8．条件付き独立性検定による構造学習

電気通信大学大学院
情報システム専攻
知識処理システム講座
植野　真臣

1．ベイジアンネットワーク学習の問題

- 前回、学習したベイジアンネットワーク学習法は厳密解と呼ばれる手法で予測を最大化する最小Iマップを見つける手法である。しかし，計算量が非常に大きすぎて数十程度のノードのネットワークしか計算できないことが問題である。
- D分離と条件付き独立性は数学的には一対一対応しないが，例外を除けばほぼ同等であると考えて，変数間の条件付き独立性を検定しながら構造を作成する手法が昔からある。これらの手法のほうが比較的計算量が小さいことから実用的であるともいえる。
- 本章では、条件付き独立性検定手法の学習法を紹介しよう。

2．因果のフェイスフルの仮定

条件付き独立性検定による構造学習では，因果のフェイスフルネスの仮定が前提となる．フェイスフルは以下のように定義される．

- 定義

ベイジアン・ネットワーク(G, Θ)において，条件付き確率集合Θをもちいた2ノード間の親ノードを所与としたときの確率分布の条件付き独立性とグラフGでの条件付き独立性が対応しているとき，(G, Θ)は フェイスフル（faithful)であるという．

この仮定は一般的には成り立たない場合があることは授業で学んだ．

3．PCアルゴリズム

(Spirts, Glymour, Scheines, and Tillman 2010, Glymour and Cooper 1999)

最も代表的な古典的条件付き独立性検定による構造学習法である．
このアルゴリズムは，SGSアルゴリズム(Spirts and Glymour 1991)とICアルゴリズム(Verma and Pearl 1990)
を発展させて開発されている．
先ずPCアルゴリズムでは，行2に示されるようにすべてのノード間に辺を引いた完全グラフから開始する．
CIテストの条件部Cは，変数数t=0，つまり，
空集合より始まり，これを0次の条件付き独立性テスト(zero-order CI test)と呼ぶ．順次，
t=1, 2, …と増やし，1次(1st order)，2次(2nd order)，…の条件付き独立性テストと呼ぶ．
定理50に従い，条件付き独立な変数間を早く発見し，その間の辺を完全グラフから順次消去していく．
t=N-2まで以上の処理を行うと，条件付き独立性テストのフェーズが終了し，無向グラフGが得られる．このあと行15より，G中の辺に方向づけを行うオリエンテーションフェーズを行う．

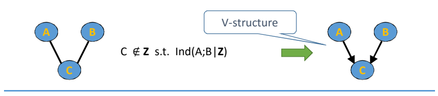
アルゴリズム 32　　PC アルゴリズム:PC(X)

- **Input:** N ノード変数，データ $\mathbf{X}$
- **Output:** ネットワーク構造 $\mathcal{G}$

main

1. ノード集合 $C \leftarrow \phi$
2. 初期グラフ　$\mathcal{G} \leftarrow$ 無向完全グラフ K_N
3. **while** $t < N-2$　**do**
4. 　$t \leftarrow 0$
5. 　**for** 各ノード X_i　**do**
6. 　**for** X_i 以外の各ノード X_j　**do**
7. 　　**for** X_i,X_j の共通隣接ノード集合中の t 個の全てのノードの組み合わせ　**do**
8. $C \leftarrow X_i,X_j$ の共通隣接ノード集合中の t 個のノードの組み合わせ
9. *if* $CI(X_i,X_j \mid C) = 1$ *then* $\mathcal{G}$ からリンク (X_i,X_i) を削除 :ただし，$CI(X_i,X_j \mid C)$ は，C を所与として X_i と X_j が条件付き独立の時 1 となりそれ以外で 0 を返す条件付き独立テスト関数
10. **end for**
11. 　**end for**
12. 　**end for**
13. 　　$t \leftarrow t+1$
14. **end while**
15. 　**for** $\mathcal{G}$ で $X-Y-Z$ を形成する全ての 3 ノード X,Y,Z　**do**
16. *if* X と Z が隣接せず，かつ $CI(X,Z \mid Y) = 0$ *then* $X \rightarrow Y \leftarrow Z$
17. **end for**
18. **while** 方向づけされていないリンクがない　**do**
19. *if* $X \rightarrow Y - Z$ かつ X と Z が隣接しない　*then* $X \rightarrow Z \rightarrow Y$
20. *if* $X \rightarrow Y \rightarrow Z$ かつ X と Z が隣接している　*then* $X \rightarrow Z$
21. *if* $X \rightarrow Y \leftarrow Z$　かつ X と Z が隣接していない，$A-D-C$ かつリンク $B-D$ が存在　*then* $D \rightarrow B$
22. **end while**
23. **return** (G) を出力

オリエンテーション　ルール



アルゴリズムは、二変数間の条件付き独立関数$CI(X_i, X_j \mid C)$を必要とする．条件付き独立テストは複数考えられるが, *Spirtes, Glymour and Scheines(2000)* にあるようにG^2テストを用いることが一般的である．

今, ノード集合Cがc番目の値をとるとき，二変数が$X_i = a$, $X_j = b$の値をとった条件付き頻度をS^{abc}_{ijC}とし, 同様にS^{ac}_{iC}, S^{bc}_{jC}, S^{c}_{C}を定義する．

このとき，G^2統計は,

$$G^2 \equiv 2 \sum_{a,b,c} S^{abc}_{ijC} \ln \frac{S^{abc}_{ijC} S^{c}_{C}}{S^{ac}_{iC} S^{bc}_{jC}} \tag{7.1}$$

