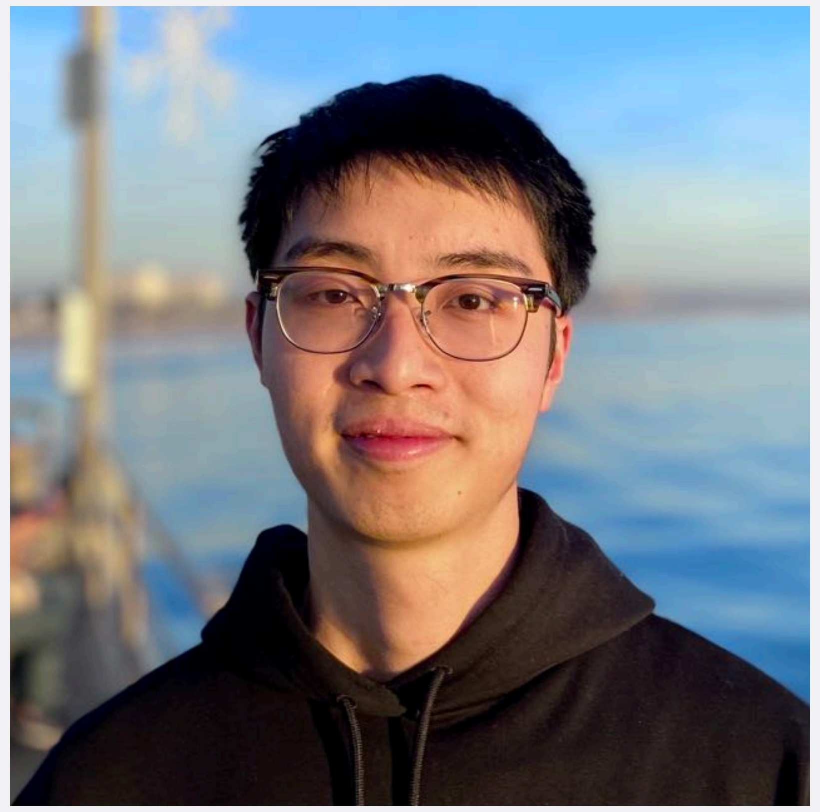
Deep Bucket Elimination






Yasaman Razeghi Kalev Kask Yadong Lu Pierre Baldi Sakshi Agarwal Rina Dechter

University of California, Irvine

International Joint Conference on Artificial Intelligence (IJCAI 2021)

Background

Graphical Models - Formal Definition

$$\mathbf{G} = \{\mathbf{X}, \mathbf{D}, \mathbf{F}\}$$

Variables: $\mathbf{X} = \{\mathbf{X_1}, \mathbf{X_2}, ..., \mathbf{X_N}\}$

Domains: $\mathbf{D} = \{\mathbf{D_{X_1}}, \mathbf{D_{X_2}}, ..., \mathbf{D_{X_N}}\}$

Factors: $\mathbf{F} = \{\mathbf{f_{\alpha_1}}, \mathbf{f_{\alpha_2}}, ..., \mathbf{f_{\alpha_M}}\}$

example

$$\mathbf{X} = \{\mathbf{A}, \mathbf{B}, \mathbf{C}\}$$

$$\mathbf{D} = \{\mathbf{D_A} = \{0, 1\}, \mathbf{D_B} = \{0, 1\}, \mathbf{D_C} = \{0, 1\}\}$$

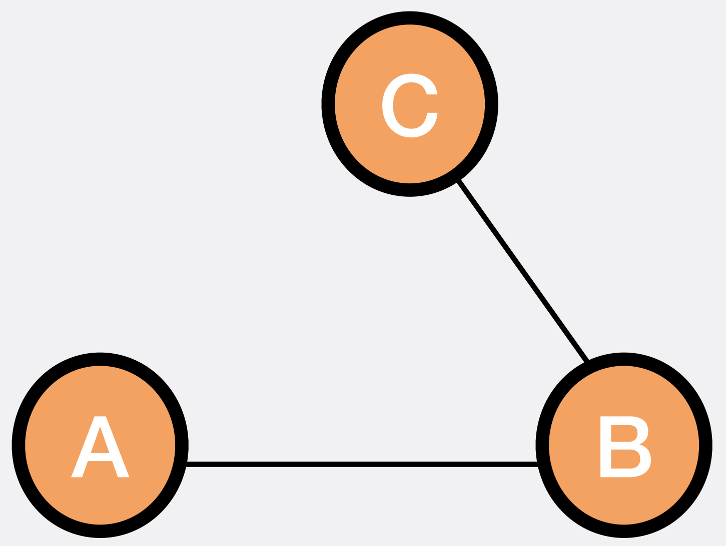
$$\mathbf{F} = \{\mathbf{f_{AB}}(\mathbf{A}, \mathbf{B}), \mathbf{f_{BC}}(\mathbf{B}, \mathbf{C})\}$$

Factors:

A	B	F(A,B)
0	0	2
0	1	4
1	0	3
1	1	1

B	C	F(B,C)
0	0	3
0	1	1
1	0	0
1	1	1

Primal Graph:



Graphical Models - Global Function

$\mathbf{X} = \{\mathbf{A}, \mathbf{B}, \mathbf{C}\}$

$\mathbf{D} = \{\mathbf{D_A} = \{0, 1\}, \mathbf{D_B} = \{0, 1\}, \mathbf{D_C} = \{0, 1\}\}$

$\mathbf{F} = \{\mathbf{f_{AB}}(\mathbf{A}, \mathbf{B}), \mathbf{f_{BC}}(\mathbf{B}, \mathbf{C})\}$

A combination operator $\otimes$ defines a global function

$$p(A, B, C) \propto f_{AB}(A, B) \times f_{BC}(B, C)$$

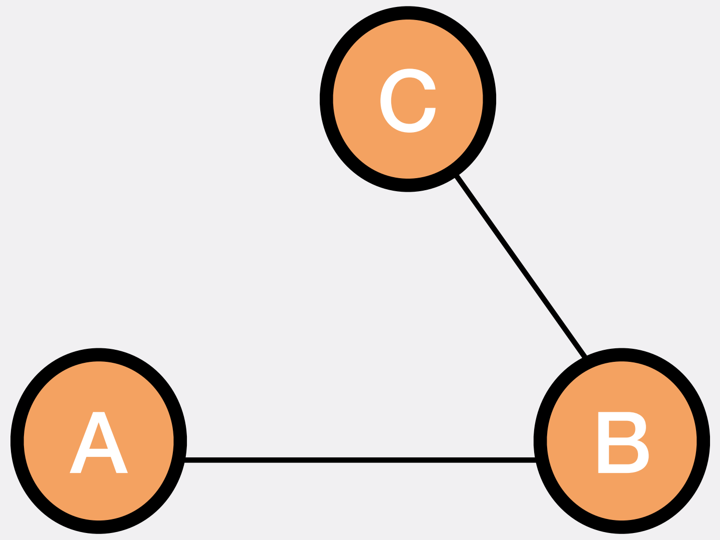
here $\otimes$ = multiplication

Factors:

A	B	F(A,B)
0	0	2
0	1	4
1	0	3
1	1	1

B	C	F(B,C)
0	0	3
0	1	1
1	0	0
1	1	1

Primal Graph:



A	B	C	P(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

The Partition Function

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

Usage in computing marginals:

$$P(X_i) = \frac{1}{Z} \sum_{X/X_i} \prod_{\alpha} f_\alpha(X_\alpha)$$

Computing is #P-complete [Cooper, 1990]

Bucket Elimination [1]

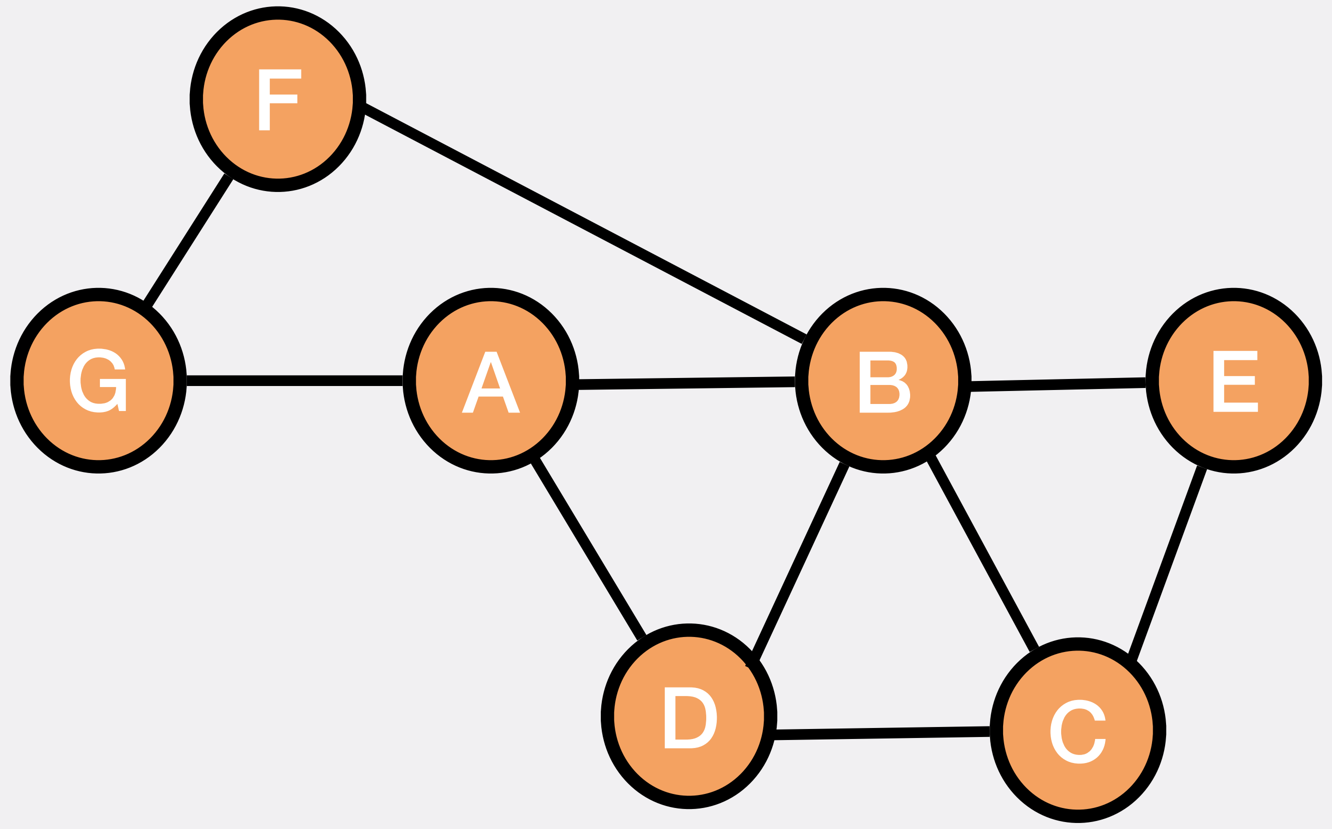
Universal inference scheme that can solve most tasks over graphical models.

Time and space exponential in the induced-width of the primal graph.

[1: Dechter, 1999]

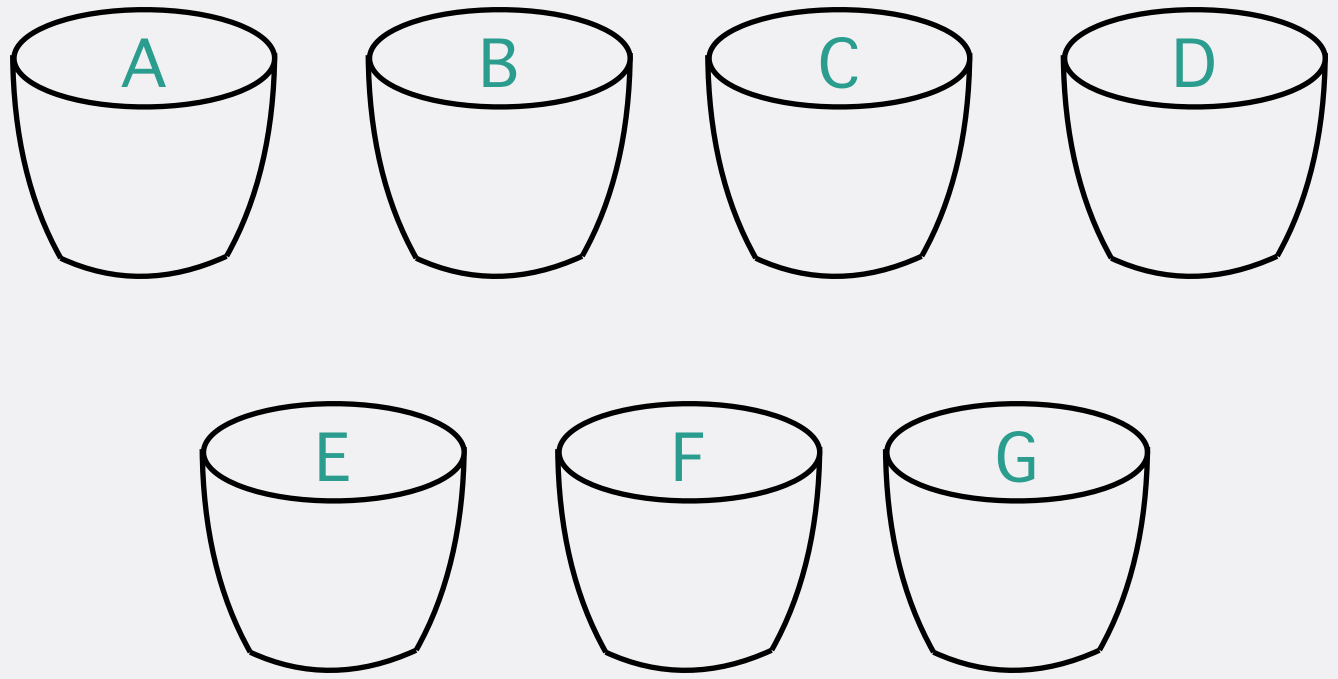
Bucket Elimination [1]

Universal inference scheme that can solve most tasks over graphical models.



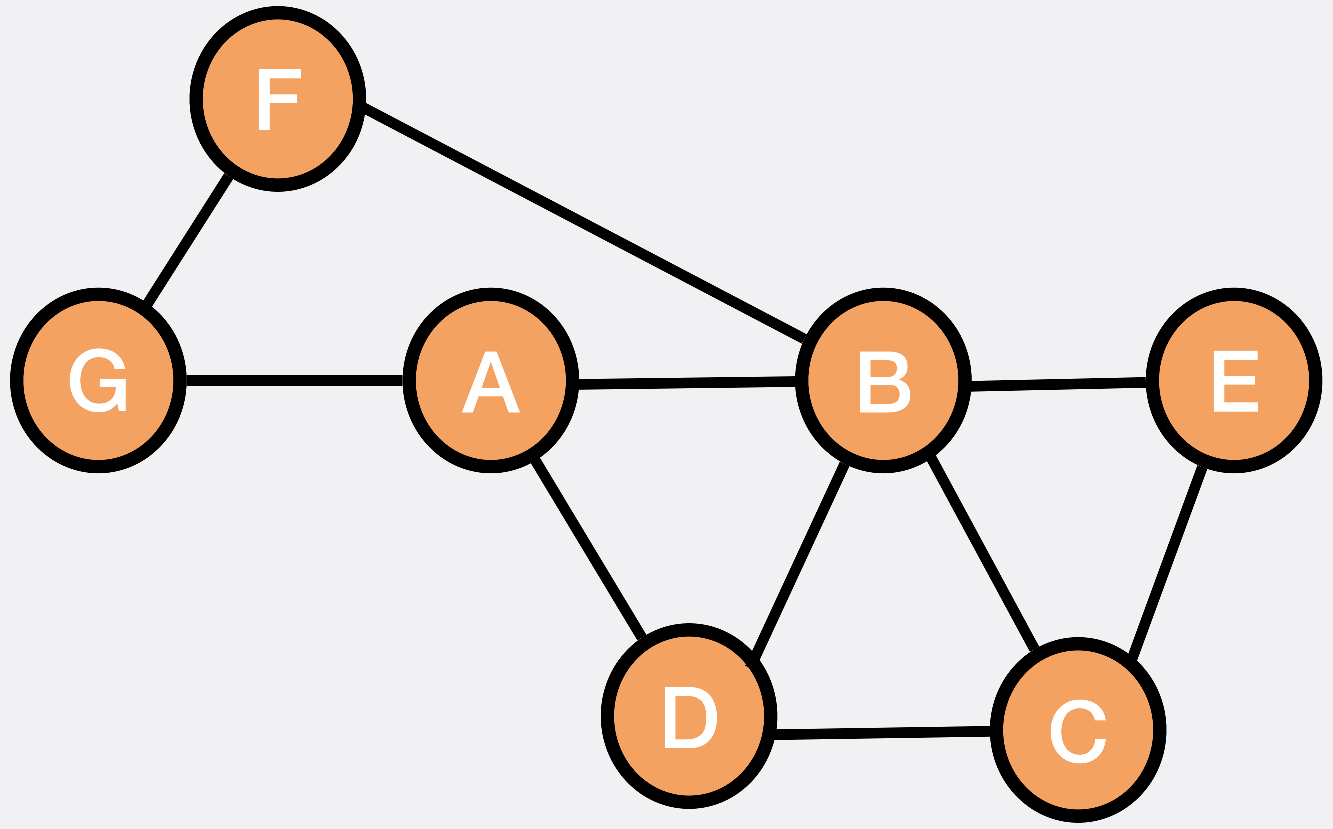
Variable ordering d: A, B, C, E, D, F, G

Factors: f(A), f(A,B), f(B,C), f(B,E), f(C,E), f(B,F),f(A,D),
f(B,D), f(C,D), f(B,E), f(C,E), f(A,G), f(F,G)



