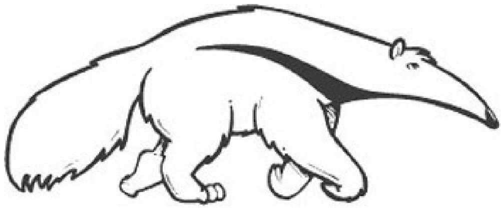
Algorithms for Causal Probabilistic Graphical Models

Class 2:
Decomposition & Variational Methods

Athens Summer School on AI
July 2024



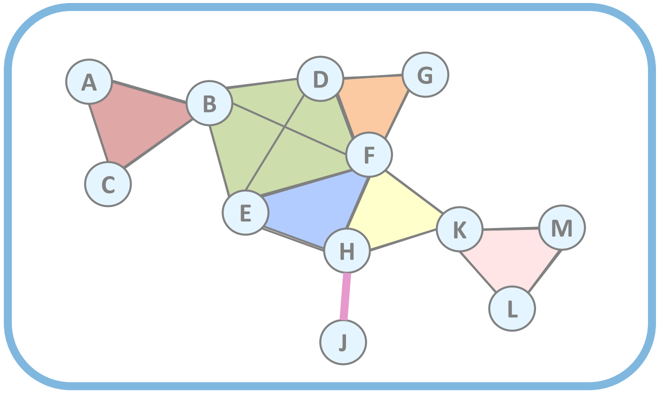
Prof. Rina Dechter
Prof. Alexander Ihler

BREN:ICS
INFORMATION AND COMPUTER SCIENCES

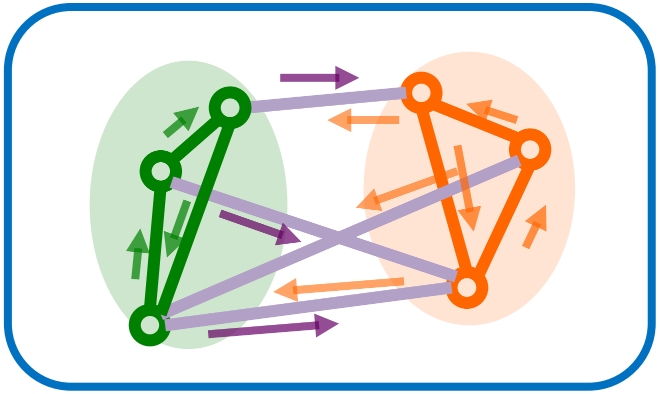
UNIVERSITY of CALIFORNIA IRVINE

Outline of Lectures

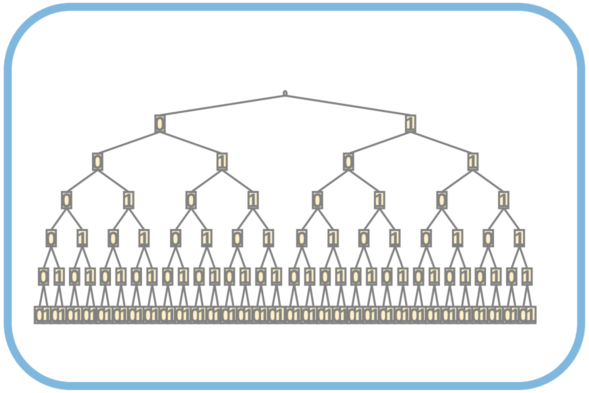
Class 1: Introduction & Inference



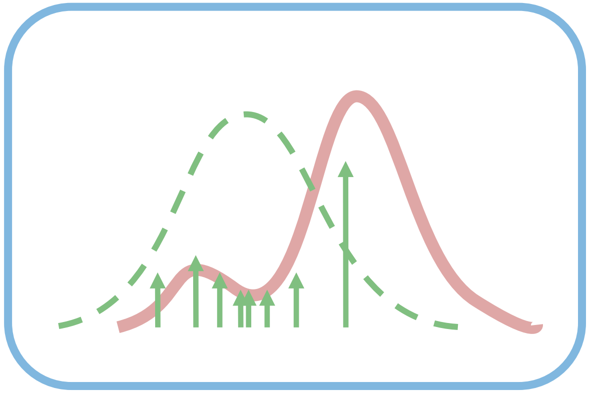
Class 2: Bounds & Variational Methods



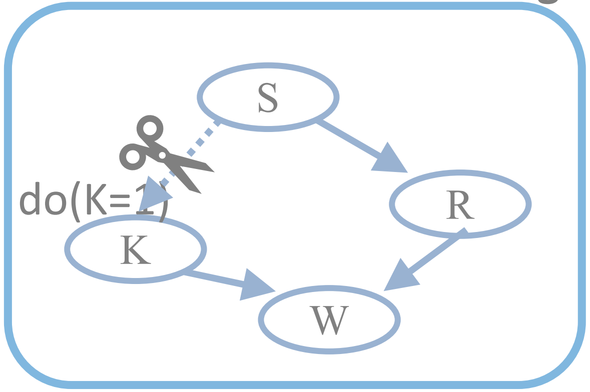
Class 3: Search Methods



Class 4: Monte Carlo Methods



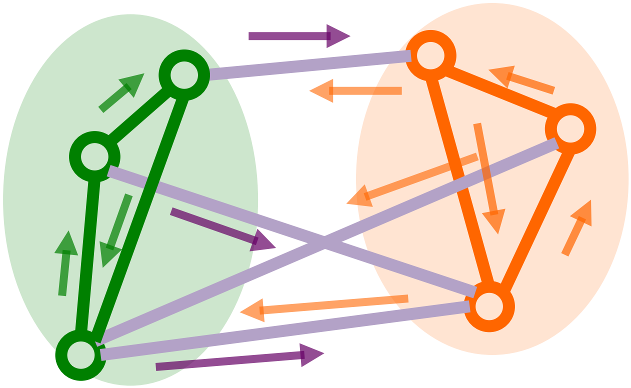
Class 5: Causal Reasoning



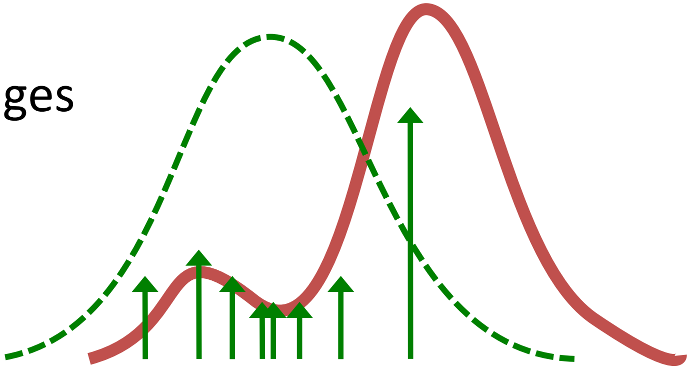
Approximate Inference

- Two main schools of approximate inference

- **Variational methods** [Class 2]
  - Frame "inference" as convex optimization & approximate (constraints, objectives)
  - Reason about "beliefs"; pass messages
  - Fast approximations & bounds
  - Quality often limited by memory



- **Monte Carlo sampling** [Class 4]
  - Approximate expectations with sample averages
  - Estimates are asymptotically correct
  - Can be hard to gauge finite sample quality



Outline

- Review: Graphical Models
- Decomposition Bounds
- Variational Optimization
- Convexity & Duality
- Regions & Higher-order Approximations

Graphical models

$A \in \{0,1\}$

$B \in \{0,1\}$

$C \in \{0,1\}$

$f_{AB}(A,B), \quad f_{BC}(B,C)$

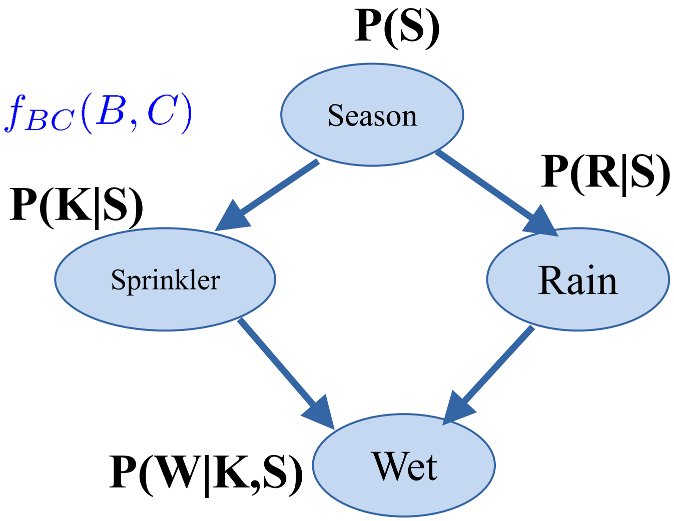
A *graphical model* consists of:

$X = \{X_1, \dots, X_n\}$ -- variables

$D = \{D_1, \dots, D_n\}$ -- domains (we'll assume discrete)

$F = \{f_{\alpha_1}, \dots, f_{\alpha_m}\}$-- functions or "factors"

and a *combination operator*



The *combination operator* defines an overall function from the individual factors,

e.g., "*" : $P(S,K,R,W) = P(S) \cdot P(K|S) \cdot P(R|S) \cdot P(W|K,S)$

Notation:

Discrete Xi values called "states"

"Tuple" or "configuration": states taken by a set of variables

"Scope" of f: set of variables that are arguments to a factor f

often index factors by their scope, e.g., $f_\alpha(X_\alpha), \quad X_\alpha \subseteq X$

Canonical forms

A *graphical model* consists of:

$X = \{X_1, \ldots, X_n\}$ -- variables

$D = \{D_1, \ldots, D_n\}$ -- domains

$F = \{f_{\alpha_1}, \ldots, f_{\alpha_m}\}$-- functions or “factors”

and a *combination operator*

Typically either multiplication or summation; mostly equivalent:

$$f_\alpha(X_\alpha) \geq 0$$

$$F(X) = \prod_\alpha f_\alpha(X_\alpha)$$

Product of nonnegative factors (probabilities, 0/1, etc.)

log / exp

$$\theta_\alpha(X_\alpha) = \log f_\alpha(X_\alpha) \in \mathbb{R}$$

$$\theta(X) = \log F(x) = \sum_\alpha \theta_\alpha(X_\alpha)$$

Sum of factors (costs, utilities, etc.)

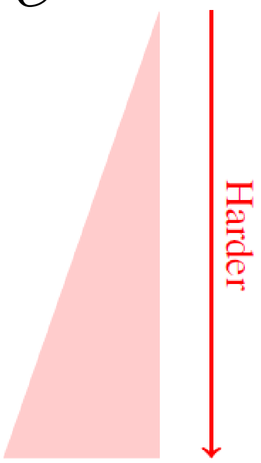
Probabilistic Reasoning Problems

- Exact Inference by elimination or search
- Complexity:

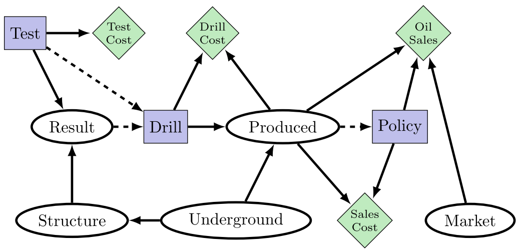
Causal effects

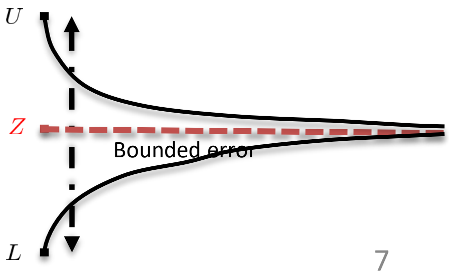
Max-Inference:	$f(x^*) = \max_x \prod_\alpha f_\alpha(x_\alpha)$
Sum-Inference:	$Z = \sum_x \prod_\alpha f_\alpha(x_\alpha)$
Mixed-Inference (MMAP):	$f_M(x_M^*) = \max_{x_M} \sum_{x_S} \prod_\alpha f_\alpha(x_\alpha)$
Mixed-Inference (MEU):	$\text{MEU} = \max_{D_1,\ldots,D_m} \sum_{X_1,\ldots X_n} (\prod_{P_i \in P} P_i) \times (\sum_{r_i \in R} r_i)$

$e^{\text{tree-width}}$



Influence diagrams & planning





Outline

Review: Graphical Models

Decomposition Bounds

Variational Optimization

Convexity & Duality

Regions & Higher-order Approximations

Decomposition bounds

- Upper & lower bounds via approximate problem decomposition
- Example: MAP inference $F(x) = f_1(x) + f_2(x)$

X	F (X)
0	2.0
1	4.0
2	5.0
3	4.0

=

X	f_1(X)
0	1.0
1	2.0
2	3.0
3	4.0

+

X	f_2(X)
0	1.0
1	2.0
2	2.0
3	0.0

$$\max_x F(x) \quad = \quad \max_x \Big[f_1(x) + f_2(x) \Big]$$

$$5.0 \quad \leq \quad \Big[\max_x f_1(x) \;+\; \max_x f_2(x) \Big] \quad = \; 4.0 \; + \; 2.0 \; = \; 6.0$$

- Relaxation: two “copies” of x, no longer required to be equal
- Bound is tight (equality) if f_1, f_2 agree on maximizing value x

Mini-Bucket Approximation

Split a bucket into mini-buckets —> bound complexity

bucket (X) =

$$\{\ \underbrace{f_1,\ f_2,\ \dots\ f_r,\ f_{r+1},\ \dots\ f_n}\ \}$$

$$\lambda_X(\cdot) = \max_x \prod_{i=1}^{n} f_i(x,\dots)$$

$$\{\ f_1,\ \dots\ f_r\ \} \qquad\qquad \{\ f_{r+1},\ \dots\ f_n\ \}$$

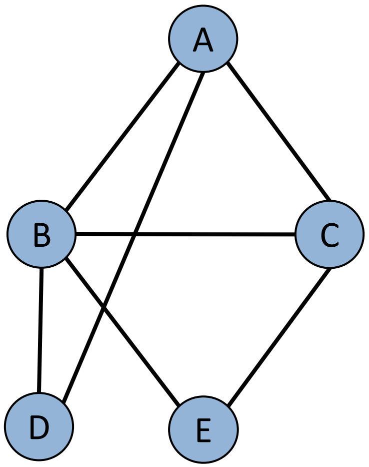
$$\lambda_{X,1}(\cdot) = \max_x \prod_{i=1}^{r} f_i(x,\dots) \qquad\qquad \lambda_{X,2}(\cdot) = \max_x \prod_{i=r+1}^{n} f_i(x,\dots)$$

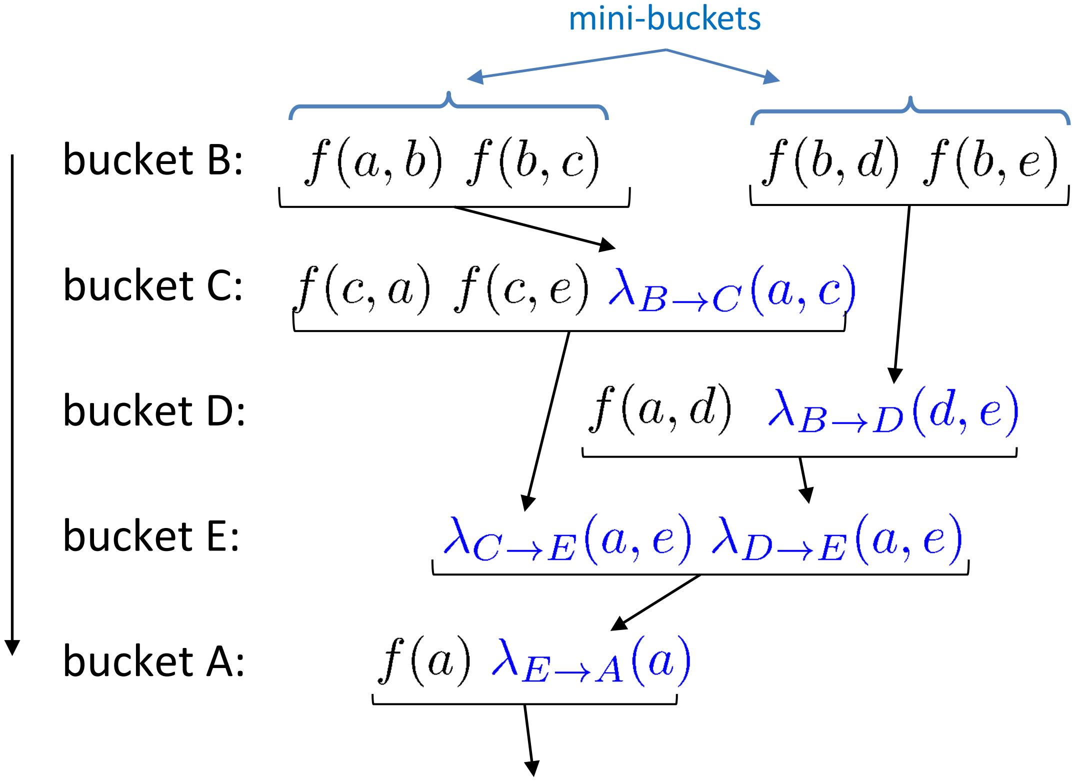
$$\lambda_X(\cdot)\ \le\ \lambda_{X,1}(\cdot)\ \lambda_{X,2}(\cdot)$$

Exponential complexity decrease: $O(e^n) \longrightarrow O(e^r) + O(e^{n-r})$

Mini-Bucket Elimination

[Dechter & Rish 2003]




mini-buckets

bucket B: $\underbrace{f(a,b)\ f(b,c)}$ $\underbrace{f(b,d)\ f(b,e)}$

bucket C: $\underbrace{f(c,a)\ f(c,e)\ \lambda_{B\to C}(a,c)}$

bucket D: $\underbrace{f(a,d)\ \ \lambda_{B\to D}(d,e)}$

bucket E: $\underbrace{\lambda_{C\to E}(a,e)\ \lambda_{D\to E}(a,e)}$

bucket A: $\underbrace{f(a)\ \lambda_{E\to A}(a)}$

U = upper bound

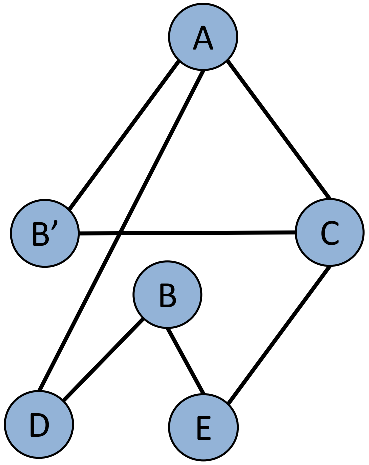
$$\lambda_{B\to C}(a,c) = \max_b f(a,b)\ f(b,c)$$

