Chapter 6: Reasoning under Uncertainty

"The mind is a neural computer, fitted by natural selection with combinatorial algorithms for causal and probabilistic reasoning about plants, animals, objects, and people.

"In a universe with any regularities at all, decisions informed about the past are better than decisions made at random. That has always been true, and we would expect organisms, especially informavores such as humans, to have evolved acute intuitions about probability. The founders of probability, like the founders of logic, assumed they were just formalizing common sense."

Steven Pinker, How the Mind Works, 1997, pp. 524, 343.

Using Uncertain Knowledge

- Agents don't have complete knowledge about the world.
- Agents need to make decisions based on their uncertainty.
- It isn't enough to assume what the world is like.
  Example: wearing a seat belt.
- An agent needs to reason about its uncertainty.

Why Probability?

- There is lots of uncertainty about the world, but agents still need to act.
- Predictions are needed to decide what to do:
  - definitive predictions: you will be run over tomorrow
  - point probabilities: probability you will be run over tomorrow is 0.002
  - probability ranges: you will be run over with probability in range [0.001,0.34]
- Acting is gambling: agents who don't use probabilities will lose to those who do — Dutch books.
- Probabilities can be learned from data.
  Bayes' rule specifies how to combine data and prior knowledge.

Probability

- Probability is an agent's measure of belief in some proposition — subjective probability.
- An agent's belief depends on its prior assumptions and what the agent observes.

Numerical Measures of Belief

- Belief in proposition, f, can be measured in terms of a number between 0 and 1 — this is the probability of f.
  - The probability f is 0 means that f is believed to be definitely false.
  - The probability f is 1 means that f is believed to be definitely true.
- Using 0 and 1 is purely a convention.
- f has a probability between 0 and 1, means the agent is ignorant of its truth value.
- Probability is a measure of an agent's ignorance.
- Probability is *not* a measure of degree of truth.

Random Variables

- A random variable is a term in a language that can take one of a number of different values.
- The range of a variable X, written $range(X)$, is the set of values X can take.
- A tuple of random variables $\langle X_1, \ldots, X_n \rangle$ is a complex random variable with range $range(X_1) \times \cdots \times range(X_n)$. Often the tuple is written as $X_1, \ldots, X_n$.
- Assignment $X = x$ means variable X has value x.
- A proposition is a Boolean formula made from assignments of values to variables.

Possible World Semantics

- A possible world specifies an assignment of one value to each random variable.
- A random variable is a function from possible worlds into the range of the random variable.
- $\omega \models X = x$
  means variable X is assigned value x in world ω.
- Logical connectives have their standard meaning:

  $\omega \models \alpha \wedge \beta$ if $\omega \models \alpha$ and $\omega \models \beta$

  $\omega \models \alpha \vee \beta$ if $\omega \models \alpha$ or $\omega \models \beta$

  $\omega \models \neg\alpha$ if $\omega \not\models \alpha$
- Let Ω be the set of all possible worlds.

Semantics of Probability

For a finite number of possible worlds:

- Define a nonnegative measure $\mu(\omega)$ to each world ω so that the measures of the possible worlds sum to 1.
- The probability of proposition f is defined by:

$$P(f) = \sum_{\omega \models f} \mu(\omega).$$

Axioms of Probability: finite case

Three axioms define what follows from a set of probabilities:

Axiom 1 $0 \leq P(a)$ for any proposition a.

Axiom 2 $P(true) = 1$

Axiom 3 $P(a \vee b) = P(a) + P(b)$ if a and b cannot both be true.

- These axioms are sound and complete with respect to the semantics.

Consequences

1. Negation of a proposition: $P(\neg\alpha) = 1 - P(\alpha)$.

   The propositions $\alpha \vee \neg\alpha$ and $\neg(\alpha \wedge \neg\alpha)$ are tautologies.
   Therefore, $P(\alpha \vee \neg\alpha) = P(\alpha) + P(\neg\alpha) = 1$.

Consequences

2. Logically equivalent propositions have the same probability: $\alpha \leftrightarrow \beta \rightsquigarrow P(\alpha) = P(\beta)$.

   If $\alpha \leftrightarrow \beta$, then $\alpha \vee \neg\beta$ is a tautology and $P(\alpha \vee \neg\beta) = 1$. α and $\neg\beta$ are contradictory statements, so with Axiom 3

   $$P(\alpha \vee \neg\beta) = P(\alpha) + P(\neg\beta) = 1$$

   Since $P(\neg\beta) = 1 - P(\beta)$ also

   $$P(\alpha) + 1 - P(\beta) = 1,$$

   and therefore $P(\alpha) = P(\beta)$.

Consequences

3. Reasoning by cases: $P(\alpha) = P(\alpha \wedge \beta) + P(\alpha \wedge \neg\beta)$.

   The propositions $\alpha \leftrightarrow ((\alpha \wedge \beta) \vee (\alpha \wedge \neg\beta))$ and $((\alpha \wedge \beta) \wedge (\alpha \wedge \neg\beta))$ are tautologies. Thus,

$$P(\alpha) = P((\alpha \wedge \beta) \vee (\alpha \wedge \neg\beta)) = P(\alpha \wedge \beta) + P(\alpha \wedge \neg\beta).$$

4. Reasoning by cases, generalized:

   If V is a random variable with domain D, then, for all propositions α,

$$P(\alpha) = \sum_{d \in D} P(\alpha \wedge V = d).$$

Consequences

5. Disjunction for non-exclusive propositions:

$$P(\alpha \vee \beta) = P(\alpha) + P(\beta) - P(\alpha \wedge \beta).$$

$(\alpha \vee \beta) \leftrightarrow ((\alpha \wedge \neg\beta) \vee \beta)$ is a tautology. Thus,

$$P(\alpha \vee \beta) = P((\alpha \wedge \neg\beta) \vee \beta) = P(\alpha \wedge \neg\beta) + P(\beta).$$

With $P(\alpha \wedge \neg\beta) = P(\alpha) - P(\alpha \wedge \beta)$.

$$P(\alpha \vee \beta) = P(\alpha) - P(\alpha \wedge \beta) + P(\beta).$$

Semantics of Probability: general case

In the general case, probability defines a measure on sets of possible worlds. We define $\mu(S)$ for some sets $S \subseteq \Omega$ satisfying:

- $\mu(S) \geq 0$
- $\mu(\Omega) = 1$
- $\mu(S_1 \cup S_2) = \mu(S_1) + \mu(S_2)$ if $S_1 \cap S_2 = \{\}$.
  Or sometimes σ-additivity:

$$\mu(\bigcup_i S_i) = \sum_i \mu(S_i) \text{ if } S_i \cap S_j = \{\} \text{ for } i \neq j$$

Then $P(\alpha) = \mu(\{\omega | \omega \models \alpha\})$.

Probability Distributions

- A probability distribution on a random variable X is a function $range(X) \rightarrow [0, 1]$ such that

$$x \mapsto P(X = x).$$

  This is written as $P(X)$.
- This also includes the case where we have tuples of variables. E.g., $P(X, Y, Z)$ means $P(\langle X, Y, Z \rangle)$.
- When $range(X)$ is infinite sometimes we need a probability density function...

Conditioning

- Probabilistic conditioning specifies how to revise beliefs based on new information.
- An agent builds a probabilistic model taking all background information into account. This gives the prior probability.
- All other information must be conditioned on.
- If evidence e is all the information obtained subsequently, the conditional probability $P(h|e)$ of h given e is the posterior probability of h.

Semantics of Conditional Probability

- Evidence e rules out possible worlds incompatible with e.
- Evidence e induces a new measure, μ_e, over possible worlds

$$\mu_e(S) = \begin{cases} c \times \mu(S) & \text{if } \omega \models e \text{ for all } \omega \in S \\ 0 & \text{if } \omega \not\models e \text{ for some } \omega \in S \end{cases}$$

  We can show that $c = \frac{1}{P(e)}$.
- The conditional probability of formula h given evidence e is

$$\begin{aligned} P(h|e) &= \mu_e(\{\omega : \omega \models h\}) \\ &= \end{aligned}$$

Semantics of Conditional Probability

- Evidence e rules out possible worlds incompatible with e.
- Evidence e induces a new measure, μ_e, over possible worlds

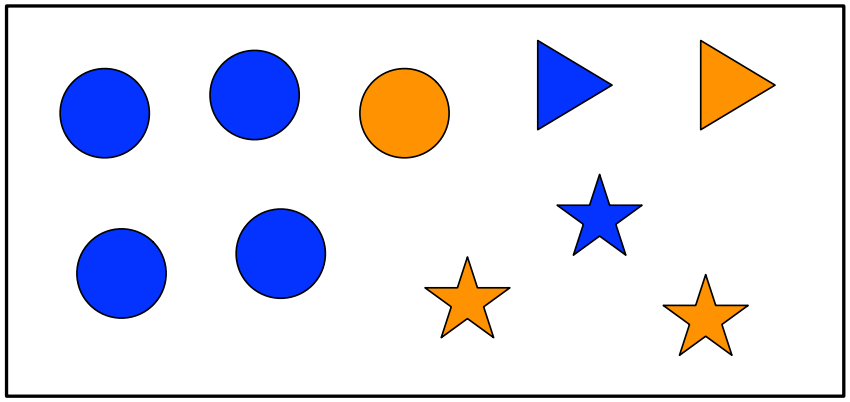
$$\mu_e(S) = \begin{cases} c \times \mu(S) & \text{if } \omega \models e \text{ for all } \omega \in S \\ 0 & \text{if } \omega \not\models e \text{ for some } \omega \in S \end{cases}$$

  We can show that $c = \frac{1}{P(e)}$.
- The conditional probability of formula h given evidence e is

$$\begin{aligned} P(h|e) &= \mu_e(\{\omega : \omega \models h\}) \\ &= \frac{P(h \wedge e)}{P(e)} \end{aligned}$$

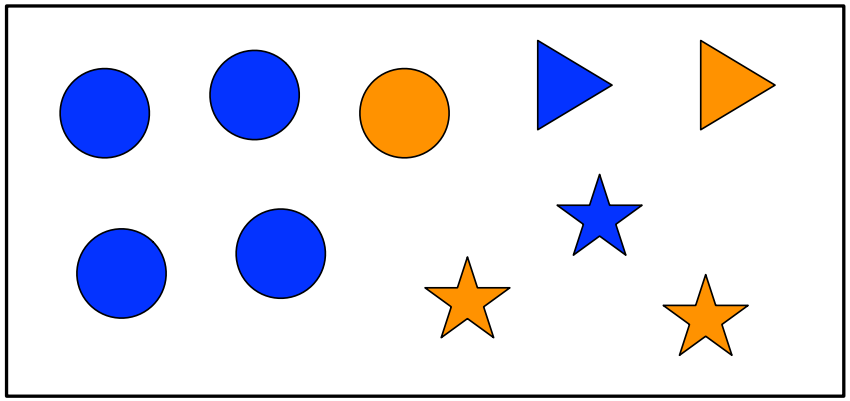
Conditioning

Possible Worlds:

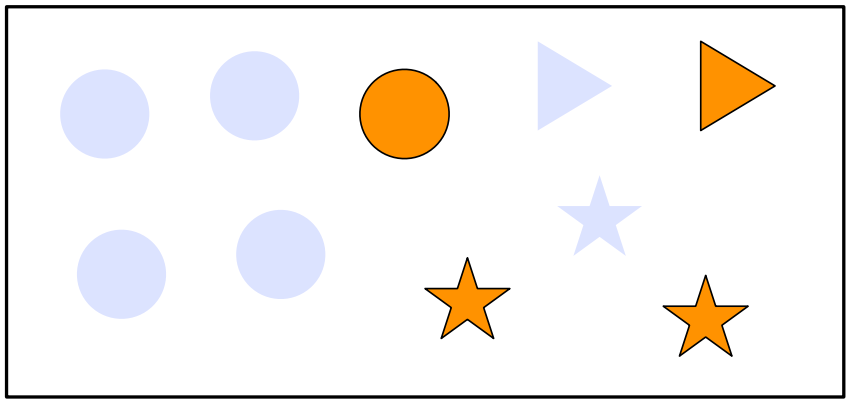


Conditioning

Possible Worlds:



Observe *Color* = *orange*:



Exercise

Flu	*Sneeze*	*Snore*	μ
true	true	true	0.064
true	true	false	0.096
true	false	true	0.016
true	false	false	0.024
false	true	true	0.096
false	true	false	0.144
false	false	true	0.224
false	false	false	0.336

What is:

(a) $P(flu \wedge sneeze)$

(b) $P(flu \wedge \neg sneeze)$

(c) $P(flu)$

(d) $P(sneeze \mid flu)$

(e) $P(\neg flu \wedge sneeze)$

(f) $P(flu \mid sneeze)$

(g) $P(sneeze \mid flu \wedge snore)$

(h) $P(flu \mid sneeze \wedge snore)$

Chain Rule

$$P(f_1 \wedge f_2 \wedge \ldots \wedge f_n)$$
$$=$$

Chain Rule

$$
\begin{aligned}
& P(f_1 \wedge f_2 \wedge \ldots \wedge f_n) \\
&= P(f_n | f_1 \wedge \cdots \wedge f_{n-1}) \times \\
&\quad P(f_1 \wedge \cdots \wedge f_{n-1}) \\
&=
\end{aligned}
$$

Chain Rule

$$
\begin{aligned}
& P(f_1 \wedge f_2 \wedge \ldots \wedge f_n) \\
&= P(f_n | f_1 \wedge \cdots \wedge f_{n-1}) \times \\
&\quad P(f_1 \wedge \cdots \wedge f_{n-1}) \\
&= P(f_n | f_1 \wedge \cdots \wedge f_{n-1}) \times \\
&\quad P(f_{n-1} | f_1 \wedge \cdots \wedge f_{n-2}) \times \\
&\quad P(f_1 \wedge \cdots \wedge f_{n-2}) \\
&= \\
&
\end{aligned}
$$

Chain Rule

$$
\begin{aligned}
& P(f_1 \wedge f_2 \wedge \ldots \wedge f_n) \\
&= P(f_n | f_1 \wedge \cdots \wedge f_{n-1}) \times \\
&\quad P(f_1 \wedge \cdots \wedge f_{n-1}) \\
&= P(f_n | f_1 \wedge \cdots \wedge f_{n-1}) \times \\
&\quad P(f_{n-1} | f_1 \wedge \cdots \wedge f_{n-2}) \times \\
&\quad P(f_1 \wedge \cdots \wedge f_{n-2}) \\
&= P(f_n | f_1 \wedge \cdots \wedge f_{n-1}) \times \\
&\quad P(f_{n-1} | f_1 \wedge \cdots \wedge f_{n-2}) \\
&\quad \times \cdots \times P(f_3 | f_1 \wedge f_2) \times P(f_2 | f_1) \times P(f_1) \\
&= \prod_{i=1}^{n} P(f_i | f_1 \wedge \cdots \wedge f_{i-1})
\end{aligned}
$$

Bayes' theorem

The chain rule and commutativity of conjunction ($h \wedge e$ is equivalent to $e \wedge h$) gives us:

$$P(h \wedge e) \quad =$$

Bayes' theorem

The chain rule and commutativity of conjunction ($h \wedge e$ is equivalent to $e \wedge h$) gives us:

$$P(h \wedge e) = P(h|e) \times P(e)$$

Bayes' theorem

The chain rule and commutativity of conjunction ($h \wedge e$ is equivalent to $e \wedge h$) gives us:

$$\begin{aligned} P(h \wedge e) &= P(h|e) \times P(e) \\ &= P(e|h) \times P(h). \end{aligned}$$

Bayes' theorem

The chain rule and commutativity of conjunction ($h \wedge e$ is equivalent to $e \wedge h$) gives us:

$$\begin{aligned} P(h \wedge e) &= P(h|e) \times P(e) \\ &= P(e|h) \times P(h). \end{aligned}$$

If $P(e) \neq 0$, divide the right hand sides by $P(e)$:

$$P(h|e) =$$

Bayes' theorem

The chain rule and commutativity of conjunction ($h \wedge e$ is equivalent to $e \wedge h$) gives us:

$$\begin{aligned} P(h \wedge e) &= P(h|e) \times P(e) \\ &= P(e|h) \times P(h). \end{aligned}$$

If $P(e) \neq 0$, divide the right hand sides by $P(e)$:

$$P(h|e) = \frac{P(e|h) \times P(h)}{P(e)}.$$

This is Bayes' theorem.

Why is Bayes' theorem interesting?

- Often you have causal knowledge:
  $P(\textit{symptom} \mid \textit{disease})$
  $P(\textit{light is off} \mid \textit{status of switches and switch positions})$
  $P(\textit{alarm} \mid \textit{fire})$
  $P(\textit{image looks like}$  $\mid \textit{a tree is in front of a car})$
- and want to do evidential reasoning:
  $P(\textit{disease} \mid \textit{symptom})$
  $P(\textit{status of switches} \mid \textit{light is off and switch positions})$
  $P(\textit{fire} \mid \textit{alarm})$.
  $P(\textit{a tree is in front of a car} \mid \textit{image looks like}$  $)$

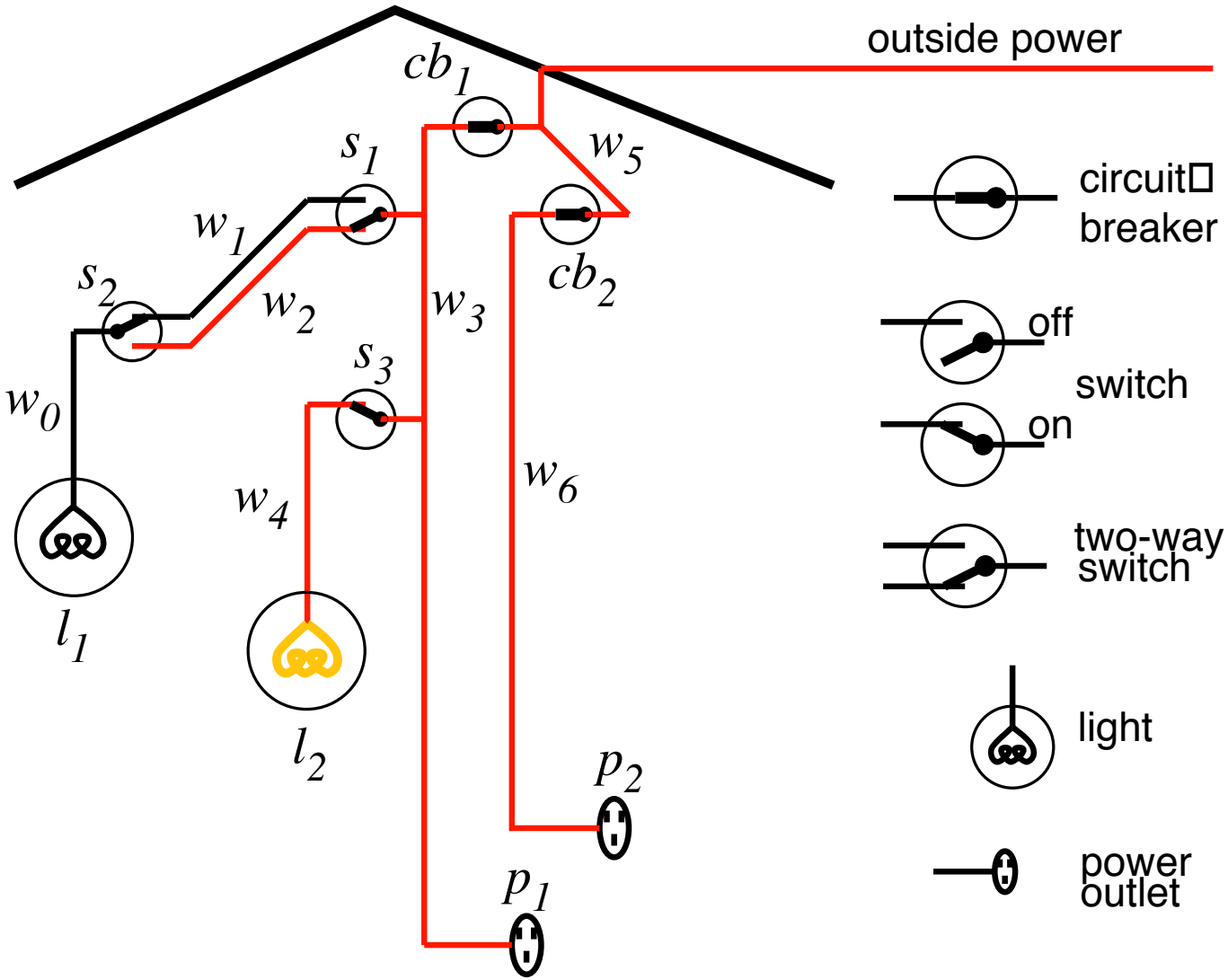
Conditional independence

Random variable X is independent of random variable Y given random variable Z if, for all $x_i \in dom(X)$, $y_j \in dom(Y)$, $y_k \in dom(Y)$ and $z_m \in dom(Z)$,

$$\begin{aligned} & P(X = x_i | Y = y_j \wedge Z = z_m) \\ = \; & P(X = x_i | Y = y_k \wedge Z = z_m) \\ = \; & P(X = x_i | Z = z_m). \end{aligned}$$

That is, knowledge of Y's value doesn't affect your belief in the value of X, given a value of Z.

Example domain (diagnostic assistant)

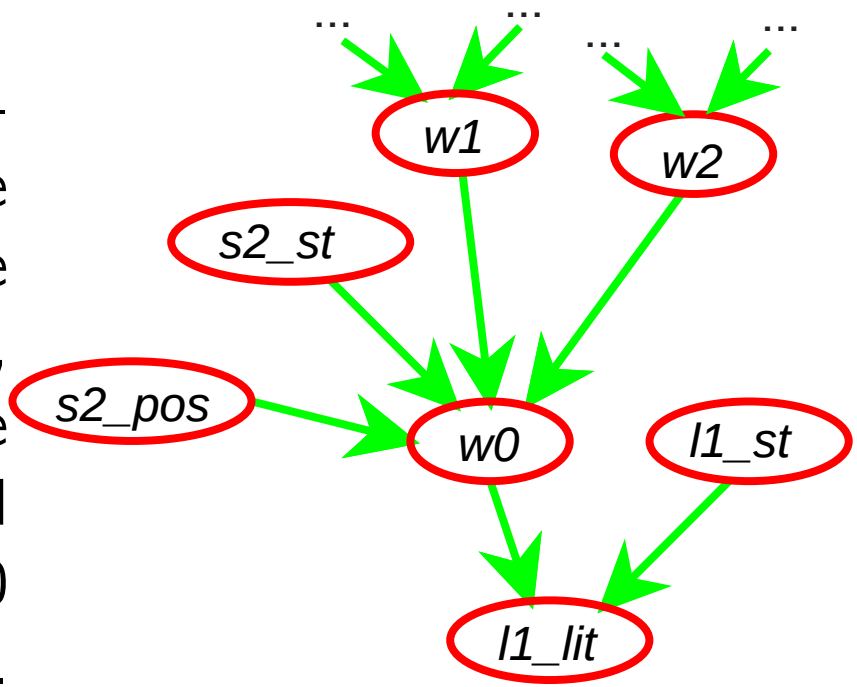


Examples of conditional independence

- The identity of the queen of Canada is independent of whether light $l1$ is lit given whether there is outside power.
- Whether there is someone in a room is independent of whether a light $l2$ is lit given the position of switch $s3$.
- Whether light $l1$ is lit is independent of the position of light switch $s2$ given whether there is power in wire w_0.
- Every other variable may be independent of whether light $l1$ is lit given whether there is power in wire w_0 and the status of light $l1$ (if it's ok, or if not, how it's broken).

