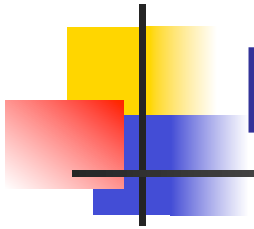
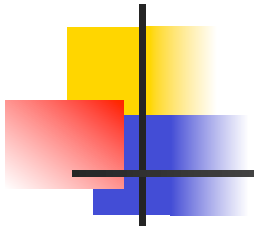


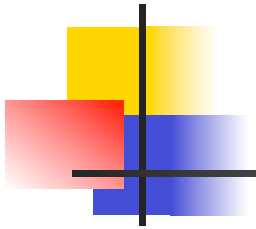
Reasoning with Uncertainty

Graphical Models



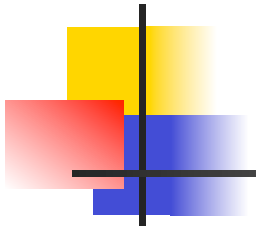
Inference Complexity and Dependence

- Inference in probabilistic and belief systems is computationally complex
  - Probabilistic inference is exponential in the number of random variables
  - Dempster-Schafer is doubly exponential in the number of state attributes (polynomial in state subsets)
- Independence relations can reduce this complexity
  - Conditional independence in probability limits the number of random variables that have to be considered to make inference



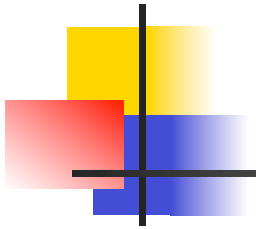
Graphical Models

- Graphical models provide an efficient structure to represent dependencies in probabilistic (and much less well developed) belief systems.
- There are two main types of graphical models for probabilistic systems:
  - Bayesian Networks are directed graphical models
  - Markov Networks (Markov Random Fields) are undirected graphical models
- Both types of models can represent different types of dependencies



Graphical Models for Probabilistic Inference

- Graphical models in probabilistic systems allow to represent the interdependencies of random variables
  - Structure shows dependency relations
  - Inference can use the structure to control the computations
- Graphical models provide a basis for a number of efficient problem solutions
  - Inference of prior and conditional probabilities
  - Learning of network structure



Bayesian Networks

- Bayesian networks are graphical representation for conditional independence providing a compact specification of joint probability distributions
  - Bayesian networks are directed, acyclic graphs
    - Nodes represent random variables

      $$N = \{X_i \mid 1 \le i \le n\}$$

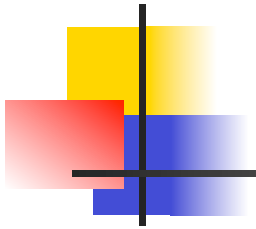
    - Links represent “direct influences”

      $$A = \{(X_{s_j}, X_{e_j}) \mid X_{s_j} \text{ and } X_{s_j} \text{ "directly influence each other"}\}$$

    - Nodes are annotated with the conditional probability distribution of the node given its parents

      $$P(X_i \mid Parents(X_i))$$

  - Probabilities in the network represent joint distribution



Markov Networks

- Markov networks (Markov Random Fields) are graphical representation for conditional independence, providing a compact specification of joint probability distributions
  - Markov networks are undirected graphs
    - Nodes represent random variables

      $$N = \{X_i \mid 1 \le i \le n\}$$

    - Links represent dependencies (nodes are not pairwise Markov)

      $$A = \{(X_{s_j}, X_{e_j}) \mid X_{s_j} \text{ and } X_{s_j} \text{ are not conditionally independ}\}$$

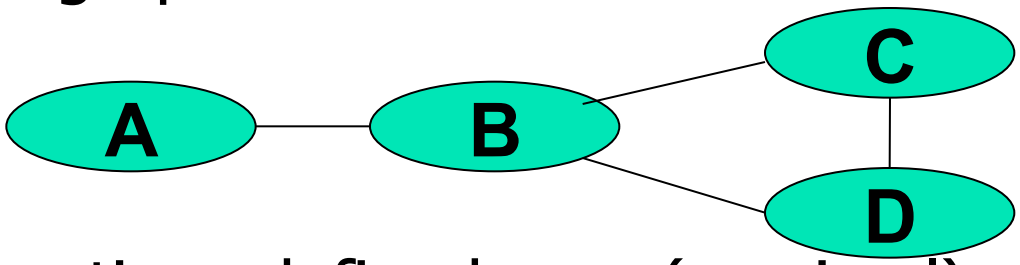
    - Cliques in the graph are annotated with "clique potentials" that allow to compute the probability distributions

