Overview of Probabilistic Graphical Models Representation: Part II

Aditya Ramesh
_@adityaramesh.com

John Cavazos Lab
University of Delaware
Newark, Delaware 19716

This presentation is based **heavily** on the material by Daphne Koller ([1]), Nir Friedman ([1], [3]), and David Sontag ([2]). I have **directly copied** or otherwise incorporated parts of their work into many of these slides, citing the sources where appropriate. If you find this presentation useful, I highly recommend that you take some time to read their work.

Outline

Conversion between Networks [1]
- Motivation
- Bayesian Networks to Markov Networks
- Markov Networks to Bayesian Networks

Partially Directed Models [1]
- Motivation
- Conditional Random Fields
- Named-Entity Recognition
- Chain Graphs

Motivation

- We saw earlier with the Misconception example that we cannot easily express interactions between cliques in Bayesian networks.
- Recall the conditions under which the common effect v-structure in Bayesian networks ($X \rightarrow Y \leftarrow Z$) is active.
- We cannot represent these types of independencies in Markov networks.

Motivation

- Each type of network can represent independence constraints that the other cannot.
- A natural question to ask is the following: when can we convert one type of network to another?
- We will first develop a procedure to convert a Bayesian network to a Markov network, and find the conditions under which the transformation is "lossless".

Bayesian Networks to Markov Networks

- We define the *reduction* of the factor ϕ to the context $\mathbf{U} = \mathbf{u}$, denoted $\phi[\mathbf{U} = \mathbf{u}]$ (and abbreviated $\phi[\mathbf{u}]$), to be a factor over scope $\mathbf{Y}' = \mathbf{Y} - \mathbf{U}$, such that

$$\phi[\mathbf{u}](\mathbf{y}') = \phi(\mathbf{y}', \mathbf{u}).$$

- A Bayesian network conditioned on evidence $\mathbf{E} = \mathbf{e}$ also induces a Gibbs distribution: the one defined by the original factors *reduced* to the context $\mathbf{E} = \mathbf{e}$.

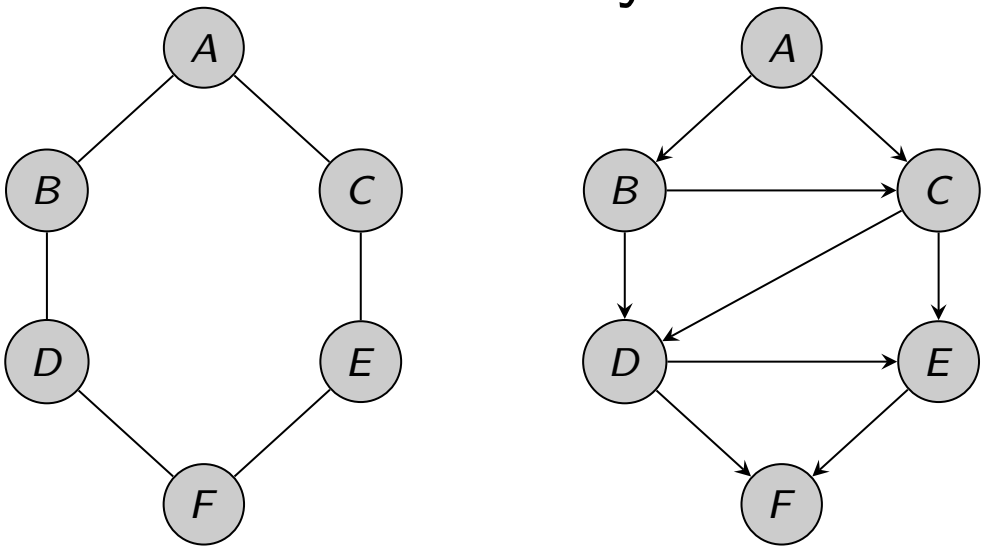
Bayesian Networks to Markov Networks

- A Markov network has the restriction that all variables belonging to the scope of a factor must be in the same clique.
- If we want to come up with a procedure to convert a Bayesian network to an undirected representation, we have to find some way of dealing with v-structures of the form $X \rightarrow Z \leftarrow Y$.
- Solution: *moralize* the network by "marrying" unconnected parents of the same node.

Bayesian Networks to Markov Networks

- The moralized graph $\mathcal{M}[G]$ is clearly an I-map for any distribution P parameterized over G.
- However, it is only a perfect map when we did not have any *immoralities* in the Bayesian network to begin with.
- Otherwise, we have lost independence information through the addition of moralizing edges.
- Caveat: we rarely encounter a directed graph that is moral. The transformation is likely to be "lossy".

