論　文

推移性を利用した大規模ベイジアンネットワーク構造学習

本田　和雅[†a)]　名取　和樹[†b)]　菅原　聖太[†c)]　磯崎　隆司[†,††d)]
植野　真臣[†e)]

Learning Huge Bayesian Network Structures Using the Transitivity

Kazunori HONDA[†a)], Kazuki NATORI[†b)], Shouta SUGAHARA[†c)],
Takashi ISOZAKI[†,††d)], and Maomi UENO[†e)]

あらまし　ベイジアンネットワークは同時確率分布の数学的な分解により高精度な予測を実現できることが知られている．しかし，ベイジアンネットワーク構造学習は膨大な計算時間を要するため，大規模変数に対して構造学習は難しい．近年の研究で，制約ベース手法である RAI アルゴリズムの条件付き独立性検定（CI テスト）に Bayes factor を用いることで漸近一致性を有しつつ 1000 変数程度の構造学習が可能になってきた．制約ベース手法では学習のできる限り早期にエッジを削除することができれば学習に要する CI テスト数を削減できることが知られている．本論文はベイジアンネットワークのある二変数の条件付き独立性からその各変数と他変数との条件付き独立性の少なくとも一つを保証できる推移性が成り立つことを明らかにする．更に RAI アルゴリズムにおいて推移性により少なくとも一つの条件付き独立が保証される二組のエッジの CI テストを優先して行い，CI テスト数を大幅に削減できる手法を提案する．複数のベンチマークネットワークと大規模なランダムネットワークを用いたシミュレーション実験により，提案手法がこれまで実現できなかった大規模なベイジアンネットワークを学習できることを示した．

キーワード　確率的グラフィカルモデル，ベイジアンネットワーク構造学習，条件付き独立性，制約ベースアプローチ

1. まえがき

ベイジアンネットワークは，確率変数をノードで表しノード間の依存関係を非循環有向グラフ（Directed Acyclic Graph: DAG）で表現する確率的グラフィカルモデルである．ベイジアンネットワークは，確率構造に DAG を仮定することにより，同時確率分布を条件付き確率の積に分解する．ベイジアンネットワークを利用した確率推論は高い予測精度をもつことから [1]，システムの故障診断や危険予測システム，医療診断システムなど様々な目的で応用されてきた．

ベイジアンネットワークの構造は一般にデータから推定する必要がある．この問題をベイジアンネットワークの構造学習と呼ぶ．

ベイジアンネットワークの構造学習法として，漸近一致性を有する学習スコアを用いて，全ての構造の候補からスコアが最も高い構造を探索する厳密解探索アプローチが従来から用いられてきた．このアプローチは構造の探索数がノード数に対し指数的に増加する NP 困難問題 [2] である．効率的に厳密解を探索するために，動的計画法 [3]～[7]，A^* 探索 [8]，整数計画法 [9] などの従来の人工知能アプローチによる構造学習法が提案されてきたが，未だ 60 ノード程度の構造学習が限界である．

一方，因果モデル分野では，漸近一致性はもたないが，より計算効率の高い構造学習法が提案されている．この手法は制約ベースアプローチと呼ばれ，完全無向グラフに，二ノード間の条件付き独立性検定（Conditional

[†] 電気通信大学大学院情報理工学研究科，調布市
Graduate School of Informatics and Engineering, The University of Electro-Communications, 1–5–1 Chofugaoka, Chofu-shi, 182–8585 Japan
[††]（株）ソニーコンピュータサイエンス研究所，東京都
Sony Computer Science Laboratories, Inc., 3–13–14 Higashigotanda, Shinagawa-ku, Tokyo, 141–0022 Japan
a) E-mail: hondak@ai.lab.uec.ac.jp
b) E-mail: natori@ai.lab.uec.ac.jp
c) E-mail: sugahara@ai.lab.uec.ac.jp
d) E-mail: isozaki@csl.sony.co.jp
e) E-mail: ueno@ai.lab.uec.ac.jp
DOI:10.14923/transinfj.2019JDP7033

Independence test: CI テスト）を適用して学習される無向グラフに対し，オリエンテーションルール [10] によるエッジの方向付けを行うことで DAG を学習する．制約ベースアプローチの研究では，PC アルゴリズム [11]，MMHC アルゴリズム [12]，RAI アルゴリズム [13] が提案されている．従来の CI テストでは漸近一致性をもたないことが問題であったが，近年の研究で，RAI アルゴリズムの CI テストに Bayes factor を用いることで漸近一致性を有する大規模ネットワーク構造学習を実現している [14]〜[16]．この手法では 1000 ノード程度のネットワークを学習できるようになったが，本研究ではより大規模のネットワーク学習を実現できるアルゴリズムの開発を目的とする．

制約ベースアプローチの時間計算量は学習に要する CI テスト数に大きく依存する [11], [13]．一般に，制約ベースアプローチでは，学習のできる限り早期にエッジを削除するほど CI テスト数を削減できる．そこで，本論文は推移性を用いて，より学習の早期に条件付き独立性を検出できる手法を提案する．

無向グラフによってノード間の依存関係を記述するマルコフネットワークの条件付き独立性では推移性が成り立つ [17], [18]．マルコフネットワークの条件付き独立性における推移性は二変数 X と Y が変数集合 $\mathbf{Z}$ を所与として条件付き独立ならば二つの変数対 X と A，A と Y（$A \notin \{X, Y\} \cup \mathbf{Z}$）のうち少なくとも一つは $\mathbf{Z}$ を所与として条件付き独立となる性質である．すなわち，ある条件付き独立性を検出したとき，推移性を利用することで少なくとも一つの条件付き独立が保証される二組のエッジの CI テストを列挙できる．しかし，ベイジアンネットワークの条件付き独立性でマルコフネットワークと同様の推移性は成り立たない．例えば図 1 の DAG において，X と Y は $\{Z\}$ を所与として条件付き独立であるが，X と A，A と Y はともに $\{Z\}$ を所与として条件付き独立でない．

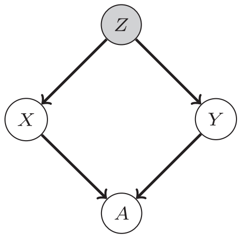

図 1　4 ノードの DAG
Fig. 1　DAG with 4 nodes.

Pearl はベイジアンネットワークの条件付き独立性で推移性を緩和した弱推移性が成り立つことを示した [18]．弱推移性は二変数 X と Y が変数集合 $\mathbf{Z}$ と $\mathbf{Z} \cup \{A\}$（$A \notin \{X, Y\} \cup \mathbf{Z}$）のどちらを所与としても条件付き独立であるならば二つの変数対 X と A，A と Y の少なくとも一つは $\mathbf{Z}$ を所与として条件付き独立となる性質である．この手法は既に制約ベースアプローチに利用され，学習精度向上に寄与している [19]．弱推移性は前述の推移性と同様に，条件部である二つの条件付き独立性を検出できれば少なくとも一つの新たな条件付き独立性の検出を保証できる．しかし，新たに条件付き独立となる候補変数 A を同定するために，$\mathbf{Z} \cup \{A\}$ を所与とした X と Y の条件付き独立性を検出しなければならず，従来の制約ベースアプローチより CI テスト数が増加してしまう．例えば図 1 において，X と Y の $\{Z\}$ を所与とした条件付き独立性を検出したとき，A が候補変数か判定するために $\{Z, A\}$ を所与とした CI テストを行わなければならない．すなわち，A が候補変数でないために弱推移性で新たな独立性を検出できないだけでなく，CI テスト数だけが増加してしまう．このように弱推移性は CI テストの精度向上には貢献するが，CI テスト数の削減は期待できない．

本論文では，二変数の条件付き独立性から各変数と条件付き独立となる他変数の候補空間を制約することで，ベイジアンネットワークでも推移性が成り立つことを示す．この候補空間はネットワークサイズに対して線形時間で探索できるため，従来の制約ベースアプローチより CI テスト数は増加しない．ベイジアンネットワークの推移性を制約ベースアプローチに利用することで少なくとも一つの条件付き独立が保証される二組のエッジの CI テストを優先して実施できる．

提案手法では，推移性を用いるために事前にエッジを方向付ける必要がある．それゆえ，無向グラフ上で CI テストを行う PC アルゴリズムや MMHC アルゴリズムは推移性を利用できない．一方で，RAI アルゴリズムは学習途中でエッジを方向付けるため，推移性を利用できる．そこで，本論文は現在最も大規模の構造学習を実現できる Bayes factor を用いた RAI アルゴリズム [14]〜[16] をベースに改良し，漸近一致性を有する新しいアルゴリズムを提案する．

提案手法は以下の利点がある．

（1） 従来の制約ベースアプローチと同等の精度を保ちつつ，学習に要する CI テスト数と計算時間をよ

り削減する．

（2） これまで実現できなかった大規模のベイジアンネットワークを厳密学習できる．

複数のベンチマークネットワークとランダムに生成した大規模ネットワークを用いたシミュレーション実験によって提案手法と従来の制約ベースアプローチを比較した．これにより，以下の有効性が示された．

（1） 30 ノードを超える中規模から大規模のネットワークにおいて，提案手法は従来の制約ベースアプローチの精度を落とさず，学習に要する CI テスト数と計算時間をより削減できる．

（2） 1000 ノードを超える大規模ネットワークにおいて，提案手法は学習速度と学習精度を大幅に向上して学習でき，従来では実現できなかった 3500 ノードの大規模ネットワーク構造学習を実現できる．

2. ベイジアンネットワーク学習

ベイジアンネットワークは，確率変数をノードとし，ノード間の依存関係を非循環有向グラフ（Directed Acyclic Graph: DAG）と各ノードの条件付き確率で表現する確率的グラフィカルモデルである．

今，$\mathbf{V} = \{X_1, \ldots, X_n\}$ を n 個の離散確率変数に対応するノード集合とし，各ノード X_i は r_i 個の状態集合 $\{1, \ldots, r_i\}$ から一つの値 k を取る（$X_i = k$ と書く）とする．このとき，ベイジアンネットワーク構造 G において，各ノード X_i の親ノード集合を $\mathbf{Pa}(X_i, G)$ としたとき，同時確率分布 $P(X_1, \ldots, X_n)$ は以下のように表現できる．

$$P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i \mid \mathbf{Pa}(X_i, G)). \quad (1)$$

