Probabilistic Graphical Models (Koller - Stanford)

PROBABILISTIC GRAPHICAL MODEL

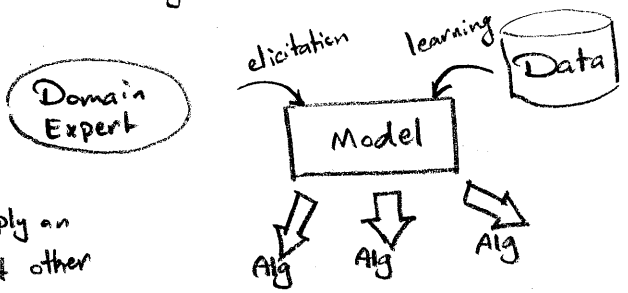
- First: Medical Diagnosis → predisposing factors, symptoms, test results, diseases, treatment outcome

  Image Segmentation → pixels ⇒ superpixels ⇒ label

- In common: Large # of Factors

  Huge amount of uncertainty about results

- Model: Declarative Representation of our understanding from the world
  - what variables are
  - How vars interact with each other



  - Declarative → standalone
    - ↳ no matter what algorithm apply on
    - ↳ we can understand it without other knowlege
  - Algs can ask different questions or the same question with different accuracy-complexity trade off

- Uncertainty
  - Partial knowledge of state of the world
  - Noisy observations
  - Phenomena not covered by our own model → modeling limitation
  - Inherent stochastic → micro level and sometimes even macro level

- Probability Theory
  - Declarative representation with clear semantics
  - Powerful reasoning patterns → Toolbox ☑ Conditioning, reasoning, decision making
  - Establish learning methods

- Complex Systems
  - Random variables → $X_1, \ldots, X_n$ (one facet of the world) → simplest case: all binary
  - Distribution over R.V. → simplest case 2^n states → intractable
  - Goal: to capture the uncertainty in form of Joint Prob → to exploit the structure in distribution and represent and manipulate in effective way

- Graphical Model
  - Baysian Networks → directed graph → nodes: random variables; edges: probabilistic connection between r.v.
    - ☑ CPCS
  - Markov Network → undirected graph ☑ Image segmentation

- Graphical Representation
  - Intuitive & Compact data structure for capturing high dimensional distribution
  - Efficient reasoning using general-purpose algorithms → exploiting graphical structure
  - Sparse parameterization → feasible elicitation ← by hand; learning from data ← automatic ⇒ Reduction in the number of params.

☑ Image Segmentation

Pixels ⟶ Superpixels ⟶ Labled : Machine Learning

Many Applications
Lecture / 17

Medical Care

Textual Information Extraction

Multi Sensor Integration

Sensors ⟶ Learned Models ⟶ Current & Future Data

Biologic Network Reconstruction

- Roadplan
  - Representation
    - ↳ Directed and Undirected
    - ↳ Temporal and Plate model ⟶ high level & complex scenarios
  - Inference (= Resoning)
    - ↳ Exact & Approximate
    - ↳ Decision Making under Uncertainty
  - Learning
    - ↳ Parameters and Structure
    - ↳ With and without complete data

* Joint Distribution

P(I, D, G)

☑ Student Example

- I : Intelligence → i^0 : low, i^1 : high
- D : Difficulty → d^0 : easy, d^1 : hard
- G : Grade → g^1 : A, g^2 : B, g^3 : C

$2 \times 2 \times 3 = 12$ possible combination
↳ free parameters

I	D	G	Prob
i^0	d^0	g^1	0.126
i^0	d^0	g^2	0.168
i^0	d^0	g^3	0.126
i^0	d^1	g^1	0.009
i^0	d^1	g^2	0.045
i^0	d^1	g^3	0.126
i^1	d^0	g^1	0.252
i^1	d^0	g^2	0.0224
i^1	d^0	g^3	0.0056
i^1	d^1	g^1	0.06
i^1	d^1	g^2	0.036
i^1	d^1	g^3	0.024
			sum = 1

- Independent variables : vars whose values are not determined by the values of other parameters
  - ↳here: $\sum_i P_i = 1$ ⟶ if we know 11 of the values, the 12th is obvious ⟶ 11

• Conditioning: assume that student got an A

$A \rightsquigarrow g^1$ ⟶ eliminates all rows of table that has $G \neq g^1$ ⟶ reduced prob. distribution

↳ not consistent with the observation

This operation is called reduction: reduce away stuff that is not consistent with observation

↳ doesn't give a probability distribution

$\sum P_i(I, D, g^1) \neq 1$

↳ $P(I, D, g^1)$ is unnormalized measure

↳ Renormalization needed.

↳ divide each of them by 0.447

↳ $P(I, D \mid g^1)$

I	D	G	P(I,D,G)
i^0	d^0	g^1	0.126
i^0	d^1	g^1	0.009
i^1	d^0	g^1	0.252
i^1	d^1	g^1	0.06
			0.447

⇓ divide by 0.447

I	D	P(I,D|g¹)
i^0	d^0	0.282
i^0	d^1	0.02
i^1	d^0	0.564
i^1	d^1	0.134
		1

• Marginalization: operation that takes the probability dist of a larger subset of variables and produces a prob. dist over a subset of these

to throw away I from P(I,D) we should sum up over I.

⇓ sum up over I

D	P(D)
d^0	0.846
d^1	0.154
	1

* Factors

• A function or a table

input: random variable ⟶ output: related to assignments of r.v. a real value

↳ accepts all possible assignment in the cross product space of r.v.

↳ all possible combinations

• $\phi : Val(X_1, \dots, X_k) \longrightarrow \mathbb{R}$

↳ $Scope(\phi) = \{X_1, \dots, X_k\}$

☑ A joint distribution is a factor. ⟶ but factors doesn't require $\sum P_i = 1$

an unnormalized measure is a factor. ⟶ $P(I, D, g^1) \rightsquigarrow scope(\phi) = \{I, D\}$

conditional probability distribution

• CPD : Conditional Probability Distribution

– a factor

☑ P(G | I, D)

* for every combination of values of I & D we have a probability dist. over G.
* each row sums to 1 → distribution ove G
  ↳ context

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

- General Factor

☑ Scope = {A, B}

A	B	ϕ
a^0	b^0	30
a^0	b^1	5
a^1	b^0	1
a^1	b^1	10

- Operations on Factors

  - Factor Product

$\phi_1(A,B) \times \phi_2(B,C)$

scope = $\{A,B\} \cap \{B,C\} = \{A,B,C\}$

$\phi_3(A,B,C)$

A	B	$\phi_1(A,B)$
a^1	b^1	0.5
a^1	b^2	0.8
a^2	b^1	0.1
...		

B	C	$\phi_2(B,C)$
b^1	c^1	0.5
b^1	c^2	0.7
b^2	c^1	0.1
...		

$T_1(\text{row } 1) \times T_2(\text{row } 1) \leftarrow a^1, b^1 \,\&\, b^1, c^1$

$T_1(\text{row } 1) \times T_2(\text{row } 2) \leftarrow a^1, b^1 \,\&\, b^1, c^2$

A	B	C	$\phi_3(A,B,C)$
a^1	b^1	c^1	0.5 × 0.5 = 0.75
a^1	b^1	c^2	0.5 × 0.7 = 0.35
a^1	b^2	c^1	0.8 × 0.1 = 0.08
...			

  - Factor Marginalization

$\phi_3(A,B,C) \longrightarrow \phi_4(A,C)$

find different values of B and sum them up

⇓

A	C	$\phi_4(A,C)$
a^1	c^1	0.75 + 0.08 = 0.33
...		

  - Factor Reduction

focus on context c^1
look for row with values $C = c^1$

$\phi_3(A,B,C) \longrightarrow \phi_5(A,B)$

A	B	C	$\phi_5(A,B)$
a^1	b^1	c^1	0.75
a^1	b^2	c^1	0.08
a^2	b^1	c^1	0.05

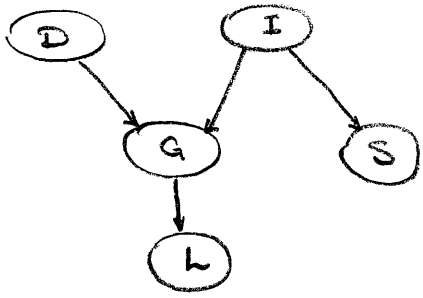
- Why factors?
  - Fundamental building blocks for defining distributions in high-dimensional spaces
  - Set of basic operations for manipulating these probability distributions

* Bayesian Networks

☑ The student example

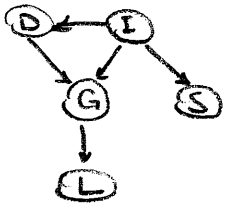
- Grade
- Difficulty of Course
- Intelligence of Student
- SAT
- Letter of Reference

$P(G, D, I, S, L)$



this is not the only model possible.

for example smarter student may go for harder classes.



- Each node have a CPD

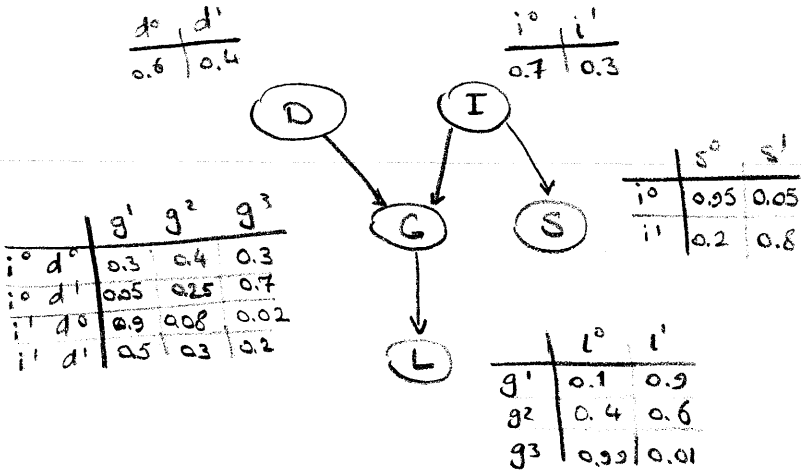
$P(D)$

$P(I)$

$P(G|I,D)$

$P(S|I)$

$P(L|G)$



- Chain Rule for BN

simply take all different CPDs in BN and multiply the together.

$$P(G, I, D, S, L) = P(D) \cdot P(I) \cdot P(G|I,D) \cdot P(S|I) \cdot P(L|G)$$

five factors with overlapping scopes $\Longrightarrow$ big factor with scope of all variables

☑ $P(d^0, i^1, g^3, s^1, l^1) = P(d^0) \times P(i^1) \times P(g^3|d^0, i^1) \times P(l^1|g^3) \times P(s^1|i^1)$

$= 0.6 \times 0.3 \times 0.02 \times 0.01 \times 0.8$

- Definition:

- BN is a directed acyclic graph (↳ DAG) G whose nodes represent the random variables ($\hookrightarrow X_1, \dots, X_n$)

- For each node X_i there is a CPD $P(X_i \mid Pa_G(X_i))$, where Pa_G ↳ parents in graph G

- BN represents a joint dist. via the chain rule for BN

$$P(X_1, \dots, X_n) = \prod_i P(X_i \mid Pa_G(X_i))$$

- Proof of BN is a joint distribution

1- BN is a legal distribution: $P \geq 0$

P is multiply of CPDs, CPDs are non-negative $\Rightarrow P \geq 0$ ✓

2- BN is legal distribution : $\sum P = 1$

$$\sum_{D,I,G,S,L} P(D,I,G,S,L) = \sum_{D,I,G,S,L} P(D)\,P(I)\,P(G|I,D)\,P(S|I)\,P(L|G)$$

chain rule

$$= \sum_{D,I,G,S} P(D)\,P(I)\,P(G|I,D)\,P(S|I) \underbrace{\sum_L P(L|G)}_{1}$$

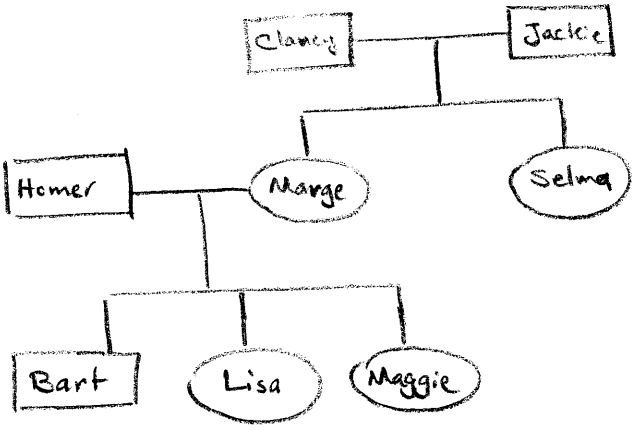
push summation inside

$$= \sum_{D,I,G} P(D)\,P(I)\,P(G|I,D) \underbrace{\sum_S P(S|I)}_{1} = \dots = 1 \checkmark$$

- P Factorizes over G — BN
  - We can represent it over graph G if we can encode it using the chain rule for
  - G: Graph over $X_1,\dots,X_n$ $\Longleftrightarrow$ P factorizes over G if

$$P(X_1,\dots,X_n) = \Pi_i P(X_i | Par_G(X_i))$$

☑ The first BN evar! ⇒ Genetic Inheritance



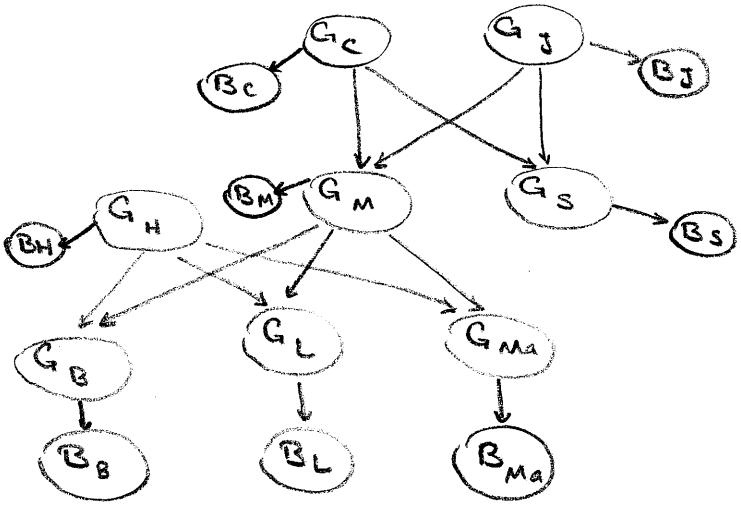
Chromosomes : A , B , O

↓ each person have two chromosomes

Genotype : AA , AB , AO , BO, BB , OO

↓ AA and AO show the same effect

Phenotype : A , B , AB , O



*Reasoning Patterns

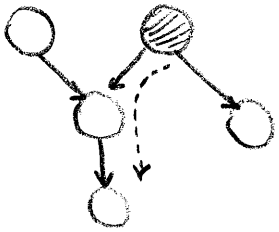
☑ Student Network

- Create graph ⟶ Joint dist. Using Chain Rule ⟶ Marginalization
- ☑ $P(i^1) \approx 0.5$
- Condition on the Parents of Query: Causal Reasoning

  Joint Dist ⟶ Reduction on Evidence ⟶ Marginalization

☑ $P(l^1 \mid i^0) \approx 0.39$

↳ causal reasoning: reason from top to down

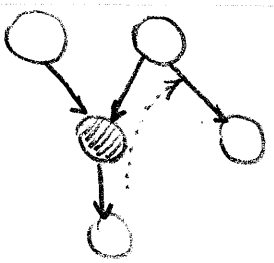


$P(l^1 \mid i^0, d^0) \approx 0.51$

- Condition on the children of Query: Evidential Reasoning

☑ $P(d^1) = 0.4 \longrightarrow P(d^1 \mid g^3) \approx 0.63$

$P(i^1) = 0.3 \longrightarrow P(i^1 \mid g^3) \approx 0.08$

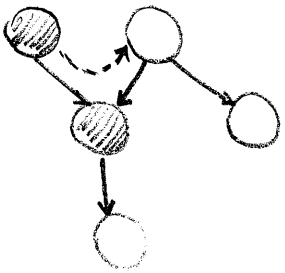


if the student get C there is more prob. that course is harder or student is less intelligent.

- Intercausal Reasoning: explain away effect

☑ class is hard, student get a C (or a B)

↳ g^3 ↳ g^2



$P(i^1) = 0.3$

$P(i^1 \mid g^3) \approx 0.08$ ⟩ increase again but not too much because

$P(i^1 \mid g^3, d^1) \approx 0.11$ $P(g^3 \mid d^1)$ is still down

---

$P(i^1) = 0.3$

$P(i^1 \mid g^2) = 0.175$ ⟩ increase even more than $P(i^1)$

$P(i^1 \mid g^2, d^1) \approx 0.34$

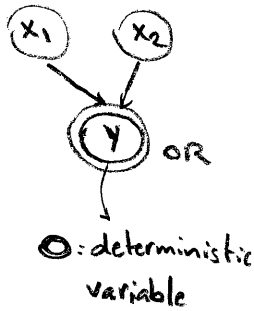
– explaination

☑ Value of Y is completely determined with X_1 and X_2 so other rows have 0 probability. we observe $Y = 1$:

$P(X_1 = 1) = \frac{2}{3}$ $P(X_2 = 1) = \frac{2}{3}$

$P(X_1 = 1 \mid X_2 = 1) = 0.5$

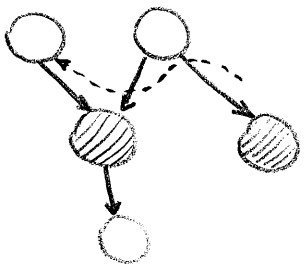
	X_1	X_2	Y	Prob
$Y=1$ ←	0	0	0	0.25
	0	1	1	0.25
	1	0	1	0.25
	1	1	1	0.25



⊙: deterministic variable

If I know $Y=1$ the chance of X_1 and X_2 is 66%
If I further know $X_2=1$ so that explains why $Y=1$ and it doesn't matter if X_1 is equal to 1 or not. It is explained away.

☑ Student example : Student aces the SAT but get C grade.
so we may say that the class is difficult. because SAT explained away that student is not intelligent.



$P(d^1) = 0.4$ $P(i^1) = 0.3$

$P(d^1 | g^3) \approx 0.63$ $P(i^1 | g^3) \approx 0.08$

$P(d^1 | g^3, s^1) \approx 0.73$ $P(i^1 | g^3, s^1) \approx 0.58$ → increase dramatically

* Flow of Probabilistic Influence

- The question is when can X influence Y? → Influence: condition on X changes belief on Y

- $X \rightarrow Y$ ✓ : connected , X is parent of Y
- $X \leftarrow Y$ ✓ : connected , Y is child of X
- $X \rightarrow W \rightarrow Y$ ✓ : causal chain
- $X \leftarrow W \leftarrow Y$ ✓ : evidential chain
- $X \leftarrow W \rightarrow Y$ ✓
- $X \rightarrow W \leftarrow Y$ X : v-structure

In general probabilistic influence is symmetrical. If X can influence Y then Y can influence X.

☑ If I say the student took a difficult class , does that tell me anything about the student intelligence?

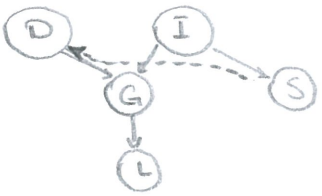
- Active trail with no evidence
  - Trail : a sequence of nodes that are connected (→ undirected) to each other by single edges in graph
  - Influence can flow from one variable to another except in v-structure → this flow can continue
  - A trail $X_1 - \cdots - X_k$ is active if it has no block (→ v-structure $X_{i-1} \rightarrow X_i \leftarrow X_{i+1}$)

- The question is when can X influence Y given evidence about $\mathcal{Z}$?
  - In general if we have an edge $X \rightarrow Y$ in some BN, there is there any set of other variables that we can condition on to make X and Y independent? NO!

- $X \rightarrow Y$ ✓
- $X \leftarrow Y$ ✓
- $X \rightarrow W \rightarrow Y$
- $X \leftarrow W \leftarrow Y$
- $X \leftarrow W \rightarrow Y$
- $X \rightarrow W \leftarrow Y$

$W \notin \mathcal{Z}$	$W \in \mathcal{Z}$	
✓	X	I know W, so Y only depend on W
✓	X	symmetry
✓	X	the root is already known, so observation can't change that
X	✓	→ explain away, but this row holds only if W and all of its decendents not in $\mathcal{Z}$

☑ S-I-D-G allow influence



I observed ✗

I not observed, nothing else ✗

I not observed, G observed ✓

- Active trail $X_1 - \cdots - X_K$ given Z
  - the v-structure is activated → node or one of descendents is observed
    - v-structure ↳ $X_{i-1} \rightarrow X_i \leftarrow X_{i+1}$
    - node ↳ X_i
    - observed ↳ $\in Z$
  - no other X_i in Z
    - ↳ not in the middle of v-structure $X_{i-1} \rightarrow X_i \leftarrow X_{i+1}$

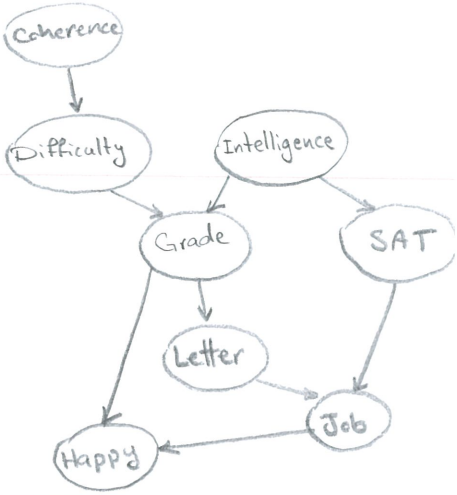
☑ Active if we observe G

$C \rightarrow D \rightarrow G \leftarrow I \leftarrow S$ ✓

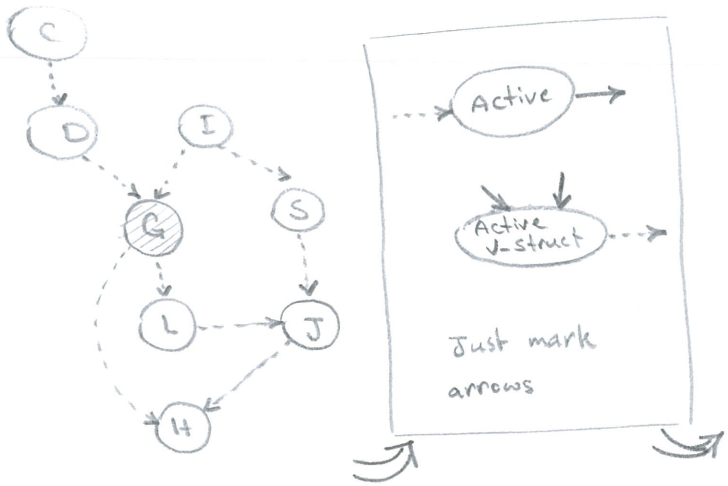
$I \rightarrow G \rightarrow L \rightarrow J \rightarrow H$ ✗

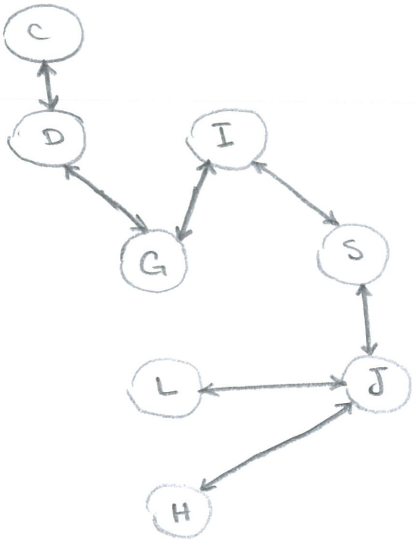
$I \rightarrow S \rightarrow J \rightarrow H$ ✓

$C \rightarrow D \rightarrow G \leftarrow I \rightarrow S \rightarrow J \leftarrow L$ ✗



* my idea: re-drawing the graph





bad idea! have problems with last example

* Independencies

- So far: graphs to encode prob. dist.

  complementary views: representation of a set of indepencies that a dist. must satisfy

- Independence in probability distribution

  - for events α and β, $P \models \alpha \perp \beta$ (read P satisfies α is independent of beta) if:
    - ↳ $P(\alpha, \beta) = P(\alpha) \times P(\beta)$ → conjunction
    - ↳ $P(\alpha \mid \beta) = P(\alpha)$ → flow of information, β doesn't effect prob in α
    - ↳ $P(\beta \mid \alpha) = P(\beta)$ → information flow is symmetrical

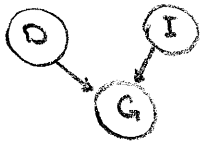
- for random variables X, Y, $P \models X \perp Y$ if:
  - $P(X,Y) = P(X)\,P(Y)$
  - $P(X|Y) = P(X)$
  - $P(Y|X) = P(Y)$

{ universal statement: $\forall x, y \quad P(x,y) = P(x) \times P(y)$

{ factor interpretation: factor $\Phi_3(X,Y)$ is the product of two lower dimension factors $\Phi_1(X)$ and $\Phi_2(Y)$

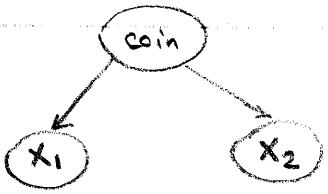
☑ $P(I,D) = P(I) \cdot P(D)$

↳ $\sum_G P(G, I, D)$



- Conditional Independence
  - for (set of) random variables X, Y, Z, $P \models (X \perp Y | Z)$ if:
    ↳ P satisfies X is independent of Y given Z
    - $P(X,Y|Z) = P(X|Z)\,P(Y|Z)$
    - $P(X|Y,Z) = P(X|Z)$ → given Z, Y gives me no additional information about X
    - $P(Y|X,Z) = P(Y|Z)$
    - $P(X,Y,Z) \propto \Phi_1(X,Z)\,\Phi_2(Y,Z)$

☑ two coins, one is fair, one had 90% of heads
we pick a coin and toss it twice



If the first toss is head → the prob of 2nd toss to be H rises!
If we are told that we pick the fair coin → no info about 2nd toss
If we are told that we pick the biased coin → (anymore!) again no info about 2nd toss!