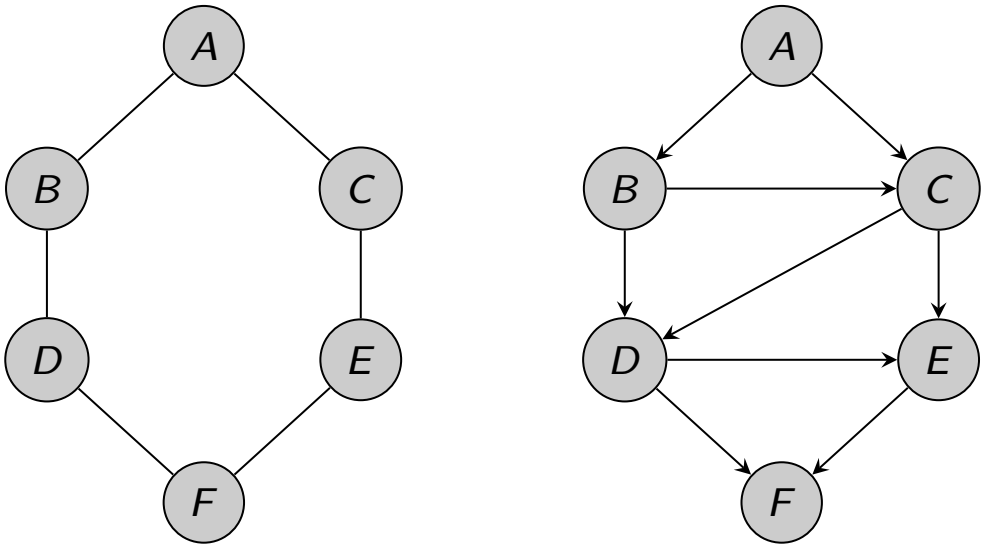
Markov Networks to Bayesian Networks



(a) (b)

- The procedure to transform a Markov network into a Bayesian network is more subtle, so we will first focus on an example.
- We aim to find a Bayesian network G that is a minimal I-map for the Markov network H shown in (a).
- Enumerating the nodes of H in the order A, B, C, D, E, F, and ensuring that the independencies $I(H)$ are satisfied, results in the Bayesian network minimal I-map shown in (b).

Markov Networks to Bayesian Networks



(a) (b)

- It turns out that any Bayesian network minimal I-map for an undirected graph H can have no immoralities, and is necessarily chordal (the longest loop is of length 3).
- The process of adding enough edges to a Markov network to turn it into a Bayesian network is called *triangulation*.

Markov Networks to Bayesian Networks

- The ordering of nodes we determined for the previous example can be thought of as derived from the ordering of cliques in the *clique tree* embedded in the undirected graph.
- In the previous example, each clique in the tree consists of exactly one node.
- When we generalize this approach to cliques containing many nodes, we need to formalize what it means for two cliques to be "separated".

Markov Networks to Bayesian Networks

- Let $\mathbf{C}_i, \mathbf{C}_j$ be two (necessarily maximal) cliques in a clique tree that are directly connected by an edge.
  - We define $\mathbf{S}_{i,j} = \mathbf{C}_i \cap \mathbf{C}_j$ to be the *sepset* between $\mathbf{C}_i$ and $\mathbf{C}_j$.
  - The sepset $\mathbf{S}_{i,j}$ is the minimal set of nodes that need to be observed to render $\mathbf{C}_i$ and $\mathbf{C}_j$ conditionally independent.
  - Let $\mathbf{W}_{<(i,j)}$ ($\mathbf{W}_{<(j,i)}$) be all of the variables that appear in any clique on the $\mathbf{C}_i$ ($\mathbf{C}_j$) side of the edge.
  - Each edge decomposes $\mathcal{X}$ into three disjoint sets: $\mathbf{W}_{<(i,j)} - \mathbf{S}_{i,j}$, $\mathbf{W}_{<(j,i)} - \mathbf{S}_{i,j}$, and $\mathbf{S}_{i,j}$.

Markov Networks to Bayesian Networks

- We say that a tree T is a clique tree for H if:
  - each node corresponds to a clique in H, and each maximal clique in H is a node in T;
  - each sepset $\mathbf{S}_{i,j}$ separates $\mathbf{W}_{<(i,j)}$ and $\mathbf{W}_{<(j,i)}$ in H.
- In other words, the clique tree describes the connections between the set of maximal cliques in H, such that observing the sepset between two cliques in the tree renders them conditionally independent.

Markov Networks to Bayesian Networks

- To generalize our procedure to convert a given Markov network H to a Bayesian network G, we do the following:

Markov Networks to Bayesian Networks

- To generalize our procedure to convert a given Markov network H to a Bayesian network G, we do the following:
  1. Select an arbitrary clique $\mathbf{C}_1$ to be the root of the clique tree.

Markov Networks to Bayesian Networks

- To generalize our procedure to convert a given Markov network H to a Bayesian network G, we do the following:
  1. Select an arbitrary clique $\mathbf{C}_1$ to be the root of the clique tree.
  2. Order the cliques $\mathbf{C}_1, \ldots, \mathbf{C}_k$ using any topological ordering such that the cliques closer to the root appear first.

Markov Networks to Bayesian Networks

- To generalize our procedure to convert a given Markov network H to a Bayesian network G, we do the following:
  1. Select an arbitrary clique $\mathbf{C}_1$ to be the root of the clique tree.
  2. Order the cliques $\mathbf{C}_1, \ldots, \mathbf{C}_k$ using any topological ordering such that the cliques closer to the root appear first.
  3. Order the nodes in any order consistent with the clique ordering: if X_l first appears in $\mathbf{C}_i$ and X_m first appears in $\mathbf{C}_j$, for $i < j$, then X_l must precede X_m in the ordering.

