Discrete Optimization for AI problems
Knowledge graphs & Bayesian Graphs

Sanjeeb Dash
IBM Research
Travel supported by ONR

12th Cargese-Porquerolles workshop, Sep 4-8, 2023

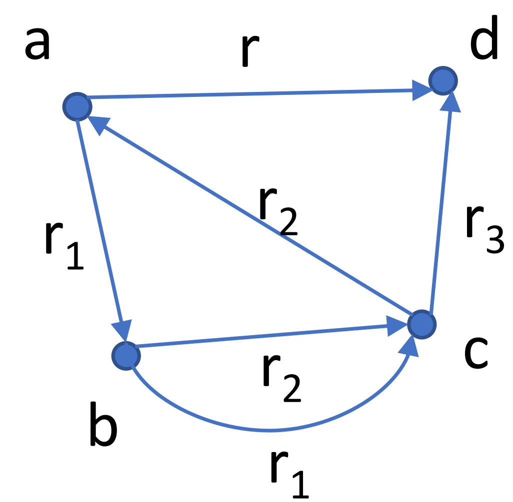
Lecture 3 Outline

- Models to learn rules/knowledge graphs
- Bayesian Network structure learning
- Integer Programming Formulation to find optimal scores
- Latent variables and IP methods
- Numerical Experiments

Knowledge Graph completion

Knowledge Graph (KG): Directed node/edge-labeled multigraph; each edge is a "fact"; edge labels represent binary relations between nodes.

Example: (a, r_1, b) is a fact or $r_1(a, b)$ is true
a, b, c, d could be individuals,
r, r_1, r_2 could *son_of, brother_of, related_to*



Knowledge graphs often have missing (and incorrect) facts.

KG completion problem:
Find missing facts e.g., *(b, brother_of, a), (c, brother_of, a)*

Popular methods: Rule based & Embedding based

YAGO3-10

Chatou	isLocatedIn	France
Boo_Young-tae	playsFor	Yangju_Citizen_FC
Toni_Kuivasto	isAffiliatedTo	Helsingin_Jalkapalloklubi
Josh_Smith_(soccer)	playsFor	Trinity_University_(Texas)
Albrecht_Dürer	diedIn	Nuremberg
Edwin_Holliday	isAffiliatedTo	Hereford_United_F.C.
William_Hopper	actedIn	The_Bad_Seed_(1956_film)
Eric_Maskin	graduatedFrom	Harvard_University
George_Mallia	playsFor	Malta_national_football_team
Héctor_Cúper	isAffiliatedTo	Aris_Thessaloniki_F.C.
Peter_Creamer	wasBornIn	Hartlepool
Robert_Bly	hasGender	male
Bangalore_Urban_district	isLocatedIn	Karnataka
Benedict_Iroha	isAffiliatedTo	D.C._United
Ariel_Garcé	playsFor	Rosario_Central
Trevor_Senior	playsFor	Dorchester_Town_F.C.
Joe_Roberts	actedIn	Our_Hospitality
Emanuele_Concetti	playsFor	U.S._Pergolettese_1932
Warren_Bradley_(footballer)	playsFor	Macclesfield_Town_F.C.
Simone_Zaza	playsFor	F.C._Esperia_Viareggio
Tom_Kouzmanis	isAffiliatedTo	York_Region_Shooters
Ricardo_Moniz	playsFor	Helmond_Sport
László_Sternberg	playsFor	New_York_Americans_(soccer)
Charlie_Chaplin	actedIn	Caught_in_a_Cabaret
Lee_Clarke	playsFor	Northern_Ireland_national_under-21_football_team
Jason_Matthews_(footballer)	isAffiliatedTo	Aberystwyth_Town_F.C.
Alan_Ainscow	isAffiliatedTo	Everton_F.C.
Antoine_Bonifaci	isAffiliatedTo	Bologna_F.C._1909

Rules

Example: $(X, \text{son\_of}, Y) \wedge (Y, \text{son\_of}, Z) \to (X, \text{grandson\_of}, Y)$

KG Completion Problem: Answer query $(a, r, ?)$

Standard Approach:

1. Learn rule-based function $f_r(X, Y)$ that gives high scores to likely facts (X, r, Y) where X, Y are nodes in the graph, and r is an edge-label

2. Answer query $(a, r, ?)$ by finding x such that $f_r(a, x)$ has highest score.

3. If the correct answer is b, measure accuracy by average rank/reciprocal rank of b (MR/MRR)

Prior work

Kok, Domingos ’05, Richardson, Domingos ’06 – Markov Logic Networks
Yang, Yang, Cohen ’17 (NeuralLP) – Neuro-symbolic methods
Rochstätel, Riedel ’17 (NTP) – „
Sadeghian, Armandpour, Ding, Wang ’19 (DRUM) – „
Evans, Grefenstette ’18 – Differential ILP
Das et al. ’18 (Minerva) – Reinforcement Learning
Qu et. al. ’21 (RNNLogic) – RNN + Probabilistic methods
Meilicke et. al. ’19 (AnyBURL) – Data mining
Teru, Denis, Hamilton ‘20 (GraIL) – Subgraph reasoning

Advantages: (1) Inductive reasoning is possible.
(2) Interpretable models when few rules are generated.

Drawbacks: (1) Lower levels of accuracy compared to embedding methods
(2) Current methods do not scale

Embedding based methods

Approach: Find $v_a \in \mathbb{R}^k$ for each node a and a mapping $T_r : \mathbb{R}^k \to \mathbb{R}^k$ for each relation r such that the score $||T_r(v_a) - v_b||$ is small for each fact (a, r, b).

Bordes, Usunier, Garcia-Duran, Weston, Yakhnenko '13 (TransE)
Yang, Yih, He, Gao, Deng '15 (DistMult)
Trouillon, Welbl, Riedel, Gaussier, Bouchard '16 (ComplEx)
Dettmers, Pasquale, Pontus, Riedel '18 (ConvE)
Lacroix, Usunier, Obozinski '18 (ComplEx-N3)
Sun, Deng, Nie, Tang '19 (RotatE)

Advantages: (1) Reasonable accuracy
(2) Scalable

Drawbacks: (1) Not effective for inductive reasoning
(2) Model is not interpretable.

Our work

Goals: Develop a scalable, rule-learner returning compact rule sets

- Interpretability is an explicit goal, and we return low-complexity rules

- We trade off complexity versus accuracy

- Scalability is attained by solving linear programming models instead of non-convex models

Our approach

Approach: Learn few (FOL) rules $R_1, \ldots, R_p$ and positive weights $w_1, \ldots, w_p$ where each R_i has the form

$$r_1(X, X_1) \wedge r_2(X_1, X_2) \wedge \cdots \wedge r_l(X_{l-1}, Y) \rightarrow r(X, Y)$$

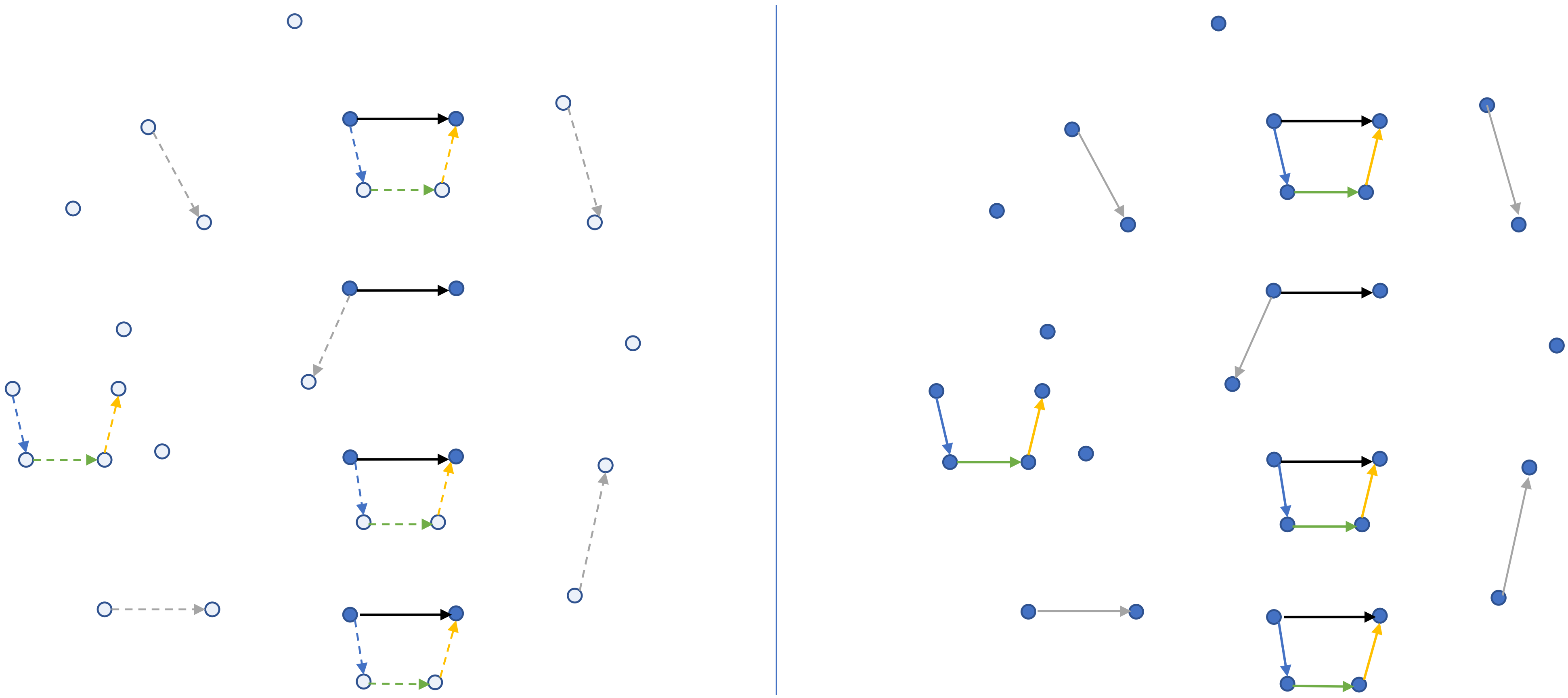
where $r_1, \ldots, r_l$ are relations in G.