として定義される．G^2統計は適当な自由度を持つχ^2分布に漸近的に従う．

欠測値がないことを仮定すると，自由度は

$$df = (\mid D(X_i) \mid -1)(\mid D(X_j) \mid -1) \prod_{C_l \in C} \mid D(C_l) \mid$$

となる．ここで,$\mid D(X) \mid$　はデータに現れるXの値の重複しない数（*distinct values*）を示す．

例 ***48*** 図 *7.1* のベイジアン・ネットワークを考えよう．*PC* アルゴリズムにおける条件付き独立性テストは，図 *7.2* の流れで実施される．

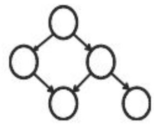


図 **7.1**　ベイジアン・ネットワーク

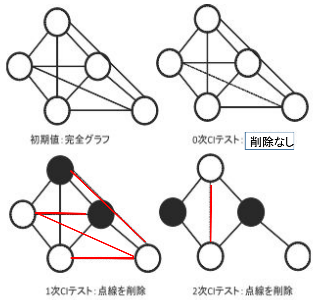


図 **7.2**　PC アルゴリズムによる条件付き独立性テスト

4. MMHCアルゴリズム

PC アルゴリズムは条件付き独立テストの数を劇的に減少させるが，それでもまだ莫大な計算が必要である．そこで，強力なヒューリスティックを用いて探索を行う **MMHC** アルゴリズム *(Max-Min Hill Climbing algorithm)* が提案されている *(Tsamardinos, Brown, and Aliferis 2006)*．*MMHC* アルゴリズムは，*1.* 辺の方向づけをする前に *MMPC(Max-Min Parents and Children)* を用いてターゲット・ノードの隣接ノード集合（スケルトンと呼ばれる）を推定するフェーズ,*2.* スケルトンを基に辺の方向づけを貪欲法（*greedy search*）で探索するフェーズ，に分類される．

今, ターゲットとなるノード変数をTとし，その親集合と子集合をPC_Tと書く．もし, 同じ変数集合を持つベイジアン・ネットワーク(G, Θ)と(G', Θ)が同じ確率分布を持ちフェイスフルであるとき, すべてのTについて, $PC^G_T = PC^{G7}_T$となる *(Verma and Pearl 1990)*．従って，フェイスフルなベイジアン・ネットワークでは，どのようなターゲット・ノードTに対しても一つの同時確率分布に対応するPC_Tはただ一つに決定する．

MMPC は，ターゲット・ノードTについて動作し，データを読み込むことによりTの隣接ノード集合（Tから向かうノード集合とTに向かうノード集合）PC_Tを同定し，返すアルゴリズムである．

Max-Min ヒューリスティック： ターゲット・ノードTのPC_Tの候補に, 所与となる条件変数をTとの関連が最小となるように選択した変数のうち, 関連が最大になる変数を加える．

直感的には，XとTの条件付き独立テストにおいて，どのように努力して条件の変数集合Zを選択してもXとTの連関が高いものは，優先してXとT間の辺を加えていけばよい，という考え方である．

具体的には，このヒューリスティックは以下のように定義できる．

定義 ***92*** XとTの条件付き独立テストで，どのように努力して条件の変数集合Zを選択して

$$MinAssoc(X, T \mid Z) = \min_{S \subseteq Z} Assoc(X, T \mid S)$$

このヒューリステックは定理 *50* における「一つでも$I_P(X, Y \mid Z)$となるZを見つければXとYの間には辺が引かれない」という性質を利用している．

また，関連度は，G^2統計のp値の逆数を用いている．

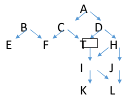
アルゴリズム *33*　　MMPC アルゴリズム:MMPC(T,X)

- **Input:** Nノード変数，ターゲット・ノードT, データ $\mathbf{X}$
- **Output:** ターゲット・ノードTの親ノード集合と子ノード集合

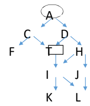
main

1. % フェーズ *1: forward*
2. $CPC = \emptyset$
3. **repeat**
4. $< F, assocF >= MaxMinHeuristic(T, CPC)$
5. **if** $assocF \neq 0$ **then**
6. $CPC = CPC \cup F$
7. **end if**
8. **until** CPC *has not changed*
9. % フェーズ *2: Backward*
10. **for** *all* $X \in CPC$ **do**
11. **if** $\exists S \subseteq CPC$, *s.t.* $CI(X, T \mid S) = 1$ **then**　Xを削除
12. **end if**
13. **end for**
14. **return** CPC
15. **end procedure**
16. **procedure** $MaxMinHeuristic(T, CPC)$
・**Input:** ターゲット・ノードT, CPC
・**Output:** 最小関連度を最大にするノード
17. $assocF = \max_{X \in V} MinAssoc(X, T \mid CPC)$
18. $F = \arg\max_{X \in V} MinAssoc(X, T \mid CPC)$
19. **return** $< F, assocF >$
20. **end procedure**

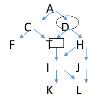
動作例



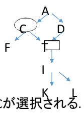
(a)真の構造



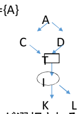
(b)　Aが選択される.CI(T,B)=1,CI(T,E)=1となり，BとEが消去される.
CPC={A}