Idea of belief networks

Whether $l1$ is lit ($L1\_lit$) depends only on the status of the light ($L1\_st$) and whether there is power in wire $w0$. Thus, $L1\_lit$ is independent of the other variables given $L1\_st$ and $W0$. In a belief network, $W0$ and $L1\_st$ are parents of $L1\_lit$. Similarly, $W0$ depends only on whether there is power in $w1$, whether there is power in $w2$, the position of switch $s2$ ($S2\_pos$), and the status of switch $s2$ ($S2\_st$).



Belief networks

A belief network is a graph: the nodes are random variables; there is an arc from the parents of each node into that node.

Suppose $\{x_1, \ldots, x_n\}$ are the variables of interest.

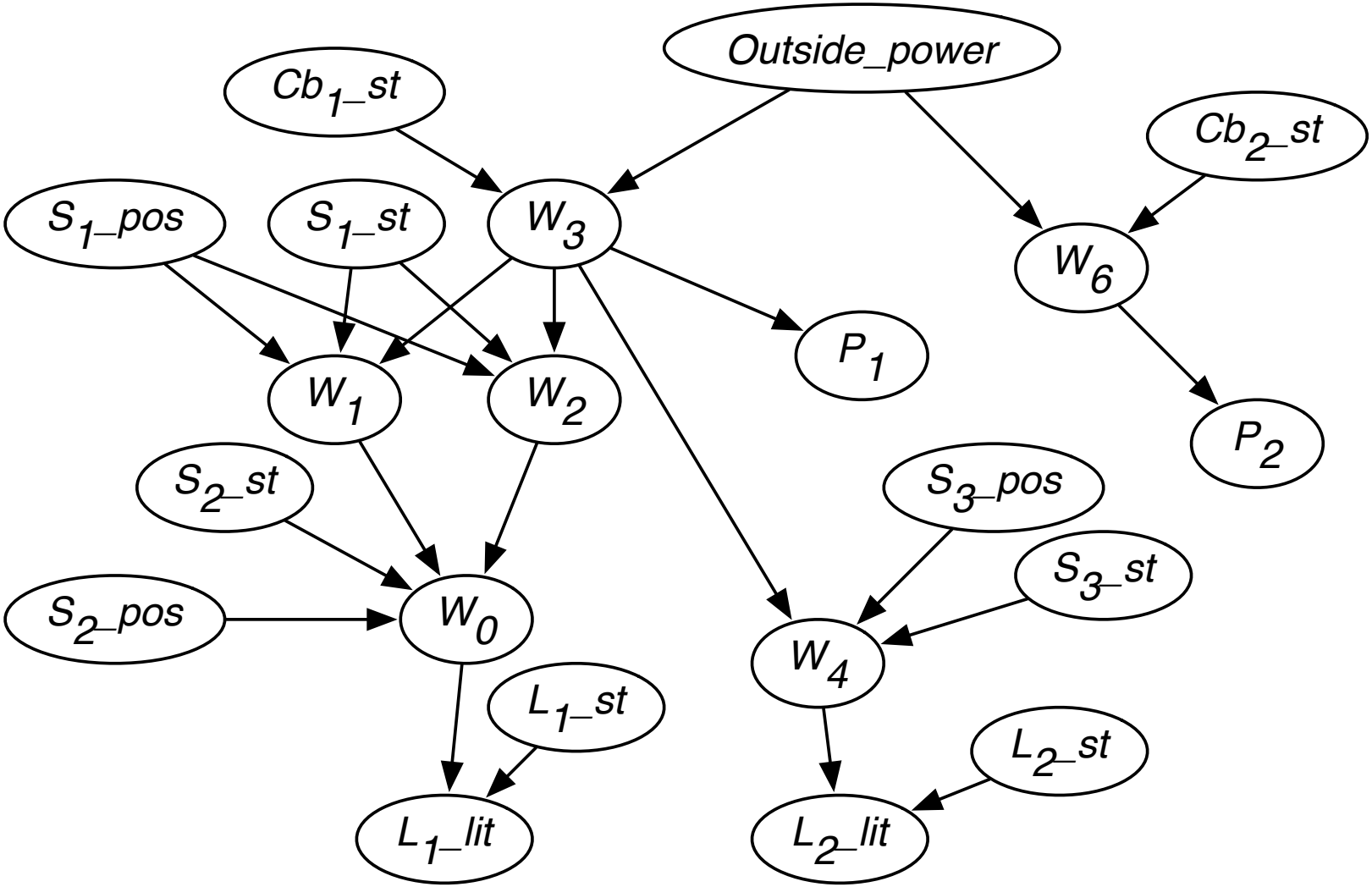
- Totally order the variables of interest: $X_1, \ldots, X_n$
- Theorem of probability theory (chain rule): $P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i | X_1, \ldots, X_{i-1})$
- The parents $parents(X_i)$ of X_i are those predecessors of X_i that render X_i independent of the other predecessors. That is, $parents(X_i) \subseteq X_1, \ldots, X_{i-1}$ and $P(X_i | parents(X_i)) = P(X_i | X_1, \ldots, X_{i-1})$
- So $P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i | parents(X_i))$

Components of a belief network

A belief network consists of:

- a directed acyclic graph with nodes labeled with random variables
- a domain for each random variable
- a set of conditional probability tables for each variable given its parents (including prior probabilities for nodes with no parents).

Example belief network



Example belief network (continued)

The belief network also specifies:

- The domain of the variables:
  $W_0, \ldots, W_6$ have domain $\{live, dead\}$
  S_1_pos, S_2_pos, and S_3_pos have domain $\{up, down\}$
  S_1_st has $\{ok, upside\_down, short, intermittent, broken\}$.
- Conditional probabilities, including:
  $P(W_1 = live | s_1\_pos = up \wedge S_1\_st = ok \wedge W_3 = live)$
  $P(W_1 = live | s_1\_pos = up \wedge S_1\_st = ok \wedge W_3 = dead)$
  $P(S_1\_pos = up)$
  $P(S_1\_st = upside\_down)$

Belief network summary

- A belief network is automatically acyclic by construction.
- A belief network is a directed acyclic graph (DAG) where nodes are random variables.
- The parents of a node n are those variables on which n directly depends.
- A belief network is a graphical representation of dependence and independence:
  - A variable is independent of its non-descendants given its parents.

Constructing belief networks

To represent a domain in a belief network, you need to consider:

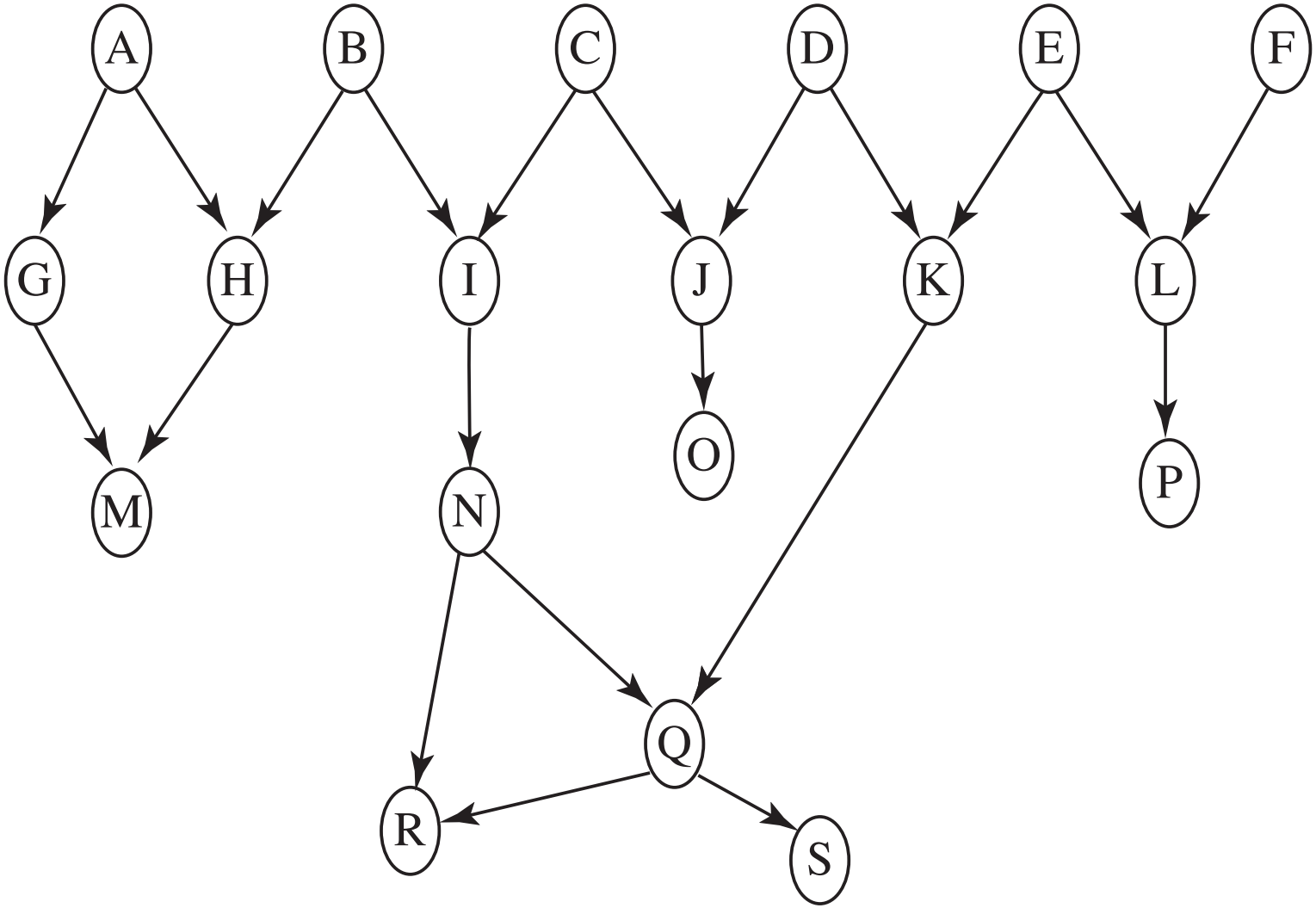
- What are the relevant variables?
  - What will you observe?
  - What would you like to find out (query)?
  - What other features make the model simpler?
- What values should these variables take?
- What is the relationship between them? This should be expressed in terms of local influence.
- How does the value of each variable depend on its parents? This is expressed in terms of the conditional probabilities.

Using belief networks

The power network can be used in a number of ways:

- Conditioning on the status of the switches and circuit breakers, whether there is outside power and the position of the switches, you can simulate the lighting.
- Given values for the switches, the outside power, and whether the lights are lit, you can determine the posterior probability that each switch or circuit breaker is *ok* or not.
- Given some switch positions and some outputs and some intermediate values, you can determine the probability of any other variable in the network.

Understanding independence: example



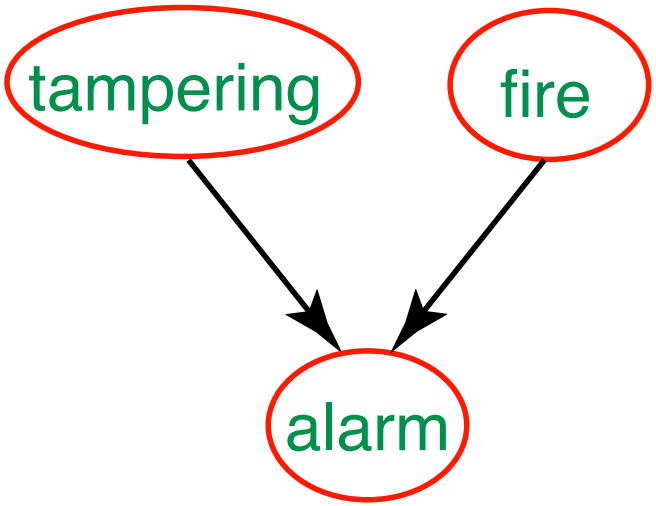
Understanding independence: questions

- On which given probabilities does $P(N)$ depend?
- If you were to observe a value for B, which variables' probabilities will change?
- If you were to observe a value for N, which variables' probabilities will change?
- Suppose you had observed a value for M; if you were to then observe a value for N, which variables' probabilities will change?
- Suppose you had observed B and Q; which variables' probabilities will change when you observe N?

What variables are affected by observing?

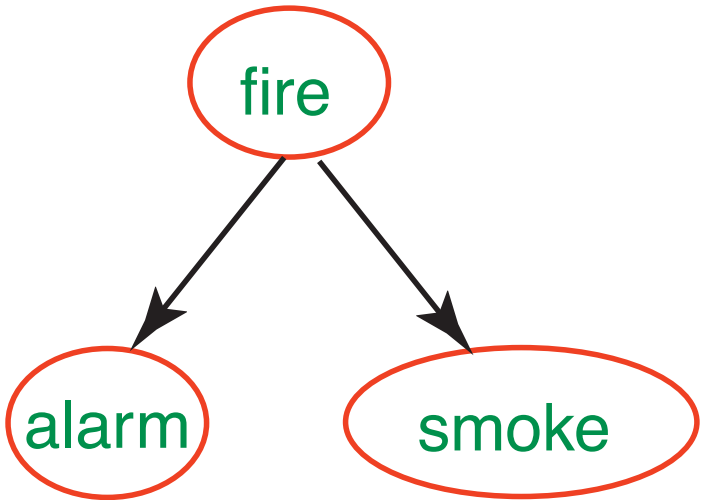
- If you observe variable $\overline{Y}$, the variables whose posterior probability is different from their prior are:
  - The ancestors of $\overline{Y}$ and
  - their descendants.
- Intuitively (if you have a causal belief network):
  - You do abduction to possible causes and
  - prediction from the causes.

Common descendants



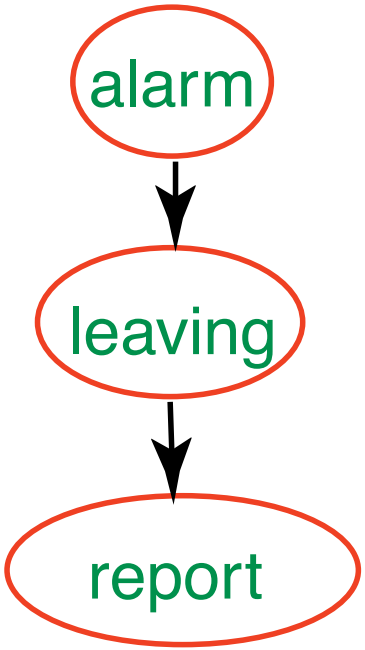
- *tampering* and *fire* are independent
- *tampering* and *fire* are dependent given *alarm*
- Intuitively, *tampering* can explain away *fire*

Common ancestors



- *alarm* and *smoke* are dependent
- *alarm* and *smoke* are independent given *fire*
- Intuitively, *fire* can explain *alarm* and *smoke*; learning one can affect the other by changing your belief in *fire*.

Chain



- *alarm* and *report* are dependent
- *alarm* and *report* are independent given *leaving*
- Intuitively, the only way that the *alarm* affects *report* is by affecting *leaving*.

Pruning Irrelevant Variables

Suppose you want to compute $P(X|e_1 \ldots e_k)$:

- Prune any variables that have no observed or queried descendents.
- Connect the parents of any observed variable.
- Remove arc directions.
- Remove observed variables.
- Remove any variables not connected to X in the resulting (undirected) graph.

Belief network inference

Four main approaches to determine posterior distributions in belief networks:

- Variable Elimination: exploit the structure of the network to eliminate (sum out) the non-observed, non-query variables one at a time.
- Search-based approaches: enumerate some of the possible worlds, and estimate posterior probabilities from the worlds generated.
- Stochastic simulation: random cases are generated according to the probability distributions.
- Variational methods: find the closest tractable distribution to the (posterior) distribution we are interested in.

Factors

A factor is a representation of a function from a tuple of random variables into a number.
We will write factor f on variables $X_1, \ldots, X_j$ as $f(X_1, \ldots, X_j)$.
We can assign some or all of the variables of a factor:

- $f(X_1 = v_1, X_2, \ldots, X_j)$, where $v_1 \in dom(X_1)$, is a factor on $X_2, \ldots, X_j$.
- $f(X_1 = v_1, X_2 = v_2, \ldots, X_j = v_j)$ is a number that is the value of f when each X_i has value v_i.

The former is also written as $f(X_1, X_2, \ldots, X_j)_{X_1 = v_1}$, etc.

Example factors

$r(X, Y, Z)$:

X	Y	Z	val
t	t	t	0.1
t	t	f	0.9
t	f	t	0.2
t	f	f	0.8
f	t	t	0.4
f	t	f	0.6
f	f	t	0.3
f	f	f	0.7

$r(X{=}t, Y, Z)$:

Y	Z	val
t	t	0.1
t	f	0.9
f	t	0.2
f	f	0.8

$r(X{=}t, Y, Z{=}f)$:

Y	val
t	0.9
f	0.8

$r(X{=}t, Y{=}f, Z{=}f) = 0.8$

Multiplying factors

The product of factor $f_1(\overline{X}, \overline{Y})$ and $f_2(\overline{Y}, \overline{Z})$, where $\overline{Y}$ are the variables in common, is the factor $(f_1 \times f_2)(\overline{X}, \overline{Y}, \overline{Z})$ defined by:

$$(f_1 \times f_2)(\overline{X}, \overline{Y}, \overline{Z}) = f_1(\overline{X}, \overline{Y}) f_2(\overline{Y}, \overline{Z}).$$

Multiplying factors example

f_1:

A	B	val
t	t	0.1
t	f	0.9
f	t	0.2
f	f	0.8

f_2:

B	C	val
t	t	0.3
t	f	0.7
f	t	0.6
f	f	0.4

$f_1 \times f_2$:

A	B	C	val
t	t	t	0.03
t	t	f	0.07
t	f	t	0.54
t	f	f	0.36
f	t	t	0.06
f	t	f	0.14
f	f	t	0.48
f	f	f	0.32

Summing out variables

We can sum out a variable, say X_1 with domain $\{v_1, \ldots, v_k\}$, from factor $f(X_1, \ldots, X_j)$, resulting in a factor on $X_2, \ldots, X_j$ defined by:

$$\begin{aligned}(\sum_{X_1} f)(X_2, \ldots, X_j) \\ = \quad f(X_1 = v_1, \ldots, X_j) + \cdots + f(X_1 = v_k, \ldots, X_j)\end{aligned}$$

Summing out a variable example

f_3:

A	B	C	val
t	t	t	0.03
t	t	f	0.07
t	f	t	0.54
t	f	f	0.36
f	t	t	0.06
f	t	f	0.14
f	f	t	0.48
f	f	f	0.32

$\sum_B f_3$:

A	C	val
t	t	0.57
t	f	0.43
f	t	0.54
f	f	0.46

Evidence

If we want to compute the posterior probability of Z given evidence $Y_1 = v_1 \wedge \ldots \wedge Y_j = v_j$:

$$P(Z | Y_1 = v_1, \ldots, Y_j = v_j)$$

$$=$$

Evidence

If we want to compute the posterior probability of Z given evidence $Y_1 = v_1 \wedge \ldots \wedge Y_j = v_j$:

$$
\begin{aligned}
& P(Z | Y_1 = v_1, \ldots, Y_j = v_j) \\
= & \frac{P(Z, Y_1 = v_1, \ldots, Y_j = v_j)}{P(Y_1 = v_1, \ldots, Y_j = v_j)} \\
= &
\end{aligned}
$$

Evidence

If we want to compute the posterior probability of Z given evidence $Y_1 = v_1 \wedge \ldots \wedge Y_j = v_j$:

$$\begin{aligned} & P(Z | Y_1 = v_1, \ldots, Y_j = v_j) \\ = & \frac{P(Z, Y_1 = v_1, \ldots, Y_j = v_j)}{P(Y_1 = v_1, \ldots, Y_j = v_j)} \\ = & \frac{P(Z, Y_1 = v_1, \ldots, Y_j = v_j)}{\sum_Z P(Z, Y_1 = v_1, \ldots, Y_j = v_j).} \end{aligned}$$

So the computation reduces to the probability of $P(Z, Y_1 = v_1, \ldots, Y_j = v_j)$.
We normalize at the end.

Probability of a conjunction

Suppose the variables of the belief network are $X_1, \ldots, X_n$.
To compute $P(Z, Y_1 = v_1, \ldots, Y_j = v_j)$, we sum out the other variables, $Z_1, \ldots, Z_k = \{X_1, \ldots, X_n\} - \{Z\} - \{Y_1, \ldots, Y_j\}$.
We order the Z_i into an elimination ordering.

$$P(Z, Y_1 = v_1, \ldots, Y_j = v_j)$$

$$=$$

Probability of a conjunction

Suppose the variables of the belief network are $X_1, \ldots, X_n$.
To compute $P(Z, Y_1 = v_1, \ldots, Y_j = v_j)$, we sum out the other variables, $Z_1, \ldots, Z_k = \{X_1, \ldots, X_n\} - \{Z\} - \{Y_1, \ldots, Y_j\}$.
We order the Z_i into an elimination ordering.

$$
\begin{aligned}
& P(Z, Y_1 = v_1, \ldots, Y_j = v_j) \\
&= \sum_{Z_k} \cdots \sum_{Z_1} P(X_1, \ldots, X_n)_{Y_1 = v_1, \ldots, Y_j = v_j}. \\
&=
\end{aligned}
$$

Probability of a conjunction

Suppose the variables of the belief network are $X_1, \ldots, X_n$.
To compute $P(Z, Y_1 = v_1, \ldots, Y_j = v_j)$, we sum out the other variables, $Z_1, \ldots, Z_k = \{X_1, \ldots, X_n\} - \{Z\} - \{Y_1, \ldots, Y_j\}$.
We order the Z_i into an elimination ordering.

$$
\begin{aligned}
& P(Z, Y_1 = v_1, \ldots, Y_j = v_j) \\
&= \sum_{Z_k} \cdots \sum_{Z_1} P(X_1, \ldots, X_n)_{Y_1 = v_1, \ldots, Y_j = v_j} \cdot \\
&= \sum_{Z_k} \cdots \sum_{Z_1} \prod_{i=1}^{n} P(X_i | parents(X_i))_{Y_1 = v_1, \ldots, Y_j = v_j} \cdot
\end{aligned}
$$

Computing sums of products

Computation in belief networks reduces to computing the sums of products.

- How can we compute $ab + ac$ efficiently?

Computing sums of products

Computation in belief networks reduces to computing the sums of products.

- How can we compute $ab + ac$ efficiently?
- Distribute out the a giving $a(b + c)$

Computing sums of products

Computation in belief networks reduces to computing the sums of products.

- How can we compute $ab + ac$ efficiently?
- Distribute out the a giving $a(b + c)$
- How can we compute $\sum_{Z_1} \prod_{i=1}^{n} P(X_i|parents(X_i))$ efficiently?

