BITS Pilani

Pilani | Dubai | Goa | Hyderabad

Probabilistic Graphical Model

Session # 11: Approximate Inference and MAP Inference

Seetha Parameswaran

seetha.p@pilani.bits-pilani.ac.in

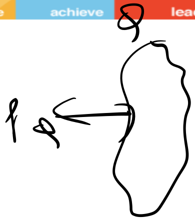
Table of Contents

1 Inference as Optimization

2 Exact Inference as an Optimization Problem

3 Propagation-Based Approximation

Constrained Optimization Problem



- Define a target class Q of **easy** distributions Q.
- Then search for an instance within that class that is the **best** approximation to P_Φ.
- Queries can then be answered using inference on Q rather than on P_Φ.
- This approach reformulates the inference task as one of optimizing an objective function over the class Q.
- This problem falls into the category of **constrained optimization**.

Table of Contents

1. Inference as Optimization

2. Exact Inference as an Optimization Problem

3. Propagation-Based Approximation

Exact Inference as an Optimization Problem

- Factorized distribution

$$P_{\Phi}(X) = \frac{1}{Z} \prod_{\varphi \in \Phi} \varphi(U_{\varphi})$$

- $U_{\varphi} = Scope[\varphi] \subseteq X$ are scope of each factor φ in the distribution P_{Φ}.
- Queries about the distribution P_{Φ} include queries about marginal probabilities of variables and queries about the partition function Z.
- Exact inference finds a set of calibrated beliefs that represent $P_{\Phi}(X)$.
- We can view exact inference as searching over the set of distributions over the set of distributions Q that are representable by the cluster tree to find a distribution Q^* that matches P_{Φ}.

Exact Inference as an Optimization Problem

- Searching for a calibrated distribution that is as close as possible to P_Φ.
- Aim is to avoid performing inference with the distribution P_Φ.
- **Relative Entropy** or **KL Divergence** is used as a distance measure to find an approximation Q to P_Φ, such that the relative entropy is minimized.

Relative Entropy

- Relative entropy between two distributions P_1 and P_2

$$D(P_1 || P_2) = E_{P_1}\left[\ln \frac{P_1(X)}{P_2(X)}\right]$$

$$\int \ln \frac{P_1(x)}{P_2(x)} P_1(x)\, dx$$

- $D \geq 0$
- $D = 0$ if and only if $P_1 = P_2$.
- Relative entropy is non-symmetric.

$$D(P_1 || P_2) \neq D(P_2 || P_1)$$

Exact Inference as an Optimization Problem

- Goal is to search for a distribution Q that minimizes $D(Q||P_\Phi)$.
- Suppose the clique tree structure T for P_Φ satisfies running intersection property and family preservation property.

$$Q = \underbrace{\{\beta_i : i \in V_T\}}\ \cup\ \underbrace{\{\mu_{i,j} : (i-j) \in E_T\}}$$

$$\boxed{P_\Phi(X) = \frac{\prod \phi}{\prod \mu}}$$

$$Q(X) = \frac{\prod_{i \in V_T} \beta_i(C_i)}{\prod_{i \in E_T} \mu_{i,j}(S_{i,j})}$$

- Due to calibration requirement we have

$$\beta_i[c_i] = Q(c_i)$$

$$\mu_{i,j}[s_{i,j}] = Q(S_{i,j})$$

- Seach for a Q that is representable by a set of beliefs over the cliques and sepsets in a particular clique tree structure T.

Exact Inference as an Optimization Problem

$\int \ln\left[Q/P\right] Q(x)$

- CTree-Optimize-KL: $-D(Q||P_\Phi) \sim -D(P_\Phi||Q)$ $\int \ln \frac{P_\Phi(x)}{Q(x)} P_\Phi$

  Find $Q = \{\beta_i : i \in V_T\} \cup \{\mu_{i,j} : (i - j) \in E_T\}$

  Maximizing $-D(Q||P_\Phi) = 0$ $Q = P_\Phi$

  subject to $\mu_{i,j}[s_{i,j}] = \sum_{C_i - S_{i,j}} \beta_i[c_i]$ $\forall (i-j) \in E_T,\ \forall s_{i,j} \in Val(S_{i,j})$

