Intro to Probabilistic Relational Models

James Lenfestey, with Tom Temple and Ethan Howe

Outline

- Motivate problem
- Define PRMs
- Extensions and future work

Our Goal

- Observation: the world consists of many distinct entities with similar behaviors
- Exploit this redundancy to make our models simpler
- This was the idea of FOL: use quantification to eliminate redundant sentences over ground literals

Example: A simple domain

- a set of students, $\mathcal{S} = \{s_1, s_2, s_3\}$
- a set of professors, $\mathcal{P} = \{p_1, p_2, p_3\}$
- `Well-Funded`, `Famous` : $\mathcal{P} \longrightarrow \{true, false\}$
- `Student-Of` : $\mathcal{S} \times \mathcal{P} \longrightarrow \{true, false\}$
- `Successful` : $\mathcal{S} \longrightarrow \{true, false\}$

Example: A simple domain

We can express a certain self-evident fact in one sentence of FOL:

$$\forall s \in \mathcal{S} \quad \forall p \in \mathcal{P}$$
$$\texttt{Famous}(p) \text{ and } \texttt{Student-Of}(s, p)$$
$$\Rightarrow \texttt{Successful}(s)$$

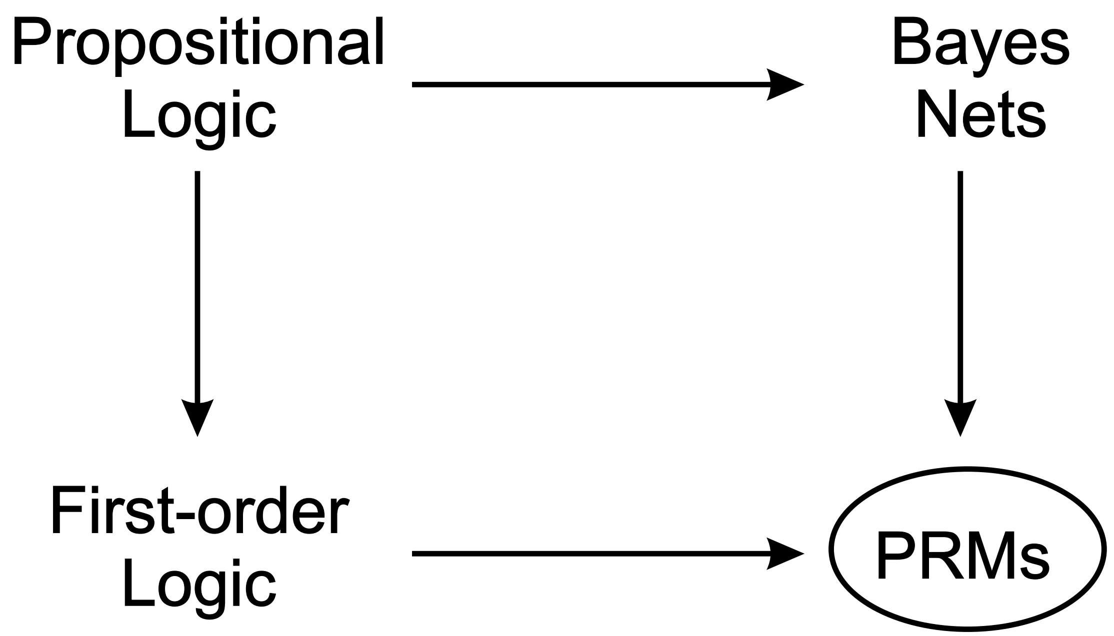
Example: A simple domain

The same sentence converted to propositional logic:

$$(\neg(p_1\_famous \text{ and } student\_of\_s_1\_p_1) \text{ or } s_1\_successful) \text{ and}$$
$$(\neg(p_1\_famous \text{ and } student\_of\_s_2\_p_1) \text{ or } s_2\_successful) \text{ and}$$
$$(\neg(p_1\_famous \text{ and } student\_of\_s_3\_p_1) \text{ or } s_3\_successful) \text{ and}$$
$$(\neg(p_2\_famous \text{ and } student\_of\_s_1\_p_1) \text{ or } s_1\_successful) \text{ and}$$
$$(\neg(p_2\_famous \text{ and } student\_of\_s_2\_p_1) \text{ or } s_2\_successful) \text{ and}$$
$$(\neg(p_2\_famous \text{ and } student\_of\_s_3\_p_1) \text{ or } s_3\_successful) \text{ and}$$
$$(\neg(p_3\_famous \text{ and } student\_of\_s_1\_p_1) \text{ or } s_1\_successful) \text{ and}$$
$$(\neg(p_3\_famous \text{ and } student\_of\_s_2\_p_1) \text{ or } s_2\_successful) \text{ and}$$
$$(\neg(p_3\_famous \text{ and } student\_of\_s_3\_p_1) \text{ or } s_3\_successful)$$

