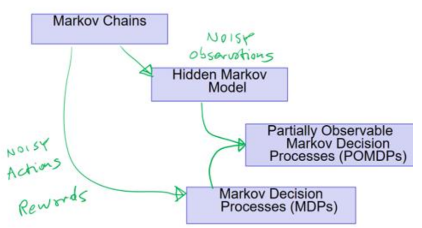
Intro and Markov Models

- value of information: EV(x is known) - EV(x is unknown) *[known in this case means an arc from x to a decision variable]*
- value of control: EV(x is a decision variable) - EV(x is a random variable)
- **limitation of decision networks:**
  - represents only a fixed # of decisions
- **advantage of markov models:**
  - network can extend indefinitely

Markov Models



A stationary Markov Chain : for all t >0

- $P(S_{t+1} | S_0,\dots,S_t) = P(S_{t+1}|S_t)$ and Markov assumption
- $P(S_{t+1}|S_t)$ is the same stationary

So we only need to specify? i-clicker.

A. $P(S_{t+1}|S_t)$ and $P(S_0)$ **B.** $P(S_0)$

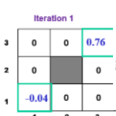
Value Iteration

Value iteration works by producing successive approximations of the optimal value function.

$$\forall s:\ V^{(k+1)}(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s,a) V^{(k)}(s')$$

What is the complexity of each iteration?

A. $O(|A|^2|S|)$ **B.** $O(|A||S|^2)$ **C.** $O(|A|^2|S|^2)$



$$V^{(1)}(3,3) = -0.04 + 1 * \max \begin{bmatrix} 0.8V^{(0)}(3,3)+0.1V^{(0)}(2,3)+0.1V^{(0)}(4,3) \\ 0.8V^{(0)}(2,3)+0.1V^{(0)}(3,3)+0.1V^{(0)}(3,2) \\ 0.8V^{(0)}(3,2)+0.1V^{(0)}(2,3)+0.1V^{(0)}(4,3) \\ 0.8V^{(0)}(4,3)+0.1V^{(0)}(3,3)+0.1V^{(0)}(3,2) \end{bmatrix} \begin{matrix} UP \\ LEFT \\ DOWN \\ RIGHT \end{matrix}$$

Markov Decision Processes

- set of states S, actions A, transition probabilities and reward function

Decision Processes

Often an agent needs to go beyond a fixed set of decisions – Examples?

- Would like to have an **ongoing decision process**

Infinite horizon problems: process does not stop
Robot surviving on planet, Monitoring Nuc. Plant, ...

Indefinite horizon problem: the agent does not know when the process may stop
reaching location

Finite horizon: the process must end at a give time N
in N steps

- MDPs are fully observable
  - Agent always knows which state its in
  - Uncertainty lies in its actions

FILTERING

- Policy only depends on current state

In general (Filtering): compute the posterior distribution over the current state given all evidence to date

$$P(X_t \mid e_{0:t})$$

POMDP belief states:

$$b'(s') = \alpha P(e|s') \sum_s P(s'|a,s) b(s)$$

- how to find optimal policy?
  - turn into MDP and apply V.I.
  - also approx. methods ie DDN using look ahead

Look Ahead Search for Optimal Policy

General Idea:

- **Expand the decision process for n steps into the future, that is**
  - "Try" all actions at every decision point
  - Assume receiving all possible observations at observation points
- **Result: tree of depth 2n+1** where
  - every branch represents one of the possible sequences of *n* actions and *n* observations available to the agent, and the corresponding belief states
  - The leaf at the end of each branch corresponds to the *belief state* reachable via that sequence of actions and observations – use filtering/belief-update to compute it
- **"Back Up" the utility values of the leaf nodes** along their corresponding branches, **combining it with the rewards** along that path
- **Pick the branch with the highest expected value**

Reinforcement Learning

- transition and reward model are unknown, can't exploit relation to adjacent states
  - this is addressed through **temporal difference**
    - takes an average through time for Q-values
- A belief state of an agent encodes what an agent has remembered
- Discounting means that more recent rewards are more valuable than rewards far in the future.