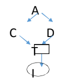
(c)　Dが選択される.
CPC={A,D}



(d)　Cが選択される.
CI(T,H|D)=1, CI(T,J|D)=1より，HとJが消去される.
CPC={A,C, D}



(e)　Iが選択される.
CI(T,F|C)=1 より，Fが消去される.
CPC={A,C, D,I}



(e)　CI(T,K|I)=1, CI(T,L|I)=1より，K,Lが消去される．残り変数がないので,
CPC={A,C, D,I}を

しかし，残念なことにこのアルゴリズム *32* はまだ完全ではない．例えば，図 *7.4* の構造の場合，アルゴリズム *33* は，正しく PC_T を返さない．

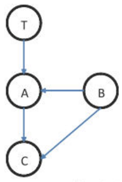


図 **7.4**　MMPC が誤った結果を出力する構造

図 *7.4* の構造の場合，A を所与としても C も条件付き従属 C となるためにこの現象は起こる．

アルゴリズム ***34***　　改良 **MMPC** アルゴリズム:**RMMPC**(T,X)

- **Input:** N ノード変数，ターゲット・ノード T，データ $\mathbf{X}$
- **Output:** ターゲット・ノード T の親ノード集合と子ノード集合

main

1. CPC=$MMPC(T,X)$
2. **for** $C \in CPC$ **do**
3. **if** $T \in MMPC(C,X)$ でない **then**
4. $CPC = CPC \setminus C$
5. **if** $assocF \neq 0$ **then**
6. $CPC = CPC \cup F$
7. **end if**
8. **end for**
9. **return** CPC
10. **end procedure**

アルゴリズム ***35***　　**MMHC** アルゴリズム:**MMHC**(X)

- **Input:** N ノード変数 $V = \{X_1, \ldots, X_N\}$，データ $\mathbf{X}$
- **Output:** ベイジアン・ネットワーク構造 G

main

1. $E = \emptyset$
2. **for** $X_i \in \{X_1, \ldots, X_N\}$ **do**
3. $PC_{X_i} = MMPC(X_i, X)$
4. **end for**
5. **for** $X_i \in V = \{X_1, \ldots, X_N\}$ **do**
6. **for** $X_j \in PC_{X_i}$ **do**
7. 3 構造 $X_j \to X_i$, $X_j \leftarrow X_i$, $X_j \perp X_i$ の $BDeu$ スコアを計算
8. **if** $X_j \to X_i$ が $BDeu$ スコアの値が最も高い
9. **then** $X_j \to X_i$ を E に追加
10. **end for**
11. **end for**
12. **return** $G = (V, E)$
13. **end procedure**

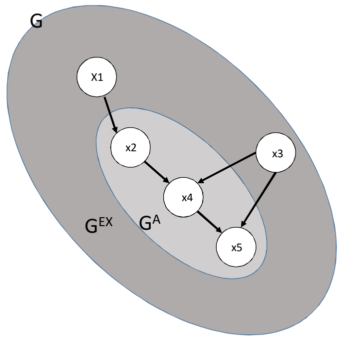
5. RAI

PC アルゴリズムや *MMHC* アルゴリズムでは，*N-2* 個のノードを所与とした条件付き独立性テストまでを繰り返し，ベイジアン・ネットワークのマルコフ・グラフを得たあとに辺の方向づけを行う．しかし，高次の条件付き独立性テスト (例えば，*N-2* 個のノードを所与とした場合) は，低次のそれにくらべて信頼性が非常に低くなることが問題である．従来の手法では，信頼性の低い条件付きテストの結果も高いものと同等に信頼し，辺の方向づけを行うので方向づけの順序によっては結果は非常に不安定になるという問題がある．そこで，信頼性の高い低次の条件付き独立性テストから辺の方向づけを行うことにより，学習の信頼性を高めようという手法が提案されている．それが **RAI アルゴリズム** *(Recursive Autonomy Identification algorithm: Yehezkel and Lerner(2009))* である．

$\mathbf{Pa}(X,G)$ をグラフ g に基づく変数 X の親ノード集合，$\mathbf{Pa}_p(X,G)$ をグラフ g に基づく変数 X の潜在的な親ノード集合 *(*後の学習で変化する可能性がある*)*，$\mathbf{Adj}(X,G)$, $\mathbf{Ch}(X,G)$ をそれぞれ，グラフ G に基づく変数 X の隣接ノード集合と子ノード集合とする．ここで，$\mathbf{Pa}_p(X,G) = \mathbf{Adj}(X,G) \setminus \mathbf{Ch}(X,G)$ となる．

定義 *93* $\mathbf{V}' \subset \mathbf{V}$ *and* $\mathbf{E}' \subset \mathbf{E}$ とする．このとき，$Y \notin \mathbf{V}'$ かつ $\forall X \in \mathbf{V}'$ で，$Y \in \mathbf{Pa_p}(X,G)$ もしくは $Y \notin \mathbf{Adj}(X,G)$ のとき，ノード $Y \in G = (\mathbf{V},\mathbf{E})$ はグラフ $G' = (\mathbf{V}',\mathbf{E}')$ の**外生因果** *(exogenous cause)* である．

定義 *94* *DAG* $G = (\mathbf{V},\mathbf{E})$ において，$\forall X \in \mathbf{V^A}$ かつ $\mathbf{Pa_p}(X,G) \subset \{\mathbf{V^A} \cup \mathbf{V_{ex}}\}$ となるとき，$\mathbf{V^A} \subset \mathbf{V}$ かつ $\mathbf{E^A} \subset \mathbf{E}$ となる 部分グラフ $G^A = (\mathbf{V^A},\mathbf{E^A})$ は，G^A の外生因果の部分集合 $\mathbf{V_{ex}} \subset \mathbf{V}$ を所与として，G において**自律的** *(autonomous)* であるという．もし，$\mathbf{V_{ex}}$ が空集合のとき，この部分集合を（完全な:*completely*）**自律的部分構造** *(autonomous sub-structure)* という．



