

5. グラフィカルモデルと ベイジアンネットワーク

植野真臣
電気通信大学
情報理工学研究科

4月15日 ベイズの定理とは？
4月22日 ベイズはどのようにして世に出たのか？
5月6日【休日出勤】 ベイズはコンピュータの父
5月13日 ベイズの躍進と人工知能の誕生
5月20日 ビリーフとベイズの定理
5月27日 尤度推定と機械学習
6月3日 ベイズ推定と機械学習(1)
6月10日 ベイズ推定と機械学習(2)
6月17日 ベイズ意思決定
7月8日 確率的グラフィカルモデルベイジアンネットワーク
7月22日 ベイジアンネットワークの推論
7月29,30日 ベイジアンネットワークと他の機械学習との関係
8月5日 テスト(※場所: 西5-109)

参考図書鈴木譲 ・ 植野真臣 編著 確率的グラフィカルモデル



本日の目標

1. AIとビッグデータの課題
2. グラフィカルモデル
3. 様々な グラフィカルモデル
4. 最も上位のベイジアンネットワーク
5. ベイジアンネットワークと様々な機械学習との関係
6. ベイジアンネットワークによる推論

0. ビック・データ時代とAI

- 90年代ー00時代 インフラストラクチャー時代
いかにデータを蓄えるか、データを蓄えさせる時代
- 有り余るデータとその有効活用が課題: 大量のデータから高精度に高度な処理ができる手法→次世代の企業コンピーテンシー
- 簡単にまねできない高精度な処理技法が注目される時代に突入
- 総合格闘技→数理統計、アルゴリズム、データベースの統合的技術

古典的人工知能(ルール)

- 古典的AI (アリストテレス)
- 論理推論、 IF then rule
- 人は死ぬ、ソクラテスは人である、ソクラテスは死ぬ

.

古典的AIの問題

- 当たり前のことしか推論できない
- 不確実な現象を推論できない
- 例外が多い
- 学習ができない

確率推論では

- より一般化した表現
- 同時確率分布

例：性別、髪の高さ、背の高さの同時確率分布

データより性別、髪の高さ、背の高さの同時確率分布が以下であることが分かっているとする。

$P(\text{男、髪短い、背高い})=0.2$
 $P(\text{男、髪長い、背高い})=0.125$
 $P(\text{男、髪長い、背低い})=0.05$
 $P(\text{男、髪短い、背低い})=0.125$
 $P(\text{女、髪短い、背高い})=0.05$
 $P(\text{女、髪長い、背高い})=0.125$
 $P(\text{女、髪長い、背低い})=0.2$
 $P(\text{女、髪短い、背低い})=0.125$

$P(\text{男})=0.5, P(\text{女})=0.5$

同時確率分布からの確率推論

- 「その人は髪が短い」ことがわかった

$$P(\text{男、髪短い、背高い})=0.2$$

$$P(\text{男、髪短い、背低い})=0.125$$

$$P(\text{女、髪短い、背高い})=0.05$$

$$P(\text{女、髪短い、背低い})=0.125$$

$$\text{男の確率}=0.325/0.5=0.65$$

同時確率分布からの確率推論

- さらに「その人は背が高い」ことがわかった

$$P(\text{男、髪短い、背高い})=0.2$$

$$P(\text{女、髪短い、背高い})=0.05$$

$$\text{男の確率}=0.2/0.25=0.8$$

確率推論の数学的定式化

データ x_d が得られたときの x_i の確率は

$$p(x_i|x_d) = \sum_{j \neq i} p(x_1, x_2, \dots, x_N|x_d)$$

世界中のすべての変数の同時確率
分布を知ればなんでも推論でき
る！！

問題

- 変数が増えると状態数が指数的に増える
- すべて2値しかとらなくても 10変数で1024パターンの同時確率を推定しないといけない。

- 100変数で

1267650000000000000000000000000000

パターンの同時確率を推定しなければならない。

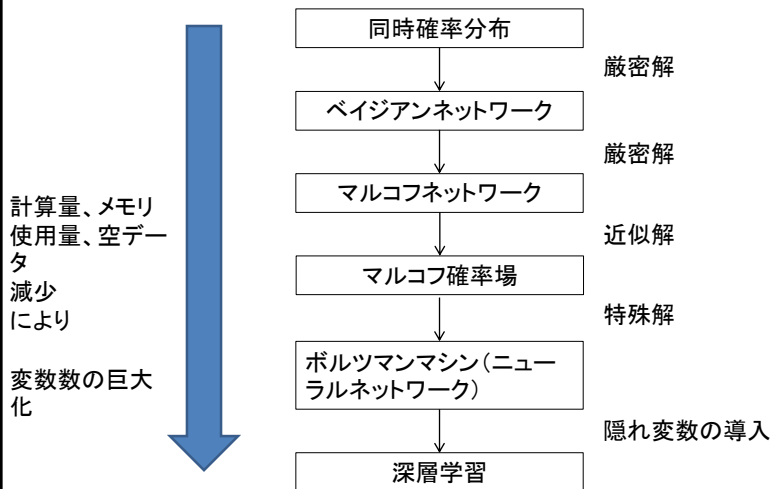
二つの問題

- 計算量が指数的に爆発する。
- データ数よりパターン数のほうが多くなってしまうと、各パターンを推定するためのデータが0になるものが大量発生。

(大量のデータがあっても空データだらけになる)

ビッグデータ問題の課題は、スパースデータ(空データの増加)と計算量(メモリ、計算速度)

解決のための数理モデル



ベイジアンネットワーク

- 確率構造が非循環有向グラフであれば、同時確率分布が条件付確率の積に因数分解できることが数学的に証明できる。
- 確率有向グラフが確率因果構造が対応し、ものごとの因果もわかる！！

現在考えられる最もよい同時確率分布の推定値

⇒ 推論の予測精度が最高のはず！！

ベイズアンネットワークの学習

未知のデータへの予測を最大化する構造は

$$P(G | X) \propto P(G) \prod_{i=1}^N \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ijk})}{\Gamma(\alpha_{ij} + n_{ij})} \prod_{k=0}^{r_i-1} \frac{\Gamma(\alpha_{ijk} + n_{ijk})}{\Gamma(\alpha_{ijk})}$$

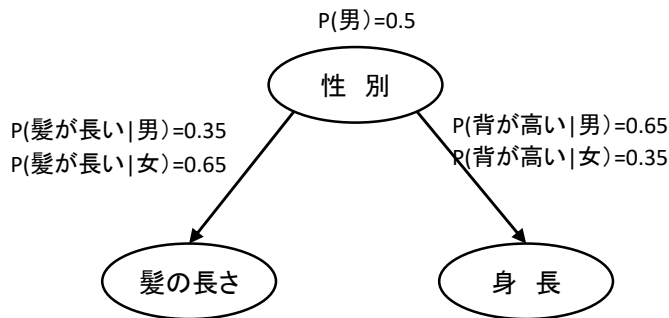
ここで、

$$\alpha_{ijk} = 1/K$$

K : モデルのパラメータ数

n_{ijk} : 変数 i が j 番目の親ノードパターン
を条件として k をとる頻度

ベイズアン・ネットワーク



$$p(\text{性別、髪の毛の長さ、身長} | G) = p(\text{性別})p(\text{髪の毛の長さ} | \text{性別})p(\text{身長} | \text{性別})$$

同時確率パラメータ
は $2^3 - 1 = 7$ 個

条件付き確率パラ
メータは 5 個

ベイジアンネットワークの問題

- 欠点として計算量の多さ
- 現在 厳密学習では、2000ノードのネットワーク(Natori, Uto, Ueno 2017))
- 厳密推論では200ノードのネットワーク(Li and Ueno 2017)
- 将来的にはこれを克服すれば最強ツールになる！！

マルコフネットワーク

- 無向グラフ構造
 - ベイジアンネットワークの互換モデル
- 利点
- 非循環有向グラフ構造の仮定がいらない。
- 欠点
- ベイジアンネットワークから変換できるがその逆は不可
 - 構造の学習ができない
 - パラメータ数が多い

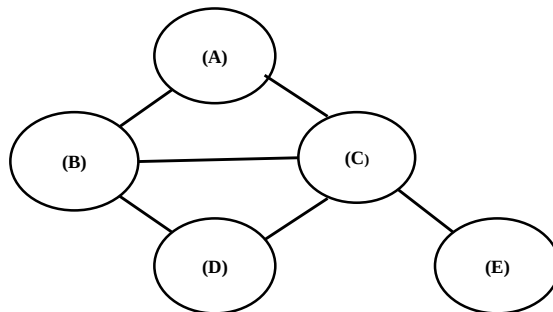
マルコフネットワークの同時確率分布 クリーク分解定理

- $P(x_1, x_2, \dots, x_N | G) = \frac{1}{Z(\theta)} \prod_{c \in C} \phi_c(x_c | \theta_c)$
- をギブス分布 (Gibbs Distribution) と呼ぶ。

