本レジュメの最後に付録として Appendix のページを添付している。適宜参照されたい。

1　グラフィカルモデルにおける推論

前回は木構造グラフ内のある 1 ノードにおける周辺分布を計算した。そのために因子ノードから変数ノードへのメッセージの流れ[*1]、逆に変数ノードから因子ノードへのメッセージの流れを定義した[*2]。

今回は議論を拡張し全ノードに対して周辺分布の計算を行う。その際に効率よく、つまり計算負荷が上がらないように計算を実行する手法を学ぶ[*3]。この手法は先のサブセクションで用いた積和アルゴリズムをより一般化したものと言ってよい。

さらに一般化された積和アルゴリズムの計算について 4 ノード因子グラフを例に取りメッセージの定式化を行う。そして定めたメッセージをもとに周辺分布が求められることを確かめる。

最後に max-sum アルゴリズムを導入し、確率[*4]が最大となる変数の組を見つけ、そのときの確率値を求める手法について学ぶ[*5]。今回は変数の組の探索について大域的な解を特定するにあたって課題が生じることを呈示するところまで議論する。

1.1　積和アルゴリズム

本サブセクションの流れは次の通りである。まず一般的な積和アルゴリズムを設計する。その際に双方向のメッセージ（葉ノード ⇔ 根ノード）をやり取りすることで計算負荷を抑えることができるということを確かめる。さらに具体的なグラフ構造において周辺分布の計算を実行する。最後に補足として観測されたノードが存在する場合に周辺分布を計算する手法について取り扱う。

まずは積和アルゴリズムの設計を行う。ただし設計方法の詳細な説明は教科書に譲り読み合わせとさせてもらう[*6]。ポイントはまず全ての葉ノードから全ての根ノードへのメッセージを全て伝播させ、逆の流れについても同様に伝播させる。そしてそれらの中間的なメッセージを保存しておくことで、どのノードの周辺分布も計算できるということである。したがって計算負荷は 1 ノードにおいて議論した場合と比べて 2 倍で済ませることができる。この考え方はやはり連鎖で議論を行ったときと同様のアナロジーで理解することができる。

一方、各ノードに対して別個に積和アルゴリズムを適用すると計算負荷はグラフの大きさ、つまりノード総数に関して二次関数的に増加する、らしい。が、わからなかった。

次は変数集合 $\boldsymbol{x}_s$ 全体上の周辺分布 $p(\boldsymbol{x}_s)$ を計算する。これが（8.72）で示されることを確かめよう。ごめん無理だった。

ここで積和アルゴリズムについて別の解釈を与える[*7]。そのために図 8.50 において x_3 に流入するメッセージが次のステップによって計算できることを確かめる。1. 因子ノード（根）4 つから送られてくるメッセージについての積を取る、2. 因子 f_s を掛け算しさらに x_1, x_2 で周辺化する。これが（8.66）を使って計算した結

[*1] 式（8.66）を見よ。
[*2] 式（8.69）を見よ。
[*3] ちなみに周辺分布の計算対象を 1 ノード→全ノードに拡張するという流れは 8.4.1 で連鎖のケースにおける議論と全く同じである。
[*4] なお、ここでいう確率とは同時分布 $p(\boldsymbol{x})$ のことを表している。
[*5] 実際の計算アルゴリズムの順番としては、同時分布の値を最大化→最大値を与える変数集合の組の決定、ということになるのだが詳細は後述。
[*6] 教科書 122 ページ中段を見よ。
[*7] ここの話のありがたみがよくわからなかった。

果と一致することを示せば良い（> Go to App.1）。

最後に規格化に関する補足を加える。無向グラフから因子グラフを導出する場合は当然規格化が必要になる。しかしながら我々はひとまず規格化されていない周辺分布（に相当するもの）を計算し、そのあとで 1 変数に関する周辺化を行うことで簡単に規格化係数を得ることができる。ある 1 変数に関して和をとればよく全変数に関する周辺化を行う必要はないことに注意せよ。