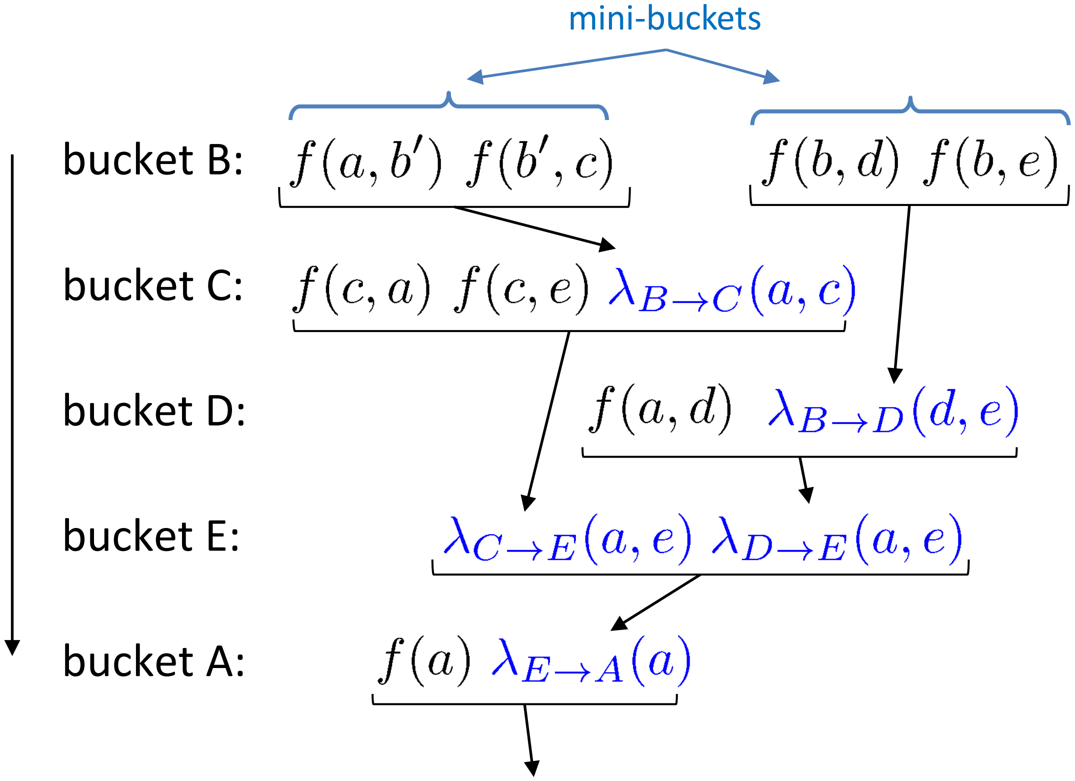
$$\lambda_{B\to D}(d,e) = \max_b f(b,d)\ f(b,e)$$

$$\lambda_{C\to E}(a,e) = \max_c \ldots$$

Mini-Bucket Elimination

[Dechter & Rish 2003]




mini-buckets

bucket B: $f(a,b')\ f(b',c)$ $f(b,d)\ f(b,e)$

bucket C: $f(c,a)\ f(c,e)\ \lambda_{B\to C}(a,c)$

bucket D: $f(a,d)\ \ \lambda_{B\to D}(d,e)$

bucket E: $\lambda_{C\to E}(a,e)\ \lambda_{D\to E}(a,e)$

bucket A: $f(a)\ \lambda_{E\to A}(a)$

U = upper bound

$$\lambda_{B\to C}(a,c) = \max_b f(a,b)\ f(b,c)$$

$$\lambda_{B\to D}(d,e) = \max_b f(b,d)\ f(b,e)$$

$$\lambda_{C\to E}(a,e) = \max_c \ldots$$

Can interpret process as “duplicating” B

[Kask et al. 2001, Geffner et al. 2007, Choi et al. 2007, Johnson et al. 2007]

Mini-Bucket Decoding

- Assign values in reverse order using approximate messages

$$\mathbf{b}^* = \arg\max_b \, f(a^*, b) \cdot f(b, c^*) \cdot f(b, d^*) \cdot f(b, e^*)$$

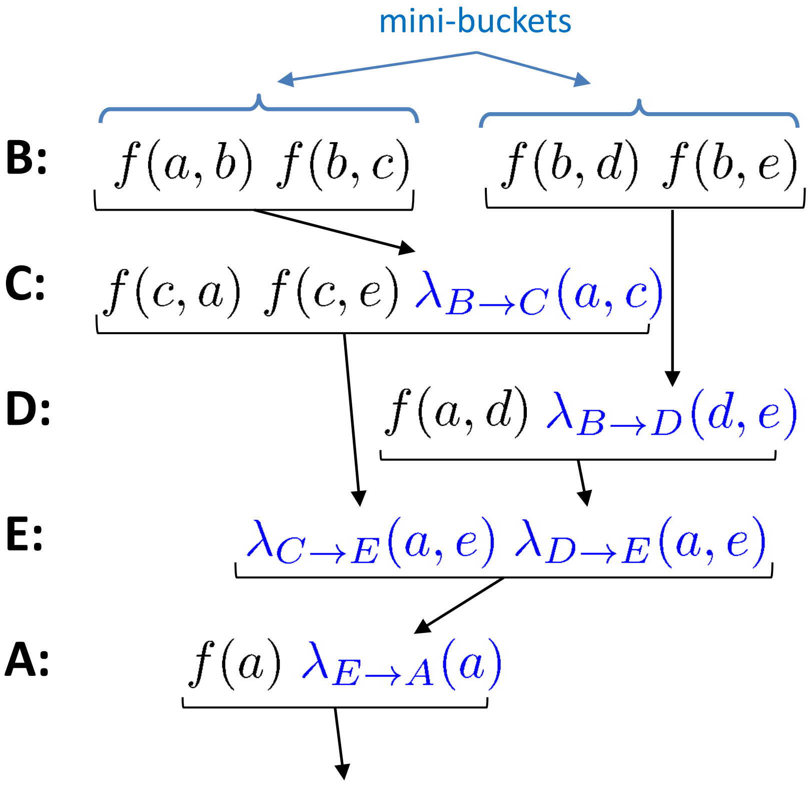
$$\mathbf{c}^* = \arg\max_c \, f(c, a^*) \cdot f(c, e^*) \cdot \lambda_{B\to C}(a^*, c)$$

$$\mathbf{d}^* = \arg\max_d \, f(a^*, d) \cdot \lambda_{B\to D}(d, e^*)$$

$$\mathbf{e}^* = \arg\max_e \, \lambda_{C\to E}(a^*, e) \cdot \lambda_{D\to E}(a^*, e)$$

$$\mathbf{a}^* = \arg\max_a \, f(a) \cdot \lambda_{E\to A}(a)$$

Greedy configuration = lower bound



mini-buckets

B: $f(a,b)\ f(b,c)$ $f(b,d)\ f(b,e)$

C: $f(c,a)\ f(c,e)\ \lambda_{B\to C}(a,c)$

D: $f(a,d)\ \lambda_{B\to D}(d,e)$

E: $\lambda_{C\to E}(a,e)\ \lambda_{D\to E}(a,e)$

A: $f(a)\ \lambda_{E\to A}(a)$

U = upper bound

Properties of MBE(i)

- **Complexity**: O(r exp(i)) time and O(exp(i)) space
- Yields a lower bound and an upper bound
- **Accuracy**: determined by upper/lower (U/L) bound
- Possible use of mini-bucket approximations
  - As anytime algorithms
  - As heuristics in search
- Other tasks (similar mini-bucket approximations)
  - Belief updating, Marginal MAP, MEU, WCSP, Max-CSP

[Dechter and Rish, 1997], [Liu and Ihler, 2011], [Liu and Ihler, 2013]

Tightening the bound

- Reparameterization (or, “cost shifting”)
  - Decrease bound without changing overall function

A	B	f_1(A,B)
0	0	2.0
1	0	3.5
0	1	1.0
1	1	3.0

+

B	C	f_2(B,C)
0	0	1.0
0	1	0.0
1	0	1.0
1	1	3.0

$$\max_{a,b} f_1(a,b) \quad + \quad \max_{b,c} f_2(b,c)$$

$$+\lambda_{B\to AB}(b) \qquad + \lambda_{B\to BC}(b)$$

=

$$f_{AB}(a,b) + f_{BC}(b,c)$$

A	B	C	F(A,B,C)
0	0	0	3.0
0	0	1	2.0
0	1	0	2.0
0	1	1	4.0
1	0	0	4.5
1	0	1	3.5
1	1	0	4.0
1	1	1	6.0

$$\lambda_{B\to AB}(b) + \lambda_{B\to BC}(b) = 0$$

=

A	B	f_1(A,B)	¸(B)
0	0	2.0	0
1	0	3.5	
0	1	1.0	+1
1	1	3.0	

+

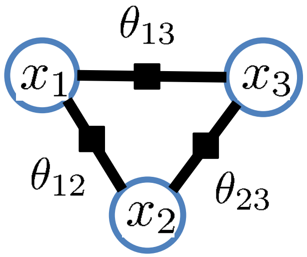
B	C	f_2(B,C)	-¸(B)
0	0	1.0	0
0	1	0.0	
1	0	1.0	-1
1	1	3.0	

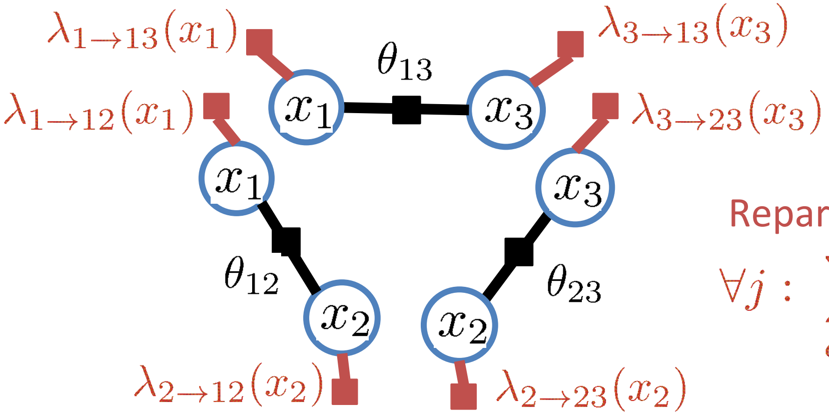
(Adjusting functions
cancel each other)

(Decomposition bound is exact)

Decomposition for MAP

Add factors that "adjust" each local term, but cancel out in total





Reparameterization:

$$\forall j : \sum_{\alpha \ni j} \lambda_{j\to\alpha}(x_j) = 0$$

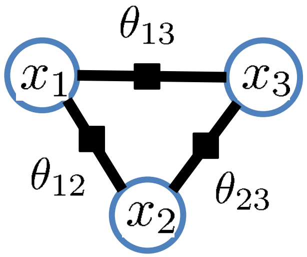
$$\log f(\mathbf{x}^*) = \max_{\mathbf{x}} \sum_{\alpha} \theta_\alpha(\mathbf{x}_\alpha) \quad \leq \min_{\{\lambda_{i\to\alpha}\}} \sum_{\alpha} \max_{\mathbf{x}_\alpha} \left[\theta_\alpha(\mathbf{x}_\alpha) + \sum_{i\in\alpha} \lambda_{i\to\alpha}(x_i) \right]$$

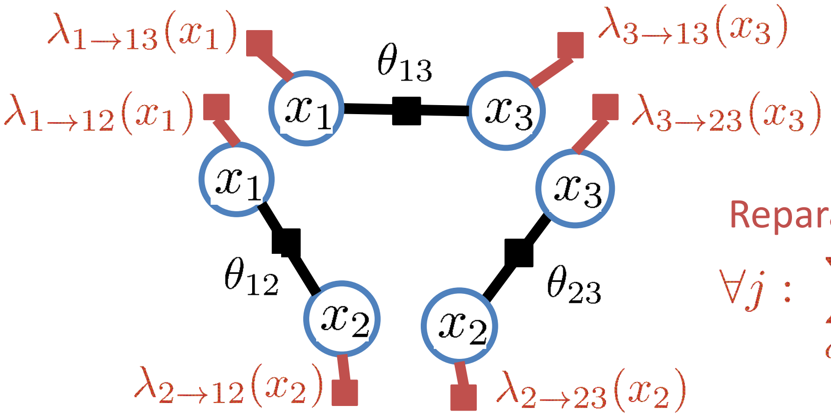
- Bound solution using decomposed optimization
- Solve independently: optimistic bound

- Tighten the bound by reparameterization
  - Enforces lost equality constraints using Lagrange multipliers

Decomposition for MAP

Add factors that "adjust" each local term, but cancel out in total





Reparameterization:

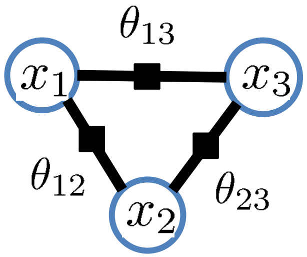
$$\forall j: \sum_{\alpha \ni j} \lambda_{j\to\alpha}(x_j) = 0$$

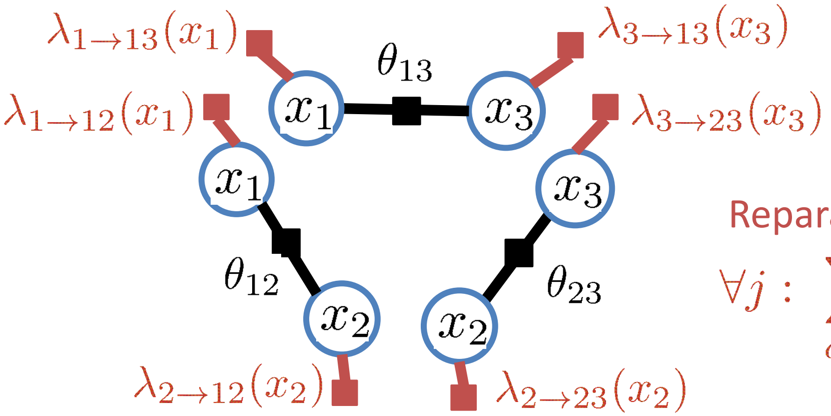
$$\log f(\mathbf{x}^*) = \max_{\mathbf{x}} \sum_{\alpha} \theta_\alpha(\mathbf{x}_\alpha) \quad \le \min_{\{\lambda_{i\to\alpha}\}} \sum_{\alpha} \max_{\mathbf{x}_\alpha} \left[\theta_\alpha(\mathbf{x}_\alpha) + \sum_{i\in\alpha} \lambda_{i\to\alpha}(x_i) \right]$$

- Many names for the same class of bounds
  - Dual decomposition [Komodakis et al. 2007]
  - TRW, MPLP [Wainwright et al. 2005; Globerson & Jaakkola 2007]
  - Soft arc consistency [Cooper & Schiex 2004]
  - Max-sum diffusion [Warner 2007]

Decomposition for MAP

Add factors that "adjust" each local term, but cancel out in total





Reparameterization:

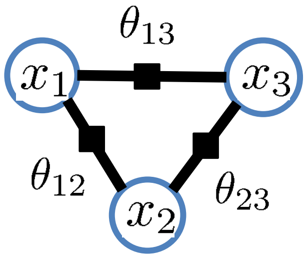
$$\forall j : \sum_{\alpha \ni j} \lambda_{j\to\alpha}(x_j) = 0$$

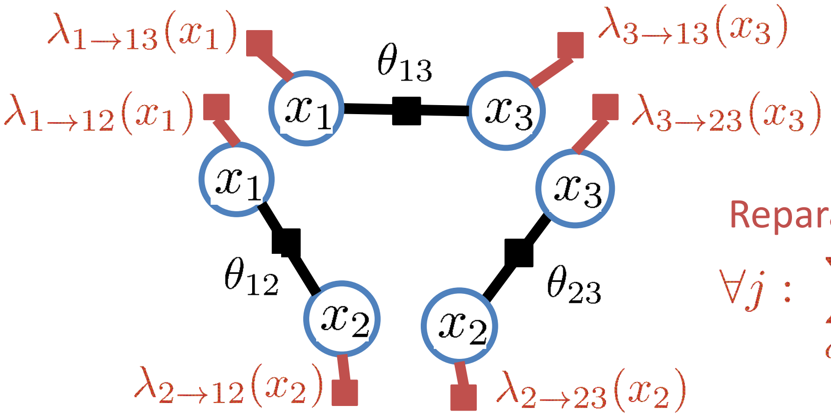
$$\log f(\mathbf{x}^*) = \max_{\mathbf{x}} \sum_{\alpha} \theta_\alpha(\mathbf{x}_\alpha) \quad \le \min_{\{\lambda_{i\to\alpha}\}} \sum_{\alpha} \max_{\mathbf{x}_\alpha} \left[\theta_\alpha(\mathbf{x}_\alpha) + \sum_{i\in\alpha} \lambda_{i\to\alpha}(x_i) \right]$$

- Many ways to optimize the bound:
  - Sub-gradient descent [Komodakis et al. 2007; Jojic et al. 2010]
  - Coordinate descent [Warner 2007; Globerson & Jaakkola 2007; Sontag 2009; Ihler et al. 2012]
  - Proximal optimization [Ravikumar et al. 2010]
  - ADMM [Meshi & Globerson 2011; Martins et al. 2011; Forouzan & Ihler 2013]

Decomposition for MAP

Add factors that “adjust” each local term, but cancel out in total



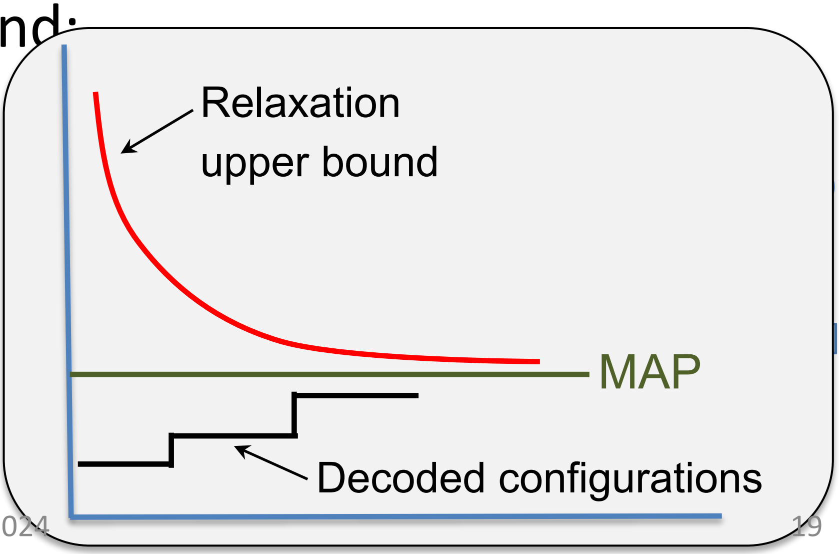


Reparameterization:

$$\forall j : \sum_{\alpha \ni j} \lambda_{j\to\alpha}(x_j) = 0$$

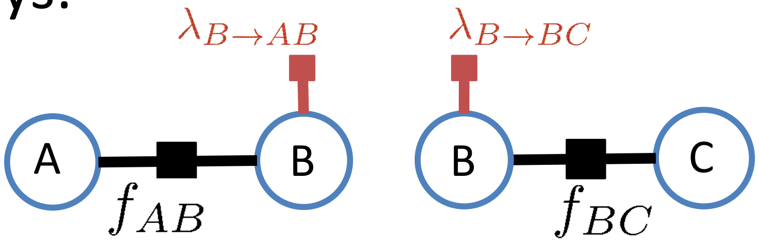
$$\log f(\mathbf{x}^*) = \max_{\mathbf{x}} \sum_{\alpha} \theta_\alpha(\mathbf{x}_\alpha) \quad \le \min_{\{\lambda_{i\to\alpha}\}} \sum_{\alpha} \max_{\mathbf{x}_\alpha} \Big[\theta_\alpha(\mathbf{x}_\alpha) + \sum_{i\in\alpha} \lambda_{i\to\alpha}(x_i) \Big]$$