C はクリーク集合

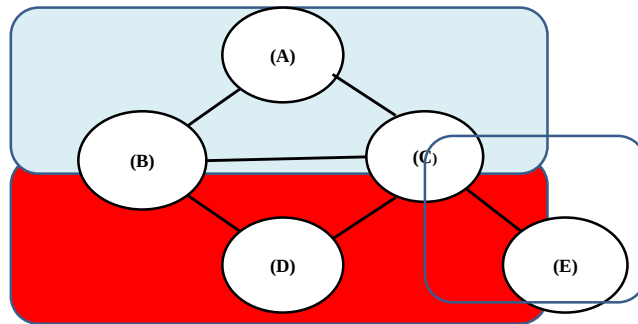
クリーク

- 頂点の部分集合が完全グラフ(すべての頂点間に辺がある)である場合



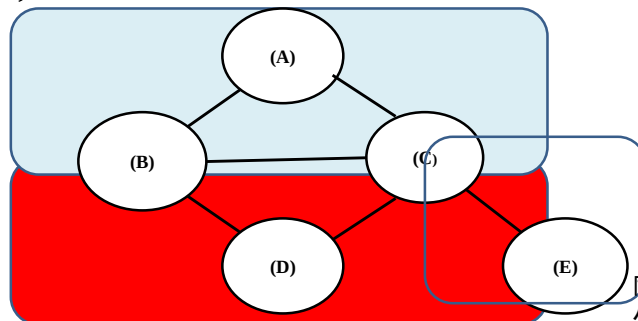
クリーク

- 頂点の部分集合が完全グラフ(すべての頂点間に辺がある)である場合



計算例 2値の場合

- $$P(x_A, x_B, \dots, x_E | G) = \frac{1}{Z(\theta)} p(x_A, x_B, x_C) p(x_B, x_C, x_D) p(x_C, x_E)$$



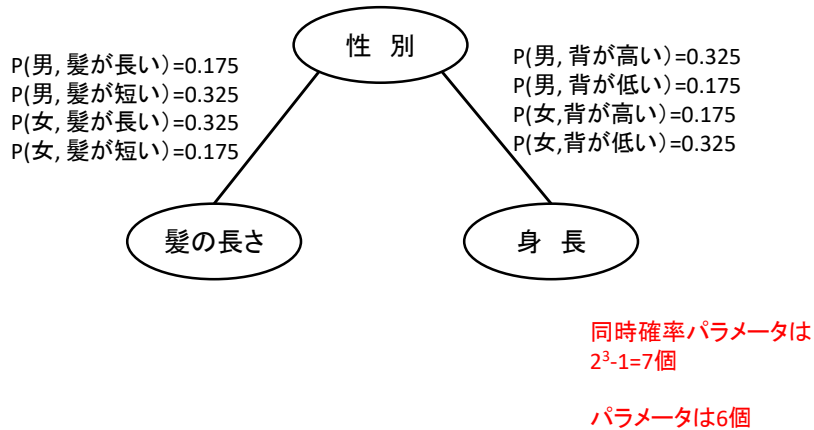
同時確率分布の
パラメータ数

31

⇒

7+7+3=17

マルコフ・ネットワーク



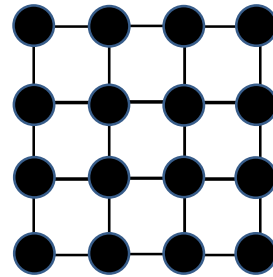
マルコフネットワークの問題

- パラメータ数はクリークの大きさに対して指数的に増え計算量が増えてしまうのでベイジアンネットワークより計算量が大きい。
- 数学的厳密性を緩和して計算が簡易なモデルの必要性

マルコフ確率場

$$P(x_1, x_2, \dots, x_N | G) = \frac{1}{Z(\theta)} \prod_i \phi(x_i) \prod_{(i,j)} \phi(x_i, x_j)$$

- クリークをまともに計算せず、グラフの辺ごとに分離して同時確率分布を近似する。
- 画像処理などで用いられる。



マルコフネットワークに戻ろう Log-Linear モデル

マルコフネットワークのファクターを

$$\phi_c(x_c | \theta_c) = \exp(-E(x_c | \theta_c))$$

と定義する。

ここで、 $E(x_c | \theta_c) = -\log(\phi_c(x_c | \theta_c)) > 0$ はクリーク c のエネルギー関数と呼ばれる。

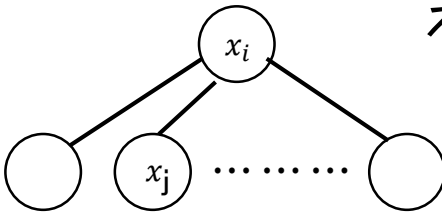
すなわち

$$P(x_1, x_2, \dots, x_N | G) = \frac{1}{Z(\theta)} \exp(-\sum_c E(x_c | \theta_c))$$

マルコフ確率場のLog-Linear モデル表現

$$P(x_1, x_2, \dots, x_N | G) = \frac{1}{Z(\theta)} \exp\left(-\sum_i E(X_i) - \sum_{(i,j)} E(X_i, X_j)\right)$$

x_i は二値しかとらずに以下の構造を考える



$$\begin{aligned} P(x_i = 1 | x_1, x_2, \dots, x_N, G) &= \frac{1}{Z(\theta)} \exp\left(-\sum_i E(x_i = 1) - \sum_{(i,j)} E(x_i = 1, x_j)\right) \\ &= \frac{\exp(-\sum_i E(x_i = 1) - \sum_{(i,j)} E(x_i = 1, x_j))}{\exp(-\sum_i E(x_i = 1) - \sum_{(i,j)} E(x_i = 1, x_j)) + \exp(-\sum_i E(x_i = 0) - \sum_{(i,j)} E(x_i = 0, x_j))} \\ &= \frac{1}{1 + \exp(-\sum_i E(x_i = 0) - \sum_{(i,j)} E(x_i = 0, x_j))} \end{aligned}$$

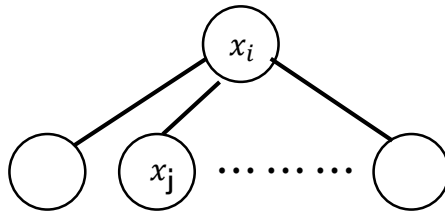
$E(x_i = 0) = b_i, E(x_i = 0, x_j) = w_{ij}x_j$ とおくと

ボルツマンマシン

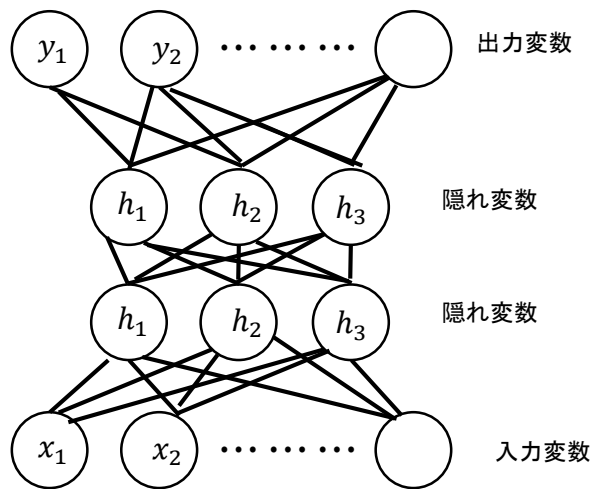
(ニューラルネットワーク)

$E(x_i = 0) = b_i$, $E(x_i = 0, x_j) = w_{ij}x_j$ とおくと

$$P(x_i = 1 | x_j \neq x_i) = \frac{1}{1 + \exp(-\sum_j w_{ij}x_j - b_i)}$$



深層学習モデル (ディープラーニング)



隠れ変数の役割

- 隠れ変数を積分消去すると
- 全変数間に辺が引かれた完全グラフ構造となる。
- 完全グラフ構造において、各辺の重みを最適化することにより、マルコフグラフの構造も同時に推定できる。
- 計算不可能な複雑な構造を 隠れ変数を導入することにより、単純で計算可能な階層構造に変換している。
- 真の確率構造が複雑な場合、隠れ変数層を増やさなければならないはず。
- ベイジアンネットワークで学習されるエッジ数が隠れ変数の数に関係している可能性が高い。

やはり脳モデルはすごい！！

- ビッグデータにおける同時確率分布の問題は変数の値のパターンがコンピュータや人間のメモリに入らないこと、計算速度が遅すぎることに、パターンが多すぎて空データが増えてしまうことである！！
- 脳モデルは メモリに乗らないほどの変数パターンは計算せず、すべて独立変数のように扱い、隠れ変数が仲介する階層モデルにより、結果として変数間の依存性を補完する。
- 計算速度、メモリ使用量、欠損データ、近似精度のトレードオフをすべて解決する！！