      $$\phi_i(X_{\{i\}})$$

  - Probabilities induced represent joint distribution

Markov Networks

- Undirected graphical models

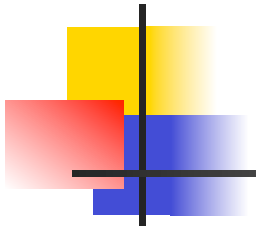


- Potential functions defined over (maximal) cliques
  - 2 maximal cliques: {A,B} , {B,C,D}

$$P(x) = \frac{1}{Z}\prod_c \phi_c(x_c)$$

$$Z = \sum_x \prod_c \phi_c(x_c)$$

A	B	ϕ(A,B)
False	False	3.5
False	True	6.5
True	False	2.1
True	True	4.3



Hammersley-Clifford Theorem

- Hammersley-Clifford Teorem states that for every distribution that is not 0 for any item, there is a corresponding Markov network

  If Distribution is strictly positive (P(x) > 0)

  And Graph encodes conditional independences

  Then Distribution is product of potentials over cliques of graph

- Inverse is also true
  - This comes from the fact that the Markov network represents the Gibbs distribution

Markov Networks

- Using the fact that all potentials are non-zero and that the probability is exponentiated log likelihood, the potential can be represented in log-linear form

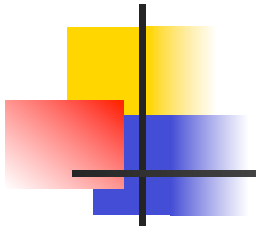
$$P(x) = 1/Z * e^{\sum_{i,j} w_{i,j} f_{i,j}(x)}$$

  - Features $f_{i,j}(x)$ can, for example, be selected to be indicators that the state matches a particular assignment to a specific clique:

$$f_{i,j}(x) = \begin{cases} 1 & \text{if the variables of x in clique i match the j}^{\text{th}} \text{ value assignment} \\ 0 & \text{otherwise} \end{cases}$$

$$w_{i,j} = \log(\phi_i(x_i))$$

    - Other feature choices can result in a more compact representation

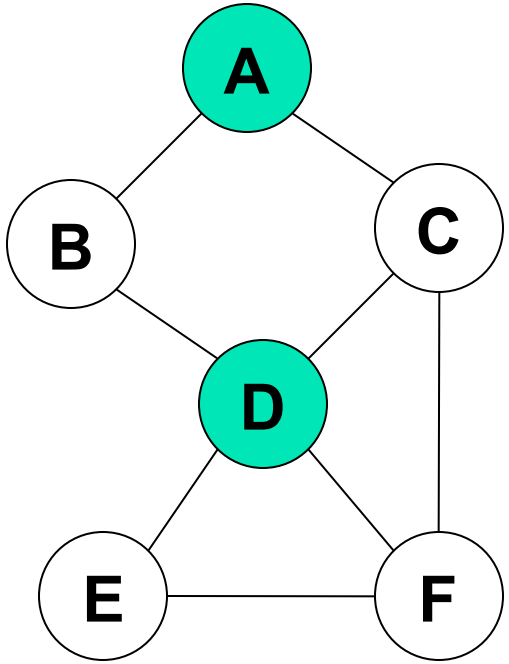


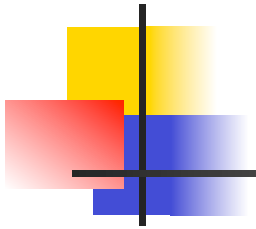
Markov Nets vs. Bayes Nets

- Bayesian and Markov Networks can be used to represent probability distributions. However, they have a different representation and different limitations to encode dependency relations in the structure of the network.
  - Markov networks can represent cyclical dependency relations. Bayesian networks can represent induced dependencies
  - In both graphical models the distribution is represented as a product of potentials
    - Potentials in Bayesian networks are conditional probabilities
  - Independence is achieved through the Markov blanket which is more local in a Markov network

Independence in Markov Networks

- Two nodes in a Markov network are independent if and only if there is no path between them that does not cross an observed (evidence) variable
  - E.g. nodes B and C are independent when A and D are observed
  - E.g. nodes B and E are independent when node A and D are observed
  - E.g. nodes C and E are not independent