So: X_1 and X_2 are not independent: $P \not\models X_1 \perp X_2$
but $P \models X_1 \perp X_2 | C$

☑ $P(S, G | i^0) = P(S|i^0) \cdot P(G|i^0)$
do we still need to check $P(S, G|i^1) = P(S|i^1) \cdot P(G|i^1)$ to say $P \models S \perp G | I$?
YES! we need to check for all values of I.

- Conditioning can lose independence!

☑ $P \models I \perp D$

$P \not\models I \perp D | g^1$

* Independence in BN
- Why independence and factorization are related to each other?
  - $P(X,Y) = P(X)\,P(Y)$ → definition of independence
    → factorization over two factors

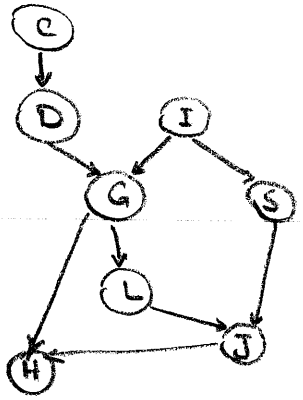
- $P(X,Y,Z) \propto \phi_1(X,Z)\,\phi_2(Y,Z) \iff (X \perp Y \mid Z)$
- factorization of a distribution P implies independencies that hold in P
- if P factorizes over G, can we read these independencies from the structure of G?

• Flow of Influence and d-Seperation

- what happens when there is no active trails in graph?
  - ↳ influence can't flow → notion of d-separation

- X and Y are d-separated in G given Z if there is no active trail in G between X and Y given Z

  ↳ $d\text{-sep}_G(X, Y \mid Z)$



☑ d-sep (D, I | L) → v-structure in G active

d-sep (D, J | L) → v-structure in G active

d-sep (D, J | L, I) ✓

d-sep (D, J | L, H, I) → v-structure in H active

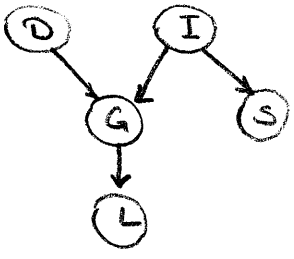
— Proof of (BN: Factorization ⇒ independence)

Theorem: If P factorizes over G, and $d\text{-sep}_G(X, Y \mid Z)$ then P satisfies $(X \perp Y \mid Z)$

↳ no active trail from X to Y given Z

☑ $D \perp S$

$$P(D, I, G, S, L) = \underbrace{P(D)\,P(I)\,P(G \mid D, I)\,P(S \mid I)\,P(L \mid G)}_{\text{chain rule for BN}}$$



$$P(D, S) = \underbrace{\sum_{G,L,I}}_{\text{push in summation}} P(D)\,P(I)\,P(G \mid D, I)\,P(S \mid I)\,P(L \mid G)$$

$$= \sum_I P(D)\,P(I)\,P(S \mid I) \underbrace{\left(\sum_G \left(P(G \mid D, I) \underbrace{\sum_L P(L \mid G)}_{1} \right) \right)}_{1}$$

$$= P(D) \underbrace{\left(\sum_I \underbrace{P(I)\,P(S \mid I)}_{P(S, I)} \right)}_{P(S)} = \underbrace{P(D)}_{\phi_1(D)}\,\underbrace{P(S)}_{\phi_2(S)} \Rightarrow P \models D \perp S \checkmark$$

- Any node is d-separated from its non-descendents given its parents

☑ L → descendants : J, H

→ non-d : C, D, S, I

→ parent : G → observed

from S to J → trail blocked

from S to H →

If P factorizes over G, then in P, any variable is independent of its non-descendants given its parents

- I-maps → Independency Map

d-seperation ⇒ P satisfies corresponding independence statement

$$I(G) = \{ (X \perp Y \mid Z) : \text{d-sep}_G(X, Y \mid Z) \}$$

all of the independence statements — correspond to d-seperation statements in graph

The set of all independencie that are derived from these seperation statements and those are the ones that graph implies hold for P.

- If P satisfies I(G) we say that G is an I-map of P

☑ P_1:

I	D	Prob
i^0	d^0	0.42
i^0	d^1	0.18
i^1	d^0	0.28
i^1	d^1	0.12

P_2:

I	D	Prob
i^0	d^0	0.282
i^0	d^1	0.02
i^1	d^0	0.564
i^1	d^1	0.134

G_1: 

$I(G_1) = \{ D \perp I \}$

$P_1 \not\models D \perp I \Rightarrow G_1 = \text{I-map}(P_1)$ ✓

$P_2 \not\models D \perp I \Rightarrow G_1 \neq \text{Imap}(P_2)$ ✗

G_2: 

$I(G_2) = \{\}$

$P_2 \Rightarrow G_2 = \text{I-map}(P_2)$ ✓

G_2 has no independence assumption

↳ both P_1 and P_2 satisfy the empty set

↳ P_1 also satisfies additional indep. assumpti
but I-map don't require that the graph
exactly represents the independencies

$P_1 \Rightarrow G_2 = \text{I-map}(P_1)$ ✓

- Proof of (BN: Factorization ⇒ Independence)

Theorem: If P factorizes over G, then G is an I-map for P

= I can read from G, independencies in P regardless of parameters

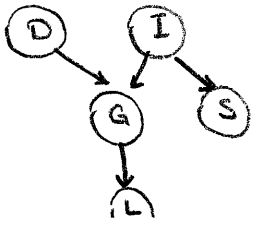
Converse: (BN: Independence ⇒ Factorization)

Theorem: If G is an I-map for P, then P factorizes over G

= If the dist. satisfies the independence assumption that are implicit in graph, w
can take that dist and represent it as a BN over the graph

☑ $P(D, I, G, S, L) = \underbrace{P(D)P(I \mid D)P(G \mid D, I)P(S \mid D, I, G)P(L \mid D, I, G, S)}_{\text{chain rule for probability}}$

↳ ... still don't know if it is BN



$P(I|D) \xrightarrow{I \perp D} P(I)$

$P(S|D,I,G) \xrightarrow{\text{last theorem}} P(S|I)$

$P(L|D,I,G,S) \xrightarrow{\text{last theorem}} P(L|G)$

last theorem : remove non-descendents

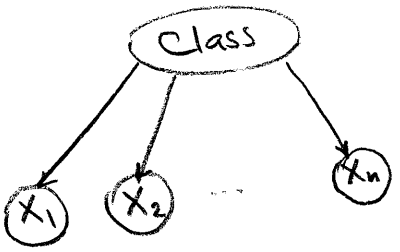
$\Rightarrow P(D,I,G,S,L) = P(D)\,P(I)\,P(G|D,I)\,P(S|I)\,P(L|G)$ ✓

- Two equivalent view of graph structure :
  - ↳ Factorization : G allow P to be represented
    - ↳ a data structure that show us how dist can be broken up to pieces (factors = CPDs)
  - ↳ I-map : Independencies encoded by G hold in P
    - ↳ structure in G encod independencies that doesn't necessarily hold in P.

- If P factorizes over a graph G we can read from the graph independencies that must hold in P (and I-map)
- Indepency is valuable → tells sth about what influence what → structure
  - ↘ if I observe sth what else might change → observation

✗ Naïve Bayes → idiot bayes

- Reason of naming: because the model make independence assumptions that indeed are very naïve and overly simplistic.
- This model is typically used for classification
  - _Given : features
  - _Infer : class



- If we observe the class variable, influence cannot flow between any of the other variables, because the only path between two other variables is through the class variable. If we dont observe the class variable, however, all the variables are dependent.

$(X_i \perp X_j \mid C)$ for all X_i, X_j

$$P(C, X_1, \dots, X_n) = P(C) \prod_{i=1}^{n} P(X_i|C)$$

- Better understanding : looking at ratio of probabilities of two different classes given the particular observation

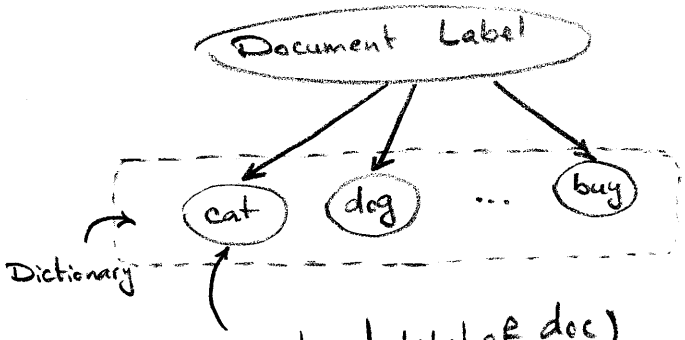
$$\frac{P(C=c_1 \mid x_1, \dots, x_n)}{P(C=c_2 \mid x_1, \dots, x_n)} = \underbrace{\frac{P(C=c^1)}{P(C=c^2)}}_{\text{odds of priors}} \times \underbrace{\prod_{i=1}^{n} \frac{P(x_i|C=c^1)}{P(x_i|C=c^2)}}_{\text{odds ratios}}$$

☑ Text classification & Categorization

1- Bernoulli Naïve Bayes for Text

- It treats every word in the dictionary as r.v.
- For each word we have a binary variable
  - ↳ 1 : word appears in a document
  - ↳ 0 : otherwise

5000 words in dict ⟹ 5000 binary var.



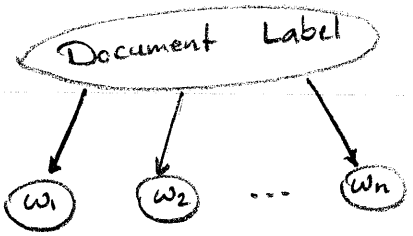
- CPD associated to each r.v. ⟶ P("cat" appears in doc | label of doc)

–	cat	dog	buy	sell
financial	0.001	0.001	0.2	0.3
pets	0.3	0.4	0.02	0.0001

Bernoulli ⟶ binary

Naïve bayes ⟶ strong indepence assumption

2- Multinomial Naïve Bayes for Text



- features are words in the document
- n: lenght of document ⟶ number of r.v.
- values of r.v. are actual words ⟶ if there are D words in dictionary cardinality of each r.v. is D
- assuming that prob. dist over the words is the same for every position
- CPD associated to all random variables

	cat	dog	buy	sell
financial	0.001	0.001	0.2	0.3
pet	0.3	0.4	0.02	0.0001

→ multinomial } all of the entries sum to one.

- Naïve ⟶ strong independence assumption ⟶ word in position i is independent of word in pos j
  - ↳ don't support two word phrases

- Simple Approach ⟶ Computationally efficient
  - ↘ Easy to construct
- Surprisingly efficient in domains with many weakly relevant features
- Strong independence assumption reduce performance when many features are strongly correlated

☑ Medical Diagnosis ⟶ Path Finder (92) ⟶ rule based system (I)
- ↘ naive bayes (II)
- ↘ naive bayes + knowledge engineering (III) ⟶ not to put in 0 probabilities

* the product of any number and 0 is 0, once a zero prob. is (wrongly) assigned to an event that is actually possible in reality, nothing can change it again. It is difficult to be sure which event had real 0 prob. and whi have really small prob. Wrongly assign 0 is dangerous, but small value for impossible events is safer.

★ better calibration of conditional prob. → P(finding | disease$_1$) to P(finding | disease$_2$) ✓

P(finding$_1$ | disease) to P(finding$_2$ | disease) X

↘ full BN (IV)

↘ removed incorrect independencies

↘ additional parents ⇒ more accurate estimate !

↘ surpas the expert that made it !

↘ CPCS (94) → 500 var, avg # of values = 4 ⇒ 4^{500} !

number of params to construct CPD = 133×10^6

↘ simplifying assumption → 1000 params

↘ Microsoft

☑ Fault Diagnosis → Microsoft troublshooters

↘ Car Repair

↳ Benefits: flexible GUI, easy to design and maintain

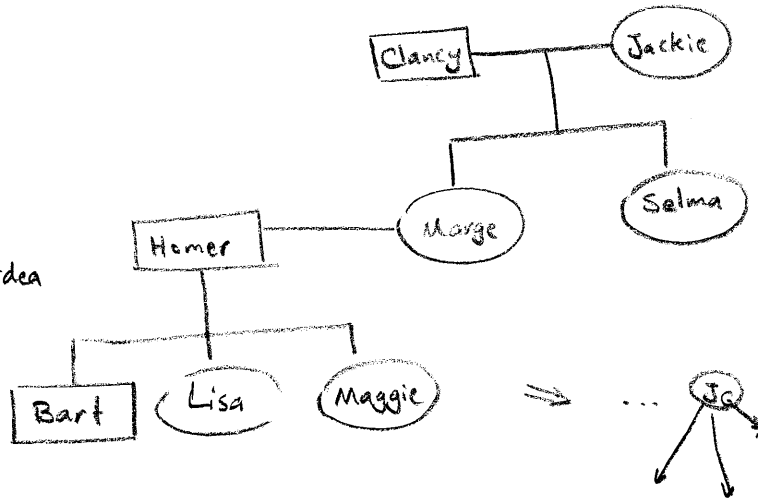
↓ skip question ...

↓ change prob., add an edge ...

* Template Models

☑ Genetic Inheritence

- Input : Pedigree
- Output : reasoning about family trait
- If we add nodes to family tree, we want have new network the same idea about Genotypes and Phenotypes



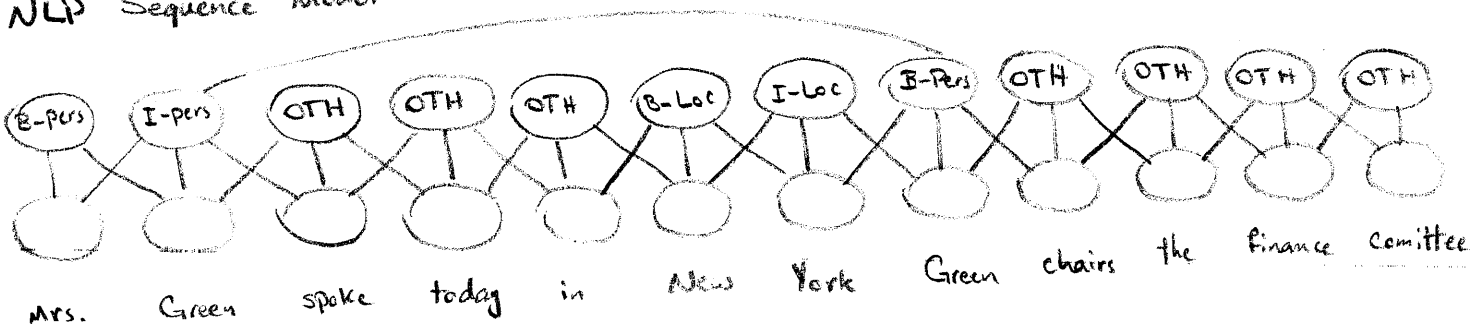
↳ Sharing between Models

- This model is too redundant

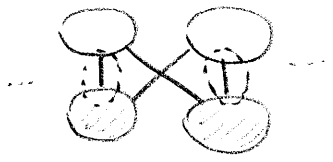
↳ Sharing within model

* CPD that tell how Selma G effects SelmaP is the same with other
* The structure of each person has two parents repeated in model

☑ NLP Sequence Model



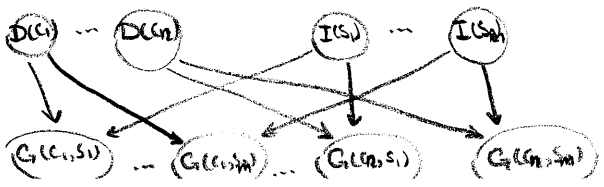
- task: Named entity recognition
- shared : relation between observation & latent var that are indep. from the place in the sequence.

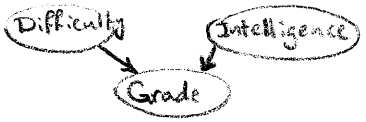


☑ Image Segmentation

→ no separate model for every superpixel in image
↘ sharing acress superpixels
↘ the model that relates class label of superpixel to image features
↘ parameters that involves adjacent superpixels are shared.
↘ pairs of superpixels shared
↘ for other images it works → between model

☑ Student Example for whole university

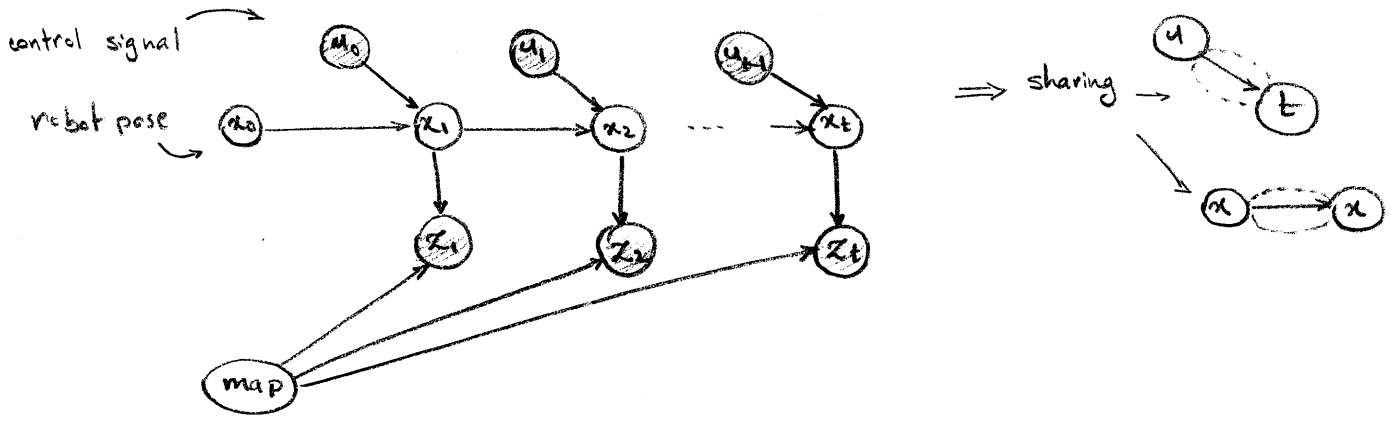




- the grade of a student within a course, depends on the relevent difficulty and intelligence of relevent student. ⟶ sharing of both structure and parameters

☑ Robot localization

- time series
- position at time t changes over time, we expect the dynamics of the robot to be fixed.



- Template variable : something that we are replecating again and again within single model as well as across model.

  - template variable ⟶ like a function that take arguments $X(U_1, \ldots, U_k)$
    ↘ instantiated / duplicated multiple times

  ☑ time points ⟶ Location (t), Sonar (t)

  person ⟶ Genotype (person), Phenotype (person)

  pixel ⟶ Label (pixel)

  Difficulty (course), Intelligence (student), Grade (course, student)

- Template model is a language that tells us the dependency model of template variables and how concrete instantiation of variables (= ground variable) inherit dependency from template.

- Benefits ⟶ CPDs in TMs can often be copied many times.
  ↘ TMs can often capture events that occur in time series.
  ↘ TMs can capture parameter sharing within the model.

- Different languages developed for TMs:
  - Dynamic BN ⟶ for temporal processes (replication over time)
  - Object-relational Models ⟶ Directed ⟶ plate models, ... ⟶ for multiple objects
    ↘ Undirected

✳ Temporal Models

- For systems that evolve over time ⟶ distribution over trajectories
- when representing over continous time ⟶ most cases forget the time is continuous ⟶ discretize
  - pick time granularity Δ ⟶ usually dictated by sensors
  - set of template r.v ⟶ $X^{(t)}$: variable X at time $t \times \Delta$

- $X^{(t:t')} = \{X^{(t)}, \dots, X^{(t')}\}$ $\quad t \leq t'$
- Want to represent $P(X^{(t:t')})$ for any t, t'

• Markov Assumption

- A type of conditional independence assumption → helps to compactify time course data

$$P(X^{(0:T)}) = \underbrace{P(X^{(0)}) \prod_{t=0}^{T-1} P(X^{(t+1)} \mid X^{(0:t)})}_{\text{chain rule for probabilities}}$$

($P(X^{(t+1)} \mid X^{(0:t)})$: state @ t+1 given state at previous times)

- Assumption: Future is independent of the past given the present

$$\underbrace{X^{(t+1)}}_{\text{next step}} \perp \underbrace{X^{(0:t-1)}}_{\text{past}} \mid \underbrace{X^{(t)}}_{\text{present}}$$

↳ a forgetting assumption → don't care any more about past

$$P(X^{(0:T)}) = P(X^{(0)}) \prod_{t=0}^{T-1} P(X^{(t+1)} \mid X^{(t)})$$

↳ Is this assumption true?

☑ X = location of robot $\quad L^{t+1} \perp L^{t-1} \mid L^{t}$

in most cases does not hold → ignores velocity (direction + speed)

to fix it → enrich the state description by adding V^t, ...

↳ make approximation better

↳ Markov assumption become much more warrented

move away Markov assumption

↳ adding dependencies that go further back in time

↳ Semi-Markov assumption

• Time Invariance Assumption

- We still need a probabilistic model for every P.

↳ we can use template probability model

- $P(X' \mid X)$ = P(next step | current time point) → we assume that model is replicated for every single time point

↳ Time Invariance

$$P(X^{(t+1)} \mid X^{(t)}) = P(X' \mid X)$$

- We assume that the dynamics of the system don't depend on current time point t.