隠れ変数の数と構造の最適化が 今後のビッグチャレンジ

- 問題: 周辺尤度や従来の情報量規準は隠れ変数の数と構造を決めることはできない。ただ、モデルが正則性を満たさないということだけではない。
- 隠れ変数の数と構造を最適にする規準(スコア)は何か？
- ベイジアンネットワークで得られる構造のパラメータ数とどのような関係にあるのか？
- 数学的に解明できるのか？
- その構造を学習できるのか？

1. グラフィカルモデル

定義60

X, Y, Z が無向グラフ G の互いに排他なノード集合であるとする。もし、 X と Y の各ノード間の全ての路が Z の少なくとも一つのノードを含んでいるとき、 Z は X と Y を分離する、といい、 $I(X, Y|Z)_G$ と書く。これは、グラフ上での条件付き独立性を表現する。一方、真の条件付き独立性、すなわち、 X と Y と Z を所与として条件付き独立であるとき、 $I(X, Y|Z)_M$ と書く。

2. パーフェクトマップ

定義61 グラフ G は, ノードに対応する変数間すべての真の条件付き独立性がグラフでの表現に一致し, さらにその逆が成り立つとき, G をパーフェクトマップ (perfect map)といい, P-mapと書く.

$$I(X, Y|Z)_M \Rightarrow I(X, Y|Z)_G$$

しかし, すべての確率モデルに対応するグラフが存在するわけではない. 例えば, 四つの変 X_1, X_2, Y_1, Y_2 が以下の真の条件付き独立性をもっているとする.

$$\begin{aligned} I(X_1, X_2|Y_1, Y_2)_M, & \quad I(X_2, X_1|Y_1, Y_2)_M, \\ I(Y_1, Y_2|X_1, X_2)_M, & \quad I(Y_2, Y_1|X_1, X_2)_M. \end{aligned}$$

問題

- パーフェクト マップは グラフィカルモデリングには, 厳しすぎる制約！！

3. Iマップ

定義62 グラフ G は、グラフでの条件付き独立性のすべての表現が真の条件付き独立性に一致しているとき、 G をインデペンデントマップ(independent map)といい、I-mapと書く。

$$I(X, Y|Z)_G \Rightarrow I(X, Y|Z)_M.$$

I-mapの定義では、完全グラフを仮定するとグラフ上に $I(X, Y|Z)_G$ が存在しないので、どんな場合でもI-mapを満たしてしまう。

最小Iマップ

そこで、以下の最小I-mapが重要となる。

定義63 グラフ G がI-mapで、かつ、一つでもエッジを取り除くとそれがI-mapでなくなってしまうとき、グラフ G は**最小I-map** (minimal I-map) と呼ばれる。

問題

グラフ表現 $I(X, Y|Z)_G$ と確率表現 $I(X, Y|Z)_M$ が必ずしも対応しない。

問題

グラフ表現 $I(X, Y|Z)_G$ と確率表現 $I(X, Y|Z)_M$ が必ずしも対応しない。

↓

グラフ表現 $I(X, Y|Z)_G$ に対応する関係性を表現する手法はないのか？

問題

グラフ表現 $I(X, Y|Z)_G$ と確率表現 $I(X, Y|Z)_M$ が必ずしも対応しない。



グラフ表現 $I(X, Y|Z)_G$ に対応する関係性を表現する手法はないのか？



D分離の導入

4. D分離

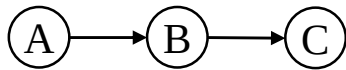


図1 逐次結合

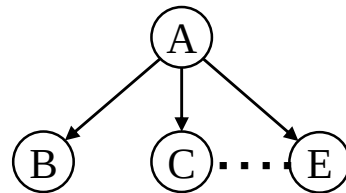


図2 分岐結合

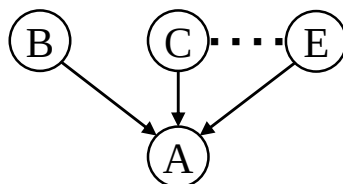
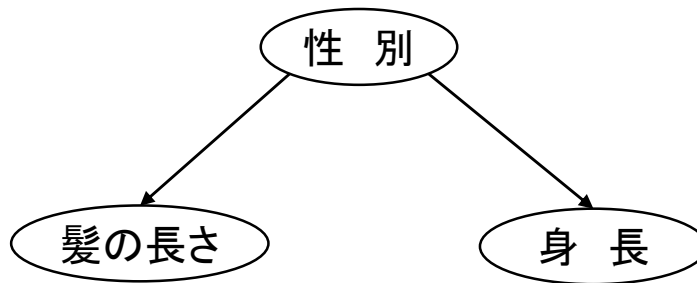


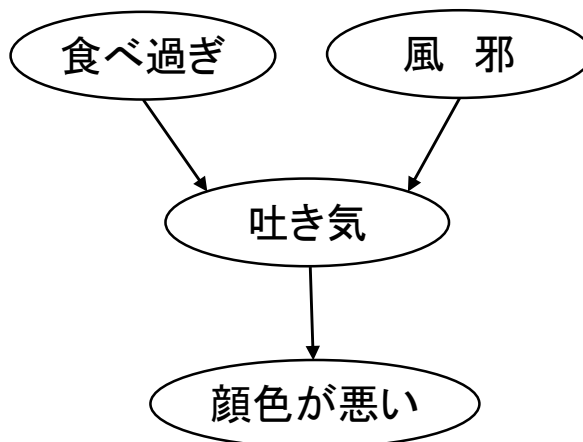
図3 合流結合

分岐結合の例



成人についての性別と髪の毛の長さ, 身長との因果ネットワーク

合流結合の例



合流結合のエビデンス

変数 $B, C, \dots, E$ は, A もしくは A の子孫についての証拠を得ない場合, 「 A を介して D 分離である」と呼ぶ. 変数についての証拠はその状態の確からしさの記述でもある. もし, その変数の値が観測されていた場合, 「変数がインスタンス化されている」と呼び, その値を「エビデンス (evidence)」と呼ぶ.

D分離

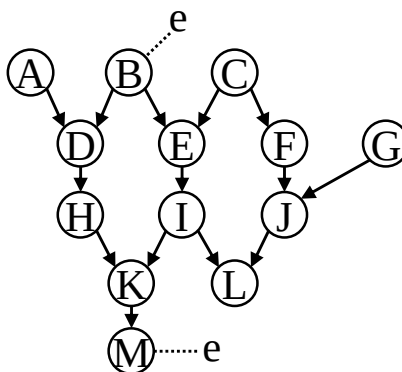
定義64 因果ネットワークにおける二つの変数 A と B は, A と B のすべてのパスに存在する以下のような変数 V (A と B を分ける) があるとき, **D分離** (D-separate) である.

- ・逐次結合もしくは分岐結合で V がインスタンス化されているとき, または,
- ・合流結合で V もしくは V の子孫がインスタンス化されていないとき,

D結合

- AとBがd分離でないとき, D結合 (D-connection) と呼ぶ.

例

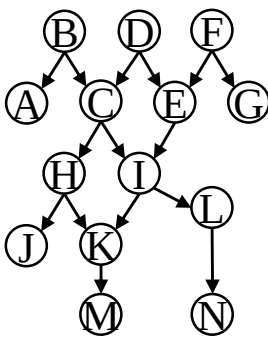


BとMがインスタンス化された
因果ネットワーク

定理(Darwiche 2009) ある因果ネットワークにおいて, 二つのノード集合 X と Y が, ノード集合 Z によってd分離されることは, 以下の枝刈り(pruning)によって得られる新しい因果ネットワークにおいて, X と Y が分離されていることと同値である.

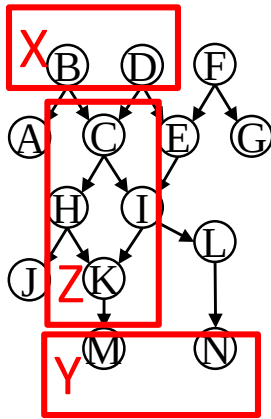
- ・ $X \cup Y \cup Z$ に属していない全てのリーフノード(leaf nodes)を削除する.
- ・ Z のノードから引かれたアークをすべて削除する.

例



図における因果ネットワークで, $X = \{B, D\}$, $Z = \{C, H, I, K\}$, $Y = \{M, N\}$ とする. $X \cup Y \cup Z = \{B, C, D, H, I, K, M, N\}$ であり, それ以外のノードを削除する. Z の要素からのアークをすべて削除すると, X と Y は完全に分離される. すなわち, X と Y がノード集合 Z を所与としてd分離されている.

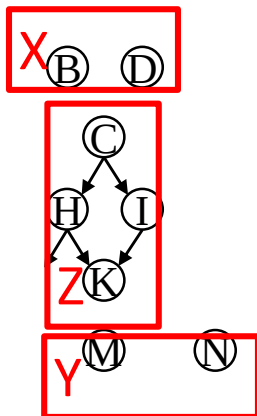
例



図における因果ネットワークで, $X = \{B, D\}$, $Z = \{C, H, I, K\}$, $Y = \{M, N\}$ とする. $X \cup Y \cup Z =$

$\{B, C, D, H, I, K, M, N\}$ であり, それ以外のノードを削除する. Zの要素からのアークをすべて削除すると, XとYは完全に分離される.

