第8章

再帰と再帰関係

フィボナッチ数列

ゼロ、11、2、35、8そして13、21！こ
の調子で**フィボナッチが現れる**;
この男の長年のシークエンス
数学の学生は遅くまで勉強している。

ゴールディ、リムリック形式の万能英語辞典

コンピュータ・サイエンスに興味を持つ人が習得しなければならない必須ツールは、再帰的思考法である。再帰的に提示される定義、概念、アルゴリズムなどを理解する能力と、思考を再帰的な枠組みに当てはめる能力は、コンピュータ・サイエンスにおいて不可欠である。本章の目標の1つは、読者がよく遭遇する形の再帰に慣れることである。

第二の目標は漸化式について議論することである。生成関数の紹介を含め、漸化関係の解法に集中する。

8.1 再帰のさまざまな顔

以下の定義について考えてみよう。これらを読むときは、それらがどのように似ているかに集中してください。

8.1.1 二項係数

ここでは、第2章で紹介した二項係数の再帰的定義を紹介する。

定義 8.1.1 二項係数 - 再帰定義。 仮定 $n \geq 0$かつ$n \geq k \geq 0$である。

- $\binom{n}{0} = 1$
- $\binom{n}{n} = 1$とする。

- $n > k > 0$の場合、$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$

◇

観察8.1.2 定義について一言：厳密に言えば、二項係数のような数学的対象が定義される場合、それらは一度だけ定義されるべきである。先に定義2.4.3で二項係数を定義したので、二項係数を説明する他の文は定理でなければならない。この場合の定理とは、上の「定義」が元の定義と矛盾しないということである。この章で再帰について論じる目的は、再帰的な性質を持つ別の定義を観察することである。演習では、2つの定義が本当に等価であることを証明する機会がある。

以下は、再帰的定義を適用して$\binom{5}{2}$を計算する方法である。

$$\begin{aligned}
\binom{5}{2} &= \binom{4}{2} + \binom{4}{1} \\
&= \left(\binom{3}{2} + \binom{3}{1}\right) + \left(\binom{3}{1} + \binom{3}{0}\right) \\
&= \binom{3}{2} + 2\binom{3}{1} + 1 \\
&= \left(\binom{2}{2} + \binom{2}{1}\right) + 2\left(\binom{2}{1} + \binom{2}{0}\right) + 1 \\
&= (1 + \binom{2}{1}) + 2(\binom{2}{1} + 1) + 1 \\
&= 3\binom{2}{1} + 4 \\
&= 3\left(\binom{1}{1} + \binom{1}{0}\right) + 4 \\
&= 3(1 + 1) + 4 = 10
\end{aligned}$$

8.1.2 多項式とその評価

定義8.1.3 *S*上の*x*における多項式表現（非帰納的） nを整数とする。$a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$ の n^{th} 次多項式は、$a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$ の形の式である。a_1 , a_0 は、係数集合と呼ばれる指定された数集合*S*の要素であり、以下の式で表される。

$a_n \neq 0$. ◇

*ここではxを*変数と呼ぶが、より正確には*不定と呼ぶ*。不定と変数には区別があるが、この議論ではその区別は関係ない。

次多項式は定数多項式と呼ばれ、単に係数の集合の要素である。

この定義は、代数学の授業で$f(n) = 4n^3 + 2n^2 8n + 9$ のような式を説明するためによく導入される。

n.この定義には、変数に値が与えられて式を評価しなければならない場合に欠点がある。例えば、$n = 7$と仮定しよう：

$$
\begin{aligned}
f(7) &= 4 \cdot 7^3 + 2 \cdot 7^2 - 8 \cdot 7 + 9 \\
&= 4 \cdot 343 + 2 \cdot 49 - 8 \cdot 7 + 9 \\
&= 1423
\end{aligned}
$$

実行された演算の数を数えてみると、5つの乗算と3つの加減算が実行されている。最初の2つの乗算は72と7^3を計算し、最後の3つは係数に7の累乗をかける。これにより4つの項が得られる。*k個の*数のリストを加減算するには、k 1回の加減算が必要である。次の多項式の定義は、より効率的な別の評価方法を示唆している。

定義8.1.4 *S*上の*x*の多項式表現（再帰的）。とする。
Sは係数の集合、xは変数である。

(a) *S*上の*x*の0次多項式は*S*の非ゼロ要素である。

(b) $n \geq 1$において、*S*上の*x*のn^{th} 次多項式は、$p(x)x+a$の形の式であり、$p(x)$は*x*の(**n**-*1*)次多項式であり、$a \in S$である。

$f(n) = 4n^3 + 2n^2 - 8n + 9$ が3次であることは簡単に検証できる。◇

この定義に基づくZ上の*n*の多項式:

$$f(n) = 4n^3 + 2n^2 - 8n + 9 = ((4n + 2)n - 8)n + 9$$

4は整数なので0次の多項式であることに注意。したがって、$4n + 2$は1次多項式であり、したがって、$(4n + 2)n$ 8はZ上の*n*の2次多項式であり、したがって、$f(n)$はZ上の*n*の3次多項式である。
Z,$f(n)$の最終式を**伸縮形と**呼ぶ。これを$f(7)$の計算に使うと、必要なのは3回の乗算と3回の加減算だけである。これは多項式を評価する**ホーナーの方法と**呼ばれる。

例8.1.5 テレスコープ多項式。

(a) $p(x) = 5x^4 + 12x^3 - 6x^2 + x + 6$ の伸縮形は、$(((5x + 12)x - 6)x+1)x+6$。ホーナーの方法を用いると、$p(c)$の値を計算するには、任意の実数*cに対して*4回の乗算と4回の加減算が必要である。

(b) $g(x) = x^5 + 3x^4 + 2x^2 + x$ は、伸縮形式 $(((x + 3)x)x - 2)x + 1)x$。

□

多くのコンピュータ言語は多項式を係数のリストとして表現する、通常は定数項から始まる。例えば，$g(x) = x^5 + 3x^4 + 2x^2 + x$は，リスト {0, 1, 2, 0, 3, 1} で表される．Mathematica とSageでは，多項式を入力し操作することができるので，リスト表現は内部的なものに過ぎない．プログラミング言語の中には，リストを使って多項式操作をプログラムすることを要求するものもある．このようなプログラミングの問題は別の機会に譲ることにする．

8.1.3 再帰検索 - バイナリ検索

次に、項目のソート{}されたリスト内の2値探索の再帰的アルゴリズムを考える。$r = r(1), r(2), \ldots r(n)$は、ソートされた$n$個の項目のリストを表すとする。

を数値キーで降順に並べたものである。j^{th} の項目を$r(j)$、そのキー値を$r(j)$.keyとする。例えば、各項目は都市の建物のデータを含み、キー値は建物の高さかもしれない。その場合、$r(1)$は最も高い建物の項目となり、$r(1)$.keyはその高さとなる。そして

これは、BinarySearch(1, n)を実行することで達成される。アルゴリズムが完了すると、変数Foundの値はtrueになる。
Foundがfalseの場合、そのような項目はリストに存在しない。このアルゴリズムの一般的な考え方を以下に示す。

図8.1.6

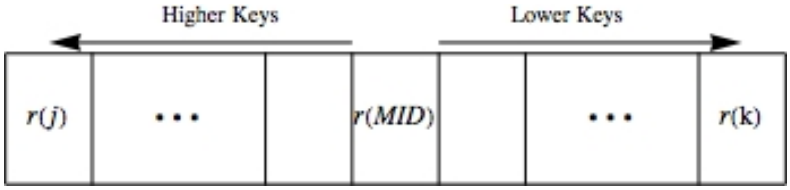

図8.1.6 バイナリサーチの一般的スキーム

以下の SageMath におけるバイナリサーチの実装では、整数のソートされたリスト内を検索する。したがって、項目そのものがキーとなります。

```
def バイナリサーチ (r,j,k,C):
    found = False
    j <= k:
        mid = floor ((j + k)/2)
        print (' プロービング␣位置␣'+ str (mid))
        if r[ mid ] == C:
            location = mid
            found = True
            print (' 見つかりました␣␣位置␣'+ str (位置 ))
            リターンロケーション
        でなければ .
          もしr[ mid ] > C:
              バイナリサーチ (r,j, mid - 1 ,C)
          でなければ .
              バイナリサーチ (r, mid + 1 ,k,C)
    でなければ .
        プリント (' not# found ')
        偽を返す
s=[1 ,9 ,13 ,16 ,30 ,31 ,32 ,33 ,36 ,37 ,38 ,45 ,49 ,50 ,52 ,61 ,63 ,64 ,69 ,77 ,79 ,80 ,81 ,83 ,86 ,90 ,93 ,96]
バイナリサーチ (s,0 , len (s) -1 ,30)
```

```
13位でプロービング 6位でプロ
ービング 2位でプロービング 4
位で発見
```

8.1.4 再帰的に定義されたシーケンス

次の2つの例では、数列を0番目の数、1番目の数、2番目の数からなる数のリストと考える、

.....

もし

シーケンスにSという名前を付ける場合、Sのk^{th} 番号は通常、S_k と書かれる。

$S(k)$。

例8.1.7 幾何学的成長数列。数列Bを次式で定義する。

$$B_0 = 100$$

$$B_k = 1.08B_{k-1} \text{ for } \mathbf{k} \geq 1.$$

このルールでは、リストの各数字は前の数字の1.08倍であり、開始数字は100に等しいと規定されている。例えば

$$\begin{aligned} B_3 &= 1.08B_2 \\ &= 1.08\ (1.08b\ )_1 \\ &= 1.08\ (1.08b_0\ )) \\ &= 1.08(1.08(1.08 - 100)) \\ &= 1.^{083} - 100 = 125.971 \end{aligned}$$

例

8.1.8 フィボナッチ数列。フィボナッチ数列は次式で定義される数列F*である。*

$$F_0 = 1,\ F_1 = 1 \text{ and}$$
$$F_k = F_{k-2} + F_{k-1} \text{ for } k \geq 2$$

□

8.1.5 再帰

これまでの例はすべて再帰的に示された。つまり、すべての "オブジェクト "は2つの形式のいずれかで記述されている。ひとつは単純な定義によるもので、これは通常、再帰の基礎と呼ばれる。もうひとつは、再帰的な記述によるもので、オブジェクトはそれ自体について記述される。再帰を適切に使用するために不可欠なのは、対象をより単純な対象で表現できることである。循環的な定義と見なされることを避けるためには、再帰を有限回適用した後に基底に到達しなければならない。

例えば、フィボナッチ数列の4番目の項目を決定するには、F_0 と F_1 を含む式が残るまで、Fの再帰的ルールを繰り返し適用する：

$$\begin{aligned} F_4 &= F_2 + F_3 \\ &= (f_0 + f_1\ ) + (f_1 + f)_2 \\ &= (f_0 + f_1\ ) + (f_1 + (f_0 + f_1\ )) \\ &= (1 + 1) + (1 + (1 + 1)) \\ &= 5 \end{aligned}$$

8.1.6 反復

一方、F_5 のようなフィボナッチ数列の項を計算するには、基底項から始めて次のように進める：

表8.1.9

$$\begin{aligned} f_2 &= f_0 + f_1 = 1 + 1 = 2 \\ f_3 &= f_1 + f_2 = 1 + 2 = 3 \\ f_4 &= f_2 + f_3 = 2 + 3 = 5 \end{aligned}$$

$$f_5 = f_3 + f_4 = 3 + 5 = 8$$

これはフィボナッチ数列の反復計算と呼ばれる。ここでは基礎から始めて、次のような単純でない数まで計算を進める。

5. F_4 でやったように、Fの再帰的定義を使ってF_5 を計算してみよ う。 上記の計算にかかった時間よりも、ずっと長い時間がかかるだろう。反復計算は通常、再帰を適用した計算よりも速くなる傾向がある。従って、可能であれば、再帰式を非再帰式、例えば繰り返し計算だけで済む式やより高速な方法に変換できることが、役に立つスキルの1つである。

また、n の反復式は、再帰的定義の適用よりもはるかに効率的である。しかし、再帰的定義にも利点がないわけではない。第一に、再帰式は一組の含みを論的解釈。1,2, ..., nからk個の要素を選ぶ、すべてのkを選ぶ方法は"n-1" 通 k 1 2よ{, 1 であり、"n-1" が存在する。の選び方 もし 要素 k nが選択され、残りのkが選択される。k-1 1 {1, 2, ..., n - } .第2章の「足し算の法則」を使って推論したことに注意しよう。

*バイナリサーチ再訪。*バイナリサーチアルゴリズムにおいて、再帰が使われている箇所を選ぶのは簡単だ。ある項目が調べられ、そのキーが目的のものでない場合、検索は以前検索していた項目の数の半分以下のサブリストに切り捨てられる。明らかに、これはより単純な検索である。基本はアルゴリズムに隠されている。検索を完了する2つのケースは、基礎と考えることができる。$j > k$のとき、欲しいアイテムが見つかるか、検索し残したサブリストが空になるかのどちらかである。

BinarySearchは、そのサブプログラムへの再帰的な呼び出しが可能な言語であれば、それほど難しいことなく翻訳することができる。このようなプログラムの利点は、バイナリサーチを行う非再帰的プログラムよりもコーディングがずっと短くなることである。しかし、ほとんどの場合、再帰バージョンは遅くなり、実行時に多くのメモリを必要とする。

8.1.7 帰納法と再帰

ペアノの定石による正の整数の定義は再帰的定義である。基本要素は数1であり、再帰はnが正の整数であればその後継も同じであるというものである。この場合、nは単純な対象であり、再帰は前進型である。もちろん、帰納法の証明の有効性は、この定義を受け入れることに基づいている。したがって、再帰が使われるときに帰納証明が現れるのは偶然ではない。

例8.1.10 Bの**公式の証明** 数列Bの公式
帰納法による証明が続く。 ≥

$k = 0$とすると、$B = 100(1.08)^0 = 100$となる。ここで、ある **k≥** 1, B_k の公式が成り立つ。

$$B_{k+1} = 1.08B_k \quad \text{再帰的定義による}$$
$$= 1.08\ 100(1.08)^k \quad \text{帰納仮説により}$$
$$= 100(1.08)^{k+1}$$

したがって、式は$k + 1$について成り立つ

Bについて証明した式を閉形式と呼ぶ。この式には再帰や和の記号は含まれない。 □

定義8.1.11 閉じた形式式。 $E = E(x_1, x_2, \ldots, x_n)$ を変数 $x_1, x_2, \ldots$ を含む代数式とする。x_n を含む代数式とする。Eが**閉形式**であるのは、Eを任意の許された

