# Lec 10: Summations + Algorithms

Sequence: an ordered list.

$$S = \{a_j\} \quad \text{where } j \in \mathbb{N} \text{ or } j \in \mathbb{Z}^+$$

Typically want the underlying formula to generate the sequence.

i.e. arithmetic sequence $\boxed{a_j = a + dj}$

for $a, d \in \mathbb{R}$ given. (often $a, d \in \mathbb{Z}$)

• geometric sequence $\boxed{a_j = ar^j}$

for $a, r \in \mathbb{R}$, $j \in \mathbb{N}, \mathbb{Z}^+$

Note is $r=1$, $a_j = a$ for all $j$

$r=1$, $S = \{a, a, a, \ldots\}$

$r \neq 1$, $S = \{a, ar, ar^2, ar^3, \ldots\}$.

Summations

$$a_0 + a_1 + a_2 + \cdots + a_n = \sum_{j=0}^{n} a_j$$

more generally

$$a_m + a_{m+1} + \cdots + a_n = \sum_{j=m}^{n} a_j$$

Summation indices are flexible.

$$\sum_{j=0}^{n} a_j = \underbrace{a_0 + \sum_{j=1}^{n} a_j}_{\text{pull out first term}} = \underbrace{\sum_{j=0}^{n-1} a_j + a}_{\text{pulled out last term}}$$

$$= a_0 + \sum_{j=1}^{n-1} a_j + a_n \quad , \text{ etc.}$$

Recall important sums:

$$S_{int}(n) = \sum_{j=0}^{n} j = \sum_{j=1}^{n} j = \frac{n(n+1)}{2}$$

$$S_{geo}(n) = \sum_{j=0}^{n} ar^j = \begin{cases} a(n+1) & \text{if } r=1 \\ \dfrac{ar^{n+1} - a}{(r-1)}, & \text{if } r \neq 1 \end{cases}$$

(recall we calculated this by considering $rS_{geo}(n)$ ).

What if instead of $j=0$ we start w/ $j=1$?

$$S'_{geo}(n) = \sum_{j=1}^{n} ar^j$$

Approach 1: $S_{geo}(n) = S'_{geo}(n) + a$

Telescopic sum

$$S = \sum_{j=1}^{N} (a_j - a_{j-1})$$

$$= (a_1 - a_0) + (a_2 - a_1) + (a_3 - a_2) + \cdots + (a_{n-1} - a_{n-2}) + (a_n - a_{n-1})$$

$$= -a_0 + a_n$$

Important series:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots \quad = \sum_{j=0}^{\infty} \frac{x^j}{j!}$$

(Recall $0! = 1$).

why $e^x \simeq 1 + x$ if $|x| \ll 1$. for $x \in \mathbb{R}$.

$$e^{ix} = \cos x + i \sin x \qquad (\text{Recall } i^2 = -1)$$

collect real terms:

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots$$

Collect imaginary terms

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots$$

1) Algorithm a proceedure for performing a computation.

Takes an iput and maps it to an output.

Every input (of the correct type) is mapped deterministically to one output.
(ie. a function)

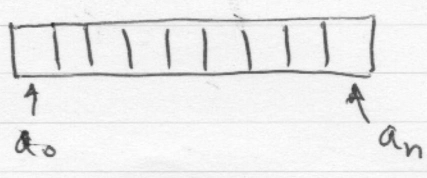
2) Properties:
- Definite
- Correct
- Effective
- Generalizable.

3) * Fundamental constructs of an algorithm:

| | |
|---|---|
| if-then statements | (the implication) |
| while loops | (while $p = T$) |
| for loops. | (for specified $x \in U$). |

4) Simple algorithm.

Find the maximum value in an unordered list




~~Best case scenario: m~~

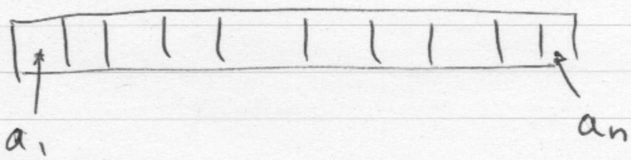
* Must search every element
(# of operations grows linearly with n).

5. Searching a list

Is it ordered? $\begin{cases} \text{No} \longrightarrow \text{linear search} \\ \text{Yes} \longrightarrow \text{binary search} \end{cases}$

Linear search.
locate an element $x$ in list $a$.

* Start from the beginning & stop once I find item $x$ in list $a$ (or else exit the list).




Is $a_j = x$ for any $j$?

How long does linear search take?

Best case; $x = a_1$ (one "while" loop execution)

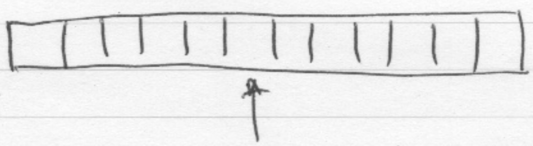
worst case; $x$ is not in list. ($n$ "while" loops)

Average ave; $x = a_{n/2}$ (search half the list)
($n/2$ "while" loops).

(worst case grows linearly w/ $n$
ave case grows linearly w/ $n$).

If the list is ordered, we can do much better.

Binary search: (list where elements are ordered from smallest to biggest)



look at middle item, $a_m$

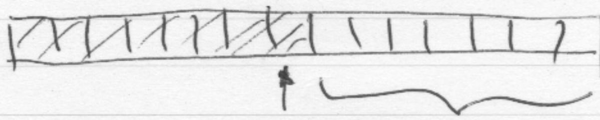
$$m = \left\lfloor \frac{n+1}{2} \right\rfloor$$

Is $x > a_m$?

yes: then any $a_j = x$ will need $j > m$.

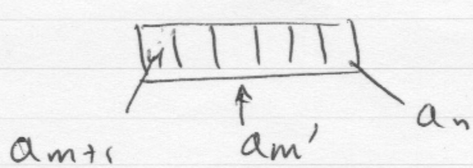
no: then any $a_j = x$ will need $j \leq m$.

Recurse if for instance $x > a_m$



left with $a_{m+1}$ to $a_n$

$x > a_m$




If $x > a_{m'}$ where $a_{m'}$ is the new middle item?

if yes then any $a_j = x$ will need $j > m'$

if no " " " " " " $j \leq m'$

CENTRE
DE RECHERCHES
MATHÉMATIQUES

Each iteration reduces the length of the list in half.

For a list of length $2^k$.

max of $k$ operations.