Markov Networks to Bayesian Networks

- To generalize our procedure to convert a given Markov network H to a Bayesian network G, we do the following:
  1. Select an arbitrary clique $\mathbf{C}_1$ to be the root of the clique tree.
  2. Order the cliques $\mathbf{C}_1, \ldots, \mathbf{C}_k$ using any topological ordering such that the cliques closer to the root appear first.
  3. Order the nodes in any order consistent with the clique ordering: if X_l first appears in $\mathbf{C}_i$ and X_m first appears in $\mathbf{C}_j$, for $i < j$, then X_l must precede X_m in the ordering.
  4. Use the algorithm discussed in the first talk (repeated shortly) to build a minimal I-map given the ordering $X_1, \ldots, X_n$ and independencies $I(H)$.

Algorithm for Constructing a Minimal I-Map

Require: An ordering $X_1, \ldots, X_n$ of random variables in $\mathcal{X}$
Require: A set of independencies I

Algorithm for Constructing a Minimal I-Map

Require: An ordering $X_1, \ldots, X_n$ of random variables in $\mathcal{X}$
Require: A set of independencies I
  Set G to an empty graph over $\mathcal{X}$

Algorithm for Constructing a Minimal I-Map

Require: An ordering $X_1, \ldots, X_n$ of random variables in $\mathcal{X}$
Require: A set of independencies I
Set G to an empty graph over $\mathcal{X}$
for $i = 1, \ldots, n$ **do**
U is the current candidate for parents of X_i
$\mathbf{U} \leftarrow \{X_1, \ldots, X_{i-1}\}$

Algorithm for Constructing a Minimal I-Map

Require: An ordering $X_1, \ldots, X_n$ of random variables in $\mathcal{X}$
Require: A set of independencies I
  Set G to an empty graph over $\mathcal{X}$
  for $i = 1, \ldots, n$ **do**
    $\mathbf{U}$ is the current candidate for parents of X_i
    $\mathbf{U} \leftarrow \{X_1, \ldots, X_{i-1}\}$
    Find the minimal set $\mathbf{U}$ satisfying
    $(X_i \perp \{X_1, \ldots, X_{i-1}\} - \mathbf{U} \mid \mathbf{U})$
    for $\mathbf{U}' \subseteq \{X_1, \ldots, X_{i-1}\}$ **do**
      if $\mathbf{U}' \subset \mathbf{U}$ and $(X_i \perp \{X_1, \ldots, X_{i-1}\} - \mathbf{U}' \mid \mathbf{U}') \in I$ **then**
        $\mathbf{U} \leftarrow \mathbf{U}'$

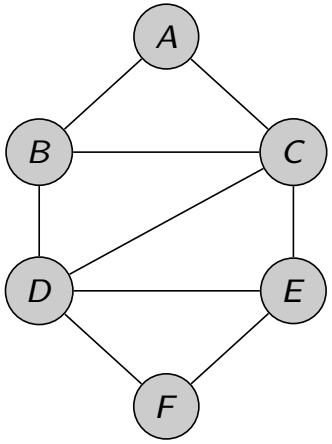
Algorithm for Constructing a Minimal I-Map

Require: An ordering $X_1, \ldots, X_n$ of random variables in $\mathcal{X}$
Require: A set of independencies I
  Set G to an empty graph over $\mathcal{X}$
  for $i = 1, \ldots, n$ **do**
    $\mathbf{U}$ is the current candidate for parents of X_i
    $\mathbf{U} \leftarrow \{X_1, \ldots, X_{i-1}\}$
    Find the minimal set $\mathbf{U}$ satisfying
    $(X_i \perp \{X_1, \ldots, X_{i-1}\} - \mathbf{U} \mid \mathbf{U})$
    for $\mathbf{U}' \subseteq \{X_1, \ldots, X_{i-1}\}$ **do**
      if $\mathbf{U}' \subset \mathbf{U}$ and $(X_i \perp \{X_1, \ldots, X_{i-1}\} - \mathbf{U}' \mid \mathbf{U}') \in I$ **then**
        $\mathbf{U} \leftarrow \mathbf{U}'$
    Now set $\mathbf{U}$ to be the parents of X_i
    for $X_i \in \mathbf{U}$ **do**
      Add X_j—X_i to G

Algorithm for Constructing a Minimal I-Map

Require: An ordering $X_1, \ldots, X_n$ of random variables in $\mathcal{X}$
Require: A set of independencies I
  Set G to an empty graph over $\mathcal{X}$
  for $i = 1, \ldots, n$ **do**
    $\mathbf{U}$ is the current candidate for parents of X_i
    $\mathbf{U} \leftarrow \{X_1, \ldots, X_{i-1}\}$
    Find the minimal set $\mathbf{U}$ satisfying
    $(X_i \perp \{X_1, \ldots, X_{i-1}\} - \mathbf{U} \mid \mathbf{U})$
    for $\mathbf{U}' \subseteq \{X_1, \ldots, X_{i-1}\}$ **do**
      if $\mathbf{U}' \subset \mathbf{U}$ and $(X_i \perp \{X_1, \ldots, X_{i-1}\} - \mathbf{U}' \mid \mathbf{U}') \in I$ **then**
        $\mathbf{U} \leftarrow \mathbf{U}'$
    Now set $\mathbf{U}$ to be the parents of X_i
    for $X_i \in \mathbf{U}$ **do**
      Add X_j—X_i to G
    return G

Markov Networks to Bayesian Networks



The clique tree for the graph H shown above is a chain
$\{A, B, C\} \to \{B, C, D\} \to \{C, D, E\} \to \{D, E, F\}$.

