Bayesian networks
Lecture 18

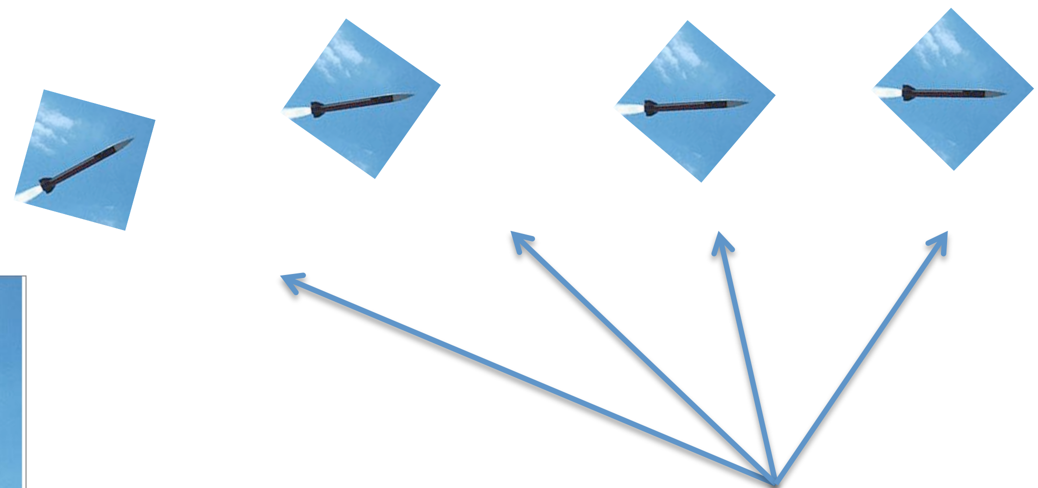
David Sontag
New York University

Outline for today

- Modeling *sequential* data (e.g., time series, speech processing) using hidden Markov models (HMMs)

- Bayesian networks
  - Independence properties
  - Examples
  - Learning and inference

Example application: Tracking



Radar

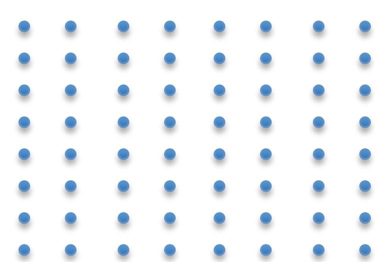
Observe noisy measurements of missile location: Y_1, Y_2, …

Where is the missile **now**? Where will it be in 10 seconds?

Probabilistic approach

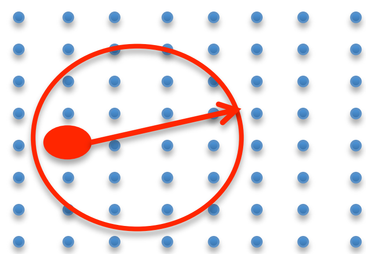
- Our measurements of the missile location were $Y_1, Y_2, \ldots, Y_n$
- Let X_t be the *true* <missile location, velocity> at time t
- To keep this simple, suppose that everything is discrete, i.e. X_t takes the values 1, …, k

Grid the space:



Probabilistic approach

- First, we specify the *conditional* distribution $Pr(X_t \mid X_{t-1})$:



From basic physics, we can bound the distance that the missile can have traveled

- Then, we specify $Pr(Y_t \mid X_t$=<(10,20), 200 mph toward the northeast>):

With probability ½, $Y_t = X_t$ (ignoring the velocity). Otherwise, Y_t is a uniformly chosen grid location

Hidden Markov models

1960's

- Assume that the **joint** distribution on $X_{1,}$ X_2, …, X_n and Y_1, Y_2, …, Y_n factors as follows:

$$\Pr(x_1, \ldots x_n, y_1, \ldots, y_n) = \Pr(x_1) \Pr(y_1 \mid x_1) \prod_{t=2}^{n} \Pr(x_t \mid x_{t-1}) \Pr(y_t \mid x_t)$$

- To find out where the missile is *now*, we do **marginal inference**:

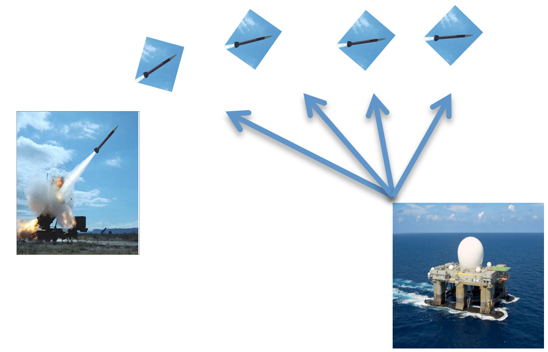
$$\Pr(x_n \mid y_1, \ldots, y_n)$$

- To find the most likely *trajectory*, we do **MAP (maximum a posteriori) inference**:

$$\arg\max_{\mathbf{x}} \Pr(x_1, \ldots, x_n \mid y_1, \ldots, y_n)$$

Inference

- Recall, to find out where the missile is now, we do marginal inference: $\Pr(x_n \mid y_1, \ldots, y_n)$



- How does one **compute** this?
- Applying rule of conditional probability, we have:

$$\Pr(x_n \mid y_1, \ldots, y_n) = \frac{\Pr(x_n, y_1, \ldots, y_n)}{\Pr(y_1, \ldots, y_n)}$$

- Naively, would seem to require k^{n-1} summations,

$$\Pr(x_n, y_1, \ldots, y_n) = \sum_{x_1, \ldots, x_{n-1}} \Pr(x_1, \ldots, x_n, y_1, \ldots, y_n)$$

Is there a more efficient algorithm?