ここまでで積和アルゴリズムの一般論に関する説明を終えよう。ここからは 4 ノード因子グラフ（図 8.51）を例に取りメッセージの定式化及び周辺分布の計算を実際に体験してみよう。

周辺分布を求めるまでの流れは次の通りである。まず周辺分布の形を無向グラフを想定して書き下す。次に葉ノードから根ノードへメッセージを伝播させ各ノード、因子におけるメッセージを求める。そして逆方向にもメッセージを伝播させてやる。最後に（8.63）を用いて周辺分布の表式が得られることを確かめる。

まず図 8.51 のグラフ形より規格化されていない同時分布は明らかに（8.73）の形で与えられる。いま x_3 を根ノードとすると x_1, x_4 は葉ノードと見てよい。

次にこれら 2 つの葉ノードから根ノード x_3 に向かうメッセージパッシングを考える。このとき各ノード、因子におけるメッセージは次のような表式で得られる。

$$\mu_{x_1\to f_a}(x_1) = 1 \ (\because \ (8.70)) \tag{1}$$

$$\begin{aligned}\mu_{f_a\to x_2}(x_2) &= \sum_{x_1} f_a(x_2,x_1)\Pi_{m\in ne(f_a)\setminus x_2}\mu_{x_m\to f_a}(x_m) \ (\because \ (8.66)) \\ &= \sum_{x_1} f_a(x_2,x_1)\mu_{x_1\to f_a}(x_1) \\ &= \sum_{x_1} f_a(x_2,x_1)\cdot 1 \ (\because \ (8.70)) \\ &= \sum_{x_1} f_a(x_2,x_1)\end{aligned} \tag{2}$$

$$\mu_{x_4\to f_a}(x_4) = 1 \ (\because \ (8.70)) \tag{3}$$

$$\begin{aligned}\mu_{f_c\to x_2}(x_2) &= \sum_{x_4} f_c(x_2,x_4)\Pi_{m\in ne(f_c)\setminus x_2}\mu_{x_m\to f_c}(x_m) \ (\because \ (8.66)) \\ &= \cdots ((8.75) \text{ と同様のため省略。}) \\ &= \sum_{x_4} f_c(x_2,x_4)\end{aligned} \tag{4}$$

$$\mu_{x_2\to f_b}(x_2) = \mu_{f_a\to x_2}(x_2)\mu_{f_c\to x_2}(x_2) \ (\because \ (8.69)) \tag{5}$$

$$\begin{aligned}\mu_{f_b\to x_3}(x_3) &= \sum_{x_2} f_b(x_3,x_2)\Pi_{m\in ne(f_b)\setminus x_3}\mu_{x_m\to f_b}(x_m) \ (\because \ (8.66)) \\ &= \sum_{x_2} f_b(x_3,x_2)\mu_{x_2\to f_b}(x_2)\end{aligned} \tag{6}$$

今度は葉ノード、根ノードを入れ替えた場合について各ノードにおけるメッセージを同様に書き下してみる。こちらは読み合わせで確認しよう。

この時点で全てのリンクに対して両方向にメッセージが送られた。これで全ノードについて周辺分布を計算できるようになった[*8]。ここから周辺分布 $p(x_2)$ を例にとり計算してみよう。そのためにまず規格化されていない $\tilde{p}(x_2)$ を求めれば良い。式（8.63）を利用することでその目的は果たされる。ノード x_2 に流入する全て

[*8] 繰り返すが、両方向からメッセージを流すことで全ての周辺分布が計算できるというフレームワークは連鎖で議論したものと全く同様であることを理解せよ。

のメッセージを掛け算すると $\tilde{p}(x_2)$ は

$$\begin{aligned}
\tilde{p}(x_2) &= \Pi_{s\in ne(x_2)}\mu_{f_s\to x_2}(x_2) \ (\because \ (8.63)) \\
&= \mu_{f_a\to x_2}(x_2)\mu_{f_b\to x_2}(x_2)\mu_{f_c\to x_2}(x_2) \\
&= \sum_{x_1} f_a(x_1,x_2)\sum_{x_3} f_b(x_2,x_3)\sum_{x_4} f_c(x_2,x_4) \ (\because \ (8.75),(8.81),(8.77)) \\
&= \sum_{x_1}\sum_{x_3}\sum_{x_4} f_a(x_1,x_2)f_b(x_2,x_3)f_c(x_2,x_4) \\
&= \sum_{x_1}\sum_{x_3}\sum_{x_4} \tilde{p}(\boldsymbol{x}) \ (\because \ (8.73)) \qquad (7)
\end{aligned}$$