変数の値の演算はT回まで（あるいは、T
時間単位）。 ◇

例題8.1.12 和を閉じた形 に する。 和
$E(n) = \sum_{k=1}^{n}$ kは閉形式にならない。
$E(n)$の値を計算する閉形式は$\frac{n(n+1)}{2}$ で、$T = 3$回の操作で済む。

□

8.1.8 エクササイズ

1. 二項係数の再帰的定義により、$\binom{7}{6} = \binom{6}{6} + \binom{6}{5}$、コンピュータ、それを基礎に定義された量で表現するために拡張を続ける。n の階乗の定義^{2}を適用して，結果をチェックしよう．$\binom{n}{k}$

2. 数列Lを$L_0 = 5$で定義し、$k \geq 1$の場合、$L_k = 2L_{k-1} - 7$。を決定する。L_4、$L_k = 7 - 2^k + 1$であることを帰納法によって証明する。

3. $p(x) = x^5 + 3x^4 - 15x^3 + x - 10$ とする。

   (a) $p(x)$を伸縮形式で書く。

   (b) 電卓を使ってp(x)の原形を使って$p(3)$を計算する。

   (c) 電卓を使って、$p(x)$の伸縮形を使って$p(3)$を計算しなさい。

   (d) パートbとcのスピードを比べてください。

4. 9個の項目からなるリスト$(r(1), r(2), ..., r(9))$が、$r(3)$.key=12、$r(4)$.key=10となるように、キーの降順でソートされているとする。を完了するために必要なBinarySearchアルゴリズムの実行を列挙せよ。

   BinarySearch(1,9) when：

   (a) 検索キーは C = 12　　(b) 検索キーはC = 11である。

   異なるアイテムは異なるキーを持つと仮定する。

5. 以下のfの定義のどこが間違っているのだろうか? $f(0) = 1$ であり $x \neq 0$ならば、$f(x) = f(x/2)/2$ となる。

6. 二項係数の2つの定義、定義2.4.3と定義8.1.1が等価であることを証明せよ。

7. n≧0 のとき、$\sum_{k=0}^{n} \binom{n}{k} = 2^n$ が成り立つこと
   を帰納法で証明せよ。

8.2 試合順

8.2.1 シークエンスとその定義

定義 8.2.1 数列。数列は自然数からある所定の集合への関数である。

任意の自然数kのイメージは$S(k)$またはS_kと書くことができ、Sのk^{th} *項*と呼ばれる。

配列の*インデックス*または*引数*と呼ばれる。 ◇

例えば、整数の列は関数$S : \mathrm{N} \rightarrow \mathrm{Z}$となる。

例8.2.2 異なる方法で定義された3つのシーケンス。

(a) $A(k) = k^2 - k, \mathbf{k} \geq 0$ で定義される数列Aは整数の数列である。

(b) $B(0)=2$、$B(k)=B(k-1)+3$で再帰的に定義される数列Bは、$k \geq 1$の整数の数列である。Bの項は次のように計算できる。

再帰の公式を適用するか、反復する。例えば

$$\begin{aligned} b(3) &= b(2) + 3 \\ &= (B(1) + 3) + 3 \\ &= ((B(0) + 3) + 3) + 3 \\ &= ((2 + 3) + 3) + 3 = 11 \end{aligned}$$

または

$$\begin{aligned} b(1) &= b(0) + 3 = 2 + 3 = 5 \\ b(2) &= b(1) + 3 = 5 + 3 = 8 \\ b(3) &= b(2) + 3 = 8 + 3 = 11 \end{aligned}$$

(c) C_r、長さrの0と1の文字列のうち、連続した0を持たないものの数とする。これらの項は整数列Cを定義する。

□

備考

(1) 数列はしばしば*離散関数*と呼ばれる。

(2) シーケンスが関数であることを念頭に置くことは重要だが、シーケンスを視覚化するもう1つの便利な方法はリストである。例えば、先の例の配列Aは（0, 0, 2, 6, 12, 20, ...）と書くことができる。有限の数列は、コンピュータへの入力やコンピュータからの出力でも同じように表示される。シーケンスのインデックスは時間変数と考えることができる。ある数列の項が1秒ごとに画面上で点滅するとしよう。そうすると、s_k、k^{th} 秒に表示されることになる。このような用語をシーケンスの説明に使うのは便利である。たとえば、Aのk番目の項（th）に先行する項は、$A(0)$、$A(1)$、...、$A(k\ 1)$となる。これらの項を先行項と呼ぶこともできる。

8.2.2 根本的な問題

任意の数列の定義が与えられたとき、我々が関心を持つ基本的な問題は、最小限の時間で任意の特定の項を決定する方法を考案することである。一般に、時間は必要な操作の回数と等しい。操作を数える場合、再帰式の適用は操作とみなされる。

(a) 例題8.2.2のAの項は、閉形式のため計算が非常に簡単である。どの項を計算するにしても、必要な操作は3つだけである。

(b) Bの項の計算方法はそれほど明確ではない。$B(100)$を知りたかったとしよう。一つの方法は、定義を再帰的に適用することである：

$b(100) = b(99) + 3 = (b(98) + 3) + 3 =$ **- - - -** $b(100) = b(99) + 3 = (b(98) + 3) + 3 =$ **- -**

Bの再帰式が100回適用され、100回の加算が続くことになる。この方法で$B(k)$を計算するには、2k回の演算が必要である。$B(k)$の反復計算は改善される：$B(1) = B(0) + 3 = 2 + 3 = 5$

$b(2) = b(1) + 3 = 5 + 3 = 8$

など。kの追加だけが必要だ。これはまだ良い状況ではない。kが大きくなると、$B(k)$を計算するのにますます時間がかかる。式

$B(k)=B(k1)+3$は、Bの漸化式と呼ばれる。$B(k)$の閉形式、つまりある一定回数以上の演算を必要としない式を求めるプロセスを、漸化式を解くと呼ぶ。

(c) C_k の決定は、組合せ論における標準的な問題である。その解決法の一つは漸近関係である。実際，組合せ論の多くの問題は，まず漸化式を探し，それを解くことで，最も簡単に解ける．次の観察は，C_k を決定するために必要な漸化式を示唆する． k 2の場合，長さkで連続する0が2つない0と1の文字列はすべて， $1s_{k1}$ または $01s_{k2}$ のいずれかである．ここで， s_{k1} と s_{k2} は，それぞれ長さk 1とk 2の連続する0が2つない文字列である．この観察から、k 2の場合、$C_k = C_{k2} + C_{k1}$ である ことがわかる。$C_0 = 1$と$C_1 = 2$は列挙によって簡単に決定できる。さて、反復によって、どのC_kも簡単に決定できる。例えば、$C_5 = 21$は5回の足し算で計算できる。C_k の閉形式があれば、さらに良くなる。C_k の漸化式はフィボナッチ数列の漸化式と同じである。基底だけが異なる。

8.2.3 エクササイズ

1. $B(k) = 3k + 2$, $k \geq 0$, が例題8.2.2の数列Bの閉形式であることを証明せよ。

2.

(a) $Q(k) = 2k + 9$, k で定義されるシーケンスQを考える。 1.以下の表を完成させ、$Q(k)$ - $Q(k - 1)$の式を記述する漸近関係を求めよ。

k	$Q(k)$	$Q(k)$ - $Q(k - 1)$
2		
3		
4		
5		
6		
7		

(b) $A(k) = k^2$ - k, $k \geq$ する。 0.下の表を完成させ、Aの漸化式を求めよ。

k	$A(k)$	$A(k)$ - $A(k$ - 1)	$A(k)$ - $2A(k$ - 1) + $A(k$ - 2)
2			
3			
4			
5	≥		

3. つの直線が平行でなく、3つの直線が同じ点で交わらないような

平面上のk本の直線（$k \geq 0$）が与えられたとき、直線が平面を分割する領域の数を$P(k)$とする（無限の領域も含む（図8.2.3参照））。$P(k) = P(k-1) + k$という漸化式がどのように導かれるかを述べよ。$P(0) = 1$とすると、$P(5)$を求めよ。

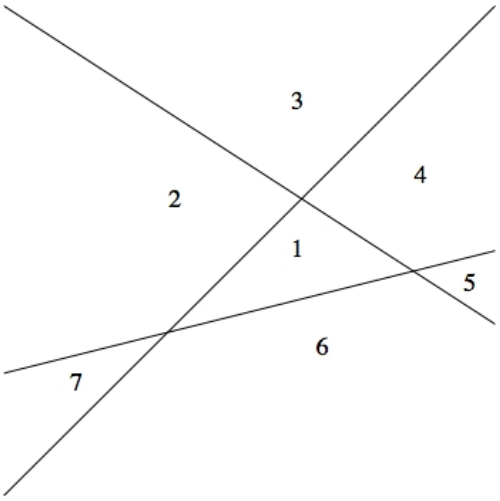

図8.2.3 3本のラインの一般的な構成

4. ある放射性物質の試料は1時間ごとに0.15%減衰すると予想される。w_t , $t \geq 0$, が実験開始t 時間後の試料の重量であるとき、w の漸化式を書きなさい。

5. n^{th} 次多項式を評価するのに必要な乗算の回数を $M(n)$ とする。多項式の再帰的定義を使って M を再帰的に定義せよ。

6. Sを整数の列とする。この2つの命題は等価か?

   (a) $(\forall M)_{\mathbb{N}}\ ((\exists n)_{\mathbb{N}}\ (S(n) \geq M))$

   (b) $(\forall M)_{\mathbb{N}}\ ((\exists N)_{\mathbb{N}}\ ((\forall n)_{\mathbb{N}}\ (n \geq N \rightarrow S(n) \geq M)))$

8.3 再帰関係

このセクションでは、漸化式とその解の研究を始める。ここでは、定数係数を持つ有限次線形漸化関係（略して有限次線形漸化関係）に焦点を当てる。まず、これらの関係がどのような閉形式から生じるかを調べる。次に、これらの関係を解くためのアルゴリズムを紹介する。後の節では、他の一般的な関係(8.4)を考察し、漸化式を研究するための2つの追加的なツール、生成関数(8.5)と行列法(第12章)を紹介する。

8.3.1 定義と用語

定義8.3.1 再帰関係。Sを数列とする。S上の漸化*関係とは*、S*の*有限個の項を除くすべての項をS*の*前の項と関係付ける式のことである。 すなわち、S*の*定義域には、k_0 、$S(k)$ *が*$S(k)$の前の項*の*いくつか（場合によってはすべて）で表現されるようなk_0 が存在する。S*の*定義域が 0, 1, 2, . . *の場合*、項$S(0)$, $S(1)$, ..., $S(k_0 - 1)$は漸化式で定義されない。そ

れらの値は、初期条件（または境界条件、または基底）であり、その値によって
Sの定義 ◇

例8.3.2 再帰関係のいくつかの例。

(a) フィボナッチ数列は漸化式 $F_k = F_{k-2} + F_{k-1}$, $k \geq 2$, 初期条件 $F_0 = 1$, $F_1 = 1$ で定義される。F_k は F の2つ前の項に依存するので、この再帰関係は2次関係と呼ばれる。セクション8.2、例題8.2.2の数列Cは、同じ漸化式で定義できるが、初期条件が異なることを思い出してほしい。

(b) $T(k) = 2T(k-1)^2 - kT(k-3)$の関係は3次の漸化式である。$T(0)$、$T(1)$、$T(2)$の値が指定されれば、$T$は完全に定義される。

(c) $S(n)=S((n/2)+5$、$n>0$、$S(0)=0$の漸化式は無限次項を持つ。nが偶数のときに$S(n)$を決定するには、$n/2$項さかのぼらなければならない。$n/2$はnとともに無限に大きくなるので、Sに有限次数を与えることはできない。

□

8.3.2 再帰関係を解く

数列は多くの場合、漸化式で最も簡単に定義できるが、漸化式を直接適用して項を計算するのは時間がかかる。漸化式から数列の項の閉形式を求めるプロセスを、漸化式を解くという。すべての漸化式を解くのに使える単一の手法やアルゴリズムはない。実際、解けない漸化式もある。上のTを定義する関係もその一例である。皆さんが今後遭遇するであろう漸化式のほとんどは、定数係数を持つ有限次線形漸化式に分類されます。本章で最も時間を費やすのはこのクラスである。

定義8.3.3 n^{th} 次線形回帰関係。 S を領域 $k \geq 0$ を持つ数列とする。定数係数を持つS上のn^{th} 次線形漸化関係とは、次の形式で書ける漸化関係である。

$$S(k) + C_1 S(k-1) + \ldots + C_n S(k-n) = f(k) \text{ for } k \geq n$$

ここで、C_1、C_2、$\ldots C_n$ は定数であり、f は次のように定義される数値関数である。$k \geq n$.