Marginal inference in HMMs

- Use **dynamic programming**

$$\Pr(A=a) = \sum_b \Pr(B=b, A=a)$$

$$\Pr(x_n, y_1, \ldots, y_n) = \sum_{x_{n-1}} \Pr(x_{n-1}, x_n, y_1, \ldots, y_n)$$

$$\Pr(\vec{A}=\vec{a}, \vec{B}=\vec{b}) = \Pr(\vec{A}=\vec{a}) \Pr(\vec{B}=\vec{b} \mid \vec{A}=\vec{a})$$

$$= \sum_{x_{n-1}} \Pr(x_{n-1}, y_1, \ldots, y_{n-1}) \Pr(x_n, y_n \mid x_{n-1}, y_1, \ldots, y_{n-1})$$

Conditional independence in HMMs

$$= \sum_{x_{n-1}} \Pr(x_{n-1}, y_1, \ldots, y_{n-1}) \Pr(x_n, y_n \mid x_{n-1})$$

$$\Pr(A=a, B=b) = \Pr(A=a) \Pr(B=b \mid A=a)$$

$$= \sum_{x_{n-1}} \Pr(x_{n-1}, y_1, \ldots, y_{n-1}) \Pr(x_n \mid x_{n-1}) \Pr(y_n \mid x_n, x_{n-1})$$

Conditional independence in HMMs

$$= \sum_{x_{n-1}} \Pr(x_{n-1}, y_1, \ldots, y_{n-1}) \Pr(x_n \mid x_{n-1}) \Pr(y_n \mid x_n)$$

- For n=1, initialize $\Pr(x_1, y_1) = \Pr(x_1) \Pr(y_1 \mid x_1)$
- Total running time is O(nk) – linear time! Easy to do **filtering**

MAP inference in HMMs

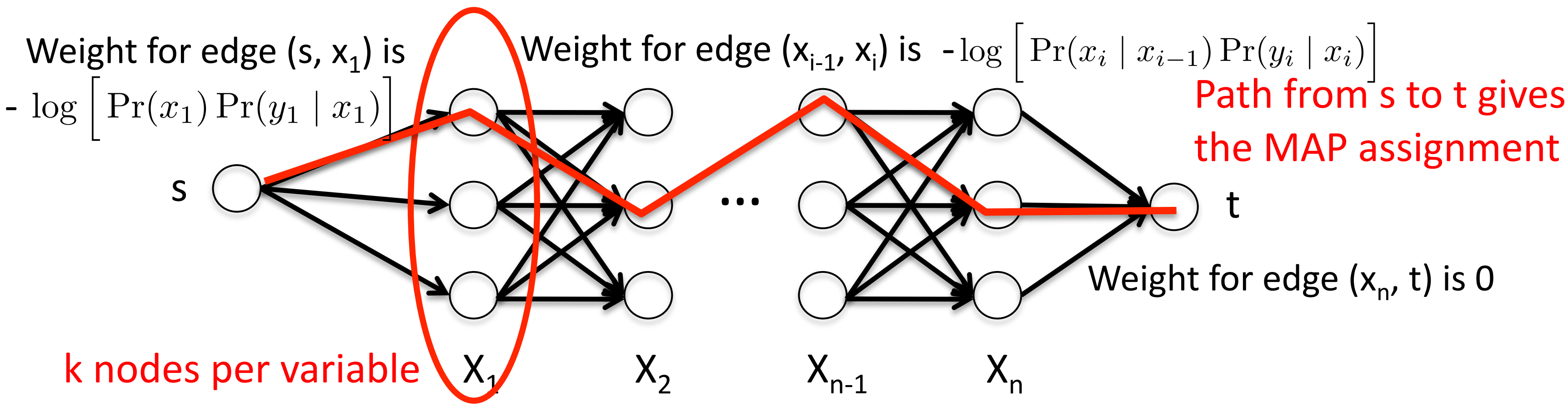
- MAP inference in HMMs can *also* be solved in linear time!

$$\arg\max_{\mathbf{x}} \Pr(x_1, \ldots x_n \mid y_1, \ldots, y_n) = \arg\max_{\mathbf{x}} \Pr(x_1, \ldots x_n, y_1, \ldots, y_n)$$

$$= \arg\max_{\mathbf{x}} \log \Pr(x_1, \ldots x_n, y_1, \ldots, y_n)$$

$$= \arg\max_{\mathbf{x}} \quad \log\Big[\Pr(x_1)\Pr(y_1 \mid x_1)\Big] + \sum_{i=2}^{n} \log\Big[\Pr(x_i \mid x_{i-1})\Pr(y_i \mid x_i)\Big]$$

- Formulate as a shortest paths problem



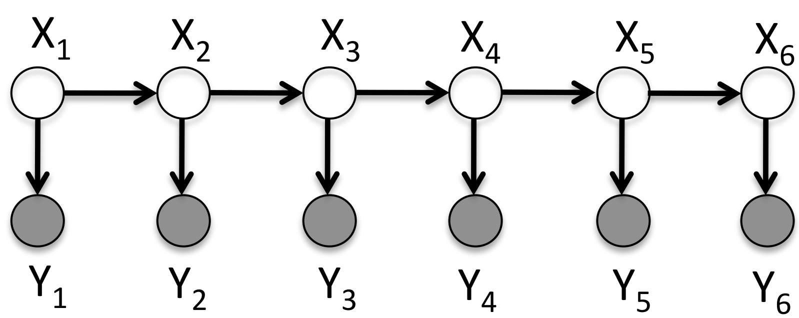
Called the Viterbi algorithm

Applications of HMMs

- Speech recognition
  - Predict phonemes from the sounds forming words (i.e., the actual signals)
- Natural language processing
  - Predict parts of speech (verb, noun, determiner, etc.) from the words in a sentence
- Computational biology
  - Predict intron/exon regions from DNA
  - Predict protein structure from DNA (locally)
- And many many more!

