エージェントシステム：第6回講義参考資料

稲邑　哲也

2003 年 11 月 8 日

1　確率的推論システム

1.1　ベイジアンネットワーク

ベイジアンネットワーク (Bayesian Network) は，別名信念ネットワーク，確率ネットワーク，因果ネットワークなどとも呼ばれ，不確実な状況での推論問題に利用される表現形式である．ベイジアンネットワークは以下の性質を有するグラフである．

- 確率変数の集合がネットワークのノードを形成する
- リンクまたは→の集合がノード対を結ぶ．ノード X からノード Y への矢印の直感的な意味は，X が Y に直接的影響を及ぼすということである．
- 各ノードは親ノードがそのノードへ及ぼす影響を定量化した条件付き確率表を持つ．ノードの親とは，そのノードを矢印によって指す全てのノードである．
- グラフは矢印の方向にサイクルを持たない．従って有向非循環グラフ (DAG) である．

具体例を図 1 に示す．これは John と Mary の隣人がおり，この二人は警報を聞けば仕事中のあなたに電話してくれると約束してるとする．John は警報を聞けば必ず電話してくれるが，時々電話のベルを警報と間違えて電話してくることがある．一方 Mary は大音響の音楽が好きなので，ときどき警報を聞き逃すことがある．誰が電話してきたかという証拠の元で，泥棒の確率を好いてする事を考えるネットワークである．

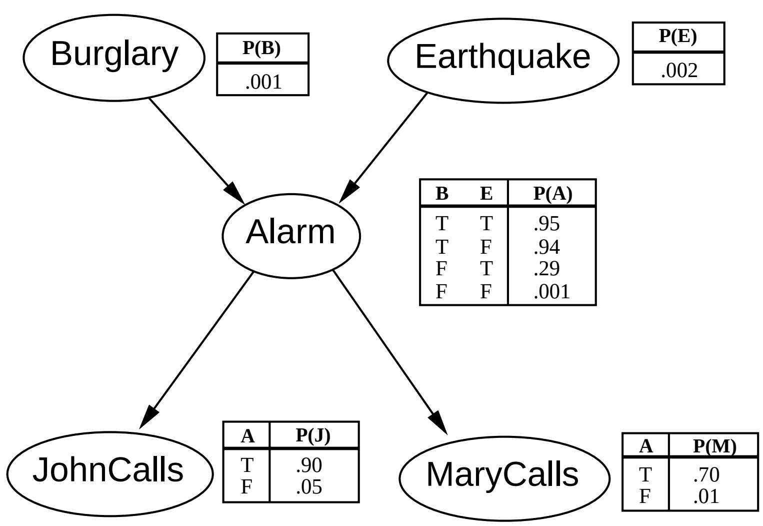

図 1: 条件付き確率テーブルを持つ典型的なベイジアンネットワーク

ベイジアンネットは扱うドメインに含まれる確率変数によって構成されるため，全ての確率変数によって構成される結合確率分布表現を代表しているとも捉える事ができる．例えば，1 の場合，$P(Buglary \wedge \neg Earthquake \wedge$

表 1: 3 つのノイズパラメータで表現される noisy-OR の条件付き確率テーブル

$Cold$	Flu	$Malaria$	$P(Fever)$	$P(\neg Fever)$
F	F	F	0.0	1.0
F	F	T	0.9	0.1
F	T	F	0.8	0.2
F	T	T	0.98	$0.02 = 0.2 \times 0.1$
T	F	F	0.4	0.6
T	F	T	0.94	$0.06 = 0.6 \times 0.1$
T	T	F	0.88	$0.12 = 0.6 \times 0.2$
T	T	T	0.977	$0.012 = 0.6 \times 0.2 \times 0.1$

$Alarm \wedge JohnCalls \wedge MaryCalls)$ を求めたい場合，それぞれの確率変数を B, E, A, J, M として，

$$P(B \wedge \neg E \wedge A \wedge J \wedge M) \tag{1}$$

$$= \quad P(B)P(\neg E)P(A|B \wedge \neg E)P(J|A)P(M|A) \tag{2}$$

として，確率値を求めることができる．この条件付き確率は，直接的な因果関係を持つ確率変数間における部分的な結合確率分布であるとも解釈できる．この部分的な表現はモデルを表現するのに必要なパラメータの次元数を削減し，推論の計算の複雑性を現象させる効果がある．例えば．全体で n 個のブール代数を持つ確率変数からなるドメインがあったとすると，全体の結合確率分布は 2^n 個の確率値を求める必要があるが，それを k 個の部分的な確率変数ごとに分割して表現すると，$n2^k$ の計算量で済む．例えば，20 ノード $n = 20$ で各々が最大 5 の親ノード ($k = 5$) を持つとすると，ベイジアンネットワークは 640 個の数値を必要とするが，全結合確率の場合だと 100 万個以上の数値が必要となる．すなわち，ベイジアンネットワークは，少ない次元数でドメイン全体の結合確率分布を表現する手段であると言える．

