CSE 1400 Applied Discrete Mathematics
Sequences

Department of Computer Sciences

College of Engineering

Florida Tech

Spring 2011

Abstract

1 Sequences

Sequences are ordered lists of values. Ordinal numbers: first, second, third, fourth, . . . , specify the positions of these values. One famous sequence is the Fibonacci sequence.

$$\langle 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, \ldots \rangle$$

The first, second, third, fourth, and fifth Fibonacci numbers are 0, 1, 1, 2, and 3. The values in the Fibonacci sequence can be named f_k where the subscript k denotes the position of the value. In computing, it is customary to start the subscript k at 0 so that the first, second, third, fourth, and fifth Fibonacci numbers are named f_0, f_1, f_2, f_3, and f_4.

Let

$$\vec{S} = \langle s_0, s_1, s_2, s_3, s_4, \ldots \rangle$$

be a sequence. The subscripted names s_0, s_1, s_2, s_3, s_4, . . . , are called TERMS and they refer to values in the first, second, third, fourth, fifth, etc., positions of the sequence. In computing theory, sequences

contain a countably infinite number of terms. In computing practice, sequences have a finite number of terms. For instance, the sequence of primes less than 30 is

$$\vec{P}{<}\,30 = \langle 2, 3, 5, 7, 11, 13, 17, 19, 23, 29\rangle$$

There is a a sequence with no terms at all, called the empty sequence. I'll use σ to denote the empty sequence, so that

$$\sigma = \vec{}$$

.

1.1 *Operations on Sequences*

There are some basic operations that can be performed on sequences.

empty() maps a sequence to `True` or `False`.

$$null(\vec{S}) = \begin{cases} \texttt{True} & \text{if } \vec{S} \text{ is empty} \\ \texttt{False} & \text{if } \vec{S} \text{ contains at least one term} \end{cases}$$

length() maps a finite sequence to the number of terms it contains.

$$length(\vec{S}) = n$$

head() maps a non-empty sequence to its first element

$$head(\vec{S}) = s_0$$

last() maps a non-empty, finite sequence to its last element

$$last(\vec{S}) = s_{n-1}$$

tail() removes the first term from a sequence

$$tail(\vec{S}) = \langle s_1, s_2, s_3, \ldots, s_{n-1}\rangle$$

concat(,) concatenates two finite sequences to form a single sequence

$$concat(\vec{S}, \vec{T}) = \langle s_1, s_2, \ldots, s_{n-1}, t_0, t_1, \ldots, t_m\rangle$$

map(,) applies a function f to each term of a sequence

$$map(f, \vec{S}) = \langle f(s_1), f(s_2), \ldots, f(_{n-1})\rangle$$

1.2 *Useful Sequences*

Here is a list of useful sequences.

- The Alice sequence

  $$\vec{A} = \langle 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, \ldots \rangle$$

  keeps a "tally."

- The Gauss sequence

  $$\vec{G} = \langle 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 \ldots \rangle$$

  computes the sum of a "tally."

- The sequence of TRIANGULAR NUMBERS

  $$\vec{T} = \langle 0, 0, 1, 3, 6, 10, 15, 21, 28, 36, \ldots \rangle$$

  computes the number of edges in a complete graph.

- The doubling sequence

  $$\vec{D} = \langle 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, \ldots \rangle$$

  computes a common growth model.

- The Mersenne sequence

  $$\vec{M} = \langle 0, 1, 3, 7, 15, 31, 63, 127, 255, 511 \ldots \rangle$$

  computes the sum of the doubling growth model.

- The FACTORIAL sequence

  $$\vec{n!} = \langle 1, 1, 2, 6, 24, 120, 720, 5040, 40320, 362880, \ldots \rangle$$

  computes the ways to order things.

- The FERMAT sequence

  $$\vec{F} = \langle 3, 5, 17, 257, 65537, 4294967297, \ldots \rangle$$

  computes the ways to order things.

- The Fibonacci sequence

  $$\vec{\Phi} = \langle 0, 1, 1, 2, 3, 5, 8, 13, 21, 34 \ldots \rangle$$

  computes an interesting growth rate based on the golden ratio φ.