HMMs as a *graphical model*

- We can represent a hidden Markov model with a graph:



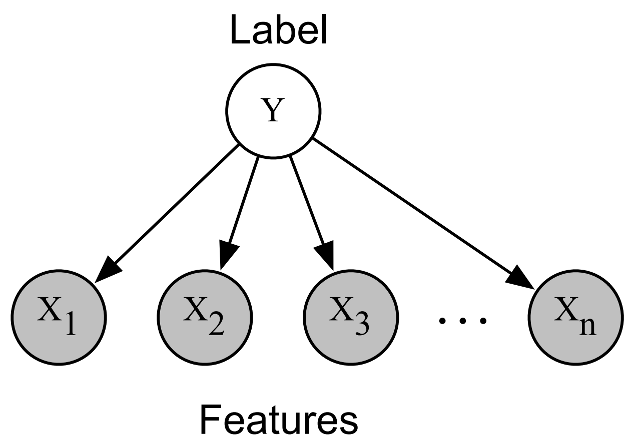
Shading in denotes *observed* variables (e.g. what is available at test time)

$$\Pr(x_1, \ldots x_n, y_1, \ldots, y_n) = \Pr(x_1) \Pr(y_1 \mid x_1) \prod_{t=2}^{n} \Pr(x_t \mid x_{t-1}) \Pr(y_t \mid x_t)$$

- There is a 1-1 mapping between the graph structure and the factorization of the joint distribution

Naïve Bayes as a *graphical model*

- We can represent a naïve Bayes model with a graph:



Shading in denotes *observed* variables (e.g. what is available at test time)

$$\Pr(y, x_1, \ldots, x_n) = \Pr(y) \prod_{i=1}^{n} \Pr(x_i \mid y)$$

- There is a 1-1 mapping between the graph structure and the factorization of the joint distribution

Bayesian networks

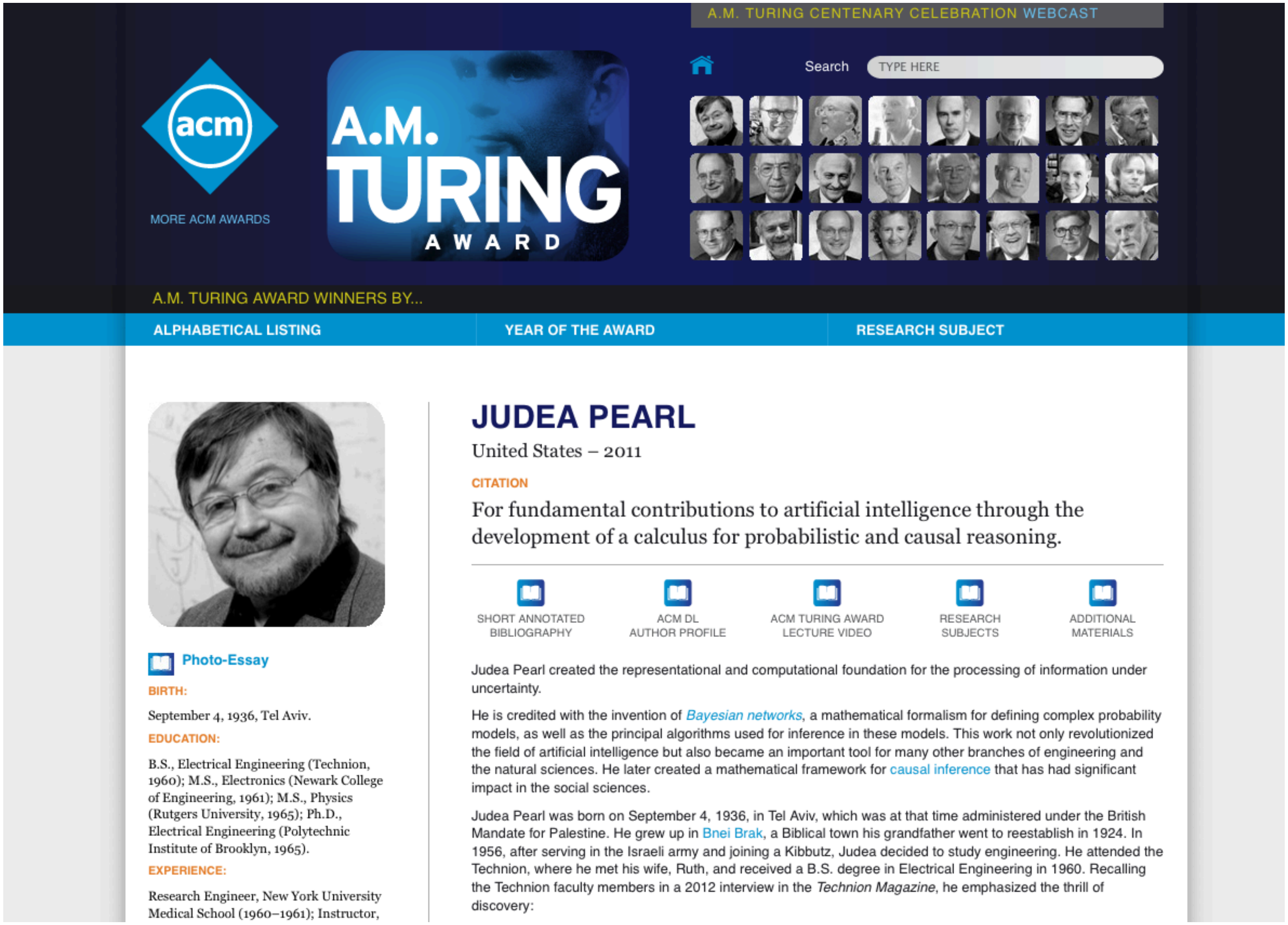
- A **Bayesian network** is specified by a directed *acyclic* graph G=(V,E) with:
  - One node *i* for each random variable X_i
  - One conditional probability distribution (CPD) per node, $p(x_i \mid \boldsymbol{x}_{Pa(i)})$, specifying the variable's probability conditioned on its parents' values