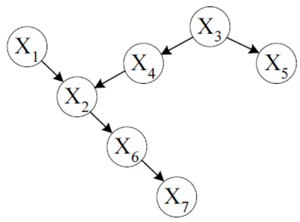
アルゴリズム *36* RAI アルゴリズム:RAI(X)

- **Input:** N ノード変数 $V = \{X_1,\ldots,X_N\}$，データ X
- **Output:** ベイジアン・ネットワーク構造 G

main $G_{out} = \mathrm{RAI}[Ns, G_{start}(\mathbf{V}_{start},\mathbf{E}_{start}), G_{ex}(\mathbf{V}_{ex},\mathbf{E}_{ex}), G_{all}]$

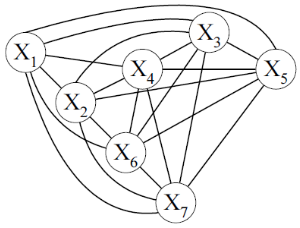
1. *if* G_{start} の全てのノードが $Ns+1$ より少ない潜在的な親ノード数を持つ *then* $G_{out} = G_{all}$ return
2. for $Y \in G_{start}, X \in G_{ex}$ do
3. *if* $X \perp Y \mid \mathbf{S}$ となるとき，$\exists \mathbf{S} \subset \mathrm{Pa_p}(Y,G_{start}) \cup \mathrm{Pa}(Y,G_{ex})$ かつ $|\mathbf{S}| = Ns$ *then* X,Y 間の辺を G_{all} から削除
4. end for
5. オリエンテーション・ルールに従い，G_{start} 中の辺を方向付け
6. for $Y \in G_{start}, X \in G_{start}$ do
7. *if* $X \perp Y \mid \mathbf{S}$ となるとき，$\exists \mathbf{S} \subset \mathrm{Pa}(Y,G_{ex}) \cup \mathrm{Pa_p}(Y,G_{start}) \setminus X$ かつ $|\mathbf{S}| = Ns$ *then* X,Y 間の辺を G_{all} と G_{start} から削除
8. **end for**
9. オリエンテーション・ルールに従い，G_{start} 中の辺を方向付け
10. 最も低次のトポロジカル・オーダーを持つノード集合を子孫部分構造 G_D としてクループ化
11. 逐次 G_{start} から G_D を削除し，その結果の無向グラフを先祖部分構造 $G_{A_1},\ldots,G_{A_k}$ として定義
12. **for** $i = 1$ *to* k **do**
13. *Call* $\mathbf{RAI}[Ns+1, G_{A_i}, G_{ex}, G_{all}]$
14. **end for**
15. $G_{ex_D} = \{G_{A_1},\ldots,G_{A_k},G_{ex}\}$ を G_D への外生因果集合とする
16. *Call* $\mathbf{RAI}[Ns+1, G_D, G_{ex_D}, G_{all}]$
17. **return** $G_{out} = G_{all}$
18. **end procedure**

RAI (Recursive Autonomy Identification)
[Yehezkel+ 09]



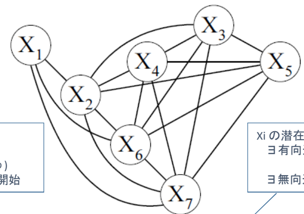
真の構造

RAI (Recursive Autonomy Identification)
[Yehezkel+ 09]



0. 初期状態は完全グラフ

RAI (Recursive Autonomy Identification)
[Yehezkel+ 09]

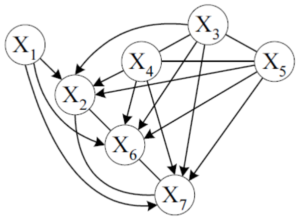


n = 0
すなわち
Ind(Xi;Xj| φ)
からテスト開始

Xi の潜在的な親変数集合
∃有向辺 Xp->Xi ⇒ Xp
or
∃無向辺 Xq—Xi ⇒ Xq

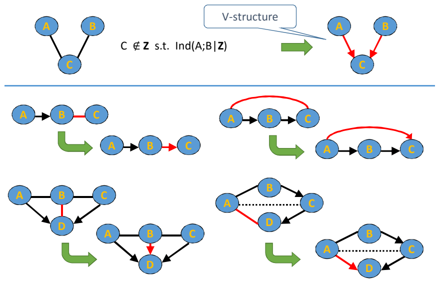
1. すべての辺 Xi—Xj について
Ind(Xi;Xj | **Z**) s.t. **Z** ⊆ **Z’** = {z| z ∈ pp(Xi) ∪ pp(Xj) – {Xi,Xj}}
and |**Z**| = n
⇒ 辺 Xi—Xj を削除

RAI (Recursive Autonomy Identification)
[Yehezkel+ 09]

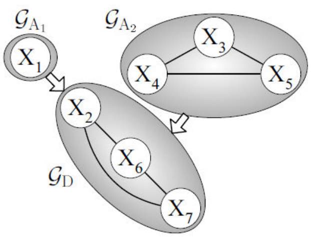


2. 方向付け(後述)を行う.