Computing sums of products

Computation in belief networks reduces to computing the sums of products.

- How can we compute $ab + ac$ efficiently?
- Distribute out the a giving $a(b + c)$
- How can we compute $\sum_{Z_1} \prod_{i=1}^{n} P(X_i|parents(X_i))$ efficiently?
- Distribute out those factors that don't involve Z_1.

Variable elimination algorithm

To compute $P(Z|Y_1 = v_1 \wedge \ldots \wedge Y_j = v_j)$:

- Construct a factor for each conditional probability.
- Set the observed variables to their observed values.
- Sum out each of the other variables (the $\{Z_1, \ldots, Z_k\}$) according to some elimination ordering.
- Multiply the remaining factors. Normalize by dividing the resulting factor $f(Z)$ by $\sum_Z f(Z)$.

Summing out a variable

To sum out a variable Z_j from a product $f_1, \ldots, f_k$ of factors:

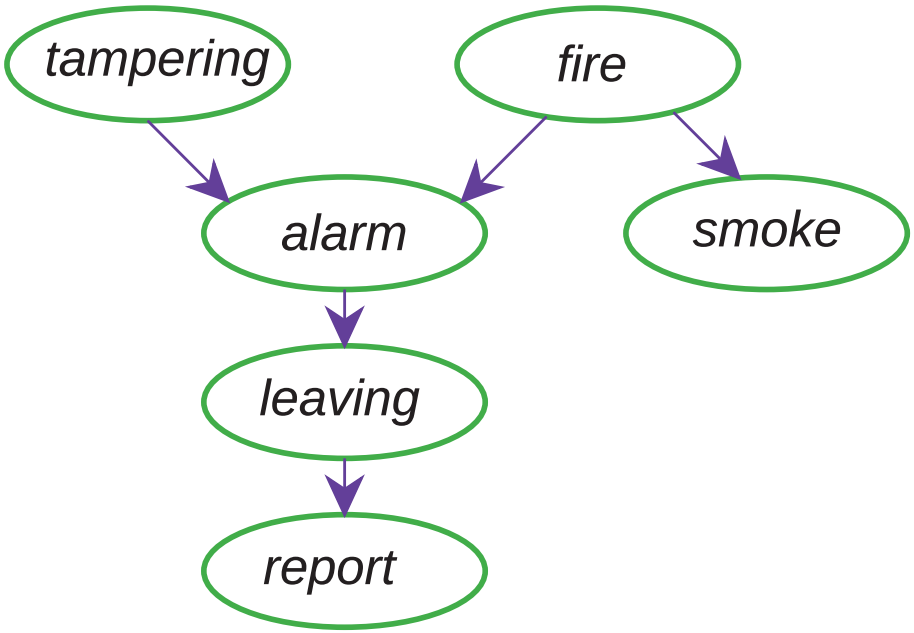
- Partition the factors into
  - those that don't contain Z_j, say $f_1, \ldots, f_i$,
  - those that contain Z_j, say $f_{i+1}, \ldots, f_k$

We know:

$$\sum_{Z_j} f_1 \times \cdots \times f_k = f_1 \times \cdots \times f_i \times \left(\sum_{Z_j} f_{i+1} \times \cdots \times f_k \right) .$$

- Explicitly construct a representation of the rightmost factor. Replace the factors $f_{i+1}, \ldots, f_k$ by the new factor.

Variable elimination example



$P(tampering) = 0.02$
$P(fire) = 0.01$
$P(alarm|fire \wedge tampering) = 0.5$
$P(alarm|fire \wedge \neg tampering) = 0.99$
$P(alarm|\neg fire \wedge tampering) = 0.85$
$P(alarm|\neg fire \wedge \neg tampering) = 0.0001$
$P(smoke|fire) = 0.9$
$P(smoke|\neg fire) = 0.01$
$P(leaving|alarm) = 0.88$
$P(leaving|\neg alarm) = 0.001$
$P(report|leaving) = 0.75$
$P(report|\neg leaving) = 0.01$

Query: $P(Tampering|Smoke = true \wedge Report = true)$.

Variable elimination example

Conditional probabilities and factors

$P(Tampering) \rightarrow f_0(Tampering) =$

true	0.02
false	0.98

$P(Fire) \rightarrow f_1(Fire) =$

true	0.01
false	0.99

Variable elimination example

Conditional probabilities and factors

$P(Alarm|Tampering, Fire)$

$\rightarrow f_2(Alarm, Tampering, Fire) =$

true	true	true	0.5
true	true	false	0.99
true	false	true	0.85
true	false	false	0.0001
false	true	true	0.5
false	true	false	0.01
false	false	true	0.15
false	false	false	0.9999

Variable elimination example

Conditional probabilities and factors

$P(Smoke|Fire)$

$$\rightarrow f_3(Smoke, Fire) = \begin{array}{|lll|} \hline \text{true} & \text{true} & 0.9 \\ \text{true} & \text{false} & 0.01 \\ \text{false} & \text{true} & 0.1 \\ \text{false} & \text{false} & 0.99 \\ \hline \end{array}$$

$P(Leaving|Alarm)$

$$\rightarrow f_4(Leaving, Alarm) = \begin{array}{|lll|} \hline \text{true} & \text{true} & 0.88 \\ \text{true} & \text{false} & 0.001 \\ \text{false} & \text{true} & 0.12 \\ \text{false} & \text{false} & 0.999 \\ \hline \end{array}$$

Variable elimination example

Conditional probabilities and factors

$P(Report|Leaving)$

$$\rightarrow f_5(Report, Leaving) =$$

true	true	0.75
true	false	0.01
false	true	0.25
false	false	0.99

Variable elimination example

Eliminate the observed variable *Smoke*

$f_3(Smoke, Fire) =$

true	true	0.9
true	false	0.01
false	true	0.1
false	false	0.99

$P(Smoke = true | Fire)$

$\rightarrow f_3'(Fire) =$

true	0.9
false	0.01

Variable elimination example

Eliminate the observed variable *Report*

$f_5(Report, Leaving) =$

true	true	0.75
true	false	0.01
false	true	0.25
false	false	0.99

$P(Report = yes | Leaving)$

$\rightarrow f_5'(Leaving) =$

true	0.75
false	0.01

Variable elimination example

Select e.g. *Fire* to be eliminated next

Collect all the factors containing *Fire*:

$f_1(Fire) =$

true	0.01
false	0.99

$f_3'(Fire) =$

true	0.9
false	0.01

Variable elimination example

Collect all the factors containing *Fire*:

$f_2(Alarm, Tampering, Fire) =$

true	true	true	0.5
true	true	false	0.99
true	false	true	0.85
true	false	false	0.0001
false	true	true	0.5
false	true	false	0.01
false	false	true	0.15
false	false	false	0.9999

Variable elimination example

Compute a new factor for them, eliminating *Fire*

$f_6(Tampering, Alarm)$

$$= \sum_{Fire} (f_1(Fire) \times f_2(Tampering, Fire, Alarm) \times f_3(Fire)$$

$$= \begin{array}{|cc|c|} \hline \text{true} & \text{true} & 0.0143 \\ \text{true} & \text{false} & 0.0076 \\ \text{false} & \text{true} & 0.0046 \\ \text{false} & \text{false} & 0.0114 \\ \hline \end{array}$$

remaining factors:
$f_0(Tampering), f_4(Alarm, Leaving), f_5(Leaving),$
$f_6(Tampering, Alarm)$

Variable elimination example

Select e.g. *Alarm* to be eliminated next.

Collect the factors containing *Alarm*

$f_4(Leaving, Alarm) =$

true	true	0.88
true	false	0.001
false	true	0.12
false	false	0.999

$f_6(Tampering, Alarm) =$

true	true	0.0143
true	false	0.0076
false	true	0.0046
false	false	0.0114

Variable elimination example

Compute a new factor for them, eliminating *Alarm*

$f_7(Tampering, Leaving)$

$$= \sum_{Alarm} f_4(Alarm, Leaving) \times f_6(Tampering, Alarm)$$

$$= \begin{array}{|cc|c|} \hline \text{true} & \text{true} & 0.0126 \\ \text{true} & \text{false} & 0.0040 \\ \text{false} & \text{true} & 0.0093 \\ \text{false} & \text{false} & 0.0119 \\ \hline \end{array}$$

remaining factors:

$f_0(Tampering), f_5(Leaving), f_7(Tampering, Leaving)$

Variable elimination example

Select *Leaving* to be eliminated next

Collect the factors containing *Leaving*

$f_5'(Leaving) =$

true	0.75
false	0.01

$f_7(Tampering, Leaving) =$

true	true	0.0126
true	false	0.0040
false	true	0.0093
false	false	0.0119

Variable elimination example

Compute a new factor for them, eliminating *Leaving*

$f_8(Tampering)$

$$= \sum_{Leaving} f_5'(Leaving) \times f_7(Tampering, Leaving)$$

$$= \begin{array}{|c|c|} \hline \text{true} & 0.0095 \\ \hline \text{false} & 0.0071 \\ \hline \end{array}$$

Variable elimination example

Multiply the remaining factors for *Tampering*

$f_0(Tampering) =$

true	0.02
false	0.98

$f_8(Tampering) =$

true	0.0031
false	0.0060

$f_9(Tampering)$

$= f_0(Tampering) \times f_8(Tampering) =$

true	0.000062
false	0.00588

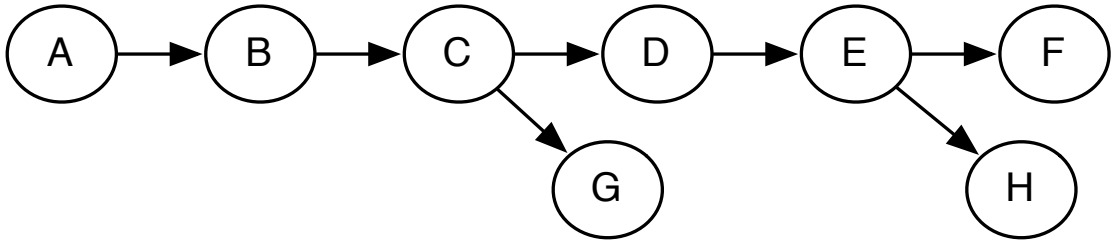
Variable elimination example

posterior distribution for *Tampering*

$$P(Tampering|Report, Smoke) =$$

$$\frac{f_9(Tampering)}{\sum_{Tampering} f_9(Tampering)} = \begin{array}{|c|c|}\hline \text{true} & 0.01 \\ \text{false} & 0.98 \\ \hline \end{array}$$

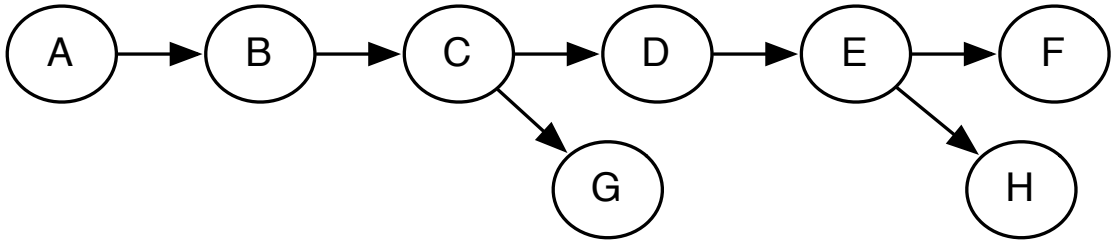
Variable Elimination example



Query: $P(G|f)$; elimination ordering: A, H, E, D, B, C

$P(G|f) \quad \propto$

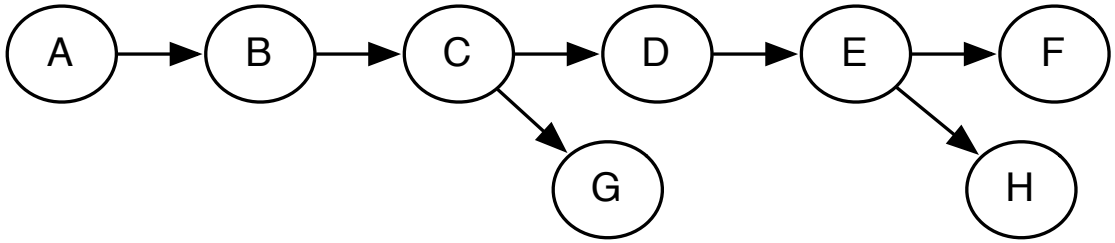
Variable Elimination example



Query: $P(G|f)$; elimination ordering: A, H, E, D, B, C

$$P(G|f) \propto \sum_C \sum_B \sum_D \sum_E \sum_H \sum_A P(A)P(B|A)P(C|B)$$
$$P(D|C)P(E|D)P(f|E)P(G|C)P(H|E)$$

Variable Elimination example



Query: $P(G|f)$; elimination ordering: A, H, E, D, B, C

$$P(G|f) \propto \sum_C \sum_B \sum_D \sum_E \sum_H \sum_A P(A)P(B|A)P(C|B)$$
$$P(D|C)P(E|D)P(f|E)P(G|C)P(H|E)$$

$$= \sum_C \left(\sum_B \left(\sum_A P(A)P(B|A) \right) P(C|B) \right) P(G|C)$$
$$\left(\sum_D P(D|C) \left(\sum_E P(E|D)P(f|E) \sum_H P(H|E) \right) \right)$$

Stochastic Simulation

- Idea: probabilities $\leftrightarrow$ samples
- Get probabilities from samples:

X	*count*
x_1	n_1
$\vdots$	$\vdots$
x_k	n_k
total	m

$\leftrightarrow$

X	*probability*
x_1	n_1/m
$\vdots$	$\vdots$
x_k	n_k/m

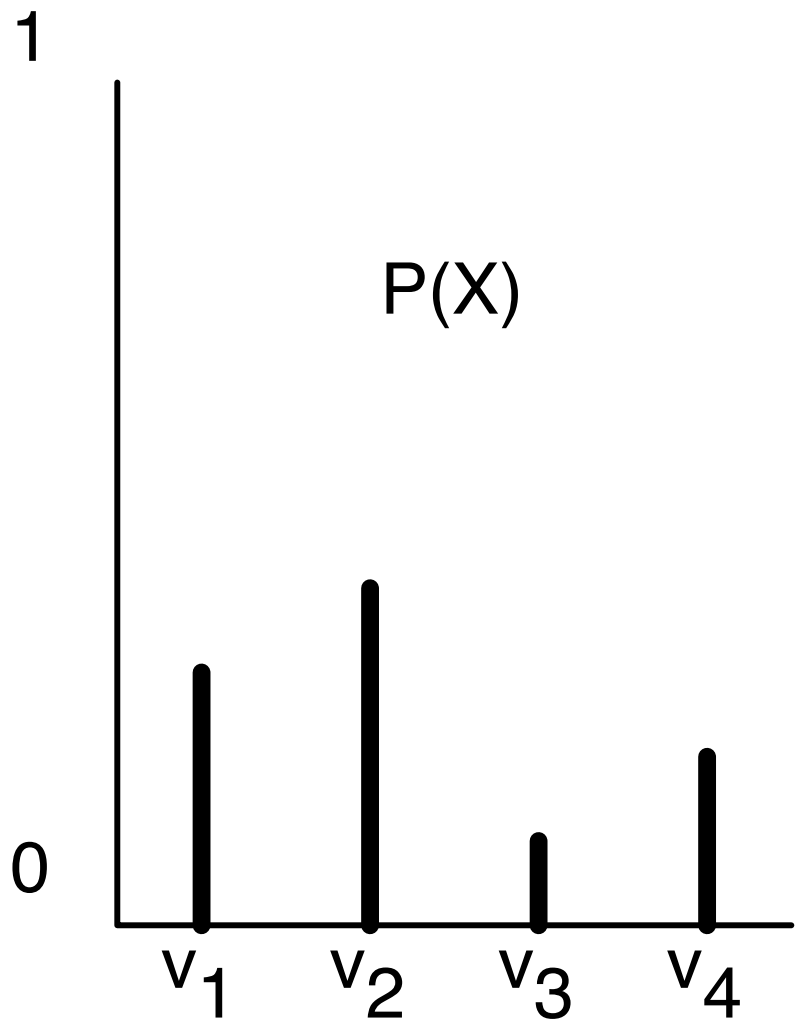
- If we could sample from a variable's (posterior) probability, we could estimate its (posterior) probability.

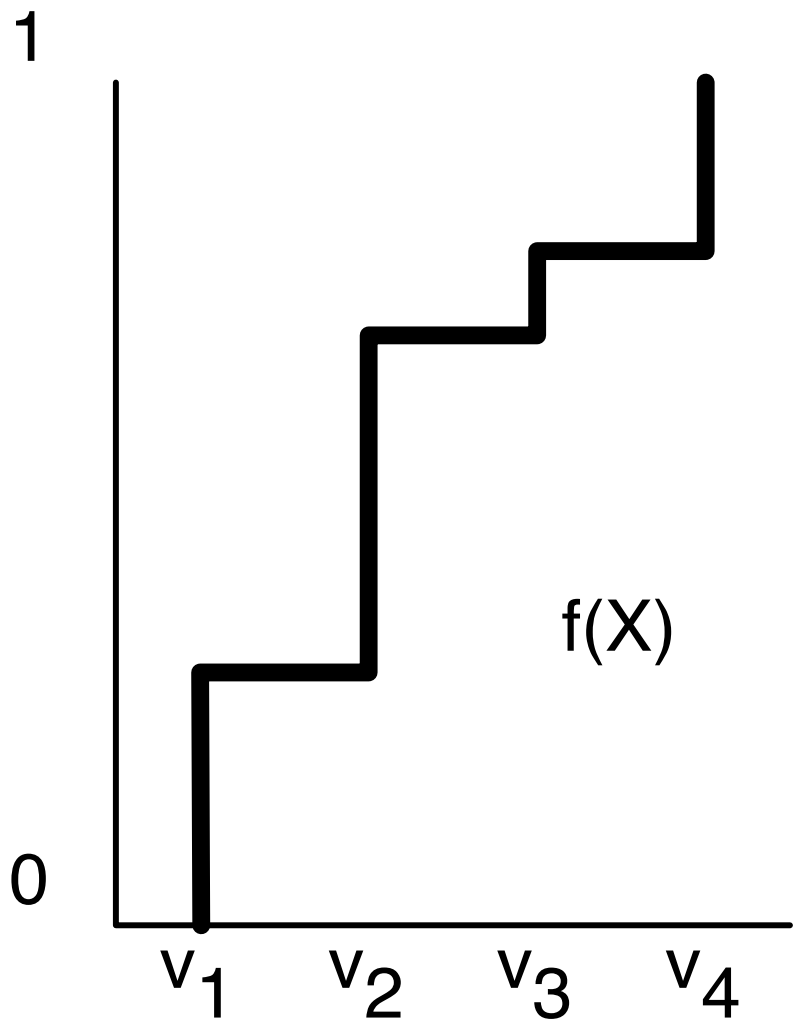
Generating samples from a distribution

For a variable X with a discrete domain or a (one-dimensional) real domain:

- Totally order the values of the domain of X.
- Generate the cumulative probability distribution: $f(x) = P(X \leq x)$.
- Select a value y uniformly in the range $[0, 1]$.
- Select the x such that $f(x) = y$.

Cumulative Distribution





Forward sampling in a belief network

- Sample the variables one at a time; sample parents of X before sampling X.
- Given values for the parents of X, sample from the probability of X given its parents.

Rejection Sampling

- To estimate a posterior probability given evidence $Y_1 = v_1 \wedge \ldots \wedge Y_j = v_j$:
- Reject any sample that assigns Y_i to a value other than v_i.
- The non-rejected samples are distributed according to the posterior probability:

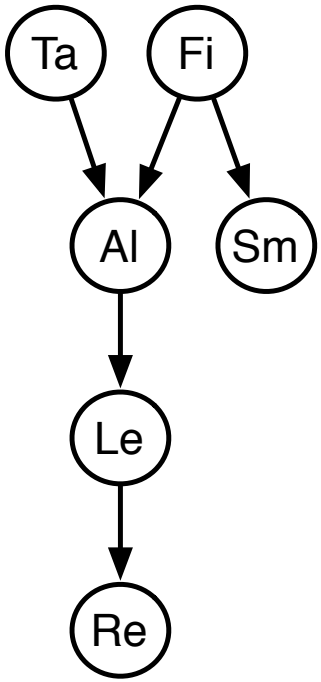
$$P(\alpha|evidence) \approx \frac{\sum_{sample \models \alpha} 1}{\sum_{sample} 1}$$

where we consider only samples consistent with evidence.