この考え方をさらに進めて，なるべく少ない数値で，なるべく大きな領域の結合確率分布を求める手法として，noisy-OR, noisy-MAX などの決定的ノードと呼ばれる結合手法がある．例えば，Cold, Flu, Malaria が真である時，しかもその時に限り Fever は真である，という状況を表現する場合，noisy-OR が使用される．もし，$P(Fever|Cold) = 0.4, P(Fever|Flu) = 0.8, P(Fever|Malaria) = 0.9$ ならこの 3 つの数値を用いるだけで，次の表 1 ような条件付き確率テーブルを定義することができる．

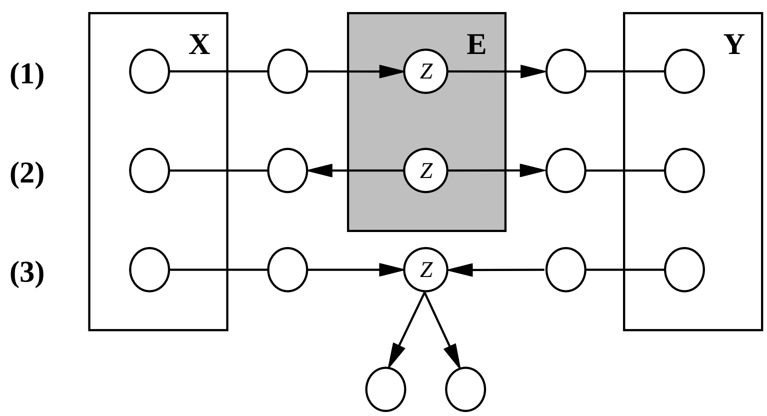

図 2: d-分離の例：(1) では Z は X と Y を d-分離している．(2) においても Z は X と Y を d-分離している．(3) では Z は X と Y を d-分離してはいない．

この計算を効率よく行う事ができるアルゴリズムが存在する事が，ベイジアンネットワークの利点の一つとなっている．

1.2 d-分離

ネットワークが与えられ証拠ノード E が与えられているときに，ノード X の集合が他のノード集合 Y と独立であるかどうか読みとることが可能であるか？この問題は推論やネットワーク設計に影響する問題であり，d-分離（方向依存性分離）という概念で解くことが出来る．d-分離の定義は,「X 中の有るノードから Y 中のあるノードへのすべての無向経路が E により d-分離されているならば，E の下で X と Y は条件付き独立である」となる．

例えば，図 3 において d-分離について分析すると次のようになる．Ignition ノードは，Radio ノードと Gas ノードを d-分離する．Battery ノードは，Radio ノードと Gas ノードを d-分離する．Starts ノードは，Radio ノードと Gas ノードを d-分離しない．

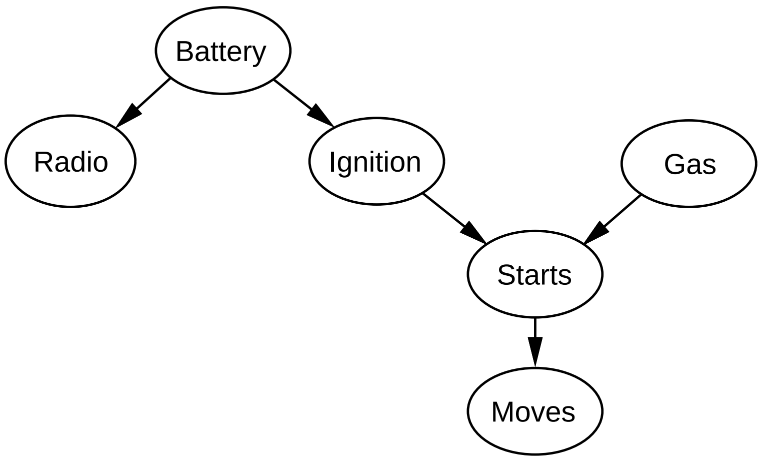

図 3: Battery, Ignition, Starts は Radio と Gas を d-分離するか？

1.3 確率的推論の性質と種類

ベイジアンネットワークの推論の一つの特徴として，推論の対象となる「質問ノード」と，その推論を行う初期条件となる「証拠ノード」を，ネットワーク上のどのノードに設定しても問題なく推論が遂行できるという点がある．すなわち，P(質問 | 証拠) を計算するのが確率的推論ということになる．質問ノードと証拠ノードがネットワーク上にどのように配置しているかによって推論の種類が異なる．図 4 はその推論の方式ごとに分類したもので，診断的推論，因果的推論，原因間推論，その複合系がある．

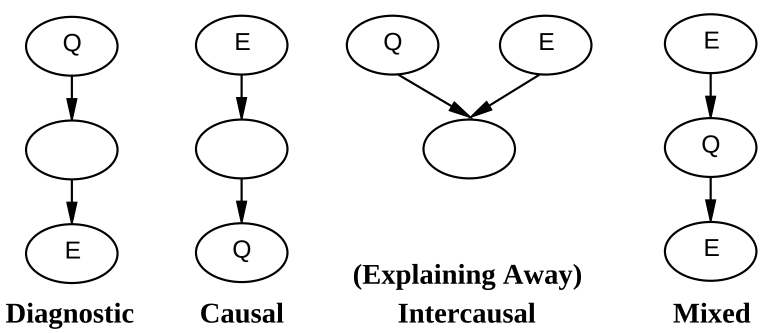

図 4: ベイジアンネットワークにおける 4 種類の推論の形式

ベイジアンネットワークを用いた推論の手法は，根本的に考えると，結合確率分布を用いた確率値の計算ということになるが，工夫をすることで計算量や計算の手間を抑えることができる．推論には，証拠として入力されたノードに数値を代入し，そこから情報を伝播させて，推論の対象となっているノードの確率変数の値を計算す

