

半環に基づく前向き後ろ向きアルゴリズムの一般化

A Generalization of Forward-Backward Algorithm Based on Semi-Rings

東 藍^{1*}
Ai Azuma¹

新保 仁¹
Masashi Shimbo¹

松本 裕治¹
Yuji Matsumoto¹

¹ 奈良先端科学技術大学院大学

¹ Nara Institute of Science and Technology

Abstract: Abstract (English) comes here.....

1 はじめに

近年、機械学習やデータマイニングの適用範囲はますます広がっており、その対象は内部に複雑な構造を有するデータにも広がりつつある。本論文における議論では、そのような構造や依存関係を有するデータのうち最も単純な部類のものである系列データを対象とする。

系列データに対する機械学習・データマイニングを行う上で、可能な全ての系列にわたって何らかの総和を計算する必要に迫られる局面は多数見受けられる。この可能な全ての系列の集合が、有向非循環グラフ上の有向パスとしてコンパクトに表現されている場合、ある種の形式の計算に対しては動的計画法に基づく効率的な計算方法が広く知られている。しかしながら、より一般的な形式の和については動的計画法による効率的な計算方法は知られていないことが多い。また既存の研究においては、個々に必要となる和の形式のうちよく知られている形式の動的計画法では計算できないものに対して、独立かつ個々に計算アルゴリズムを設計している例も散見される。

本論文では、より幅広い形式の和を有向非循環グラフ上で計算するためのより統一的な枠組みとして、代数的な抽象化に基づいた前向き後ろ向きアルゴリズムの一般化を提案する。また、この一般化の枠組みに当てはまる具体的な代数のうち、系列データに対する機械学習・データマイニングにおいて重要な役割を果たすと思われるものをいくつか紹介する。そして、この一般的なアルゴリズムの形式が系列データに対する具体的な機械学習・データマイニングの文脈において、どのように具体化されることで有用となるのか実演する。

2 研究背景

まず最初に、本論文において何を一般化の対象とするかについて述べておく。有向非循環グラフ $G = (V, E)$ に対して、入次数 0 の頂点を始点とし、出次数 0 の頂点を終点とする有向パスの集合を $\Pi(G)$ とおく。また、 V 上に実数値関数 ϕ, f が定義されているとする。このように定めた表記の下で、本論文で一般の対象とするのは、我々が系列データに対する機械学習・データマイニングにおいて極めて重要な役割を果たすと考える以下のように定式化される計算である。

$$\sum_{\pi \in \Pi(G)} \prod_{v \in \pi} \phi(v) \quad (1)$$

$$\sum_{\pi \in \Pi(G)} \left(\sum_{v \in \pi} f(v) \right) \left(\prod_{v \in \pi} \phi(v) \right) \quad (2)$$

ここで有向パスを表す記号 π を、 π 上に現れる頂点の集合を表す記号としても扱っている。この表記は特に混乱を来たさないと思われるので、以下でも特に断りなく同様に扱う。

以下、(1) および (2) の形式、およびその一般化が系列データを対象とする機械学習・データマイニングにおいて重要であると我々が考える理由と背景について述べる。

(1) は、考慮している系列の集合が $\Pi(G)$ として表現されており、各系列に対する確率ポテンシャルがその系列に現れる頂点上のポテンシャル関数の積として表現されているときに、集合 $\Pi(G)$ に対する周辺確率あるいは正規化定数を計算することに対応する。たとえば、系列データの解析に頻繁に用いられる図 ?? のような直鎖状のグラフィカルモデル G を考え、 G の周辺確率あるいは正規化定数を計算することを考えてみる。今、仮に G の各頂点に A, B, C の 3 つの状態が割り当てられるものとする。その可能な状態の割り

*連絡先：奈良先端科学技術大学院大学
情報科学研究科 自然言語処理学講座
〒 630-0192 奈良県生駒市高山町 8916-5
E-mail: ai-a@is.naist.jp