Markov Blanket

- In a Markov network, the Markov blanket of a node consists of that node and its neighbors
  - Markov blanket in a Markov network is easier to represent and analyze than in a Bayesian network since it does not include any nodes at a distance larger than 1.
    - In a Bayesian network there can be a relation through parents of children and thus a remaining dependence is much more difficult to evaluate.
    - Dependencies in Markov networks are inherently local

Converting Between Bayesian and Markov Network Structure

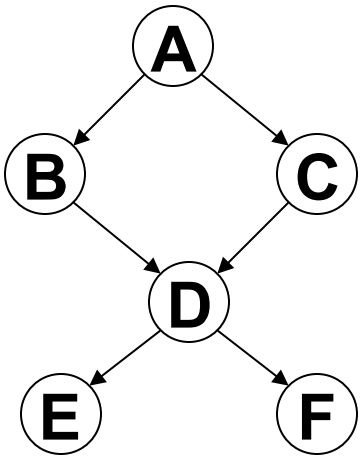
- Bayesian and Markov networks can both represent arbitrary probability distributions
- To convert between the two network types a number of properties have to be considered
  - Same data flow must be maintained in the conversion
    - Sometimes new dependencies must be introduced to maintain data flow
  - For efficient conversion of the structure independent of the specific potential functions the original set of immediate dependencies has to be maintained
    - When converting to a Markov network, the dependencies of Markov net must be a superset of the Bayes net dependencies.

      $I(Bayes) \subseteq I(Markov)$
    - When converting to a Bayes net the dependencies of Bayes net must be a superset of the Markov net dependencies.