Q-Learning

- Search based applications' policies are evaluated as a whole, and cannot take into account local good/bad actions
- n states, m actions = m^n
- Q-Learning learns after every action for what to do in a given state
- Learning is linear instead of exponential with # of states

$$Q[s,a] \leftarrow Q[s,a] + \alpha((r + \gamma \max_{a'} Q[s',a']) - Q[s,a])$$

- For the approximation in Q-Learning to work, try each action an unbounded # of times
  - Unlikely transitions will be observed much less frequently than likely ones
- Any strategy should be greedy in the limit of infinite exploration **(GLIE)**
  - Choose the predicted best action in the limit, and try an unbounded # of times

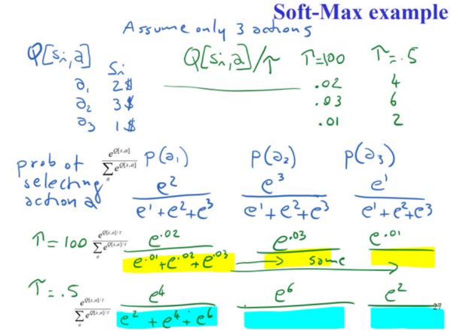
E-greedy

- choose random action with probability E, choose best action with probability 1-E
- reduce E overtime to predict best action in GLIE

Soft-max

- High Tau = exploration
- Low Tau = exploitation

Soft-Max example



On and Off-Policy learning

- If an agent is not deployed it should do random all the time and Q-Learning
  - Deploy once Q values have converged
- If an agent is deployed use one of the explore/exploit strategies and do SARSA
- Q-Learning is off-policy; it focuses on the optimal policy
  - On-policy learning addresses problem of high risk exploration and can revise policy

Hoeffding's inequality

You can tolerate an error greater than 0.1 only in 5% of your cases

Set $\varepsilon = 0.1$, $\delta = 0.05$

Equation (1) gives you n > 184

$$n > \frac{-\ln\frac{\delta}{2}}{2\varepsilon^2} \qquad (1)$$

$$2e^{-2n\varepsilon^2} < \delta$$

MCMC – why is it called MCMC?

- states of the chain are possible samples

Theorem: chain approaches stationary distribution:
long-run fraction of time spent in each state is exactly proportional to its posterior probability ..given the evidence

Filtering

$$P(X_1|e_1) = \alpha P(e_1|X_1) * \sum_{x_0} P(X_1|x_0) P(x_0)$$

$$= \alpha\langle 0.8, 0.3\rangle * (0.5\langle 0.7, 0.3\rangle + 0.5\langle 0.4, 0.6\rangle)$$

Backwards

$$P(e_3|X_2) = \sum_{x_3} P(e_3|x_3) P(|x_3) P(x_3|X_2)$$

$$= (0.8*1*\langle 0.7, 0.4\rangle) + (0.3*1*\langle 0.3, 0.6\rangle)$$

Smoothing

$P(X_k \mid e_{0:t}) = P(X_k \mid e_{0:k}, e_{k+1:t})$ dividing up the evidence

$= \alpha\, P(X_k \mid e_{0:k})\, P(e_{k+1:t} \mid X_k, e_{0:k})$ using Bayes Rule

$= \alpha\, P(X_k \mid e_{0:k})\, P(e_{k+1:t} \mid X_k)$ By Markov assumption on evidence

The backwards phase is initialized with making an unspecified observation e_{t+1} at t+ 1......