Our Goal

- Unfortunately, the real world is not so clear-cut
- Need a probabilistic version of FOL
- Proposal: PRMs



Defining the Schema

- The world consists of base entities, partitioned into classes $X_1, X_2, ..., X_n$
- Elements of these classes share connections via a collection of relations $R_1, R_2, ..., R_m$
- Each entity type is characterized by a set of attributes, $\mathcal{A}(X_i)$. Each attribute $A_j \in \mathcal{A}(X_i)$ assumes values from a fixed domain, $V(A_j)$
- Defines the *schema* of a relational model

Continuing the example...

We can modify the domain previously given to this new framework:

- 2 classes: $\mathcal{S}, \mathcal{P}$
- 1 relation: $\texttt{Student-Of} \subset \mathcal{S} \times \mathcal{P}$
- $\mathcal{A}(\mathcal{S}) = \{\texttt{Success}\}$
- $\mathcal{A}(\mathcal{P}) = \{\texttt{Well-Funded}, \texttt{Famous}\}$

Instantiations

An instantiation I of the relational schema defines

- a set of base entities $O^I(X_i)$ for each class X_i
  $O^{I'}(\mathcal{P}) = \{p_1, p_2, p_3\}$, $O^{I'}(\mathcal{S}) = \{s_1, s_2, s_3\}$

Instantiations

An instantiation I of the relational schema defines

- a set of base entities $O^I(X_i)$ for each class X_i
  $O^{I'}(\mathcal{P}) = \{p_1, p_2, p_3\}$, $O^{I'}(\mathcal{S}) = \{s_1, s_2, s_3\}$
- $R_i(X_1, ..., X_k) \subset O^I(X_1) \times ... \times O^I(X_k)$ for each R_i
  $\texttt{Student-Of} = \{(s_1, p_1), (s_2, p_3), (s_3, p_3)\}$

Instantiations

An instantiation I of the relational schema defines

- a set of base entities $O^I(X_i)$ for each class X_i
  $O^{I'}(\mathcal{P}) = \{p_1, p_2, p_3\}$, $O^{I'}(\mathcal{S}) = \{s_1, s_2, s_3\}$
- $R_i(X_1, ..., X_k) \subset O^I(X_1) \times ... \times O^I(X_k)$ for each R_i
  $\texttt{Student-Of} = \{(s_1, p_1), (s_2, p_3), (s_3, p_3)\}$
- values for the attributes of each base entity for each class
  $p_1.\texttt{Famous} = false$,
  $p_3.\texttt{Well-Funded} = true$,
  $s_2.\texttt{Success} = true, ...$

Slot chains

We can project any relation $R(X_1, ..., X_k)$ onto its ith and jth components to obtain a binary relation $\rho(X_i, X_j)$

Notation: for $x \in O^I(X_i)$, let

$$x.\rho = \{y \in O^I(X_j) | (x,y) \in \rho(X_i, X_j)\}$$

We call ρ a *slot* of X_i. Composition of slots (via transitive closure) gives a *slot chain*

E.g. x_1.`Student-Of`.`Famous` is the fame of x_1's adviser

Probabilities, finally

- The idea of a PRM is to express a joint probability distribution over all possible instantiations of a particular relational schema
- Since there are infinitely many possible instantiations to a given schema, specifying the full joint distribution would be very painful
- Instead, compute marginal probabilities over remaining variables given a *partial* instantiation

Partial Instantiations

A partial instantiation I' specifies

- the sets $O^{I'}(X_i)$
  $O^{I'}(\mathcal{P}) = \{p_1, p_2, p_3\}$, $O^{I'}(\mathcal{S}) = \{s_1, s_2, s_3\}$

Partial Instantiations

A partial instantiation I' specifies

- the sets $O^{I'}(X_i)$
  $O^{I'}(\mathcal{P}) = \{p_1, p_2, p_3\}$, $O^{I'}(\mathcal{S}) = \{s_1, s_2, s_3\}$
- the relations R_j
  $\texttt{Student-Of} = \{(s_1, p_1), (s_2, p_3), (s_3, p_3)\}$