- Many ways to optimize the bound:
  - Sub-gradient descent [Komodakis et
  - Coordinate descent [Warner 2007; 12]
  - Proximal optimization [Ravikumar et
  - ADMM [Meshi & Glob



Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent



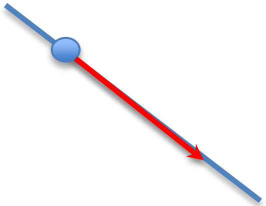
=

A	B	f₁(A,B)	λ(B)
0	0	1.0	0
1	0	0.0	
0	1	0.0	0
1	1	2.5	
0	2	1.0	0
1	2	3.0	

+

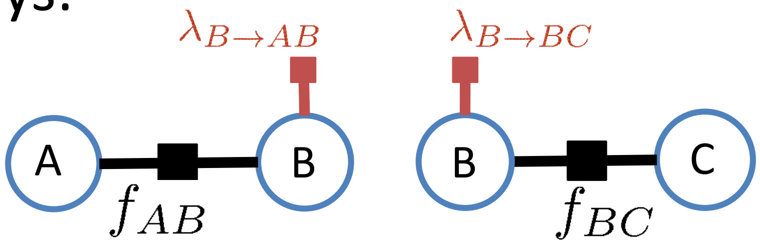
B	C	f₂(B,C)	-λ(B)
0	0	5.0	0
0	1	2.0	
1	0	1.0	0
1	1	1.5	
2	0	0.2	0
2	1	0.0	

$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$



Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent



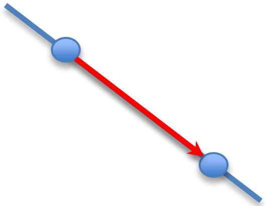
=

A	B	f$_1$(A,B)	λ(B)
0	0	1.0	+1
1	0	0.0	
0	1	0.0	0
1	1	2.5	
0	2	1.0	-1
1	2	3.0	

+

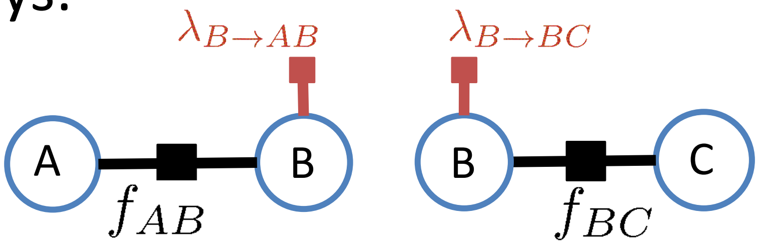
B	C	f$_2$(B,C)	-λ(B)
0	0	5.0	-1
0	1	2.0	
1	0	1.0	0
1	1	1.5	
2	0	0.2	+1
2	1	0.0	

$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$



Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent



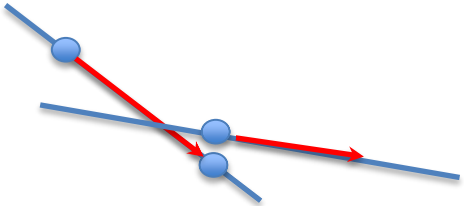
=

A	B	f_1(A,B)	λ(B)
0	0	1.0	+1
1	0	0.0	
0	1	0.0	0
1	1	2.5	
0	2	1.0	-1
1	2	3.0	

+

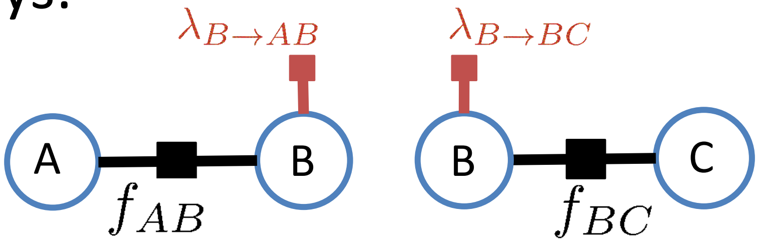
B	C	f_2(B,C)	-λ(B)
0	0	5.0	-1
0	1	2.0	
1	0	1.0	0
1	1	1.5	
2	0	0.2	+1
2	1	0.0	

$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$



Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent



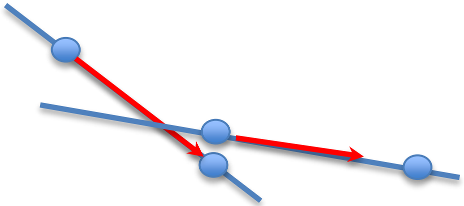
=

A	B	$f_1(A,B)$	$\lambda(B)$
0	0	1.0	+2
1	0	0.0	
0	1	0.0	-1
1	1	2.5	
0	2	1.0	-1
1	2	3.0	

+

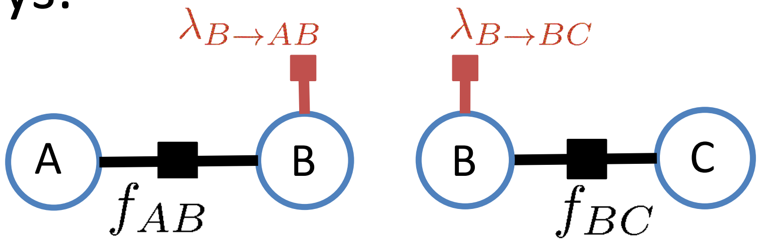
B	C	$f_2(B,C)$	$-\lambda(B)$
0	0	5.0	-2
0	1	2.0	
1	0	1.0	+1
1	1	1.5	
2	0	0.2	+1
2	1	0.0	

$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$



Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent



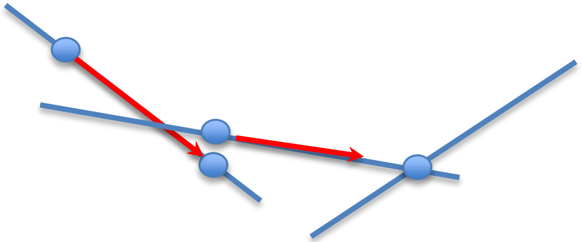
Both parts agree on the optima value(s): zero subgradient

=

A	B	f₁(A,B)	λ(B)
0	0	1.0	+2
1	0	0.0	
0	1	0.0	-1
1	1	2.5	
0	2	1.0	-1
1	2	3.0	

+

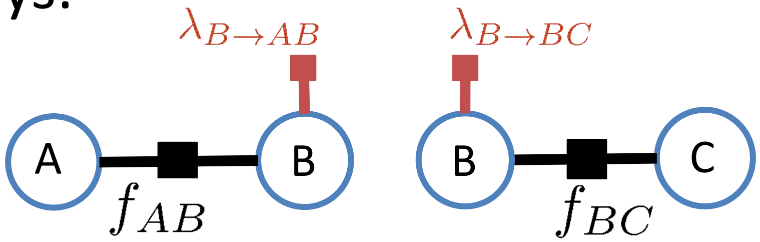
B	C	f₂(B,C)	-λ(B)
0	0	5.0	-2
0	1	2.0	
1	0	1.0	+1
1	1	1.5	
2	0	0.2	+1
2	1	0.0	



$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$

Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent
  - Coordinate descent



Easy to minimize over a single variable, e.g. B:

Find maxima for each B
Match values between f's

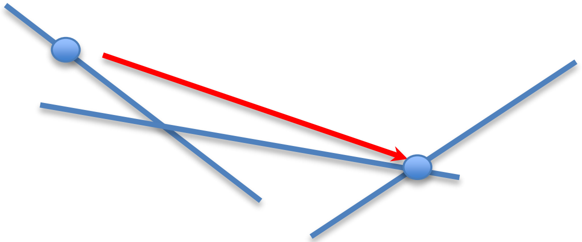
=

A	B	f$_1$(A,B)	λ(B)
0	0	1.0	
1	0	0.0	
0	1	0.0	
1	1	2.5	
0	2	1.0	
1	2	3.0	

+

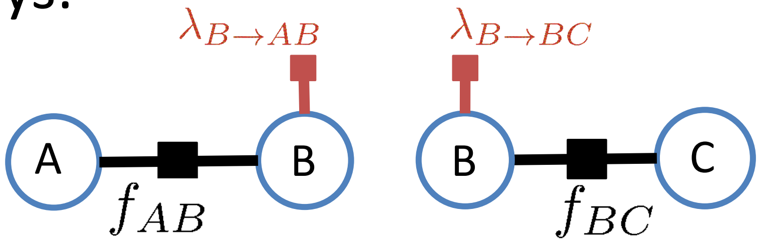
B	C	f$_2$(B,C)	-λ(B)
0	0	5.0	
0	1	2.0	
1	0	1.0	
1	1	1.5	
2	0	0.2	
2	1	0.0	

$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$



Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent
  - Coordinate descent



Easy to minimize over a single variable, e.g. B:

Find maxima for each B
Match values between f's

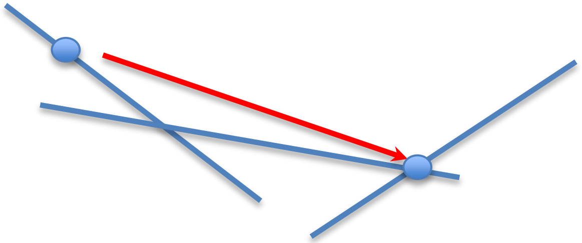
=

A	B	f_1(A,B)	λ(B)
0	0	1.0	- 0.5 +2.5
1	0	0.0	
0	1	0.0	- 1.25 +0.75
1	1	2.5	
0	2	1.0	- 1.5 +0.1
1	2	3.0	

+

B	C	f_2(B,C)	-λ(B)
0	0	5.0	+0.5 - 2.5
0	1	2.0	
1	0	1.0	+1.25 - 0.75
1	1	1.5	
2	0	0.2	+1.5 - 0.1
2	1	0.0	

$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$



Mini-Bucket as Decomposition

[Ihler et al. 2012]

$$\max_{a,c,b} \log \Big[f(a,b) \cdot f(b,c) / \lambda_{B\to C}(a,c) \Big] = 0$$

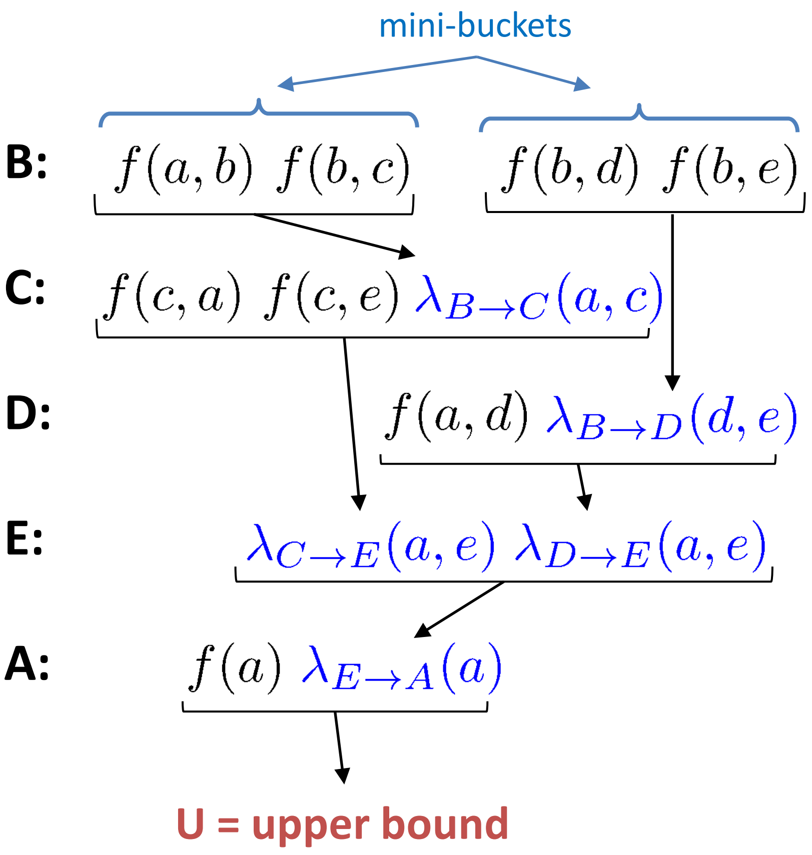
$$\max_{b,d,e} \log \Big[f(b,d) \cdot f(b,e) / \lambda_{B\to D}(d,e) \Big] = 0$$

$$\max_{a,e,c} \log \Big[f(c,a)\, f(c,e)\, \lambda_{B\to C} / \lambda_{C\to E} \Big] = 0$$

$$\max_{a,d,e} \log \Big[f(a,d)\, \lambda_{B\to D} / \lambda_{D\to E} \Big] = 0$$

$$\max_{a,d} \log \Big[\lambda_{C\to E}\, \lambda_{D\to E} / \lambda_{E\to A} \Big] = 0$$

$$\max_{a} \log \Big[f(a)\, \lambda_{E\to A}(a) \Big] = \log U$$



mini-buckets

B: $f(a,b)\ f(b,c)$ $\quad$ $f(b,d)\ f(b,e)$

C: $f(c,a)\ f(c,e)\ \lambda_{B\to C}(a,c)$

D: $f(a,d)\ \lambda_{B\to D}(d,e)$

E: $\lambda_{C\to E}(a,e)\ \lambda_{D\to E}(a,e)$

A: $f(a)\ \lambda_{E\to A}(a)$

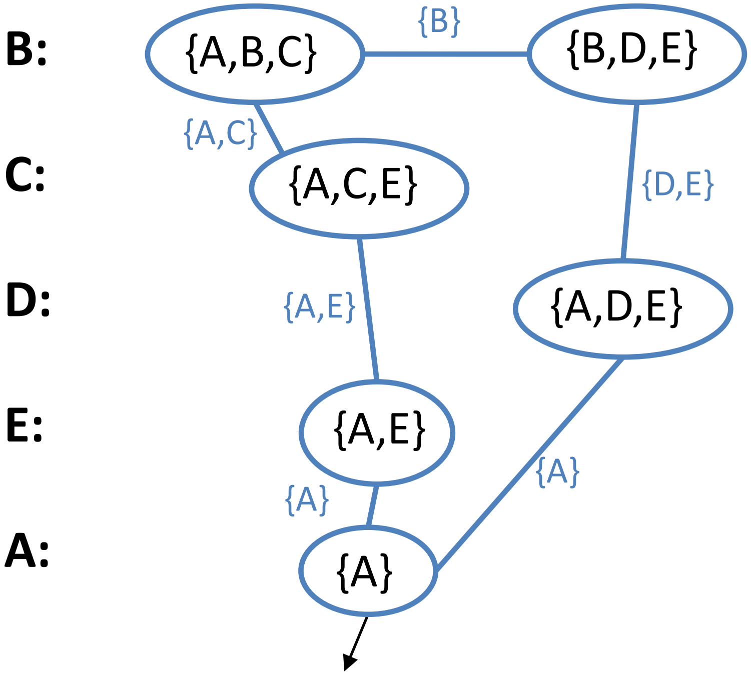
U = upper bound

Mini-Bucket as Decomposition

[Ihler et al. 2012]

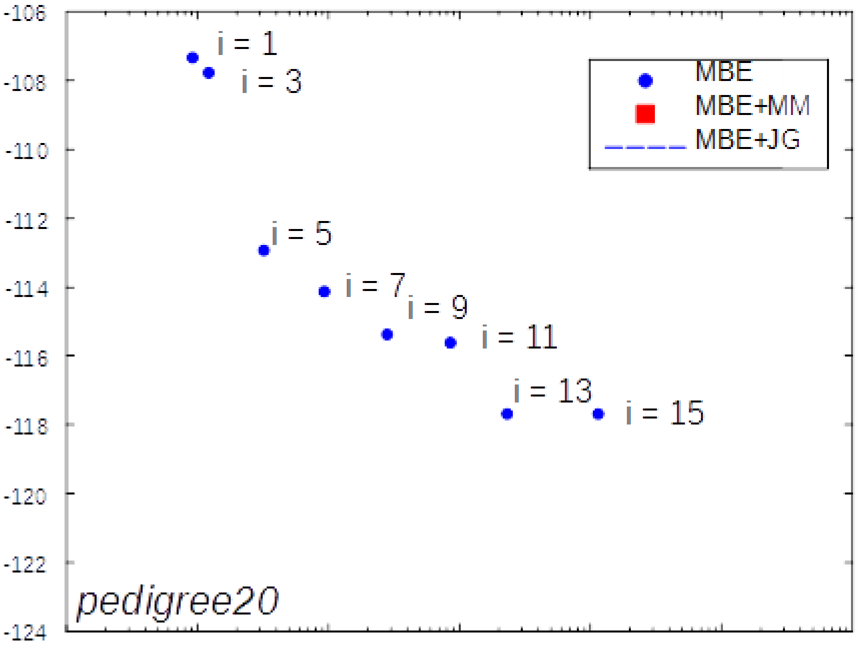
- Downward pass as cost shifting
- Can also do cost shifting within mini-buckets:
  "Join graph" message passing
- "Moment-matching" version:
  One message exchange within each bucket, during downward sweep
- Optimal bound defined by cliques ("regions") and cost-shifting f'n scopes ("coordinates")

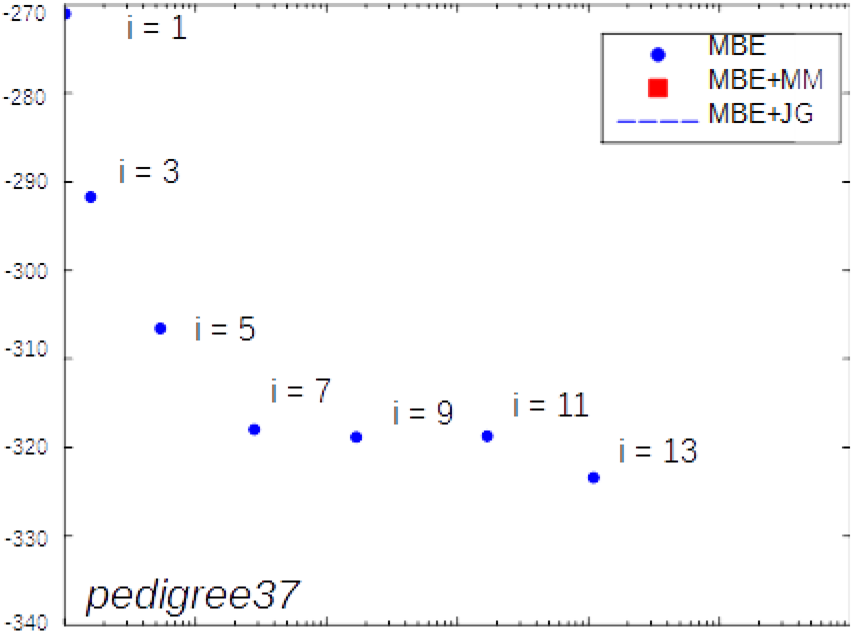
Join graph:



U = upper bound

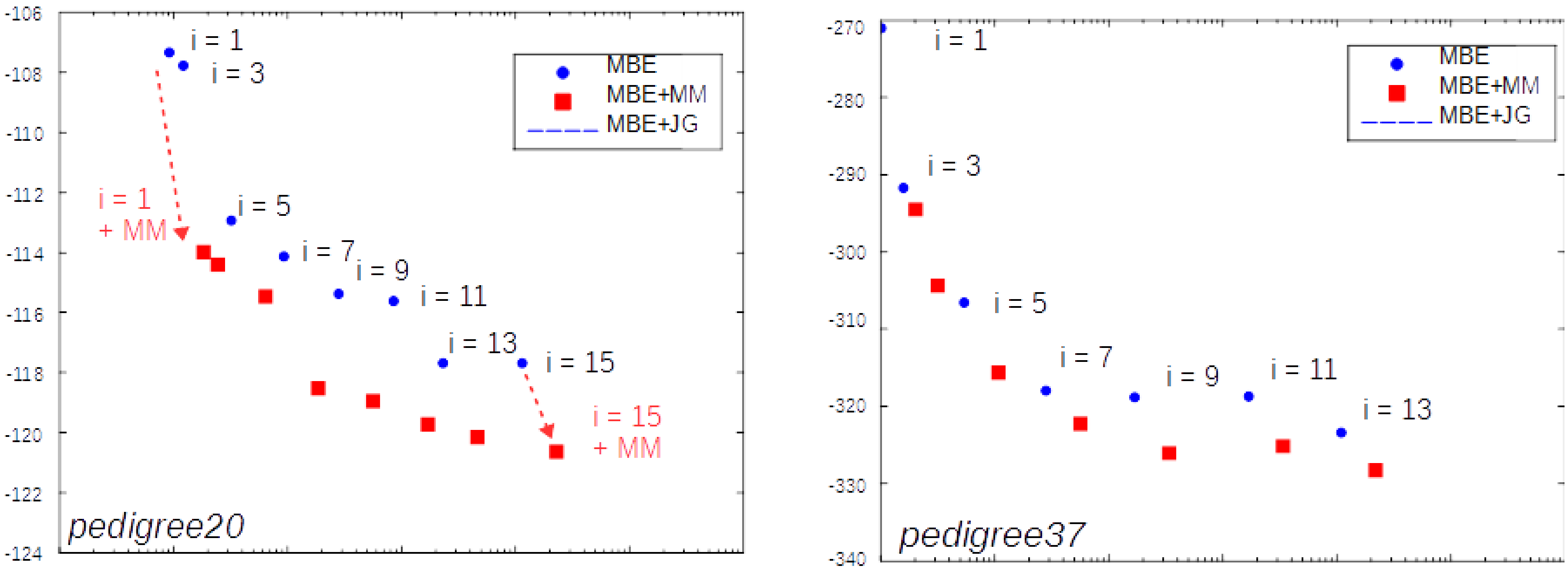
Anytime Approximation





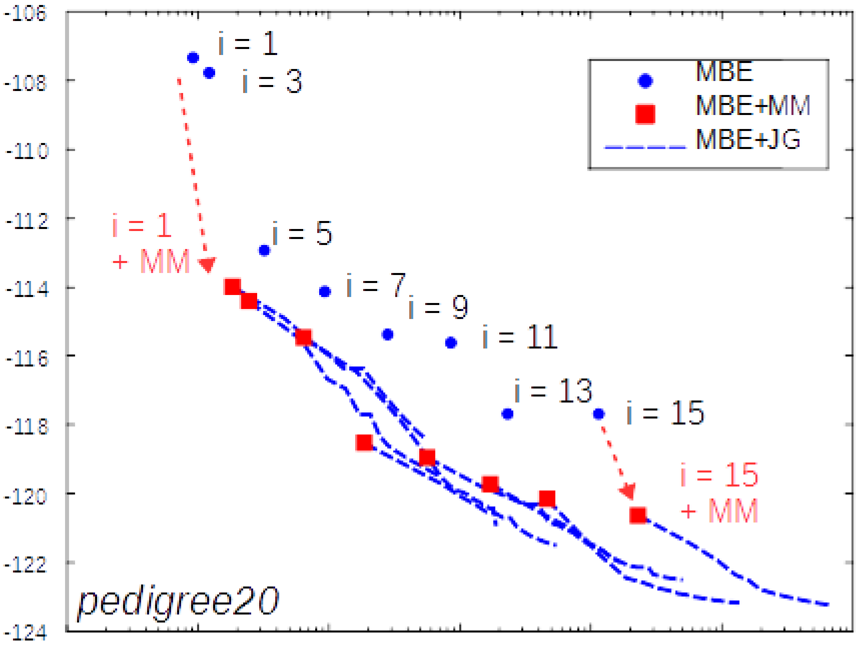
- Can tighten the bound in various ways
  - Cost-shifting (improve consistency between cliques)
  - Increase i-bound (higher order consistency)
- Simple moment-matching step improves bound significantly

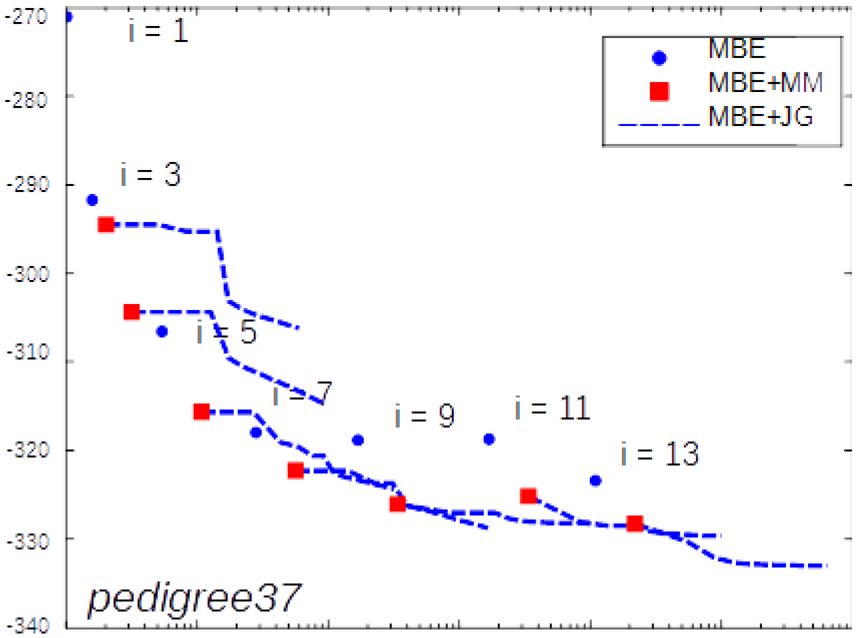
Anytime Approximation



- Can tighten the bound in various ways
  - Cost-shifting (improve consistency between cliques)
  - Increase i-bound (higher order consistency)
- Simple moment-matching step improves bound significantly

Anytime Approximation





- Can tighten the bound in various ways
  - Cost-shifting (improve consistency between cliques)
  - Increase i-bound (higher order consistency)
- Simple moment-matching step improves bound significantly

Decomposition for Sum

$$F(x) = f_1(x) \cdot f_2(x)$$

- Generalize technique to sum via Holder's inequality:

$$\sum_x f_1(x) \cdot f_2(x) \quad \leq \quad \Big[\sum_x f_1(x)^{\frac{1}{w_1}}\Big]^{w_1} \cdot \Big[\sum_x f_2(x)^{\frac{1}{w_2}}\Big]^{w_2}$$

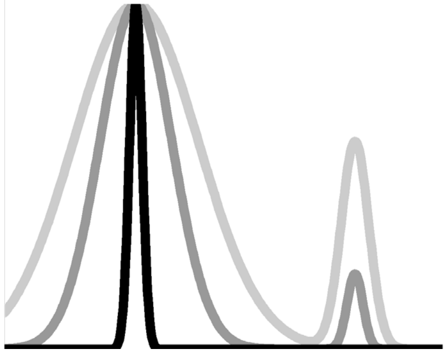
$$w_1 + w_2 = 1$$

- Define the weighted (or powered) sum:

$$\sum_{x_1}^{w_1} f(x_1) = \Big[\sum_{x_1} f(x_1)^{\frac{1}{w_1}}\Big]^{w_1}$$

  - "Temperature" interpolates between sum & max:
  - Different weights do not commute:

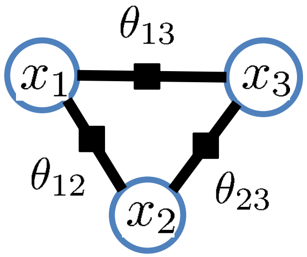
$$\sum_{x_1}^{w_1}\sum_{x_2}^{w_2} f(x_1,x_2) \neq \sum_{x_2}^{w_2}\sum_{x_1}^{w_1} f(x_1,x_2)$$

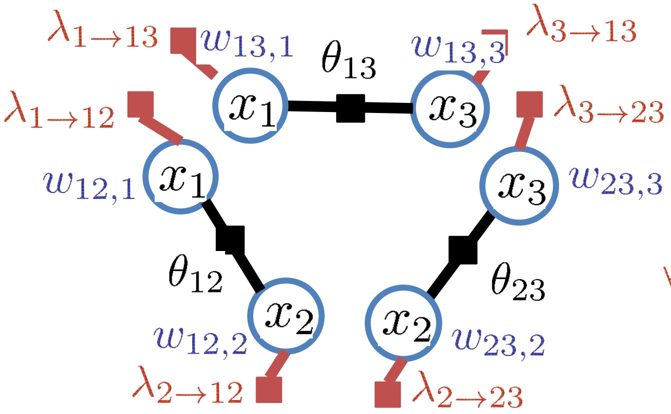


$$\lim_{w\to 0^+} \sum_x^{w} f(x) = \max_x f(x)$$

Decomposition for Sum

[Peng, Liu, Ihler 2015]





Reparameterization:

$$\forall j : \sum_{\alpha \ni j} \lambda_{j\to\alpha}(x_j) = 0$$

$$\log Z = \log \sum_{\mathbf{x}} \exp\Big[\sum_{\alpha} \theta_\alpha(\mathbf{x}_\alpha)\Big] \le \min_{\substack{\{\lambda_{i\to\alpha}\}\\ \{w_{\alpha,i}\}}} \sum_{\alpha} \log \sum_{\mathbf{x}_\alpha}^{\mathbf{w}_\alpha} \exp\Big[\theta_\alpha(\mathbf{x}_\alpha) + \sum_{i\in\alpha} \lambda_{i\to\alpha}(x_i)\Big]$$

- Fixed elimination order
- Assign weight per clique & variable

- Again, tighten bound by reparameterization
  - Can also optimize over weights

Weights:

$$\forall j : \sum_{\alpha \ni j} \mathbf{w}_{\alpha,j} = 0$$

Ex: w_{12} = [0.5 0.3 -]
w_{13} = [0.5 - 0.6]
w_{23} = [- 0.7 0.4]

Weighted Mini-bucket

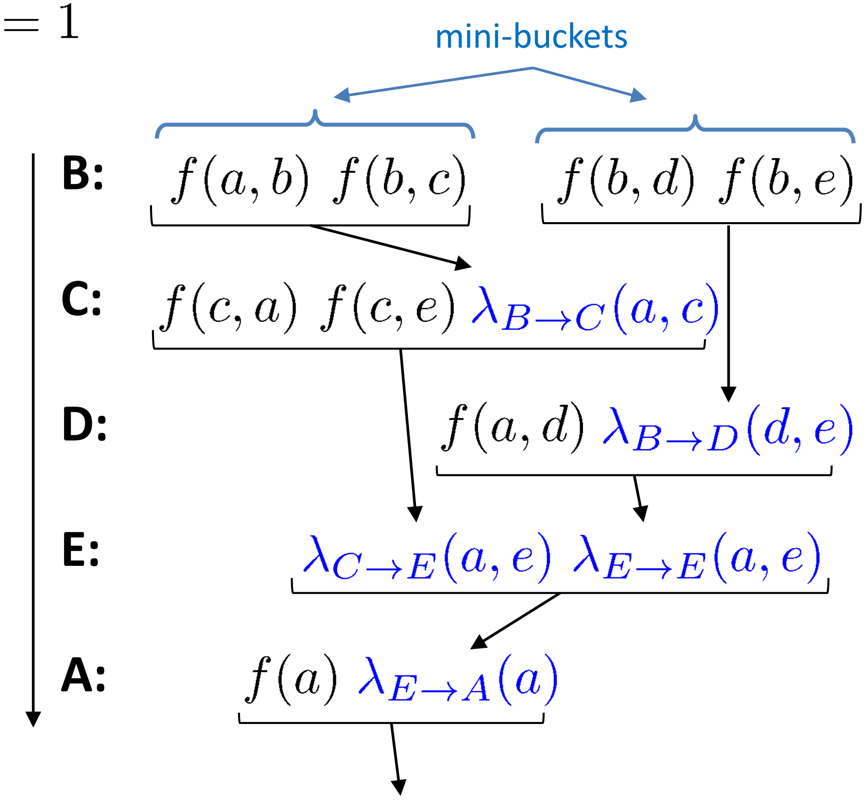
[Liu & Ihler 2011]

$$\lambda_{B\to C} = \sum_b^{w_{B1}} f(a,b)\cdot f(b,c) \qquad w_{B1} + w_{B2} = 1$$

$$\lambda_{B\to D} = \sum_b^{w_{B2}} f(b,d)\cdot f(b,e)$$

$$\lambda_{C\to E} = \sum_c f(c,a)\cdot f(c,e)\cdot \lambda_{B\to C}$$

⋮



mini-buckets

B: $f(a,b)\ f(b,c)$ $f(b,d)\ f(b,e)$

C: $f(c,a)\ f(c,e)\ \lambda_{B\to C}(a,c)$

D: $f(a,d)\ \lambda_{B\to D}(d,e)$

E: $\lambda_{C\to E}(a,e)\ \lambda_{E\to E}(a,e)$

A: $f(a)\ \lambda_{E\to A}(a)$

U = upper bound

Compute downward messages
using weighted sum

Upper bound if all weights positive
(corresponding lower bound if only one positive, rest negative)

Outline

Review: Graphical Models

Decomposition Bounds

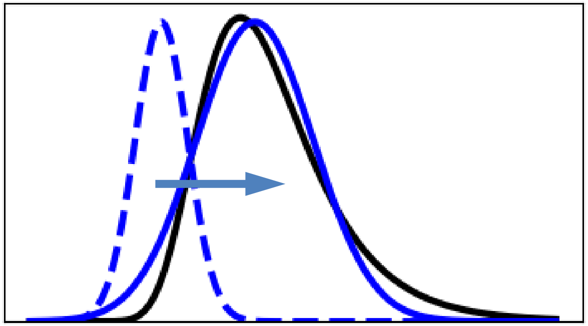
Variational Optimization

Convexity & Duality

Regions & Higher-order Approximations

Variational methods

- "Variational" = calculus of variations
  - Optimization of a "functional" (function of a function)
- Idea:
  - frame "inference" (maximization or marginals, partition f'n) as a (continuous) optimization problem
- Ex: fit a surrogate model q(x); use to answer questions about p(x)



- Why?
  - We're really good at continuous optimization: (stochastic) gradient descent, etc.
- Problem?
  - How can we optimize q(x) without inference about p(x)?

Ex: BN with Evidence

- Suppose we have a Bayesian network with some evidence E=e

$$p(x) = \tilde{p}(x|E=e)$$

“target” distribution we’re interested in

$$\propto f(x) = \tilde{p}(x, E=e)$$

but, only able to evaluate up to a constant

$$p(x) = f(x)/Z \qquad Z = p(E=e)$$

“probability of evidence”

- The KL-divergence between q & p works out very conveniently:

$$D(q||p) = -H(x;q) - \mathbb{E}_q[\log f(x)] + \log Z \ \geq 0$$

$$\Rightarrow \qquad \underline{\log Z} \geq \underline{H(x;q) + \mathbb{E}_q[\log f(x)]}$$

probability
of evidence

Evaluate or estimate from q(x)
We can maximize this over q(x)!

Sometimes called the ELBO = “Evidence Lower BOund”

Stochastic Variational Inference (in Pyro)

(1) Define our target, unnormalized model (may have evidence, etc.)

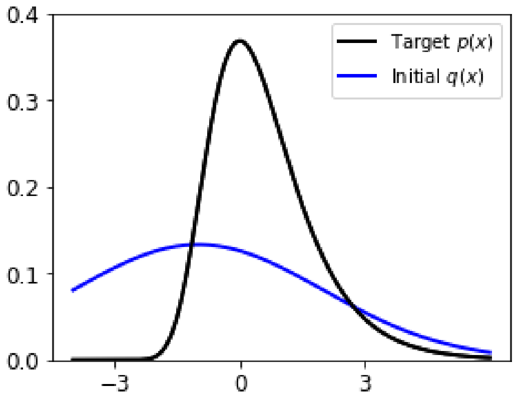
```
def model():
    X = pyro.sample('X', dist.Gumbel(torch.tensor([0.0]), torch.tensor([1.0])))
```

(2) Define our variational approximation, q(x) and initialize its parameters:

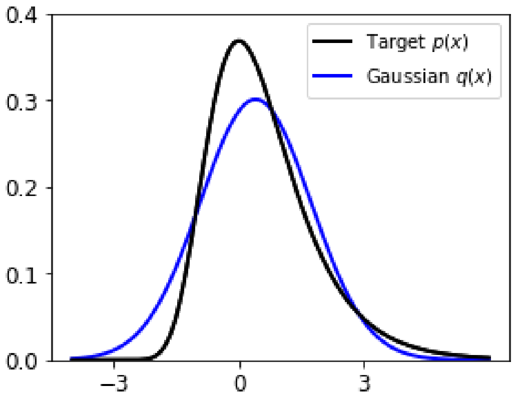
```
def guide():
    mu = pyro.param("mu", torch.tensor(-1.0) )
    var = pyro.param("var", torch.tensor(3.0), constraint=constraints.positive)
    X = pyro.sample("X", dist.Normal(mu,var))
```

(3) Optimize the bound using gradient descent

```
optimizer = pyro.optim.Adam({"lr": 0.01})
svi = pyro.infer.SVI(model, guide, optimizer, loss=pyro.infer.Trace_ELBO())
for step in range(3000): svi.step()
```



(gradient descent)



Variational methods

- Answer queries by fitting a simpler “proxy” model
  - Optimize the KL divergence (proxy to target)

$$D(q\|p) = \sum_x q(x) \log \Big[\frac{q(x)}{\frac{1}{Z} f(x)} \Big]$$

$$= \underbrace{-H(x;q) - \mathbb{E}_q[\log f(x)]}_{\text{Can evaluate or estimate from q(x)}} + \underbrace{\log Z}_{\text{Constant – depends only on f(x)!}}$$

- Needs proxy q(x) to be “easy” or “nice”!
  - What kinds of q(x) are nice?
  - Need to be able to evaluate expectations & evaluate/estimate entropy
  - Continuous-valued x? q(x) Gaussian, etc.
  - Discrete x? High-dimensional x?   Make q(x) simple in terms of its graph!