- Corresponds 1-1 with a particular factorization of the joint distribution:

$$p(x_1, \ldots x_n) = \prod_{i \in V} p(x_i \mid \mathbf{x}_{\mathrm{Pa}(i)})$$

- Powerful framework for designing *algorithms* to perform probability computations

2011 Turing award was for Bayesian networks



A.M. TURING CENTENARY CELEBRATION WEBCAST

Search TYPE HERE

MORE ACM AWARDS

A.M. TURING AWARD

A.M. TURING AWARD WINNERS BY...

ALPHABETICAL LISTING | YEAR OF THE AWARD | RESEARCH SUBJECT

JUDEA PEARL

United States – 2011

CITATION

For fundamental contributions to artificial intelligence through the development of a calculus for probabilistic and causal reasoning.

SHORT ANNOTATED BIBLIOGRAPHY | ACM DL AUTHOR PROFILE | ACM TURING AWARD LECTURE VIDEO | RESEARCH SUBJECTS | ADDITIONAL MATERIALS

Judea Pearl created the representational and computational foundation for the processing of information under uncertainty.

He is credited with the invention of *Bayesian networks*, a mathematical formalism for defining complex probability models, as well as the principal algorithms used for inference in these models. This work not only revolutionized the field of artificial intelligence but also became an important tool for many other branches of engineering and the natural sciences. He later created a mathematical framework for causal inference that has had significant impact in the social sciences.

Judea Pearl was born on September 4, 1936, in Tel Aviv, which was at that time administered under the British Mandate for Palestine. He grew up in Bnei Brak, a Biblical town his grandfather went to reestablish in 1924. In 1956, after serving in the Israeli army and joining a Kibbutz, Judea decided to study engineering. He attended the Technion, where he met his wife, Ruth, and received a B.S. degree in Electrical Engineering in 1960. Recalling the Technion faculty members in a 2012 interview in the *Technion Magazine*, he emphasized the thrill of discovery:

Photo-Essay

BIRTH:

September 4, 1936, Tel Aviv.

EDUCATION:

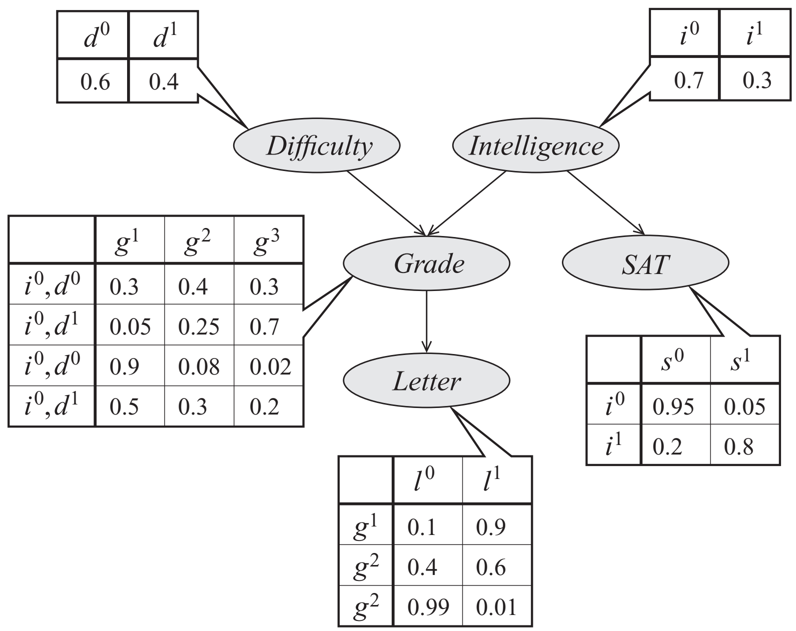
B.S., Electrical Engineering (Technion, 1960); M.S., Electronics (Newark College of Engineering, 1961); M.S., Physics (Rutgers University, 1965); Ph.D., Electrical Engineering (Polytechnic Institute of Brooklyn, 1965).