ノード X からノード Y までの隣接ノードの連なりを二ノード X，Y を結ぶ道と言う．道上で，三ノード A，B，C が $A \to C \to B$，$A \leftarrow C \to B$，$A \to C \leftarrow B$ と結合することをそれぞれ，ノード C で逐次結合，分岐結合，合流結合すると言う．ベイジアンネットワーク構造は二ノードを結ぶ道をブロックすることで条件付き独立性を表現する．ブロックを以下で定義する．

［定義 2.1］ 二ノード X，Y を結ぶ道 p が以下のいずれかの条件を満たすとき，道 p はノード集合 $\mathbf{Z}$ でブロックされる．

（1） 道 p がノード $Z \in \mathbf{Z}$ で逐次結合か分岐結合をする．

（2） 道 p がノード $Z \notin \mathbf{Z}$ で合流結合をし，かつ Z の子孫が $\mathbf{Z}$ に属さない．

二ノード X，Y を結ぶ全ての道がノード集合 $\mathbf{Z}$ でブロックされるとき，ネットワーク構造 G において X，Y は $\mathbf{Z}$ を所与として条件付き独立であり，$X \perp Y \mid \mathbf{Z}$ と表す．ネットワーク構造における条件付き独立性は合成性や弱結合性，弱推移性などの性質を満たす [18]．

ベイジアンネットワークの構造学習では漸近一致性を有する周辺ゆう度スコアを最大化する構造を探索することが一般的である．今，条件付き確率パラメータ集合 $\Theta = \{\theta_{ijk}\}$，$(i = 1, \ldots, n,\ j = 1, \ldots, q_i,\ k = 1, \ldots, r_i)$ の事前分布として，以下のディレクレ分布 $P(\Theta)$ を仮定する．

$$P(\Theta) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma\left(\sum_{k=1}^{r_i} \alpha_{ijk}\right)}{\prod_{k=1}^{r_i} \Gamma(\alpha_{ijk})} \prod_{k=1}^{r_i} \theta_{ijk}^{\alpha_{ijk}-1}. \quad (2)$$

ここで，q_i はノード X_i の親ノード集合 $\mathbf{Pa}(X_i, G)$ の取り得るパターン数を表し，α_{ijk} はディレクレ事前分布のハイパーパラメータを表す．ノード集合 $\mathbf{V}$ に対する N 個のデータを $\mathbf{D} = \{D_1, \ldots, D_N\}$ とするとき，周辺ゆう度スコアは次式で表される．

$$\begin{aligned} P(\mathbf{D} \mid G, \boldsymbol{\alpha}) &= \int_{\Theta} P(\mathbf{D} \mid \Theta, G) P(\Theta) d\Theta \\ &= \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})}. \quad (3) \end{aligned}$$

ここで，$\boldsymbol{\alpha} = \{\alpha_{ijk}\}$，$(i = 1, \ldots, n, j = 1, \ldots, q_i, k = 1, \ldots, r_i)$ であり，$\alpha_{ij} = \sum_{k=1}^{r_i} \alpha_{ijk}$ である．また，N_{ijk} はノード X_i の親ノード集合 $\mathbf{Pa}(X_i, G)$ が j 番目のパターンを取るときの $X_i = k$ となる頻度を表し，$N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$ である．データ数 N は $N = \sum_{j=1}^{q_i} N_{ij}$，$(i = 1, \ldots, n)$ となる．近年では，$\alpha_{ijk} = \alpha/(r_i q_i)$ とした Bayesian Dirichlet equivalent uniform（BDeu）が最も用いられる [20], [21]．ここで，α は Equivalent Sample Size (ESS) と呼ばれる事前知識の重みを示す擬似サンプルである．

一般にこの構造学習法は，厳密解探索アプローチと呼ばれる．しかし，このアプローチによる構造学習は NP 困難であり，ノード数の増加に伴い，計算量が爆発的に増加してしまう．厳密解を効率的に探索するために，動的計画法 [3]〜[7]，A^* 探索 [8]，整数計画法 [9] といった従来の探索手法を用いた構造学習法が提案さ

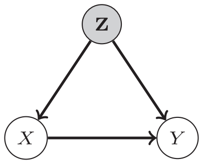

図 2　従属モデル G_1
Fig. 2　Dependent model.

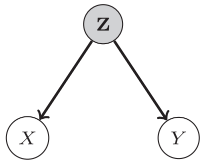

図 3　独立モデル G_2
Fig. 3　Independent model.

れてきた．しかし，最先端手法 [9] でさえ 60 ノード程度の構造学習が限界であり，大規模ネットワークを学習できない．

一方，因果モデル分野では，大幅に計算量を削減できる制約ベースアプローチと呼ばれる構造学習法が提案されてきた．このアプローチの基本的なアルゴリズムは以下のとおりである．

（1）　完全無向グラフを生成する．

（2）　(1) で生成された完全無向グラフに対し条件付き独立性検定（Conditional Independence test: CI テスト）によりエッジを削除する．

（3）　(2) で得られた無向グラフに対してオリエンテーションルール [10] を用いて方向付けを行う．

一般に，制約ベースアプローチの学習精度は CI テストの精度に依存し，学習速度は学習に要する CI テスト数に依存する．

制約ベースアルゴリズムとして，PC アルゴリズム [11]，MMHC アルゴリズム [12]，RAI アルゴリズム [13] が提案されてきた．しかし，これらのアルゴリズムでは χ^2 検定，G^2 検定，条件付き相互情報量を CI テストに用いるため，漸近一致性をもたない．

一方で，Steck らは，二変数間が独立・従属モデルの周辺ゆう度の比による Bayes factor を用いた漸近一致性を有する CI テストを提案した [22]．例として，X と Y 間について各ノードの共通親ノード集合を $\mathbf{Z}$ としたときの従属なモデルを G_1，独立なモデルを G_2 とし，それぞれ図 2，3 に示す．このときの Bayes factor を $\mathrm{BF}(X, Y \mid \mathbf{Z})$ とすると，対数 Bayes factor は，

$$\log \mathrm{BF}(X, Y \mid \mathbf{Z}) = \log \frac{P(\mathbf{D} \mid G_1, \boldsymbol{\alpha})}{P(\mathbf{D} \mid G_2, \boldsymbol{\alpha})}, \tag{4}$$

と表される．ここで，$P(\mathbf{D} \mid G_1, \boldsymbol{\alpha})$，$P(\mathbf{D} \mid G_2, \boldsymbol{\alpha})$ は式 (3) の BDeu を用いる．Bayes factor を用いた CI テストでは対数 Bayes factor が 0 以上か否かで図 2，3 のどちらを選択するか判定する．しかし，Steck らはこの CI テストを理論分析に用いるだけでベイジアンネットワーク学習には用いていない．

Abellán らは Bayes factor を用いた CI テストを PC アルゴリズムに組み込み，従来の制約ベースアプローチの手法と比較している [23]．しかし，実験結果より，χ^2 検定，G^2 検定を用いた従来の CI テストのほうが Bayes factor を用いた CI テストより学習精度が高かったと報告している．名取らは以下の定理を示すことで，Abellán らの結果はデータ数が少ないときの結果であり，理論的にはデータ数を増やせば Bayes factor を用いた手法が従来手法より高精度となると報告した [14]～[16]．

［定理 2.1］　データ数 $N \to \infty$ のとき，

（1）　真の構造が $\mathbf{Z}$ を所与として X と Y が条件付き独立でないとき，$\log \mathrm{BF}(X, Y \mid \mathbf{Z}) > 0$．

（2）　真の構造が $\mathbf{Z}$ を所与として X と Y が条件付き独立のとき，$\log \mathrm{BF}(X, Y \mid \mathbf{Z}) < 0$．

証明は名取ら [16] を参照してほしい．定理 2.1 より，Bayes factor を用いた CI テストは漸近的に真の条件付き独立性を判定できる．名取らは Bayes factor を用いた CI テストを制約ベースアプローチである RAI アルゴリズムに組み込んだ手法を提案し，これまでにできなかった漸近一致性を有して 1000 変数を超える大規模構造学習を実現した．

本論文では，更に大規模のベイジアンネットワーク学習の実現のために，ある二変数の条件付き独立性を検出すればその各変数と他変数との条件付き独立性の少なくとも一つを保証できる推移性が成り立つことを証明し，名取らの手法 [14]～[16] を拡張する．これにより，CI テスト数を大幅に削減し，従来の制約ベースアプローチでは実現していない大規模のネットワーク学習を実現する．

3. Bayes factor を用いた RAI アルゴリズム

RAI アルゴリズムは，制約ベースアプローチにおいて最初に提案された PC アルゴリズム [11] を改良したものである．PC アルゴリズムでは，$n-2$ 個のノードを所与とした高次の CI テストまで繰り返す．しかし高次の CI テストは，低次のときに比べて信頼性が非常に低くなり，精度が著しく悪化する問題がある．RAI アルゴリズムは，その高次の CI テストを抑えるために開発された学習アルゴリズムである．RAI アルゴリズムは，各次数の CI テスト後にオリエンテーションルールによるエッジの方向付けを行い，その結果を用いて全体グラフを部分グラフに分割する処理を繰り

返すことで構造を学習する．RAI アルゴリズムの CI テストに Bayes factor を用いることで漸近一致性を有する学習アルゴリズムが提案されている [14]〜[16].

今，グラフを $G = (\mathbf{V}, \mathbf{E})$ と表し，$\mathbf{V}$，$\mathbf{E}$ はそれぞれ G に含まれるノード集合，エッジ集合を表す．ここで，G は有向エッジと無向エッジを併せもつとする．また，$\mathbf{Adj}(X, G)$ はグラフ G におけるノード X の隣接ノード集合を表し，$\mathbf{Ch}(X, G)$ はグラフ G におけるノード X の子ノード集合を表す．このとき，$\mathbf{Pa}_p(X, G)$ は $\mathbf{Adj}(X, G) \setminus \mathbf{Ch}(X, G)$ を表し，$\mathbf{Pa}(X, G)$ はグラフ G におけるノード X の親ノード集合を表す．また，$\mathbf{Pa}(X, \mathbf{G})$ はグラフ集合 $\mathbf{G}$ において $\cup_{G \in \mathbf{G}} \mathbf{Pa}(X, G)$ を表す．G の部分構造 $G' = (\mathbf{V}', \mathbf{E}')$ が存在するとき，RAI アルゴリズムのグラフ分割では，以下に定義される外生因果及び自律的部分構造に分割を行う．

［定義 3.1］ Y が $G' = (\mathbf{V}', \mathbf{E}')$ の外生因果 $\Leftrightarrow$ $\forall Y \in \mathbf{V} \setminus \mathbf{V}'$，$\forall X \in \mathbf{V}'$，$Y \in \mathbf{Adj}(X, G) \Rightarrow Y \in \mathbf{Pa}(X, G)$.

