SHAPE Final Project “Cupboard Jumps”

8.16.24

Dexter Theisen, Siddharth Nair

01

Problem Statement

What are we solving?!

“Cupboard Jumps” – 3500* rated

From Moscow-based CodeForces Round 707
***3500 is the highest rating a problem can receive**

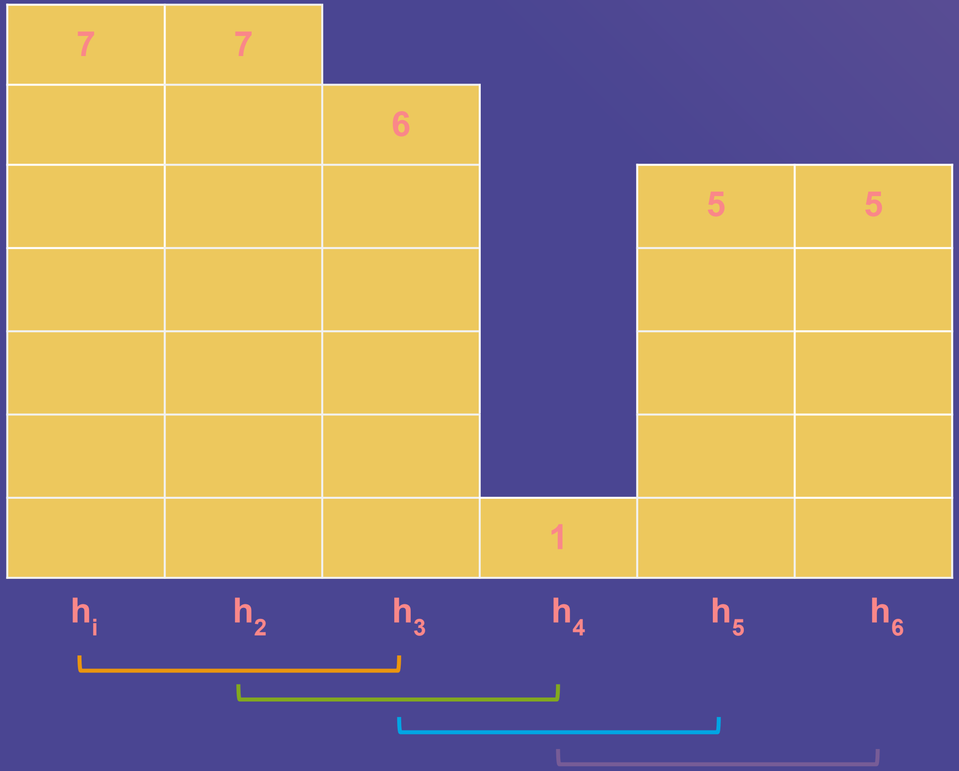
In the house where Krosh used to live, he had n cupboards standing in a line, the i-th cupboard had the height of h_i. Krosh moved recently, but he wasn't able to move the cupboards with him. Now he wants to buy n new cupboards so that they look as similar to old ones as possible.

Krosh does not remember the exact heights of the cupboards, but instead for every three consecutive cupboards he remembers the height difference between the tallest and the shortest of them. In other words, if the cupboards' heights were h_1, h_2, h_3, ..., h_n, then Krosh remembers the values $w_i = \max(h_i, h_{i+1}, h_{i+2}) - \min(h_i, h_{i+1}, h_{i+2})$ for all $1<=i<=n-2$.

Krosh wants to buy such n cupboards that all the values w_i remain the same. Help him determine the required cupboards' heights, or determine that he remembers something incorrectly and there is no suitable sequence of heights.

Note: all cupboards must have a height greater than or equal to 0.

Example Diagram



In this example, the cupboards have heights: 7, 7, 6, 1, 5, 5. This means that:

w_1 = max(7,7,6) - min(7,7,6) = 7-6 = 1
w_2 = max(7,6,1) - min(7,6,1) = 7-1 = 6
w_3 = max(6,1,5) - min(6,1,5) = 6-1 = 5
w_4 = max(1,5,5) - min(1,5,5) = 5-1 = 4

Example Input/Output

Input:

n	C
6	9

w_1	w_2	w_3	w_4
1	6	5	4

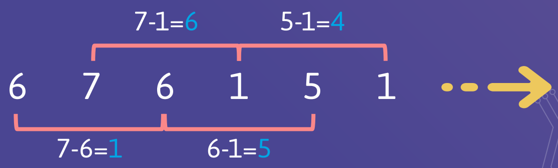
Here, n denotes the number of cupboards Krosh has. Each w_i denotes respectively the range for every 3 cupboards. And C represents an upper bound on w_i (used primarily in the code and not the logic.)