方向付け(オリエンテーションルール)
[Verma+ 92, Meek 95]



RAI (Recursive Autonomy Identification)
[Yehezkel+ 09]

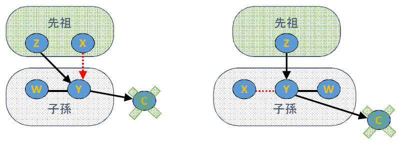


3. 子孫グラフと先祖グラフに分割

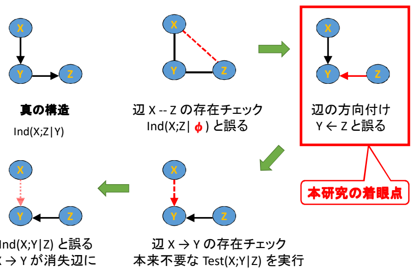
RAI: 親変数(のみ)を所与とした CI テスト

補題 (Pearl 88; Spirtes et al. 00)
DAG において, X と Y が非隣接かつ X が Y の子孫でないとき,
X と Y は Y の親変数集合を所与として有向分離.

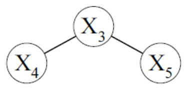
定理 (Yehezkel et al. 09)
DAG $G^A = (\mathbf{V}^A, \mathbf{E}^A)$ が $G = (\mathbf{V}, \mathbf{E})$ の自律部分構造で G^A の外生因子集合 $\mathbf{V}_{ex} \subset \mathbf{V}$ が与えられ, Ind(X;Y|**S**) (s.t. X,Y $\in \mathbf{V}^A$, $\mathbf{S} \subset \mathbf{V}$) ならば, $\mathbf{S}' \subset \{\mathbf{V}^A \cup \mathbf{V}_{ex}\}$ かつ Ind(X;Y|**S'**) なる **S'** が存在する.



RAI の問題点: 消失辺発生メカニズム

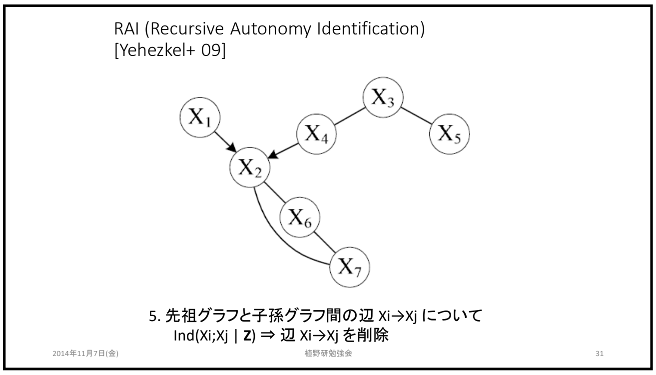


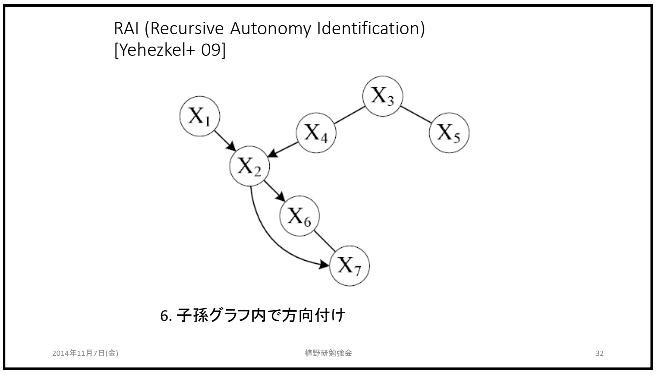
RAI (Recursive Autonomy Identification)
[Yehezkel+ 09]

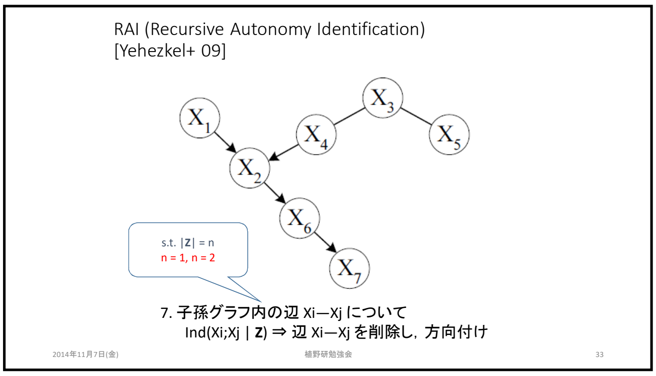


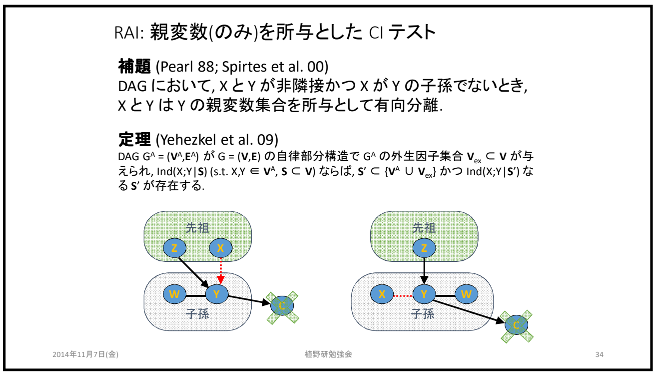
4. 先祖グラフに RAI を再帰的に適用する.
   1. ∀辺 Xi—Xj, Ind(Xi;Xj | **Z**) ⇒ 辺 Xi—Xj を削除
   2. 方向付け
   3. 子孫グラフと先祖グラフに分割
   …