［定義 3.2］ G' が自律的部分構造 $\Leftrightarrow$ $\forall X \in \mathbf{V}'$ で，$\mathbf{V}_{ex}$ は G' の外生因果からなる集合とするとき，$\mathbf{Pa}_p(X, G) \subset \mathbf{V}' \cup \mathbf{V}_{ex}$.

CI テストに Bayes factor を用いる RAI アルゴリズムの詳細を Algorithm1 に示す．Algorithm1 では完全無向グラフ G_{uc} とデータ $\mathbf{D}$ を入力として関数 RAI を再帰的に実行することで，学習結果の構造を出力として得る．ここで，関数 RAI 内の $\log \mathrm{BF}(X, Y \mid \mathbf{Z})$ は式 (4) の対数 Bayes factor を表し，0 未満のとき条件付き独立と判定する．また，$\mathbf{V}[i]$ はノード集合 $\mathbf{V}$ の i 番目の要素を，$\mathbf{G}[i]$ はグラフ集合 $\mathbf{G}$ の i 番目の要素を，E_{XY} は XY 間のエッジを表す．関数 RAI の概略は次のとおりである．入力グラフを $G_s = (\mathbf{V}_s, \mathbf{E}_s)$ とし，(1) 各次数の CI テストにおいて $\log \mathrm{BF}(X, Y \mid \mathbf{Z}) < 0$ となり X，Y が条件付き独立と判定されるとき，XY 間のエッジ E_{XY} を削除する（8 行目から 24 行目）.(2)（1）により得られた無向グラフにオリエンテーションルールを適用して方向付ける（25 行目）.(3) 方向付けの結果から自律的部分構造を取り出す．具体的には，$\mathbf{V}_s$ の要素から子ノードをもつ集合 $\mathbf{V}_p$ と子ノードをもたない集合 $\mathbf{V}_c$ を取り出す．ここで，$\mathbf{V}_c$ の要素が $\mathbf{E}_s$ の無向エッジ集合 $\mathbf{E}_U$ の要素を用いて $\mathbf{V}_p$ のいずれかの要素に到達可能 [24] な場合，その要素を $\mathbf{V}_c$ から削除する．また，$\mathbf{E}_U$ の要素のうち $\mathbf{V}_c$ の要素を頂点にもつエッジ集合を $\mathbf{E}_c$ とし，$\mathbf{V}_c$ と $\mathbf{E}_c$ で構成されるグラフを自律的部分構造として G_s から取り出す（26 行目から 37 行目）.(4) G_s から外生因果を構成するノード集合とそのノードを頂点にもつエッジ集合を取り出す．このとき，取り出したノード集合とエッジ集合で定義されるグラフが非連結グラフとなる場合，非連結グラフ内の個々の連結グラフを列挙する．具体的には，$\mathbf{V}_s$ の要素がなくなるまで以下の手順を繰り返す．まず，$\mathbf{V}_s$ の任意の要素 V から到達可能な G_s のノード集合と V の和集合を $\mathbf{V}_e$ とする．次に $\mathbf{E}_s$ において，$\mathbf{V}_e$ の要素を頂点にもつエッジ集合を $\mathbf{E}_e$ とする．$\mathbf{V}_e$ と $\mathbf{E}_e$ で構成されるグラフをグラフ集合 $\mathbf{G}_e$ に追加し，G_s から $(\mathbf{V}_e, \mathbf{E}_e)$ を取り除く.（38 行目から 47 行目）.(5)

Algorithm 1 The RAI algorithm using Bayes factor

```
1: function MAIN(G_uc, D)
     G_uc = (V_uc, E_uc): 完全無向グラフ
     D: データ
2:     return RAI (0, G_uc, φ, G_uc, D)
3: end function

4: function RAI(n_z, G_s, G_ex, G_all, D)
     n_z: CI テストの次数
     G_s = (V_s, E_s): 入力グラフ
     G_ex: 分割されたグラフの集合
     G_all = (V_all, E_all): CI テストと方向付けによって得られる出力グラフ
5:     if 全ての V ∈ V_s について |Pa_p(V, G_all)|<n_z + 1 then
6:         return G_all
7:     end if
     // CI テストによるエッジの削除
8:     for G_ex = (V_ex, E_ex) ∈ G_ex do
9:         for X ∈ V_s, Y ∈ V_ex do
10:            for Z ⊆ Pa_p(X, G_s) ∪ Pa(X, G_ex) \ {Y} do
11:                if |Z| = n_z かつ log BF(X, Y | Z) < 0 then
12:                    E_all ← E_all \ {E_XY}
13:                                                ▷ E_XY: XY 間のエッジ
14:                end if
15:            end for
16:        end for
17:    end for
18:    for X ∈ V_s, Y ∈ V_s do
19:        for Z ⊆ Pa_p(X, G_s) ∪ Pa(X, G_ex) \ {Y} do
20:            if |Z| = n_z かつ log BF(X, Y | Z) < 0 then
21:                E_all ← E_all \ {E_XY}, E_s ← E_s \ {E_XY}
22:            end if
23:        end for
24:    end for
25:    オリエンテーションルールを用いて E_all, E_s を方向付け
     // G_s から自律的部分構造を分離
26:    E_U ← E_s の無向エッジ集合
27:    V_c ← V_s の子ノードをもたないノード集合
28:    V_p ← V_s \ V_c
29:    for i = 1 to |V_c| do
30:        if V_c[i] が E_U の要素を用いて V_p のいずれかの要素に到達可能 then
31:            V_c ← V_c \ V_c[i]
32:        end if
33:    end for
34:    E_c ← E_U において V_c の要素を頂点としてもつエッジ集合
35:    E_s ← E_s \ E_c
36:    V_s ← V_s \ V_c
37:    G_D ← (V_c, E_c)
     //G_s から外生因果を分離
38:    G_e ← φ
39:    i ← 1
40:    for V ∈ V_s do
41:        V_e ← {V} ∪ (V から到達可能な G_s のノード集合)
42:        E_e ← E_s において V_e の要素を頂点としてもつエッジ集合
43:        G_e[i] ← (V_e, E_e)
44:        V_s ← V_s \ V_e
45:        E_s ← E_s \ E_e
46:        i ← i + 1
47:    end for
     // 再帰的に関数 RAI を呼び出す
48:    for i = 1 to |G_e| do
49:        G_all ← RAI(n_z + 1, G_e[i], G_ex, G_all, D)
50:    end for
51:    G_ex ← G_ex ∪ G_e
52:    return RAI(n_z + 1, G_D, G_ex, G_all, D)
53: end function
```

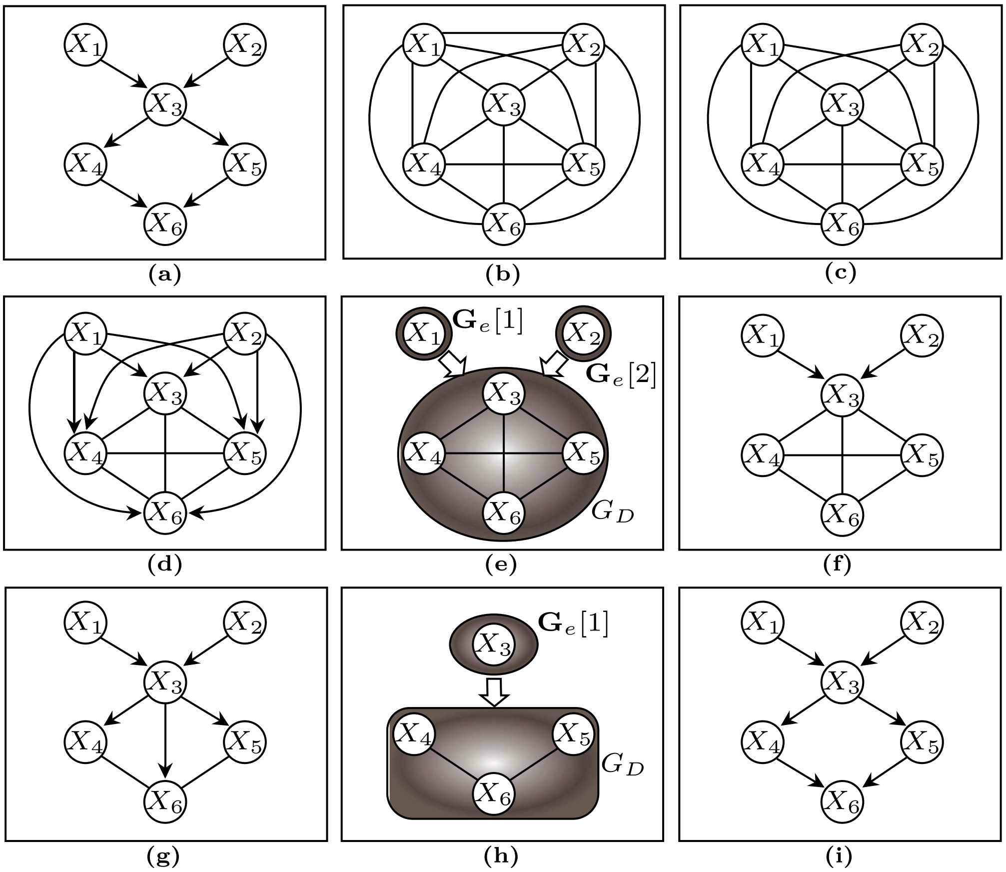

図 4　Algorithm1 の動作例
Fig. 4　A running example of Algorithm1.

