Exercise 1 David Riemer

Consider the backtracking algorithm for the Turnpike problem from class. How does an input difference multiset D look like that leads to an exponential running time of the algorithm?

Describe a family of inputs (i.e., for infinitely many sizes n) and prove that the backtracking algorithm needs exponential time.

When looking at how the algorithm works, we look at one example case:

We apply the backtracking Turnpike algorithm to

$D = \{1, 2, 3, 3, 5, 6, 8, 8, 9, 11\}$

At the Initialization

The maximum distance is $max(D) = 11$, We start with $P = \{0, 11\}$

Current differences: $\Delta P = \{11\}$.

Remaining differences: $D \setminus \Delta P = \{1, 2, 3, 3, 5, 6, 8, 8, 9\}$.

Then we continue with the next candidate which is $d = 9$:

When trying to choose the next distance d form the left at the 3rd recursion (Current $\Delta P = \{1, 2, 3, 8, 9, 11\}$, remaining $D \setminus \Delta P = \{3, 5, 6, 8\}$ and $d = 8$) Option 1 fails and we need to go back and try option 2, using 3 in the new $P\acute{}$.

In General

From this example we can deduct that the algorithm performs bad on an input, on which both options seem perfectly feasible at first, not harming any constraints on the algorithm; The algorithm then has to explore both paths, what causes $\Theta(2^n)$ runtime.

The algorithm always keeps an invariant $\Delta P \subseteq D$ and checks tries 2 placementes for the next largest d. (One from the right and oen from the left). It recurses whenever the new differences still lie in D.

An example for such a worst case (for binary tree of depth 3):

$D = \{1, 2, 2, 3, 3, 5\}$

At the Initialization

The maximum distance is $max(D) = 5$, We start with $P = \{0, 5\}$

Current differences: $\Delta P = \{5\}$.

Try to place point **at distance 3**:
Option 1 (From the left):

Add 3

$P\prime = \{0, 3, 5\}$, $\Delta P\prime = \{2, 3, 5\}$, which is a subset of D.

Option 2 (From the right):

Add $7 - 6 = 1$

$P\prime = \{0, 2, 5\}$ $\Delta P\prime = \{2, 3, 5\}$, which is a subset of D.

We now have 2 branches that again call 2 branches recursively.

From the first branch with $P\prime = \{0, 3, 5\}$, remaining $\{1, 2, 3\}$

Option 1 (From the left):

Try to place point **at distance 3**:

$P\prime = \{0, 3, 3, 5\}$, distance 0, try next

Option 2 (From the right):

Add $5 - 3 = 2$

$P\prime = \{0, 2, 3, 5\}$, $\Delta P\prime = \{1, 2, 2, 3, 3, 5\}$, which is D.

From the second branch with $P = \{0, 2, 5\}$, remaining $\{1, 2, 3\}$

Option 1 (From the left):

Try to place point **at distance 3**:

$P\prime = \{0, 2, 3, 5\}$, $\Delta P\prime = \{1, 2, 2, 3, 3, 5\}$,which is D.

Option 2 (From the right):

Add $5 - 3 = 2$

$P = \{0, 2, 2, 5\}$, distance 0, try next