Identifying Mixtures of Bayesian Network Distributions

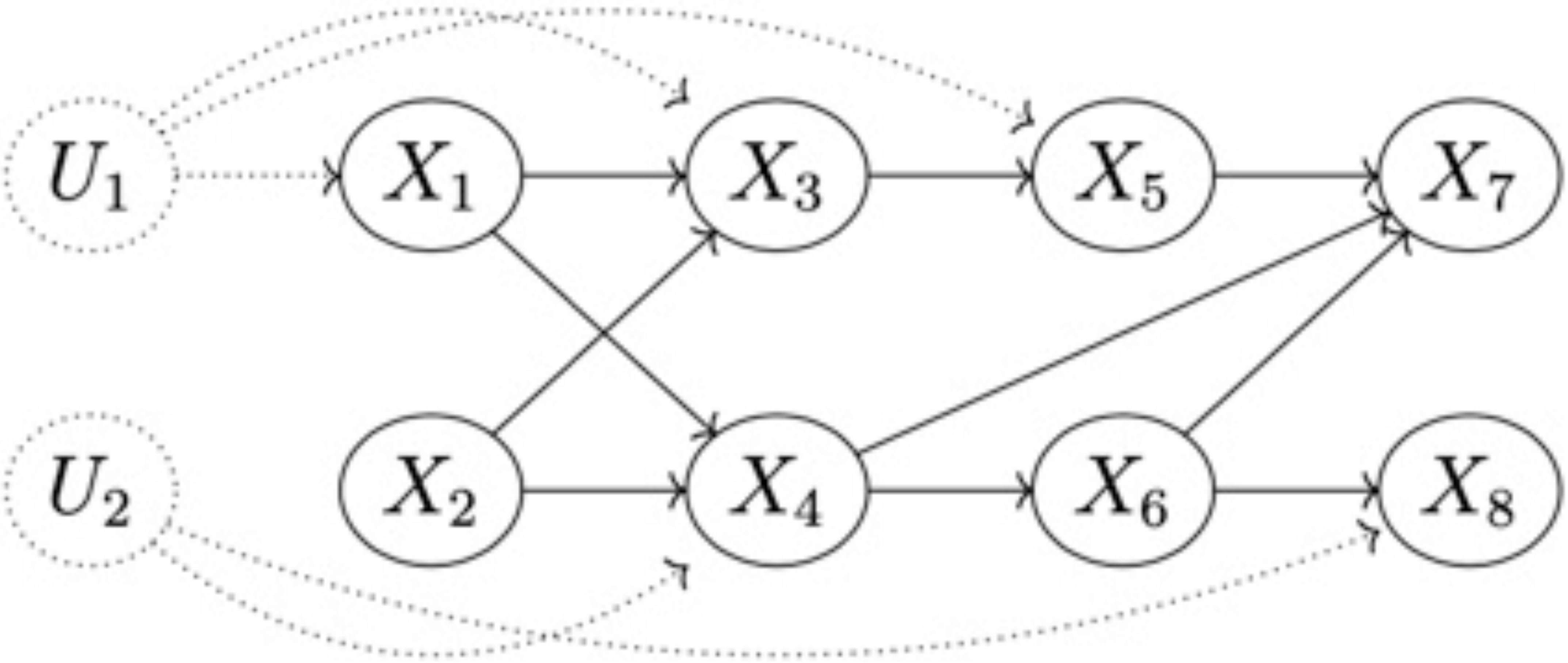
Yuval Rabani - The Hebrew University of Jerusalem

Joint work with
Spencer L. Gordon, Bijan Mazaheri, Leonard J. Schulman - Caltech

Bayesian networks [Pearl 1985]

- A directed acyclic graph G, the nodes are random variables
- The joint probability distribution is Markovian with respect to G:

