

Bayesian Networks

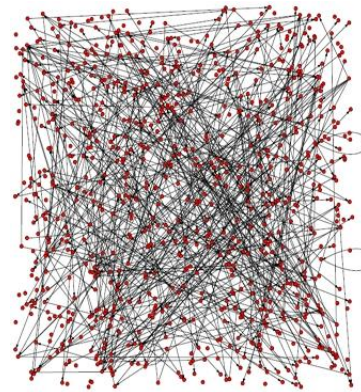
A belief about many things is a graph

Jinkyoo Park · KAIST

Structure tames the joint — and once a decision hangs off the graph, every later lecture is already in the room

Where we are — one unknown becomes many

Bayesian Network is System Modeling



Probability + **Statistics** + **Graph Theory**

Probability + statistics + graph theory.

Lecture 2 updated one unknown. A Bayesian network connects many interacting variables: **represent a system, infer its hidden parts, then decide.**

Learning route — from a joint probability to a decision

Bring: Bayes' rule and marginalization from Lecture 2.

FIRST PASS	WHAT TO DO
Follow the idea	Represent → infer → learn local tables → track a hidden state → choose by utility
Work without the solution	Compute a posterior, one filtering update and an expected-utility choice.
Return later	Structure search, hybrid/time-series extensions and full elimination derivations are references.

For the temperature thread: **predict** → **calculate** → **reveal and check** → **change one condition**. Complete the core calculation before reading the research extensions.

— ACT 1

The joint as a graph

Q1. Every distribution factorises somehow. A Bayesian network is a factorisation you can draw — and drawing it is what makes it cheap.

A joint, written out in full

Q1 Why a graph?

Q2 What does it encode?

Q3 How do I infer?

Q4 How do I decide?

A	B	C	$P(A, B, C)$
0	0	0	0.08
0	0	1	0.15
0	1	0	0.05
0	1	1	0.10
1	0	0	0.14
1	0	1	0.18
1	1	0	0.19
1	1	1	0.11

THREE BINARY VARIABLES, AND WHAT THE TABLE COSTS

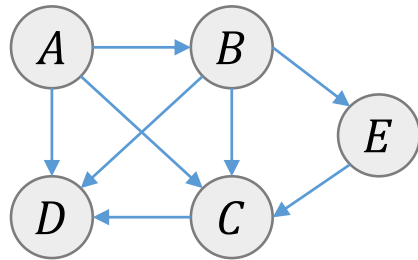
$2^3 = 8$ rows — every combination A , B and C can take. The last is forced by the others, so $2^3 - 1 = 7$ free numbers.

The table is *complete*: every question about A , B and C is a sum over some of its rows, and nothing is assumed.

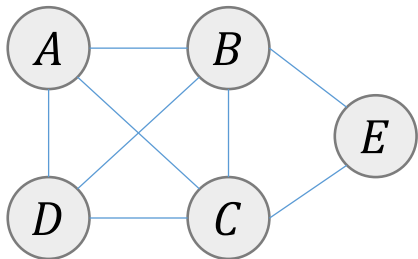
That completeness is exactly the problem. It is **useless at scale**, because a flat table has no way to say that A is irrelevant to C once B is known.

A graph can say it. That is the whole content of this act.

First read the arrows — directed and undirected graphs



A **directed graph** G consists of directed edges between nodes

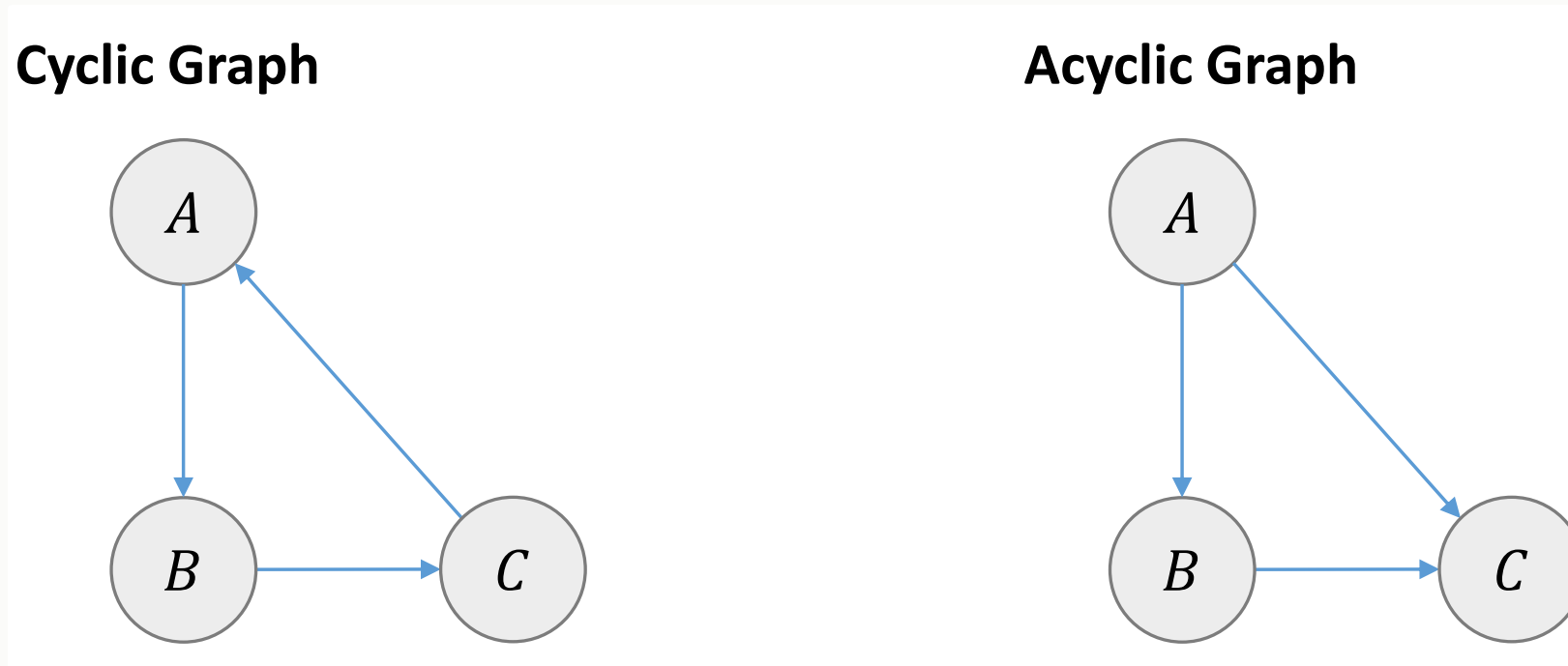


An **undirected graph** G consists of undirected edges between nodes

Original node positions and arrow directions, PDF p. 6.

An edge connects two nodes. An arrow also gives a direction. A Bayesian network uses directed edges.

A directed acyclic graph, and the words for reading one

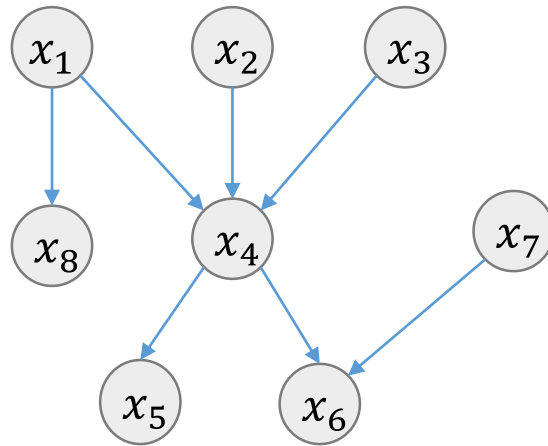


Follow the arrows and try to return to the starting node.

A **directed acyclic graph (DAG)** has no directed cycle. On the left, $A \rightarrow B \rightarrow C \rightarrow A$ returns to A ; on the right it cannot.

A DAG gives a **topological order**: generate each variable after its parents. Arbitrary tables on a directed cycle do not provide this guarantee.

Read one graph three ways — nodes, edges and a matrix



Representations

- Edge list

$$E = \{(x_1, x_4), (x_1, x_8), (x_2, x_4), (x_3, x_4), (x_4, x_5), (x_4, x_6), (x_7, x_6)\}$$

- Adjacency matrix

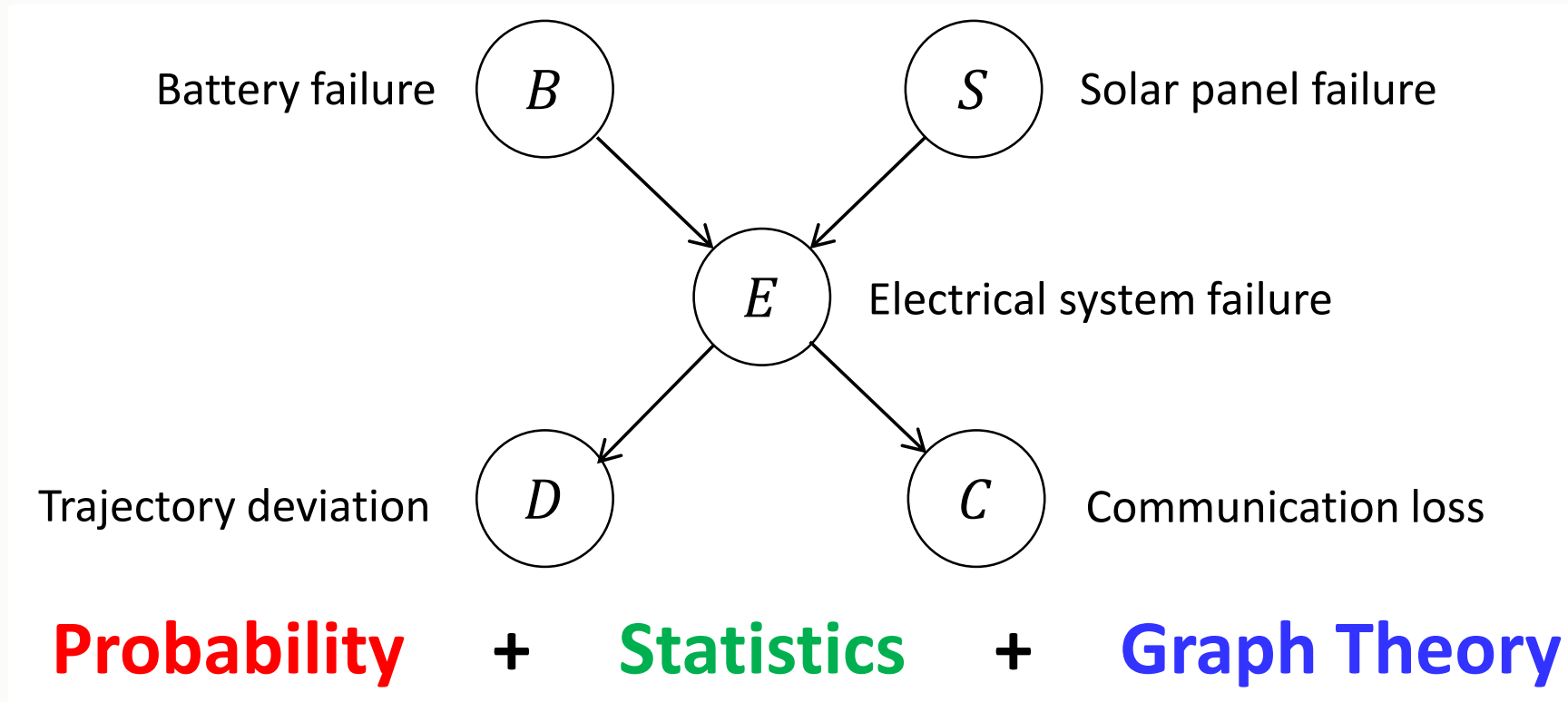
$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

- ✓ A path $x_1 \rightarrow x_6$ is $x_1 \rightarrow x_4 \rightarrow x_6$
- ✓ The ancestors of x_6 are $ac(x_6) = \{x_1, x_2, x_3, x_4, x_7\}$
- ✓ The descendants of x_2 are $dc(x_2) = \{x_4, x_5, x_6\}$
- ✓ The parents of x_4 are $pa(x_4) = \{x_1, x_2, x_3\}$
- ✓ The children of x_4 are $ch(x_4) = \{x_5, x_6\}$

The same eight-node graph, edge list and adjacency matrix.

Read arrows entering x_4 : its parents are x_1, x_2, x_3 . Follow arrows leaving it: its children are x_5, x_6 .

A satellite — two causes and two visible symptoms

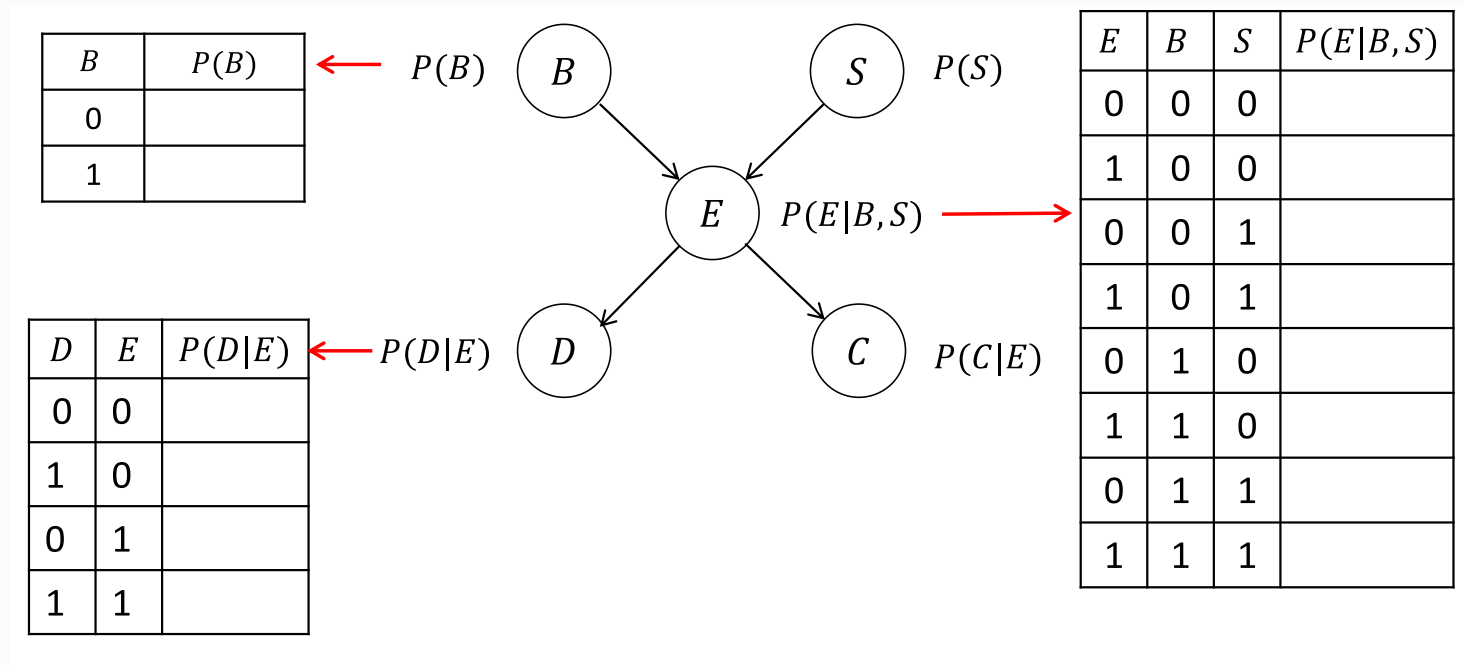


Retain the diamond-shaped system diagram.

Battery B and solar panel S feed electrical state E ; E feeds trajectory D and communication C . Read each node as a random variable.

The factorisation — one joint, written locally

A **Bayesian network** specifies $p(x_1, \dots, x_n) = \prod_i p(x_i \mid \text{pa}_i)$ on a DAG.



Each table sits beside the node whose local distribution it specifies.

$$p(B, S, E, D, C) = p(B)p(S)p(E \mid B, S)p(D \mid E)p(C \mid E).$$

Free parameters: $1 + 1 + 4 + 2 + 2 = 10$, compared with $2^5 - 1 = 31$ in the full joint.

Multiply first, then sum — a two-node calculation

an illustrative sensor model; each row has a clear role

Let F mean a machine fault and A mean an alarm: $F \rightarrow A$. Use $P(F) = 0.1$, $P(A \mid F) = 0.8$, $P(A \mid \neg F) = 0.2$.

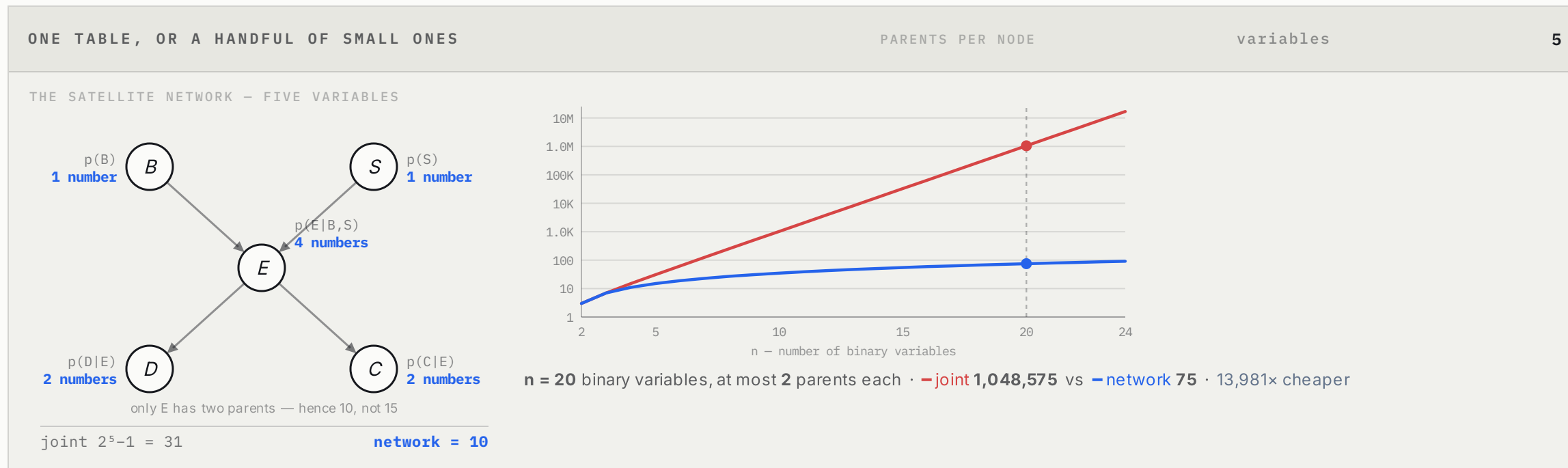
CASE	LOCAL PROBABILITIES	JOINT PROBABILITY
Fault and alarm	0.1×0.8	0.08
No fault and alarm	0.9×0.2	0.18

The alarm probability sums both possible causes: $P(A) = 0.08 + 0.18 = 0.26$.

$$P(F \mid A) = \frac{P(F, A)}{P(A)} = \frac{0.08}{0.26} \approx 0.308.$$

Multiply along a complete case, add mutually exclusive cases, normalise after observing evidence. An alarm increases the fault probability from 10% to about 31%; it does not make a fault certain.

The collapse, in numbers



The joint's cost is $2^n - 1$ whatever you do. With a fixed upper bound on the number of parents, each binary node needs a bounded-size table. **The network then grows linearly in the number of nodes** while the full joint grows exponentially. Slide n and watch the gap open; the satellite sits at $n = 5$, where 31 becomes 10.

The wall — a joint distribution is exponentially large

For n binary variables the joint $p(x_1, \dots, x_n)$ is a table of 2^n rows, so specifying it takes

$$2^n - 1 \text{ independent numbers} \quad (M^n - 1 \text{ if each variable has } M \text{ values})$$

Five binary variables need 31 numbers. Twenty need 1 048 575. Thirty need over a billion. **Too many to store, far too many to estimate from data, and hopeless to sum over.**

The escape is structure: **a belief about many things is a graph.**

Every distribution is a Bayesian network — and that is the catch

The chain rule already writes any joint as a product of conditionals:

$$p(x_1, \dots, x_n) = p(x_n \mid x_1, \dots, x_{n-1}) p(x_{n-1} \mid x_1, \dots, x_{n-2}) \cdots p(x_1)$$

which is a Bayesian network — on the *fully connected* DAG. It costs exactly $2^n - 1$ numbers and buys nothing.

The saving is not in the factorisation. It is in **the edges you leave out**.