る事ができる．詳細については付録に記す．この計算の手法はニューラルネットワークにおける情報伝播とも類似性がある．

情報を伝播させる事からわかるとおり，ネットワークが閉ループ構造を持っていると問題が生じる．無限に情報が伝播し続けるからである．

1.4 閉ループを含むネットワークにおける推論

上記の推論アルゴリズムでは，図 5 のようにネットワーク構造に閉ループが存在した場合，情報伝播が無限に繰り返されてしまい，計算が終了しない．閉ループが存在した場合の推論手法として，以下の３通りの手法が存在する．

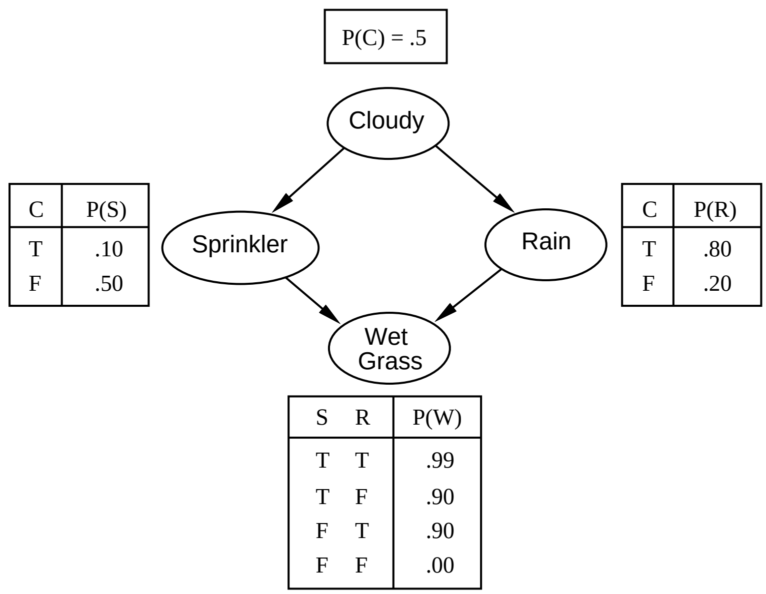

図 5: 閉ループを含むベイジアンネットワーク

- クラスタリング法
- カットセット条件付け法
- 確率的論理シミュレーション法

クラスタリング法では，図 6 のように確率変数を統合することでメガノードを形成し，閉ループ構造が通常の多重木構造になるような変換を行う．クラスタリング法は線形時間アルゴリムの利用を可能にするが，問題の NP 困難性は解消されない．逆にネットワークの大きさは指数的に像だする危険も含む．

カットセット条件付け法では，図 7 のように，ネットワークを一つの複雑な多重木に変形するのではなく，ネットワークを複数の簡単な多重木に変形する．単純な各ネットワークは確定的に具体化された一つ以上の変数をカットセットとして持つ．

確率論的シミュレーション法では，ベイジアンネットワークで記述される世界のシミュレーションを繰り返し，関連事象が発生した頻度を計数する事によって，求めたい確率を推定する．シミュレーションの各ラウンドは，ネットワークの各根ノードに対して事前確率により重み漬けしながらランダムに値を選ぶ．以後，CPT の事後確率に従って値を選んでいく．この手法は常に正しい値に収束するが，入力証拠に対して滅多に起こらない値の割り当てに興味を持っている場合に問題が生じる．

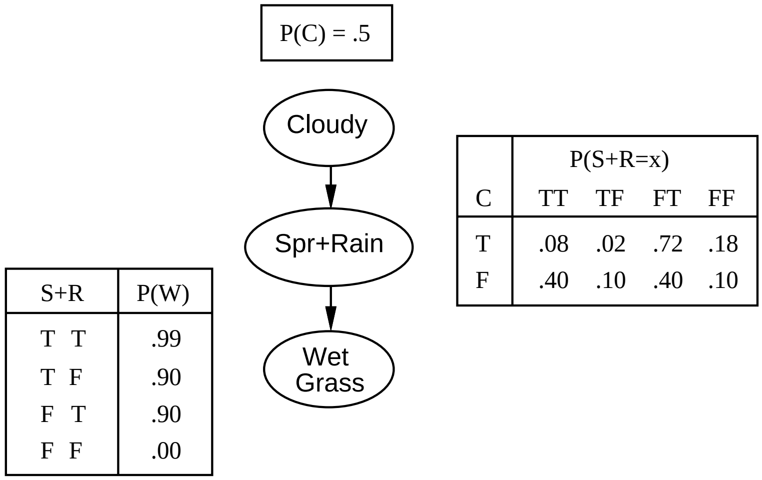

図 6: 多重結合ネットワークのクラスタ化による等価なネットワーク

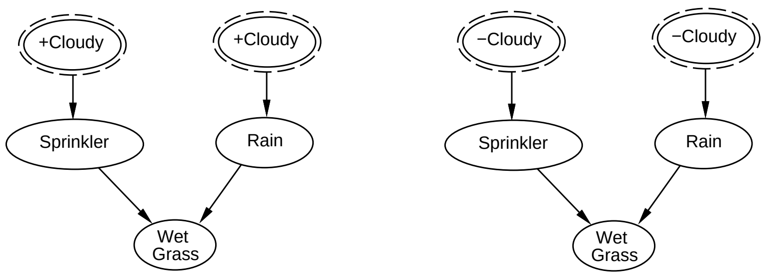

図 7: カットセットによって具体化されたネットワーク

A　Bayesian Networ における再帰的な確率伝播計算法

後ろ向きの推論アルゴリズムを図 9 に示す.