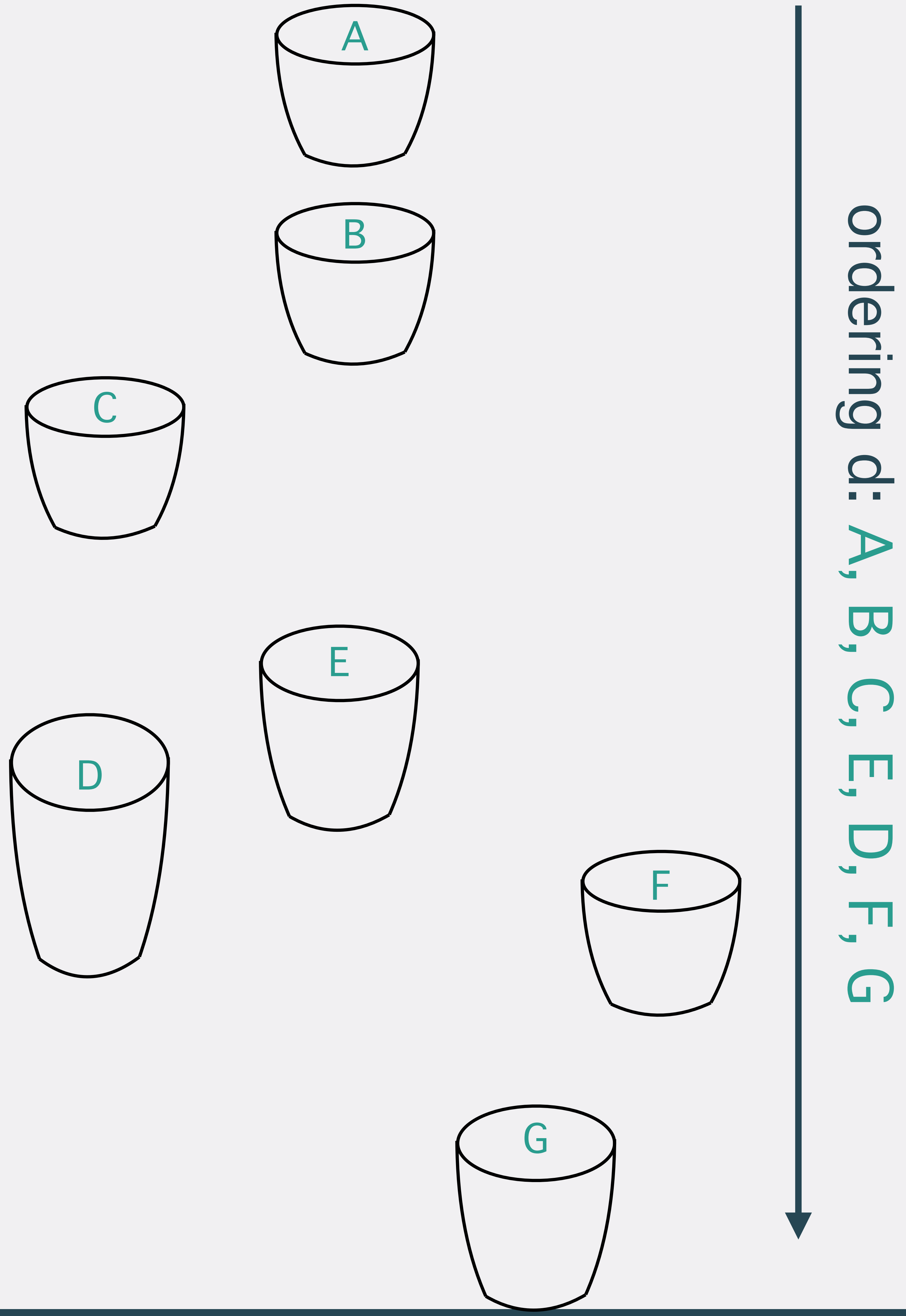
[1: Dechter, 1999]

Bucket Elimination [1]



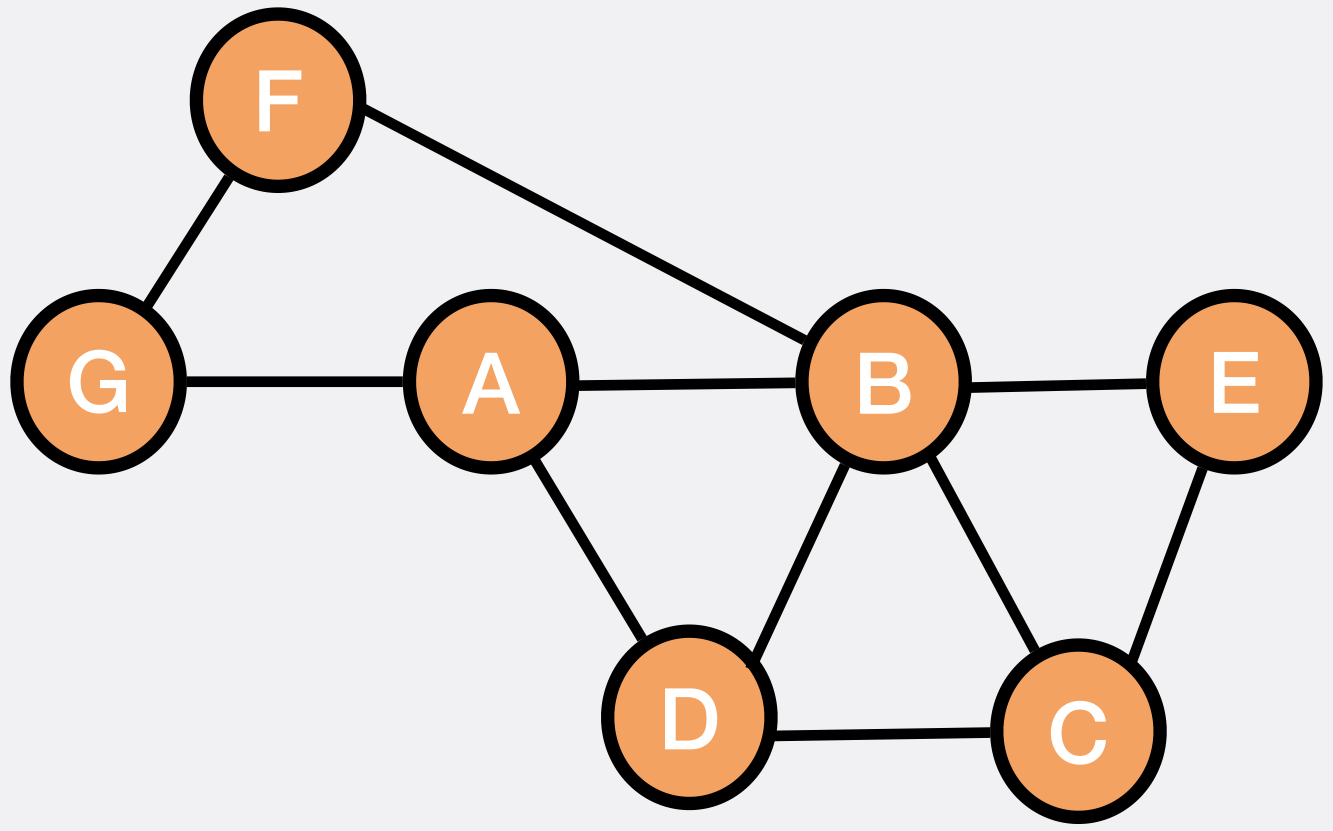
Variable ordering d: A, B, C, E, D, F, G

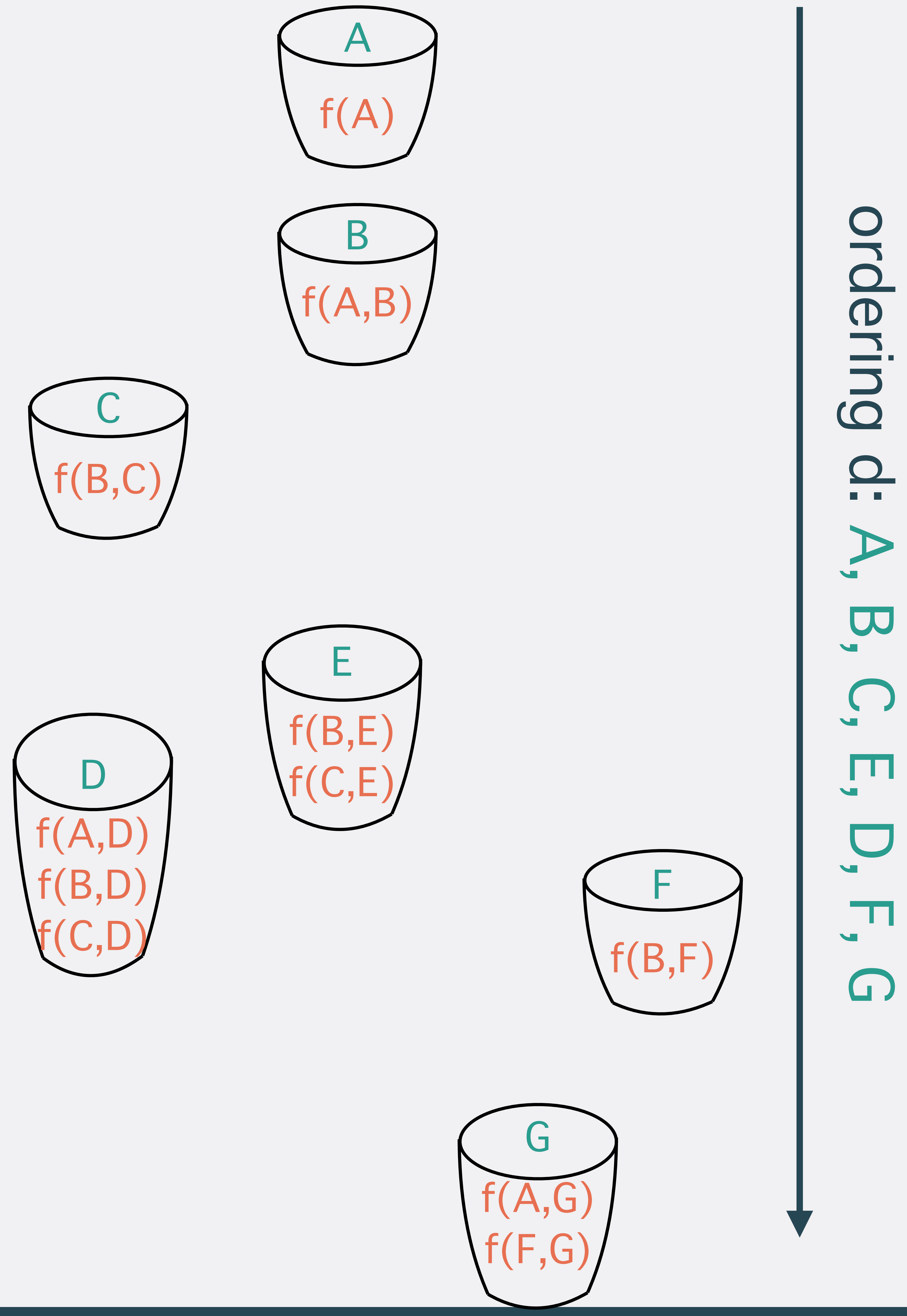
Factors: f(A), f(A,B), f(B,C), f(B,E), f(C,E), f(B,F),f(A,D), f(B,D), f(C,D), f(B,E), f(C,E), f(A,G), f(F,G)



[1:Dechter, 1999]

Bucket Elimination [1]



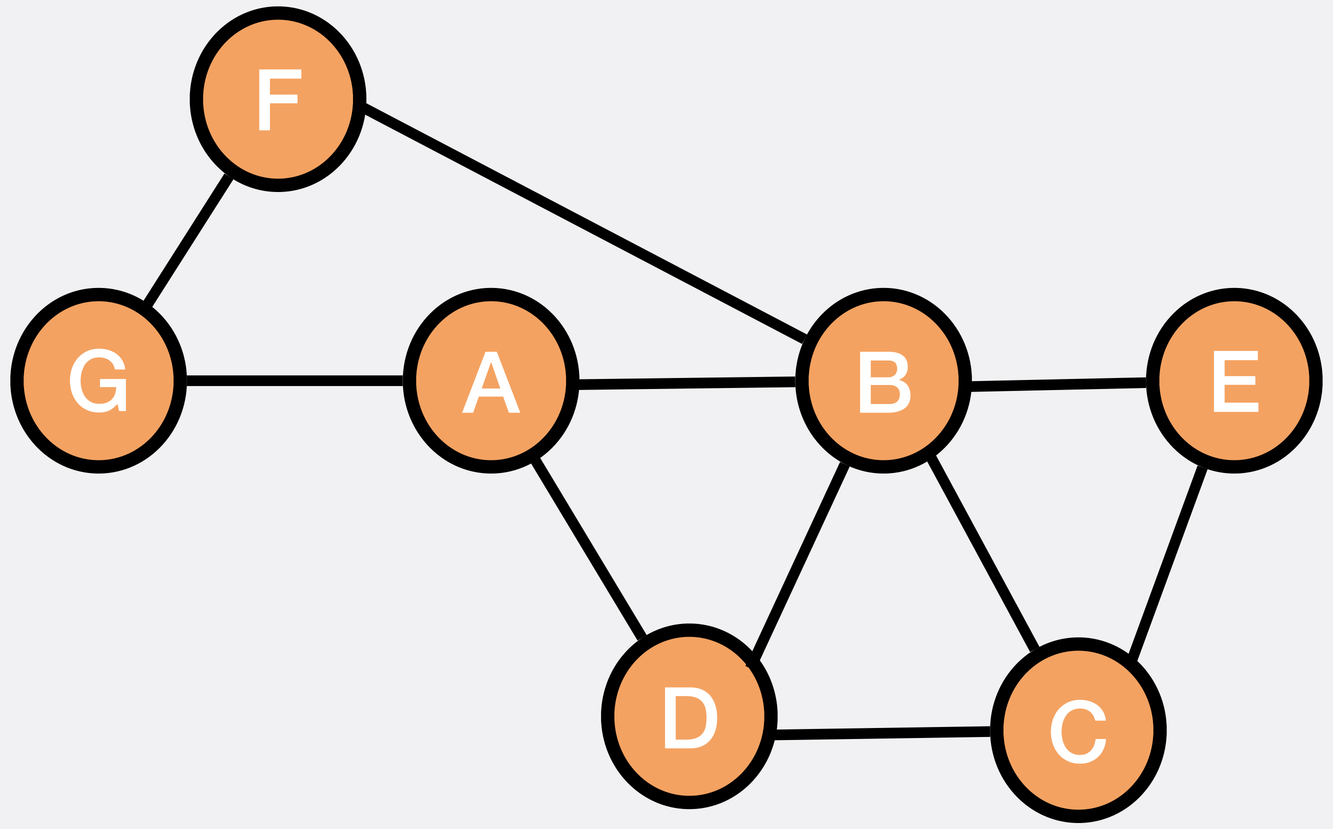


Variable ordering d: A, B, C, E, D, F, G

Factors: f(A), f(A,B), f(B,C), f(B,E), f(C,E), f(B,F),f(A,D), f(B,D), f(C,D), f(B,E), f(C,E), f(A,G), f(F,G)

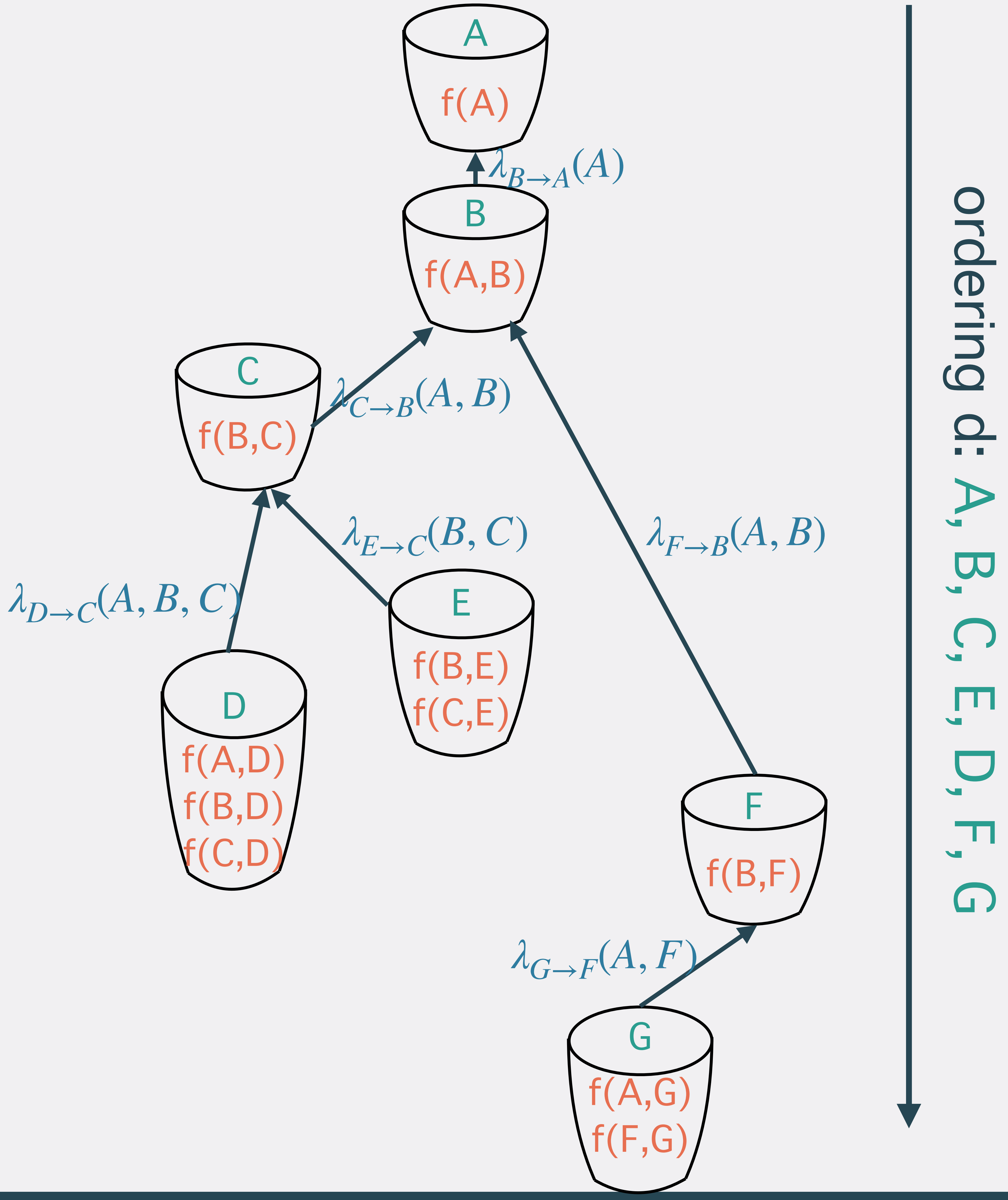
[1:Dechter, 1999]