$p(C \mid E)$ rather than $p(C \mid E, B, S, D)$ is a conditional-independence claim: once the electrical state is known, the other listed variables add no information about communication loss. Observing descendants D, C can still change our belief about E , even when B, S are known. The DAG *is* the model, and every missing arrow is an assumption you are on the hook for. Act 2 says exactly which assumptions a given DAG makes.

Check — where the saving comes from

A Bayesian network factorises the joint as $\prod_i p(x_i \mid \text{pa}(x_i))$. Where does the reduction from 2^n numbers actually come from?

- ☒ A From the edges left *out*: each factor conditions only on parents, not on all predecessors
- ☐ B From the factorisation itself — writing the joint as a product of conditionals
- ☐ C From assuming every variable is binary
- ☐ D From the topological ordering chosen for the chain rule

The chain rule $p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \cdots$ is **always** true and saves nothing — the last factor still conditions on everything. What buys the reduction is each missing edge, every one of which is a claimed conditional independence. A Bayesian network is therefore a statement about what does *not* influence what, and its compactness is only as honest as those claims.

— ACT 2

What the graph encodes

Q2. The full graph specifies conditional independences. A missing direct arrow alone does not rule out dependence through another path.

Conditional independence — the meaning of a missing edge

Q1 Why a graph?

Q2 What does it encode?

Q3 How do I infer?

Q4 How do I decide?

INDEPENDENCE

$$X \perp Y \iff p(X, Y) = p(X)p(Y)$$

equivalently $p(X | Y) = p(X)$: learning Y tells you nothing about X .

CONDITIONAL INDEPENDENCE

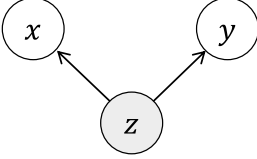
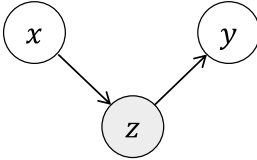
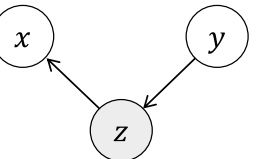
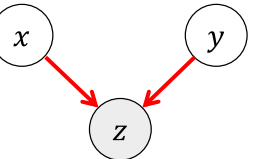
$$X \perp Y | Z \iff p(X, Y | Z) = p(X | Z)p(Y | Z)$$

equivalently $p(X | Y, Z) = p(X | Z)$: **once you know Z , learning Y tells you nothing further about X .**

The DAG's basic promise is the **local Markov property**: a variable is independent of its non-descendants given its parents. In the satellite network, $C \perp B | E$ — knowing the battery failed does not move your belief about communication loss **if the electrical system's state is already known**. Likewise $D \perp S | E$.

That single sentence is what the missing edges $B \rightarrow C$ and $S \rightarrow D$ mean. This is also why the binary conditional $p(C | E)$ has two free numbers; conditioning on four binary variables would require sixteen.

See why the collider is different

	$p(x, y z) = \frac{p(x, y, z)}{p(z)} = \frac{p(z)p(x z)p(y z)}{p(z)} = p(x z)p(y z)$
	$p(x, y z) = \frac{p(x, y, z)}{p(z)} = \frac{p(x)p(z x)p(y z)}{p(z)} = \frac{p(x, z)p(y z)}{p(z)} = p(x z)p(y z)$
	$p(x, y z) = \frac{p(x, y, z)}{p(z)} = \frac{p(y)p(z y)p(x z)}{p(z)} = \frac{p(y, z)p(x z)}{p(z)} = p(y z)p(x z)$
	$p(x, y z) = \frac{p(x, y, z)}{p(z)} = \frac{p(x)p(y)p(z x, y)}{p(z)}$ BN with $x \rightarrow z \leftarrow y$ ✓ x and y are unconditionally independent Dependence can appear given z.

Blue: conditioning separates; red: conditioning can couple.

The three blue rows factor after observing z . In the red row, $p(z \mid x, y)$ still connects the two causes.

Three structures, three verdicts

Chain any two variables through a third and there are only three shapes. Whether X and Y are independent depends on the shape *and* on whether the middle node is observed.

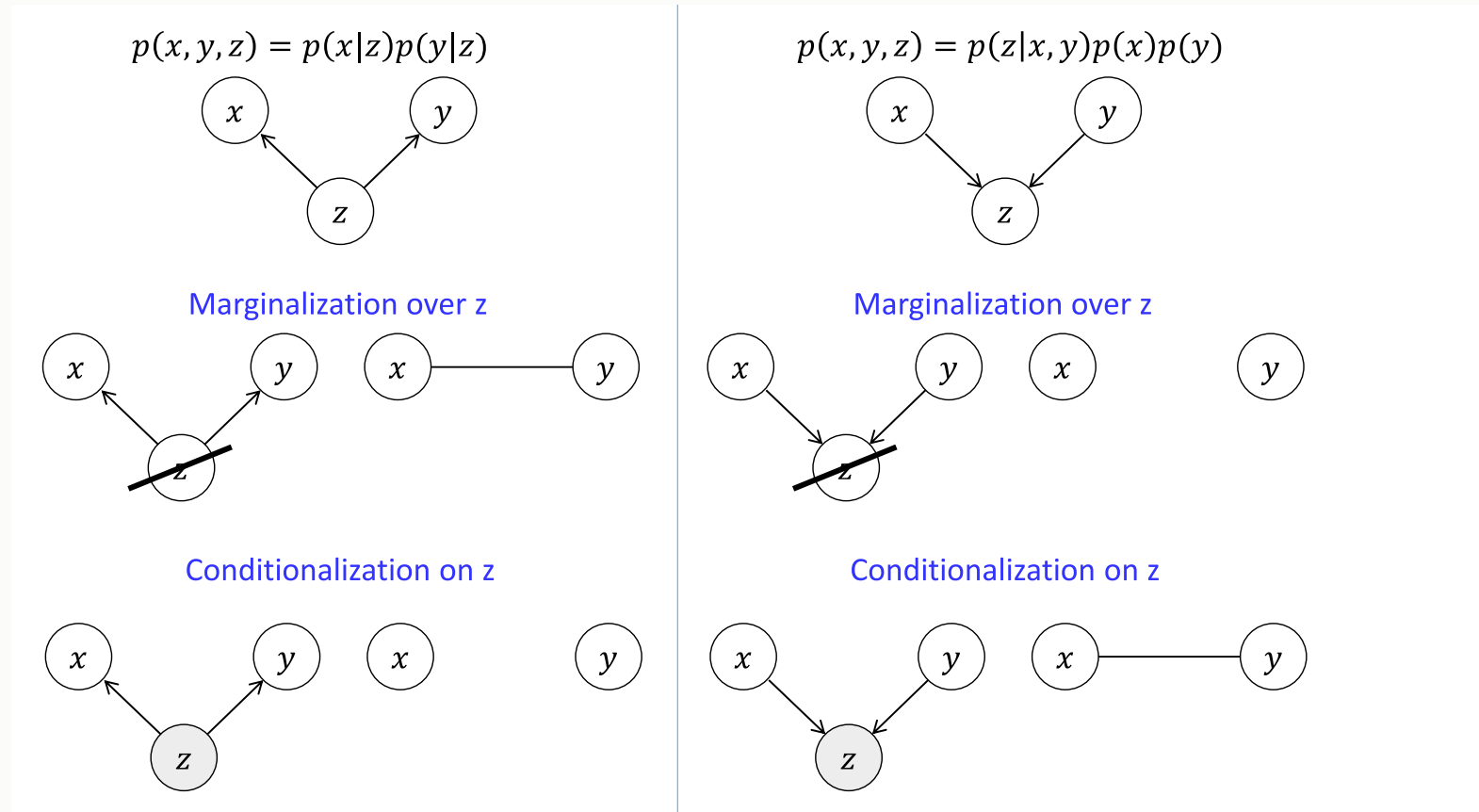
STRUCTURE	SHAPE	$X \perp Y$ WITH Z UNOBSERVED	$X \perp Y$ GIVEN Z
chain	$X \rightarrow Z \rightarrow Y$	not guaranteed	yes
fork (common cause)	$X \leftarrow Z \rightarrow Y$	not guaranteed	yes
collider (common effect)	$X \rightarrow Z \leftarrow Y$	yes	not guaranteed

For these isolated graphs, observing a chain or fork middle node **blocks** the path; observing a collider **opens** it. An open path permits dependence:

$$\begin{aligned}
 \text{fork: } & p(x, y \mid z) = p(x \mid z)p(y \mid z), \\
 \text{collider: } & p(x, y \mid z) = \frac{p(x)p(y)p(z \mid x, y)}{p(z)} \not\equiv p(x \mid z)p(y \mid z).
 \end{aligned}$$

This structural test guarantees independence for **every** compatible probability table. An open path gives no such guarantee; special tables can still make the variables independent.

Marginalize or condition — watch the connection change

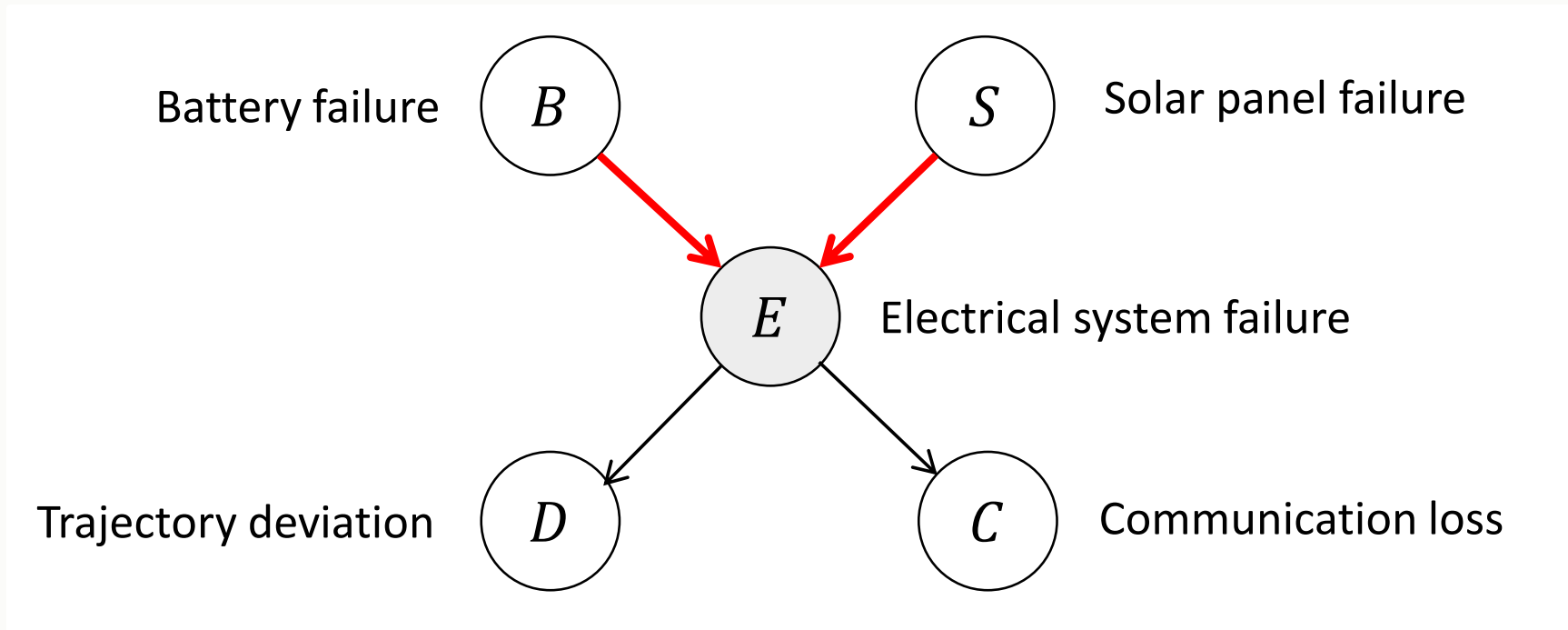


Left: common cause; right: common effect.

Crossed-out z : sum it out. Shaded z : observe it. A remaining line permits dependence; disconnected nodes are independent.

Explaining away — the collider, in words

nothing changed in the world; something changed in what you know

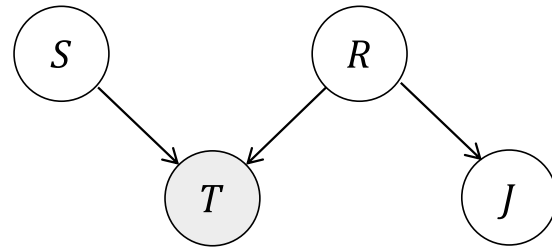


The electrical failure is observed; the competing causes are highlighted.

Before: B and S are independent. **Observe E :** the electrical system failed. **Then learn B failed:** in a competing-causes model, less evidence is needed for S to explain E .

The collider permits **explaining away**. Its probability table determines the sign and size of the dependence.

Wet grass — the graph and its probability tables



$R \in \{0,1\} : R = 1$ means that it has been raining
 $S \in \{0,1\} : S = 1$ Sprinkler is turned on
 $J \in \{0,1\} : J = 1$ Jack's grass is wet
 $T \in \{0,1\} : T = 1$ Tracey's grass is wet

$$p(T, J, R, S) = p(T|R, S)p(J|R)p(R)p(S)$$

$p(T | R, S)$

Tracey's Grass wet=1	Rain	Sprinkler
1	1	1
1	1	0
0.9	0	1
0	0	0

$p(J|R)$

Jack's Grass wet=1	Rain
1	1
0.2	0

$$p(S = 1) = 0.1$$

$$p(R = 1) = 0.2$$

Graph above, local probability tables below; the table label is corrected to $p(T | R, S)$.

First specify the model. Next ask whether wet grass supports the sprinkler, the rain, or both.

Wet grass — explaining away, with numbers

Example 3.1

THE MODEL

S sprinkler on · R raining · T Tracey's grass wet · J Jack's grass wet

$$S \rightarrow T \leftarrow R \rightarrow J$$

$$p(T, J, R, S) = p(T \mid R, S) p(J \mid R) p(R) p(S)$$

$p(S=1) = 0.1$, $p(R=1) = 0.2$; the grass is certainly wet if it rained, wet with probability 0.9 if only the sprinkler ran, and dry otherwise; $p(J=1 \mid R=1) = 1$, $p(J=1 \mid R=0) = 0.2$.

THE TWO QUERIES

$$p(S=1) = 0.100$$

$$p(S=1 \mid T=1) = \frac{0.092}{0.272} = 0.338$$

$$p(S=1 \mid T=1, J=1) = \frac{0.0344}{0.2144} = 0.160$$

Tracey's wet grass more than triples the suspicion of the sprinkler. Then Jack's wet grass — **evidence for the rival cause** — halves it again.

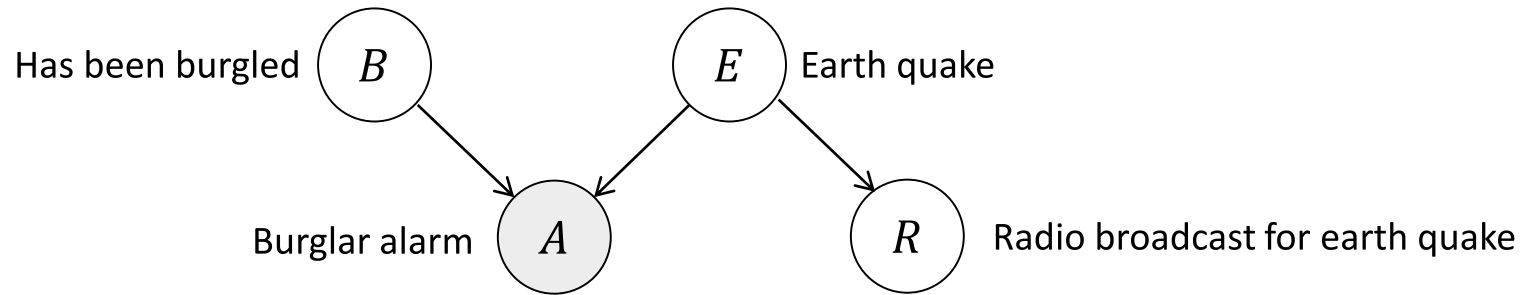
Note that J is not on the collider path at all; it is a child of R . It moves the answer only because it moves R , and R competes with S to explain T . Note also that $p(S=1 \mid J=1) = 0.100$ exactly — with T unobserved, Jack's lawn says nothing whatever about Tracey's sprinkler.

Evidence, entered by hand



Click a node to observe it. The two blue edges are the collider path $S \rightarrow T \leftarrow R$; they light when observing T opens it. The readout is the exact posterior over the sprinkler, computed by enumerating all sixteen states. Observe T alone and the collider opens; **add J and the rain explains the sprinkler away**; observe R directly and the sprinkler snaps back to its prior, because with the rain known, wet grass says nothing.

Burglar alarm — two competing explanations



$$p(B, E, A, R) = p(A|B, E)p(R|E)p(E)p(B)$$

$$p(A|B, E)$$

Alarm=1	Burglar	Earthquake
0.9999	1	1
0.99	1	0
0.99	0	1
0.0001	0	0

$$p(R|E)$$

Radio=1	Earthquake
1	1
0	0

$$p(B = 1) = 0.01$$

$$p(E = 1) = 0.000001$$

The first prior label is corrected to $P(B = 1) = 0.01$.

The alarm has two parents. A radio report is evidence about earthquake E ; after hearing the alarm, it can change our belief about burglary B .

The burglar alarm — the same move, sharper

Example 3.2

THE MODEL

B burgled · E earthquake · A alarm sounds · R radio reports a quake

$$B \rightarrow A \leftarrow E \rightarrow R$$

$p(B=1) = 0.01$, $p(E=1) = 10^{-6}$. The alarm is near-certain given either cause and almost never fires without one; the radio reports a quake exactly when there is one.

THE COLLAPSE

$$p(B=1) = 0.01$$

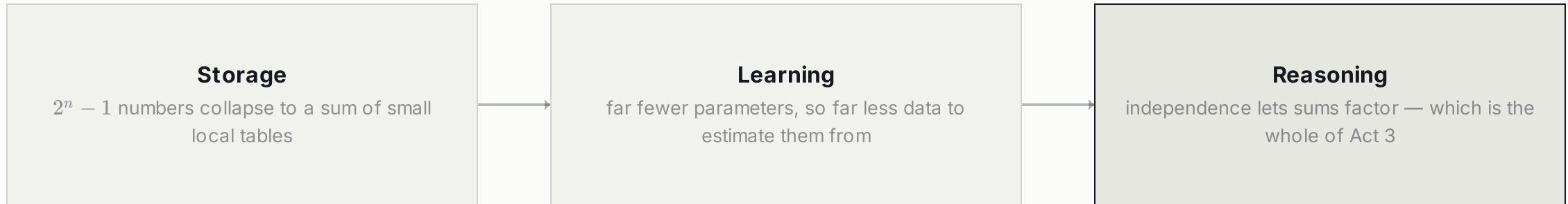
$$p(B=1 \mid A=1) = 0.99$$

$$p(B=1 \mid A=1, R=1) = 0.01$$

The alarm makes a burglary near-certain. One radio bulletin — which says nothing about burglars — sends it **straight back to the prior**.

R is not connected to B by any edge, and the two are marginally independent. They become strongly dependent the instant the alarm is heard — the same collider move as the wet grass, on a graph built to make it matter.

What the missing edges buy



All three are the same fact read three ways: **a missing edge is a conditional independence**, and the chain, fork and collider rules read every one of them off the graph.

Two cautions. An open path *permits* dependence; it does not force it, and special parameter values can cancel. And an arrow encodes a factorisation, not a causal intervention, unless a causal reading is explicitly assumed.

Check — the collider

Two independent causes point at one effect: $X \rightarrow Z \leftarrow Y$. What is the relationship between X and Y ?