注：このクラスの関係をn^{th} order linearと略す。◇

の関係にある。したがって、今後の議論では、$S(k) + 2kS(k-1) = 0$ は一次線形関係とはみなされない。

例8.3.4 いくつかの有限次線形関係。

(a) $F_k - F_{k-1} - F_{k-2} = 0$ であるから、フィボナッチ数列は2次の線形関係によって定義される。

(b) $P(j) + 2P(j-3) = j$。$P(j-3) = j^2$ は3次の線形関係である。この

場合、$C_1 = C_2 = 0$となる。

(c) $A(k)$=2($A(k1)$+k)の関係は、$A(k)$ $2A(k\ 1) = 2$kと書くことができる。したがって、これは一次の線形関係である。

□

8.3.3 "解 "から得られる再帰関係

有限次線形関係を解くアルゴリズムを与える前に、ある閉形式から生じる漸化式を調べる。閉形式は、そこから有限次線形関係が得られるように選ばれる。このアプローチは少々作為的に見えるかもしれないが、簡単な代数式をいくつか書いてみると、そのほとんどがこれから調べる式と似ている可能性がある。

最初の例として、$D(k) = 5^{2k}, k \geq 0$ で定義されるDを考える。 $D(k) = 5^{2k} = 2 \cdot 5^{2k-1} = 2D(k-1)$。したがって、$D$は一次の線形関係 $D(k) - 2D(k-1) = 0$を満たし、初期条件$D(0) = 5$がDの初期条件となる。

第二の例として、$C(k) = 3^{k-1} + 2^{k+1} + k, k \geq 0$ を考えてみよう。この結果を得るためには、さらに代数的な操作が必要である：

表8.3.5

$C(k) = 3^{k-1} + 2^{k+1} + k$	元の式 $3C(k-1) =$
$3^{k-1} + 3 \cdot 2^{k} + 3(k-1)$	$k-1$をkに代入
	に3を掛ける
	最初の式から2番目の式を
	引く。
$C(k) - 3C(k-1) = 2^{k} - 2k + 3$	3^{k-1}項は消去される。
	これは一次の関係である
。$2C(k-1) - 6C(k-2) = 2^{k} - 2(2(k-1)) + 6$	k のに1を代
入する。	
	3番目の方程式に2を掛ける。
	3番目の方程式から4番目の方程式を引く。
$C(k) - 5C(k-1) + 6C(k-2) = 2k - 7$	2k+1項は消去される。
	これは2次の関係である。

先ほど求めた再帰関係は、$k \geq 2$に対して定義され、初期条件$C(0) = 7/3$ と$C(1) = 6$とともに、Cを定義する。

表8.3.6は、読者に導出させる他のいくつかの例とともに、我々の結果をまとめたものである。これらの結果から、指数式と多項式を組み合わせた数列の閉形式はすべて有限次線形関係の解になると推測できる。これは真であるだけでなく、逆も真である：有限次線形関係は、今調べたものと似た閉形式を定義する。追加で必要な情報は初期条件のセットだけである。

表8.3.6 与えられた系列から得られる再帰関係

閉形式	再帰関係
$D(k) = 5 - {}^{2k}$	$D(k) - 2D(k - 1) = 0$
$C(k) = {}^{3k-1} + {}^{2k+1} + k$	$C(k) - 2C(k - 1) - 6C(k - 2) = 2k - 7$
$Q(k) = 2k + 9$	$Q(k) - Q(k - 1) = 2$
$A(k) = k^2 - k$	$A(k) - 2A(k - 1) + A(k - 2) = 2$
$B(k) = 2k^2 + 1$	$B(k) - 2B(k - 1) + B(k - 2) = 4\ G(k)$
$= 2 - {}^{4k} - 5(-3)^k$	$G(k) - G(k - 1) + 12G(k - 2) = 0$
$J(k) = (3 + k)^{2k}$	$J(k) - 4J(k - 1) + 4J(k - 2) = 0$

定義8.3.7 同次漸化関係。 n^{th} 、すべてのkについて$f(k) = 0$のとき、次数線形関係は同次である。

$S(k) + C_1\ S(k - 1) + . . + C_n\ S(k - n) = f(k)$, 関連する同次関係は $S(k) + C_1\ S(k - 1) + . . + C_n\ S(k - n) = 0$ ◇

例8.3.8 一次の同次漸化式関係。 $D(k)\ 2D(k\ 1) = 0$ は一次の同次関係である。これは$D(k) = 2D(k\ 1)$とも書けるので、2の累乗を含む式から生じても不思議ではない。 より一般的には、$L(k)\ aL(k\ 1)$の解は^{k}を含むと予想される。 実際には解は
$L(k) = L(0)a^k$ 、$L(0)$の値は初期条件によって与えられる。 □

例題8.3.9 二次の例題。 $S(k)\ 7S(k\ 1) + 12S(k\ 2) = 0$と初期条件$S(0) = 4$と$S(1) = 4$の2次同次関係式を考える。上記の議論から、この関係式の解には ba^k という形の項が含まれることが予測できる。ここで b と a は決定されなければならない非ゼロ定数である。解がこの量に正確に等しいとすると

$$S(k) = ba^k$$
$$S(k - 1) = ba^{k-1}$$
$$S(k - 2) = ba^{k-2}$$

これらの式を漸化式に代入すると、次のようになる。

$$BA^k - 7BA^{k-1} + 12BA^{k-2} = 0$$

この方程式の左辺の各項は、0でないba^{k-2} の因子を持つ。この共通因数で除算すると

$$a^2 - 7a + 12 = (a - 3)(a - 4) = 0 \qquad (8.3.1)$$

したがって、aの取りうる値は3と4だけである。式(8.3.1)は漸化式の特性方程式と呼ばれる。このことは、$S(k) = b_1\ ^{3k} + b_2\ ^{4k}$（ここで、$b_1$ 、b_2 は実数） という形の任意の数列に対して、われわれの元の漸化式が成り立つということである。この数列の集合を漸化式の一般解と呼ぶ。もしSの初期条件がなかったら、ここで止まってしまうだろう。初期条件があれば、b_1 と b_2 の確定値を求めることができる。

$$\begin{array}{l} S(0) = 4 \\ S(1) = 4 \end{array} \Rightarrow \begin{array}{l} b_1\ ^{30} + b_2\ ^{40} = 4 \\ b_1\ ^{31} + b_2\ ^{41} = 4 \end{array} \Rightarrow \begin{array}{l} b_1 + b_2 = 4 \\ 3b_1 + 4b_2 = 4 \end{array}$$

この連立方程式の解は、b_1 ＝12、b_2 **＝-8で**あり、解は$S(k)$＝12 - ^{3k} - 8 - ^{4}kとなる。 □

定義 8.3.10 特性方程式. 同次 n^{th} ��線形関係式 $S(k)+C_1$ S(k 1)+. .+C S(k n) = 0 の特性方程式は、n 次多項式方程式である。$.+C_n$ S(k n) = 0 はn次の多項式方程式である。

$$a^n + \sum_{j=1} C\ a_j^{n-j} = a^n + C\ a + + \text{-}C\ a + C = 0_1^{n-1} \qquad + C_{n-1}\ a + C_n = 0$$

この方程式の左辺を特性多項式と呼ぶ。特性多項式の根を方程式の特性根と呼ぶ。 ◇

例8.3.11 いくつかの特性方程式。

(a) $F(k)-F(k-1)-F(k-2)=0$の特性*方程式は*、$x^2-x-1=0$である。

(b) $Q(k)+2Q(k-1)-3Q(k-2)-6Q(k-4)=0$の特性方程式。

は、$^4 + 2a^3 - 3a^2 - 6 = 0$である。$Q(k\ 3)$の項がないことは、特性方程式に$x^{4-3} = x$の項が現れないことを意味する。

アルゴリズム 8.3.12 同次有限次線形関係を解くアルゴリズム。

(a) $S(k) + C_1\ S(k - 1) +$ *の関係式の特性方程式を書き出す。*
$.. .+ C_n\ S(k - n) = 0$ *であり、これは*$^n + C\ a_1^{n-1} + - - + C_{n-1}\ a + C_n = 0$である。

(b) 特性方程式のすべての根、特性根を求める。

(c) n *個の異なる特性根、*a_1 、a_2 、$... a_n$ *がある場合、漸近関係の一般解は* $S(k) = b\ a_{11}^k + b\ a_{22}^k + + b\ a_{nn}^k$ *である。特徴的な根が*n *より少ない場合、少なくとも*1 *つの根は重根である。*$a^{p-1}{}_j$ *が重根の場合，* $b\ a_{jj}^k$ *項は* $(b_{j,0} + b_{j,1}\ k)\ a^k$ *に置き換えられる． 一般に，* a_j *が多重度* p *の根の場合，* $b\ a_{jj}^k$ *項は* $b_{j,0} + b_{j,1}\ k + - - + b_{j,(p-1)}\ k\ a^k$ *に置き換えられる．*

(d) n *個の初期条件が与えられた場合、代入によって*n *個の未知数（ステップ*3 *の*$b_j's$）のn *個の一次方程式が得られる。可能であれば、これらの方程式を解いて*$S(k)$*の最終形を決定する。*

このアルゴリズムはnのすべての値に対して有効であるが、このアルゴリズムが実行可能なnの大きさには限界がある。鉛筆と紙を使えば、いつでも2次方程式を解くことができる。根の2次式

のは$ax^2 + bx + c = 0$

である。
$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

$a^2 + C_1\ a + C_2 = 0$ の解は次の通りである。

$$\frac{1}{2}(-C_1 + \sqrt{C_1^2 - 4C_2}) \text{ および } \frac{1}{2}(-C_1 - \sqrt{C_1^2 - 4C_2})$$

次式と4次式は存在するが、ここでは紹介するには長すぎる。このため、手で解くことが期待できる高次関係式（$n \geq 3$）は、特性多項式の因数分解が容易なものだけである。

例8.3.13 アルゴリズムを使った解。 Tが$T(k) = 7T(k - 1) - 10T(k - 2)$で定義され、$T(0) = 4$、$T(1) = 17$であるとする。この漸化式をアルゴリズム8.3.12で解くことができる：

(a) 漸化式を「非標準」形式で書いたことに注意。この簡単なステップでのミスを避けるために、この場合、$T(k) - 7T(k - 1) + 10T(k - 2) = 0$ に方程式の並べ替えを考えるかもしれない。したがって、特

(b) 特徴的な根は、$\frac{1}{2}(7 + \sqrt{49 - 40}) = 5$ と $\frac{1}{2}(7 - \sqrt{49 - 40}) = 5$ である。性方程式は $a^2 - 7a + 10 = 0$ となる。

2.これらの根は、特性多項式を$(a - 5)(a - 2)$に因数分解することで、同様に簡単に求めることができる。

(c) 反復関係の一般解は、$T(k) = b_1 2^k + b_2 5^k$である。

(d) $\begin{cases} T(0) = 4 \\ T(1) = 17 \end{cases} \Rightarrow \begin{cases} B_1 2^0 + B_2 5^0 = 4 \\ B_1 2^1 + B_2 5^1 = 17 \end{cases} \Rightarrow \begin{cases} b_1 + b_2 = 4 \\ b_1 + 5b_2 = 17 \end{cases}$

連立方程式は、解 $b_1 = 1$ と $b_2 = 3$ を持つ。したがって、$T(k) = 2^k + 3 - 5^k$ となる。

□

ここで便利なルールを一つ紹介しよう：特性多項式の係数がすべて整数で、定数項がmに等しい場合、可能な有理特性根はmの約数（正負両方）だけである。

コンピューター（あるいは電卓だけかもしれないが）の助けを借りれば、次のようなことができる。

n.特性根の近似は、いくつかのよく知られた方法のいずれかによって得ることができ、そのうちのいくつかは標準的なソフトウェアパッケージの一部となっている。数値近似が実行可能なnの値を特定する一般的なルールはない。得られる精度は，解こうとする関係式に依存する．(このセクションの練習問題17を参照）。

例題8.3.14 3次の漸化式の解法。解く

$S(k)-7S(k-2)+6S(k-3)=0$、ここで$S(0)=8$、$S(1)=6$、$S(2)=22$。

(a) 特性方程式は $a^3 - 7a + 6 = 0$ である。

(b) 有理根で試せるのは、±1、±2、±3、±6だけである。これらをチェックすると、1、2、-3の3つの根が得られる。

(c) 一般解は $S(k) = b_1 1^k + b_2 2^k + b_3 (-3)^k$ である。第一項は単純に b_1 と書くことができる。

(d) $\begin{cases} S(0) = 8 \\ S(1) = 6 \\ S(2) = 22 \end{cases} \Rightarrow \begin{cases} b_1 + b_2 + b_3 = 8 \\ b_1 + 2b_2 - 3b_3 = 6 \\ b_1 + 4b_2 + 9b_3 = 22 \end{cases}$ このシステムを解決することができる。

消去法で $b_1 = 5, b_2 = 2, b_3 = 1$ となる。したがって

$S(k) = 5 + 2 \cdot 2^k + (-3)^k = 5 + 2^{k+1} + (-3)^k$

例題8.3.15 二重特性根を持つ解。$D(k) - 8D(k-1) + 16D(k-2) = 0$、ここで $D(2) = 16$、$D(3) = 80$ を解け。

(a) 特性方程式： $a^2 - 8a + 16 = 0$.

(b) $a^2 - 8a + 16 = (a-4)^2$.したがって、二重特性根4が存在する。

(c) 一般解：$D(k) = (b_{1,0} + b_{1,1} k) 4^k$. (d)

$$\begin{cases} D(2) = 16 \\ D(3) = 80 \end{cases} \Rightarrow \begin{cases} (b_{1,0} + b_{1,1} 2) 4^2 = 16 \\ (b_{1,0} + b_{1,1} 3) 4^3 = 80 \end{cases} \Rightarrow \begin{cases} 16b_{1,0} + 32b_{1,1} = 16 \\ 64b_{1,0} + 192b_{1,1} = 80 \end{cases} \Rightarrow \begin{cases} b_{1,0} = 1 \\ b_{1,1} = \frac{1}{4} \end{cases}$$

したがって、$D(k) = (1/2 + (1/4)k)4^k = (2 + k)4^{k-1}$ となる。

□

8.3.4 非均質有限次線形関係の解法

非同質関係に対する我々のアルゴリズムは、同質の場合ほど完全ではない。これは、異なる右辺（$f(k)$）が特定の解を得るために異なるルールを要求するという事実によるものである。

アルゴリズム8.3.16 非同次有限次線形関係を解くためのアルゴリズム。 *再帰関係* $S(k) + C_1\,S(k-1) + \ldots + C_n\,S(k-n) = f(k)$

(1) 関連する同次関係を書き、その一般解を求める(アルゴリズム8.3.12のステップ(a)から(c))。これを同次解 $S^{(h)}(k)$ と呼ぶ。

(2) 特定の解の形を経験的に推測して、再帰関係の特定の解と呼ばれるもの $S^{(p)}(k)$*を求め始める。右辺の大きなクラスでは、特定の解は* $f(k)$*と同じタイプの関数であることが多いので、これは本当の推測ではありません（表8.3.17参照）。*

(3) ステップ2で推測したものを漸化式に代入する。推測が正しければ、未知の係数を求めることができるはずである。間違った推測をした場合は、この代入の結果から明らかになるはずなので、ステップ2に戻る。

(4) 漸化式の一般解は、同次解と特殊解の和である。条件が与えられない場合は、これで終わりである。n個の初期条件が与えられれば、n個の未知数のn個の連立一次方程式に変換し、系を解いて完全な解を得る。

表8.3.17 与えられた右辺に対する特定の解

右辺、$f(k)$	特殊解の形、$S^{(p)}(k)$
定数、d	定数、q
一次関数、$q_0 + q\,k_1$	一次関数、$d_0 + d\,k_1$
m^{th} 次数の多項式、$q_0 + q_1 k + \ldots + q\,k\,m_m$	m^{th} 次数多項式、$d_0 + d_1 k + \ldots + d\,k_m{}^m$
指数関数、qa^k	指数関数、da^k

例題8.3.18 非一様一次回帰関係の解。 $S(0) = 6$として、$S(k) + 5S(k-1) = 9$を解け。

(a) 関連する同次関係式、$S(k) + 5S(k-1) = 0$ は、特性方程式 $a + 5 = 0$ を持つ。同次解は $S^{(h)}(k) = b(-5)^k$ である。

(b) 右辺は定数なので、特定の解は定数dになると推測される。

(c) $S^{(p)}(k) = d$を再帰関係に代入すると、$d + 5d = 9$、または$6d = 9$となる。したがって、$S^{(p)}(k) = 1.5$となる。