- It turns out that every undirected graph H has a clique tree T.
- If H is a nonchordal Markov network, then there is no Bayesian network G that is a perfect map for H.
- On the other hand, if H is a chordal Markov network, then there is a Bayesian network G such that $I(H) = I(G)$, and it can be found using the procedure we just discussed.

Motivation

- Consider a set of variables $\mathcal{X} = \mathbf{X} \cup \mathbf{Y}$, where $\mathbf{Y}$ is a set of *target variables* and $\mathbf{X}$ is a (disjoint) set of *observed variables*.
- The dependencies among the variables in $\mathbf{X}$ may be quite complex and poorly understood.

Motivation

- So far, we have been trying to model the joint distribution $P(\mathbf{Y}, \mathbf{X})$; we could instead try modeling the distribution $P(\mathbf{Y} \mid \mathbf{X})$.
- By introducing directed dependencies between the $\mathbf{Y}$ and $\mathbf{X}$, we can avoid encoding the distribution $P(\mathbf{X})$ altogether. In other words, we disallow any factor whose scope is a subset of $\mathbf{X}$.
- This allows us to focus on using our domain knowledge to encode a rich set of features into the model.

Conditional Random Fields

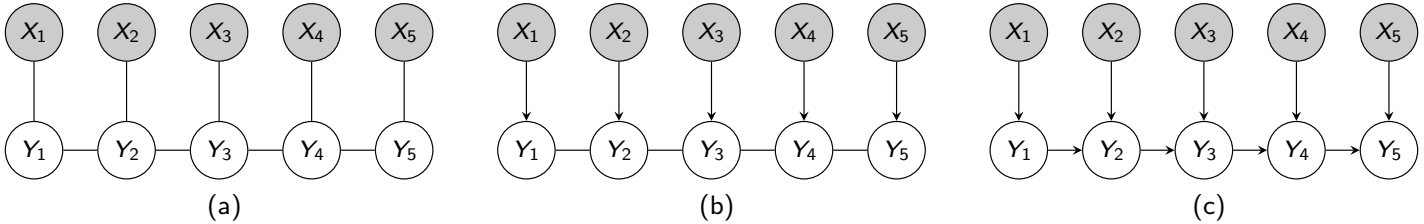
- A *conditional random field* is an undirected graph H whose nodes correspond to $\mathbf{X} \cup \mathbf{Y}$; the network is annotated with a set of factors $\phi_1(\mathbf{D}_1), \ldots, \phi_m(\mathbf{D}_m)$ such that each $\mathbf{D}_i \not\subseteq \mathbf{X}$. The network encodes a distribution as follows:

$$P(\mathbf{Y} \mid \mathbf{X}) = \frac{1}{\mathbf{Z}(\mathbf{X})} \tilde{P}(\mathbf{Y}, \mathbf{X})$$

$$\tilde{P}(\mathbf{Y}, \mathbf{X}) = \prod_{i=1}^{m} \phi_i(\mathbf{D}_i)$$

$$Z(\mathbf{X}) = \sum_{\mathbf{Y}} \tilde{P}(\mathbf{Y}, \mathbf{X}).$$

Conditional Random Fields



- The undirected CRF shown in (a) has the form:

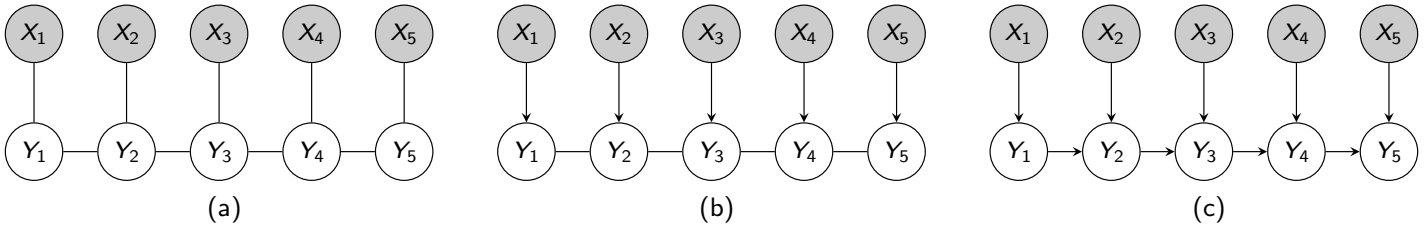
$$P(\mathbf{Y} \mid \mathbf{X}) = \frac{1}{Z(\mathbf{X})} \tilde{P}(\mathbf{Y}, \mathbf{X})$$

$$\tilde{P}(\mathbf{Y}, \mathbf{X}) = \prod_{i=1}^{k-1} \phi(Y_i, Y_{i+1}) \prod_{i=1}^{k} \phi(Y_i, X_i)$$

$$Z(\mathbf{X}) = \sum_{\mathbf{Y}} \tilde{P}(\mathbf{Y}, \mathbf{X}).$$

- The partially directed CRF shown in (b) is equivalent.