- ☐ A Dependent, and conditioning on Z makes them independent
- ☒ B Independent before conditioning; observing Z can make them dependent
- ☐ C Independent, and they stay independent whatever you condition on
- ☐ D Dependent, and no conditioning changes that

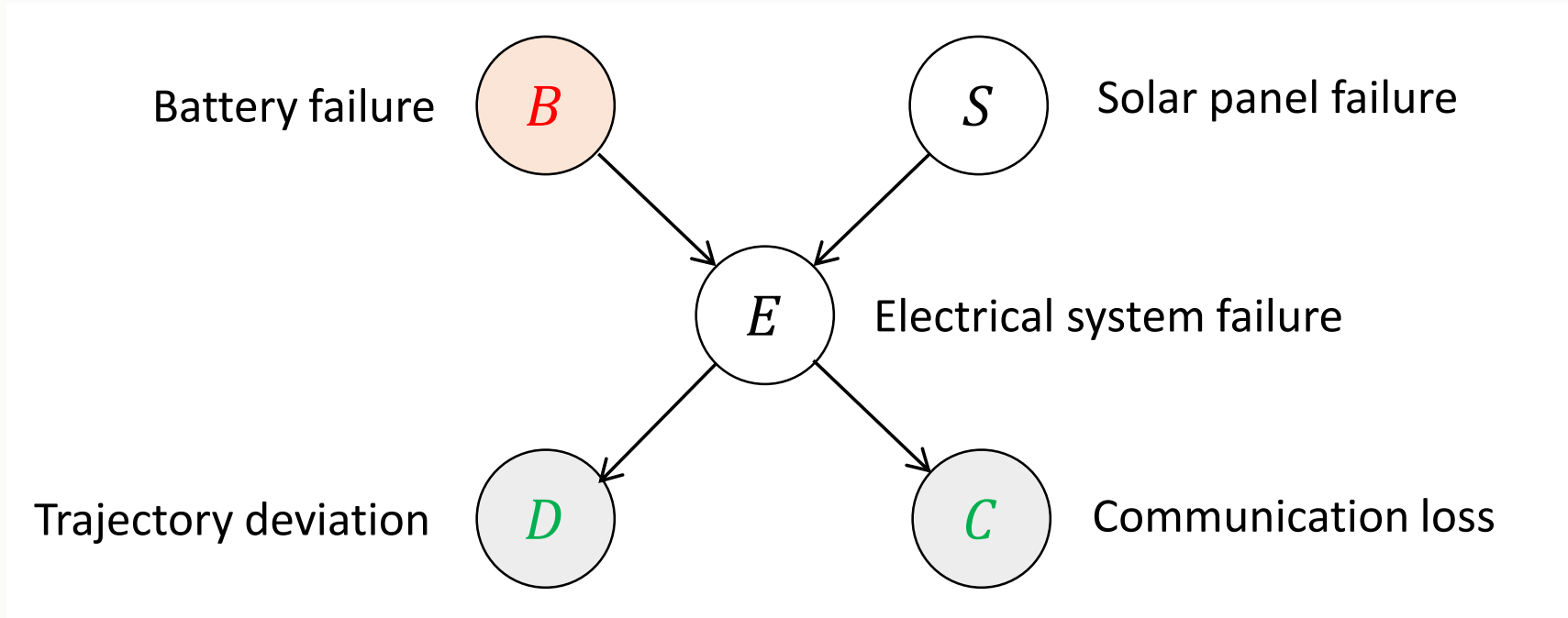
This is the collider, and it runs opposite to the chain and the fork. Learning the alarm went off makes burglary and earthquake **compete** to explain it, so hearing that there was an earthquake lowers your belief in a burglary — they became dependent the moment you conditioned on their shared effect. *Explaining away* is why a collider behaves opposite to a chain or a fork, and it is a standard way to introduce a correlation that is not there.

— ACT 3

Reasoning with the network

Q3. A model you cannot query is decoration. Inference is where the factorisation earns its keep — and where it still, sometimes, fails.

The query, and three kinds of variable



Orange: query; green: observed evidence; white: hidden variables.

$$P(B \mid D = 1, C = 1)$$

Keep *B* as the query. **Fix** *D*, *C* at their observed values. **Sum over** *E*, *S*, then normalise.

Exact inference — and why it hurts

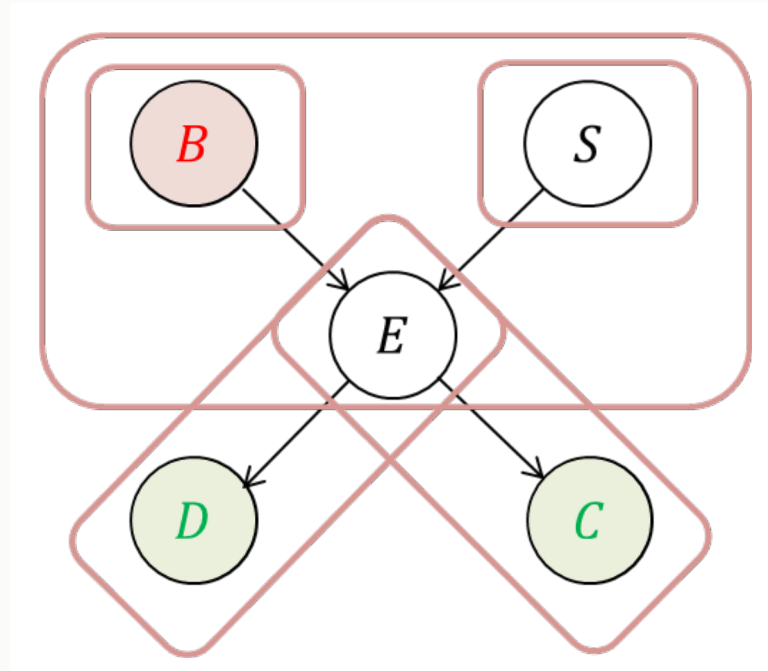
Marginalise the joint over the hidden variables and normalise:

$$P(b^1 \mid d^1, c^1) \propto \sum_s \sum_e P(b^1)P(s)P(e \mid b^1, s)P(d^1 \mid e)P(c^1 \mid e)$$

This is correct, and it is a disaster. The number of terms is the number of joint assignments to the hidden variables — **exponential in how many there are**. The curse the factorisation was supposed to have removed comes back the moment we sum.

Storing the joint cheaply is not the same as **summing over it cheaply**.

A factor covers the variables in one local table



Each outline groups the variables of one factor.

The large outline covers E, B, S : it is $p(E \mid B, S)$. Eliminating E combines every factor that mentions E .

Variable elimination — push each sum past what it cannot touch

A sum can slide past any factor that does not mention its variable. Do that twice:

$$\begin{aligned}
 P(b^1 \mid d^1, c^1) &\propto \sum_s \sum_e P(b^1) P(s) P(e \mid b^1, s) P(d^1 \mid e) P(c^1 \mid e) \\
 &= P(b^1) \sum_s \sum_e P(s) P(e \mid b^1, s) P(d^1 \mid e) P(c^1 \mid e) \quad P(b^1) \text{ has no } s \text{ and no } e \\
 &= P(b^1) \sum_s P(s) \sum_e P(e \mid b^1, s) P(d^1 \mid e) P(c^1 \mid e) \quad P(s) \text{ has no } e
 \end{aligned}$$

The value has not changed. What has changed is **how often each product is formed**: the inner sum is a function of s alone once b^1 is fixed, so it is computed once per value of s — not once per pair (s, e) .

Every rearrangement is a choice of **which variable to sum out first**. Here e goes first, then s ; the other order is just as legal, and two slides on it becomes clear why that choice is the whole game.

The same thing as tables — eliminate E , then S

Write each conditional distribution as a **factor**, a table over the variables it mentions, and run the rearrangement as three table operations.

start	$T_1(B) T_2(S) T_3(E, B, S) T_4(D, E) T_5(C, E)$	
fix the evidence	$T_1(B) T_2(S) T_3(E, B, S) T_6(E) T_7(E)$	keep only the d^1 row of T_4 , the c^1 row of T_5
eliminate E	$T_1(B) T_2(S) T_8(B, S)$	$T_8(B, S) = \sum_e T_3(e, B, S) T_6(e) T_7(e)$
eliminate S	$T_1(B) T_9(B)$	$T_9(B) = \sum_s T_2(s) T_8(B, s)$

Each elimination is the same move: **multiply every factor that mentions the variable, then sum it out**, leaving a smaller table. Two factors remain and neither mentions a hidden variable, so normalise:

$$P(b \mid d^1, c^1) = \frac{T_1(b) T_9(b)}{\sum_{b'} T_1(b') T_9(b')}.$$

The cost is the largest table you build — and the order decides it

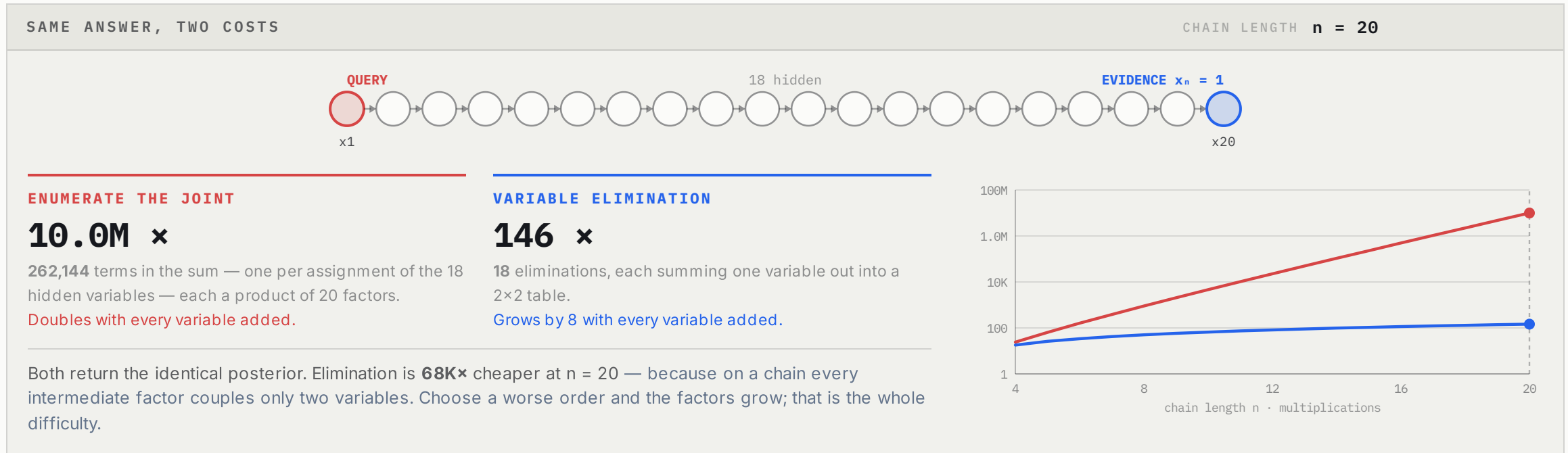
Each elimination first **builds the product** of the factors that mention the variable, and that product is a table over their union of variables:

STEP	FACTORS MULTIPLIED	TABLE BUILT	ENTRIES
eliminate E	$T_3(E, B, S) \cdot T_6(E) \cdot T_7(E)$	over (E, B, S)	$2^3 = 8$
eliminate S	$T_2(S) \cdot T_8(B, S)$	over (B, S)	$2^2 = 4$

The largest table was 8 entries — the size of $P(E \mid B, S)$ itself. Nothing bigger than a table we already had was ever built. Brute enumeration instead forms one term per assignment of the hidden variables, **and that count doubles with every hidden variable added.**

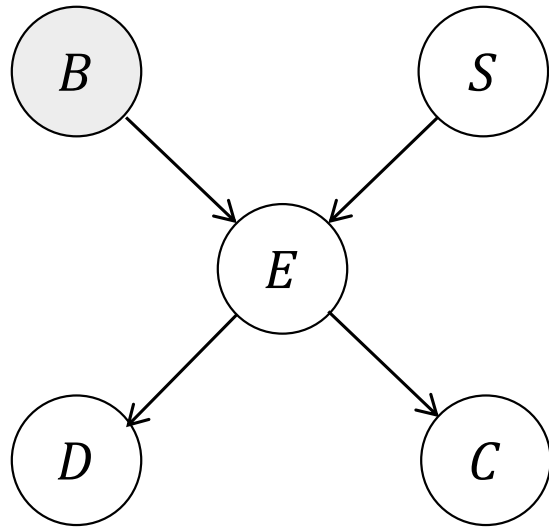
The largest intermediate table depends on the **elimination order**, and finding the best order is itself NP-hard. So the ordering is a heuristic: **often linear, sometimes still exponential** — the sum has not been made cheap, it has been made cheap *when the graph allows it*.

What the ordering is worth



Both columns compute the identical number. On the left, enumeration: one term per assignment of the hidden variables, so the count doubles with every variable added. On the right, elimination: each hidden variable is summed out once into a small table, so the count grows by a constant. **At twenty variables it is 262 144 terms in one sum against 146 multiplications in total** — and both answers agree to the last digit.

Generate one sample — 1/5 · Draw the battery



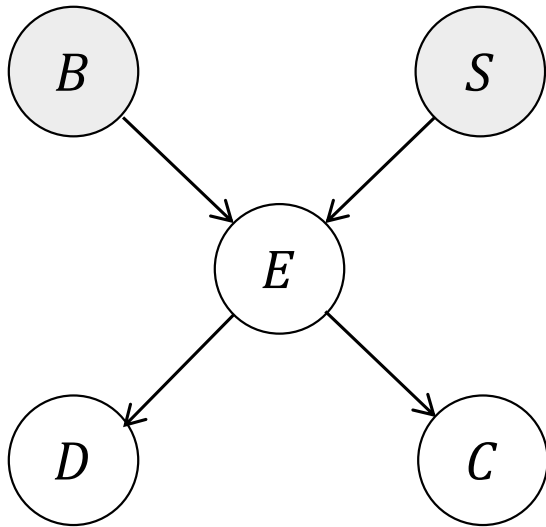
B	S	E	D	C
1				

Sample from $P(B)$

Sample each variable once its parents are available.

Draw B from $p(B)$. This sample starts with $B = 1$.

Generate one sample — 2/5 · Draw the solar panel



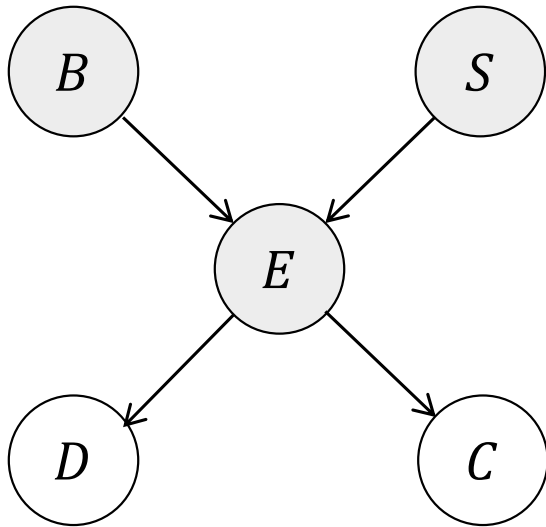
B	S	E	D	C
1	1			

Sample from $P(S)$

Sample each variable once its parents are available.

Draw S from $p(S)$. The two root nodes have no parents.

Generate one sample — 3/5 · Draw the electrical state



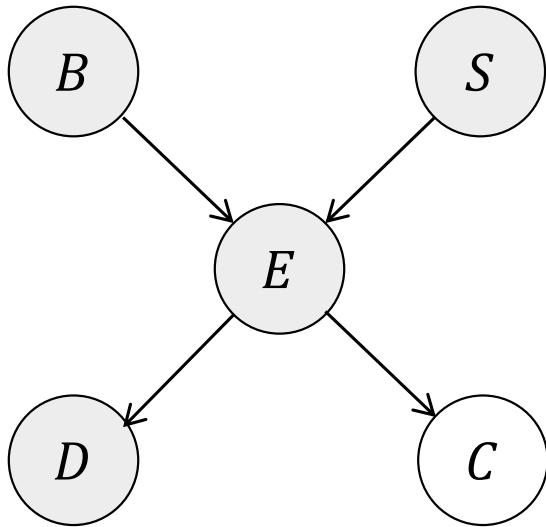
B	S	E	D	C
1	1	1		

Sample from $P(E|B = 1, S = 1)$

Sample each variable once its parents are available.

Now both parents are available: draw E from $p(E | B = 1, S = 1)$.

Generate one sample — 4/5 · Draw the trajectory



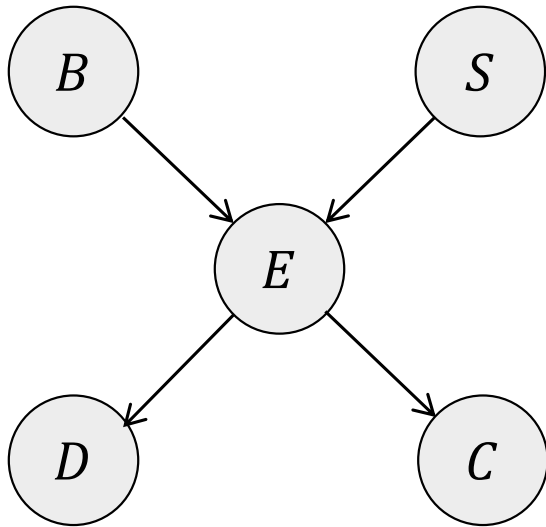
B	S	E	D	C
1	1	1	0	

Sample from $P(D|E = 1)$

Sample each variable once its parents are available.

Draw D from $p(D | E = 1)$. A sampled outcome can be zero even when its parent is one.

Generate one sample — 5/5 · Draw communication



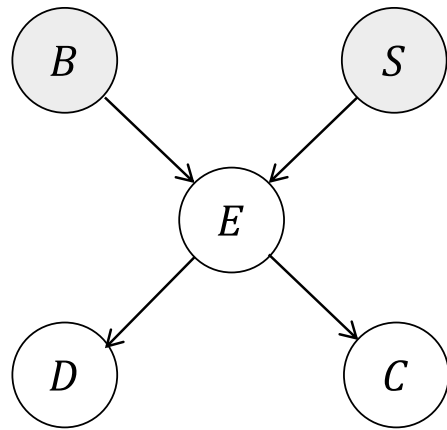
B	S	E	D	C
1	1	1	0	0

Sample from $P(C|E = 1)$

Sample each variable once its parents are available.

Draw C from $p(C | E = 1)$. The row is now one complete sample from the joint.

Reject rows that disagree with the evidence



$$P(b^1|d^1, c^1) = 1/3$$

$$P(b^0|d^1, c^1) = 2/3$$

B	S	E	D	C
1	1	1	0	0
0	1	0	1	0
1	0	1	1	1
0	1	0	0	1
0	1	1	1	1
0	1	0	0	1
0	0	0	1	0
0	1	1	1	0
0	1	0	1	1

Three cases coincide observations d^1, c^1

Three of the nine displayed rows match $D = C = 1$.

Among the three matching rows, one has $B = 1$: the sample estimate is $1/3$. This is an estimate, not an exact posterior.

Weight the evidence instead of rejecting a row

every row is kept — but each one pays for the evidence it was handed

B	S	E	D	C	weight
1	0	1	1	1	$P(d^1 e^1) P(c^1 e^1)$
0	1	1	1	1	$P(d^1 e^1) P(c^1 e^1)$
0	1	0	1	1	$P(d^1 e^0) P(c^1 e^0)$

The red D, C columns were never sampled; the last column is what each row pays.

Rejection threw away six rows in nine. The waste is in sampling D and C at all, only to check them against values we already know. So **do not sample them** — write $D = 1, C = 1$ straight into every row.

That is cheating, and the weight is the price. A row whose sampled E would have produced $D = C = 1$ on its own has earned them; a row whose E makes them improbable was *handed* them. So each row counts in proportion to how plausibly it could have produced the evidence:

$$w = \underbrace{P(D = 1 \mid E)}_{\text{how likely this row's } E \text{ makes } D=1} \cdot \underbrace{P(C = 1 \mid E)}_{\text{and } C=1}$$

Nothing is discarded: every row votes, and its vote is scaled by its weight — full strength if it earned its evidence, faintly if it was forced.

How different a 0.72 and a 0.02 really are

two illustrative conditionals, the same evidence, very different weight

Give $E \rightarrow D$ and $E \rightarrow C$ illustrative tables: $P(D=1 \mid E=1) = 0.9$, $P(C=1 \mid E=1) = 0.8$, and $P(D=1 \mid E=0) = 0.1$, $P(C=1 \mid E=0) = 0.2$.

$$w(E=1) = 0.9 \times 0.8 = 0.72$$

$$w(E=0) = 0.1 \times 0.2 = 0.02$$

A row with $E = 1$ explains $D = C = 1$ **thirty-six times more plausibly** than a row with $E = 0$. Both rows are kept — one just earns its evidence, and the other barely does.

Three rows, weighed instead of counted

the same three surviving rows from p. 36, now scored instead of tallied

B	S	E	WEIGHT $w = P(D=1 \mid E)P(C=1 \mid E)$
1	0	1	$0.9 \times 0.8 = 0.72$
0	1	1	$0.9 \times 0.8 = 0.72$
0	1	0	$0.1 \times 0.2 = 0.02$

$$\hat{P}(B=1 \mid D=C=1) = \frac{0.72}{0.72 + 0.72 + 0.02} = \frac{0.72}{1.46} \approx \mathbf{0.493}.$$

Not 1/3. The earlier rejection-sampling count gave every surviving row one vote regardless of how well it explained the evidence. The row with $E = 0$ survived too — but it barely earned its place, and its vote counts accordingly.