Partial Instantiations

A partial instantiation I' specifies

- the sets $O^{I'}(X_i)$
  $O^{I'}(\mathcal{P}) = \{p_1, p_2, p_3\}$, $O^{I'}(\mathcal{S}) = \{s_1, s_2, s_3\}$
- the relations R_j
  $\texttt{Student-Of} = \{(s_1, p_1), (s_2, p_3), (s_3, p_3)\}$
- values of some attributes for some of the base entities
  $p_3.\texttt{Famous} = true$, $s_1.\texttt{Success} = false$

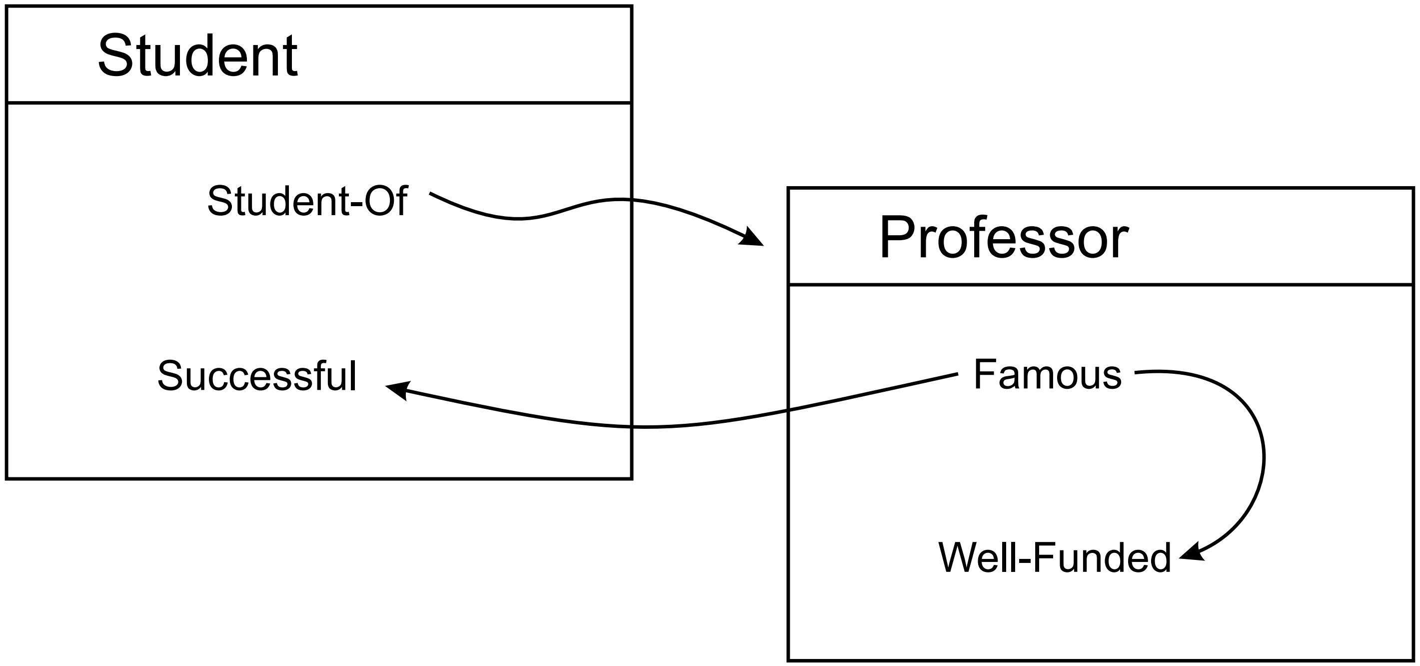
Locality of Influence

- BNs and PRMs are alike in that they both assume that real-world data exhibits *locality of influence*, the idea that most variables are influenced by only a few others
- Both models exploit this property through *conditional independence*
- PRMs go beyond BNs by assuming that there are few distinct patterns of influence in total

Conditional independence

- For a class X, values of the attribute $X.A$ are influenced by attributes in the set $Pa(X.A)$ (its parents)
- $Pa(X.A)$ contains attributes of the form $X.B$ (B an attribute) or $X.\tau.B$ (τ a slot chain)
- As in a BN, the value of $X.A$ is conditionally independent of the values of all other attributes, given its parents