- The sequence of PRIME numbers

$$\vec{P} = \langle 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, \ldots \rangle$$

are fundamental to the understanding of number theory.

- The sequence of COMPOSITE numbers

$$\vec{C} = \langle 4, 6, 8, 9, 10, 12, 14, 15, 16, 18, \ldots \rangle$$

are likewise fundamental in number theory.

- The DIVISOR sequence

$$\vec{\Delta} = \langle 1, 2, 2, 3, 2, 4, 2, 4, 3, 4, \ldots \rangle$$

counts the divisors of $\langle 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 \rangle$.

- The relations sequence

$$\vec{R} = \langle 1, 2, 16, 512, 65536, 33554432, \ldots \rangle$$

counts the number of relations on an n element set for $n = 0, 1, 2, 4, \ldots$.

- The sequence of BINOMIAL COEFFICIENT

$$\vec{B} = \langle 1, 1, 1, 1, 2, 1, 1, 3, 3, 1, \ldots \rangle$$

counts the number of m-elements subsets of an n-element set as m ranges from 0 to n and n goes from 0 onward.

- The CATALAN sequence

$$\vec{\chi} = \langle 1, 1, 2, 5, 14, 42, 132, 439, \ldots \rangle$$

counts the number of binary trees with n nodes for $n = 0, 1, 2, 3, 4, \ldots$

- STIRLING NUMBERS OF THE FIRST KIND

$$\vec{S}_1 = \langle 1, 0, 1, 0, 1, 1, 0, 2, 3, 1, 0, 6, 11, 6, 1, \ldots \rangle$$

counts the number of m-cycle permutations of an n-element set as m ranges from 0 to n and n goes from 0 onward.

- STIRLING NUMBERS OF THE SECOND KIND

$$\vec{S}_2 = \langle 1, 0, 1, 0, 1, 1, 0, 1, 3, 1, 0, 1, 7, 6, 1, \ldots \rangle$$

counts the number of m-subset partitions of an n-element set as m ranges from 0 to n and n goes from 0 onward.

- The sequence of HARMONIC NUMBERS

$$\vec{H} = \left\langle 0, \frac{1}{1}, \frac{3}{2}, \frac{11}{6}, \frac{25}{12}, \frac{137}{60}, \ldots \right\rangle$$

1.3 Growth Rates

The above examples show that values of terms in many sequence increase as the position in the sequence increases. The above examples also show that values of terms in some sequence oscillate. In any case, we can often roughly describe a sequence by saying its order of growth is bounded by some function.

- A sequence of constants $\langle c, c, c, c, \ldots \rangle$ is said to be of order 1, or big-Oh of 1, written $O(1)$.
- A arithmetic sequence $\langle b, a+b, 2a+b, 3a+b, \ldots \rangle$ is said to be of order n, or big-Oh of n, written $O(n)$. This is a LINEAR rate of growth.
- The triangular $\langle 0, 0, 1, 3, 6, \ldots, n(n-1)/2, \ldots \rangle$ is said to be of order n^2, or big-Oh of n squared, written $O(n^2)$. This is a QUADRATIC rate of growth.
- A geometric sequence $\langle 1, a, a^2, a^3\, a^4, \ldots \rangle$ is said to be of order a to the n, or big-Oh of a to the n, written $O(a^n)$. This is an EXPONENTIAL rate of growth.
- The doubling, Mersenne, and Fibonacci sequence grow exponentially
  - The doubling and Mersenne sequences grow like 2^n

    $$d_n = 2^n \qquad m_n = 2^n - 1$$

  - The Fibonacci sequence grows like φ^n, where φ is the golden ratio, which is equal to $\frac{1+\sqrt{5}}{2}$.

    $$f_n = O(\varphi^n) = O\left(\left(\frac{1+\sqrt{5}}{2}\right)^n\right)$$

- The factorial sequence grows like the function

  $$\sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$

- The sequence of Harmonic numbers grow like $\ln(n)$

Polynomial or slower growth is called TRACTABLE, which loosely means that an algorithm with polynomial time complexity can be solves problems in a reasonable amount of time. Tractable problems are considered easy. Exponential grow is called INTRACTABLE, which loosely means that an algorithm with exponential time complexity takes an unacceptable amount of time solve problems. It is hard to solve with an intractable problems.