(d) 再帰関係の一般解は、$S(k) = S^{(h)}(k) + S^{(p)}(k) = b(-5)^k + 1.5$ である。初期条件は、bを決定するために解くべき方程式を1つ与えてくれる。$S(0) = 6 \Rightarrow b(-5)^0 + 1.5 = 6 \Rightarrow b + 1.5 = 6$ したが

って、$b = 4.5$ であり、$S(k) = 4.5(-5)^k + 1.5$ である。

例

8.3.19 非同次二次再結合関係の解。 $T(k) - 7T(k-1) + 10T(k-2) = 6 + 8k$、$T(0) = 1$、$T(1) = 2$ を考える。

(a) 例題8.3.13から、$T^{(h)}(k) = b_1 2^k + b_2 5^k$ であることがわかる。注意：$T^{(p)}$ を加えるまでは、$T^{(h)}$ に初期条件を適用しないこと！

(b) 右辺は線形多項式なので、$T^{(p)}$ は線形である、

$T^{(p)}(k) = d_0 + d_1 k.$

(c) 再帰関係に代入すると、以下のようになる： $(d_0 + d_1 k) - 7(d_0 + d_1(k-1)) + 10(d_0 + d_1(k-2)) = 6 + 8k \Rightarrow (4d_0 - 13d_1) + (4d_1)k = 6 + 8k$ Two多項式が等しいのは、その係数が等しい場合だけである。したがって

$$\begin{cases} 4d_0 - 13d_1 = 6 \\ d_1 = 8 \end{cases} \Rightarrow \begin{cases} d_0 = 8 \\ d_1 = 2 \end{cases}$$

(d) 一般解$T(k) = b_1 2^k + b_2 5^k + 8 + 2k$と初期条件を使用する。

$$\begin{cases} T(0) = 1 \\ T(1) = 2 \end{cases} \Rightarrow \begin{cases} b_1 + b_2 + 8 = 1 \\ 2b_1 + 5b_2 + 10 = 2 \end{cases}$$

の式を用いて最終的な解を求める：

$$\Longrightarrow \begin{cases} b_1 + b_2 = -7 \\ 2b_1 + 5b_2 = -8 \end{cases}$$

$$\Longrightarrow \begin{cases} b_1 = -\frac{27}{9} \\ b_2 = 2 \end{cases}$$

したがって、$T(k) = -9 \cdot 2^k + 2 \cdot 5^k + 8 + 2k$ となる。

□

注8.3.20 金利についての簡単なメモ。普通預金口座の残高のように、ある数量が一定の割合で増加する場合、乗数を用いて計算するのが最も簡単である。8%の増加の場合、乗数は次のようになる。1.08は、元の金額Aに0.08Aが加算されるため、新しい残高は$A + 0.08A = (1 + 0.08)A = 1.08A$となる。

別の例として、金利が3.5%の場合、乗数は1.035となる。これは、年利3.5%（しばしば**単利と**呼ばれる）の場合、年末に利息が適用されることを前提としている。利息が毎月適用される場合、各月の長さが同じという単純化されたケースを想定すると、毎月後の乗数は$1 + \frac{0.035}{12} \approx 1.00292$. 1年経過すると、この倍率は12回適用されることになり、これは$1.0029212 \approx 1.03557$を掛けるのと同じである。この1.035から1.03557への増加が**複利の**効果である。

例8.3.21 一種の年金。年利8%の普通預金口座を開設したとする。さらに、口座開設時に1ドルを預金し、毎年預金額を2倍にするつもりだとする。k年後の残高を$B(k)$とする。Bは$B(k)=1.08B(k-1) + 2^k$の関係式で表すことができ、$S(0)=1$とする。毎年預金を2倍にする代わりに、一定額qを預けたとすると、2^kの項はqに置き換えられる。このような定期的な預け入れの連続は、単純年金と呼ばれる。

元の状況に戻る、

(a) $B^{(h)}(k) = b_1(1.08)^k$

(b) $B^{(p)}(k)$は$d2^k$の形でなければならない。

$$d2k = 1.^{08d2k-1} + 2k \Rightarrow (2d)^{2k-1} = 1.^{08d2k-1} + 2 - 2k-1$$

$$\Rightarrow 2d = 1.08d + 2$$

$$\Rightarrow .92d = 2$$

$$\Rightarrow \text{したがって、} B^{(p)}(k) = 2.174 - 2k$$

となる。

(d) $B(0) = 1 \Rightarrow b_1 + 2.174 = 1$

$\Rightarrow b_1 = -1.174$

したがって、$B(k) = -1.174 - 1.^{08k} + 2.174 - {}^{2k}$ となる。

□

例題8.3.22 根のマッチング。 $S(k)$の一般**解を**求めよ。
$3S(k - 1) - 4S(k - 2) = {}^{4k}$.

(a) 関連する同次関係の特性根は-1と4である。したがって、$S^{(h)}(k) = b_1 (-1)^k + b_2 {}^{4k}$。

(b) ${}^{d}4k$ の形の関数は、関連する同次関係を解くので、非同次関係の特定の解ではない。右辺が特性根に等しい底を持つ指数関数を含むときは、特定の解の推測に k を掛けるべきである。$S^{(p)}(k)$ の推測は ^{dk4k} となる。このルールのより完全な説明は、観察8.3.23を参照。

(c) $S(k)$の漸化式に${}^{dk}4k$ を代入する：

$${}^{dk4k} - 3d(k - 1)^{4k-1} - 4d(k - 2)^{4k-2} = {}^{4k}\ 16dk4k-2 - 12d(k - 1)^{4k-2} - 4d(k - 2)^{4k-2} = {}^{4k}$$

左辺の各項の因数は4^k-2である。

$$16dk - 12d(k - 1) - 4d(k - 2) = 4220d = 16 \Rightarrow d = 0.8$$

したがって、$S^{(p)}(k) = 0.8^k 4k$となる。

(d) 再帰関係の一般解は次の通りである。

$$S(k) = b_1 (-1)^k + b_2 {}^{4k} + 0.^{8k4k}$$

観

察8.3.23 右辺の底が特性根に等しいとき。非同次関係式の右辺が底 a を持つ指数を含み、a が多重度 p の特性根でもある場合、表8.3.17 に規定された特定の解の推測に k^p を掛ける。

例8.3.24 マッチングベースの例。

(a) $S(k)\ 9S(k\ 1) + 20S(k\ 2) = 2\cdot 5^k$とすると、特性根は4と5である。5は右辺の底と一致するので、$S^{(p)}(k)$はdk^5kの形になる。

(b) $S(n)\ 6S(n\ 1) + 9S(n\ 2) = 3^n+1$とすると、唯一の特徴根は3であるが、これは二重根（多重度2）である。したがって、特定の解の形は$dn^2\ 3^n$である。

(c) $Q(j) - Q(j - 1) - 12Q(j - 2) = (-3)^j + 6 - {}^{4j}$ とすると、特性根は次のようになる。-3 と 4 である。特定の解の形は $d_1\ \mathbf{j}(-3)^j + d_2\ \mathbf{j} - {}^{4j}$ となる。

(d) $S(k)\ 9S(k\ 1) + 8S(k\ 2) = 9k + 1 = (9k + 1)^{1k}$ とすると、特性根は1と

8である。この場合のように右辺が多項式であれば、指数因子1^kを導入することができる。特定の解は$k\,(d_0 + d_1\,\mathrm{k})$の形になる。

□

このセクションの最後を締めくくる。
特性方程式は複素数根を生じる。有限次線形関係の係数を実数、あるいは整数に限定しても、根が複素数の特性方程式に遭遇することがある。ここでは、我々のアルゴリズムは複素数の特性根でも有効であるが、これらの関係の解を表現するための慣習的な方法が異なることを指摘する。これらの表現を理解するには複素数の知識が必要なので、興味のある読者には、漸化式のより高度な扱い（差分方程式も参照）を参照することを勧める。

8.3.5 エクササイズ

演習グループ以下の漸化式と初期条件のセットを解け.

1. $S(k) - 10S(k-1) + 9S(k-2) = 0, S(0) = 3, S(1) = 11$

2. $S(k) - 9S(k-1) + 18S(k-2) = 0, S(0) = 0, S(1) = 3$

3. $S(k) - 0.25S(k-1) = 0, S(0) = 6$

4. $S(k) - 20S(k-1) + 100S(k-2) = 0, S(0) = 2, S(1) = 50$

5. $S(k) - 2S(k-1) + S(k-2) = 2$、$S(0) = 25$、$S(1) = 16$

6. $S(k) - S(k-1) - 6S(k-2) = \mathbf{-30}, S(0) = 7, S(1) = 6$

7. $S(k) - 5S(k-1) = {}^{5k}, S(0) = 3$

8. $S(k) - 5S(k-1) + 6S(k-2) = 2, S(0) = -1, S(1) = 0$

9. $S(k) - 4S(k-1) + 4S(k-2) = 3k + {}^{2k}, S(0) = 1, S(1) = 1$

10. $S(k) = rS(k-1) + a, S(0) = 0, r, a \geq 0, r \neq 1$

11. $S(k) - 4S(k-1) - 11S(k-2) + 30S(k-3) = 0$、$S(0) = 0$、$S(1) = 0$
$-35, S(2) = -85$

12. 8.2節の練習問題3において、$P(k)$の閉じた形の式を求めよ。

13.

(a) フィボナッチ数列の項の閉形式を求めよ（例題8.1.8参照）。

(b) 数列Cは、C_r = 連続するゼロを持たない長さrのゼロと1の文字列の数 で定義される（例8.2.2(c)）。その漸化式はフィボナッチ数列と同じである。$C_r, r \geq 1$ の閉形式を求めよ。

14. $S(n) = \sum_{j=1}^{n} g(j), n \geq 1$とすると、Sは次の漸化式で記述できる。関係 $S(n) = S(n - 1) + g(n)$.和を使って定義された以下の各列について、閉じた形の式を求めよ：

(a) $S(n) = \sum_{j=1}^{n} j, n \geq 1$

(b) $Q(n) = \sum_{j=1}^{n} j^2, n \geq 1$

(c) $P(n) = \sum_{j=1}^{n} \frac{1}{2}, n \geq 0$

(d) $T(n) = \sum_{j=1}^{n} j^3, n \geq 1$

15. 集合1, 2, ..., n , n **{ } ≥** 1,を2つ
の空でない部分集合に分割することができる。

(a) Dの漸化式を求めよ（ヒント：一次線形関係）。

(b) 再帰関係を解く。

16. 毎年末に一定額を何年間か預ける場合、この一連の支払いは年金と呼ばれる（例8.3.21参照）。

(a) qドルをiの利率で支払う年金の残高または価値を表す閉じた形の式を求めよ。

(b) 金利が5.5%の場合、18年後に100万ドルの価値を持つためには、いくら年金に預ける必要がありますか?

(c) ローンの支払いは年金の一形態であり、当初の価値はある負の金額（ローンの金額）であり、価値がゼロになった時点で年金は終了する。年間5,000ドルを11%の利息で25年間支払う余裕がある場合、いくら借りられるでしょうか?

17. C は小さな正数であるとする。初期条件$B(0) = 1$、$B(1) = 1$の漸化式 $B(k)\ 2B(k\ 1)+\ 1\ C^2\ B(k\ 2) = C^2$ を考える。C が十分小さい場合は，$1\ C^2$ を 1 に，C^2 を 0 に置き換えて近似することを考える．近似の解を B_a a とする。$B(k)$ と $B_a\ (k)$ の閉じた形の式を比較せよ。元の関係式の特性根は近接しており、近似は1つの二重特性根をもたらしたので、両者の形は大きく異なる。関係の特性根が比較的離れていれば、この問題は起こらない。例えば、$S(k) + 1.001S(k - 1) - 2.004002S(k - 2) = 0.0001$ と $S_a\ (k) + S_a\ (k - 1) -$ の一般解を比較する。$2S_a\ (k - 2) = 0.$

8.4 一般的な再帰関係

本節では、定数係数を持つ有限次線形ではない様々な漸化式を検討する。本節の各部分では、具体的な例を考え、解を示し、可能であれば元の関係のより一般的な形を検討する。

8.4.1 最初の基本例

$S(n)\ nS(n\ 1) = 0,\ n\ 1$、初期条件$S(0) = 1$という定係数のない同次一次線形関係を考える。この関係を$n \geqq 1$よく調べると、n番目の項が$(n\ 1)$番目の項のn倍であることがわかる。$S(n)=n!$ は、$n \geqq 1$であれば、この関係の解である、

$$S(n) = n! = n - (n - 1)!\ = n - S(n - 1)$$

加えて、$0!\ =1$なので、初期条件は満たされる。計算の観点からは、

$n!$ の正確な計算には$n - 1$回の乗算が必要であるため、我々の「解」は実際には大した改善にはならないことを指摘しておく。

同様の関係を調べると、$G(k) - 2kG(k-1), k \geq 1$ を$G(0)=1$とすると、Gの値の表は可能な解を示唆している：

k	0	1	2	3	4	5
$G(k)$	1	2	23	26	210	215

$G(k)$の2の指数は、$E(k)=E(k)$の関係に従って成長している。$E(k) = E(k-1) + k$、$E(0) = 0$。したがって、$E(k) = \frac{k(k+1)}{2}$、$G(k) = 2^{k(k+1)/2}$。以下のことに注意されたい。

$G(k)$は$2^{0+1+2+\cdots+k}$と書くこともできる。$k \geq 0$ と書くこともできるが、これは閉じたフォームの表現。

一般に、$P(0) = f(0)$で $n \geq 1$ のとき、$P(n) = f(n-1)$の関係があり、f はすべての $n \geq 0$ に対して定義される関数である。

$$P(n) = \prod_{k=0}^{n} f(k)$$

この$P(n)$の積の形は、nが大きくなるにつれて乗算の数が増えるので、閉じた形の式ではない。したがって、これは本当の解ではない。上の$G(k)$のように、積の形から閉じた形の式が導かれることがよくある。

8.4.2 二分探索アルゴリズムの分析

8.4.2.1

バイナリ検索アルゴリズム（8.1.3節参照）を、0個以上のソートされた項目のリストに対して使用し、各項目に簡単にアクセスできるように、項目が配列に格納されているとする。自然な質問は、"検索を完了するのにかかる時間は？"である。このような質問がなされるとき、私たちが言及する時間は、いわゆる最悪の場合の時間であることが多い。つまり、*n個の項目を*検索するとしたら、検索を完了するのに必要な最長時間はどれくらいか？このような分析を使用するコンピュータに依存しないようにするため、時間は実行されるステップの数を数えることによって測定される。各ステップ（または一連のステップ）には絶対時間、つまり重みが割り当てられる。したがって、答えは秒単位ではなく、絶対時間単位になる。もし2つの異なるアルゴリズムのステップに一貫性のある重みが割り当てられていれば、アルゴリズムの分析を使って相対的な効率を比較することができる。バイナリサーチアルゴリズムの呼び出しで実行しなければならない主要なステップが2つある：

