Algorithms Homework 3

BRANDON YUEN, by161, Section 06 & RYAN LEE, rsl82, Section 07 & JAMES ALBA, jma

This is a joint assignment between Brandon Yuen, Ryan Lee, and James Alba
October 2022

Problem 1: Brandon Yuen, by161, section 06, Ryan Lee, rsl82, section07, and James Alba, jma390, section 07

1.

```
B ← empty Dictionary
for int i in A do
   if B.search(i-1) = NIL then
      B.add(i,1)
   else
      B.add(i,B.search(i-1) + 1)
   end if
end for
answer ← 0
for int i in A do
   if B.search(i) > answer then
      answer ← B.search(i)
   end if
end for
return  answer
```

Problem 2: Brandon Yuen, by161, section 06, Ryan Lee, rsl82, section07, and James Alba, jma390, section 07

Algorithm 1 problem2(A,k)

```
1.  B ← Empty Dictionary
    answer ← []
    count ← 0
    for i=0 to k-1 do
      if B.search(A[i]) = NIL then
        B.add(A[i],1)
        count ← count+1
      else
        B.search(A[i]) ← B.search(A[i]) + 1
      end if
    end for
    answer.add(count)
    for i=1 to n-k do
      if B.search(A[i-1]) = 1 then
        B.delete(A[i-1])
        count ← count -1
      else
        B.search(A[i-1]) ← B.search(A[i-1]) - 1
      end if
      if B.search(A[i+k-1]) = NIL then
        B.add(A[i+k-1],1)
        count ← count + 1
      else
        B.search(A[i+k-1]) ← B.search(A[i+k-1]) + 1
      end if
      answer.add(count)
    end for
    return  answer
```

Problem 3: Brandon Yuen, by161, section 06, Ryan Lee, rsl82, section07, and James Alba, jma390, section 07

1.
```
A ← sorted(A)
for i=1 to len(A) do
  if A[i-1]=-i then
    return A[i-1]
  end if
end for
return No Solution
```

Algorithm 2 FindSpecial(A,offset) ← FindSpecial(A,0)

2.
```
Split A into groups of 5: A[1...5], A[6...10], A[11...15], ...A[n-4...n]
Compute mi ← the median of group i.
Create array M of length n/5 with all mi.
m ← SELECT(M,n/10)
A1,A2 ← PARTITION(A,m)
Let L1, L2 be lengths of A1, A2.
if L1 + 1 = -m + offset then
  return m
end if
if L1 + 1 < -m + offset then
  return FindSpecial(A2, offset + L1 + 1)
end if
if L1 + 1 > -m + offset then
  return FindSpecial(A1, offset)
end if
return No solution
```

We decide to pivot in such a way because in order to find a special number, -m has to match L1 + 1 + offset(initially 0). A1 and the length of it gives us information on the rank of our chosen middle, and with that information we will have to pivot left or right depending on if our number's rank is the actual rank based on A1, or has overshoot/undershoot.

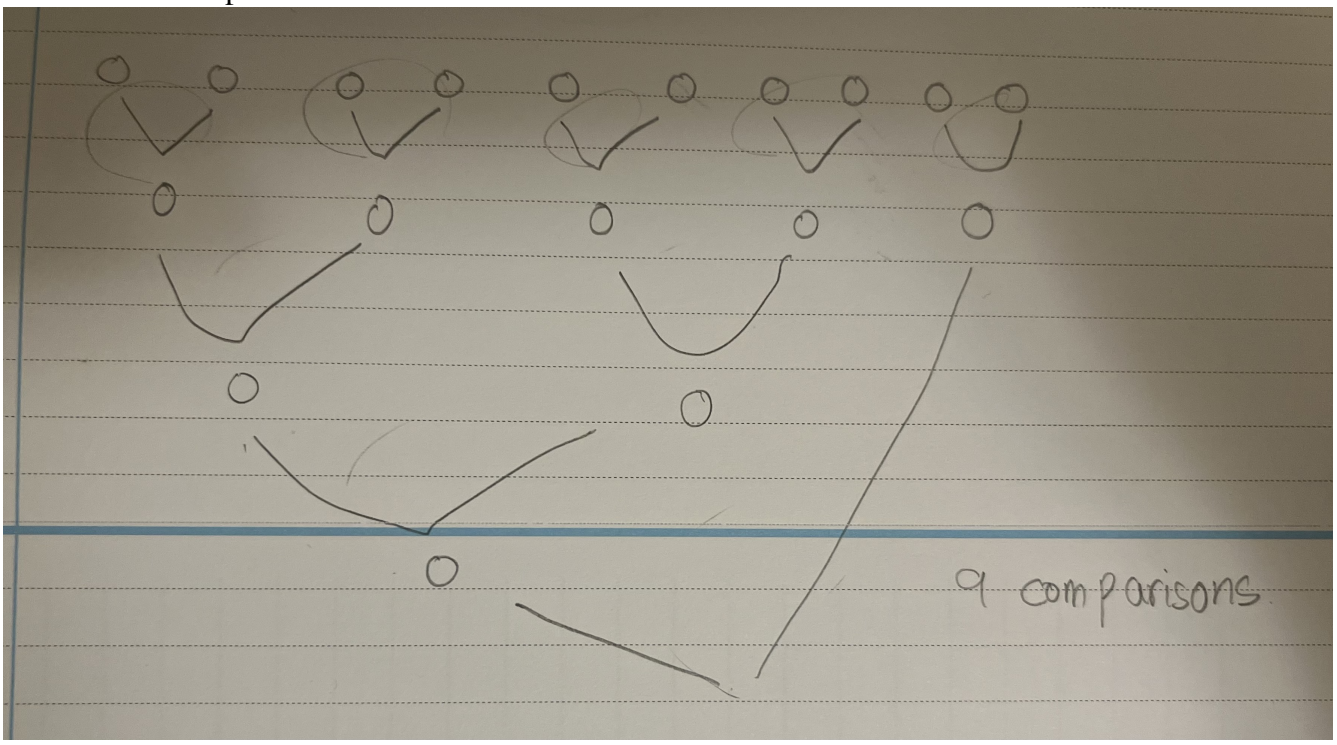
For example, say we get a initial middle of -8 and the size of A1 is 2, it is then impossible for A1 to have any special numbers because in order for that to happen, the potential special numbers are (-1,-2) in A1 but they are all bigger than -8 which is not possible. However it is still possible that a special number can exist in the other half so we will check there. If the size of A1+1 is bigger than -m then we ignore A2 because in order for a special number to exist there has to be a number less than -m that exists in A2 which is not possible.

We will update offset based on if we are pivoting towards the right to preserve the overall rank in the list.

Recurrence Formula: T(n) = T(n/5) + T(7n/10) + O(n) = O(n)

Problem 4: Brandon Yuen, by161, section 06, Ryan Lee, rsl82, section07, and James Alba, jma390, section 07

Pair up with two toys and find lighter one.
For example, (t1,t2),(t3,t4)...(t1023,t1024). (512 comparisons)
And we get 512 toys, do the same thing until we get lightest toys.
Comparisons = 512 + 256 + 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 1023 comparisons.
Now, pick 10 toys that are defeated by lightest toy. They are potential 2nd lightest toys because 2nd lightest toy should win against everything else except lightest toy.
Do the same comparison for 10 losers.



It should look like this, so finding second lightest toy needs 9 comparison.
So, overall comparisons to find two lightest toys are 1023 + 9 = 1032 comparisons.