例



図における因果ネットワークで, $X = \{B, D\}$, $Z = \{C, H, I, K\}$, $Y = \{M, N\}$ とする. $X \cup Y \cup Z =$

$\{B, C, D, H, I, K, M, N\}$ であり, それ以外のノードを削除する. Zの要素からのアークをすべて削除すると, XとYは完全に分離される.

5. マルコフ ブランケット

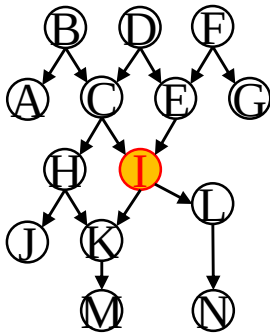
定義65 変数 A のマルコフブランケット (Markov blanket) とは, A の親集合, 子集合, A と子を共有している変数集合の和集合より成り立つ.

マルコフブランケットとD分離

ノート2 A についてのマルコフブランケットがすべてエビデンスがある場合, A はネットワーク中の残りの変数すべてとd分離である.

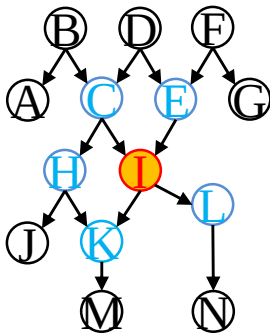
例

図において, II についてのマルコフブ
ランケットは？

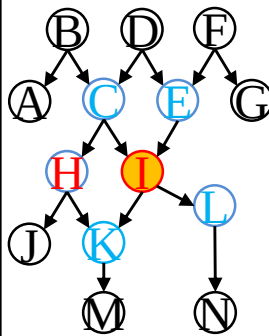


例

図において, II についてのマルコフブ
ランケットは(C,E,H,K,L)となる.



例



注: Iの近傍の変数のみが所与の場合, JはIとはd分離でないことに注意してほしい. なぜならば, この場合, Iはインスタンス化されないのであるが, Kを所与として合流結合である親HはIから影響を受け, Jにも影響を与える.

D分離の表記

定義66 X, Y, Z がグラフGのたがいに排他なノード集合であるとする. X と Y が Z によってD分離されるとき, $I(X, Y|Z)_d$ と書く.

D分離と最小Iマップ

定理19 $I(X, Y|Z)_d$ と $I(X, Y|Z)_G$ は同値である.

すなわち, $I(X, Y|Z)_d \Leftrightarrow I(X, Y|Z)_G$.

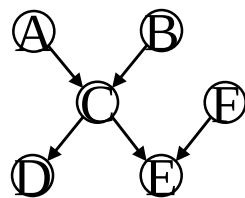
このように, D分離はグラフ上で表現するには都合のよい概念であり, この仮定がベイジアンネットワークのモデルそのものである. 真の条件付き独立性すべては表現できておらず, 最小I-mapを仮定していることになる.

モラルグラフを用いたD分離の定理

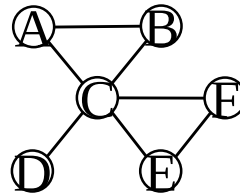
定義67 X, Y, Z がグラフ G のたがいに排他なノード集合であるとする. X, Y, Z を含む最小のモラルグラフで, Z が X と Y を分離するとき, Z は X と Y をD分離する.

例

左図の有向グラフを考える. これを無向化し, モラル化したグラフは右図である. 例えば, EはCとFをd分離しない, CはDとFをd分離する, などがわかる.



有向グラフ



モラルグラフ

6. ベイジアンネットワークモデル

定義 N 個の変数集合 $x = \{x_1, x_2, \dots, x_N\}$ をもつベイジアンネットワークは, (G, Θ) で表現される.

- G は x に対応するノード集合によって構成される **非循環有向グラフ** (directed acyclic graph, **DAG**) ネットワーク構造と呼ばれる.

- Θ は, G の各アークに対応する条件付き確率パラメータ集合 $\{p(x_i | \Pi_i, G)\}$, $(i = 1, \dots, N)$ である. ただし, Π_i は変数 x_i の親変数集合を示している.

同時確率分布

定理20 変数集合 $x = \{x_1, x_2, \dots, x_N\}$ をもつベイジアンネットワークの同時確率分布 $p(x)$ は以下で示される.

$$p(x|G) = \prod_i p(x_i | \Pi_i, G)$$

ここで, G は最小I-mapを示している.

演習

- 定理20を証明せよ.

逐次結合のd分離

[逐次結合の場合] AがBを通じてCに逐次結合している場合を考えよう. このとき, $p(C|B, A) = p(C|B)$ を示せばよい. 定理20より,

$$p(A, B, C) = p(A)p(B|A)p(C|B) = p(A, B)p(C|B).$$

よって

$$p(C|B, A) = \frac{p(A, B, C)}{p(A, B)} = p(C|B)$$

となる.

分岐結合のd分離

[分岐結合の場合] Aの独立な二つの子がB, Cであるとする. このとき, $p(C|A, B) = p(C|A)$ を示せばよい. 定理20より,

$$p(A, B, C) = p(A)p(B|A)p(C|A) = p(A, B)p(C|A).$$

よって

$$p(C|A, B) = \frac{p(A, B, C)}{p(A, B)} = p(C|A)$$

となる.

合流結合のd分離

[合流結合の場合] AとBをCの親としよう. いま, $p(A|B) =$

$p(A)$ を示せばよい. 定理20より,

$$p(A, B, C) = p(A)p(B)p(C|B, A).$$

Cについて周辺化し,

$$p(A, B) = \sum_C p(A)p(B)p(C|B, A).$$

ここで $\sum_C$ は $p(C|B, A)$ にのみかかっているので,

$$\sum_C p(A)p(B)p(C|B, A) = p(A)p(B) \sum_C p(C|B, A).$$

条件付き確率表で列の和は1になるので, $\sum_C p(C|B, A)$ は1となる. したがって

$p(A, B) = p(A)p(B)$ となるのである.

7. CPT(Conditional probabilities tables)

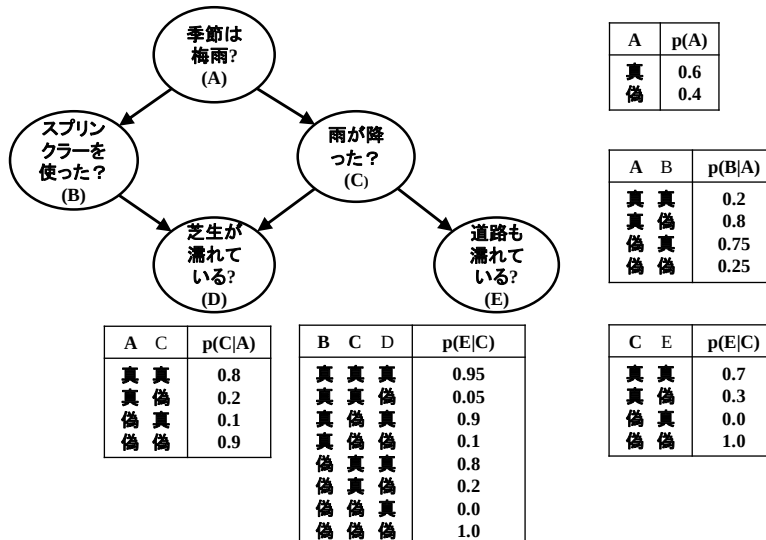


図7 バイジアンネットワークのCPT

演習：以下の確率を求めよ

$$p(A = 1, B = 1, C = 1, D = 1, E = 1|G)$$

8.同時確率分布表: joint probability distribution table, JPDT

表3.1 図3.2の同時確率分布表:JPDT

A	B	C	D	E	p(A,B,C,D,E)	A	B	C	D	E	p(A,B,C,D,E)
1	1	1	1	1	0.06384	0	1	1	1	1	0.01995
1	1	1	1	0	0.02736	0	1	1	1	0	0.00855
1	1	1	0	1	0.00336	0	1	1	0	1	0.00105
1	1	1	0	0	0.00144	0	1	1	0	0	0.00045
1	1	0	1	1	0.0	0	1	0	1	1	0.0
1	1	0	1	0	0.02160	0	1	0	1	0	0.24300
1	1	0	0	1	0.0	0	1	0	0	1	0.0
1	1	0	0	0	0.00240	0	1	0	0	0	0.02700
1	0	1	1	1	0.21504	0	0	1	1	1	0.00560
1	0	1	1	0	0.09216	0	0	1	1	0	0.00240
1	0	1	0	1	0.05376	0	0	1	0	1	0.00140
1	0	1	0	0	0.02304	0	0	1	0	0	0.00060
1	0	0	1	1	0.0	0	0	0	1	1	0.0
1	0	0	1	0	0.0	0	0	0	1	0	0.0
1	0	0	0	1	0.0	0	0	0	0	1	0.0
1	0	0	0	0	0.09600	0	0	0	0	0	0.09000