Output:

An example output may look like:

h'_1	h'_2	h'_3	$h'_{4,}$	h'_5	h'_6
6	7	6	1	5	1

where h'_i denotes the i-th cupboard in Krosh's new house. Note that there can exist more than one solution for any given w_i, but it is our job only to output one.



It can be easily seen that our solution yields the correct w_i:

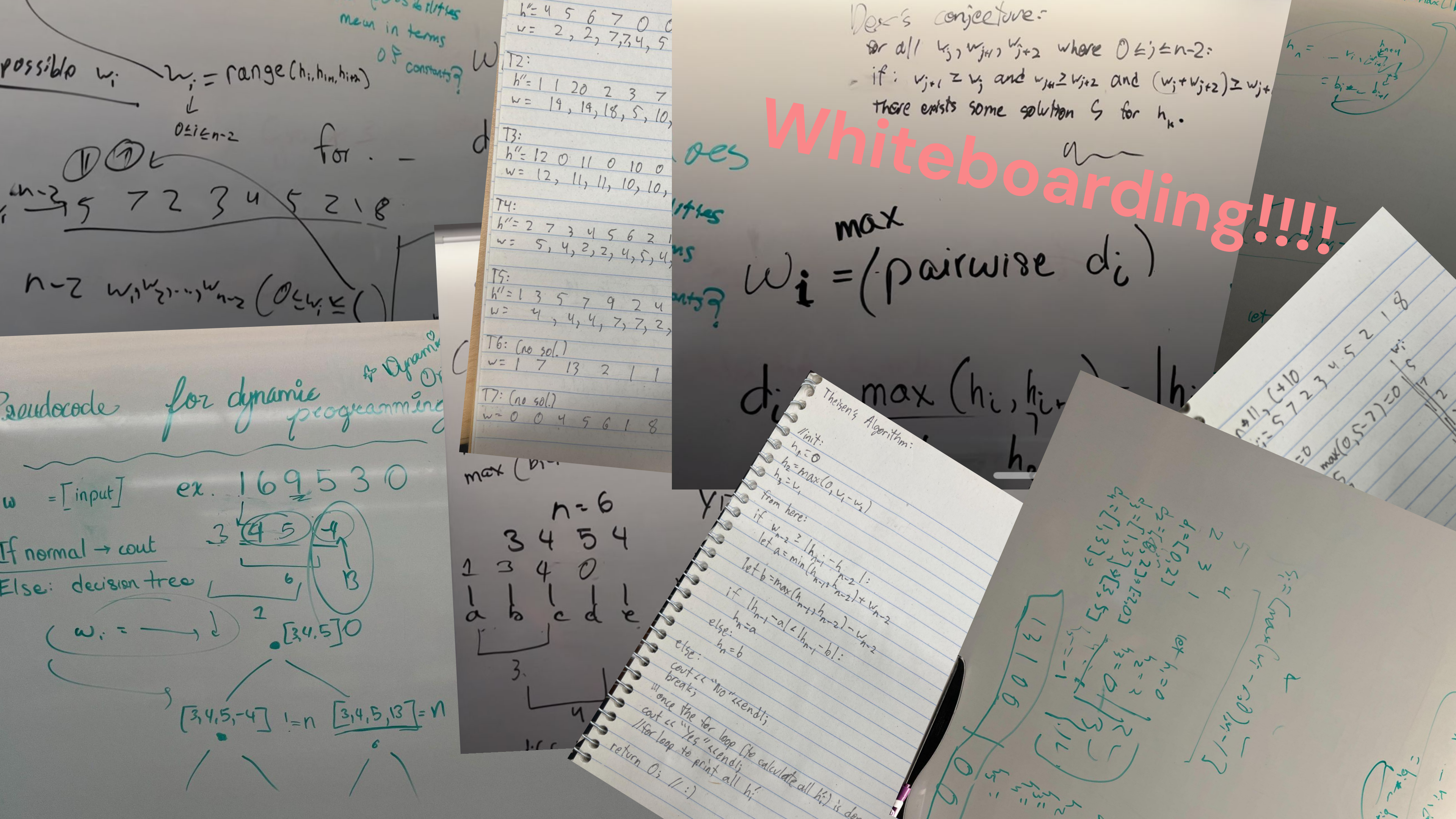
1 6 5 4

02

Brainstorming

Okay great! What do we do now?