Conditional Random Fields



- The Bayesian network in (c) is not equivalent to (a) or (b), as it implies that $(Y_i \perp X_{i+1})$ (recall the common effect v-structure $X \rightarrow Z \leftarrow Y$).
- In the CRFs shown in (a) and (b), the unnormalized measure of $\mathbf{Y}$ depends on the entire parameterization of the chain (the assignment to $\mathbf{X}$).
- To encode these dependencies in (c), we would have to draw edges from all the variables in $\mathbf{X}$ to each of the variables in $\mathbf{Y}$.

Conditional Random Fields

- Factors in conditional random fields are typically parameterized as log-linear functions, which were discussed in the first talk.
- Consider a CRF over the binary-valued variables $\mathbf{X} = \{X_1, \ldots, X_k\}$ and Y, and a pairwise potential between each Y and each X_i (aka the *naive Markov model*).
- We parameterize the potentials as follows:

$$\phi_i(X_i, Y) = \exp\{w_i \mathbb{1}\{X_i = 1, Y = 1\}\}$$
$$\phi_0(Y) = \exp\{w_0 \mathbb{1}\{Y = 1\}\}.$$

Conditional Random Fields

- We have

$$\tilde{P}(Y = 1 \mid x_1, \ldots, x_k) = \exp\left\{ w_0 + \sum_{i=1}^{k} w_i x_i \right\}$$

$$\tilde{P}(Y = 0 \mid x_1, \ldots, x_k) = \exp(0) = 1,$$

so

$$\begin{aligned} P(Y = 1 \mid x_1, \ldots, x_k) \\ &= \frac{1}{Z(x_1, \ldots, x_k)} \tilde{P}(Y = 1 \mid x_1, \ldots, x_k) \\ &= \frac{\tilde{P}(Y = 1 \mid x_1, \ldots, x_k)}{\tilde{P}(Y = 0 \mid x_1, \ldots, x_k) + \tilde{P}(Y = 1 \mid x_1, \ldots, x_k)} \\ &= \text{sigmoid}\left(w_0 + \sum_{i=1}^{k} w_i x_i \right). \end{aligned}$$

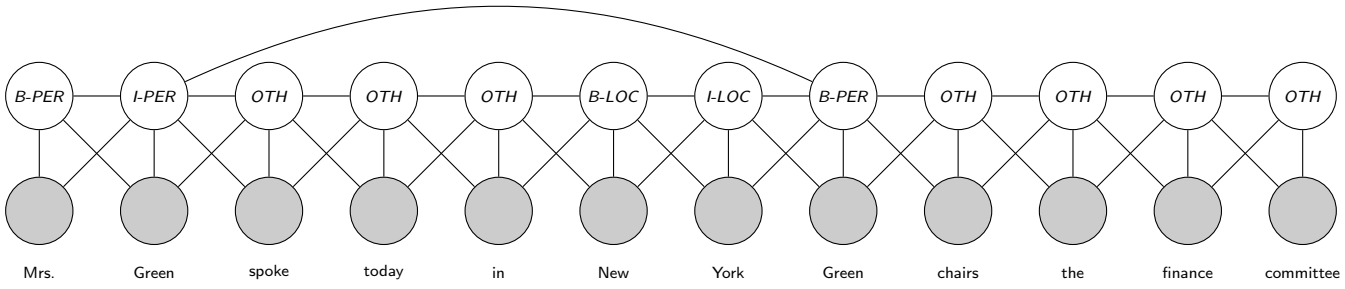
Named-Entity Recognition [2]

- Given a sentence, determine the people and organizations involved, along with their relevant locations in the sentence: "Mrs. Green spoke today in New York. Green chairs the finance committee."
- Entities sometimes span multiple words. The entity of a word is not obvious without considering its *context*.

Named-Entity Recognition [2]

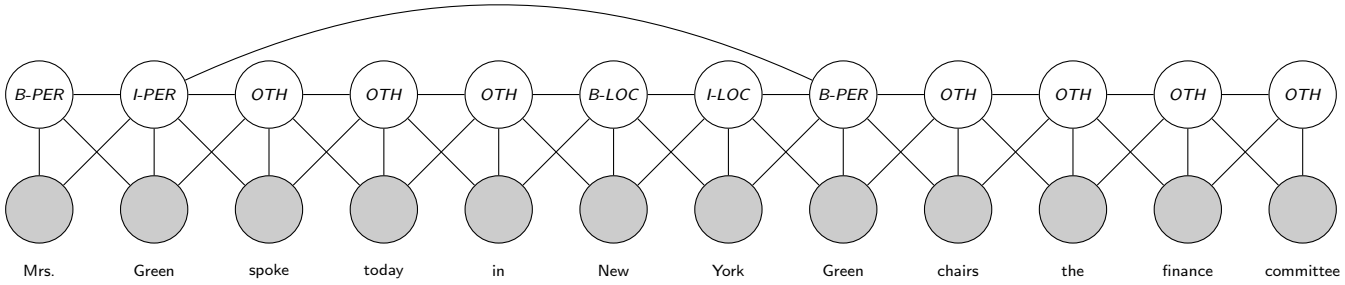
- The CRF has one variable X_i for each word, which encodes the possible labels of that word.
- The targets are, for example, "B-person", "I-person", "B-location", "I-location", "B-organization", and "I-organization".
- Having "B" mark the beginning of an entity and "I" mark the inside or end of an entity allows the model to segment adjacent entities.