EXPERIENCE:

Research Engineer, New York University Medical School (1960–1961); Instructor,

Example

- Consider the following Bayesian network:



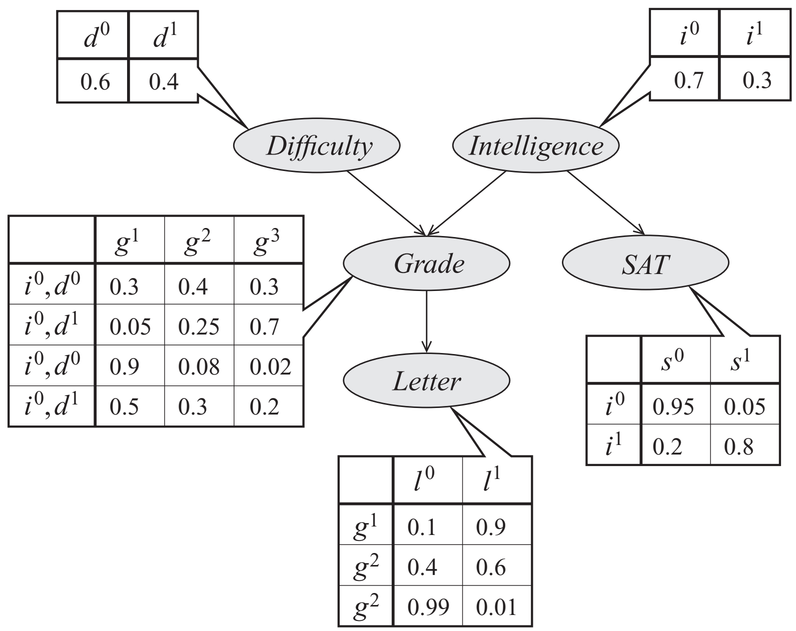
Example from Koller & Friedman, P*robabilistic Graphical Models,* 2009

- What is its joint distribution?

$$p(x_1, \dots x_n) = \prod_{i \in V} p(x_i \mid \mathbf{x}_{\mathrm{Pa}(i)})$$
$$p(d, i, g, s, l) = p(d)p(i)p(g \mid i, d)p(s \mid i)p(l \mid g)$$

Example

- Consider the following Bayesian network:



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

	g^1	g^2	g^3
i^0, d^0	0.3	0.4	0.3
i^0, d^1	0.05	0.25	0.7
i^0, d^0	0.9	0.08	0.02
i^0, d^1	0.5	0.3	0.2

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^2	0.99	0.01

Example from Koller & Friedman, P*robabilistic Graphical Models,* 2009

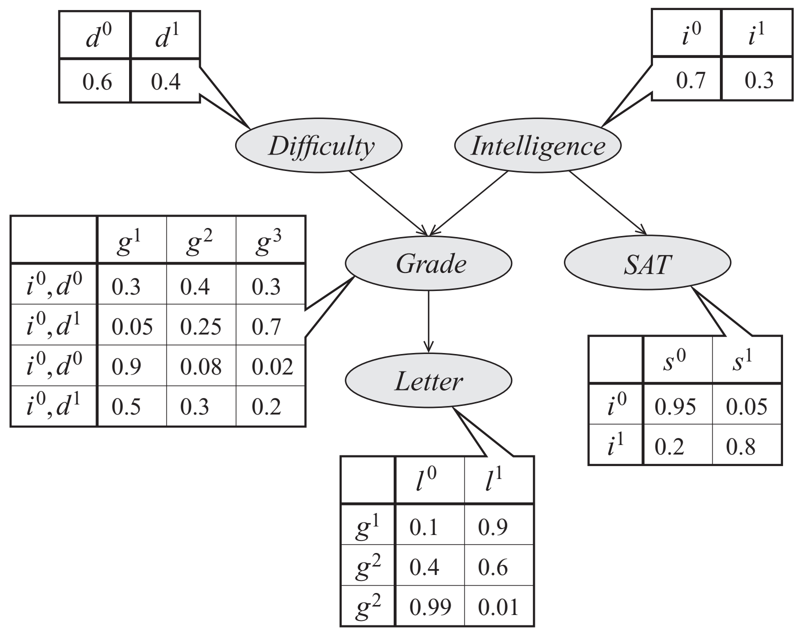
- What is this model assuming?

$$\text{SAT} \not\perp \text{Grade}$$

$$\text{SAT} \perp \text{Grade} \mid \text{Intelligence}$$

Example

- Consider the following Bayesian network:



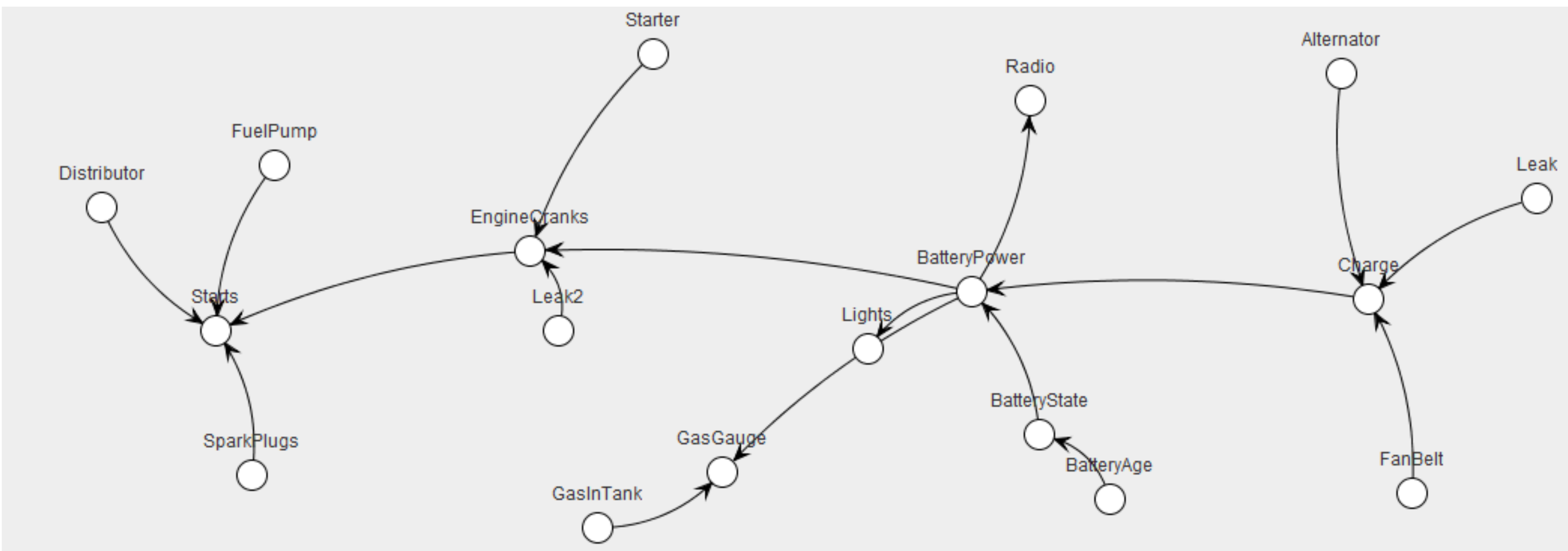
Example from Koller & Friedman, P*robabilistic Graphical Models,* 2009

- Compared to a simple log-linear model to predict intelligence:
  - Captures *non-linearity* between grade, course difficulty, and intelligence
  - *Modular*. Training data can come from different sources!
  - Built in *feature selection*: letter of recommendation is irrelevant given grade

Bayesian networks enable use of domain knowledge

$$p(x_1, \ldots x_n) = \prod_{i \in V} p(x_i \mid \mathbf{x}_{\mathrm{Pa}(i)})$$

Will my car start this morning?



Heckerman *et al.*, Decision-Theoretic Troubleshooting, 1995

Bayesian networks enable use of domain knowledge

$$p(x_1, \dots x_n) = \prod_{i \in V} p(x_i \mid \mathbf{x}_{\mathrm{Pa}(i)})$$

What is the differential diagnosis?

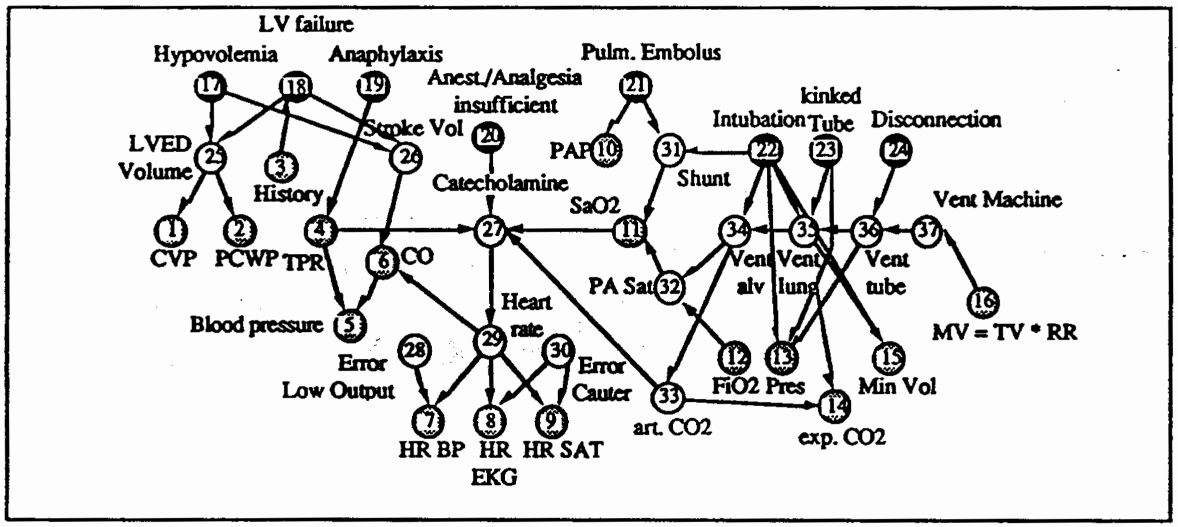
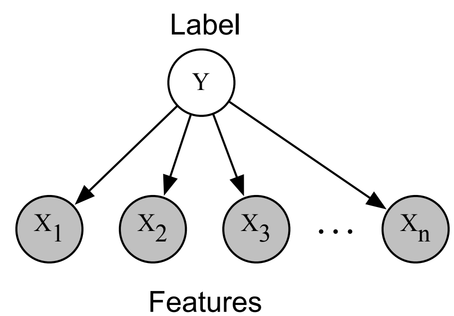


Fig. 1 The ALARM network representing causal relationships is shown with diagnostic (●), intermediate (○) and measurement (◎) nodes. CO: cardiac output, CVP: central venous pressure, LVED volume: left ventricular end-diastolic volume, LV failure: left ventricular failure, MV: minute ventilation, PA Sat: pulmonary artery oxygen saturation, PAP: pulmonary artery pressure, PCWP: pulmonary capillary wedge pressure, Pres: breathing pressure, RR: respiratory rate, TPR: total peripheral resistance, TV: tidal volume