当てを全て展開したものを有向非循環グラフとして表現したものを図 ?? に示す．図 ?? で表現されるような図，あるいはデータ構造はしばしばラティスあるいはトレリスと呼ばれる．ここでトレリス中の辺の向きは単に系列の前後を区別するために便宜的に与えるものであり，グラフィカルモデルにおける有向・無向の概念とは無関係である． \mathcal{G} のような直鎖状のグラフィカルモデルにおいては，全頂点に対する状態の割り当てに対する結合確率は頂点と辺の上に定義された非負実数値のポテンシャル関数の積に比例する．一方，トレリス G 上に定義される有向パスの集合 $\Pi(G)$ は \mathcal{G} に対する可能な状態の割り当ての集合に 1 対 1 で対応するのは明らかである．図 ?? に示されるトレリス $G = (V, E)$ に対して，辺集合 E の全ての要素を新たに頂点に置き換えるように変形した結果得られるトレリスを $G' = (V', E')$ と置く．つまり， $V' \stackrel{\text{def}}{=} V \cup E$ とし， E' の要素 (x, y) は $x = v \wedge y = (v, v')$ ， $(v, v') \in E$ または $x = (v, v') \wedge y = v'$ ， $(v, v') \in E$ のときかつこのときのみ存在するとする．この変形は， \mathcal{G} をファクターグラフに変形することに対応する．¹このような定義の下で， G' に対する (1) はまさにグラフィカルモデル \mathcal{G} に対する周辺確率あるいは正規化定数を与える．このとき，(1) における ϕ は \mathcal{G} のクリーク上に定義されたポテンシャル関数となる．

より一般に，ある状態に滞留する時間がある決められた単一時間長さとは限らないようなモデル，すなわち準マルコフモデルを考えることもできる．このようなモデルもやはり系列データに対する機械学習・データマイニングで実用上多用されている．[3][1] (1) の形式はこのような準マルコフモデルも包含している形式であることに注意されたい．

さて，このようなトレリス上にエンコードされた系列集合を対象とした計算のうち，実用上重要な他の例として最尤パスの尤度計算が挙げられる．対数関数が単調増加関数であることから，最尤パスの対数尤度は以下のように定式化される．

$$\max_{\pi \in \Pi(G)} \sum_{v \in \pi} \log \phi(v) \quad (3)$$

本論文における一般化の出発点として，(1) と (3) の代数的な共通性を論じることとする．すなわち，加算および乗算を抽象化して 2 項演算 \oplus, \otimes として表記することになると，(1) と (3) はともに以下のように代数的に抽象化した形式で統一的に記述できる．

$$\bigoplus_{\pi \in \Pi(G)} \bigotimes_{v \in \pi} \phi(v) \quad (4)$$

¹この変形によって，頂点および辺の上で定義された関数を頂点の上だけで定義された関数として扱える．以下，表記を簡潔にするためにこのような頂点上にのみ関数が定義された形式のみで論じるものとする．読者におかれては，文脈の必要に応じてこのような変形が暗黙になされているものとして読み替えていただきたい．

ここで， \oplus は 2 項演算 \oplus の列 $\cdot \oplus \cdot \oplus \cdots$ を表し， \otimes は 2 項演算 \otimes の列 $\cdot \otimes \cdot \otimes \cdots$ を表す．(1) は \oplus が実数の通常の加算で \otimes が実数の通常の乗算の場合である．また，(3) は \oplus が実数に対する \max であり \otimes が実数の通常の加算の場合である．(4) が明確に定義できるためには，これらの 2 項演算が満たすべき規則の集合が必要となるが，これは半環と呼ばれる代数的枠組みとして形式化される．この半環の厳密な定義については次の節で述べる．

他に (4) の形式の範疇となる例を挙げてみよう．たとえば集合 $B \stackrel{\text{def}}{=} \mathbb{R} \times \Pi(G)$ 上の 2 項演算 \oplus, \otimes を

$$(p, \pi) \oplus (q, \pi') \stackrel{\text{def}}{=} \begin{cases} (p, \pi), & p \geq q \\ (q, \pi'), & \text{otherwise} \end{cases} \quad (5)$$

$$(p, \pi) \otimes (q, \pi') \stackrel{\text{def}}{=} (p + q, \pi \cup \pi') \quad (6)$$