$$\Pr[X_1=x_1, X_2=x_2, \ldots, X_n=x_n] = \prod_i \Pr[X_i=x_i \mid pa(X_i)]$$



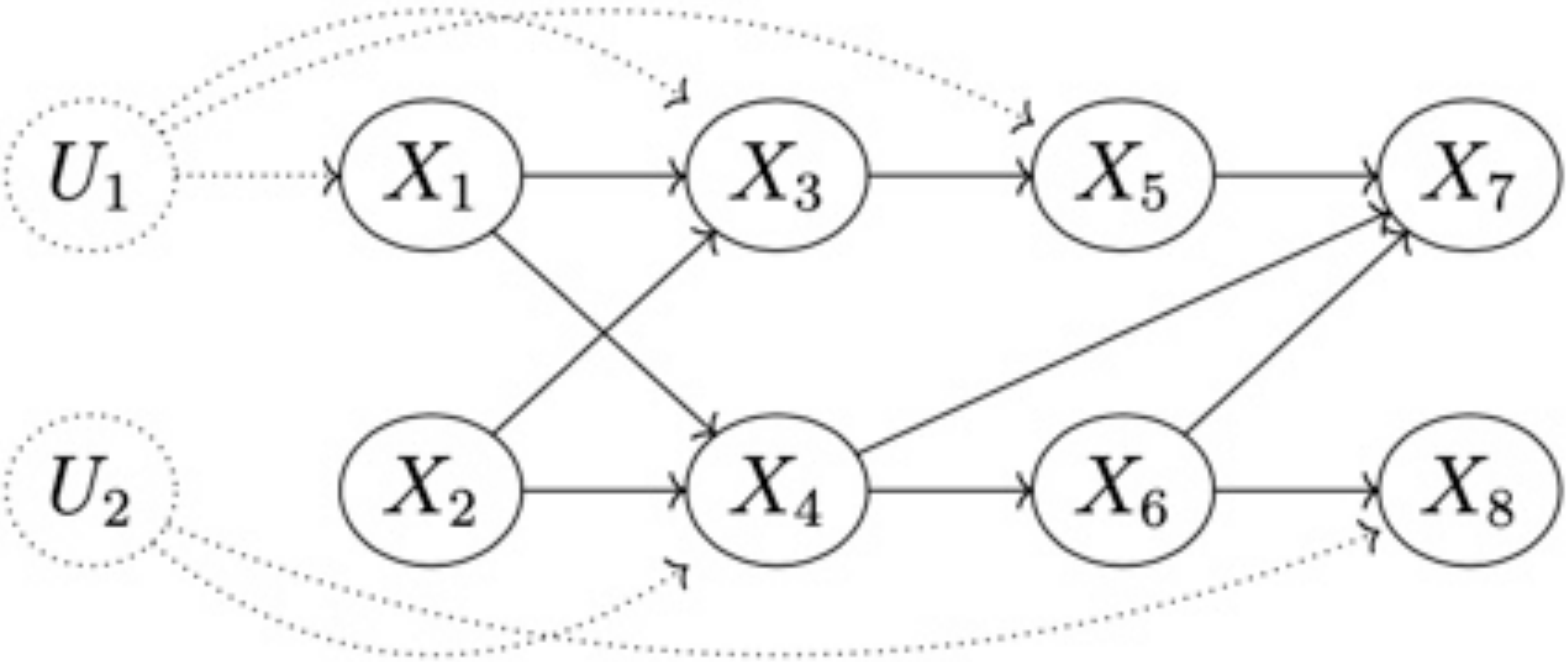
U_1, U_2 are hidden variables, X_1, …, X_8 are observed variables

Bayesian network

The assignment to $Pa(X_i)$, the parents of X_i

- A directed acyclic graph G, the n variables
- The joint probability distribution is Markovian with respect to G:

$$\Pr[X_1=x_1, X_2=x_2, \ldots, X_n=x_n] = \prod_i \Pr[X_i=x_i \mid pa(X_i)]$$

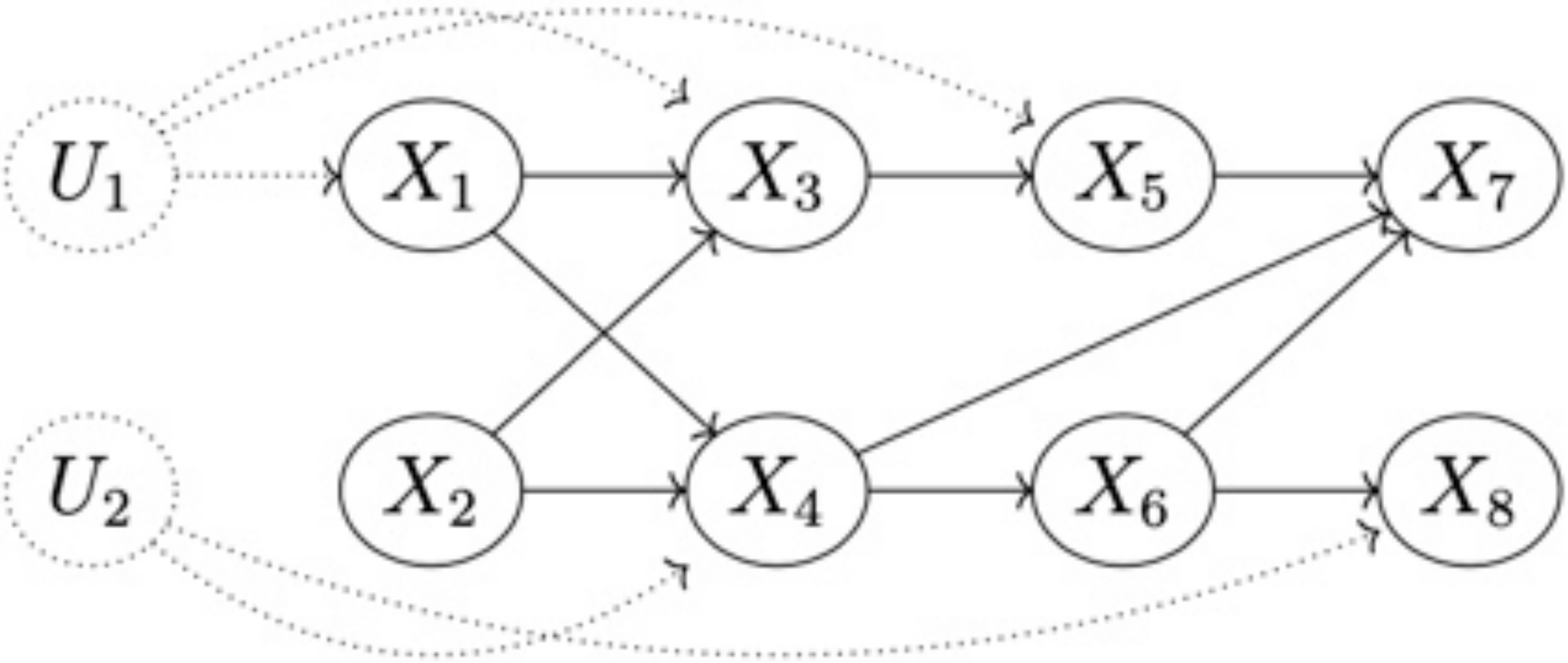


U_1, U_2 are hidden variables, X_1, …, X_8 are observed variables

Bayesian networks [Pearl 1985]

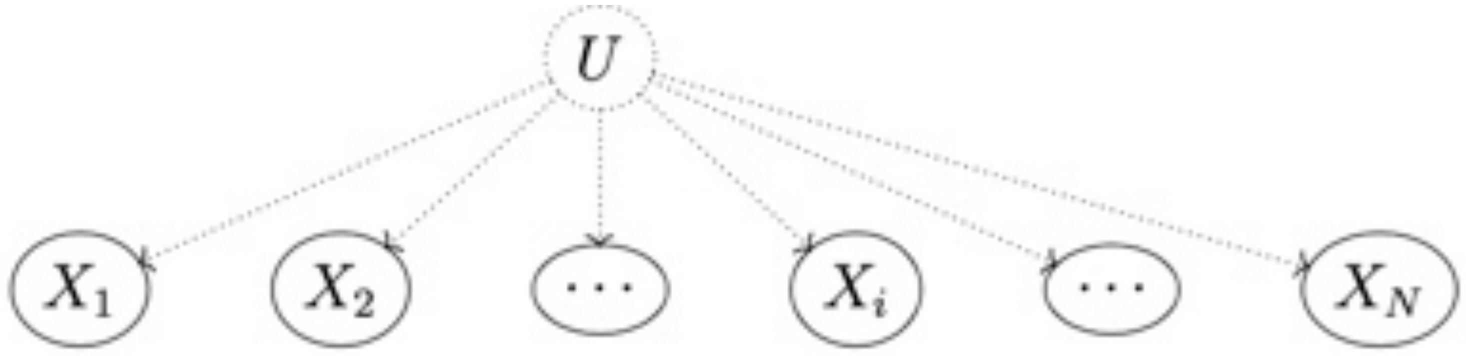
- A directed acyclic graph G, the nodes are random variables
- The joint probability distribution is Markovian with respect to G:

$$\Pr[X_1=x_1, X_2=x_2, \ldots, X_n=x_n] = \prod_i \Pr[X_i=x_i \mid pa(X_i)]$$

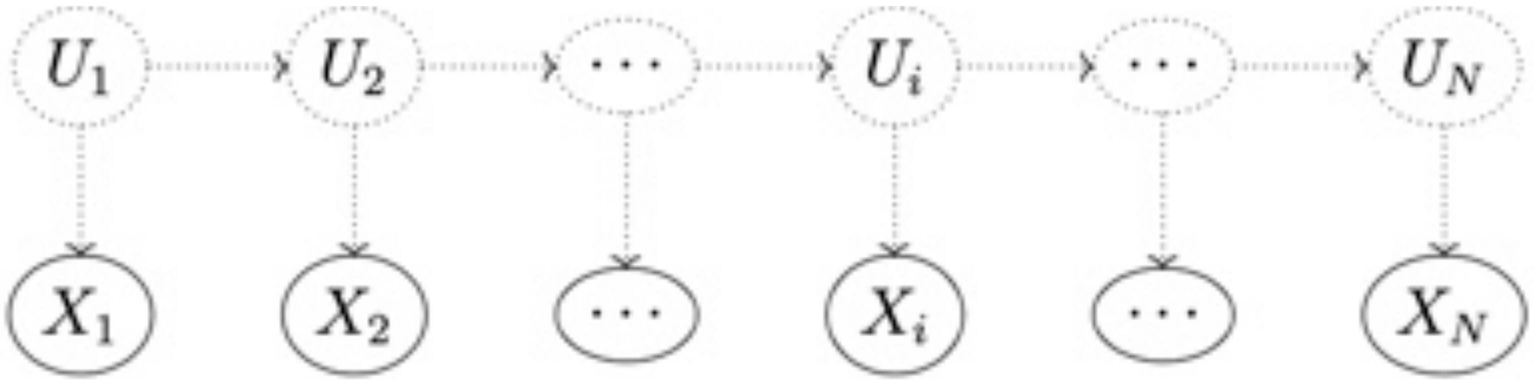


U_1, U_2 are hidden variables, X_1, …, X_8 are observed variables

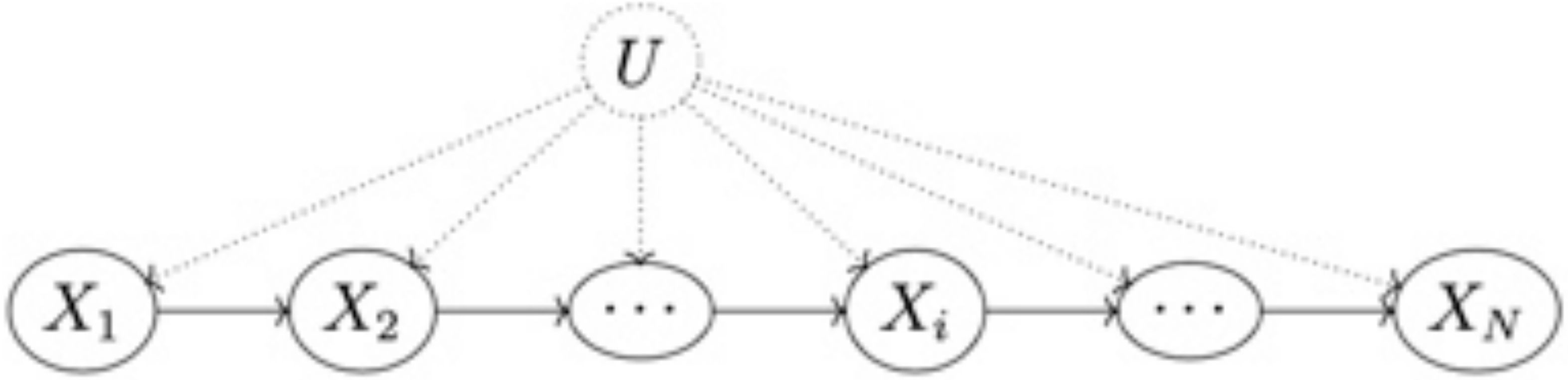
Some examples



A mixture of product distributions (MixProd)



A hidden Markov model (HMM)



A mixture of Markov models (MixMM)

The setting

- Sample space: each random variable is distributed in a finite set; let's assume observed variables are Bernoulli (i.e., in {0,1})
- Identification: computing a good estimate of the unique probabilistic model that explains the observed data
- Observations: independent samples from the joint distribution on the observed random variables
- The actual causal relations are known (or a subgraph of the known graph)

The setting

More than learning,
not always possible

- Sample space: each rand ed in a finite set; let's assume observed variables are Bernoulli (i.e., {0,1})

- Identification: computing a good estimate of the unique probabilistic model that explains the observed data

- Observations: independent samples from the joint distribution on the observed random variables

- The actual causal relations are known (or a subgraph of the known graph)