Beinlich *et al.*, The ALARM Monitoring System, 1989

Bayesian networks are *generative models*

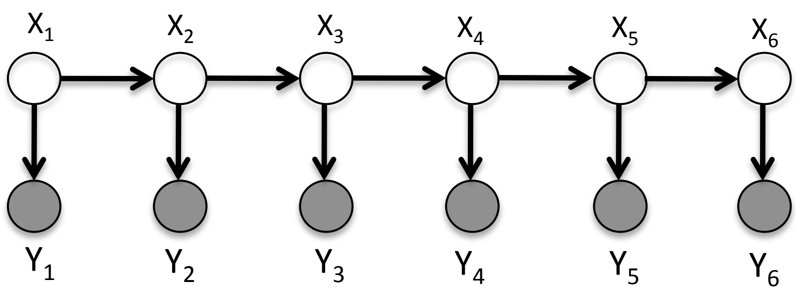
- Can sample from the joint distribution, top-down
- Suppose Y can be “spam” or “not spam”, and X_i is a binary indicator of whether word *i* is present in the e-mail
- Let’s try generating a few emails!

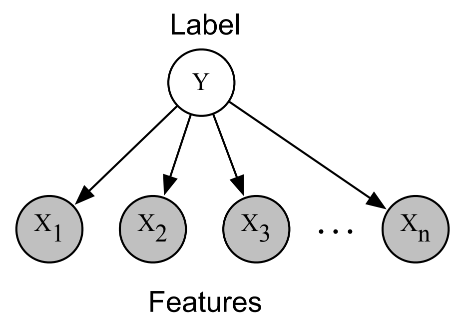


- Often helps to think about Bayesian networks as a generative model when constructing the structure and thinking about the model assumptions

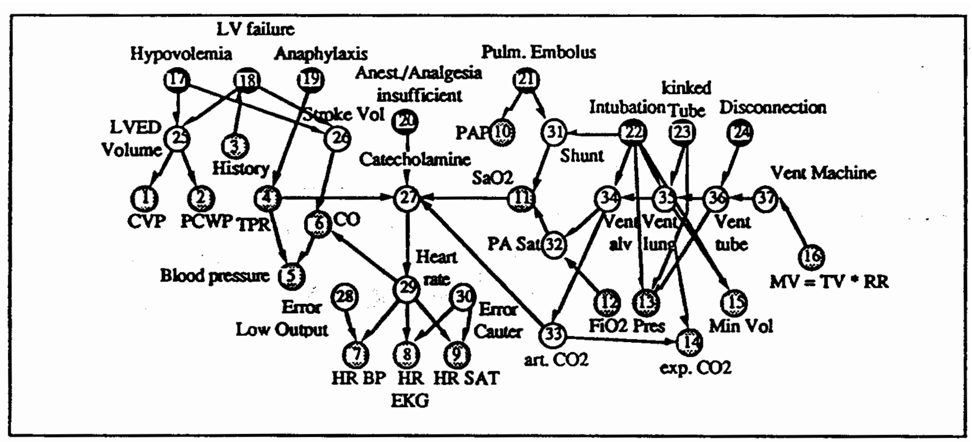
Inference in Bayesian networks

- Computing marginal probabilities in **tree** structured Bayesian networks is easy
  - The algorithm called “belief propagation” generalizes what we showed for hidden Markov models to arbitrary trees



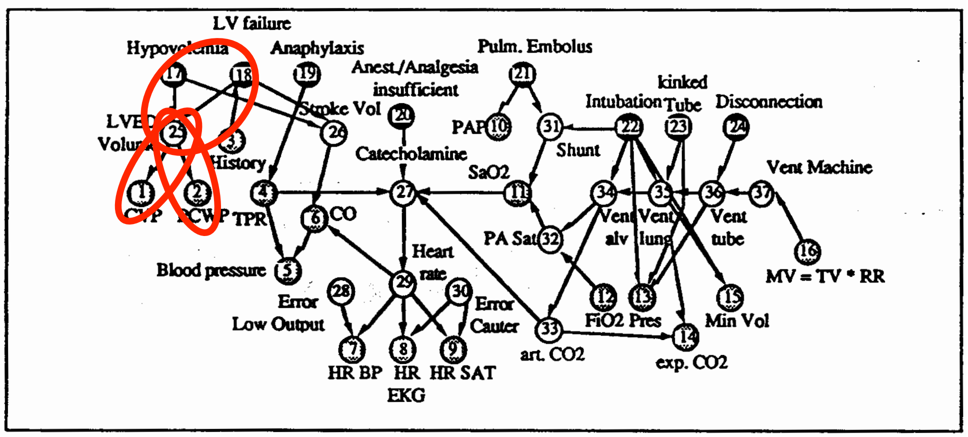


- Wait… this isn’t a tree! What can we do?