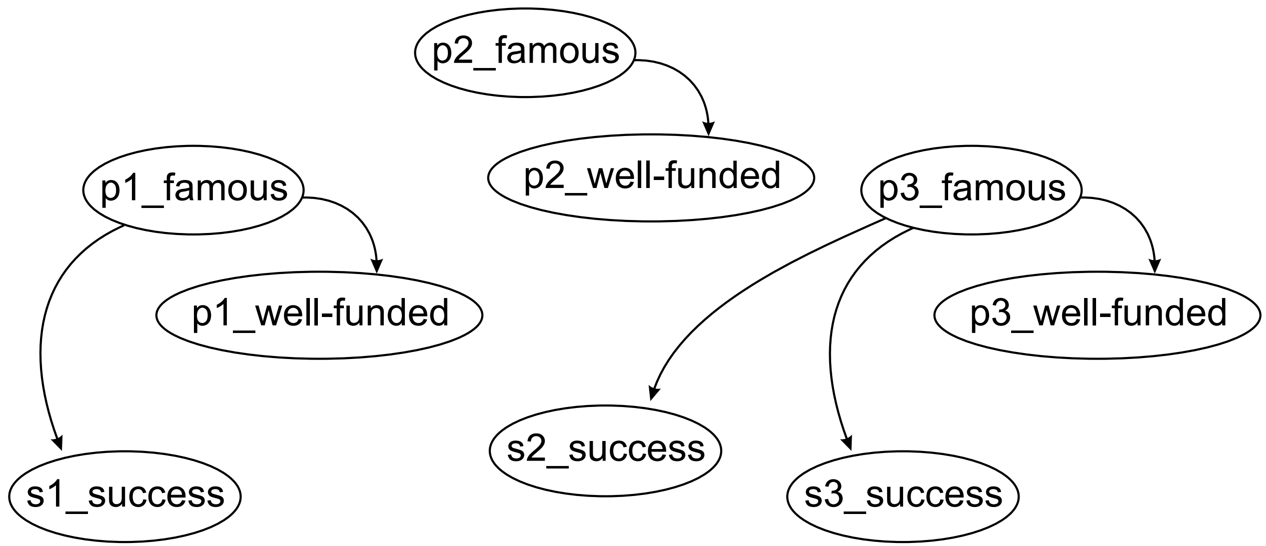
An example



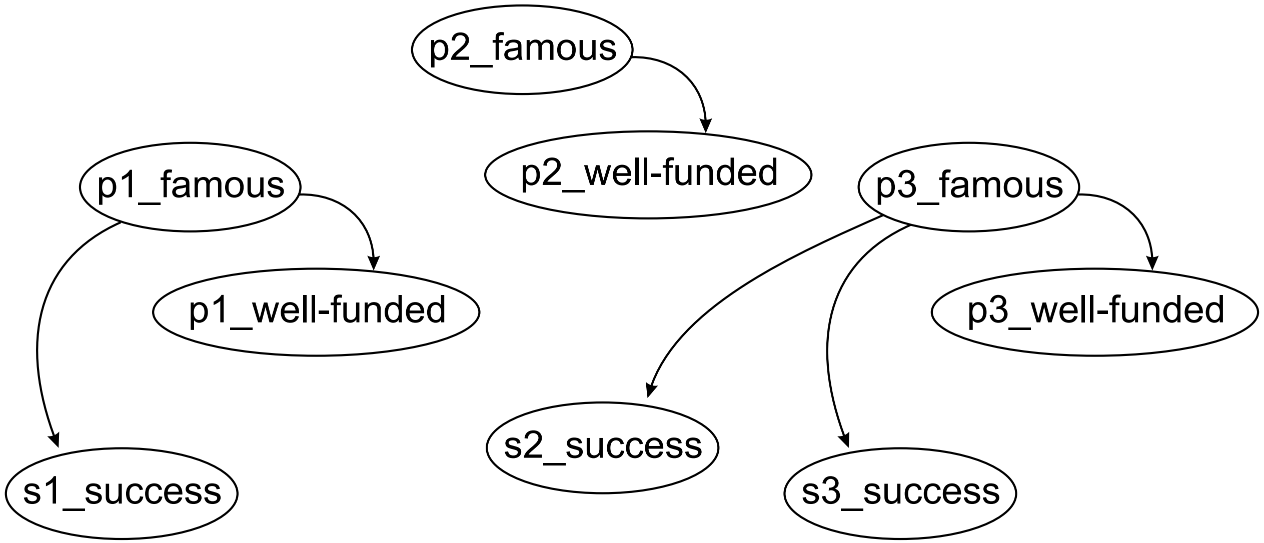
Captures the FOL sentence from before in a probabilistic framework.