各部分グラフで再帰的に RAI を呼び出す（48 行目から 52 行目）．

Algorithm1 の動作例として，ベイジアンネットワークのリポジトリ [25] に登録されている survey（図 4 (a)）を真の構造とした学習過程を示す（図 4）．この例では，Bayes factor を用いた CI テストが真の独立性を判定できると仮定する．2 行目より，CI テストの次数 $n_z = 0$ と完全無向グラフ G_{uc}（図 4 (b)）を引数として関数 RAI を呼び出す．$n_z = 0$ より，18〜24 行目で条件変数集合 $\mathbf{Z}$ を空集合とした 0 次の CI テストを実施する．その結果，$X_1 \perp X_2$ を検出し，エッジ $E_{X_1X_2}$ を削除する（図 4 (c)）．25 行目で図 4 (c) の構造のエッジを方向付ける（図 4 (d)）．27 行目より，図 4 (d) の構造において $\mathbf{V}_s$ から子ノードをもたないノード集合 $\mathbf{V}_c = \{X_3, X_4, X_5, X_6\}$ を得る．35〜37 行目より，$\mathbf{V}_s, \mathbf{E}_s$ から $\mathbf{V}_c$ とそのノード間のエッジ集合 $\mathbf{E}_c$ からなる構造を自律的部分構造 G_D として取り出す（図 4 (e)）．残った $\mathbf{V}_s = \{X_1, X_2\}$ の各要素において，40〜47 行目の処理を X_1, X_2 の順で以下のように行う．X_1 は X_2 に到達可能でないため，41〜43 行目より，$\mathbf{V}_s$, $\mathbf{E}_s$ から $\mathbf{V}_e = \{X_1\}$, $\mathbf{E}_e = \phi$ からなる構造を外生因果 $\mathbf{G}_e[1]$ として取り出す（図 4 (e)）．残った $\mathbf{V}_s$ に含まれるノードは X_2 のみとなるため，41〜43 行目より，$\mathbf{V}_s, \mathbf{E}_s$ から $\mathbf{V}_e = \{X_2\}$, $\mathbf{E}_e = \phi$ からなる構造を外生因果 $\mathbf{G}_e[2]$ として取り出す（図 4 (e)）．次に 48〜50 行目より，$\mathbf{G}_e[1]$, $\mathbf{G}_e[2]$ について CI テストの次数 $n_z + 1 = 1$，$\mathbf{G}_{ex} = \phi$ を引数として関数 RAI を再帰的に呼び出す．$\mathbf{G}_e[1]$, $\mathbf{G}_e[2]$ はそれぞれ X_1, X_2 のみからなる構造であるため，共に 5 行目の終了条件を満たす．52 行目より，G_D について，$n_z + 1 = 1$, $\mathbf{G}_{ex} = \{\mathbf{G}_e[1], \mathbf{G}_e[2]\}$ を引数として関数 RAI を再帰的に呼び出す．$n_z = 1$ より，8〜17 行

目で 1 次の CI テストを実施し，$X_1 \perp X_4 \mid \{X_3\}$，$X_1 \perp X_5 \mid \{X_3\}$，$X_1 \perp X_6 \mid \{X_3\}$，$X_2 \perp X_4 \mid \{X_3\}$，$X_2 \perp X_5 \mid \{X_3\}$，$X_2 \perp X_6 \mid \{X_3\}$ を検出して，$E_{X_1X_4}$，$E_{X_1X_5}$，$E_{X_1X_6}$，$E_{X_2X_4}$，$E_{X_2X_5}$，$E_{X_2X_6}$ を削除する（図 4 (f)）．次に，18〜24 行目の CI テストによって $X_4 \perp X_5 \mid \{X_3\}$ を検出して $E_{X_4X_5}$ を削除する．25 行目で構造 G_{all} のエッジを方向付ける（図 4 (g)）．26〜37 行目より，$\mathbf{V}_s$, $\mathbf{E}_s$ から，$\mathbf{V}_c = \{X_4, X_5, X_6\}$, $\mathbf{E}_c = \{E_{X_4X_6}, E_{X_5X_6}\}$ からなる構造を自律的部分構造 G_D として取り出す（図 4 (h)）．残った $\mathbf{V}_s$ に含まれるノードは X_3 のみとなるため，41〜43 行目より，$\mathbf{V}_s$, $\mathbf{E}_s$ から $\mathbf{V}_e = \{X_3\}$, $\mathbf{E}_e = \phi$ からなる構造を外生因果 $\mathbf{G}_e[1]$ として取り出す（図 4 (h)）．次に，48〜50 行目より，$\mathbf{G}_e[1]$ について，CI テストの次数 $n_z + 1 = 2$, $\mathbf{G}_{ex} = \{(\{X_1\}, \phi), (\{X_2\}, \phi)\}$ を引数として関数 RAI を再帰的に呼び出すが，5 行目の終了条件を満たす．52 行目より，G_D について $n_z + 1 = 2, \mathbf{G}_{ex} = \{(\{X_1\}, \phi), (\{X_2\}, \phi), \mathbf{G}_e[1]\}$ を引数として関数 RAI を再帰的に呼び出す．$n_z = 2$ より，8〜17 行目で 2 次の CI テストを実施し，$X_3 \perp X_6 \mid \{X_4, X_5\}$ を検出して，$E_{X_3X_6}$ を削除する．25 行目で構造 G_{all} のエッジを方向付ける（図 4 (i)）．26〜37 行目より，$\mathbf{V}_s, \mathbf{E}_s$ から $(\{X_6\}, \phi)$ を自律的部分構造 G_D として取り出す．また，38〜47 行目で $\mathbf{V}_s, \mathbf{E}_s$ から，$(\{X_4\}, \phi), (\{X_5\}, \phi)$ をそれぞれ外生因果 $\mathbf{G}_e[1]$, $\mathbf{G}_e[2]$ として取り出す．48〜52 行目より，$\mathbf{G}_e[1]$, $\mathbf{G}_e[2]$, G_D について CI テストの次数 $n_z + 1 = 3$ を引数として関数 RAI を再帰的に呼び出し，5 行目の終了条件を満たすので学習を終了し，図 4 (i) の構造を得る．

今，真のベイジアンネットワークに以下の 3.1 を仮定すると，Bayes factor を用いた RAI アルゴリズムは漸近一致性を有する [14]〜[16]．

［仮定 3.1］ $G = (\mathbf{V}, \mathbf{E})$ を真のベイジアンネットワーク構造とし，$\mathbf{V}$，$\mathbf{E}$ をそれぞれ，G に含まれるノード集合，エッジ集合とする．このとき，G は以下を満たす．

$$\forall X, \forall Y \in \mathbf{V},\ \forall \mathbf{Z} \subset \mathbf{V} \setminus \{X, Y\}\ s.t.\ X \perp Y \mid \mathbf{Z}$$
$$\Leftrightarrow E_{XY} \notin \mathbf{E}.$$

［定理 3.1］ 真のベイジアンネットワークが仮定 3.1 を満たし，データ数 $N \to \infty$ となるとき，Algorithm1 は真のベイジアンネットワーク構造を推定する．

定理 3.1 の証明は名取ら [14]〜[16] を参照してほしい．

名取らの手法 [14]〜[16] はこれまでの制約ベースアプローチの中で最も高精度な大規模構造学習を実現した．本論文では，名取らの手法 [14]〜[16] のエッジ削除に推移性を利用することで漸近一致性を有しつつ，更に大規模の構造学習を実現する．

4. 提 案 手 法

本章では，ベイジアンネットワークにおいて，ある二変数の条件付き独立性を検出すればその各変数と他変数との条件付き独立性の少なくとも一つを保証できる推移性を導入し，それを名取らの手法 [14]〜[16] に利用することで漸近一致性を有しつつ学習に要する CI テスト数を削減できるアルゴリズムを提案する．

4.1 推 移 性

無向グラフによって確率変数間の依存関係を記述するマルコフネットワークの条件付き独立性では以下の推移性が成り立つ [17], [18]．今，$\mathbf{V}$ を n 個の変数集合とし，X，$Y \in \mathbf{V}$，$\mathbf{Z} \subseteq \mathbf{V} \setminus \{X, Y\}$ とする．また，$A \in \mathbf{V} \setminus (\{X, Y\} \cup \mathbf{Z})$ とする．このとき，推移性は以下の性質を示す．

$$X \perp Y \mid \mathbf{Z} \Rightarrow X \perp A \mid \mathbf{Z}\ or\ A \perp Y \mid \mathbf{Z}. \qquad (5)$$

ここで，$X \perp Y \mid \mathbf{Z}$ は X と Y が $\mathbf{Z}$ を所与として条件付き独立であることを表す．式 (5) より，ある二変数の条件付き独立性（左辺）からその各変数と他変数との条件付き独立性（右辺）の少なくとも一つを保証できる．推移性は既にマルコフネットワーク学習に利用されている [26] が，前述のようにベイジアンネットワークの条件付き独立性では成り立たない．

Pearl は推移性を緩和した以下の弱推移性がベイジアンネットワークの条件付き独立性で成り立つことを示した [18]．

$$X \perp Y \mid \mathbf{Z}\ and\ X \perp Y \mid \mathbf{Z} \cup \{A\}$$
$$\Rightarrow X \perp A \mid \mathbf{Z}\ or\ A \perp Y \mid \mathbf{Z}. \qquad (6)$$

式 (6) より，弱推移性は条件部の条件付き独立性から右辺の条件付き独立性の少なくとも一つを保証でき，既にベイジアンネットワーク学習に利用されている [19]．しかし，前述のようにこの手法は CI テストの精度を向上させるが，条件部の真偽判定で従来の制約ベースアプローチでは行わない $\mathbf{Z} \cup \{A\}$ を所与とした CI テストを行うために CI テスト数が増加する問題がある．

そこで，本論文では，弱推移性における変数 A の候補空間を条件付けることにより，条件部の $\mathbf{Z}\cup\{A\}$ を所与とした条件付き独立性を検出せずとも新たな条件付き独立性の検出を保証できる以下の定理を導く．

［定理 4.1］ $G=(\mathbf{V},\mathbf{E})$ を DAG とし，X，$Y\in\mathbf{V}$ で，Y は X の非子孫とする．このとき，$A\in\mathbf{V}\setminus(\{X,Y\}\cup\mathbf{Pa}(X,G)\cup\mathbf{W})$ とすると，以下が成り立つ．

$$X\perp Y\mid\mathbf{Pa}(X,G)$$
$$\Rightarrow X\perp A\mid\mathbf{Pa}(X,G)\ \mathit{or}\ A\perp Y\mid\mathbf{Pa}(X,G). \tag{7}$$

ここで，$\mathbf{W}$ は X の子孫であり，X と Y が合流結合するノードとその子孫からなるノード集合を表し，$\mathbf{Pa}(X,G)$ は X の G における親ノード集合を表す．定理 4.1 の証明は付録で示す．定理 4.1 より，マルコフネットワークの推移性と同様に，ベイジアンネットワークのある二変数の条件付き独立性（左辺）からその各変数と他変数との条件付き独立性（右辺）の少なくとも一つを保証できる．すなわち，定理 4.1 はベイジアンネットワークにおける条件付き独立性の推移性を示し，従来の制約ベースアプローチより CI テスト数を増加せずに新たな条件付き独立性の候補変数を同定できる．例えば図 1 において，二ノード X，Y はノード集合 $\{Z\}$ を所与として条件付き独立だが，$A\in\mathbf{W}$ より A は新たな条件付き独立性の候補変数でないと判定でき，CI テスト数は増加しない．

