Class Meeting #7

COS 226 — Spring 2018

Mark Braverman

(based on slides by

Jérémie Lumbroso)

Problem #3: Duplicate Element

Stream: x[1] x[2] x[3] x[4] … x[N+1]
Elements from 1…N

At least 1 duplicate: 1, 5, 4, 3, 2, 1
Possibly many: 2, 1, 1, 1, 1, 2

Problem: find one duplicate element
Requirements:
- (~) constant auxiliary memory
- as few passes as possible

Problem #3: Naïve One Pass

As before, we can build a histogram:

array **a**[1..N] initially all 0

foreach element x[i]

    increment a[x[i]] by 1

for i = 1 to N

    if a[i] > 1 then print "Duplicate is " + i

One pass, but linear space

Problem #3: Naïve One Word

We can flip the naïve solution to use linear passes and constant memory

```
for i = 1 to N
    counter := 0
    foreach element x[j]:
        if x[j] == i then counter = counter + 1
    if counter > 1 then print "Found duplicate " + i
```

Sketch of better solution

Solution: constant memory, log N passes

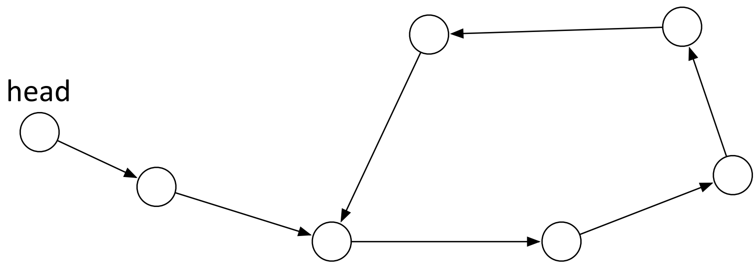
Use two counters

- one counter to track values in [1, N/2)
- one counter to track values in [N/2, N]

The counter that is larger indicates to range to visit (lower half or higher half)

Recursively look at half

Problem #4: Cycle Detection



Single linked list got corrupted and has cycle

<u>Question 1:</u> how to detect cycle?

<u>Question 2 (harder):</u> how to fix cycle?

Problem #4: Suboptimal ideas (1)

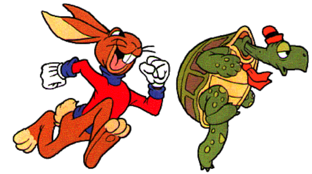
Try to traverse the list (possibly does not terminate)

Keep track of all elements seen so far (requires linear extra memory + does not allow duplicates)

Keep track of the pointer addresses seen so far (requires linear extra memory)

Problem #4: Tortoise and Hare

Traverse the list using two pointers



- Tortoise which follows each **node.next**
- Hare which follows at twice the pace, going each time to **node.next.next**

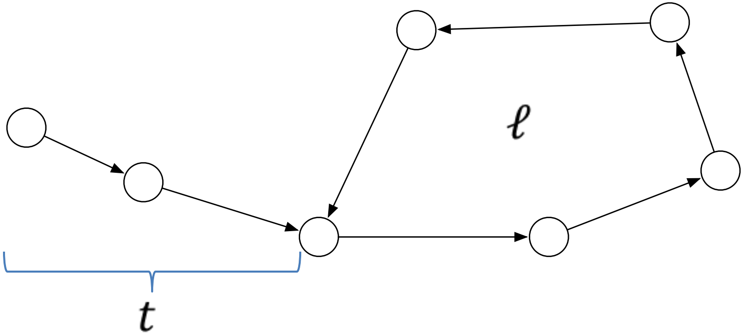
If there is a loop, they will at some point be on the same element

We can then compare addresses

(and if not, then the Hare eventually will get to a null element)

*How to find the cycle's location?