Bayesian Network の特徴の一つは，従来のベイズ則を元に算出する方法にくらべ，非常にシンプルな確率算出の方法にある．複数の事象がノードとしてネットワーク構造に配置されているため，それぞれの事象に対する観測結果から，推論目的の事象の条件付き確率を求めようとすると，計算式が煩雑になり，かつ簡単なアルゴリズムで計算する事ができない.

ここでアルゴリズムを説明するにあたって必要となる用語について説明を加える．確信度:BEL は証拠となる事象が観測された時に，推論しようとしている事象がどの程度の確率で起こるかを示すものである．証拠となる観測事象を e として,

$$BEL(x) = P(x|e) \tag{3}$$

となる．この確信度は事象 x が持つ確率変数の次元と同じ次元のベクトルとなる．ノード x とノード y の確率的な関係を表すには，条件付き確率表 (CPT:Conditional Probability Table) を使用する．(以下 CPT と呼ぶ) この CPT はマトリクスの形で表現される．例えば，ノード X における確率変数を $(x_1, x_2, \ldots, x_m)$ とし，ノード Y における確率変数を $(y_1, y_2, \ldots, y_n)$ とすると，CPT を表すマトリクス $M_{y|x}$ は

$$\begin{aligned} M_{y|x} &\stackrel{\text{def}}{=} P(Y=y|X=x) \\ &= \begin{pmatrix} P(y_1|x_1) & P(y_2|x_1) & \ldots & P(y_n|x_1) \\ P(y_1|x_2) & P(y_2|x_2) & \ldots & P(y_n|x_2) \\ \vdots & \vdots & \ddots & \vdots \\ P(y_1|x_m) & P(y_2|x_m) & \ldots & P(y_n|x_m) \end{pmatrix} \end{aligned} \tag{4}$$

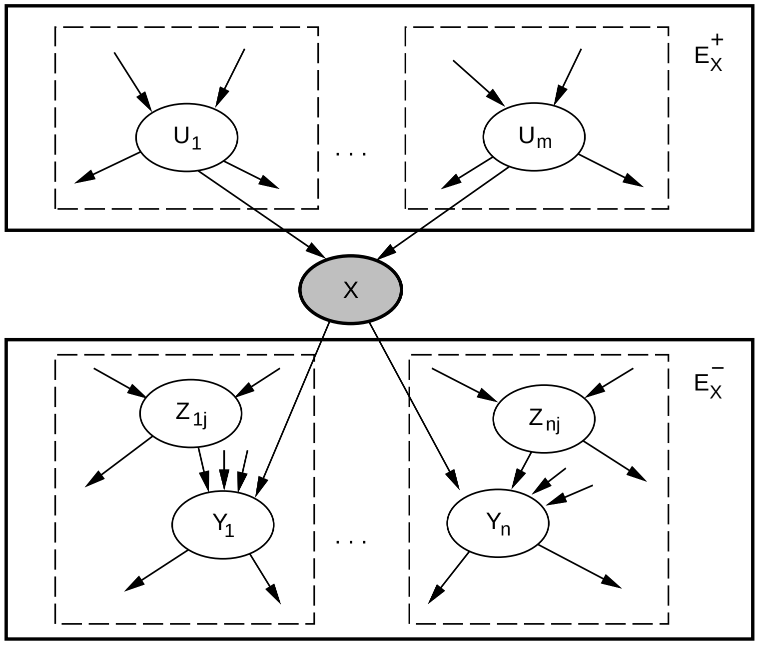

図 8:

となる．また，二つのベクトル $f(x)$ $g(x)$ の積として，項ごとに積演算を行なうものとする．

$$(1,2,3)(3,2,1) = (1 \times 3,\ 2 \times 2,\ 3 \times 1) = (3,4,3) \tag{5}$$

内積の場合は「$\cdot$」を用いる

$$(1,2,3) \cdot (3,2,1) = 1 \times 3 + 2 \times 2 + 3 \times 1 = 10 \tag{6}$$

また，次の式

$$f(x) \cdot M_{y|x} \stackrel{\text{def}}{=} \sum_x f(x) M_{y|x} \tag{7}$$

を良く使用するので留意されたい．ベクトルの各項の和が 1 となるように，正規化係数として α を用いる．

$$\alpha(1,1,3) = (0.2,\ 0,2,\ 0.6) \tag{8}$$

以下から，確率伝播のアルゴリズムについて概説するが，ネットワークの構造上大きく 3 つ分けて説明を行なう．それぞれ，Chain 型，Tree 型，PolyTree 型である．