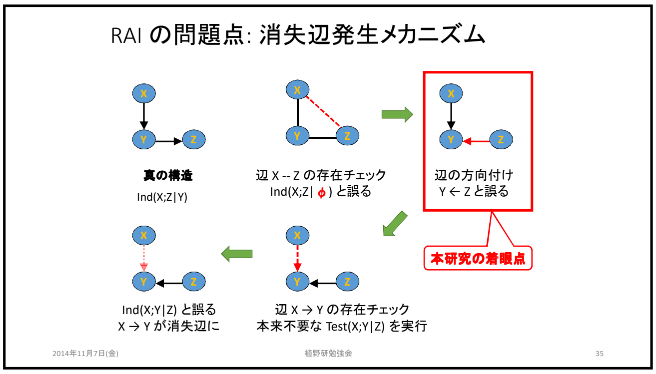
s.t. |**Z**| = n
n = 1

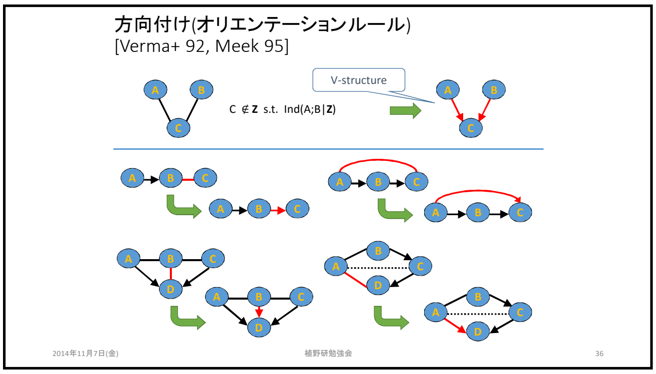

RAI (Recursive Autonomy Identification)
[Yehezkel+ 09]
X1 X2 X3 X4 X5 X6 X7
5. 先祖グラフと子孫グラフ間の辺 Xi→Xj について
Ind(Xi;Xj | Z) ⇒ 辺 Xi→Xj を削除
2014年11月7日(金)
植野研勉強会
31


RAI (Recursive Autonomy Identification)
[Yehezkel+ 09]
X1 X2 X3 X4 X5 X6 X7
6. 子孫グラフ内で方向付け
2014年11月7日(金)
植野研勉強会
32

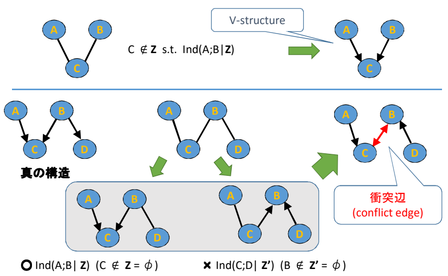

RAI (Recursive Autonomy Identification)
[Yehezkel+ 09]
X1 X2 X3 X4 X5 X6 X7
s.t. |Z| = n
n = 1, n = 2
7. 子孫グラフ内の辺 Xi—Xj について
Ind(Xi;Xj | Z) ⇒ 辺 Xi—Xj を削除し，方向付け
2014年11月7日(金)
植野研勉強会
33


RAI: 親変数(のみ)を所与とした CI テスト
補題 (Pearl 88; Spirtes et al. 00)
DAG において, X と Y が非隣接かつ X が Y の子孫でないとき,
X と Y は Y の親変数集合を所与として有向分離.
定理 (Yehezkel et al. 09)
DAG GA = (VA,EA) が G = (V,E) の自律部分構造で GA の外生因子集合 Vex ⊂ V が与えられ, Ind(X;Y|S) (s.t. X,Y ∈ VA, S ⊂ V) ならば, S′ ⊂ {VA ∪ Vex} かつ Ind(X;Y|S′) なる S′ が存在する.
先祖
子孫
Z X W Y C
2014年11月7日(金)
植野研勉強会
34


RAI の問題点: 消失辺発生メカニズム
真の構造
Ind(X;Z|Y)
辺 X -- Z の存在チェック
Ind(X;Z| φ) と誤る
辺の方向付け
Y ← Z と誤る
本研究の着眼点
辺 X → Y の存在チェック
本来不要な Test(X;Y|Z) を実行
Ind(X;Y|Z) と誤る
X → Y が消失辺に
2014年11月7日(金)
植野研勉強会
35


方向付け(オリエンテーションルール)
[Verma+ 92, Meek 95]
V-structure
C ∉ Z s.t. Ind(A;B|Z)
A B C D
2014年11月7日(金)
植野研勉強会
36

方向付けの誤り検出



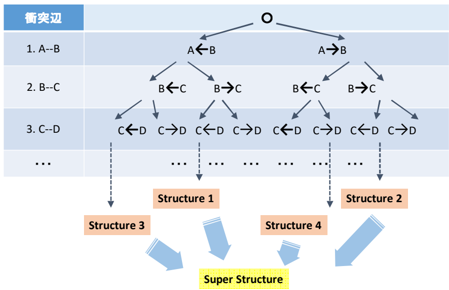
CORAI (Concurrently Oriented RAI)
提案アルゴリズム

考え方：衝突辺の各方向付けを可能な限り探索，結果を合成することで消失辺を抑制する
.

