Review

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Search

• Basic definition of the search problem
  – Search space, operators, initial state, goal condition

• Formulation of a problem:
  – We have some control over the complexity of the search space size

• Two basic types of search problems:
  – Path vs. configuration search
Search

- **Methods for searching the search space:**
  - Search trace captured by the search tree

- **Search methods properties:**
  - Completeness, Optimality, Space and time complexity.

- **Complexities**
  - measured in terms of a branching factor ($b$), depth of the optimal solution ($d$), maximum depth of the state space ($m$)

---

Search

- **Uninformed methods:**
  - Breadth first search, Depth first search, Iterative deepening, Bi-directional search, Uniform cost search (for the weighted path search)

- **Informed methods:**
  - **Heuristic function** ($h$): potential of a state to reach the goal
  - **Evaluation function** ($f$): desirability of a state to be expanded next
  - **Best first search:**
    - Greedy $f(n) = h(n)$
    - A*: $f(n) = g(n) + h(n)$

  the role of admissible heuristics, optimality
Search

• **Constraint satisfaction problem (CSP)**
  – Variables, constraints on values (reflect the goal)
  – Formulation of a CSP as search
  – Methods and heuristics for CSP search
    • Backtracking, constraint propagation, most constrained variable, least constrained value
• **Combinatorial optimization (search). Find the best configuration.**
  – **Iterative algorithms**: Hill climbing, Simulated annealing, Genetic algorithms
  – **Advantage: memory!!**
• **Parametric optimization (search):**
  – **Methods**: Linear program, quadratic, convex optimization, gradient methods

Search

• **Adversarial search (game playing)**
  – Specifics of a game search, game problem formulation
  – rational opponent
• **Algorithms:**
  – **Minimax algorithm**
    • Complexity bottleneck for large games
  – **Alpha-Beta pruning**: prunes branches not affecting the decision of players
  – **Cutoff** of the search tree and heuristics
KR and logic

- **Knowledge representation:**
  - Syntax (how sentences are build), Semantics (meaning of sentences), Computational aspect (how sentences are manipulated)

- **Logic:**
  - A formal language for expressing knowledge and ways of reasoning
  - **Three components:**
    - A set of sentences
    - A set of interpretations
    - The valuation (meaning) function

Propositional logic

- A language for symbolic reasoning
- **Language:**
  - Syntax, Semantics
- **Satisfiability** of a sentence: at least one interpretation under which the sentence can evaluate to True.
- **Validity** of a sentence: True in all interpretations
- **Entailment:** $KB \models \alpha$
  - $\alpha$ is true in all worlds in which KB is true
- **Inference procedure**
  - Soundness  If $KB \vdash \alpha$ then $KB \models \alpha$
  - Completeness  If $KB \models \alpha$ then $KB \vdash \alpha$
Propositional logic

- **Logical inference problem:** $KB \models \alpha$ ?
  - Does $KB$ entail the sentence $\alpha$ ?
- Logical inference problem for the propositional logic is **decidable**.
  - A procedure (program) that stops in finite time exists
- **Approaches:**
  - Truth table approach
  - Inference rule approach
  - Resolution refutation

\[
KB \models \alpha \quad \text{if and only if} \quad (KB \land \neg \alpha) \text{ is unsatisfiable}
\]

- **Normal forms:** DNF, CNF, Horn NF (conversions)

First order logic

- Deficiencies of propositional logic
- **First order logic (FOL):** allows us to represent objects, their properties, relations and statements about them
  - Variables, predicates, functions, quantifiers
  - Syntax and semantics of the sentences in FOL
- **Logical inference problem** $KB \models \alpha$ ?
  - **Undecidable.** No procedure that can decide the entailment for all possible input sentences in a finite number of steps.
- **Inference approaches:**
  - Inference rules
  - Resolution refutation
First order logic

- Methods for making inferences work with variables:
  - Variable substitutions
  - Unification process that takes two similar sentences and computes the substitution that makes them look the same, if it exists

- Conversions to CNF with universally quantified variables
  - Used by resolution refutation
    - The procedure is refutation-complete

Knowledge-based systems with HNF

- KBs in Horn normal form:
  - Not all sentences in FOL can be translated to HNF
  - Modus ponens is complete for Horn databases

- Inferences with KBs in Horn normal form (HNF)
  - Forward chaining
  - Backward chaining

- Production systems
  - Problem: Conflict resolution
Planning

• **Find a sequence of actions** that lead to a goal
  – Much like path search, but for very large/complex domains
  – Need to represent the dynamics of the world
• **Two basic approaches** planning problem representation:
  – **Situation calculus**
    • Explicitly represents situations (extends FOL)
    • **Solving:** theorem proving
    • **Frame problem**
  – **STRIPS**
    • Add and delete list
    • Solves the frame problem
    • **Solving:** (goal progression, goal regression)