NLP Example: Named-Entity Recognition [2]



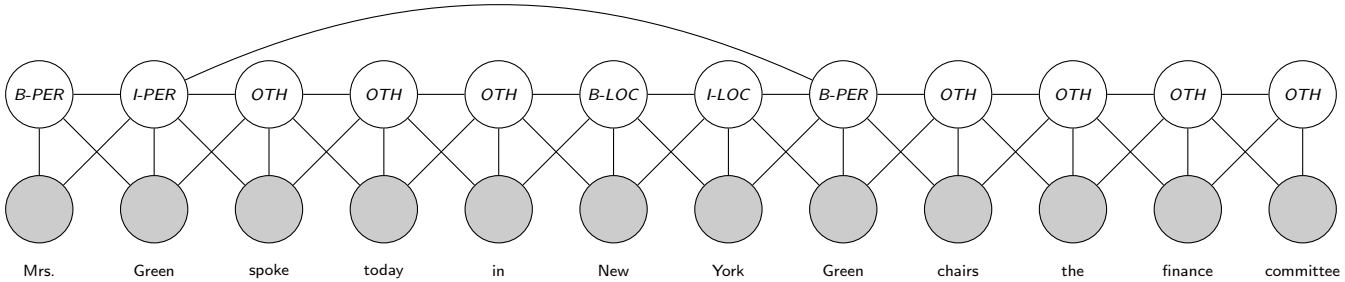
- This is typically represented by having two factors for each word:
  - $\phi^1(Y_t, Y_{t+1})$ represents the dependencies between neighboring target variables.
  - $\phi^2(Y_t, X_1, \ldots, X_T)$ represents the dependencies between a target and its context in the word sequence.

NLP Example: Named-Entity Recognition [2]



- A large number of features of the word X_t and its neighboring words are relevant to the named entity decision (e.g. whether it is capitalized, appears in a list of common person names, appears in an atlas of common names, ends in the string "ton", or is exactly the string "York" followed by the word "Times").

NLP Example: Named-Entity Recognition [2]



- These features are often sparse, though highly indicative.
- A common practice is to connect identical words together, in order to enforce that they are assigned to the same labels.
- Such a CRF is called a linear-chain CRF, and achieves accuracies in the high 90 percent range on many natural data sets.

Chain Graphs

- We can generalize our ideas used to formulate the notion of the conditional random field by allowing arbitrary, undirected *chain components* in a PDAG to be connected to one another in a directed fashion.
- We will begin by formalizing the notion of the chain component, and thus the notion of the chain graph.

Chain Graphs

- Let K be a PDAG over $\mathcal{X}$. Let $\mathbf{K}_1, \ldots, \mathbf{K}_l$ be a disjoint partition of $\mathcal{X}$ such that:
  - the induced subgraph over $\mathbf{K}_i$ contains no directed edges;
  - for any pair of nodes $X \in \mathbf{K}_i$ and $Y \in \mathbf{K}_j$ for $i < j$, an edge between X and Y can only be a directed edge $X \to Y$.
- Each component $\mathbf{K}_i$ is called a chain componment.
- Because of its chain structure, a PDAG is also called a *chain graph*.

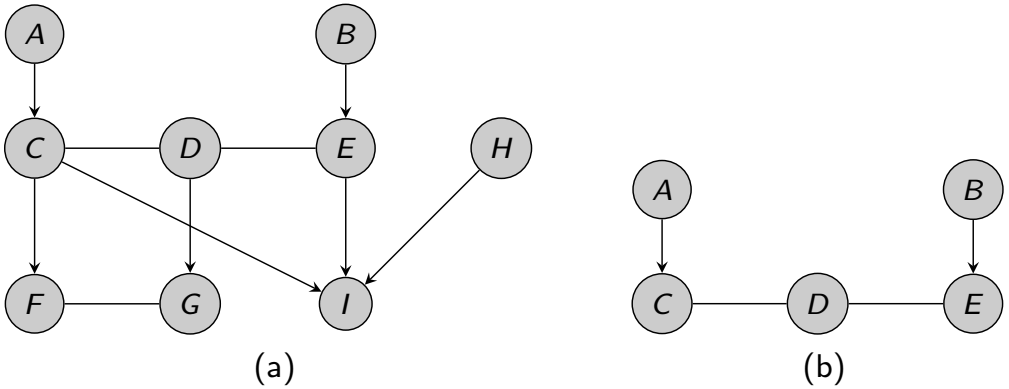
Chain Graphs

- In a PDAG, we have both the notion of parents of X (variables Y such that $Y \to X$) and neighbors of X (variables Y such that Y—X).
- The union of these two sets is the *boundary* of X, denoted Boundary_X.