Rejection Sampling Example: $P(ta|sm, re)$

Observe $Sm = true, Re = true$

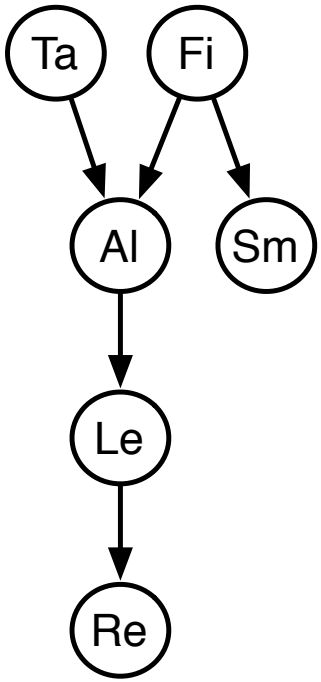
	Ta	Fi	Al	Sm	Le	Re
s_1	false	true	false	true	false	false



Rejection Sampling Example: $P(ta|sm, re)$

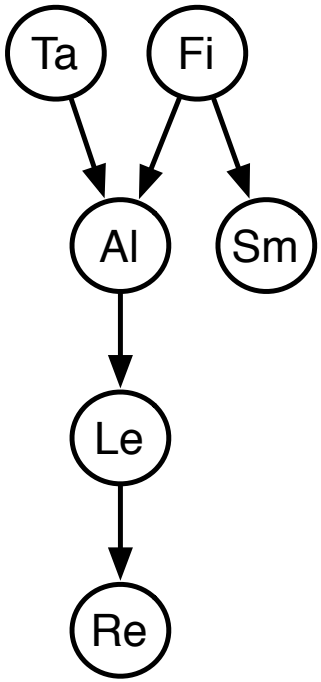
Observe $Sm = true, Re = true$

	Ta	Fi	Al	Sm	Le	Re	
s_1	false	true	false	true	false	false	✘
s_2	false	true	true	true	true	true	



Rejection Sampling Example: $P(ta|sm, re)$

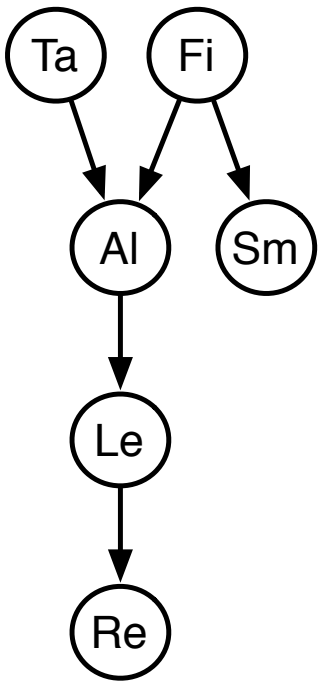
Observe $Sm = true, Re = true$



	Ta	Fi	Al	Sm	Le	Re	
s_1	false	true	false	true	false	false	✘
s_2	false	true	true	true	true	true	✔
s_3	true	false	true	false			

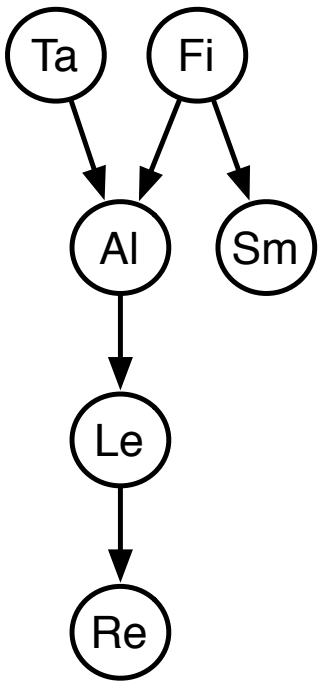
Rejection Sampling Example: $P(ta|sm, re)$

Observe $Sm = true, Re = true$



	Ta	Fi	Al	Sm	Le	Re	
s_1	false	true	false	true	false	false	✘
s_2	false	true	true	true	true	true	✔
s_3	true	false	true	false	—	—	✘
s_4	true	true	true	true	true	true	

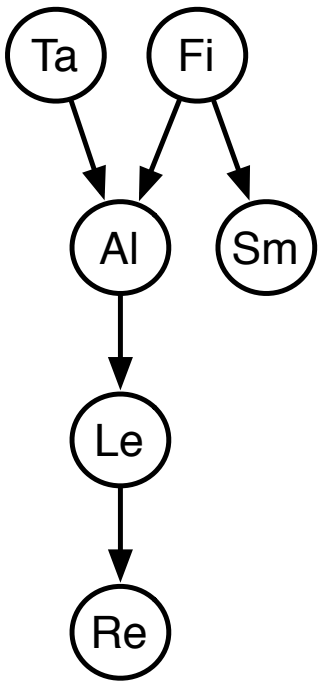
Rejection Sampling Example: $P(ta|sm, re)$



Observe $Sm = true, Re = true$

	Ta	Fi	Al	Sm	Le	Re	
s_1	false	true	false	true	false	false	✘
s_2	false	true	true	true	true	true	✔
s_3	true	false	true	false	—	—	✘
s_4	true	true	true	true	true	true	✔
...							
s_{1000}	false	false	false	false			

Rejection Sampling Example: $P(ta|sm, re)$



Observe $Sm = true, Re = true$

	Ta	Fi	Al	Sm	Le	Re	
s_1	false	true	false	true	false	false	✘
s_2	false	true	true	true	true	true	✔
s_3	true	false	true	false	—	—	✘
s_4	true	true	true	true	true	true	✔
...							
s_{1000}	false	false	false	false	—	—	✘

$P(sm) = 0.02$

$P(re|sm) = 0.32$

How many samples are rejected?

How many samples are used?

Importance Sampling

- Samples have weights: a real number associated with each sample that takes the evidence into account.
- Probability of a proposition is weighted average of samples:

$$P(\alpha|evidence) \approx \frac{\sum_{sample \models \alpha} weight(sample)}{\sum_{sample} weight(sample)}$$

- Mix exact inference with sampling: don't sample all of the variables, but weight each sample according to $P(evidence|sample)$.

Applications of Bayesian Networks

- modelling human multimodal perception
  - human sensor data fusion
  - top down influences in human perception
- multimodal human-computer interaction

Human Sensor Data Fusion

- two general strategies (Ernst and Bülthoff, 2004)
  - sensory combination: maximize information delivered from the different sensory modalities
  - sensory integration: reduce the variance in the sensory estimate to increase its reliability

Sensor Data Fusion

- sensory integration has to produce a coherent percept
- Which modality is the dominating one?
  - visual capture: e.g. vision dominates haptic perception
  - auditory capture: e.g. number of auditory beeps vs. number of visual flashes
- modality precision, modality appropriateness, estimate precision: the most precise modality wins

Sensor Data Fusion

- two possible explanations:
  - maximum likelihood estimation: weighted sum of the individual estimates
    - all cues contribute to the percept
  - cue switching:
    - the most precise cue takes over
    - the less precise cues have no influence

Sensor Data Fusion

- maximum likelihood estimate:
  - weighted sum of the individual estimates
  - weights are proportional to their inverse variance

$$\hat{s} = \sum_i w_i \, \hat{s}_i \text{ with } \sum_i w_i = 1$$

$$w_i = \frac{1/\sigma_i^2}{\sum_j 1/\sigma_j^2}$$

  - most reliable unbiased estimate possible (estimate with minimal variance)
  - optimality not really required; good approximation might be good enough

Sensor Data Fusion

- overwhelming evidence for the role of estimate precision
- weighting within modalities
  - visual depth perception: motion + disparity, texture + disparity
  - visual perception of slant
  - visual perception of distance
  - haptic shape perception: force + position
- cross modal weighting:
  - vision + audition
  - vision + haptic
  - vision + proprioception

Sensor Data Fusion

- no conclusive evidence for the reliability hypothesis so far
- How to estimate the variance of a stimulus?
  - requires an independence assumption
  - difficult to achieve in a unimodal task
  - cues within one modality are correlated
  - $\rightarrow$ multi-modal experiments

Sensor Data Fusion

- Ernst and Banks (2002): vision-haptic integration
  - modifying the visual reliability by adding noise to the visual channel
  - two extreme cases:
    - vision dominates (little noise)
    - haptics dominate (high noise)

$\rightarrow$ perception requires dynamic adjustment of weights

$\rightarrow$ nervous system has online access to sensory reliabilities

Sensor Data Fusion

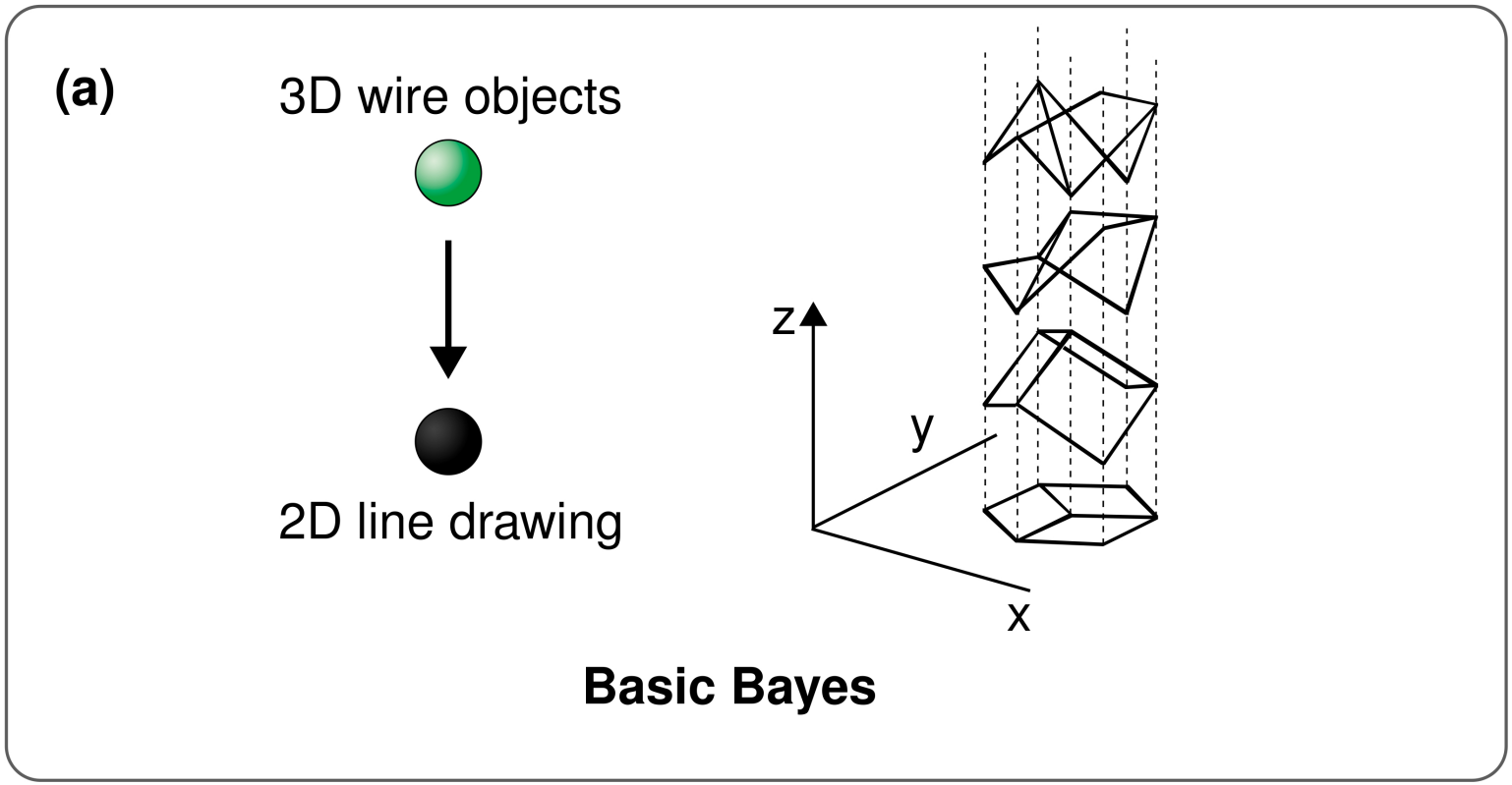
- But where do the estimates come from?

- prior experience vs. on-line estimation during perception
- on-line is more likely: observing the fluctuations of responses to a signal
  - over some period of time
  - across a population of independent neurons (population codes)

Top Down Influence

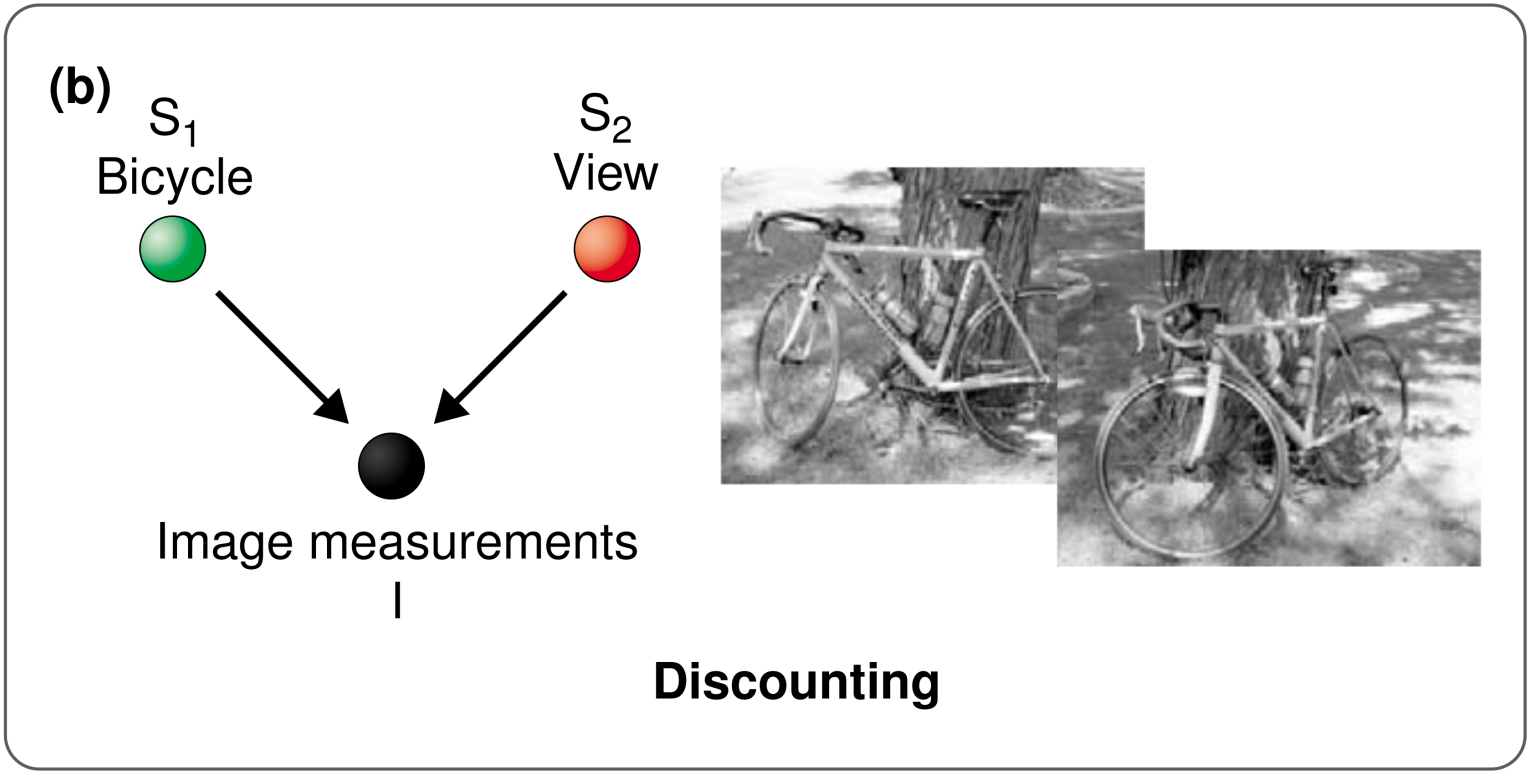
- perception is modulated by contextual factors, e.g scene or object properties
- How to model top-down influences?
  - can be captured by prior probabilities
  - prior probabilities can be integrated by means of Bayes rule
    $\rightarrow$ Bayesian reasoning

Top Down Influence



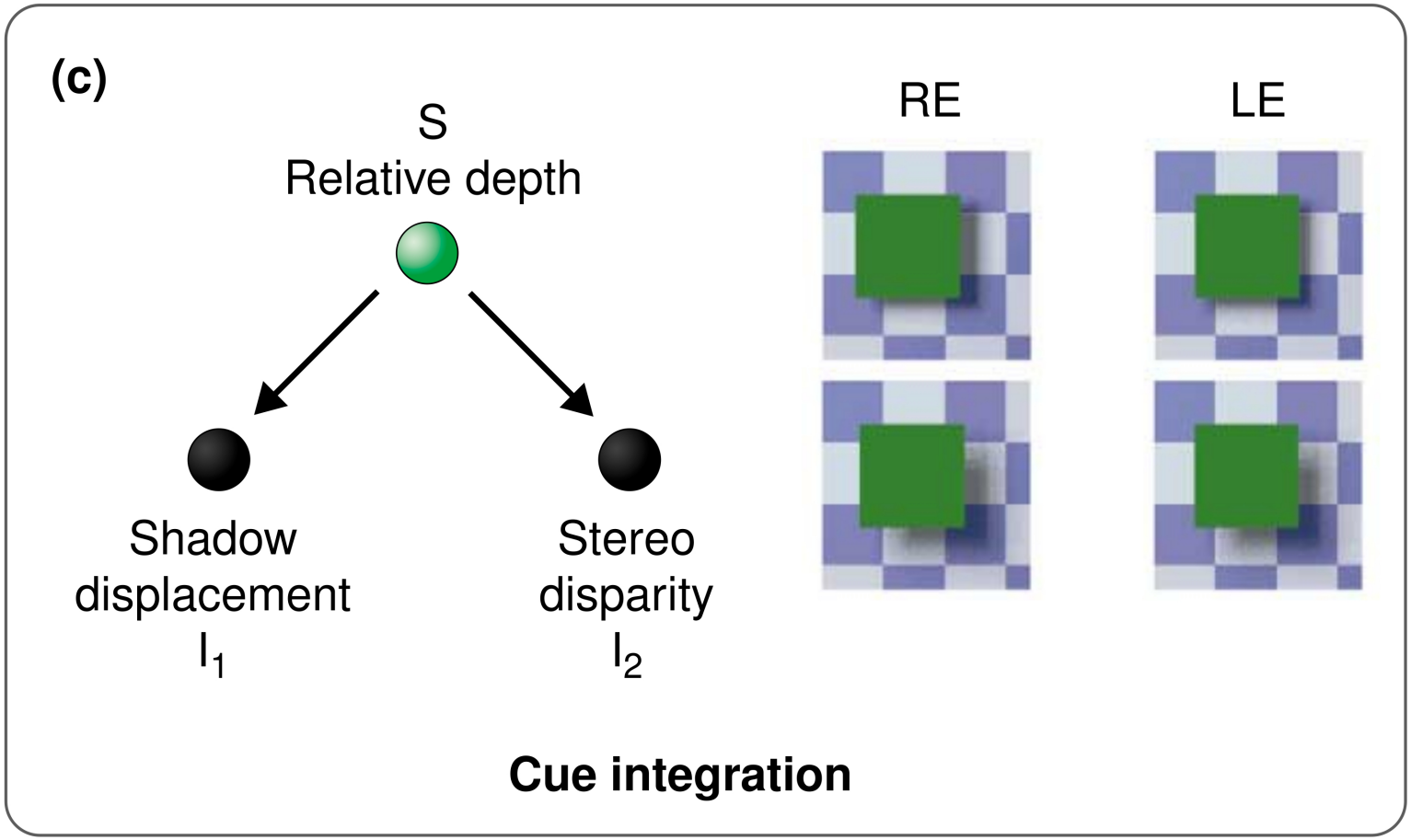
Kersten and Yuilley (2003)

Top Down Influence



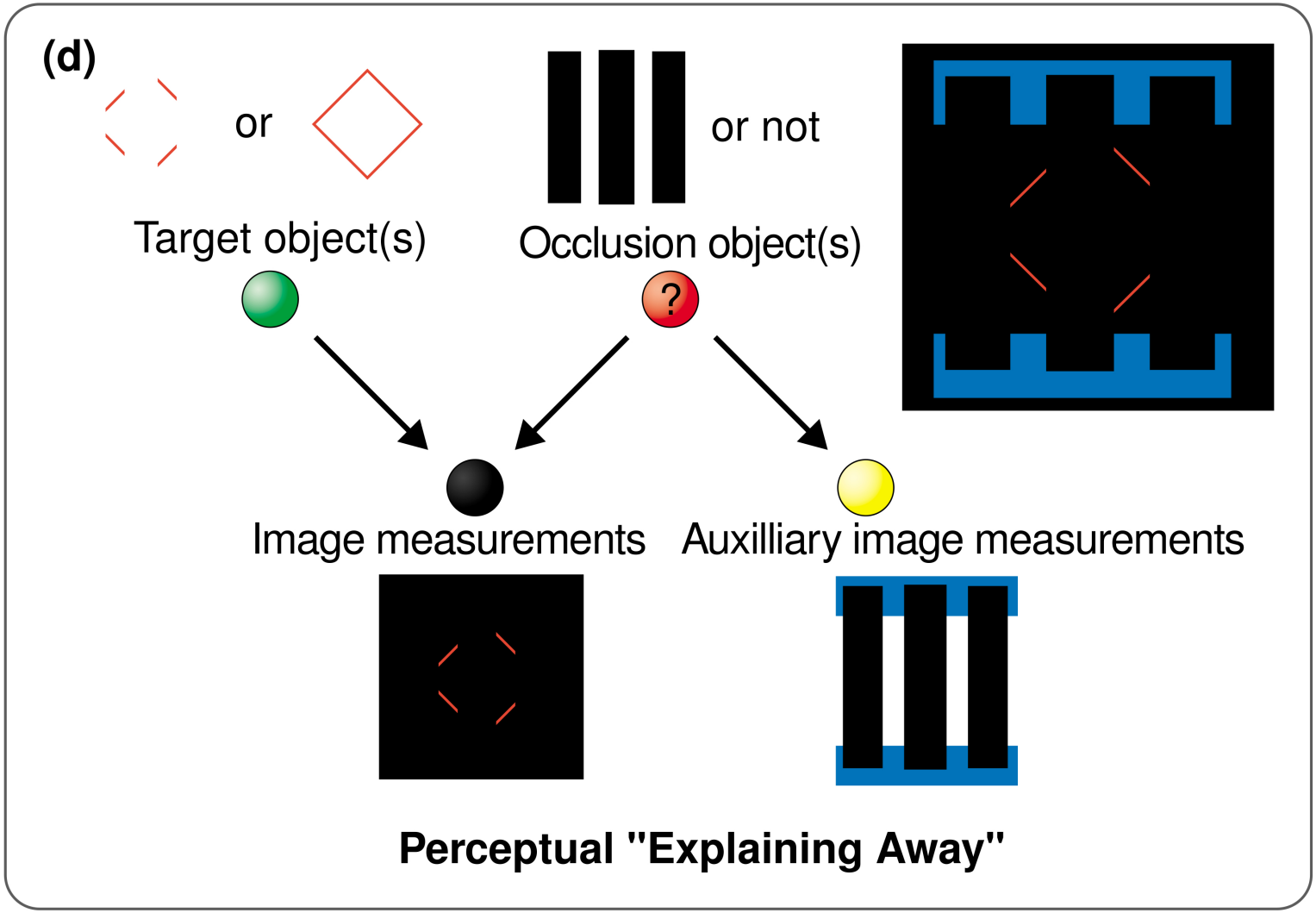
Kersten and Yuilley (2003)

Top Down Influence



Kersten and Yuilley (2003)

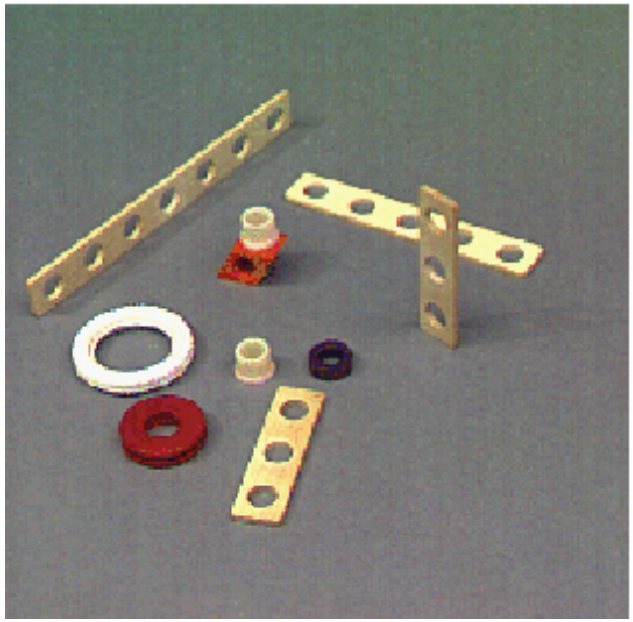
Top Down Influence



Kersten and Yuilley (2003)

Multimodal Human-Computer Interaction

- Socher, Sagerer, Perona (2000), Wachsmuth, Sagerer (2002)
  - multi-modal human machine interaction using
    - speech
    - vision
    - (pointing gestures)



- data fusion from different reference systems
  - spatial (vision) vs. temporal (speech)
  - language based instruction: fusion on the level of concepts