The setting

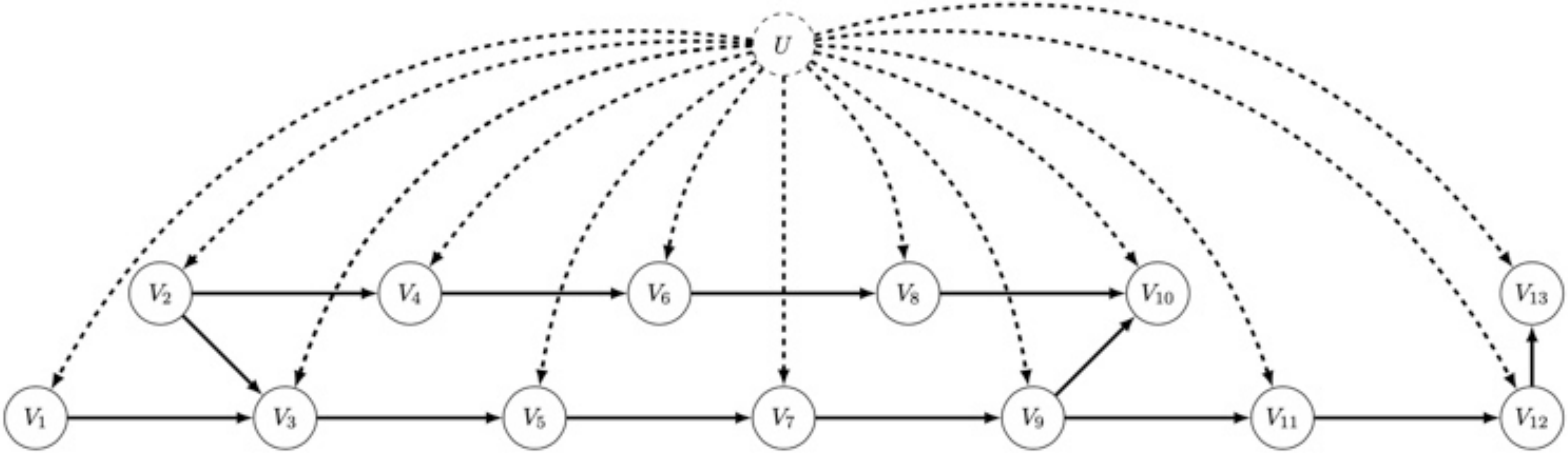
- Sample space: each random variable is distributed in a finite set; let’s assume observed variables are Bernoulli (i.e., in {0,1})
- Identification: computing a good estimate of the unique probabilistic model that explains the observed data
- Observations: independent samples from the joint distribution on the observed random variables
- The actual causal relations are known (or a subgraph of the known graph)

Mixture models

- A single confounding (hidden) variable U, affects all observed variables

- G is known, we want to identify the joint probability distribution

- Even just verifying the existence of U is impossible without assumptions



Conditions for identifiability

- Let U range in $\{1, 2, \ldots, k\}$

  $w_j \triangleq \Pr[U=j]$ $p_{ij} \triangleq \Pr[V_i=1 \mid U=j]$ $N \triangleq$ #observed random variables

- If $N=1$, all we can learn is $E[V_1=1]$. So we need G to be sufficiently large. Just V_1 has $2k-1$ degrees of freedom ($w_1, \ldots, w_{k-1}, p_{11}, \ldots, p_{1k}$).

- If two values of U produce the same distribution, we can't identify. We'll require sufficiently many ζ-separated or ζ-informative observables.

- V_i is ζ-separated iff $\min_{j \neq j'} |p_{ij} - p_{ij'}| > \zeta$

- We need at least $2k-1$ ζ-separated observed variables.

- In general, $2k-1$ 0-separated observables are necessary [RSS, TMMA].

Problems and reductions

ε = desired output accuracy, Δ = max (in+out) degree of G

- MixIID: special case of MixProd with all observables identically distributed (i.e., it's a mixture of Binomial distributions), $N \geq 2k$

  Sample size: $\varepsilon^{-2} (w_{min})^{-2} \zeta^{-O(k)}$ (for constant success probability)

  Runtime: $k^{2+o(1)} + O(k \log^2 k \operatorname{loglog} \varepsilon^{-1})$

- MixProd reduces to MixIID, $N \geq 3k-3$

  Sample size + runtime: $\varepsilon^{-2} (w_{min})^{-O(\log k)} \zeta^{-O(k \log k)} N \log N$

- MixBND (general case) reduces to MixProd, $N \geq (\Delta+1)^4 (3k-3)$

  Sample size + runtime: $\varepsilon^{-2} (w_{min})^{-O(\log k)} \zeta^{-O(k (\Delta^2 + \log k))} N \log N$

Problems and reductions

ε = desired output accuracy, Δ = max (in+out) degree of G

- MixIID: special case of MixProd with all observables identically distributed (i.e., it's a mixture of Binomial distributions), $N \geq 2k$

  Sample size: $\varepsilon^{-2} (w_{min})^{-2} \zeta^{-O(k)}$ (for constant success probability)

  Runtime: $k^{2+o(1)} + O(k \log^2 k \log\log \varepsilon^{-1})$

- MixProd reduces to MixIID, $N \geq 3k-3$

  Sample size + runtime: $\varepsilon^{-2} (w_{min})^{-O(\log k)} \zeta^{-O(k \log k)} N \log N$