Compiling into a BN

A PRM can be compiled into a BN, just as a statement in FOL can be compiled to a statement in PL



PRM



We can us this network to support inference over queries regarding base entities

Aggregates

- $Pa(X.A)$ may contain $X.\tau.B$ for slot chain τ, which is generally a multiset.
- $Pa(X.A)$ dependent on the value *of* the set, not just the values *in* the multiset
- Representational challenge, again $|X.\tau.B|$ has no bound a priori

Aggregates

- γ summarizes the contents of $X.\tau.B$
- Let $\gamma(X.\tau.B)$ be a parent of attributes of X
- Many useful aggregates: mean, cardinality, median, etc
- Require computation of γ to be deterministic (we can omit it from the diagram)

Example: Aggregates

- Let $\gamma(A) = |A|$
- Let $\texttt{Adviser-Of} = \texttt{Student-Of}^{-1}$
- e.g. $p_1.\texttt{Adviser-Of} = \{s_1\}$,
  $p_2.\texttt{Adviser-Of} = \{\}$,
  $p_3.\texttt{Adviser-Of} = \{s_2, s_3\}$
- To represent the idea that a professor's funding is influenced by the number of advisees:

$$Pa(\mathcal{P}.\texttt{Well-Funded}) =$$
$$\{\mathcal{P}.\texttt{Famous}, \gamma(\mathcal{P}.\texttt{Adviser-Of})\}$$

Extensions

- Reference uncertainty. Not all relations known a priori; may depend probabilistically on values of attributes. E.g., students prefer advisers with more funding
- Identity uncertainty. Distinct entities might not refer to distinct real-world objects
- Dynamic PRMs. Objects and relations change over time; can be unfolded into a DBN at the expense of a very large state space

Acknowledgements

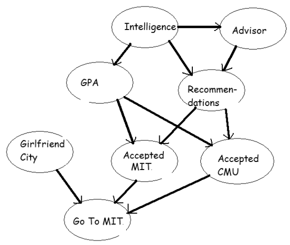
- "Approximate inference for first-order probabilistic languages", Pasula and Russell. Running example.
- "Learning Probabilistic Relational Models", Friedman et al. Borrowed notation.

Resources

- “Approximate inference for first-order probabilistic languages” gives a promising MCMC approach for addressing relational and identity uncertainty.
- “Inference in Dynamic Probabilistic Relational Models”, Sanhai et al. Particle-filter based DPRM inference that uses abstraction smoothing to generalize over related objects.

Extensions of Bayesian Networks

Ethan Howe,
James Lenfestey,
Tom Temple



Outline

- Intro to Dynamic Bayesian Nets (Tom)
- Exact inference in DBNs with demo (Ethan)
- Approximate inference and learning (Tom)
- Probabilistic Relational Models (James)

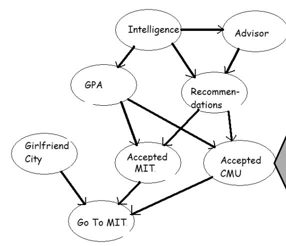
Reasoning under Uncertainty

- How do we use prior knowledge and new observations to judge what is likely and what is not?

$$P(s \mid observations, prior\ knowledge)$$

- But that is a very large joint distribution

Bayesian Network



GPA	Recs	Accepted
Low	Great	.05
Low	Good	.01
Ave	Great	.1
Ave	Good	.05
High	Great	.25
High	Good	.1

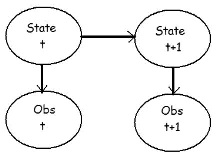
Bayesian Network

- A Bayesian Network is a Directed Acyclic Graph (DAG) with variables as nodes.
- Edges go from parent to child such that
  - Each child, x, has a conditional probability table P(x|parents(x)) that defines the affects that x feels from its parents.
- Intuitively, these edges codify *direct relationships* between nodes in a cause-effect manner.
- The lack of an edge between nodes implies that they are conditionally independent.

Features of Bayesian Networks

- Arbitrarily descriptive; allows encapsulation of all the available prior knowledge
- The model makes no distinction between observed variables and inferred variables
- The DAG restriction is somewhat limiting

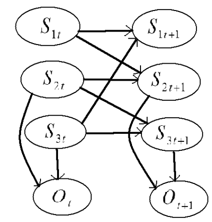
HMM as a Bayes Net



	S1	S2	S3	S4	…
S1	.8	.2	.2	.01	…
S2	.01	.5	.1	.02	…
S3	.05	.1	.6	.01	…
S4	.01	.01	.01	.9	…
…	…	…	…	…	…

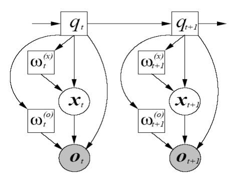
Dynamic Bayesian Networks

- DBNs are BNs with a temporal regularity analogous to the HMM
- The full representation of a DBN grows with time.