  $\sum_{c_i} \beta_i[c_i] = 1$ $\forall i \in V_T$

  Q as the objective function

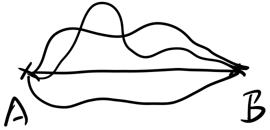
- Optimization problem CTree-Optimize-KL has a unique solution

$$-\int Q(x) \ln \Phi(x) = H_{\Phi}(X)$$

Exact Inference as an Optimization Problem

$$P_\Phi = \frac{1}{Z}\tilde{P}_\Phi$$

- Applying Relative entropy equation in P_Φ



Calculus of Variations

$$\rightarrow D(Q||P_\Phi) = \ln Z - F[\tilde{P}_\Phi, Q]$$

$$F[\tilde{P}_\Phi, Q] = \sum_{\varphi \in \Phi} E_Q[\ln \varphi] + H_Q(X)$$

$$D(Q||P_\Phi) \geq 0$$

$$\ln Z \geq F[\tilde{P}_\Phi, Q]$$

functional $\rightarrow f(Q)$ = scalar value

$f(x)$ = function of a variable

- $F[\tilde{P}_\Phi, Q]$ is called the **energy functional.**
- The first term is called the **energy term**.
- The second term is called the **entropy term**.
- Minimizing the relative entropy is equivalent to maximizing the energy functional.

Exact Inference as an Optimization Problem

$\min f(x, y, z)$

s.t $g(x, y, z) = c$

Lagrange multiplier

$\Rightarrow \nabla f(x, y, z) = \lambda \nabla g(x, y, z)$

- CTree-Optimize:

Find $\quad Q = \{\beta_i : i \in V_T\} \cup \{\mu_{i,j} : (i - j) \in E_T\}$

Maximizing $\quad F[\tilde{P}_\Phi, Q]$

subject to $\quad \mu_{i,j}[s_{i,j}] = \sum_{C_i - S_{i,j}} \beta_i[c_i] \quad \forall (i - j) \in E_T,\ \forall s_{i,j} \in Val(S_{i,j})$

$$\sum_{c_i} \beta_i[c_i] = 1 \qquad \forall i \in V_T$$

$$\beta_i(c_i) \geq 0 \qquad \forall i \in V_T;\ c_i \in Val(C_i)$$

Fixed-point characterisation

We need first the concept of Lagrange multipliers since we will convert the given <u>constrained optimization problem</u> to an unconstrained one.

To understand the derivation in the book, we need Lagrange multipliers + functionals ⇒ out of scope

Fixed-point characterisation

Setting up Lagrange multiplier equations for the constrained optimization problem and finding stationary points leads to the same sort of equations as the message passing algorithm

Table of Contents

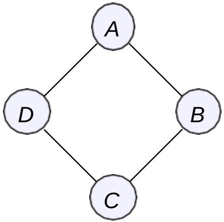
1 Inference as Optimization

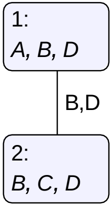
2 Exact Inference as an Optimization Problem

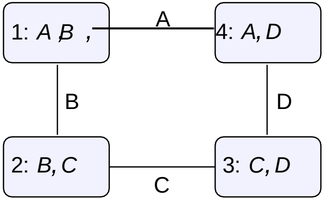
3 Propagation-Based Approximation

Propagation-Based Approximation

- Use the same message propagation as in exact inference.
- Use Cluster graph instead of clique tree.
- The cluster graph contains loops (undirected cycles), such graphs are often called **loopy**.
- The BP algorithm is called **Loopy belief propagation**, since it uses propagation steps used by algorithms for Markov trees, but applied to networks with loops.

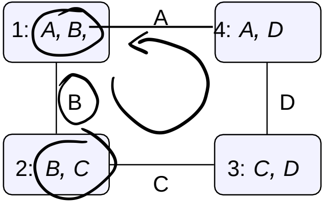


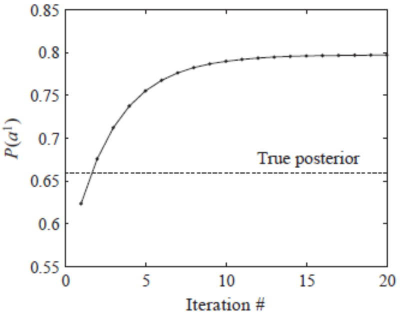




Propagation-Based Approximation

- Message propagation process may not converge in two passes, since information from one pass will circulate and affect the next round.
- In some cases, the propagation of beliefs may not converge at all.
- An example run of loopy belief propagation is given below.