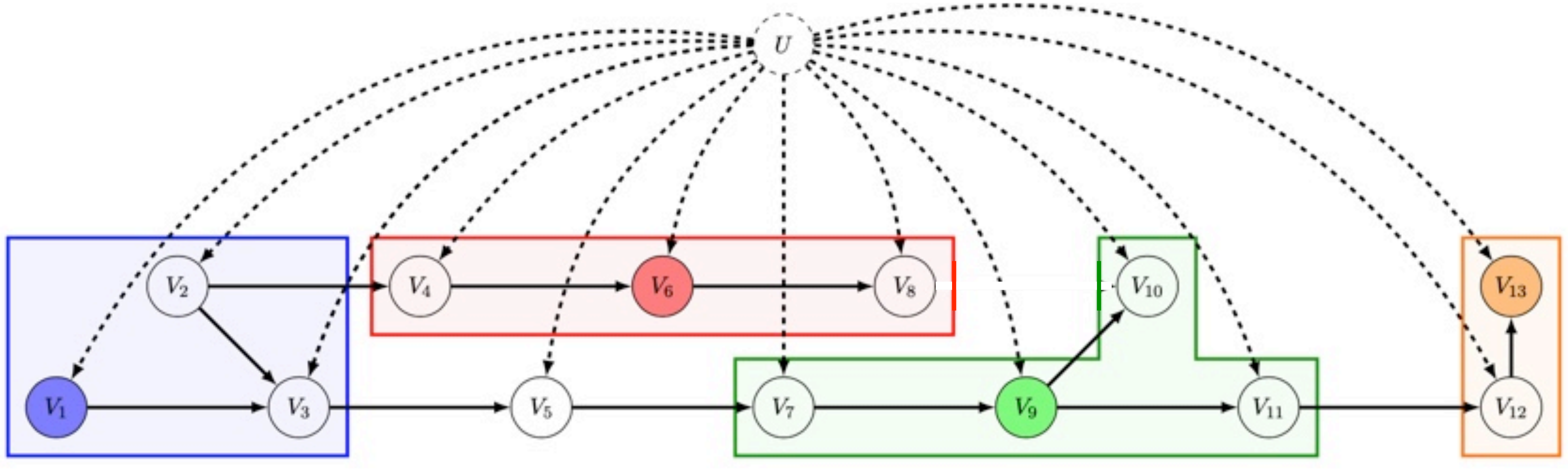
- MixBND (general case) reduces to MixProd, $N \geq (\Delta+1)^4 (3k-3)$

  Sample size + runtime: $\varepsilon^{-2} (w_{min})^{-O(\log k)} \zeta^{-O(k (\Delta^2 + \log k))} N \log N$

Reducing MixBND to MixProd

- The *Markov boundary* of V is $\mathrm{Mb}(V) = \mathrm{Pa}(V) \cup \mathrm{Ch}(V) \cup (\mathrm{Pa}(\mathrm{Ch}(V)) \setminus V)$

  We need $3k-3$ variables with mutually disjoint Markov boundaries



4 observables (V_1, V_6, V_9, V_{13}) with disjoint Markov boundaries

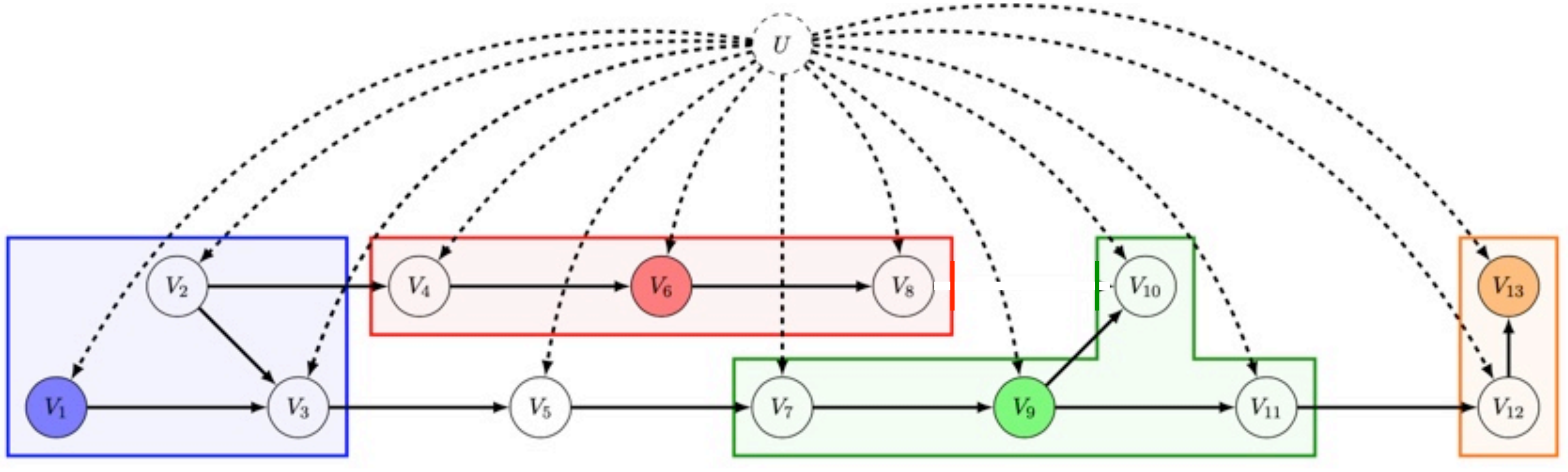
- Chosen V_is are independent conditional on U and the $\mathrm{Mb}(V_i)$s
- A *run*: assign the Mbs and identify conditionally independent variables
- We need to *align* runs (values of U can be permuted)
- Then, recover $\Pr[V \mid U \wedge \mathrm{Pa}(V)]$ for all V — *Bayesian unzipping*