Hybrid State in DBNs

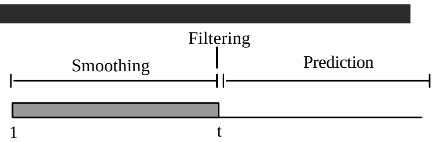
- For basis functions, parent variables are used as function parameters
- Usually the function is a mixture of Gaussians with known means and variances and the parent variables decides the mixture weights.



Exact Inference in Bayesian Networks

- Queries of the network are fielded by "summing out" all of the non-query variables
- A variable is summed out by collecting all of the factors that contain it into a new factor. Then sum over all of the possible states of the variable to be eliminated.

Temporal Inference



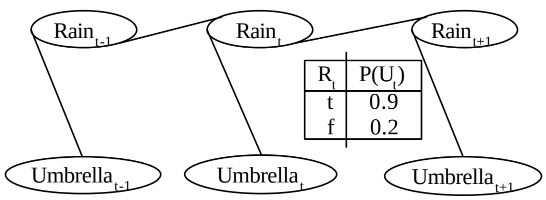
- Filtering
  - Probabilities of current states
- Prediction
  - Probabilities of future states
- Smoothing
  - Probabilities of past states

Notation

- $\mathbf{X}_{t+1}$: set of hidden variables in the t+1 time slice
- $\mathbf{x}_{t+1}$: set of values for those hidden variables at t+1
- **$\mathbf{e}_{t+1}$: set of evidence at time t+1**
- **$\mathbf{e}_{1:t}$: set of evidence from all times from 1 to t**
- **a: normalization constant**

Toy Example

R_{t-1}	$P(R_t)$
t	0.7
f	0.3



R_t	$P(U_t)$
t	0.9
f	0.2

Markov Assumptions

- First-order Markov process
  - $P(\mathbf{X}_t|\mathbf{X}_{0:t-1}) = P(\mathbf{X}_t|\mathbf{X}_{t-1})$
- **Markov assumption of evidence**
  - $P(\mathbf{E}_t|\mathbf{X}_{0:t},\mathbf{E}_{0:t-1}) = P(\mathbf{E}_t|\mathbf{X}_t)$

Prediction: Separating Time Slices

$P(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) = P(\mathbf{X}_{t+1}|\mathbf{e}_{1:t},\mathbf{e}_{t+1})$ (divide evidence)

$= a\ P(\mathbf{e}_{t+1}|\mathbf{X}_{t+1},\mathbf{e}_{1:t})\ P(\mathbf{X}_{t+1}|\mathbf{e}_{1:t})$ (Bayes' rule)

$= a\ P(\mathbf{e}_{t+1}|\mathbf{X}_{t+1})\ P(\mathbf{X}_{t+1}|\mathbf{e}_{1:t})$ (Markov property

of evidence)

Predict Next Time Slice from Current

$P(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1})$

$= a\ P(\mathbf{e}_{t+1}|\mathbf{X}_{t+1})\ S_{\mathbf{x}_t}\ P(\mathbf{X}_{t+1}|\mathbf{x}_t,\mathbf{e}_{1:t})\ P(\mathbf{x}_t|\mathbf{e}_{1:t})$

$= a\ P(\mathbf{e}_{t+1}|\mathbf{X}_{t+1})\ S_{\mathbf{x}_t}\ P(\mathbf{X}_{t+1}|\mathbf{x}_t)\ P(\mathbf{x}_t|\mathbf{e}_{1:t})$

Example: Umbrella Seen on Day 1

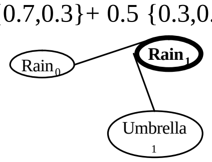
$P(R_0) = \{0.5,0.5\}$

$P(R_1|u_1) = a\ P(\mathbf{u_1}|\mathbf{R_1})\ S\ P(\mathbf{R_1}|\mathbf{r_0})\ P(\mathbf{r_0})$