Chain Graphs

- We say that a subset of nodes $\mathbf{X} \in \mathcal{X}$ is *upwardly closed* in K if, for any $X \in \mathbf{X}$, we have that $\text{Boundary}_X \subset \mathbf{X}$.
- We define the *upward closure* of $\mathbf{X}$ to be the minimal upwardly closed subset $\mathbf{Y}$ that contains $\mathbf{X}$.
- The definition of the upward closure of X is somewhat subtle. Obviously the upward closure $\mathbf{Y}$ must contain $\mathbf{X}$. But this means that for all $U \in \text{Boundary}_{\mathbf{X}}$, we must also have $U \in \mathbf{Y}$.
- So to obtain the upward closure $\mathbf{Y}$ of $\mathbf{X}$, we must recursively add the boundaries of all nodes in $\mathbf{Y}$ until we cannot proceed any further.

Chain Graphs



- The *upwardly closed subgraph* of $\mathbf{X}$, denoted $K^+[\mathbf{X}]$, is the induced subgraph over $\mathbf{Y}$, $K[\mathbf{Y}]$.
- Recursively expanding the boundary of C in (a) yields the upward closure of C, which is $\{A, B, C, D, E\}$.
- The upwardly closed subgraph of C, $K^+[\{A, B, C, D, E\}]$, is shown in (b).

Chain Graphs

- Let K be a PDAG and $\mathbf{K}_1, \ldots, \mathbf{K}_l$ be its chain components. We define $\mathrm{Pa}_{\mathbf{K}_i}$ to be the parents of the nodes in $\mathbf{K}_i$. The moralized graph of K is an undirected graph $\mathcal{M}[K]$ produced by first connecting, using undirected edges, any pair of nodes $X, Y \in \mathrm{Pa}_{\mathbf{K}_i} \forall i = 1, \ldots, l$ and then converting all directed edges into undirected edges.
- The definition of the moralized graph here is the same as the the one discussed in the first talk, even for PDAGs.

Chain Graphs

- Let K be a PDAG, and $\mathbf{K}_1, \ldots, \mathbf{K}_l$ be its chain components. A chain graph distribution is defined via a set of factors $\phi_i(\mathbf{D}_i)$ $(i = 1, \ldots, m)$ such that each $\mathbf{D}_i$ is a complete subgraph in the moralized graph $\mathcal{M}[K^+[\mathbf{D}_i]]$.
- We associate each factor $\phi_i(\mathbf{D}_i)$ with a single chain component $\mathbf{K}_i$ such that $\mathbf{D}_i \subseteq \mathbf{K}_i \cup \mathrm{Pa}_{\mathbf{K}_i}$, and define $P(\mathbf{K}_i)$ as a CRF with these factors, and with $\mathbf{Y}_i = \mathbf{K}_i$ and $\mathbf{X}_i = \mathrm{Pa}_{\mathbf{K}_i}$.

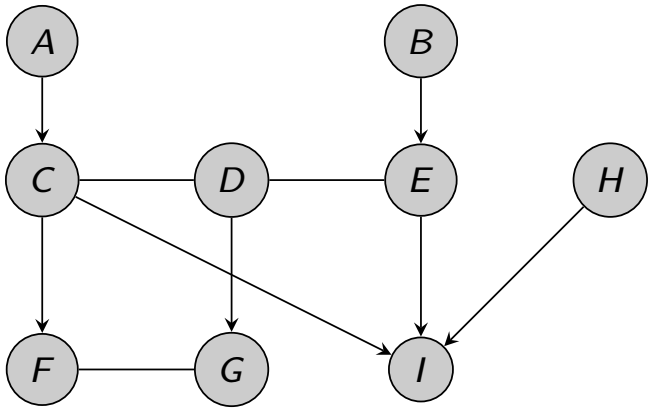
Chain Graphs

- We now define

$$P(\mathcal{X}) = \prod_{i=1}^{I} P(\mathbf{K}_i \mid \mathrm{Pa}_{\mathbf{K}_i}).$$

- We say that a distribution P factorizes over K if it can be represented as a chain graph distribution over K.

Chain Graphs



- In the chain graph shown above, we must have

$$P(C, D, E \mid A, B) = \frac{1}{Z(A, B)} \phi_1(A, C)\phi_2(B, E)\phi_3(C, D)\phi_4(D, E).$$

- A similar factorization applies to $P(F, G \mid C, D)$.

Independence

- Now that we know what a chain graph is, we will use our results from directed and undirected graphs to analyze how independence flows in a graph that incorporates both.
- Repeated for convenience: Boundary_X is defined as the union of the parents of X (all Y such that $Y \to X$) and neighbors of X (all Y such that Y—X).

Independence

- We define Descendants_X to be the set of nodes that are reachable from X via a directed path (one that involves at least one directed edge from X to Y, but no directed edges from Y to X).
- It follows that if Y is a descendant of X, then Y must be in a "lower" chain component.

Independence

- For a PDAG K, we define the pairwise independencies associated with K to be:

$$I_p(K) = \{(X \perp Y \mid (\text{NonDescendants}_X - \{X, Y\})) : X, Y \textit{ non-adjacent}, Y \in \text{NonDescendants}_X\}.$$

- This generalizes the notion of pairwise independencies in directed graphs to undirected graphs: in an undirected graph, $\text{NonDescendants}_X = \mathcal{X}$.