Pa — parents
Ch — children

Reduci Prod

- The *Markov boundary* of V is $Mb(V) = Pa(V) \cup Ch(V) \cup (Pa(Ch(V)) \setminus V)$

We need $3k-3$ variables with mutually disjoint Markov boundaries



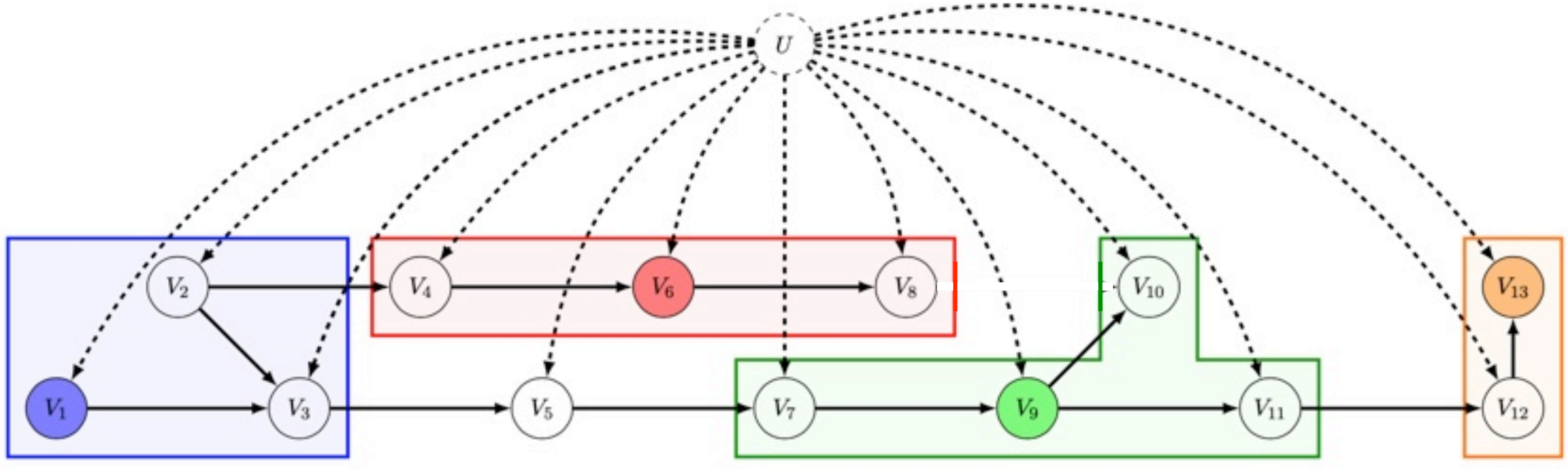
4 observables (V_1, V_6, V_9, V_{13}) with disjoint Markov boundaries

- Chosen V_is are independent conditional on U and the $Mb(V_i)$s
- A *run*: assign the Mbs and identify conditionally independent variables
- We need to *align* runs (values of U can be permuted)
- Then, recover $Pr[V \mid U \wedge Pa(V)]$ for all V — *Bayesian unzipping*

Reducing MixBND to MixProd

- The *Markov boundary* of V is $\mathrm{Mb}(V) = \mathrm{Pa}(V) \cup \mathrm{Ch}(V) \cup (\mathrm{Pa}(\mathrm{Ch}(V)) \setminus V)$

We need $3k-3$ variables with mutually disjoint Markov boundaries



4 observables (V_1, V_6, V_9, V_{13}) with disjoint Markov boundaries

- Chosen V_is are independent conditional on U and the $\mathrm{Mb}(V_i)$s
- A *run*: assign the Mbs and identify conditionally independent variables
- We need to *align* runs (values of U can be permuted)
- Then, recover $\Pr[V \mid U \wedge \mathrm{Pa}(V)]$ for all V — *Bayesian unzipping*

A good collection of runs

- Two runs are alignable iff at least one V_i has the same sequence of k distributions $\Pr[V_i \mid U=j]$ in both of them.

- We need a collection of runs with the following properties:
  - They can all be aligned together.
  - Each has $3k-3$ independent variables, conditional on the assignment of values to the Markov boundaries.
  - Every observed variable V + every assignment to $Pa(V)$ is covered by at least one run in the collection.
  - … (some additional conditions)

A good collection of runs

- Two runs are alignable iff at least one V_i has the same sequence of k distributions $Pr[V_i | U=j]$ in both of ther
- We need a collection of runs with
  - They can all be aligned together.
  - Each has $3k-3$ independent variables, co assignment of values to the Markov boundaries.
  - Every observed variable V + every assignment to $Pa(V)$ is covered by at least one run in the collection.
  - … (some additional conditions)

V is included in the independent set

A good collection of runs

- Two runs are alignable iff at least one V_i has the same sequence of k distributions $\Pr[V_i \mid U=j]$ in both of them.