と定義する．ここで $\pi \cup \pi'$ は有向パスの結合を表す．これらと $\phi(v) \stackrel{\text{def}}{=} (\log \phi'(v), \{v\})$ なる定義の下で (4) は最尤パスとその対数尤度を同時に与え，これはすなわちヴィタビアルゴリズムによって求まる結果を定式化したものに他ならない．また， \min_k を $(\mathbb{R} \cup \{\infty\})^m$ ($m \geq k$) から $(\mathbb{R} \cup \{\infty\})^k$ へ写す次のような写像とする． $\min_k(x_1, \dots, x_m) = (y_1, \dots, y_k)$ ，ここで (y_1, \dots, y_k) は (x_1, \dots, x_m) を昇順ソート済みリストとしたときの最初の k 要素である． \mathbb{T}_k を $(\mathbb{R} \cup \{\infty\})^k$ の k -タプルの集合で， $\mathbb{R} \cup \{-\infty\}$ における自然な順序で並んでいるものとする．つまり

$$\mathbb{T}_k = \{(a_1, \dots, a_k) \in (\mathbb{R} \cup \{-\infty\})^k \mid 0 \leq a_1 \leq \dots \leq a_k\} \quad (7)$$

このとき \mathbb{T}_k には次の 2 つの 2 項演算が定義できる．

$$a \oplus_k b = \min_k((a_1, \dots, a_k) \cup (b_1, \dots, b_k)) \quad a \otimes_k b = \min_k((a_i + b_j)_{i,j}) \quad (8)$$

この定義，および $\phi(v) = (x_v, \dots, x_v)$ なる定義の下で (4) は上位 k 個の最尤パスの対数尤度を与える．言い換えればビーム幅 k のビーム探索を定式化する．

$$((p_1, \pi_1), \dots, (p_n, \pi_n)) \otimes ((q_1, \pi'_1), \dots, (q_n, \pi'_n)) \stackrel{\text{def}}{=} ((p_{i_1}, \pi_{i_1}), \dots, (p_{i_n}, \pi_{i_n})) \quad (9)$$

$$((p_1, \pi) \quad (10)$$

(4) はビーム幅 n のビームサーチによっても止まる対象を定式化したものである．

半環に対して (4) の形式に代数的に抽象化し，これを計算するアルゴリズムを定式化した既存の研究としていくつか挙げることができる．たとえば，[2] の枠組みでは，任意のグラフに対して半環によって一般化した最短距離問題を定式化している．(4) の形式は，彼の枠組みにおいて非循環有向グラフに限定し queue

decipline として topological order による優先順位付キューを採用することに対応する。

しかしながら, [2] で例示されている具体例以外にも系列データの機械学習・データマイニングにおいて非常に重要な役割を果たすと考えられるものをいくつか挙げる事ができる。本論文の新規性の1つは, 系列データに対する機械学習・データマイニングにおいて重要な役割を果たすと考えられる他の半環の具体例を挙げ, その実際的な適用事例を示すことにある。

一方で, 系列データに対する機械学習・データマイニングにおいては, たとえばグラフィカルモデルのある頂点に特定の状態が割り当てられる確率を計算する必要に迫られる局面も多い。これはトレリスの表記で言い換えれば, トレリス上のある頂点を通る有向パスの確率ポテンシャルのみに関して和を取る計算となる。たとえば, 図??の $X_3 = A$ なる頂点 v を通る有向パスの確率ポテンシャルのみを総和する計算は

$$\begin{aligned} & \sum_{\pi \in \Pi(\text{src}, \text{snk})} \delta_{v \in \pi} \prod_{v' \in \pi} \phi(v') \\ &= \sum_{\pi \in \Pi(\text{src}, \text{snk})} \left(\sum_{v' \in \pi} \delta_{v'=v} \right) \left(\prod_{v' \in \pi} \phi(v') \right) \end{aligned} \quad (11)$$

で与えられる。ここで $\delta_{\mathcal{P}}$ は叙述 \mathcal{P} に対する指示関数である。

また, 状態 A が割り当てられる期待値は

$$\sum_{\pi \in \Pi(\text{src}, \text{snk})} \left(\sum_{v \in \pi} \delta_{\text{state}(v)=A} \right) \left(\prod_{v' \in \pi} \phi(v') \right) \quad (12)$$

で計算できる。state(v) = A は, v が状態 A の割り当てに対応する頂点であることを意味するものとする。 $\sum_{v \in \pi} \delta_{\text{state}(v)=A}$ が状態 A を割り当てられた頂点の有向パス π 中に現れる個数を表すことに注意されたい。これらの計算はグラフィカルモデルにおいては前向き再帰における中途の状態を記録した変数と後ろ向き再帰における中途の状態を記録した変数とを組み合わせることで計算可能である。前向き再帰によって(1)の形式が可能であり, これが代数的に抽象化できる一方, 前向きと後ろ向き再帰を組み合わせる形式に対して代数的抽象化を施す議論は我々が知る限り成されていないと思われる。