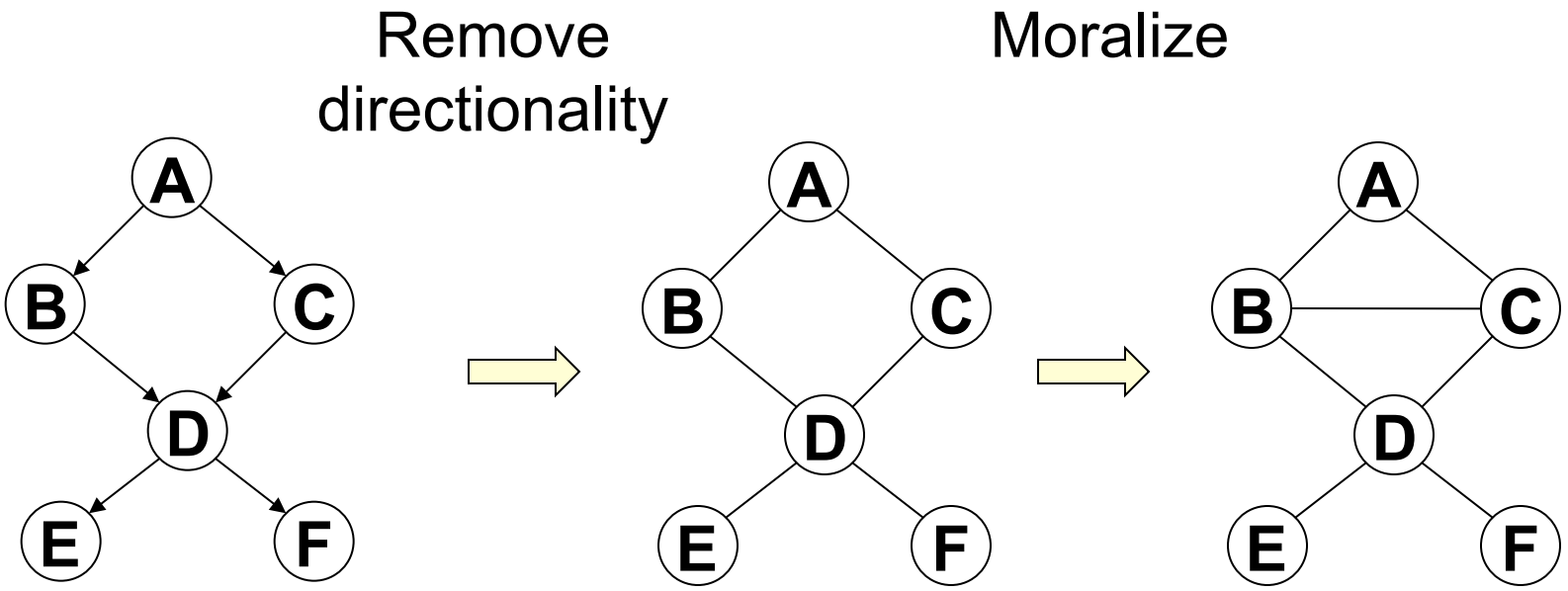
      $I(Markov) \subseteq I(Bayes)$

Converting Bayesian Networks to Markov Networks

- Conversion from directed to undirected model has to maintain the dependencies independent of the evidence (observed variables)
  - All direct dependencies in the Bayesian network are potential dependencies in the Markov network
  - Structure must be able to handle any evidence.
    - Difference in Markov blanket leads to additional potential dependencies. If a child of a node is observed, the parents of this child are not independent. E.g. if D is observed there is still data flowing between B and C.
    - Common parents of a node have to be connected to represent the additional dependencies ("moralizing")

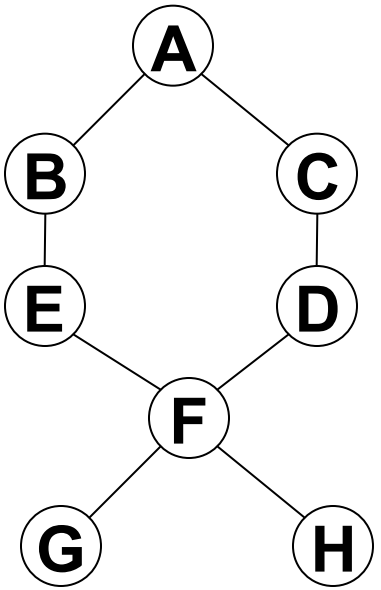


Converting Bayesian Networks to Markov Networks



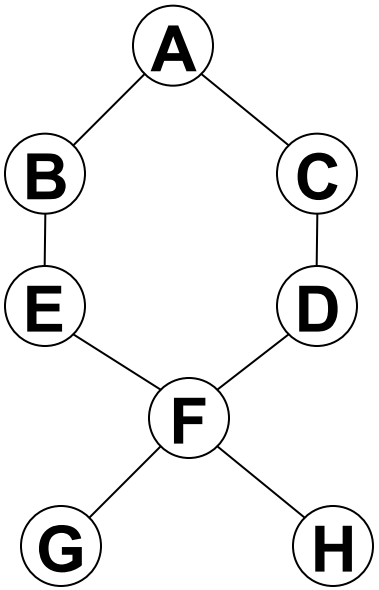
Converting Markov Networks to Bayesian Networks

- Conversion from undirected to directed model has to maintain the dependencies and independence relations, independent of the evidence (observed variables)
  - All direct dependencies in the Markov network are potential dependencies in the Bayesian network
  - Structure has to be able to handle any evidence and preserve dependencies and independence relations
    - Differences in Markov blanket imply that observations of children nodes can cause conditional independence in Markov networks that are not mirrored in a Bayesian network with the same connectivity