$b_{t+1:t} = P(e_{t+1} \mid X_t) = P(\mathit{unspecified} \mid X_t) = 1$

Viterbi and Approximate inference in Temporal Models (Particle Filtering)

$$\max_{x_{1:t-1}} P(x_{1:t-1}, X_t, e_{1:t-1}) =$$

$$P(e_t|X_t) \max_{x_{t-1}} \left(P(X_t|x_{t-1}) \max_{x_{1:t-2}} P(x_{1:t-2}, x_{t-1}, e_{1:t-1}) \right)$$

$$\langle 0.2, 0.7\rangle \langle \max(0.7*0.76522, 0.4*0.23478), \max(0.3*0.76522, 0.6*0.23478)\rangle$$

Viterbi complexity:

- T = # of time slices, S = # of states
- Time Complexity: $O(T S^2)$
- Space Complexity: $O(T S)$

- Limitations of exact algorithms
- HMM has very large # of states
  - Algorithms do not scale up
    - Use **Approximate inference**
      - **Getting N samples is faster than computing the right answer (via filtering)**
    - **Particle filtering**: run all N samples together through the network one slice at a time

1. Generate a population on N initial-state samples by sampling from initial state distribution $P(x_0)$ **2.** Propagate each sample for x_t forward by sampling the next state value x_{t+1} based on $P(x_{t+1} | x_t)$ **3.** Weight each sample by the likelihood it assigns to the evidence **4.** Resample the population so that the probability that each sample is selected is proportional to its weight **5.** Start this cycle again with this new sample

HMM Application

Bioinformatics: Gene Finding

- *States:* coding / non-coding region x x ✓ ✓ ✓ x x
- *Observations:* DNA Sequences → A T C G G A A

For these problems the critical inference is:

find the most likely sequence of states given a sequence of observations Viterbi Algo

CPSC 322, Lecture 32

Graphical Models

- Variable elimination algorithm for Bnets and Markov networks are the same!

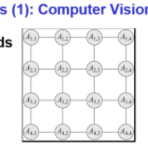
A **CRF** models $P(y_1 \dots y_k | x_1 \dots x_k)$

- special case of Markov Networks where x_i's are all observed
- it is an undirected graphical model whose nodes = X U Y
  - logistic regression is a naïve markov model
  - number of param w_l is linear instead of exponential the # of parents
  - natural model

Markov Networks Applications (1): Computer Vision

Called **Markov Random Fields**

- Stereo Reconstruction
- Image Segmentation
- Object recognition



Typically **pairwise MRF**

- Each *vars* correspond to a *pixel* (or *superpixel*)
- Edges (factors) correspond to interactions between adjacent pixels in the image
  - E.g., in segmentation: from generically penalize discontinuities, to road under car

Propositional logic and Resolution

Validity and satisfiability

A sentence is valid if it is true in all interpretations
e.g., $True$, $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$

Validity is connected to inference via the Deduction Theorem:
$KB \models \alpha$ if and only if $(KB \Rightarrow \alpha)$ is valid

A sentence is satisfiable if it is true in some interpretation
e.g., $A \vee B$, C

A sentence is unsatisfiable if it is true in no interpretations
e.g., $A \wedge \neg A$

Satisfiability is connected to inference via the following:
$KB \models \alpha$ if and only if $(KB \wedge \neg\alpha)$ is unsatisfiable
i.e., prove α by *reductio ad absurdum*

- Resolution algorithm tries to prove KB |= a

- Convert to CNF, apply resolution
- Until two clauses resolves into an empty clause

-> contradiction, therefore unsatisfiable

- No new clauses can be added; no contradiction

Walksat

- Start from randomly generated implementation
  - Pick a proposition/atom to flip by
    - 1) Randomly
    - 2) To minimize # of unsatisfied clauses
- problems get hard as #clauses/#symbols ↑ (~4.3)
- resolution procedure can be generalized to **FOL**
  - every formula can be rewritten in a logically equivalent CNF form
    - additional rewriting rules for quantifiers
    - variables need to be unified (like datalog)

Wordnet – database containing lexical relations

- To compute similarities between two senses in WordNet, use:
  - the distance between the two concepts in the underlying hierarchies/graphs
  - the glosses of the concepts
- **lesk** can be disadvantageous because most glosses are short and don't have overlaps

Path-length sim based on is-a/hypernyms hierarchies

$$\text{sim}_{\text{path}}(c_1,c_2)=1/\,pathlen(c_1,c_2)$$

- comparing two words would be semantically ambiguous - comparing two synsets we are confident it's close in distance/overlap

Relations among words and their meanings (*paradigmatic*)