Why B, S and E carry no weight of their own

they already paid — by how often they were drawn

Sample B, S, E honestly from $P(B)$, $P(S)$, $P(E \mid B, S)$ and their probabilities are already built into *how many* rows carry each value. With $P(B=1) = 0.1$, roughly one row in ten already has $B = 1$ — not because a weight said so, but because that is how often the sampler drew it.

D and C get no such chance: they are never drawn, only *decreed* to be 1 in every row. The only way their probability can still be counted is to pay it **afterwards**, as a weight. **Sample a variable and it counts by frequency; clamp it and it counts by weight — never both.** That is why $P(B)$, $P(S)$ and $P(E \mid B, S)$ never reappear inside w .

Likelihood weighting in one sentence: **whatever you did not sample, you must pay for.**

Why the weighted vote is the right answer

Two facts, and the method follows from their product. A row (b, s, e) is drawn with the **prior** probability of its unclamped part, and it is then weighted by the evidence's probability given that part:

$$\underbrace{P(b) P(s) P(e \mid b, s)}_{\text{how often the row is drawn}} \times \underbrace{P(D = 1 \mid e) P(C = 1 \mid e)}_{\text{its weight}} = P(b, s, e, D = 1, C = 1).$$

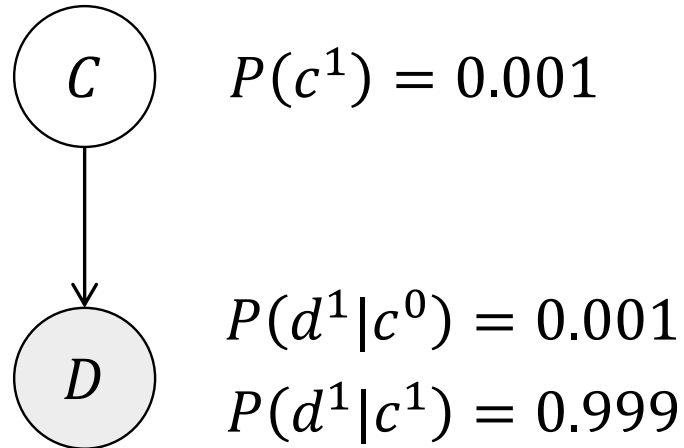
Drawn-frequency times weight is the **joint probability of the row together with the evidence** — the very quantity rejection sampling estimates by keeping only matching rows. So weighted rows carry the same information as surviving rows, and none had to be thrown away. Averaging the weights over rows with $B = 1$, against the weights over all rows, cancels the constant $P(D = 1, C = 1)$:

$$\hat{P}(B = 1 \mid D = C = 1) = \frac{\sum_k w_k \mathbf{1}[B_k = 1]}{\sum_k w_k} \longrightarrow \frac{P(B = 1, D = 1, C = 1)}{P(D = 1, C = 1)}.$$

Rejection and weighting estimate the same posterior. The difference is only that **weighting spends every sample** — and when the evidence is rare, that is the difference between an answer and none.

Rare causes — why weighting can still struggle

Compare an exact posterior with rare prior draws



Bayesian approach :

$$\begin{aligned}
 P(c^1 | d^1) &= \frac{P(d^1 | c^1)P(c^1)}{P(d^1 | c^1)P(c^1) + P(d^1 | c^0)P(c^0)} \\
 &= \frac{0.999 \times 0.001}{0.999 \times 0.001 + 0.001 \times 0.999} \\
 &= 0.5
 \end{aligned}$$

A rare cause C can have posterior probability 0.5 after observing $D = 1$.

But likelihood weighting still proposes $C = 1$ with probability 0.001. In 1,000 draws, the probability of missing it entirely is $0.999^{1000} \approx 0.368$.

A run with no $C = 1$ sample estimates zero. That is a sampling failure; the model's conditional probability has not become zero.

When exact inference is hopeless — sample

DIRECT SAMPLING, WITH REJECTION

Walk the DAG in topological order, drawing each variable from $p(x_i \mid \text{pa}_i)$; keep only the runs that happen to match the evidence, and count.

Nine samples of the satellite network, three of which show d^1, c^1 , one of those with b^1 :

$$\hat{P}(b^1 \mid d^1, c^1) = 1/3$$

The flaw. If the evidence is unlikely, almost every sample is thrown away.

LIKELIHOOD WEIGHTING, AND ITS FLAW

Do not reject. Clamp each evidence variable to its observed value and carry a weight $w \leftarrow w \times P(x_i \mid \text{pa}_i)$ for the clamping.

A different failure mode. Take $C \rightarrow D$ with $p(c^1) = 0.001$, $p(d^1 \mid c^1) = 0.999$, $p(d^1 \mid c^0) = 0.001$. The exact posterior is $p(c^1 \mid d^1) = 0.5$. But C is still drawn from its prior. The chance of seeing no c^1 in 1,000 draws is $0.999^{1000} \approx 0.368$. Such a run reports **0 for an answer that is a half**.

The cure is to stop sampling variables independently: **Gibbs sampling** sweeps through the variables, redrawing each from its conditional given the current value of all the others, and, under suitable irreducibility and mixing conditions, approaches the posterior. A finite burn-in does not guarantee exact posterior samples. Approximate inference is not a shortcut; it is a second set of failure modes, traded for the first.

Check — cheap to store, cheap to use?

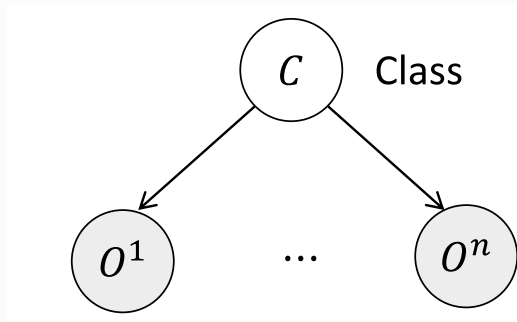
A network is sparse: every node has at most three parents, so the whole model is a few hundred numbers. What does that guarantee about the cost of exact inference?

- ☐ A It is linear in the number of nodes
- ☐ B It is at worst quadratic in the number of nodes
- ☒ C Nothing in general — exact inference can still be intractable; the cost follows the elimination order, not the storage
- ☐ D Inference is always cheaper than storage, since it never builds the full joint

Storing the joint cheaply and **summing over it** cheaply are different problems. Exact inference is NP-hard in general, and its real cost is governed by how large the intermediate factors grow as variables are eliminated — the treewidth — which a sparse-looking graph can still make enormous. This is why the chapter has to talk about elimination order at all.

Naive Bayes — classification is posterior inference

A familiar use of the same graph



One class node, many observed

For an illustrative two-class, two-feature problem:

CLASS	PRIOR × FIRST-FEATURE LIKELIHOOD × SECOND-FEATURE LIKELIHOOD	UNNORMALISED SCORE
<i>A</i>	$0.4 \times 0.8 \times 0.5$	0.16
<i>B</i>	$0.6 \times 0.2 \times 0.5$	0.06

Thus $P(A \mid o_1, o_2) = 0.16/0.22 \approx 0.727$. Choose *A* under equal misclassification costs.

“Naive” describes the **conditional-independence assumption**. It does not mean that the observed features must be marginally independent.

— LEARNING

Learning a model from data

Inference assumes the local tables are given. Learning estimates those tables from observations.

Learning the tables — reuse Lecture 2 locally

Parameter learning, with the graph fixed

For complete discrete data, count each node separately for each configuration of its parents:

$$\hat{p}(X_i = k \mid \text{pa}_i = j) = \frac{N_{ijk}}{\sum_{k'} N_{ijk'}}.$$

In the illustrative fault model, suppose 8 of 10 faulty machines and 18 of 90 healthy machines trigger the alarm.

CONDITIONAL ROW	MLE	POSTERIOR MEAN WITH A SEPARATE BETA(1,1) PRIOR
$P(A = 1 \mid F = 1)$	$8/10 = 0.8$	$9/12 = 0.75$
$P(A = 1 \mid F = 0)$	$18/90 = 0.2$	$19/92 \approx 0.207$

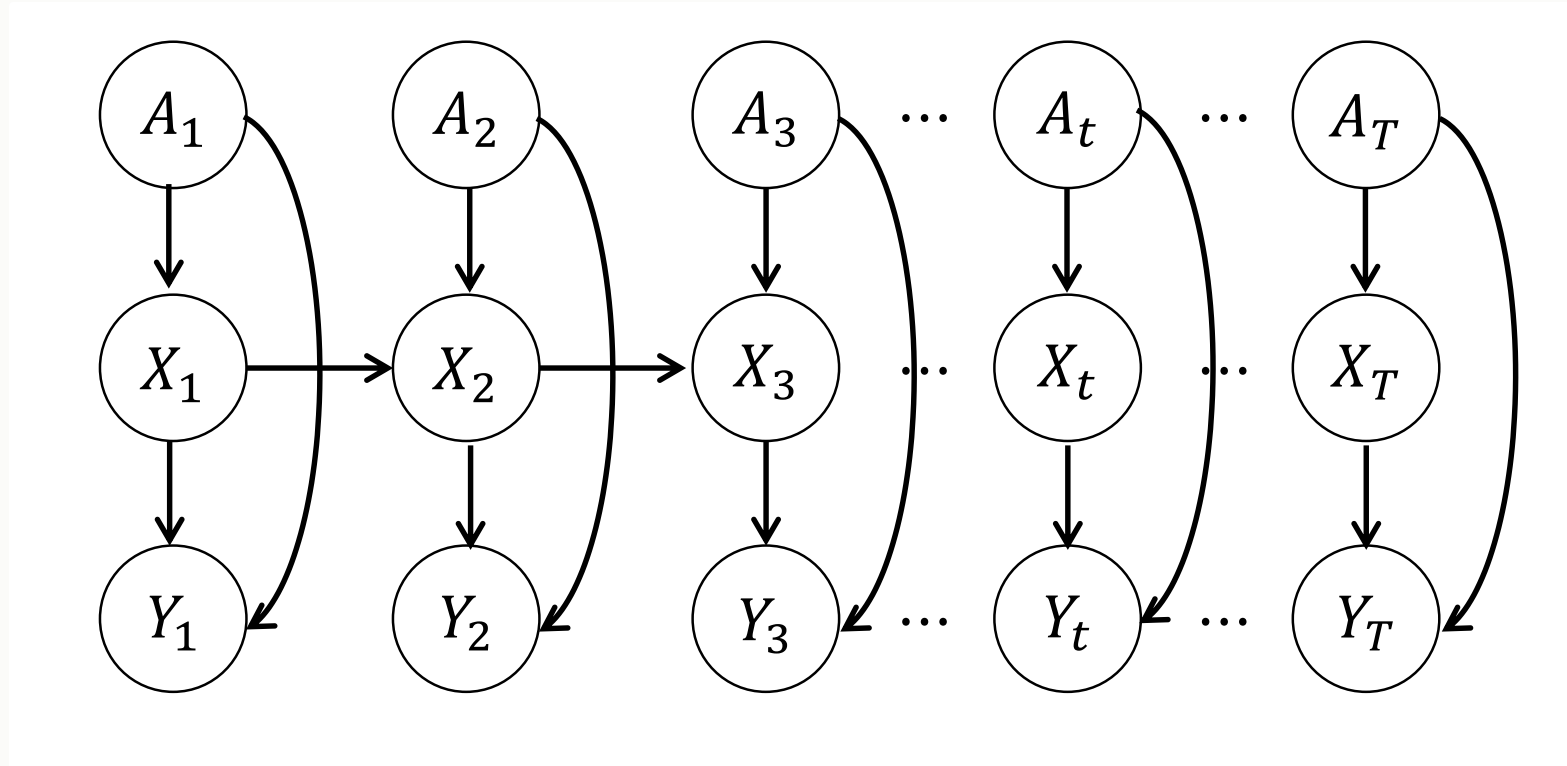
A Bayesian network gives **many small estimation problems**. Missing or latent variables require inference as part of learning; counting observed rows alone is then insufficient.

— FILTERING

Tracking a hidden state through time

Use the transition model to predict, then use a new observation to update.

A dynamic Bayesian network — repeat the same local structure



Input, hidden state and observation stay in three aligned rows.

The horizontal arrows carry the state forward. Vertical and curved arrows describe what happens within one time step.

The same graph, unrolled through time

Nothing so far said the variables were simultaneous. Index them by time and the identical machinery becomes a model of a system evolving.

$$P(S_1, \dots, S_T) = P(S_1) \prod_{t=2}^T P(S_t \mid S_{1:t-1}) \xrightarrow{\text{two assumptions}} P(S_1) \prod_{t=2}^T P(S_t \mid S_{t-1})$$

1 • THE MARKOV ASSUMPTION

$$P(S_t \mid S_1, \dots, S_{t-1}) = P(S_t \mid S_{t-1})$$

the future is conditionally independent of the past given the present, $S_{t+1} \perp S_{1:t-1} \mid S_t$. Which is a *conditional independence* — a chain structure, read off a graph.

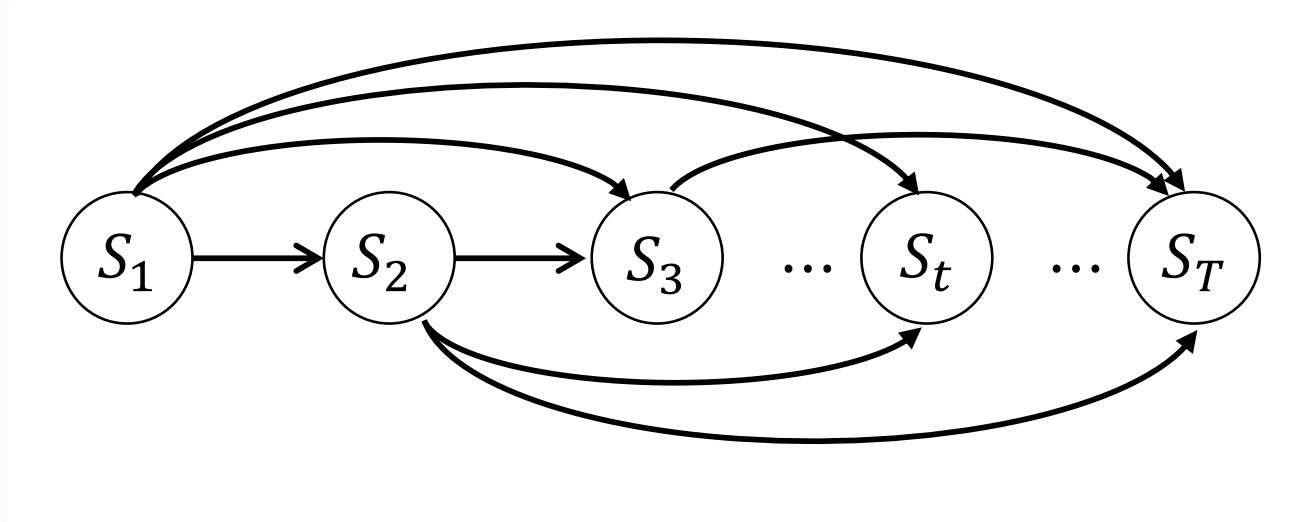
2 • THE STATIONARITY ASSUMPTION

$$P(S_{t+1}=s' \mid S_t=s) = P(S'=s' \mid S=s) \quad \forall t$$

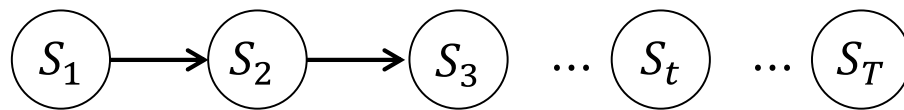
one transition table, shared by every time step, instead of T of them.

Lecture 7 will open by *assuming* both of these. Markov structure is a conditional-independence assumption. Time homogeneity is a separate parameter-sharing assumption: the transition table does not change with time.

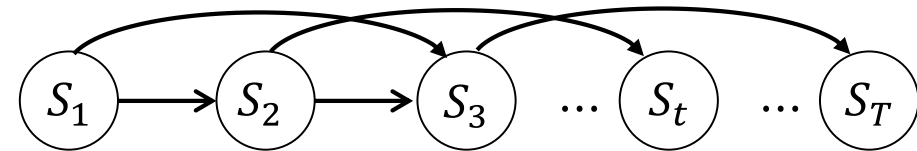
Markov order — how far back do the arrows reach?



Without an assumption, a state may depend on the whole past.



First Order Markov Chain



Second Order Markov Chain

First-order and second-order chains, side by side.

A first-order model keeps only the previous state. A second-order model keeps the previous two.

Fit a transition matrix — count where each state goes

Use count-and-normalise rule

Use the source convention $M_{ij} = P(S_{t+1} = i \mid S_t = j)$: **column j is the distribution of the next state given the current state.**

If we have $s_{1:t} = 1, 3, 2, 4, 1, 4, 3, 5, 1, 3, 4, 2, 1, 4, 4, 2, 4, 5, 1, 3, 3, 4, \dots$ $\longrightarrow \theta_{3|1} = \frac{3}{5}$

Five departures from state 1; three go to state 3, giving $\hat{\theta}_{3|1} = 3/5$.

For the illustrative observed sequence A, A, B, A, B, B , the four transition counts are $N_{AA} = 1, N_{BA} = 2, N_{AB} = 1, N_{BB} = 1$.

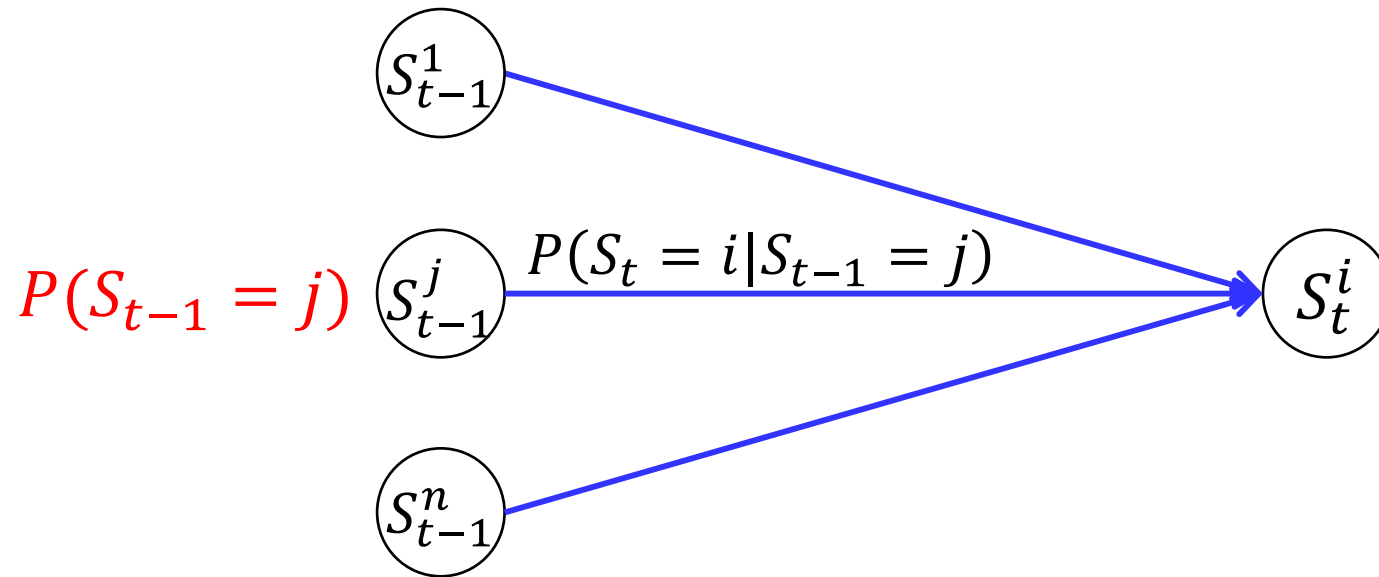
$$\hat{M} = \begin{bmatrix} 1/3 & 1/2 \\ 2/3 & 1/2 \end{bmatrix}, \quad p_{t+1} = \hat{M}p_t.$$

Starting at A gives $p_t = [1, 0]^\top$, so the next-state probabilities are $[1/3, 2/3]^\top$. Every column sums to one.

This is **model learning** from a trajectory. Predicting with $p_{t+1} = \hat{M}p_t$ is a different operation, performed after the matrix is specified or estimated.

Predict a marginal — collect probability from every predecessor

One destination receives mass from several possible previous states



Each blue arrow contributes transition probability times previous-state probability.

$$P(S_t = i) = \sum_j P(S_t = i | S_{t-1} = j) P(S_{t-1} = j), \quad p_t = M p_{t-1}.$$

Add all routes into state i . Repeating this calculation evolves the marginal; a stationary distribution is one that the update leaves unchanged.