と書ける。これで規格化されていない周辺分布を計算することができた[*9]。

最後に一部のノードが観測されている場合について事後分布を計算するフレームワークを確認しよう。ここは読み合わせ。最後の「$\boldsymbol{v}$ に含まれる〜置き換わる」の意味がわからなかった。

1.2 max-sum アルゴリズム

積和アルゴリズムを用いることで周辺分布を効率よく、つまり計算負荷を高めることなく計算することができた。実はこの周辺分布を計算するということ以外に次の 2 つのタスクが必要になることがある。

・確率[*10]が最大となる変数の組を見つけること

・上記の場合について、そのときの確率値を求めること

これらのタスクを max-sum アルゴリズムと呼ばれる手法で解決する。

本サブセクションの内容は大きく 2 つのパートに分けることができる。ひとつが確率の最大値計算。もうひとつが最大値を与える変数の組の探索である。

最初に確率の最大値計算について流れを記す。まず我々が求めるべき確率の最大値が同時分布の最大値であることを確認する。そしてその最大値が最大値演算の形式で表されることを見たのちノード連鎖の例に適用することを考える。さらに拡張した議論として木構造に対しても最大値演算を適用することを考える。最後に実用のため最大値演算の表式に対して対数を取ることで小数点の丸め誤差が発生しないようにするためのフレームワークを確認する。

変数の組の探索についての流れは次回確認しよう。ひとまず今回は最大値を与える x^{max} が従来のメッセージパッシングによる手法では正しく求められないことを呈示する。

確率の最大値計算についての議論から入ろう。結論から言うと我々は同時分布の最大化、及びその最大値を与える変数集合の組を考えれば良い。

ところで最大値の考え方について次のような主張が考えられる：ノードごとに個別に周辺分布を全て計算しそれらの中で最も大きい値が最大値である。そしてその最大値を与える x を見つけてくれば良い。

しかし実用上は最大確率を同時に達成する変数集合の組を求めたいことの方が多い[*11]。そのため我々は確率の最大値と言ったときには同時分布の最大値のことを言っていると考えれば良い。したがって個別に周辺分布を計算するという考え方は採用しない。

ここで同時分布を最大にする変数の組と個々の周辺分布を最大にする変数の組とは一致しないことを具体例でもって確かめる。ここは読み合わせ。

[*9] 規格化の議論は省略。教科書にも特に記述ないため。

[*10] ここでいう「確率」は同時分布を指している。詳細は後ほど説明する。

[*11] 正直なところ「実用上」の意味はまだよくわかっていない。今後しっかりと理解できることを願っている。

ここからは同時分布の最大化演算を実行していく。まずやることは次の数式の通り。

$$\max_{\boldsymbol{x}} p(\boldsymbol{x}) = \max_{x_1} \cdots \max_{x_M} p(\boldsymbol{x}) \tag{8}$$

次にこの同時分布をポテンシャル関数の積で表す[*12]。さらに $a \geq 0$ のときに成立する最大値演算に関する次の法則

$$\max(ab, ac) = a \max(b, c) \tag{9}$$

を利用し最大値演算と積演算とを入れ替えられるようにする。この 2 つのポイントを連鎖グラフに対して適用すると

$$\begin{aligned} \max_{\boldsymbol{x}} p(\boldsymbol{x}) &= \frac{1}{Z} \max_{x_1} \cdots \max_{x_N} [\psi_{1,2}(x_1, x_2) \cdots \psi_{N-1,N}(x_{N-1}, x_N)] \\ &= \frac{1}{Z} \max_{x_1} [\max_{x_2} [\psi_{1,2}(x_1, x_2) [\cdots \max_{x_N} \psi_{N-1,N}(x_{N-1}, x_N)]]] \ (\because \text{式 } 9) \end{aligned} \tag{10}$$