Converting Markov Networks to Bayesian Networks

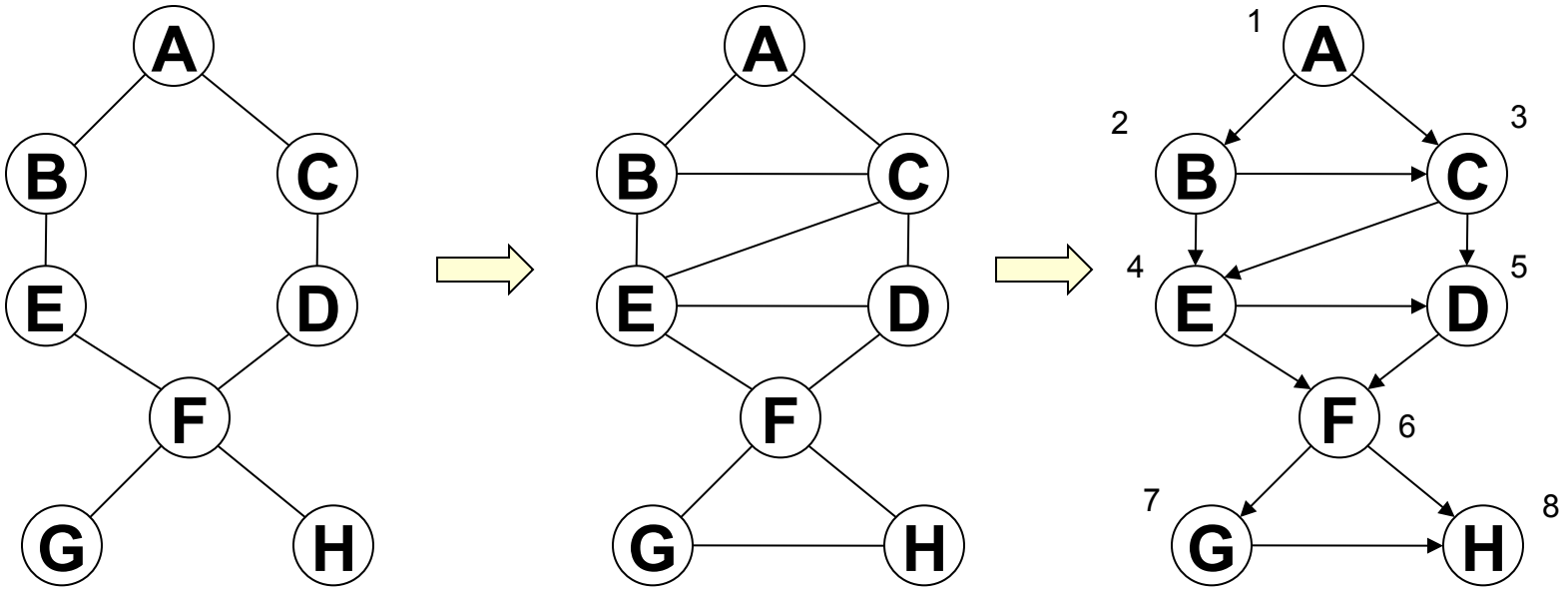
- Structure has to be able to handle any evidence and preserve dependencies and independence relations
  - Difference in the Markov blanket implies that while parents of a common observed node are dependent in a Bayesian network, they are independent in the Markov network (given the remainder of the Markov blanket). E.g. if A and F are observed no information flows between B and C in the Markov network but does in a Bayesian network.
  - To address this, additional connections between the common descendants of a node have to be used to allow the representation of independence by compensating the influence ("triangulation")
- Resulting structure has to be made directional and acyclic through topological sorting of nodes

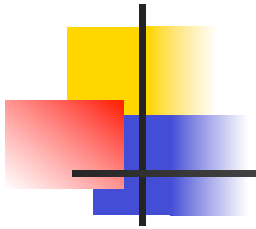


Convert Bayesian Networks to Markov Networks

Triangulation

Topological sorting
And adding
directionality



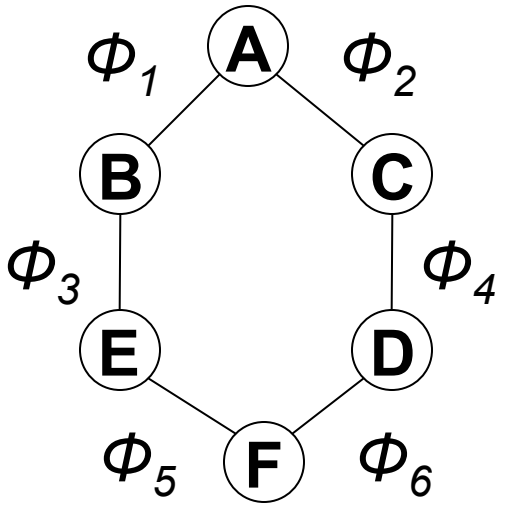


Inference in Markov Networks

- Inference in Markov networks is similar to inference in Bayesian networks
  - Analytic inference in Markov networks is *#P* complete and thus intractable in general (Note that this was true for Bayesian networks, too.)
    - Particular classes of Markov networks are solvable in Polynomial time but the exact definition of these classes is still research (in Bayesian networks the class of all singly connected networks is polynomial)
  - Monte Carlo inference provides approximate inference and is more tractable
    - Markov Chain Monte Carlo is most commonly used here and simpler than in Bayesian networks due to the more local dependencies