本論文の新規性として, この形式に対しても代数的に抽象化された表記に基づいた一般的なアルゴリズムを導出する。

3 定義

本節では, 前向き後ろ向きアルゴリズムを一般化するにあたって, 本論文で用いるいくつかの定義について述べておく。

3.1 半環および群上の半環

定義 1 (半群, monoid) 半群とは, 閉じた2項演算 \oplus の定義された集合 \mathbb{K} であり, 以下を満たす。

1. $\forall a, b, c \in \mathbb{K}$ に対して $(a \oplus b) \oplus c = a \oplus (b \oplus c)$ が成立 (結合法則)
2. $\bar{0} \in \mathbb{K}$ が存在し $\forall a \in \mathbb{K}$ に対して $\bar{0} \oplus a = a \oplus \bar{0} = a$ が成立 (単位元の存在)

定義 2 (可換半群, commutative monoid) 可換半群とは, $\forall a, b \in \mathbb{K}$ に対して $a \oplus b = b \oplus a$ (交換法則) が成り立つ半群 $(\mathbb{K}, \oplus, \bar{0})$ である。

定義 3 (半環, semiring) \mathbb{K} を集合とする。 \oplus, \otimes を \mathbb{K} 上の2項演算, $\bar{0}, \bar{1}$ を \mathbb{K} の要素とする。このとき, システム $\mathcal{T} = (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ が半環 (semiring) であるとは以下が成り立つことである。

1. $(\mathbb{K}, \oplus, \bar{0})$ が $\bar{0}$ を単位元とする可換半群 (commutative monoid) である,
2. $(\mathbb{K}, \otimes, \bar{1})$ が $\bar{1}$ を単位元とする半群 (monoid) である,
3. \otimes が \oplus に対して分配する。すなわち, $\forall a, b, c \in \mathbb{K}$ に対して以下が成り立つ
 - (a) $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$
 - (b) $c \otimes (a \oplus b) = (c \otimes a) \oplus (c \otimes b)$
4. $\bar{0}$ が \otimes に対するゼロイデアルとなる。つまり $\forall a \in \mathbb{K}, a \otimes \bar{0} = \bar{0} \otimes a = \bar{0}$

定義 4 (群上の半環) 半環 $\mathcal{T}' = (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ が群 $(\mathbb{K}', +, 0)$ 上の半環であるとは以下を満たすことである。

1. $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ が半環である
2. $(\mathbb{K}', +, 0)$ が半群をなす
3. 2項演算 $\cdot : \mathbb{K}' \times \mathbb{K} \rightarrow \mathbb{K}$ が存在し, 以下を満たす
 - (a) $\forall x \in \mathbb{K}$ に対して $0 \cdot x = \bar{0}$
 - (b) $\forall x, y \in \mathbb{K}, c \in \mathbb{K}'$ に対して $(c \cdot x) \otimes y = c \cdot (x \otimes y)$

3.2 最短距離および周辺化最短距離

トレリス $G = (V, E)$ および半環 $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ が与えられたとき, V から \mathbb{K} への関数 w を考える。 v を引数としたときの w の値を $w[v]$ と書き, 頂点 v の重みと呼ぶことにする。また, この定義を拡張し G 上の有向パス

```

Data: A trellis  $G = (V, E)$ ,
         vertex weight  $w : V \rightarrow \mathbb{K}$ 
Result:  $\alpha[v] = D_{\mathcal{T},w}(\text{src}, v)$ 

for  $v \in \text{src}(G)$  do
  |  $\alpha[v] \leftarrow w[v]$ 
end
for  $v \in V \setminus \text{src}$  with a topological order do
  |  $\alpha[v] \leftarrow \bigoplus_{x \in \text{prev}(v)} \alpha[x] \otimes w[v]$ 
end

```

Algorithm 1: 一般化した前向き再帰

$\pi = (v_1, \dots, v_{|\pi|})$ の重みを $w[\pi] \stackrel{\text{def}}{=} w[v_1] \otimes \dots \otimes w[v_{|\pi|}]$ と定義する。このとき、(1) を一般化した次のような和の形式を定義する。

定義 5 (最短距離) トレリス上の任意の 2 頂点 $u, v \in V$ に対して、半環 $\mathcal{T} = (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ および頂点の重み w による u, v 間の最短距離 $D_{\mathcal{T},w}(u, v)$ を以下で定義する。

$$D_{\mathcal{T},w}(u, v) \stackrel{\text{def}}{=} \bigoplus_{\pi \in P(u \rightarrow v)} w[\pi] \quad (13)$$