1.4 *Defined by Recurrence Equations*

Terms in many sequences can be computed by using a recurrence equation. Let

$$\vec{S} = \langle s_0, s_1, s_2, s_3, s_4, \ldots \rangle$$

be a sequence. A recurrence equation computes s_n as an expression involving previous sequence values $s_0, s_1, s_2, \ldots, s_{n-1}$. For instance, terms in the Fibonacci sequence can be computed by the recurrence equation.

$$f_n = f_{n-1} + f_{n-2}, \quad n \in \mathbb{N}, n \geq 2$$

To be used, initial values for f_0 and f_1 must be specified. In particular, set $f_0 = 0$ and $f_1 = 1$ to compute the values

$$\begin{aligned} f_2 &= f_0 + f_1 = 0 + 1 = 1 \\ f_3 &= f_2 + f_1 = 1 + 1 = 2 \\ f_4 &= f_3 + f_2 = 2 + 1 = 3 \\ &\vdots \end{aligned}$$

Here is list of recurrence relations for some useful sequences.

- Terms in the Alice sequence can be computed by the recursion

  $$a_n = a_{n-1} \quad \forall n \in \mathbb{Z}^+ = \{1, 2, 3, 4, \ldots\}$$

  and initial condition $a_0 = 1$.

- Terms in the Gauss sequence can be computed by the recursion

  $$g_n = g_{n-1} + 1 \quad \forall n \in \mathbb{Z}^+ = \{1, 2, 3, 4, \ldots\}$$

  and initial condition $g_0 = 1$.

- Terms in the sequence of triangular numbers can be computed by the recursion

  $$t_n = t_{n-1} + (n-1) \quad \forall n \in \mathbb{Z}^+ = \{1, 2, 3, 4, \ldots\}$$

  and initial condition $t_0 = 0$.

- Terms in the sequence of doubling numbers can be computed by the recursion

  $$d_n = 2d_{n-1} \quad \forall n \in \mathbb{Z}^+ = \{1, 2, 3, 4, \ldots\}$$

  and initial condition $d_0 = 1$.

- Terms in the Mersenne sequence can be computed by the recursion

$$m_n = 2m_{n-1} + 1 \quad \forall n \in \mathbb{Z}^+ = \{1, 2, 3, 4, \ldots\}$$

and initial condition $m_0 = 0$.

- Terms in the factorial sequence can be computed by the recursion

$$n! = n(n-1)! \quad \forall n \in \mathbb{Z}^+ = \{1, 2, 3, 4, \ldots\}, \quad 0! = 1$$

and initial condition $0! = 1$.

- Terms in the Fibonacci sequence can computed by the recursion

$$f_n = f_{n-1} + f_{n-2}, \quad n \in \{2, 3, 4, \ldots\}$$

and two initial conditions $f_0 = 0$ and $f_1 = 1$.

- Terms in the sequence of harmonic numbers can computed by the recursion

$$H_n = H_{n-1} + \frac{1}{n} \quad \forall n \in \mathbb{Z}^+$$

and initial condition $H_0 = 0$.

- Terms in the Catalan sequence are computed by the recursion

$$c_n = c_0 c_{n-1} + c_1 c_{n-2} + \cdots + c_{n-2} c_1 + c_{n-1} c_0 \quad \forall n \in \mathbb{N}$$

and initial condition $c_0 = 1$.

The recursion for Catalan numbers is called a CONVOLUTION.

- Terms in the binomial sequence can be computed by the recursion

$$\binom{n}{m} = \binom{n-1}{m-1} + \binom{n-1}{m} \quad \forall n, m \in \mathbb{N}, n \geq m$$

and boundary conditions $\binom{n}{0} = 1$ and $\binom{n}{n} = 1$

- Stirling numbers of the first kind can be computed by

$$\begin{bmatrix} n \\ m \end{bmatrix} = \begin{bmatrix} n-1 \\ m-1 \end{bmatrix} + (n-1) \begin{bmatrix} n-1 \\ m \end{bmatrix}, \; n > 0$$

and boundary conditions $\begin{bmatrix} n \\ 0 \end{bmatrix} = 1$ and $\begin{bmatrix} n \\ n \end{bmatrix} = 1$