アルゴリズム

1. 衝突辺解決ルールの異なる複数の RAI を並行実行する.
   - ✓ 各 RAI は，衝突辺ごとに向きをランダムに固定する.
2. 各 RAI の辺集合の和をスーパーストラクチャとして出力

CORAI：衝突辺可能解の並行探索

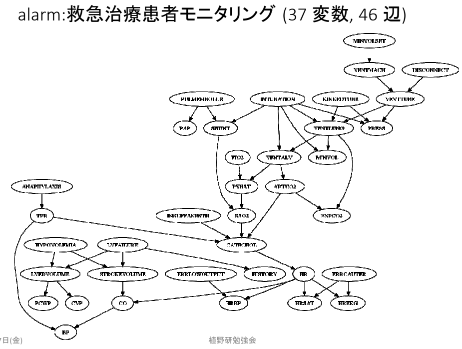


実験

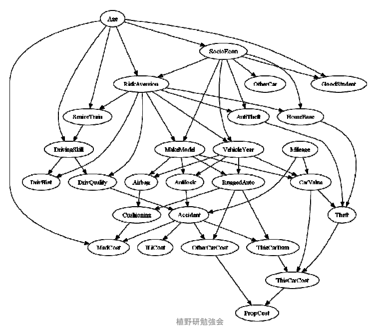
- 比較対象: MMPC, RAI, PC
- G 検定：有意水準を 0.01 から 0.95 まで変化
- 10,000 件のサンプルを 10 セットずつ学習
- 有意水準 0.05 で SS 学習後, SS 内で exact 学習 (宇都さん実装の DP [Malone+11])

ネットワーク	変数	辺	親変数最大	変数状態数
alarm	37	46	4	2-4
insurance	27	52	3	2-5
water	32	66	5	3-4
win95pts	76	112	7	2-2

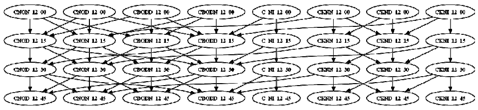
alarm:**救急治療患者モニタリング** (37 **変数**, 46 **辺**)



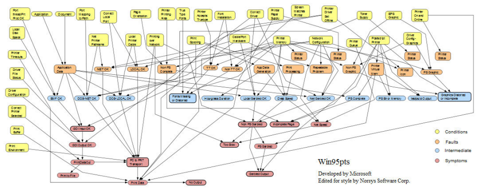
insurance:**保険加入申請評価** (27 **変数**, 52 **辺**)



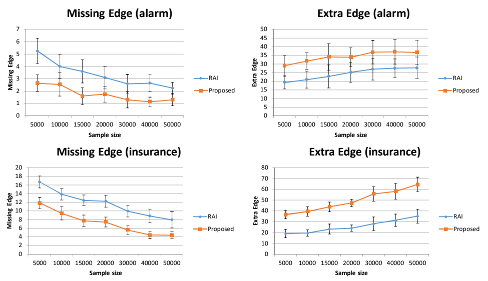
water:浄水設備のバイオロジカルプロセスモデル
(32 変数, 66 辺)



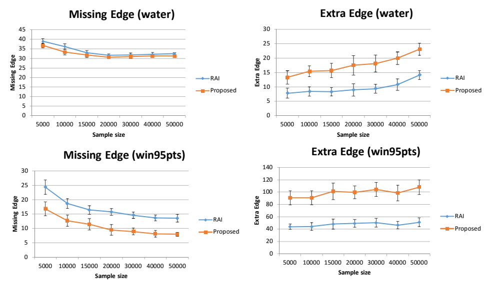
win95pts: Windows 95 プリンタ
トラブルシューティングモデル (76 変数, 112 辺)



実験1: RAI との精度比較
alarm, insurance



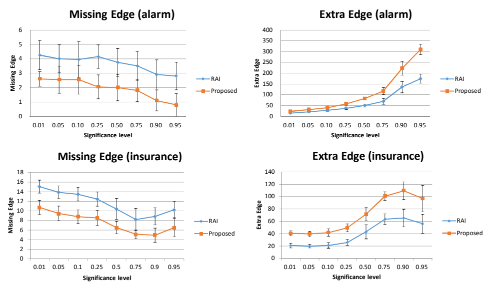
実験1: RAI との精度比較
water, win95pts



実験2:有意水準ごとの消失辺, 余剰辺

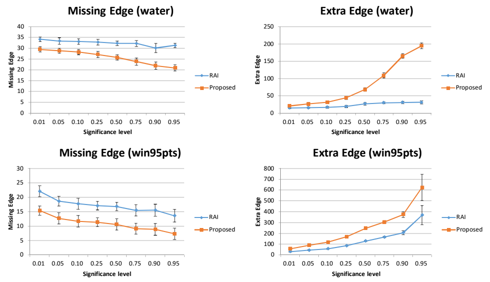
- Q: 既存手法(RAI)でも検定の有意水準を上げれば消失辺は減るのでは?
- 有意水準を 0.01 から 0.95 まで変化
- 10,000 件のサンプルを各 20 セット学習し, 消失辺と余剰辺の数を比較

実験2: 有意水準ごとの消失辺, 余剰辺
alarm, insurance



実験2: 有意水準ごとの消失辺, 余剰辺

water, win95pts



結論

1. 少ない CI テストで高精度とされる RAI [Yehezkel+09] の消失辺発生メカニズムを示した.
2. 消失辺の原因となる方向付けの誤りを衝突辺の発生により検知できることを示した.
3. 消失辺を, 衝突辺の可能解並行探索により抑制するスーパーストラクチャ構築アルゴリズム CORAI を提案した.
4. 比較評価実験により, 以下を示した.
   1. CORAI は, 従来より消失辺を削減.
   2. CORAI は, 検定の有意水準を上げるより消失辺を削減でき, 余剰辺も少なく抑えられる.
   3. CORAI による SS は, 従来手法より高精度に BN を推定