Length of this rule is l; left-hand-side is the clause $C_i : V \times V \rightarrow \{0, 1\}$

The learned prediction/scoring function $f_r : V \times V \rightarrow \mathbb{R}_+$ for r is:

$$f_r(X, Y) = \sum_{i=1}^{p} w_i C_i(X, Y) \;\; \forall X, Y \in V$$

Main idea

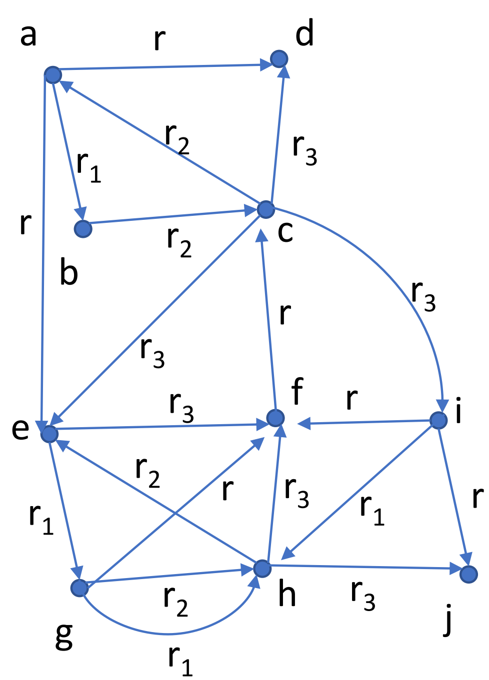


Details

KG:

a-j are entities

r, r_1, r_2, r_3 are relations



$C_1(X, Y)$

Rule $r_1(X, X_1) \wedge r_2\,(X_1, X_2) \wedge r_3\,(X_2, Y) \rightarrow r(X, Y)$ and associated clause-edge vector

$r_1(a, b) \wedge r_2\,(b, c\,) \wedge r_3\,(c, d)$ is true and $r(a, d)$ is true

edge	$r_1 \wedge r_2 \wedge r_3$	r
(a,d)	1	**1**
(a,e)	1	**1**
(f,c)	0	**1**
(g,f)	1	**1**
(i,f)	1	**1**
(i,j)	0	**1**
(e,f)	**1**	0
(a,i)	**1**	0
(e,j)	**1**	0

a_{i1}

positive instances: edges in KG = E_r

negative instances: non-edges (sample)

LP to learn KG rules

Minimize error for weighted collection of rules:

loss on positive instances   loss on negative instances

$$\min_{w,\xi} \sum_{i:y_i=1} \xi_i + \tau \sum_{k \in K} \mathrm{neg}_k w_k$$

cover positives $\longrightarrow$ $\xi_i + \underbrace{\sum_{k \in K} a_{ik} w_k}_{\text{value of scoring fn.}} \geq 1, \quad \xi_i \geq 0, \ (t_{i,} h_i) \in E_r$

complexity bound $\longrightarrow$ $\sum_{k \in K} c_k w_k \leq C$

select clause *k* or not $\longrightarrow$ $w_k \in [0,1], \qquad k \in K$

Model details

- E_r = set edges labeled by r, and (t_i, h_i)= th edge in E_r
- w_k variable gives weight for rule k; $w_k > 0$ implies rule k is chosen
- a_{ik} is a constant $= C_k(t_i, h_i)$
- c_k is a constant = 1+ rule length
- C is a parameter bounding weighted complexity of chosen rules
- τ is a parameter, neg_k is a constant

Modeling – Use all positive facts for a relation + sample some negative facts for the LP model

Algorithmic issues – Use simple shortest path heuristics to find relational paths, and associated rules – Iterate over different values of tau and complexity

Code available at: https://github.com/IBM/LPRules

Column generation

Step 0 – Fix an initial complexity and tau value

Step 1 – Use simple heuristics to create an initial collection of rules

Step 2 – Set up LP model and solve it

Step 3 – Obtain dual values of LP model

Step 4 – Dual values indicate which facts are "well-covered" and which are not. Heuristically generate new rules that "cover" facts that are not well-covered.

Step 5 – Repeat Steps 2 – 4 till termination criterion

Sizes of datasets

Datasets	# Relations	# Entities	# Train	# Test	# Valid
Kinship	25	104	8544	1074	1068
UMLS	46	135	5216	661	652
FB15k-237	237	14541	272115	20466	17535
WN18RR	11	40943	86835	3134	3034
YAGO3-10	37	123182	1079040	5000	5000

Neuro-symbolic methods take a long time on FB15k-237 and cannot handle YAGO3-10

Experiments (accuracy)

Datasets	ComplEx-N3	AnyBURL	NeuralLP	DRUM	RNNLogic	LPRules
Kinship	0.889	0.626	0.652	0.566	*0.687*	**0.746**
UMLS	0.962	**0.940**	0.750	0.845	0.748	*0.869*
FB15k-237	0.362	0.226	0.222	0.225	†**0.288**	*0.255*
WN18RR	0.469	*0.454*	0.381	0.381	0.451	**0.459**
YAGO3-10	0.574	**0.449**				**0.449**

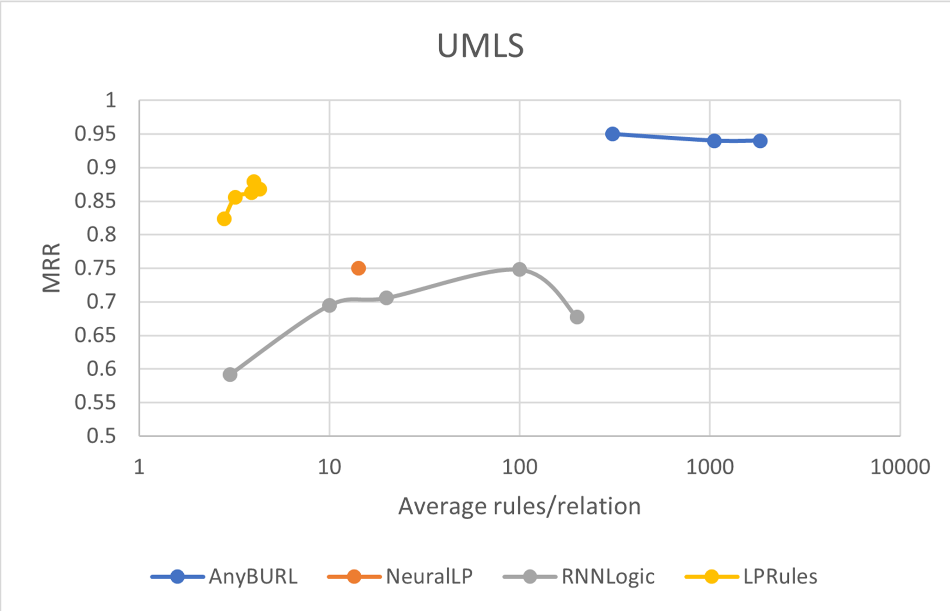
† We could not run RNNLogic on FB15k-237 and report numbers taken from Qu et al. (2021)