Problem 4: locating the cycle

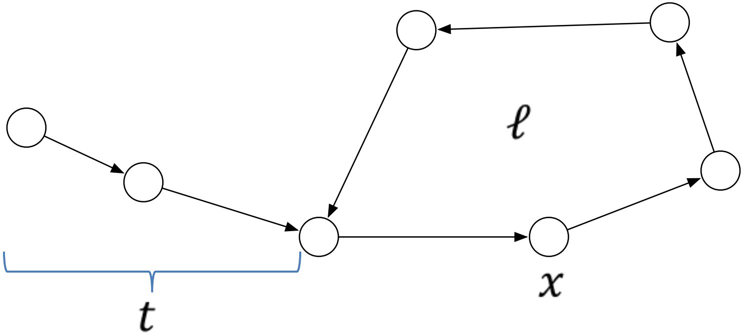


Need to figure out the number of steps t to reach cycle and cycle length ℓ

In the picture above, $t = 2$, $\ell = 5$

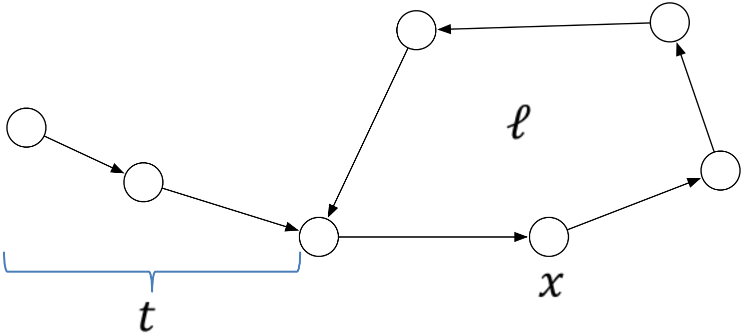
After the tortoise and hare meet, let tortoise rest, and let hare run another lap, counting its steps will give us ℓ.

Problem 4: finding t



Let x be the number of steps from the start of the cycle that the hare and tortoise meet for the first time.
Let m be the number of steps the tortoise makes before the meeting. Then $m = t + x$

Problem 4: finding t



$$m = t + x$$

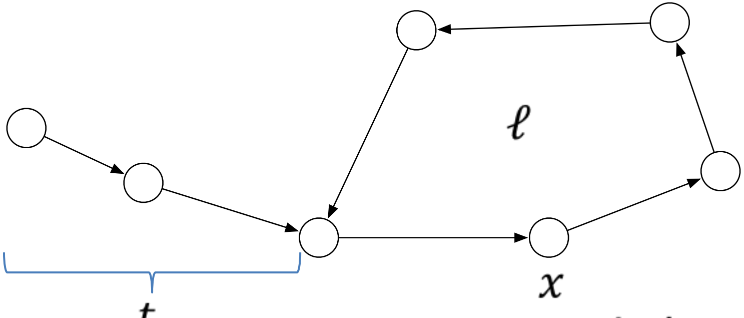
The hare makes $2m$ steps. Therefore

$$2m = t + x + k \cdot \ell$$

for some integer k. $\mathrm{m} = t + x = k \cdot \ell$

Place turtle at x, and hare at the beginning, and let them run at speed 1.

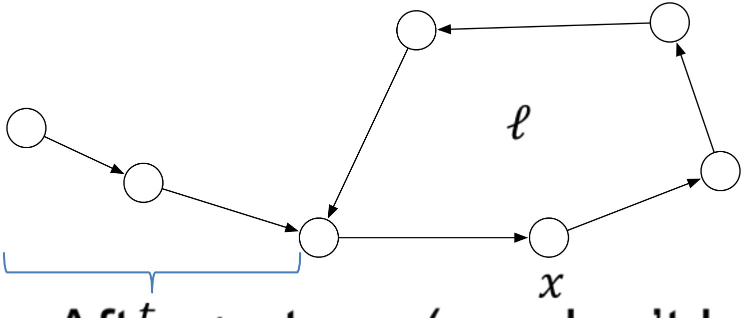
Problem 4: finding t



$$\mathrm{m} = t + x = k \cdot \ell$$

Place turtle at x, and hare at the beginning, and let them run at speed 1.
After t steps (we don't know t), hare will be at the start of the cycle. Turtle will be $x + t = k \cdot \ell$ from the beginning of the cycle, i.e. at the beginning of the cycle too!