制約ベースアプローチは学習の早期に条件付き独立性を検出してエッジを削除するほど少ない CI テスト数で学習できる．定理 4.1 の推移性はある条件付き独立性から少なくとも一つの条件付き独立性を保証できるため，これを用いれば少なくとも一つの条件付き独立が保証される二組のエッジの CI テストを優先して実施できる．これにより，より学習の早期に条件付き独立性を検出してエッジを削除でき，CI テスト数を削減できる．次節では，推移性を用いたエッジ削除法を提案し，制約ベースアプローチに組み込む．

4.2 提案アルゴリズム

定理 4.1 の推移性を制約ベースアプローチに利用するためには，推移性を有するノード A の探索空間を同定しなければならない．今，$\mathbf{V}$ を n 個のノード集合とし，X，$Y\in\mathbf{V}$，$\mathbf{Pa}(X,G)$ を X のグラフ G における親ノード集合とする．また，$\mathbf{W}$ を X の子孫で，X と Y が合流結合するノードとその子孫からなるノード集合とする．このとき，A の探索空間は $\mathbf{V}\setminus(\{X,Y\}\cup\mathbf{Pa}(X,G)\cup\mathbf{W})$ であるため，まずノード集合 $\mathbf{W}$ の要素を列挙しなければならない．そのためには事前にエッジを方向付ける必要がある．それゆえ，無向グラフ上で CI テストを行う PC アルゴリズム [11] や MMHC アルゴリズム [12] は $\mathbf{W}$ の要素を列挙できない．一方，RAI アルゴリズム [13] は学習途中でエッジを方向付けるため，ノード集合 $\mathbf{W}$ の要素を列挙できる．そこで，本論文は Bayes factor を用いた RAI アルゴリズム [14]～[16] のエッジ削除に推移性を利用する．

前述のように $\mathbf{W}$ の要素を列挙するためにはノード X の子孫で，二ノード X，Y が合流結合するノードを同定する必要がある．この合流結合を厳密に同定することは X，Y 間で X の子孫をもつ全ての道を探索しなければならず，計算量が大きい．しかし，推移性を有するノード A の探索空間は式 (7) を用いて更に縮約できる．今，X と Y の条件付き独立性を検出してエッジを削除したとき，X と A か A と Y（若しくは両方）のエッジが既に削除されていたとする．このとき，X と A か A と Y（若しくは両方）の条件付き独立性は既に検出されているため，式 (7) は既に成り立っており，推移性を利用する必要がない．新たに X か Y（若しくは両方）と条件付き独立な A を検出するためには，A の探索空間は X と Y の共通隣接ノード集合 $\mathbf{Adj}(X,G)\cap\mathbf{Adj}(Y,G)$ に含まれなければならない．このうち，X と Y の共通子ノード集合 $\mathbf{Ch}(X,G)\cap\mathbf{Ch}(Y,G)$ はノード集合 $\mathbf{W}$ に含まれるために探索空間から除外する．また，定理 4.1 より，$\mathbf{Pa}(X,G)$ も除外する．したがって，$\mathbf{Adj}(X,G)\cap\mathbf{Adj}(Y,G)\setminus(\mathbf{Ch}(X,G)\cap\mathbf{Ch}(Y,G)\cup\mathbf{Pa}(X,G))$ が A の探索空間となる．この探索空間は X，Y の共通隣接ノードのみを探索すれば同定できるため，ノード数に対して線形時間で計算できる．

定理 4.1 の推移性を利用した CI テストとエッジ削除法を Algorithm2 に示す．関数 TRANSITIVE_CUT は学習途中のグラフ G_s と G，$X\perp Y\mid\mathbf{Z}$ となる二ノード X，Y とノード集合 $\mathbf{Z}$，データ $\mathbf{D}$ を入力とし，推移性に基づくエッジの削除を行ったあとにグラフを出力する．具体的には，$\mathbf{Adj}(X,G)\cap\mathbf{Adj}(Y,G)\setminus(\mathbf{Ch}(X,G)\cap\mathbf{Ch}(Y,G)\cup\mathbf{Z})$ を前述の探索空間とし，この探索空間に属する任意のノードを A とする．二つのノード対 X と A，A と Y のそれぞれに対して $\mathbf{Z}$ を所与とした Bayes factor を用いる CI テストを行

い，独立と判定されたノード対の間のエッジをグラフから削除する．以上を探索空間に属する全てのノードに対して繰り返す．

推移性を用いたエッジ削除法を RAI アルゴリズムに利用するためには関数 TRANSITIVE_CUT を関数 RAI におけるエッジの削除後（Algorithm1 の 12 行目と 21 行目の直後）に呼び出せば良い．ここで，CI テストの次数が 0 のとき，関数 RAI は，推移性の利用の有無にかかわらず，全てのノード対に対して一回ずつの CI テストを行わなければならない．そのため，提案手法は関数 TRANSITIVE_CUT を 0 次の CI テストでは呼び出さず，1 次以降のときにのみ呼び出す．

提案手法は推移性によって少なくとも一つの条件付き独立性が保証される二組のエッジの CI テストを優先的に実施できるため，学習の早期にエッジを削除でき，CI テスト数を削減できる．制約ベースアプローチの時間計算量は CI テスト数に依存するため，提案手法は計算時間も削減できる．また，データ数が十分に大きくないとき，信頼性の低い CI テストを削減するため，従来の制約ベースアプローチ以上の学習精度を保証する．更に，提案手法では従来の制約ベースアプローチが実現してこなかった大規模構造学習の実現を期待できる．

4.3 漸近一致性

本節では提案手法が以下の漸近一致性を有することを示す．

［定理 4.2］ 真のベイジアンネットワーク構造が仮定 3.1 を満たし，データ数 $N \to \infty$ となるとき，提案手法は真のベイジアンネットワーク構造を推定する．

［証明］ 定理 3.1 より，Algorithm1 は仮定 3.1 の下で $N \to \infty$ のとき真のベイジアンネットワーク構造を推定する．これは CI テストの実施順序にかかわらず成り立つ．提案手法は Algorithm1 における CI テストの実施順序を変更しただけであり，漸近一致性をもち，定理 4.2 が成り立つ． □

Algorithm 2 Edge cutting with transitivity

```
function TRANSITIVE_CUT(G_S, G, X, Y, Z, D)
G_S = (V_S, E_S): 自律的部分構造
G = (V, E): 全体グラフ
X, Y, Z: X ⊥ Y | Z となる二ノード X, Y とノード集合 Z
    A ← Adj(X,G)∩Adj(Y,G)\(Ch(X,G)∩Ch(Y,G)∪Z)
    for A ∈ A do
        if log BF(X, A | Z) < 0 then
            if X ∈ V_S かつ A ∈ V_S then
                E_S ← E_S \ {E_XA}, E ← E \ {E_XA}
                                        ▷ E_XA: XA 間のエッジ
            else
                E ← E \ {E_XA}
            end if
        end if
        if log BF(A, Y | Z) < 0 then
            if A ∈ V_S かつ Y ∈ V_S then
                E_S ← E_S \ {E_AY}, E ← E \ {E_AY}
            else
                E ← E \ {E_AY}
            end if
        end if
    end for
    return (G_S, G)
end function
```

5. 評価実験

本章では提案手法の有効性を示すために数種類の条件で実験を行う．具体的には，PC アルゴリズム [11]，MMHC アルゴリズム [12]，RAI アルゴリズム [13]，提案手法に対して BDeu に基づく Bayes factor（ESS = 1.0 [27]～[29]）を適用し，様々な規模のベンチマークネットワークとランダム生成した大規模ネットワークの構造学習において学習速度と精度を比較する．PC アルゴリズムは Bayes Net Toolbox for Matlab（注1）の実装を，RAI アルゴリズムは Lerner が公開している実装（注2）を使用し，MMHC アルゴリズムと提案手法は独自に実装した．また，Bayes factor を用いた CI テストも独自に実装し，各アルゴリズムに組み込んだ．各手法の計算環境を表 1 に示す．

5.1 ベンチマークネットワークを用いた評価

本節では，ベイジアンネットワークのリポジトリ *bnlearn* [25] に登録されている 7 種類のベンチマークネットワークを用いて実験を行う．これは，様々な規模のネットワークに対して，提案手法が従来手法の精度を保ちつつ，より高速に学習できることを示すためである．ベンチマークネットワークの情報と実験を行ったデータ数について表 2 に示す．表中のノード数，エッジ数，最大親ノード数，パラメータ数はネットワークの規模を表す．

本実験では，各ネットワークについて表 2 のように

表 1 計 算 環 境
Table 1 Computational environment.

PC アルゴリズム，RAI アルゴリズム，提案手法	
CPU	3.5 GHz 6-Core Intel Xeon E5 Mac Pro
System Memory	64GB
OS	Mac OSX 10.11.6
ソフトウェア	MATLAB
MMHC アルゴリズム	
CPU	3.5 GHz 6-Core Intel Xeon E5 Mac Pro
System Memory	64GB
OS	Mac OSX 10.11.6
ソフトウェア	Java

（注1）：https://github.com/bayesnet/bnt

（注2）：http://www.ee.bgu.ac.il/∼boaz/software.html

表 2　ベンチマークネットワーク
Table 2　Benchmark networks.

network	ノード数	エッジ数	最大親ノード数	パラメータ数	データ数
cancer	5	4	2	10	10,000 ～ 2,000,000
earthquake	5	4	2	10	10,000 ～ 200,000
sachs	11	17	3	178	10,000 ～ 200,000
alarm	37	46	4	509	10,000 ～ 2,000,000
win95pts	76	112	7	574	10,000 ～ 2,000,000
andes	223	338	6	1157	10,000 ～ 2,000,000
munin	1041	1397	3	6314	10,000 ～ 200,000

データ数を増加させるときの学習速度と精度を比較した．実験手順は以下のとおりである．

（1）　各パターンの真のネットワーク構造からデータセットを表 2 のデータ数だけランダムに発生する．

（2）　手順（1）で発生させたデータを用いて，各手法により構造学習する．

（3）　手順（2）を 10 回繰り返す．

ただし，6 時間の制限時間を設け，超過する場合は学習を打ち切った．

本実験の CI テスト数と計算時間，Structural Hamming Distance（SHD）[12] の結果を表 3 から表 9 に示す．CI テスト数は各学習において行われた条件付き独立性検定（Conditional Independence test: CI テスト）の回数を表す．また，SHD は，真には存在するが学習において削除されたエッジ数と真には存在しないが学習において残されたエッジ数，エッジの方向付けの誤り数の和によって真の構造と推定された構造の距離を表す．SHD が 0.0 に収束することで真の構造と推定された構造が一致したことを表す．表中の “-” は制限時間内に学習できなかったことを表す．また，表中の PC，MMHC，名取ら，提案手法はそれぞれ，PC アルゴリズム，MMHC アルゴリズム，Bayes factor を用いた RAI アルゴリズム [14]～[16]，提案手法の学習結果を表す．

提案手法と PC アルゴリズムを比べると，提案手法は全てのネットワーク学習において CI テスト数を削減し，cancer を 10,000 個のデータで学習したときを除いて計算時間も削減した．これより，提案手法は PC アルゴリズムより CI テスト数と計算時間を削減して学習できることが確認された．提案手法は大規模ネットワーク学習において PC アルゴリズムより SHD を減少したが，小規模から中規模のネットワーク学習において SHD をわずかに増加した．これと同様の傾向が名取らの手法における SHD でも確認できる．大規模のネットワークではパラメータ数が増大し，データがスパースになるため，CI テストの信頼性が低下す

表 3　cancer の実験結果
Table 3　The experiment results for cancer.

