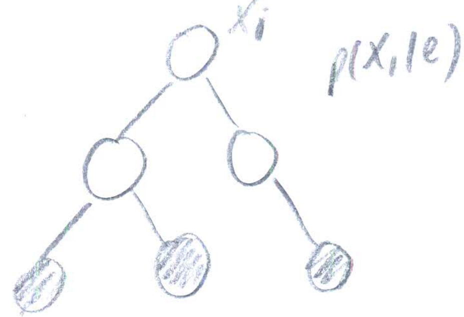
10-708 - L4 04/05/2020

L4 - exact inference, variable elimination - You have notes already, record intuitions

- EX: 3 lectures on GM representation
- EX: Focus on inference; then learning (which uses inference as a subroutine)

Query 1 - likelihood
- casino - quantit. specification of probability of fishy sequence of dice outcome - estimation/likelihood.
- marginal probability of evidence, likelihood
- marginalise over r.v.s. whose observations you do not have

Query 2 - C.P.
- use previous query as subsolution
- A posterior belief
- don't query all; interested in subset of hidden r.v.s.
- marginalise out hidden variables which we are not interested in



Applications of post. belief

Prediction: $p(C|A,B) = p(C|B)$

as $C \perp A | B$

Diagnosis: $p(A|B,C) = p(A|B)$

as $C \perp A|B$.

- systematic way of using PGMs for potentially large GMs.
- e.g. social network
- Reduces necessity to deal with entire network due to C.I. properties

---

- DBN

- optimisation semantics; representation learning/embedding is also an inference problem.

layers may correspond to different granularities of features. can be viewed as hidden r.v.s.

Query 3 - MPA (maximum a posterior config. given evidence)
or (most probable assignment)

- Again previous query is sub soln.

Ⓠ: recall the distinctions made by JP for HMM

- which y gives highest C.P. mass

- application - classification, explanation
- PGM may yield be useful in situations where simple linear class not ~~good~~ approp.

(*) Nuances

- MPA answer depends on framing

i) $\arg\max_{y_1} p(y_1, y_2)$ $y_1^* = 1$ (expected value?)

ii) $\arg\max_{y_1, y_2} p(y_1, y_2)$

$y_1^* = 0$ $y_2^* = 0$

- MPA is different depending on whether there is context in the form of y_2
- PGM makes this explicit → joint label or separate?
- EX: y_1, y_2 connected / in Markov Blanket → 'use context'

y_1, y_2 d-separated → 'no need for context'

Complexity of Inference

EX. Proof of NP results → signpost of how to allocate your time

- computing $P(X = x | e)$ - NP hard
in GM

- no assumptions about graphical models → no. of configuration increases exponentially.

(*) Avoid answer for any subset → need to enumerate (without help of GM)

Ex: Hardness does not mean not soluble

- 10-708 - In many cases, certain graph structures offer polynomial time solution methods, or approximate with poly. complexity

Ex: class focuses on these generic (not special) cases ; tradeoff between rich graphical models and comput. feasibility

Ex: deep learning paradigm → no thinking about algorithm ; not luxury in GM.

Approaches to inference

- Exact inference - guaranteed theoretically to get exact answer entailed by model
- Approx inference - approx 'true answer' ; more heavily used practically

Ex: many deep learning methods/model architectures may be good ; but inference algorithm not → hamper performance

---

Marginalisation / Elimination. - trivial statistically

- Q: Likelihood of protein E being active (binary state)
- P(e)
- marginalisation
- Q: How expensive is marginalisation ; exponential⁴ (?) A: clarity

- Computation difficult with very long chain (trad. summation, enum; non PGM)
- use chain decomp (PGM) :-

- Two strategies:

1) Exponential cost - Hold node together, 2^4 configurations ; change only a record values of joint probability node (?) A: clarity complexity: $O(k^n)$ (naive)

P(e) =

2)(*) enumeration savings $\rightarrow$ not all c.p.s. function of a

(*) Note $\sum_a p(a)p(b|a) = \phi(b) = p(b)$ (i.e. a function of b)

k - no. of states
n - no. of r.v.s.

- more systematic method than 1)
- Repeat until we have $p(e) = \sum_d p(e|d)p(d)$
- one off summation has cost(k^2) cost of e.g. $\sum_a p(a)p(b|a)$ is $|b| \times |a|$ i.e. 4 (as a = 2 states) (quadratic)
- ~~H~~ for n eliminations $\rightarrow O(nk^2)$ (quadratic complexity in lrgest config of nodes k, linear in no. of nodes)

- GM structure gives opportunities

HMM

- C.P. :- $p(y_i|x_1, \dots, x_T) = \sum_{\{y\}_{t=1}^T \setminus y_i} \cdots \sum p(y_1, \dots, y_T, x_1, \dots, x_T)$ $\rightarrow$ prob hidden state given entire seq.