A.1 Chain 型ネットワークにおける確率伝播

次式

$$\begin{aligned} \lambda(x) &= P(\boldsymbol{e}^-|x) \\ \pi(x) &= P(x|\boldsymbol{e}^+) \end{aligned} \tag{9}$$

において，$\lambda(x)$ と $\pi(x)$ を定義する．ここで $\boldsymbol{e}^-, \boldsymbol{e}^+$ は証拠変数であり，

$$\begin{aligned} BEL(x) &\stackrel{\text{def}}{=} P(x|\boldsymbol{e}^+, \boldsymbol{e}^-) \\ &= \alpha P(\boldsymbol{e}^-|x, \boldsymbol{e}^+) P(x|\boldsymbol{e}^+) \\ &= \alpha P(\boldsymbol{e}^-|x) P(x|\boldsymbol{e}^+) \\ &= \alpha \lambda(x) \pi(x) \end{aligned} \tag{10}$$

function BELIEF-NET-ASK(X) **returns** a probability distribution over the values of X
  inputs: X, a random variable

  SUPPORT-EXCEPT(X, *null*)

---

function SUPPORT-EXCEPT(X, V) **returns** $\mathbf{P}(X|E_{X\setminus V})$

  if EVIDENCE?(X) **then return** observed point distribution for X
  else
    calculate $\mathbf{P}(E^-_{X\setminus V}|X)$ = EVIDENCE-EXCEPT(X, V)
    $\boldsymbol{U} \leftarrow$ PARENTS[X]
    if $\mathbf{U}$ is empty
      then return $\alpha\ \mathbf{P}(E^-_{X\setminus V}|X)\ \mathbf{P}(X)$
    else
      for each U_i **in** U
        calculate and store $\mathbf{P}(U_i|E_{U_i\setminus X})$ = SUPPORT-EXCEPT(U_i, X)
      return $\alpha\ \mathbf{P}(E^-_{X\setminus V}|X) \sum_{\mathbf{u}} \mathbf{P}(X|\mathbf{u}) \prod_i \mathbf{P}(U_i|E_{u_i\setminus X})$

---

function EVIDENCE-EXCEPT(X, V) **returns** $\mathbf{P}(E^-_{X\setminus V}|X)$

  $\mathbf{Y} \leftarrow$ CHILDREN[X] $-\ V$
  if $\mathbf{Y}$ is empty
    then return a uniform distribution
  else
    for each Y_i **in Y do**
      calculate $\mathbf{P}(E^-_{Y_i}|y_i)$ = EVIDENCE-EXCEPT(Y_i, null)
      $\mathbf{Z}_i \leftarrow$ PARENTS[Y_i] $-\ X$
      for each Z_{ij} **in** $\mathbf{Z}_i$
        calculate $\mathbf{P}(Z_{ij}|E_{Z_{ij}\setminus Y_i})$ = SUPPORT-EXCEPT(Z_{ij}, Y_i)
    return $\beta \prod_i \sum_{y_i} P(E^-_{Y_i}|y_i) \sum_{\mathbf{z}_i} \mathbf{P}(y_i|X, \mathbf{z}_i) \prod_j P(z_{ij}|E_{Z_{ij}\setminus Y_i})$

図 9: 信念確率の推論アルゴリズム（後ろ向き推論）

として，確信度 $BEL(x)$ を表現する事にする．これらのベクトル $\lambda(x)$,$\pi(x)$ はノード X に関する情報であるが，この情報は，ノード間のアークを伝わって隣接するノードの λ, π に影響を与える．すなわち，証拠のノードに入力された確率の値を伝播させて，推論すべきノードの確信度を求めて行くわけである．λ について，確率の伝播は

$$\begin{aligned}\lambda(x) &= \sum_y P(\boldsymbol{e}^-|y,x)P(y|x) \\ &= \sum_y P(\boldsymbol{e}^-|y)P(y|x) \\ &= M_{y|x}\cdot\lambda(y)\end{aligned} \tag{11}$$

のように行なわれる．π については，

$$\begin{aligned}\pi(x) &= P(x|\boldsymbol{e}^+) \\ &= \sum_u P(x|u,\boldsymbol{e}^+)P(u|\boldsymbol{e}^+) \\ &= \sum_u P(x|u)\pi(u) \\ &= \pi(u)\cdot M_{x|u}\end{aligned} \tag{12}$$

つまり，π の情報[1] は親ノードから子ノードへ，λ の情報は子ノードから親ノードへ，と伝播して行く．親ノードを持たないルートノード (Root Node) では，あらかじめ与えられた事前確率 (Prior Probability) を π とする．また，子ノードを持たない葉ノード (Leaf Node) は主に観測や証拠を表すノードとして扱われ，事象が観測された場合は，その観測結果が λ に代入され，観測されない場合は，デフォルト値として，λ の全ての要素を 1 とするベクトルが代入される．