CI テスト数				
データ数	PC	MMHC	名取ら	提案手法
10,000	36.6	17.1	32.1	32.1
20,000	39.2	17.1	34.0	34.0
50,000	48.6	17.7	39.9	40.9
100,000	49.5	17.8	40.1	40.3
200,000	52.5	18.0	41.4	41.8
500,000	55.2	18.0	42.9	43.4
1,000,000	56.1	18.0	43.8	43.7
2,000,000	57.0	18.0	44.6	44.0
計算時間（s）				
10,000	0.08	0.40	0.06	0.16
20,000	0.13	0.92	0.10	0.07
50,000	0.27	1.06	0.16	0.17
100,000	0.53	1.99	0.23	0.25
200,000	1.01	3.98	0.50	0.42
500,000	2.66	10.28	1.14	1.32
1,000,000	7.02	19.63	2.69	3.09
2,000,000	15.36	37.36	9.19	6.59
SHD				
10,000	2.5	1.3	2.7	2.7
20,000	2.4	1.5	2.8	2.8
50,000	1.9	0.3	3.3	3.3
100,000	1.6	0.3	2.9	2.7
200,000	1.3	0.3	2.8	2.5
500,000	0.6	0.2	1.2	0.6
1,000,000	0.3	0.2	0.6	0.3
2,000,000	0.0	0.1	0.0	0.0

表 4　earthquake の実験結果
Table 4　The experiment results for earthquake.

CI テスト数				
データ数	PC	MMHC	名取ら	提案手法
10,000	60.9	57.0	46.5	45.2
20,000	57.0	57.7	45.0	44.0
50,000	57.0	57.0	45.0	44.0
100,000	57.0	57.0	45.0	44.0
200,000	57.0	57.0	45.0	44.0
計算時間（s）				
10,000	0.11	0.23	0.11	0.10
20,000	0.15	0.61	0.14	0.10
50,000	0.29	0.65	0.24	0.16
100,000	0.55	1.27	0.35	0.24
200,000	0.97	2.50	0.64	0.47
SHD				
10,000	1.5	1.3	1.5	1.5
20,000	0.0	1.4	0.0	0.0
50,000	0.0	0.1	0.0	0.0
100,000	0.0	1.3	0.0	0.0
200,000	0.0	1.0	0.0	0.0

表 5　sachs の実験結果
Table 5　The experiment results for sachs.

CI テスト数				
データ数	PC	MMHC	名取ら	提案手法
10,000	604.1	144.9	427.0	426.7
20,000	731.5	166.9	529.3	522.4
50,000	859.5	193.0	624.8	614.8
100,000	932.6	194.2	695.2	687.0
200,000	988.0	197.0	733.0	731.0
計算時間（s）				
10,000	3.02	1.37	1.04	0.99
20,000	5.57	2.80	2.02	1.65
50,000	10.75	7.16	4.00	3.08
100,000	20.90	25.37	7.36	6.13
200,000	28.24	63.17	14.34	13.11
SHD				
10,000	14.1	17.0	16.2	14.3
20,000	12.9	17.0	17.3	16.1
50,000	10.2	17.0	14.0	14.0
100,000	10.0	17.0	12.6	12.6
200,000	0.0	17.0	0.0	0.0

表 6　alarm の実験結果
Table 6　The experiment results for alarm.

CI テスト数				
データ数	PC	MMHC	名取ら	提案手法
10,000	2352.3	3060.1	1625.7	1314.8
20,000	2537.8	3331.0	1797.5	1404.6
50,000	2904.9	3751.6	2188.4	1498.0
100,000	3153.1	4029.0	2429.7	1608.3
200,000	3372.2	4322.7	2522.3	1648.6
500,000	3673.5	4449.2	2746.7	1735.2
1,000,000	3871.8	4512.0	3041.1	1892.5
2,000,000	4024.8	4393.3	3320.0	1917.4
計算時間（s）				
10,000	6.31	5.31	2.58	1.98
20,000	11.29	11.54	4.74	2.83
50,000	21.37	30.81	9.86	4.89
100,000	43.08	68.46	21.49	8.45
200,000	90.75	150.28	44.63	15.45
500,000	241.49	384.17	135.91	44.95
1,000,000	463.17	751.52	330.33	103.05
2,000,000	1127.21	1464.26	701.68	273.58
SHD				
10,000	18.7	24.1	24.3	17.8
20,000	18.9	24.6	26.1	18.2
50,000	15.2	25.6	30.2	20.1
100,000	12.2	24.3	30.2	18.8
200,000	8.8	23.3	25.6	15.2
500,000	3.0	22.9	17.0	10.2
1,000,000	3.0	23.5	16.2	10.5
2,000,000	2.0	21.7	15.9	8.8

表 7　win95pts の実験結果
Table 7　The experiment results for win95pts.

CI テスト数				
データ数	PC	MMHC	名取ら	提案手法
10,000	8204.1	-	6154.7	4650.7
20,000	9140.4	-	6496.2	4911.4
50,000	10237.1	-	6955.2	5186.9
100,000	11431.2	-	7269.6	5419.8
200,000	12590.4	-	7524.3	5673.3
500,000	13836.2	-	7987.6	6012.7
1,000,000	14742.5	-	8333.5	6301.3
2,000,000	15605.9	-	8503.8	6415.3
計算時間（s）				
10,000	13.25	-	7.96	4.22
20,000	24.11	-	15.16	6.01
50,000	56.27	-	29.90	13.25
100,000	138.42	-	55.04	25.40
200,000	326.64	-	114.26	51.96
500,000	1051.26	-	251.47	167.71
1,000,000	2201.24	-	724.37	440.86
2,000,000	5429.31	-	2251.42	896.56
SHD				
10,000	52.8	-	59.6	57.0
20,000	46.6	-	53.5	48.8
50,000	41.2	-	44.2	41.7
100,000	37.2	-	36.5	35.5
200,000	35.7	-	36.8	35.5
500,000	31.3	-	35.5	33.5
1,000,000	30.2	-	33.3	32.3
2,000,000	28.6	-	33.3	32.2

る傾向がある．提案手法と名取らの手法が大規模ネットワークで PC アルゴリズムより高精度であった理由は CI テストを PC アルゴリズムより大幅に削減したことで信頼性の低い CI テスト数を減少できたためと考えられる．一方，提案手法と名取らの手法は CI テストの次数ごとにエッジの削除と方向付けを繰り返すために方向付けの誤りがエッジの削除に影響するが，PC アルゴリズムはエッジの削除を完了した後にエッジを方向付けるため，エッジの方向付けによる誤りがエッジの削除に影響しない．それゆえ，PC アルゴリズムは CI テストの信頼性が高い小規模から中規模のネットワークを高精度に学習できると考えられる．一方，PC アルゴリズムは 2,000,000 個のデータによる andes の学習と munin の学習を制限時間内に実行できなかったが，提案手法は全てのパターンで制限時間内に学習できた．これより，提案手法は PC アルゴリズムが学習できない大規模のネットワーク学習を実現できることが示された．

表 8　andes の実験結果
Table 8　The experiment results for andes.

データ数	PC	MMHC	名取ら	提案手法
CI テスト数				
10,000	45750.5	93179.8	29097.9	28030.0
20,000	55772.4	101756.1	30630.1	29022.9
50,000	70752.2	123526.8	33282.9	30690.8
100,000	85640.0	149086.2	35666.7	32124.8
200,000	94747.0	185620.9	38807.5	33822.7
500,000	99722.7	-	44130.3	36405.1
1,000,000	112619.0	-	49750.0	38920.5
2,000,000	-	-	57408.4	41915.1
計算時間（s）				
10,000	65.86	238.66	35.66	32.34
20,000	120.83	523.37	83.90	44.90
50,000	316.94	1769.46	132.53	80.86
100,000	745.34	4497.28	241.17	141.37
200,000	1911.08	12103.61	479.40	279.65
500,000	9560.77	-	1992.89	754.87
1,000,000	19996.00	-	4257.23	1831.74
2,000,000	-	-	9706.24	4942.97
SHD				
10,000	125.7	223.9	70.1	72.2
20,000	102.3	216.3	51.8	55.8
50,000	83.0	211.9	29.8	39.0
100,000	67.8	205.6	23.3	26.5
200,000	44.9	199.8	18.9	19.2
500,000	22.4	-	15.7	13.8
1,000,000	15.8	-	14.7	11.2
2,000,000	-	-	12.4	9.5

表 9　munin の実験結果
Table 9　The experiment results for munin.