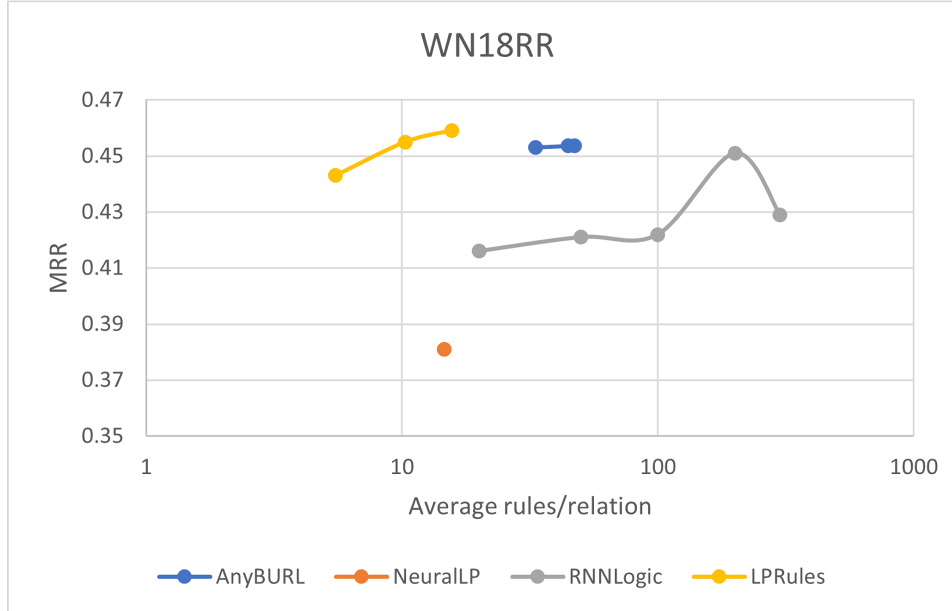
Running time + number of rules

Metric	Datasets	AnyBURL	NeuralLP	RNNLogic	LPRules
Average # rules per relation	Kinship	6653.1	10.4	200.0	21.0
	UMLS	1837.6	15.1	100.0	4.2
	FB15k-237	79.9	8.1		14.2
	WN18RR	47.3	14.3	200.0	15.6
	YAGO3-10	63.0			7.8
Running time	Kinship	1.7	1.6	108.8	0.5
	UMLS	1.9	1.1	133.4	0.2
	FB15k-237	3.9	14565.9		234.5
	WN18RR	1.8	399.9	104.0	11.0
	YAGO3-10	34.3			1648.4

Avg number of rules per relation and wall clock running time on a 60 core machine

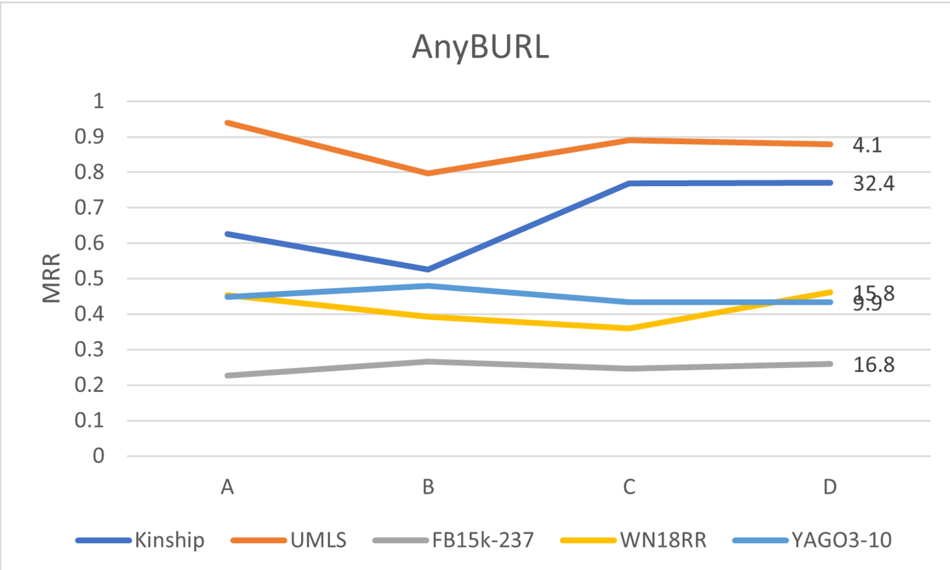
Accuracy versus Complexity tradeoff

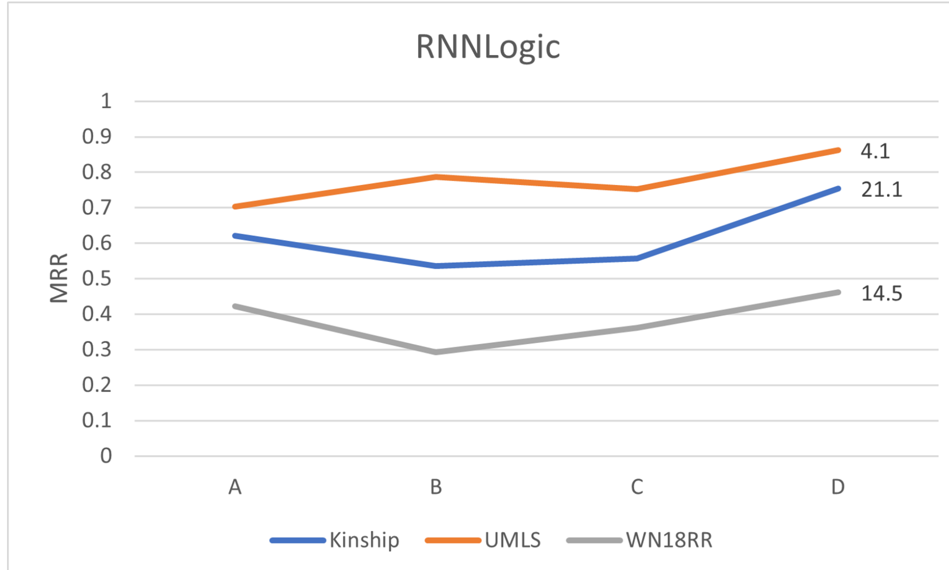




Change in MRR with change in average rules per relation

LPRules + rules from other codes





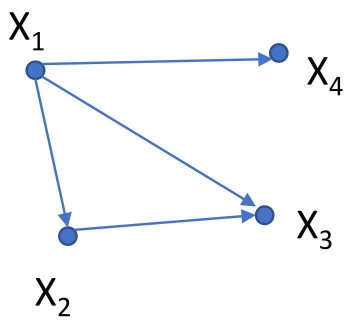
MRR values using rules generated by AnyBURL and RNNLogic (experiments A-D)

A – Use other rule-based code
B – Take rules and weights and use in our prediction function
C – Recalculate weights using complexity bound
D – Add our rules and recalculate weights

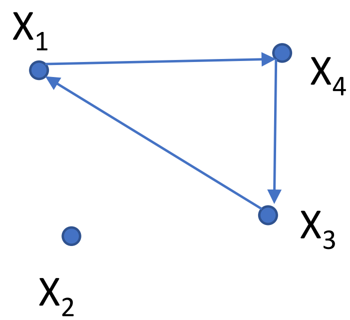
Bayesian Network Structure Learning

Bayesian Network: Directed acyclic graph (DAG) representing conditional probability relationships between variables

dag



directed cycle



$$P(X_1, X_2, X_3, X_4) = P(X_4|X_1)P(X_3|X_1, X_2)P(X_2|X_1)P(X_1)$$

BNSL Problem - Learn DAG from data:

DP methods: Koivisto, Sood ’04, Silander, Myllymäki ’06

A* search: Yuan, Malone ’13

Branch-and-bound: Campos, Ji ’11

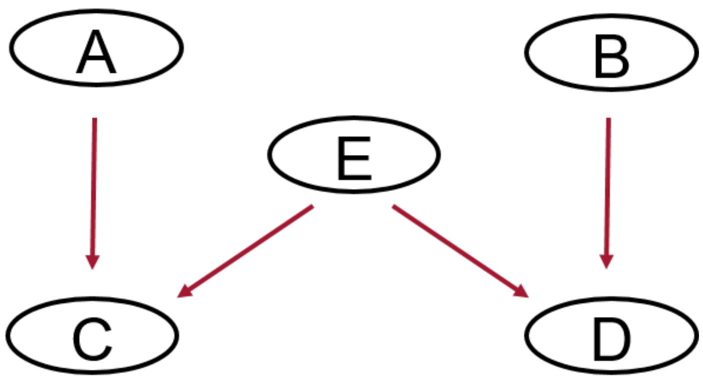
IP based solver GOBNILP: Bartlett, Cussens ’13, ’17

GOBNILP is a state-of-the-art method: Malone et. al. ’17

Causal Graphs/Causal BN

- Graphical Models where directed edges represent causal relationships
- DAG encodes *structural equations*