What happens if we use CTree-BU-Update?

Let us propagate messages in the order $\mu_{12}, \mu_{23}, \mu_{34}, \mu_{41}$

In the first message μ_{12}, the cluster AB passes information to cluster BC using a marginal distribution on B.

What happens if we use CTree-BU-Update?

Suppose all clusters favour consensus joint assignments

$\beta_1(a^0, b^0)$ and $\beta_1(a^1, b^1)$ much larger than $\beta_1(a^0, b^1)$ and $\beta_1(a^1, b^0)$

If μ_{12} strengthens the belief that $B = b^1$,

then μ_{23} strengthens the belief that $C = c^1$

What happens if we use CTree-BU-Update?

Going around the loop, cluster AB will get a message that strengthens the belief that $A = a'$

This message will be treated as being independent of the initial propagation when it is not so. ⇒ This procedure overestimates the marginal probability of A

Cluster-graph Belief Propagation

We say that U satisfies the running intersection property if, whenever there is a variable X such that X ∈ C_i and X ∈ C_j , then there is a single path between C_i and C_j for which X ∈ S_e for all edges e in the path

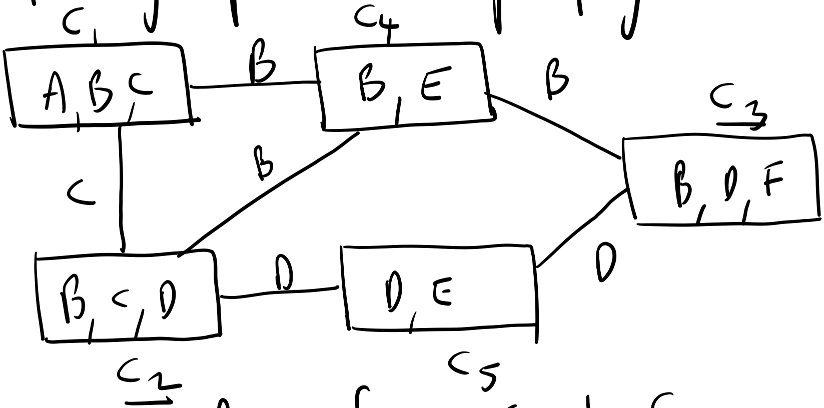
This is a <u>generalized running intersection</u> <u>property</u>

Cluster-graph belief propagation

There is only a single path in which information about X can flow in the graph.

- All clusters must agree on a marginal distribution of X.
- At most one path means information about X cannot endlessly cycle in a loop.

Cluster-graph belief propagation



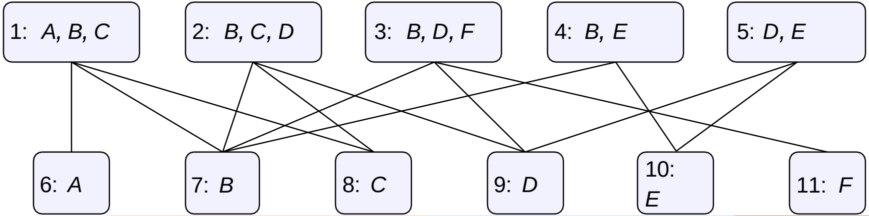
Two paths from C_3 to C_2

Cluster-graph belief propagation

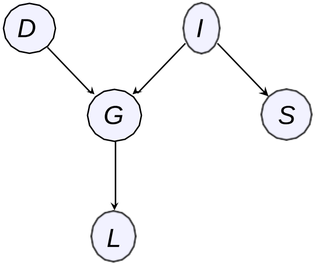
- The first path from C_3 to C_2 goes through C_4 and propagates information about B
- The second path from C_2 to C_3 goes through C_5 and propagates information about D
- We can still get circular reasoning
- For edge i-j connecting C_i, C_j $\sum_{C_i - S_{i,j}} \beta_i = \sum_{C_j - S_{i,j}} \beta_j$

Bethe Cluster Graph

- Bethe cluster graph, uses a bipartite graph.
- The first layer graph consists of "large" clusters, with one cluster for each factor φ in Φ, whose scope is $Scope[\varphi]$.
- These clusters satisfy the family-preservation property.
- The second layer consists of "small" univariate clusters, one for each random variable.
- Place an edge between each univariate cluster X on the second layer and each cluster in the first layer that includes X ; the scope of this edge is X itself.



