploring Wumpus World

A. Sensors, actuators, and goal
   1) Can perceive smell, breeze, glitter, bump, and scream, denoted by 5 variables (s,b,g,p,m).
   2) Can turn, forward, shoot, and grab.
   3) Get gold without being eaten or falling into pit.

B. How will a knowledge-based agent explore WW?
   a) What is agent’s knowledge state before entering WW?
   b) What is the knowledge state at (1,1)?
   c) What does agent know after moving to (2,1)?
   d) What does agent know after moving to (1,2)?
   e) What should agent do next?

Reflection on Wumpus World

• Agent draws conclusions on unobservable events from background knowledge and observations.
• In deterministic environments, the conclusions are guaranteed to be correct, if available information is correct.
• This is a fundamental property of logical reasoning.
• How can such logical agents be implemented?

Syntax and Semantics of Language

• Components of representation language
   - Syntax specifies what sentences are well-formed.
   - Semantics determines truth for each sentence in each model (possible world).

1. Ex Language for sets
2. Ex S₁: \( X \cup Y = Z \).
3. Ex S₂: \( X \cap Y = Z \).
   - Model M₁: \( X = \{2,5,7\}, Y = \{3,7,8\}, Z = \{7\} \).
4. Ex S₃: “There is smell in (4,3)”.
   - Model M₂
5. Ex M₃ is similar to M₂ but wumpus is in (4,2).

Entailment

• Sentence \( \alpha \) entails sentence \( \beta \) iff, in every model where \( \alpha \) is true, \( \beta \) is true. It is denoted by \( \alpha \models \beta \).
• Ex For sets \( X \) and \( Y \), we have \( X = Y \models X \subseteq Y \).
• Entailment captures the intuitive notion follow.
• When \( \alpha \models \beta \) holds, does \( \beta \models \alpha \) hold?
KB and Entailment

- A KB can be viewed as a statement that asserts each sentence in the KB.
  \( \text{"KB } \models \beta \text{" is well-formed, where } \beta \text{ is a sentence.} \)
- Ex Agent Ag is in WW with at most one pit.
  1. Perceive nothing unusual at (1,1) and breeze at (2,1).
  2. What should Ag’s KB contain at the moment?
  3. For S1 = "no pit in (1,2)", does KB \( \models S_1 \) hold?
  4. For S2 = "no pit in (2,2)", does KB \( \models S_2 \) hold?
  5. For S3 = "pit in (2,2)", does KB \( \models S_3 \) hold?

Inference Algorithm

- Inference derives a new sentence from those in KB. It is performed by an algorithm.
- If inference algorithm \( \text{inf} \) can derive sentence \( \alpha \) from KB, it is denoted as KB \( \vdash_{\text{inf}} \alpha \).
- How does agent utilize inference?

Evaluating Inference Algorithms

1. Result of “derive” (\( \vdash_{\text{inf}} \)) depends on capacity of inf.
2. Given a KB, a sentence \( \alpha \) and an algorithm \( \text{inf} \), either KB \( \vdash_{\text{inf}} \alpha \) or KB \( \not\vdash_{\text{inf}} \alpha \).
3. Given KB and sentence \( \gamma \), either KB \( \not\vdash \gamma \) or KB \( \vdash \gamma \).
4. Algorithm \( \text{inf} \) is sound, if every derived sentence is entailed.
5. Algorithm \( \text{inf} \) is complete, if it can derive every entailed sentence.
6. Sound and complete inference algorithms

WW KB in Propositional Logic (PL)

- Language: Propositional logic
- Symbols
  A. Room: W13, P31, S12, B21
  B. Agent: L110, East0
  C. Percept: Breeze0, Smell2, Bump1
  D. Action: Forward0, TurnLeft3, Shoot5
- Sentences
  1. Rooms next to wumpus are smelly and vice versa.
  2. Breeze is sensed if and only if the room is breezy.
  3. Exactly one pit exists among (2,1), (3,1) and (3,2).
Inference by Model Checking

- Aim: Decide whether \( KB \models \alpha \) for sentence \( \alpha \).
- Inference by model checking
  A. Enumerate all models. Check whether \( \alpha \) is true in each model where \( KB \) is true.
  B. Algorithm \texttt{isEntailed}()
  C. Example on WW

