COP 4516 2/17/26

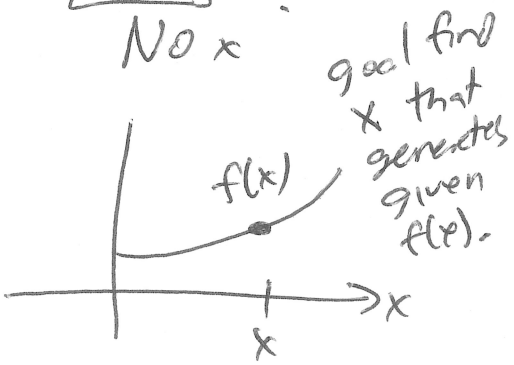
1) Intro idea by Etch
   real-valued binary search
2) real-valued B.S. – bootstrapping number (Kaths)
3) Integer Binary Search
   – Airport Shuttle
4) Int example: financial planning (Kaths)

Reg binary search

Input sorted array, value
Output – is the value in the array

| 2 | 3 | 17 | 22 | 99 | 106 |

45 ?
No x



How we apply this?

$$\frac{f_2 - f_1}{f_1 f_2} = at + b(1 - e^{-ct})$$

A real valued binary search is useful <u>if</u>

1) it's easy to calculate func forwards
2) function is ↗ or ↘
3) hard to invert function

* Qualities for a real-valued binary search!

To determine low and high, must do an analysis of the given problem.

low ≤ real ans
high ≥ real ans
} and cause NO arithmetic "~~expts~~ exceptions"

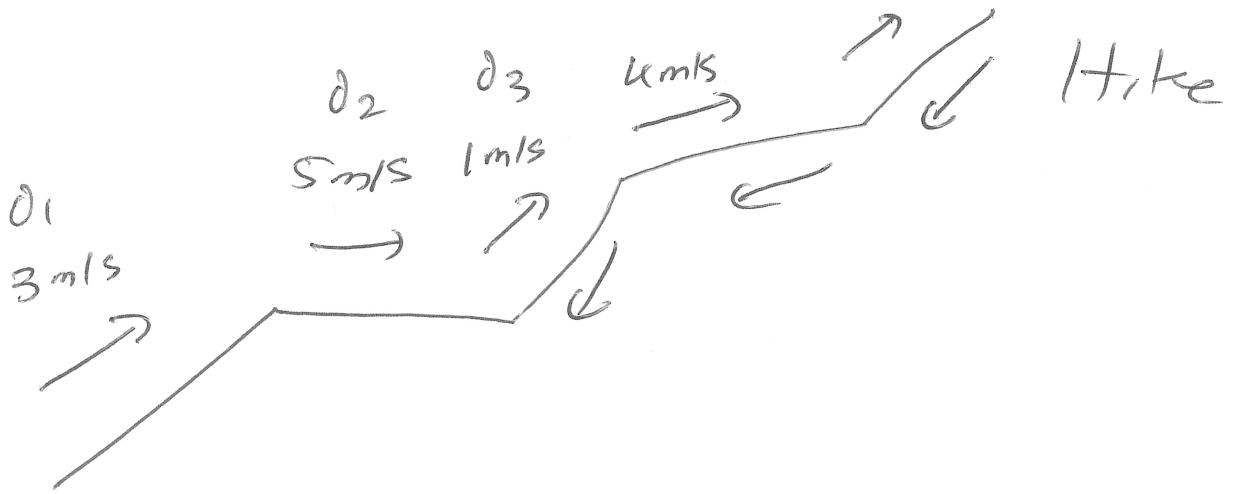
low = 0 easy

high = $\frac{\text{targetval}}{a}$, since $b(1-e^{-ct}) > 0$.

FIXED # iter

either low = mid
or high = mid
} exactly

example of another real valued b.s.



Int Binary Search

Airport Shuttle

→ easy to calc forwards (given ans)
→ hard to calc backwards
→ ans is an int
→ Careful setting low, high (problem specific)
NOT a fixed # iterations

```
while (low < high) {

}
```

TRICKY PART resetting low, high

If airport shuttle
- if can't do it
  - low = mid+1
- else
  - high = mid

mid = (low+high)/2
lo=2 hi=3
= (low+high+1)/2
int div

ceil(x/y) = (x+y-1)//y

Fin Plan

p[0] c[0]

n + p[0] - c[0] ≥ M
n + p[0] ≥ M + c[0]
n = ⌈ (M + c[0]) / p[0] ⌉