Internal structure of individual words (*syntagmatic*)

FrameNet – database containing frames and syntactic/semantic argument structures

- depth vs breadth vs density (breadth and density = understanding)

Concept Distance: info content

- **Similarity should be proportional to the information that the two concepts share… what is that?**

P(root) = 1

The lower the concept/sense the lower its prob.

probability $P(c)=\dfrac{count(c)}{N}$ $IC(c)=-\log P(c)$ Information Content $LCS(c_1,c_2)$ Lowest Common Subsumer

$$\text{sim}_{\text{resnik}}(c_1,c_2)=-\log P(LCS(c_1,c_2))$$

$$dist_{JC}(c_1,c_2)=2\times\log P(LCS(c_1,c_2))-(\log P(c_1)+\log P(c_2))$$

Treebanks

- DEF. corpora in which each sentence has been paired with a parse tree
- These are generally created
  - Parse collection with parser
  - human annotators revise each parse
- Requires detailed annotation guidelines
  - POS tagset
  - Grammar
  - instructions for how to deal with particular grammatical constructions.

WS Distributional Methods (2)

- More informative values (referred to as weights or measure of association in the literature)
- Point-wise Mutual Information

$$assoc_{PMI}(w,w_i)=\log_2\frac{P(w,w_i)}{P(w)P(w_i)}$$

P(ω) P(ωᵢ) if independent, unrelated

- t-test

$$assoct_{t\text{-}test}(w,w_i)=\frac{P(w,w_i)-P(w)P(w_i)}{\sqrt{P(w)P(w_i)}}$$

CPSC 422, Lecture 24

Context Free Grammar

CFGs

- Define a Formal Language (un/grammatical sentences)
- Generative Formalism
  - Generate strings in the language
  - Reject strings not in the language
  - Impose structures (trees) on strings in the language

Top-Down vs. Bottom-Up

Top-down
- Only searches for trees that can be answers +
- But suggests trees that are not consistent with the words −

Bottom-up
- Only forms trees consistent with the words +
- Suggest trees that make no sense globally −

- CFGs cover most syntactic structures in English
- vanilla PCFGs assume independence of non-terminal expansions - **Invalid assumption!**
  - structural/lexical dependencies
    - syntactic subject of a sentence as a pronoun
    - Jim likes dogs. But he lost one as a child.
    - **solution:** split non-terminals
      - add lexical dependencies to the scheme
        - infiltrate influence of particular words into the prob. of the rules

CKY Algorithm

Definitions

- $w_1 \dots w_n$ an input string composed of n words
- w_{ij} a string of words from word i to word j
- $\mu[i, j, A]$: a table entry holds the maximum probability for a constituent with non-terminal A spanning words $w_i \dots w_j$

Markov Logic

Prob. Rel. Models vs. Markov Logic

PRM
- Relational Skeleton
- Dependency Graph
- Parameters (CPT)

=> BNET

ML
- weighted logical formulas
- set of constants

=> MARKOV LOGIC NETWORK

Markov Logic: Intuition(1)

- A logical KB is a set of **hard constraints** on the set of possible worlds

$\forall x\ Smokes(x) \Rightarrow Cancer(x)$

Individuals = {a, b}

Smokes(a) = T
Cancer(a) = F
Smokes(b) = F
Cancer(b) = F

in FOL ω is impossible

- Let's make them **soft constraints**: When a world violates a formula, the world becomes less probable, not impossible

i>clicker.

if f is True → P(ω) decreases
False → P(ω) increases

CPSC 322, Lecture 29

Markov Logic: Definition

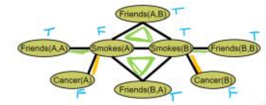
- A Markov Logic Network (MLN) is
  - a set of pairs (F, w) where
    - F is a **formula** in first-order logic
    - w is a **real number**
  - Together with a set C of **constants**,
- It defines a **Markov network** with
  - One *binary node* for each **grounding** of each **predicate** in the MLN
  - One *feature/factor* for each **grounding** of each **formula** F in the MLN, with the corresponding weight w