1. Is the algorithm sound?
2. Is the algorithm complete?
3. What is the time complexity?

Algorithm \texttt{isEntailed}

\[
\texttt{isEntailed(kb, } \alpha \texttt{)}
\]

\{
\texttt{symbols = list of symbols in kb and } \alpha \texttt{; for each model m (truth assignment of all symbols), compute truth value of \( kb \) in m; if \( kb \) is true, get truth value of } \alpha \texttt{ in m; if } \alpha \texttt{ is false, return false; return true; }
\}

\[
\begin{align*}
\text{kb} &= \{ \neg P11, \neg B12, B21, B12 \equiv (P11 \lor P22 \lor P13), B21 \equiv (P11 \lor P22 \lor P31) \}
\end{align*}
\]

Theorems on Entailment

1. Equivalence theorem
   For sentences \( \alpha \) and \( \beta \), \( \alpha \models \beta \) and \( \beta \models \alpha \) hold iff \( \alpha \equiv \beta \).
2. Deduction theorem
   For sentences \( \alpha \) and \( \beta \), \( \alpha \models \beta \) holds iff sentence \( \alpha \Rightarrow \beta \) is valid.
3. Contradiction theorem
   For sentences \( \alpha \) and \( \beta \), \( \alpha \models \beta \) holds iff sentence \( \alpha \land \neg \beta \) is unsatisfiable.
Using Inference Rules

- Inference by proof
  - In order to confirm $KB \models \alpha$, derive entailed sentences from $KB$, until $\alpha$ is derived.
- Proof is an alternative to model checking.
- Approach
  - Equip algorithm $inf$ with one or more inference rules ...
    1. How do we ensure that $inf$ is sound?
    2. How do we ensure that each rule is sound?
    3. How do we obtain sound rules?

Inference by Proof

- Ex From $KB = \{S_1, S_2, S_3\}$, decide whether rooms (1,2) and (2,1) are pit-free.
  - $S_1$: $B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$
  - $S_2$: $B_{1,2} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{1,3})$
  - $S_3$: $\neg B_{1,1}$
  1. What is $\alpha$? Derive $\alpha$ by inference rules.
  2. Proof may be more efficient than model checking. Why?
  3. Will proof be always more efficient than model checking?

The Resolution Rule

1. Ex WW agent Ag perceives smell in (1,2).
2. What can Ag infer about wumpus?
3. If Ag also knows that (1,1) is wumpus-free, what can it conclude?
- Unit resolution:
  \[ l_1 \lor \ldots \lor l_k \lor m_i \]
  \[ \frac{l_j \lor \ldots \lor l_{i-1} \lor l_{i+1} \lor \ldots \lor l_k}{l_j \lor \ldots \lor \neg l_i \lor \ldots \lor l_k} \]
  where $l_i, \ldots, l_k, m_i$ are literals and $m_i = \neg l_i$
  a) A clause is a disjunction of literals.
  b) Two literals are complementary, if one is the negation of the other.
  c) The resultant clause is called resolvent.

Full Resolution

- Full resolution
  \[ l_j \lor \ldots \lor l_k \lor m_1 \lor \ldots \lor m_n \]
  \[ \frac{l_j \lor \ldots \lor l_{i-1} \lor l_{i+1} \lor \ldots \lor l_k \lor m_1 \lor \ldots \lor m_{j-1} \lor m_{j+1} \lor \ldots \lor m_n}{l_i \lor \ldots \lor \neg l_i} \]
  where $m_j = \neg l_i$
  - Resolvent contains one copy of each literal, with duplicates removed.
  - Theorem The full resolution inference rule is sound.
Conjunctive Normal Form

1. When sentences in KB are not clauses, how can resolution be applied?
2. A sentence expressed as a conjunction of clauses is in conjunctive normal form (CNF).
3. KB is a conjunction of sentences.
4. If each sentence can be converted into CNF, KB will be in CNF and resolution is applicable.