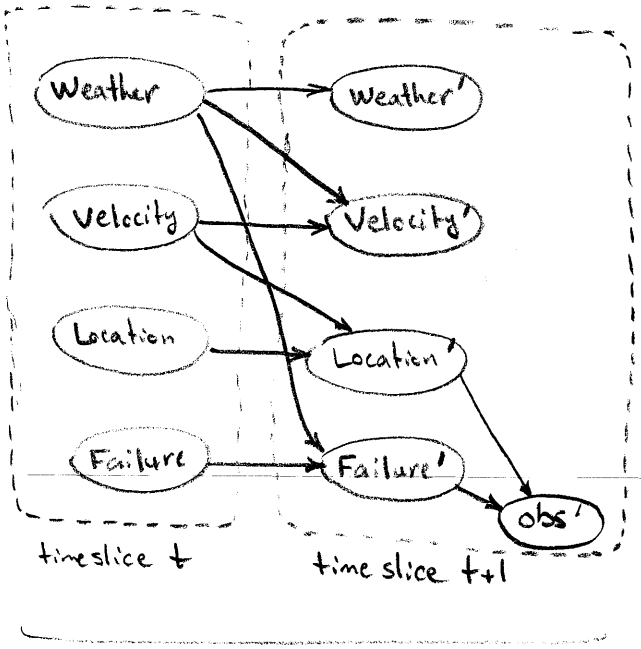
↳ Is this assumption true?

☑ traffic on road → does the dynamics of that traffic depends on time of system?

in most cases does not hold → depends on time of day, day of week, big football match

to correct the inaccuracies → enriching the model by including these variable

template transition model



O has no effect

$$P(W', V', L', F', O' \mid W, V, L, F) = P(W' \mid W) \times P(V' \mid V, W) \times P(L' \mid L, V) \times P(F' \mid F, W)\, P(O' \mid L')$$

dependancy across time $t \to t+1$

within ti $t+1 \to t+1$ ↓ rapidly-acting before the next time step

* ↓ intra time-slice edge

* → inter time-slice edge

net work fragment → gives conditional dist given t

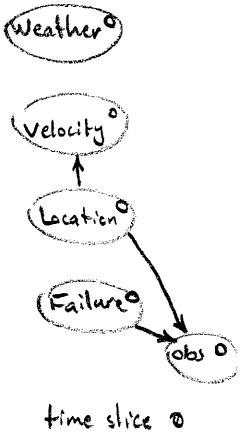
* inter time-slice edges from $X^{(t)} \to X^{(t+1)}$ are often called persistence edges

* model don't have CPD for time t, only tries to represent the probability of the next time slice given the previous one.

* we need initial state distribution

$$P(W^{(0)}, V^{(0)}, L^{(0)}, F^{(0)}, O^{(0)}) = P(W^{(0)})\, P(V^{(0)} \mid L^{(0)})\, P(L^{(0)})\, P(F^{(0)})\, P(O^{(0)} \mid F^{(0)}, L^{(0)})$$

chain rule for BN



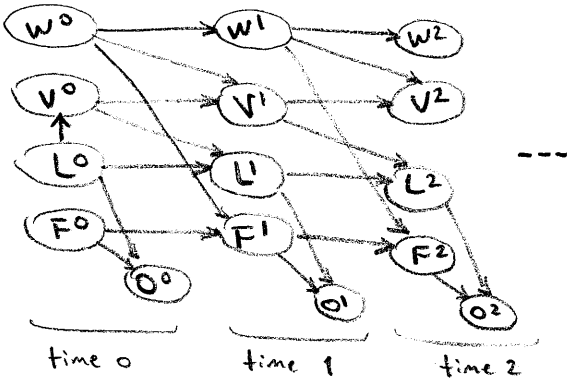
time slice 0

* ground BN

↳ copy time slice 0

↳ copy prob. dist. @ t1 given t0

↳ copy prob.dist @ t2 given t1



- 2-Time-Slice BN

A transition model (2TBN) over $X_1, \dots, X_n$ is specified as a BN fragment such that:

- the nodes have 2 copies: $X_1', \dots, X_n'$ and a subset of $X_1, \dots, X_n$

- this subset of $X_1, \dots X_n$ at time t are variables that directly affect the state at $t+1$
- only the nodes $X_1', \dots, X_n'$ have parents and CPD
- the 2TBN defines a conditional distribution using chain rule $P(X'|X) = \prod_{i=1}^{n} P(X_i' | Pa(X_i'))$

maybe in $t+1$
in t
Combination of both

- **Dynamic BN**

A Dynamic Bayesian Network (DBN) over $X_1, \dots, X_n$ is defined by a:

- 2TBN $BN_{\rightarrow}$ over $X_1, \dots, X_n$ → dynamics
- a Bayesian network $BN^{(0)}$ over $X_1^{(0)}, \dots, X_n^{(0)}$ → time 0 → initial state

- **Ground Network**

For trajectory over $0, \dots, T$ we define a ground (unrolled network) such that:

- the dependency model for $X_1^{(0)}, \dots, X_n^{(0)}$ is copied from $BN^{(0)}$ → time 0
- the dependency model for $X_1^{(t)}, \dots, X_n^{(t)}$ for all $t > 0$ is copied from $BN_{\rightarrow}$ → transition

- **Summary**

- DBN → compact representation → encoding structured dist.
  ↘ arbitrarily long temporal trajectories

- Assumptions → Markov assumption
  ↘ Time invariance

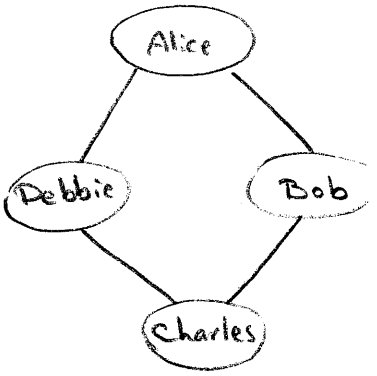
↳ some times are not true → need model redesign for better approximation

* Markov Network Fundamentals:

- Undirected graphical models are typically called MN. → Markov Random Field
- Simplest case: Pairwise Markov Network

* Pairwise Markov Network

- Misconception Example
  - Alice & Charles don't get along, Bob & Debbie had breakup
  - Random variables indicates whether the students have a particular misconception
  - Connection: if two students study together they influence each other
    - ↳ influence goes both ways → undirected
  - Parameterization → undirected: no conditional prob. dist.



↘ using factors → general factor: numbers not in [0,1]

↳ names: affinity functions, compatibility function, soft constraints

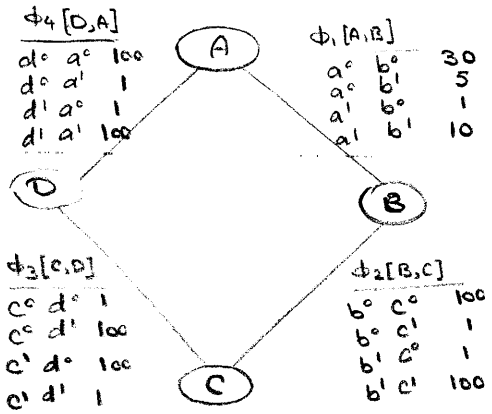
↳ means: local happiness of vars A and B in $\phi_1[A,B]$ in a joint assignment

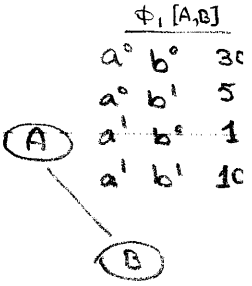
↳ the happiest assignment for A and B in isolation of everything else is $a^0 b^0$ ⇒ neither have misconception

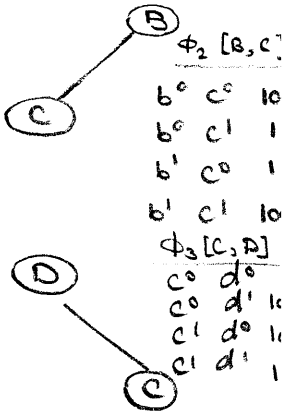
↳ the second happiest is $a^1 b^1$ where students both have misconception

↳ B & C really like to agree with eachother. A & D like to agree the same of B & C.

↳ C & D likes to argue with each other all the time.







↘ describe the whole situation by a bunch of little pieces together → using product of factors

$$\tilde{P}(A,B,C,D) = \phi_1(A,B) \times \phi_2(B,C) \times \phi_3(C,D) \times \phi_4(D,A)$$

not in range of $[0,1]$ and $\sum \tilde{P}_{(A,B,C,D)} \neq 1$

so $\tilde{P}$ is unnormalized measure.

↳

assignment	unnormalized
a^0 b^0 c^0 d^0	300,000
a^0 b^0 c^0 d^1	300,000

↘ to normalize it

$$P(A,B,C,D) = \frac{1}{Z}\tilde{P}(A,B,C,D)$$

Z: partition function ↝ come from statistical physics

↳ constant factor that help P to sum to 1

↳ $Z = \sum_{A,B,C,D} \tilde{P}(A,B,C,D)$

↘ $\Phi_1(A,B)$ = local happiness between A and B, how it related to prob. dist?

↳ the marginal prob. $P(A,B)$ ✗

↳ the conditional prob. $P(A|B)$ ✗

↳ the conditional prob. $P(A,B|C,D)$ ✗

A	B	$P_\Phi(A,B)$
a^0	b^0	0.13
a^0	b^1	0.69
a^1	b^0	0.14
a^1	b^1	0.04

→ P_Φ derives from the set of factors $\Phi = \{\phi_1, \phi_2, \phi_3, \phi_4\}$

→ this does not agree by the fact A&B likes to agre

→ highest assignment : $a^0 b^1$

↳ this P_Φ comes from multiplying all four

B really likes to agree with C

A really likes to agree with D

C and D strongly like to diagree

these 3 assignments are bigger than values in $\phi_1(A,B)$

so $P(A,B)$ should break the loop of $P \to A \to B \to C \not\to D$

because it's weaker. so it is a complicated aggregate o

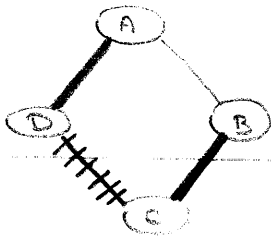
these different factors that are used to compose the MN

↳ there isn't a natural mapping between $P(A,B)$ and the factors that are used to compose it.

↳ In contrast with BN

↳ Impossible to look at the prob. dist and say eg. this piece of it is what ϕ_1 ought to be.

↳ Makes it harder to learn factors from data becaus we can't extract them directly from the prob. dist



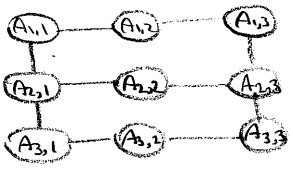
- Pairwise MN is an undirected graph whose nodes are $\underbrace{X_1, \ldots, X_n}_{\text{r.v.}}$ and each edge X_i-X_j associated with a factor $\Phi_{i,j}(X_i, X_j)$

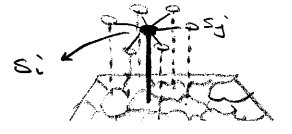
☑ 2D lattice

Images → variables = pixels, edges = neighbor

☑ Image segmentation

no regular grid

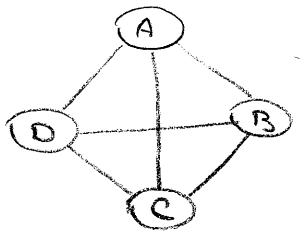




* General Gibbs Distribution

- More general than the pairwise case
- Most expressive pairwise MN: fully connected

$P(A,B,C,D)$



↳ is this fully expressive? (= can I represent any prob. dist. over four random variable?)

↳ If each X_i has d values in fully connected pairwise MN over $X_1, \dots, X_n$ → # of parameters?

# of edges in network : $\binom{n}{2}$

# of parameters per edge: d^2

$\Rightarrow O(n^2 d^2)$

Can we represent any prob. dist. using $O(n^2d^2)$ parameters? No!

↳ How many prob. dist. over N random variables where each has D values? $O(d^n)$

$O(d^n) \gg O(n^2d^2)$ → pairwise MN is not sufficiently expressive to have all prob. dist.

$\underbrace{d \times d \times \dots \times d}_{n \text{ variables}}$

to increase the expressiveness → move away from pairwise edges.

- General Gibbs Distribution

= parameterize using general factors, each with scope that may contain more that two variables

$\Phi_i(D_i)$ → general factors

$\Phi = \{\Phi_i(D_i)\}$ → set of factors

– Can we represent any prob.dist.? Yes!

↳ prob. dist. is a factor whose scope is the exponential of X_n.

– Formal: A Gibbs dist is parameterized by a set of factors Φ

$\Phi = \{\Phi_1(D_1), \dots, \Phi_K(D_K)\}$ → set of factors

$\tilde{P}_\Phi(X_1, \dots, X_n) = \prod_{i=1}^{K} \Phi_i(D_i)$ → factor product, not a prob. dist, unnormalized measure

$Z_\Phi = \sum_{X_1, \dots, X_n} \tilde{P}_\Phi(X_1, \dots, X_n)$ → partition function, normalizing constant

$$P_\Phi(X_1, \dots, X_n) = \frac{1}{Z_\Phi} \tilde{P}_\Phi(X_1, \dots, X_n)$$

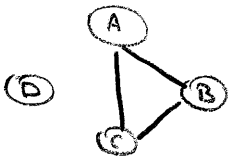
– what is the MN that we would like to have for a Gibbs dist. of the certain set of fac Φ?

☑ Two factors

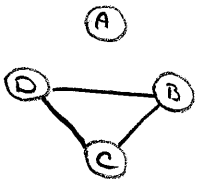
$\Phi_1(A,B,C)$ → A, B and C all get to interact with each other

↳ links between AB, BC and AC

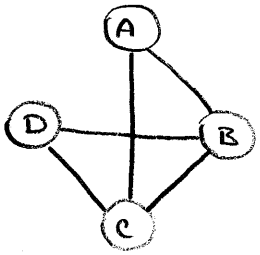
↳ direct prob. relationships between all of them



$\Phi_2(B,C,D)$



Induced MN by fators Φ_1 and Φ_2

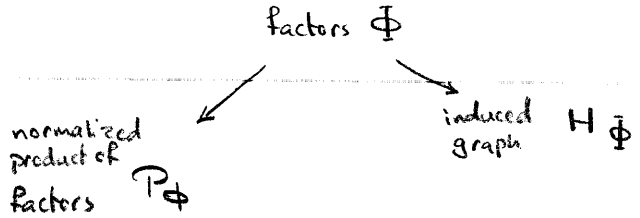


↳ Generally if we have a set of factors Φ, where each Φ_i has a scope D_i the induced Markov Network H_Φ has an edge $X_i - X_j$ whenever there exist a $\Phi_m \in \Phi$ such that $X_i, Y_i \in D_m$.

• Factorization → Answer the question: is there set of factors Φ that are going to let me represent P?

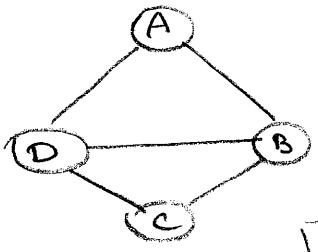
_P factorizes over H if there exist $\Phi = \{\Phi_1(D_1), \dots, \Phi_K(D_K)\}$ such that $P = P_\Phi$ and H is induced graph for Φ.

normalized product of factors



_ If we have graph H can we tell what factorization of dist. P over that graph would be?

☑ Which Gibbs distribution would induce the graph H?



↳ $\Phi_1(A,B,D), \Phi_2(B,C,D)$ ✓

↳ $\Phi_1(A,B), \Phi_2(B,C), \Phi_3(C,D), \Phi_4(D,A), \Phi_5(B,D)$ ✓

↳ $\Phi_1(A,B,D), \Phi_2(B,C), \Phi_3(C,D)$ ✓

We cannot read the factorization from the graph

↳ different factorization → different expressive power → all may induce the same graph.

↳ Pairwise $O(n^2 d^2)$

↳ one giant factor $O(d^n)$

↳ ...

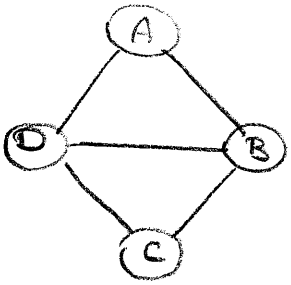
↳ why the graph is the same? what does it tell us? it's not telling as the structure of the factorization so what?

— Flow of Influence

↳ when can one var. influence another?

* when B influence D?

Case 1: $\phi_1(A,B,D)$, $\phi_2(B,C,D)$
↳ tie B & D directly



Case 2: $\phi_1(AB)$, $\phi_2(B,C)$, $\phi_3(C,D)$, $\phi_4(D,A)$, $\phi_5(B,D)$
tie B & D directly

* when A influence C?

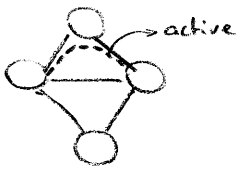
Case 1: $A \to D \to C$: $\phi_1(A,B,D) \to \phi_2(B,C,D)$

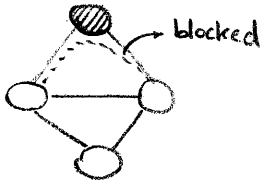
Case 2: $A \to D \to C$: $\phi_4(A,D) \to \phi_3(C,D)$

↳ the parameterization of the two dist. are different BUT
the trail in the graph through which influence can flow is the same REGARDLESS
finer grain structure of the factorization → this why graphs are the same
∴ Influence can flow along any trail, regardless of the form of the factors.

— Active trail in MN

A trail $X_1 - \cdots - X_n$ is active given $\mathcal{Z}$ if no X_i is in $\mathcal{Z}$
no block





↳ Difference of active trails in BN and MN

↳ V-structure → BN: $\mathcal{Z}$ observed → create active trail
→ MN: $\mathcal{Z}$ observed → block the trail

— Summary

1. Gibbs dist represent dist as a product of factors
2. Induced MN connects every pair of nodes that are in the same factor
3. MN structure doesn't fully specify the factorization of P
4. But active trails depend only on graph structure

*Conditional Random Fields (CRF)

- This class of models are intended to deal with «task specific prediction»

— input / observed variables X

target variables Y

task: predict Y ⟶ classic models : same type of variables in input and output

☑ Image segmentation — X : pixel values & processed features (histograms, colors, textures, ...

Y : class for every pixel (e.g. grass, sky, cow, water, ...)

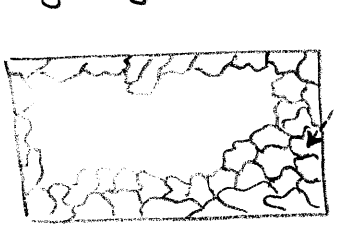
Task : going from X ⟶ Y

☑ Text processing — X : words in sentence

Y : labels of the words (e.g. person, location, organization, ...)

— Why don't use the models we know so far?

↳ Image Seg. task ⟶ predict C_i



↳ high level features ⟶ more informative
⟶ more correlated with each other

↳ we represented it in naïve bayes model ( ) in which features should be independent given label ⟶ we ignore the correlation structure ⟶ we count the same feature again and again and again ⟶ push us toward very skewed dist ⟶ far from true belief because of indep. assumption

↳ to correct indep. assumption ⟶ add some edges to capture the correlations
↳ hard to figure out the correlations
↳ gives rise to densly connected models

↳ Alternative solution : I don't care about the image features !
I don't want to predict the prob. dist. over pixels !
I don't want to do image synthesis !
I don't care about prob. of having green pixel next to another green pix
I only use X to model distribution over Y !

Reformulate : Instead of modeling $P(X,Y)$
Model $P(Y|X)$ ⟶ not trying to capture dist over X ⟶ I should not care abou

- CRF Representation
  - Just like a Gibbs dist
  - $\Phi = \{\phi_1(D_1), \ldots, \phi_K(D_K)\}$ → set of factors ϕ_i with scope D_i

$$\tilde{P}_\Phi(X,Y) = \prod_{i=1}^{K} \phi_i(D_i)$$ → an unnormalized measure = product of factors

difference with Gibbs

$$Z_\Phi(X) = \sum_Y \tilde{P}_\Phi(X,Y)$$ → partition function, a function of X

difference with Gibbs

↳ means: for any given X, we sum over all Y's corresponding to that x

$$P_\Phi(Y|X) = \frac{1}{Z_\Phi(X)} \tilde{P}_\Phi(X,Y)$$

☑

x^0	y^0	A
x^0	y^1	B
x^1	y^0	C
x^1	y^1	D

A + B ⇒ $Z(x^0)$; C + D ⇒ $Z(x^1)$ ⟹

$A/Z(x^0)$
$B/Z(x^0)$
$C/Z(x^1)$
$D/Z(x^1)$

☑ consider a new factor $\phi(D)$ such that $D \subseteq X$. what is the effect of adding this factor ϕ on distribution $P(Y|X)$?

$$P_\Phi(Y|X) = \frac{\tilde{P}_\Phi(X,Y)}{Z_\Phi(X)} = \frac{\prod_i \phi_i(D_i)}{\sum_Y \tilde{P}_\Phi(X,Y)} = \frac{\prod_i \phi_i(D_i)}{\sum_Y \prod_i \phi_i(D_i)}$$

adding a new factor → multiply to numerator & denominator → cancels out

- CRFs and Logistic Model
  - X_i and Y are binary r.v.

$\mathbb{1}$ indicator function → 1 if all conditions are met, 0 otherwise

using log-linear model

$$\phi_i(X_i, Y) = \exp\{\omega_i \mathbb{1}\{X_i = 1, Y = 1\}\}$$

$$\phi_i(X_i, Y=1) = \exp\{\omega_i \mathbb{1}\{X_i=1, \underbrace{Y=1}_{\text{always 1}}\}\} = \exp\{\omega_i \mathbb{1}\{\underbrace{X_i=1}_{\text{binary } X_i \in \{0,1\}}\}\} = \exp\{\omega_i x_i\}$$

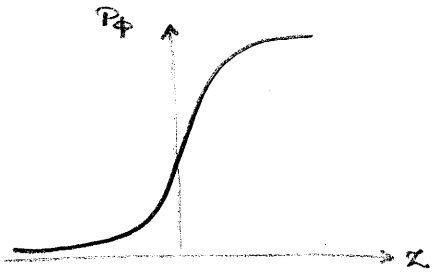
$$\phi_i(X_i, Y=0) = \exp\{\omega_i \mathbb{1}\{X_i=1, \text{false}\}\} = \exp\{\omega_i \times 0\} = e^0 = 1$$

Calculating unnormalized density (2 cases: $Y=0$ & $Y=1$)

$$\tilde{P}_\Phi(X, Y=1) = \underbrace{\exp\left\{\sum_i (\omega_i X_i)\right\}}_{\prod_i e^{\omega_i X_i}} \qquad \underbrace{\tilde{P}_\Phi(X, Y=0) = 1}_{\text{product of ones}}$$

$$Z_\Phi(X) = \sum_Y \tilde{P}_\Phi(X, Y) = \tilde{P}_\Phi(X, Y=1) + \tilde{P}_\Phi(X, Y=0) = 1 + \exp\left\{\sum_i \omega_i X_i\right\}$$

$$P_\Phi(Y=1 \mid X) = \frac{\exp\left\{\sum_i \omega_i X_i\right\}}{1 + \exp\left\{\sum_i \omega_i X_i\right\}} = \frac{e^x}{1+e^x}$$



Logistic is very simple CRF.

☑ CRFs for Image Segmentation

- Node factors can use any features of the image → Color histograms
  → Texture features
  → Discriminative Patches → like looking for the eye on a cow
- Features can be in and out of the superpixel
  ↳ green under this superpixel ☑ on grass
  ↳ counting redundant superpixels? I don't care
- Correlation doesn't matter
- Can train a discriminative classifier (SVM, Boosting) to improve performance
  ↳ e.g. train very strong classifiers for node potentials $P(Y_i \mid X_{1\ldots n})$ then adding pairwise or higher order features between the Y's.
  features of log-linear not image features

☑ CRFs for Languages

- High level features → high correlation
  → word capitalized
  → word in Atlas or NameList
  → previous word is "Mrs."
  → next word is "Times"
  (correlated)
- P(Labels | words)

• Summary
- CRF is parameterize the same as a Gibbs dist. but normalized differently → $P(Y \mid X)$
  ↳ Generalizes logistic regression models
- Don't need to model dist. over vars. we don't care about
- Allows models with highly expressive features without worrying about wrong independencies.
  ↳ Enable design really expressive predictors of pieces of the model

* Markov Networks and Independencies

what kind of indepencies are encoded by the structural graph of MN

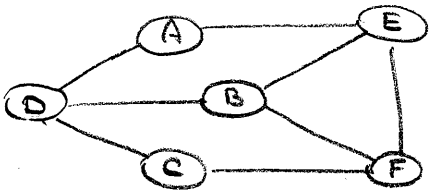
- Separation in MNs

- X and Y are separated in H given Z if there is no active trail in H between X and Y.

