Chapter 14: Approximate Inference by Belief Propagation

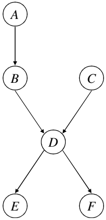
Adnan Darwiche[1]

[1] Lecture slides for *Modeling and Reasoning with Bayesian Networks*, Adnan Darwiche, Cambridge University Press, 2009.

Outline

We discuss in this chapter a class of approximate inference algorithms which are based on belief propagation. These algorithms provide a full spectrum of approximations, allowing one to trade-off approximation quality with computational resources.

The Belief Propagation Algorithm



A	Θ_A
true	.01
false	.99

A	B	$\Theta_{B\|A}$
true	true	.100
true	false	.900
false	true	.001
false	false	.999

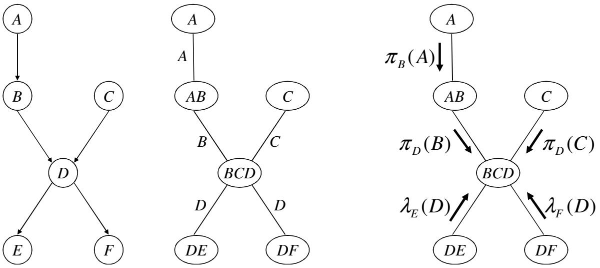
C	Θ_C
true	.001
false	.999

B	C	D	$\Theta_{D\|BC}$
true	true	true	.99
true	true	false	.01
true	false	true	.90
true	false	false	.10
false	true	true	.95
false	true	false	.05
false	false	true	.01
false	false	false	.99

D	E	$\Theta_{E\|D}$
true	true	.9
true	false	.1
false	true	.3
false	false	.7

D	F	$\Theta_{F\|D}$
true	true	.2
true	false	.8
false	true	.1
false	false	.9

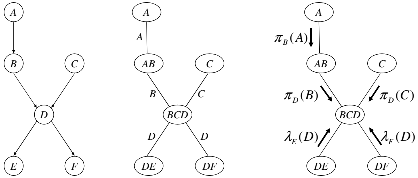
The Belief Propagation Algorithm



- A node i in the jointree, which corresponds to variable X in the polytree, will have its cluster as $\mathbf{C}_i = X\mathbf{U}$, where $\mathbf{U}$ are the parents of X in the polytree.
- An undirected edge i–j in the jointree, which corresponds to edge $X \rightarrow Y$ in the polytree, will have its separator as $\mathbf{S}_{ij} = X$.

$$\Pr(X\mathbf{U}, \mathbf{e}) = \lambda_{\mathbf{e}}(X)\Theta_{X|\mathbf{U}} \prod_i \pi_X(U_i) \prod_j \lambda_{Y_j}(X)$$

The Belief Propagation Algorithm



Messages propagated towards node D under evidence $E =$ true.
We have three π-messages in this case:

A	$\pi_B(A)$
true	.01
false	.99

B	$\pi_D(B)$
true	.00199
false	.99801

C	$\pi_D(C)$
true	.001
false	.999

We also have two λ-messages:

D	$\lambda_E(D)$
true	.9
false	.3

D	$\lambda_F(D)$
true	1
false	1

The Belief Propagation Algorithm

To compute the joint marginal for the family of variable D, we simply evaluate:

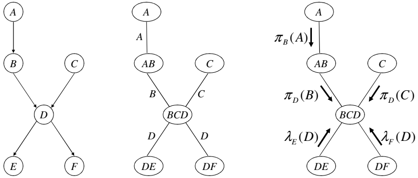
$$\Pr(BCD, \mathbf{e}) = \Theta_{D|BC} \cdot \pi_D(B)\pi_D(C) \cdot \lambda_E(D)\lambda_F(D)$$

leading to the following:

B	C	D	$\Pr(BCD, \mathbf{e})$
true	true	true	1.7731×10^{-6}
true	true	false	5.9700×10^{-9}
true	false	true	1.6103×10^{-3}
true	false	false	5.9640×10^{-5}
false	true	true	8.5330×10^{-4}
false	true	false	1.4970×10^{-5}
false	false	true	8.9731×10^{-3}
false	false	false	2.9611×10^{-1}

By summing all table entries, we find that $\Pr(\mathbf{e}) \approx 0.3076$.