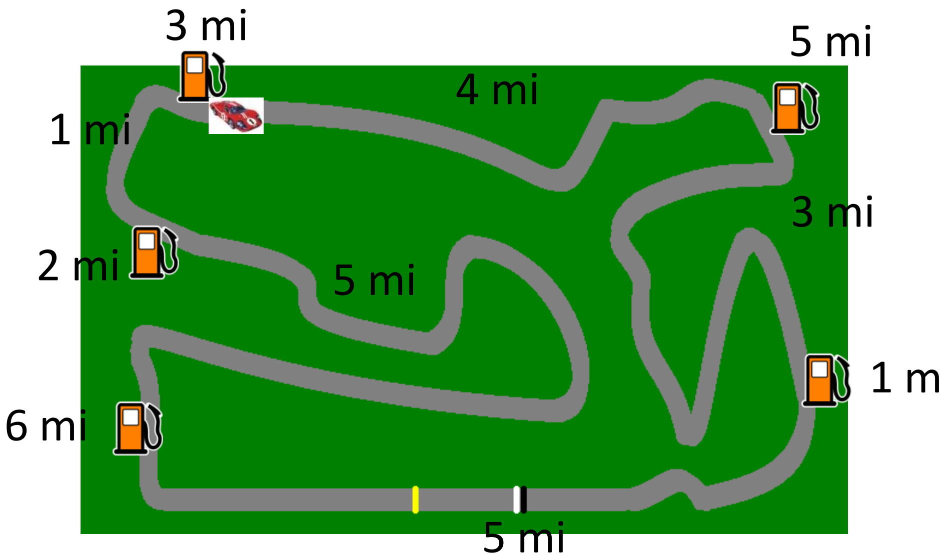
Problem 4: finding t



After t steps (we don't know t), hare will be at the start of the cycle. Turtle will be $x + t = k \cdot \ell$ from the beginning of the cycle, i.e. at the beginning of the cycle too!
They meet for the first time after exactly t steps at the beginning of the cycle.

Problem #5

(Dynamic programming preparation)
Just enough gas to complete the course.
Where to start?

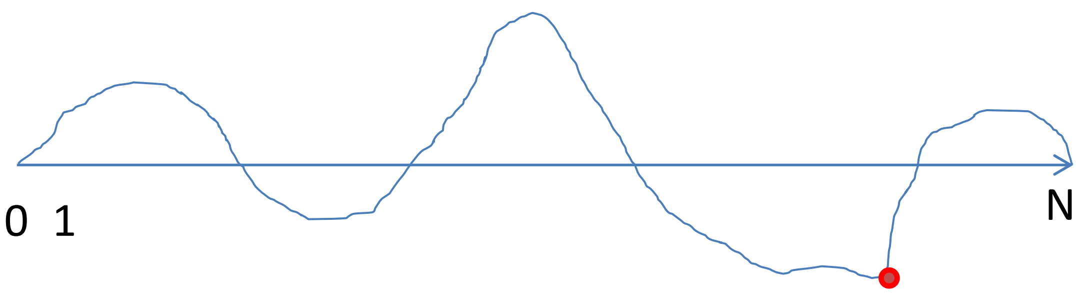


Problem #5

G[1] … G[N]
C[1] …C[N]
G[i] – gas at location i
C[i] – cost of segment after location i
G[1]+…+G[N]=C[1]+...+C[N]
Want: a location j such that for all k
$G[j]+G[j+1]+\ldots+G[j+k] \geq C[j]+\ldots C[j+k]$,
where addition is modulo N (so (N-4)+7=3).

Problem #5

Imagine that we could "overdraw" gas.
Start at 1.
After i steps have
A[i]:=G[1]-C[1]+G[2]-C[2]+…+G[i]-C[i]
gas.



Problem #5

Let j be such that A[j] is the smallest. Start from j+1.

Gas after k steps:

G[j+1]-C[j+1]+G[j+2]-C[j+2]+…+G[j+k]-C[j+k]= A[j+k]-A[j]

Problem #6

Find the (contigeous) subarray with largest sum.

Problem #6 (sketch)

Calculate

Sum[i]=A[0]+…+A[i-1]

Calculate

Min[i]=min(Sum[0],…,Sum[i+1])

Max[i]=Sum[i+1]-Min[i]

Find the maximum value of Max[i] for i=1..N

Note that only a constant amount of extra memory is needed for these calculations