A stationary model need not start in a stationary distribution

Eparate two uses of "stationary"

Time-homogeneous transitions: the same matrix M is used at each step. **Stationary distribution:** a probability vector p_∞ satisfying $Mp_\infty = p_\infty$.

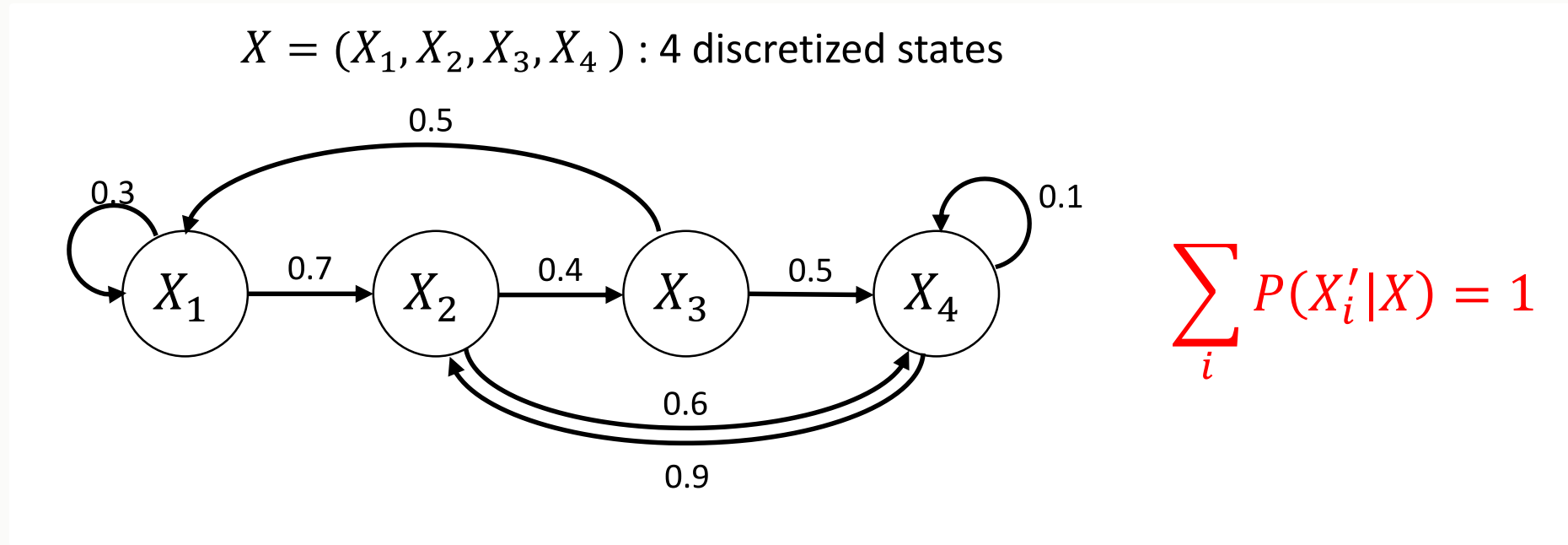
For a separate illustrative two-state model,

$$M = \begin{bmatrix} 0.9 & 0.2 \\ 0.1 & 0.8 \end{bmatrix}, \quad p_\infty = \begin{bmatrix} 2/3 \\ 1/3 \end{bmatrix}.$$

From $p_0 = [1, 0]^\top$, the first two distributions are $p_1 = [0.9, 0.1]^\top$ and $p_2 = [0.83, 0.17]^\top$: the marginal distribution changes even though M stays fixed.

For a finite irreducible, aperiodic chain, the distribution converges to its unique stationary distribution. A general Markov chain need not have that convergence property.

State transitions — a graph of possible next states

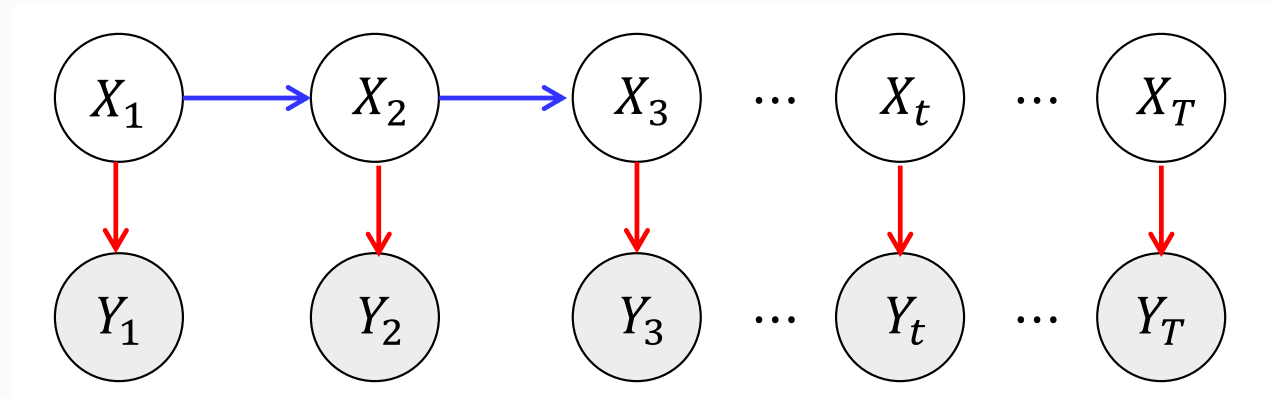


These four nodes are state values, not four time steps.

From X_2 , move to X_3 with probability 0.4 or X_4 with probability 0.6. Outgoing probabilities sum to one.

Hidden Markov models — name the question before computing

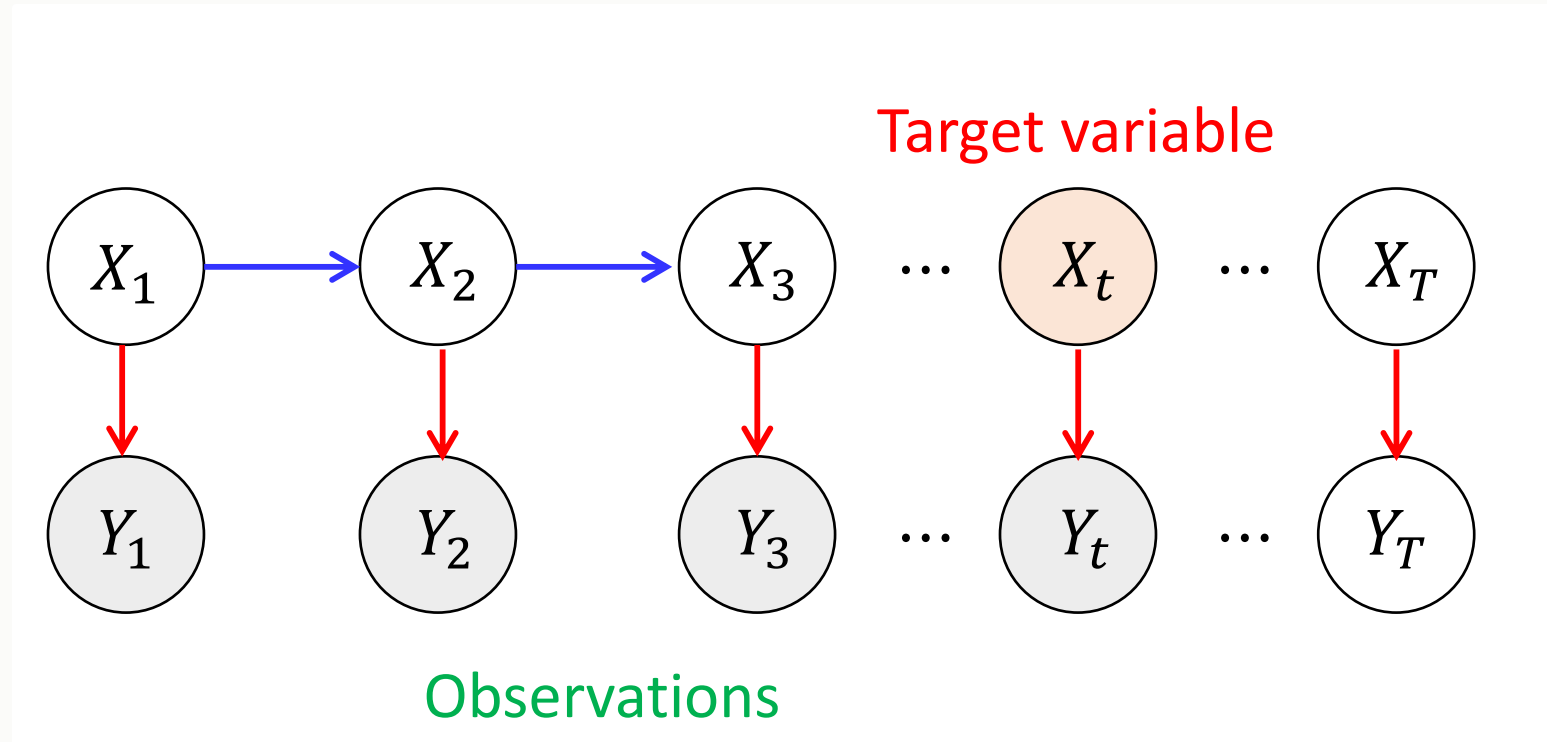
Same hidden states, different information sets



Blue transitions, red emissions and grey observed nodes.

QUESTION	TARGET	WHAT THE ANSWER USES
Filtering	$p(x_t \mid y_{1:t})$	observations available now
Prediction	$p(x_{t+h} \mid y_{1:t})$	current belief, propagated into the future
Smoothing	$p(x_t \mid y_{1:T}), t < T$	later observations to revise the past
Sequence likelihood / best hidden path	$p(y_{1:T}) / \arg \max_{x_{1:T}} p(x_{1:T} \mid y_{1:T})$	model comparison / Viterbi decoding

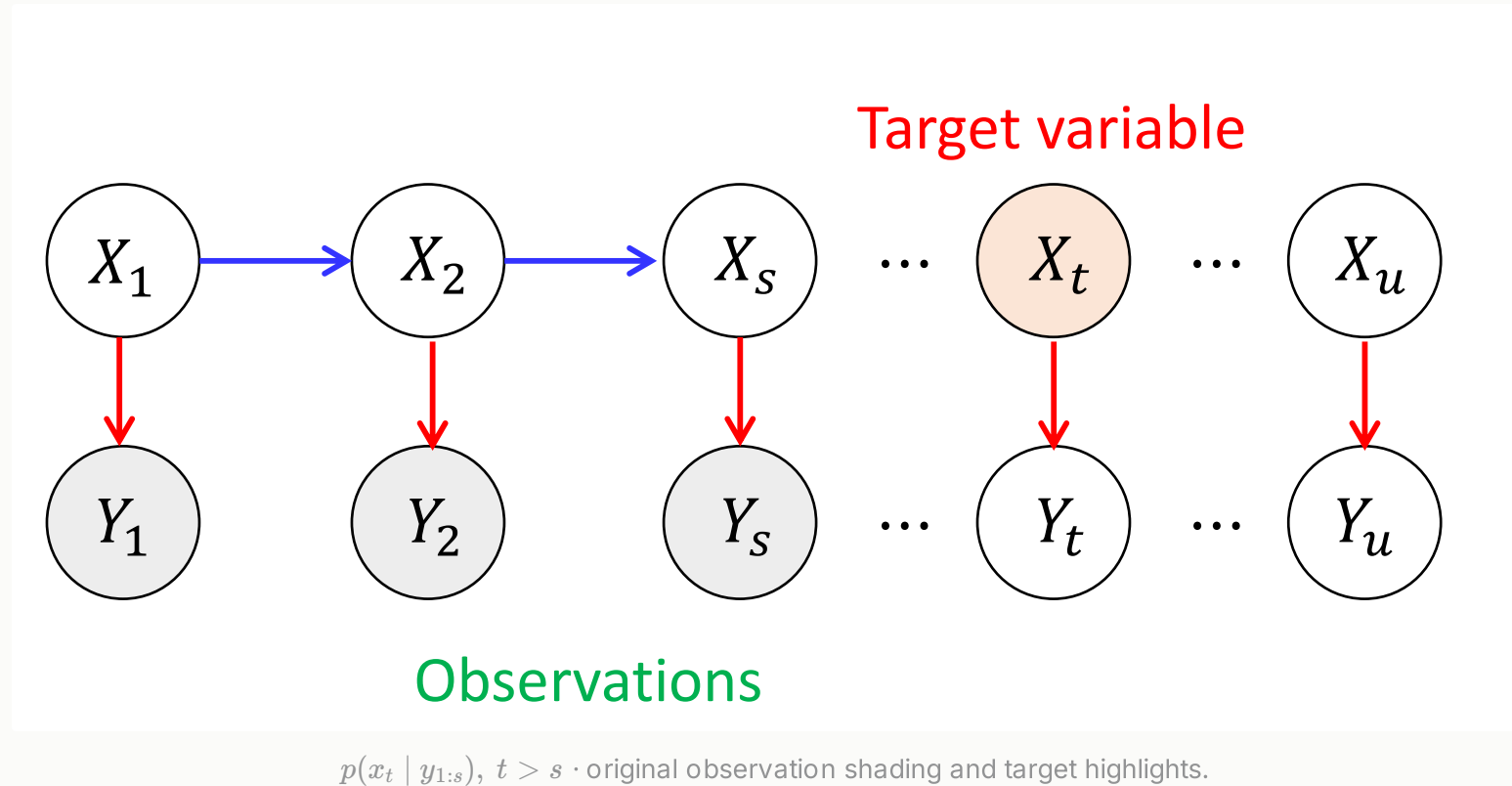
Filtering — locate the present



$$p(x_t \mid y_{1:t}) \cdot \text{original observation shading and target highlights.}$$

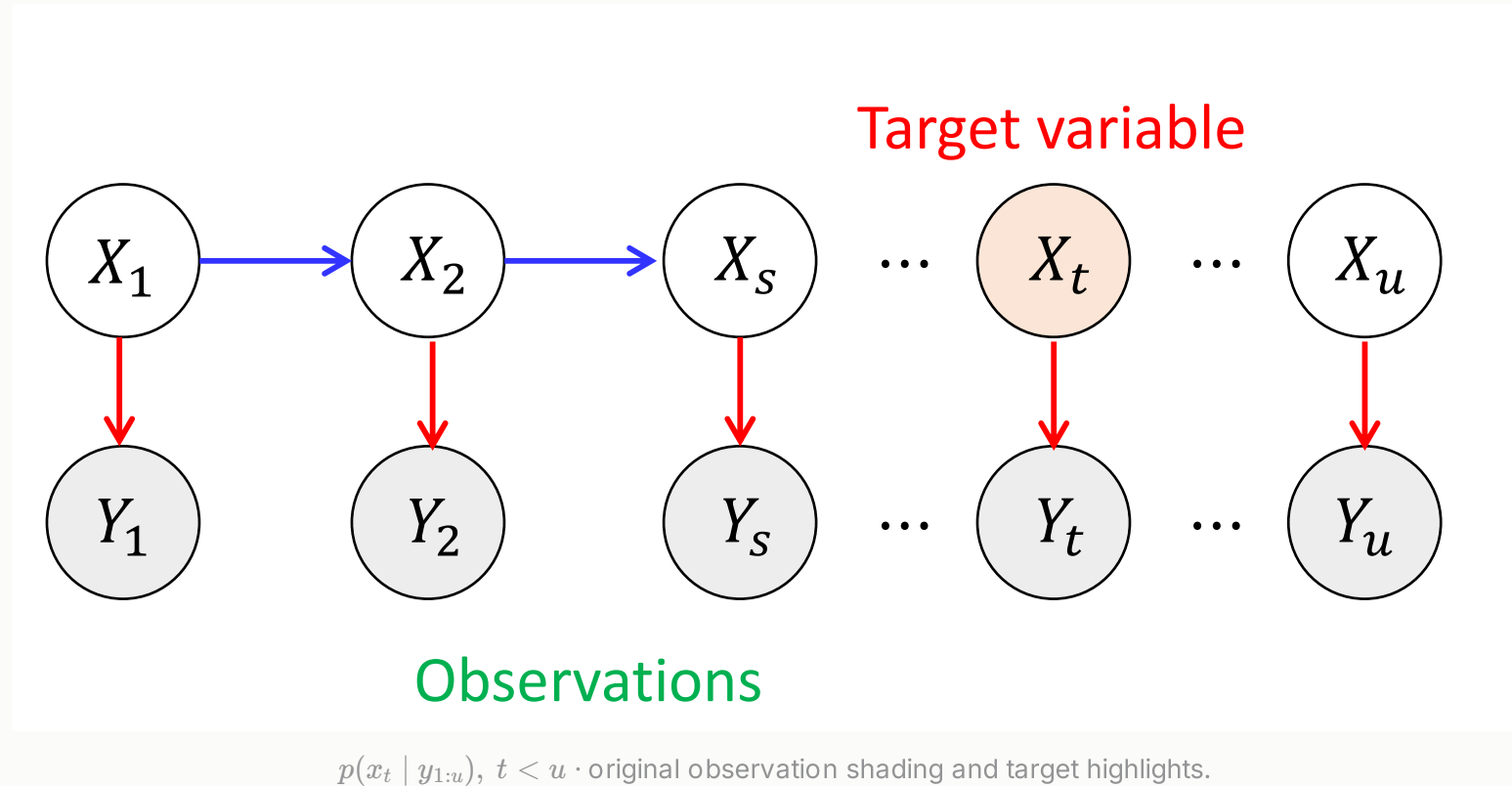
The orange node is the current target. Use the grey observations up to now; the future observation is still unknown.

Prediction — look beyond the observations



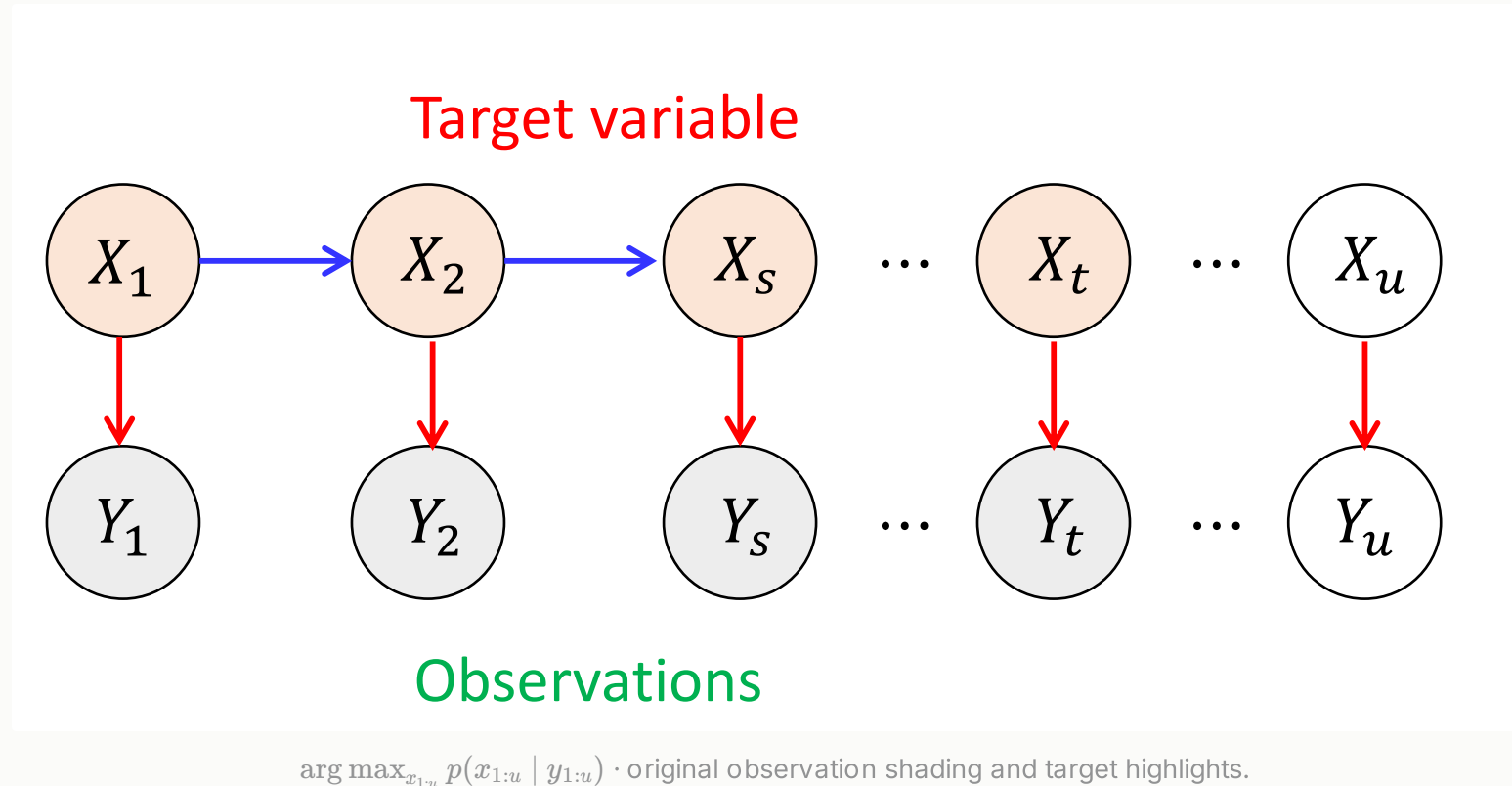
The observations stop at s . Push the current belief forward through transitions to the orange future node.