データ数	PC	MMHC	名取ら	提案手法
CI テスト数				
10,000	-	-	560361.8	553392.1
20,000	-	-	588880.7	562427.8
50,000	-	-	653391.1	575784.0
100,000	-	-	705078.8	582138.4
200,000	-	-	665294.7	588161.2
計算時間（s）				
10,000	-	-	772.71	620.87
20,000	-	-	1592.53	870.84
50,000	-	-	2587.48	1456.83
100,000	-	-	4783.72	2195.40
200,000	-	-	7255.80	3907.85
SHD				
10,000	-	-	504.4	461.8
20,000	-	-	504.1	430.4
50,000	-	-	525.9	404.0
100,000	-	-	552.5	405.6
200,000	-	-	540.6	404.1

次に，提案手法と MMHC アルゴリズムを比較する．提案手法は cancer と sachs を除いたネットワーク学習において MMHC アルゴリズムよりも CI テスト数を削減でき，全てのネットワーク学習において計算時間が短い．したがって，提案手法は 30 ノードを超える中規模から大規模のネットワークにおいて MMHC アルゴリズムより CI テスト数と計算時間を削減できることが分かる．更に，提案手法は cancer を除いたネットワーク学習において MMHC アルゴリズムより SHD を減少している．特に大規模ネットワークの andes では，提案手法の SHD が MMHC アルゴリズムの SHD より大きく下回っている．この結果より，提案手法は短い計算時間で MMHC アルゴリズムと同等以上の精度で学習でき，特に大規模構造学習で MMHC アルゴリズムより学習精度を向上することが確認できた．MMHC アルゴリズムは 500,000 個以上のデータによる andes の学習と win95pts，munin の学習を制限時間内に実行できなかったが，提案手法は制限時間内にこれらのネットワークを学習できた．それゆえ，提案手法が MMHC アルゴリズムでは学習できない規模のネットワーク構造を学習できることが示された．

最後に提案手法と名取らの手法を比較する．提案手法は 30 ノード以下の小規模ネットワークを名取らの手法と同程度の CI テスト数と計算時間で学習し，中規模から大規模ネットワークを名取らの手法より CI テスト数と計算時間を削減して学習できた．また，小規模ネットワークにおいて，提案手法は名取らの手法と同様に，データ数が十分大きくなるときに真の構造を推定できた．これより，提案手法は名取らの手法と同様に漸近一致性をもつことが示された．30 ノード以上のネットワークでは提案手法は名取らの手法と同等かそれ以下の SHD で学習できた．特に最大規模のベンチマークネットワークである munin において，提案手法は名取らの手法より SHD を大幅に減少した．以上より，提案手法は短い計算時間で名取らの手法と同等以上の学習精度を保ち，特に大規模のネットワークにおいて学習精度を向上することが示された．

以上より，以下を確認できた．

（1） 30 ノードを超える中規模から 1000 ノード程度の大規模ネットワークにおいて，提案手法は従来の制約ベースアプローチより学習に要する CI テスト数と計算時間を削減できる．

（2） 提案手法は小規模から大規模のネットワークで MMHC アルゴリズムや名取らの手法と同等以上の

表 10　ランダムネットワーク
Table 10　Random networks.

network	ノード数	エッジ数	最大親ノード数	パラメータ数
random2000	2000	4959	5	33528
random2500	2500	6250	5	42382
random3000	3000	7454	5	50710
random3500	3500	8694	5	59436

表 11　ランダムネットワークの実験結果
Table 11　The experiment results for random networks.

CI テスト数		
ネットワーク	名取ら	提案手法
random2000	2071906.2	2062591.0
random2500	3227667.6	3214255.7
random3000	4609199.8	4597386.1
random3500	-	6242323.0
計算時間（s）		
random2000	5346.09	3351.71
random2500	9742.30	3748.95
random3000	18141.00	8970.68
random3500	-	12714.20
SHD		
random2000	4037.6	3841.8
random2500	5171.7	4900.0
random3000	6272.3	5962.8
random3500	-	6744.4

学習精度で学習できる．

（3） 提案手法は PC アルゴリズムや MMHC アルゴリズムで学習できないほど大規模のネットワーク学習を実現できる．

5.2 大規模ランダムネットワークを用いた評価

前節では，PC アルゴリズムや MMHC アルゴリズムでは学習できない規模のネットワークを提案手法が学習できることを示した．本節では，ランダムに生成したネットワーク（ランダムネットワーク）を用いて実験を行い，名取らの手法では学習できない規模のネットワークを提案手法が学習できることを示す．

ランダムネットワークの生成には BNGenerator(注3) [30], [31] を使用する．BNGenerator はマルコフ連鎖モンテカルロ法により一様分布からネットワークをランダムに生成する．本論文では，ノード数を 2000, 2500，3000，3500 とし，各ネットワークの最大次数を 5 と指定して BNGenerator を使用することで表 10 に示すネットワークを得た．本節では，生成されたネットワークからデータ数 10,000 のデータセットをランダムに発生させ，前節と同様の実験手順で名取らの手法と提案手法を比較する．

本実験の CI テスト数と計算時間，SHD の結果を表 11 に示す．表 11 より，提案手法は全てのネットワーク学習で名取らの手法より CI テスト数と計算時間を削減し，SHD を減少させた．これより，提案手法は 2000 ノード以上の大規模ネットワークを名取らの手法より高速かつ高精度に学習できることが示された．名取らの手法は random3500 を制限時間内で学習できなかったが，提案手法は全てのネットワークを制限時間内に学習できた．したがって，提案手法は RAI アルゴリズムが学習できなかった 3500 ノードのネットワーク学習を実現できる．

6. む す び

本論文では，ベイジアンネットワークのある二変数の条件付き独立性からその各変数と他変数との条件付き独立性の少なくとも一つを保証できる推移性が成り立つことを示し，それを Bayes factor を用いた RAI アルゴリズムへ利用することで従来の制約ベースアプローチより CI テスト数を大幅に削減する学習アルゴリズムを提案した．シミュレーション実験により，提案手法は中規模から大規模の構造学習において従来手法より CI テスト数と計算時間を削減することを示した．また，提案手法は，特に大規模ネットワークにおいて，信頼性の低い CI テストを削減するために学習精度を向上した．更に，従来手法では学習できないほど大規模のベイジアンネットワーク学習を実現した．

今後の課題として，大規模構造学習の精度向上が挙げられる．大規模構造学習ではデータがスパースになるため，CI テストの信頼性が低下する傾向がある．この問題に対し，磯崎らは Minimum Free Energy（MFE）に基づくパラメータ推定法と CI テストを提案している [32]～[34]．この MFE に基づくパラメータ推定法と CI テストを組み込むことで大規模構造学習の精度向上が見込める．また，実用面の課題として推論精度を最適化する学習 [35] やベイジアンネットワーク分類器の厳密学習 [36] への拡張が挙げられる．

謝辞　本論文の一部は本田 [37] によって発表されている．

文　　献

[1] 植野真臣，ベイジアンネットワーク，コロナ社，2013.
[2] D.M. Chickering, “Learning Bayesian networks is

（注3）：http://sites.poli.usp.br/pmr/ltd/software/bngenerator/

NP-Complete," in Learning from Data: Artificial Intelligence and Statistics, vol.V, pp.121–130, Springer, 1996.

[3] R.G. Cowell, "Efficient maximum likelihood pedigree reconstruction," Theoretical Population Biology, vol.76, no.4, pp.285–291, Dec. 2009.

[4] M. Koivisto and K. Sood, "Exact Bayesian structure discovery in Bayesian networks," J. Machine Learning Research, vol.5, pp.549–573, Dec. 2004.

[5] A. Singh and A. Moore, "Finding optimal Bayesian networks by dynamic programming," Technical report, Carnegie Mellon University, pp.1–16, June 2005.

[6] T. Silander and P. Myllymaki, "A simple approach for finding the globally optimal Bayesian network structure," Proc. Uncertainty in Artificial Intelligence (UAI), pp.445–452, 2006.

[7] B. Malone, C. Yuan, and E.A. Hansen, "Memory-efficient dynamic programming for learning optimal Bayesian networks," Proc. 25th AAAI Conference, pp.1057–1062, 2011.

[8] C. Yuan, B. Malone, and W. Xiaojian, "Learning optimal Bayesian networks using A* search," Int. Joint Conference on Artificial Intelligence (IJCAI), pp.2186–2191, 2011.

[9] J. Cussens, "Bayesian network learning with cutting planes," Proc. Uncertainty in Artificial Intelligence (UAI), pp.153–160, 2011.

[10] J. Pearl, Causality: Models, Reasoning, and Inference, Cambridge University Press, 2000.

[11] P. Spirtes, C. Glymour, and R. Scheines, Causation, Prediction, and Search, MIT Press, 2000.

[12] I. Tsamardinos, L.E. Brown, and C.F. Aliferis, "The max-min hill-climbing Bayesian network structure learning algorithm," Machine Learning, vol.65, no.1, pp.31–78, 2006.

[13] R. Yehezkel and B. Lerner, "Bayesian network structure learning by recursive autonomy identification," J. Machine Learning Research, vol.10, pp.1527–1570, 2009.

[14] K. Natori, M. Uto, Y. Nishiyama, S. Kawano, and M. Ueno, "Constraint-based learning Bayesian networks using Bayes factor," Advanced Methodologies for Bayesian Networks (AMBN) 2015, vol.LNAI 9505, pp.15–31, Springer, 2015.

[15] K. Natori, M. Uto, and M. Ueno, "Consistent learning Bayesian networks with thousands of variables," Advanced Methodologies for Bayesian Networks (Proc. Machine Learning Research), vol.73, pp.57–68, 2017.

[16] 名取和樹，宇都雅輝，植野真臣，"Bayes factor を用いた RAI アルゴリズムによる大規模ベイジアンネットワーク学習，" 信学論（D），vol.J101-D, no.5, pp.754–768, May 2018.

[17] J. Pearl and A. Paz, Graphoids, A graph-based logic for reasoning about relevance relations, University of California (Los Angeles). Computer Science Department, 1985.

[18] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan-Kaufmann, 1988.

[19] F. Bromberg and D. Margaritis, "Improving the reliability of causal discovery from small data sets using argumentation," J. Machine Learning Research, vol.10, pp.301–340, Feb. 2009.

[20] W. Buntine, "Theory refinement on Bayesian networks," Proc. Uncertainty in Artificial Intelligence (UAI), pp.52–60, 1991.

[21] D. Heckerman, D. Geiger, and D.M. Chickering, "Learning Bayesian networks: The combination of knowledge and statistical data," Mach. Learn., vol.20, pp.197–243, 1995.

[22] H. Steck and T.S. Jaakkola, "On the dirichlet prior and Bayesian regularization," Neural Information Processing Systems (NIPS 2002), pp.697–704, 2002.

[23] J. Abellán, M. Gómez-Olmedo, S. Moral, et al., "Some variations on the PC algorithm," International Conference on Probabilistic Graphical Models, pp.1–8, 2006.

[24] R. Sedgewick and K. Wayne, Algorithms, 4th ed., Pearson, 2011.

[25] M. Scutari, "Learning Bayesian networks with the bnlearn R package," J. Statistical Software, vol.35, no.3, pp.1–22, 2011.