MAP and Variable Elimination



$$\max_{S,I,D,L,G} P(D, I, G, S, L)$$

$$= \max_{L,G}[\varphi_L(L, G)] \cdot \mathrm{Max}_D[\varphi_D(D) \cdot T_2(G, D)]$$

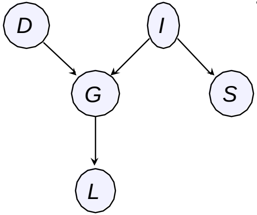
$$= \max_{L,G}[\varphi_L(L, G)] \cdot T_3(G)$$

$$= \max_{G}[T_3(G)] \cdot \max_L[\varphi_L(L, G)]$$

$$= \max_{G}[T_3(G)] \cdot T_4(G)$$

$$= \boxed{T_5(\theta)}$$

MAP and Variable Elimination



Step	Variable eliminated	Factors used	Intermediate factor	New factor
1	S	$\varphi_S(S, I)$	$\psi_1(I, S)$	$T_1(I)$
2	I	$\varphi_I(I) \cdot \varphi_G(G, I, D) \cdot T_1(I)$	$\psi_2(G, I, D)$	$T_2(G, D)$
3	D	$\varphi_D(D) \cdot T_2(G, D)$	$\psi_3(G, D)$	$T_3(G)$
4	L	$\varphi_L(L, G)$	$\psi_4(L, G)$	$T_4(G)$
5	G	$T_3(G) \cdot T_4(G)$	$\psi_5(G)$	$T_5(\theta)$

Now choose the maximizing value x_i^* for X_i.

g^*

MAP, Variable Elimination and Traceback

$\max P(b/a)$

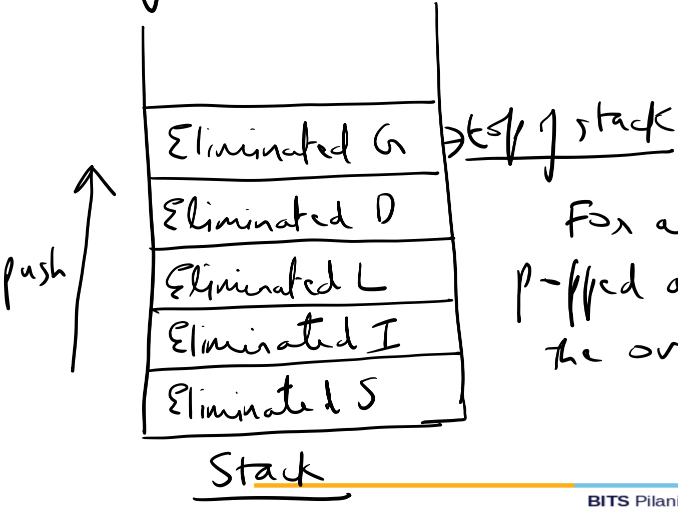
- Determine a conditional maximizing value – their maximizing value given the values of the variables that have not yet been eliminated.
- Pick the value of the final variable.
- Then go back and pick the values of the other variables accordingly.
- For the last variable eliminated X, the factor for the value x contains the probability of the most likely assignment that contains $X = x$.
- This process is called **traceback** of the solution.