Smoothing — use later evidence to revise the past



The target stays at t , but grey observations extend to u . Later measurements can change what we believe happened earlier.

Decoding — choose a whole hidden path



All hidden nodes are targets together. Find the most probable complete path; separate per-time guesses need not form that path.

Filtering — Bayes' rule, once per time step

Hide the state and observe an emission — $X_t \rightarrow Y_t$ over a chain $X_{t-1} \rightarrow X_t$ — and you have a **hidden Markov model**. The standard query is *filtering*, the belief about now given everything seen so far:

$$P(x_t \mid y_{1:t}) \propto \underbrace{P(y_t \mid x_t)}_{\text{corrector}} \sum_{x_{t-1}} \underbrace{P(x_t \mid x_{t-1}) P(x_{t-1} \mid y_{1:t-1})}_{\text{predictor}}$$

Read it as Lecture 2's loop, run once per tick: yesterday's posterior is pushed through the dynamics to become today's **prior**; today's measurement is the **likelihood**; the product is today's posterior. Make the chain linear and the noise Gaussian and every term stays Gaussian — the recursion then carries only a mean and a covariance, and it has a name: the **Kalman filter**.

One filtering step — predict, observe, normalise

a two-state machine, healthy or faulty

Yesterday's posterior fault probability is 0.2. A fault persists with probability 0.8; a healthy machine develops a fault with probability 0.1.

Predict today's state:

$$P(F_t) = 0.8(0.2) + 0.1(0.8) = 0.24.$$

Now an alarm occurs, with $P(A_t | F_t) = 0.9$ and $P(A_t | \neg F_t) = 0.1$.

Correct using the observation:

$$P(F_t | A_t) = \frac{0.9(0.24)}{0.9(0.24) + 0.1(0.76)} \approx 0.740.$$

The transition changes 0.20 to 0.24; the measurement changes 0.24 to 0.74. **The corrected belief becomes the starting belief at the next time step.**

Temperature thread — infer a hidden heater mode

shared teaching example · predict before revealing the calculation

In a two-mode variant, the heater is good (G) or weak (W). Initially $P(G) = 0.6$. Next-step probabilities are $P(G' \mid G) = 0.9$ and $P(G' \mid W) = 0.3$. A warm reading has likelihood 0.8 under G' and 0.2 under W' .

Predict: First predict the next mode without the reading. Then decide whether a warm reading should raise or lower its good-mode probability.

Calculate and check. $P(G') = 0.9(0.6) + 0.3(0.4) = 0.66$. After a warm reading,

$$P(G' \mid \text{warm}) = \frac{0.8(0.66)}{0.8(0.66) + 0.2(0.34)} = \frac{132}{149} \approx 0.886.$$

Prediction uses the transition table; updating uses the observation likelihood. The two tables have different jobs.

Try it — the next reading is cool

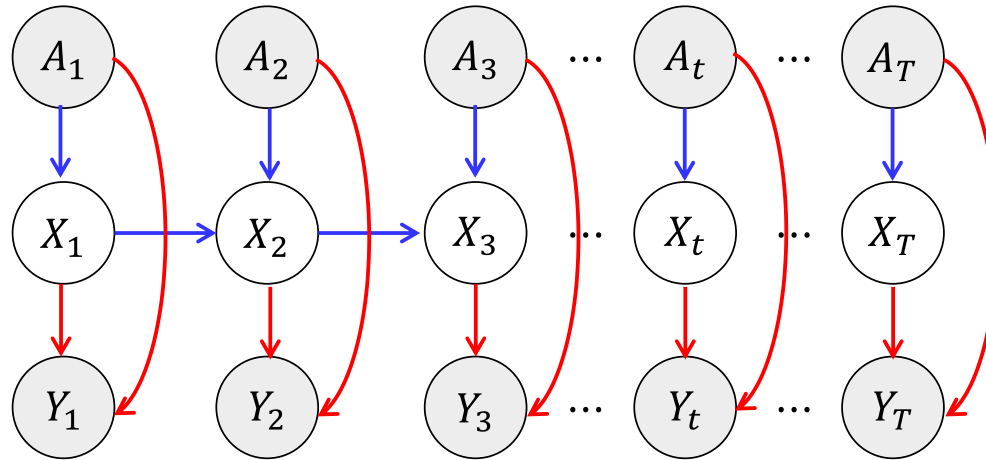
work independently · reveal only after writing an answer

Keep the same predicted probabilities (0.66, 0.34). Instead observe “cool,” the complement of “warm.” Its likelihoods are therefore 0.2 under G' and 0.8 under W' . Calculate the new posterior.

Check your answer. $P(G' \mid \text{cool}) = 0.132 / (0.132 + 0.272) = 33/101 \approx \mathbf{0.327}$. Do not start from the posterior of the warm-reading example; these are alternative observations.

When changing a condition, identify exactly which factors change. Do not reuse evidence from a different scenario.

Add an input — the original IOHMM



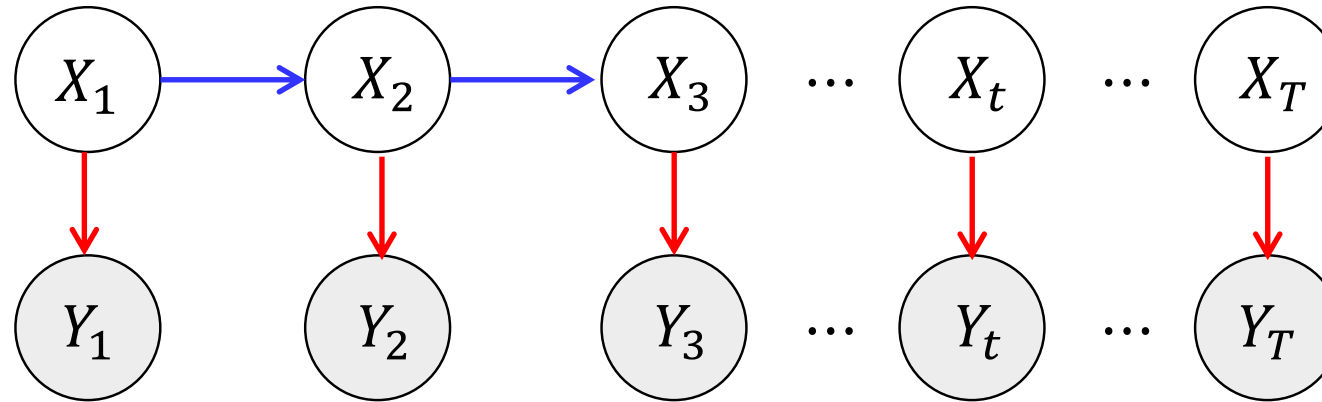
The state transition model : $P(X_t | X_{t-1}, A_t)$

The observation model : $P(Y_t | X_t, A_t)$

A_t affects both the transition and the observation.

This is an input-output HMM. Choosing inputs becomes a decision problem only after specifying utilities and the information available when acting.

A continuous hidden state — the same two-row picture



Transition model : $x_t = \mathbf{A}_t x_{t-1} + \eta_t^x,$

Emission model : $y_t = \mathbf{B}_t x_t + \eta_t^y,$

$\eta_t^x \sim \mathcal{N}(\bar{x}_t, \Sigma_t^x)$

$\eta_t^y \sim \mathcal{N}(\bar{y}_t, \Sigma_t^y)$

The source writes the emission matrix as B_t ; the next slide uses C .

The graph still separates transitions and measurements. Linear equations and Gaussian noise let us carry a mean and covariance instead of a probability table.

Kalman filtering — carry the mean and covariance forward

Ource names the mean f and covariance F



$$P(x_{t-1}|y_{1:t-1}) = N(x_{t-1}|f_{t-1}, F_{t-1})$$

$$P(x_t|y_{1:t}) = N(x_t|f_t, F_t)$$

The old Gaussian belief becomes a new Gaussian belief after prediction and measurement correction.

The update carries two objects: **where the state is likely to be**, and **how uncertain that estimate remains**. The next slide writes these quantities as m_t, P_t .

The whole posterior changes. Linear Gaussian assumptions let us represent that distribution using only its mean and covariance.

Linear Gaussian state space — the Kalman version of the same update

Transition and observation are different models

$$x_t = Ax_{t-1} + w_t, \quad y_t = Cx_t + v_t, \quad w_t \sim \mathcal{N}(0, Q), \quad v_t \sim \mathcal{N}(0, R).$$

With a Gaussian initial belief and independent Gaussian noises, the posterior remains Gaussian. Track its mean and covariance instead of a table.

STAGE	CALCULATION
Predict state	$m_t^- = Am_{t-1}, P_t^- = AP_{t-1}A^\top + Q$
Weight the measurement	$K_t = P_t^- C^\top (CP_t^- C^\top + R)^{-1}$
Correct belief	$m_t = m_t^- + K_t(y_t - Cm_t^-), P_t = (I - K_t C)P_t^-$

A scalar example with $m^- = 10$, $P^- = 4$, $C = 1$, $R = 1$, and $y = 12$ gives $K = 0.8$, $m = 11.6$ **and** $P = 0.8$.

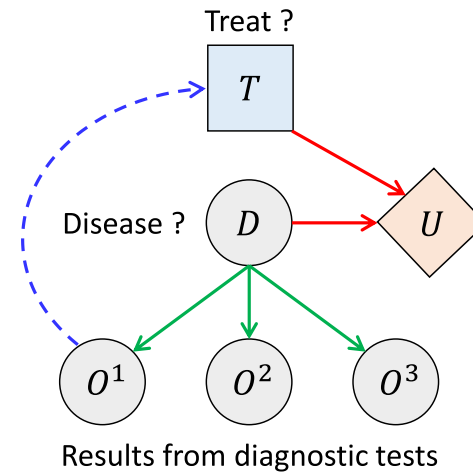
The Kalman filter is **predict** → **observe** → **update**, just like the discrete fault example. Gaussian conditioning makes those steps closed form.

— ACT 4

From belief to decision

Q4. Everything so far describes the world. Add two node types and the graph starts choosing.

Bayesian network + decision + utility



T	D	$U(T, D)$
0	0	0
0	1	-10
1	0	-1
1	1	-1

→ Conditional edge
 → Functional edge
 → Information edge (often omitted)

- A **chance node** corresponds to a random variable
- A **decision node** corresponds to each decision to be made
- ◇ A **utility node** corresponds to an additive utility component

Circle: chance. Square: decision. Diamond: utility. Dashed blue: information known before choosing.

Read the dashed arrow first: the result of test O^1 is known before treatment T is chosen.

Utility — a number for comparing outcomes

Probabilities describe **how likely** outcomes are. Utilities describe **how desirable** those outcomes are to the decision maker.

A CERTAIN OUTCOME

Assign a utility $U(s)$ to outcome s . Higher is preferred; the units can be profit, comfort or a stated combination of objectives.

AN UNCERTAIN OUTCOME

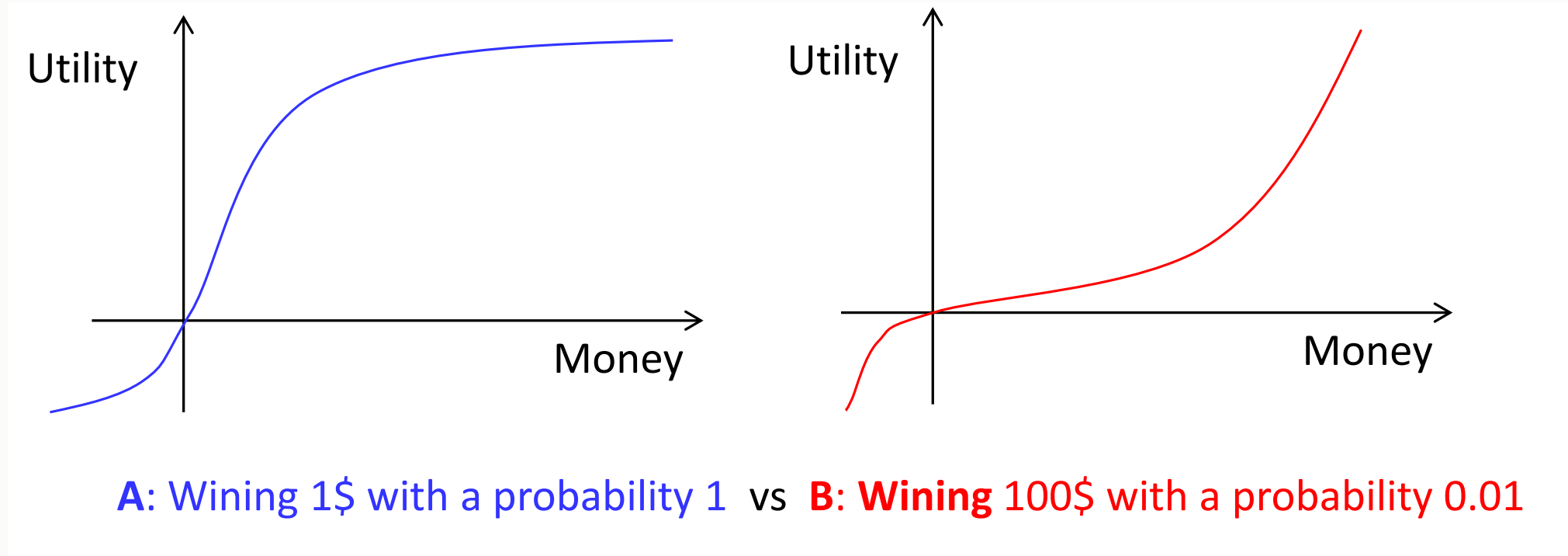
Under the expected-utility model, an action producing outcomes with probabilities p_i is valued by

$$EU(a) = \sum_i p_i U(s_i).$$

Average utilities under belief, then take the best action. That is the entire operation a decision node performs — and, run over many stages, it is the Bellman equation of Lecture 7.

Utility need not equal money. A concave utility of money represents risk aversion; the same expected-utility calculation still applies.

Utility curves — the same money can have different value



Compare the two utility shapes on the nonnegative money range.

On the nonnegative money range: blue bends downward (risk averse); red bends upward (risk seeking). Compare expected utility, not just expected money.

Utility is not necessarily money — the original lottery example

Probabilities describe beliefs; utilities describe preferences

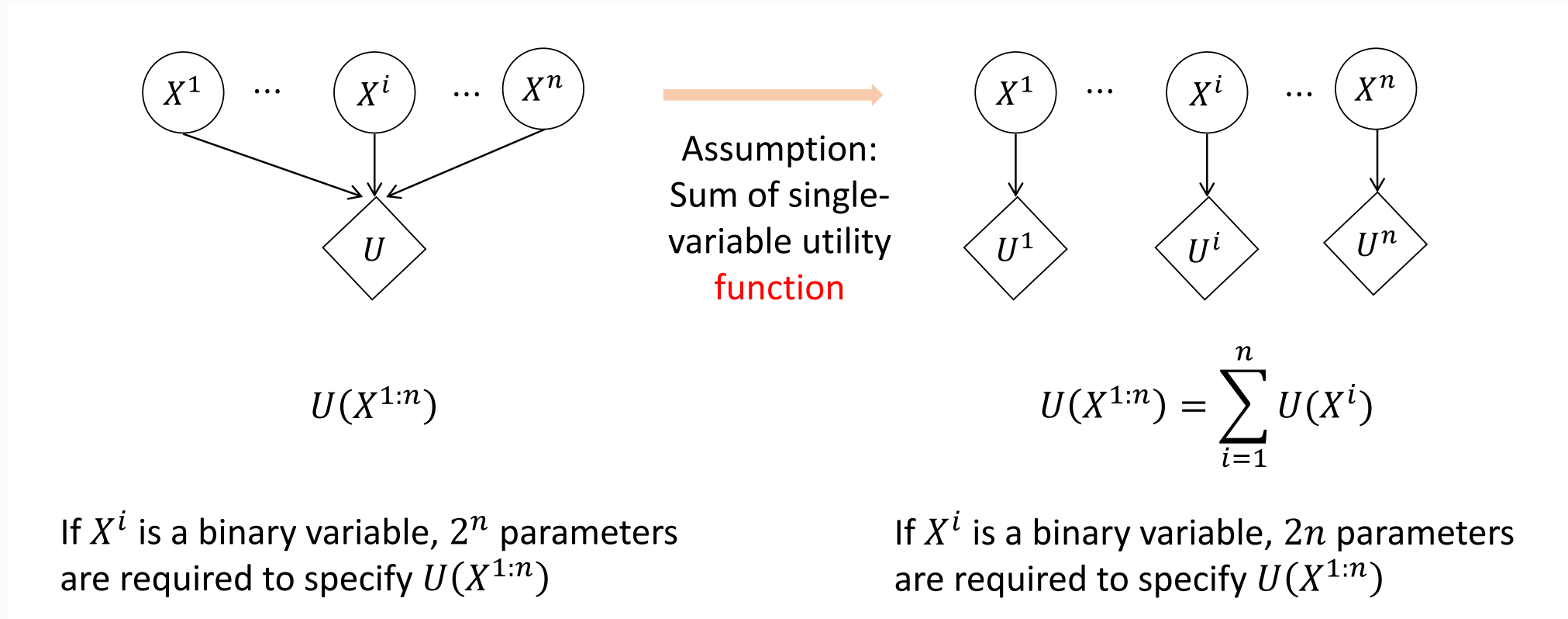
Compare **A: receive 1 dollar for sure** with **B: receive 100 dollars with probability 0.01, otherwise 0**. Both have expected money 1 dollar.

UTILITY OF MONEY $m \geq 0$	$\mathbb{E}[U(A)]$	$\mathbb{E}[U(B)]$	PREFERRED
$U(m) = \sqrt{m}$	1	$0.01(10) = 0.1$	A: risk averse
$U(m) = m$	1	$0.01(100) = 1$	indifferent: risk neutral
$U(m) = m^2$	1	$0.01(10\,000) = 100$	B: risk seeking

Same money on average, three different decisions — the shape of U is doing all the work.

Choose by $\mathbb{E}[U(M)]$, which need not equal $U(\mathbb{E}[M])$. The utility function is part of the model, and it has to be stated before two decisions can be compared.

Utility factorisation — one large function or several small ones



One diamond with many parents versus separate utility diamonds.

Additivity is an assumption about preferences. It does not say the random variables are independent.

Several consequences — when can utilities be added?

Collision avoidance

The original example distinguishes horizontal closeness H , vertical closeness V , and whether an alarm is raised A .

