Improving binary search: interpolation search

InterpolationSearch(list a, key), a sorted in ascending order

```
i = 0, j = len(a)-1
while a[i] < key < a[j] do
    mid = ⌊ i + ((key − a[i])(a[j] − a[i])) / (a[j] − a[i]) ⌋
    if a[mid] < key then
        j = mid - 1
    else if a[mid] > key then
        i = mid + 1
    else
        return mid
    end if
end while
if a[i] == key then
    return i
else
    return -1
end if
```

→ picking proportionally

Improving binary search: ternary search

TernarySearch(list a, key, i, j), a sorted ascending

if $i \geq j$ **then**
  if a[i] == key **then**
    Return i
  else
    Return −1 {Key not present}
  end if
else
  $mid_1 = \left\lfloor i + \frac{j-i}{3} \right\rfloor$, $mid_2 = \left\lfloor i + \frac{2(j-i)}{3} \right\rfloor$
  if a[mid_1] == key **then**
    Return mid_1
  else if a[mid_2] == key **then**
    Return mid_2
  else if key < a[mid_1] **then**
    Return BinarySearch(a, key, i, mid_1−1)
  else if key > a[mid_2] **then**
    Return BinarySearch(a, key, $mid_2 + 1$, j)
  else
    Return BinarySearch(a, key, $mid_1 + 1$, $mid_2 - 1$)
  end if
end if

Finding duplicates: SCAN

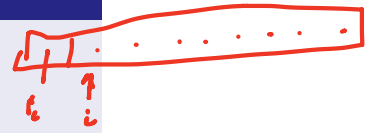
Suppose we have a list A of n elements, and we want to know if they contain a *pair* of duplicate values.

SCAN algorithm

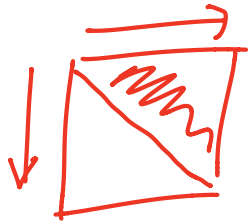
```
for i in 0 to n − 2 do
    for j in i + 1 to n − 1 do
        if A[i] = A[j] then
            Return True
        end if
    end for
end for
Return False
```

n times

roughly n times



Very roughly n^2

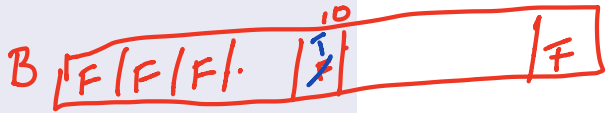


Finding duplicates: STOR

Suppose we have a list A of n elements, and we want to know if they contain a *pair* of duplicate values.

STOR algorithm: Uses second list B

```
for i in 0 to n − 1 do
    if B[A[i]] = True then
        Return True
    else
        B[A[i]] ← True
    end if
end for
Return False
```



How big does B need to be?

Efficiency of SCAN vs STOR

Random selections

- How could we make a random selection of k distinct items from a total of N?
- What if we don't know what N is? For example, suppose we want to select 50 ticket numbers from the crowd at a rugby match.
- Can we do it **without** storing all N items?

Reservoir sampling

Reservoir sampling: assumes stream a of unknown length N, $k < N$ items to be uniformly randomly chosen for sample S

$S = []$
for i in 0 to $k - 1$ **do**
  Add $a[i]$ to S
end for
for i in k to $N - 1$ **do**
  $x = randInt(0, i)$
  if $x < k$ **then**
    Replace $S[x]$ by $a[i]$
  end if
end for