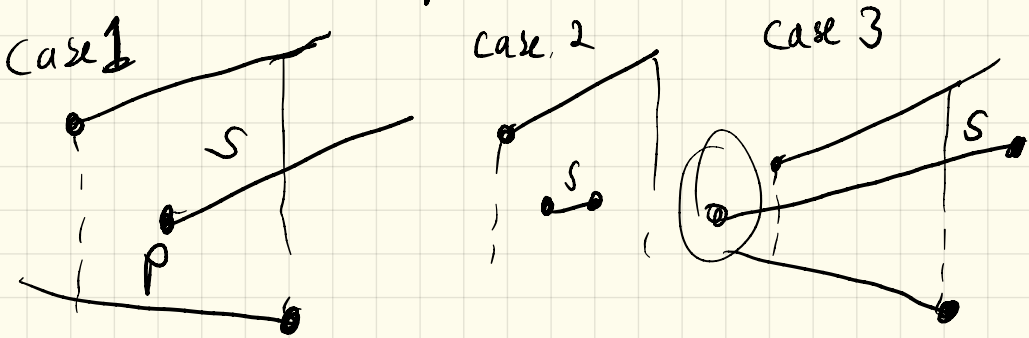
# Trap Map 3



For construction for our point location data structure




1 point in trap

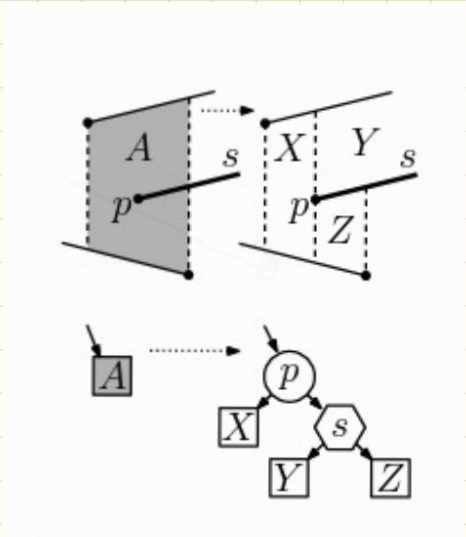
2 pts trap

Case 1 (single Left or right endpoint in trap)

- replace A w/ 3 traps X, Y, Z
- let p be the endpoint in A

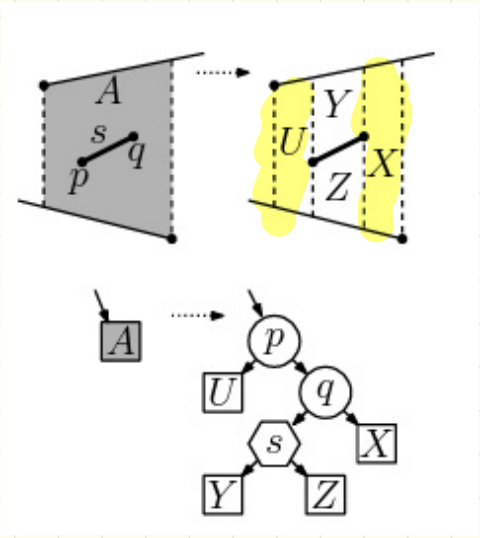
replace leaf of A:

1. create an x-node for p
2. add one child for trap X
3. add one child for y-node
   - 3a. add one child for Y (above)
   - 3b. add one child for Z (below)

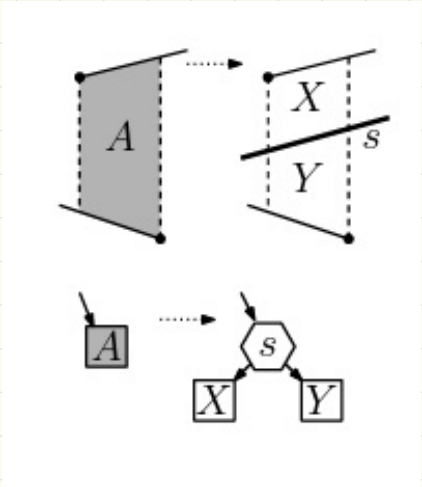


Case 2 (2 seg endpoints in trap)
- replace A w/ traps X, Y, Z, U
- let p and q be left and right endpoints of seg (resp)

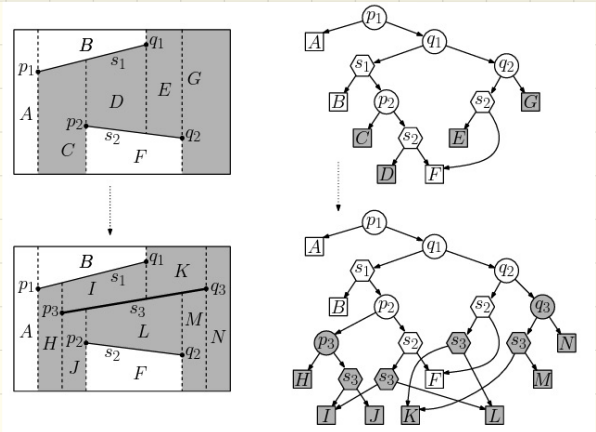
create x-node for p
1. add L child for trap U
2. add R child for x-node for q
   2a. add L child for y-node of seg
      * add L child for trap Y (above)
      * add R child for trap Z (below)
   2b. add R child for trap X