- simpler than d-separation → no different cases
  (d-separation: directed)

☑ What does it take to separate A from E?

A and E separated given B, D
given D
given C



- Factorization ⇒ Independence

- Theorem: If P factorizes over H, and $sep_H(X, Y | Z)$ then P satisfies $(X \perp Y | Z)$

$I(H) = \{ (X \perp Y | Z) : sep_H(X, Y | Z) \}$

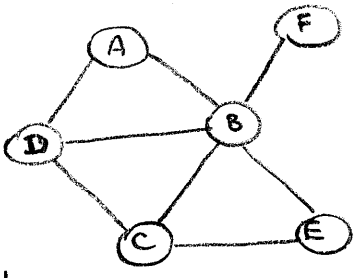
If P satisfies I(H) we say that H is an I-map of P
(satisfies I(H): all the independencies that we can read from graph H)

- Theorem: If P factorizes over H, then H is an I-map of P

P factorizes over H → satisfy all of the independencies that one can read from H.

☑ For the graph H shown here, which independence assertions are in I(H)

$A \perp C | B, D$ ✓
$A \perp E | B$ ✗
$C \perp F | B$ ✓
$A \perp F \perp B, D$ ✓



- In BN the converse holds, in MN the converse sort of holds.

- Independence ⇒ Factorization

- Theorem (Hammersley-Clifford): For a positive dist. P, if H is an I-map for P then P factorizes over H.
  (positive: difference with BN) $P(X_i) > 0$

↳ if distribution involves deterministic relationship this property no longe holds.
(deterministic: $P(x_i) \in \{0, 1\}$)

- Summary:
  - Two almost equivalent views of graph structure: Factorization, I-map (for positive dist.)
  - If P factorizes over a graph H, we can read from the graph independencies that must hold in P
    I-map (P)

* I-maps and Perfect Maps :

• A graph structure —encode→ set of independencies —necessarily hold→ every dist. that can be encoded as BN over that graph

↳ How to take a distribution that has a certain set of independencies that satisfies and encode it within a graph structure?

• what is the independencies of a dist? Capturing indep. in P

$$I(P) = \{(X \perp Y \mid Z) : P \models (X \perp Y \mid Z)\}$$

↳ set of all independence that hold for dist P

↳ write all of the exponential numbers of possible indep. statements ⟶ I(P) is the ones that hold

if P factorizes over G ⟹ G is I-map for P

↳ every independence that hold in G that is implied by d-separation properties of G also holds in P

↳ $I(G) \subseteq I(P)$

↳ there can be indep. in I(P) that are not in I(G)

• We have a graph that doesn't capture some of the independencies in P then it's unneccessarily complicated

↳ we want a sparser graph ⟶ encodes more indep.
- fewer parameters → more efficient inference
- more informative

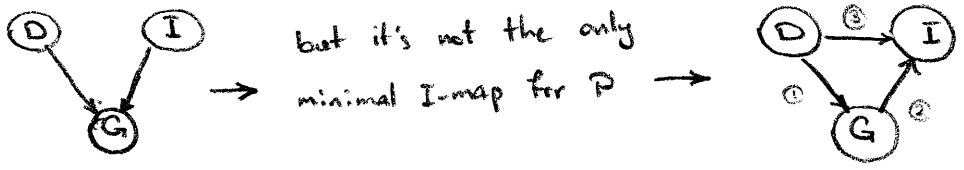
– Want a graph that captures as much of the structure in P as possible.

– Minimal I-map : I-map without redundant edges

☑ If we have (X)→(Y) but Y doesn't really depend on X in the sense $P(Y \mid x^0) = P(Y \mid x^0)$ we can remove the edge and it remains an I-map, so the first I-map was not minimal

↳ Minimal I-map may still not capture I(P) !

☑ we have dist P that corresponds to left minimal I-map



it is a minimal I-ma because no edge could be removed.

remove:

① ⇒ D ⊥ G ✗

② ⇒ G ⊥ I | D ✗

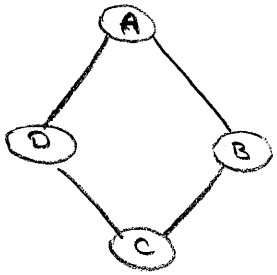
③ ⇒ D ⊥ I | G ✗

– So I-map is not the best tool for capturing structure in data distribution

• Perfect Map

- Perfect map: $I(G) = I(P)$ ⇝ BN as a perfect map
  - ↳ G perfectly captures indepedencies in P
  - ↳ usually hard to come by perfect maps ⟶ sometime dist. doesn't have perfect maps.

☑

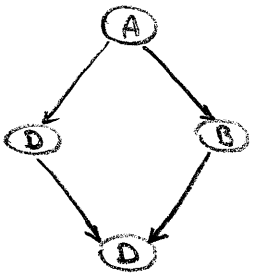


$P \models A \perp C \mid B, C$

$P \not\models B \perp D \mid A, C$

⟹

attemp 1

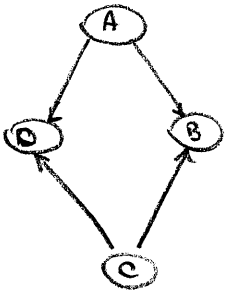


G: $B \perp D \mid A$

$P \not\models B \perp D \mid A$ ✗

not an I-map

attemp 2



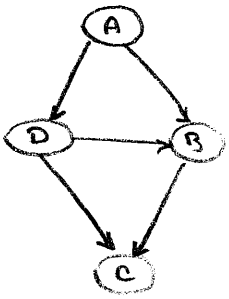
G: $B \perp D \mid A, C$ ✓

G: $A \perp C$

$P \not\models A \perp C$ ✗

not an I-map

attemp 3



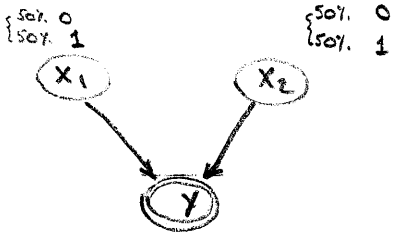
G: $A \perp C \mid B, D$

$I(G) \subset I(P)$

I-map ✓

perfect map ✗

☑ XOR example



X_1	X_2	Y	Prob
0	0	0	0.25
0	1	1	0.25
1	0	1	0.25
1	1	0	0.25

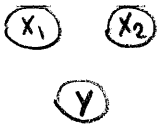
$X_1 \perp X_2$

$X_1 \perp Y$

$X_2 \perp Y$

no I-map for this graph

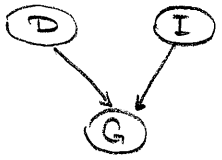
because the only way is:



that is not an I-map

- perfect map: $I(H) = I(P)$ ⇝ MN as a perfect map
  - ↳ H perfectly captures independecies in P
  - ↳ can we capture all possible dist. in terms of a MN as a perfect map? NO!

☑



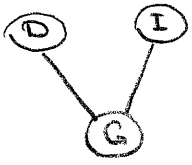
$P \not\models D \perp I$

Has perfect map as a BN

doesn't have perfect map as a MN

v-structure example

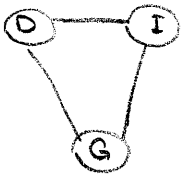
⟹



H: $G \perp I \mid G$

$P \not\models G \perp I \mid G$

⟹



$P \not\models D \perp I$

✗

- Uniqueness of Perfect Map

☑ simplest case

G_1: 

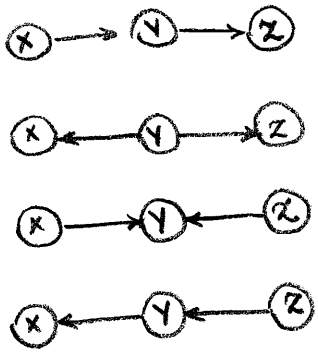
$I(G_1) = \phi$

G_2: 

$I(G_2) = \phi$

⟹ can represent exactly the same set of all of distributions

☑ Which of the following graphs does not encode the same indep. as the others?



$\{X \perp Z \mid Y\}$

$\{X \perp Z \mid Y\}$

✓ v-structure $\{X \perp Z\}$

$\{X \perp Z \mid Y\}$

- I-equivalence

Two graphs G_1 and G_2 over $X_1, \ldots, X_n$ are I-equivalent if $I(G_1) = I(G_2)$

☑ $X \to Y \to Z$ and $X \leftarrow Y \to Z$ and $X \leftarrow Y \leftarrow Z$ are I-equivalent

↳ tells us that there's certain aspects of the graphical model that are unidentifiable.

↳ without prior knowledge or preference of who is the parent of who we have no single choic

↳ most graphs have large numbers of I-equivalents.

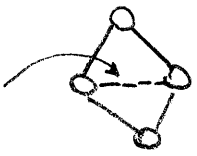
↳ complicated e.g. for learning

- Summary

- Graphs that capture more of $I(P)$ are more compact and provide more insight
- A minimal I-map may fail to capture alot of structure even if present and representable as a PG
- A perfect map is great, but may not exist
- Convertin BN $\rightleftharpoons$ MN loses independencies

↳ BN → MN : loses indep. in v-structures

↳ MN → BN : must add triangulating edges to loops



* Log-Linear Models

- Local structure that doesn't need full table representations is important in both directed and undirected models

↳ how do we incorprate input structure into undirected models? → Log-linear models!

- Log-Linear Representation

$\tilde{P} = \prod_i \phi_i(D_i)$ → potentially a full table

⇩

$\tilde{P} = \exp\left(-\sum_j \omega_j f_j(D_j)\right)$ → linear model → because logarithm is linear fu

coefficients, features, scope of feature j

But different features can have the same scope. Each feature have specific coeff.

$\tilde{P} = \prod_j \underbrace{\exp(-w_j f_j(D_j))}_{\text{factor}}$ → product of factors that each has a single variable w_j

- Representing Table Factors

$$\phi(X_1, X_2) = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix}$$

factor ϕ over two binary r.v. X_1 and X_2

↳ full table has 4 params : $a_{00}, a_{01}, a_{10}, a_{11}$

we can capture this model using log-linear model using a set of such features

↳ indicator functions

$$f_{12}^{00} = \mathbb{1}\{X_0 = 0, X_2 = 0\}$$

$$f_{12}^{01} = \mathbb{1}\{X_0 = 0, X_2 = 1\}$$

$$f_{12}^{10} = \mathbb{1}\{X_0 = 1, X_2 = 0\}$$

$$f_{12}^{11} = \mathbb{1}\{X_0 = 1, X_2 = 1\}$$

$$\Rightarrow \quad \phi(X_1, X_2) = \exp\left(-\sum_{k=0,1} \sum_{\ell=0,1} w_{k\ell} f_{ij}^{k\ell}(X_1, X_2)\right)$$

↳ when $X_1 = 0$ and $X_2 = 0$: $\phi = \exp(-w_{00})$

↳ when $X_1 = 0$ and $X_2 = 1$: $\phi = \exp(-w_{01})$

if we assume $w_{k\ell} = -\log a_{k\ell}$ we get the original factor back.

but generally we want a much finer grain set of features.

☑ Language model

Y: represent the annotations for each word in the sequence corresponding to what category ...

↳ Begin-Person, Intermediate-Person, OTH, B-location, I-loc

OTH (nothing) → no person or location, other

↳ value Y tells us what category it belongs to

X: actual words in the sentence

full table rep. : tries to relate each and every Y that has a feature, to every word in Language

features : $f(Y_i, X_i) = \mathbb{1}\{Y_i = \text{B-person}, X_i = \text{capitalized}\}$

↳ how important is capitalization for recognizing that sth is a person

$= \mathbb{1}\{Y_i = \text{B-location}, X_i \text{ appears in Atlas}\}$

☑ Ising Model : from statistical physics

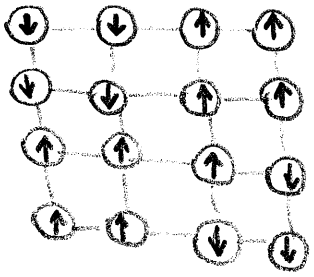
- looks at the pairs of variables → Pairwise MN

looks at pairs of adjacent variables and gives a coefficient for their product

$E(x_1, \dots, x_n) = -\sum_{i<j} w_{ij} \cdot x_i \cdot x_j - \sum_i u_i x_i$

$f(x_i, x_j) = x_i \cdot x_j \qquad x_i \in \{-1, +1\}$

- modeling spins of electrons in a grid



↳ P = the joint spins in the model

↳ depends on weather adjacent atoms have the same spins or opposite spin.

$$P(X) \propto e^{-\frac{1}{T}E(X)}$$

↳ depends on ω

↳ systems that atoms spin at same direction (ferro magnetic) VS. system that atoms have opposite spi (anti ferro magnetic)

↳ temperature T : how strong is this connection

↳ $T\uparrow$: $\frac{\omega_{ij}}{T} \to 0$ → atoms become decoupled from each other

$T\downarrow$: $\frac{\omega_{ij}}{T} \uparrow$ → atoms impose a much stronger effect on each other

• Metric MRF

- another kind of features that are used in real applications is metric MRF

↳ sth that comes up mostly when random var. X_i all take values in label space V

☑ all binary, all $\in \{1,2,3,4\}$

↳ we want X_i and X_j that are connected to each other take similar values.

↳ we need similarity notion

- Distance function : $\mu : V \times V \to \mathbb{R}^+$ (for x_i, for x_j)

↳ need to satisfy the standard condition on a distance function or metric

* Reflexivity $\mu(v,v) = 0 \quad , \forall v$

* Symmetry $\mu(v_1, v_2) = \mu(v_2, v_1) \quad , \forall v_1, v_2$

* Triangle Inequality $\mu(v_1, v_2) \leq \mu(v_1, v_3) + \mu(v_3, v_2) \quad , \forall v_1, v_2, v_3$

↳ if a function just satisfies the first two → semi-metric

↳ put in MRF $f_{i,j}(x_i, x_j) = \mu(x_i, x_j)$

$$\phi(x_i, x_j) = \exp(-\omega_{ij} \cdot f_{ij}(x_i, x_j)) \quad , \omega_{ij} > 0$$

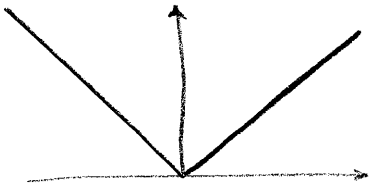
↳ lower distance $\mu\downarrow \Rightarrow \phi\uparrow$

values of x_i and x_j are far ⇒ lower probability

☑ $\mu(v_k, v_l) = \begin{cases} 0 & v_k = v_l \\ 1 & \text{otherwise} \end{cases}$ ⇒ step function ⇒ $\Phi = \begin{bmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & 0 & 1 & & \\ 1 & 1 & 0 & & \\ \vdots & & & \ddots & \\ 1 & & & & 0 \end{bmatrix}_{V \times V}$

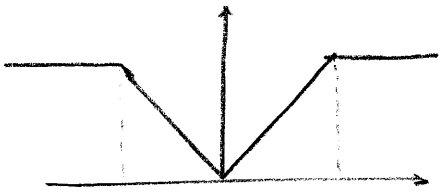
☑ linear penalty

$\mu(v_k, v_l) = |v_k - v_l|$



☑ truncated linear penalty

$\mu(v_k, v_l) = \min(|v_k - v_l|, d)$



☑ Image Segmentation

we want the adjacent super pixels to take the same class $\mu(v_k, v_l) = \begin{cases} 0 & v_k = v_l \\ 1 & \text{otherwise} \end{cases}$

☑ Image Denoising ↝ Gaussian Noise

X : noisy pixels

Y : clean pixels

{ we want Y_i to be close to X_i ⟶ recover original image
{ adjacent pixels tend to have the same value ⟶ constraint

↳ linear penalty ⟶ fragile model ⟶ tend to set all the pixels equal

↳ truncated linear ⟶ only close pixels are important

☑ Stereo Reconstruction

Y_i : depth disparity for a given pixel in the image ⟶ we have spatial continuity.

↳ depth of a pixel is close to the depth of adjacent pixel

we don't want to push this constraint too hard ⟶ truncated linear model + some trick

tricks ⟶ pixel appearance : color, texture, ... ⟶ adj pixel are nearly the same ⇒ same object

↘ adj pixel are very different ⇒ another object

• Shared Features in Log-Linear Models

↳ In most MRFs, same feature and weight are used over many scopes

☑ Ising Model

$$E(x_1, \cdots, x_n) = -\sum_{(i,j) \in \text{edges}} \underbrace{w_{ij}}_{\text{weights}} \underbrace{x_i x_j}_{\hookrightarrow f(x_i, x_j)} - \sum_i d_i x_i$$

↳ same weight for every adjacent pair ⟶ $w = w_{ij}$

↳ feature $f(x_i, x_j) = x_i \times x_j$ is reused throughout the model

☑ Natural Language Processing

↳ same energy terms $w_K f_K(X_i, Y_i)$ repeat for all positions i in the sequence

↳ Same energy terms $w_m f_m(Y_i, Y_{i+1})$ also repeat for all positions i

☑ Image Segmentation

↳ same features and weights for all superpixels in the image

– Need to specify for each feature f_k a set of scopes $Scopes[f_k]$

↳ for each $D_k \in Scopes[f_k]$ we have a term $w_k f_k(D_k)$ in the energy function

↳ e.g. if we want f_k to be applied to adjacent superpixels in the image

$$Scope[f_k] = \{Y_i, Y_j : i,j \text{ adjacent}\}$$

↳ Weight (w_k) depends on the feature (k), but not in the scope (D_k)

$$w_k \sum_{D_k \in Scopes(f_k)} f_k(D_k)$$

– Summary

↳ Same feature & weight can be used for multiple subsets of variables

↳ pairs of adjacent pixels / atoms / words

↳ occurance of same word in the document

↳ Can provide a single template for multiple MNs

↳ different images

↳ different sentences

↳ Parameters and structure are reused within an MN and across different MNs.

↳ Need to specify set of scopes for each feature

* Knowledge Engineering

↳ How to build a graphical model for some application having all of this info → Art, Black Magic

- Important Distinctions

1– Template based vs. Specific

↳ specific model for fixed set of r.v

2– Directed vs. Undirected

3– Generative vs. Discriminative

⇒ Hybrids are also comm

☑ Hybrids

template-based pieces + stuff that is not in template level

directed components + undirected ones

- Template - Based vs. Specific

Specific

Medical Diagnosis
↳ particular set of symptoms, diseases,...

Template - Based

Image Segmentation
↳ different images with same model

fault diagnosis

unique elements ← generic printer → particular printer → shared elements

- the place decided to sit on this spectrum changes knowledge engineering problem
- template-based → usually have a small number of variable types
  - ☑ Image segmentation → class label only
    - ↳ most effort goes to features
    - ↳ the main design challenge goes to figuring out that which features are most predictive
      - ↳ feature engineering
- Specific → usually a large number of unique vars. → each requires its own model

- Generative vs. Discriminative

Generative

- Don't have a predetermined task
  - ↳ task shifts
    - ↳ e.g. medical dignosis ~ every patient present differently
- Flexibility
  - ↳ to measure different vars. and predict others
- Easier to train in certain regimes
  - ↳ the case that data are not fully labled

Discriminitive

- Particular Prediction task
  - ↳ Richly expressive features
  - ↳ Avoid dealing with correlations
    - ↳ Higher performance

? what are the key decisions about designing a graphical model

- Variable Types
  - Target
    - ↳ regardless of weather we have a fix or varying task in hand we have a set of target vars. (↳ we care about'e
  - Observed
    - ↳ including complex, constructed features
  - Latent / Hidden

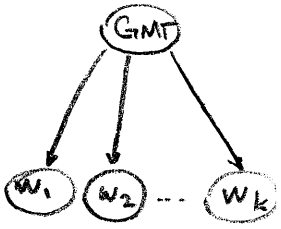
↳ we don't care about predicting neccessarily it is not observed ⇒ WTF it is used for then!?

☑ What time does your watches show?

w_i : the time watch show → all correlated with each other

↳ not because everybody set it with eachother

↳ using GMT



naive bayes ⇒ $w_i \perp w_j \mid GMT$ → GMT is latent

↘ if we don't have GMT ⇒

Fully connected model required to capture correlation

↳ latent variables can simplify our structure

• Structure

— Casual vs. non-casual ordering

↳ in BN the question is: do arrows show causality?

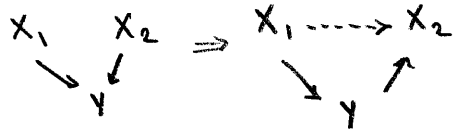
(X) → (Y) $\overset{?}{\Longrightarrow}$ causal connection between X and Y ↳ YES and NO!

↳ Any dist that I model this way ((X → Y)) I can equally model this way ((Y → X))

↳ we can reverse the edges and have a model that is equally expressive → for any distribution

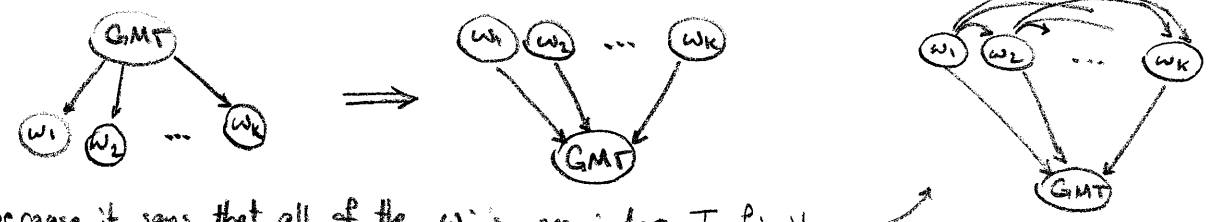
↳ nasty in case of v-structure

if we reverse edge $X_2 \to Y$ we should add extra link $X_1 \dashrightarrow X_2$



↳ causal directionality is often simpler

☑ Can we invert the edges? Is this the correct model?



No, because it says that all of the w_i's are indep. To fix it ...

↳ So, causal ordering although is not more correct than a non causal ordering, it's sparser

↳ Causal ordering → sparser, more intuitive, easier to parameterize

— Extending the Conversation

↳ usually we don't know how to construct distribution using variables $X_1 \dots X_n$.

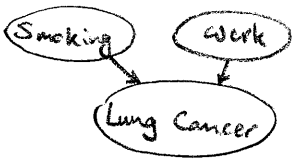
↳ we need to figure out how to encode them using a graph

↳ first step is to have set of variables that we wish to reason about

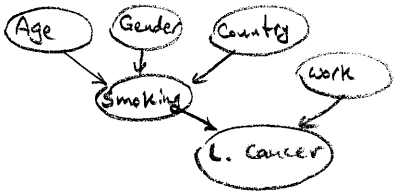
Lung Cancer

1.

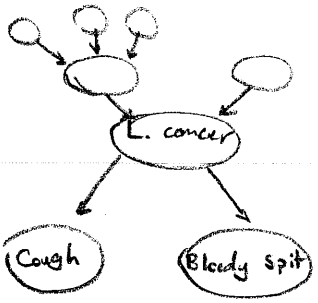
2. what influences weather somebody's going to get lung cancer?
   - ↳ ask doctor about reasons



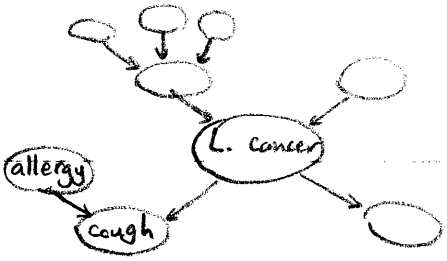
3. Continue questioning: what is the prob. of someone smokes?
   - ↳ Continue backward up to the point there's no way to extend the conversation
     - ↳ e.g. what is the prob. of Gender being male vs. female



4. factors that help us indicate weather someone has Lung cancer.