$a^* \to \max P(b/a^*)$

MAP, Variable Elimination and Traceback

Step	Variable eliminated	Factors used	Intermediate factor	New factor	Traceback
1	S	$\varphi_S(S, I)$	$\psi_1(I, S)$	$\tau_1(I)$	$s^* = arg\ \max_s \psi_1(i^*, s)$
2	I	$\varphi_I(I) \cdot \varphi_G(G, I, D) \cdot \tau_1(I)$	$\psi_2(G, I, D)$	$\tau_2(G, D)$	$i^* = arg\ \max_i \psi_2(g^*, d^*, i)$ d^*
3	D	$\varphi_D(D) \cdot \tau_2(G, D)$	$\psi_3(G, D)$	$\tau_3(G)$	$= arg\ \max_d \psi_3(g^*, d)$
4	L	$\varphi_L(L, G)$	$\psi_4(L, G)$	$\tau_4(G)$	$l^* = arg\ \max_l \psi_4(g^*, l)$
5	G	$\tau_3(G) \cdot \tau_4(G)$	$\psi_5(G)$	$\tau_5(\theta)$	$g^* = arg\ \max_g \psi_5(g)$

Think of it like a stack....



For assignment the stack is popped and variables assigned in the order $G^* \to D^* \to L^* \to I^* \to S^*$

MAP and Variable Elimination Algorithm

$X_1 = S,\ X_2 = I,\ X_3 = L,\ X_4 = D,\ X_5 = G$

Procedure Max-Product-VE (
  Φ, // Set of factors over $\boldsymbol{X}$
  $\prec$ // Ordering on $\boldsymbol{X}$
)

Let $X_1, \ldots, X_k$ be an ordering of $\boldsymbol{X}$ such that
  $X_i \prec X_j$ iff $i < j$
  for $i = 1, \ldots, k$
    $(\Phi, \phi_{X_i}) \leftarrow$ Max-Product-Eliminate-Var(Φ, X_i)
  $\boldsymbol{x}^* \leftarrow$ Traceback-MAP$(\{\phi_{X_i} : i = 1, \ldots, k\})$
  return $\boldsymbol{x}^*, \Phi$ // Φ contains the probability of the MAP

Procedure Max-Product-Eliminate-Var (
  Φ, // Set of factors
  Z // Variable to be eliminated
)

$\Phi' \leftarrow \{\phi \in \Phi : Z \in Scope[\phi]\}$
  $\Phi'' \leftarrow \Phi - \Phi'$
  $\psi \leftarrow \prod_{\phi \in \Phi'} \phi$
  $\tau \leftarrow \max_Z \psi$
  return $(\Phi'' \cup \{\tau\}, \psi)$

2

Procedure Traceback-MAP (
  $\{\phi_{X_i} : i = 1, \ldots, k\}$
)

for $i = k, \ldots, 1$
  $\boxed{\boldsymbol{u}_i \leftarrow (x^*_{i+1}, \ldots, x^*_k)\langle Scope[\phi_{X_i}] - \{X_i\}\rangle}$
  // The maximizing assignment to the variables eliminated after X_i
  $x^*_i \leftarrow \arg\max_{x_i} \phi_{X_i}(x_i, \boldsymbol{u}_i)$
  // x^*_i is chosen so as to maximize the corresponding entry in the factor, relative to the previous choices $\boldsymbol{u}_i$
return $\boldsymbol{x}^*$

Table of Contents

1 Inferences

2 Maximum a Posteriori (MAP) Query

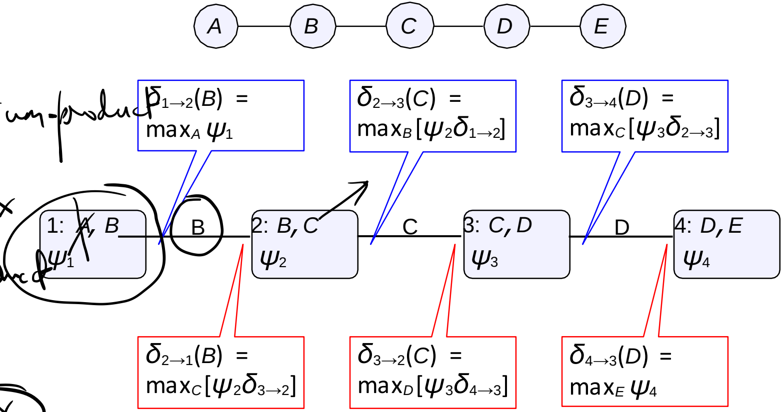
3 Max Product and Max Marginals

4 MAP and Variable Elimination

5 MAP using Belief Propagation

MAP using Belief Propagation

Same as sum-product except
sum → max



sum product

$\sum_A \psi_1$

max

MAP using Belief Propagation

- An exact solution to the MAP problem via a variable elimination procedure is intractable.
- Use message passing procedures in cluster graphs to compute approximate max-marginals.
- These pseudo-max-marginals can be used for selecting an assignment.
- The task has two parts: computing the max-marginals and decoding them to extract a