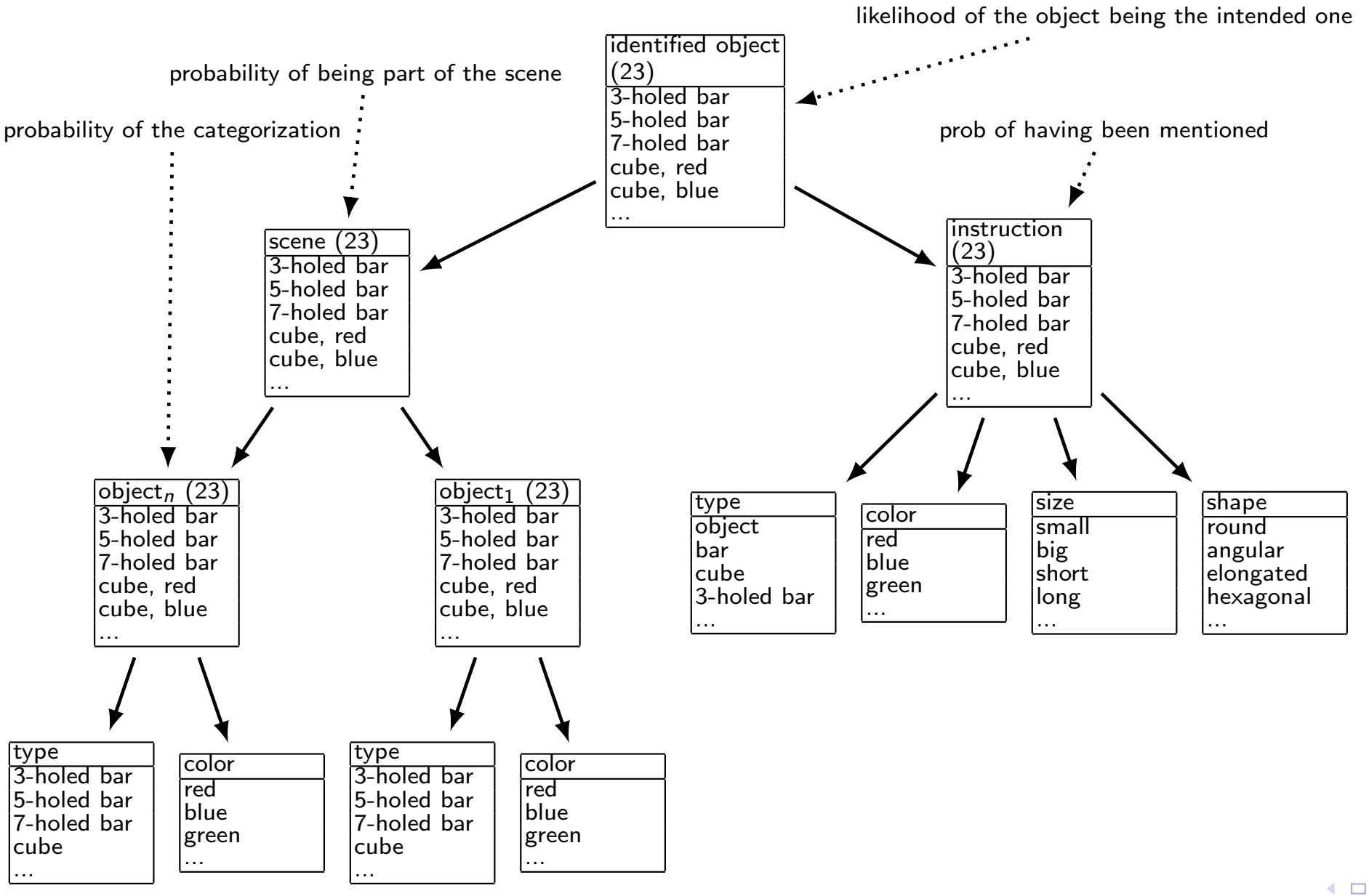
Multimodal Human-Computer Interaction

- noisy and partial interpretation of the sensory signals
- dealing with referential uncertainty
- goal: cross modal synergy

- sensory data: properties (color) and (spatial) relationships: degree-of-membership representation (fuzzyness)

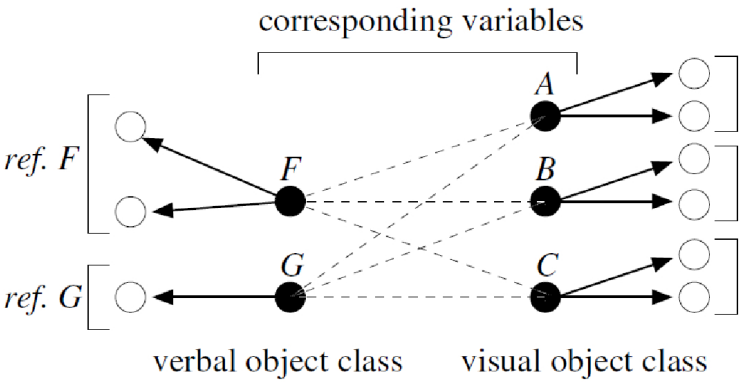
- combination using Bayesian Networks
- estimating the probabilities by means of psycholinguistic experiments
  - how do humans categorize objects and verbalize object descriptions

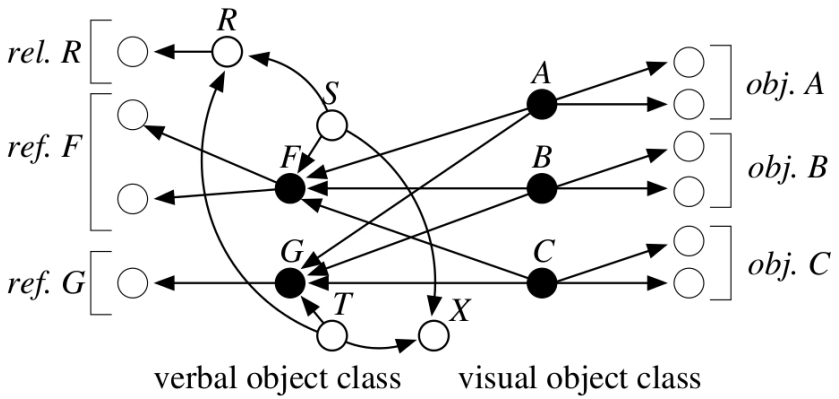
Multimodal Human-Computer Interaction



Multimodal Human-Computer Interaction

- more sophisticated fusion model (Wachsmuth, Sagerer 2002)
  - solution to the correspondence problem using selection variables





- more adequate modelling of naming habits

Multimodal Human-Computer Interaction

- results for object identification

	correct input	noisy speech	noisy vision	noisy input
recognition error rates	–	15%	20%	15%+20%
identification rates	0.85	0.81	0.79	0.76
decrease of identification rates	–	5%	7%	11%

- synergy between vision and speech
- higher robustness due to redundancy between modalities

Bayesian Models for Sequences

- the world is dynamic
  - old information becomes obsolete
  - new information is available
  - the decisions an agent takes need to reflect these changes
- the dynamics of the world can be captured by means of state-based models

Bayesian Models for Sequences

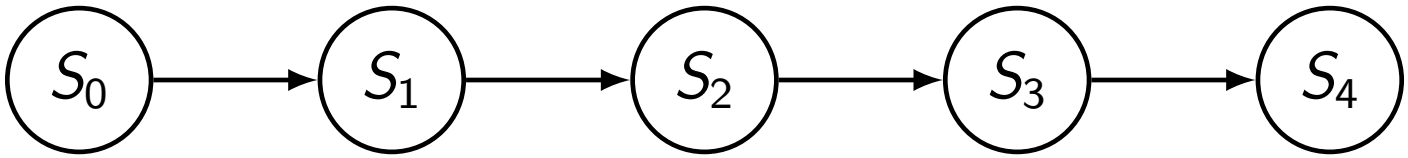
- changes in the world are modelled as transitions between subsequent states
- state transitions can be
  - clocked, e.g.
    - speech: every 10 ms
    - vision: every 40 ms
    - stock market trends: every 24 hours
  - triggered by external events
    - language: every other word
    - travel planning: potential transfer points

Bayesian Models for Sequences

- main purpose:
  - predicting the probability of the next event
  - computing the probability of a (sub-)sequence
- important application areas:
  - speech and language processing, genome analysis, time series predictions (stock market, natural desasters, ...)

Markov chain

- Markov chain: special sort of belief network for sequential observations



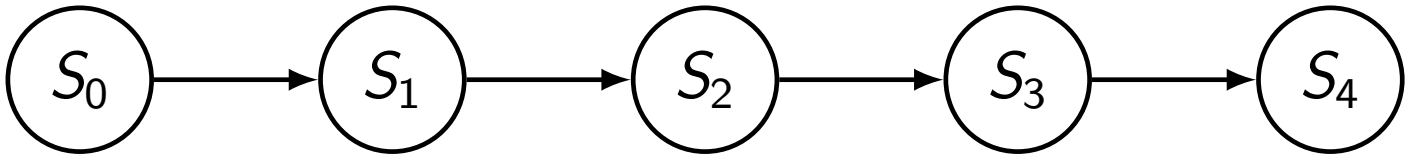
- Thus, $P(S_{t+1}|S_0, \ldots, S_t) = P(S_{t+1}|S_t)$.
- Intuitively S_t conveys all of the information about the history that can affect the future states.
- "The future is independent of the past given the present."

Stationary Markov chain

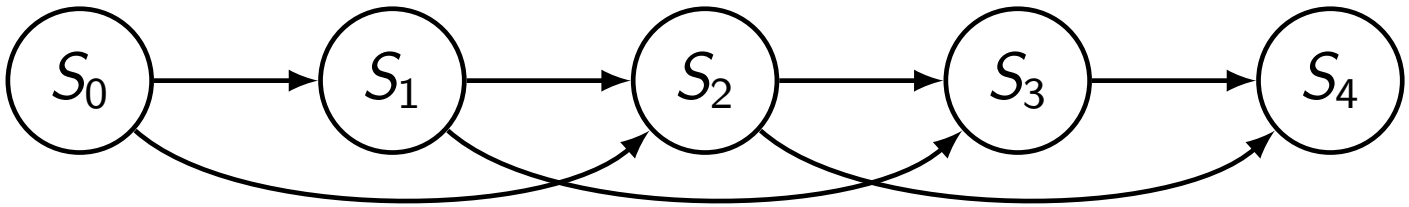
- A stationary Markov chain is when for all $t > 0$, $t' > 0$, $P(S_{t+1}|S_t) = P(S_{t'+1}|S_{t'})$.
- We specify $P(S_0)$ and $P(S_{t+1}|S_t)$.
  - Simple model, easy to specify
  - Often the natural model
  - The network can extend indefinitely

Higher-order Markov Models

- modelling dependencies of various lengths
- bigrams



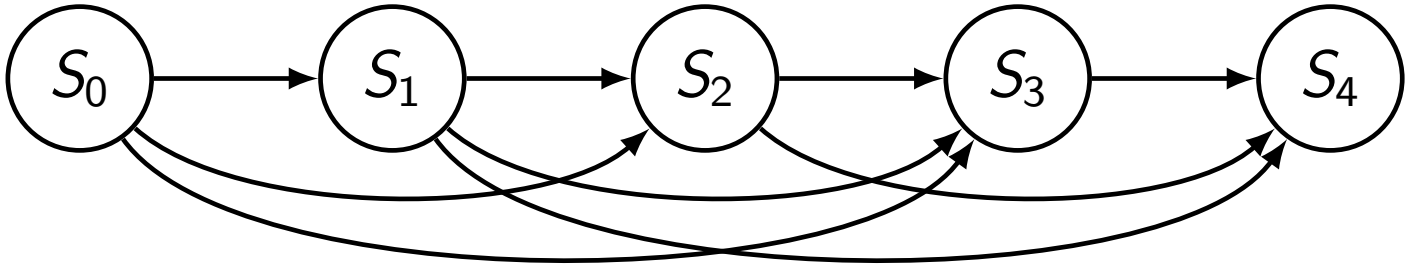
- trigrams



  - three different time slices have to modelled
    - for S_0: $P(S_0)$
    - for S_1: $P(S_1|S_0)$
    - for all others: $P(S_i)|S_{i-2}S_{i-1})$

Higher-order Markov Models

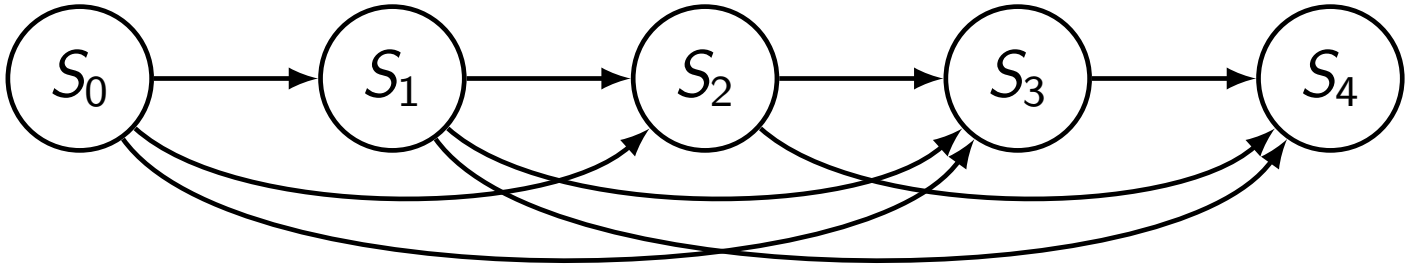
- quadrograms: $P(S_i|S_{i-3}S_{i-2}S_{i-1})$



- four different kinds of time slices required

Higher-order Markov Models

- quadrograms: $P(S_i|S_{i-3}S_{i-2}S_{i-1})$



- four different kinds of time slices required

- Markov models can be applied to predict the probability of the next event
- e.g. for speech and language processing, genome analysis, time series predictions (stock market, natural desasters, ...)

Markov Models

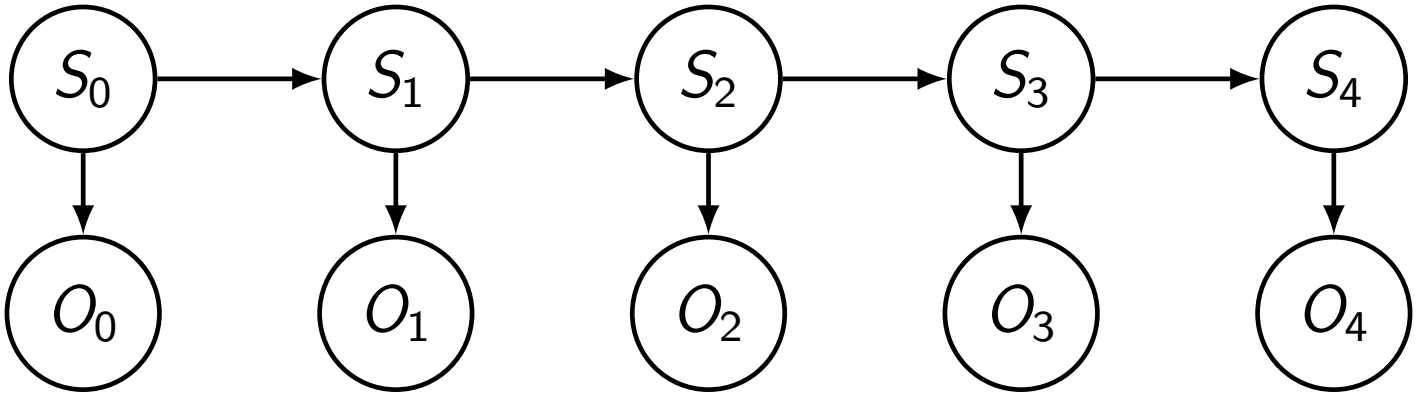
- examples of Markov chains for German letter sequences
- unigrams:
  aiobnin*tarsfneonlpiitdregedcoa*ds*e*dbieastnreleeucdkeaitb*
  dnurlarsls*omn*keu**svdleeoieei* ...
- bigrams:
  er*agepteprteiningeit*gerelen*re*unk*ves*mterone*hin*d*an*
  nzerurbom* ...
- trigrams:
  billunten*zugen*die*hin*se*sch*wel*war*gen*man*
  nicheleblant*diertunderstim* ...
- quadrograms:
  eist*des*nich*in*den*plassen*kann*tragen*was*wiese*
  zufahr* ...

Hidden Markov Model

- Often the observation does not deterministically depend on the state of the model
- This can be captured by a Hidden Markov Model (HMM)
- ... even if the state transitions are not directly observable

Hidden Markov Model

- A HMM is a belief network where states and observations are separated



- $P(S_0)$ specifies initial conditions
- $P(S_{t+1}|S_t)$ specifies the dynamics
- $P(O_t|S_t)$ specifies the sensor model

Hidden Markov Models

A Hidden Markov Model consists of

- $\mathcal{S}$: a finite set of states s_i
- $\mathcal{O}$: a finite set of observations o_i
- transition probabilities $\mathcal{T} : \mathcal{S} \times \mathcal{S} \mapsto \mathbb{R}_+$
- emission probabilities $\mathcal{E} : \mathcal{S} \times \mathcal{O} \mapsto \mathbb{R}_+$
- prior probabilities (for the initial states): $\pi : \mathcal{S} \mapsto \mathbb{R}_+$

Hidden Markov Models

A Hidden Markov Model consists of

- $\mathcal{S}$: a finite set of states s_i
- $\mathcal{O}$: a finite set of observations o_i
- transition probabilities $\mathcal{T} : \mathcal{S} \times \mathcal{S} \mapsto \mathbb{R}_+$
- emission probabilities $\mathcal{E} : \mathcal{S} \times \mathcal{O} \mapsto \mathbb{R}_+$
- prior probabilities (for the initial states): $\pi : \mathcal{S} \mapsto \mathbb{R}_+$

- HMMs are a special case of belief networks
  $\rightarrow$ arbitrary distributions can be computed by means of variable elimination
- HMMs make strong assumptions about the model topology
  $\rightarrow$ special algorithms for inference are available

Inference with Hidden Markov Models

Evaluation:

- What's the probability of an HMM λ having produced an observation sequence $O = (o_1, o_2, \ldots, o_t)$
- problem: hidden state sequence $S = (s_1, s_2, \ldots, s_t)$ is not known

$$P(O|\lambda) = \sum_{\forall S} P(O|S, \lambda)\ P(S|\lambda)$$

with

$$P(O|S, \lambda) = \prod_{i=1}^{t} P(O_i = o_i|S_i = s_i) = E_{s_1,o_1}\ E_{s_2,o_2} \ldots E_{s_t,o_t}$$

$$P(S|\lambda) = P(S_1 = s_1) \prod_{i=2}^{t} P(S_i = s_i|S_{i-1} = s_{i-1})$$

$$= \pi_{s_1}\ T_{s_1,s_2}\ T_{s_2,s_3} \ldots T_{s_{t-1},s_t}$$

Inference with Hidden Markov Models

Forward algorithm

- there are exponentially many state sequences
  $\rightarrow$ naive computation requires exponentially many multiplications: $\mathcal{O}(t \cdot |\mathcal{S}|^t)$
- efficient calculation using a recursive reformulation based on the Markov property
- forward coefficients: $\alpha_k(s) = P(O_{1:k}, S_k = s|\lambda)$

Inference with Hidden Markov Models

Forward algorithm

- initialize $\alpha_1(s) = \pi_s \, E_{s,o_1} \quad \forall s \in \mathcal{S}$
- repeat, for $k = 1$ to $k = t - 1$ and for all $s \in \mathcal{S}$

$$\alpha_{k+1}(s) = E_{s,o_{k+1}} \sum_{q \in \mathcal{S}} \alpha_k(q) \, T_{q,s}$$

- aggregate:

$$P(O|\lambda) = \sum_{s \in \mathcal{S}} \alpha_t(s)$$

- complexity reduced to $\mathcal{O}(t \cdot |\mathcal{S}|^2)$

Inference with Hidden Markov Models

Backward algorithm

- initialize $\beta_k(s) = 1 \quad \forall s \in \mathcal{S}$
- repeat, for $k = t - 1$ to $k = 1$ and for all $s \in \mathcal{S}$

$$\beta_k(s) = \sum_{q \in \mathcal{S}} \beta_{k+1}(q)\, T_{s,q}\, E_{q,o_{k+1}}$$

- aggregate:

$$P(O|\lambda) = \sum_{s \in \mathcal{S}} \pi(s)\, \beta_1(s)\, E_{s,o_1}$$

- complexity reduced to $\mathcal{O}(t \cdot |\mathcal{S}|^2)$

Inference with Hidden Markov Models

Explanation

- Filtering: $P(S_k|O_{1:k})$
  given an observation sequence $O_{1:k} = (o1, o2, \ldots, o_k)$
  compute the distribution of S_k
- Smoothing: $P(S_j|O_{1:k}), j < k$
  given an observation sequence $O_{1:k} = (o1, o2, \ldots, o_k)$
  compute the distribution of $S_j, j < k$
- Prediction: $P(S_j|O_{1:k}), j > k$
  given an observation sequence $O_{1:k} = (o1, o2, \ldots, o_k)$
  compute the distribution of $S_j, j > k$
- Decoding: $\hat{S}_{1_k} = \arg\max_S P(S_{1:k}|O_{1:k})$
  given an observation sequence $O_{1:t} = (o1, o2, \ldots, o_k)$
  compute the most likely state sequence $S_{1:k}$

Filtering

Filtering:

$$P(S_i|o_1, \ldots, o_i)$$

What is the current belief state based on the observation history?