(1) 下位インデックスが上位インデックス以下であれば、リストの中央を探し、そのキーと検索対象の値を比較する。

(2) 最悪の場合、アルゴリズムは前の実行の約半分の大きさのリストで実行されなければならない。ステップ1にかかる時間を1単位とし、$T(n)$をn個のアイテムからなるリストのワーストケース時間とすると

$$T(n) = 1 + T(\lfloor n/2 \rfloor), \qquad n > 0 \qquad (8.4.1)$$

簡単のため、次のように仮定する。

$$T(0) = 0 \qquad (8.4.2)$$

なぜ(8.4.1)で$n/2$が切り捨てられるのか不思議に思うかもしれない。もしnが奇数なら

あるkが0の場合、$n = 2k + 1$となり、リストの真ん中が$(k + 1)$番目の項目となり、リストのどの半分に検索が向けられても、縮小されたリストは$k = n/2$の項目を持つことになる。一方、nが偶数の場合、$k > 0$で$n = 2k$となる。リストの真ん中はk^{th}の項目となり、真ん中より後のk個の項目（$(k + 1)$番目から$(2k)$番目の項目）に誘導された場合、最悪のケースが発生する。この場合も、縮小リストには↪Ps_230A/2」項目がある。

(8.4.1)と(8.4.2)の解答。$T(n)$を決定する最も簡単なケースは、以下の場合である。nは2のべき乗である。$T(2^m), m \geq 0$を繰り返し計算すると、結果は次のようになる。

$$t(1) = 1 + t(0) = 1$$
$$t(2) = 1 + t(1) = 2$$
$$t(4) = 1 + t(2) = 3$$
$$t(8) = 1 + t(4) = 4$$

このパターンから、$T(2^m) = m + 1$であることがわかる。この結果は、リストのサイズを2倍にするたびに、探索時間が1単位だけ増加することを示しているようだ。

nを2進数で表すと、より完全な解が得られる。
各 $n \geq 1$ に対して、次のような非負整数 r が存在する。

$$2^{r-1} \leq n < 2^r \qquad (8.4.3)$$

例えば、$n = 21$の場合、$2^4 \leq 21 < 2^5$であり、したがって$r = 5$である。nが(8.4c)を満たす場合、その2進表現はr桁を必要とする。例えば、$21_{ten} = 10101_{two}$。

一般に、$n = (a\ a_{12} \ldots a_r)_2$。ここで、$a_1 = 1$。この形式では⌊n/2⌋は簡単に説明できる。-1桁の2進数$(a\ a_{12} \ldots a_{r-1})$である。だから

$$\begin{aligned} T(n) &= T(a\ a_{12} \ldots a)_r \\ &= 1 + T(a\ a_{12} \ldots a)_{r-1} \\ &= 1 + (1 + T(a\ a_{12} \ldots a_{r-2})) \\ &= 2 + T(a\ a_{12} \ldots a)_{r-2} \\ &\quad . \\ &\quad . \\ &= (r - 1) + T(a)_1 \\ &= (r - 1) + 1 \qquad T(1) = 1 \\ &= r \end{aligned}$$

この文の正式な帰納的証明は可能である。しかし、ほとんどの読者は上の論証で満足するだろう。懐疑的な人は、帰納的な証明をお願いしたい。

数字の例を見たい人のために、n＝21とする。

$$\begin{aligned} t(21) &= t(10101) \\ &= 1 + T(1010) \\ &= 1 + (1 + T(101)) \end{aligned}$$

$$
\begin{aligned}
&= 1 + (1 + (1 + T(10))) \\
&= 1 + (1 + (1 + (1 + T(1)))) \\
&= 1 + (1 + (1 + (1 + (1 + T(0))))) \\
&= 5
\end{aligned}
$$

我々の一般的な結論は、(8.4.1)と(8.4.2)の解は、以下のようになる。$n \geq 1$、$T(n) = r$、ここで$2^{r-1} \leq n < 2^r$である。

$T(n) = \lfloor \log_2 n \rfloor + 1$である。例えば、$T(21) = \lfloor \log_2 21 \rfloor + 1 = 4 + 1 = 5$。

8.4.2.2 対数の復習

定理5を除いて、我々の底は2である。異なる底（10とe 2.171828は他の一般的なものです）を使用しても、各対数に定数を掛ける効果しかないことがわかります。したがって、使用する底はあまり重要ではない。底2の対数の選択は、私たちが考えている問題にとって便利である。

定義8.4.1 底2対数。正の数の底2対数は指数を表し、任意の正の実数aに対して以下の等価で定義される。

$$\log_2 a = x \quad \Leftrightarrow \quad 2^x = a.$$

◇

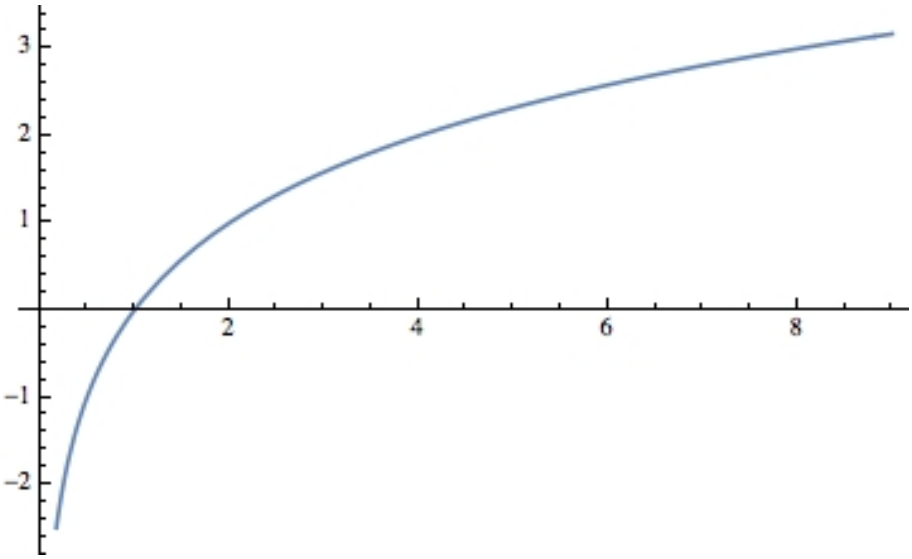

図8.4.2 基数2の対数関数のプロット

例えば、$\log_2 8 = 3$は$2^3 = 8$だからであり、$\log_2 1.414$ は$2^{0.5} = 1.414$だからである。図8.4.2の関数$f(x) = \log_2 x$のグラフは、$a < b$の場合、$\log_2 a < \log_2 b$、つまりxが増加すると$\log_2 x$も増加することを示している。しかし、xを$2^{10}=1024$から$2^{11}=2048$に移すと、$\log_2 x$は10から11にしか増加しない。この対数関数の増加速度の遅さは、覚えておくべき重要なポイントである。n個のデータに対して

$\log_2 n$個の時間単位で実行できるアルゴリズムは、$n/100$や$\sqrt{n}$個の時間単位で実行できるアルゴリズムよりも、かなり大きなデータセットを扱うことができる。グラフの$T(n) = \lfloor \log_2 n \rfloor + 1$も同じ挙動を示すだろう。

この後の対数に関する議論で使用するいくつかの性質は、以下の定

理にまとめられている。

定理8.4.3 対数の基本的性質。 *a* と *b* を

正の実数、rは実数。

$$\log_2 1 = 0 \tag{8.4.4}$$
$$\log_2 ab = \log_2 a + \log_2 b \tag{8.4.5}$$
$$\log_2 \frac{a}{b} = \log_2 a - \log_2 b \tag{8.4.6}$$
$$\log_2 a^r = r \log_2 a \tag{8.4.7}$$
$$2^{\log_2 a} = a \tag{8.4.8}$$

定義8.4.4 対数の底*b*. $b > 0$, $b \neq 1$のとき、$a > 0$のとき、

$$\log_b a = x \Leftrightarrow b^x = a$$

◇

定理8.4.5 異なる基底を持つ対数はどのように関連しているか。 *とする。* $b > 0$, $b \neq 1$. *すべての*$a > 0$について $a = \frac{\log_2 a}{\log_2 b}$. *したがって、*$b > 1$*の場合、ベース*$b$、$\log_b$ *対数は、正のスケーリングファクター*$\log_2 b$ *で割ることにより、底2の対数から計算することができる。*$b < 1$*の場合、このスケーリングファクターは負である。*

証明。(8.4.8)の類推により、$a = b^{\log_b a}$. したがって、この等式の両辺の底2の対数をとれば、次が得られる：

$$\log_2 a = \log_2 b^{\log_b a} \Rightarrow \log_2 a = \log_b a \cdot \log_2 b$$

最後に、最後の式の両辺を$\log_2 b$ で割る。 ■

注8.4.6 $\log_2 \mathbf{10} \approx 3.32192$ and $\log_2 \mathbf{e} \approx 1.4427$.

8.4.2.3

バイナリーサーチアルゴリズムに戻ると、対数関数は増加するため、数の対数を取るときに不等式が維持されるなど、対数の特性を使用してT（n）の最終式を導出することができます。

$$\begin{aligned} T(n) = r &\Leftrightarrow 2^{r-1} \leq n < 2^r \\ &\Leftrightarrow \log_2 2^{r-1} \leq \log_2 n < \log_2 2^r \\ &\Leftrightarrow r - 1 \leq \log_2 n < r \\ &\Leftrightarrow r - 1 = \lfloor \log_2 \mathbf{n} \rfloor \\ &\Leftrightarrow T(n) = r = \lfloor \log_2 \mathbf{n} \rfloor + 1 \end{aligned}$$

対数のこれらの性質をいくつか応用して、T（n）の代替式を得ることができる：

$$\begin{aligned} \lfloor \log_2 \mathbf{n} \rfloor n + 1 &= \lfloor \log_2 n + \mathbf{1} \rfloor \\ &= \lfloor \log_2 n + \log_2 \mathbf{2} \rfloor. \\ &= \lfloor \log_2 \mathbf{2n} \rfloor \end{aligned}$$

バイナリ探索アルゴリズムのステップ1に割り当てられていた時間が変更されても、解の形が大きく変わることは期待できない。$T(0) = c$ で、$T(n) = a + T(\lfloor n/2 \rfloor)$ とすると、$T(n) = c + a \lfloor \log_2 2n \rfloor$ となる。

さらに一般化すると、$T(\lfloor n/2 \rfloor)$ に係数を加えることになる：$T(n) = a + bT(\lfloor n/2 \rfloor)$ with $T(0) = c$, ここで $a, b, c \in R$, and $b \neq 0$ はそれほど単純ではない

を導く。まず、*n*の値が2の累乗であることを考える：

$$T(1) = a + bT(0) = a + bc$$
$$T(2) = a + b(a + bc) = a + ab + cb^2$$
$$T(4) = a + b\,a + ab + cb^2 = a + ab + ab^2 + cb^3$$

$$T(2^r) = a + ab + ab^2 + \cdots + ab^r + cb^{r+1}$$

*n*が2のべき乗でない場合、推論により、(8.4.1)と(8.4.2)で使ったものと同じになる、

$$T(n) = \sum_{k=0} ab^k + cb^{r+1}$$

ここで、$r = \lfloor \log_2 n \rfloor$。

この式の第1項は幾何和であり、閉じた形で書くことができる。その和を*x*とする：

$$x = a + ab + ab^2 + \cdots + ab^r$$
$$bx = ab + ab^2 + \cdots + ab^r + ab^{r+1}$$

*x*の各項に*b*を乗じ、xとbの同じ項を揃えた。

*bx*となる。ここで、2つの方程式を引いてみよう、

$$x - bx = a - ab^{r+1} \Rightarrow x(1 - b) = a\,1 - b^{r+1}$$

したがって、$x = a\frac{b^{r+1} - 1}{b - 1}$。

$T(n)$の閉形式は次のようになる。

$$T(n) = a\frac{b^{r+1} - 1}{b - 1} + cb^{r+1}$$ ここで $r = \lfloor \log_2 n \rfloor$ である。

8.4.3 バブルソートとマージソートの分析

バイナリサーチのような検索アルゴリズムの効率は、検索リストがキー値に従ってソートされ、検索がキー値に基づいて行われるという事実に依存している。リストをソートする方法はいくつかある。その一例がバブルソートである。これはよく使われる "最初のソート・アルゴリズム" なので、ご存知の方も多いだろう。このアルゴリズムの時間分析によると、*n*個のアイテムに対してバブルソートを完了するのに必要なワーストケース時間を$B(n)$とすると、$B(n)=(n - 1) + B(n - 1)$、$B(1)=0$となる。

などは、その2乗項によって制御される。それ以外の項は矮小化される*n*が大きくなるにつれて、nは2nになる。バブルソートの場合、ソートするリストのサイズを2倍にすると、*n*は2nに変わり、n^2 は$4n^2$ になる。したがって、バブルソートに必要な時間は4倍になる。バブルソ

ートに代わるものとして、マージソートがある。以下は、$F = \{r(1), r(2), \ldots r(n)\}$, $n \geq 1$.$n = 1$の場合、リストは些細なことでソートされる。$n \geq 2$なら

(1) Fを$F_1 = \{r(1), \ldots r(\lfloor n/2 \rfloor)\}$と$F_2 = \{r(\lfloor n/2 \rfloor + 1), \ldots r(n)\}$とする。

(2) F_1 と F_2 をマージソートで並べ替える。

(3) ソートされたリストF_1 とF_2 を1つのソートされたリストにマージする。ソートがキー値の降順で行われる場合、F_1 と F_2 の前方から、より高いキー値を選び続け、F の後方に置く。

F_1 は常に$n/2$個の項目を持ち、F_2 は$n/2$個の項目を持つことに注意すること。したがって、nが奇数の場合、F_2 はF_1 よりも1個多い項目を得る。アルゴリズムのステップ1を実行するのに必要な時間は、他のステップに比べて重要でないと仮定する。したがって、このステップには時間値0を割り当てる。ステップ3は、おおよそn回の比較と、F_1 と F_2 からFへのアイテムのn回の移動を必要とする；したがって、その時間は n に比例する。ステップ2には T (⌊n/2B) + T (⌈n/2↪Pe_2309) 時間単位が必要である、

$$T(n) = n + T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) \tag{8.4.9}$$

