4. 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12日　ベイズ的人工知能への招待
4月19日　ベイズの定理
4月26日　ベイズの定理
5月10日　　ベイズはコンピュータ、人工知能の父である！！
5月17日　　アランチューリングとベイズ
5月24日　　ベイズから機械学習へ
5月31日　　ベイズ推定ビリーフとベイズの定理
6月7日　　尤度推定と機械学習
6月14日　　尤度推定と機械学習2
6月21日　　ベイズ推定と機械学習
7月5日　　ベイズ的推定の基礎
7月12日　　MCMC法とベイズ推定
7月19日　　ベイジアンネットワーク
8月16日　　ベイジアンネットワークと他の機械学習モデルとの関係

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，$I(X,Y|Z)_{\mathrm{G}}$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，$I(X,Y|Z)_{\mathrm{M}}$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．

$I(X_1, X_2|Y_1, Y_2)$　　$I(Y_1, Y_2|X_1, X_2)$

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

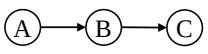


図1 逐次結合

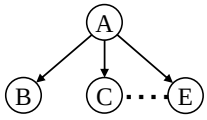


図2 分岐結合

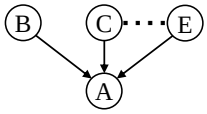
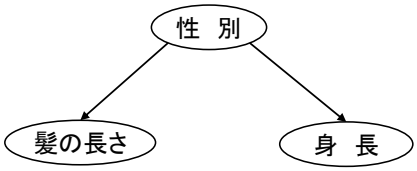


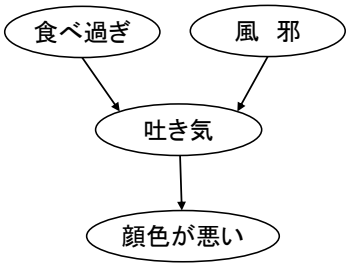
図3 合流結合

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,…,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．

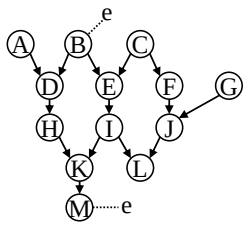
D分離

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，

例

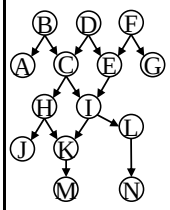


BとMがインスタンス化された
因果ネットワーク

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．

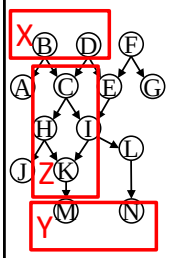
・$X \cup Y \cup Z$に属していない全てのリーフノード (leaf nodes) を削除する．

例



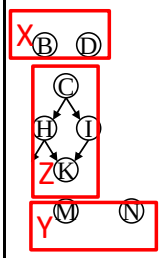
図における因果ネットワークで，$X = \{B, D\}$, $Z = \{C, H, I, K\}$, $Y = \{M, N\}$とする．$X \cup Y \cup Z = \{B, C, D, H, I, K, M, N\}$であり，それ以外のノードを削除する．Zの要素からのアークをすべて削除すると，XとYは完全に分離される．

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．

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

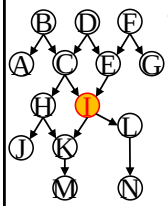
定義65 変数Aのマルコフブランケット(Markov blanket)とは，Aの親集合，子集合，Aと子を共有している変数集合の和集合より成り立つ．

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．

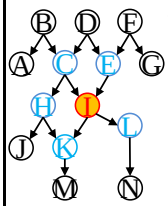
例

図において，Iについてのマルコフブランケットは？

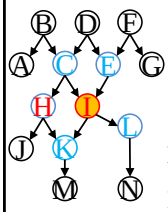


例

図において，I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，　$I(X,Y|Z)_d \Leftrightarrow I(X,Y|Z)_G$.