Bucket Elimination [1]



Variable ordering d: A, B, C, E, D, F, G

Factors: f(A), f(A,B), f(B,C), f(B,E), f(C,E), f(B,F),f(A,D), f(B,D), f(C,D), f(B,E), f(C,E), f(A,G), f(F,G)



[1:Dechter, 1999]

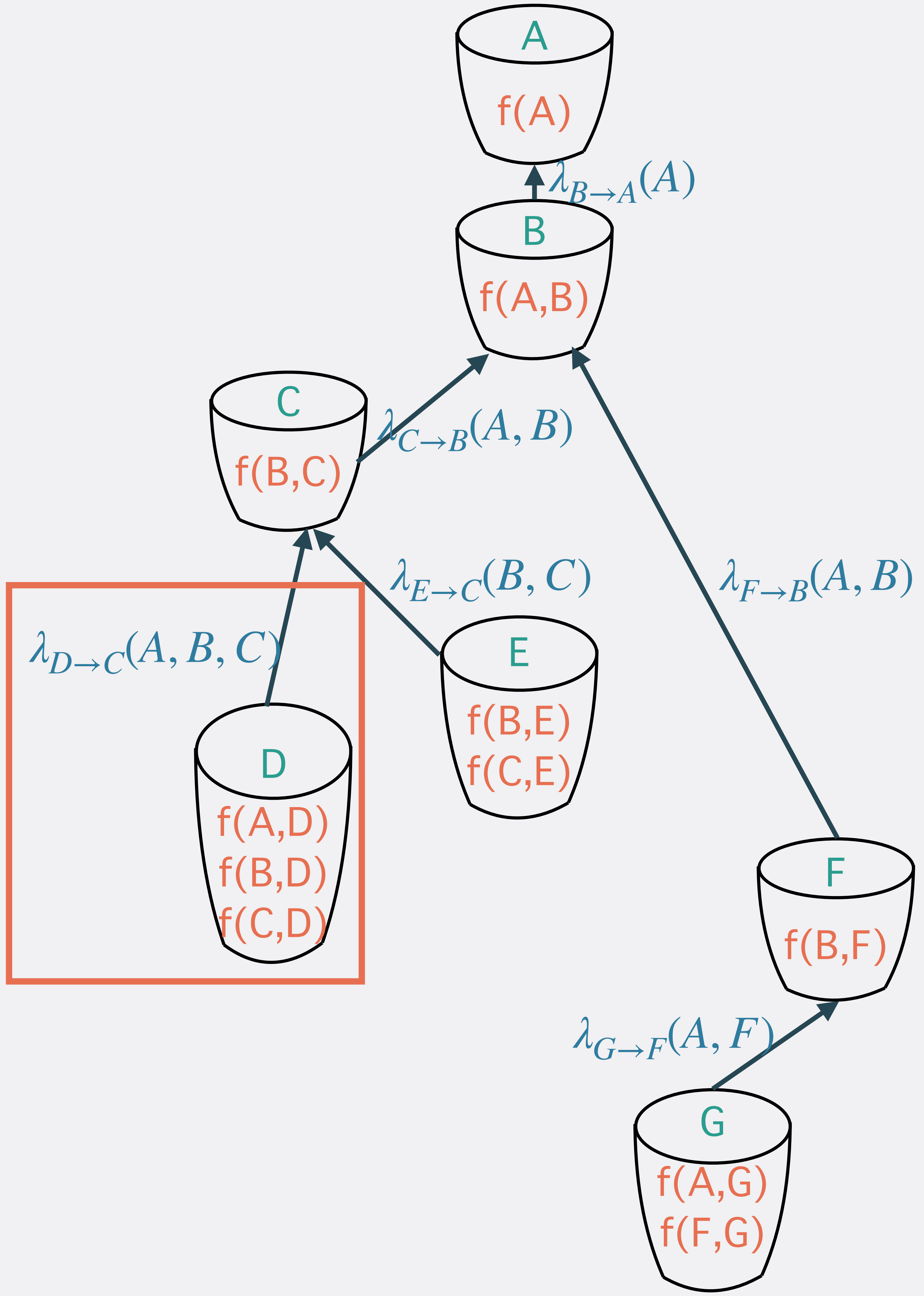
Bucket Elimination - Processing a Bucket

Processing a bucket:

$$\lambda_{(p\to a)} = \sum_{X_p} \prod_{f_\alpha \in B_i} f_\alpha$$

$$\lambda_{(D\to C)}(A,B,C) = \sum_{D} f(A,D)f(B,D)f(C,D)$$

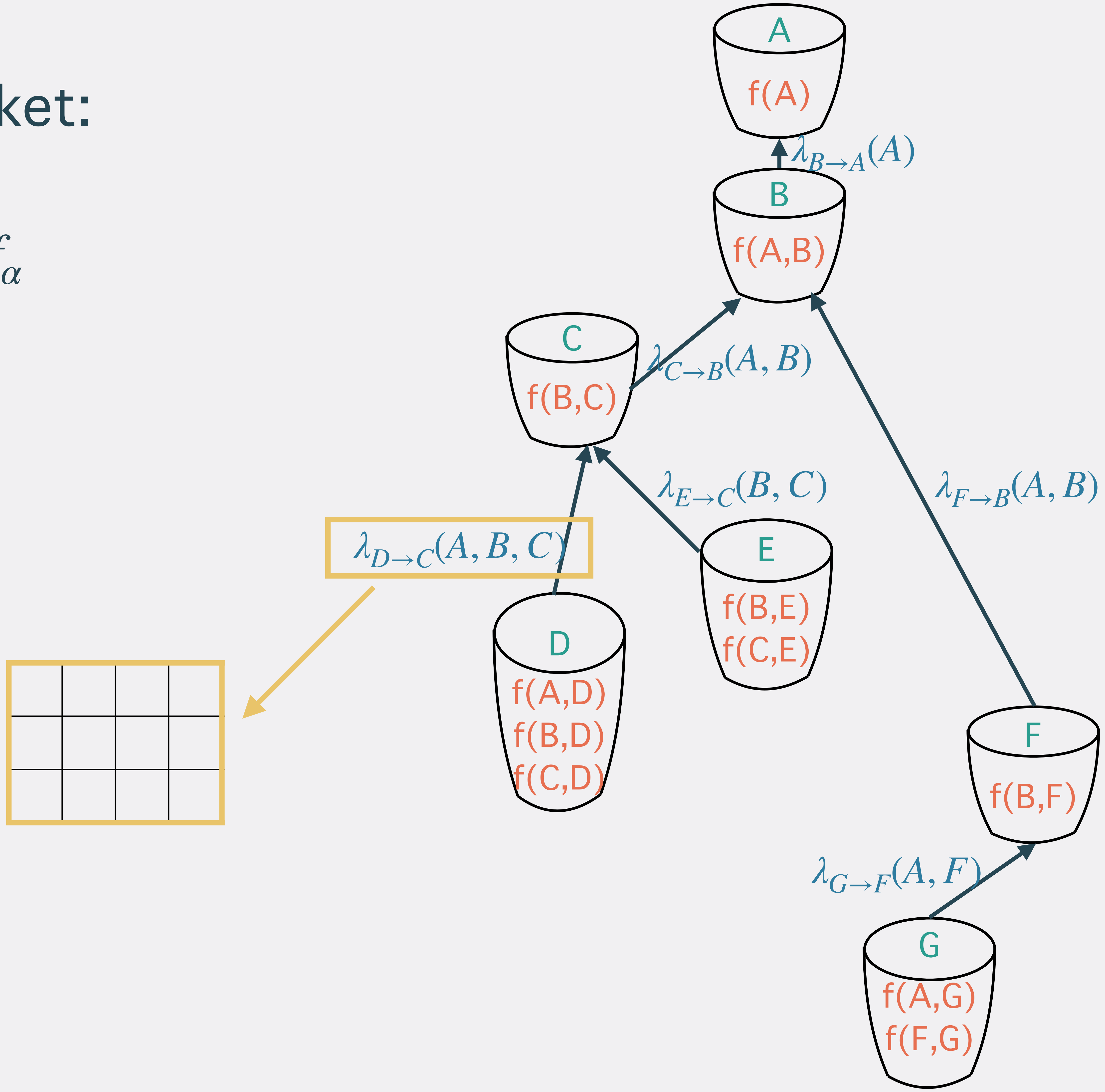
A	B	C	λ(A,B,C)
0	0	0	6
0	1	0	0
1	0	0	9
1	1	0	0
0	0	1	2
0	1	1	4
1	0	1	3
1	1	1	1



Bucket Elimination [1]

Processing a bucket:

$$\lambda_{(p\to a)} = \sum_{X_p} \prod_{f_\alpha \in B_i} f_\alpha$$

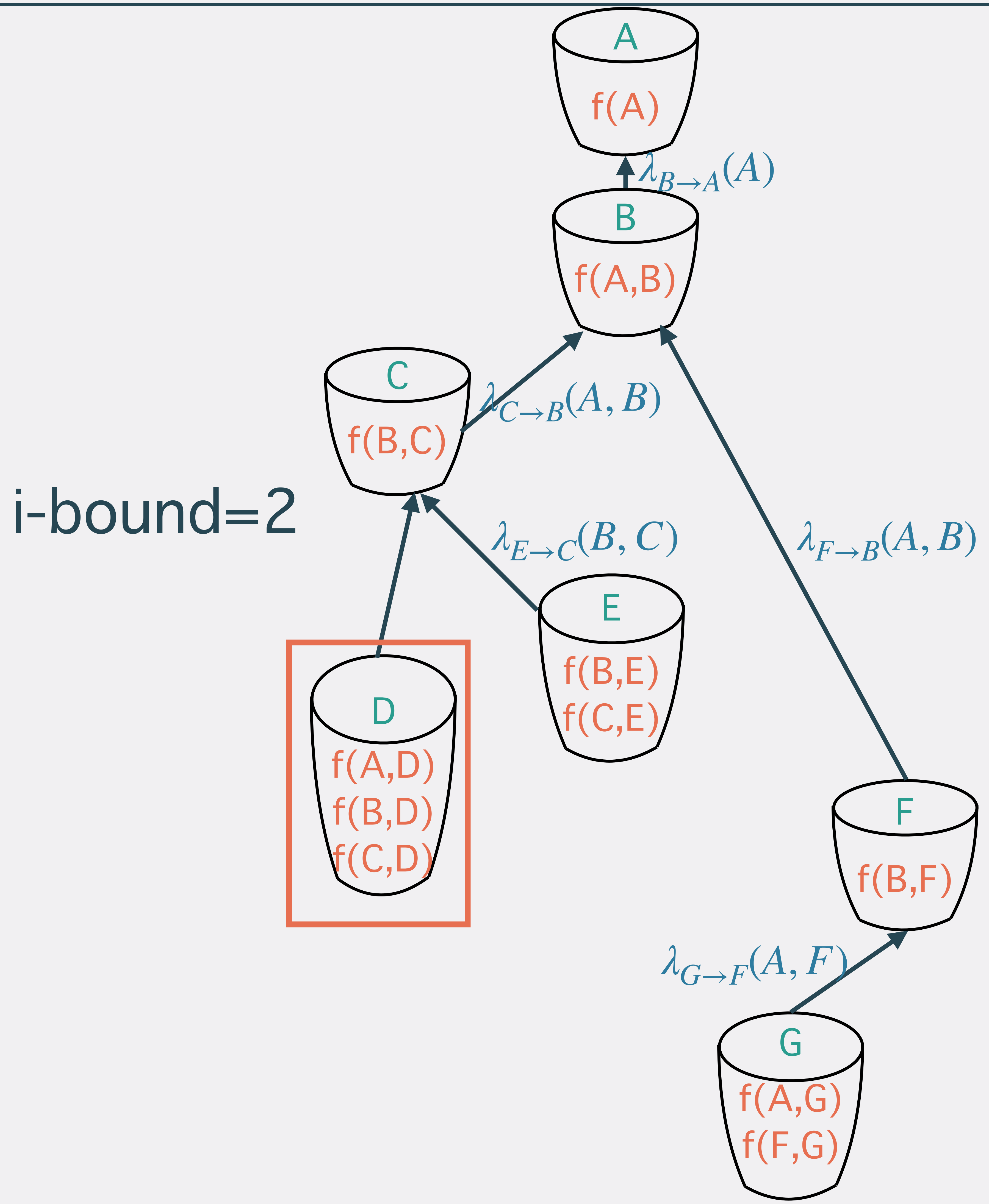


Memory is exponential in induced width!

[1: Dechter, 1999]

Weighted-MiniBucket Elimination [1,2]

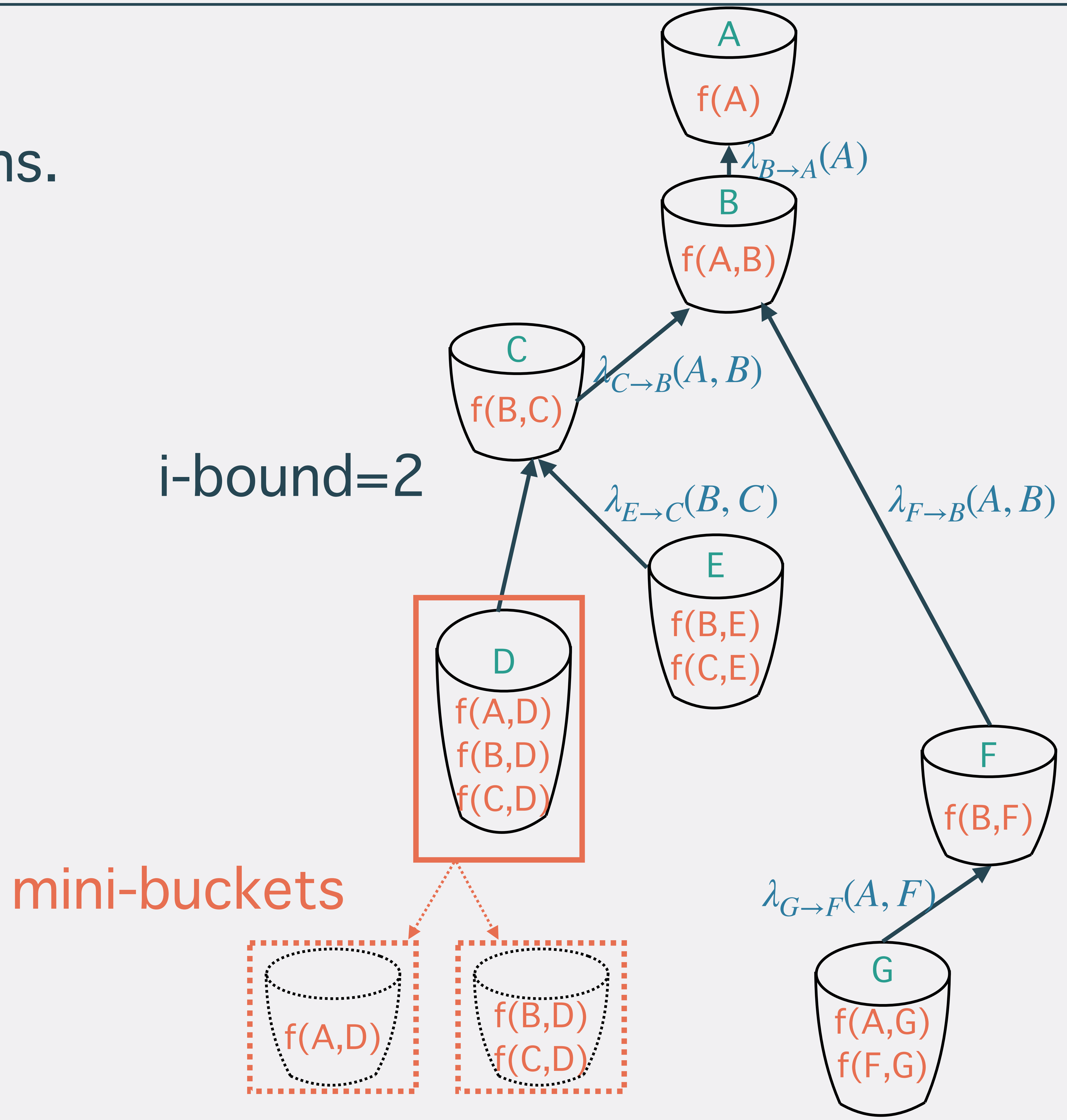
An algorithm that approximates the bucket functions.



[1: Dechter and Rish, 2003]
[2: Liu and Ihler, 2012]

Weighted-MiniBucket Elimination [1,2]

An algorithm that approximates the bucket functions.



[1: Dechter and Rish, 2003]
[2: Liu and Ihler, 2012]

Weighted-MiniBucket Elimination [1,2]

An algorithm that approximates the bucket functions.