9. ファクター

定義69 変数集合 X のファクター (factor) φ (Jensen らはポテンシャル関数 (potential function) と呼ぶ) とは、変数集合 X の各値 x を非負値に写像させる関数であり、 $\varphi(x)$ と書く。

10. 周辺消去

実際には、各変数の状態確率に興味があるので、以下のように N 個の変数をもつ同時確率分布 $p(x) = (x_1, x_2, \dots, x_N | G)$ で対象となる変数 x_i 以外の変数を周辺化して $p(x_i | G)$ を求めればよい。

$$p(x_i | G) = \sum_{x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_N} p(x_1, x_2, \dots, x_N | G)$$

11. 周辺消去アルゴリズム

アルゴリズム6 (周辺事前確率のための変数消去アルゴリズム)

•Input: ベイジアンネットワーク $\{G, \Theta\}$, ベイジアンネットワークでのクエリ (query) 変数集合 Q

•Output: 周辺確率 $p(Q|G)$

1. Main
2. $S \leftarrow \text{CPTの値}$
3. For $i=1$ to N do
4. $\varphi \leftarrow \prod_k \varphi_k$, ここで φ_k は Q に含まれないノード i に関する (を含む) S に属する条件付き確率
5. $\varphi_i \leftarrow \sum \varphi$
6. S のすべての φ_k を φ_i によって置き換える.
7. end for
8. return $\prod_{\varphi \in S} \varphi$
9. end procedure

例

例38 例えば, 図7について周辺確率 $p(E = 1|G)$ をアルゴリズム6に従い計算すると以下ようになる.

$$\sum_D \sum_C p(E|C) \sum_B p(D|B, C) \sum_A p(A)p(B|A)p(C|A) = 0.364$$

例

変数消去の順を $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E$ の順としたとき、計算のステップは以下のようになる。

1. (step1) $i = 1$

消去する変数Aに関する変数はB,Cなので φ_k は以下の通り。

$$\begin{aligned}\varphi &\leftarrow \prod_k \varphi_k = p(A)p(B|A)p(C|A) \\ &= p(A, B, C)\end{aligned}\quad (1)$$

具体的なポテンシャルは表1のようになる。

(2)

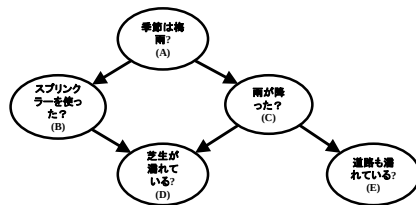


表1: $p(A, B, C)$

A	B	C	$p(A, B, C)$
真	真	真	$0.6 \times 0.2 \times 0.8 = 0.096$
真	真	偽	$0.6 \times 0.2 \times 0.2 = 0.024$
真	偽	真	$0.6 \times 0.8 \times 0.8 = 0.384$
真	偽	偽	$0.6 \times 0.8 \times 0.2 = 0.096$
偽	真	真	$0.4 \times 0.75 \times 0.1 = 0.03$
偽	真	偽	$0.4 \times 0.75 \times 0.9 = 0.03$
偽	偽	真	$0.4 \times 0.25 \times 0.1 = 0.01$
偽	偽	偽	$0.4 \times 0.25 \times 0.9 = 0.09$

次に、Aを消去したポテンシャルは以下のようになる。

$$\varphi_1 \leftarrow \sum_A \varphi = \sum_A p(A, B, C) \quad (3)$$

$$= p(B, C) \quad (4)$$

具体的なポテンシャルは表2のようになる。

表1: $p(A, B, C)$

A	B	C	$p(A, B, C)$
真	真	真	$0.6 \times 0.2 \times 0.8 = 0.096$
真	真	偽	$0.6 \times 0.2 \times 0.2 = 0.024$
真	偽	真	$0.6 \times 0.8 \times 0.8 = 0.384$
真	偽	偽	$0.6 \times 0.8 \times 0.2 = 0.096$
偽	真	真	$0.4 \times 0.75 \times 0.1 = 0.03$
偽	真	偽	$0.4 \times 0.75 \times 0.9 = 0.03$
偽	偽	真	$0.4 \times 0.25 \times 0.1 = 0.01$
偽	偽	偽	$0.4 \times 0.25 \times 0.9 = 0.09$

表2: $p(B, C)$

B	C	$p(B, C)$
真	真	$0.096 + 0.03 = 0.126$
真	偽	$0.024 + 0.27 = 0.294$
偽	真	$0.384 + 0.01 = 0.394$
偽	偽	$0.096 + 0.09 = 0.186$

例

2. (step2) $i = 2$

消去する変数Bに関する変数はC,Dなので φ_k は以下の通り.

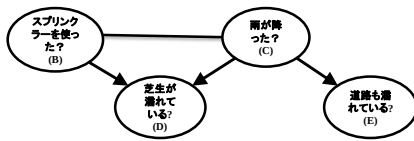
$$\varphi \leftarrow \prod_k \varphi_k = p(B, C) p(D | B, C) \quad (5)$$

$$= p(B, C, D) \quad (6)$$

具体的なポテンシャルは表3のようになる.

表3: $p(B, C, D)$

B	C	D	$p(B, C, D)$
真	真	真	$0.126 \times 0.95 = 0.1197$
真	真	偽	$0.126 \times 0.05 = 0.0063$
真	偽	真	$0.294 \times 0.9 = 0.2646$
真	偽	偽	$0.294 \times 0.1 = 0.0294$
偽	真	真	$0.394 \times 0.8 = 0.3152$
偽	真	偽	$0.394 \times 0.2 = 0.0788$
偽	偽	真	$0.186 \times 0.0 = 0.0$
偽	偽	偽	$0.186 \times 1.0 = 0.186$



例

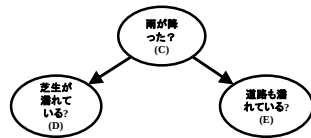
次に, Bを消去したポテンシャルは以下のようになる.

$$\varphi_2 \leftarrow \sum_B \varphi = \sum_B p(B, C, D) = p(C, D)$$

具体的なポテンシャルは表4のようになる.

表4: $p(C, D)$

C	D	$p(C, D)$
真	真	$0.1197 + 0.3152 = 0.4349$
真	偽	$0.0063 + 0.0788 = 0.0851$
偽	真	$0.2646 + 0.0 = 0.2646$
偽	偽	$0.0294 + 0.186 = 0.2154$



例

3. (step3) $i = 3$

消去する変数Cに関する変数はD,Eなので φ_k は以下の通り.

$$\begin{aligned}\varphi &\leftarrow \prod_k \varphi_k = p(C, D)p(E|C) \\ &= p(C, D, E)\end{aligned}\tag{9}$$

具体的なポテンシャルは表5のようになる.\tag{10}

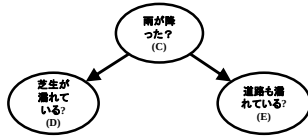


表5: $p(C, D, E)$

C	D	E	$p(C, D, E)$
真	真	真	$0.4349 \times 0.7 = 0.30443$
真	真	偽	$0.4349 \times 0.3 = 0.13047$
真	偽	真	$0.0851 \times 0.7 = 0.05957$
真	偽	偽	$0.0851 \times 0.3 = 0.02553$
偽	真	真	$0.2646 \times 0.0 = 0.0$
偽	真	偽	$0.2646 \times 1.0 = 0.2646$
偽	偽	真	$0.2154 \times 0.0 = 0.0$
偽	偽	偽	$0.2154 \times 1.0 = 0.2154$

例

次に, Cを消去したポテンシャルは以下のようになる.

$$\varphi_2 \leftarrow \sum_C \varphi = \sum_C p(C, D, E)\tag{11}$$

$$= p(D, E)\tag{12}$$

具体的なポテンシャルは表6のようになる.

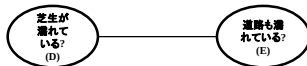


表6: $p(D, E)$

D	E	$p(D, E)$
真	真	$0.30443 + 0.0 = 0.30443$
真	偽	$0.13047 + 0.2646 = 0.39507$
偽	真	$0.05957 + 0.0 = 0.05957$
偽	偽	$0.02553 + 0.2154 = 0.24093$

例

4. (step4) $i = 4$

消去する変数Dに関する変数はEなので φ_k は以下の通り.

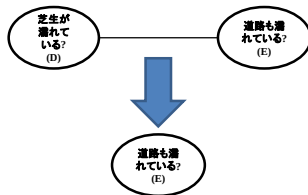
$$\varphi \leftarrow \prod_k \varphi_k = p(D, E)$$

次に, Dを消去したポテンシャルは以下ようになる.

$$\begin{aligned} \varphi_4 &\leftarrow \sum_D \varphi = p(D, E) \\ &= p(E) \end{aligned}$$

具体的なポテンシャルは表7のようになる.