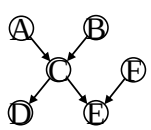
このように，d分離はグラフ上で表現するには都合のよい概念であり，この仮定がベイジアンネットワークのモデルそのものである．真の条件付き独立性すべては表現できておらず，最小I-mapを仮定していることになる．

モラルグラフを用いたD分離の定理

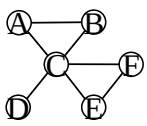
定義67　X,Y,ZがグラフGのたがいに排他なノード集合であるとする．X,Y,Zを含む最小のモラルグラフで，ZがXとYを分離するとき，ZはXとYをd分離する．

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68　N個の変数集合$x = \{x_1, x_2, \ldots, x_N\}$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$\{p(x_i|\Pi_i, G)\}$，$(i = 1, \ldots, N)$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.

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

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,

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

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,

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

Cについて周辺化し,

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

ここで$\sum_C$ は$p(C|B,A)$にのみかかっているので,

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

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)

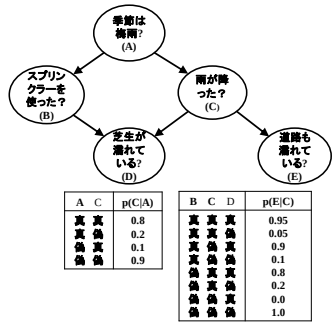


図3.12　ベイジアンネットワークのCPT[1)]

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，φ(x)と書く.

カルバックライブラー

- P 、Q を離散確率分布とするとき、P の Q に対する**カルバック・ライブラー情報量**は以下のように定義される。
- $\mathrm{D}=\sum_i\ P(i)log\frac{P(i)}{Q(i)},\ \mathrm{D}\geq 0$
- 確率分布PとQの乖離度を示す。
- ベイズでは情報利得I Information Gain)

ベイジアンネットワークでのKL

定理

- P 、Q をベイジアンネットワーク確率分布とするとき、P の Q に対するカルバック・ライブラー情報量は以下で求められる。
- $\mathrm{D}=\sum_i^N\sum_{\mathrm{j}}^{q_i}P(\mathrm{x}_i,\pi_i)log\frac{P(\mathrm{x}_i|\pi_i)}{Q(\mathrm{x}_i|\pi_i)}$

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$\mathrm{p}(x)=(x_1,x_2,\dots,x_N|\mathrm{G})$で対象となる変数$x_i$以外の変数を周辺化して$\mathrm{p}(x_i|\mathrm{G})$を求めればよい.

$$\mathrm{p}(x_i|\mathrm{G})=\sum_{x_1,\dots,x_{i-1},x_{i+1},\dots,x_N}p(x_1,x_2,\dots,x_N|\mathrm{G})\quad(3.2)$$

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,Θ}, 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$p(Q|\mathrm{G})$

1. Main
2. S←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に関係する (を含む) Sに属する条件付き確率
5. $\varphi_i \leftarrow \sum_i \varphi$
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

例

例38 例えば, 図3.12について周辺確率$p(E=1|\mathrm{G})$をアルゴリズム6に従い計算すると以下のようになる.

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

例

変数消去の順をA→B→C→D→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に関係する変数はB,Cなのでφ_kは以下の通り.

$$\varphi \leftarrow \prod_k \varphi_k = p(A)p(B|A)p(C|A) \quad (1)$$
$$= p(A,B,C) \quad (2)$$

具体的なポテンシャルは表1のようになる.

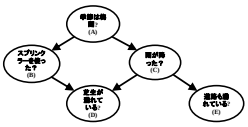


表1:p(A,B,C)

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	$0.4 \times 0.25 \times 0.9 = 0.09$

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に関係する変数は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.

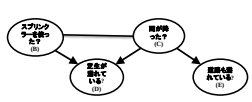


表3:p(B,C,D)

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.

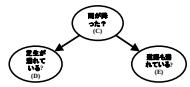


表4:p(C,D)

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に関係する変数はD,Eなのでφ_kは以下の通り.

$$\varphi \leftarrow \prod_k \varphi_k = p(C,D)p(E|C) \quad (9)$$
$$= p(C,D,E) \quad (10)$$

具体的なポテンシャルは表5のようになる.