Directed Acyclic Graph

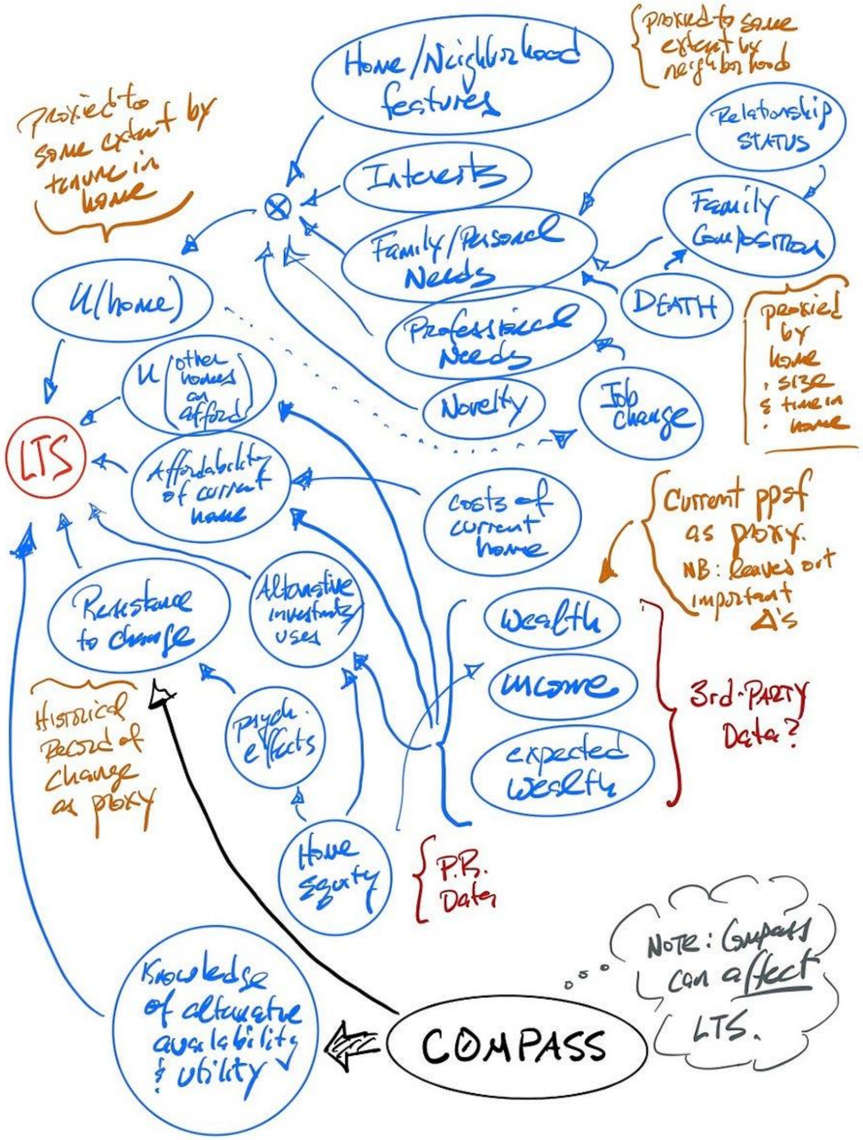


(Linear) Structural equations

$$\Leftrightarrow \begin{cases} x_A = \epsilon_A \\ x_B = \epsilon_B \\ x_C = b_{CA}x_A + b_{CE}x_E + \epsilon_C \\ x_D = b_{DB}x_B + b_{DE}x_E + \epsilon_D \\ x_E = \epsilon_E \end{cases}$$

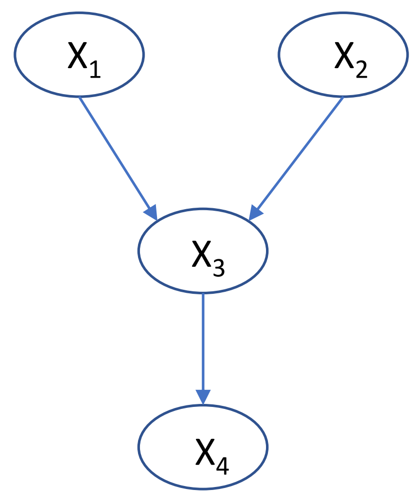
In a BN, $X \to Y \to Z$ and $X \leftarrow Y \leftarrow Z$ are indistinguishable.

Creating causal graphs



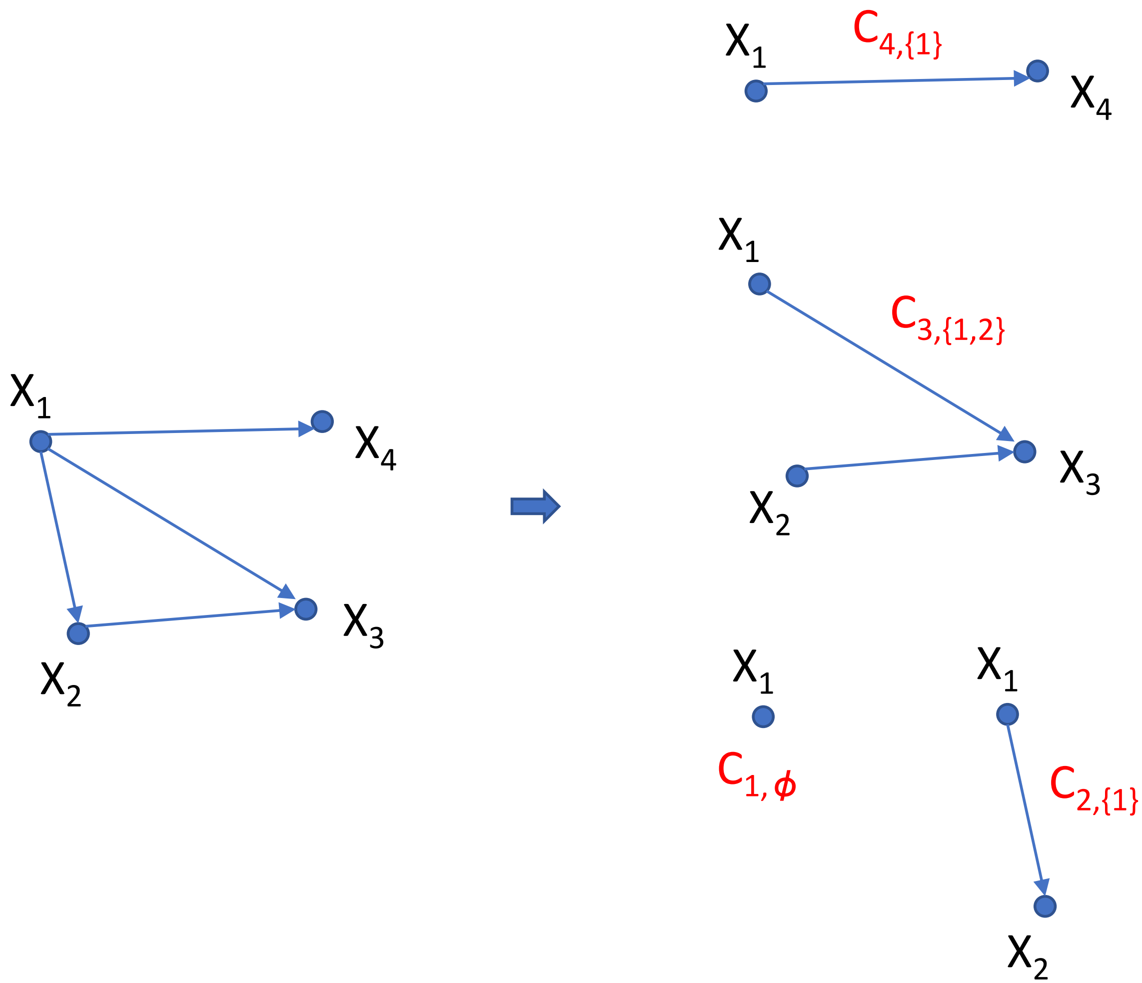
Foster, Ipeirotis 2022

X_1	X_2	X_3	X_4
1	0	1	0
0	1	1	1
1	1	1	0
0	1	1	1
0	0	0	1



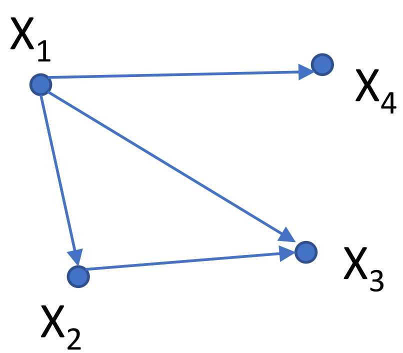
Score decompositions for BNSL

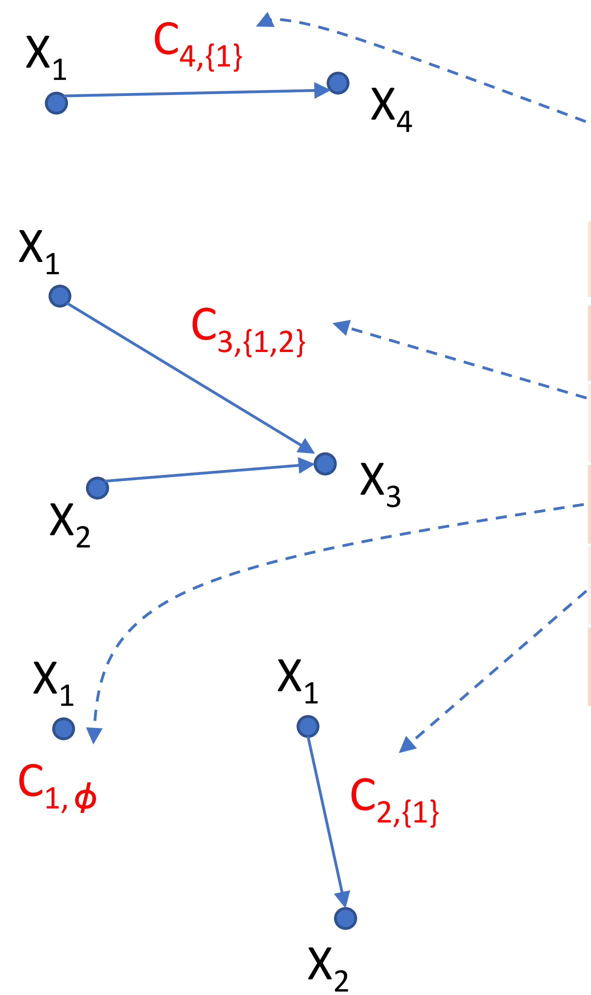
Score of DAG is sum of scores of “in-stars” (inward directed star)