Case 3 (no seg endpoint)
- replace A w/ 2 traps X, Y
- create y-node w/ segment
  1. add L child for X (above)
  2. add R child for Y (below)




Example of not adding a trap multiple times

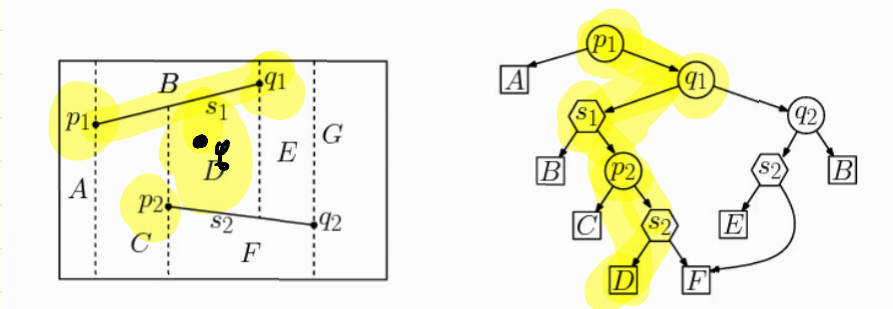


Analysis:
Claim expected $O(n)$ space
and $O(\lg n)$ search

Pf space:
- # of new nodes is proportional to # of new traps
- we already show $O(1)$ traps added in expection
  $\Rightarrow O(1)$ new nodes added in expetation
  $\Rightarrow O(n)$ size in expetation

Search:
idea for a fixed query $q$
expected len of search path for $q$
(expetation is taken ove all permutations of the segments)

- Let $q$ be a querypoint
consider how $q$ moves through search data structure
as addition of each line segment
Let $\Delta_i$ be the trap that $q$ is in after $s_i$ is inserted
if $\Delta_i = \Delta_{i-1} \Rightarrow s_i$ did not affect the trap containing $q$
if $\Delta_i \neq \Delta_{i-1} \Rightarrow s_i$ caused trap $\Delta_{i-1}$ to be replaced
find the next trap $O(1)$ work (at most 3 leafs to traverse - case 2)
$\Rightarrow$ expected length of search path is 3x expected # of times $q$ 'changes' traps



Let $X_i(q)$ be the random event that $q$ changes traps on $i^{th}$ insertion
Let $Pr(X_i(q))$ be the prob of the event
Let $D(q)$ be the average depth of $q$ in the search tree
↖ expected path length

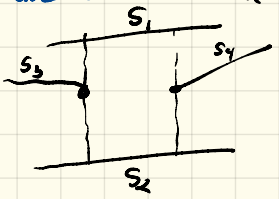
$$D(q) \le 3 \sum_{i=1}^{n} Pr(X_i(q)) \quad *$$

Wts $D(q) < O(\lg n)$ in expectation
we will show $Pr(X_i(q)) \le 4/i$ ← show w/ backwards analysis

Consider trap $\Delta_i$ contains $q$ after $i^{th}$ segment
$\Delta_i$ only changed if $s_i$ is a segment defing $\Delta_i$

What is the prob that $s_i$ is one of the segments?
- any order of segments is equally likely
- at most 4 segments define $\Delta_i$

$\Rightarrow Pr(X_i(q)) \le 4/i$



Case realizing 4 seg defing trap

now we can consider *

$$D(q) \le 3 \sum_{i=1}^{n} Pr(X_i(q)) \le 3 \sum_{i=1}^{n} 4/i = 12 \sum_{i=1}^{n} 1/i > O(\ln n) = O(\log n) \quad \checkmark$$

fact $\sum_{i=1}^{n} 1/i$ is called Harmonic Series
for large $n$
$\sum_{i=1}^{n} 1/i = \Theta(\ln n)$

Guarantee on search time

**Lemma:** Given a set of $n$ non-crossing line segments in the plane, and a parameter $\lambda > 0$, the probability that the total depth of the randomized search structure exceeds $3\lambda \ln(n+1)$, is at most $2/(n+1)^{\lambda \ln 1.25 - 3}$.

alg:
run the alg and track the depth as we go
if depth is too long (bigger than $c(\log n)$ for some $c$)
throw away and shuffle segs again and rerun
(lemma $\Rightarrow$ we don't need to restart too many times)

$\Rightarrow$

**Theorem:** Given a set of $n$ non-crossing line segments in the plane, in expected $O(n \log n)$ time, it is possible to construct a point location data structure of (worst case) size $O(n)$ that can answer point location queries in (worst case) time $O(\log n)$.

Note!
- Same idea leads to expected $O(I + n \lg n)$ time segment intersection algo
  ↳ recall plane sweep $O((n+I) \lg n)$