研究の現況と先行研究

- 既存 Exact 学習の学習可能変数数(例えば[Malone+11]の33個)に届いていない.
  - 宇都さん作 SS 上の Exact スコア学習 (Dynamic Programming [Malone+11]) で実行できず.
- H2PC [Gasse+12]:
  - SS 消失辺少ない(対MMPC)
  - スコアはヒューリスティック
  - [Gasse+12] Maxime Gasse et al., An Experimental Comparison of Hybrid Algorithms for Bayesian Network Structure Learning, In proc of ECMLPKDD 2012.

リソース

- ソースコード
  - コアアルゴリズム : Java
    - CORAI (RAI)
    - UtoDP
  - 実験スクリプト : Ruby, sh
  - 動作環境 : Windows 上の Cygwin
- 先行研究の実験コード
  - MMPC・MMHC, PC: Matlab
  - H2PC: R

参考文献

- [Bouchaert 94] Bouchaert et al., Probabilistic Network Construction Using the Minimum Description Length Principle, in Proc. of SQUARAU, pp.41-48, 1994.
- [Cheng+02] Cheng et al., Learning Bayesian networks from data: An information-theory based approach, *Artificial Intelligence*, Vol.137, pp.43–90, 2002.
- [Cussens 11] Cussens, Bayesian network learning with cutting planes, In Proc. of UAI'11, 2011.
- [Heckerman+95] Heckerman et al, Learning Bayesian Networks: The Combination of Knowledge and Statistical Data, Machine Learning, Vol.20, No.3, 1995.
- [Jaakkola+ 10] Jaakkola et al., Learning Bayesian Network Structure using LP Relaxations, In Proc. of AISTATS'10, 2010.
- [Jensen+07] Jensen et al., *Bayesian Networks and Decision Graphs, 2nd Ed.*, Springer, 2007.
- [Koivisto+ 04] Koivisto et al., Exact Bayesian Structure Discovery in Bayesian Networks, Journal of Machine Learning Research, Vol.5, pp.549-573, 2004.
- [Malone+ 11] Malone et al., Improving the Scalability of Optimal Bayesian Network Learning with External-Memory Frontier Breadth-First Branch and Bound Search, In Proc. of UAI'11, 2011.
- [Silander+ 06] Silander et al., A Simple Approach for Finding the Globally Optimal Bayesian Network Structure, In Proc. of UAI'06, 2006.
- [Spirtes+00] Sprites et al., *Causation, Prediction, and Search, 2nd Ed.*, Springer-Verlag, 2000.
- [Suzuki 93] Suzuki, A Construction of Bayesian Networks from Databases on an MDL Principle, in Proc. of UAI'93, pp.266-273, 1993.
- [Tamada+ 11] Tamada et al., Parallel Algorithm for Learning Optimal Bayesian Network Structure, Journal of Machine Learning Research, Vol.12, pp.2437-2459, 2011.
- [Tsamardinos+06] Tsamardinos et al., The max-min hill-climbing Bayesian network structure learning algorithm, Machine Learning, Vol.65, pp.31-78, 2006.
- [Ueno 11] Ueno, Robust learning Bayesian networks for prior belief, in Proc. of UAI'11, 2011.
- [Yehezkel+ 09] Yehezkel et al., Bayesian Network Structure Learning by Recursive Autonomy Identification, Journal of Machine Learning Research, Vol.10, pp.1527-1570, 2009.
- [Yuan+ 11] Yuan et al., Learning Optimal Bayesian Networks Using A* Search, In Proc. of IJCAI'11, 2011.

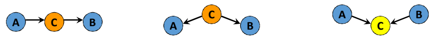
バックアップ
スライド

ベイジアンネットワーク
(Bayesian networks) [Pearl88] の定義

- ベイジアンネットワーク $\mathcal{N} = (G, \mathcal{P})$ は非循環グラフ $G = (V, E)$ と，条件付確率分布 $\mathcal{P}$ によって構成される．
- G の頂点集合 V の各要素は，$\mathcal{P}$ の各確率変数に1対1対応する．
- $\mathcal{N}$ は，V 上の結合確率分布 $P(V)$ を次式のように導入する．ここでは，$pa(X)$ は X の親頂点にあたる変数の集合である．

$$P(V) = \prod_{X \in V} P(X|pa(X))$$

ベイジアンネットワークの性質
有向分離（d-separation）



逐次結合 (serial connection)　分岐結合 (diverging connection)　収斂結合 (converging connection)

- 変数 A, B は，変数間のすべてのパスについて，中間変数 C があり，かつ以下のどちらかの場合に 有向分離しているという．
  - パスが逐次結合または分岐結合で，中間変数 C がインスタンス化されている．
  - パスが収斂結合で，中間変数 C および C の子孫がインスタンス化されていない．
- 変数 A, B が 有向分離されていれば，A の確からしさの変更は，B の確からしさに影響を与えない．

7月26日発売予定　共立出版

鈴木 譲 (編著), 植野 真臣 (編著)

黒木 学 (著), 清水 昌平 (著),湊 真一 (著), 石畠 正和 (著), 樺島 祥介 (著), 田中 和之 (著), 本村 陽一(著), 玉田 嘉紀 (著)