Whiteboarding!!!!



Dex's conjecture:
for all w_j, w_{j+1}, w_{j+2} where $0 \le j \le n-2$:
if: $w_{j+1} \ge w_j$ and $w_{j+1} \ge w_{j+2}$ and $(w_j + w_{j+2}) \ge w_{j+}$
there exists some solution S for h_k.

$$w_i = \overset{max}{(\text{pairwise } d_i)}$$

d_i max $(h_i, h_i$

h''= 4 5 6 7 0 0
w= 2, 2, 7, 7, 4, 5

T2:
h''= 1 1 20 2 3 7
w= 19, 19, 18, 5, 10,

T3:
h''= 12 0 11 0 10 0
w= 12, 11, 11, 10, 10,

T4:
h''= 2 7 3 4 5 6 2 1
w= 5, 4, 2, 2, 4, 5, 4,

T5:
h''= 1 3 5 7 9 2 4
w= 4, 4, 4, 7, 7, 2,

T6: (no sol.)
w= 1 7 13 2 1 1

T7: (no sol.)
w= 0 0 4 5 6 1 8

```
Theisen's Algorithm:
//init:
h_1 = 0
h_2 = max(0, w_1 - w_2)
h_3 = w_1
from here:
if w_{n-2} ≥ |h_{n-1} - h_{n-2}|:
    let a = min(h_{n-1}, h_{n-2}) + w_{n-2}
    let b = max(h_{n-1}, h_{n-2}) - w_{n-2}
    if |h_{n-1} - a| < |h_{n-1} - b|:
        h_n = a
    else:
        h_n = b
else:
    cout << "No" << endl;
    break;
//once the for loop (to calculate all h_i) is don
cout << "Yes" << endl;
//for loop to print all h_i
return 0; // :)
```

Pseudocode for dynamic programming

```
w = [input]          ex. 1 6 9 5 3 0
If normal → cout         3 4 5 4
Else: decision tree
  w_i = → l                [3,4,5] 0
  [3,4,5,-4] != n      [3,4,5,13] = n
```

First Thoughts

After a few hours of whiteboarding, we had derived some key insights (not all of which did we strictly implement):

- Transform the w_i into pairwise differences, $d_i = h'_{i+1} - h'_i$, and subsequently rewrite w_i as $\max(|d_i|, |d_{i+1}|, |d_i+d_{i+1}|)$.
- WLOG*, assume we can fix h'_1 to 0, as even if we end up with some negative h'_i, we know there must be some k such that all $h'_i + k >= 0$.
- We need to recursively define h'_{n+1} from h'_n, h'_{n-1}, and w_{i+1} – but how? There are multiple ideas here, in minimizing vs maximizing $|h'_{n+1} - h'_n|$ (or doing neither).
- Dex's Conjecture: if there exists some i such that $w_i > w_{i+1} + w_{i-1}$, then there exists no h'_i that satisfy the given w_i. This can be proven mathematically by assigning ranges to each h'_i.
- Direction (ie increasing vs decreasing) of the w_i affects how to define h_{n+1}.
- At the start of the sequence, w_1 will be in increasing order.
- Set h_3' to w_1, and define h'_2 such that $h_1 <= h_2 <= h_3$.
- Utilize some type of dynamic programming or tree system to generate h'_{n+1} faster

Some cases

1 The case of 0

Key points: all h'_i need to be equivalent, meaning for the next w_i there is some constraint/forced h'_i.

2 Alternating w_i

Key points: very hard to know what value to set intermediate h' values to, because they affect 3 w_i.

3 No solution

Key points: Dex’s conjecture only finds about 95% of these cases, meaning our main algo will have to check for impossible cases as well.

03

Implementation

How exactly did we solve this problem?!