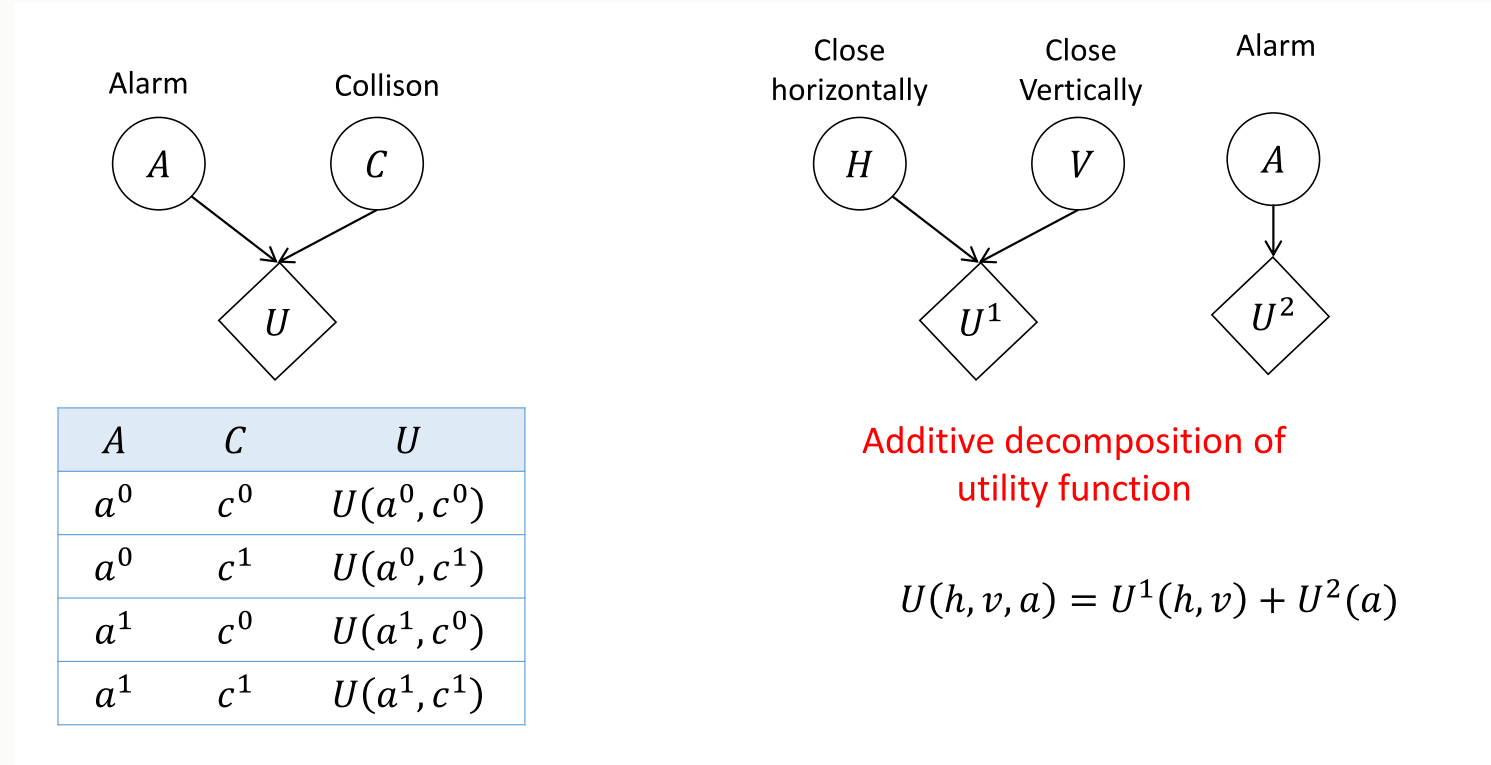
$$U(h, v, a) = U_{\text{separation}}(h, v) + U_{\text{alarm}}(a).$$

The first term represents the consequence of unsafe separation; the second can represent alarm cost. This **additive preference assumption** is separate from any independence assumption about the random variables.

REPRESENTATION FOR n BINARY VARIABLES	ENTRIES TO SPECIFY BEFORE ACCOUNTING FOR UTILITY SCALING
Arbitrary joint utility $U(x_1, \dots, x_n)$	2^n
Additive utility $\sum_i U_i(x_i)$	$2n$

Probability factorisation makes beliefs manageable. Utility factorisation can make preferences manageable. Both gain simplicity by making explicit modelling assumptions.

Collision avoidance — safety and alarm cost in one picture



Joint alarm/collision utility on the left; additive components on the right.

Unsafe horizontal and vertical separation contribute $U^1(h, v)$. Raising an alarm contributes $U^2(a)$. Add the two only if that preference model is intended.

Maximum expected utility

For action a after observation o , combine the belief model with the value of each outcome:

Probabilistic model Utility function

$P(s'|o, a)$ $U(s')$

$$EU(a|o) = \sum_{s'} P(s'|o, a) U(s')$$

Green supplies the probability; red supplies the utility.

$$a^* = \arg \max_a EU(a | o).$$

The Bayesian network supplies probabilities. The utility function supplies values. **Average outcomes, then choose the best action.**

Read the treatment diagram — compare two actions

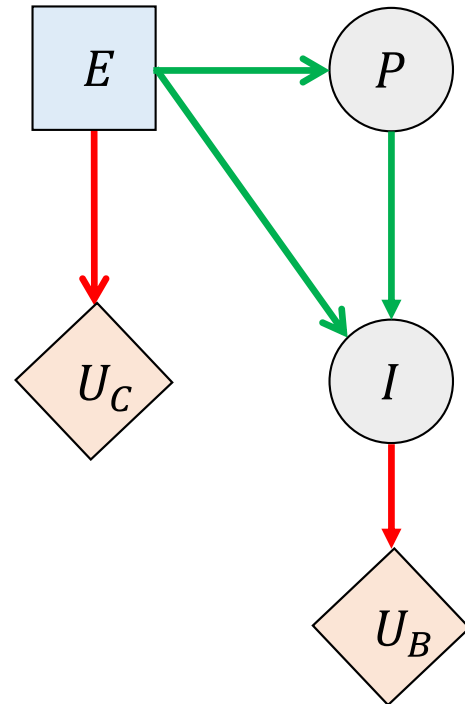
the original utility table, now used for a decision

Let $p = P(D = 1 \mid O^1)$ be the disease probability after reading the test. The graph supplies p ; the utility table supplies the consequences.

CHOOSE	HEALTHY: $D = 0$	DISEASED: $D = 1$	EXPECTED UTILITY
No treatment	0	-10	$-10p$
Treatment	-1	-1	-1

Treat when $-1 > -10p$, or $p > 0.1$; at $p = 0.1$ the actions tie. These are the illustrative utilities specified in the model.

The PhD decision — follow cost and benefit through the graph



Do PhD to wind a Nobel Prize?

The ordering : $E^* < \{I, P\}$

Domains

- $\text{dom}(E) = \{\text{do PhD, no PhD}\}$
- $\text{dom}(P) = \{\text{prize, no prize}\}$
- $\text{dom}(I) = \{\text{low, average, high}\}$

Utilities

- $U_C(\text{do PhD}) = -50000$, $U_C(\text{no PhD}) = 0$
- $U_B(\text{low}) = 100000$, $U_B(\text{average}) = 200000$, $U_B(\text{high}) = 500000$



Education changes both study cost and the income distribution.

Choose education E first. Average over prize P and income I , then add study cost U_C and income benefit U_B .

Should I do a PhD?

Example 3.4 · a decision network with real numbers

THE NETWORK

E do PhD / no PhD is the **decision**; P win the prize? and I income: low / average / high are **chance**; U_C (cost of study) and U_B (benefit of income) are **utility**.

$$E \rightarrow P, \quad \{E, P\} \rightarrow I, \quad E \rightarrow U_C, \quad I \rightarrow U_B$$

$U_C(\text{do PhD}) = -50\,000$, $U_C(\text{no PhD}) = 0$; $U_B = 100\,000/200\,000/500\,000$ for low / average / high income;
 $p(\text{prize} \mid \text{PhD}) = 0.001$ against 10^{-7} without.

THE COMPUTATION

$$U(E) = \sum_{I,P} p(I \mid E, P) p(P \mid E) [U_C(E) + U_B(I)]$$

$$U(\text{do PhD}) = 260\,174$$

$$U(\text{no PhD}) = 240\,000$$

The doctorate wins by about 20 000 — and **almost none of that comes from the prize**. It comes from the income table: a PhD shifts the chance of high income from 0.2 to 0.4, which is worth more than the 50 000 it costs.

Pick a decision, watch the utility

SHOULD I DO A PHD?

EXPECTED UTILITY $EU(E) = \sum_p \sum_I P \cdot [U_C + U_B]$

do the PhD ◀ **choose** 260,174

do not 240,000

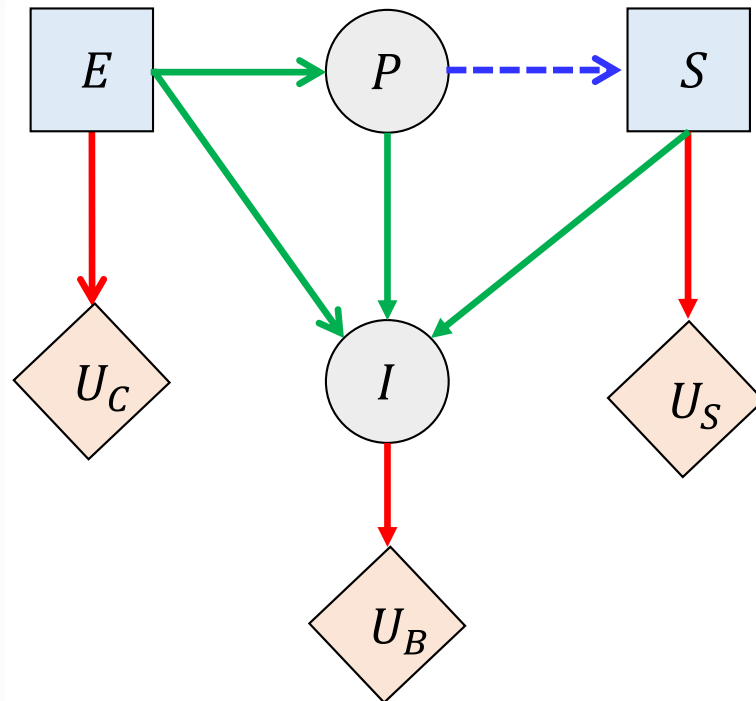
BRANCH BY BRANCH — DO THE PHD

wins the prize	$p = 0.001$	434,000
no prize	$p = 0.999$	260,000
weighted total		260,174

Read it as one step of an MDP. E is the action a , (P, I) is the next state s' , $U_C + U_B$ is the reward, and $\text{argmax}_E EU$ is a Bellman backup with the horizon set to one.

The same network, live. Choose an action and the expected utility is computed by exactly the machinery of Acts 1–3: the network gives $p(I, P \mid E)$, the utility nodes give the value, the sum gives EU . Then switch on the second decision and [watch the recommendation reverse](#).

A later start-up decision — the information arrow matters



Do PhD to wind a Nobel Prize and start-up?

The ordering : $E^* < P < S^* < I$

Domains

- $\text{dom}(E) = \{\text{do PhD, no PhD}\}$
- $\text{dom}(P) = \{\text{prize, no prize}\}$
- $\text{dom}(I) = \{\text{low, average, high}\}$
- $\text{dom}(S) = \{\text{yes, no}\}$

Utilities

- $U_C(\text{do PhD}) = -50000$, $U_C(\text{no PhD}) = 0$
- $U_B(\text{low}) = 100000$, $U_B(\text{average}) = 200000$, $U_B(\text{high}) = 500000$
- $U_S(\text{start up}) = -200000$, $U_S(\text{no start up}) = 0$



The dashed edge says the prize is known before the start-up decision.

Read the order $E \prec P \prec S \prec I$. The income parents also change to S, P , so this is a different income model.

Add one option, and the answer flips

Example 3.5 · PhD and start-ups

WHAT CHANGES

A second decision S *found a start-up?* is added, with $U_S(\text{yes}) = -200\,000$, and an information edge $P \dashrightarrow S$ — the prize is known before the start-up is founded. Crucially, **income now hangs off S and P , not off E .**

$$U(E) = U_C(E) + \sum_P p(P \mid E) \max_S \left[U_S(S) + \sum_I p(I \mid S, P) U_B(I) \right].$$

WHAT HAPPENS

$$U(\text{do PhD}) = 190\,195$$

$$U(\text{no PhD}) = 240\,000$$

Do not do the PhD. And the start-up is never founded either — "no start-up" wins the inner **max** in every branch.

Changing the income model has **reversed the decision**, because the doctorate's only surviving route to income is a 0.001 chance at the prize, which does not repay 50 000.

The income distribution now depends on start-up choice rather than education. This changes the model, not just the option set. Merely adding an optional action to an unchanged decision problem cannot reduce its maximum expected utility.

The value of information

If the decision can wait, it may pay to observe something first. Let $\text{EU}^*(o) = \max_a \text{EU}(a \mid o)$ be the value of deciding well given what is known. Then observing a new variable is worth

$$\text{VOI}(O^{\text{new}} \mid o) = \left(\sum_{o^{\text{new}}} P(o^{\text{new}} \mid o) \text{EU}^*(o^{\text{new}}, o) \right) - \text{EU}^*(o)$$

the expected value of the *better decision* the observation lets you make, minus what you would have got anyway. With the same available actions and the option to ignore the observation, its expected value is nonnegative. It must be weighed against the **cost of the observation**, which the formula does not include.

WHERE YOU WILL MEET THIS AGAIN

Lecture 4

"How much is it worth to look here?" is precisely the question an **acquisition function** answers in Bayesian optimisation. The knowledge gradient measures improvement in the value of a final recommendation. Expected improvement instead measures expected improvement over an incumbent; both guide queries, but they are different objectives. Act 4 has already written the formula.

A decision and the value of looking first

illustrative maintenance costs; this is a utility calculation

A machine is faulty with probability 0.2. Running it earns utility 10 if healthy and -30 if faulty. Stopping gives utility 0.

DECIDE WITHOUT A TEST

$$EU(\text{run}) = 0.8(10) + 0.2(-30) = 2.$$

Run is better than stop, so the best expected utility is **2**.

A PERFECT TEST BEFORE DECIDING

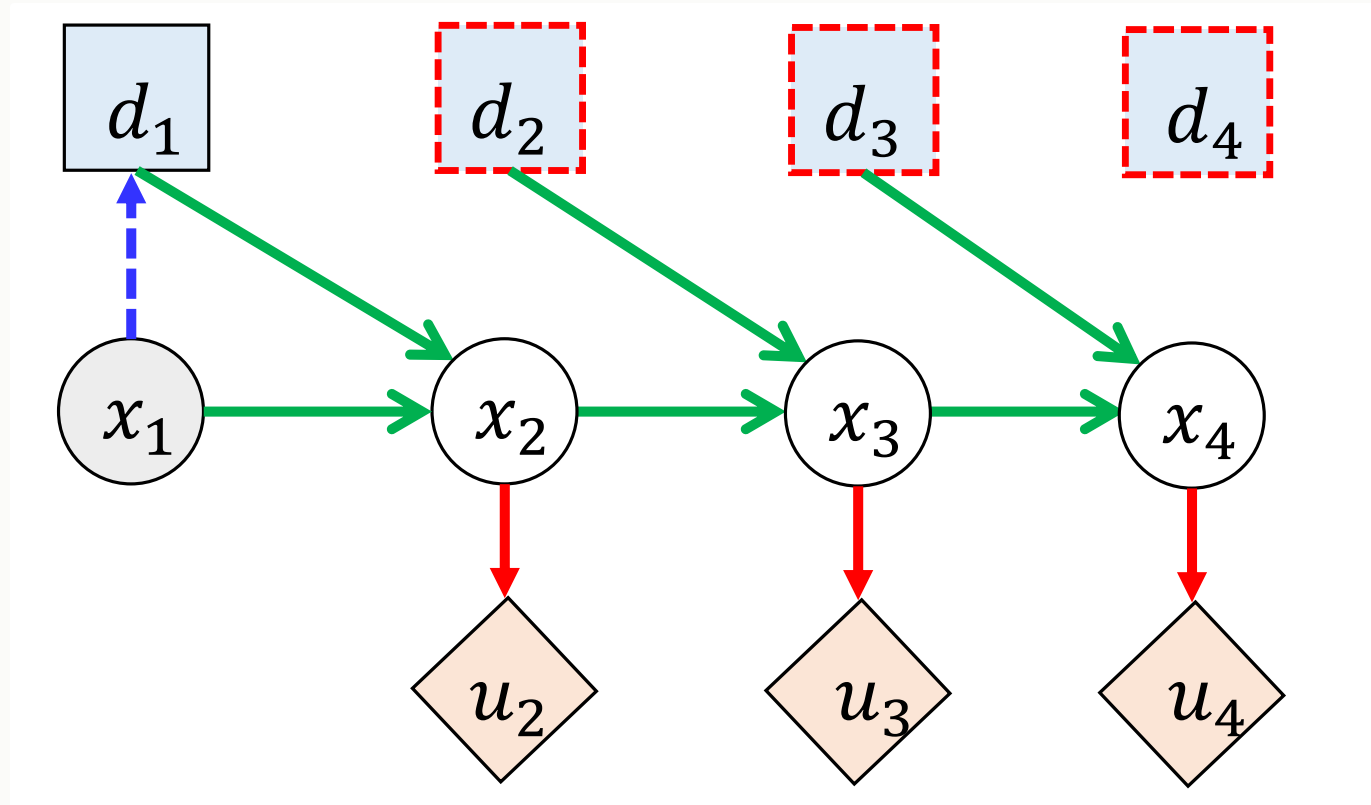
If healthy, run; if faulty, stop.

$$EU(\text{test then decide}) = 0.8(10) + 0.2(0) = 8.$$

The information is worth $8 - 2 = \mathbf{6}$ before paying for the test.

Information matters because it can change the action. A test costing 7 utility units is not worth buying here; a cost of 3 gives a net gain of 3.

Sequential decisions — average, observe, then choose again



Decisions above states, utilities below; later decisions are still pending.

At d_1 , average the uncertain x_2 . Once x_2 is known, choose d_2 . The order of sums and maxima follows this information sequence.

From an influence diagram to an MDP — state the extra assumptions

A sequential influence diagram records **what is known before each decision**. For additive utility, let h_t contain the observed history and define the best remaining value by

$$V_t(h_t) = \max_{a_t} \mathbb{E}[r_{t+1} + V_{t+1}(h_{t+1}) \mid h_t, a_t], \quad V_T = 0.$$

FULLY OBSERVED MARKOV STATE

If s_t contains all information relevant to the next transition and reward, replace the history with s_t . This gives the finite-horizon MDP recursion used in Lecture 7.

HIDDEN PHYSICAL STATE

An observation may not be a sufficient state. A belief $b_t(s) = P(s_t = s \mid h_t)$ can serve as the information state in a partially observed model. This is why filtering matters.

The common structure is **choose an action, average unknown outcomes, then choose again with the new information**. A general influence diagram does not become a fully observed MDP merely by adding a time index.

Check — the Bellman equation, four lectures early

An influence diagram adds decision and utility nodes to the network. Evaluating one alternates which two operations?

- ☐ A Maximising over chance nodes and summing over decision nodes
- ☐ B Summing over both, weighted by utility
- ☐ C Maximising over both, in topological order
- ☒ D Summing over chance nodes and maximising over decision nodes

You **average** over what you cannot control and **maximise** over what you can, in the order the information actually arrives. That alternating $\sum \max \sum \max$ is the structure of the Bellman equation, met here in a one-shot setting. Lecture 7 adds a Markov state, a transition model, and a return objective so that this pattern becomes an MDP recursion.

— CLOSING

Closing

Belief structured, and action attached. Two roads lead out of here, and they are the rest of the course.

Where we are — the graph, and the seed

MODEL-BASED		DATA-DRIVEN
Static, single	optimisation (<i>Lec 1 ✓</i>)	Bayesian statistics (<i>Lec 2 ✓</i>) → Bayesian networks (<i>Lec 3 ✓</i>) → Lec 4

We can now represent belief over a whole system as a graph, read its independences off the missing edges, answer questions within it by elimination or by sampling, and — newly — attach a decision and a value to it.

This lecture hands on [structured belief, plus decision and utility nodes — the influence diagram, a precursor to MDPs](#).

LECTURE 4 — BELIEF THAT ACTS

Put belief to work on an *unknown function*. A Gaussian process specifies consistent Gaussian distributions over function values. A Bayesian optimiser uses that uncertainty to choose measurements; EI and value-of-information criteria answer related but different questions.

LECTURE 7 — BELIEF THROUGH TIME

Unroll the decision network. One decision becomes a sequence, the utility node becomes a reward per step, and $\arg \max_a EU$ becomes [the Bellman optimality operator](#).

A joint distribution over a system is unmanageable
— until conditional independence draws it as a
graph.

Small local tables in place of one exponential one; sums that factor because the edges are missing; and then a decision node and a value node, which together are, in miniature, every decision problem still to come.

Questions?

Remember the decision network. It is the static, one-shot ancestor of the MDP, of dynamic programming, and of the reinforcement learner — belief, action and value, before time enters the picture.

— APPENDIX

Backup slides

The complete arguments, kept out of the narrative.

Reading guide — representation, inference, learning and decisions

one main idea to explain, one comparison, one application

ROLE	READ OR REVISIT	QUESTION TO ANSWER
Core	Koller & Friedman, <i>Probabilistic Graphical Models</i> (2009): selected sections on Bayesian networks and inference	What factorization and independence claims does a graph encode?
Compare	Discrete HMM filtering versus the linear Gaussian Kalman update	Which quantities change when a state becomes continuous?
Apply	The original alarm, aircraft and PhD-decision examples	Which question asks for a probability, a learned model, or an action?

The book is a reference for selected concepts. The core exercise is a small factorization and one predict–observe–update step; structure search and advanced time-series models are second-pass reading.