同様に、群 $(\mathbb{G}, +, 0)$ 上の半環 $\mathcal{T} = (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ 、重み $w : V \rightarrow \mathbb{K}$ 、 f が与えられたとき、 $f : V \rightarrow \mathbb{G}$ によって周辺化した最短距離を以下のように定義する。

定義 6 (周辺化最短距離)

$$E_{\mathcal{T},w}[f] \stackrel{\text{def}}{=} \bigoplus_{\pi \in P(q)} \left(\sum_{v \in \pi} f(v) \right) w[\pi] \quad (14)$$

4 Lawler のアルゴリズムおよび前向き後ろ向きアルゴリズムの一般形

3 における定義に基づいて、一般化した前向き後ろ向きアルゴリズムを Algorithm 1 に掲げる。

アルゴリズムにおいて必要とされる演算を代数の言葉として抽象化することは自然で合理的な一般化はそれ自身重要である。代数の言葉で表現することで個別のアルゴリズムに対する統一かつ俯瞰的な視点と説明を得られる。最後に代数的抽象化にはそれ自身、ソフトウェア工学的な有用性を確かめることも出来る。特に汎用プログラミングの枠組みにおいては、対象の型に対する必要最小限かつ直交した操作 (axiom) のみから対象のアルゴリズムを構築する。これにより、柔軟で再利用性に優れたソフトウェアコンポーネントを提供できる。

Elements of Programming

```

Data: A trellis  $G = (V, E)$ ,
         vertex weight  $w : V \rightarrow \mathbb{K}$ 
         vertex function  $f : V \rightarrow \mathbb{G}$ 
Result:  $E = E_{\mathbb{K},w}[f]$ 

for  $v \in \text{snk}(G)$  do
  |  $\beta[v] \leftarrow \bar{1}$ 
end
for  $v \in V \setminus \text{snk}(G)$  with a topological order do
  |  $\beta[v] \leftarrow \bigoplus_{x \in \text{next}(v)} w[x] \otimes \beta[x]$ 
end
 $E \leftarrow \bar{0}$ 
for  $v \in V$  do
  |  $E \leftarrow E \oplus f(v) \cdot \alpha[v] \otimes \beta[v]$ 
end

```

Algorithm 2: 一般化した前向き後ろ向きアルゴリズム

5 重要と思われる具体例

本節では、上に挙げた枠組みに当てはまる具体的な半環および群上の半環のうち、系列データに機械学習・データマイニングを適用する上で非常に重要な役割を担うと思われるいくつかの具体例を挙げておく。

5.1 2 項畳み込み半環

n 次元実数ベクトル上における代数系 $\{\mathbb{R}^n, +, \diamond, \bar{0}, \bar{1}\}$ を以下のように定義する。ただし、 $\vec{x} = (x_1, \dots, x_n)^T \in \mathbb{R}^n$ 、 $\vec{y} = (y_1, \dots, y_n)^T \in \mathbb{R}^n$ とする。

$$\begin{aligned}
 \vec{x} + \vec{y} &= (x_1 + y_1, \dots, x_n + y_n)^T \\
 \vec{x} \diamond \vec{y} &= (x_1 y_1, x_1 y_2 + x_2 y_1, \dots \\
 &\quad \sum_{m=1}^k \binom{k}{m-1} x_m y_{k-m+1}, \dots)^T \quad (15) \\
 \bar{0} &= (0, \dots, 0)^T \\
 \bar{1} &= (1, 0, \dots, 0)^T
 \end{aligned}$$

$+$ は n 次元実数ベクトルに対する通常の加算であり、 \diamond は 2 項畳み込み (binomial convolution) などと呼ばれる演算である。これらの定義によるシステム $(\mathbb{R}^n, +, \diamond, \bar{0}, \bar{1})$ が半環を成すことは容易に確かめられる。

$\phi : V \rightarrow \mathbb{R}$ とする。この半環において特記すべきこととして、 $\vec{\phi} : V \rightarrow \mathbb{R}^n$ 、 $\vec{\phi}(v) = (1, \phi(v), \phi^2(v), \dots, \phi^{n-1}(v))^T$ とすると

$$\bigotimes_{v \in \pi} \vec{\phi}(v) = (1, \phi(\pi), \phi^2(\pi), \dots, \phi^{n-1}(\pi))^T \quad (16)$$