Processing a bucket:

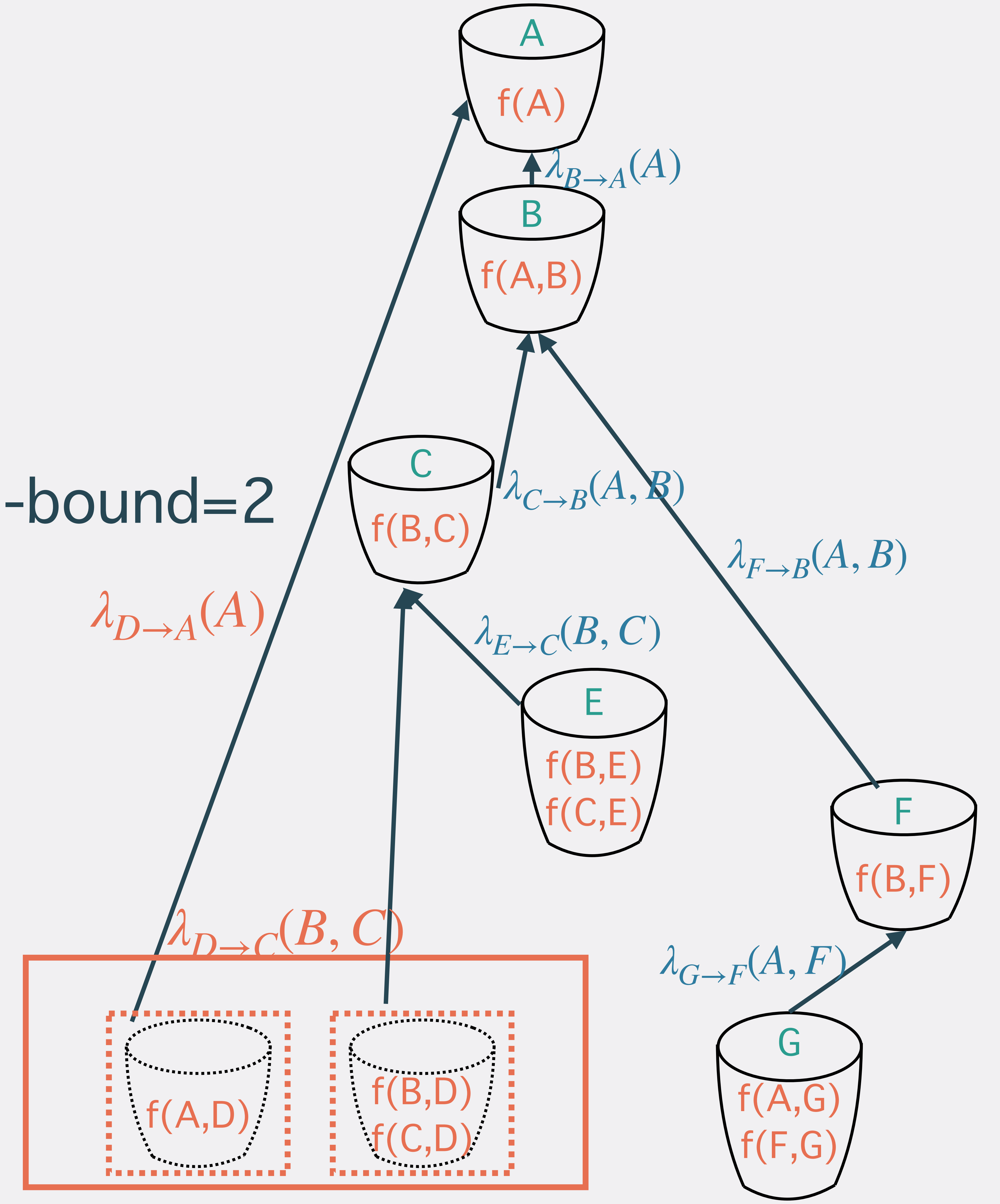
$$\lambda_{(p\to a)} = \sum_{X_p}\prod_{f_\alpha\in B_i} f_\alpha \simeq [\sum_{X_p}\prod_{f_\alpha\in B_{i1}} f_\alpha^{\frac{1}{w_1}}]^{w_1}\cdot[\sum_{X_p}\prod_{f_\alpha\in B_{i2}} f_\alpha^{\frac{1}{w_2}}]^{w_2}$$

$$w_1 + w_2 = 1$$

$$\lambda_{(D\to A)}(A) = [\sum_D f(A,D)^{\frac{1}{w_1}}]^{w_1}$$

$$\lambda_{(D\to C)}(A) = [\sum_D f(B,D)f(C,D)^{\frac{1}{w2}}]^{w2}$$

i-bound=2



[1: Dechter and Rish, 2003]
[2: Liu and Ihler, 2012]

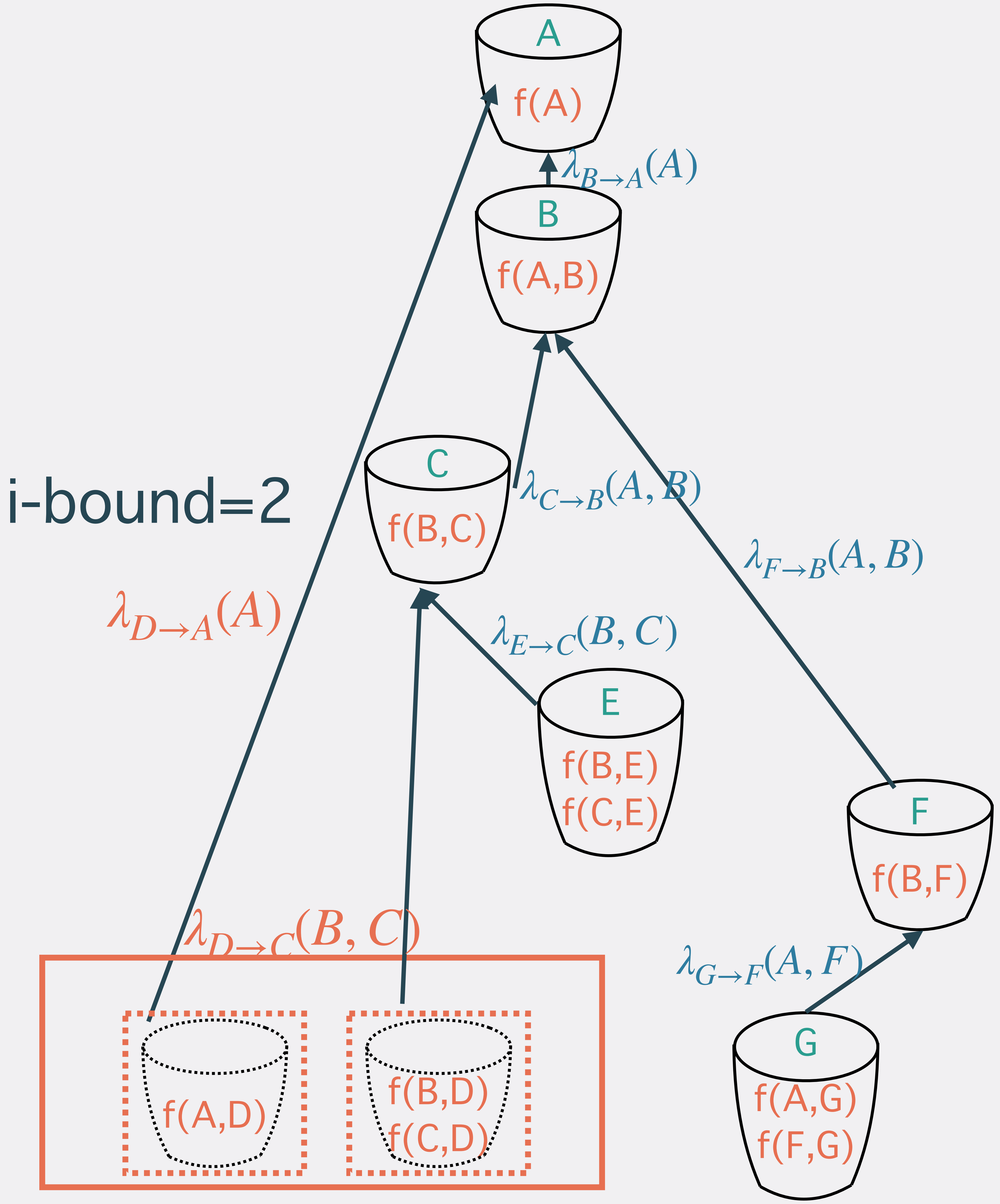
Weighted-MiniBucket Elimination [1,2]

An algorithm that approximates the bucket functions.

Generates upper bound on the partition function.

Time and space exponential in the i-bound.

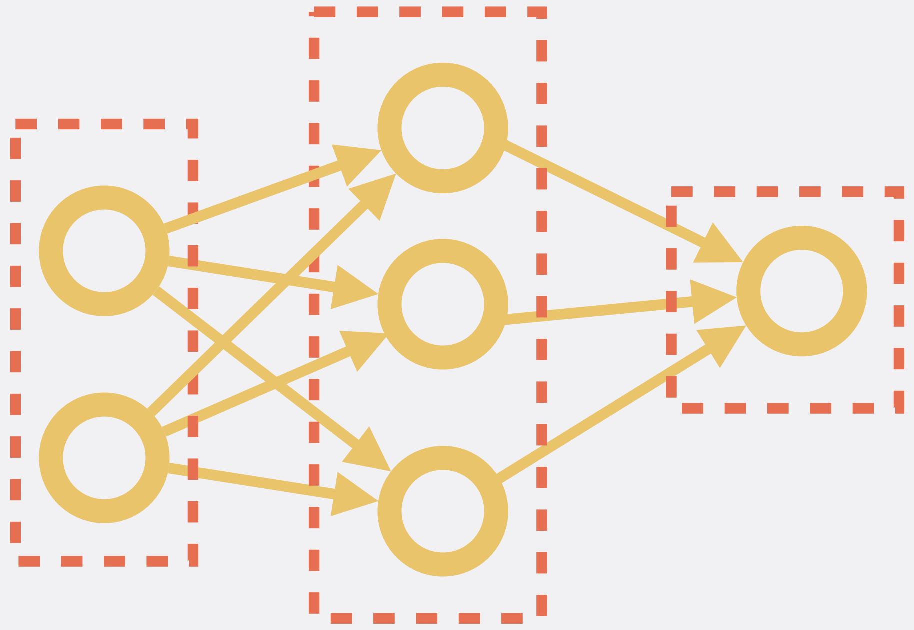
Can not improve with more time!

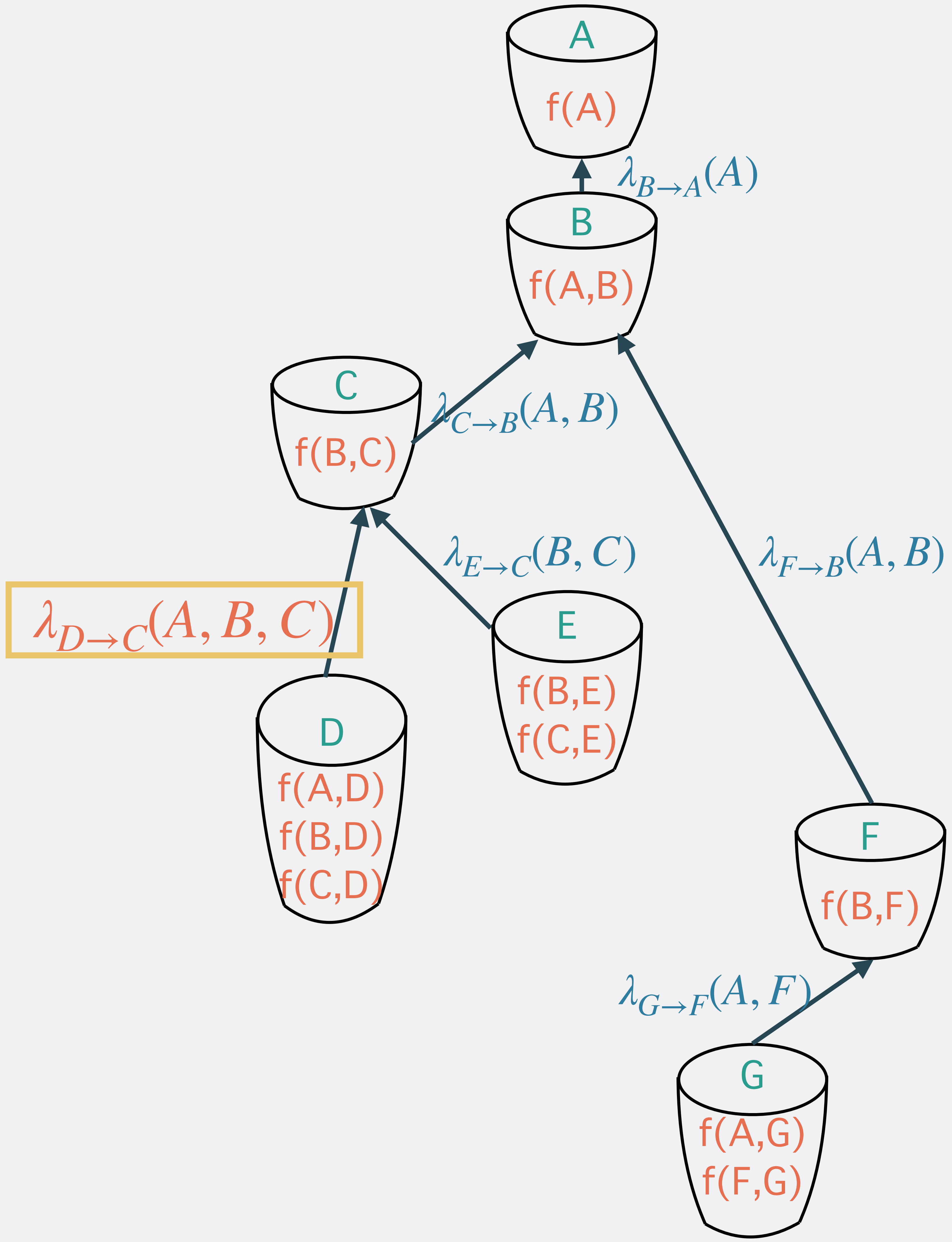


[1: Dechter and Rish, 2003]
[2: Liu and Ihler, 2012]

Deep Bucket Elimination

Deep Bucket Elimination

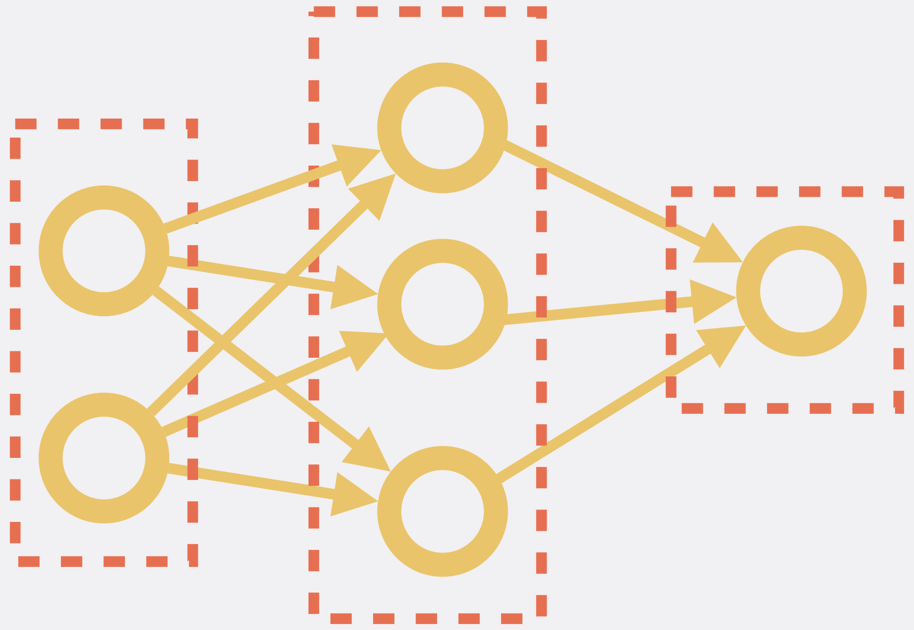


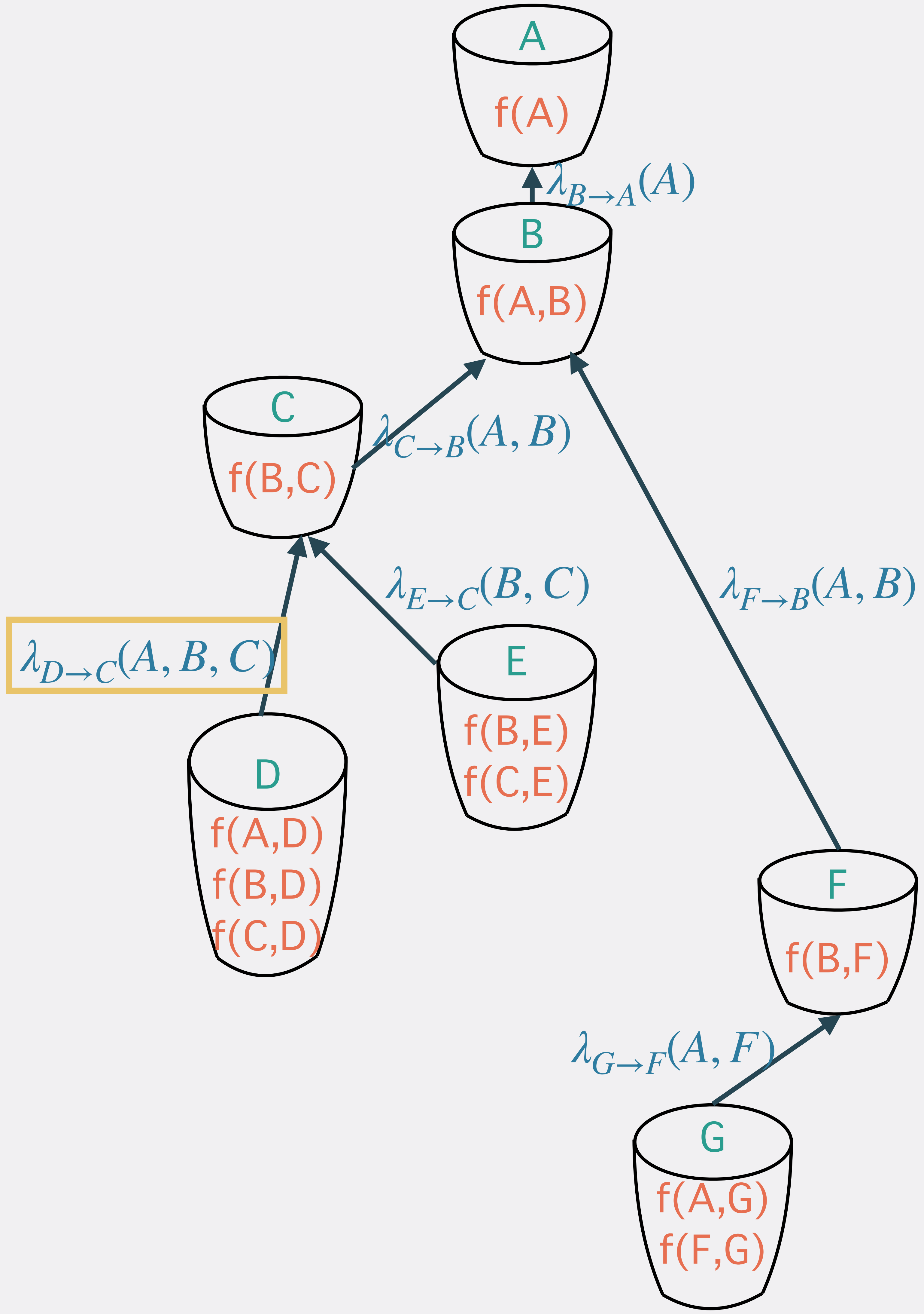


approximate the bucket's function by training a neural network to have a manageable size!