のように和と積の順序を入れ替えた表式が得られる。ここで式 8 の M は N に書き換えた。また Z はポテンシャル関数についての分配関数である。こうすると式 8 を適用した場合よりも効率よく計算できるらしい。しかしそれぞれのオーダーがどれほどのものかよくわかっていない。ちなみに式 10 はノード x_N から後ろ向きに連鎖を伝わって[*13]ノード x_1 に到達するメッセージと解釈できるものが存在することがわかる。

木構造における議論については読み合わせ。ポイントは”max”-product アルゴリズムがその名の通り積「和」アルゴリズムにおける和演算 Σ を最大値演算 max に置き換えたものに過ぎないということである[*14]。この置き換えによりメッセージの表式（8.66）は次のように書き直せる。

$$\mu_{f_x \to x}(x) = \max_{x_1} \cdots \max_{x_M} f(x, x_1, \cdots, x_M) \Pi_{m \in ne(f) \backslash x} \mu_{x_m \to f}(x_m) \tag{11}$$

原理的には以上の手続きによって確率の最大値を求めることができる。しかしながらここで計算上の問題が発生することが懸念される。式 10 を見ると小さい確率の値がたくさん掛け算された形になっていることがわかる。よって丸め誤差[*15]の問題が発生しうるためこれを避けたい。その方法は単純であり最大値演算の表式に対して対数をとってやるだけである[*16]。これにより（8.91）を得る。

ここからは max-sum アルゴリズムを導出し確率の最大値を計算していく。そのためには先で導入した最大値演算と対数演算の入れ替えに加えて次の分配則

$$\max(a + b, a + c) = a + \max(b, c) \tag{12}$$

を利用すればいい。式 11 において両辺に ln を適用し、$\ln \mu \to \mu$ と書き直す。

$$\begin{aligned} \ln \mu_{f \to x}(x) &= \ln [\max_{x_1} \cdots \max_{x_M} f(x, x_1, \cdots, x_M) \Pi_{m \in ne(f) \backslash x} \mu_{x_m \to f}(x_m)] \\ &= \max_{x_1} \cdots \max_{x_M} \ln [f(x, x_1, \cdots, x_M) \Pi_{m \in ne(f) \backslash x} \mu_{x_m \to f}(x_m)] \\ &= \max_{x_1} \cdots \max_{x_M} [\ln f(x, x_1, \cdots, x_M) + \sum_{m \in ne(f) \backslash x} \ln \mu_{x_m \to f}(x_m)] \\ \mu_{f \to x}(x) &= \max_{x_1} \cdots \max_{x_M} [\ln f(x, x_1, \cdots, x_M) + \sum_{m \in ne(f) \backslash x} \mu_{x_m \to f}(x_m)] \ (\because \ln \mu \to \mu \text{と置き換え}) \end{aligned} \tag{13}$$

[*12] 和積の入れ替えを行うため無向グラフの方が議論しやすい。

[*13] 木で言うところの葉ノードから根ノードへの伝播である。

[*14] このことに気づいた瞬間全身の力が抜けた。先週の段階で”max-product”という名前を見てビビった自分をビンタしてやりたい。

[*15] 教科書でアンダーフローと言っているやつ。「丸め誤差」の方が分かりやすいと思ってそのように書いた。

[*16] 対数関数の単調増加性と積を和に変えることができる性質を利用する。丸め誤差を避けるには掛け算を足し算に直すことが定石。

葉ノードにおけるメッセージの設定については教科書を参照されたい。また根ノードにおける確率の最大値は（8.63）において総積を最大値演算と総和に書き換えてやることで（8.97）のように表すことができる。ごめん。かなり天下りな説明になっている。メッセージの表式からどういうロジックで（8.97）導けるのが正直わかっていない。。