Mean Field

- We can design lower bounds by restricting q(x)
  - Naïve mean field: q(x) is fully independent
  - Entropy H(q) is then easy:

$$q(x) = \prod_i q_i(x_i)$$

) $H(q) = \sum_i H(q_i)$

- Optimizing the bound via coordinate ascent:

$$\mathbb{E}_q[\theta(x)] + H(q) = \mathbb{E}_q\Big[\sum_{\alpha\ni i} \theta_\alpha(x_\alpha)\Big] + H(q_i) + \text{const}$$

$$= \mathbb{E}_{q_i}\Big[\log g(x_i)\Big] + H(q_i)$$

$$= D(\, q_i \,\|\, g_i \,)$$

$$\log g_i(x_i) = \mathbb{E}_{q_{\neg i}}\Big[\sum_{\alpha\ni i} \theta_\alpha(x_\alpha)\Big]$$

$$q_{\neg i}(x) = \prod_{j\neq i} q_j(x_j)$$

Coordinate update:

$$\Longrightarrow \quad q_i(x_i) \propto \exp\Big[\, \mathbb{E}_{q_{\neg i}}\Big[\sum_{\alpha\ni i} \theta_\alpha(x_\alpha)\Big]\Big]$$

Mean Field

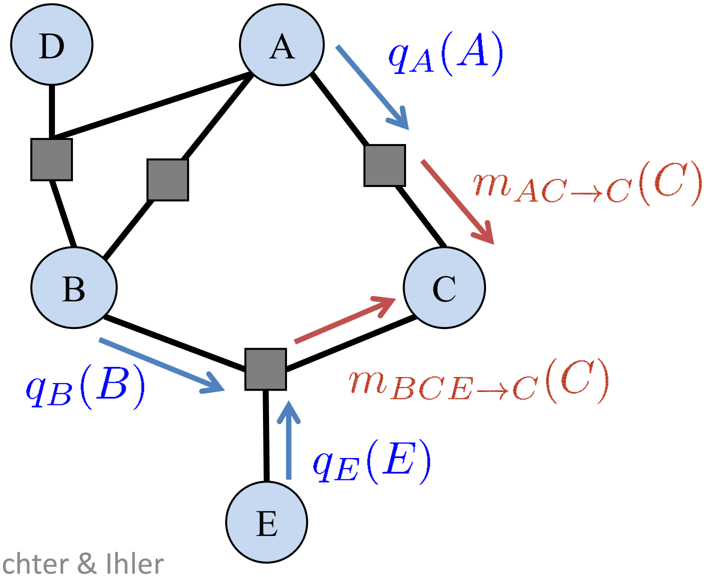
- We can design lower bounds by restricting q(x)
  - Naïve mean field: q(x) is fully independent
  - Entropy H(q) is then easy:

$$q(x) = \prod_i q_i(x_i)$$

$$\Rightarrow \quad H(q) = \sum_i H(q_i)$$

- Optimizing the bound via coordinate ascent:

$$q_i(x_i) \propto \exp\Big[\mathbb{E}_{q_{\neg i}}\big[\sum_{\alpha \ni i} \theta_\alpha(x_\alpha) \big] \Big]$$



"Message passing" interpretation:
Updates depend only on Xi's Markov blanket

Naïve Mean Field

```
1: Initialize $\{q_i(X_i)\}$
2: while not converged do
3:     for $i = 1 \dots n$ do
4:         $m_{\alpha\to i}(x_i) = \exp\Big[\sum_{x_{\alpha\setminus i}} \theta_\alpha(x_\alpha) \prod_{j\in\alpha\setminus i} q_j(x_j)\Big]$
5:         $q_i(x_i) \propto \prod_{\alpha\ni i} m_{\alpha\to i}(x_i)$
```

Optimization Perspective

- “Variational” = calculus of variations
  - Optimization of a “functional” (function of a function)

- **Exponential family distributions**
  - Inference tasks are **convex** in the model natural parameters!

- Very elegant perspective based on convex optimization

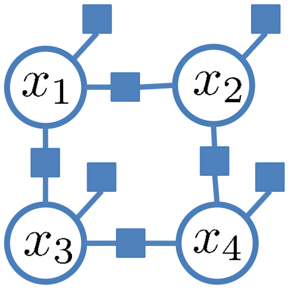
  (discussed here for background / perspective)

Vector space representation

- Represent the (log) model and state in a vector space

$$\theta(x) = \theta_1(x_1) + \theta_2(x_2) + \ldots + \theta_{12}(x_1, x_2) + \ldots$$

$$x = [x_1, x_2, x_3, x_4] = [\,1\ ,\ 0\ ,\ 1\ ,\ 1\,]$$



X_1	$\theta(X_1)$
0	0.5
1	0.75

…

X_1	X_2	$\theta(X_1, X_2)$
0	0	-1.0
0	1	2.0
1	0	0.0
1	1	1.0

…

$\theta_1(x_1)$

$\theta_2(x_2)$

$\theta_{12}(x_1, x_2)$

$\vec{\theta}$	$\vec{x}$
0.5	0
0.75	1
.	1
.	0
…	…
-1.0	0
2.0	0
0.0	1
1.0	0
…	….

$x_1 = 1$

$x_2 = 0$

$(x_1, x_2) = (1, 0)$

Evaluating the function is a dot product in the vector space:

$$\theta(x) \;=\; \vec{\theta} \cdot \vec{x}$$

Inference Tasks & Convexity

- Distribution is log-linear (exponential family):

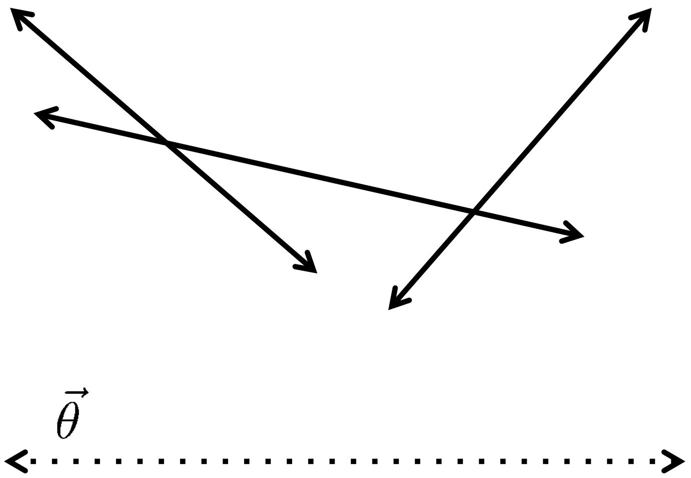
$$p(x) = \frac{1}{Z} f(x) \propto \exp\left[\vec{\theta} \cdot u(x)\right]$$

$\vec{\theta}$ “natural parameters”

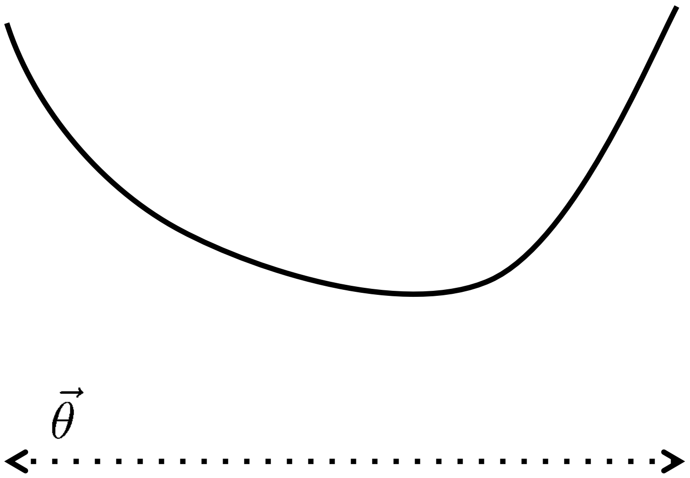
$u(x) = \vec{x}$ “features”

- Tasks of interest are convex functions of the model:

$$\log f(\mathbf{x}^*) = \max_{\vec{x} \in \mathcal{X}} \vec{\theta} \cdot \vec{x}$$



$$\log Z = \log \sum_{\vec{x} \in \mathcal{X}} \exp\left[\vec{\theta} \cdot \vec{x}\right]$$

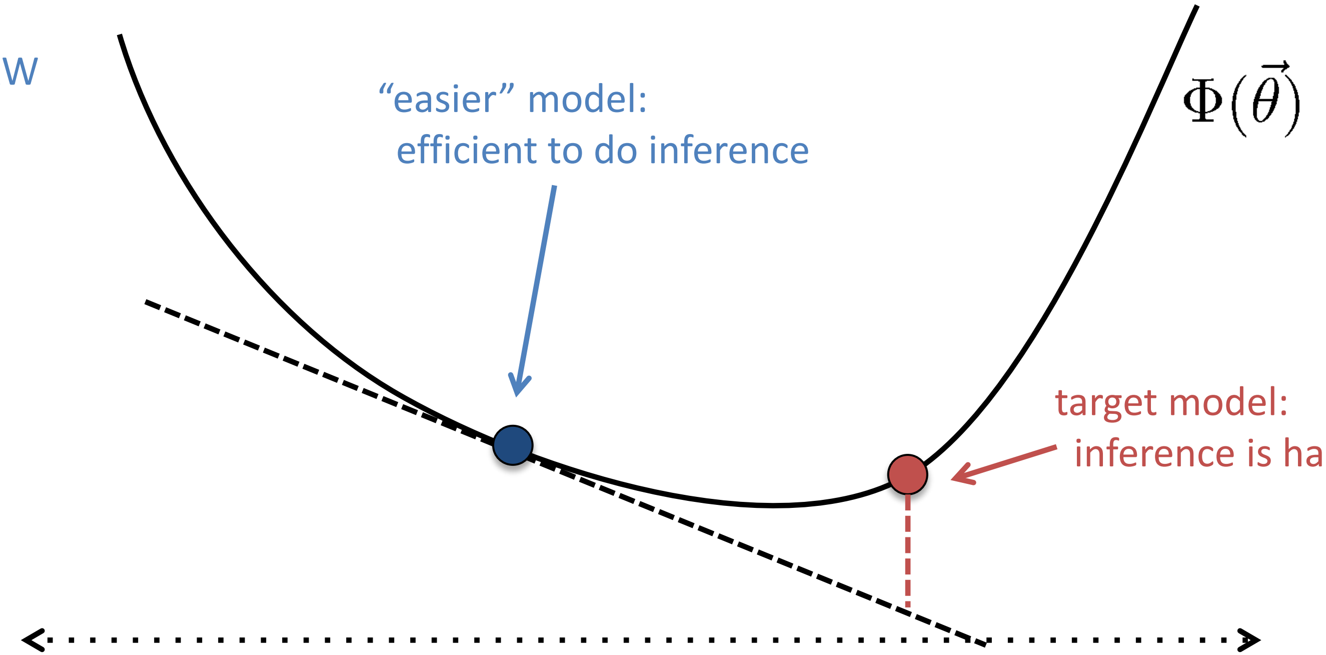


Bounds via Convexity

- Convexity relates target to “nearby” models
  - Some of these models are easy to solve! (trees, etc.)
  - Inference at easy models + convexity tells us something about our model!

- Lower bounds:

Mean field
Negative TRW
…



“easier” model:
efficient to do inference

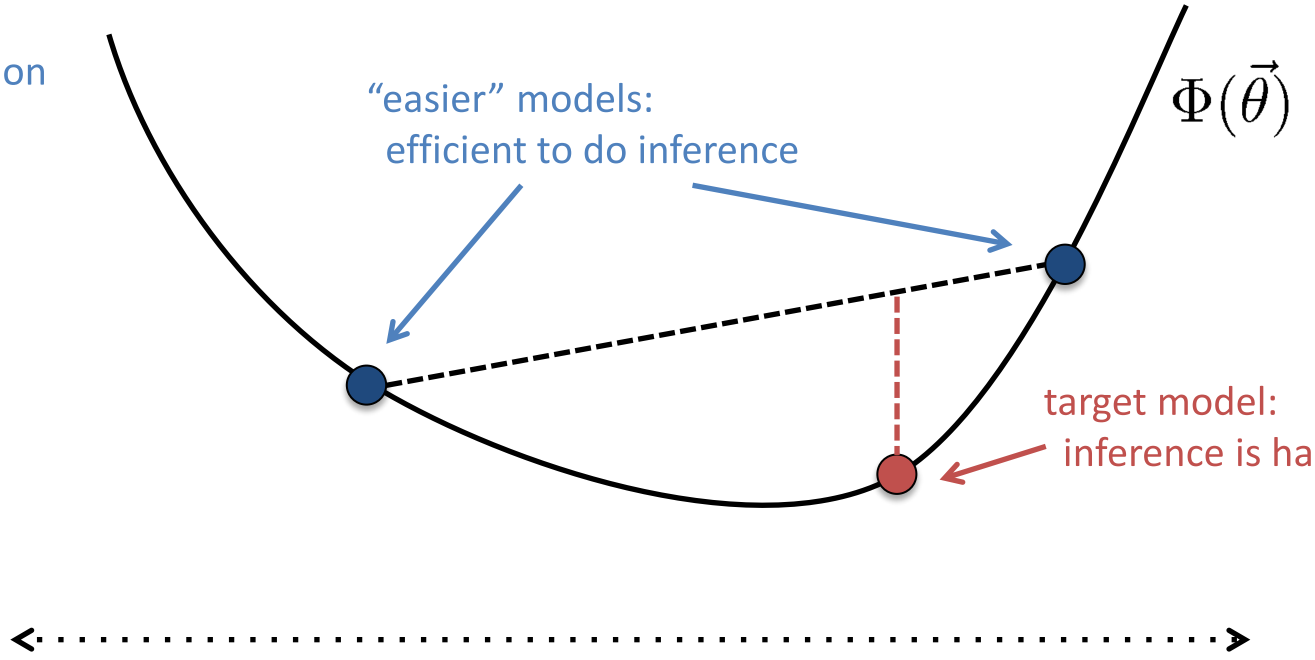
target model:
inference is hard!

Bounds via Convexity

- Convexity relates target to “nearby” models
  - Some of these models are easy to solve! (trees, etc.)
  - Inference at easy models + convexity tells us something about our model!

- Upper bounds:

TRW
Decomposition
…



Tree-reweighted MAP

$$\Phi_0(\vec{\theta}) = \max_{\vec{x} \in \mathcal{X}} \vec{\theta} \cdot \vec{x}$$

- Let T_1, T_2 be two (or more) tree-structured models, with

$$\vec{\theta} \; = \; w_1 \, \vec{\theta}^{(1)} + w_2 \, \vec{\theta}^{(2)}$$

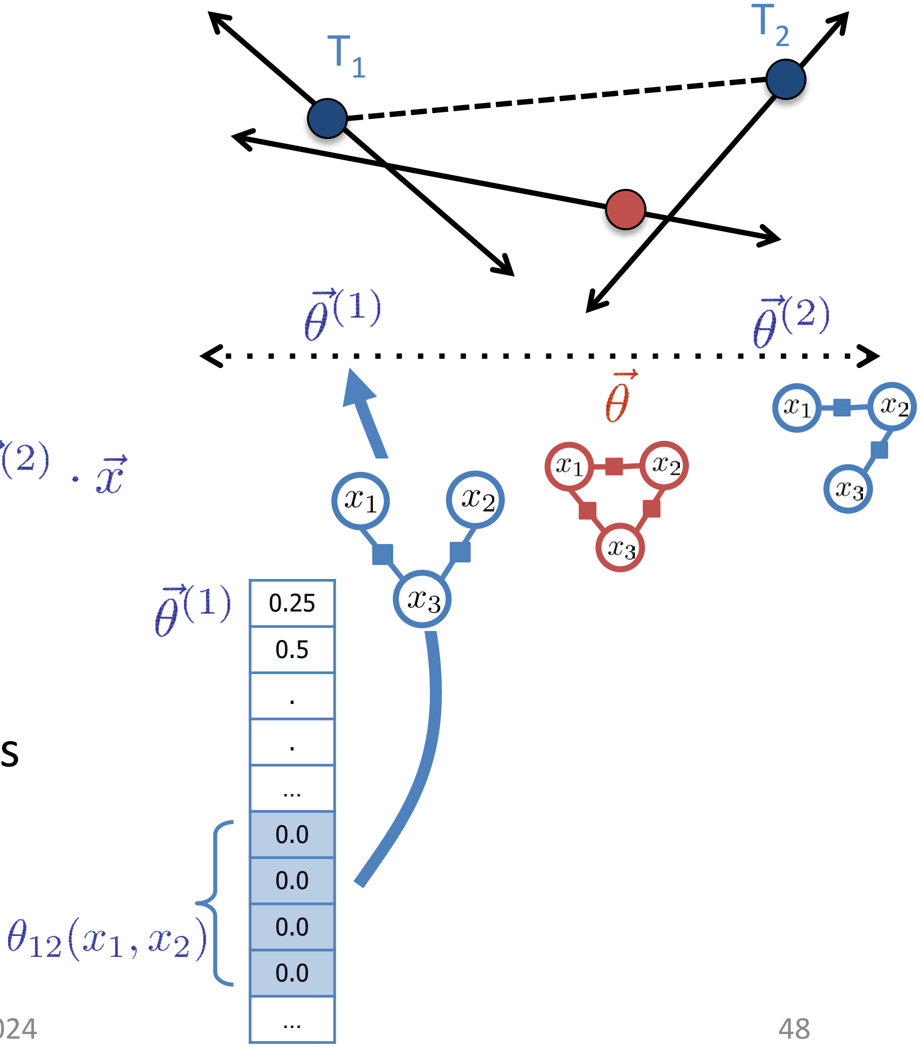
- Each T_i is easy to solve:

$$\vec{x}^{*(1)} = \max_{\vec{x}} \; \vec{\theta}^{(1)} \cdot \vec{x}$$

- And by convexity,

$$\max_{\vec{x}} \vec{\theta} \cdot \vec{x} \; \leq \; w_1 \max_{\vec{x}} \vec{\theta}^{(1)} \cdot \vec{x} + w_2 \max_{\vec{x}} \vec{\theta}^{(2)} \cdot \vec{x}$$

- Minimize bound?
  - Convex objective, linear constraints



Decomposition Bounds

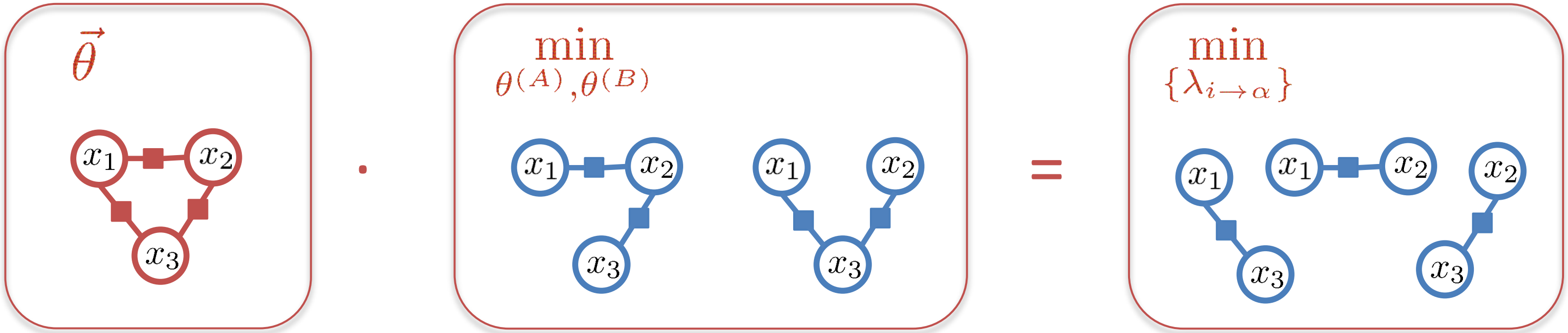
- TRW MAP is equivalent to MAP decomposition

$$\max_{\vec{x}} \left[\vec{\theta}\cdot\vec{x}\right] \le \min_{\theta^{(1)},\theta^{(2)}} \max_{\vec{x}} \left[w_1\,\vec{\theta}^{(1)}\cdot\vec{x}\right] + \max_{\vec{x}} \left[w_2\vec{\theta}^{(2)}\cdot\vec{x}\right] \qquad \vec{\theta} = w_1\,\vec{\theta}^{(1)} + w_2\,\vec{\theta}^{(2)}$$

$$= \min_{\theta^{(A)},\theta^{(B)}} \max_{\vec{x}} \left[\vec{\theta}^{(A)}\cdot\vec{x}\right] + \max_{\vec{x}} \left[\vec{\theta}^{(B)}\cdot\vec{x}\right] \qquad \vec{\theta} = \vec{\theta}^{(A)} + \vec{\theta}^{(B)}$$

$$= \min_{\{\lambda_{i\to\alpha}\}} \sum_{\alpha} \max_{\vec{x}_\alpha} \left[(\vec{\theta}_\alpha + \sum_{i} \vec{\lambda}_{i\to\alpha})\cdot\vec{x}_\alpha\right] \qquad \vec{0} = \sum_{\alpha\ni i} \vec{\lambda}_{i\to\alpha}$$