Deep Bucket Elimination

if width > i-bound =2

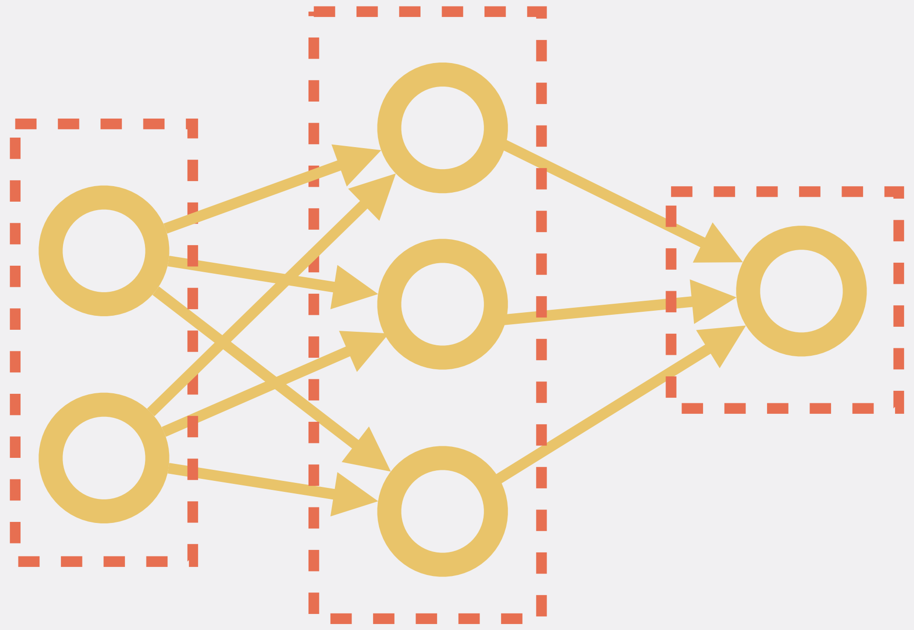


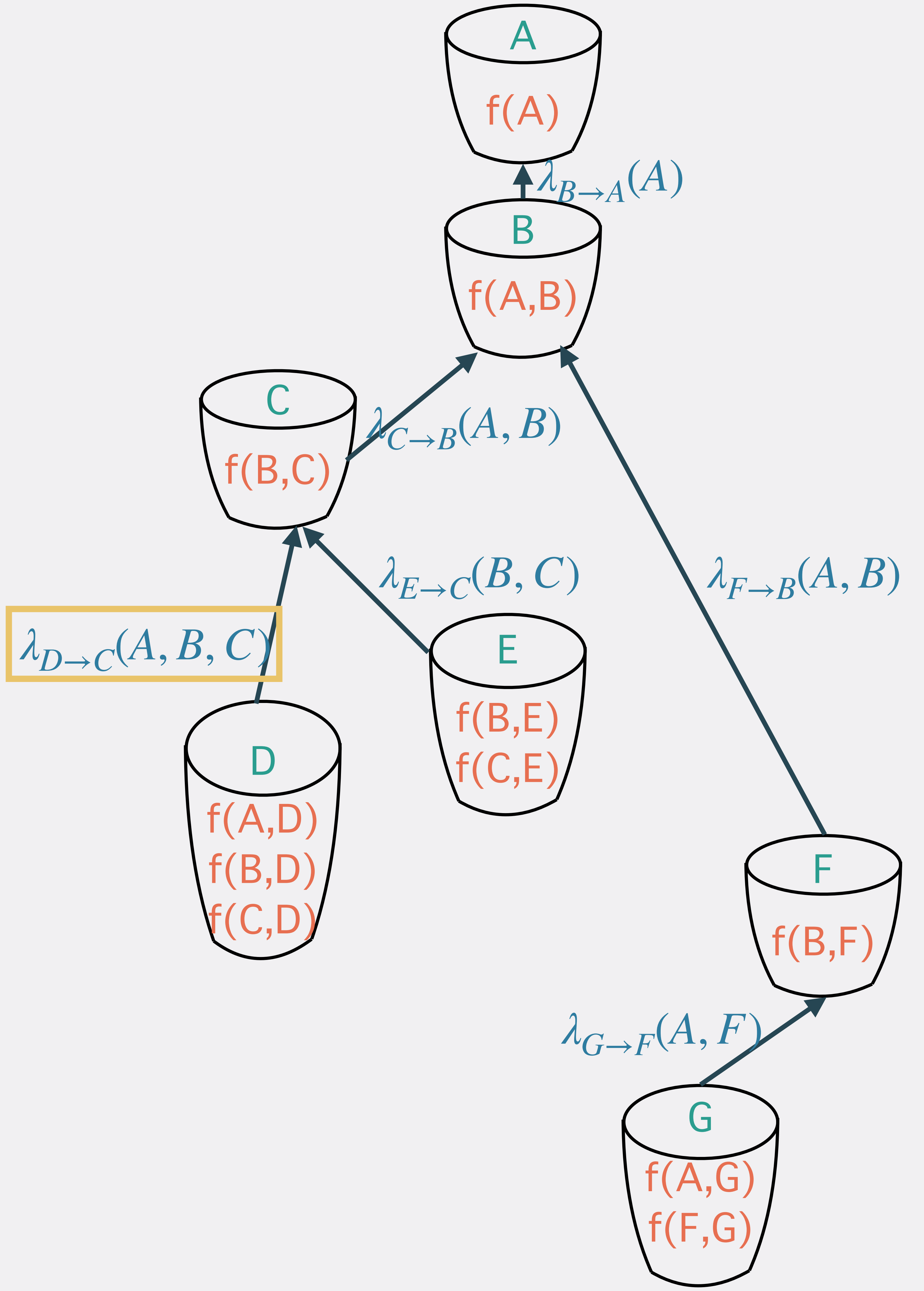


Deep Bucket Elimination

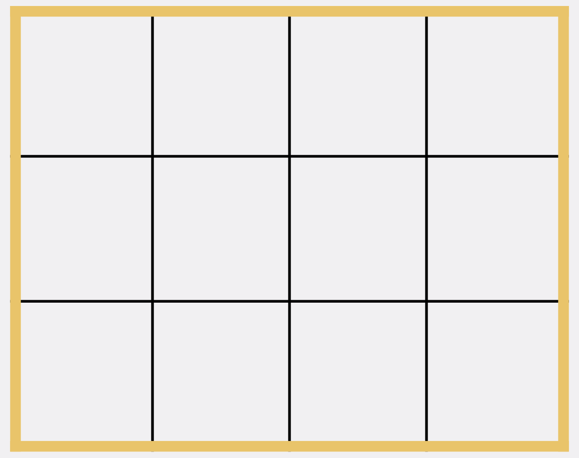
i-bound = 2

if width > i-bound





if width <= i-bound



Training a Neural Networks for Bucket Functions

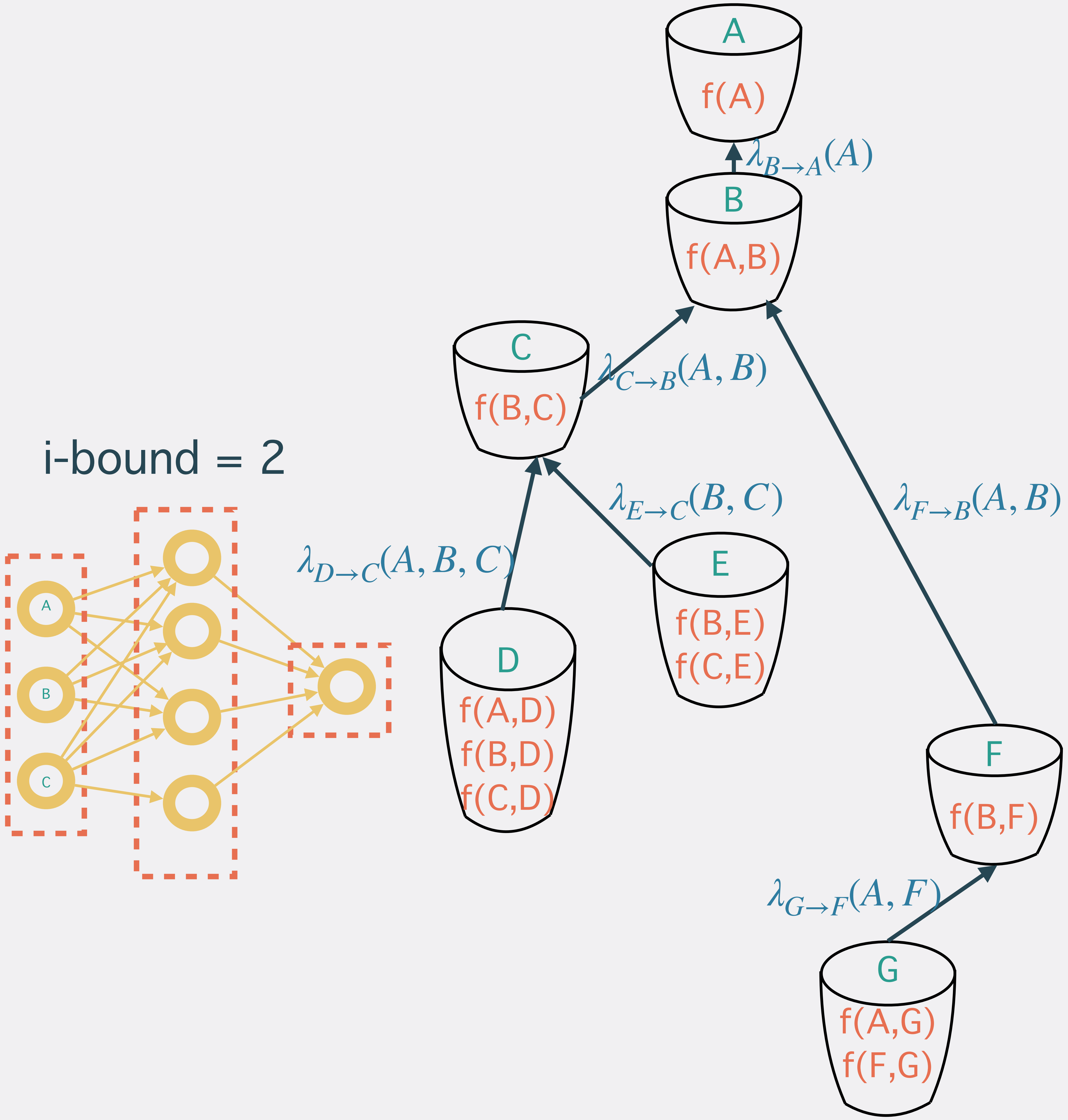
1. Learning the functions as neural networks
    1. What is the appropriate architecture?
    2. How to train the neural networks?
2. Samples
    1. How many samples we need for learning the messages?
    2. How to generate the samples?

Generating the Samples

1. Fixed number of samples

A	B	C	λ(A,B,C)