表7: $p(E)$



E	$p(D, E)$
真	$0.30443 + 0.05957 = 0.364$
偽	$0.39507 + 0.24093 = 0.636$

例

5. (step4) $i = 4$

消去する変数はEだがEはクエリQに含まれているため処理を行わない.

6. return

S注に含まれるポテンシャルをすべて掛け合わせたものを出力する. この例では7が出力される.

ここでの変数の周辺化は, ベイジアンネットワークのいくつかの変数がインスタンス化される (エビデンスを得る) 前の事前確率分布について行われるものであり, 得られた各変数の周辺確率を周辺事前確率 (marginal prior) と呼び, この操作を事前分布周辺化 (prior marginals) と呼ぶ. それに対して, ベイジアンネットワークでいくつかの変数がインスタンス化 (エビデンスを得る) された場合の各変数の周辺確率を周辺事後確率 (marginal posterior) と呼び, この操作を事後分布周辺化 (posterior marginals) と呼ぶ.

9. エビデンスを得た後の周辺事後確率

エビデンス e を所与とした種変事後確率を計算する場合 (エビデンスを得た場合の変数消去の場合), まず同時周辺確率 (joint marginals) $p(Q, e|G)$ を計算する. そのために, エビデンスに一致しないファクターの値を0にするように, ファクターを以下のように再定義する.

定義70 エビデンス e を所与としたときのファクター $\varphi^e(x)$ は以下のように定義される.

$$\varphi^e(x) \begin{cases} \varphi(x) & (x \text{ が } e \text{ に一致しているとき}) \\ 0 & (\text{上記以外}) \end{cases}.$$

さらに, この変換について以下の分配法則が成り立つ.

定理21 φ_1 と φ_2 が二つの異なるファクターであり, エビデンス e を得たとき,

$$(\varphi_1 \varphi_2)^e = \varphi_1^e \varphi_2^e$$

が成り立つ.

10. 周辺事後確率のための変数消去アルゴリズム

アルゴリズム7 (周辺事後確率のための変数消去アルゴリズム)

•Input: ベイジアンネットワーク $\{G, \Theta\}$, ベイジアンネットワークでのクエリ変数集合 Q , エビデンス e

•Output: 周辺確率 $p(Q, e|G)$

1. Main
2. $S \leftarrow \varphi^e \leftarrow \varphi$
3. For $i=1$ to N do
4. $\varphi \leftarrow \prod_k \varphi_k$, ここで φ_k はノード i に関する (を含む) S に属する φ^e
5. $\varphi_i \leftarrow \prod_i \varphi$
6. S のすべての φ_k を φ_i によって置き換える.
7. end for
8. return $\prod_{\varphi \in S} \varphi$
9. end procedure

12. エビデンスを得た後の周辺事後確率

エビデンス e を所与とした種変事後確率を計算する場合 (エビデンスを得た場合の変数消去の場合), まず同時周辺確率 (joint marginals) $p(Q, e|G)$ を計算する. そのために, エビデンスに一致しないファクターの値を0にするように, ファクターを以下のように再定義する.

定義70 エビデンス e を所与としたときのファクター $\varphi^e(x)$ は以下のように定義される.

$$\varphi^e(x) \begin{cases} \varphi(x) & (x \text{ が } e \text{ に一致しているとき}) \\ 0 & (\text{上記以外}) \end{cases}.$$

さらに, この変換について以下の分配法則が成り立つ.

定理21 φ_1 と φ_2 が二つの異なるファクターであり, エビデンス e を得たとき,

$$(\varphi_1 \varphi_2)^e = \varphi_1^e \varphi_2^e$$

が成り立つ.

12. 周辺事後確率のための変数消去アルゴリズム

アルゴリズム7 (周辺事後確率のための変数消去アルゴリズム)

•Input: ベイジアンネットワーク $\{G, \Theta\}$, ベイジアンネットワークでのクエリ変数集合 Q , エビデンス e

•Output: 周辺確率 $p(Q, e|G)$

1. Main
2. $S \leftarrow \varphi^e \leftarrow \varphi$
3. For $i=1$ to N do
4. $\varphi \leftarrow \prod_k \varphi_k$, ここで φ_k はノード i に関する (を含む) S に属する φ^e
5. $\varphi_i \leftarrow \prod_i \varphi$
6. S のすべての φ_k を φ_i によって置き換える.
7. end for
8. return $\prod_{\varphi \in S} \varphi$
9. end procedure

演習

例39 今, エビデンス $e=\{A=1, B=0\}$ (真のとき1, 偽のとき0)とし,
 $Q=\{D, E\}$ とする.

各クエリの周辺確率を求めよ。

計算量

アルゴリズムで重要なことは、少ない計算量(短い時間)で、解に達することである。

例えば、 $2x^3 + 4x^2 + 2x + 5$ を計算するのに、

$2 \times x \times x \times x + 4 \times x \times x + 2 \times x + 5 \quad \dots \textcircled{1}$
 とすると、乗算の回数は6回になる。

この式を変形して、

$((2 \times x + 4) \times x + 2) \times x + 5 \quad \dots \textcircled{2}$
 とすれば、乗算の回数は3回になる。

一般化して、 n 次多項式、

$a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x^1 + a_0$
 を①のように計算するならば、乗算回数は

$(n+1) + n + (n-1) + \dots + 2 + 1 = \frac{1}{2}n(n+1) = \frac{1}{2}n^2 + \frac{1}{2}n \quad \dots \textcircled{3}$

になる。また、②の形式に変形して

$(\dots (a_n \times x + a_{n-1}) \times x) \dots) \times x + a_0$
 とすれば、乗算回数は n 回になる。

ここで、乗算の回数を計算量の尺度にしたのは、加減算に比べて乗除算に要する時間は非常に大きいからである。

このように、アルゴリズムでの主要な操作に着目して、解に達するまでに必要な操作回数を反映する量を**計算量**(オーダー)という。

計算量O（ビッグオー）

- $O(n)$ の式)の表現では、次のように簡略化する。
 - ・多項式になる場合、 $n \rightarrow$ 大としたとき最大になる項だけにする。
 (多項式③の場合: $(1/2)n^2 + (1/2)n$ を $(1/2)n^2$ とする)
 - ・係数を無視する。(Cの場合: $(1/2)n^2$ を n^2 とする)
 すなわち、多項式③の計算量は $O(n^2)$ となる。
- このようにする理由は、
 - ・計算量が問題になるのは、 n が大きいときに限られる。
 - ・ n の式による違いに対して、係数の違いは相対的に影響が小さい。

P問題

- 計算量が多項式 $O(n)$ になるアルゴリズムが存在する問題を**P問題**と呼ぶ。
- $a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x^1 + a_0$ の計算量は $O(n^2)$
- P問題は、数学者にとっては、効率的に解ける問題だとして扱われる。
 ところが、一般的な解法が見つからない問題や、解法はあるのだが、 $O(n^2)$ にはならない問題 ($O(2^n)$ や $O(n!)$) が多くある。それは**NP問題**に分類される。
- NPというのは多項式時間で正解が本当に正しいか判定できる問題のクラスのこと。

P問題の例

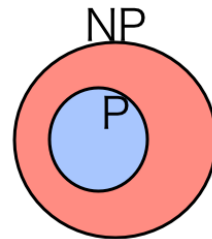
- 与えられたグラフが一筆書きできるか判定する問題はPである。
- なぜなら, すべての頂点の次数(その頂点から出ている辺の数)が偶数かどうかを調べればよい。
- $O(|V||E|)$

NP問題の例

- ハミルトン閉路問題(頂点を1回ずつ通って元に戻ってくる道があるのか判定する問題)はNPである。実際にそのような道を探すのは難しい(多項式時間で解く方法は発見されていない)が, 答えが与えられたらそれが確かに正しいということは簡単に確認できる。

PとNPの包含

- 多項式時間で解けるなら, 多項式時間で確認できるので, Pに属する問題はNPにも属する。NPはPを包含しているはずである。
- $P \neq NP$ 予想は, 「NPに属するのにPに属さない問題(図の赤い部分)」があるのかどうかをめぐる問題。



$P \neq NP$ 予想

- 数学的な関心事としては、
 $P = NP$ (本当はP問題として解けるのに、未だ発見されていないだけだ)
 $P \neq NP$ (本当にP問題にはならない問題があるのだ)
 のどちらなのだろうか、大きな課題となる。
- 未だに解けていない数学の大問題。
- $P \neq NP$ 予想が解けたら約一億円もらえる。