(on trees, decomposition bound = exact inference)



More compact
Faster optimization
Reparameterization “messages”

Tree-reweighted Sum

$$\Phi_1(\vec{\theta}) = \log \sum_{\vec{x} \in \mathcal{X}} \exp\left[\vec{\theta} \cdot \vec{x}\right]$$

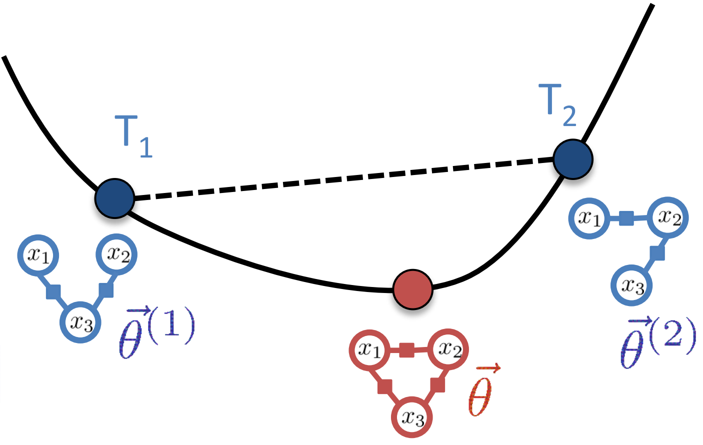
- Let T_1, T_2 be two (or more) tree-structured models, with

$$\vec{\theta} \;=\; w_1\, \vec{\theta}^{(1)} + w_2\, \vec{\theta}^{(2)} \;=\; \vec{\theta}^{(A)} + \vec{\theta}^{(B)}$$

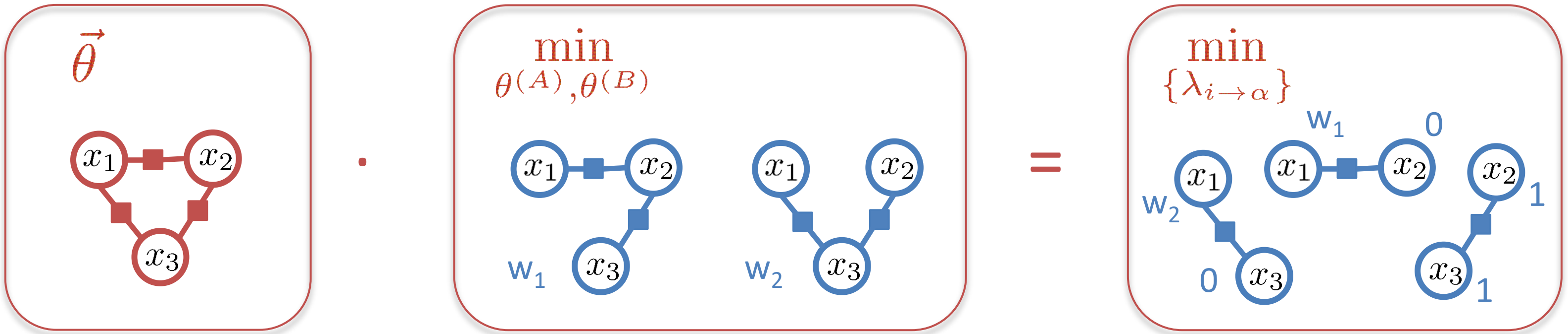
- Again, we have

$$\Phi_1(\vec{\theta}) \;\le\; w_1\, \Phi_1(\vec{\theta}^{(1)}) + w_2\, \Phi_1(\vec{\theta}^{(2)})$$

$$= \log \sum_{\vec{x}}^{w_1} \exp\left[\vec{\theta}^{(A)} \vec{x}\right] + \log \sum_{\vec{x}}^{w_2} \exp\left[\vec{\theta}^{(B)} \vec{x}\right]$$



$$\sum_{x}^{w} f(x) = \left[\sum_{x} f(x)^{\frac{1}{w}}\right]^{w}$$



$\vec{\theta}$
x_1
x_2
x_3
·
$\min_{\theta^{(A)},\theta^{(B)}}$
w_1
w_2
=
$\min_{\{\lambda_{i \to \alpha}\}}$
w_1
0
w_2
0
1
1

(if T_1, T_2 share an elimination order)

Outline

Review: Graphical Models

Decomposition Bounds

Variational Optimization

Convexity & Duality

Regions & Higher-order Approximations

Variational forms

- Reframe inference task as an optimization over distributions q(x)
- Ex: MAP inference $\max_x \log f(x) = \log f(x^*) = \max_{q \in \mathbb{P}} \; \mathbb{E}_q[\log f(x)]$

Optimal q(x) puts all mass on optimal value(s) of x: $q^*(x) = \mathbb{1}[x = x^*]$
(mass on any other values of x reduces the average)

- Sum inference: $\log Z = \log \sum_x f(x) = \max_{q \in \mathbb{P}} \; \mathbb{E}_q[\log f(x)] + H(x\,;\,q)$

Proof:

$$D(q\|p) = \sum_x q(x) \log \left[\frac{q(x)}{\frac{1}{Z} f(x)} \right]$$

(Kullback–Leibler divergence)

$$= -H(x\,;\,q) - \mathbb{E}_q[\log f(x)] + \log Z$$

$$\Rightarrow \log Z \geq \mathbb{E}_q[\log f(x)] + H(x\,;\,q)$$

Equal iff
$q(x) = p(x) = \frac{1}{Z} f(x)$

- How to optimize over distributions q?

The marginal polytope

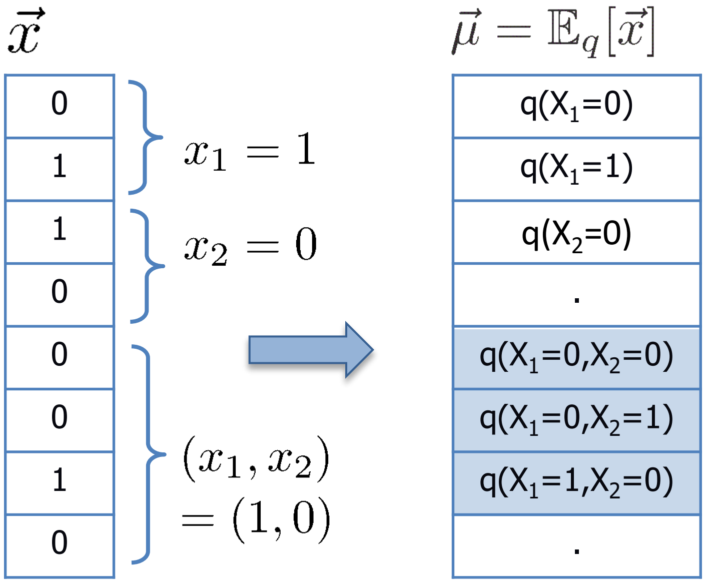
- Rewrite $\log f(x^*) = \max_{q \in \mathbb{P}} \mathbb{E}_q[\log f(x)] = \max_{q \in \mathbb{P}} \mathbb{E}_q[\vec{\theta} \cdot \vec{x}] \quad = \quad \max_{\vec{\mu} \in \mathcal{M}} \vec{\theta} \cdot \vec{\mu}$

and similarly, $\log Z \quad = \quad \max_{\vec{\mu} \in \mathcal{M}} \vec{\theta} \cdot \vec{\mu} + H(\vec{\mu})$

(max entropy given [1])

$\vec{\mu} = \mathbb{E}_q[\vec{x}]$

(the marginal probabilities of q)

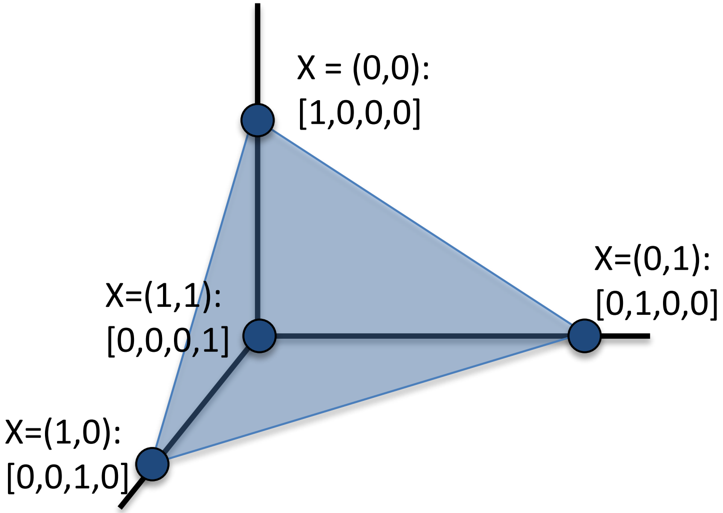


$\vec{x}$		$\vec{\mu} = \mathbb{E}_q[\vec{x}]$
0	$x_1 = 1$	q(X_1=0)
1		q(X_1=1)
1	$x_2 = 0$	q(X_2=0)
0		.
0	$(x_1, x_2) = (1, 0)$	q(X_1=0,X_2=0)
0		q(X_1=0,X_2=1)
1		q(X_1=1,X_2=0)
0		.

$\mathcal{M} = \{\, \vec{\mu} \;:\; \exists q : \vec{\mu} = \mathbb{E}_q[\vec{x}] \,\}$

(set of all valid marginal probabilities of q)

"marginal polytope"



Variational perspectives

- Replace q 2 P and H(q) with simpler approximations

$$\log p(x^*) = \max_{q \in \mathbb{P}} \; \mathbb{E}_q[\log f(x)]$$

$$\log Z = \max_{q \in \mathbb{P}} \; \mathbb{E}_q[\log f(x)] + H(x\,;\,q)$$

- Algorithms and their properties:

	Method	distributions	entropy	value
Max:	Linear programming	$q \in \mathbb{L} \supseteq \mathbb{P}$	n/a	$\hat{p}_{lp} \geq p(x^*)$
Sum:	Mean field	$\{q = \prod q_i(x_i)\} \subseteq \mathbb{P}$	exact	$Z_{mf} \leq Z$
	Belief propagation	$q \in \mathbb{L} \supseteq \mathbb{P}$	$H_\beta \approx H(q)$	$Z_\beta \approx Z$
	Tree-reweighted	$q \in \mathbb{L} \supseteq \mathbb{P}$	$H_{tr} \geq H(q)$	$Z_{tr} \geq Z$

Variational perspectives

- Replace q 2 P and H(q) with simpler approximations

$$\log p(x^*) = \max_{q \in \mathbb{P}} \; \mathbb{E}_q[\log f(x)]$$

$$\log Z = \max_{q \in \mathbb{P}} \; \mathbb{E}_q[\log f(x)] + H(x\,;\,q)$$

- Algorithms and their properties:

	Method	distributions	entropy	value
Max:	Linear programming	$q \in \mathbb{L} \supseteq \mathbb{P}$	n/a	$\hat{p}_{lp} \geq p(x^*)$
Sum:	**Mean field**	$\{q = \prod q_i(x_i)\} \subseteq \mathbb{P}$	exact	$Z_{mf} \leq Z$
	Belief propagation	$q \in \mathbb{L} \supseteq \mathbb{P}$	$H_\beta \approx H(q)$	$Z_\beta \approx Z$
	Tree-reweighted	$q \in \mathbb{L} \supseteq \mathbb{P}$	$H_{tr} \geq H(q)$	$Z_{tr} \geq Z$

Naïve Mean Field

- Subset of M corresponding to independent distributions?
  - Includes all vertices (configurations of x), but not all distributions
  - Non-convex set; coordinate ascent has local optima

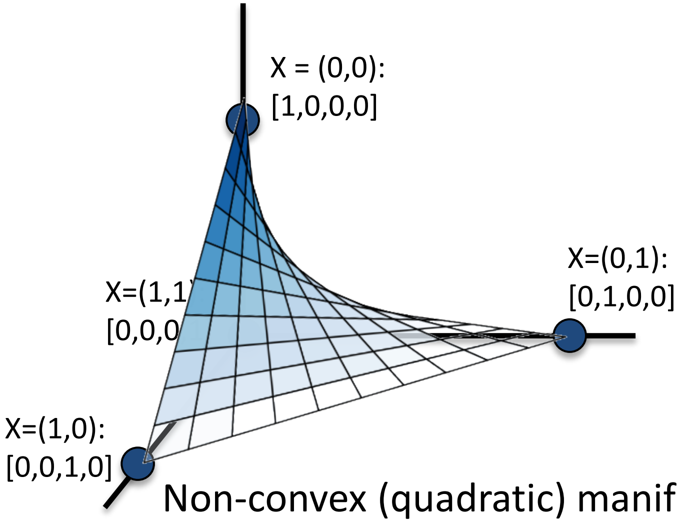
$$q(x) = \prod_i q_i(x_i)$$

$\vec{\mu} = \mathbb{E}_q[\vec{x}]$
$q(X_1=0)$
$q(X_1=1)$
$q(X_2=0)$
$q(X_2=1)$
$q(X_1=0,X_2=0)$
$q(X_1=0,X_2=1)$
$q(X_1=1,X_2=0)$
.

$\vec{\mu} = \mathbb{E}_q[\vec{x}]$
$1-q_1$
q_1
$1-q_2$
q_2
$(1-q_1) \times (1-q_2)$
$(1-q_1) \times q_2$
$q_1 \times (1-q_2)$
.

$$\mathrm{MF} = \{\, \vec{\mu} \;:\; \exists \{q_i\} : \vec{\mu} = \mathbb{E}_q[\vec{x}] \,\}$$

(set of marginal probabilities of independent q)



Non-convex (quadratic) manifold!

Variational perspectives

- Replace q 2 P and H(q) with simpler approximations

$$\log p(x^*) = \max_{q \in \mathbb{P}} \; \mathbb{E}_q[\log f(x)]$$

$$\log Z = \max_{q \in \mathbb{P}} \; \mathbb{E}_q[\log f(x)] + H(x\,;\,q)$$

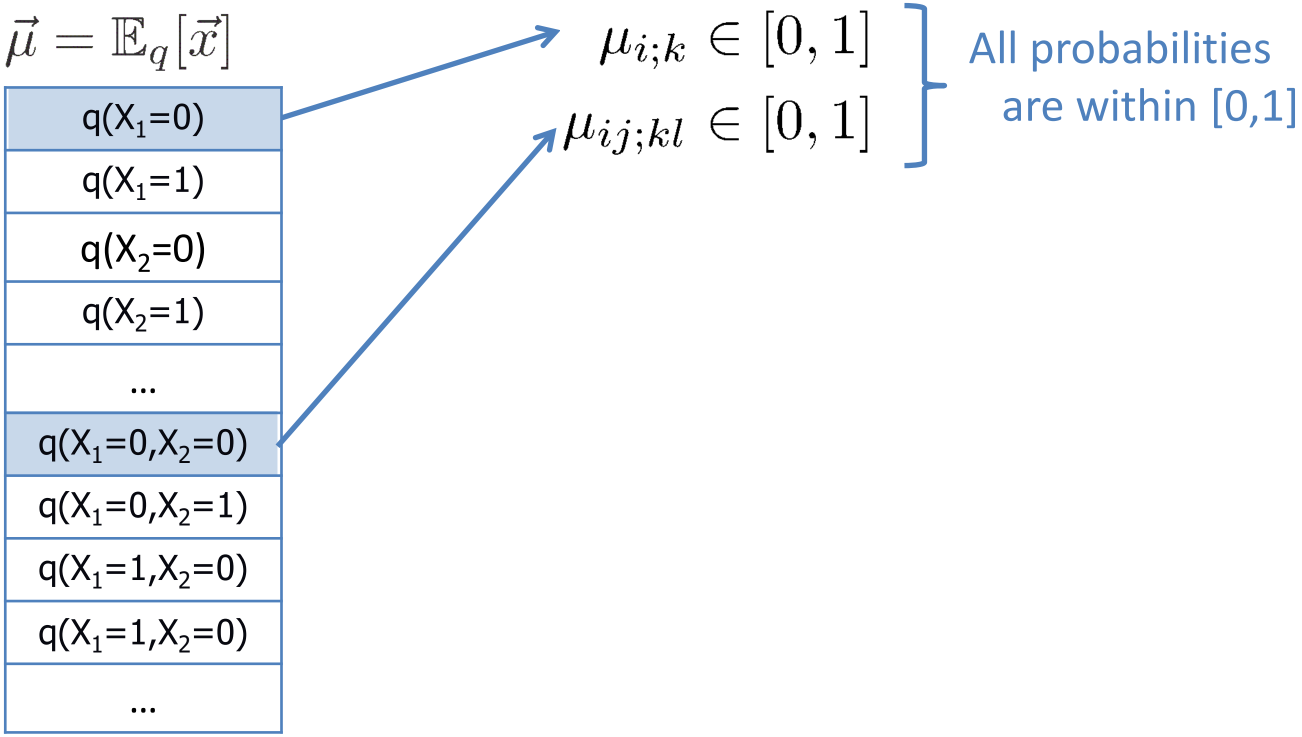
- Algorithms and their properties:

	Method	distributions	entropy	value
Max:	**Linear programming**	$q \in \mathbb{L} \supseteq \mathbb{P}$	n/a	$\hat{p}_{lp} \geq p(x^*)$
Sum:	Mean field	$\{q = \prod q_i(x_i)\} \subseteq \mathbb{P}$	exact	$Z_{mf} \leq Z$
	Belief propagation	$q \in \mathbb{L} \supseteq \mathbb{P}$	$H_\beta \approx H(q)$	$Z_\beta \approx Z$
	Tree-reweighted	$q \in \mathbb{L} \supseteq \mathbb{P}$	$H_{tr} \geq H(q)$	$Z_{tr} \geq Z$

The local polytope

- Unfortunately, M has a large number of constraints
  - Enforce only a few, easy to check constraints?
  - Equivalent to a linear programming relaxation of original ILP