4

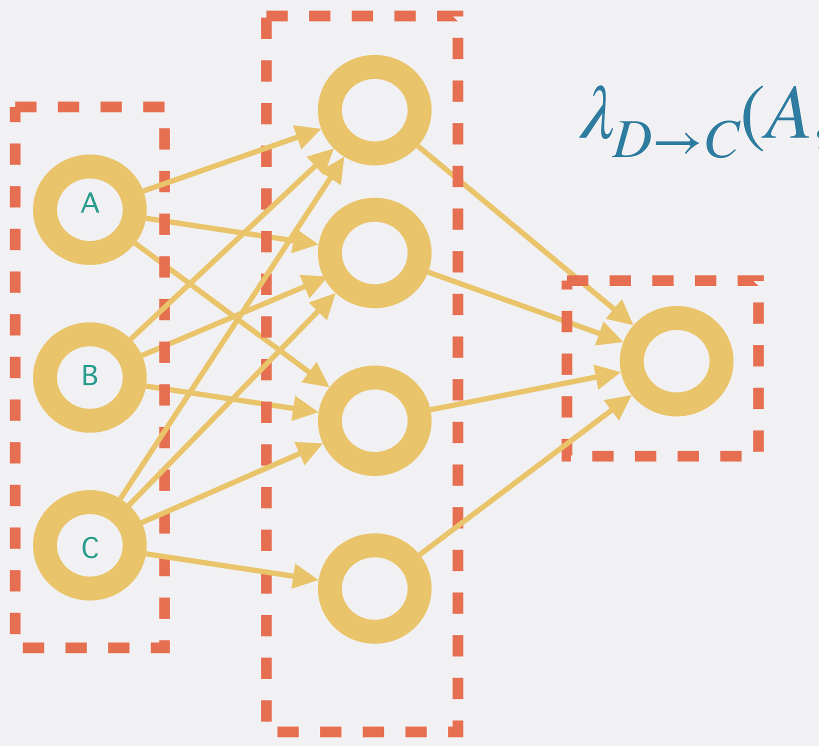


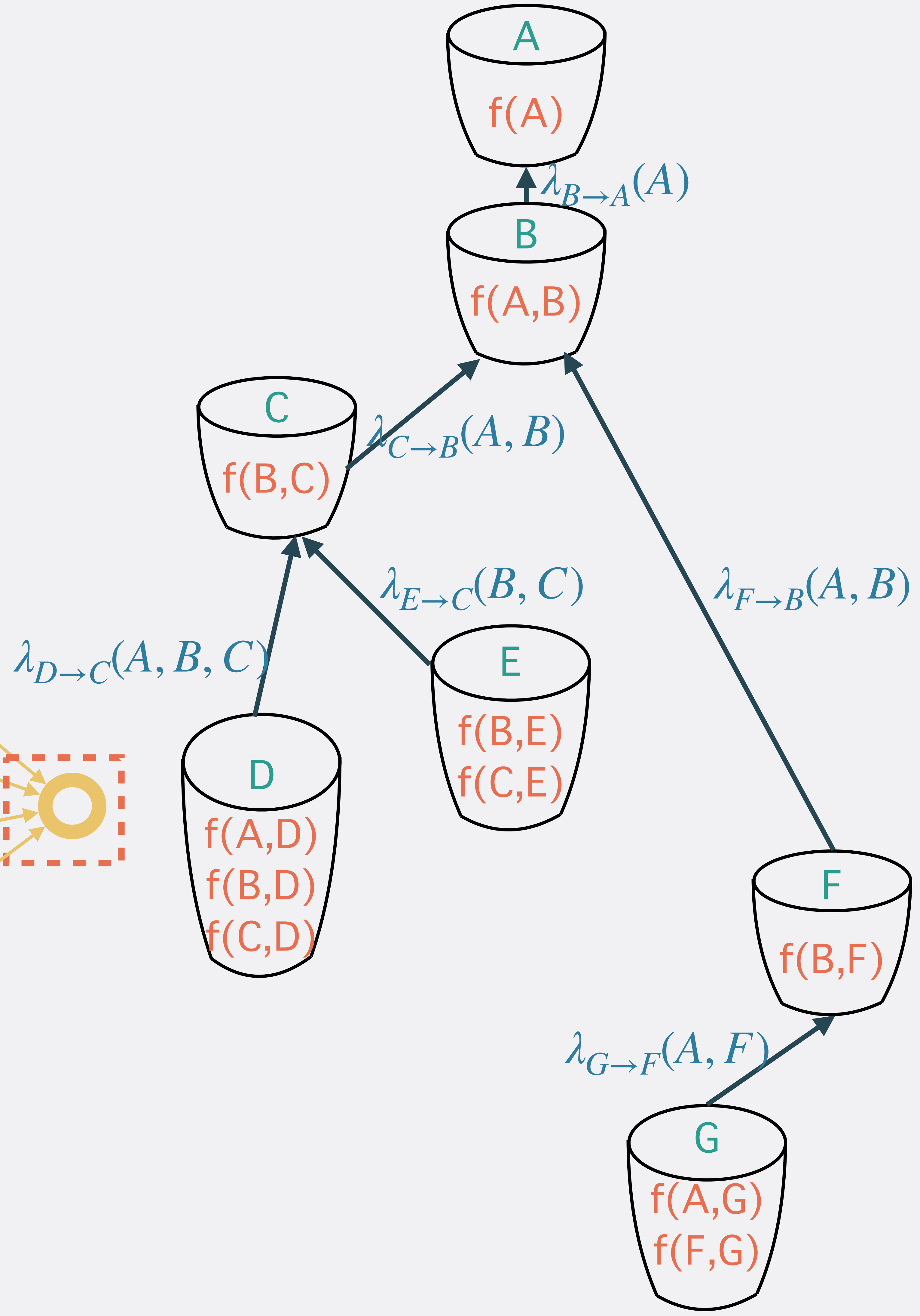
Generating the Samples

1. Fixed number of samples
2. Sample the configuration uniformly at random

A	B	C	λ(A,B,C)
0	0	1	
0	1	1	
1	0	1	
1	1	1	

i-bound = 2





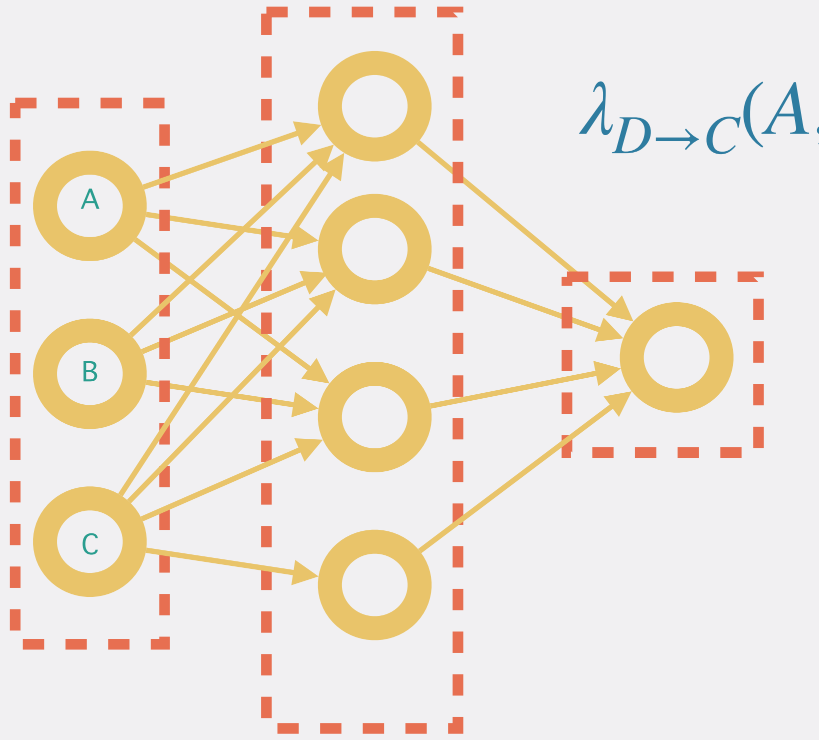
Generating the Samples

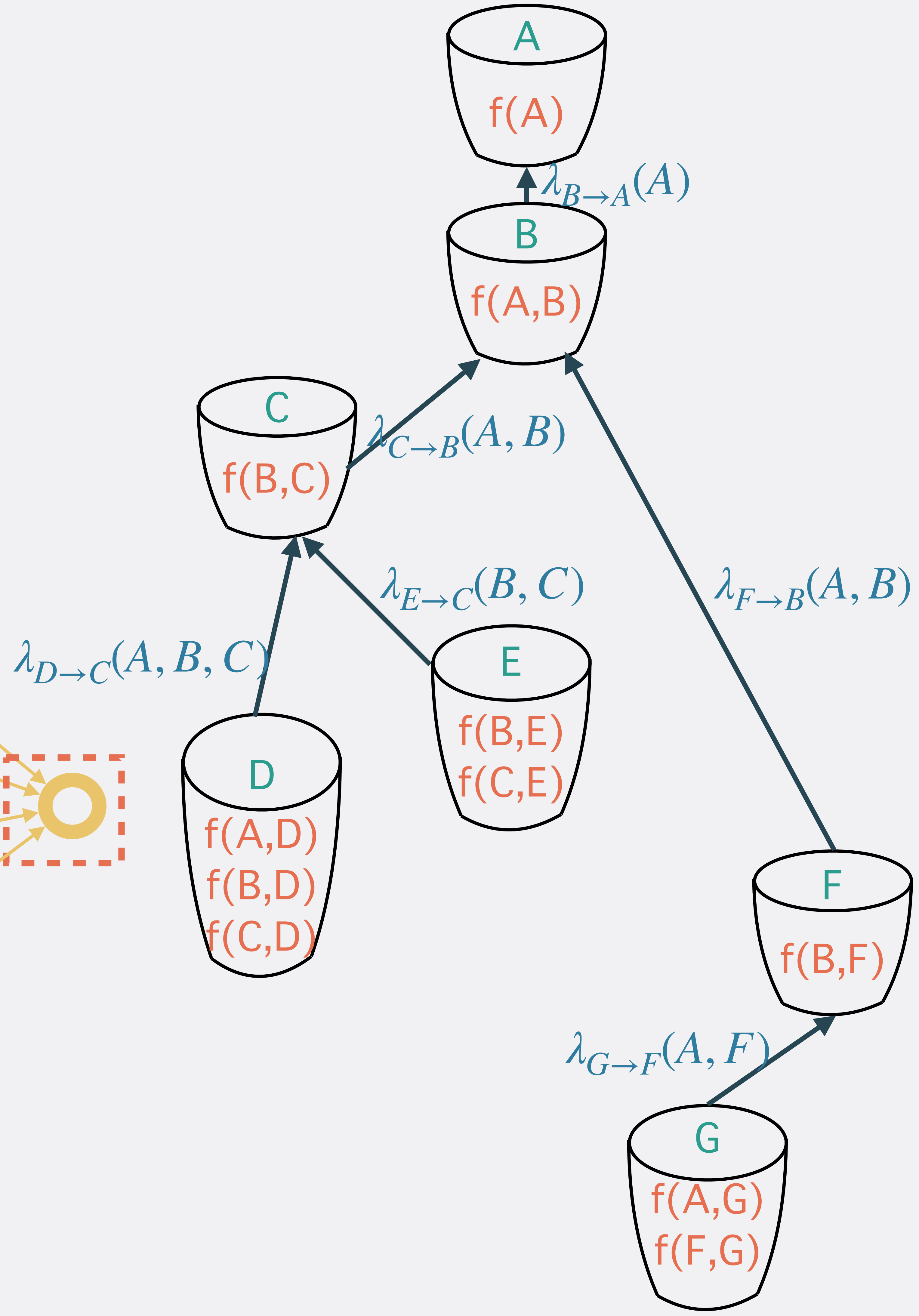
1. Fixed number of samples
2. Sample the configuration uniformly at random
3. Generate the corresponding values