- Via fact law :$\rightarrow$ How does this affect complexity of inference

EX: Illustrates inference complex. reduction for HMM (W A3 - Key juncture for full underst.)

(*) $\underbrace{\sum_{y_2}\sum_{y_3} \cdots \sum_{y_T}}_{\text{excl. } y_i \text{ (query)}} \cdots \sum_{y_1} \underbrace{p(y_1)p(x_1|y_1)p(y_2|y_1)}_{m(x_1, y_2) = p(x_1, y_2)}$ (?) W A3 - check

(*) $\sum_{y_3} \cdots \sum_{y_T} \cdots \sum_{y_2} \underbrace{p(x_2|y_2)p(y_3|y_2)f(x_1,y_2)}_{m(x_1,x_2,y_3)}$

- Repeat
- algorithm is linear in no. of nodes, quadratic in no. of states
- This is the HMM forward algorithm (elimination on a chain) simple as

- HMM - different elimination (sequence)

(*) $\sum_{y_1} \cdots \sum_{y_T} \underbrace{p(x_T|y_T)p(y_T|y_{T-1})}_{= n(x_T, y_{T-1}) = p(x_T|y_{T-1})}$

- Ultimately: - $n' = p(x_{T:T}|y_{T-1})$ ...

(h) (15)

- Backward algorithm: eliminating nodes from the tail.

- n' takes semantics
  of condit. prob. of partial sequence of observed given 1 hidden state.
  n term (prob. of late ½ of sequence and given latent states)

- n term takes semantics: -
  of joint prob of 1st half of sequence given n hidden states.

(h) (15): Have to solidify understanding of HMM forward-backward

- Major invention of the 70s (figuring it out algebraically not trivial)
- ordering the nodes for elimination → highly specialised c.i. insight.
- PGMS allow determination of poss. / feasibility via simplified alg.

- undirected chains } chain models have recurring pattern

- CRFS

(*)
- Sum-product operation

$\sum_{z} \prod_{\phi \in F} \phi$ — F - set of factors, ϕ - factors

- Ex: A key vehicle for understanding complexity for inference on GMs:-
  (*) count no. of summations, multiplications

(*) Appl. of sum-product to GMS beyond chains

(*) Query: $P(X_1, e) = \sum_{x_n} \cdots \sum_{x_2} \prod_i P(x_i|pa_i)$ write

(*) (define an ordering of summation signs) with this ordering:

Iteratively: i) Move irrelevant terms outside innermost sum

ii) insert new term into product

iii) insert new term → prod.

ultimately :- $p(X_1|\underline{e}) = \dfrac{\phi(X_1,\underline{e})}{\sum_{x_1} \phi(X_1,\underline{e})}$

Outcome of Elimin

Ex: Factors ϕ are general (local marginal / local cond / potential / intermediate eg $m(\cdot), n(\cdot)$)

- Ex: each factor has scope ( )

- Queries, variables tbe.

Dealing with evidence

- evidence - non random variables (observed / clamped)
- programming / implementation → reduce no. of operations via evidence potentials.

(*) Incorporate evidence into gm. via sum-product rule.

(*) Total evidence potential is merely a product; treat as addit. set of factors in GM representation

$$\tau(\underline{Y},\bar{e}) = \sum_{z,e} \prod_{\phi \in F} \phi \times \partial(E,\bar{e})$$

- Elimin algorithm (W) (AB) (Q) ✓✓

PGM - visual prompts at suitable ordering

F

∂
∂
ϕ
ϕ
ϕ
ϕ

- "stack of factors"

(*) S-P-V-E

Ex: Input ($\mathcal{F}, \underline{Z}, \prec$)

stack of factors

variables to be elimin

ordering of $\underline{Z}$

- sum-product-variable-elim

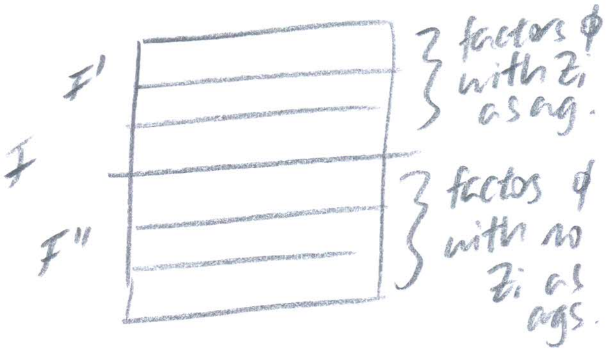
(*) eliminate one r.v. Z_i from set $\mathcal{F}$

- repeat above subroutine continuously until all variables Z_i eliminated

(*) sum-Product-Eliminate var.