Filtering

Filtering:

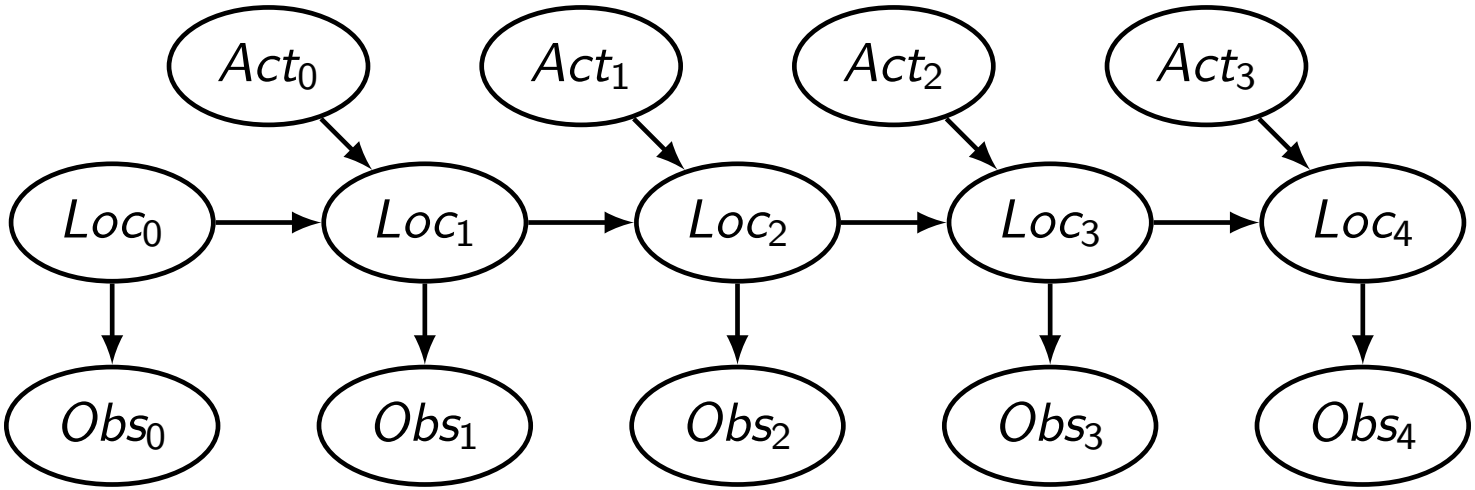
$$P(S_i|o_1,\ldots,o_i)$$

What is the current belief state based on the observation history?

$$\begin{aligned}
P(S_i|o_1,\ldots,o_i) &\propto P(S_i,o_1,\ldots,o_i)\\
&= P(o_i|S_i)P(S_i,o_1,\ldots,o_{i-1})\\
&= P(o_i|S_i)\sum_{S_{i-1}} P(S_i,S_{i-1},o_1,\ldots,o_{i-1})\\
&= P(o_i|S_i)\sum_{S_{i-1}} P(S_i|S_{i-1})P(S_{i-1},o_1,\ldots,o_{i-1})\\
&\propto P(o_i|S_i)\sum_{S_{i-1}} P(S_i|S_{i-1})P(S_{i-1}|o_1,\ldots,o_{i-1})
\end{aligned}$$

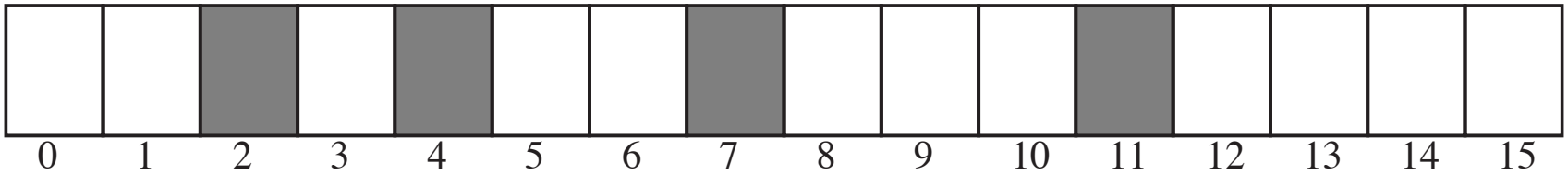
Example (1): robot localization

- Suppose a robot wants to determine its location based on its actions and its sensor readings: Localization
- This can be represented by the augmented HMM:



Example localization domain

- Circular corridor, with 16 locations:



- Doors at positions: 2, 4, 7, 11.
- Noisy Sensors
- Stochastic Dynamics
- Robot starts at an unknown location and must determine where it is.

Example Sensor Model

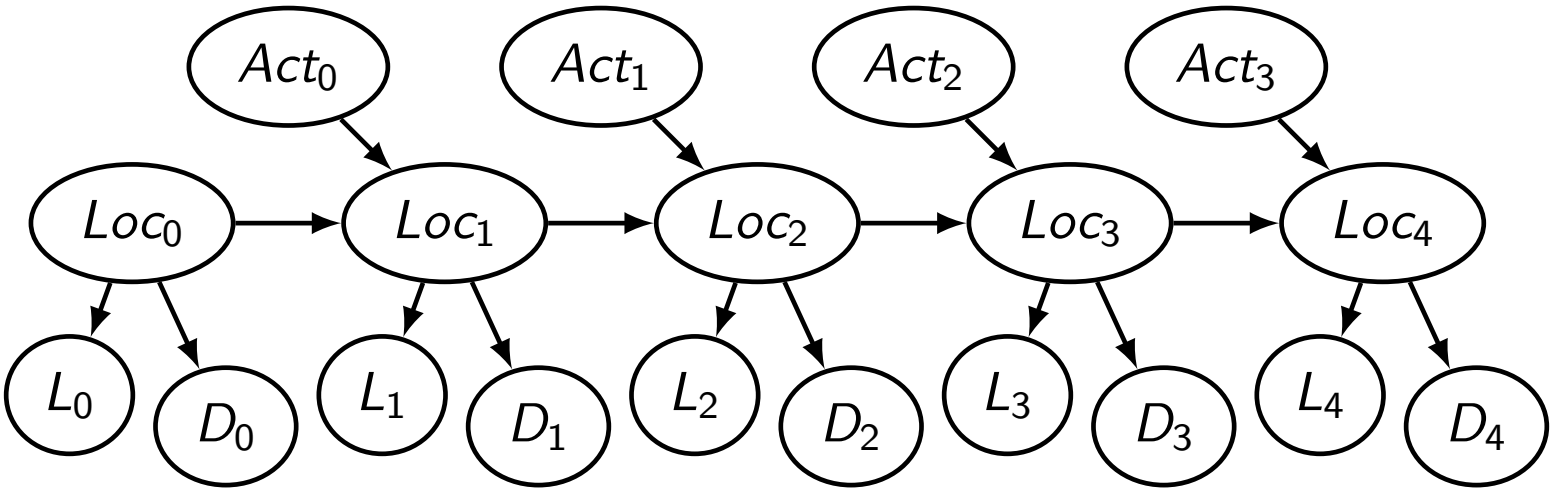
- $P(\mathit{Observe\ Door} \mid \mathit{At\ Door}) = 0.8$
- $P(\mathit{Observe\ Door} \mid \mathit{Not\ At\ Door}) = 0.1$

Example Dynamics Model

- $P(loc_{t+1} = L | action_t = goRight \wedge loc_t = L) = 0.1$
- $P(loc_{t+1} = L + 1 | action_t = goRight \wedge loc_t = L) = 0.8$
- $P(loc_{t+1} = L + 2 | action_t = goRight \wedge loc_t = L) = 0.074$
- $P(loc_{t+1} = L' | action_t = goRight \wedge loc_t = L) = 0.002$ for any other location L'.
  - All location arithmetic is modulo 16.
  - The action *goLeft* works the same but to the left.

Combining sensor information

- the robot can have many (noisy) sensors for signals from the environment
- e.g. a light sensor in addition to the door sensor
- Sensor Fusion: combining information from different sources

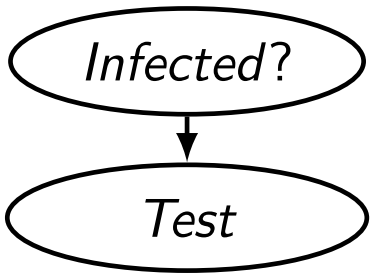


D_t door sensor value at time t

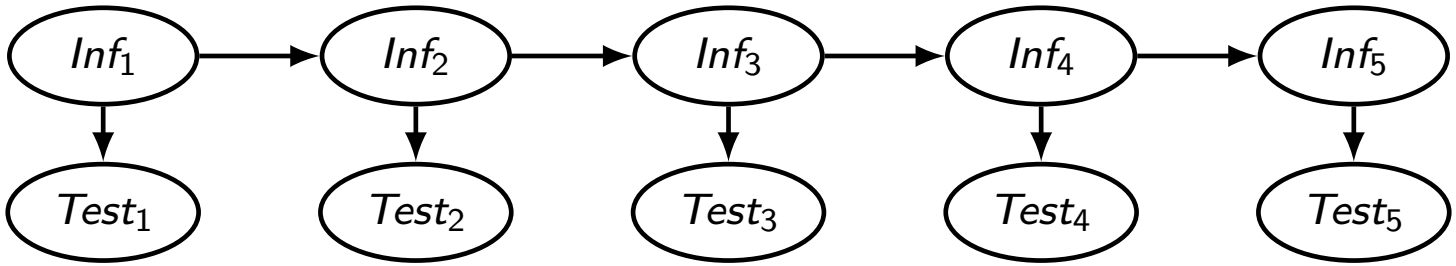
L_t light sensor value at time t

Hidden Markov Models

- Example (2): medical diagnosis (milk infection test) (Jensen and Nielsen 2007)
- the probability of the test outcome depends on the cow being infected or not

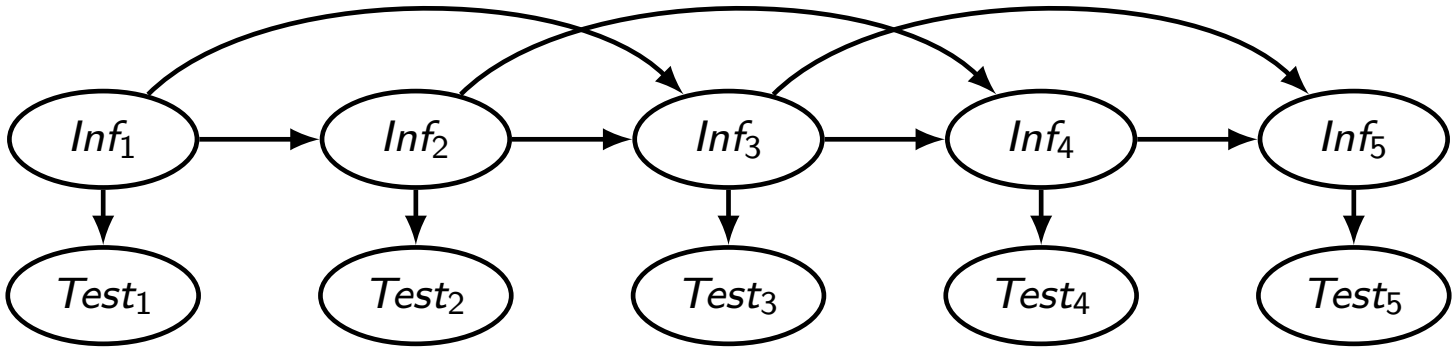


- the probability of the cow being infected depends on the cow being infected on the previous day
  - first order Markov model



Hidden Markov Models

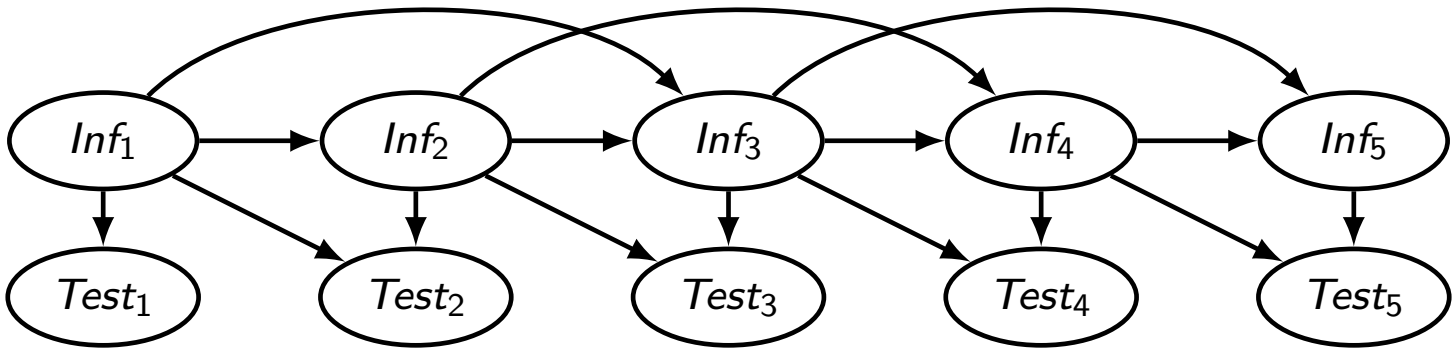
- the probability of the cow being infected depends on the cow being infected on the two previous days
  - incubation and infection periods of more than one day
  - second order Markov model



  - assumes only random test errors
- weaker independence assumptions
  - more powerful model
  - more data required for training

Hidden Markov Models

- the probability of the test outcome also depends on the cow's health and the test outcome on the previous day
  - can also capture systematic test errors
  - second order Markov model for the infection
  - first order Markov model for the test results



Inference in Hidden Markov Models

Decoding: What's the state sequence which most likely produced the observation?

- filtering and smoothing produce probability distributions for the values of a state variable
- choosing the value with the highest probability gives only a pointwise best estimation
  - a sequence of pointwise best estimation need not be the best state value sequence
  - the model need not even be able to produce the pointwise best sequence

Inference in Hidden Markov Models

- VITERBI coefficients:

$$\delta_k(s) = \max_{S_{1:k-1}} P(S_{1:k} = (S_{1:k-1}, s), O_{1:k}|\lambda)$$

  - $\delta_k(s)$ is the probability of the most likely path ending in state s and generating the observation sequence $O_{1:k}$

- because of the Markov property the computation simplifies to

$$\delta_{k+1}(s) = \max_q (\delta_k(q)\, T_{q,s})\, E_{s,o_{k+1}}$$

Inference in Hidden Markov Models

- VITERBI coefficients:

$$\delta_k(s) = \max_{S_{1:k-1}} P(S_{1:k} = (S_{1:k-1}, s), O_{1:k}|\lambda)$$

  - $\delta_k(s)$ is the probability of the most likely path ending in state s and generating the observation sequence $O_{1:k}$

- because of the Markov property the computation simplifies to

$$\delta_{k+1}(s) = \max_q (\delta_k(q)\, T_{q,s})\, E_{s,o_{k+1}}$$

  this corresponds to the principle of dynamic programming

Inference in Hidden Markov Models

- VITERBI coefficients:

$$\delta_k(s) = \max_{S_{1:k-1}} P(S_{1:k} = (S_{1:k-1}, s), O_{1:k}|\lambda)$$

  - $\delta_k(s)$ is the probability of the most likely path ending in state s and generating the observation sequence $O_{1:k}$
- because of the Markov property the computation simplifies to

$$\delta_{k+1}(s) = \max_q (\delta_k(q)\, T_{q,s})\, E_{s,o_{k+1}}$$

  this corresponds to the principle of dynamic programming
- first computing the deltas in a forward pass
- afterwards reconstructing the best path by means of pointers p from each state q_k to its most likely predecessor q_{k-1}

Inference in Hidden Markov Models

VITERBI algorithm

- intitialize for all $s \in \mathcal{S}$
  - $\delta_1(S) = \pi_s \; E_{s,z_1}$
  - $p_1(s) = \mathit{null}$
- repeat recursively
  - $\delta_{k+1}(s) = \max_q(\delta_k(q) \; T_{q,s}) \; E_{s,o_{k+1}}$
  - $p_{k+1}(s) = \arg\max_q(\delta_k(q) \; T_{q,s})$
- select the most likely terminal state $\hat{s}_t = \arg\max_s \delta_t(s)$ with $\hat{p} = \delta(\hat{s}_t)$ being the probability of the most likely path
- reconstruct the most likely path backwards: $\hat{q}_k = p_{k+1}(\hat{q}_{k+1})$

Inference in Hidden Markov Models

VITERBI algorithm

- is similar to the forward algorithm
- uses maximization instead of summation

Inference in Hidden Markov Models

VITERBI algorithm

- is similar to the forward algorithm
- uses maximization instead of summation
- has many applications signal processing, pattern recognition, biocomputing, natural language processing, etc.
  - message reconstruction for noisy wireless communication
  - speech recognition / speech synthesis
  - machine translation
  - swype keyboards
  - intron/exon detection
  - ...

Hidden Markov Models

- Example (3): Tagging for Natural Language Processing
- annotating the word forms in a sentence with

part-of-speech information
$Yesterday_{RB}$ the_{DT} $school_{NNS}$ was_{VBD} $closed_{VBN}$

topic areas: *He did some field work.*
$field_{military}$, $field_{agriculture}$, $field_{physics}$, $field_{social\ sci.}$, $field_{optics}$, ...

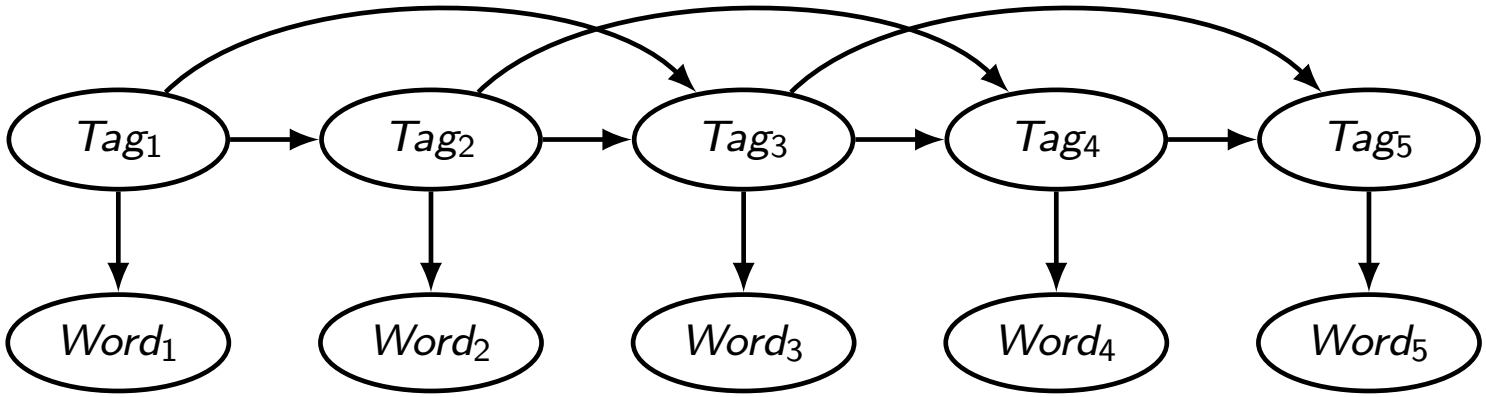
semantic roles
$The\ winner_{Beneficiary}$ $received\ the\ trophy_{Theme}$ $at\ the$
$town\ hall_{Location}$

Hidden Markov Models

- sequence labelling problem
  - the label depends on the current state and the most recent history
- one-to-one correspondence between states, tags, and word forms

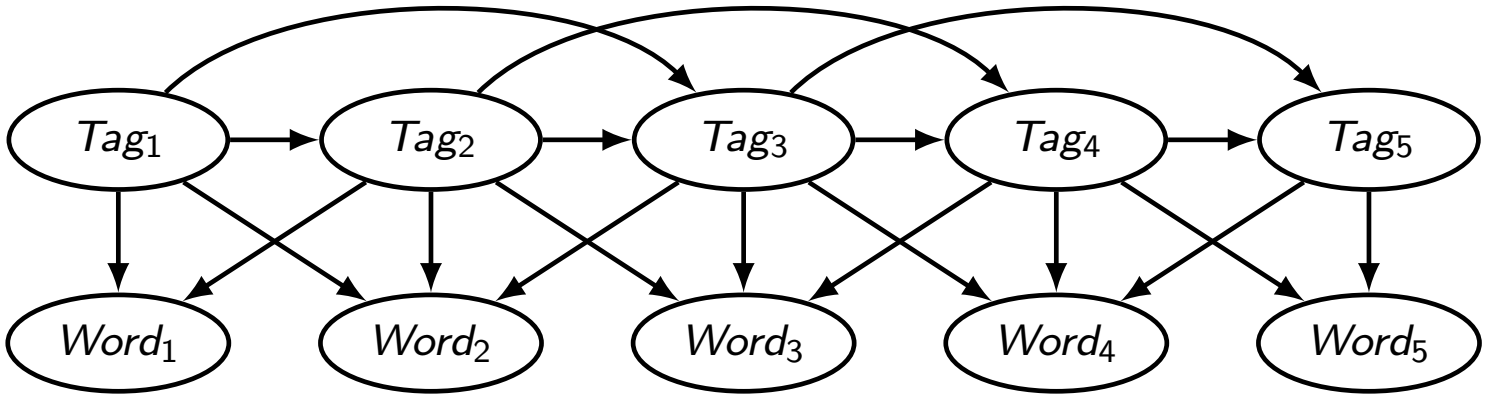
Hidden Markov Models

- causal (generative) model of the sentence generation process
  - tags are assigned to states
  - the underlying state (tag) sequence produces the observations (word forms)
- typical independence assumptions
  - trigram probabilities for the state transitions
  - word form probabilities depend only on the current state



Hidden Markov Model

- weaker independence assumption (stronger model):
  - the probability of a word form also depends on the previous and subsequent state



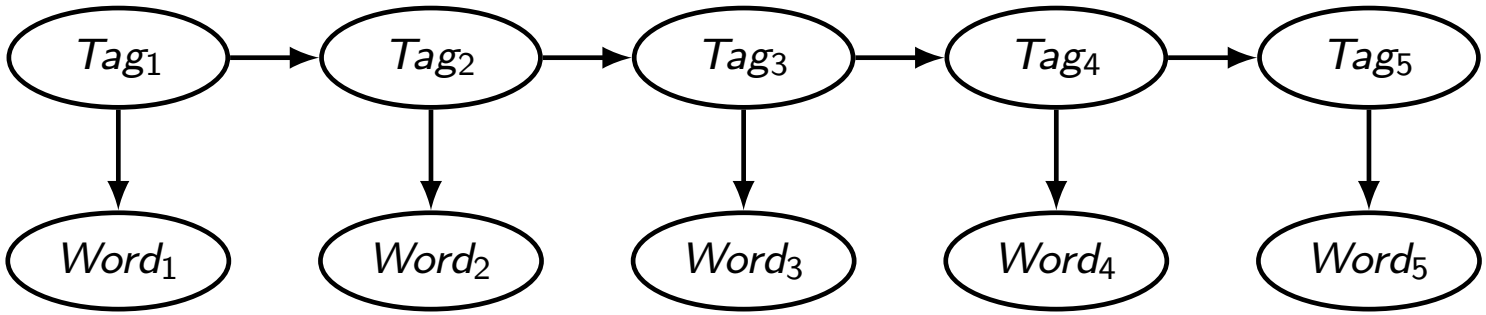
Two alternative graphical representations

- influence diagrams, belief networks, Bayesian networks, causal networks, graphical models, ...
- state transition diagrams (probabilistic finite state machines)

	Bayesian networks	State transition diagrams
state nodes	variables with states as values	states
edges into state nodes	causal influence and their probabilities	possible state transitions
# state nodes	length of the observation sequence	# model states
observation nodes	variables with observations as values	observation values
edges into observ. nodes	conditional probability tables	conditional probabilities