NP困難

- $P \neq NP$ 予想に基づき、NPに属すとは限らないが、NPに属する任意の問題と比べて、少なくとも同等以上に難しいこと

定理

- ベイジアンネットワーク推論は、NP困難問題である。
- 変数消去法は、計算量は $O(N^2 \exp(w))$

13. 枝刈りによる高速化

ベイジアンネットワークにおいて、変数消去法によるエビデンス e を所与としたクエリ変数 Q の確率推論は、 (Q, e) の組み合わせにより枝刈り (pruning) を適用し、高速化することができる。すなわち、以下の定理が知られている。

定理22 (Q, e) を所与としたとき、ノード集合 $(Q \cup e)$ に含まれない葉ノード集合を削除できる。

定理23 (Q, e) を所与としたとき、ノード集合 e から張られたすべてのエッジ集合を削除できる。

14. 例

例40 図7において、 $Q=\{D\}$ 、 $e:A=\text{真}(1)$ 、 $C=\text{偽}(0)$ が得られているとき、 $(Q \cup e)$ に含まれない葉ノード集合はノード E のみであり、ノード E が削除される。さらに、ノード集合 e から張られたすべてのエッジ集合は、ノード A, C から張られたすべてのエッジを削除すればよいので、結果として、図7の縮約されたベイジアンネットワーク構造を得ることができる。変換されたベイジアンネットワークについて、 $Q'=(Q \cup e)$ として e を新たなクエリ変数 Q' に組み込み、アルゴリズム6の周辺事前確率を求めるアルゴリズムに $Q' \rightarrow Q$ として適用すればよい。結果として、 $p(D, e|G)$ は

$$\begin{aligned}
 p(D=1, e|G) &= \sum_{B=\{0,1\}} \varphi^e(B) \varphi^e(D|B) \\
 &= 0.2 \times 0.9 + 0.8 \times 0.0 = 0.18 \\
 p(D=0, e|G) &= 0.2 \times 0.1 + 0.8 \times 1.0 = 0.82
 \end{aligned}$$

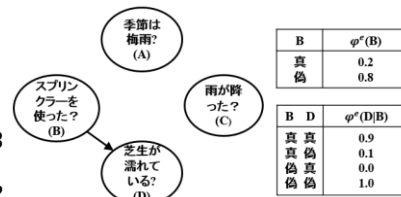


図14 図7についてクエリ $Q=\{D\}$ 、エビデンス $e=\{A=1, C=0\}$ を得た時の枝刈りされたベイジアンネットワーク

15. 変数消去順序の決定

本章では、ベイジアンネットワークの推論の基本手法である変数消去アルゴリズムを紹介した。ここでは、詳しく述べなかったが、アルゴリズム6や7の変数消去アルゴリズムでは、変数消去順序 (variables elimination order) が全体の計算量に大きく影響を及ぼすことが知られている。例えば、簡単なDAG: $A \rightarrow B \rightarrow C$ を考え周辺事前確率 $p(C|G)$ を求めたいとしよう。二つの変数消去順序 $\{A, B\}$, $\{B, A\}$ の二つの候補が挙げられるが、変数消去順序 $\{A, B\}$ の場合、

$$\sum_B p(C|B) \sum_A (p(A)p(B|A))$$

となり、 $\sum_A (p(A)p(B|A))$

$= p(B)$ なので、最初の変数消去で一変数のみが残される。

15. 変数消去順序の決定(続き)

変数消去順序 $\{A, B\}$ の場合に対し、変数消去順序 $\{B, A\}$ の場合、

$$\sum_A p(A) \sum_B (p(B|A)p(C|B))$$

となり、 $\sum_B (p(B|A)p(C|B)) = p(C|A)$ なので、最初の変数消去で二変数が残ってしま

う。1回目の変数消去でのアルゴリズム6の3行、4行での計算量が変数消去順序 $\{B, A\}$ のほうが、 $\{A, B\}$ よりも2倍大きくなってしまふことがわかる。このように、変数消去順序によりすべての変数消去プロセスのうち変数消去後に残される最大の変数数 w が決定され、さらに w がベイジアンネットワーク推論の計算量を決定していることがわかる。ただし、上の例では変数消去順序 $\{A, B\}$ では、 $w = 1$ 、変数消去順序 $\{B, A\}$ では、 $w = 2$ となる。

16. Widthと計算量

定理24 すべての変数消去プロセスのうち変数消去後に残される最大の変数数を w とすると, N 個の変数をもつベイジアンネットワークの変数消去アルゴリズムの計算量は $O(N^2 \exp(w))$ となる. ただし, w をベイジアンネットワークの”width” (ウィズ) と呼ぶ.

17.例

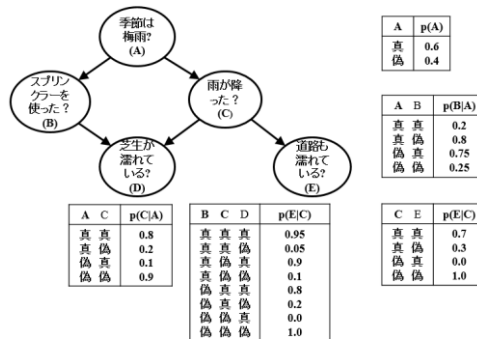


図3.12 ベイジアンネットワークのCPT¹⁾

表3.2 図3.12についての変数消去順序{B,C,A,D}でのwidth

i	Order	S	φ_i	width
1	B	$p(A), p(C A),$ $p(E C), \varphi_1(D A, C)$	$\varphi_1(D A, C)$ $= \sum_B p(B A) p(D B, C)$	3
2	C	$p(A), \varphi_2(D, E A)$	$\varphi_2(D, E A)$ $= \sum_C p(C A) p(E C) \varphi_1(D A, C)$	3
3	A	$\varphi_3(D, E)$	$\varphi_3(D, E) = \sum_A p(A) \varphi_2(D, E A)$	2
4	D	$\varphi_4(E)$	$\varphi_4(E) = \sum_D \varphi_3(D, E)$	1

17. インターラクショングラフ Widthの計算法 (Darwiche 2009)

定義71 $\varphi_1, \dots, \varphi_N$ をファクター集合とする. これらのファクターのインターラクショングラフ (interaction graph) G_i を以下のように定義する. G_i のノードは, $\varphi_1, \dots, \varphi_N$ に出現する各変数であり, もし二つの変数が同一のファクターに出現するとき, 対応する二つのノード間にエッジが張られる.

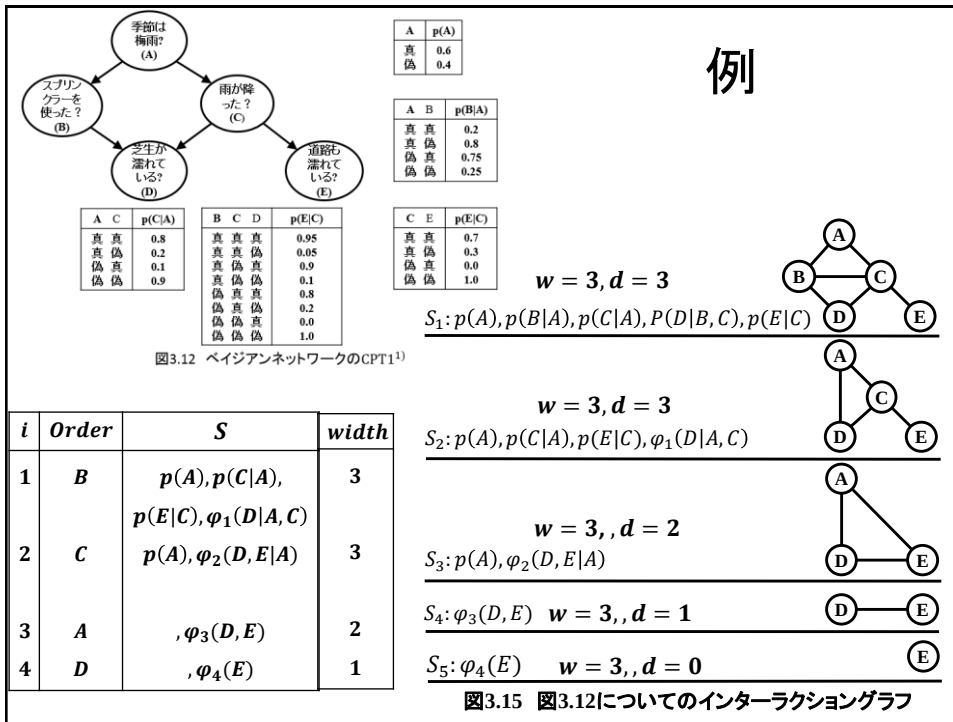
定義72 $\varphi_1, \dots, \varphi_N$ をファクター集合とする. インターラクショングラフ G_i のノードは $\varphi_1, \dots, \varphi_N$ に出現する各変数であり, 各ファクター φ_i に出現するノード集合でクリークを形成する.