[1] π-message とも言う

A.2 Tree 型ネットワークにおける確率伝播

次に，図 10 のように親ノードに複数の子ノードが連結する，Tree 型のネットワーク構造の場合の確率伝播について説明する．ただし，あるノードに対して，親ノードは唯一しか存在しないものとする．この場合はほぼ Chain 型の場合と同様の伝播である．$\boldsymbol{e}_X^-$ をノード X より下側に存在するノードにおける証拠とし，$\boldsymbol{e}_X^+$ をノード X より上側に存在するノードにおける証拠とする．このような定義のもとでも確信度 $BEL(x)$ の定義は同様に，

$$\begin{aligned} BEL(x) &= P(x|\boldsymbol{e}_X^+, \boldsymbol{e}_X^-) \\ &= \alpha P(\boldsymbol{e}_X^+|\boldsymbol{e}_X^-, x)P(x|\boldsymbol{e}_X^+) \\ &= \alpha P(\boldsymbol{e}_X^-|x)P(x|\boldsymbol{e}_X^+) \end{aligned} \tag{13}$$

ここで，

$$\lambda(x) = P(\boldsymbol{e}_X^-|x) \tag{14}$$

および，

$$\pi(x) = P(x|\boldsymbol{e}_X^+) \tag{15}$$

とすることにより，

$$BEL(x) = \alpha\lambda(x)\pi(x) \tag{16}$$

となる．ここで，確率の伝播のルールをまとめる．まず，ボトムアップの方向として，$\lambda_X(u)$ を $\lambda(x)$ から計算する伝播則と，$\lambda(x)$ を $\lambda_{Y_i}(x)$ から計算する伝播則と二つある．それぞれ．

$$\begin{aligned} \lambda_X(u) &= \sum_x P(\boldsymbol{e}_X^-|u, x)P(x|u) \\ &= \sum_x P(\boldsymbol{e}_X^-|x)P(x|u) \\ &= \sum_x \lambda(x)P(x|u) \\ &= M_{x|u} \cdot \lambda(x) \end{aligned} \tag{17}$$

および

$$\lambda(x) = \prod_j \lambda_{Y_j}(x) \tag{18}$$

である．

一方，トップダウン方向の伝播として，$\pi_{Y_j}(x)$ および $\pi(x)$ を求める法則は次の通りである．

$$\begin{aligned} \pi(x) &= \sum_u P(x|u)\pi_X(u) \\ &= \pi_X(u) \cdot M_{x|u} \end{aligned} \tag{19}$$

$$\pi_{Y_j}(x) = \alpha\pi(x) \prod_{k \neq j} \lambda_{Y_k}(x) \tag{20}$$

A.3 Poly Tree 型ネットワークにおける確率伝播

最後に，Poly Tree 型のネットワークにおける伝播について説明する．Poly Tree とは Tree 型とは異なり，複数の親を持つ事ができる．今まで使用して来た証拠を表す $\boldsymbol{e}_X^-$ と $\boldsymbol{e}_X^+$ について再定義を行なう．

$$\begin{aligned} \boldsymbol{e}_X^- &= \left\{\boldsymbol{e}_{XY_1}^-, \ldots, \boldsymbol{e}_{XY_m}^-\right\} \\ \boldsymbol{e}_X^+ &= \left\{\boldsymbol{e}_{XY_1}^+, \ldots, \boldsymbol{e}_{XY_m}^+\right\} \end{aligned} \tag{21}$$

$\boldsymbol{e}_{XY_j}^-$ は，ノード Y_j について，ノード X より上側に存在する証拠を意味する．同様に $\boldsymbol{e}_{U_iX}^+$ は，ノード U_i について，ノード X より下側に存在する証拠を意味する．

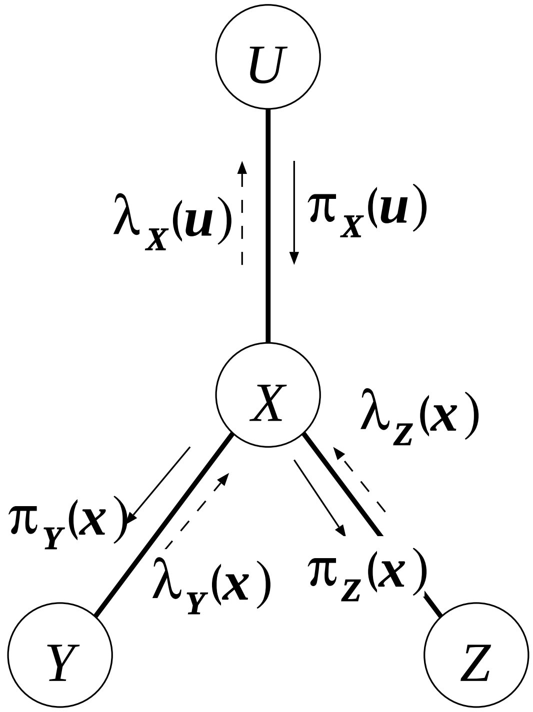

図 10: Network of Tree Type Structure

この場合の伝播則は，$\lambda(x)$ については計算方法が変わらない．

$$\begin{aligned}\lambda(x) &= P(\boldsymbol{e}_X^-|x) \\ &= P(\boldsymbol{e}_{XY_1}^-, \ldots, \boldsymbol{e}_{XY_m}^-|x) \\ &= P(\boldsymbol{e}_{XY_1}^-|x) \cdot P(\boldsymbol{e}_{XY_2}^-|x) \ldots P(\boldsymbol{e}_{XY_m}^-|x) \\ &= \prod_j^m \lambda_{Y_j}(x) \end{aligned} \tag{22}$$