Two alternative graphical representations

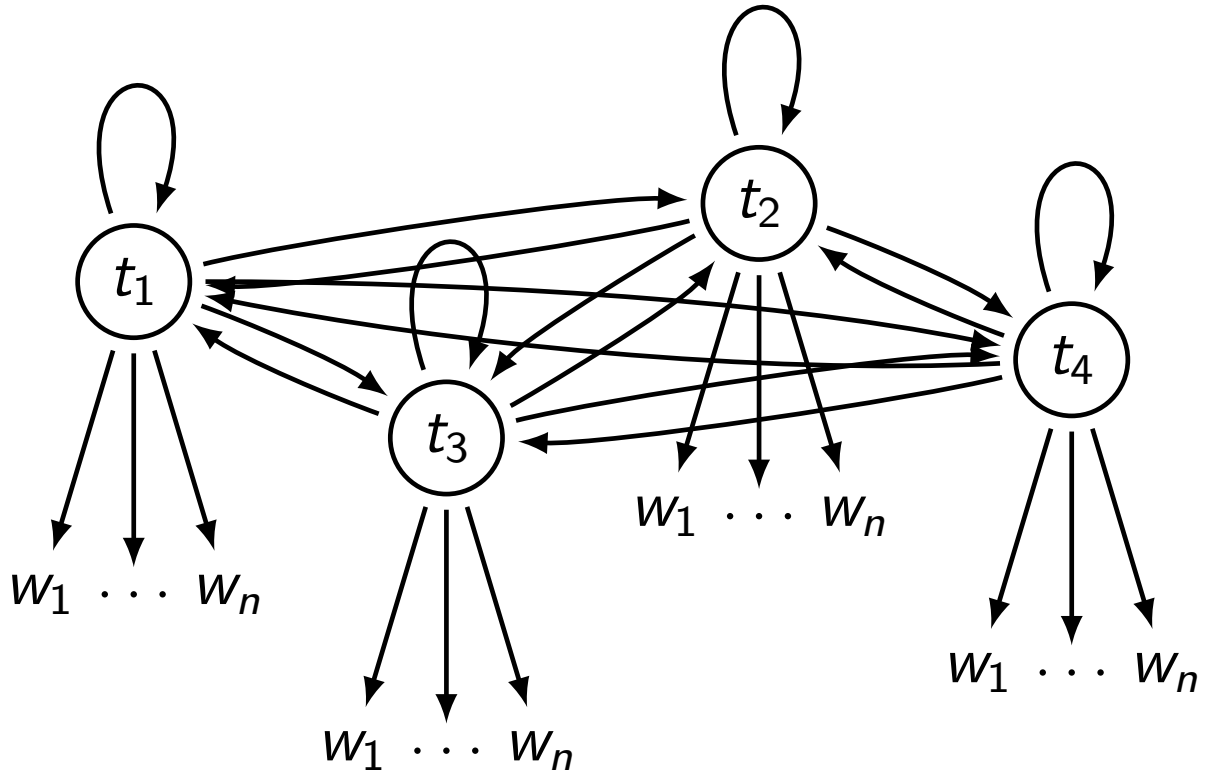
- Bigram-tagging as a Bayesian network



- possible state transitions are not directly visible
  - indirectly encoded in the conditional probability tables
- sometimes state transition diagrams are better suited to illustrate the model topology

Two alternative graphical representations

- Bigram-Tagging as a state transition diagram (can only be depicted for bigram models)



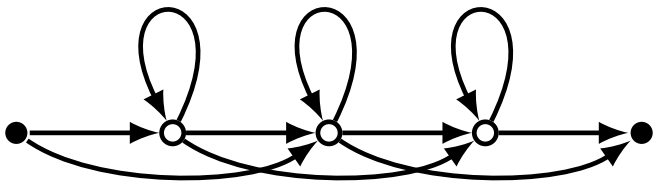
- ergodic model: full connectivity between all states

Hidden Markov Models

- Example (4): Speech Recognition, Swype gesture recognition
- observation subsequences of unknown length are mapped to one label
  $\rightarrow$ alignment problem
- full connectivity is not desired
- a phone/syllable/word realization cannot be reversed

Hidden Markov Models

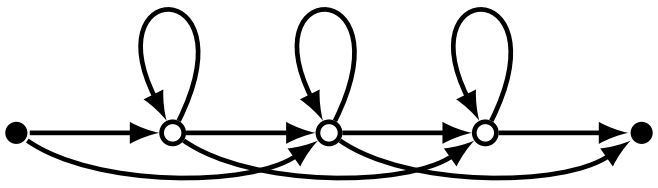
- possible model topologies for phones (only transitions depicted)



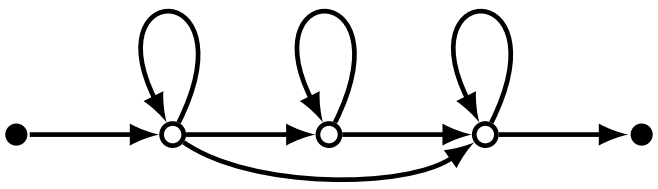
$$\begin{array}{ccccc} P(1|0) & P(1|1) & 0 & 0 & 0 \\ P(2|0) & P(2|1) & P(2|2) & 0 & 0 \\ 0 & P(3|1) & P(3|2) & P(3|3) & 0 \\ 0 & 0 & P(4|2) & P(4|3) & 0 \end{array}$$

Hidden Markov Models

- possible model topologies for phones (only transitions depicted)



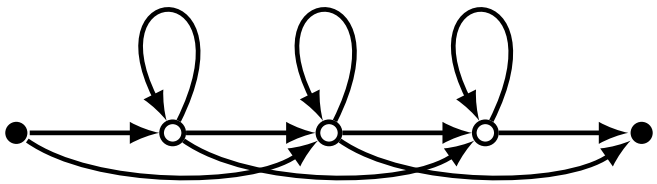
$$\begin{matrix} P(1|0) & P(1|1) & 0 & 0 & 0 \\ P(2|0) & P(2|1) & P(2|2) & 0 & 0 \\ 0 & P(3|1) & P(3|2) & P(3|3) & 0 \\ 0 & 0 & P(4|2) & P(4|3) & 0 \end{matrix}$$



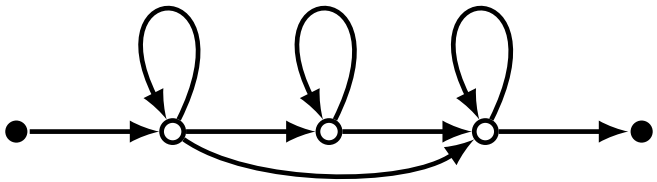
$$\begin{matrix} P(1|0) & P(1|1) & 0 & 0 & 0 \\ 0 & P(2|1) & P(2|2) & 0 & 0 \\ 0 & P(3|1) & P(3|2) & P(3|3) & 0 \\ 0 & 0 & 0 & P(4|3) & 0 \end{matrix}$$

Hidden Markov Models

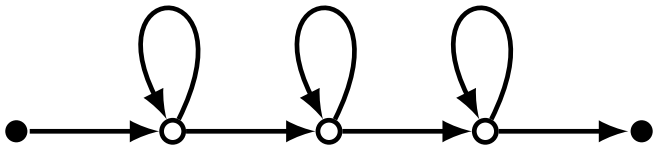
- possible model topologies for phones (only transitions depicted)



$$\begin{matrix} P(1|0) & P(1|1) & 0 & 0 & 0 \\ P(2|0) & P(2|1) & P(2|2) & 0 & 0 \\ 0 & P(3|1) & P(3|2) & P(3|3) & 0 \\ 0 & 0 & P(4|2) & P(4|3) & 0 \end{matrix}$$



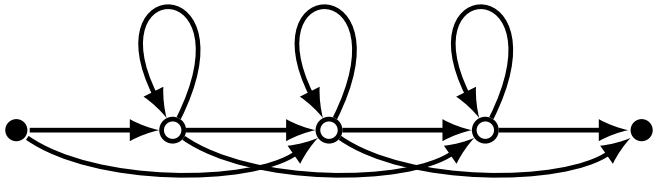
$$\begin{matrix} P(1|0) & P(1|1) & 0 & 0 & 0 \\ 0 & P(2|1) & P(2|2) & 0 & 0 \\ 0 & P(3|1) & P(3|2) & P(3|3) & 0 \\ 0 & 0 & 0 & P(4|3) & 0 \end{matrix}$$



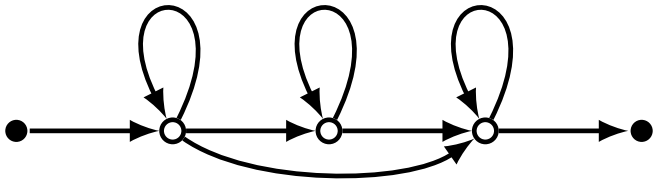
$$\begin{matrix} P(1|0) & P(1|1) & 0 & 0 & 0 \\ 0 & P(2|1) & P(2|2) & 0 & 0 \\ 0 & 0 & P(3|2) & P(3|3) & 0 \\ 0 & 0 & 0 & P(4|3) & 0 \end{matrix}$$

Hidden Markov Models

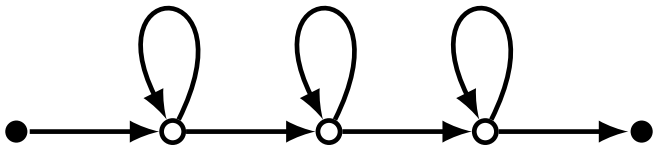
- possible model topologies for phones (only transitions depicted)



$$\begin{matrix} P(1|0) & P(1|1) & 0 & 0 & 0 \\ P(2|0) & P(2|1) & P(2|2) & 0 & 0 \\ 0 & P(3|1) & P(3|2) & P(3|3) & 0 \\ 0 & 0 & P(4|2) & P(4|3) & 0 \end{matrix}$$



$$\begin{matrix} P(1|0) & P(1|1) & 0 & 0 & 0 \\ 0 & P(2|1) & P(2|2) & 0 & 0 \\ 0 & P(3|1) & P(3|2) & P(3|3) & 0 \\ 0 & 0 & 0 & P(4|3) & 0 \end{matrix}$$

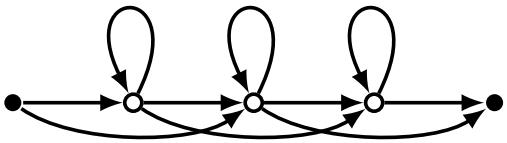


$$\begin{matrix} P(1|0) & P(1|1) & 0 & 0 & 0 \\ 0 & P(2|1) & P(2|2) & 0 & 0 \\ 0 & 0 & P(3|2) & P(3|3) & 0 \\ 0 & 0 & 0 & P(4|3) & 0 \end{matrix}$$

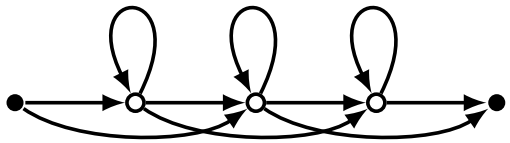
- the more data available the more sophisticated (and powerful) models can be trained

Hidden Markov Models

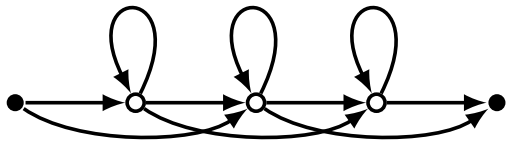
- composition of submodels on multiple levels
  - phone models have to be concatenated into word models
  - word models are concatenated into utterance models



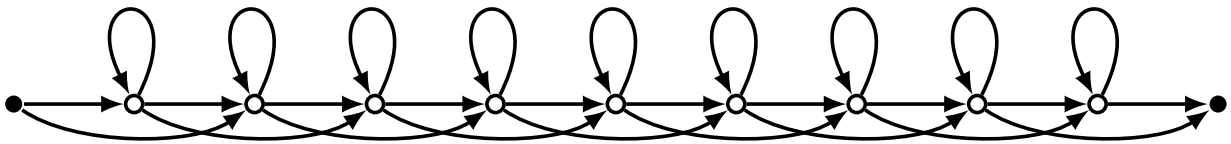
[f]



[a]



[n]



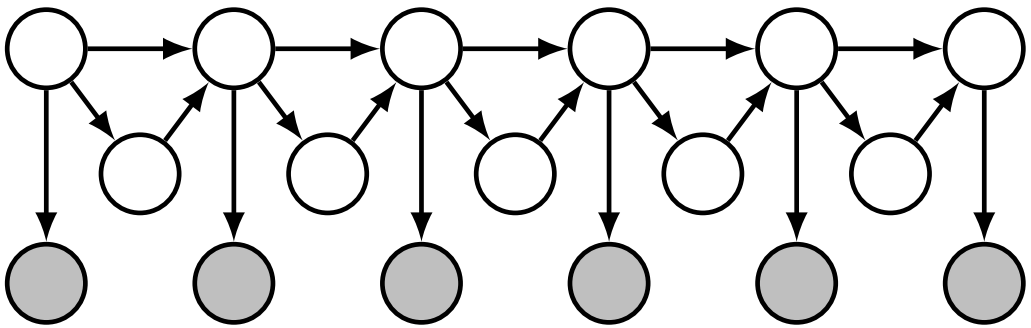
[f a n]

Dynamic Bayesian Networks

- using complex state descriptions, encoded by means of features
  - model can be in "different states" at the same time
- more efficient implementation of state transitions
- modelling of transitions between sub-models
- factoring out different influences on the outcome
  - interplay of several actuators (muscles, motors, ...)
- modelling partly asynchronized processes
  - coordinated movement of different body parts (e.g. sign language)
  - synchronization between speech sounds and lip movements
  - synchronization between speech and gesture
  - ...

Dynamic Bayesian Networks

- problem: state-transition probability tables are sparse
  - contain a large number of zero probabilities
- alternative model structure: separation of state and transition variables



- causal links can be stochastic *or* deterministic
  - stochastic: conditional probabilities to be estimated
  - deterministic: to be specified manually (decision trees)

Dynamic Bayesian Networks

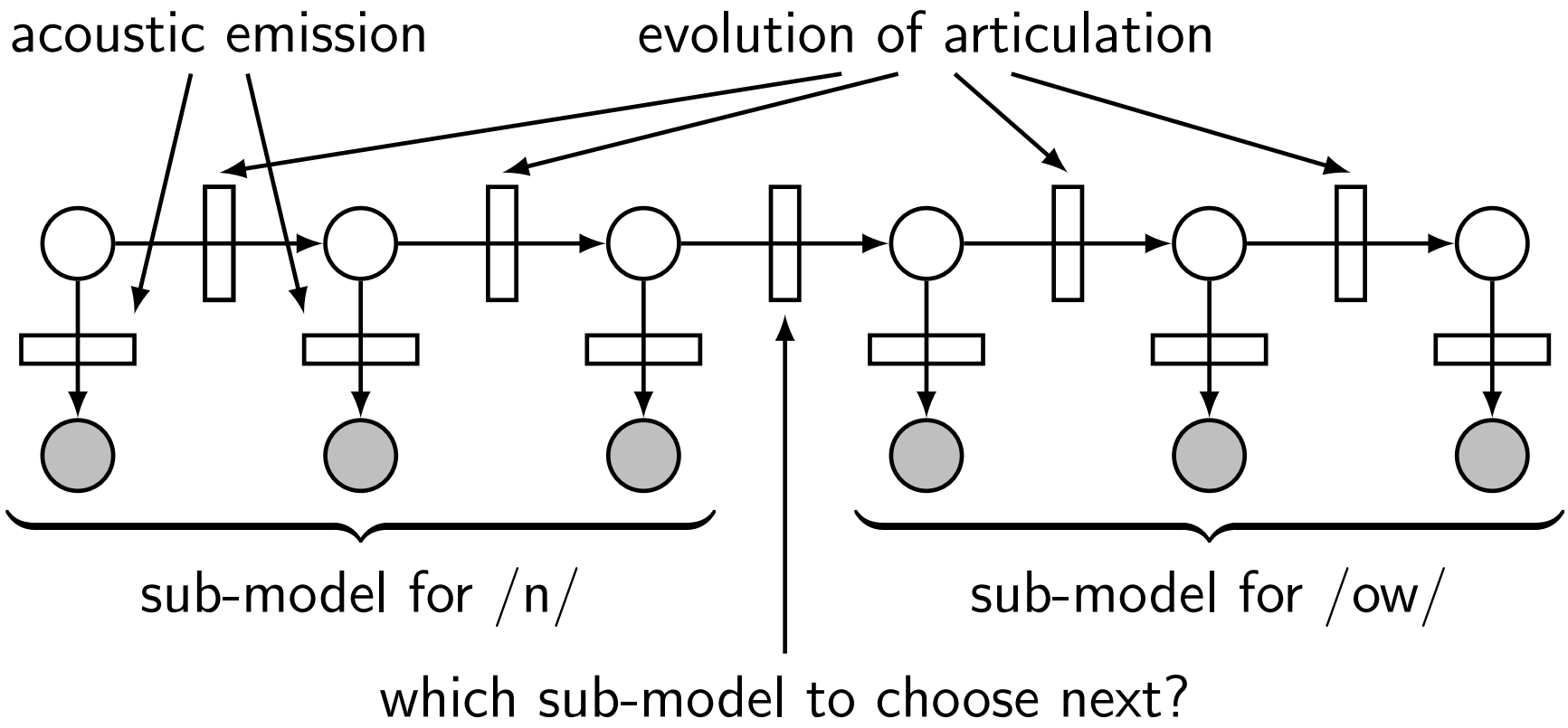
- state variables
  - distinct values for each state of the corresponding HMM
  - value at slice $t + 1$ is a deterministic function of the state and the transition of slice t
- transition variables
  - probability distribution
  - which arc to take to leave a state of the corresponding HMM
  - number of values is the outdegree of the corresponding state in an HMM
- use of transition variables is more efficient than using stochastic state variables with zero probabilities for the impossible state transitions

Dynamic Bayesian Networks

- composite models: some applications require the model to be composed out of sub-models
  - speech: phones $\rightarrow$ syllables $\rightarrow$ words $\rightarrow$ utterances
  - vision: sub-parts $\rightarrow$ parts $\rightarrow$ composites
  - genomics: nucleotides $\rightarrow$ amino acids $\rightarrow$ proteins

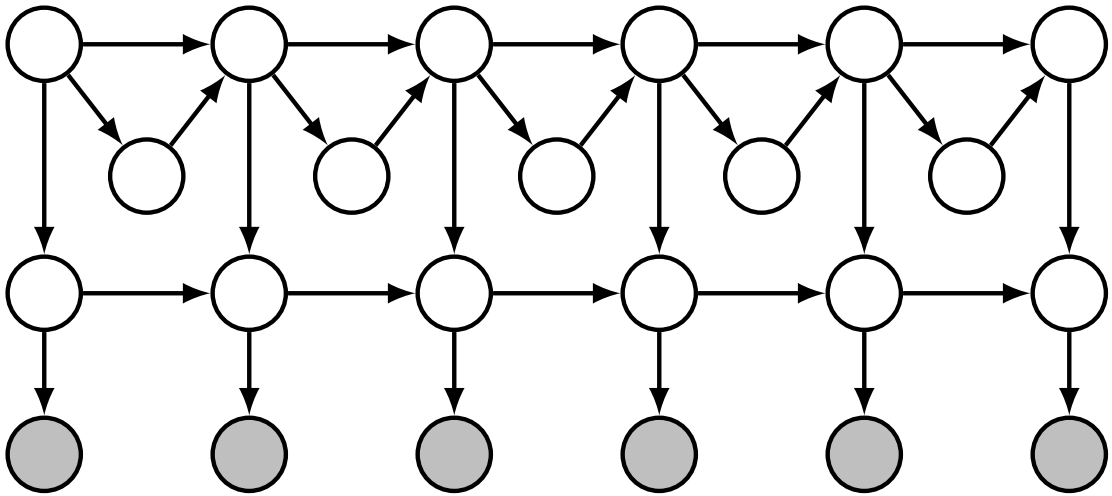
Dynamic Bayesian Networks

- composite models:
  - length of the sub-segments is not kown in advance
  - naive concatenation would require to generate all possible segmentations of the input sequence



Dynamic Bayesian Networks

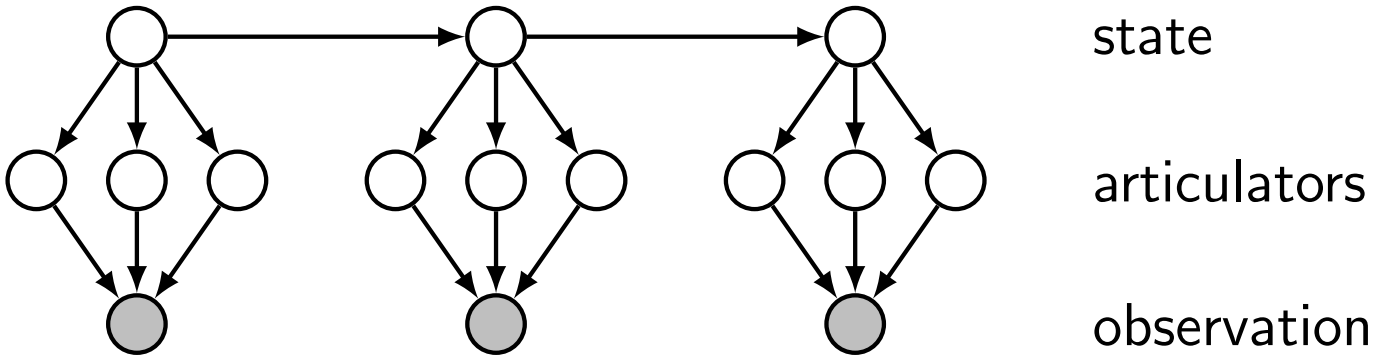
- additional sub-model variables select the next sub-model to choose



- sub-model index variables: which submodel to use at each point in time
- sub-model index and transition variables model legal sequences of sub-models (control layer)
- several control layers can be combined

Dynamic Bayesian Networks

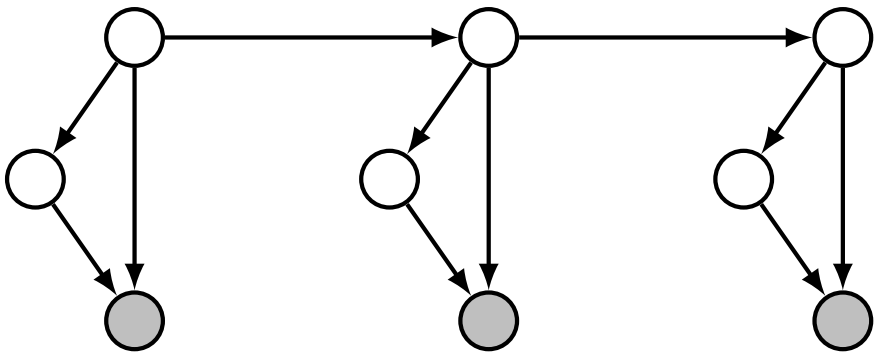
- factored models (1): factoring out different influences on the observation
- e.g. articulation:
  - asynchroneous movement of articulators (lips, tongue, jaw, ...)