$= a\ \{0.9,0.2\}\ (0.5\ \{0.7,0.3\} + 0.5\ \{0.3,0.7\})$

$= a\ \{0.45, 0.1\}$

$= \{0.818, 0.182\}$



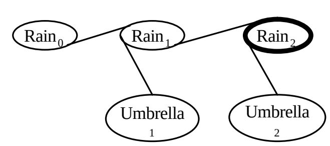
Example: Umbrella Seen Again on Day 2

$P(R_2|u_1) = S\ P(R_2|\mathbf{r_1})\ P(r_1|\mathbf{u_1})$

$P(R_2|u_1,u_2) = a\ P(u_2|\mathbf{R_2})\ P(R_2|u_1)$

$= a\{0.9, 0.2\}(0.818\{0.7,0.3\}+0.182\{0.3,0.7\})$

$= a\{0.565,0.075\} = \{0.883, 0.117\}$



Encapsulate Prediction

- Repeating actions at each step of time
- Formalize as procedure
  - $P(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1})$ = FORWARD(**previous state set, new evidence set)**
- **Only need to pass message from previous time step**
  - **$f_{1:t+1}$= a FORWARD($f_{1:t}$,e_{t+1})**
- **Can forget about previous times**

Smoothing: Break into data before and after

$P(\mathbf{X}_k|\mathbf{e}_{1:t}) = P(\mathbf{X}_k|\mathbf{e}_{1:k},\mathbf{e}_{k+1:t})$ (divide evidence)

$= a\ P(\mathbf{X}_k|\mathbf{e}_{1:k})\ P(\mathbf{e}_{k+1:t}|\mathbf{X}_k,\mathbf{e}_{1:k})$ (Bayes' rule)

$= a\ P(\mathbf{X}_k|\mathbf{e}_{1:k})\ P(\mathbf{e}_{k+1:t}|\mathbf{X}_k)$ (Markov Indep.)

$= a\ \mathbf{f}_{1:k}\,\mathbf{b}_{k+1:t}$

Backwards message

$P(\mathbf{X}_{k+1:t}|\mathbf{e}_k)$

$= S_{\mathbf{x}_{k+1}}\ P(\mathbf{e}_{k+1:t}|\mathbf{X}_k,\mathbf{x}_{k+1})\ P(\mathbf{x}_{k+1}|\mathbf{X}_k)$ (cond. on $\mathbf{X}_{k+1}$)

$= S_{\mathbf{x}_{k+1}}\ P(\mathbf{e}_{k+1:t}|\mathbf{x}_{k+1})\ P(\mathbf{x}_{k+1}|\mathbf{X}_k)$ (Markov indep.)

$= S_{\mathbf{x}_{k+1}}\ P(\mathbf{e}_{k+1},\mathbf{e}_{k+2:t}|\mathbf{x}_{k+1})\ P(\mathbf{x}_{k+1}|\mathbf{X}_k)$

$= S_{\mathbf{x}_{k+1}}\ P(\mathbf{e}_{k+1}|\mathbf{x}_{k+1})\ P(\mathbf{e}_{k+2:t}|\mathbf{x}_{k+1})\ P(\mathbf{x}_{k+1}|\mathbf{X}_k)$

Backwards procedure

- Repeating actions at each step of time
- Formalize as procedure
  - $P(\mathbf{e}_{k+1:t}|\mathbf{X}_k)$ = **BACKWARD**(next **state set, future evidence set)**
- **Message passing goes backwards in time**
  - **$b_{k+1:t}$= BACKWARD($b_{k+2:t}$,$e_{k+1:t}$)**

Example: Was it really raining day 1?

$P(u_2|R_1) = S_{r_2}\ P(u_2|r_2)\ P(u_{3:2}|r_2)\ P(r_2|R_1)$

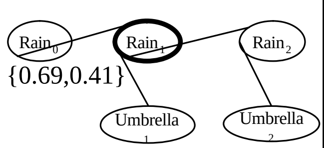
$= (0.9*1*\{0.7,0.3\})+(0.2*1*\{0.3,0.7\})$

$= \{0.69, 0.41\}$

$P(R_1|u_{1:2}) = a\ f_{1:1}\ b_{2:2}$

$= a\ \{0.818,0.182\}\ \{0.69,0.41\}$

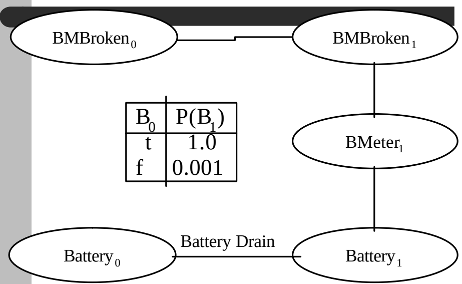
$= \{0.883,0.117\}$



Forward-Backwards Algorithm

- Save FORWARD values at each time step
- Each time new data arrives, recompute BACKWARD values.
- Time Complexity O(t)
- Space Complexity O(t)
- Both can be bounded if only care about last k slices.