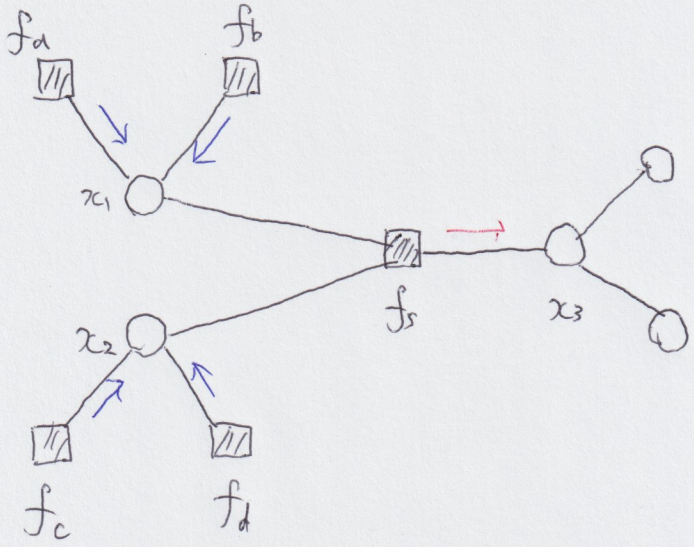
ここからは確率の最大値を与える変数の組を求めよう。まずあるノード x において（8.98）を適用してやれば確率が最大となる x^{max} を計算することができる。そして次に根ノードから葉ノードにメッセージを伝播させたのち残り全てのノードに対しても（8.98）を適用してやればいい。これで $\boldsymbol{x}^{max}$ の組が求められたように思える。しかしこのやり方はダメである。なぜかについては別資料を参照されたい[*17]。

次回はメッセージパッシングのやり方を変更することで上記の問題を解決する手法について学ぶ。

[*17] http://tomerun.info/presen/PRMLRevenge/8_4_5/8_4_5.pdf の 10 枚目を参照。

App. 1　積和アルゴリズムの別解釈と(8.66)との一致

図 8.50 において各因子に名前を付けておく。



以下では赤矢印のメッセージが

① $f_a \sim f_d$ の因子達を持ち

② f_s をかけ算し

③ x_1, x_2 で周辺化する

ことによって求められること、そして

これが(8.66)に一致することを確かめる。

求めるメッセージ $\mu_{f_s \to x_3}(x_3)$ は ① ~ ③ を使って次のように書ける。

$$\mu_{f_s \to x_3}(x_3) = \underbrace{\sum_{x_1}\sum_{x_2}}_{③} \underbrace{f_s(x_3, x_1, x_2)}_{②} \underbrace{\mu_{f_a \to x_1}(x_1)\,\mu_{f_b \to x_1}(x_1)\,\mu_{f_c \to x_2}(x_2)\,\mu_{f_d \to x_2}(x_2)}_{①}$$

$$= \sum_{x_1}\sum_{x_2} f_s(x_3, x_1, x_2) \prod_{i \in ne(x_1)\setminus f_s} \mu_{f_i \to x_1}(x_1) \prod_{j \in ne(x_2)\setminus f_s} \mu_{f_j \to x_2}(x_2)$$

$$= \sum_{x_1}\sum_{x_2} f_s(x_3, x_1, x_2)\, \mu_{x_1 \to f_s}(x_1)\, \mu_{x_2 \to f_s}(x_2) \quad (\because (8.69))$$

$$= \sum_{x_1}\sum_{x_2} f_s(x_3, x_1, x_2) \prod_{m \in ne(f_s)\setminus x_3} \mu_{x_m \to f_s}(x_m)$$

これは(8.66)において $x \to x_3$, $M \to 2$ としたものと一致する。

よって「別解釈」によって定義したメッセージは、因子ノードから変数ノードへのメッセージを正しく与えることがわかった。