- if the data is drawn from a factored source, full DBNs are superior to the special case of HMMs

Dynamic Bayesian Networks

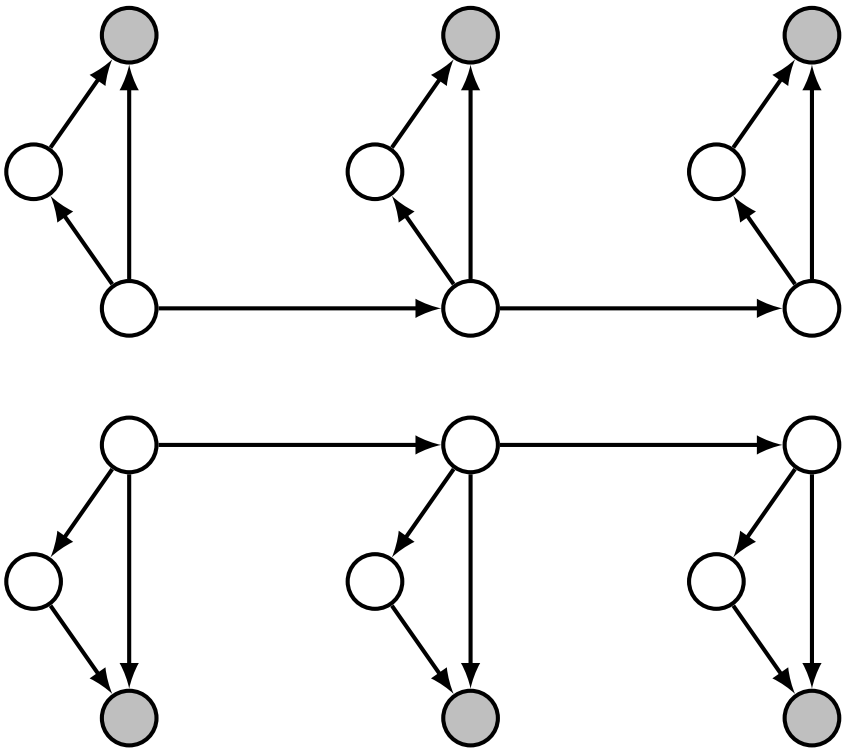
- factored models (2): coupling of different input channels
  - e.g. acoustic and visual information in speech processing
- naïve approach (1): data level fusion



- too strong synchronisation constraints

Dynamic Bayesian Networks

- naïve approach(2): independent input streams



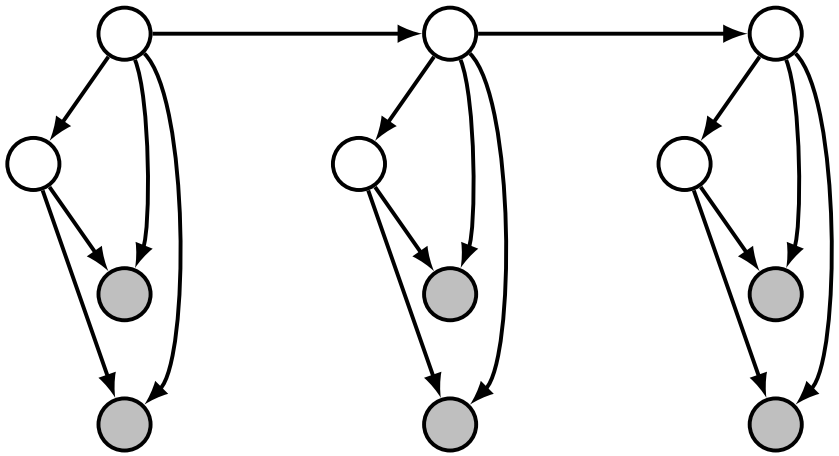
acoustic channel

visual channel

- no synchronisation at all

Dynamic Bayesian Networks

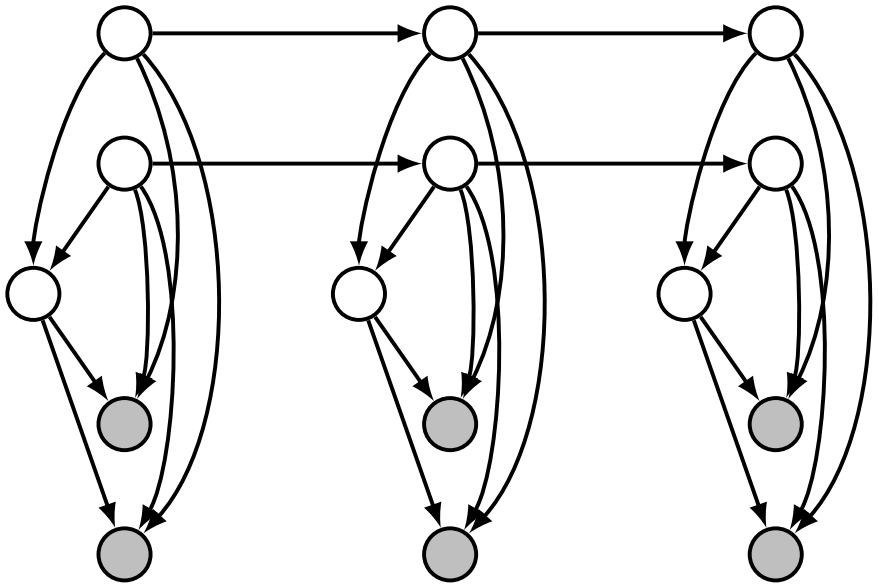
- product model



- state values are taken from the cross product of acoustic and visual states
- large probability distributions have to be trained

Dynamic Bayesian Networks

- factorial model (Nefian et al. 2002)



- independent (hidden) states
- indirect influence by means of the "explaining away" effect
- loose coupling of input channels

Dynamic Bayesian Networks

- inference is extremely expensive
  - nodes are connected across slices
  - domains are not locally restricted
  - cliques become intractably large
- but: joint distribution usually need not be computed
  - only maximum detection required
  - finding the optimal path through a lattice
  - dynamic programming can be applied (Viterbi algorithm)

Learning of Bayesian Networks

- estimating the probabilities for a given structure
  - for complete data:
    - maximum likelihood estimation
    - Bayesian estimation
  - for incomplete data
    - expectation maximization
    - gradient descent methods
- learning the network structure

Parameter estimation

- complete data
  - maximum likelihood estimation
  - Bayesian estimation
- incomplete data
  - expectation maximization
  - (gradient descent techniques)

Maximum Likelihood Estimation

- likelihood of the model M given the (training) data $\mathcal{D}$

$$L(M|\mathcal{D}) = \prod_{d \in \mathcal{D}} P(d|M)$$

- log-likelihood

$$LL(M|\mathcal{D}) = \sum_{d \in \mathcal{D}} log_2 P(d|M)$$

- choose among several possible models for describing the data according to the principle of maximum likelihood

$$\hat{\Theta} = \arg\max_{\Theta} L(M_\Theta|\mathcal{D}) = \arg\max_{\Theta} LL(M_\Theta|\mathcal{D})$$

- the models only differ in the set of parameters Θ

Maximum Likelihood Estimation

- complete data: estimating the parameters by counting

$$P(A = a) = \frac{N(A = a)}{\sum_{v \in dom(A)} N(A = v)}$$

$$P(A = a | B = b, C = c) = \frac{N(A = a, B = b, C = c)}{N(B = b, C = c)}$$

Rare events

- sparse data results in pessimistic estimations for unseen events
  - if the count for an event in the data base is 0, the event is considered impossible by the model
  - in many applications most events will never be observed, irrespective of the sample size

Rare events

- Bayesian estimation: using an estimate of the prior probability as starting point for the counting
  - estimation of maximum a posteriori parameters
  - no zero counts can occur
  - if nothing else available use an even distribution as prior
  - Bayesian estimate in the binary case with an even distribution

$$P(yes) = \frac{n+1}{n+m+2}$$

    n: counts for yes, m: counts for no
  - effectively adding virtual counts to the estimate

Rare events

- alternative: smoothing as a post processing step
- remove probability mass from the frequent observations ...
- ... and distribute it to the not observed ones
  - floor method
  - discounting
  - ...

Rare events

Backoff

- interpolate with the estimates of a less sophisticated model, e.g. compine trigram probabilities with bigram or unigram probabilities

$$\hat{P}(o_n|o_{n-2}, o_{n-1}) =$$
$$c_3\, P(o_n|o_{n-2}, o_{n-1}) + c_2\, P(o_n|o_{n-1}) + c_1\, P(o_n)$$

- good/acceptable coefficients c_i can be estimated on held out data

Incomplete Data

- missing at random:
  - probability that a value is missing depends only on the observed value
  - e.g. confirmation measurement: values are available only if the preceding measurement was positive/negative
- missing completely at random
  - probability that a value is missing is also independent of the value
  - e.g. stochastic failures of the measurement equipment
  - e.g. hidden/latent variables (mixture coefficients of a Gaussian mixture distribution)
- nonignorable:
  - neither MAR or MCAR
  - probability depends on the unseen values, e.g. exit polls for extremist parties

Expectation Maximization

- estimating the underlying distribution of not directly observable variables
- expectation:
  - "complete" the data set using the current estimation $h = \Theta$ to calculate expectations for the missing values
  - applies the model to be learned (Bayesian inference)
- maximization:
  - use the "completed" data set to find a new maximum likelihood estimation $h' = \Theta'$

Expectation Maximization

- full data consists of tuples $\langle x_{i1}, \ldots, x_{ik}, z_{i1}, \ldots, z_{il} \rangle$
  only x_i can be observed
- training data: $X = \{\vec{x}_1, \ldots, \vec{x}_m\}$
- hidden information: $Z = \{\vec{z}_1, \ldots, \vec{z}_m\}$
- parameters of the distribution to be estimated: Θ
- Z can be treated as random variable with $p(Z) = f(\Theta, X)$
- full data: $Y = \{\vec{y} \mid \vec{y} = \vec{x_i} || \vec{z_i}\}$
- hypothesis: h of Θ, needs to be revised into h'

Expectation Maximization

- goal of EM: $h' = \arg\max E(\log_2 p(Y|h'))$
- define a function $Q(h'|h) = E(\log_2 p(Y|h')|h, X)$
- Estimation (E) step:
  Calculate $Q(h'|h)$ using the current hypothesis h and the observed data X to estimate the probability distribution over Y

$$Q(h'|h) \leftarrow E(\log_2 p(Y|h')|h, X)$$

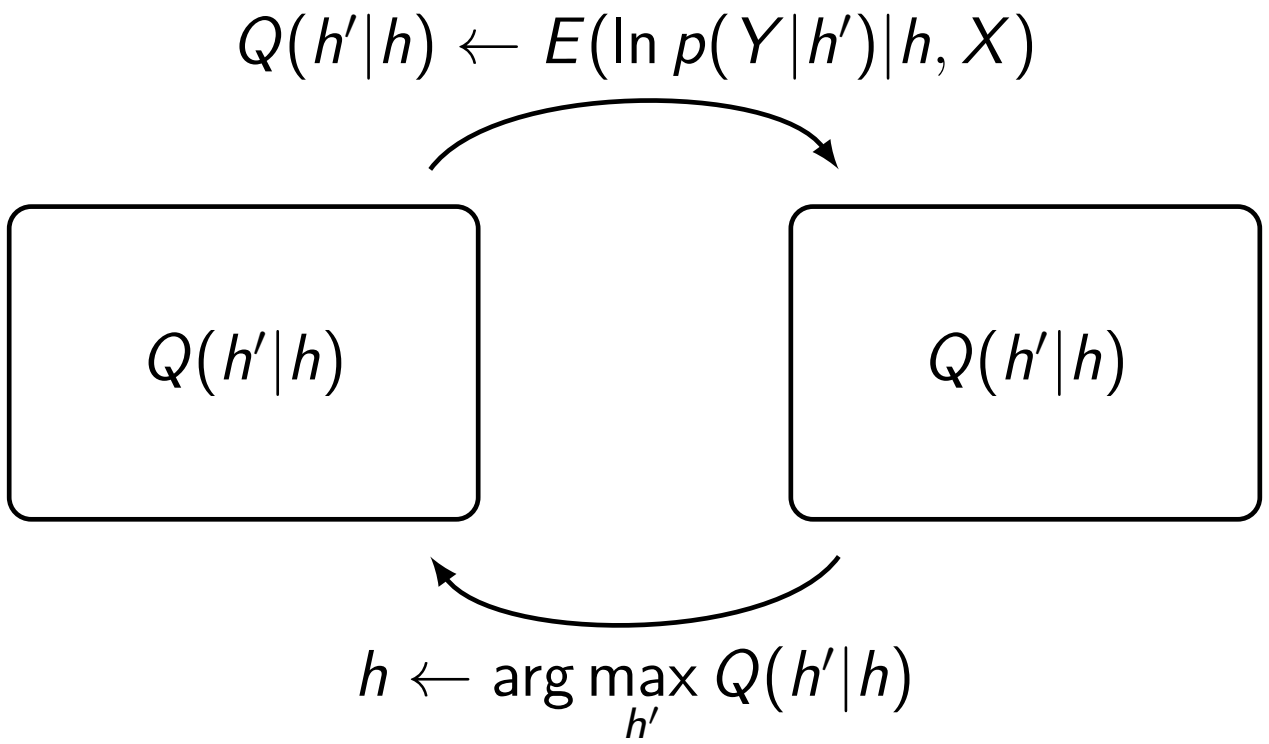
- Maximization (M) step
  Replace hypothesis h by h' that maximizes the function Q

$$h \leftarrow \arg\max_{h'} Q(h'|h)$$

Expectation Maximization

- expectation step requires applying the model to be learned
  - Bayesian inference
- gradient ascent search
  - converges to the next local optimum
  - global optimum is not guaranteed

Expectation Maximization



- If Q is continuous, EM converges to the local maximum of the likelihood function $P(Y|h')$

Learning the Network Structure

- learning the network structure
- space of possible networks is extremely large ($> \mathcal{O}(2^n)$)
- a Bayesian network over a complete graph is always a possible answer, but not an interesting one (no modelling of independencies)
- problem of overfitting
- two approaches
  - constraint-based learning
  - (score-based learning)

Constraint-based Structure Learning

- estimate the pairwise degree of independence using conditional mutual information
- determine the direction of the arcs between non-independent nodes

Estimating Independence

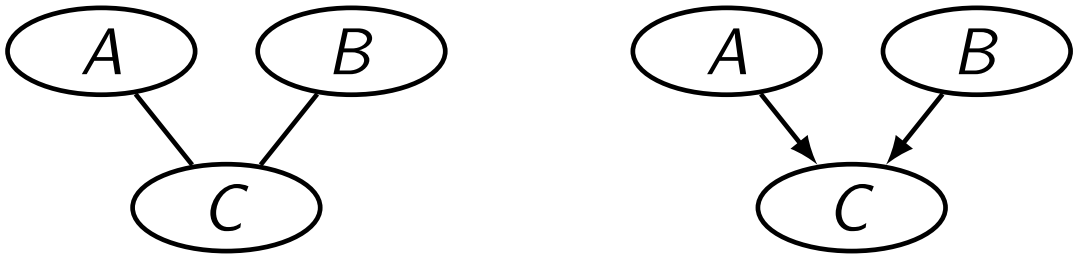
- conditional mutual information

$$CMI(A, B|\mathcal{X}) = \sum_{\mathcal{X}} \widehat{P}(\mathcal{X}) \sum_{A,B} \widehat{P}(A, B|\mathcal{X}) log_2 \frac{\widehat{P}(A, B|\mathcal{X})}{\widehat{P}(A|\mathcal{X})\widehat{P}(B|\mathcal{X})}$$

- two nodes are independent if $CMI(A, B|\mathcal{X}) = 0$
- choose all pairs of nodes as non-independent, where the significance of a χ^2-test on the hypothesis $CMI(A, B|\mathcal{X}) = 0$ is above a certain user-defined threshold
- high minimal significance level: more links are established
- result is a skeleton of possible candidates for causal influence

Determining Causal Influence

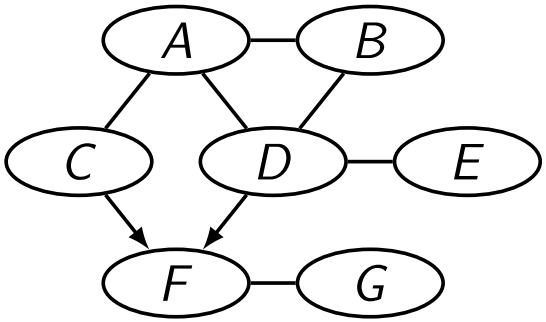
- Rule 1 (introduction of v-structures): $A - C$ and $B - C$ but not $A - B$ introduce a v-structure $A \rightarrow C \leftarrow B$ if there exists a set of nodes $\mathcal{X}$ so that A is d-separated from B given $\mathcal{X}$



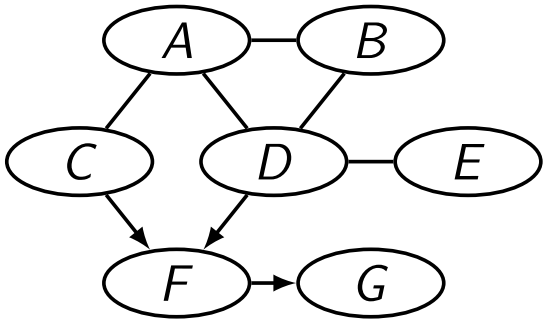
Determining Causal Influence

- Rule 2 (avoid new v-structures): When Rule 1 has been exhausted and there is a structure $A \rightarrow C - B$ but not $A - B$ then direct $C \rightarrow B$
- Rule 3 (avoid cycles): If $A \rightarrow B$ introduces a cycle in the graph do $A \leftarrow B$
- Rule 4 (choose randomly): If no other rule can be applied to the graph, choose an undirected link and give it an arbitrary direction

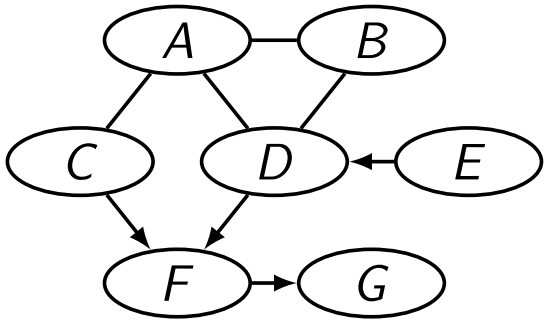
Determining Causal Influence



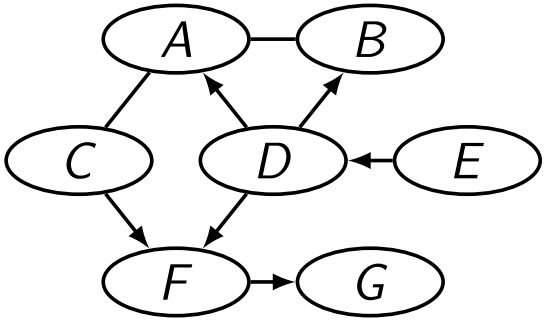
Rule 1



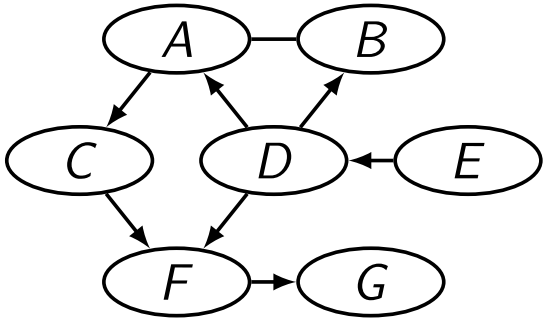
Rule 2



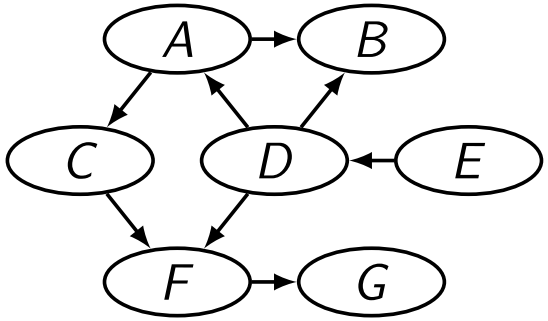
Rule 4



Rule 2



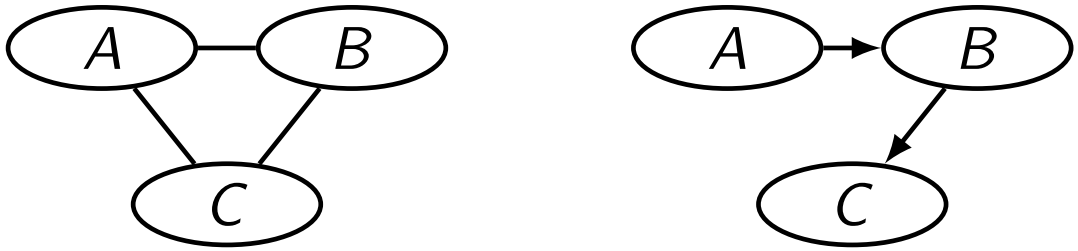
Rule 2



Rule 4

Determining Causal Influence

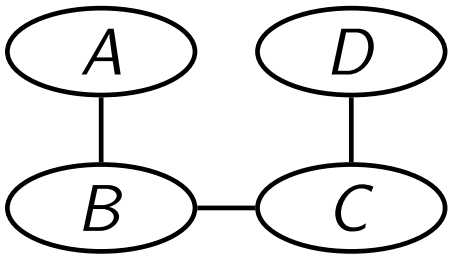
- independence/non-independence candidates might contradict each other
- $\neg I(A, B), \neg I(A, C), \neg I(B, C)$, but $I(A, B|C), I(A, C|B)$ and $I(B, C|A)$
  - remove a link and build a chain out of the remaining ones

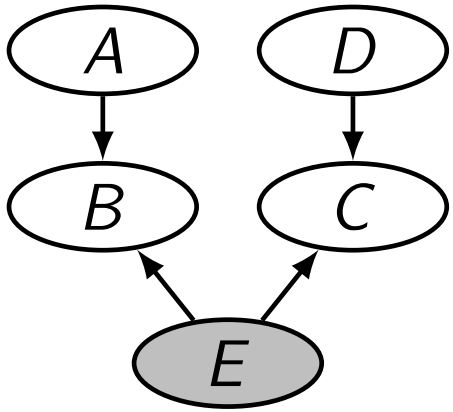


  - uncertain region: different heuristics might lead to different structures

Determining Causal Influence

- $I(A, C), I(A, D), I(B, D)$





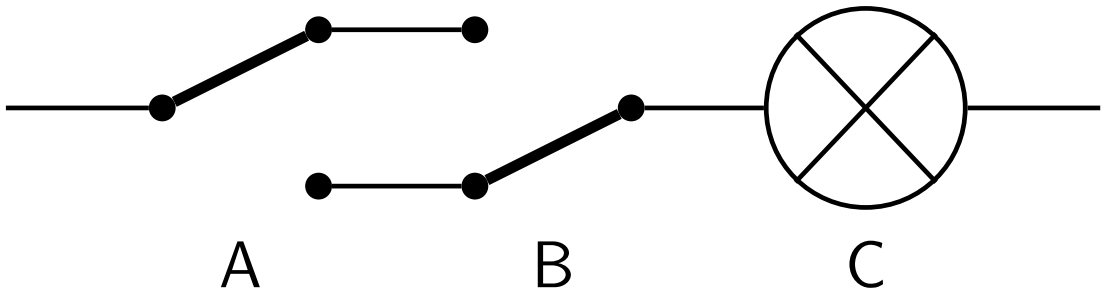
  - problem might be caused by a hidden variable $E \rightarrow B$ $E \rightarrow C$ $A \rightarrow B$ $D \rightarrow C$

Constraint-based Structure Learning

- useful results can only be expected, if
  - the data is complete
  - no (unrecognized) hidden variables obscure the induced influence links
  - the observations are a faithful sample of an underlying Bayesian network
    - the distribution of cases in $\mathcal{D}$ reflects the distribution determined by the underlying network
    - the estimated probability distribution is very close to the underlying one
  - the underlying distribution is recoverable from the observations

Constraint-based Structure Learning

- example of an unrecoverable distribution:
  - two switches: $P(A = up) = P(B = up) = 0.5$
  - $P(C = on) = 1$ if $val(A) = val(B)$
  - $\rightarrow I(A, C), I(B, C)$



- problem: independence decisions are taken on individual links (CMI), not on complete link configurations

$$P(C|A, B) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$