、初期条件

$$T(1) = 0 \tag{8.4.10}$$

これらの方程式の厳密解の代わりに、$T(n)$の推定値で満足することにする。まず、$n = 2^r$, $r \geq 1$の場合を考える：

$$t(2^1) = t(2) = 2 + t(1) + t(1) = 2 = 1 \cdot 2$$
$$t(2^2) = t(4) = 4 + t(2) + t(2) = 8 = 2 \cdot 4$$
$$t(2^3) = t(8) = 8 + t(4) + t(4) = 24 = 3 \cdot 8$$

$$T(2^r) = r2^r = 2^r \log_2 2^r$$

したがって、nが2のべき乗であれば、$T(n) = n \log_2 n$ である。ここで、ある$r \geq 2$, $2^{r-1} < n \leq 2^r$のとき、$(r-1)2^{r-1} < T(n) < r2^r$である。$n$が$2^{r-1}$から$2^r$に増加するにつれて、$T(n)$は$(r-1)2^{r-1}$から$r2^r$に増加し、$n \log_2 n$よりわずかに大きくなる。この不一致は十分に小さいので、$T_e(n) = n \log_2 n$ は、マージソートを他のアルゴリズムと比較する目的で、(8.4.9)と(8.4.10)の解とみなすことができる。表8.4.7は、$B(n)$と$T_e(n)$をnの値で比較したものである。

表8.4.7 バブルソートとマージソートの時間比較

n	$B(n)$	$T_e(n)$
10	45	34
50	1225	283
100	4950	665
500	124750	4483
1000	499500	9966

8.4.4 紆余曲折

錯乱とは、「固定点」を持たない集合上の順列のことである。これが正式な定義である：

定義8.4.8 派生。 空でない集合Aの錯乱とは、すべての$a \in A$に対して$f(a) = a$となるようなAの並べ替え（すなわち、AからAへの双射）である。 ◇

A=1,2,{・・・,n}とすると、「いくつのデランジャーがあるのか?のメントがあるか? 答えは$n!$$n$は$A$の並べ替えの$(n - 1)$通りであるから、答えは$n$よりかなり小さくなる。

1,2,・・・,**n**}の錯乱の数を$D(n)$とする。n**≧3と**する。1, 2, . .n**}の錯乱、**fを構成するので**あれば、**$f(n) = k \neq n$である、

nの画像はn - 1の異なる方法で選択できる。どのn - 1のどれを選んでも、fの定義は2つの方法のいずれかで完了する。まず、$f(k) = n$とする。 2)の方法でfの定義を完成させることができる。, n n, k .次に、$f(k) = n$を選択することにすると、$D(n$ 1) の各離散は、1, 2, .n 1の各D(n 1) ディレンジメントは、fを定義するために使用することができる。g が1, 2, ... , n 1の錯乱であり、g(p)が1, 2, ... , n 1の錯乱である。$g(p) = k$ となるような1, 2, ... , n 1 の 錯 乱 で あ る 場 合 、 fを以下のように定義する。

$$f(j) = \begin{cases} n & j = p \\ k & j = n \\ g(j) & \text{さもなければ} \end{cases}$$

従って、$f(n) = k$を持つ1, 2, ..., nの脱線は、どちらの方法でも構成できない。

この結果をまとめると、fは、まずnのn - 1個の像のうちの1個を選び、次にfの余りを$D(n-2) + D(n-1)$通りのいずれかで構成することによって決定される。したがって

$$D(n) = (n-1)(D(n-2) + D(n-1)) \tag{8.4.11}$$

初期条件$D(1)=0$および$D(2)=1$とともに、この可変係数を持つ同次の2次線形関係は、次のように完全に定義される。

D.この関係を解析的に求める代わりに、$D(n)$の近似値を経験的に求める。1,2...,nの錯乱は、$n!$ の順列のプールから引かれるので、$n!$, $D(n)$,$\frac{D(n)}{n!}$の値を列挙することで、これらの順列のうち何パーセントが錯乱であるかを確認する。その結果、n

が大きくなっても、$\frac{D(n)}{n!}$はほとんど変化しない。この商を小数点以下8桁まで計算すると、nが12 の場合、$D(n)/n!$ = 0.36787944.$D(n)/n!$が向かっていると思われるこの数の逆数は、8桁で2.7182818である。この数は、数学の中で非常に多くの場所に現れるので、それ自身の名前eを持っている。Dに関する再帰関係の近似解は、$D(n) \approx \frac{n!}{e}$

```
def D(n):
    n <=2の場合:
        n -1を返す
    でなければ .
        return ( n -1)*( D(n -2) +D(n -1))
リスト ( マップ ( ラムダ
    k:[k,D(k) ,(D(k)/ factorial(k)).n( digits =8)], 範囲 (1 ,16)))
```

```
[[1 , 0 , 0.00000000],
 [2 , 1 , 0.50000000],
```

```
[3 , 2 , 0.33333333],
[4 , 9 , 0.37500000],
[5 , 44 ,0.36666667],
[6 , 265 ,0.36805556],
[7 , 1854 ,0.36785714],
[8 , 14833 ,0.36788194],
[9 , 133496 ,0.36787919],
[10 , 1334961 ,0.36787946],
[11 , 14684570 ,0.36787944],
[12 , 176214841 ,0.36787944],
[13 , 2290792932 ,0.36787944],
```

[14,32071101049,0.36787944],
[15,481066515734,0.36787944]]

8.4.5 エクササイズ

1. 以下の漸化式を解け。あなたの解が反復より改善されているかどうかを示しなさい。

(a) $nS(n) - S(n-1) = 0, S(0) = 1.$

(b) $T(k) + 3kT(k-1) = 0, T(0) = 1.$

(c) $U(k) - \frac{k-1}{k}U(k-1) = 0, k \geq 2, U(1) = 1.$

2. もし$n \geq 0$のとき、$\lceil n/2 \rceil + \lfloor n/2 \rfloor = n$.であることを証明せよ。奇数と偶数は別々に)

3. できるだけ完全に解く：

(a) $T(n) = 3 + T(\lfloor n/2 \rfloor), T(0) = 0.$

(b) $T(n) = 1 + \frac{1}{2}T(\lfloor n/2 \rfloor), T(0) = 2.$

(c) $V(n) = 1 + V\lfloor n/8 \rfloor), V(0) = 0.$ (ヒント: nを8進数で書く)。

4. $T(n) = 1 + T(\hookrightarrow \text{Ps\_230An}/2\rfloor)$、$T(0) = 0$、および $2^{r-1} \leq$が成り立つことを帰納法によって証明する。
$n < 2^r, r \geq 1,$ then $T(n) = r.$

ヒント *rに対する*帰納法で証明せよ。

5. 代入$S(n) = T(n+1)/T(n)$ を使って、$T(n)T(n-2) =$を解く。
T（0）＝1、T（1）＝6、T（n）≧0である。

6. $G(n)=T(n)^2$の代入を使って、$T(n)^2 - T(n-1)^2=1$を解く。

$n \geq 1$、$T(0) = 10$。

7. できるだけ完全に解く：

(a) $Q(n) = 1 + Q(\lfloor \sqrt{n} \rfloor), n \geq 2, Q(1) = 0.$

(b) $R(n) = n + R(\lfloor n/2 \rfloor), n \geq 1, R(0) = 0.$

8. マージ・ソート・アルゴリズムのステップ1にかなりの時間がかかったとする。nの値とは無関係に、0.1時間単位でかかると仮定する。

(a) この因子を考慮した$T(n)$の新しい漸化式を書き出しなさい。

(b) $T(2^r)$、$r \geq 0$について解く。

(c) 2の累乗に対する解がすべてのnに対して良い推定値であると仮定して、あなたの結果をテキストの解と比較してください。大きくなるにつれて、本当に大きな違いがあるのだろうか?

8.5 関数の生成

このセクションでは、生成関数のトピックと、他の問題の中でも特に漸化式を解くために生成関数がどのように使われるかを紹介します。生成関数を使う方法は，数列を含む問題を生成関数を含む問題に変換できるという概念に基づいています．新しい問題を解いた後、数列に戻せば元の問題を解くことができる。

このセクションでは、以下の項目を取り上げる：

(1) 生成関数の定義。

(2) 生成関数を使用するために必要なスキルを確認するために、生成関数を使用して漸化式を解く。

(3) 2.で挙げたスキルの紹介および/またはレビュー。

(4) 生成関数のいくつかの応用。

8.5.1 定義

定義8.5.1 数列の生成関数。 項S_0, S_1, S_2, ..., を持つ数列Sの生成関数は、次の無限和である。

$$G(S; z) = \sum_{n=0}^{\infty} S_n z^n = S_0 + S_1 z + S_2 z^2 + S_3 z^3 + \quad (S; z = S + S z + S z + S z + \text{)} \text{。}$$

生成関数のドメインとコドメインは、代数的な演算を行うだけなので、われわれには関係ない。 ◇

例 8.5.2 最初の例

(a) $S_n = 3^n, n \geq 0$とすると、次のようになる。

$$G(S; z) = 1 + 3z + 9z^2 + 27z^3 + \cdots$$
$$= \sum_{n=0}^{\infty} 3^n z^n$$
$$= \sum_{n=0}^{\infty} (3z)^n$$

$G(S;z)$の閉形式は、$G(S;z)$-を観察することで得られる。$3zG(S; z) = 1$。したがって、$G(S; z) = \frac{1}{1}$。

(b) 有限列は生成関数を持つ。例えば、2項係数の列 $\binom{n}{0}$、$\binom{n}{1}$、...、$\binom{n}{n}$、$n \geq 1$は生成関数を持つ。

$$G\left(\binom{n}{0}; z\right) = \binom{n}{0} + \binom{n}{1} z + \cdots + \binom{n}{n} z^n$$
$$= \sum_{k=0}^{\infty} \binom{n}{k} z^k$$
$$= (1 + z)^n$$

を二項式に当てはめる。

(c) $Q(n) = n^2$ の場合、$G(Q; z) = \sum_{n=0}^{\infty} n^2 z^{2n} = \sum_{k=0}^{\infty} k^2 z^{2k}$。和に使われるインデックスには意味がないことに注意。また、$Q(0) = 0$なので、和の下限は1から始まる可能性がある。

□

8.5.2 生成関数を用いた漸近関係の解法

$S(0)=3$、$S(1)=1$として、$S(n) - 2S(n-1) - 3S(n-2) = 0,\ n \geqq 2$を解くことにより、生成関数の使い方を説明する。

(1) 漸化式を生成関数に関する方程式に変換せよ。

$V(n) = S(n)$ とする。 $- 2S(n-1) - 3S(n-2), n \geq 2$、$V(0) = 0$ および

したがって、$V(1) = 0$である、

$$G(V; z) = 0 + 0z + \sum_{n=2}^{\infty} (S(n) - 2S(n-1) - 3S(n-2))z^n = 0$$

(2) 未知数列の生成関数$G(S; z) = \sum_{n=0}^{\infty} S_n z^n$を解く。

$$\begin{aligned} 0 &= \sum_{n=2}^{\infty} (S(n) - 2S(n-1) - 3S(n-2))z^n \\ &= \sum_{n=2}^{\infty} S(n)z^n - 2\sum_{n=2}^{\infty} S(n-1)z^n - 3\sum_{n=2}^{\infty} S(n-2)z^n \end{aligned}$$

上記の3つの合計を精査すると、次のことがわ

かる：(a)

$$\begin{aligned} \sum_{n=2}^{\infty} S_n z^n &= \sum_{n=0}^{\infty} S_n z^n - S(0) - S(1)z \\ &= G(S; z) - 3 - z \end{aligned}$$

$S(0)=3$、$S(1)=1$だからである。

(b)

$$\begin{aligned} \sum_{n=2}^{\infty} S(n-1)z^n &= z\sum_{n=2}^{\infty} S(n)z^n \\ &= z\sum_{n=1}^{\infty} S(n)z^n - S(0) \\ &= z\sum_{n=0}^{\infty} S(n)z^n \\ &= z(G(S; z) - 3) \end{aligned}$$

(c)

$$\begin{aligned} \sum_{n=2}^{\infty} S(n-2)z^n &= z^2\sum_{n=2}^{\infty} S(n-2)z^{n-2} \\ &= z^2 G(S; z) \end{aligned}$$

だから

$$\begin{aligned} &(G(S; z) - 3 - z) - 2z(G(S; z) - 3) - 3z^2 G(S; z) = 0 \\ &\Rightarrow G(S; z) - 2zG(S; z) - 3z^2 G(S; z) = 3 - 5z \\ &\Rightarrow G(S; z) = \frac{3 - 5z}{1 - 2z - 3z^2} \end{aligned}$$

(3) 生成関数がステップ2で得たものである数列を決定する。

この例では、指数数列の閉形式について一般的な事実を一つ知っておく必要がある（証明は後で行う）：

$$T(n) = ba^n,\ n \geq 0 \Leftrightarrow G(T;z) = \frac{b}{1 - az} \tag{8.5.1}$$

さて、この例でSを認識するためには、$G(S;z)$の閉形式を上記の$G(T;z)$のような項の和として書かなければならない。$G(S;z)$の分母は因数分解できることに注意：

$$G(S;z) = \frac{3 - 5z}{1 - 2z - 3z^2} = \frac{3 - 5z}{(1 - 3z)(1 + z)}$$

この$G(S;z)$の最後の式をよく見ると、2つの分数の足し算の結果であることが想像できる、

$$\frac{3 - 5z}{(1 - 3z)(1 + z)} = \frac{A}{1 - 3z} + \frac{B}{1 + z} \tag{8.5.2}$$

ここで、AとBは決定されなければならない2つの実数である。(8.5.2)の右辺から始めると、どのようなA、Bでも和は左辺のようになることは明らかであろう。(8.5.2)を真にするAとBの値を求める過程は、左辺の**部分分数分解**と呼ばれる：

$$\frac{A}{1 - 3z} + \frac{B}{1 + z} = \frac{A(1 + z)}{(1 - 3z)(1 + z)} + \frac{B(1 - 3z)}{(1 - 3z)(1 + z)}$$
$$= \frac{(A + B) + (A - 3B)z}{(1 - 3z)(1 + z)}$$

だから

$$\begin{cases} A + B = 3 \\ A - 3B = -5 \end{cases} \Rightarrow \begin{cases} A = 1 \\ B = 2 \end{cases}$$

そして

$$G(S;z) = \frac{1}{1 - 3z} + \frac{2}{1 + z}$$

$G(S;;z)$の各項に対して(8.5.1)を適用できる：

- $\frac{1}{1-3z}$ は、$S_1(n) = 1 - 3^n = 3^n$の生成関数であり、S_2
- $\frac{2}{1+z}$
  $(n) = 2(-1)^n$の生成関数である。