Demo: Battery Meter Model



Exact Inference in DBNs

- There are some additional difficulties with exact inference in DBNs
  - The factors grow to include all the states which sharply decreases the advantage over HMMs
  - Constant memory requirement

Approximate Inference

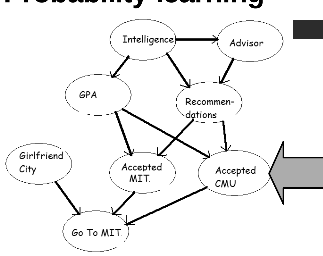
- While a number of inference algorithms exist for the standard Bayes Net, few of them adapt well to the DBN.
- One that does adapt is Particle filtering, due to its ingenuitive use of resampling
  - Particle filtering deals well with hybrid state
- Sampling has trouble with unlikely events

Learning

- The three components of a BN
  - the probabilities
  - the structure
  - the variables

can all be learn automatically, though with varying degrees of complexity

Probability learning

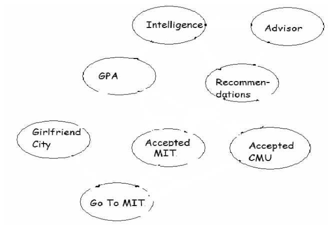


GPA	Recs	Accepted
Low	Great	??
Low	Good	??
Ave	Great	??
Ave	Good	??
High	Great	??
High	Good	??

How do we generate the Network?

- The network contains variables, edges and probabilities all of which need to be known ahead of time.
- Given all the variables and their relationships, it is relatively easy to estimate the probabilities from sample data, even if that sample data does not include every variable.

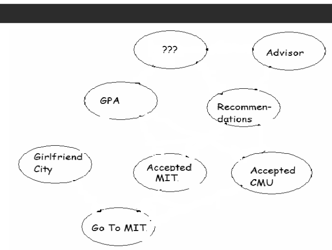
Structure Learning



Structure Learning

- Given the variables only, we can guess a structure, learn parameters for that structure and then rate it based on how well it "predicts" the data.
- Care must be taken not to overfit.
- Search the structure space with greedy-ascent on this rating
- Randomization helps avoid local maxima

Hidden Variable Learning

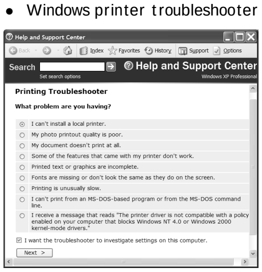


Hidden Variable Learning

- Given a subset of the variables, generate new variables, structure and parameters.
- Greedy ascent on the output of Structure learning.
- Some risk of overfit

Applications

- Windows printer troubleshooter and goddamn paper-clip



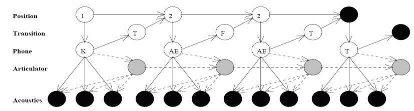
More exciting applications

- Speech and Vision recognition
  - Fewer parameters than traditional HMMs mean better learning
  - Allows for easier experimentation with intuitive (i.e. hand coded) hierarchical models

In Depth Training

- Since the DBN model doesn't make a distinction between observed variables and hidden variables, it can learn a model with access to data that we don't have during prediction.
- For example, in Speech Recognition,
  - We know that sounds are made by the pose of the lips mouth and tongue
  - While training we can measure the pose

Isolated Word Recognition



	WER	# Param.
Acoustics Only (baseline)	9.8%	31488
2 Discrete Articulatory Values	8.5%	62976
4 Discrete Articulatory Values	7.7%	126690
8 Discrete Articulatory Values	8.4%	257070

Recall a few important points

- Bayesian networks are an arbitrarily expressive model for joint probability distributions
- DBNs can be more compact than HMMs and therefore easier to learn and faster to use
- Dynamic Bayesian networks allow for reasoning in a temporal model
  - Tolerant of missing data, likewise able to exploit bonus data
  - Computationally, the compactness advantage is often lost in exact reasoning
- Particle Filters are an alternative for exploiting this compactness as long as the important probabilities are large enough to be sampled from.