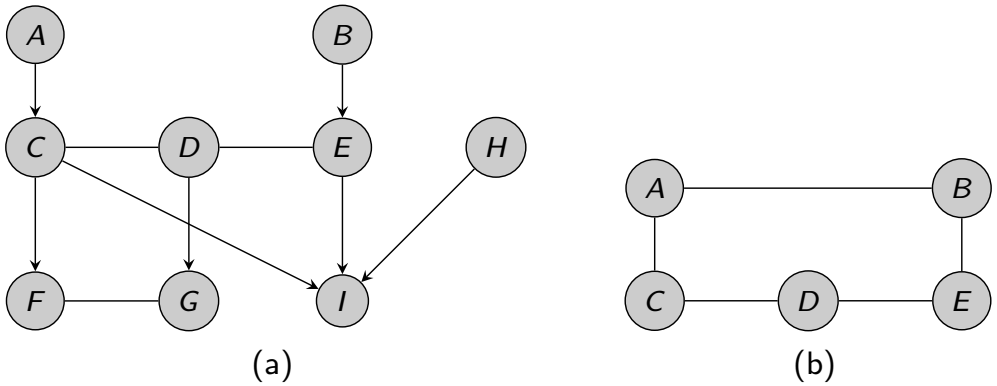
Independence

- For a PDAG K, we define the local independencies associated with K to be:

$$I_\ell(K) = \{(X \perp \text{NonDescendants}_X - \text{Boundary}_X \mid \text{Boundary}_X : X \in \mathcal{X})\}.$$

- This generalization follows directly from the discussion of Markov blankets that was presented in the first talk.
- For directed graphs, NonDescendants_X is precisely the set of nondescendants, whereas $\text{Boundary}_X = \pi(X)$.
- For undirected graphs, $\text{NonDescendants}_X = \mathcal{X}$, whereas $\text{Boundary}_X = \text{Nb}_X$.

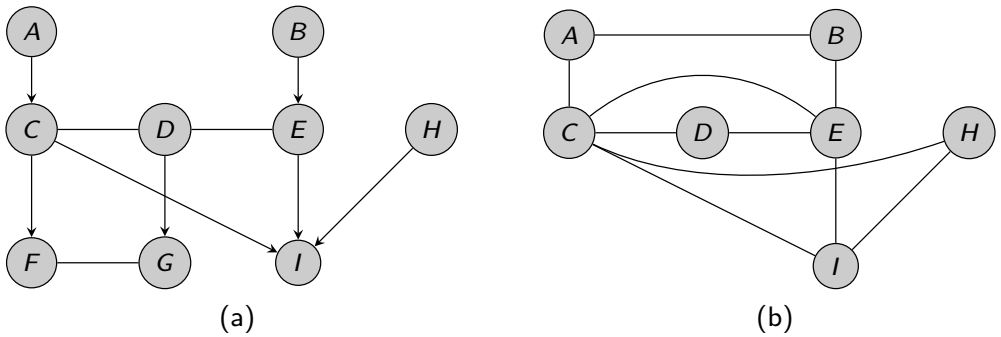
Independence



A chain graph K and $\mathcal{M}[K^+[\{C, D, E\}]]$.

- Let $\mathbf{X}, \mathbf{Y}$ and $\mathbf{Z}$ be three disjoint sets, and let $\mathbf{U} = \mathbf{X} \cup \mathbf{Y} \cup \mathbf{Z}$. We say that $\mathbf{X}$ is c-separated from $\mathbf{Y}$ given $\mathbf{Z}$ if $\mathbf{X}$ is separated from $\mathbf{Y}$ given $\mathbf{Z}$ in the undirected graph $\mathcal{M}[K^+[\mathbf{X} \cup \mathbf{Y} \cup \mathbf{Z}]]$.
- In the figure above, C is c-separated from E given D, A, since C is c-separated from E given D, A in the undirected graph.
- However, C is not c-separated from E given only D, since there a path from C to E in the undirected graph.

Independence



A chain graph K and $\mathcal{M}[K^+[\{C, D, E, I\}]]$.

- The introduction of I into the set of nodes in consideration results in a direct edge between C and E in the moralized graph, so we cannot render them independent by observing any of the other nodes.

Independence

- Let K be a PDAG. We define the global independencies associated with K to be:

$$I(K) = \{(\mathbf{X} \perp \mathbf{Y} \mid \mathbf{Z} : \mathbf{X}, \mathbf{Y}, \mathbf{Z} \subset \mathcal{X},$$
$$\mathbf{X} \textit{ is c-separated from } \mathbf{Y} \textit{ given } \mathbf{Z})\}.$$

- As in undirected graphs, if the distribution is positive:
  - The three forms of independence and equivalent.
  - P factorizes over a PDAG K if and only if $P \models I(K)$.

- Daphne Koller and Nir Friedman, *Probabilistic Graphical Models: Principles and Techniques*, MIT Press, 1st Edition, 2009.
- David Sontag, lecture slides on Probabilistic Graphical Models. Accessible at `http://cs.nyu.edu/~dsontag/courses/pgm12/`.
- Nir Friedman, lecture slides on the theory of Bayesian networks. Accessible at `classes.soe.ucsc.edu/cmps290c/Spring04/paps/nir2.pdf`.