したがって、$S(n) = 3^n + 2(-1)^n$ となる。

この例から、生成関数を扱うにはいくつかのスキルを習得する必要があることがわかる。以下のことができなければならない：

(a) 合計式とそのインデックスを操作する（ステップ2）。

(b) 代数方程式を解き、部分関数の分解を含む代数式を操作する（ステップ2、3）。

(c) その生成関数で数列を識別する（ステップ1と3）。

他のスキルの習熟は、できるだけ多くの練習をこなし、できるだけ多くの例題を読むことから生まれる。まず、数列の操作と関数生成の操作を確認する。

ーションである。

8.5.3 数列の操作

定義8.5.3 数列に対する演算。 SとTを数列とし、cを実数とする。和$S + T$、スカラー積cS、積ST、畳み込み$S * T$、ポップ演算SŮ（「S pop」と読む）、およびプッシュ演算S↓（「S push」と読む）を、$k \geq 0$の項ごとに次式で定義する。

$$(S + T)(k) = S(k) + T(k) \tag{8.5.3}$$

$$(cS)(k) = cS(k) \tag{8.5.4}$$

$$(S \cdot T)(k) = S(k)T(k) \tag{8.5.5}$$

$$(S * T)(k) = \sum_{j=0}^{k} S(j)T(k-j) \tag{8.5.6}$$

$$(S\_1D6D9)(k) = S(k+1) \tag{8.5.7}$$

$$(S \downarrow)(k) = \begin{cases} 0 & \text{もし} k = 0 \text{なら} \\ S(k-1) & \text{もし} k > 0 \text{なら} \end{cases} \tag{8.5.8}$$

◇

数列を1行と無限の列を持つ行列と想像すれば、S＋TとcSは行列の加算とスカラー倍と全く同じである。他の演算と行列演算の間には明らかな類似点はない。

ポップ操作とプッシュ操作は、図8.5.4aのように、$S(0)$が一番上、$S(1)$がその次.....という数の無限のスタックである数列を想像することで理解できる。数列Sは，図8.5.4bのように，S(1)を一番上，S(2)をその次...とするスタックを残して，スタックからS(0)を「ポッピング」することによって得られる．スタックの先頭にゼロを置くと、図8.5.4cのようなスタックになり、シーケンスSが得られる。ポップ操作とプッシュ操作について説明するときは、これらの図を念頭に置いてください。

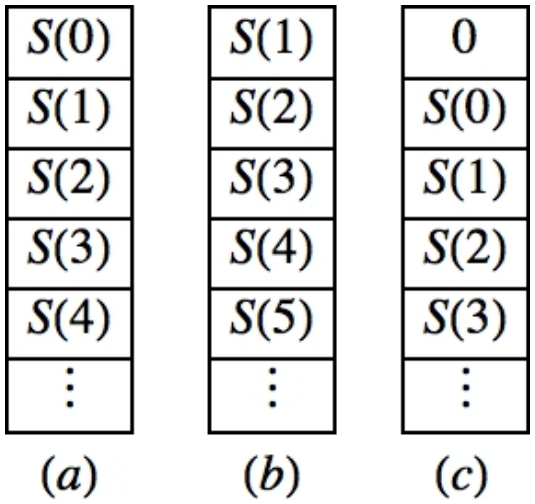

図8.5.4 ポップ操作とプッシュ操作のスタック解釈

例8.5.5 いくつかの数列操作。 $S(n) = n$, $T(n) = n^2$, $U(n) = 2^n$, $R(n) = n^2 n$ とする:

(a) $(S + T)(n) = n + n^2$

(b) $(U + R)(n) = 2^n + n^2 2^n = (1 + n)^2 2^n$

(c) $(2U)(n) = 2 \cdot 2^n = 2^{n+1}$

(d) $\left(\frac{1}{2}R\right)(n) = \frac{1}{2} n^2 2^n = n^2 2^{n-1}$

(e) $(S \cdot T)(n) = n n^2 = n^3$

(f) $$(S * T)(n) = \sum_{j=0}^{n} S(j)T(n-j) = \sum_{j=0}^{n} j(n-j)^2$$
$$= \sum_{j=0}^{n} \left(jn^2 - 2nj^2 + j^3\right)$$
$$= n^2 \sum_{j=0}^{n} j - 2n \sum_{j=0}^{n} j^2 + \sum_{j=0}^{n} j^3$$
$$= n^2 \left(\frac{n(n+1)}{2}\right) - 2n \left(\frac{(2n+1)(n+1)n}{6}\right) + \frac{1}{4} n^2 (n+1)^2$$
$$= \frac{n^2 (n+1)(n-1)}{12}$$

(g) $$(U * U)(n) = \sum_{j=0}^{n} U(j)U(n-j)$$
$$= \sum_{j=0}^{n} 2^j 2^{n-j}$$
$$= (n+1)2^n$$

(h) $(S\_1D6D9)(n) = n + 1$

(i) $(S \downarrow)(n) = \max(0, n - 1)$

(j) $((S \downarrow) \downarrow)(n) = \max(0, n - 2)$

(k) $(U\downarrow)(n) = \begin{cases} 2n-1 & n > 0 \text{ の場合} \\ 0 & n = 0 \text{の場合} \end{cases}$

(l) $((U\downarrow)\hookrightarrow \mathrm{Ll\_1D(D9)})(n) = (U\downarrow)(n+1) = {}^{2n} = U(n)$

(m) $((U\_1D9)\downarrow)(n) = \begin{cases} 0 & \text{もし} n = 0 \text{なら} \\ U(n) & n > 0 \text{の場合} \end{cases}$

□

$(U\downarrow)\psi = (U\psi)\downarrow$**であること**に注意。

定義8.5.6 複数のポップとプッシュ。Sを数列とし、pを1以上の正の整数とするとき、以下を定義する。

$$S\,\psi\,p = (S\,\check{\mathrm{U}}\,(p-1))\,\psi \qquad p \geq 2 \text{ かつ } S\,đ\,1 = S\,đ$$

同様に

$$S\downarrow p = (S\downarrow (p-1))\downarrow \textbf{となる。}\ p \geq 2 \text{ かつ } S\downarrow 1 = S\downarrow\downarrow \textbf{と}\text{する。}$$

◇

一般に、$(S\,\psi(k)) = S(k+p)$である。

$$(S\downarrow p)(k) = \begin{cases} 0 & \text{もし} k < p \text{なら} \\ S(k-p) & k \geq p \text{の場合} \end{cases}$$

8.5.4 生成関数の操作

定義8.5.7 生成関数に対する操作。 $G(z) = {}^{\infty}{}_{=0} a\ z\ {}_k{}^k$ and $H(z) = {}^{\infty}\ {}_{=0}\ b\ z\ {}_k{}^k$ is generating function and c is real number, then sum $G + H$, scalar product cG, product GH, and monomial product z^p G, $\mathbf{p} \geq$ 1 is generating function, where

$$(G+H)(z) = \sum_{k=0} (a_k + b_k)\, z^k \tag{8.5.9}$$

$$(cG)(z) = \sum_{k=0} ca\, z_k{}^k \tag{8.5.10}$$

$$(GH)(z) = \sum_{k=0}^{\infty} cz^k \text{ ここで } c_k = \sum_{j=0}^{k} a\ b_{jk-j} \tag{8.5.11}$$

$$(z^p\ \mathrm{G})\ (z) = z^p \sum_{k=0}^{\infty} a\ z_k{}^k = \sum_{k=0}^{\infty} a\ z_k{}^{k+p} = \sum_{n=p} a\ z_{n-p}{}^n \tag{8.5.12}$$

最後の和は、前の和のkにn-pを代入して得られる。 ◇

例8.5.8 生成関数に対するいくつかの操作。 $D(z) = \Sigma_{k\infty=0}\, kz^k$ and $H(z) = \Sigma_{k\infty=0}\, 2kz^k$ then

$$(D+H)(z) = \sum_{k=0} \left(k + 2^k\right) z$$

∞ ∞

$$(2H)(z) = \sum_{k=0} 2 \cdot 2kz^k = \sum_{k=0} 2k+1z^k$$

$$(zD)(z) = z\sum_{k=0}^{\infty} kz^k = \sum_{k=0}^{\infty} kz^{k+1}$$

$$= \sum_{k=1}^{\infty} (k-1)z^k = D(z) - \sum_{k=1}^{\infty} z^k$$

$$(DH)(z) = \sum_{k=0}^{\infty} \sum_{j=0}^{k} 2^{k-j} z^k$$

$$(HH)(z) = \sum_{k=0}^{\infty} \sum_{j=0}^{k} 2^{k-j} z^k = \sum_{k=0}^{\infty} (k+1)2^k z^k$$

注：例5より、$D(z) = G(S; z)$、$H(z) = G(U; z)$。 □

ここで、数列の操作と生成関数の関係を確立する。SとTを数列とし、cを実数とする。

$$G(S + T; z) = G(S; z) + G(T; z) \quad (8.5.13)$$

$$G(cS; z) = cG(S; z) \quad (8.5.14)$$

$$G(S * T; z) = G(S; z)G(T; z) \quad (8.5.15)$$

$$G(S_{1D6D9}; z) = (G(S; z) - S(0))/z \quad (8.5.16)$$

$$G(S\downarrow; z) = zG(S; z) \quad (8.5.17)$$

つまり、(8.5.13)は、2つの数列の和の生成関数は、それらの数列の生成関数の和に等しいと言っている。他の4つの恒等式を自分の言葉で書き出してみよう。これまでの例から、最後の2つを除いて、これらの恒等式は自明であろう。(8.5.16)は次の定理で証明し、(8.5.17)の証明は読者に任せる。つまり、$G(S - T; z)$は簡約化できない。

定理8.5.9 PopとPushに関連する生成関数。 *もし* $p > 1$,

(a) $$G(S\psi p; z) = G(S; z) - \sum_{k=0}^{p-1} S(k) z^k / z^k$$

(b) $G(S\downarrow p; z) = z^p G(S; z)$。

証明(a)を帰納法で証明し、(b)の証明は読者に委ねる。

基本：

$$G(S\psi 1) = \sum_{k=0}^{\infty} S(k+1)z^k$$

$$= \sum_{k=1}^{\infty} S(k)z^{k-1}$$

$$= \sum_{k=1}^{\infty} S(k)z^k / z$$

$$= AS(0) + \sum_{k=1}^{\infty} S(k)z^k - S(0) / z$$

$$= (G(S; z) - S(0))/z$$

従って、(a)の部分はp=1の場合に正しい。

帰納法ある$p \geq 1$において、(a)の部分が真であるとする：

$$G(S^{(p+1)}; z) = G((S^{p})'; z)$$
$$= (G(S^{p}; z) - \sum_{k=0}^{p-1} S(k) z^{k}) / z \text{ である。}$$
$$= \frac{\frac{G(S;z)}{z^{p}} - S(p)}{z}$$

である。ここで、上の最後の式の$S(p)$を$(S(p)z^{p}) / z^{p}$と書き、有限和に収まるようにする：

$$G(S^{(p+1)}; z) = \frac{G(S; z) - \sum_{k=0}^{p} S(k) z^{k}}{z^{p+1}}$$

従って、$p+1$についてこの文は正しい。 ■

8.5.5 生成関数の閉形式

生成関数を閉じた形で表現するために使われる最も基本的なツールは、幾何級数の閉じた形の式であり、これは $a + ar + ar^2 +$ という形の式である。これは終端することも、無限に拡張することもできる。

有限幾何学シリーズ：

$$a + ar + ar^2 + \cdots + ar^n = \frac{1 - r^{n+1}}{1 - r} \qquad (8.5.18)$$

無限の幾何学シリーズ：

$$a + ar + ar^2 + \cdots = \frac{a}{1 - r} \qquad (8.5.19)$$

制限：aおよびrは定数を表し、2つの方程式の右辺は以下の条件下で適用される：

(1) 有限の場合、rは1であってはならない。$a + ar + ar^n = (n + 1)a$ if $r = 1$.

(2) 無限の場合、rの絶対値は1より小さくなければならない。

生成関数では、このような制限はない。$S(n) = a + ar + ar^n$, $n > 0$ならば、$S(n) = rS(n-1) + a$ であることに注意すれば、(8.5.18)を導くことができる（セクション8.3の練習問題10参照）。セクション8.4では別の導出法を用いた。同じ手順で(8.5.19)を導出する。$x = a + ar + ar^2 + \cdots$とする。すると

$$rx = ar + ar^2 + \cdots = x - a \Rightarrow x - rx = a \Rightarrow x = \frac{a}{1-r}$$

例 8.5.10 幾何和を含む関数の生成。

(a) $S(n)=9\cdot 5^n$、$n \geq 0$のとき、$G(S; z)$は$a=9$の無限等比級数である。
したがって、$G(S; z) = \frac{9}{1-5z}$。

(b) $T(n) = 4$, $n \geq 0$ ならば、$G(T; z) = 4/(1 - z)$となる。

(c) $U(n) = 3(-1)^n$ ならば、$G(U; z) = 3/(1 + z)$となる。

(d) $C(n) = S(n) + T(n) + U(n) = 9 - 5^n + 4 + 3(-1)^n$ とする。とすると

$$G(C; z) = G(S; z) + G(T; z) + G(U; z)$$
$$= \frac{9}{1 - 5z} + \frac{4}{1 - z} + \frac{3}{1 + z}$$
$$= \frac{14z^2 + 34z - 16}{-5z^3 - z^2 - 5z + 1}$$

$G(C;z)$の最後の形と、先ほどの3つの分数の和のどちらかを選べるのであれば、3つの関数の和のままにしておく方がいいだろう。先の例で見たように、最後の式のような分数の部分分数分解は、生成に多少の労力を要する。

(e) $G(Q;z)=34/(2-3z)$であれば、Qは以下のように決定される。分子と分母を1/2で割って1を得る $\frac{17}{1 - \frac{3}{2}z}$.我々はこのことを認識している。

2したがって$Q(n)=17(3/2)^n$となる。

(f) $G(A; z) = (1 + z)^3$ とすると、$(1 + z)^3$ を $1 + 3z + 3z^2 + z^3$ に展開する。したがって、$A(0) = 1$、$A(1) = 3$ $A(2) = 3$、$A(3) = 1$となり、高次の項がないので、$A(k) = 0, k \geq 4$ となる。をより簡潔に表現すると

$$A(k) = \binom{3}{k}$$

□

表8.5.11にいくつかの一般的な数列の生成関数の閉じた形の式を示す。

表8.5.11 生成関数の閉形式

シーケンス	生成機能
$S(k) = ba^k$	$G(S; z) = \frac{b}{1 - az}$
$S(k) = k$	$G(S; z) = \frac{z}{(1 - z)^2}$
$S(k) = bka^k$	$G(S; z) = \frac{abz}{(1 - az)^2}$
$S(k) = \frac{1}{k!}$	$G(S; z) = e^z$
$S(k) = \binom{n}{k}$ $0 \leq k \leq n$ 0 $k > n$	$G(S; z) = (1 + z)^n$