5. what is the Probability P(cough | lung cancer)?
   - ↳ extend the conversation backwards



- Parameters
  - Values
    - ↳ what matters?
      - ↳ Zeros : make big difference
        cause many of the mistakes → giving zero to unlikely but not impossible even
      - ↳ Orders of Magnitude : big difference between 0.1 and 0.01
      - ↳ Relative Values : relative values between conditional prob. make much bigger difference to th answer than the absolute prob.
        - ↳ compare values in each CPD is useful to evaluate the PGM
          - ↳ values used for relative ratios should make sense
  - Structured CPDs
    - ↳ CPD tables are rare → small application
    - ↳ context-specific: vars make a difference in some circumstance but not in others
    - ↳ aggregating of multiple weak influences

	Context Specific	Aggregating
Discrete	Tree CPD	Sigmoid CPD Noisy OR
Continous	Regression Tree CLG	Linear Gaussian

↳ regression tree : continuous version of tree CPD

↳ breaks up context based on some thresholds on continuous variables

↳ CLG : conditional linear Gaussian

- Iterative Refinement

↳ the model is rarely done the first time it is written → start ↷ test ↷ improve over time

– Model Testing

↳ ask queries, check results

– Sensitivity Analysis for parameters

↳ look at query and ask which parameters have the biggest difference on the value of the query

↳ those are probably the one we should fine tune

– Error Analysis

↳ add features → maybe help to eliminate certain types of error

↳ add dependency → give the model the ability to capture the structure

☑ For which of the following applications might a generative model be a better choice than discriminative model?

✗ making fast predictions of Y given X where the dist. of X is not important

✓ medical diagnosis where obtaining ground-truth labels for data is expensive

✗ image segmentation where the extracted features are high dimensional

✓ modeling different music composers for classification of music and creating new music pieces

↳ discriminative models directly model $P(Y|X)$ and avoid dealing with joint distribution

↳ generative models handles missing data or partially labled data better so they are preferable if obtaining lable is hard

↳ High-D variables are in general more difficult to model, and disc. models avoid this by conditioning on the

↳ generative models can be used to generate new observable data

* Inference :: Conditional Probability Queries

- set of observations $E = e$ → evidence

  a subset of variables Y → query

  compute $P(Y \mid E = e)$ → task

☑ applications

- medical / fault diagnosis → observations : symptoms, test results
  → task : predicting the probability of different failure modes / diseases
- Pedigree analysis → observations : phenotypes, genotypes of some individuals
  → task : predicting about other individuals

- NP-Hard
  - Inference in PGM is NP-Hard
    - ↳ if the problem is shown to be NP-Hard ⇒ it is extremely unlikely to have an efficient solution.
    - ↳ means that all algorithms that people have devised for this problem (and a bunch of problems that are equally hard) require at the very least time exponential in the size of representation of the problem
      - ↳ unlikely to be solvable for any problem that is bigger than small number of vars.
  - Which particular problems in PGM is NP-Hard?

1. Exact Inference : Given a PGM P_Φ
   a variable X and a value $x \in Val(X)$
   compute $P_\Phi(X = x)$

2. $P_\Phi(X = x) > 0$

ok! exact inference is NP-hard. but can we approximate it? NO!

3. Approximate Inference : Given a PGM P_Φ
   a variable X and a value $x \in Val(X)$
   and observation $e \in Val(E)$
   problem of finding an approximation of answer p that we guarantee

$$|P_\Phi(X = x \mid E = e) - p| < \varepsilon \quad , \quad \varepsilon < 0.5$$

↳ the case $\varepsilon = 0.5$ can be obtained by simply random guessing

↳ should we get depressed? this is a worst case result!

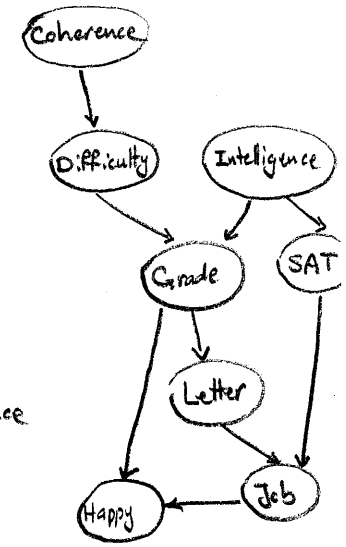
↳ we can construct this bizzare PGMs with exponential time at worst case but in general case one can't do better

↳ so we are looking at where to get better results ...

- Joint dist. defined by chain rule for BN.

$\Phi_C(C)\ \Phi_D(D)\ \Phi_I(I)\ \Phi_G(G,I,D)\ \Phi_S(S,I)\ \Phi_L(L,G)\ \Phi_J(J,L,S)\ \Phi_H(H,G,J)$

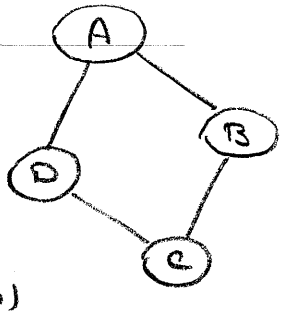
($\Phi_C(C)$ → $P(C)$; $\Phi_G(G,I,D)$ → $P(G|I,D)$)



- Transform a BN to a set of factors.
  - ↳ each CPD to a factor of variable and its parents
- goal : compute $P(J)$
  - ↳ $\sum_{C,D,I,G,S,L,H}$: marginalization
  - ↳ this is why the problem of inference in conditional prob. inference is called **Sum-Product** : sum over a product of factors
    - ↳ root of becoming interactable

- The same for MN

$\Phi_1(A,B) \cdot \Phi_2(B,C) \cdot \Phi_3(C,D) \cdot \Phi_4(A,D)$



goal : compute $P(D)$

- ↳ $\sum_{A,B,C} \Phi_1(A,B)\ \Phi_2(B,C)\ \Phi_3(C,D)\ \Phi_4(A,D)$
- ↳ but it is not quite right, because it's not equal to $P(A,B,C,D)$
  - ↳ it is unnormalized measure $\tilde{P}(A,B,C,D)$
    - ↳ should be devided by partition function

$\frac{1}{Z} \sum_{A,B,C} \Phi_1(A,B)\ \Phi_2(B,C)\ \Phi_3(C,D)\ \Phi_4(A,D)$

- ↳ how do we compute Z? to goal is $P(D)$ not Z!
  - ↳ $P(D) = \frac{1}{Z}\tilde{P}(D)$

so we calculate $\tilde{P}(D)$ and do renormalization

- Evidence : Reduced Factors
  - Only requires simple preprocessing step of factor reduction

$$P(Y \mid E=e) = \frac{P(Y, E=e)}{P(E=e)}$$

$W = \{X_1, \dots, X_n\} - Y - E$ → variables that are neithe query nor evidence

$$P(Y, E=e) = \sum_W P(Y, W, E=e)$$ → marginalizing out the W, the form of Sum-Product

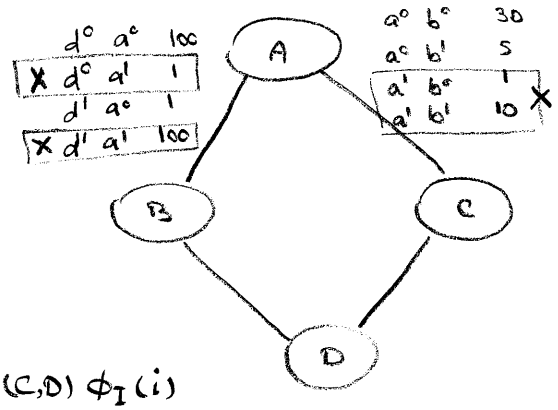
$$= \sum_W \frac{1}{Z} \prod_k \Phi_k(D_k, E=e)$$ → only the factors that are consistent with $E=e$

↳ reduce the factors by evidence

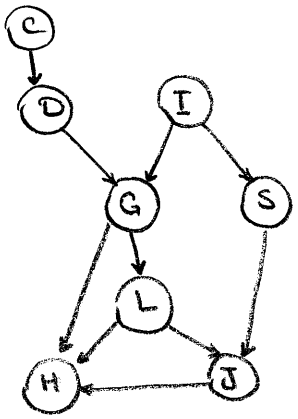
$$= \frac{1}{Z} \prod_k \Phi_i(D_i)$$

☑ $A = a^0$

after reducing the factors we can treat it exactly the same way we treat it before.



☑ $P(J, I=i, H=h)$



$P(J, I=i, H=h) \propto$

$$\sum_{C,D,I,G,S,L,H} \phi_C(C)\,\phi_D(C,D)\,\phi_I(i)$$

$$\phi_G(G,i,D)\,\phi_S(S,i)\,\phi_L(L,G)\cdot\phi(J,L,S)\,\phi_H(h,G,J$$

then renormalize it to have

$P(J \mid I=i, H=h)$

- Sum-Product Algorithm Summary

1. conditional prob ▷ ratio $\quad P(Y \mid E=e) = \dfrac{P(Y, E=e)}{P(E=e)}$

2. numerator = product of reduced factors $\quad P(Y, E=e) = \sum_W \frac{1}{Z} \prod_k \phi'_k(D'_k)$

3. denominator = numerator summed up over the vars and query vars Y. $\quad P(E=e) = \sum_Y \sum_W \frac{1}{Z} \prod_k \phi'_k(D'_k)$

↳ OR compute $\sum_W \prod_k \phi'_k(D'_k)$ and normalize at the end

- many algorithms for computing conditional prob. queries

↳ Push summation into factor products ⇒ Variable Elimination

↳ a special case of Dynamic Programming

↳ form of exact inference

↳ Message passing over a graph

↳ exct variations : Belief propagation

↳ variational approximations

↳ Random sampling instantiations

↳ Markov Chain Monte Carlo (MCMC), Importance Sampling

↳ approximation inference

- # of enteries in a table describing the full joint dist, where each of the n variables has k possible values?

○ n^k ○ n^2k ○ nk^2 ✓ k^n $\quad \underset{\text{var1}}{k} \; \underset{\text{var2}}{k} \; \underset{\text{var3}}{k} \cdots \underset{\text{var n}}{k} = k \times k \times \cdots \times k = k^n$

* Inference :: MAP Inference

- Maximum a Posteriori (MAP)

  Evidence : $E = e$

  Query : all other variables Y ($Y = \{X_1, \dots, X_n\} - E$)

  Task : map assignment $\quad MAP(Y|E=e) = \arg\max_y P(Y=y|E=e)$

  ↳ finding assignment y to variables Y that maximizes the conditional prob. of Y given evid

  ↳ the solution might not be unique → several different assignment that give exact prob.

☑ application

- message decoding : most likely transmitted message → assignment to the transmitted bits
- image segmentation : most likely segmentation

- MAP ≠ Max over Marginals

☑

P(A)	
a^0	0.4
a^1	0.6

P(B|A)	
$a^0 b^0$	0.1
$a^0 b^1$	0.9
$a^1 b^0$	0.5
$a^1 b^1$	0.5

P(A, B)	
$a^0 b^0$	0.04
$a^0 b^1$	0.36
$a^1 b^0$	0.3
$a^1 b^1$	0.3

⟹ → MAP assignment = highest probabili

look seperately at A and B ⟹ most likely assignment is a^1 X

we can't look separately at the marginals over A and over B

↳ we are looking for a single assignment over all of the variables that together has the highest prob.

- NP-Hard

  _ which problems are NP-Hard?

  1. Given a PGM P_Φ

     find a joint assignment x with highest probability $P_\Phi(x)$

  2. Given a PGM P_Φ

     and a probability p

     decide if there is an assignment x such that $P_\Phi(x) > p$

  ↳ Should we give up? NO!

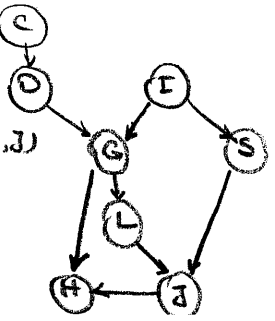
  ↳ we have even broader set of solutions compared to conditional prob. queries case

- Max-Product in BN

  ↳ now instead of marginalizing and summing out, we have max

$$\max_{C,D,I,G,S,L,J,H} \Phi_C(C)\,\Phi_D(D,C)\,\Phi_I(I)\,\Phi_G(G,I,D)\,\Phi_S(S,I)\,\Phi_L(L,G)\,\Phi_J(J,L,S)\,\Phi_H(H,G,J)$$

↳ max of product ~ Max-Product



- Max-Product Algorithm Summary

1. conditional probability ▷ ratio

$$P(Y \mid E=e) = \frac{P(Y, E=e)}{P(E=e)}, \quad Y = \{X_1, \ldots, X_n\} - E$$

↳ constant w.r.t Y

2. numerator = unnormalized measure

$$P(Y, E=e) = \frac{1}{Z} \prod_k \phi_k'(D_k')$$

↳ normalizing by partition function

3. find max assignment

$$\operatorname*{argmax}_Y P(Y \mid E=e)$$

constant w.r.t Y

OR compute $\operatorname*{argmax}_Y \prod_k \phi_k'(D_k')$

— many algorithms to compute MAP

↳ Push maximization into factor product ⇒ Variable Elimination (max Product)

↳ Message passing over a graph

↳ Max Product Belief Propagation

↳ It is an optimization problem : we can use specific algorithms for optimization

↳ Integer Programming : general class of optimization over discrete spaces

↳ state-of-the-art algs are based on this especially in approximate case

↳ for certain types of networks there are specific very efficient algorithms

↳ Graph Cuts

↳ Combinatorial Search

- Summary

— MAP : single coherent assignment of highest probability

↳ not the same as maximizing individual marginal probs.

— Maxing over factor product

— Combinatorial optimization problem

— Many exact and approximate solutions

✳ Inference :: Variable Elimination Algorithm

- simplest and most fundamental algorithm

☑ Elimination in Chains

$$P(E) \propto \sum_D \sum_C \sum_B \sum_A \tilde{P}(A,B,C,D,E)$$

$$= \sum_D \sum_C \sum_B \sum_A \Phi_1(A,B)\,\Phi_2(B,C)\,\Phi_3(C,D)\,\Phi_4(D,E) \quad \rightarrow \text{pairwise factors}$$

→ move factors that doesn't mention a particular var. in their scope out of summation over it.

$$= \sum_D \sum_C \sum_B \Phi_2(B,C)\,\Phi_3(C,D)\,\Phi_4(D,E) \underbrace{\sum_A \Phi_1(A,B)}$$

↳ summation over pairwise factor

↳ result is a factor over single variable B $\rightarrow \tau_1(B)$

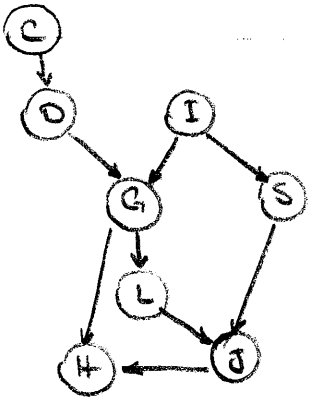
↳ eliminated A

$$= \sum_D \sum_C \Phi_3(C,D)\,\Phi_4(D,E) \underbrace{\left(\sum_B \Phi_2(B,C)\,\tau_1(B)\right)}$$

↳ $\tau_2(C)$ → eliminated B

$$= \cdots = \tau_5(E)$$

☑ Elimination in BN



$P(J) = ?$, eliminate C, D, I, G, L, S, H

$$\sum_{L,S,G,H,I,D,C} \Phi_J(J,L,S)\,\Phi_L(L,G)\,\Phi_S(S,I)\,\Phi_G(G,I,D)\,\Phi_H(H,G,J)\,\Phi_I(I)\,\Phi_D(C,D)\,\Phi_C(C)$$

$$\sum_{L,S,G,H,D,I} \Phi_J(J,L,S)\,\Phi_L(L,G)\,\Phi_S(S,I)\,\Phi_G(G,I,D)\,\Phi_H(H,G,J)\,\Phi_I(I) \underbrace{\sum_C \Phi_D(C,D)\,\Phi_C(C)}_{\tau_1(D)}$$

$$\sum_{L,S,G,H,D,I} \Phi_J(J,L,S)\,\Phi_L(L,G)\,\Phi_S(S,I)\,\Phi_G(G,I,D)\,\Phi_H(H,G,J)\,\Phi_I(I)\,\tau_1(D)$$

$$\sum_{L,S,G,H,I} \Phi_J(J,L,S)\,\Phi_L(L,G)\,\Phi_S(S,I)\,\Phi_H(H,G,J)\,\Phi_I(I) \underbrace{\sum_D \Phi_G(G,I,D)\,\tau_1(D)}_{\tau_2(G,I)}$$

$$\sum_{L,S,G,H} \Phi_J(J,L,S)\,\Phi_L(L,G)\,\Phi_H(H,G,J) \underbrace{\sum_I \Phi_S(S,I)\,\Phi_I(I)\,\tau_2(G,I)}_{\tau_3(G,S)}$$

$$\sum_{L,S,G} \Phi_J(J,L,S)\,\Phi_L(L,G)\,\tau_3(G,S) \underbrace{\sum_H \Phi_H(H,G,J)}_{\tau_4(G,J)}$$

$\Phi_H(H,G,J) = P(H|G,J) \rightarrow \sum_H = 1$

not every algorithm is clever enough to do this

$$\sum_{L,S} \Phi_J(J,L,S) \underbrace{\sum_G \Phi_L(L,G)\,\tau_3(G,S)\,\tau_4(G,J)}_{\tau_5(L,S,J)}$$

$$\sum_{L,S} \Phi_J(J,L,S) \cdot \tau_5(L,S,J) \quad \rightarrow \text{two factors} \rightarrow \text{multiply them}$$

☑ Variable Elimination with Evidence

$P(J, I=i, H=h)$: ? → eliminate C, D, G, S, L (no H, no I) → have single value

reduce the factors to correspond to scope

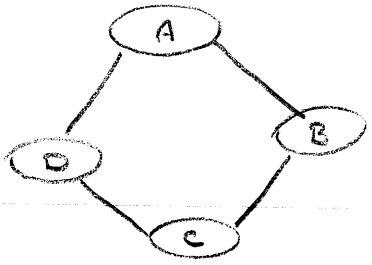
$$\sum_{L,S,G,D,C} \phi_J(J,L,S)\, \phi_L(L,G)\, \phi_S'(S)\, \underbrace{\phi_G'(G,D)}_{P(G \mid D, I=i)}\, \phi_H'(G,J)\, \underbrace{\phi_I'()}_{\phi_I(i)=cte}\, \phi_D(G,D)\, \phi_C(C)$$

elimination as before

renormalize it to have $P(J, I=i, H=h)$ by dividing it to $P(I=i, H=h)$

☑ Variable Elimination in MN

$P(D)$: ? → eliminate A, B, C



$$\sum_{A,B,C} \phi_1(A,B)\, \phi_2(B,C)\, \phi_3(C,D)\, \phi_4(A,D)$$

$$\sum_{BC} \phi_2(B,C)\, \phi_3(C,D) \underbrace{\sum_A \phi_1(A,B)\, \phi_4(A,D)}_{\tau_1(B,D)}$$

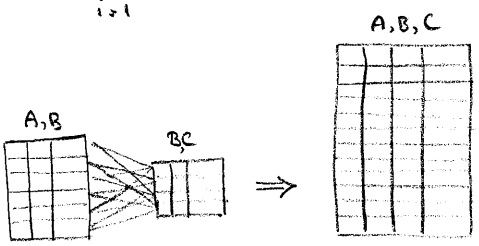
$$\sum_B \tau_1(B,D) \underbrace{\sum_C \phi_2(B,C)\, \phi_3(C,D)}_{\tau_2(B,D)}$$

$$\sum_B \tau_1(B,D)\, \tau_2(B,D) = \tau_3(D) = \tilde{P}(D) \rightarrow \text{renormalize}$$

- Routine : Eliminate var Z from Φ
  - look in Φ, and find Φ' which is all factors that involve Z — $\Phi' = \{\phi_i \in \Phi : Z \in Scope[\phi_i]\}$
  - multiply all of those factors — $\psi = \prod_{\phi_i \in \Phi'} \phi_i$
  - sum out the variable we want to eliminate, Z. — $\tau = \sum_Z \psi$
  - the factors in Φ' is used and we don't want to reuse them. but we create τ so introduce it. — $\Phi := \Phi - \Phi' \cup \{\tau\}$
- Variable Elimination Algorithm Summary
  - Reduce all factors by evidence
    - ↳ remove incosistant rows of CPD
    - ↳ get us a set of factors Φ
  - For each non-query variable Z
    - ↳ Eliminate var Z from Φ → removes $\Phi' \Rightarrow$ adds τ
  - Multiply all remaining factors
  - Renormalize to get a distribution

- VE Summary
  - Simple algorithm
  - Works for both BN and MN
  - Factor product and summation steps can be done in any order, subject to:
    - ↳ when $\mathcal{Z}$ is eliminated, all factors involving $\mathcal{Z}$ have been multiplied in → ensure correctness
- Complexity Analysis of VE
  - Basic operations in eliminating $\mathcal{Z}$
    - ↳ factor product

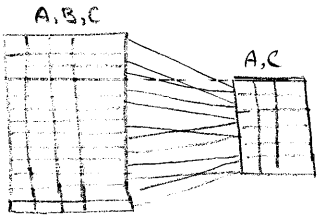
$$\psi_k(X_k) = \prod_{i=1}^{m_k} \phi_i$$



      - * every row of result = one row of each table
      - * $N_k = |Val(X_k)|$ → row of results
        - ↳ need to multiply m_k different factors
        - ↳ $m_k - 1$ product
      - * total cost : $(m_k - 1) N_k$ multiplications
    - ↳ factor marginalizations

$$\tau_k(X_k - \{\mathcal{Z}\}) = \sum_{\mathcal{Z}} \psi_k(X_k)$$

      - * every row of result = sum of all values of marginalized variable
      - * each number of original table is used exactly once
      - * $N_k = |Val(X_k)|$
      - * total cost : $\sim N_k$ additions → upper bound



  - Complexity of VE
    - ↳ start with m factors
      - ↳ $m \leq n$ for BN ← $m = n$ reduced by evidence; one factor for every variable = CPD
      - ↳ can be larger for MN → grid MN, fully connected MN
    - ↳ at each elimination step
      - ↳ get in some factors
      - ↳ generate exactly 1 factor
    - ↳ we have at most n elimination steps
      - ↳ worst case is when each elimination step use exactly 1 factor and generate 1
    - ↳ total number of factors that we ever produce : m^*
      - ↳ m^* < original factors + generated factors = $m + n$
    - ↳ size of the largest factor ever created N
      - ↳ $N = \max(N_i)$

↳ full calculation

↳ product operations : $\sum_k (m_k - 1) N_k$

each factor is multiplied in at most once

↳ as soon as we multiply it in, it goes away

↳ $\sum_k (m_k - 1) \leq$ # of factors $= m^*$

$$\sum_k (m_k - 1) N_k \leq N \sum_k (m_k - 1) \leq \underline{N \cdot m^*}$$

↳ sum operations : $\leq \sum_k N_k \leq N$ (# of elimination steps) $\leq \underline{N \cdot n}$

↳ total work is linear in N and m^*

↳ $N_k = |Val(X_k)| = O(d^{r_k})$ → total number of values in a factor

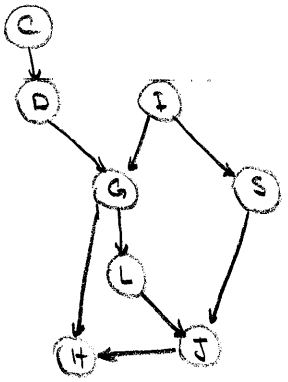
↳ $d = \max(|Val(X_i)|)$ → all values have maximum d values in their scope

binary $\Rightarrow d = 2$

↳ $r_k = |X_k|$ → cardinality of the scope of the k^{th} factor

↳ the source of exponential blow up

☑ Complexity Example



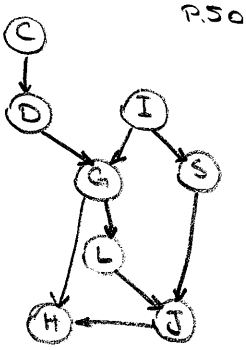
	# of vars in scope
$\tau_1(D) = \sum_C \phi_C(C)\,\phi_D(C,D)$	C,D — 2
$\tau_2(G,I) = \sum_D \phi_G(G,I,D)\,\tau_1(D)$	G,I,D — 3
$\tau_3(S,G) = \sum_I \phi_S(S,I)\,\phi_I(I)\,\tau_2(G,I)$	S,I,G — 3
$\tau_4(G,J) = \sum_H \phi_H(H,G,J)$	H,G,J — 3
$\tau_5(J,L,S) = \sum_G \phi_L(L,G)\,\tau_3(S,G)\,\tau_4(G,J)$	L,G,S,J — 4 → d
$\tau_6(J) = \sum_{L,S} \phi(J,L,S)\,\tau_5(J,L,S)$	JLS — 3

– Elimination Order

↳ variables can be eliminated in any order → if we multiply things correctly and correct time

↳ order can affect the complexity

☑ if we start elimination by G !