Score calculation

Score of each"in-star" is calculated from data

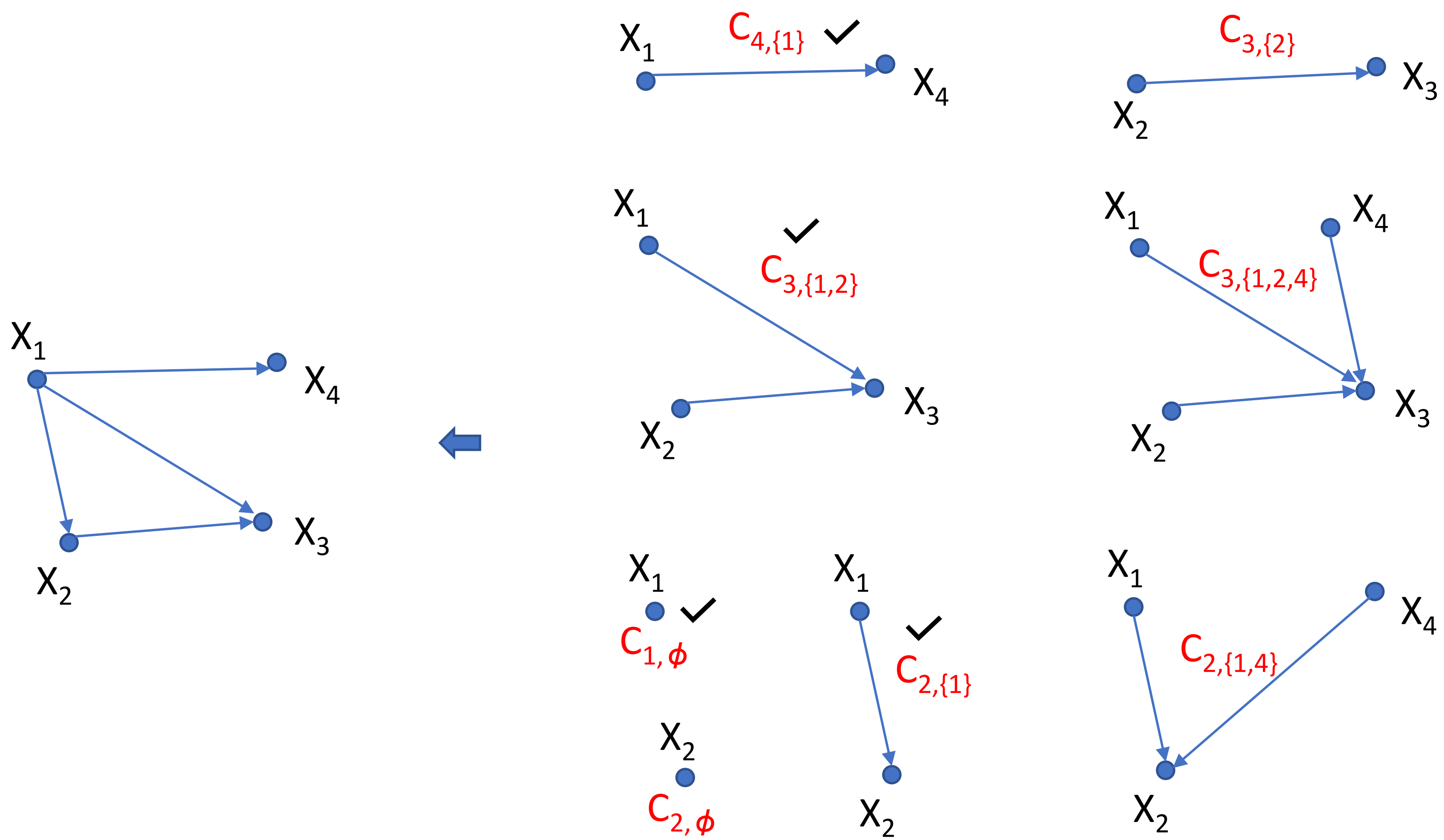




X_1	X_2	X_3	X_4
1	0	1	1
0	1	0	1
1	0	1	1
0	1	0	1
1	0	1	1

MIP for score based approach

MIP has one variable per in-star, equations choosing one in-star per node, and *cluster inequalities* preventing cycles.



Opt. formulations

Notation: Node set - $V = \{1, \ldots, n\}$, $P(i)$ = set of parent sets of i.

MIP (parent set variables):

$$\max \sum_{i \in V} \sum_{P \in P(i)} c_{i,P} z_{i,P}$$

$$\sum_{P \in P(i)} z_{i,P} = 1, \quad \forall i \in V$$

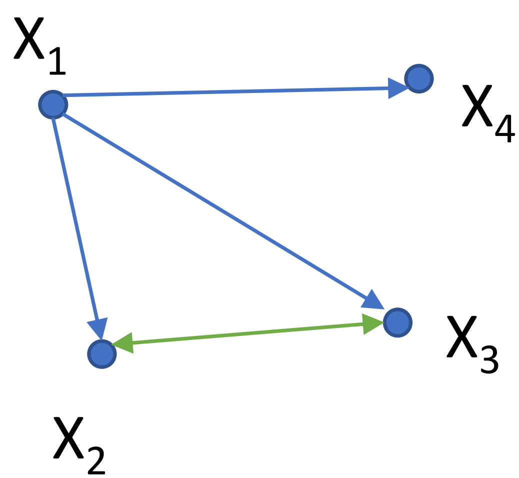
$$\sum_{i \in S, P \cap S = \emptyset} z_{i,P} \geq 1, \quad \forall S \subseteq V \quad *$$

$$z_{i,P} \in \{0, 1\}$$

Jaakkola, Sontag, Globerson, Meila '10: cluster constraints(*)
Bartlett, Cussens '13, 17: IP + software (GOBNILP)
Grotschel, Junger, Reinelt '85: Acyclic subgraph polytope

Latent Variables

Goal: Learn causal network structures in the presence of latent vars.

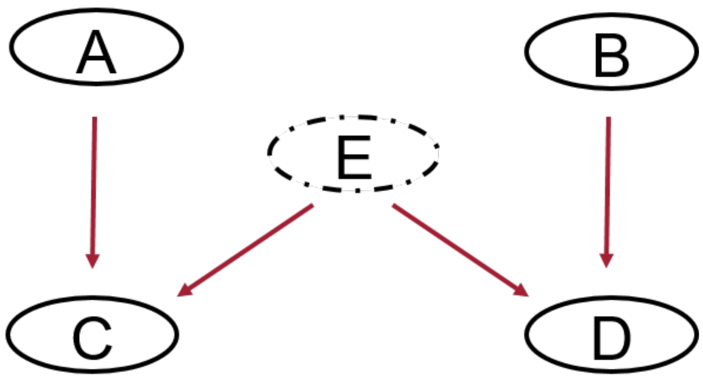


We use **ancestral acyclic directed mixed graphs** (with directed + bidirected edges) as models of data with latent confounders.

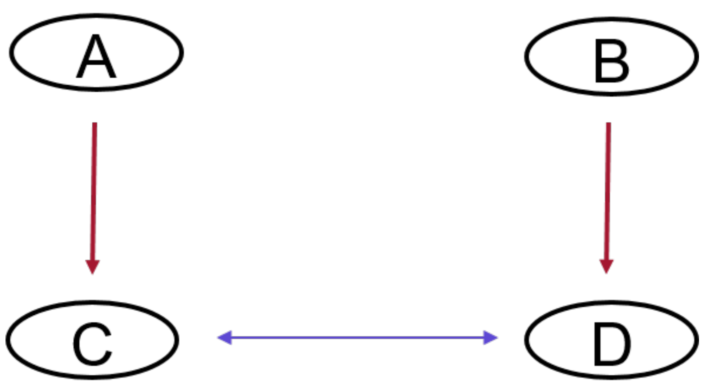
Chen, Dash, Gao '21: MIP formulation & first exact score-based method to find optimal AADMG for continuous Gaussian variables.

Ancestral graphs (AGs)

► DAGs are not closed under marginalization!