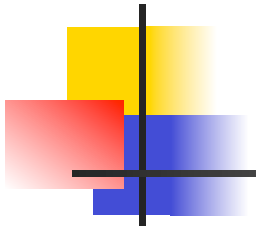
Variable Elimination

- Variable elimination in Markov networks works similar to Bayesian networks
  - Given a set of potentials, Φ_i, defined over all maximal cliques
    - Change all clique potentials for clique assignments that are inconsistent with the observed variables to 0
    - Combine and marginalize potentials in the same way as for Bayesian network variable elimination



Markov Chain Monte Carlo

- Markov Chain Monte Carlo (MCMC) is a common family of inference algorithms for Markov networks
  - Metropolis-Hastings is a very general algorithm (with some attributes of rejection sampling) but uses more samples than required and does not take full advantage of local Markov blanket
    - Samples the next state given the current one according to the transition probabilities in the MCMC model
    - Reject the new state with a given probability to maintain balance
  - Gibbs sampling is the most popular algorithm
    - Gibbs sampling resamples each of the non-observed variables sequentially, treating all other variables as if they were observed according to the values in the last generated sample
      - This implies that all variables in the Markov blanket are observed and the transition probability can therefore be computed by looking only at the potentials of cliques containing the variable that is being resampled



Gibbs Sampling

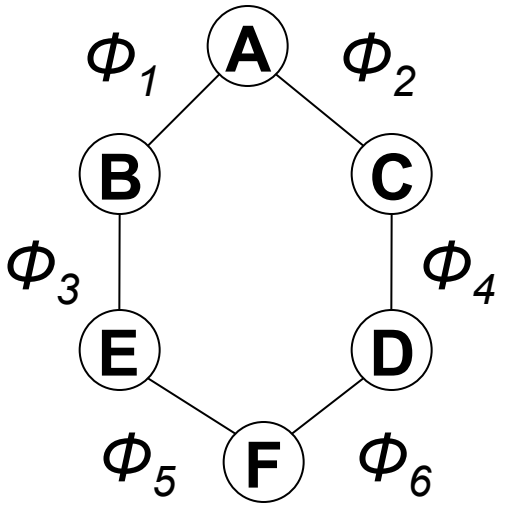
- Gibbs sampling is one of the simplest MCMC algorithms where samples are generated by sampling one variable at a time while considering all others to have the previous value
  - Initial samples depend on starting point and are discarded (burn-in)
  - Samples generated later on represent the distribution
    - After burn-in different sample selection strategies can be used
      - Use all samples generated in consecutive sampling steps
        - Consecutive samples are not independent (only one variable changed) and thus a large number of samples is needed to correctly represent the distribution
        - Generation of a sample only requires sampling one variable
      - Use only every n^{th} sample (i.e. after sampling n variables – often n is chosen to be the number of random variables in the system)
        - If n is large enough the samples will be approximately independent
        - A smaller number of samples can represent the distribution but generation of each sample requires n sampling steps
      - Generate one sample per run from different, random starting points
        - Samples are independent
        - Sample generation is extremely expensive since it needs on burn-in per sample

Gibbs Sampling

- Assume P(A | C, E), i.e. C, E are observed
  - Since no data flows from D and F to A if C and E are observed, these variables do not influence P(A | C, E) and only the other variables have to be sampled
- Start with a random sample, e.g.

A	B	C	D	E	F
0	0	1	x	1	x

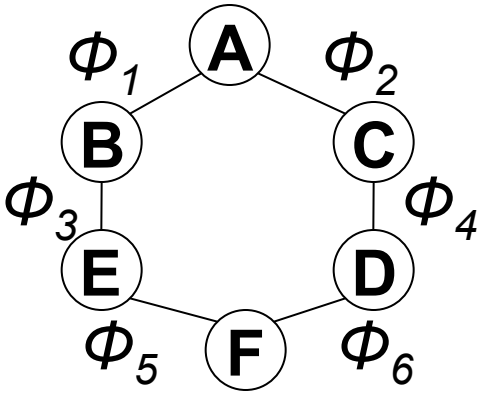
- To generate the next samples, all relevant variables (A and B) have to be resampled
  - Sampling order of variables can either be random or in a fixed order (they have to be sampled uniformly)



Gibbs Sampling

- Picking a fixed order (A, B), first A and then B are resampled assuming that all other variables are observed at their current value
  - Sampling A requires P(A | ~B,C,E).
    Due to the definition of the Markov blanket in Markov networks this requires only the potentials of cliques that contain the node that is being sampled (Φ_1,Φ_2)

A	B	C	D	E	F
0	0	1	x	1	x



Φ_1:

	a	¬a
b	1	5
¬b	4.3	0.2

Φ_2:

	a	¬a
c	1	2
¬c	3	4

$\Phi_1 x \Phi_2$:

a	¬a
12.9	0.8

Normalized probability to sample A:

a	¬a
0.94	0.06

Gibbs Sampling

- Assume that sampling resulted in A. B is resampled next, requiring P(B | A,C,E).