- Stirling numbers of the second kind can be computed by

$$\begin{Bmatrix} n \\ m \end{Bmatrix} = \begin{Bmatrix} n-1 \\ m-1 \end{Bmatrix} + m \begin{Bmatrix} n-1 \\ m \end{Bmatrix}, \; n > 0$$

and boundary conditions $\begin{Bmatrix} n \\ 0 \end{Bmatrix} = 1$ and $\begin{Bmatrix} n \\ n \end{Bmatrix} = 1$

1.5 *Defined by Functions on the Natural Numbers*

Terms in many sequences can be computed by using simple functions. Here is short list of useful sequences and the simple functions that compute the value of their terms.

- Terms in the Alice sequence are computed by the constant function

$$a(n) = 1 \quad \forall n \in \mathbb{N} = \{0, 1, 2, 3, 4, \ldots\}$$

- Terms in the Gauss sequence are computed by the linear function

$$g(n) = n \quad \forall n \in \mathbb{N}$$

- Terms in the triangular sequence are computed by the quadratic function

$$t(n) = \frac{n(n-1)}{2} \quad \forall n \in \mathbb{N} = \{0, 1, 2, 3, 4, \ldots\}$$

- Terms in the Mersenne sequence are computed by the exponential function

$$m(n) = 2^n - 1 \quad \forall n \in \mathbb{N}$$

- Terms in the Fermat sequence are computed by the hyper-exponential function

$$r(n) = 2^{2^n} + 1 \quad \forall n \in \mathbb{N}$$

- Terms in the relation count sequence are computed by the function

$$l(n) = 2^{n^2} \quad \forall n \in \mathbb{N}$$

1.6 *Computed by Algorithms*

Some sequences do not have simple functions or simple recurrence equations that compute their terms. There is no simple recurrence equation or simple function that will compute the values in the sequence of prime numbers. However there are algorithms that will compute the sequence of prime numbers. These algorithms will not halt because there are a countably infinite number of primes, but given any natural number n, the algorithms will compute all primes less than n.

Although there is a simple recurrence equation for the factorial sequence there is no simple function for the terms. Although factorials are often defined by the expression

$$n! = n(n-1)(n-2)\cdots(2)(1)$$

with initial condition $0! = 1$, this expression is more an algorithm than a function. By extension, computing the value of binomial coefficients by the factorial expression

$$\binom{n}{m} = \frac{n!}{m!(n-m)!}$$

is more an algorithm, than a simple function.

1.7 Non-Computable Sequence

Let $\mathbb{S}$ be the set of all countably infinite sequences on the natural numbers, that is,

$$\mathbb{S} = \left\{ \vec{S} : (\vec{S} = \langle s_0, s_1, s_2, \ldots \rangle) \wedge (\forall k \in \mathbb{N})(s_k \in \mathbb{N}) \right\}$$

Consider the decision problem:

Is every sequence in $\mathbb{S}$ computable?

That is, for each sequence, is there an algorithm that will compute the sequence's terms? It turns out that the answer is NO. There are sequences of natural numbers that cannot be algorithmically computed.

It is beyond the scope of this class to present formal definitions of computability, but the basic concepts can be appreciated without complete understanding of the detail. As an introduction, a decision problem is computable if there is a Turing machine that solves the problem. Basically, a Turing machine takes in some input values, runs a program on them, and out puts the answer.

Because of the way in which Turing machines are constructed, there are only a countably infinite number of them. That is, there is a one-to-one and onto correspondence between Turing machines and the natural numbers $\mathbb{N}$. Therefore, there are only a countably infinite number of sequences. On the other hand, there are more than a countable number of sequences in the set $\mathbb{S}$. Thus, there must sequences of natural numbers that cannot be computed.

The HALTING sequence gives the number of steps Turing machine T_k executes on input I_j. If machine T_k does not halt on input I_j, then define the number of steps to be the symbol ∞. The halting sequence cannot be computed.

The BUSY BEAVER sequence gives the maximal number of steps an n state Turing machine can make on an initially blank tape and halt. The busy beaver sequence cannot be computed. Most sequences cannot even be described!