Backup 1— variable elimination, step by step

Query $P(B \mid d^1, c^1)$ on $p(B, S, E, D, C) = p(B)p(S)p(E \mid B, S)p(D \mid E)p(C \mid E)$. Write each conditional table as a factor over its variables:

$$T_1(B) T_2(S) T_3(E, B, S) T_4(D, E) T_5(C, E)$$

1. **Insert the evidence.** Fix $D = d^1$ and $C = c^1$, which reduces T_4 and T_5 to factors over E alone: $T_6(E) = T_4(d^1, E)$, $T_7(E) = T_5(c^1, E)$.
2. **Eliminate E .** Multiply every factor mentioning E and sum it out: $T_8(B, S) = \sum_e T_3(e, B, S) T_6(e) T_7(e)$.
3. **Eliminate S .** $T_9(B) = \sum_s T_2(s) T_8(B, s)$.
4. **Combine and normalise.** $P(B \mid d^1, c^1) \propto T_1(B) T_9(B)$.

Cost. Each elimination creates an intermediate factor whose size is 2^k where k is the number of variables it couples — the *induced width* of the chosen ordering. A good ordering keeps every factor small; a bad one can build a factor over most of the network. Finding the optimal ordering is NP-hard, so heuristics (min-fill, min-degree) are used. This is the practical ceiling of exact inference, and the reason sampling exists.

Backup 2 — approximate inference, and where each method breaks

METHOD	HOW IT WORKS	WHERE IT BREAKS
Direct (rejection) sampling	draw whole assignments in topological order; discard those disagreeing with the evidence	rare evidence throws away almost every sample
Likelihood weighting	clamp the evidence, weight each sample by $\prod P(x_i \text{pa}_i)$ over the clamped nodes	the <i>non</i> -evidence variables are still drawn from their priors, so a rare cause is never proposed
Gibbs sampling	redraw each variable from its conditional given all the others; discard a burn-in	correct in the limit, but can mix arbitrarily slowly

The likelihood-weighting counter-example. Take $C \rightarrow D$ with $p(c^1) = 0.001$, $p(d^1 | c^1) = 0.999$, $p(d^1 | c^0) = 0.001$, so that exactly $p(c^1 | d^1) = \frac{0.999 \times 0.001}{0.999 \times 0.001 + 0.001 \times 0.999} = 0.5$. A 1,000-draw run has probability $0.999^{1000} \approx 0.368$ of never proposing c^1 , in which case the estimate is zero. The normalised weighted estimator is generally biased at finite sample sizes but consistent under suitable support conditions; rare important samples can make convergence slow.

Backup 3 — from maximum expected utility to Bellman

One decision. Chance variables X with a Bayesian network, a decision D , a utility $U(D, X)$. Given evidence e ,

$$\text{EU}(d \mid e) = \sum_x p(x \mid d, e) U(d, x), \quad d^* = \arg \max_d \text{EU}(d \mid e), \quad \text{MEU}(e) = \max_d \text{EU}(d \mid e)$$

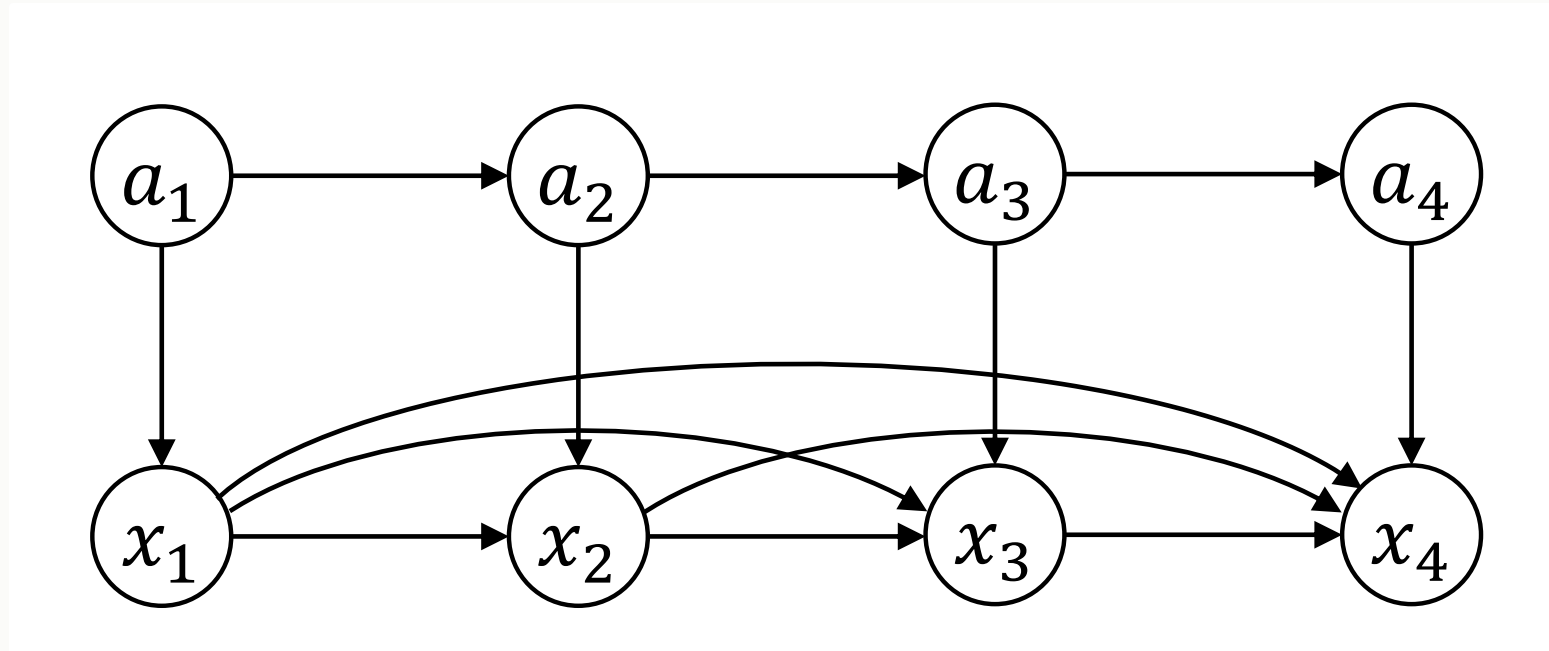
Many decisions. With the partial ordering $\mathcal{X}_0 \prec D_1 \prec \dots \prec D_n \prec \mathcal{X}_n$, the value of the first decision is

$$U(d_1 \mid x_0) = \sum_{\mathcal{X}_1} \max_{D_2} \dots \sum_{\mathcal{X}_{n-1}} \max_{D_n} \sum_{\mathcal{X}_n} \prod_{i \in \mathcal{L}} p(x_i \mid \text{pa}(x_i)) \sum_{j \in \mathcal{T}} U_j(\text{pa}(u_j))$$

with $\mathcal{L}$ the chance variables and $\mathcal{T}$ the utility variables.

The translation. Let each decision move a state, $p(x_{t+1} \mid x_t, d_t)$; let utility accumulate as a per-step reward, $\sum_t u(x_t)$. Then MEU becomes the value function V^* , the alternating $\sum \max$ becomes the dynamic-programming recursion, and $\arg \max_d \text{EU}$ becomes the Bellman optimality operator. Lecture 7 adds a discount factor γ and an infinite horizon; Lecture 8 removes p and estimates the expectation from samples. The object itself was built here.

Changing coefficients — a hidden process above the observations



The upper coefficient chain drives the lower autoregressive observations.

A changing coefficient is a hidden state to infer. It is different from fitting one fixed regression coefficient.

Backup — when the time-series model itself changes

Extensions to the AR model

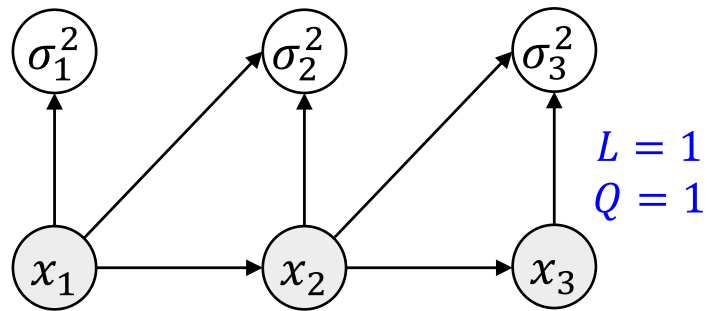
WHAT CHANGES?	A MODEL FOR THE CHANGE	INTERPRETATION
AR coefficients	$a_t = a_{t-1} + \eta_t, x_t = h_t^\top a_t + \epsilon_t$	infer slowly changing coefficients as hidden states
Innovation variance: ARCH	$\sigma_t^2 = \omega + \sum_i \alpha_i \epsilon_{t-i}^2$	recent large residuals predict more variability
Innovation variance: GARCH	$\sigma_t^2 = \omega + \sum_i \alpha_i \epsilon_{t-i}^2 + \sum_j \beta_j \sigma_{t-j}^2$	past variance also persists

Use $\omega > 0$ and nonnegative variance coefficients; stationarity requires additional parameter restrictions. A changing coefficient, a changing state, and changing observation noise are **different modelling choices**.

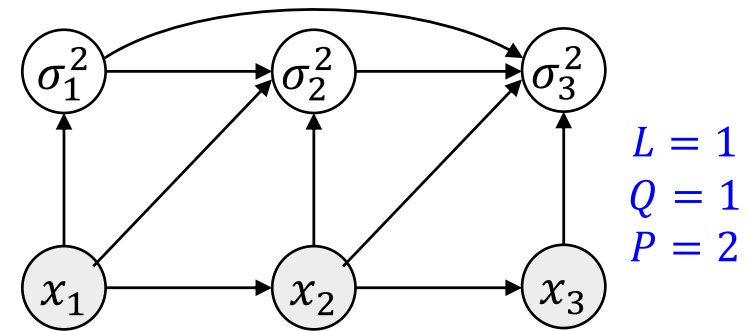
These are extensions of the same graphical model. Bayesian updating estimates the quantities the model allows to vary; it does not make a fixed model automatically adapt to every kind of change.

Changing variance — ARCH and GARCH side by side

Auto Regressive Conditional Heteroscedasticity (ARCH)



Generalized ARCH model (GARCH)



Variance is modelled explicitly in the upper row.

ARCH uses past squared residuals. GARCH also carries past conditional variance forward.

— EXTENSION

Extensions — model classes and structure learning

Read after completing the main route.

Hybrid networks — a table is not the only local model

Aircraft example

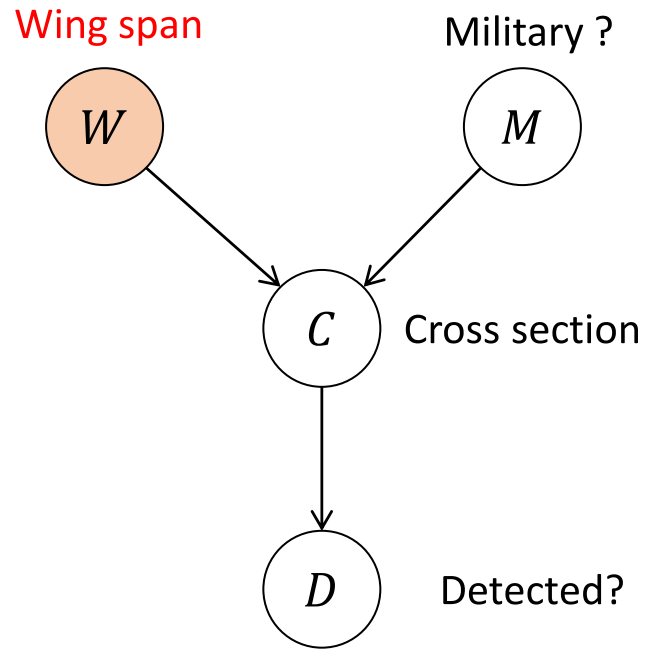
The source network mixes **wing span** W (continuous), **military type** M (binary), **radar cross section** C (continuous), and **detection** D (binary).

$$p(w, m, c, d) = p(w) p(m) p(c \mid w, m) p(d \mid c).$$

NODE	A POSSIBLE LOCAL MODEL	MEANING
W	$\mathcal{N}(\mu_W, \sigma_W^2)$	distribution of wing spans
M	Bernoulli probability θ	frequency of the aircraft type
$C \mid W, M = m$	$\mathcal{N}(a_m W + b_m, \sigma_m^2)$	each type has its own regression
$D \mid C$	$P(D = 1 \mid C) = 1/(1 + e^{-(C-c_0)/b}), b > 0$	detection becomes more likely as the cross section grows

The **graph factorisation stays the same**. Sum discrete hidden variables and integrate continuous ones. Non-Gaussian factors can require approximate inference.

A continuous root — wing span



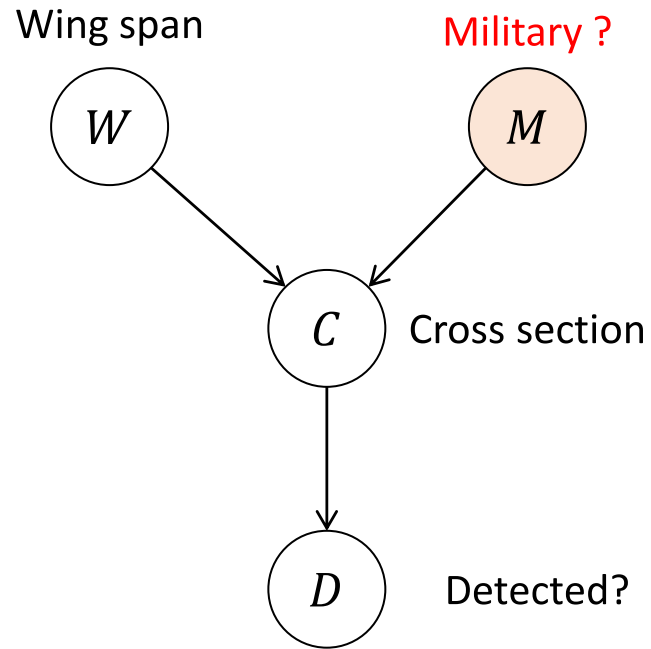
Wing span is a continuous variable and modeled as a Gaussian distribution

$$P(w) = N(w|\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{w-\mu}{\sigma}\right)^2}$$

The graph stays fixed while the highlighted local model changes.

The highlighted root W has a Gaussian density. It is a continuous variable, so its local model is not a finite table.

A discrete root — aircraft type



Whether a target is a military vehicle can be modeled with a single parameter θ

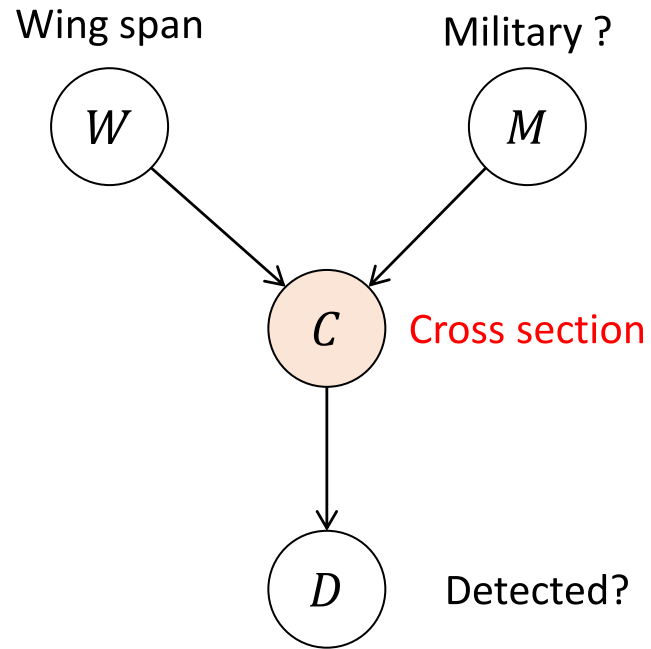
$$P(m^1) = \theta$$

$$P(m^0) = 1 - \theta$$

The graph stays fixed while the highlighted local model changes.

The highlighted root *M* is binary, so one Bernoulli parameter specifies its local distribution.

A conditional Gaussian — radar cross section



Radar cross section can be modeled as a conditional Gaussian

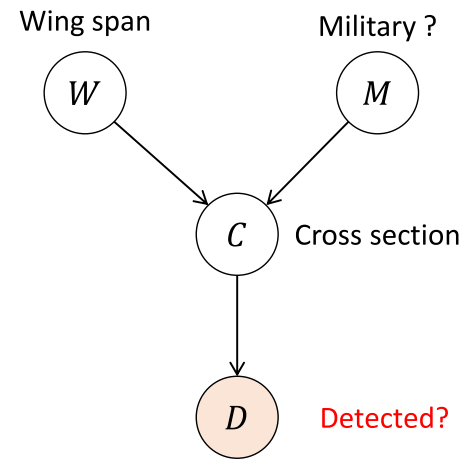
$$P(c|w, m) = \begin{cases} N(c|a_0w + b_0, \sigma_0^2) & \text{if } m = m^0 \\ N(c|a_1w + b_1, \sigma_1^2) & \text{if } m = m^1 \end{cases}$$

(Conditional linear Gaussian)

The graph stays fixed while the highlighted local model changes.

The highlighted *C* depends on wing span and type. Each type selects its own linear Gaussian regression.

A binary observation — detection probability

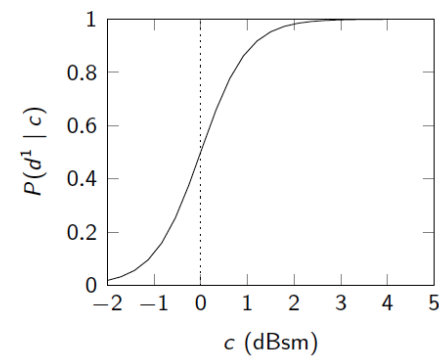


- Logit model:

$$P(d^1|c) = \frac{1}{1 + \exp\left(-2\frac{c - \alpha}{\beta}\right)}$$

- Probit model:

$$P(d^1|c) = \Phi\left(\frac{c - \alpha}{\beta}\right)$$



The graph stays fixed while the highlighted local model changes.

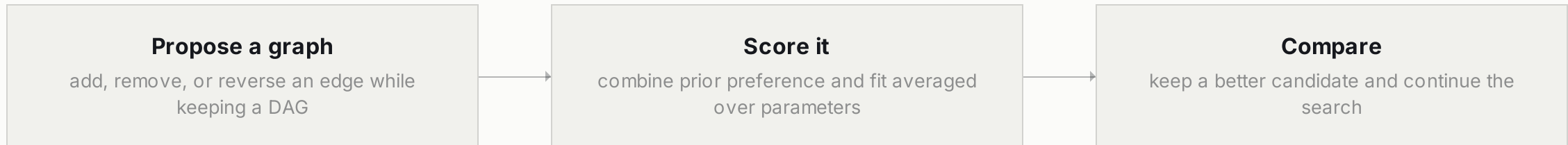
The highlighted *D* is binary. Its probability changes smoothly with continuous radar cross section *C*.

Learning the graph — compare explanations, not just fitted tables

Tructure learning

If the arrows are unknown, the candidate model is the graph G as well as its parameters θ :

$$P(G \mid D) \propto P(G) \underbrace{\int p(D \mid \theta, G) p(\theta \mid G) d\theta}_{p(D|G), \text{ the model evidence}}.$$

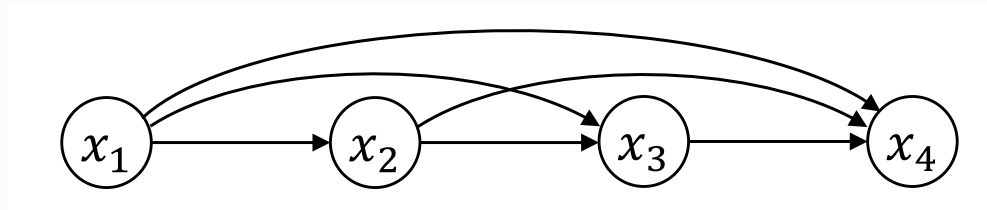


Exhaustively checking every DAG is usually impractical. Search can stop at a local solution, and different graphs may encode the same observational independences.

Inference: unknown variables in a fixed model. **Parameter learning:** unknown tables. **Structure learning:** unknown arrows. An observational graph alone does not establish causality.

Continuous time-series data — regression becomes a transition model

Time is discrete; the state is continuous



Lagged observations point to the next value.

An AR(L) model uses the last L values as regression features:

$$x_t = a^\top h_t + \epsilon_t, \quad h_t = [x_{t-1}, \dots, x_{t-L}]^\top, \quad \epsilon_t \sim \mathcal{N}(0, \sigma^2).$$

Lecture 2's regression now specifies a **transition model**. Gaussian MLE still minimises $\sum_t (x_t - a^\top h_t)^2$.

AR(2) is generally not first-order Markov in x_t . It is first-order in $[x_t, x_{t-1}]$. **Choose a state that keeps the information needed for prediction.**