Planning

• **Divide and conquer approach**
  – Sussman’s anomaly

• **State space vs. plan space search**
  – Search the state space or search the space of plans that are gradually built

• **Partial order (non-linear) planners:**
  – Search the space of partially build plans

• **Hierarchical planners**
Uncertainty

• Basics of probability:
  – random variable, values, probability distribution

• Joint probability distribution
  – Over variables in a set, full joint over all variables
  – Marginalization (summing out)

• Conditional probability distribution
  \[ P(A | B) = \frac{P(A, B)}{P(B)} \quad \text{s.t.} \quad P(B) \neq 0 \]

• Product rule \[ P(A, B) = P(A | B)P(B) \]

• Bayes rule
  \[ P(A | B) = \frac{P(B | A)P(A)}{P(B)} \]

Full joint probability distribution
  – Over variables in a set, full joint over all variables

Two important things to remember:
• Any probabilistic query can be computed from the full joint distribution
• Full joint distribution can be expressed as a product of conditionals via the chain rule
Bayesian belief networks

- **Full joint distribution** over all random variables defining the domain can be very large
  - Complexity of a model, inferences, acquisition
- **Solution:** Bayesian belief networks (BBNs)

- **Two components of BBNs:**
  - Structure (directed acyclic graph)
  - Parameters (conditional prob. distributions)
- **BBN** build upon conditional independence relations:
  \[ P(A, B \mid C) = P(A \mid C)P(B \mid C) \]

- **Joint probability distribution for BBNs:**
  - Product of local (variable-parents) conditionals
  \[ P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i \mid pa(X_i)) \]

Bayesian belief networks

- **More compact model of the joint distribution:**
  - Reduction in the number of parameters
- **Inferences:**
  - Queries on joint probabilities
  - Queries on conditionals expressed as ratios of joint probabilities
  - Joint probabilities can be expressed in terms of full joints
  - Full joints are product of local conditionals
- **Smart way to do inferences:**
  - Interleave sums and products (variable elimination)
Decision-making in the presence of uncertainty

- **Decision tree:**
  - Decision nodes (choices are made)
  - Chance nodes (reflect stochastic outcome)
  - Outcomes (value) nodes (value of the end-situation)

- **Rational choice:**
  - Decision-maker tries to optimize the expected value

- **Use utilities to define the rational choice:**
  - Utility (or expected utility) is typically different from the expected value under uncertainty;
  - Example: the utility function for the risk-averse investor differs from the expected value

---

Machine learning

- **Types of machine learning:**
  - Supervised
  - Unsupervised
  - Reinforcement learning

- **Typical learning:**
  - Find a model with parameters to fit the data
  - Optimize the parameters to assure the best fit
  - **Different error criteria:**
    - Mean squared error
    - Likelihood of data
Machine learning

- **Simple learning problem:**
  - A model of a biased coin
  - \( \theta = P(\text{outcome} = \text{head}) \)
  - \( P(\text{outcome} = \text{tail}) = 1 - P(\text{outcome} = \text{head}) = 1 - \theta \)

- **Maximum likelihood estimate the parameter**
  - calculated from data (observed sequence of outcomes)
  
  \[
  \theta_{ML} = \frac{N_1}{N} = \frac{N_1}{N_1 + N_2}
  \]
  - \( N_1 \) – number of heads seen, \( N_2 \) – number of tails seen

- **Learning parameters of the BBN**
  - Convert to many simple (coin) learning problems

---

**Discriminative classification models**

- **A classification model is defined using**
  - discriminant functions

- **Idea:**
  - For each class \( i \) define a function \( g_i(x) \) mapping \( X \rightarrow \mathbb{R} \)
  - When the decision on input \( x \) should be made choose the class with the highest value of \( g_i(x) \)

  \[
  \text{class} = \arg \max_i g_i(x)
  \]
Classification models

• Discriminative models
  – discriminative function learned directly
  – Logistic regression
    \[ g_1(x) = g(w^Tx + w_0) \]
    \[ g_0(x) = 1 - g(w^Tx + w_0) \]
    where \( g(z) = 1/(1 + e^{-z}) \)
  – Support vector machines
    \[ g_1(x) = w^Tx + w_0 \]
    \[ g_0(x) = -(w^Tx + w_0) \]

• Generative models
  – Model and learn
    \[ p(x, y) = p(y)p(x | y) \]
  – Make decision by calculating
    \[ p(y | x) \propto p(y)p(x | y) \]
  – Example: Naïve Bayes model