$\mu \in \mathbb{L}$: “local consistency” polytope



$$\vec{\mu} = \mathbb{E}_q[\vec{x}]$$

q(X_1=0)

q(X_1=1)

q(X_2=0)

q(X_2=1)

…

q(X_1=0,X_2=0)

q(X_1=0,X_2=1)

q(X_1=1,X_2=0)

q(X_1=1,X_2=0)

…

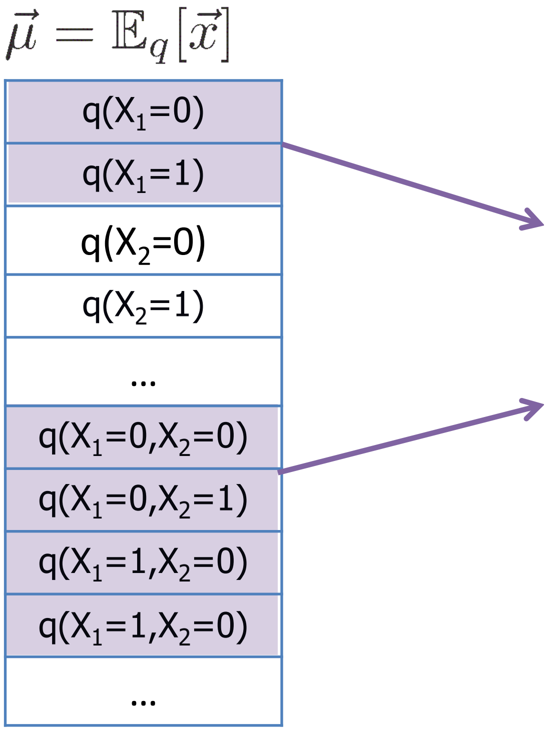
$$\left.\begin{array}{l} \mu_{i;k} \in [0,1] \\ \mu_{ij;kl} \in [0,1] \end{array}\right\}$$

All probabilities are within [0,1]

The local polytope

- Unfortunately, M has a large number of constraints
  - Enforce only a few, easy to check constraints?
  - Equivalent to a linear programming relaxation of original ILP

$\mu \in \mathbb{L}$: "local consistency" polytope



$$\vec{\mu} = \mathbb{E}_q[\vec{x}]$$

$q(X_1=0)$
$q(X_1=1)$
$q(X_2=0)$
$q(X_2=1)$
…
$q(X_1=0,X_2=0)$
$q(X_1=0,X_2=1)$
$q(X_1=1,X_2=0)$
$q(X_1=1,X_2=0)$
…

$$\mu_{i;k} \in [0,1]$$

$$\mu_{ij;kl} \in [0,1]$$

All probabilities are within [0,1]

$$\sum_k \mu_{i;k} = 1$$

$$\sum_{k,l} \mu_{ij;kl} = 1$$

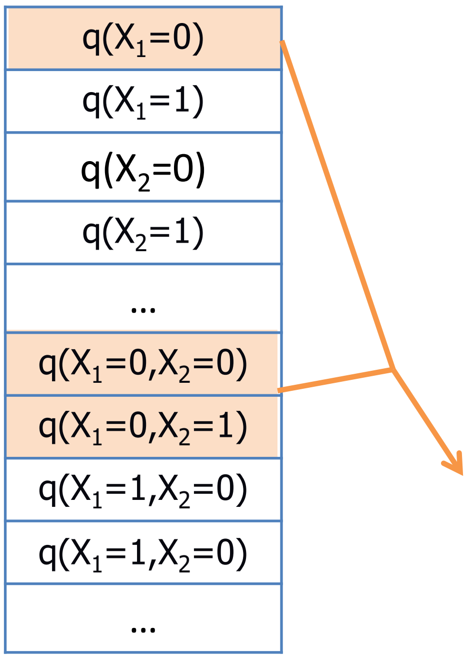
Each marginal probability is normalized to sum to one

The local polytope

- Unfortunately, M has a large number of constraints
  - Enforce only a few, easy to check constraints?
  - Equivalent to a linear programming relaxation of original ILP

$\mu \in \mathbb{L}$: "local consistency" polytope

$\vec{\mu} = \mathbb{E}_q[\vec{x}]$



q(X_1=0)
q(X_1=1)
q(X_2=0)
q(X_2=1)
…
q(X_1=0,X_2=0)
q(X_1=0,X_2=1)
q(X_1=1,X_2=0)
q(X_1=1,X_2=0)
…

$$\left.\begin{aligned} \mu_{i;k} &\in [0,1] \\ \mu_{ij;kl} &\in [0,1] \end{aligned}\right\}$$

All probabilities are within [0,1]

$$\left.\begin{aligned} \sum_k \mu_{i;k} &= 1 \\ \sum_{k,l} \mu_{ij;kl} &= 1 \end{aligned}\right\}$$

Each marginal probability is normalized to sum to one

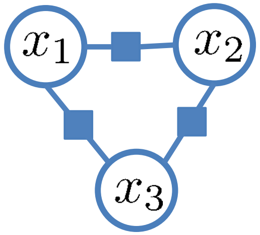
$$\left.\sum_l \mu_{ij;kl} = \mu_{i;k}\right\}$$

Marginal of (x_i, x_j) is consistent with marginal of x_i

$$\sum_k \mu_{ij;kl} = \mu_{j;l}$$

(& similarly, consistent with x_j)

The local polytope



- Local polytope does not enforce all the constraints of M:
  - Ex: all pairwise probabilities locally consistent, but no joint q(x) exists:

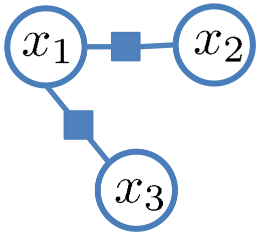
$$\mu_1 = \mu_2 = \mu_3 \quad \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

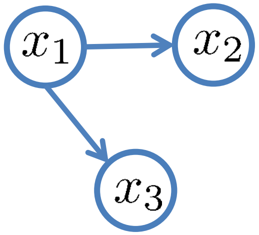
$$\mu_{12} \quad \begin{array}{c|c} & x_2 \\ \hline x_1 & \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \end{array} \qquad \mu_{13} \quad \begin{array}{c|c} & x_3 \\ \hline x_1 & \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \end{array} \qquad \mu_{23} \quad \begin{array}{c|c} & x_3 \\ \hline x_2 & \begin{bmatrix} 0 & 0.5 \\ 0.5 & 0 \end{bmatrix} \end{array}$$

$$(x_1 = x_2) \qquad (x_1 = x_3) \qquad (x_2 \neq x_3)$$

(also illustrates connection to arc consistency in CSPs, etc.)

- But, trees remain easy
  - If we only specify the marginals on a tree, we can construct q(x)





$$\begin{aligned} q(x) &= q(x_1) \cdot q(x_2|x_1) \cdot q(x_3|x_1) \\ &= \mu_1 \cdot \frac{\mu_{12}}{\mu_1} \cdot \frac{\mu_{13}}{\mu_1} \end{aligned}$$

$\mathbb{L} = \mathbb{M}$ on tree-structured distributions

Duality relationship

- Local polytope LP & MAP decomposition are Lagrangian duals:

$$\log f(x^*) \le \max_{\mu} \Big[\sum_{i,k} \theta_{i;k}\,\mu_{i;k} + \sum_{i,j,k,l} \theta_{ij;kl}\,\mu_{ij;kl} \Big]$$

subject to (a) normalization constraints (enforce explicitly)
(b) consistency: $\sum_l \mu_{ij;kl} = \mu_{i;k}$, $\sum_k \mu_{ij;kl} = \mu_{j;l}$ (use Lagrange)

$$L = \max_{\mu} \min_{\lambda} \sum_{i,k} \theta_{i;k}\,\mu_{i;k} + \sum_{i,j,k,l} \theta_{ij;kl}\,\mu_{ij;kl} + \sum_{i,j,k} \lambda_{i\to ij;k} \big(\sum_{l} \mu_{ij;kl} - \mu_{i;k} \big)$$

$$\le \min_{\lambda} \max_{\mu} \sum_{i,k} \theta_{i;k}\,\mu_{i;k} + \sum_{i,j,k,l} \theta_{ij;kl}\,\mu_{ij;kl} + \sum_{i,j,k} \lambda_{i\to ij;k} \big(\sum_{l} \mu_{ij;kl} - \mu_{i;k} \big)$$

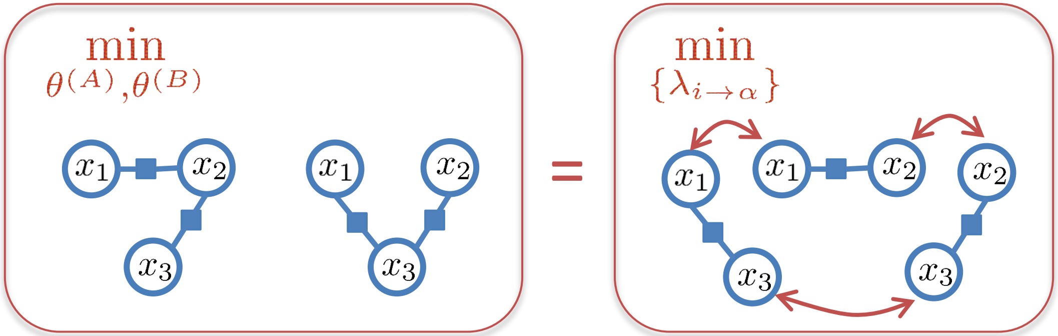
$$= \min_{\lambda} \max_{\mu} \sum_{i,k} \big(\theta_{i;k} - \sum_{j} \lambda_{i\to ij;k}\big)\mu_{i;k} + \sum_{i,j,k,l} \big(\theta_{ij;kl} + \lambda_{i\to ij;k} + \lambda_{j\to ij;l}\big)\mu_{ij;kl}$$

$$= \min_{\lambda} \sum_{i,k} \max_{k} \big(\theta_{i;k} - \sum_{j} \lambda_{i\to ij;k}\big) + \sum_{i,j,k,l} \max_{k,l} \big(\theta_{ij;kl} + \lambda_{i\to ij;k} + \lambda_{j\to ij;l}\big)$$

Duality: MAP

Primal

$$\min_{\{\lambda_{i\to\alpha}\}} \sum_{\alpha} \max_{\mathbf{x}_\alpha} \left[\theta_\alpha(\mathbf{x}_\alpha) + \sum_{i \in \alpha} \lambda_{i\to\alpha}(x_i) \right]$$

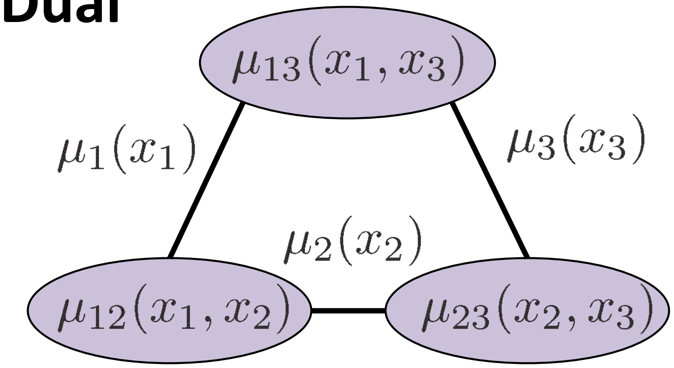

$\min_{\theta^{(A)},\theta^{(B)}}$
x_1
x_2
x_3
x_1
x_2
x_3
=
$\min_{\{\lambda_{i\to\alpha}\}}$
x_1
x_1
x_2
x_2
x_3
x_3

Reason about subproblems

"Messages" adjust overlapping subproblems

Reparameterize subproblems to decrease upper bound

Dual



$$\max_{\vec{\mu} \in \mathbb{L}} \; \vec{\theta} \cdot \vec{\mu}$$

Reason about "beliefs" (marginals)

Constraints enforce overlapping beliefs are consistent

Optimum over beliefs gives upper bound

Outline

Review: Graphical Models

Decomposition Bounds

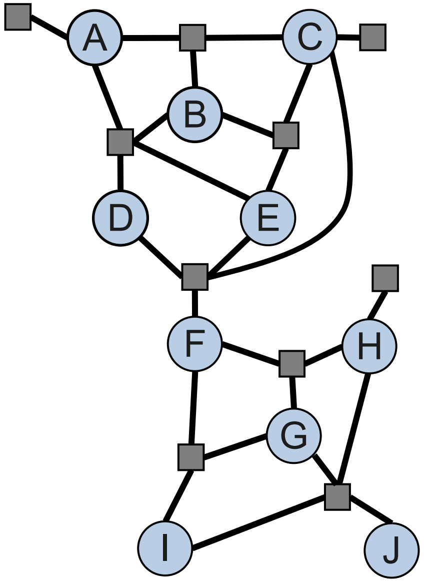
Variational Optimization

Convexity & Duality

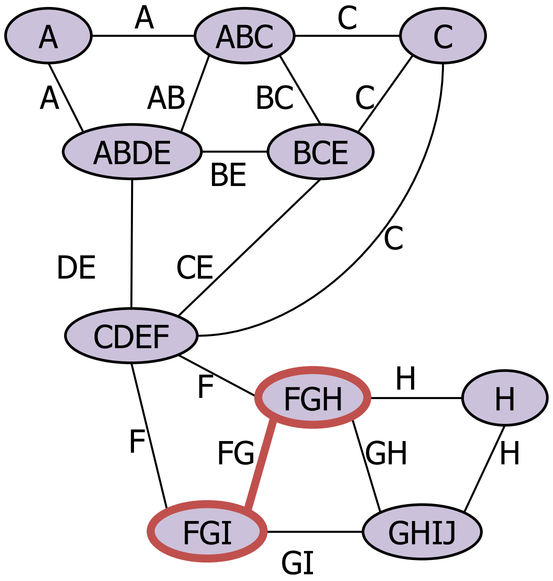
Regions & Higher-order Approximations

Regions

- Generalize local consistency enforcement



Factor graph



Dual graph

Beliefs: $\mu_{FGH}, \; \mu_{FGI}, \; \dots$

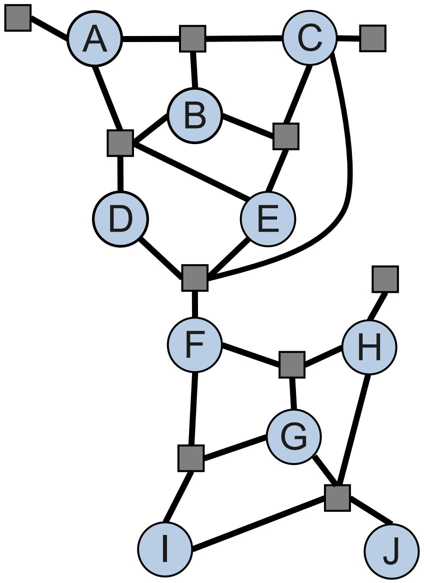
Consistency:

$$\sum_{a} \mu_{FGH}(f,g,h) = \mu_{FG}(f,g) = \sum_{i} \mu_{FGI}(f,g,i)$$

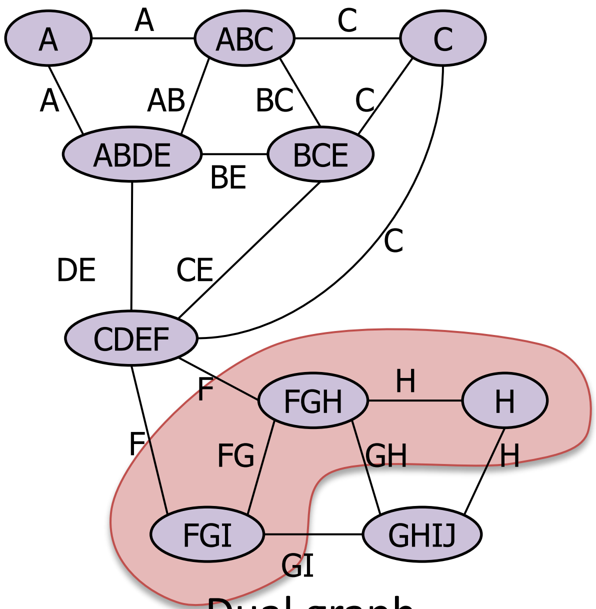
Separators = coordinates
of bound optimization (¸)

Regions

- Generalize local consistency enforcement
- Larger regions: more consistent; more costly to represent



Factor graph



Dual graph

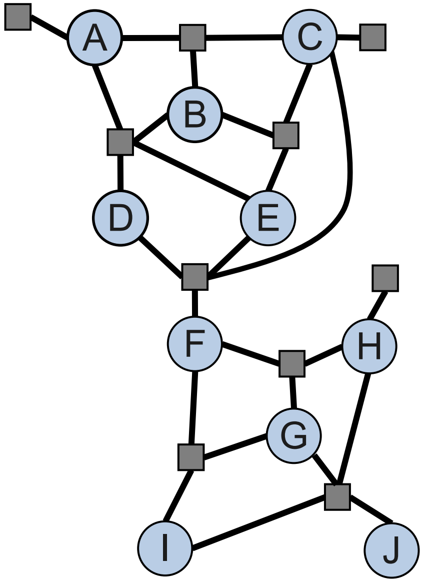
Beliefs: $\mu_{FGH}, \ \mu_{FGI}, \ \ldots$

Consistency:

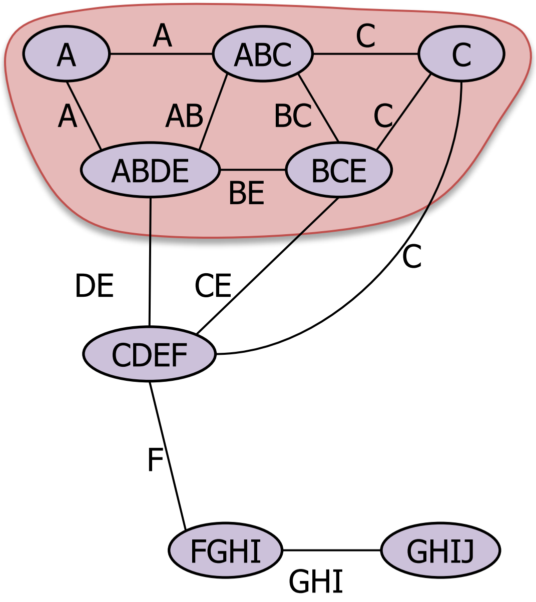
$$\sum_a \mu_{FGH}(f,g,h) = \mu_{FG}(f,g) = \sum_i \mu_{FGI}(f,g,i)$$

Regions

- Generalize local consistency enforcement
- Larger regions: more consistent; more costly to represent



Factor graph



Dual graph

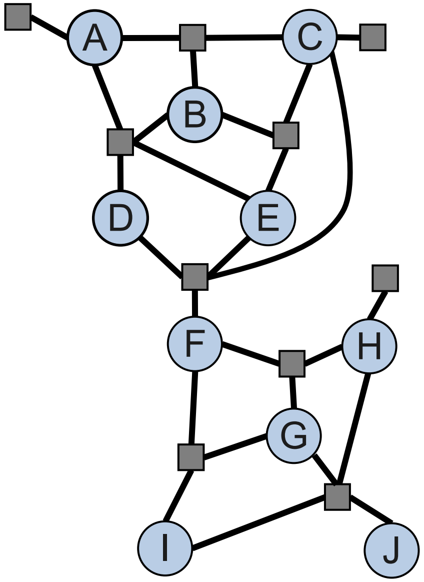
Beliefs: $\mu_{FGHI}, \ldots$

Consistency:

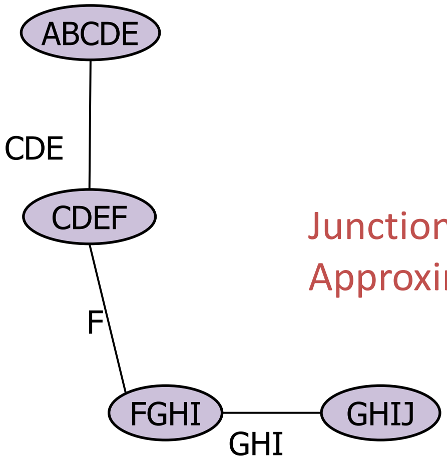
$$\sum_a \mu_{FGHI}(f,g,h,i) = \mu_{GHI}(g,h,i) = \ldots$$

Regions

- Generalize local consistency enforcement
- Larger regions: more consistent; more costly to represent



Factor graph



Dual graph

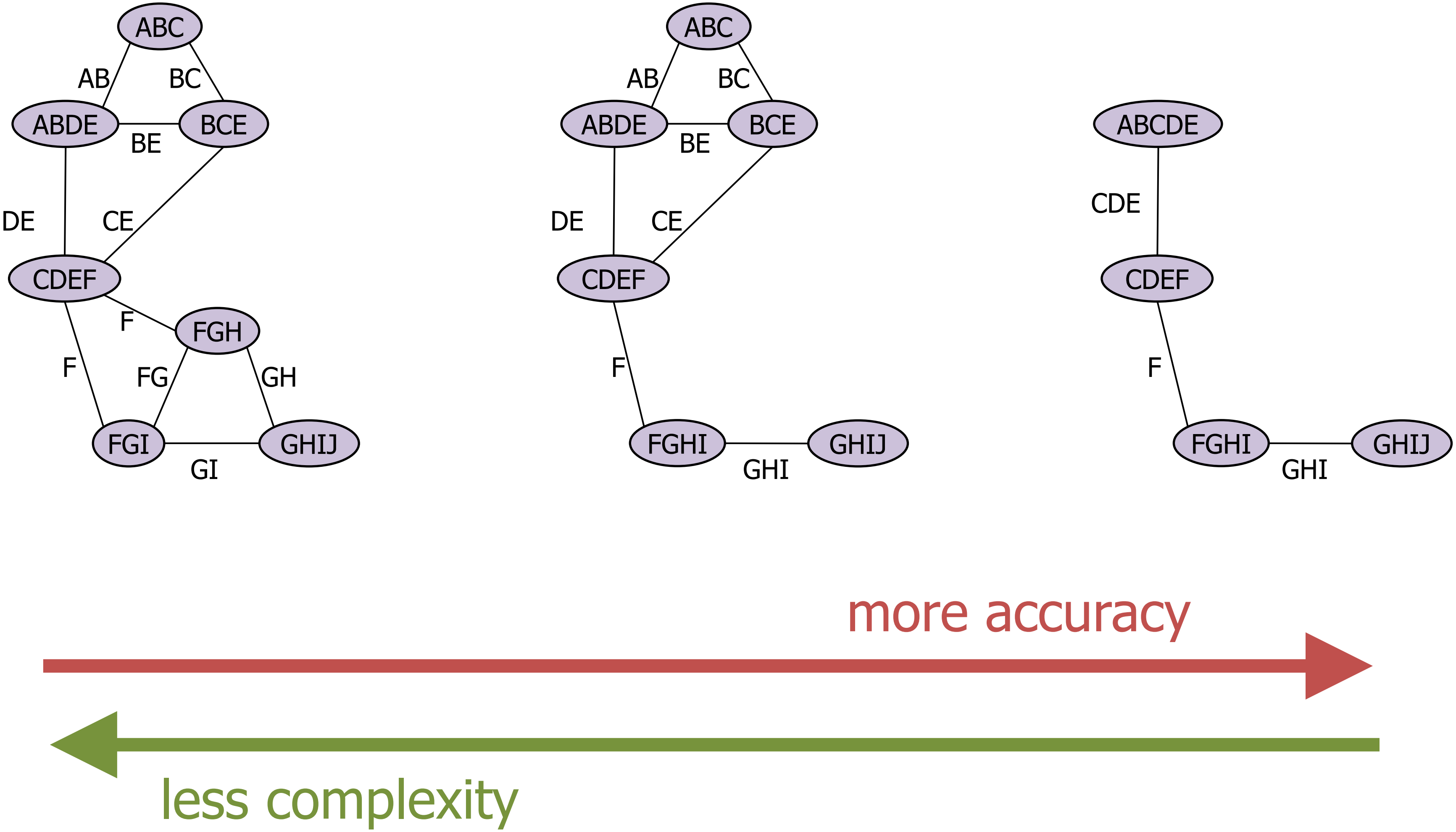
Junction tree: Approximation is exact!

Beliefs: $\mu_{FGHI}, \; \ldots$

Consistency:

$$\sum_a \mu_{FGHI}(f,g,h,i) = \mu_{GHI}(g,h,i) = \ldots$$

Regions

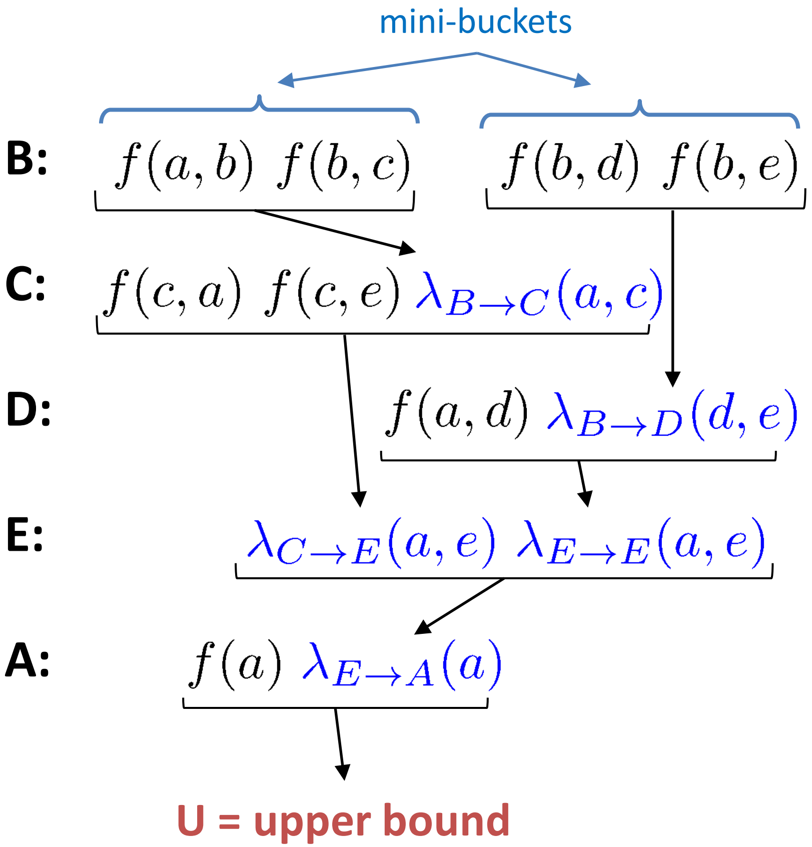


more accuracy

less complexity

Mini-bucket Regions

- Mini-bucket elimination defines regions with bounded complexity



mini-buckets

B: $f(a,b)\ f(b,c)$ $f(b,d)\ f(b,e)$

C: $f(c,a)\ f(c,e)\ \lambda_{B\to C}(a,c)$

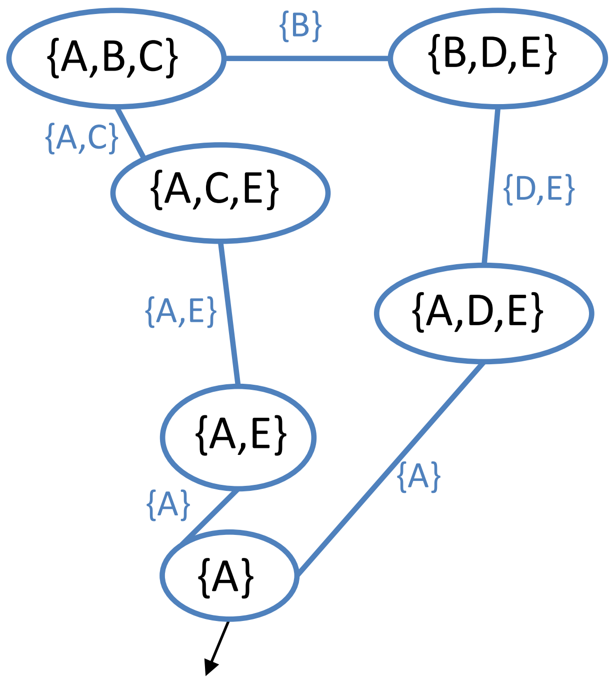
D: $f(a,d)\ \lambda_{B\to D}(d,e)$

E: $\lambda_{C\to E}(a,e)\ \lambda_{E\to E}(a,e)$

A: $f(a)\ \lambda_{E\to A}(a)$

U = upper bound

Join graph:



U = upper bound

Variational perspectives

- Replace q 2 P and H(q) with simpler approximations

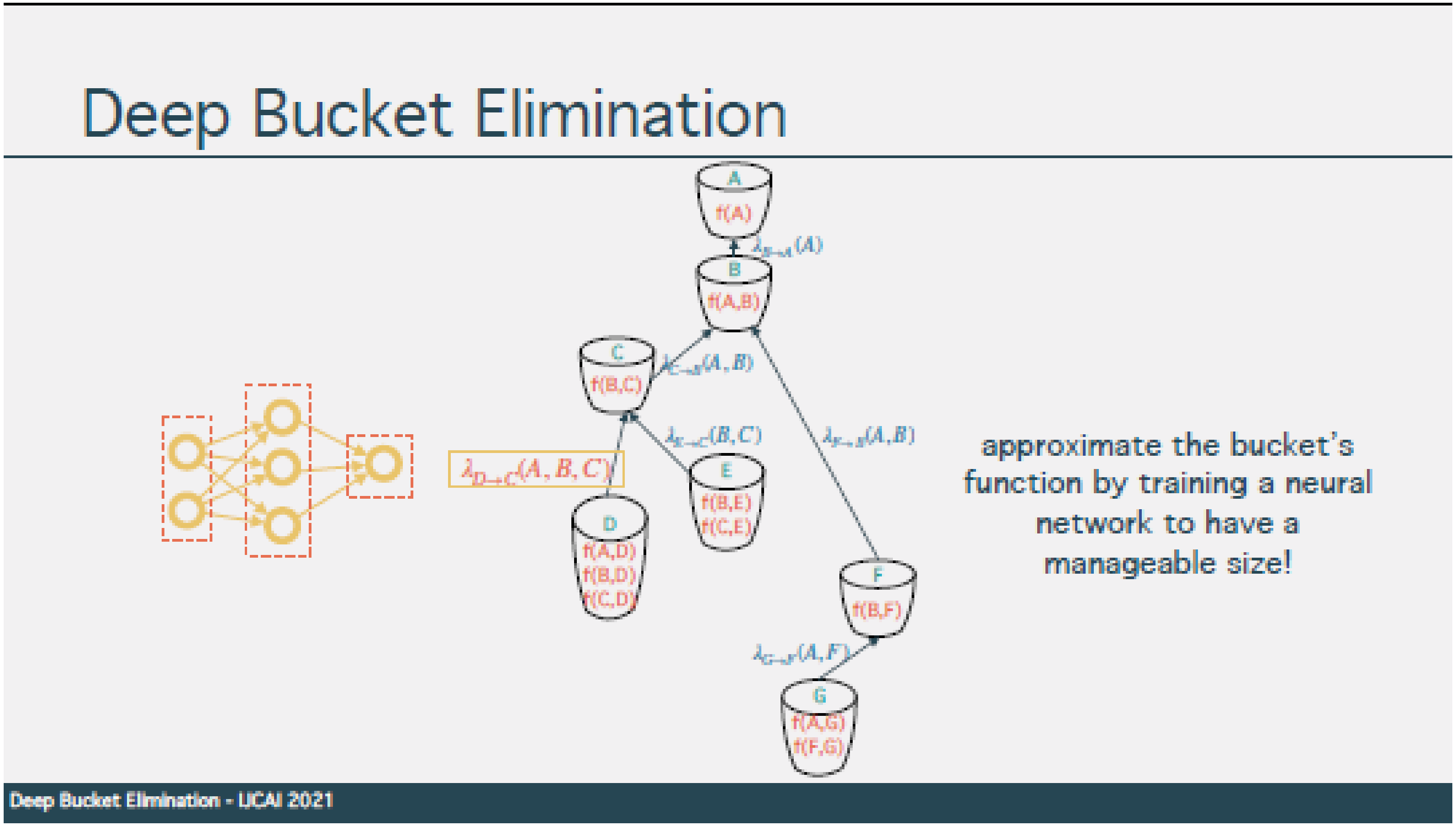
$$\log p(x^*) = \max_{q \in \mathbb{P}} \; \mathbb{E}_q[\log f(x)]$$

$$\log Z = \max_{q \in \mathbb{P}} \; \mathbb{E}_q[\log f(x)] + \boxed{H(x\,;\,q)}$$

Approximate entropy in terms of local beliefs

- Algorithms and their properties:

	Method	distributions	entropy	value
Max:	Linear programming	$q \in \mathbb{L} \supseteq \mathbb{P}$	n/a	$\hat{p}_{lp} \geq p(x^*)$
Sum:	Mean field	$\{q = \prod q_i(x_i)\} \subseteq \mathbb{P}$	exact	$Z_{mf} \leq Z$
	Belief propagation	$q \in \mathbb{L} \supseteq \mathbb{P}$	$H_\beta \approx H(q)$	$Z_\beta \approx Z$
	Tree-reweighted	$q \in \mathbb{L} \supseteq \mathbb{P}$	$H_{tr} \geq H(q)$	$Z_{tr} \geq Z$

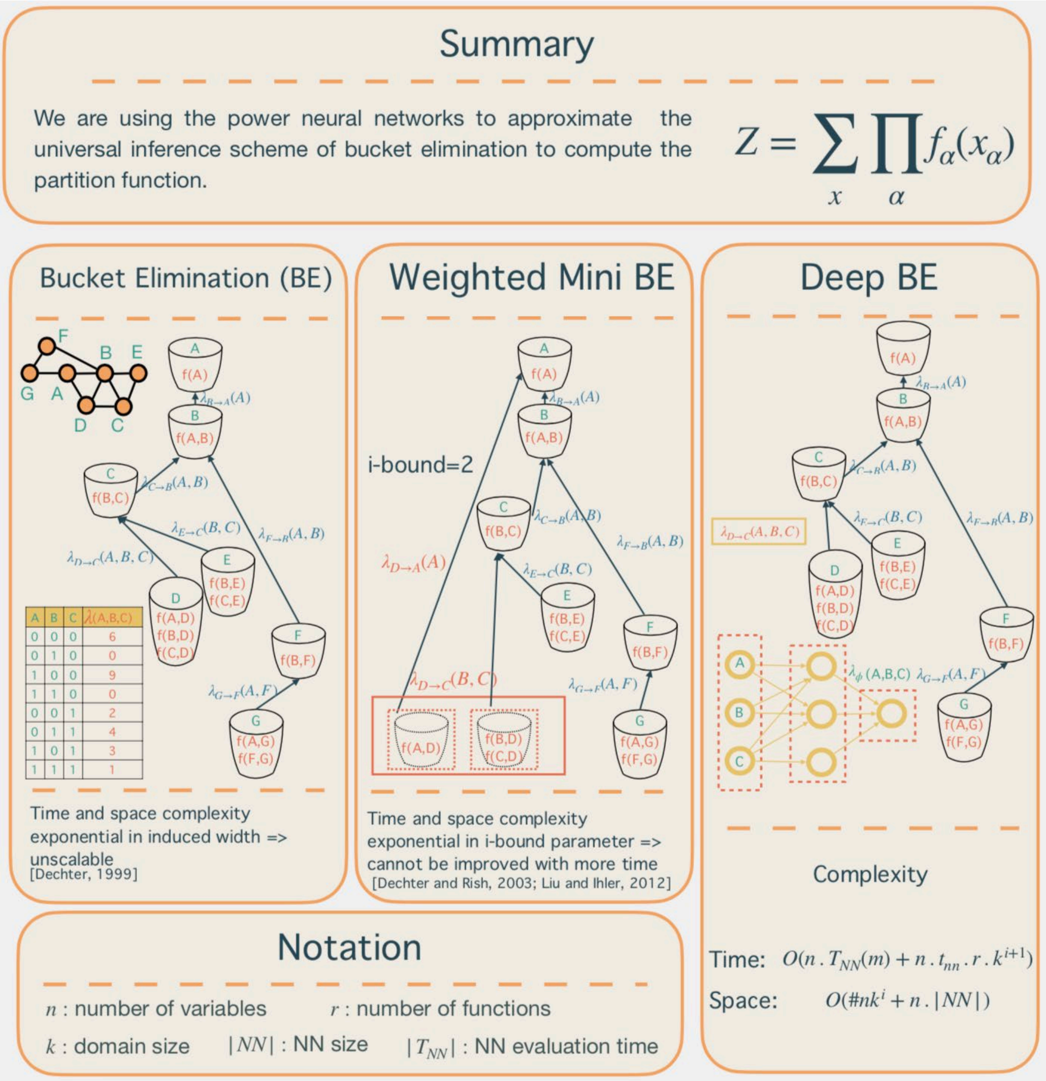


Deep Bucket Elimination

approximate the bucket's function by training a neural network to have a manageable size!

Deep Bucket Elimination - IJCAI 2021

Neuro BE



Summary

We are using the power neural networks to approximate the universal inference scheme of bucket elimination to compute the partition function.

$$Z = \sum_{x} \prod_{\alpha} f_{\alpha}(x_{\alpha})$$

Bucket Elimination (BE)

A	B	C	λ(A,B,C)
0	0	0	6
0	1	0	0
1	0	0	9
1	1	0	0
0	0	1	2
0	1	1	4
1	0	1	3
1	1	1	1

Time and space complexity exponential in induced width => unscalable
[Dechter, 1999]

Weighted Mini BE

Time and space complexity exponential in i-bound parameter => cannot be improved with more time
[Dechter and Rish, 2003; Liu and Ihler, 2012]

Deep BE

Complexity

Time: $O(n \,.\, T_{NN}(m) + n \,.\, t_{nn} \,.\, r \,.\, k^{i+1})$

Space: $O(\#nk^{i} + n \,.\, |NN|)$

Notation

n : number of variables r : number of functions

k : domain size $|NN|$: NN size $|T_{NN}|$: NN evaluation time

Summary: Variational methods

- Build approximations via an optimization perspective
  - **Primal** form: decomposition into simpler problems
  - **Dual** form: optimization over local "beliefs"

- Deterministic bounds and approximations
  - Convex upper bounds
  - Non-convex lower bounds
  - Bethe approximation & belief propagation

- Scalable, "local approximation" viewpoint
  - Optimization as local message passing

- Can improve quality through increasing region size
  - But, requires exponentially increasing memory & time, or approximation