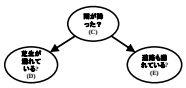


表5:p(C,D,E)

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.

$$\varphi_2 \leftarrow \sum_C \varphi = \sum_C p(C,D,E) \quad (11)$$
$$= p(D,E) \quad (12)$$

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に関係する変数は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のようになる.

$$\varphi_4 \leftarrow \sum_D \varphi = p(D,E)$$
$$= p(E)$$

具体的なポテンシャルは表7のようになる.

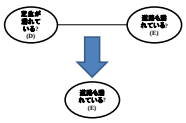


表7:p(E)

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) $p(Q,e|\mathrm{G})$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$\varphi^e(\mathrm{x})$は以下のように定義される.

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

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$\{\mathrm{G},\Theta\}$，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$p(Q,e|\mathrm{G})$

1. Main
2. S←$\varphi^e \leftarrow \varphi$
3. For i=1 to N do
4. $\varphi \leftarrow \prod_k \varphi_k$,ここで$\varphi_k$はノードiに関係する (を含む) Sに属する$\varphi^e$
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) $p(Q,e|\mathrm{G})$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$\varphi^e(\mathrm{x})$は以下のように定義される．

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

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$\{\mathrm{G},\Theta\}$，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$p(Q,e|\mathrm{G})$

1. Main
2. S←$\varphi^e \leftarrow \varphi$
3. For i=1 to N do
4. 　$\varphi \leftarrow \prod_k \varphi_k$,ここで$\varphi_k$はノードiに関係する (を含む) Sに属する$\varphi^e$
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、

$$2\times x\times x\times x+4\times x\times x+2\times x+5 \quad \cdots ①$$

とすると、乗算の回数は6回になる。
この式を変形して、

$$\big(\big((2\times x+4)\times x+2\big)\times x\big)+5 \quad \cdots ②$$

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、

$$a_nx^n+a_{n-1}x^{n-1}+\cdots+a_1x^1+a_0$$

を①のように計算するならば、乗算回数は

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

になる。また、②の形式に変形して

$$(\cdots(a_n\times x++a_{n-1})\times x)\cdots)\times x+a_0$$

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→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。
- $a_nx^n+a_{n-1}x^{n-1}+\cdots+a_1x^1+a_0$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取り扱われる。
  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

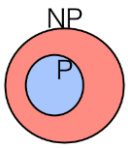
- 与えられたグラフが一筆書きできるか判定する問題はPである。
- なぜなら，すべての頂点の次数（その頂点から出ている辺の数）が偶数かどうかを調べればよい。
- O(|V||E|)

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≠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≠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≠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が、大きな課題となる。
- 未だに解けていない数学の大問題。
- P≠NP予想が解けたら約一億円もらえる。

NP困難

- P≠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 ∪ 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3.12において，Q={D}，e：A=真 (1)，C=偽 (0) が得られているとき，(QUe) 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3.14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'= (QUe) 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'→Qとして適用すればよい．結果として，p(D,e|G)は

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

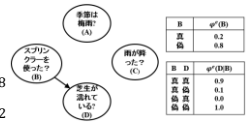


図3.14 図3.12についてクエリQ={D}, エビデンスe={A=1,C=0}を得た時の枝刈りされたベイジアンネットワーク

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→B→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うことがわかる．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

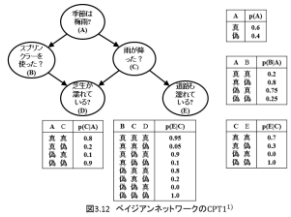


図3.12 ベイジアンネットワークのCPT[1)]

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　$\varphi_1, \ldots, \varphi_N$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) $\mathcal{G}_i$を以下のように定義する．$\mathcal{G}_i$のノードは，$\varphi_1, \ldots, \varphi_N$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　$\varphi_1, \ldots, \varphi_N$をファクター集合とする．インターラクショングラフ$\mathcal{G}_i$のノードは$\varphi_1, \ldots, \varphi_N$に出現する各変数であり，各ファクター$\varphi_i$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**:ベイジアンネットワーク{G,Θ}，　*Order* :変数消去順序

・**Output**：変数消去順序*Order*の*width*

1. Main
2. $\mathcal{G}_i \leftarrow$ 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$\mathcal{G}_i$におけるノードiの隣接ノード数
6. $\mathcal{G}_i$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. $\mathcal{G}_i$からノードiを消去する．
8. end for
9. return w
10. end procedure