例題8.5.12 もう一つの完全解。 $S(k) + 3S(k - 1) - 4S(k - 2) = 0, k \geq 2, S(0) = 3, S(1) = 2$.この解は、このセクションの前半で使用したのと同じ手順を使用して導出されるが、1つのバリエーションがある。

(1) 生成関数に関する式に変換する。ここで、$V(n) = S(n + 2) + 3S(n + 1) - 4S(n)$ とすると、Vは零列であり、零の生成関数を持つ。さらに、$V = S$ Ψ $2 + 3(S$ Ŭ$) - 4$Sである。

だから

$$\begin{aligned} 0 &= G(V\,;z) \\ &= G(S\ddot{U}2;z) + 3G(S\check{U};z) - 4G(S;z) \\ &= \frac{G(S;z) - S(0) - S(1)z}{z2} + 4\,\frac{(G(S;z) - S(0))}{z} - 4G(S;z) \end{aligned}$$

(2) ここで、$G(S;z)$について以下の方程式を解きたい：

$$\frac{G(S;z) - S(0) - S(1)z}{z2} + 4\,\frac{(G(S;z) - S(0))}{z} - 4G(S;z) = 0$$

z^2 を掛ける：

$$G(S;z) - 3 + 2z + 3z(G(S;z) - 3) - 4z^2\,G(S;z) = 0$$

$G(S;z)$を片辺に含むすべての項を展開して集める： $(\,;\,) + 3 \qquad (\,;\,)$

$$G\,S\,z \qquad zG\,S\,z - 4\,z^2\,G(\,S)z = 3 + 7 \qquad z$$

$$1 + 3z - 4z^2 \quad G(S;z) = 3 + 7z$$

だから

$$(\,;\,) = \frac{3 + 7z}{G\,S\,z} \qquad \frac{}{1 + 3z - 4z2}$$

(3) その生成関数からSを決定する。$1 + 3z \qquad - 4z^2 = (1 + 4z)(1 - z)$

したがって、$G(S;z)$の部分分数分解は次のようになる：

$$\frac{A}{1 + 4z} + \frac{B}{1 - z} = \frac{Az - A - 4Bz - B}{(z - 1)(4z + 1)} = \frac{(A + B) + (4B - A)z}{(z - 1)(4z + 1)}$$

したがって、$A+B = 3$、$4B\,A = 7$。この一連の方程式の解は、$A = 1$、$B = 2$である。$G(S;z) = \frac{1}{1+4z} + \frac{2}{1-z}$.

$\frac{1}{1+4z}$は、$S_1(n) = (-4)^n$ の生成関数である。

$\frac{2}{1-z}$ はSの生成関数である$_2(n) = 2(1)^n = 2$

結論として、$G(S;z) = G(S_1\,;z) + G(S_2\,;z)$ なので、$S(n) = 2 + (-4)^n$ となる。

□

例8.5.13 カウントへの応用。$A = a, b, c, d, e$ とし、Aの各要素を0回以上使って作ることができる長さ0以上の文字列の集合を A^* とする。一般化された積の法則により、長さnを持つこのような文字列は5^n個存在する。$\in 0, X_n$ は、長さnの文字列の集合であり、a'sとb'sのすべてがc's、d's、e'sのすべてに先行するという性質を持つとする。したがって $aaabde\ {}_{66}R(n) = X_n$ とする。Rの閉形式は、Rを2つの数列の畳み込みとして認識することで得られる。我々の主張を説明するために、$R(6)$の計算を考えてみよう。

文字列がX_6 に属する場合、a, bのk文字で始まり、c, d, eのk文字が続く。k文字が続く。長さkのaとbの文字列の数を$S(k)$、長

さkのc,d,eの文字列の数を$T(k)$とする。一般化された積の法則により、$S(k)=^{2}$k、$T(k)=^{3}$kとなる。X_6 の文字列のうち、2つのaとbで始まり、c's, d's, e'sで終わるものがある。このような文字列は$S(2)T(4)$個ある。法則によれば

足し算の、

$$|x_6| = r(6) = s(0)t(6) + s(1)t(5) + \cdots + s(5)t(1) + s(6)t(0)$$

Rの第6項は、SとTの畳み込みの第6項であることに注意されたい。この一般的な状況をしばらく考えてみれば、$R = S * T$ であることは明らかであろう。さて、我々の行動は次のようになる：

(a) SとTの生成関数を決定する、

(b) $G(S;z)$と$G(T;z)$を掛け合わせ、$G(S * T;z) = G(R;z)$を得る。

(c) $G(R;z)$の基底でRを決定する。

(a) $G(S;z) = \sum_{k=0}^{\infty} 2kz^k = \frac{1}{(1-2z)}$ および$G(T;z) = \sum_{k=0}^{\infty} \frac{3kz^k}{(-3z)} = \frac{1}{(1-3z)}$

(b) $G(R;z) = G(S;z)G(T;z) = \frac{1}{(1-2z)(1-3z)}$ (1)

(c) $G(R;z)$からRを認識するためには、部分分数分解をしなければならない：

$$\frac{1}{(1-2z)(1-3z)} = \frac{A}{1-2z} + \frac{B}{1-3z} = \frac{-3Az + A - 2Bz + B}{(2z-1)(3z-1)} = \frac{(A+B) + (-3A-2B)z}{(2z-1)(3z-1)}$$

したがって、$A + B = 1$、$-3A - 2B = 0$ となる。
$G(R;z) = \frac{1}{(1-2z)} + \frac{2}{(1-3z)}$ である。
の生成関数の和である。

$-2(2)^k$ および$3(3)^k$, R_k

$-2(2)^k + 3(3)^k = 3^{k+1} - 2^{k+1}$

例えば、$R(6) = 3^7 - 2^7 = 2187 - 128 = 2059$である。当然、これは$(S * T)(6)$から得られる和に等しい。この数を考慮すると、制限のない長さ6の文字列の総数は$5^6 = 15625$である、

そ $\frac{2059}{15625} \approx 0.131776$.したがって、文字列の約13パーセントが
の長さ6は問題の条件を満たす。
し
て

□

8.5.6 専門家のためのおまけ

このセクションの残りの部分は、組合せ論のコースを取ったことのある読者、あるいは取るつもりのある読者を対象としている。一般的なコースに組み込むことは勧めない．前の例で使われた方法は非常に強力なものであり，組合せ論の多くの問題を解くのに使うことができる．このセクションの最後に，この方法で解ける問題の一般的な説明と，いくつかの例を示す．

$P_1, P_2, \ldots, P_m$ が、それぞれ明確に定義された結果をもたらす、取らなければならない m 個の行動である状況を考える。各 $k = 1, 2, \ldots, m$ に対して、P_k の可能な結果の集合を X_k と定義する。各結果は何らかの方法で定量化でき、X_k の要素の定量化は関数 $Q_k : X_k \to \{0, 1, 2, \}$ したが

って定義されると仮定する、

各結果は、それに関連する非負の整数を持つ。最後に、頻度関数 F_k : $\{\ \}\to\{\ \}$ $0, 1, 2, \ldots$ を定義する。 $0, 1, 2, F_k(n)$ は、次の数である。

Xの要素の$_k$ 、nの量化を持っている。

さて、これらの前提に基づき、解決可能な問題を定義することができる。プロセスPを、上記のような一連の行動$P_1, P_2, , P_m$ と定義する、
そして、Pの結果が、$X_1 \times X_2 \times \times X_m$ の要素である場合、次のようになる。

によって定
量化される

$$Q(a_1, a_2, \ldots, a) = \sum_{k=1}^{m} Q(a)_m \quad Q_k(a)_k$$

。

とすれば、Pの周波数関数Fは、Pの周波数関数$F_1, P_2, \ldots, P_m$ の畳み込みであり、周波数関数$F_1, F_2, \ldots, F_m$ の生成関数の積に等しい生成関数を持つ、

$$G(F; z) = G(F_1; z)\, G(F_2; z)\, (F_m; z)$$

例題8.5.14 二つのサイコロを振る。サイコロを2回振り、出た目の数を合計するとする。ダイスの面は1から6の整数を表しているので、合計は2から6の間でなければならない。

12.これらの和はいくつの方法で得られるだろうか? 明らかに、2は1つの方法でしか得られない。和が3になる数列は2つある: 1-2と2-1。2から12までの数が得られる頻度をすべて求めるために、次のように状況を設定する。$j = 1, 2$; P_j は，j^{th} 回のダイスの出目である．$X_j = 1, 2, \ldots, 6$ とし、$Q_j : X_j\ 0, 1, 2, 3, \ldots$ は $Q_j(x) = x$ で定義される。各数字はダイスにちょうど一度だけ現れるので、度数関数は $1 \le k \le 6$ のとき $F_j(k) = 1$ となり、それ以外のとき $F_j(k) = 0$ となる。つまり、$Q(a_1, a_2) = Q_1(a_1) + Q_2(a_2)$ である。ダイスを2回振る頻度関数の生成関数は次のようになる。

$$\begin{aligned} G(F; z) &= G(F_1; z)\, G(F_2; z) \\ &= (z^6 + z^5 + z^4 + z^3 + z^2 + z)^2 \\ &= z^{12} + 2z^{11} + 3z^{10} + 4z^9 + 5z^8 + 6z^7 + 5z^6 + 4z^5 + 3z^4 + 2z^3 + z^2 \end{aligned}$$

さて、$F(k)$を求めるには、z^k の係数を読めばよい。例えば、z^5 の係数は4なので、合計5を振るには4通りある。

この方法を適用するために重要なステップは、大きなプロセスを適切な方法で分解し、我々が説明した一般的な状況に適合させることである。

□

例8.5.15 委員会の配分ある組織が地理的にA、B、Cの3つのセクションに分かれているとする。どのセクションからも5人以上の委員が出ないように11人の委員からなる執行委員会を選出しなければならず、セクションA、B、Cの委員はそれぞれ最低3人、2人、2人でなければならないとする。各セクションからの委員の数だけを見ると、委員会は何通り構成できるでしょうか。有効な委員会の一例としては、Aが4名、Bが4名、Cが3名となります。

P_A を、セクション A から何人の委員 (誰ではない) が委員を務めるかを決定する行為とする。$X_A = 3, 4, 5$, $Q_A(k) \in k$. 度数関数 F_A は、k X_k のと

き $F_A(k) = 1$、それ以外のとき $F_A(k) = 0$ で定義される。$G(F_A ; z)$ は、$z^3 + z^4 + z^5$ となる。同様に、$G(F_B ; z) = z^2 + z^3 + z^4 + z^5 = G(F_C ; z)$となる。委員会のメンバーは11人でなければならないので、答えは$G(F_A ; z)$ $G(F_B ; z)$ $G(F_C ; z)$におけるz^{11} の係数となり、10となる。

```
ラテックス表示 var('z')
拡大 (( z^3+ z^4+ z^5)*( z^2+ z^3+ z ^4 + z^5) ^2)
```

```
z^15 + 3* z^14 + 6* z^13 + 9* z^12 + 10* z^11 + 9* z^10 + 6* z^9 + 3* z^8 +
    z^7
```

□

8.5.7 エクササイズ

1. 次の生成関数を持つ数列は?

(a) 1

(b) $\frac{10}{2 - z}$

(c) $1 + z$

(d) $\frac{3}{1 + 2z} + \frac{3}{1 - 3z}$

2. 次の生成関数を持つ数列は?

(a) $\frac{1}{1 + z}$

(b) $\frac{1}{4 - 3z}$

(c) $\frac{2}{1 - z} + \frac{1}{1 + z}$

(d) $\frac{z + 2}{z + 3}$

3. 次の数列の生成関数の閉形式を求めよ:

(a) $V(n) = 9^n$

(b) $P(k) - 6P(k - 1) + 5P(k - 2) = 0$ である。 $2, P(0) = 2, P(1) = 2$ である。

(c) フィボナッチ数列 $F(k + 2) = F(k + 1) + F(k), k \geq 0$, ただし $f(0) = f(1) = 1$.

4. 次の数列の生成関数の閉形式を求めよ:

(a) $W(n) = \binom{5}{n} 2^n$ for $0 \leq n \leq 5$ and $W(n) = 0$ for $n > 5$.

(b) Q、ここで $Q(k) + Q(k - 1) - 42Q(k - 2) = 0$ である。 $2, Q(0) = 2, Q(1) = 2$ である。

(c) ここで、$G(k + 3) = G(k + 2) + G(k + 1) + G(k)$ for $k \geq 0$ である。 $g(0) = g(1) = g(2) = 1$である。

5. 次の各式について、部分分数分解を求め、その式を生成関数とする数列を特定せよ。

(a) $\frac{5 + 2z}{1 - 4z - 12z^2}$

(b) $\frac{32 - 22z}{2 - 3z + z^2}$

(c) $\frac{6 - 29z}{1 - 11z + 30z^2}$

6. 部分分数の分解を求め、次の式を持つ数列を特定せよ:

(a) $\frac{1}{1 - 9z^2}$

(b) $\frac{1 + 3z}{16 - 8z + z^2}$

(c) $\frac{2z}{1 - 6z - 7z^2}$

7. $S(k)=k$、$T(k)=10k$とすると、以下の各列の生成関数のk^{th} 項は何であるか:

(a) $S + T$

(b) $S * T$

(c) $\mathbf{S} * T$

(d) $S \psi * S \psi$! "

8. $P(k) = 10^k$、$Q(k) = k!(k^k)$としたとき、次の各列の生成関数のk^{th} 項は何か:

(a) $\mathbf{P} * P$

(b) $P + P \psi$

(c) $\mathbf{P} * Q$

(d) $\mathbf{Q} * Q$

9. ゲームはダイスを5回振って行う。k（th）の出目で、kより大きい数字が出た場合、得点に1点が加算される。そうでない場合、得点は0点となる。例えば、2,3,4,1,2 と出れば合計3点、1,2,3,4,5と出れば0点となる。6^5＝7776通りの出目のうち、0点、1点、.....5点となる出目はいくつあるか。5点?

10. ダイスを10回続けて振り、出た目のマスを記録するとする。出た目の2乗の和が40になるのは何通りあるだろうか。最も一般的な結果は?