$\mathcal{F}$ // set of factors (queue of potentials)

Z // variables to be eliminated



(*) Partition $\mathcal{F}$ into $\mathcal{F}'$ and $\mathcal{F}''$

- depending on whether the factor ϕ contains Z as an argument

(*) corresponds to step where we put all terms that contain a particular variable in innermost summation

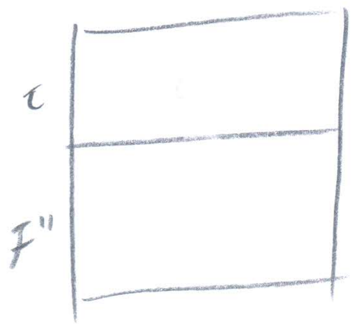
(*) $\mathcal{F}''$ is the complement of $\mathcal{F}'$

- (3) (*) - Product

- (4) (*) - Sum

- $\tau(\cdot)$ does not contain Z_i

- Returns



(*) stack $|F_1| < |F_2| < \ldots$

(*) Normalise over product of remaining terms.

Ex: There may be multiple query r.v.s.

- Ex: An automated way of addressing all GMs (despite NP-hard)

- What are issues?

- Hand-wavy about ordering

- Irreducibility where factor is as big as model itself.

(*) Not guaranteed that factors are readily identified.

- ordering may change size of factors (coupling)

Ex: move onto special cases

- complexity of variable elimination → concrete method for algo complexity

- (Q) (A6): understand how this is done

(*) - c is subset of r.v.s captured by a particular factor that co-occurs with query x.

- HMM: factor size of 2

∴ complexity :- K^2

- (A7) (*) - This example

Ex: move away from chain models Ex: walkthrough elim. algorithm

- Food web/complex network ; elimination in non-chain models

Query $P(A|H)$

- initial factor stack
- elimination order (not covered now)

- use graph too keep track of factors, newly formed factors

mechanical recording (not prob semant.)

(*) - check you understand this example ✓

- After all terms eliminated ...

(Q) (A8): step 8 - clarify. ✓

- understanding variable elimination

- turn original graph → undirected moralised graph ✓

- Graph elimination - algebraic

- sequence of graph elimination ⇒ model with only query nodes

(*) new structures along the way giving meaningful graphical, algebraic info. ✓

(through connection) ⟶ intermediate cliques corres. to algebraically eliminated

There is one-to-one corres of graph eliminants and algebraic eliminants
(operational equivalence)

(*) - Allows visual inspection of largest clique ⇒ info about largest intermediate terms ✓

(*) - operationalisable way of determining inference complexity

---

Ex: Must take each elimination term in context.

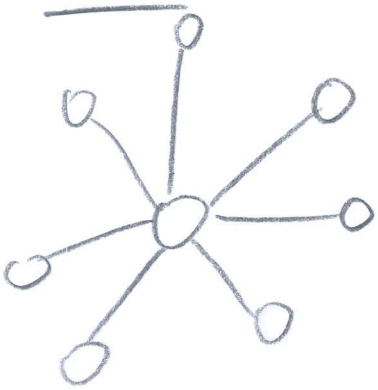
- clique tree
  - Each graph eliminant can be connected by an edge when they share a subset of random variables.
- Following elimination order ; ⇒ message passing along a clique tree
- Form of message is a sum-product term

- Ex: message-passing as a general inference algo.
  (*) elimination is analogous to passing a message over a particular ordering (over clique trees).

(Q) where does elimination ordering come from
How does it affect algo complex.

- examples:- star and Tree.

star



- can be DGM/UGM
- inference on star : how to do elimination of r.v.s?
- (A) "from degree 1 nodes"
- orderings (heuristic example)

· EX: ~~we are~~ Are we always able to find a good ordering that gives polynomial-time elimination algorithm

(W) (A8) - we eliminate, therefore need to connect - need clarity on this aspect ✓

- Ising model → eg 100 x 100 pixel
  can get
  - cliques with 100 nodes.

EX:
- However smart you are with elimination ordering; can still get exponentially large intermediate factors → still exponentially hard. ✓

(*) A pathway from elimination → message passing

(Q) If we decide to query another r.v. after graph elimin. do we have to redo?

- (*) minimum computation; redo messages (recompute a subset of messages)

EX: Revise lecture

· elimination - tractable inference on PGM by exploiting ~~via~~ elim. ordering; potential for max clique to be manageable
- And can determine complexity precisely before working on it.

Appen.
- messages can be bi-directional