Inference in Bayesian networks

- In some cases (such as this) we can *transform* this into what is called a “junction tree”, and then run belief propagation



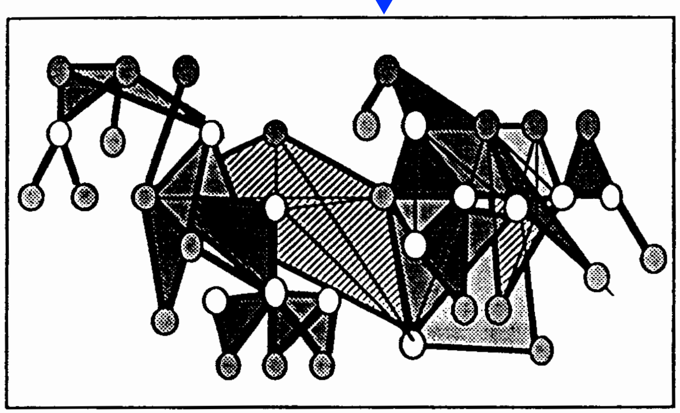
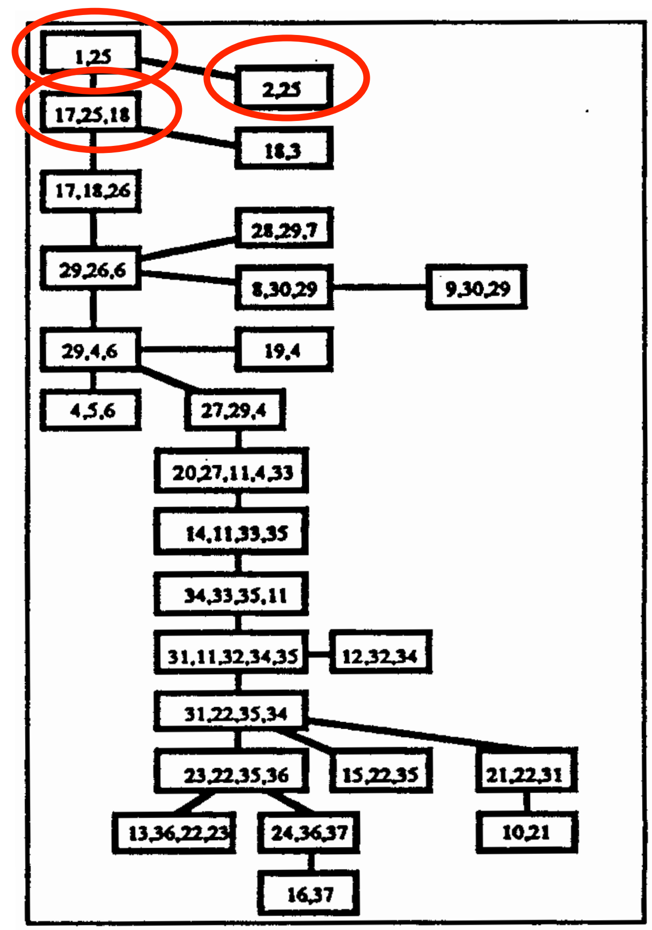


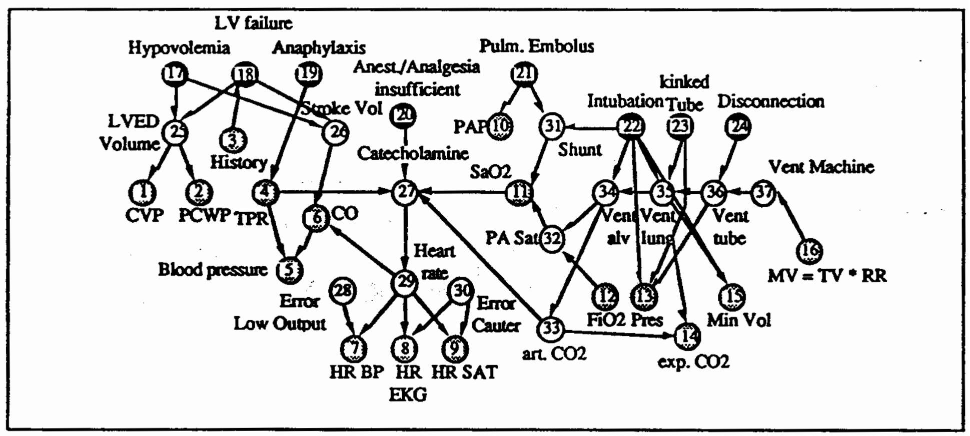
Fig. 7

Spiegelhalter's algorithm rearranges the ALARM network by triangulation and clique formation. The cliques are shaded differently to make them visible.



Approximate inference

- There is also a wealth of **approximate** inference algorithms that can be applied to Bayesian networks such as these



- Markov chain Monte Carlo algorithms repeatedly sample assignments for estimating marginals
- Variational inference algorithms (deterministic) find a simpler distribution which is "close" to the original, then compute marginals using the simpler distribution

Maximum likelihood estimation in Bayesian networks

- Suppose that we know the Bayesian network structure G
- Let $\theta_{x_i|\mathbf{x}_{pa(i)}}$ be the parameter giving the value of the CPD $p(x_i \mid \mathbf{x}_{pa(i)})$
- Maximum likelihood estimation corresponds to solving:

$$\max_{\theta} \frac{1}{M} \sum_{m=1}^{M} \log p(\mathbf{x}^M; \theta)$$

subject to the non-negativity and normalization constraints

- This is equal to:

$$\begin{aligned}
\max_{\theta} \frac{1}{M} \sum_{m=1}^{M} \log p(\mathbf{x}^M; \theta) &= \max_{\theta} \frac{1}{M} \sum_{m=1}^{M} \sum_{i=1}^{N} \log p(x_i^M \mid \mathbf{x}_{pa(i)}^M; \theta) \\
&= \max_{\theta} \sum_{i=1}^{N} \frac{1}{M} \sum_{m=1}^{M} \log p(x_i^M \mid \mathbf{x}_{pa(i)}^M; \theta)
\end{aligned}$$

- The optimization problem decomposes into an independent optimization problem for each CPD! Has a simple closed-form solution.