  MAP assignment.

$$\psi(C_i) = \psi_i \cdot \prod \delta_{k->i}$$

$$\tau(S_{i,j}) = \max_{C_i - S_{i,j}} \psi(C_i)$$

MAP using Belief Propagation

- For each clique C_i , and each assignment c_i to C_i ,

$$\beta_i(c_i) = MaxMarg_{P_\Phi}(c_i)$$

- Any two adjacent cliques must agree on their sepset. The cliques are said to be max-calibrated.

Σ Σ

$$\boxed{\max_{C_i - S_{i,j}} \beta_i = \max_{C_j - S_{i,j}} \beta_j = \mu_{ij}(S_{i,j})}$$

- The beliefs in a clique tree resulting from an upward and downward pass of the max-product clique tree algorithm are max-calibrated.

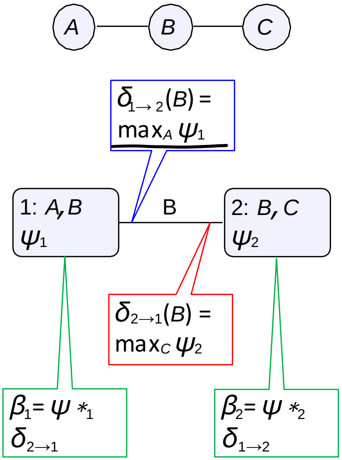
MAP using Belief Propagation

- The assignment ξ^* has the local optimal assignment property ξ^* given a max-calibrated set of beliefs $\beta_i(C_i)$, if for each clique

$$\xi^*(C_i) \in \boxed{\arg\max_{C_i} \beta_i(c_i)}$$

- The task of finding a locally optimal assignment ξ^* given a max-calibrated set of beliefs is called the decoding task.

MAP + BP – Most Likely Assignment – Example



ψ_1		
	a^1b^1	3
	a^2b^1	-1
	a^1b^2	0
	a^2b^2	1

ψ_2		
	b^1c^1	4
	b^1c^2	-1
	b^2c^1	1
	b^2c^2	2

$\delta_{1\to2}$		
	b^1	3
	b^2	1

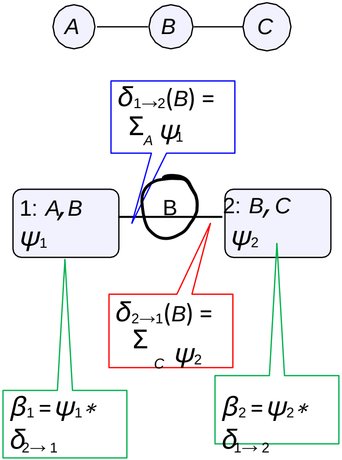
$\delta_{2\to1}$		
	b^1	4
	b^2	2

β_1		
	a^1b^1	3*4 = 12
	a^2b^1	-1*4 = -4
	a^1b^2	0*2 = 0
	a^2b^2	1*2 = 2

β_2		
	b^1c^1	4*3 = 12
	b^1c^2	-1*3 = -3
	b^2c^1	1*1 = 1
	b^2c^2	2*1 = 2

Most Likely assignment = (a^1, b^1, c^1)

MAP + BP – Calibration – Example



β_1		
	a^1b^1	3*4 = 12
	a^2b^1	-1*4 = -4
	a^1b^2	0*2 = 0
	a^2b^2	1*2 = 2

β_2		
	b^1c^1	4*3 = 12
	b^1c^2	-1*3 = -3
	b^2c^1	1*1 = 1
	b^2c^2	2*1 = 2

$\max_A \beta_1$		
	b^1	12
	b^2	2

=

$\max_C \beta_2$		
	b^1	12
	b^2	2