アルゴリズム8 (Orderが与えられたときのwidth計算アルゴリズム)

• **Input:** ペイジアンネットワーク $\{G, \Theta\}$, *Order* : 変数消去順序

• **Output:** 変数消去順序 *Order* の *width*

1. Main
2. $G_i \leftarrow \text{CPTのインターラクショングラフ}$
3. $w \leftarrow 0$
4. For $i=1$ to N do
5. $w \leftarrow \max(w, d)$, ここで d は G_i におけるノード i の隣接ノード数
6. G_i におけるノード i の隣接ノード中でたがいに非隣接なノード間にエッジを張る.
7. G_i からノード i を消去する.
8. end for
9. return w
10. end procedure



18. 変数消去順序の最適化 最小次数法

当然,アルゴリズム8を用いてwidthを最小にする変数消去順序Orderを求めれば計算量を最適化できる. しかし, 残念なことに最適な変数消去順序Orderを求めることはNP困難な問題であり, 大きなベ이지アンネットワークには利用することができない.

そこで, 最適な変数消去順序Orderを求めるためのいくつかのヒューリスティック手法が知られている. 最も簡単な方法は変数数の少ないファクターから消去していく方法である. インターラクショングラフを用いると, 最小の隣接ノード数をもつノードから消去していけばよい. この手法は, 「最小次数法」(min-degree method) と呼ばれ, アルゴリズム9に示されている.

アルゴリズム9 (最小次数法アルゴリズム)

• **Input:** ベイジアンネットワーク $\{G, \Theta\}$

• **Output:** 変数消去順序 $Order$

1. Main
2. $G_i \leftarrow$ CPTのインターアクショングラフ
3. For $i=1$ to N do
4. $\pi(i), Order \leftarrow G_i$ の中で最小の隣接ノードをもつ変数
5. $\pi(i)$ の隣接ノード中でたがいに非隣接なノード間にエッジを張る.
6. G_i からノード $\pi(i)$ を削除する.
7. end for
8. return $Order$
9. end procedure

最小フィルイン法

もう一つの手法は、元のCPTのサイズをなるべく大きくしないように消去変数のフィルイン数を最小にするような順序づけ法である。アルゴリズム9の5行目の各変数のすべての隣接ノード中でたがいに非隣接なノード間にエッジ (フィルイン (fill-in)) を張った後、最小の隣接ノードをもつ変数の順に修正すればより有効であることが知られている。このアルゴリズムは「最小フィルイン (min-fill-in) 法」と呼ばれ、アルゴリズム10に示されている。

アルゴリズム10 (最小フィルイン法アルゴリズム)

• **Input:** ベイジアンネットワーク $\{G, \Theta\}$

• **Output:** 変数消去順序 $Order$

1. Main
2. $G_i \leftarrow$ CPTのインターラクショングラフ
3. For $i=1$ to N do
4. $\pi(i), Order \leftarrow G_i$ の中の各変数のすべての隣接ノード中でたがいに非隣接なノード間にエッジを張った場合に、その中で最小の隣接ノードをもつ変数
5. $\pi(i), Order \leftarrow G_i$ の中で最小の隣接ノードをもつ変数
6. G_i からノード $\pi(i)$ を削除する.
7. end for
8. return $Order$
9. end procedure

ハイブリッド手法

- 一般的には、最小次数法と最小フィルイン法を組み合わせたハイブリッド法が用いられる。

授業で学んだ厳密推論アルゴリズム

- 逐次変数消去法 $O(N^2 \exp(w))$
- 実際には、論文などで発表された正式なアルゴリズムではないが、理解しやすいので本授業では用いた。
- 市販のベイジアンネットワーク推論では厳密推論は用いられずに近似解が用いられるので結果が違うかもしれないことに注意。

ジョイントツリーアルゴリズム

- $O(N \exp(w))$
- クリークグラフが木で、二つのクラスターに属するすべての共通ノードがそれらの間に存在する路のすべてのクラスターにも属しているとき、ジョイントツリー (join tree) またはジャンクションツリー (junction tree) と呼ぶ。

すごく荒い説明

- もし、授業時間が大幅にあまりそうならそのときに詳しく説明します。

手順1 : Join treeの構築

BNの構造を次の手順で変形し join treeを構築

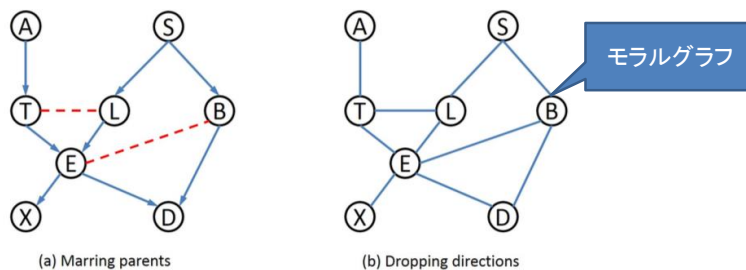
- 1.1: モラルグラフへの変形
- 1.2: 三角グラフへの変形(三角化と呼ぶ)
- 1.3: クリークの識別
- 1.4: Join treeの構築

ジョイントツリー

- クリークグラフが木で、二つのクラスターに属するすべての共通ノードがそれらの間に存在する路のすべてのクラスターにも属しているとき、ジョイントツリー (join tree) またはジャンクシオンツリー (junction tree) と呼ぶ。

手順1.1: モラルグラフへの変形

共通の子ノードを持つ親変数間にエッジを加え、アーケの方向を取り除くモラル化を実行



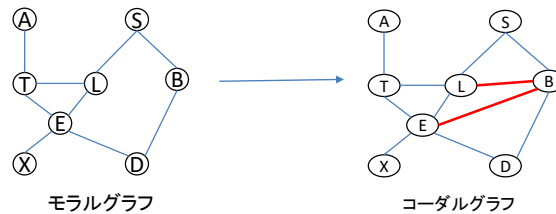
手順1.2: コーダルグラフ

コーダルグラフの定義

長さが4以上の全てのサイクルに少なくとも一個エッジがあるグラフ

コーダル化とは

長さが4つ以上のサイクルにエッジを挿入し、コーダルグラフに変換すること。コーダル化の仕方は一意ではなく、入れたエッジ集合により得られるコーダルグラフは変化する。

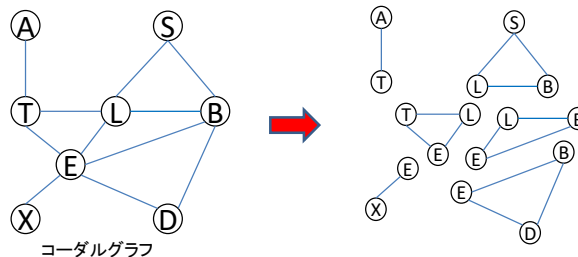


119

手順1.3: クリークの識別

三角グラフから全てのクリークを抽出

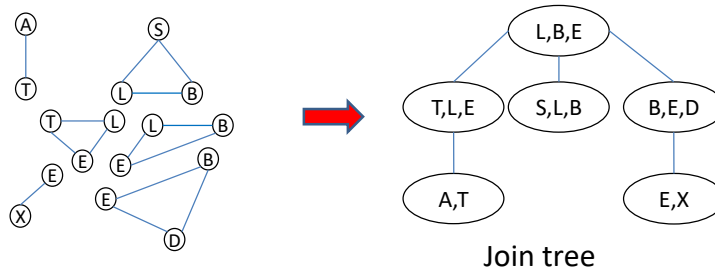
クリーク: 全ての頂点をエッジで繋いだ構造



120

手順1.4: Join tree の構築

クリーク間を接続してJoin treeを構築



121

Junction tree アルゴリズム

手順1: Join treeの構築

BNの構造をjoin treeに変形

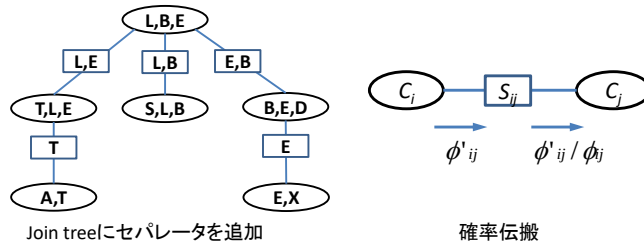
手順2: 確率推論

確率伝搬法により各変数の事後確率を計算

122

手順2: 確率推論

- 2.1: Join treeのセパレータ
(隣接するクリークの共通変数集合)の作成
- 2.2: クリークとセパレータの確率分布表の作成
- 2.3: 確率伝搬 (belief propagation) の実行
- 2.4: 各クリークの確率分布計算



123

まだ、計算量が大きい

- $O(N \exp(w))$
- 100変数が限界

市販のプログラムでは、厳密推論は使われていない。おおざっぱにいうと木に近似するプログラム。線形時間。

まとめ

1. AIとビッグデータの課題
2. グラフィカルモデル
3. 様々な グラフィカルモデル
4. 最も上位のベイジアンネットワーク
5. ベイジアンネットワークと様々な機械学習との関係
6. ベイジアンネットワークによる推論