Grounding: substituting vars with constants

- prob. of possible world:

$$P(x)=\frac{1}{Z}\exp\left(\sum_i w_i f_i(x)\right)$$

Weight of Feature i — Feature i



$P(pw)=(e^{1}\cdot e^{1}\cdot e^{0}\cdot e^{0}\cdot e^{1.5}\cdot e^{1.5}\cdot e^{0})/Z$

First order logic (with some mild assumptions) is a special Markov Logics obtained when
- all the weight are equal
- and tend to infinity

Inference in MLN

- MLN acts as a template for a Markov Network
- We can always answer prob. queries using standard Markov network inference methods on the instantiated network
- **However**, due to the size and complexity of the resulting network, this is often infeasible.
- Instead, we combine **probabilistic methods** with ideas from **logical inference**, including **satisfiability** and **resolution**.
- This leads to efficient methods that take full advantage of the logical structure.

Finding truth assignment that maximizes sum of weights of satisfied formulas: $\arg\max_{pw}\sum_i w_i n_i(pw)$

-> this is just a weighted MaxSAT problem!

- instead of minimizing, maximize # of satisfied formulas

Entity Resolution (ML application)

- determining which observations correspond to the same real-world objects
  - can perform *collective* entity resolution, where resolving one pair of entities can help resolve other relevant entities

Statistical Parsing Representation: Summary

- For each rule of the form A → B C: Formula of the form `B(i,j) ^ C(j,k) => A(i,k)` E.g.: `NP(i,j) ^ VP(j,k) => S(i,k)`
- For each rule of the form A → a: Formula of the form `Token(a,i) => A(i,i+1)` E.g.: `Token("pizza", i) => N(i,i+1)`
- For each nonterminal: state that exactly one production holds (solve problem 1)
- Mutual exclusion rules for each part of speech pair (solve problem 2)

PRMs

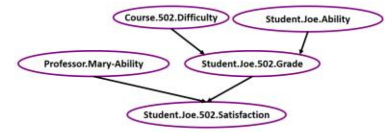
- The representation of PRMs is a direct mapping from that of relational databases

Motivation for PRMs

- Most real-world data are stored in relational DBMS
- Combine advantages of relational logic & Bayesian networks:
  - natural domain modeling: objects, properties, relations;
  - generalization over a variety of situations;
  - compact, natural probability models.
- Integrate uncertainty with relational model:
  - properties of domain entities can depend on properties of related entities;
  - uncertainty over relational structure of domain.

Limitations of Bayesian Networks

A Bayesian networks (BNs) represents a pre-specified set of attributes/variables whose relationship to each other is fixed in advance.



How PRMs extend BNs?

1. PRMs conceptually extend BNs to allow the specification of a probability model for classes of objects rather than a fixed set of simple attributes
2. PRMs also allow properties of an entity to depend probabilistically on properties of other related entities

- Each class has a set of attributes + primary keys
  - A set of **reference slots** (corresponds to foreign keys) *ie. Course.instructor -> Professor.name*
  - An instance has values for each attribute and reference slot (fixed vs probabilistic)
    - ie. name, id vs ranking, grade
    - Skeleton is a partial specification of an instance (doesn't have probabilistic att.)
      - Completion I of the skeleton is a possible world, specifies values for probabilistic attributes
- A student is registered in many courses - student is a field in registration, Registered_in() is an **inverse slot** of a reference slot

Slot-chain:*Student.Registered_in.Course.Instructor*

- PRM contains a CPD for each attribute
  - The parameters in all these CBDs comprise θ
  - Problem: There are variable # of parents
    - Solution: use aggregation
      - Example: a pro gamer's ELO can depend on average KDA
- Given a skeleton, apply local cond. probabilities to define a Joint prob. dist. over all completions of the skeleton
- Disallow uncertainty over the relational structure of the model because of our skeleton structure x
  - Parameter sharing/CPT reuse also found in temporal models (stationary assumption!)
  - We must ensure our probabilistic dependencies are acyclic
    - This is guaranteed by some prior knowledge about the domain
      - Male chromosome from a person