Ancestral graphs (Richardson and Spirtes ’02)



► Include all DAGs and are closed under marginalization

► Properties:

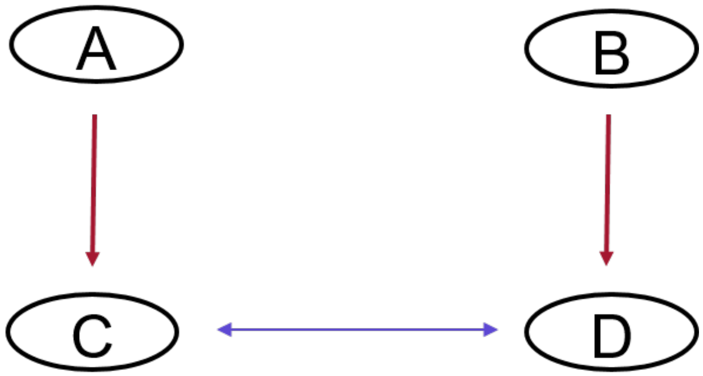
No directed cycles

$(a \to b \to \ldots \to a)$

No almost directed cycles

$(a \leftrightarrow b \to c \to \ldots \to a)$

Continuous Guassian distributions



(Linear) Structural equations

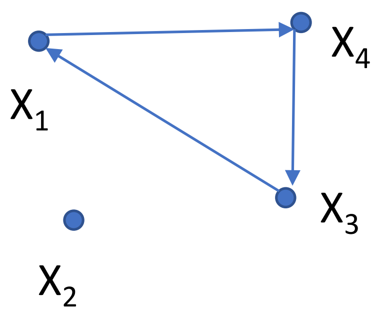
$$\Leftrightarrow \begin{cases} x_A = \epsilon_A \\ x_B = \epsilon_B \\ x_C = b_{CA} x_A + \epsilon_C \\ x_D = b_{DB} x_B + \epsilon_D \\ cov(\epsilon_C, \epsilon_D) = \Omega_{CD} \end{cases}$$

If $\epsilon_A - \epsilon_D$ are normally distributed random variables, then x has a multivariate normal distribution with covariance matrix Σ given by

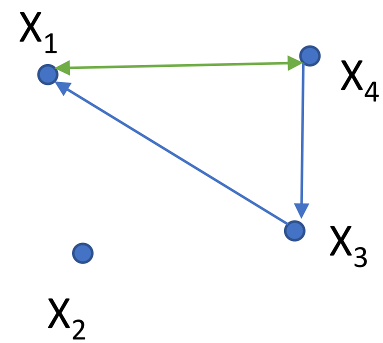
$$(I - B)^{-1}\Omega(I - B)^{-T}$$

Forbidden structures

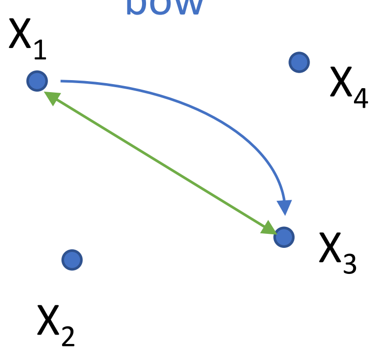
directed cycle



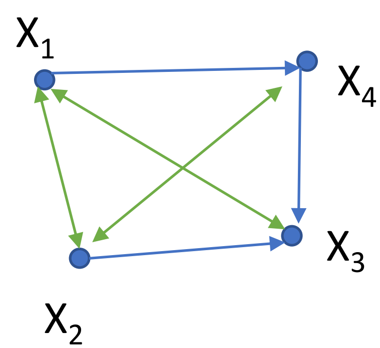
almost directed cycle



bow



rooted arborescence +
bidirected component



Learning methods

Constraint-based methods:

► Apply conditional independence test on the data to infer the graph structure: FCI (Sprites et al., ’00), cFCI (Ramsey et al., ’12)

Score-based methods:

► Optimize a scoring criterion that measures the likelihood of the graph: GSMAG (Triantafillou and Tsamardinos, ’16)

Hybrid methods:

► Use both a scoring criterion and conditional independence tests: M^{3}HC (Tsirlis et al., ’18), SPo (Bernstein et al., ’20), CCHM (Chobtham and Constantinou, ’20)

Current score-based and hybrid methods are all greedy or local search algorithms!

Scoring a DMG

- The BIC score (Schwarz '78) for graph $\mathcal{G}$ is given by

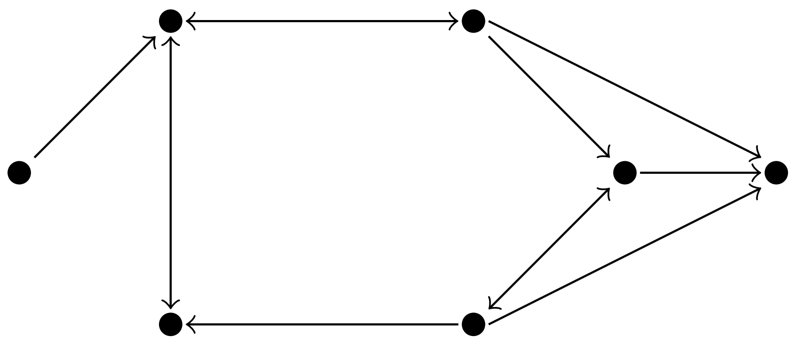
$$\text{BIC}_{\mathcal{G}} = 2\,\mathbf{ln}(l_{\mathcal{G}}(\hat{\Sigma})) - \mathbf{ln}(N)(2|V| + |E|)$$

- The maximum log-likelihood $\mathbf{ln}(l_{\mathcal{G}}(\hat{\Sigma}))$ can be decomposed by c-components in $\mathcal{G}$ (Nowzohour et al., '17)

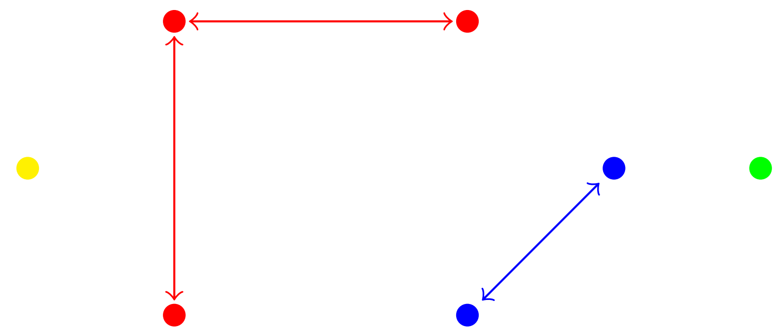
$$\mathbf{ln}(l_{\mathcal{G}}(\hat{\Sigma})) = -\frac{N}{2}\sum_{D\in\mathcal{D}}\Big[|D|\,\mathbf{ln}(2\pi) + \mathbf{log}\big(\frac{|\hat{\Sigma}_{\mathcal{G}_D}|}{\prod_{j\in pa_{\mathcal{G}}(D)}\hat{\sigma}^2_{Dj}}\big) + \frac{N-1}{N}tr(\hat{\Sigma}^{-1}_{\mathcal{G}_D}S_D - |pa_{\mathcal{G}}(D)\setminus D|)\Big]$$

district = component defined by bidirected edges
c-component = district + in-edges per node in district