$\sum \Phi_J \Phi_L \Phi_S \Phi_G \Phi_H \Phi_I \Phi_D \Phi_C$

eliminate G

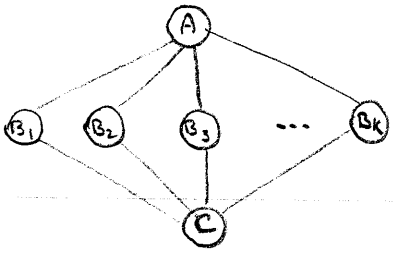
↳ which factors do we need to multiply to eliminate G?

$$\sum_G \underbrace{\Phi_L(L,G) \times \Phi_G(G,I,D) \times \Phi_H(H,G,I)}_{\Psi_1(L,G,I,D,H,J)}$$

6 variables → previously $d = 4$

↘ this order $d \geq 6$

☑



Eliminate A first

↳ factors involved $\Phi_1(A,B_1) \cdots \Phi_K(A,B_K)$

↳ scope of factor generated $\{A,B_1,\ldots,B_K\} \rightarrow K+1$

↳ size of the factor is exponential in K

Eliminate B_i first

↳ factors involved $\Phi_{A_i}(A,B_i) \times \Phi_{C_i}(C,B_i) \rightarrow \Psi_i(A,B_i,C) \rightarrow \tau_i(A,C)$

↳ scope of factor generated $\{A,B_i,C\} \rightarrow 3$

↳ size of the factor is exponential in 3

—Summary

↳ complexity of variable elimination linear in

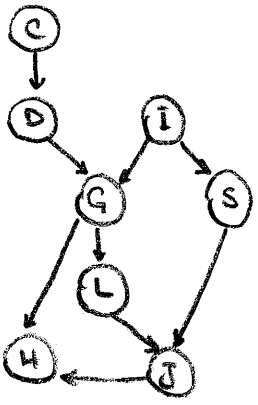
* size of model (# factors, # variables)
* size of largest factor generated

↳ size of factor is exponential in its scope

↳ complexity of algorithm depends heavily on elimination ordering

- Graph Based Perspective

☑

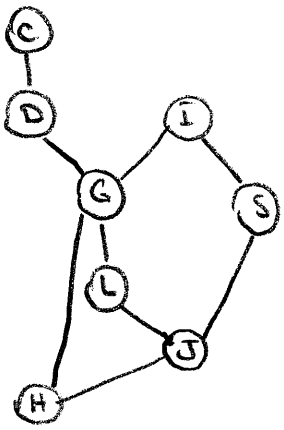


$\Phi_J(J,L,S) \times \Phi_L(L,G) \times \Phi_S(S,I) \times$

$\Phi_G(G,I,D) \times \Phi_H(H,G,J) \times \Phi_I(I) \times$

$\Phi_D(C,D) \times \Phi_C(C)$

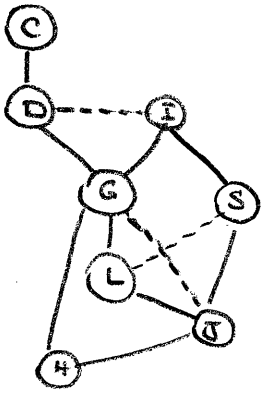
→ structure of graph that corresponds to this factors → make all edges undirected



that's not enough!

↳ $\Phi_G(G,I,D)$ so we need to add an edge

D-I to represent they are together in a factor



↳ $\Phi_J(J,L,S)$ → L-S → J's parents

↳ $\Phi_H(H,G,J)$ → G-J → H's parents

↳ for any v-structure we need to add an edge between parents

↳ called moralization : marrying the parents of child

↳ marry all of them no matter how many of them

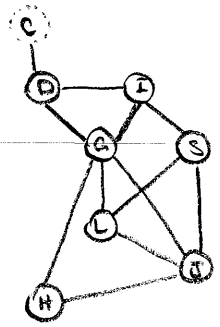
↳ the result is induced MN for the current set of factors

*eliminate: C

$$\tau_1(D) = \sum_C \Phi_C(C)\,\Phi_D(C,D)$$

$\Phi_I(I)$

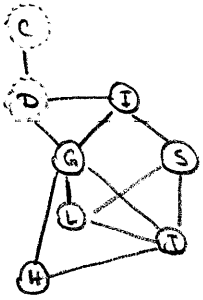
$$\Phi_J(J,L,S)\,\Phi_L(L,G)\,\Phi_S(S,I)\,\Phi_G(G,I,D)\,\Phi_H(H,G,J)\,\tau_1(D)$$



*eliminate: D

$$\tau_2(G,I) = \sum_D \Phi_G(G,I,D)\,\tau_1(D)$$

$$\Phi_J(J,L,S)\,\Phi_L(L,G)\,\Phi_S(S,I)\,\Phi_I(I)\,\Phi_H(H,G,J)\,\tau_2(D)$$



*eliminate : I

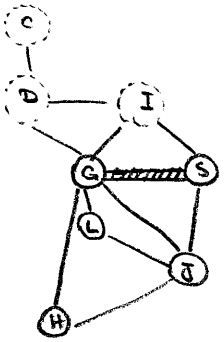
$$\tau_3(S,G) = \sum_I \Phi_S(S,I)\,\Phi_I(I)\,\tau_2(G,I)$$

↳ the factor that we introduced $\tau_3(S,G)$ doesn't have an edge in the graph

↳ we should introduce G-S edge in graph

↳ called fill edge : an edge we have to introduce because of elimination order

↳ all variables connected to I become connected directly

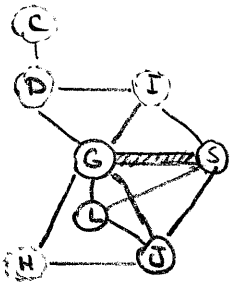


$$\Phi_J(J,L,S)\,\Phi_L(L,G)\,\Phi_H(H,G,J)\,\tau_3(S,G)$$

* eliminate : H

$$\tau_4(G,J) = \sum_H \Phi_H(H,G,J)$$

$$\Phi_J(J,L,S)\,\Phi_L(L,G)\,\tau_3(S,G)\,\tau_4(G,J)$$



* eliminate : G

* eliminate : L & S

this VE order only introduce only 1 fill edge

↳ conservative → pretty good elimination ordering

we can put all of those edges in a graph and call it **Induced Graph.**

– Induced Graph

induced graph $I_{\Phi,\alpha}$ over factors Φ and ordering α

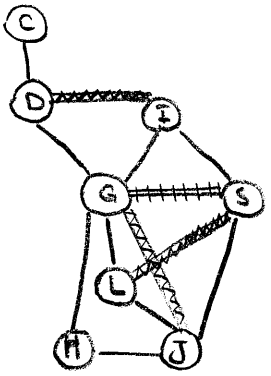
↳ undirected graph

↳ X_i and X_j are connected if they appear in the same factor in a run of VE algorithm using α as the ordering

↳ ▬ begin with

▬ moralization

▬ fill edge



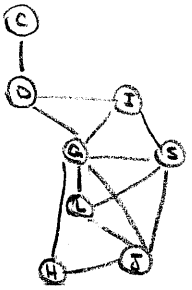
this ordering has 1 fill edge

– Cliques in the Induced tree

*Theorem: Every factor produced during VE is a clique in the induced graph

↳ clique: maximal fully connected subgraph e.g. DIG ✓

GLJ ✗ → fully connected, ↘ not maximal, ↳ we can add S



cliques	generated factors
DIG → ②	① $\tau_1(D) = \sum_C \Phi_C \Phi_D$
GIS → ③	② $\tau_2(G,I) = \sum_D \Phi_G \tau_1$
GLSJ → ⑤	③ $\tau_3(S,G) = \sum_I \Phi_S \Phi_I \tau_2$
CD → ①	④ $\tau_4(G,J) = \sum_H \Phi_H$
GHJ → ④	⑤ $\tau_5(L,J,S) = \sum_G \Phi_L \tau_3 \tau_4$
	⑥ $\tau_6(J) = \sum_{L,S} \Phi_J \tau_5$ → no clique

*Theorem: Every (maximal) clique in the induced graph is a factor produced during VE

e.g. GSLJ as a max clique

↳ one of the variables has to be the one who is eliminated first

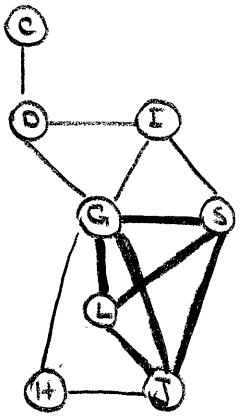
↳ once it eliminated it remove all corresponding edges

↳ so no new neighbors adds to it later

⇒ when eliminated it had all the neighbors that it had in clique.

⇒ it participated in factors with all other variables

⇒ when multiplied together, we have a factor over all of them



– Induced width

↳ the number of nodes in the largest clique in the graph − 1

↳ Minimal Induced width of a graph K is $\min$ (width $(I_{K,\alpha})$)

↳ provides a lower bound on best performance of VE to a model factorizing over K

- Summary

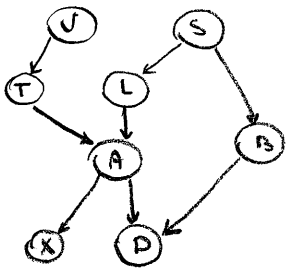
↳ VE can be viewed as transformations on undirected graph

↳ elimination connects all node's current neighbors ⇒ gives induced graph

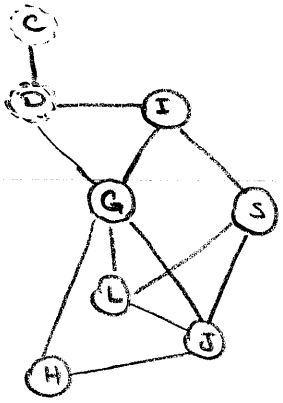
↳ cliques in resulting induced graph directly correspond to alg. complexity

☑ How many edges should be added to the graph to make it moralized?

= # of v-structures = 2



☑ what edge if any should be added to the graph when I is eliminated?



$I \rightarrow \phi(I, G, S) \xrightarrow{\text{elimination}} \tau(G, S) \rightarrow G - S$

• Finding Elimination Orderings

- How good of an elimination ordering can we construct?

Theorem: for a graph H, determining weather there exists an elimination ordering for H with induced width $\leq K$ is NP-complete

↳ this NP-hardness result is distinct from the NP-hardness result of inference.

↳ even given the optimal ordering, inference may still be exponential

- Greedy search using heuristic cost function

↳ at each point: eliminate node with smallest cost

↳ possible cost functions

↳ min-neighbors: # neighbors in current graph → produces smallest factor

↳ min-weight: weight (# values) of factors formed

☑ 2 var with 100 values each

5 var with binary values ✓

↳ these two look at the size of factor → ignores the fact that some of them are inevitable

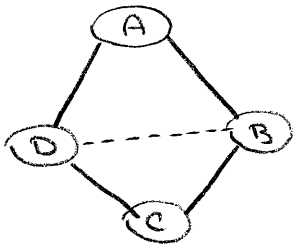
↳ min-fill: number of new fill edges → conting add complexity introduced by eliminating

↳ weighted min-fill: total weight of new fill edge

↳ edge weight = product of weights of the two nodes

practice

- Theorem : The induced graph is triangulated
  - ↳ regardless of elimination ordering
  - ↳ triangulated : no loop of length >3 with a "bridge"
  - ☑ proof via example



assume we have a loop of size $4 > 3$

assume that C is the first to be eliminated

↳ after eliminating C we end up introducing B-D edge

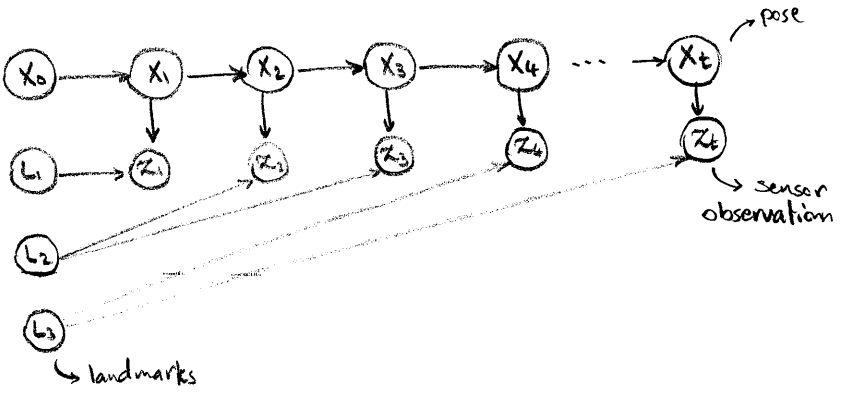
↳ fill edge

↳ we can find elimination ordering by finding a low-width triangulation of original graph H ☑

↳ using graph triangulation literature

☑ Robot Localization & Mapping

robot sees landmark and distinguish them uniquly → can measure the distance to each landmark

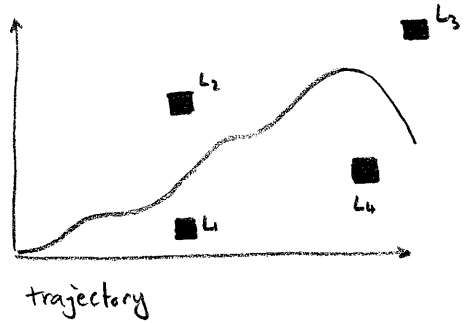


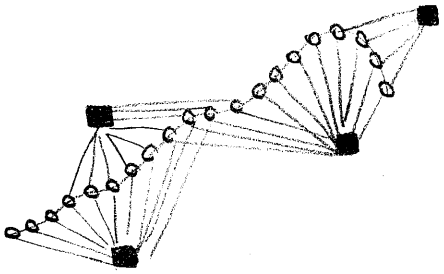
@ t_1 : robot sees L_1

@ t_2 and t_3 : robot sees L_2

@ t_4 : robot sees L_3

observed distance is a function of robot pose and landmark distance





- o robot pose at every time
- — edge between $p_t - p_{t+1}$
- ■ landmark positions

eliminate robot poses then landmarks ? → induced graph big spagetti:

↳ every landmark connected to all other landmarks

eliminate landmarks then poses ? → still densely connected

↳ but not as bad as the spagetti before

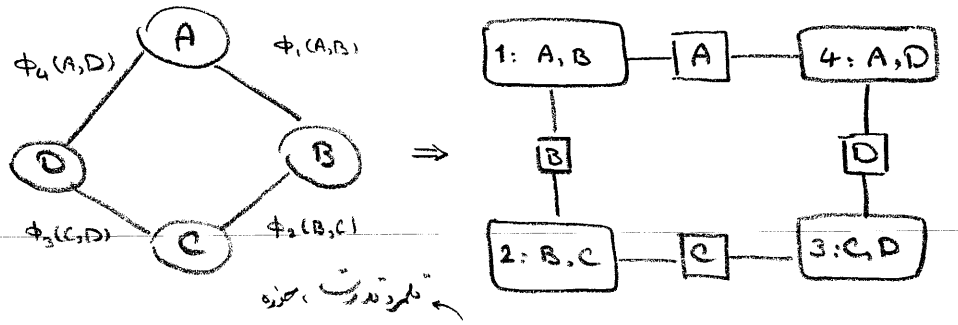
min-Fill elimination → very sparse

- Summary
  - ↳ finding the optimal elimination ordering is NP-hard
  - ↳ simple heuristics that try to keep induced graph small often provide reasonable performance

* Message Passing :: Belief Propagation

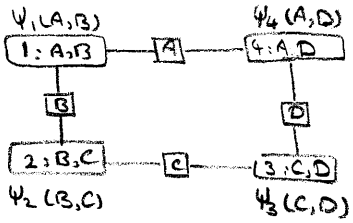
- Msg Passing class of algorithms
  - ↳ closely related to VE
  - ↳ more flexible → how to do summation and factor product steps
    - ↳ lower complexity compared even to minimal VE ordering

- Cluster Graph
  - a data structure to get bits of knowlege from a PGM

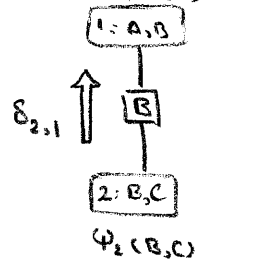


- Cluster 1 : a cluster whose jurisdiction of influence is pair of variables A, B
- clusters are going to talk to each other → convince each other that what they think a var. that they both have under jursdiction is correct
  - ↳ cluster 1 talk to cluster 2 about B → might tell C2 sth about B that it become more informed about dist. over B

- $\phi_1(A,B)$ → C1 : A,B
  $\phi_2(B,C)$ → C2 : B,C
  ...
- variables are communicating via messages.
- name the initial set of beliefs (= evidence that a cluster has over vars. in jurisdiction) → ψ
  - ↳ can be more complicated than single ϕ's.



- cluster 2 wants to send a msg to cluster 1
  - ↳ initially we are going to initialize with a totally uninformed msg.
    - ↳ $\delta_{2,1} = 1$ → from 2 to 1
    - ↳ they haven't even start to talk with each other
    - ↳ cluster 1 get that msg and multiply it with current thoughts about var. A and B
  - ↳ now that cluster 1 get some info, it's gonna send it to C4.
    - ↳ C4 doesn't care about B
      - ↳ $\delta_{1,4} = \sum_B \delta_{2,1}(B)\ \psi_1(A,B)$ → $\sum$ incoming msg × initial beliefs

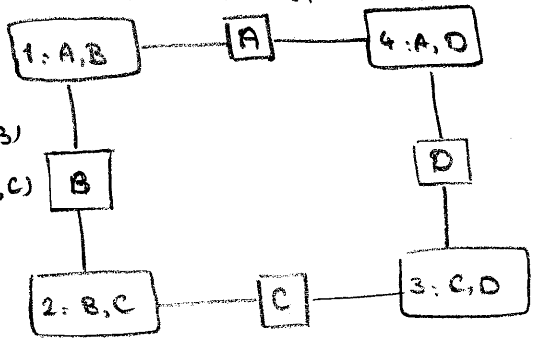


$\delta_{1,4} = \sum_B \delta_{2,1}(B)\ \psi_1(A,B)$

$\delta_{4,1} = \sum_D \delta_{3,4}(D)\ \psi_4(A,D)$

$\delta_{1,2} = \sum_A \delta_{4,1}(A)\ \psi_1(A,B)$

$\delta_{2,1} = \sum_C \delta_{3,2}(C)\ \psi_2(B,C)$



$\delta_{3,4} = \sum_C \delta_{2,3}(C)\ \psi_3(C,D)$

$\delta_{4,3} = \sum_A \delta_{1,4}(A)\ \psi_4(A,D)$

$\delta_{2,3} = \sum_B \delta_{1,2}(B)\ \psi_2(B,C)$

$\delta_{3,2} = \sum_D \delta_{4,3}(D)\ \psi_3(C,D)$

- each cluster send messages to its adjacent cluster that reflect this exact same process
  - ↳ the msg that C3 send to C4 doesn't take into consideration, the information that it got from C4 → خودرایی، خودخنکی؟ عجب مرد زرنگی!
    - ↳ avoid repeating back a rumor that you just told me or else we all become more convinced about the truth of this rumor
      - ↳ restrict our attention that come in from other sources. → communication protocol

- Cluste Graphs (ctd)
  - Undirected graph such that
    - ↳ nodes are clusters $C_i \subseteq \{X_1, \ldots, X_n\}$ → subsets of variables
    - ↳ edge between C_i and C_j associated withe sepset $S_{i,j} \subseteq C_i \cap C_j$ → variable they both know abo[ut]
      - ↳ the variables they choose to talk about
  - Given set of factors Φ

Initialize the model by giving each cluster a certain amount of information

↳ each of initial factors ϕ_k is going to be assigned to one and only one cluster $C_{\alpha(k)}$

- ↳ it needs to be at least one → to be taken to account
- ↳ shouldn't be more than one → avoid double counting the evidence

Assign ϕ_i to $C_{\alpha(k)}$ → $\text{Scope}[\phi_k] \subseteq C_{\alpha(k)}$

↳ where do we put the information corresponding to factor k?

↳ we can only put it in a cluster that understands every single var. in factor

Define initial beliefs

↳ product of all factors that are assigned to it

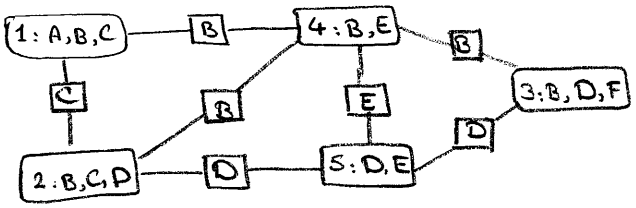
$$\psi_i(C_i) = \prod_{k:\alpha(k)=i} \phi_k$$

↳ some cluster have just one factor → $\psi = \phi$

some have several factors → multiply those factors → total informed belief of cluster

some have no factor → null product ...