ここで，$\lambda_{Y_j}(x)$ はもちろん，$P(\boldsymbol{e}_{XY_j}^-|x)$ を意味する．また，$\pi(x)$ については，

$$\begin{aligned}\pi(x) &= P(x|\boldsymbol{e}_X^+) \\ &= P(x|\boldsymbol{e}_{U_1X}^+, \ldots, \boldsymbol{e}_{U_nX}^+) \\ &= \sum_{u_1,\ldots,u_n} P(x|u_1,\ldots,u_n)P(u_1,\ldots,u_n|\boldsymbol{e}_{U_1X}^+, \ldots, \boldsymbol{e}_{U_nX}^+) \\ &= \sum_{u_1,\ldots,u_n} P(x|u_1,\ldots,u_n)P(u_1|\boldsymbol{e}_{U_1X}^+)P(u_2|\boldsymbol{e}_{U_2X}^+)\ldots P(u_n|\boldsymbol{e}_{U_nX}^+)\end{aligned}$$

ここで，

$$\pi_X(u_i) = P(u_i|\boldsymbol{e}_{U_iX}^+) \tag{23}$$

として，

$$\begin{aligned}\pi(x) &= \sum_{u_1,\ldots,u_n} P(x|u_1,\ldots,u_n)\pi_X(u_1)\pi_X(u_2)\ldots\pi_X(u_n) \\ &= \sum_{\boldsymbol{u}} P(x|\boldsymbol{u})\prod_{i=1}^n \pi_X(u_i)\end{aligned} \tag{24}$$

ここで注意しなければならないのが，$u_1,\ldots,u_n$ の扱いである．ここで簡単のために，U_1, U_2 の二つの親ノードが存在したとして，その確率変数がそれぞれ $U_1 = (True, False)$，$U_2 = (Good, Bad)$ だとする．この時の u_1, u_2

はこの二つのノードをまとめて一つのノードであるかのように扱った時の，確率変数を意味する．すなわち，

$$(u_1, u_2) = \Big\{(True, Good), (True, Bad), (Flase, Good), (False, Bad)\Big\}$$

となる．すなわち，式 23 の $\sum$ は 4 項の和となる．さらに，$\pi(x)$ がベクトルである事を考慮すると，

$$\begin{aligned}
\pi(x) &= P(x|\boldsymbol{e}_X^+) \\
&= \sum_{u_1,u_2} P(x|u_1,u_2)P(u_1|\boldsymbol{e}_{U_1X}^+)P(u_2|\boldsymbol{e}_{U_2X}^+) \\
&= P(x|u_1u_2 = TG)P(u_1 = T|\boldsymbol{e}_{U_1X}^+)P(u_2 = G|\boldsymbol{e}_{U_2X}^+) \\
&\quad + P(x|u_1u_2 = TB)P(u_1 = T|\boldsymbol{e}_{U_1X}^+)P(u_2 = B|\boldsymbol{e}_{U_2X}^+) \\
&\quad + P(x|u_1u_2 = FG)P(u_1 = F|\boldsymbol{e}_{U_1X}^+)P(u_2 = G|\boldsymbol{e}_{U_2X}^+) \\
&\quad + P(x|u_1u_2 = FB)P(u_1 = F|\boldsymbol{e}_{U_1X}^+)P(u_2 = B|\boldsymbol{e}_{U_2X}^+)
\end{aligned}$$

これを $\pi_X(u_i)$ を使って表現すると，

$$\begin{aligned}
\pi(x) &= \sum_{u_1,u_2} P(x|u_1,u_2)\pi_X(u_1)\pi_X(u_2) \\
&= P(x|u_1u_2 = TG)\pi_X(u_1 = T)\pi_X(u_2 = G) \\
&\quad + P(x|u_1u_2 = TB)\pi_X(u_1 = T)\pi_X(u_2 = B) \\
&\quad + P(x|u_1u_2 = FG)\pi_X(u_1 = F)\pi_X(u_2 = G) \\
&\quad + P(x|u_1u_2 = FB)\pi_X(u_1 = F)\pi_X(u_2 = B)
\end{aligned}$$

すなわち，$\pi_X(u_1)\pi_X(u_2)$ の結果（これを $\pi_X(\boldsymbol{u})$ と呼ぶ）は，単なるベクトルの積ではなく，

$$\pi_X(u_1) = \begin{pmatrix} a \\ b \\ c \end{pmatrix} \pi_X(u_2) \ \begin{pmatrix} d \\ e \end{pmatrix}$$

とした時,

$$\pi_X(\boldsymbol{u}) = \begin{pmatrix} ad \\ ae \\ bd \\ be \\ cd \\ ce \end{pmatrix}$$

と計算する事である．この $\pi_X(\boldsymbol{u})$ と $P(x|\boldsymbol{u})$ を用いて，

$$\begin{aligned}
\pi(x) &= \sum_{\boldsymbol{u}} P(x|\boldsymbol{u})\pi_X(\boldsymbol{u}) \\
&= \pi_X(\boldsymbol{u})M_{x|\boldsymbol{u}}
\end{aligned} \tag{25}$$