Full implementation

Algo

1. define basic variables, arrays, and references

```
#include <iostream>
#include <deque>
using namespace std;

typedef pair<long, long> pll;
#define MP make pair
#define fi first
#define se second

const int mn = 1000005;
int n;
long  , w[mn], a[mn], v1[mn], d[mn][3];
bool g[mn], rev[mn];
deque<pll> q;
```

2. define a function solve()

3. iterate over w_i, and store ranges for pairwise differences d_i in array d

```
bool solve() {
   long a = 1, b = 0;
   q.emplace back(0ll, w[1]);
   for (int i = 1; i <= n; i++) {
       if (a == 1) {
           while (!q.empty() && q.back().se + b > w[i]) {
               pll p = q.back();
               q.pop back();
               if (p.fi + b <= w[i]) {
                   q.emplace_back(p.fi, w[i] - b);
                   break;
               }
           }
           if (q.empty()) return false;
           if (q.back().se + b == w[i]) {
               g[i] = true;
               while (!q.empty()) q.pop back();
               q.emplace_back(0, w[i]), a = 1, b = 0;
           } else {
               v1[i] = q.back().se + b;
               a = -1, b = w[i] - b;
               q.emplace_front(b - w[i], b - w[i]);
           }
       else {
           while (!q.empty() && -q.front().fi + b
> w[i]) {
               pll p = q.front();
               q.pop front();
               if (-p.se + b <= w[i]) {
                   q.emplace_front(b - w[i],
p.se);
                   break;
               }
           }
           if (q.empty()) return false;
           if (-q.front().fi + b == w[i]) {
               g[i] = true;
               while (!q.empty()) q.pop back();
               q.emplace_back(0, w[i]), a = 1, b
= 0;
           } else {
               v1[i] = -q.front().fi + b;
               a = 1, b = w[i] - b;
               q.emplace_back(w[i] - b, w[i] -
b);
           }
       }
   }
```

4. backpropagate final ranges in array d that satisfy all w_i

```
if (q.empty()) return false;
   d[n][2] = a * q.back().se + b;
   for (int i = n; i; i--) {
       if (i < n) d[i][2] = d[i + 1][0];
       if (d[i][2] != w[i]) {
           if (g[i]) d[i][0] = w[i], d[i][1] =
w[i] - d[i][2];
           else d[i][1] = w[i], d[i][0] = w[i] -
d[i][2];
       } else if (g[i]) d[i][0] = w[i], d[i][1] =
0;
       else d[i][0] = v1[i], d[i][1] = w[i] -
v1[i];
   }
   return true;
}
```

5. grab inputs n, C, and w_i

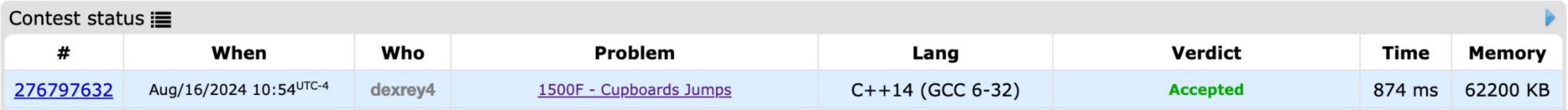
6. iterate over array d and finalize h'_i values

7. ensure all values are nonnegative

8. print out either YES followed by h'_i or NO

```
int main() {
   cin >> n >> _;
   n-=2;
   for (int i = 1; i <= n; i++) cin >> w[i];
   if (solve()) {
       a[1] = 0, a[2] = d[1][0];
       for (int i = 1; i <= n; i++) {
           if (w[i] == d[i][0] && !rev[i - 1])
d[i][2] = -d[i][2], rev[i] = true;
           else if (w[i] == d[i][1] && rev[i -
1]) d[i][2] = -d[i][2], rev[i] = true;
           else if (w[i] == d[i][2] && !rev[i -
1]) d[i][2] = -d[i][2], rev[i] = true;
           a[i + 2] = a[i + 1] + d[i][2];
       }
       long dlt = 0;
       for (int i = 1; i <= n+2; i++) dlt =
min(dlt, a[i]);
       cout << "YES" << endl;
       for (int i = 1; i <= n+2; i++) cout <<
a[i] - dlt << " ";
   } else { cout << "NO"; }
   cout << endl;
   return 0;
}
```

Thank you!



Contest status

#	When	Who	Problem	Lang	Verdict	Time	Memory
276797632	Aug/16/2024 10:54$^{\text{UTC-4}}$	dexrey4	1500F - Cupboards Jumps	C++14 (GCC 6-32)	Accepted	874 ms	62200 KB

Special thanks to our SSLs Abhishek and Courtney for their support