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

$$\lambda_{(D\rightarrow C)}(A,B,C) = \sum_{D} f(A,D)f(B,D)f(C,D)$$

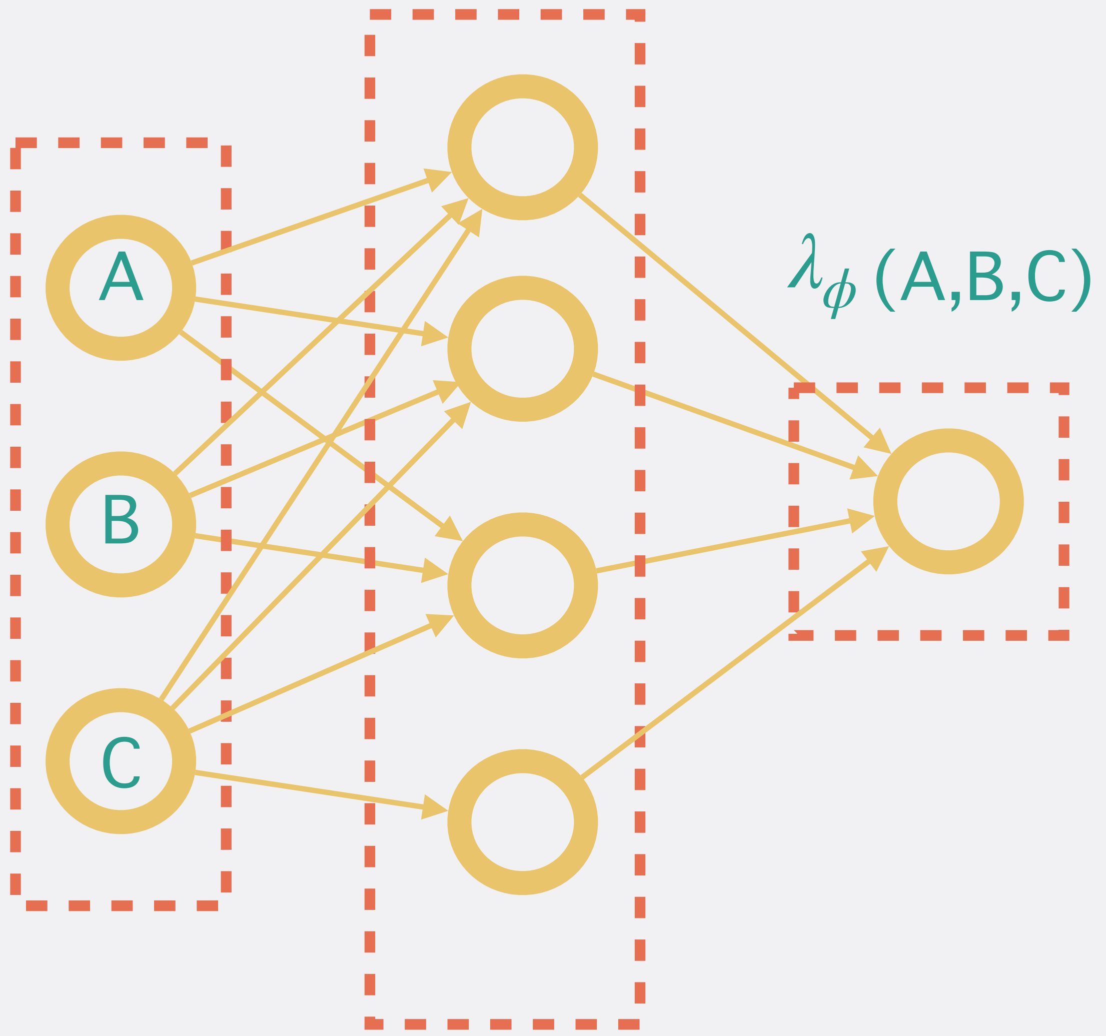
i-bound = 2

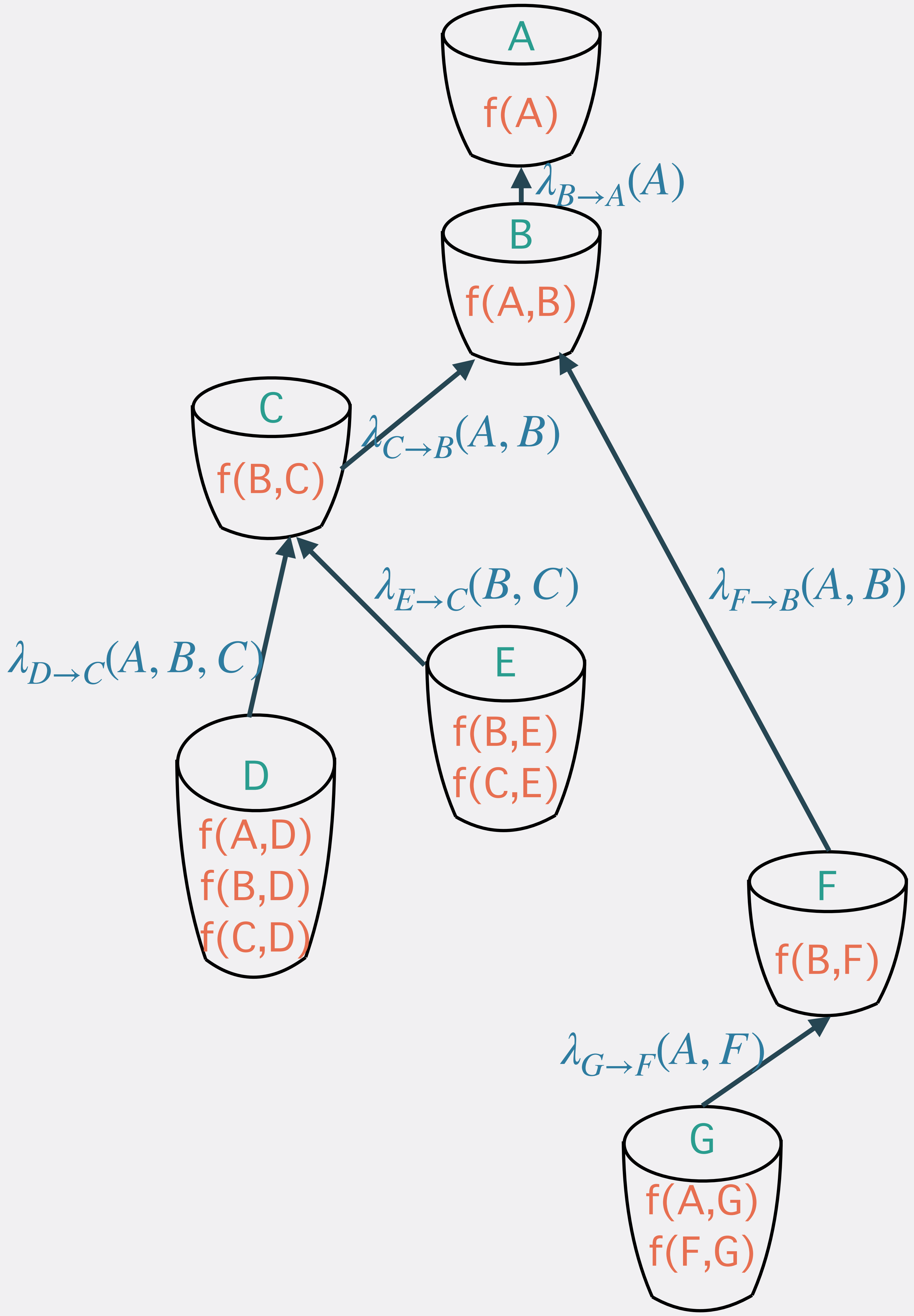




Learning the Functions

1. Fixed architecture





i-bound = 2

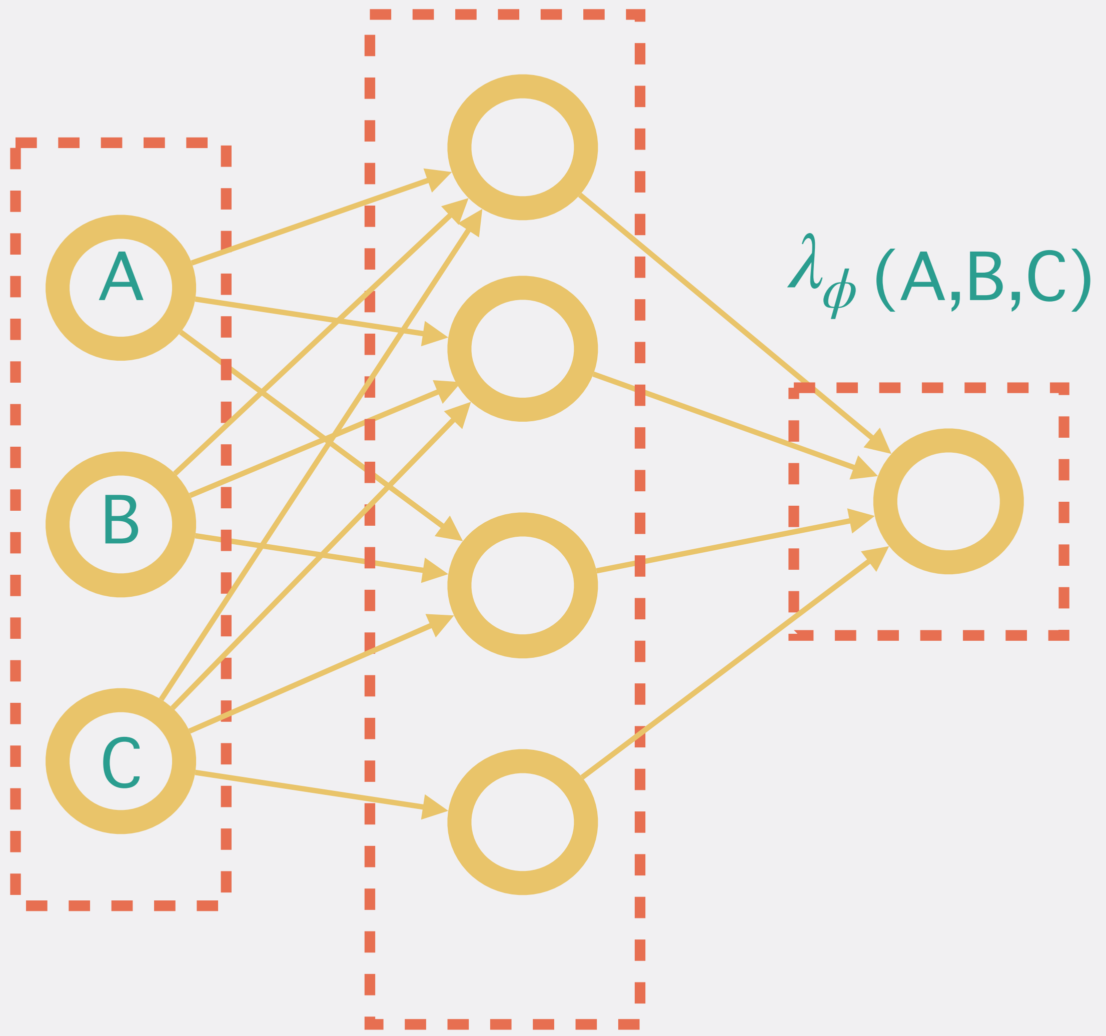
Learning the Messages

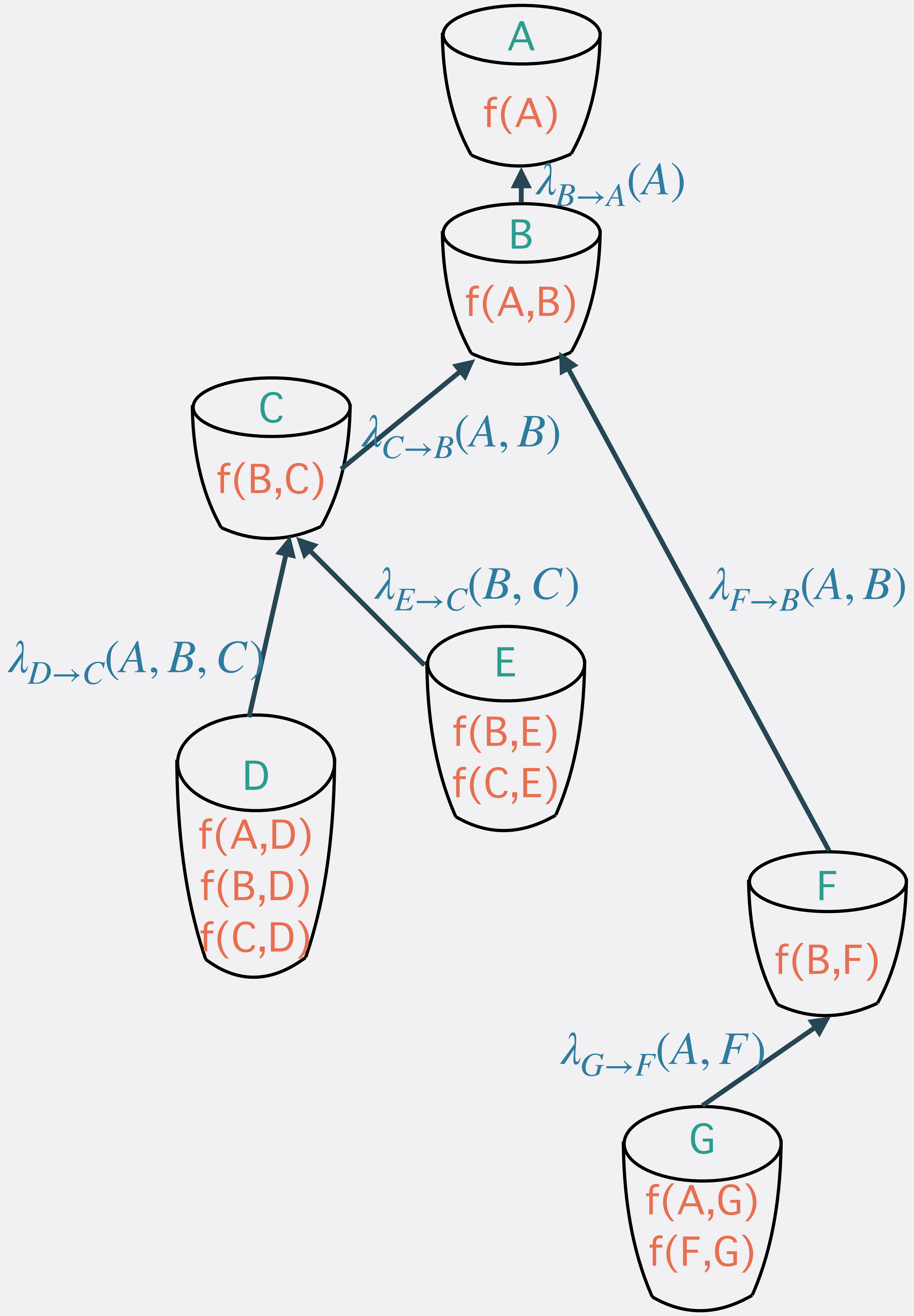
1. Fixed architecture

2. Using the generated samples for training

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

i-bound = 2



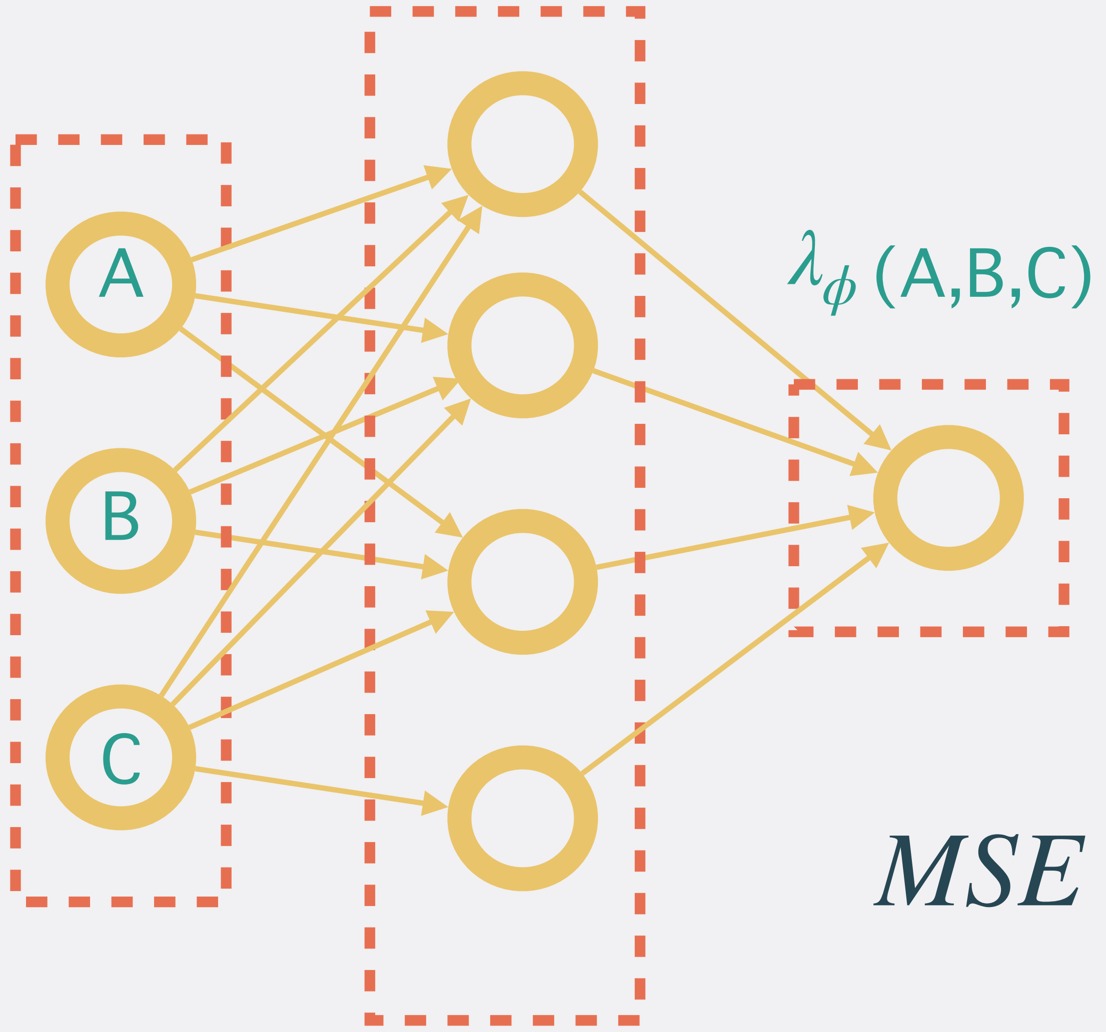


Learning the Messages

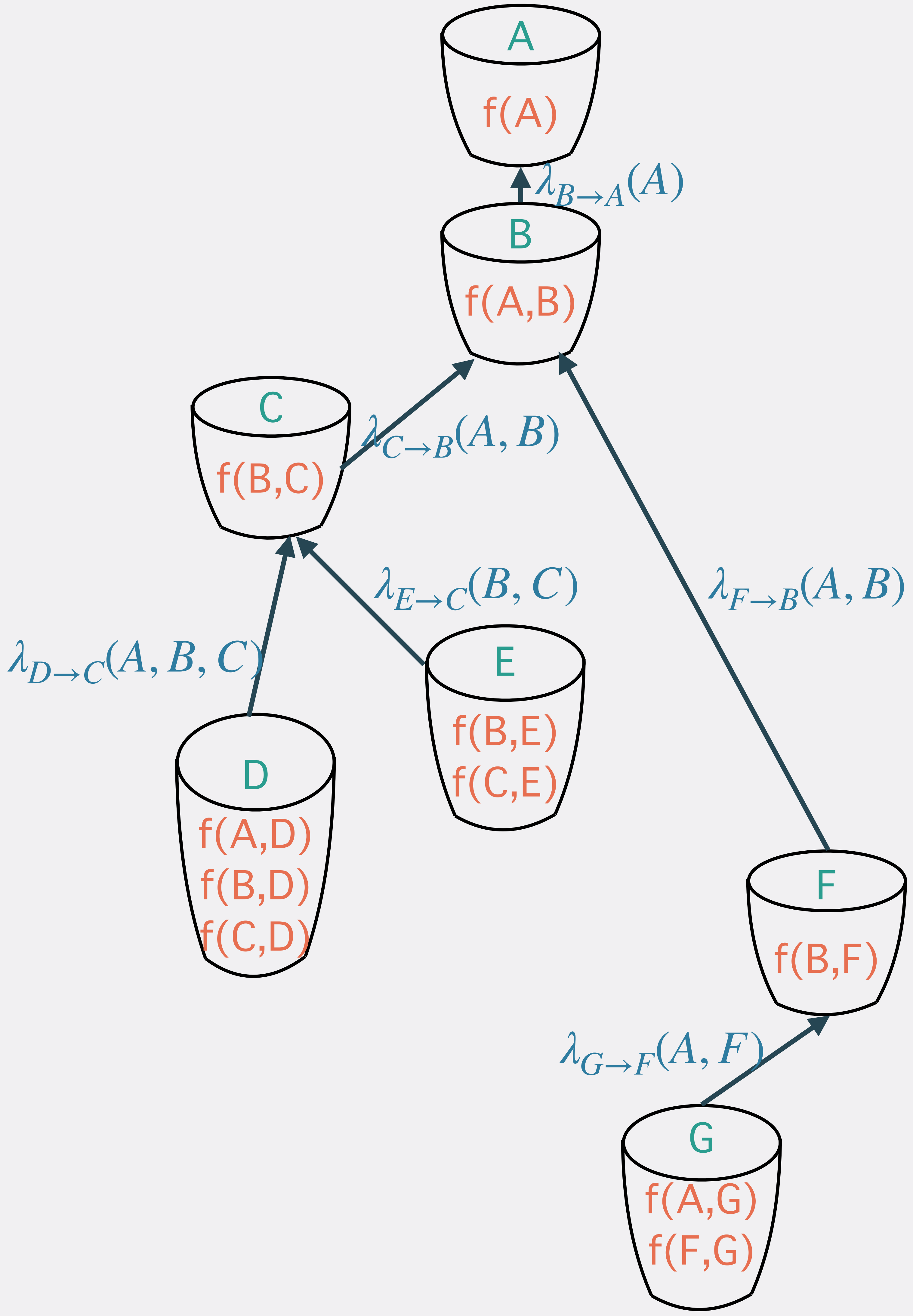
1. Fixed architecture
2. Using the generated samples for training
    1. Minimize MSE loss function with Adam optimizer.

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

i-bound = 2



$$MSE = \frac{1}{ns}\sum_{n=1}^{ns}(\lambda_\phi(s_n) - \lambda(s_n))^2$$



Deep Bucket Elimination - Complexity Analysis

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

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

n : number of variables

k : domain size

r : number of functions

$|NN|$: NN size

$|T_{NN}|$: NN evaluation time

For details please see the paper.

Methodology

Methodology - Algorithms

Comparing with the Weighted Mini Bucket (WMB) [1, 2] Scheme

A. Is in the same class with DBE

B. Has i-bound as approximating parameter

C. Used as a preprocessing step for more powerful algorithms [3]

[1: Dechter and Rish, 2003]
[2: Liu and Ihler, 2012]
[3: Kask, Pezeshki, Broka, Ihler, Dechter, IJCAI 2020]

Methodology - Benchmarks

- Diverse set of benchmarks from UAI repository
  A. Easy or hard
  B. With deterministic or without-deterministic

- Benchmarks
  A. Grid(vision domain): easy and hard instances, without-deterministic, 12 instances
  B. Pedigree(genetic linkage analysis domain): hard, with-deterministic, 7 instances
  C. DBN: medium- without determinism, 6 instances

Methodology - Performance Measure

$$error = |\, log_{10}Z^* - log_{10}\hat{Z}\,|$$

$\hat{Z}$ is the approximated partition function

Z^* is the reference partition function

Exact Z^* when available for easy instances

Estimated Z^* from an advanced sampling scheme [1] for hard problems

[1:Kask, Pezeshki, Broka, Ihler, Dechter, IJCAI 2020]

Methodology - Design Choices

Feed Forward and MaskedNet architectures.

Fixed number of Samples: 500000

Empirical Evaluation

Empirical Evaluation - Results Grids

i-bound=20					DBE						WMB	ref Z
								statistics on error over 10 runs				
Id	name	k	#v	w	Arch	#NB	avg val mse	stdev	avg error	smallest error	error	
1	grid4040f10	2	1600	55	ff-2layers, 100 hidden units each	308	9.29E-06	65.15	**97.14**	**11.81**	215.45	5490
2	grid4040f5	2	1600	55		308	9.17E-06	34.96	**39.9**	**6.28**	84.92	2800
3	grid4040f2	2	1600	55		308	7.50E-06	5.4	**7.34**	**1.2**	25.24	1220
4	grid4040f2w	2	1600	55		376	1.07E-05	20.52	**15.12**	**0.92**	32	1231
5	grid4040f15	2	1600	55		308	9.38E-06	34.2	**83.46**	**41.78**	338.2	8200
6	grid4040f15w	2	1600	55		376	1.37E-05	192.2	**220.91**	**95.23**	657.03	8230

Grids (vision domain), hard instances, without-deterministic

Empirical Evaluation - Results Grids

i-bound=20					DBE						WMB	ref Z
Id	name	k	#v	w	Arch	#NB	avg val mse	statistics on error over 10 runs			error	
								stdev	avg error	smallest error		
1	grid4040f10	2	1600	55	ff-2layers, 100 hidden units each	308	9.29E-06	65.15	**97.14**	**11.81**	215.45	5490
2	grid4040f5	2	1600	55		308	9.17E-06	34.96	**39.9**	**6.28**	84.92	2800
3	grid4040f2	2	1600	55		308	7.50E-06	5.4	**7.34**	**1.2**	25.24	1220
4	grid4040f2w	2	1600	55		376	1.07E-05	20.52	**15.12**	**0.92**	32	1231
5	grid4040f15	2	1600	55		308	9.38E-06	34.2	**83.46**	**41.78**	338.2	8200
6	grid4040f15w	2	1600	55		376	1.37E-05	192.2	**220.91**	**95.23**	657.03	8230

Grids (vision domain), hard instances, without-deterministic

Empirical Evaluation - Results Grids

i-bound=20					DBE						WMB	ref Z
								statistics on error over 10 runs				
Id	name	k	#v	w	Arch	#NB	avg val mse	stdev	avg error	smallest error	error	
1	grid4040f10	2	1600	55	ff-2layers, 100 hidden units each	308	9.29E-06	65.15	**97.14**	**11.81**	215.45	5490
2	grid4040f5	2	1600	55		308	9.17E-06	34.96	**39.9**	**6.28**	84.92	2800
3	grid4040f2	2	1600	55		308	7.50E-06	5.4	**7.34**	**1.2**	25.24	1220
4	grid4040f2w	2	1600	55		376	1.07E-05	20.52	**15.12**	**0.92**	32	1231
5	grid4040f15	2	1600	55		308	9.38E-06	34.2	**83.46**	**41.78**	338.2	8200
6	grid4040f15w	2	1600	55		376	1.37E-05	192.2	**220.91**	**95.23**	657.03	8230

Grids (vision domain), hard instances, without-deterministic

Empirical Evaluation - Results Grids

i-bound=20					DBE						WMB	ref Z
								statistics on error over 10 runs				
Id	name	k	#v	w	Arch	#NB	avg val mse	stdev	avg error	smallest error	error	
1	grid4040f10	2	1600	55	ff-2layers, 100 hidden units each	308	9.29E-06	65.15	**97.14**	**11.81**	215.45	5490
2	grid4040f5	2	1600	55		308	9.17E-06	34.96	**39.9**	**6.28**	84.92	2800
3	grid4040f2	2	1600	55		308	7.50E-06	5.4	**7.34**	**1.2**	25.24	1220
4	grid4040f2w	2	1600	55		376	1.07E-05	20.52	**15.12**	**0.92**	32	1231
5	grid4040f15	2	1600	55		308	9.38E-06	34.2	**83.46**	**41.78**	338.2	8200
6	grid4040f15w	2	1600	55		376	1.37E-05	192.2	**220.91**	**95.23**	657.03	8230

Grids (vision domain), hard instances, without-deterministic

Empirical Evaluation - Results Grids

i-bound=20					DBE						WMB	ref Z
								statistics on error over 10 runs				
Id	name	k	#v	w	Arch	#NB	avg val mse	stdev	avg error	smallest error	error	
1	grid4040f10	2	1600	55	ff-2layers, 100 hidden units each	308	9.29E-06	65.15	**97.14**	**11.81**	215.45	5490
2	grid4040f5	2	1600	55		308	9.17E-06	34.96	**39.9**	**6.28**	84.92	2800
3	grid4040f2	2	1600	55		308	7.50E-06	5.4	**7.34**	**1.2**	25.24	1220
4	grid4040f2w	2	1600	55		376	1.07E-05	20.52	**15.12**	**0.92**	32	1231
5	grid4040f15	2	1600	55		308	9.38E-06	34.2	**83.46**	**41.78**	338.2	8200
6	grid4040f15w	2	1600	55		376	1.37E-05	192.2	**220.91**	**95.23**	657.03	8230

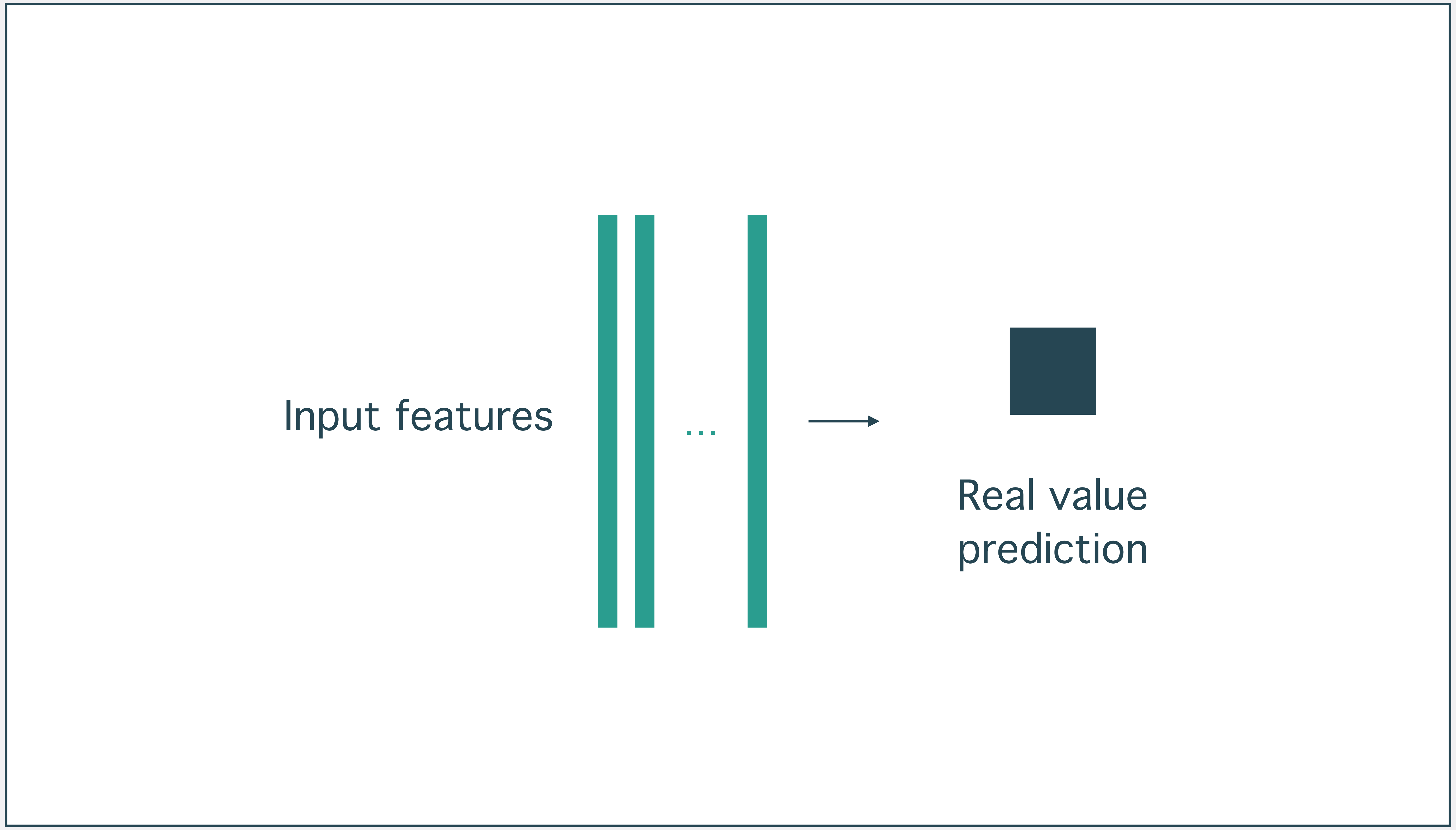
Grids (vision domain), hard instances, without-deterministic