[26] F. Bromberg, D. Margaritis, and V. Honavar, "Efficient Markov network structure discovery using independence tests," J. Artificial Intelligence Research, vol.35, pp.449–484, 2009.

[27] M. Ueno, "Learning likelihood-equivalence Bayesian networks using an empirical Bayesian approach," Behaviormetrika, vol.35, no.2, pp.115–135, 2007.

[28] M. Ueno, "Learning networks determined by the ratio of prior and data," Proc. Uncertainty in Artificial Intelligence (UAI), pp.598–605, 2010.

[29] M. Ueno, "Robust learning Bayesian networks for prior belief," Proc. Uncertainty in Artificial Intelligence (UAI), pp.689–707, 2011.

[30] J.S. Ide and F.G. Cozman, "Random generation of Bayesian networks," in Brazilian symposium on artificial intelligence, pp.366–376, Springer, 2002.

[31] J.S. Ide, F.G. Cozman, and F.T. Ramos, "Generating random Bayesian networks with constraints on induced width," Proc. European Conference on Artificial Intelligence (ECAI), vol.16, pp.323–327, 2004.

[32] T. Isozaki, N. Kato, and M. Ueno, "Minimum free energies with "data temperature" for parameter learning of Bayesian networks," 20th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'08), vol.1, pp.371–378, 2008.

[33] T. Isozaki, N. Kato, and M. Ueno, " "Data

temperature" in minimum free energies for parameter learning of bayesian networks," Int. J. Artificial Intelligence Tools, vol.18, no.05, pp.653–671, 2009.

[34] T. Isozaki and M. Ueno, "Minimum free energy principle for constraint-based learning Bayesian networks," Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML), pp.612–627, 2009.

[35] C. Li and M. Ueno, "An extended depth-first search algorithm for optimal triangulation of Bayesian networks," International Journal of Approximate Reasoning, vol.80, pp.294–312, 2017.

[36] S. Sugahara, M. Uto, and M. Ueno, "Exact learning augmented naive Bayes classifier," International Conference on Probabilistic Graphical Models, pp.439–450, 2018.

[37] 本田和雅，推移性を利用した大規模ベイジアンネットワーク構造学習，電気通信大学大学院大学院情報理工学研究科情報・ネットワーク工学専攻修士論文（未公刊），2019.

付　　録

定理 4.1 の証明

以下の補題 1 を証明した後に定理 4.1 を証明する．

［補題 1］ $G = (\mathbf{V}, \mathbf{E})$ を非循環有向グラフ（Directed Acyclic Graph: DAG）とし，$X, Y \in \mathbf{V}$ で，Y は X の非子孫とする．このとき，$\forall A \in \mathbf{V} \setminus (\{X, Y\} \cup \mathbf{Pa}(X, G) \cup \mathbf{W})$ とすると，以下が成り立つ．

$$X \perp Y \mid \mathbf{Pa}(X, G) \Rightarrow X \perp Y \mid \mathbf{Pa}(X, G) \cup \{A\}. \tag{A·1}$$

ここで，$\mathbf{W}$ は X の子孫であり，X と Y が合流結合するノードとその子孫からなるノード集合を表し，$\mathbf{Pa}(X, G)$ は X の G における親ノード集合を表す．

［証明］（補題 1） ノード A が X の非子孫である場合と子孫である場合に分けられる．

（1） ノード A が X の非子孫である場合

X の非子孫は $\mathbf{Pa}(X, G)$ を所与として条件付き独立である [18]．すなわち，$X \perp Y \mid \mathbf{Pa}(X, G)$ かつ $X \perp A \mid \mathbf{Pa}(X, G)$．条件付き独立性の合成性と弱結合性より，

$$\begin{aligned}
& X \perp Y \mid \mathbf{Pa}(X, G) \ \mathit{and}\ X \perp A \mid \mathbf{Pa}(X, G) \\
& \Rightarrow X \perp Y \cup A \mid \mathbf{Pa}(X, G) \qquad (\because \text{合成性}) \\
& \Rightarrow X \perp A \mid \mathbf{Pa}(X, G) \cup \{Y\} \qquad (\because \text{弱結合性}) \\
& \quad \mathit{and}\ X \perp Y \mid \mathbf{Pa}(X, G) \cup \{A\} \\
& \Rightarrow X \perp Y \mid \mathbf{Pa}(X, G) \cup \{A\}.
\end{aligned}$$

したがって，式 (A·1) が成り立つ．

（2） ノード A が X の子孫である場合

X と Y を結ぶ全ての道は，内点に A が存在する道 p と内点に A が存在しない道 q に分けられる．それぞれの道が $\forall A \in \mathbf{Des}(X, G) \setminus \mathbf{W}$ を加えたノード集合 $\mathbf{Pa}(X, G) \cup \{A\}$ でブロックされるならば，X と Y を結ぶ全ての道が $\mathbf{Pa}(X, G) \cup \{A\}$ でブロックされる．ここで，$\mathbf{Des}(X, G)$ は X の G における子孫集合を表す．

（a） 道 p が存在する場合

A が X の子孫であるから，道 p の内点に X の子孫が存在する．G が DAG で Y が X の非子孫であるため，X の子孫で合流結合するノード $\exists W \in \mathbf{W}$ が道 p の内点となることに注意する．$A \notin \mathbf{W}$ であるため，道 p はノード集合 $\mathbf{Pa}(X, G) \cup \{A\}$ でもブロックされる．

（b） 道 q が存在する場合

道 q の内点に X の子孫が存在する場合と存在しない場合に分けられる．

i. 道 q の内点に X の子孫が存在しない場合

ノード Y とノード A を結ぶ道の内点には必ず $\mathbf{Pa}(X, G)$ に属するノードが合流結合せずに存在する．すなわち，$X \perp Y \mid \mathbf{Pa}(X, G)$ かつ $A \perp Y \mid \mathbf{Pa}(X, G)$ である．条件付き独立性の合成性と弱結合性より，

$$\begin{aligned}
& X \perp Y \mid \mathbf{Pa}(X, G) \ \mathit{and}\ A \perp Y \mid \mathbf{Pa}(X, G) \\
& \Rightarrow Y \perp X \cup A \mid \mathbf{Pa}(X, G) \qquad (\because \text{合成性}) \\
& \Rightarrow Y \perp A \mid \mathbf{Pa}(X, G) \cup \{X\} \qquad (\because \text{弱結合性}) \\
& \quad \mathit{and}\ Y \perp X \mid \mathbf{Pa}(X, G) \cup \{A\} \\
& \Rightarrow X \perp Y \mid \mathbf{Pa}(X, G) \cup \{A\}.
\end{aligned}$$

したがって，式 (A·1) が成り立つ．

ii. 道 q の内点に X の子孫が存在する場合

道 q の内点に A が存在しないことから，道 q の内点で X の子孫となるノードは必ず $\mathbf{W}$ に属する．G が DAG で，Y が X の非子孫であることから，道 q は $\exists W \in \mathbf{W}$ で合流結合をする．$A \notin \mathbf{W}$ であるから，道 q はノード集合 $\mathbf{Pa}(X, G) \cup \{A\}$ でもブロックされる．

2(b)i，2(b)ii より，2(b) で式 (A·1) が成り立つ．

（c） 道 p と道 q がどちらも存在しない場合

X と Y を結ぶ道が存在しないため，任意の条件集合を所与として X と Y は条件付き独立である．したがって，X と Y はノード集合 $\mathbf{Pa}(X, G) \cup \{A\}$ を所

与として条件付き独立である．
2(a)，2(b)，2(c) より，(2) で式 (A·1) が成り立つ．
(1)，(2) より，$\forall A \in \mathbf{V} \setminus (\{X, Y\} \cup \mathbf{Pa}(X, G) \cup \mathbf{W})$ で式 (A·1) が成り立つ． □
［証明］（定理 4.1） 補題 1 より，$\forall A \in \mathbf{V} \setminus (\{X, Y\} \cup \mathbf{Pa}(X, G) \cup \mathbf{W})$ で，

$$X \perp Y \mid \mathbf{Pa}(X, G) \Rightarrow X \perp Y \mid \mathbf{Pa}(X, G) \cup \{A\},$$

であるから，

$$X \perp Y \mid \mathbf{Pa}(X, G) \\ \Rightarrow X \perp Y \mid \mathbf{Pa}(X, G) \; and \; X \perp Y \mid \mathbf{Pa}(X, G) \cup \{A\}. \tag{A·2}$$

条件付き独立性の弱推移性より，

$$X \perp Y \mid \mathbf{Pa}(X, G) \; and \; X \perp Y \mid \mathbf{Pa}(X, G) \cup \{A\} \\ \Rightarrow X \perp A \mid \mathbf{Pa}(X, G) \; or \; A \perp Y \mid \mathbf{Pa}(X, G). \tag{A·3}$$

よって，式 (A·2) と式 (A·3) より，$\forall A \in \mathbf{V} \setminus (\{X, Y\} \cup \mathbf{Pa}(X, G) \cup \mathbf{W})$ で式 (7) が成り立つ． □

（2019 年 3 月 27 日受付，5 月 30 日再受付，
8 月 6 日早期公開）

本田　和雅

2017 年山形大学工学部卒，同年電気通信大学大学院情報理工学研究科情報・ネットワーク工学専攻博士前期課程入学，現在に至る．

名取　和樹　（正員）

2014 年電気通信大学情報理工学部卒．2016 年同大学院情報システム学研究科社会知能情報学専攻博士前期課程了．同年，同大学院情報理工学研究科情報·ネットワーク工学専攻博士後期課程入学，現在に至る．

菅原　聖太

2018 年電気通信大学情報理工学部卒．同年，同大学院情報理工学研究科情報・ネットワーク工学専攻博士前期課程入学，現在に至る．

磯崎　隆司

1995 年東京工業大学理学部物理学科卒，1997 年東北大学大学院理学研究科物理学専攻博士前期課程了，2010 年電気通信大学大学院情報システム学研究科社会知能情報学専攻博士後期課程了，博士（工学）．現在（株）ソニーコンピュータサイエンス研究所リサーチャー．2016 年より電気通信大学大学院情報理工学研究科客員准教授を併任．

植野　真臣　（正員）

1992 年神戸大学大学院教育学研究科了，1994 年東京工業大学大学院総合理工学研究科了．博士（工学）．東京工業大学，千葉大学，長岡技術科学大学を経て 2006 年より電気通信大学助教授，2013 年より教授，現在に至る．