とも書くことができる．

A.3.1 λ の更新

次に

$$\lambda_X(u_i) = P(\boldsymbol{e}_{U_iX}^-|u_i) \tag{26}$$

の更新について説明する．説明の中で新しくノード V を定義する．このノードは，X の親ノードのうち，現在対象としている親ノード u_i を除いた全ての親ノードを表しており，

$$V = \boldsymbol{U} - U_i = \{U_1, \ldots, U_{i-1}, U_{i+1}, \ldots, U_n\} \tag{27}$$

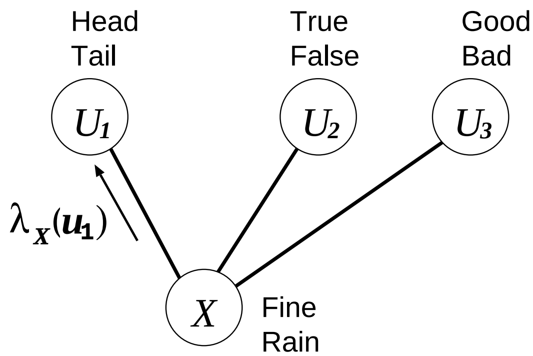

図 11: Example of poly tree structure network

で表されるテンポラリのノードである．ここで $\boldsymbol{e}^-_{U_iX}$ を

$$\boldsymbol{e}^-_{U_iX} = \{\boldsymbol{e}^+_{VX}, \boldsymbol{e}^-_X\} \tag{28}$$

のように分解する．ただし，

$$\boldsymbol{e}^+_{VX} = \bigcup_{k\neq i} \boldsymbol{e}^+_{U_kX} \tag{29}$$

である．順に式を追って行くと，

$$\begin{aligned}
\lambda_X(u_i) &= P(\boldsymbol{e}^+_{VX}, \boldsymbol{e}^-_X|u_i) \\
&= \sum_x \sum_v P(\boldsymbol{e}^+_{VX}, \boldsymbol{e}^-_X|u_i, v, x)P(v, x|u_i) \\
&= \sum_x \sum_v P(\boldsymbol{e}^-_X|x)P(\boldsymbol{e}^+_{VX}|v)P(v, x|u_i) \\
&= \beta \sum_x \sum_v P(\boldsymbol{e}^-_x|x)\frac{P(v|\boldsymbol{e}^+_{VX})}{P(v)}P(x|v, u_i)P(v|u_i) \\
&= \beta \sum_x \sum_v P(\boldsymbol{e}^-_x|x)P(v|\boldsymbol{e}^+_{VX})P(x|v, u_i)
\end{aligned}$$

ここで，式 27 と式 29 より，

$$\begin{aligned}
P(x|v, u_i) &= P(x|\boldsymbol{u}) \\
P(v|\boldsymbol{e}^+_{VX}) &= \prod_{k\neq i} P(u_k|\boldsymbol{e}^+_{U_kX}) = \prod_{k\neq i} \pi_X(u_k)
\end{aligned}$$

であるので，これを用いて，

$$\lambda_X(u_i) = \beta \sum_x \lambda(x) \sum_{u_k:k\neq i} P(x|\boldsymbol{u}) \prod_{k\neq i} \pi_X(u_k) \tag{30}$$

となる．このままでは少々理解しづらいので，実際に具体例を使って解説を加える．

図 11 に示すような，親ノードが u_1 から u_3 までである状況を考える．求めるものは $\lambda_X(u_1)$ である．

$$\begin{aligned}
\lambda_X(u_1) &= \beta \sum_x \sum_{u_k:k\neq i} \lambda(x)P(x|\boldsymbol{u}) \prod_{k\neq i} \pi_X(u_k) \\
&= \beta \sum_x \sum_{u_2,u_3} \lambda(x)P(x|\boldsymbol{u})\pi_X(u_2)\pi_X(u_3) \\
&= P(x = Fi|u_1, u_2 = Tr, u_3 = Go)\lambda(x = Fi)\pi_X(u_2 = Tr)\pi_X(u_3 = Go) \\
&\quad + P(x = Fi|u_1, u_2 = Tr, u_3 = Ba)\lambda(x = Fi)\pi_X(u_2 = Tr)\pi_X(u_3 = Ba) \\
&\quad \vdots \\
&\quad + P(x = Ra|u_1, u_2 = Fa, u_3 = Ba)\lambda(x = Ra)\pi_X(u_2 = Fa)\pi_X(u_3 = Ba)
\end{aligned}$$

となる．ここで，この $P(x=Fi|u_1,u_2=Tr,u_3=Go)$ は図 12 に示す通り，網がけの部分の要素を意味している．すなわち，この要素は u_i の次元数と同じベクトルを表現しており，そのベクトルを x,v の組み合わせの数だけ足し合わせる計算を行なう事になる．

u_1,u_2,u_3		X	Fine	Rain
Head	True	Good		
Head	True	Bad		
Head	False	Good		
Head	False	Bad		
Tail	True	Good		
Tail	True	Bad		
Tail	False	Good		
Tail	False	Bad		

図 12: Lambda part of CPT