- We need a collection of runs with the following properties:
    - They can all be aligned together.
    - Each has $3k-3$ independent variables, conditional on the assignment of values to the Markov boundaries.
    - Every observed variable V + every assignment to $Pa(V)$ is covered by at least one run in the collection.
    - … (some additional conditions)

Constructing a good collection of runs

- Start with V_1, V_2, …, V_{3k-3} with mutually disjoint Markov boundaries.

- Base run: arbitrary assignment to $Mb(V_1)$, …, $Mb(V_{3k-3})$
  other runs modify the base run:

- Runs for every $i=1,2,\ldots,3k-3$, and $mb \in \{0,1\}^{Mb(V_i)}$
  replace assignment to $Mb(V_i)$ by mb

- Runs for every $V \notin \{V_1,\ldots,V_{3k-3}\}$ and $pa \in \{0,1\}^{Pa(V)}$
  if $V \in Mb(V_i)$ then replace V_i by V, otherwise add V
  assign pa to $Pa(V)$
  assign any remaining variables in $Mb(V)$ arbitrarily

Bayesian unzipping

- We have: $\Pr[V \mid U \wedge \mathrm{Mb}(V)]$, for all nodes V (same permutation on U).

  We want: $\Pr[V \mid U \wedge \mathrm{Pa}(V)]$, for all nodes V.

- By definition, for an assignment mb to Mb(V),

$$\Pr[V{=}1 \mid U \wedge mb] = \frac{\Pr[V{=}1 \wedge mb \mid U]}{\Pr[V{=}1 \wedge mb \mid U] + \Pr[V{=}0 \wedge mb \mid U]} \qquad ❊$$

- Plug in (for ch, pa being the restrictions of mb to Ch(V), Pa(V))

  $\Pr[V \wedge mb \mid U] = \Pr[mb\text{-}ch \mid U]\ \Pr[V \mid U \wedge pa]\ \Pr[ch \mid U \wedge V \wedge mb\text{-}ch]$

  In (❊) the first term $\Pr[mb\text{-}ch \mid U]$ cancels

- $\Pr[ch \mid U \wedge V \wedge mb\text{-}ch]$ factors into a product over Ch(V), and can be computed inductively in reverse topological order

Final remarks

- For all V, |Mb(V)| = poly(Δ), so n = (3k-3) poly(Δ) suffices.
- In special cases (e.g., a path) we can do better.
- The case of observables over a larger domain reduces to the {0,1} case.
- The ζ-informative condition guarantees that all product distribution instances that need solving are ζ-separated.
- Compared with related literature, it’s a fairly mild condition.
- Better sample size? computation time?

Beyond final remarks

- This is a (two-step) reduction to MixIID. Lots of applications for MixIID:
- Identifying topic models reduces to MixIID [RSS, LRSS]:

  A topic is a probability distribution on the dictionary {1, 2, …, n}.

  To produce a document, draw a topic in {1, 2, …, k}, then draw words.

  Documents with 2k-1 words suffice.
- Inferring (haploid) population histories (evolving according to Wright-Fisher dynamics) [KKMMR] is equivalent to MixIID:

  Reduces to hyper-exponential mixture problem (Kingman coalescent); same as MixIID (linear transformation of the moments polynomials).
- Network evaluation, …

Beyond final rema

Instead of {0, 1} in MixIID

- This is a (two-step) reducti pplications for MixIID:
- Identifying topic models reduces to MixIID [RSS, RSS]:

  A topic is a probability distribution on the dictionary {1, 2, …, n}.

  To produce a document, draw a topic in {1, 2, …, k}, then draw words.

  Documents with 2k-1 words suffice.
- Inferring (haploid) population histories (evolving according to Wright-Fisher dynamics) [KKMMR] is equivalent to MixIID:

  Reduces to hyper-exponential mixture problem (Kingman coalescent); same as MixIID (linear transformation of the moments polynomials).
- Network evaluation, …

Beyond final remarks

- This is a (two-step) reduction to MixIID. Lots of applications for MixIID:
- Identifying topic models reduces to MixIID [RSS, LRSS]:

  A topic is a probability distribution on the dictionary {1, 2, …, n}.

  To produce a document, draw a topic in {1, 2, …, k}, then draw words.

  Documents with 2k-1 words suffice.
- Inferring (haploid) population histories (evolving according to Wright-Fisher dynamics) [KKMMR] is equivalent to MixIID:

  Reduces to hyper-exponential mixture problem (Kingman coalescent); same as MixIID (linear transformation of the moments polynomials).
- Network evaluation, …