例

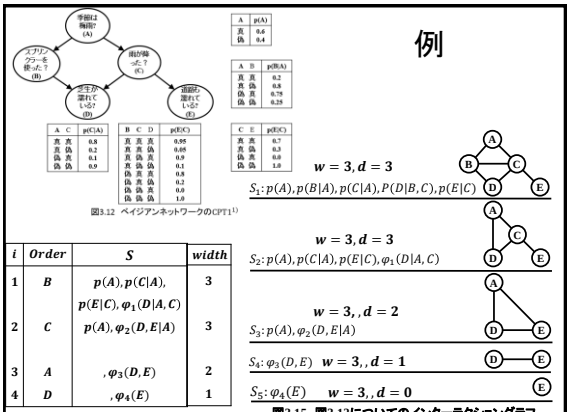


図3.12　ベイジアンネットワークのCPT[1)]

i	$Order$	S	$width$
1	B	$p(A), p(C\|A), p(E\|C), \varphi_1(D\|A,C)$	3
2	C	$p(A), \varphi_2(D,E\|A)$	3
3	A	$, \varphi_3(D,E)$	2
4	D	$, \varphi_4(E)$	1

図3.15　図3.12についてのインターラクショングラフ

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，大きなベイジ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,Θ}

・**Output**：変数消去順序*Order*

1. Main
2. $\mathcal{G}_i \leftarrow$ CPTのインターラクショングラフ
3. For i=1 to N do
4. $\pi(i), Order \leftarrow \mathcal{G}_i$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. $\mathcal{G}_i$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,Θ}

・**Output**: 変数消去順序*Order*

1. Main
2. $\mathcal{G}_i \leftarrow$ CPTのインターラクショングラフ
3. For i=1 to N do
4. $\pi(i), Order \leftarrow \mathcal{G}_i$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. $\pi(i), Order \leftarrow \mathcal{G}_i$の中で最小の隣接ノードをもつ変数
6. $\mathcal{G}_i$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。

ジョインツリーアルゴリズム

- $O(\mathrm{N}\exp(w))$
- 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　コーダルグラフへの変形（コーダル化と呼ぶ）

1.3：クリークの識別

1.4：Join treeの構築

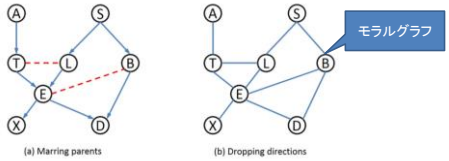
84

ジョインツリー

- 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，**ジョインツリー**（join tree）または**ジャンクションツリー**（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、アークの方向を取り除くモラル化を実行



86

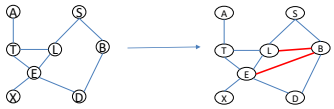
手順1.2：コーダルグラフ

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。

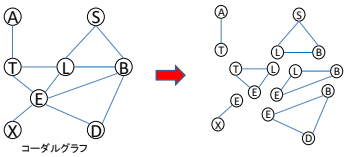


87

手順1.3：クリークの識別

コーダルグラフから全ての**クリーク**を抽出

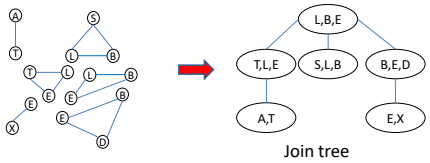
クリーク：全ての頂点をエッジで繋いだ構造



88

手順1.4：Join tree の構築

クリーク間を接続してJoin treeを構築



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

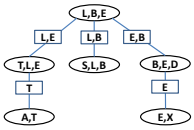
手順2：確率推論

確率伝搬法により各変数の事後確率を計算

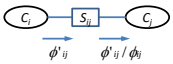
90

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



Join treeにセパレータを追加



確率伝搬

91

まだ、計算量が大きい

- $O(\mathrm{Nexp}(w))$
- 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。

授業時間が余ったら

- 最新の推論についての詳しいアルゴリズムを紹介します！！