$\phi_1(A,B,C)$, $\phi_2(B,C)$, $\phi_3(B,D)$, $\phi_4(D,E)$,
$\phi_5(B,E)$, $\phi_6(B,D)$, $\phi_7(B,D,F)$



find for each factor a cluster in which to put it:

$\phi_1(A,B,C)$ → only one choice → C1 → $\alpha(1)=1$

$\phi_2(B,C)$ → two choices: C1, C2 → let's choose C2 → $\alpha(2)=2$

$\phi_3(B,D)$ → two choices: C2, C3 → let's choose C2 → $\alpha(3)=2$

$\phi_4(D,E)$ → one choice → C5 → $\alpha(4)=5$

$\phi_5(B,E)$ → one choice → C4 → $\alpha(5)=4$

$\phi_6(B,D)$ → two choices: C2, C3 → let's choose C3 → $\alpha(6)=3$

$\phi_7(B,D,F)$ → only one choice → C3 → $\alpha(7)=3$

there are other alternative assignments

↳ $\psi_2 = \phi_2(B,C) \times \phi_3(B,D)$

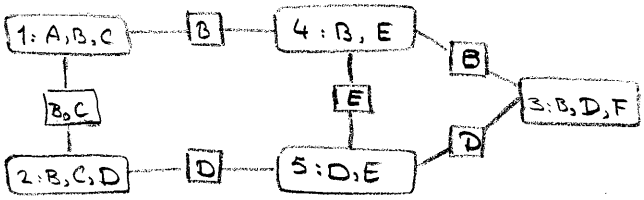
$\psi_1 = \phi_1(A,B,C)$

$\psi_3 = \phi_6(B,D) \times \phi_7(B,D,F)$

for alternative assignments:

$\psi_1 = \phi_1$
$\psi_2 = \phi_2\,\phi_3\,\phi_6$
$\psi_3 = \phi_7$
$\psi_4 = \phi_5$
$\psi_5 = \phi_4$

☑ Same factors, different cluster Graph



clusters haven't changed

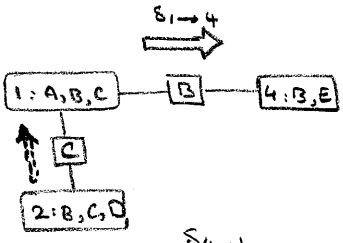
only sepsets between them changed

↳ e.g. C1: A,B,C & C2: B,C,D → sepset above: C
→ sepset now: B,C

↳ e.g. C2: B,C,D × C4: B,E → sepset above: B
→ sepset now: ∅

- Message Passing

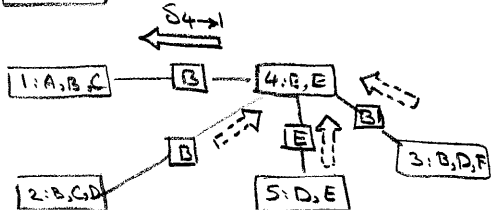
$$\delta_{1\to4}(B) = \sum_{A,C} \psi_1(A,B,C)\ \delta_{2\to1}(C)$$



$$\delta_{4\to1}(B) = \sum_{E} \psi_4(B,E) \times \delta_{2\to4}(B) \times \delta_{5\to4}(E) \times \delta_{3\to4}(B)$$

current most informed beliefs about B & E

↳ message from c1 itself was not used.



$$\delta_{i \to j}(S_{ij}) = \sum_{C_i - S_{ij}} \psi_i \times \prod_{k \in (N_i - \{j\})} \delta_{k \to j}$$

- from i
- to j
- sepset
- the vars cluster j doesn't know about
- factors initially assigned to cluster i
- all of the incoming messages other than the one from j

- Belief Propagation Algorithm
  - ↳ clusters propagates informed beliefs together

— Algorithm

I. Assign each factor $\phi_k \in \Phi$ to a cluster $C_{\alpha(k)}$

II. Construct initial potentials $\psi_i(C_i) = \prod_{k: \alpha(k) = i} \phi_k$

III. Initialize all messages to be 1

VI. Repeat

(i) select edge (i,j) and pass message $\delta_{i \to j}(S_{ij}) = \sum_{C_i - S_{ij}} \psi_i \times \prod_{k \in (N_i - \{j\})} \delta_{k \to i}$

V. Compute final belief $\beta_i(C_i) = \psi_i \times \sum_{k \in N_i} \delta_{k \to i}$

↳ at the end of process a cluster needs to know what to believe about the variable that it understands

↳ takes its own initial beliefs and all of the information it got from all of neighbors

↳ produce beliefs (β_i)

— Repeat until when?

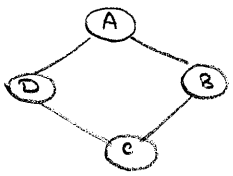
— How do we select edges to pass messages along?

↳ Prespecified order? → round robin

↳ there are better strategies

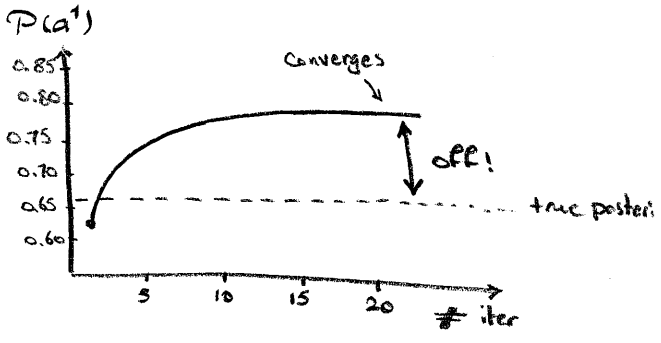
— Is this algorithm good? Yes & No → بستگی دارد!

☑

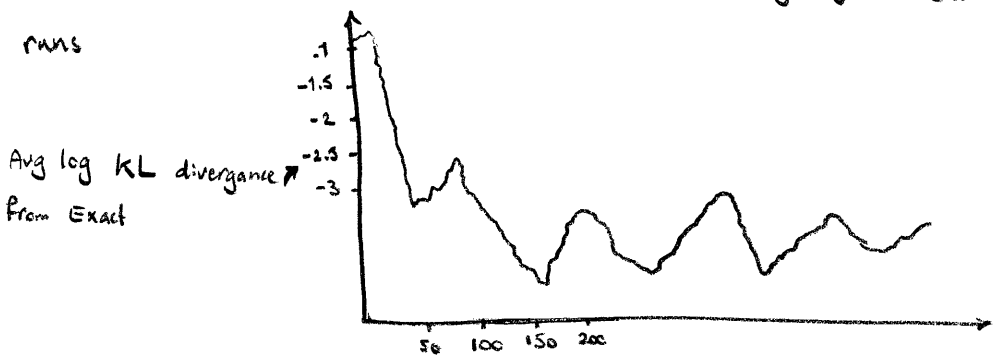


convergance achieved but the result is a little bit off!

↳ the answer is not exact

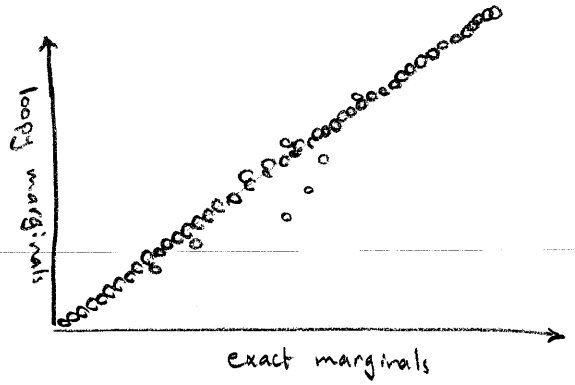


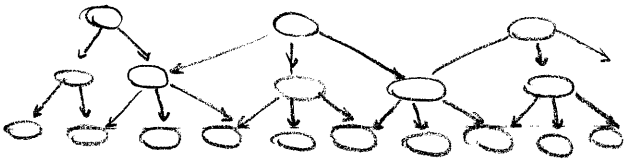
↳ In general this is an approximate algorithm → no free lunch (NP-hard)
↳ no easy way to solve it
↳ have exceptions
↳ different BP runs



- So why bother using?

↳ it missbehave in loops
☑ pyramid network
↳ happens in image analysis
↳ compute exact marginals
compute loopy BP
with a few outliers the results are the same





- Summary
↳ Graph of clusters connected by sepsets
↳ Adjacent clusters pass information to each other about variables in sepsets
↳ The msg from i to j summarizes everything i knows, except information obtained from j.
↳ Algorithm may not converge
↳ may have oscilatory behavior
↳ The resulting belief are psudo marginals → in theory
↳ but actually performs well in practical applications

• Cluster graph properties
- In order to → support reasonable message passing

Reminder: Undirected graph → nodes : clusters $C_i \subseteq \{X_1, \dots, X_n\}$
↘ edges : sepsets $S_{i,j} \subseteq C_i \cap C_j$

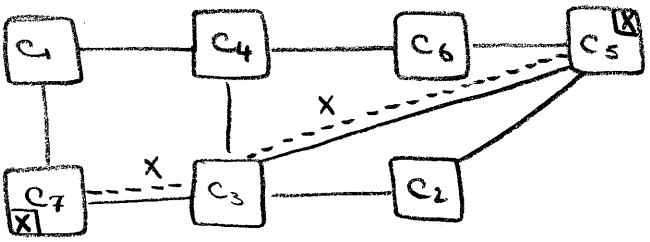
- Family Preservation Property
↳ given a set of factors Φ
we need to be able to assign each factor ϕ_k to some cluster $C_{\alpha(k)}$
such that the cluster can accomodate ϕ_k ($Scope[\phi_k] \subseteq C_{\alpha(k)}$)
↳ we need an apropriate cluster in cluster graph
↳ for each factor $\phi_k \in \Phi$, there exists a cluster C_i such that $Scope[\phi_k] \subseteq C_i$

- Running Intersection Property

↳ let's assume that we have a pair of clusters C_i and C_j, and a variable X that belongs to bot

e.g. X in C_5 & C_7

the property says that there exist a unique path between C_5 & C_7 for which all clusters an sepsets that is along the path contains X



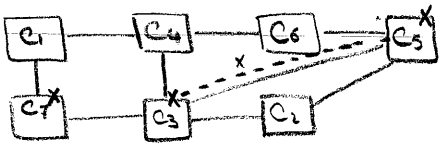
e.g. if the path is $C_7 - C_3 - C_5$ there should be X all along the way

↳ for each pair of clusters C_i, C_j and variable $X \in C_i \cap C_j$ there exist a unique path between C_i and C_j for which all clusters and sepsets contain X

↳ existance ? uniqueness?

↳ existance proof : assume that there is no path ⇒ no way

for C_7 to communicate C_3 about the variable X. ⇒ seperated communities each know some information about X and they can never talk to eachother about X ⇒ never going to get agree about X

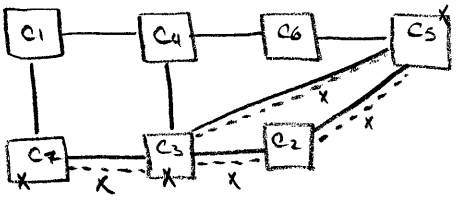


↳ uniqueness proof: assume that there are two paths involved X

↳ trace msg passing algorithm

$C_3 \xrightarrow{X} C_5$ eg. I suggest $X = x^1$

$C_5 \xrightarrow[\delta_{35}]{X} C_2$



$C_2 \xrightarrow[\delta_{53}]{X} C_3$ the C_3 suggestion about $X = x^1$ reinforces ⟶ prob ↑ ⟶ again to loop $C_3 - C_5 - C_2$

↳ self-reinforcing loop

↳ skewed prob. in many examples

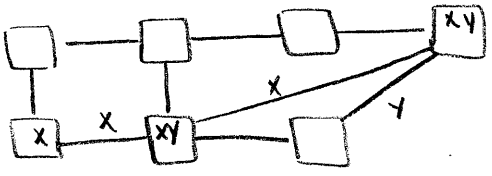
↳ we should avoid that ⟶ only reduce the problem not eliminate it

☑ X and Y are strongly correlated

$C_3 \xrightarrow{X} C_5$

C_5 translates that to information about Y

$C_5 \xrightarrow{Y} C_2 \xrightarrow{Y} C_3$ reinforced!



↳ BP does very poorly when we have strong correlations

↳ The more skew probabilities are, the harder for BP to get the results

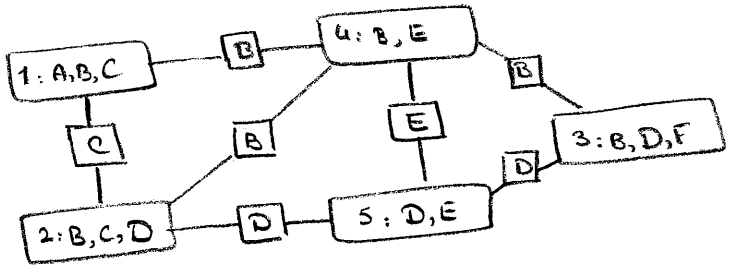
↳ equivalently: for any X, the set of clusters and sepsets containing X form a tree

↳ has to be connected = existance of the path

↳ tree, no loop = uniqueness

↳ each variable induce its own little tree across which info about that var. flows in the graph

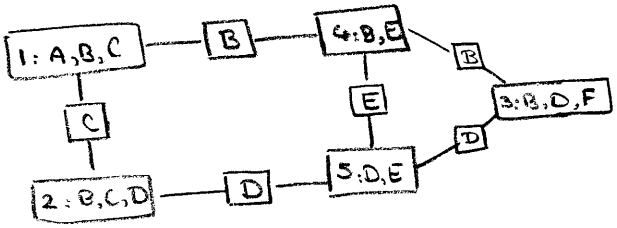
☑



Var B

⇒ tree → running intersection

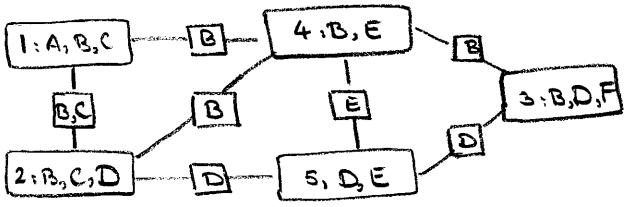
☑ Illegal cluster graph



⇒ isolated C2 → no running intersection

(violated existance)

☑ Illegal cluster graph



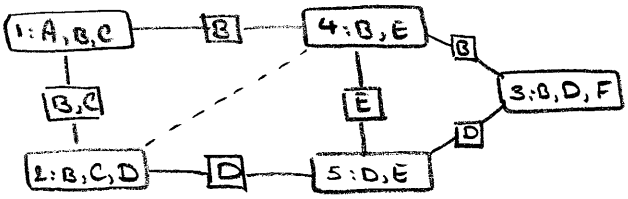
⇒ loop → no running intersection

(violated uniqueness)

if we want to take the cluster graph and still allow communication between B & C

↳ e.g. want C1 & C2 to talk about B and C

↳ just eliminate one edge



all variables satisfies RIP

B ✓

D ✓

E ✓

↳ how do we construct a cluster graph that has desired properties?

↳ one very simple and in some ways degerate version → Bethe Cluster graph

- **Bethe Cluster Graph**

↳ simple → often used

↳ from statistical physics: people use this kind of approximation to energy functions

↳ degenerate

↳ two types of clusters

↳ big clusters : correspond to factors in Φ

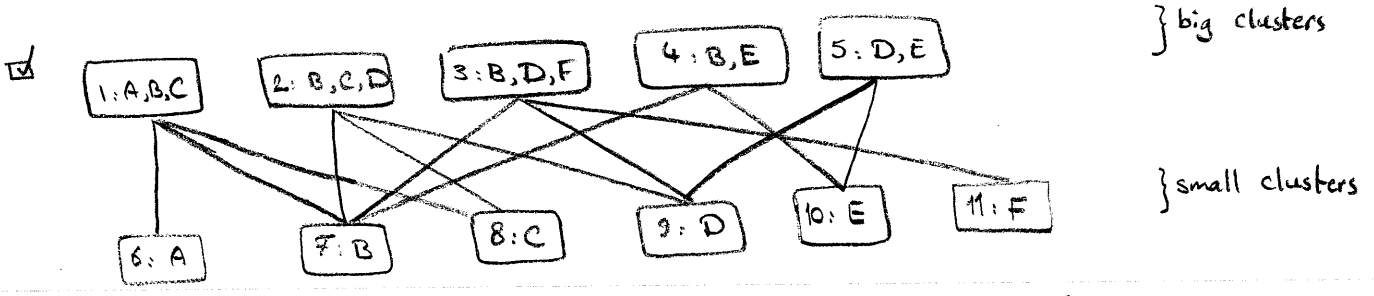
for each $\phi_k \in \Phi \Rightarrow$ a factor cluster $C_k = \text{Scope}[\phi_k]$

↳ little clusters : correspond to individual variables

for each $X_i \rightarrow$ a singleton cluster

↳ connect C_k to X_i exactly when X_i is a member of C_k

edge $C_k - X_i$ if $X_i \in C_k$



:(ABC is connected to A, B and C → how the msgs are passed

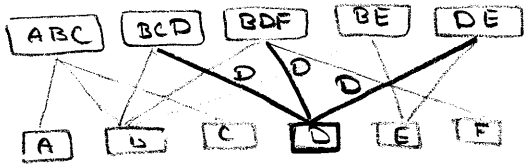
↳ degenerate : it only allows info about singletons to be passed

↳ in every msg passing step : loses any info about the correlation between variables

:) simple to construct, guaranteed RIP

↳ why RIP is guaranteed in Bethe Cluster Graph

☑ in the above graph, any variable involve var D, sepsets have only D. It's a tree because D doesn't appear on any other sepsets except these ones



— Summary

↳ Cluster graph must satisfy two properties

↳ family preservation : allow Φ to be encoded

↳ running intersection : connects all info about any var, but without feedback loops

↳ Bethe Cluster Graph is often first default

↳ Richer cluster graph structure can offer different tradeoffs wrt computational cost and preservation of dependencies

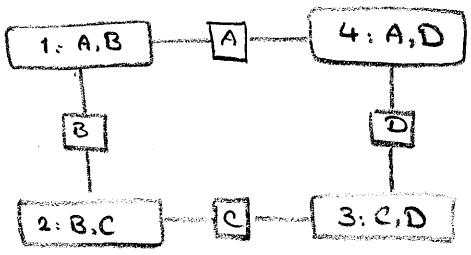
↳ the amount of the sizes of the messages that are passed grow more expensive BUT allow us to improve the preservation of the dependencies as msgs are passed

↳ keep more info

- Properties of BP Algorithm

— Calibration

☑ cluster beliefs $\beta_i(C_i) = \psi_i \times \prod_{k \times N_i} \delta_{k \to i}$



$\beta_1(A,B) = \psi_1(A,B) \times \delta_{4 \to 1}(A) \times \delta_{2 \to 1}(B)$

↳ a cluster graph is calibrated if every pair of adjacent clusters C_i, C_j agree on their sepset $S_{i,j}$

↳ if we ask C1 what it thinks about var B : $\beta_1(A,B)$

if we ask C2 what it thinks about var B : $\beta_2(B,C)$

they should agree → $\sum_A \beta_1(A,B) = \sum_C \beta_2(B,C)$

$$\sum_{C_i - S_{i,j}} \beta_i(C_i) = \sum_{C_j - S_{i,j}} \beta_j(C_j)$$

↳ convergence of BP algorithm implies calibration.

Convergance ▶ Calibration

Convergence of BP : msg from next time step equals to previous one.

↳ $\delta_{i \to j}(S_{i,j}) = \delta'_{i \to j}(S_{i,j})$ ①

$$\delta'_{i \to j}(S_{i,j}) = \sum_{C_i - S_{i,j}} \left(\psi_i \times \prod_{k \in (N_i - \{j\})} \delta_{k \to i} \right) \quad ②$$

$k \in (N_i - \{j\})$ → msgs except incoming from j

$$\beta_i(C_i) = \psi_i \times \prod_{k \in N_i} \delta_{k \to i} \quad ③$$

$k \in N_i$ → all msgs

(2,3) → $\delta'_{i \to j}(S_{i,j}) = \sum_{C_i - S_{i,j}} \frac{\beta_i(C_i)}{\delta_{j \to i}(S_{i,j})} \overset{①}{=} \delta_{i \to j}(S_{i,j})$

$\delta_{i \to j}(S_{i,j}) \times \delta_{j \to i}(S_{i,j}) = \sum_{C_i - S_{i,j}} \beta_i(C_i)$ ④

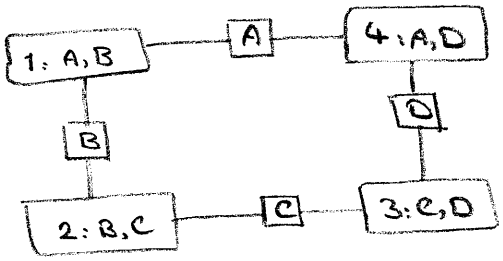
equally → $\delta_{i \to j}(S_{i,j}) \times \delta_{j \to i}(S_{i,j}) = \sum_{C_j - S_{i,j}} \beta_j(C_j)$ ⑤

(4,5) → $\sum_{C_i - S_{i,j}} \beta_i(C_i) = \sum_{C_j - S_{i,j}} \beta_j(C_j)$ ✓

↳ sepset beliefs $\mu_{i,j}(S_{i,j}) = \delta_{j \to i}\, \delta_{i \to j} = \sum_{C_j - S_{i,j}} \beta_j(C_j)$

– Reparameterization

☑ $\beta_i(C_i) = \psi_i \times \sum_{k \in N_i} \delta_{k \to i}$ $\quad \mu_{i,j}(S_{i,j}) = \delta_{j \to i}\, \delta_{i \to j}$



$\beta_1(A,B) = \psi_1 \times \delta_{4 \to 1} \times \delta_{2 \to 1}$ $\quad \mu_{1,2} = \delta_{1 \to 2} \times \delta_{2 \to 2}$

$\beta_2(B,C) = \psi_2 \times \delta_{1 \to 2} \times \delta_{3 \to 2}$ $\quad \mu_{2,3} = \delta_{2 \to 3} \times \delta_{3 \to 2}$

$\beta_3(C,D) = \psi_3 \times \delta_{2 \to 3} \times \delta_{4 \to 3}$ $\quad \mu_{3,4} = \delta_{3 \to 4} \times \delta_{4 \to 3}$

$\beta_4(A,D) = \psi_4 \times \delta_{1 \to 4} \times \delta_{3 \to 4}$ $\quad \mu_{1,4} = \delta_{1 \to 4} \times \delta_{4 \to 1}$

↳ lots of repetition → $\delta_{4 \to 1}$ in β_1 and $\mu_{1,4}$

↘ $\delta_{1 \to 2}$ in β_2 and $\mu_{1,2}$

⋮

⟹ every δ appears exactly twice
one in cluster belief
one in sepset belief

$$\frac{\prod_i \beta_i}{\prod_{i,j} \mu_{i,j}} = \frac{\prod_i \psi_i \prod_{j \in N_i} \delta_{i \to j}}{\prod_{i,j} \delta_{i \to j}} = \prod_i \psi_i = \tilde{P}_\Phi(X_1, \dots, X_n)$$

↳ $\tilde{P}_\Phi(X_1, \dots, X_n) = \frac{\prod \beta_i}{\prod_{i,j} \mu_{i,j}}$ → shown with different sets of parameters

↳ no information loss as the result of BP

– Summary

↳ At Convergance of BP, cluster graph beliefs are calibrated:

↳ beliefs at adjacent clusters agree on sepsets

↳ Cluster graph beliefs are an alternative, calibrated parameterization of the original unnormalized density

↳ we can read a variable from any clique in which it appears

↳ no info is lost by msg passing

☑ T or F? at convergance:

– the beliefs of neighboring clusters agree on all variables → F

– the belifs of neighboring clusters agree on all variables in the sepset → T

– " " " " " " " " " " they have in common → F

– " " " " " " " " " " in either cluster → F

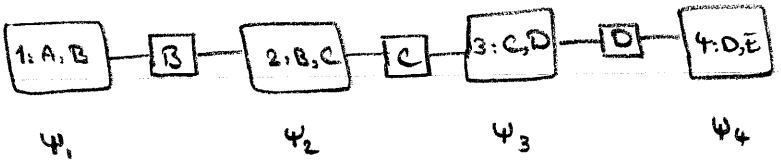
* Message Passing :: Clique Tree Algorithm

- BP → over a fairly general DS called cluster graph

Clique Tree Alg → over a special DS called Clique tree
↘ special case of BP
↳ clique tree is a special case of cluster graph
↳ brings stronger performance guarantee to BP
↳ faster, convergance to the exact answer \o/

☑ Passing Msg in a tree

Pairwise MRF



pass msg from the left :

$$\delta_{1\to 2}(B) = \sum_A \psi_1$$

$$\delta_{2\to 3}(C) = \sum_B \psi_2 \times \delta_{1\to 2}$$

$$\delta_{3\to 4}(D) = \sum_C \psi_3 \times \delta_{2\to 3}$$

pass on the other direction :

$$\delta_{4\to 3}(D) = \sum_E \psi_4$$

$$\delta_{3\to 2}(C) = \sum_D \psi_3 \times \delta_{4\to 3}$$

$$\delta_{2\to 1}(B) = \sum_C \psi_2 \times \delta_{3\to 2}$$

regardless of when the msg get passed there is no other msg that C4 would multiply in when sending msg to C3.
↳ no msg recieved
↳ got a msg from C3 itself

- Correctness
  - Surprisingly relevant to what we've seen before the resulting beliefs are correct