Converting Sentences to CNF

- Algorithm getCnf(s) returns CNF of a sentence s, where s consists of literals, ¬, ∧, ∨, ⇒, ⇔, and ()
1. If s is a literal, return s.
2. a) If s = ¬(¬p), return getCnf(p).
   b) If s = ¬(p ∧ q), return getCnf(¬p ∨ ¬q).
   c) If s = ¬(p ∨ q), return getCnf(¬p ∧ ¬q).
3. If s = p ∧ q, return getCnf(p) ∧ getCnf(q).
4. If s = p ⇒ q, return getCnf(¬p ∨ q).
5. If s = p ⇔ q, return getCnf((p ∧ q) ∨ (¬p ∧ ¬q)).

Converting to CNF

6. If s = p ∨ q,
   call getCnf(p) that returns p₁ ∧ ... ∧ p_m and
   call getCnf(q) that returns q₁ ∧ ... ∧ q_n.
   Return (p₁ ∨ q₁) ∧ ... ∧ (p₁ ∨ q_n) ∧ ... ∧ (p_m ∨ q₁) ∧ ... ∧ (p_m ∨ q_n).

- Ex Convert R ⇔ (P ∨ ¬Q) to CNF.

Idea of Resolution Algorithm

1. Determine KB ├ α by showing that KB ∧ ¬α is unsatisfiable. Why is it correct?
2. How can resolution determine unsatisfiability?
3. Theorem Sentence β is unsatisfiable iff β ├ P and ¬P for some literal P.
4. Resolution algorithm
5. Ex Determine no pit in (2,1), from KB = {S₁, S₂}.
   S₁: B₁,₁ ⇔ (P₁,₂ ∨ P₂,₁)
   S₂: ¬B₁,₁
Resolution Algorithm

Resolution(kb, α) {
    clau = clauses of kb∧¬α in CNF; new={};
    do {
        for each p, q in clau with complementary literals,
            resolvent = Resolve(p, q);
            if resolvent = ∅, return “kb ⊨ α”;
            new = new ∪ {resolvent};
            if new ⊆ clau, return “kb ⊭ α”;
            clau = clau ∪ new; new={};
    } }

Properties of Resolution Algorithm

A. Does Resolution(kb, α) always terminate?
   1. Resolution closure is the set of all clauses derivable by repeated resolution.
   2. Is resolution closure finite?
B. Is Resolution(kb, α) complete?
C. Theorem If a set of clauses is unsatisfiable, their resolution closure contains the empty clause.
D. What is the time complexity of Resolution()?
FC(kb, q) { // kb: Horn clauses; q: positive literal;
    agenda = facts in kb as a list of symbols;
    for each symbol s in kb, inferred[s] = false;
    for each clause c in kb, count[c] = # of literals in body(c);
    if q is in agenda, return true;
    while agenda is not empty, do
        p = pop(agenda);
        if inferred[p] == false, do
            inferred[p] = true;
            for each clause c with p in body(c), do
                count[c]--;
                if count[c] == 0, do
                    if head(c) == q, return true;
                    push(head(c), agenda);
        return false;
}

Example on Forward Chaining

• Ex For KB below and query Q, decide if KB □ Q.

C1: E ∧ X ⇒ P
C2: E ∧ Y ⇒ Q
C3: D ∧ X ⇒ Y
C4: B ∧ D ⇒ X
C5: A ∧ B ⇒ E
C6: A
C7: B
C8: D

Properties of Forward Chaining

1. Is forward chaining sound?
2. Is forward chaining complete?
3. Modify FC(kb, q) to FC(kb) as follows.
   A. Delete argument q and all statements with q.
   B. Change last statement to "return".
4. FC(kb) derives every atomic sentence entailed.
5. Implication to FC(kb, q)
6. What is the time complexity of forward chaining?
7. What is the price paid for efficiency?

Backward Chaining

1. Find implications in KB that conclude query literal q, and try to derive their premises.
2. Ex Backward chaining
3. Cost is often much less than forward chaining.
4. Ex Forward chaining derived P before Q.
   ❏ Is P relevant to query Q?
Summary

1. In partially observable environment, agent can infer the unobservable from knowledge and percepts.
2. Knowledge representation (KR) and inference are fundamental issues of AI.
3. Focus: Deterministic environment, propositional logic for KR, and inference to decide entailment
4. Inference: model checking, resolution and chaining
   A. They are sound and complete.
   B. Trade-off between expressiveness and efficiency