  This requires the potentials of cliques Φ_1, Φ_3

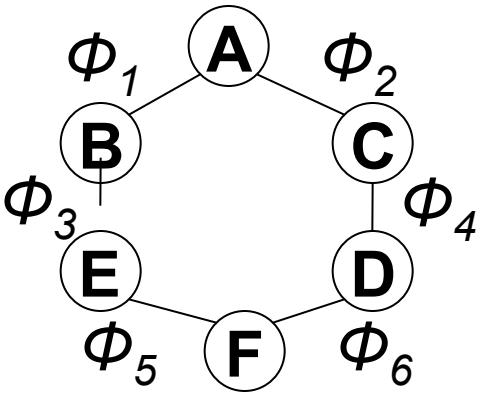
A	B	C	D	E	F
0	0	1	x	1	x
1	0	1	x	1	x

Φ_1:

	a	¬a
b	1	5
¬b	4.3	0.2

Φ_3:

	e	¬e
b	1	2
¬b	2	1



⇨ $\Phi_1 x \Phi_3$:

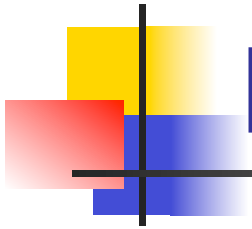
b	¬b
1	8.6

Normalized probability to sample B:

b	¬b
0.11	0.89

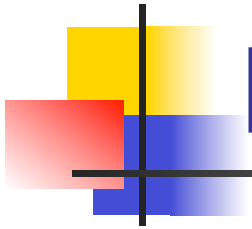
- Assume ~B is sampled. This completes the first full Gibbs sampling iteration (over all relevant variables)
  - Depending on sample retention policy either all samples from consecutive sampling or only the ones at the end of full iterations are retained

A	B	C	D	E	F
0	0	1	x	1	x
1	0	1	x	1	x
1	0	1	x	1	x



Learning Markov Networks

- As with Bayesian networks, Markov networks can be learned from data
  - Parameter learning
    - Given the connectivity, learn the potentials (or the weights in the log-linear model)
      - Learning a generative model
        - No assumptions are made which variables will be observed and which ones will be inferred. Therefore the complete joint probability distribution has to be inferable
      - Learning a discriminative model
        - Which variables are being observed is assumed to be known. Therefore only conditional probabilities for the other variables have to be inferred – Conditional Markov Networks / Conditional Random Fields (CRF)



Learning Markov Networks

- Structure learning
  - Learning of the connectivity / cliques in the network (or of features in the log-linear model)

Generative Parameter Learning

- Given data D, maximize the likelihood, *P(D | Φ)*, that the network would generate the data
  - Generally formulated using log likelihood

$$\log\left(P(D\mid\Phi)\right)=\log\left(\prod_{d\in D}P(d\mid\Phi)\right)=\sum_{d\in D}\log\left(\frac{1}{Z}\prod_{c}\phi_c(d_c)\right)=\sum_{d}\left(\sum_{c}\log\left(\phi_c(d_c)\right)-\log(Z)\right)$$

    - In contrast to Bayesian networks where the model parameters (and thus the maximum of the likelihood) can be solved analytically, this maximum has to be found here numerically using optimization (since *Z* does not decompose over network parameters)
    - Given that $c_{i,j}$ represents the j^{th} assignment to the variables in clique *i* and $N_{i,j}$ represents the number of data items which match this variable assignment, the derivative can be determined

$$\frac{\partial\log\left(P(D\mid\Phi)\right)}{\partial\Phi_i(c_{i,j})}=\frac{N_{i,j}}{\phi_i(c_{i,j})}-\frac{N*P(c_{i,j}\mid\Phi)}{\phi_i(c_{i,j})}=\frac{N_{i,j}}{\phi_i(c_{i,j})}-\frac{E\left[N_{i,j}\mid\Phi\right]}{\phi_i(c_{i,j})}$$

Generative Parameter Learning

- The derivative requires the computation of the expected number of data samples that have a particular assignment to each clique (and thus a network inference) in each iteration of the optimization.