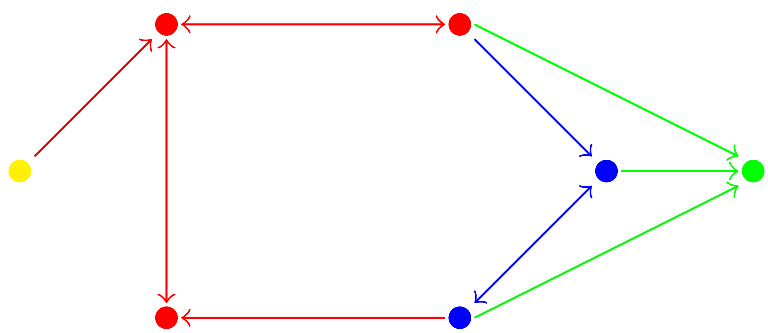
Decomposition into c-components



Ancestral ADMG



Districts

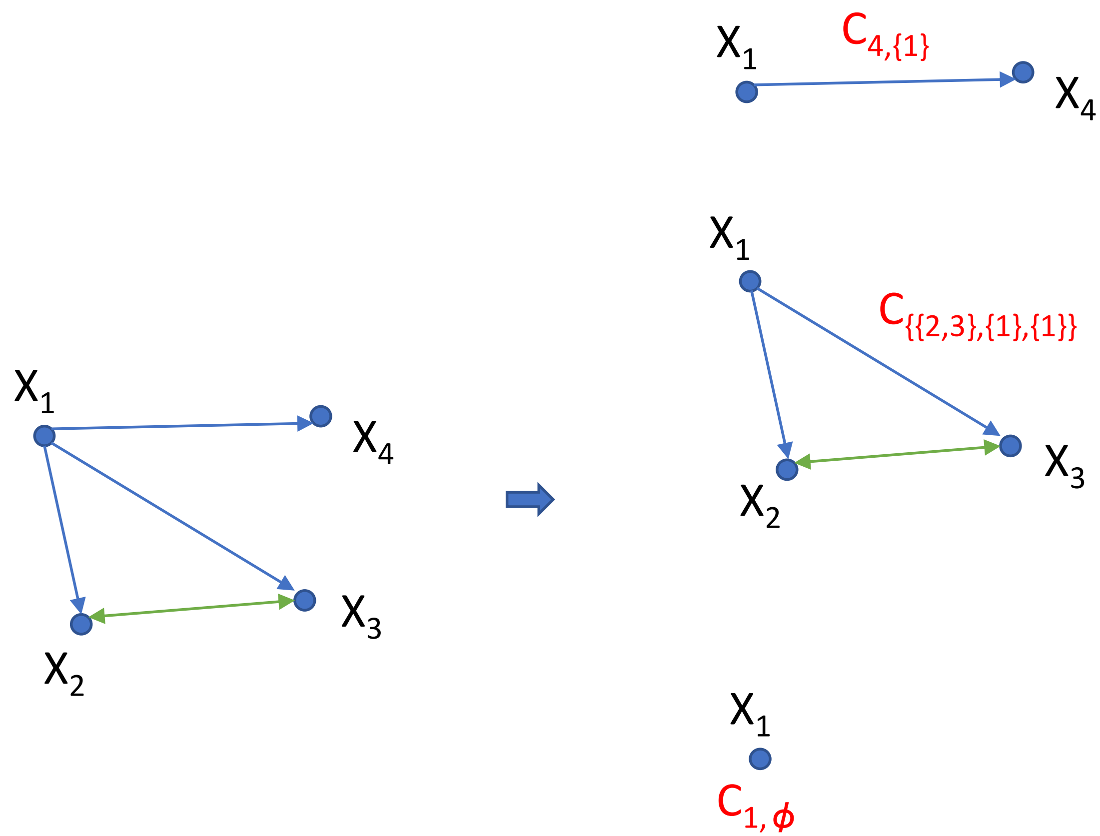


c-components

► We obtain a (BIC) score-maximizing ancestral ADMG for a set of continuous variables that follow a multivariate Gaussian distribution.

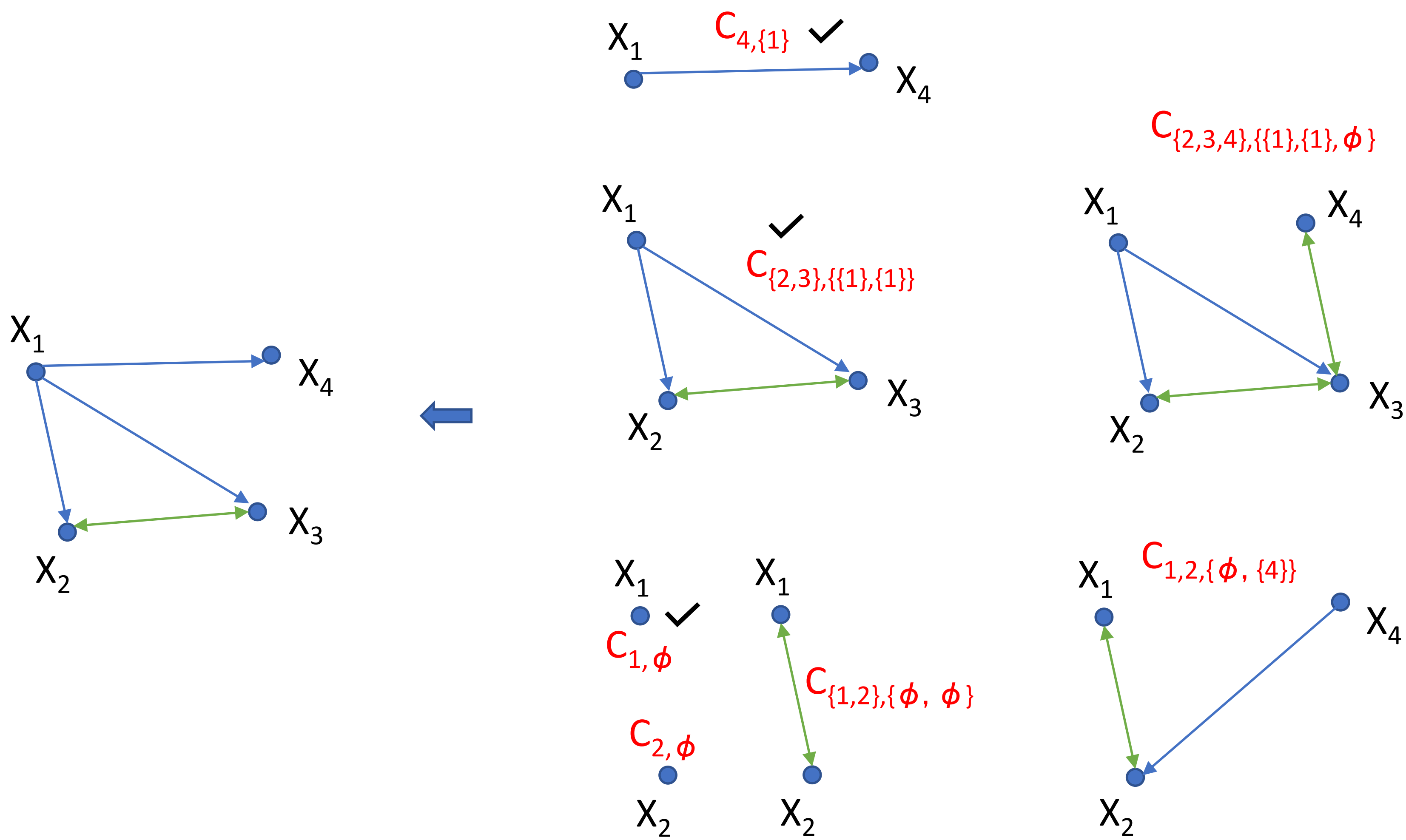
Score decomposition for AADMG

Score of AADMG is sum of scores of c-components



Approach

Our work: Learn an AADMG with maximum score from c-components



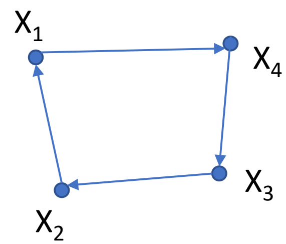
MIP formulation

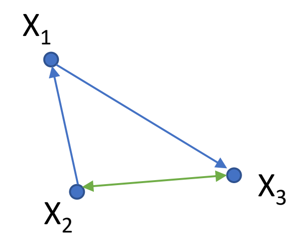
Let $\mathcal{C}$ be set of all c-components, and let $D(C)$ be the district of a c-component C.

MIP to find optimal AADMG:

$$
\begin{aligned}
\text{max} \quad & \sum_{c \in \mathcal{C}} s_C z_C \\
& \sum_{C : i \in D(C)} z_C = 1, \;\; \forall i \in V \\
& G(z) \text{ has no directed and almost directed cycles} \\
& z_C \in \{0, 1\}
\end{aligned}
$$

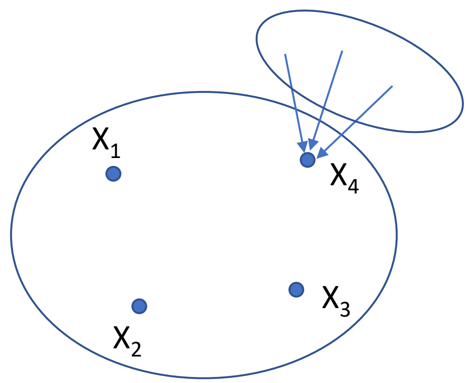
Cutting planes to avoid cycles

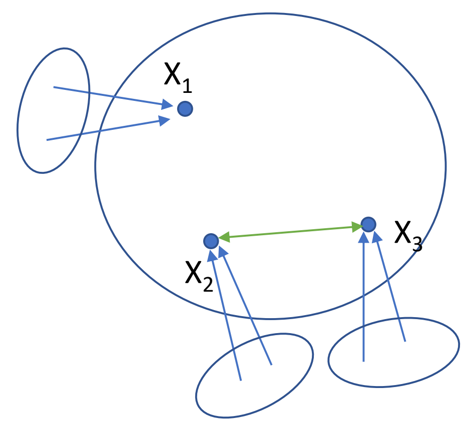




Cluster Inequalities:

$$\sum_{i \in S, P \cap S = \emptyset} z_{i,P} \geq 1, \;\; \forall S \subseteq V$$





Bicluster inequalities: $(w_{i,j} = \sum_{C: i \leftrightarrow j \in D(C)} z_C)$

$$\sum_{v \in S \setminus \{i,j\}} \sum_{P: P \cap S = \emptyset} z_{v,P} + \sum_{P^1: P^1 \cap S = \emptyset} \sum_{P^2: P^2 \cap S = \emptyset} z_{i,j,P^1,P^2} \geq w_{i,j}$$

Cutting planes generation

► Karger's ('93) random contraction algorithm for min-cut problems: Randomly contract edge ij with probability $\propto$ edge weight

► *Separation heuristic* for cluster inequalities:
- Let $\mu^k(S)$ denote the LHS of the cluster inequality at iteration k and

$$w_{ij}^k = \mu^k(\{i\}) + \mu^k(\{j\}) - \mu^k(\{i,j\}),\ \forall i,j$$

- At iteration k, randomly contract edge ij with probability $\propto w_{ij}^k$
- Remove nodes i and j, create a pseudo-node i' and replace all occurrences of i and j in the original graph by the pseudo-node
- Repeat until $\mu^k(\{i\}) < 1$ for some $i \Rightarrow$ a violated cluster inequality

► Similar separation heuristic for bi-cluster inequalities

Numerical Experiments

- Test set 1:

  1. Randomly generated DAGs with 20 nodes
  2. $l =$ 2,4,6 variables set to be latent
  3. $d =$ remaining observed variables
  4. A sample of $N = 1000/10,000$ realizations of observed variables per instance

- Candidate c-components:

  1. Single-node districts with up to three parents
  2. Two-node districts with up to one parent each node

- Compared methods:

  1. AGIP: our IP model
  2. DAGIP: our IP model with only single-node districts
  3. M^{3}HC: a greedy hybrid method by Tsirlis et al. (2018)
  4. FCI: an exact constraint-based method by Sprites et al. (2000)
  5. cFCI: an exact constraint-based method by Ramsey et al. (2012)

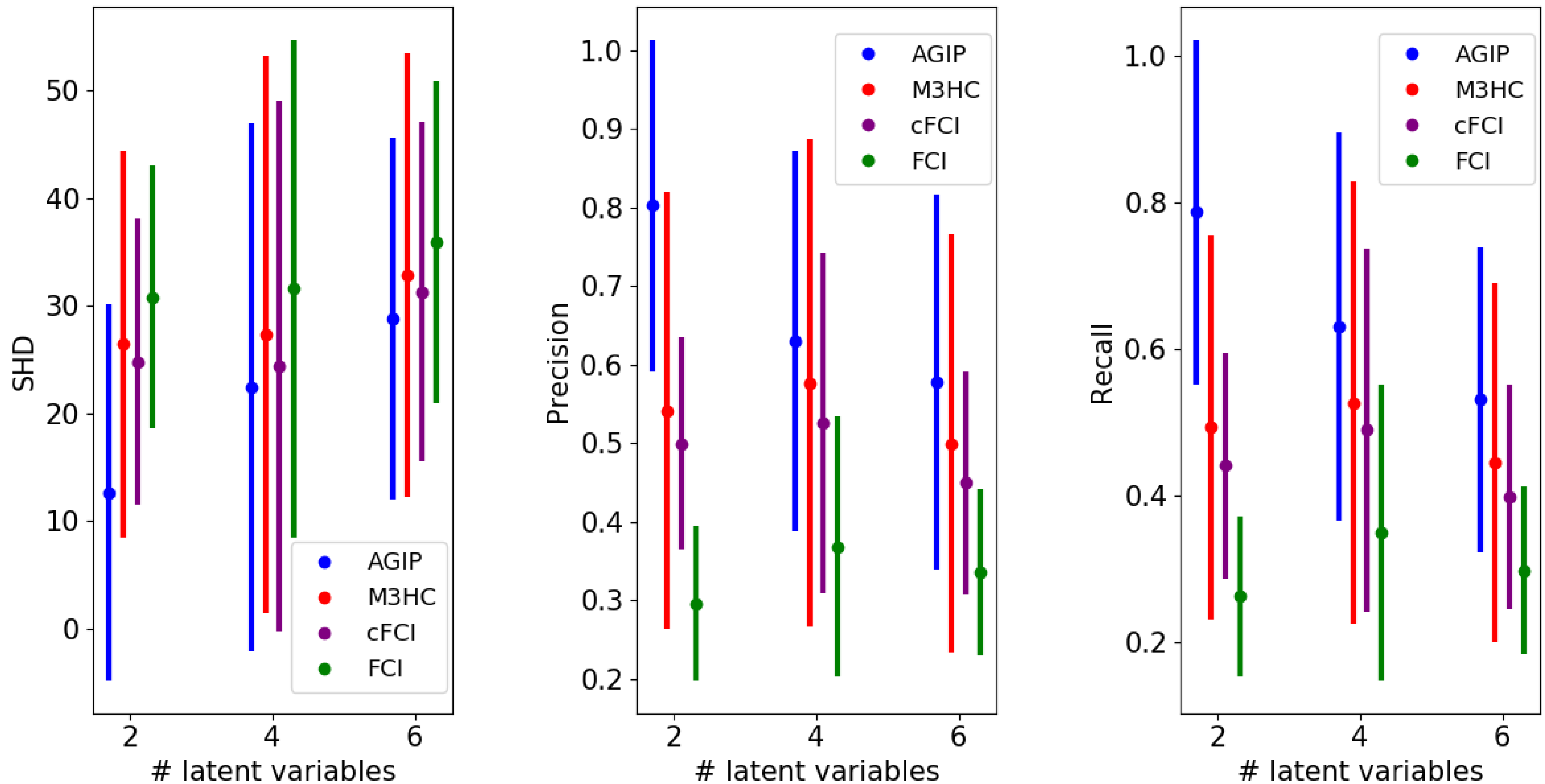
Quality of formulation

20-node graphs; d = number of observed nodes, l = number of latent variables (removed from graph), N = number of samples.

(d, l, N)	Avg # bin vars before pruning	Avg # bin vars after pruning	Avg pruning time (s)	Avg root gap (%)	Avg soln. time (s)
$(18, 2, 1000)$	59229	4116	19.1	0.65	60.4
$(16, 4, 1000)$	39816	3590	13.6	0.43	41.0
$(14, 6, 1000)$	20671	1788	3.9	0.54	8.9
$(18, 2, 10000)$	59229	9038	33.0	0.67	323.2
$(16, 4, 10000)$	39816	7378	21.4	0.53	215.4
$(14, 6, 10000)$	20671	3786	6.4	0.56	47.2

Results for varying number of latent vars.

$d = 18, l = 2, 4, 6, N = 10,000,$



Current work

- Find optimal bow-free/arid graphs (supersets of AADMGs) using MIP

Use BSNL formulation, but extra variables for c-components with > 1 node districts and no bows

MIP (parent set variables):

$$\max \sum_{i \in V} \sum_{P \in P(i)} c_{i,P} z_{i,P}$$

$$\sum_{P \in P(i)} z_{i,P} = 1, \quad \forall i \in V$$

$$\sum_{i \in S, P \cap S = \emptyset} z_{i,P} \geq 1, \quad \forall S \subseteq V \quad *$$

$$z_{i,P} \in \{0, 1\}$$

Sparse instances

Dataset	Ground Truth	AADMG	Bow-free	Bhattacharya
0	-17741.6	**-17741.6**	**-17741.6**	-17765.1
1	-17508.5	**-17508.5**	**-17508.5**	-17511.9
2	-17872.5	*-17871.2*	*-17871.2*	**-17872.5**
3	-19055.6	-19093.6	**-19055.6**	-19123.7
4	-17888.1	-17884.1	-17881.6	-17908.4
5	-18584.9	-18595.5	**-18584.9**	-18625.4
6	-17791.2	-17790.1	-17789.5	-17795.6
7	-18964.8	-19010.8	-18964.8	-20438.8
8	-17562.1	-17562.1	-17562.1	-17565.6
9	-17627.9	-17655.9	-17627.9	-17681.6

Scores for sparse randomly generated datasets

Method	Precision			Recall		
	skeleton	dir.	bidir.	skeleton	dir	bdir
AADMG	0.906	0.711	0.450	0.950	0.818	0.283
Bow-free	0.969	0.812	0.633	0.975	0.873	0.517
Bhattacharya	0.830	0.749	0.179	0.949	0.774	0.383

Average results

Medium density instances

Dataset	Ground Truth	AADMG	Bow-free	Bhattacharya	LP-heuristic
0	-19057.4	-19169.2	-19117.4	-19071.4	-19061.3
1	-19802.3	-20082.1	-19916.3	-19830.9	-19825.3
2	-20606.4	-21074.8	-20857.5	-20613.9	-20623.2
3	-21178.7	-21332.9	-21267.9	-21207.7	-21190.7
4	-20865.8	-20993.5	-20962.1	-20876.5	-20870.1
5	-18846.5	-19031.6	-18936.4	-18848.3	-18855.4
6	-21268.7	-21405.1	-21347.0	-21716.6	-21288.2
7	-18906.2	-18924.9	-18921.7	-18927.6	-18908.4
8	-22152.7	-22517.5	-22320.3	-22226.1	-22189.1
9	-21059.0	-21118.6	-21100.4	-21110.3	-21070.5

Method	Precision			Recall		
	skeleton	dir.	bidir.	skeleton	dir	bdir
AADMG	0.840	0.442	0.100	0.693	0.488	0.050
Bow-free	0.837	0.336	0.083	0.732	0.383	0.034
Bhattacharya	0.799	0.641	0.388	0.946	0.783	0.398
LP-heuristic	0.812	0.424	0.367	0.858	0.589	0.074

Open questions

- How does one deal with the exponentially many variables
- Find valid inequalities for bounded indegree acyclic graphs

Cussens, Jarvisalo, Korhonen, Bartlett '17: detailed study of associated polytopes

References

1. S. Dash, J. Goncalves, Rule induction in knowledge graphs using linear programming, AAAI 2023.
2. R. Chen, S. Dash, T. Gao, Integer programming for causal structure learning in the presence of latent variables. ICML 2021, PMLR 139:1550-1560.
3. J. Cussens, M. Jarvisalo, J. H. Korhonen, M. Bartlett, Bayesian Network Structure Learning with Integer Programming: Polytopes, Facets and Complexity, JAIR **58** (2017).