Empirical Evaluation - Results Grids

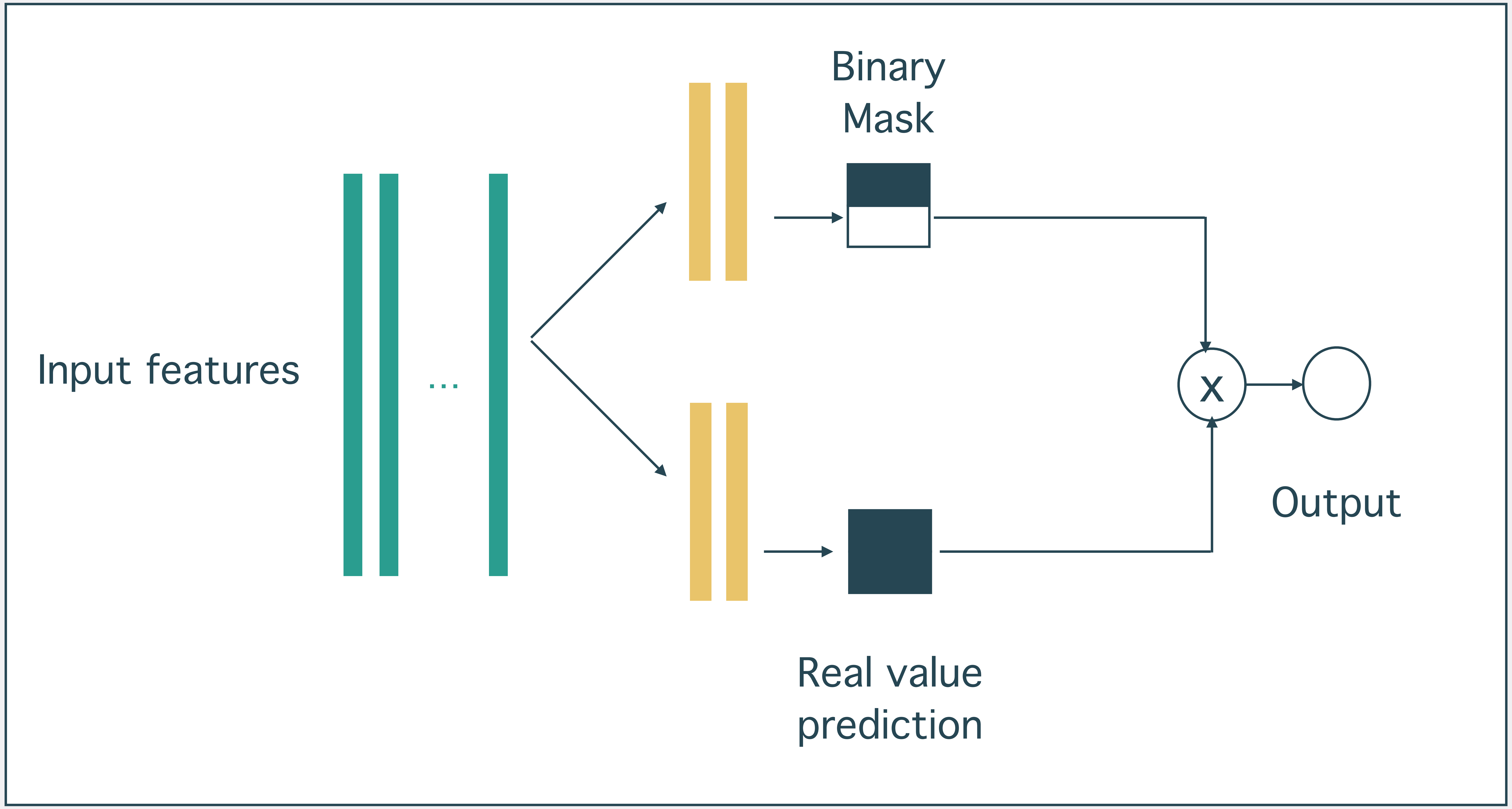
i-bound=20					DBE						WMB	ref Z
								statistics on error over 10 runs				
Id	name	k	#v	w	Arch	#NB	avg val mse	stdev	avg error	smallest error	error	
1	grid4040f10	2	1600	55	ff-2layers, 100 hidden units each	308	9.29E-06	65.15	**97.14**	**11.81**	215.45	5490
2	grid4040f5	2	1600	55		308	9.17E-06	34.96	**39.9**	**6.28**	84.92	2800
3	grid4040f2	2	1600	55		308	7.50E-06	5.4	**7.34**	**1.2**	25.24	1220
4	grid4040f2w	2	1600	55		376	1.07E-05	20.52	**15.12**	**0.92**	32	1231
5	grid4040f15	2	1600	55		308	9.38E-06	34.2	**83.46**	**41.78**	338.2	8200
6	grid4040f15w	2	1600	55		376	1.37E-05	192.2	**220.91**	**95.23**	657.03	8230

Grids (vision domain), hard instances, without-deterministic

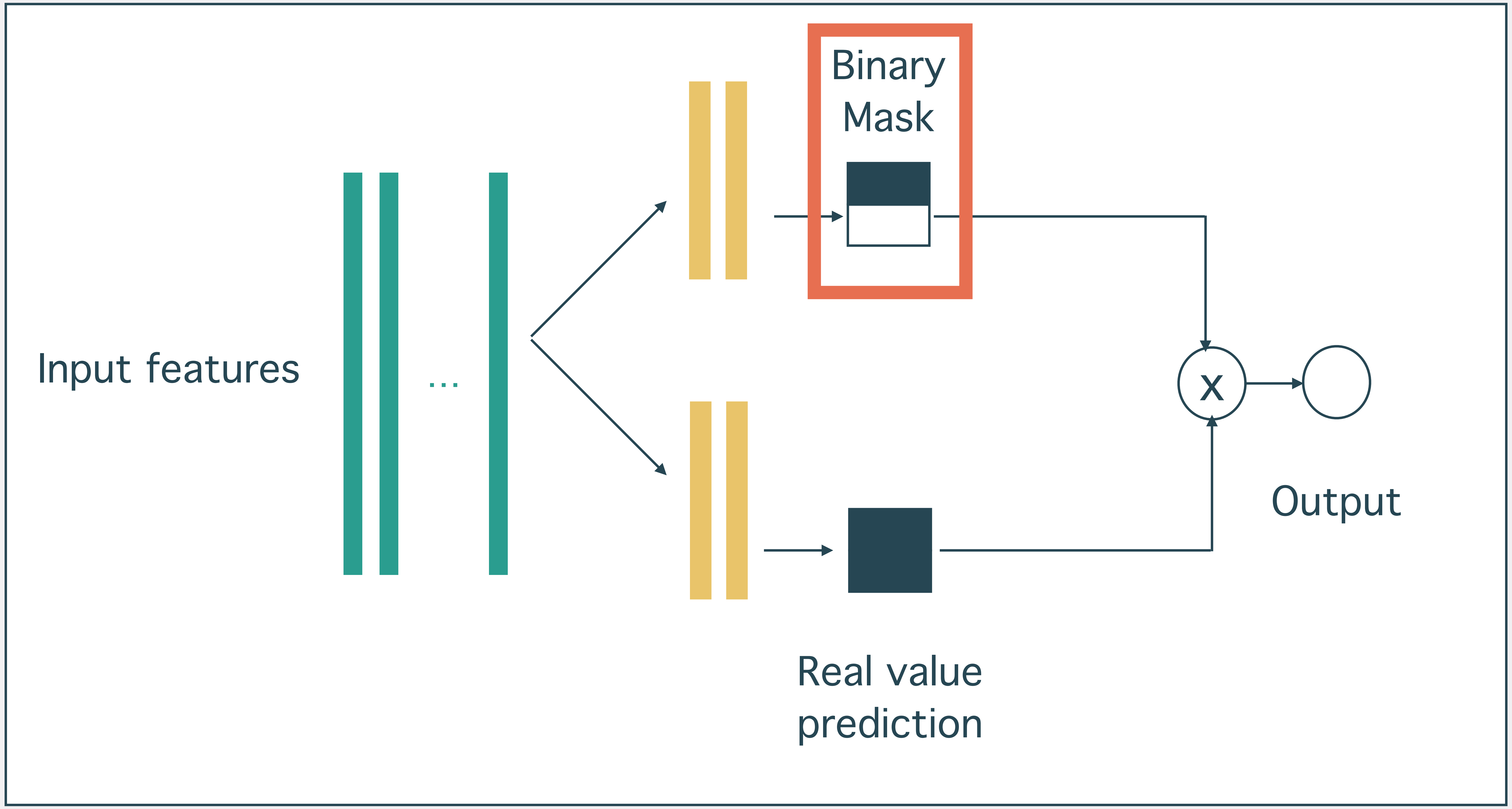
Deep Bucket Elimination - Masked-Net



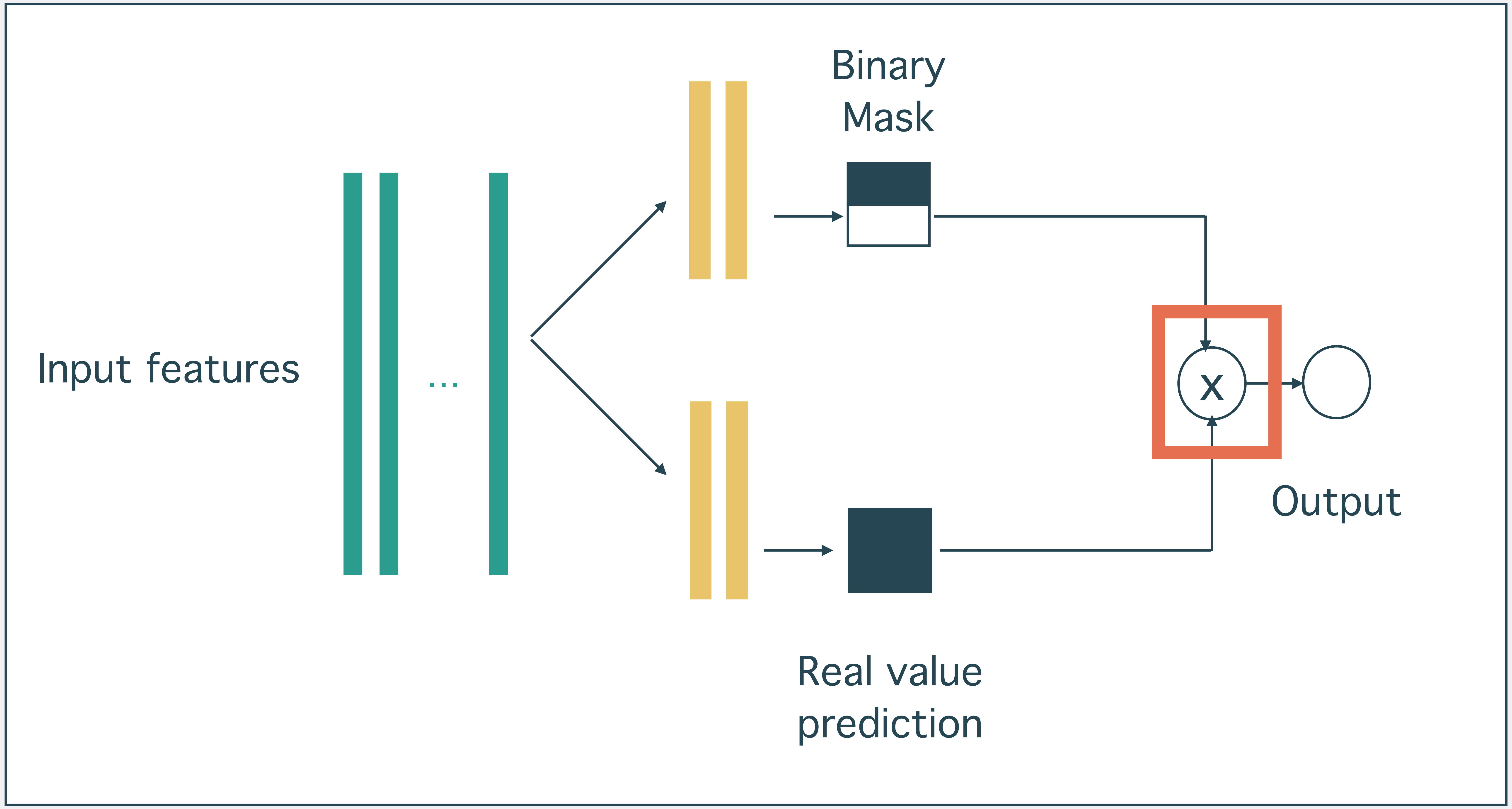
Deep Bucket Elimination - Masked-Net



Deep Bucket Elimination - Masked-Net



Deep Bucket Elimination - Masked-Net



Empirical Evaluation - Results Pedigree

i-bound=20					DBE						WMB	ref Z
								statistics on error over 10 runs				
Id	name	k	#v	w	Arch	#NB	avg val mse	stdev	avg error	smallest error	error	
1	pedigree13	3	888	33	masked-net, ff-3layers, 100 hidden units each	127	2.18E-02	3.374	**3.873**	**0.8927**	6.4696	-31.18
2	pedigree41	5	885	32		92	4.63E-03	0.701	**2.894**	**1.933**	4.1497	-76.04
3	pedigree51	5	871	35		120	4.85E-03	3.193	**8.662**	**4.539**	9.7624	-77.27
4	pedigree34	5	922	33		106	9.71E-03	1.191	**5.93**	**4.14**	7.0762	-64.23
5	pedigree7	4	867	34		108	8.04E-03	0.84	**6.002**	**4.628**	**6.0012**	-64.82
6	pedigree31	5	1006	30		85	9.16E-03	2.169	**5.863**	**0.0178**	12.3603	-78.52
7	pedigree19	5	693	28		43	3.76E-03	1.364	3.663	**1.5882**	2.5809	-59.020

Pedigree (genetic linkage analysis domain), hard instances, with-deterministic

For more results and details please see the paper

Empirical Evaluation - Results Pedigree

i-bound=20					DBE						WMB	
								statistics on error over 10 runs				ref Z
Id	name	k	#v	w	Arch	#NB	avg val mse	stdev	avg error	smallest error	error	
1	pedigree13	3	888	33	masked-net ff-3layers, 100 hidden units each	127	2.18E-02	3.374	**3.873**	**0.8927**	6.4696	-31.18
2	pedigree41	5	885	32		92	4.63E-03	0.701	**2.894**	**1.933**	4.1497	-76.04
3	pedigree51	5	871	35		120	4.85E-03	3.193	**8.662**	**4.539**	9.7624	-77.27
4	pedigree34	5	922	33		106	9.71E-03	1.191	**5.93**	**4.14**	7.0762	-64.23
5	pedigree7	4	867	34		108	8.04E-03	0.84	**6.002**	**4.628**	**6.0012**	-64.82
6	pedigree31	5	1006	30		85	9.16E-03	2.169	**5.863**	**0.0178**	12.3603	-78.52
7	pedigree19	5	693	28		43	3.76E-03	1.364	3.663	**1.5882**	2.5809	-59.020

Pedigree (genetic linkage analysis domain), hard instances, with-deterministic

For more results and details please see the paper

Empirical Evaluation - i-bound

Grid					DBE i-bound 20			DBE i-bound 15		
Id	name	H	#v	w	#NB	error	T(h)	#NB	error	T(h)
1	grid2020f10	e	400	27	31	4.12	1.196	69	18.32	3.156
2	grid2020f5	e	400	27	31	2.07	1.187	69	4.715	3.086
3	grid4040f10	h	1600	55	308	70.5	11.74	421	122.14	19.264
4	grid4040f5	h	1600	55	308	33.4	11.8	421	54.61	19.299

Higher i-bound achieves higher accuracy with less time.

Empirical Evaluation - i-bound

Grid					DBE i-bound 20			DBE i-bound 15		
Id	name	H	#v	w	#NB	error	T(h)	#NB	error	T(h)
1	grid2020f10	e	400	27	31	4.12	1.196	69	18.32	3.156
2	grid2020f5	e	400	27	31	2.07	1.187	69	4.715	3.086
3	grid4040f10	h	1600	55	308	70.5	11.74	421	122.14	19.264
4	grid4040f5	h	1600	55	308	33.4	11.8	421	54.61	19.299

Higher i-bound achieves higher accuracy with less time.

Empirical Evaluation- i-bound

Grid					DBE i-bound 20			DBE i-bound 15		
Id	name	H	#v	w	#NB	error	T(h)	#NB	error	T(h)
1	grid2020f10	e	400	27	31	4.12	1.196	69	18.32	3.156
2	grid2020f5	e	400	27	31	2.07	1.187	69	4.715	3.086
3	grid4040f10	h	1600	55	308	70.5	11.74	421	122.14	19.264
4	grid4040f5	h	1600	55	308	33.4	11.8	421	54.61	19.299

Higher i-bound achieves higher accuracy with less time.

Conclusion

DBE uses the power of neural networks to approximate the bucket elimination algorithm.

Better accuracy in comparison with WMB algorithm has shown.

Limitations:

The algorithm time is much higher than WMB.

We used simple design choices to provide a proof of concept.

https://github.com/dechterlab/DBE