$$\frac{\partial \log\left(P(D \mid \Phi)\right)}{\partial \Phi_i(c_{i,j})} = \frac{N_{i,j}}{\phi_i(c_{i,j})} - \frac{N * P(c_{i,j} \mid \Phi)}{\phi_i(c_{i,j})} = \frac{N_{i,j}}{\phi_i(c_{i,j})} - \frac{E\left[N_{i,j} \mid \Phi\right]}{\phi_i(c_{i,j})}$$

  - Derivative for log-linear form where features $f_{i,j}$ are strict indicators (i.e. 1 if the feature is present and 0 otherwise) is even simpler (but still requires inference)

$$\frac{\partial \log\left(P(D \mid w)\right)}{\partial w_{i,j}} = N_{f_{i,j}} - N * P(f_{i,j} \mid w) = N_{f_{i,j}} - E\left[N_{f_{i,j}} \mid w\right]$$

- Standard optimization approaches can be used to solve for parameters (fixed-point iteration, gradient ascent, …)
  - Parameter learning is relatively slow since it requires complete inference in each optimization iteration

Pseudolikelihood Learning

- Faster (approximate) parameter learning can be achieved by optimizing the pseudolikelihood PL(D|Φ) rather than the likelihood

$$PL(X) \equiv \prod_i P(x_i \mid x_{j_1},\ldots,x_{j_k} : x_{j_l} \in \text{Markov Blanket of } x_i)$$

  - Derivative of log pseudolikelihood only requires computation of pseudolikelihoods of clique assignments which does not require network inference
  - Pseudolikelihood is a consistent estimator and thus results in consistent approximations to the parameters
    - Parameters work well for basic inference but might not result in very good approximations for long inference chains (errors accumulate)

Discriminative Parameter Learning

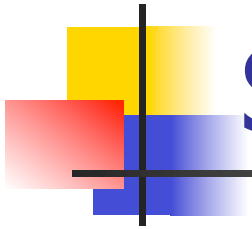
- Given data *D* and a subset of the variables, X_o, that will always be observed, maximize the conditional likelihood, $P(D \mid \Phi, X_o)$, that the network would generate the data
  - Generally formulated again using log likelihood

$$\log\left(P(D \mid \Phi, x_o)\right) = \log\left(\prod_{d \in D} P(d \mid \Phi, x_o)\right) = \sum_{d \in D} \log\left(\frac{1}{Z(x_o)} \prod_c \phi_c(d_c, d_o)\right)$$

$$= \sum_d \left(\sum_c \log\left(\phi_c(d_c, d_o)\right) - \log(Z(d_o)\right)$$

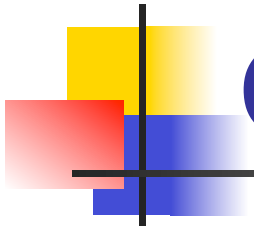
    - In a log-linear representation, and assuming using $N_{x,j,xo}$ to represent the number of data items that match feature $f_{i,j}$ and correspond to observation xo, the derivative of the conditional probability for the CRF can be computed and optimization be used (again requiring inference in each step)

$$\frac{\partial \log\left(P(D \mid w, x_o)\right)}{\partial w_{i,j}} = N_{i,j,x_o} - N_{x_o} * P(c_{i,j} \mid w, x_o) = N_{i,j,x_o} - E\left[N_{i,j,x_o} \mid w\right]$$



Structure Learning

- As in Bayesian networks the structure can be learned from data by evaluating dependencies
  - Adding nodes in a pre-determined order is not possible in Markov networks since there is no directionality and pairwise markov can not be evaluated on a subset of the variables
  - Stucture learning in Markov networks can be performed in a log-linear model by aggregating features such as to maximize the probability of the data
    - Start with atomic features
    - Greedily combine features to improve score
      - Need to reestimate parameters for each new candidate
      - Approximation: Keep weights of previous features constant



Graphical Models

- Graphical models provide compact ways to represent probability distributions by using information about dependencies
  - Bayesian networks are directed models that represent the joint distribution in terms of conditional probabilities
  - Markov networks are undirected models that represent the joint distribution in terms of clique potentials
- Different causal relations can be represented with different efficiency in the two types of models
- For both models, algorithms exist to make arbitrary probabilistic inferences and to learn the parameters and the structure from