The Belief Propagation Algorithm



We can also compute the joint marginal for variable C once we compute the message passed from D to C:

C	$\lambda_D(C)$
true	0.8700
false	0.3071

C	$\Pr(C, \mathbf{e})$
true	0.0009
false	0.3067

Note that we could have also computed the joint marginal for variable C using the joint marginal for the family of D: $\Pr(C, \mathbf{e}) = \sum_{BD} \Pr(BCD, \mathbf{e})$.

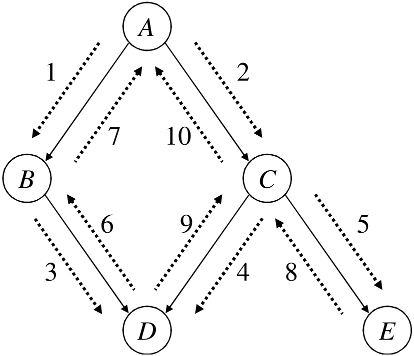
The Belief Propagation Algorithm

$BEL(X\mathbf{U})$ denotes the conditional marginal $\Pr(X\mathbf{U}|\mathbf{e})$:

$$
\begin{aligned}
BEL(X\mathbf{U}) &= \eta\, \lambda_{\mathbf{e}}(X)\Theta_{X|\mathbf{U}} \prod_i \pi_X(U_i) \prod_j \lambda_{Y_j}(X) \\
\lambda_X(U_i) &= \eta \sum_{X\mathbf{U}\setminus\{U_i\}} \lambda_{\mathbf{e}}(X)\Theta_{X|\mathbf{U}} \prod_{k\neq i} \pi_X(U_k) \prod_j \lambda_{Y_j}(X) \\
\pi_{Y_j}(X) &= \eta \sum_{\mathbf{U}} \lambda_{\mathbf{e}}(X)\Theta_{X|\mathbf{U}} \prod_i \pi_X(U_i) \prod_{k\neq j} \lambda_{Y_k}(X)
\end{aligned}
$$

η a constant that normalizes a factor to sum to one (to simplify the notation, we will refrain from distinguishing between constants η that normalize different factors).

Iterative Belief Propagation



Iterative Belief Propagation (IBP)

$\texttt{IBP}(\mathcal{N}, \mathbf{e})$

input:
$\mathcal{N}$: a Bayesian network inducing distribution $\Pr$
$\mathbf{e}$: an instantiation of some variables in network $\mathcal{N}$

output: approximate marginals, $BEL(X\mathbf{U})$, of $\Pr(X\mathbf{U}|\mathbf{e})$ for each family $X\mathbf{U}$ in $\mathcal{N}$.

main:

$t \leftarrow 0$
initialize all messages π^0, λ^0 (uniformly)
while messages have not converged **do**
  $t \leftarrow t + 1$
  for each node X with parents $\mathbf{U}$ **do**
    for each parent U_i **do**
      $\lambda^t_X(U_i) = \eta \ \sum_{X\mathbf{U} \setminus \{U_i\}} \lambda_{\mathbf{e}}(X)\Theta_{X|\mathbf{U}} \prod_{k \neq i} \pi^{t-1}_X(U_k) \prod_j \lambda^{t-1}_{Y_j}(X)$
    end for
    for each child Y_j **do**
      $\pi^t_{Y_j}(X) = \eta \ \sum_{\mathbf{U}} \lambda_{\mathbf{e}}(X)\Theta_{X|\mathbf{U}} \prod_i \pi^{t-1}_X(U_i) \prod_{k \neq j} \lambda^{t-1}_{Y_k}(X)$
    end for
  end for
end while
return $BEL(X\mathbf{U}) = \eta \ \lambda_{\mathbf{e}}(X)\Theta_{X|\mathbf{U}} \prod_i \pi^t_X(U_i) \prod_j \lambda^t_{Y_j}(X)$ for families $X\mathbf{U}$

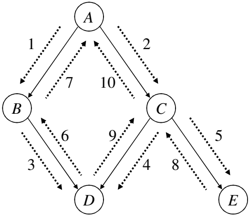
When IBP converges, the values of its messages at convergence are called a fixed point.

In general, IBP may have multiple fixed points on a given network.