である。ただし， G 上の有向パス π に対して $\phi^n(\pi) \stackrel{\text{def}}{=} (\sum_{v \in \pi} \phi(v))^n$ とする。²

5.2 複素数

実数と同様，複素数 $\{x+iy \mid x, y \in \mathbb{R}, i \text{ は虚数単位}\}$ もまたその通常の加算・通常の乗算によって半群を成す。すなわち， $(\mathbb{C}, +, \times, 0, 1)$ は半群となり，頂点上に複素数の重みを定義した場合でも最短距離や周辺化最短距離は定義でき，それらは Algorithm 1, Algorithm 2 で計算可能である。ここで特記すべきことは $\phi: V \rightarrow \mathbb{R}$ に対して，頂点の重み w を $w[v] = \cos(\phi(v)) + i \sin(\phi(v))$ と定義することで以下が成立することである。

$$w[\pi] = \cos\left(\sum_{v \in \pi} \phi(v)\right) + i \sin\left(\sum_{v \in \pi} \phi(v)\right) \quad (17)$$

この事実とフーリエ級数展開を用いて，非常に広範な形式の和が近似計算できる可能性があることを示しておく。

今， $h(\sum_{v \in \pi} \phi(v))$ と定義されており， h が区間 $[\min_{\pi \in P(\text{src}, \text{snk})} \sum_{v \in \pi} \phi(v), \max_{\pi \in P(\text{src}, \text{snk})} \sum_{v \in \pi} \phi(v)]$ でフーリエ級数展開可能，たとえば h が 2 乗可積分であれば

$$\begin{aligned} & \sum_{\pi} h\left(\sum_{v \in \pi} \phi(v)\right) \\ &= \sum_{k=-\infty}^{+\infty} \left(a_k \sum_{\pi} \cos\left(\sum_{v \in \pi} \phi(v)\right) + b_k \sum_{\pi} \sin\left(\sum_{v \in \pi} \phi(v)\right) \right) \end{aligned} \quad (18)$$

ここで $a_k = \int_{-\text{inf}}^{+\text{inf}} h(x) \cos\left(\frac{2k\pi}{n}x\right) dx$, $b_k = \int_{-\text{inf}}^{+\text{inf}} h(x) \sin\left(\frac{2k\pi}{n}x\right) dx$ である。定式において a_k, b_k はトレリスの構造と無関係に h だけから決まる値であり，また，和 $\sum_{\pi} \cdot, \sum_{\pi} \cdot$ は先に述べたとおり $\{\mathbb{C}, +, \times\}$ に対する Lawler のアルゴリズムによって求まる。このことから，この形式の和はフーリエ級数展開したものを有限項数で打ち切って計算することで近似計算できる。

5.3 双対数

双対数 \mathbb{D} を $\{x + y\varepsilon \mid x \in \mathbb{R}, y \in \mathbb{R}, \varepsilon \neq 0, \varepsilon^2 = 0\}$

² $\mathcal{P}_n(x)$ を n 次多項式とするとき， $\mathcal{P}_n(x+y) = \sum_{k=0}^n \binom{n}{k} \mathcal{P}_k(x) \mathcal{P}_{n-k}(y)$ が成立する多項式列 $\{\mathcal{P}_n\}_n$ を 2 項型の多項式列 (polynomial sequence of binomial type) と呼ぶ。ここでの結果は単項式の列 $\{x^n\}_n$ が 2 項型の多項式列であるから，とも説明できる。他に下側階乗列 $\{x(x-1)\cdots(x-n+1)\}_n$ ，上側階乗列 $\{x(x+1)\cdots(x+n-1)\}_n$ などが 2 項型の多項式列の例である。

6 関連研究

謝辞

....

参考文献

- [1] T. Kudo, K. Yamamoto, and Y. Matsumoto. Applying conditional random fields to Japanese morphological analysis. In *Proc. of EMNLP*, Vol. 2004, 2004.
- [2] M. Mohri. Semiring frameworks and algorithms for shortest-distance problems. *Journal of Automata, Languages and Combinatorics*, Vol. 7, No. 3, pp. 321–350, 2002.
- [3] S. Sarawagi and W.W. Cohen. Semi-markov conditional random fields for information extraction. *Advances in Neural Information Processing Systems*, Vol. 17, pp. 1185–1192, 2005.