☑ $\beta_3(C,D) = \psi_3 \times \delta_{2\to 3} \times \delta_{4\to 3}$

$$= \psi_3 \times \left(\sum_B \psi_2 \times \delta_{1\to 2}\right) \times \left(\sum_E \psi_4\right)$$

$$= \psi_3 \times \left(\sum_B \psi_2 \times \sum_A \psi_1\right) \times \left(\sum_E \psi_4\right)$$

we have a product of all the network $\psi_{1\dots 4}$ + we have a summation over all vars not in C

⇒ product of factors marginalized out unneccessary variables ⇝ same as VE
↳ legal order of operations
↳ correct result

- Clique Tree

— Undirected tree

↳ nodes : clusters $C_i \subseteq \{X_1, \dots, X_n\}$

↳ edges : sepsets $C_i - C_j \Rightarrow S_{i,j} = C_i \cap C_j$

- in cluster graphs sepsets could be a subset of $C_i \cap C_j$
  but in clique tree sepset should be equal to $C_i \cap C_j$

— Family Preservation

↳ Given set of factors Φ
assign each ϕ_k to a cluster $C_{\alpha(k)}$ such that $Scope[\phi_k] \subseteq C_{\alpha(k)}$ somewhere

↳ each factor should go to a cluster → each piece of information should be represeted

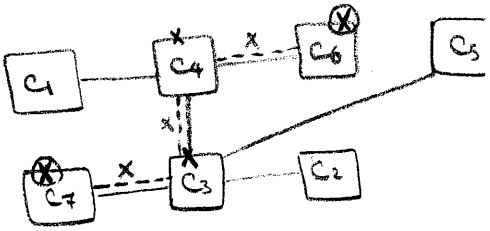
↳ For each factor $\phi_k \in \Phi$
there exist a cluster C_i such that $Scope[\phi_k] \subseteq C_i$

↳ there must exist one cluster whose scope is big enough to accomodate that factor

— Running Intersection Property

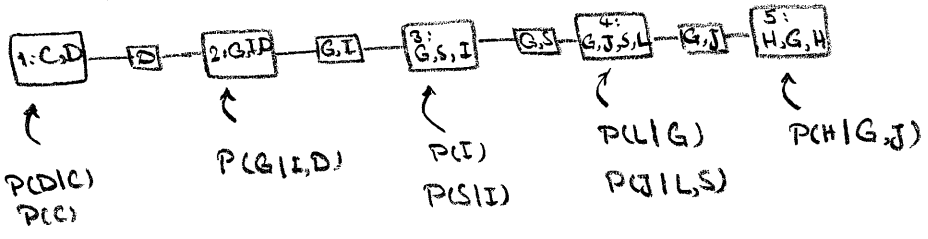
↳ For each pair on clusters C_i, C_j → as variants of cluster graph
and variable $X \in C_i \cap C_j$
there exist a unique path between C_i and C_j
for which all clusters and sepsets contain X

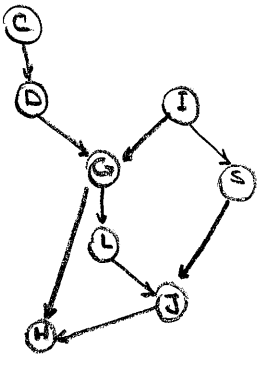
☑ C_6 and C_7
it's a tree → only one path between C_6 and C_7
so X has to be all over the path



↳ (clique tree version)
For each pair of clusters C_i, C_j
and variable $X \in C_i \cap C_j$
in the unique path between C_i and C_j
all clusters and sepsets contain X

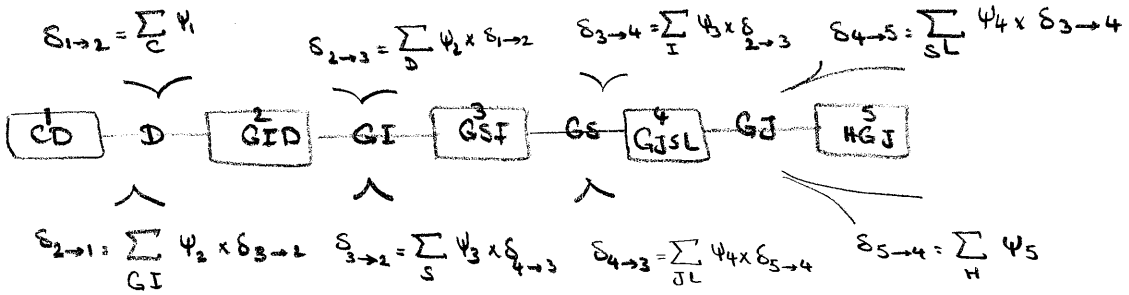
☑ more complicate clique tree





because of Family preservation property ⟶ each factor can be assigned to some clique

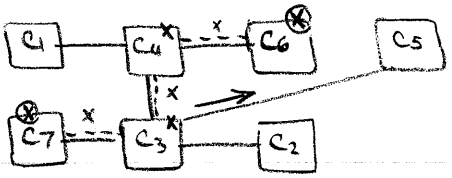
tree satisfies RIP ⟶ G : $C_2 \sim C_5$



- RIP ⇒ Clique Tree Correctness

↳ we want to prove that if we eliminated X while passing msg from C_i to C_j, then we have multiplied in all factors that involve X.

☑ let's imagine that X is eliminated while passing msg from C_3 to C_5.



X cannot be in C_5, because it is not in the sepset $S_{3,5}$. ⟶ if $X \in C_5 \overset{R.I.P}{\Longrightarrow} X \in S_{3,5} \Rightarrow$ X wouldn't be eliminated

↳ so the RIP relates the structure of the graph to the time that those variable are eliminated.

☑

$$\beta_3(G,S,I) = \psi_3 \times \delta_{2\to3} \times \delta_{4\to3}$$

$$= \psi_3 \times \sum_C \psi_1 \times \sum_D \psi_2 \times \sum_{J,L} \psi_4 \times \sum_H \psi_5$$

$$= \sum_{C,D,J,L,H} \psi_1 \times \psi_2 \times \psi_3 \times \psi_4 \times \psi_5$$

↳ when we eliminate one variable, we multiply all of factors that involve the variable

- Summary

↳ BP can be run over a tree-structured cluster graph

↳ In this can, computation is a variant of variable elimination

↳ The resulting beliefs are guaranteed to be correct marginals of the unnormalized density

- Computation of Clique Tree Alg

↳ exploit tree structure to improve behavior of alg

☑

$\delta_{1\to2}(B) = \sum_A \psi_1$ $\quad \delta_{2\to3}(C) = \sum_B \psi_2 \times \delta_{1\to2}$ $\quad \delta_{3\to4}(D) = \sum_C \psi_3 \times \delta_{2\to3}$

$\delta_{2\to1}(B) = \sum_C \psi_2 \times \delta_{3\to2}$ $\quad \delta_{3\to2}(C) = \sum_D \psi_3 \times \delta_{4\to3}$ $\quad \delta_{4\to3}(D) = \sum_E \psi_4$

↳ observation

$\delta_{1\to 2}$: once computed never changes ⟶ instant convergence

$\delta_{2\to 3}$: depend one when it is passed

↳ if passed first before C_2 get msg from C_1 then after getting that msg changes

↳ if C_2 is clever and it wait until $\delta_{1\to 2}$ is passed it converges and never changes

$\delta_{3\to 4}$: needs to wait longer

↳ if it wait for $\delta_{1\to 2}$ then $\delta_{2\to 3}$ now can fix $\delta_{3\to 4}$ with one computation

from right-to-left it happens the same : $\delta_{4\to 3}$ converge instantly then $\delta_{3\to 2}$ then $\delta_{2\to 1}$

∴ so we can compute all the msgs in one pass in either direction ltr or rtl.

↳ for chains it has its own name : Forward-Backward Algorithm

↳ commonly used for HMMs

↳ to total effort that we had to compute all of these msgs is just one pass in each direction

↳ once all msgs have been passed we have beliefs that are marginals of $\tilde{P}_\Phi(C_i)$

↳ Convergance : once C_i recieved a final msg from all of its neighbors except C_j (if it waits enough to recieve msgs from everyone else and then send it to j) then that msg $\delta_{i\to j}$ is final

↳ Can you always find a msg passing order to achieve that goal?

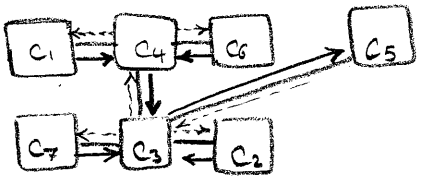
↳ is there any deadlock situations ? = all ppl waiting for somebody else to send'em msg.

↳ msgs from leaves are immediately final ⟶ then the parent of that ⟶ ...

↳ because it is a tree it is guaranteed to find a legal msg-passing order

↳ if we pass msg in correct order (= from leaves inward) only need to pass $2(K-1)$ msgs (K = # of clusters ⟶ $K-1$ edges × 2 msg for each edge)

☑ message passing order I



⟶ forward

⟵ backward

we can start with any leaf . say C_2

↳ C_3 can't pass : waiting

↳ we had to go for another leaf . say C_1

↳ C_3 & C_4 can't pass : waiting

↳ another leaf : C_6

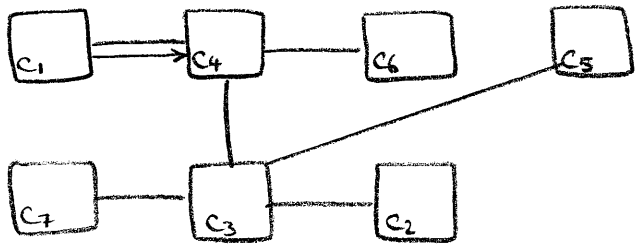
↳ now we have the option of activating C_4

↳ another leaf : C_7

↳ C_3 can be activated

at this point every one recieve all the msgs but one

↳ another direction: $C_5 \to C_3 \to C_2 \to C_7 \to C_4 \to C_6 \to C_1$



$C_1 \rightarrow C_4$

$C_4 \rightarrow C_3$ x illegal

↳ does not have all the info it needs to pass a final msg.

↳ still waiting for C_6

— Answering Query

↳ clique tree have good properties → make it a useful DS

↘ which kind of queries does it answer?

↳ Posterior distribution queries on variables that apear together in a clique

↳ find any clique that contains all of those variablels

sum out irrelevant variables from that clique → results in $\tilde{P}_\Phi$

renormalize them

↳ Introducing new evidence $\mathcal{Z}=z$ and querying X

↳ a form of incremental inference : you already calibrated clique tree and say wait a minute! I made another observation

↳ clique trees are good for that

↳ Case 1 : If X appears in clique with $\mathcal{Z}$

$X, \mathcal{Z} \in \phi_i$ → what the posterior X would be if I further observe $\mathcal{Z}$

↳ $\tilde{P}_\Phi(\mathcal{Z}, X, \dots)$

↳ we can condition or reduce the clique

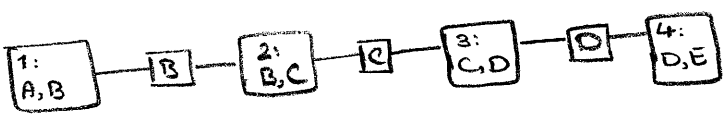
↳ just restricting the attention to the enteries that are consistent with evidence $\mathcal{Z}$

$= \phi_i \times \mathbb{1}(\mathcal{Z}=z) = \tilde{P}_\Phi(\mathcal{Z}=z, X, \dots)$

sum out irrelevant variables

renormalize

↳ Case 2 : If X does not share a clique with $\mathcal{Z}$



observe $A=a$

query D

↳ reduction of factors including $\mathcal{Z}$

↳ multiply by $\mathbb{1}(\mathcal{Z}=z)$

Observe $A=a$ → $\psi_1 = \psi_1 \times \mathbb{1}(A=a)$ → removing all enteries inconsistent with $A=a$

what happens when we change ψ_1?

↳ strategy 1 : recompute everything from new → all msgs and beliefs

↳ strategy 2. look which msgs would change

ψ_1 change → $\delta_{1\to2}$ change → $\delta_{2\to3}$ change → $\delta_{3\to4}$ change but since we are interested in C_3 we don't care

$\delta_{4\to3}$ doesn't change → $\delta_{3\to2}$ and $\delta_{2\to1}$ doesn't change

↳ we can reuse at least half of messages since they don't change

↳ the only msgs that we need to recompute are the msgs that are along the path to clique containing X

— Summary

↳ In clique tree with k cliques, if messages are passed starting at leaves, 2(k-1) msgs suffic to compute all beliefs

↳ we get posterior over every single var in the model

↳ contrast that with VE complexity and it only gives posterior over 1 var in model

↳ Can compute marginals over all variables at only twice the cost of VE

↳ all marginals → VE : # var x O(VE)

↳ clique tree : 2 x O(VE)

↳ By storing msgs, inference can be reused in incremental queries.

• Clique tree & Independence

— clique tree alg properties → guarantee we achieve the correct marginals at every single clique

↳ marginals agree with each other

↳ marginals can be computed using a single upward and downward pass

↳ Inference in PGM is NP-hard → there should be a catch somewhere.

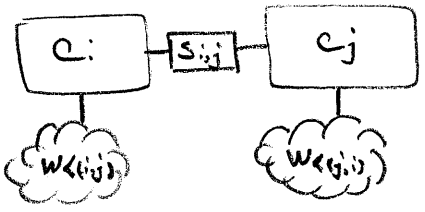
↳ a computational cost that occurs at least in certain cases → where is it?

— For an edge (i,j) in T let break tree in 3 parts:

↳ $W_{<i,j>}$ = all vars that appear only on C_i side of T

↳ $W_{<j,i>}$ = all vars that appear only on C_j side of T

↳ variables on both sides are in the sepset S_{ij}



_Theorem : T satisfies RIP if and only if for every (i,j)

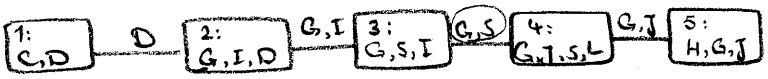
$$P_\Phi \models (W_{<(i,j)} \perp W_{<(j,i)} \mid S_{i,j})$$

variables on left side of edge are independent of the variables on the right side of the edge given the variables on the sepset → sepset separate the left side from the right side.

↳ RIP is the critical property to prove correctness of clique tree alg

↳ so RIP is important to have all those nice properties

☑ focus on sepset $S_{3,4}$ (G,S)



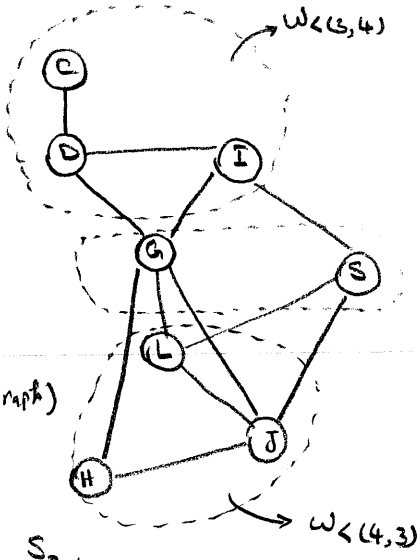
$W_{<(3,4)} = \{I, D, C\}$

$W_{>(4,3)} = \{J, L, H\}$

$S_{3,4}$ (G,S)

looking at the induced graph

(BN CPDs → factors in MN → induced graph)

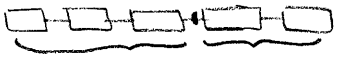


there are no path between $W_{<(3,4)}$ and $W_{<(4,3)}$ that are go through $S_{3,4}$

↳ C, I, D separated from H, L, J given G, S

↳ $P_\Phi \models (\{C, I, D\} \perp \{J, L, H\} \mid \{G, S\})$ ✓

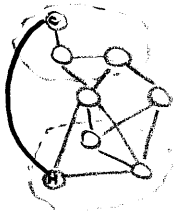
↳ Proof by contradiction: let's assume that vars on left side are not independent of vars on right side, given sepst.



⇓

there needs to be path in the induced MN between L-side and R-side. that doesn't go through $S_{i,j}$ $W_{<(i,j)}$ $W_{<(j,i)}$

⇓



$\Rightarrow \phi$ (C,H)

there has to be an edge where one node in one side, and one node in the other

⇓

there needs to be a factor ϕ, that involves those two variables

⇓

ϕ(C,H) → C_1(C,D,H)

↓

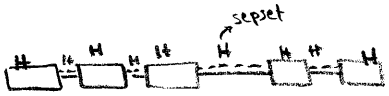
left side

because of family preservation, that factors should sit in one of the cliques.

⇓

that clique is either on the L-side or R-side.

⇓



now we have variable H in one clique in L-side and in one-clique in R-side

⇓

R.I.P tells that H should be everywhere in between and specifically in sepset

⇓

violation of the assumption that H is not in sepset

— Implications

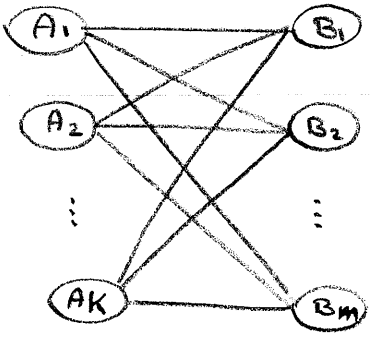
↳ where are computational implications? what we conclude from the fact that sepset divide the graph into conditionally indep. pieces?

↳ Each sepset needs to separate graph into two conditionally independent pieces.

↳ In many graphs that implies a certain minimal complexity → sometimes quite large

↳ Case 1: complete bipartite graph → two sets of vars
→ edges only cross the sets

plate models for students course difficulty & student intelligence



↳ what is the smallest sepset that we could construct?

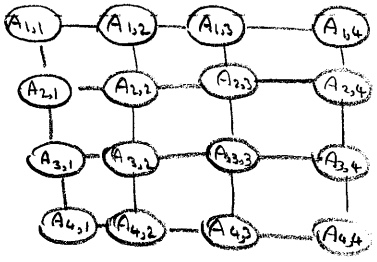
↳ we need to seperate graph into 2 conditionally indep pieces

↳ smallest sepset that we could construct that actually break up the graph into meaningful pieces is all of var on A side or all the variables on B side

↳ smallest sepset size $\geq \min(K, m)$

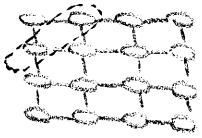
↳ Case 2: grid

↳ how to break up graph into separate conditionally indep pieces



↳ we can construct clique trees to have smaller sepsets
optimal tree → smallest

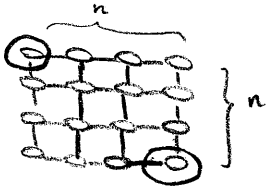
e.g.



→ leaves a very large everything else

↳ if we want to break the grid such that $A_{1,1}$ appears in one side and $A_{4,4}$ in another

↳ any clique tree satisfying this separation must have a sepset of size $\geq n$



↳ breaking up in other ways doesn't make it any better

↳ In these cases we could put a lower bound on the size of the sepset that is required for running clique tree inference

↳ this is where we pay exponential blow-up

— Summary

↳ Correctness of clique tree inference relies on running intersection property

↳ RIP implies separation in original dist.

↳ Implies minimal complexity incurred by any clique tree

↳ (as seen before) Related to width of minimal induced graph

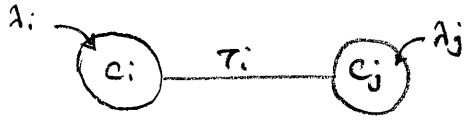
induced width talks about cliques but it is related to sepsets too

☑ Consider the optimal clique tree for a pairwise MN structured as nxn grid. what is the size (# var)

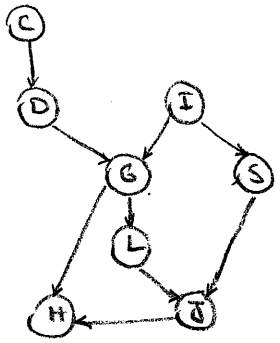
of the largest sepset in the clique tree?

→ smallest sepset possible
→ sepset that separates two parts yet to be smallest $\Rightarrow O(n)$

- Clique tree and VE
  - VE → a way of proving clique tree alg. correctness
    - ↘ help answering an important question: where clique trees come from?
      - ↳ or the we might construct a little clique tree?
  - Variable Elimination
    - I. each step create a large factor λ_i through factor product
    - II. a variable is eliminated in λ_i to generate new factor τ_i
    - III. τ_i is used in computing other factors λ_j
      - ↳ each generated τ_i will be used in context of computing another large factor
    - ↳ Let's think about this process as passing msgs
      - * Intermediate factors λ_i are cliques
      - * τ_i's are messages generated by clique λ_i and transmitted to another clique λ_j
    - ↳ VE defines a graph
      - ↳ cluster C_i for each factor λ_i used in computation
      - ↳ draw edge $C_i - C_j$ if the factor generated from λ_i is used in the computation of λ_j

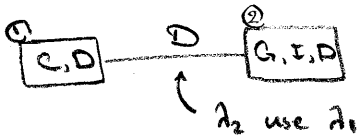


☑ $\phi_C(C)\,\phi_D(C,D)\,\phi_I(I)\,\phi_G(D,I,G)\,\phi_S(S,I)\,\phi_L(S,L,G)$
$\phi_H(G,J,H)\,\phi_J(J,L,S)$



① eliminate C: $\tau_1(D) = \sum_C \underbrace{\phi_C(C)\,\phi_D(C,D)}_{\lambda_1(C,D)}$

② eliminate D: $\tau_2(G,I) = \sum_D \underbrace{\phi_G(G,I,D)\,\tau_1(D)}_{\lambda_2(G,I,D)}$



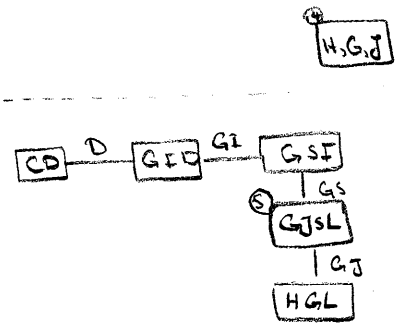
③ eliminate I: $\tau_3(G,S) = \sum_I \underbrace{\phi_I(I)\,\phi_S(S,I)\,\tau_2(G,I)}_{\lambda_3(G,I,S)}$

④ eliminate H : $\tau_4(G,J) = \sum_H \underbrace{\phi_H(H,G,J)}_{\lambda_4(H,G,J)}$

↳ didn't use any factor produced by other factors so far

⑤ eliminate G : $\tau_5(J,L,S) = \sum_G \underbrace{\phi_L(L,G)\,\tau_3(G,S)\,\tau_4(G,J)}_{\lambda_5(G,J,S,L)}$

↳ use factors that produced in two separate computations



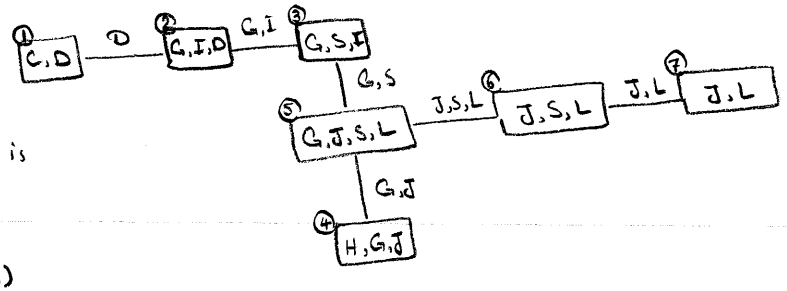
⑥ eliminate S : $\tau_6(J,L) = \sum_S \underbrace{\phi_J(J,L,S)\ \tau_5(J,L,S)}_{\lambda_6(J,L,S)}$

⑦ eliminate L : $\tau_7(J) = \sum_L \underbrace{\tau_6(J,L)}_{\lambda_7}$

↳ this clique tree is bigger than previous ones

↳ reason: we have 3 adjacent cliques where one is subset of another.

$$C_7(J,L) \subset C_6(J,S,L) \subset C_5(J,G,S,L)$$



↳ require post processing step: remove redundant cliques

↳ those whose scope is a subset of adjacent clique's scope