Iterative Belief Propagation

- How fast will IBP converge, or should we even expect it to converge?
- One can identify networks where the messages computed by IBP can oscillate, and if left to run without limit, IBP could loop forever.
- The convergence rate of IBP can depend crucially on the order in which messages are propagated, which is known as a message schedule.
- IBP is said to use a parallel schedule, since we wait until all messages for an iteration are computed before they are propagated (in parallel) in the following iteration.
- One can adopt a sequential schedule where messages are propagated as soon as they are computed.
- All schedules have the same fixed points (if IBP starts at a fixed point, it stays at a fixed point, independent of the schedule).

Iterative Belief Propagation



For a concrete example of sequential schedules, we compute messages in the following order:

$$\pi_B(A), \pi_C(A), \pi_D(B), \pi_D(C), \pi_E(C), \lambda_D(B), \lambda_B(A), \lambda_E(C), \lambda_D(C), \lambda_C(A).$$

- When we are ready to compute the message $\pi_E(C)$ using messages $\pi_C(A)$ and $\lambda_D(C)$, we can use the most up-to-date ones: $\pi_C(A)$ from the current iteration and $\lambda_D(C)$ from the previous iteration.
- Information available at node A is able to propagate to E, two steps away, in the same iteration.
- Computing messages in parallel, this same information would have taken two iterations to reach E.

The Semantics of IBP

We will provide a semantics for Algorithm IBP showing how it can be viewed as searching for an approximate probability distribution $\Pr'$ that attempts to minimize the Kullback-Leibler divergence with the distribution $\Pr$ induced by the given Bayesian network.

The Kullback-Leibler Divergence

The Kullback-Leibler divergence (KL–divergence)

$$\mathrm{KL}(\mathrm{Pr}'(\mathbf{X}|\mathbf{e}), \mathrm{Pr}(\mathbf{X}|\mathbf{e})) = \sum_{\mathbf{x}} \mathrm{Pr}'(\mathbf{x}|\mathbf{e}) \log \frac{\mathrm{Pr}'(\mathbf{x}|\mathbf{e})}{\mathrm{Pr}(\mathbf{x}|\mathbf{e})}$$

- $\mathrm{KL}(\mathrm{Pr}'(\mathbf{X}|\mathbf{e}), \mathrm{Pr}(\mathbf{X}|\mathbf{e}))$ is non-negative
- equal to zero if and only if $\mathrm{Pr}'(\mathbf{X}|\mathbf{e})$ and $\mathrm{Pr}(\mathbf{X}|\mathbf{e})$ are equivalent.

Optimizing the KL-Divergence

The approximations computed by IBP are based on assuming an approximate distribution $\Pr'(\mathbf{X})$ that factors as follows:

$$\Pr'(\mathbf{X}|\mathbf{e}) = \prod_{X\mathbf{U}} \frac{\Pr'(X\mathbf{U}|\mathbf{e})}{\prod_{U \in \mathbf{U}} \Pr'(U|\mathbf{e})}$$

- This choice of $\Pr'(\mathbf{X}|\mathbf{e})$ is expressive enough to describe distributions $\Pr(\mathbf{X}|\mathbf{e})$ induced by polytree networks $\mathcal{N}$
- In the case where $\mathcal{N}$ is not a polytree, then we are simply trying to fit $\Pr(\mathbf{X}|\mathbf{e})$ into an approximation $\Pr'(\mathbf{X}|\mathbf{e})$ as if it were generated by a polytree network.
- The entropy of distribution $\Pr'(\mathbf{X}|\mathbf{e})$ can be expressed as:

$$\mathrm{ENT}'(\mathbf{X}|\mathbf{e}) = -\sum_{X\mathbf{U}} \sum_{x\mathbf{u}} \Pr'(x\mathbf{u}|\mathbf{e}) \log \frac{\Pr'(x\mathbf{u}|\mathbf{e})}{\prod_{u \sim \mathbf{u}} \Pr'(u|\mathbf{e})}$$

Optimizing the KL-Divergence

- IBP fixed points are stationary points of the KL–divergence: they may only be local minima, or they may not be minima.
- When IBP performs well, it will often have fixed points that are indeed minima of the KL–divergence.
- For problems where IBP does not behave as well, we will next seek approximations Pr' whose factorizations are more expressive than that of the polytree-based factorization.