T211 - Snow Way

David Wai {wjx}

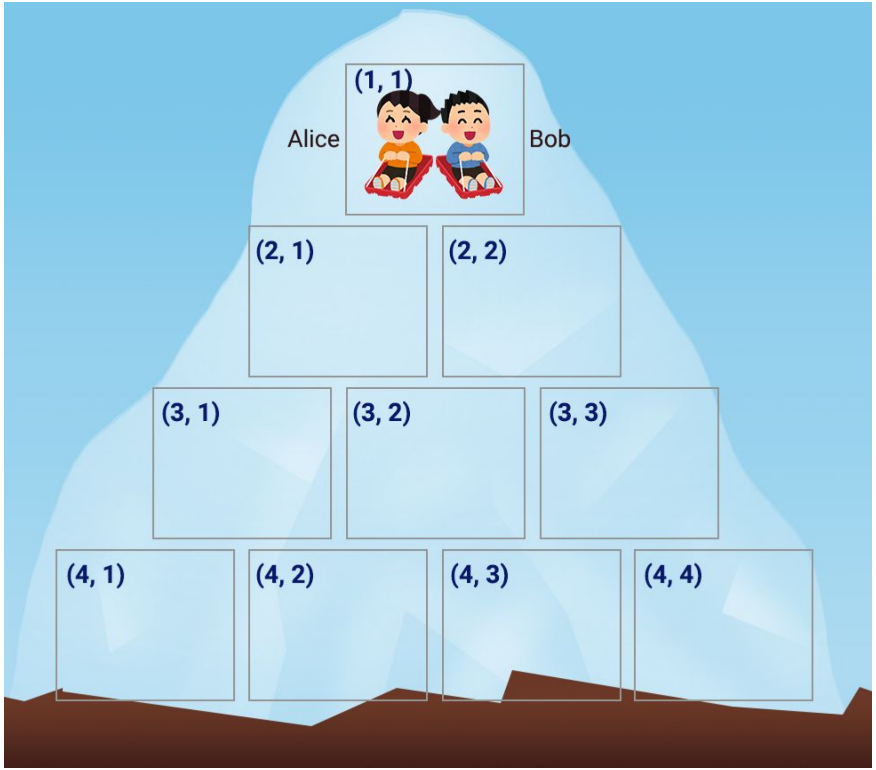
2021-04-18

Background

Problem Idea by Alex Tung

Preparation by David Wai, Ethen Yuen

Pictures by Tony Wong



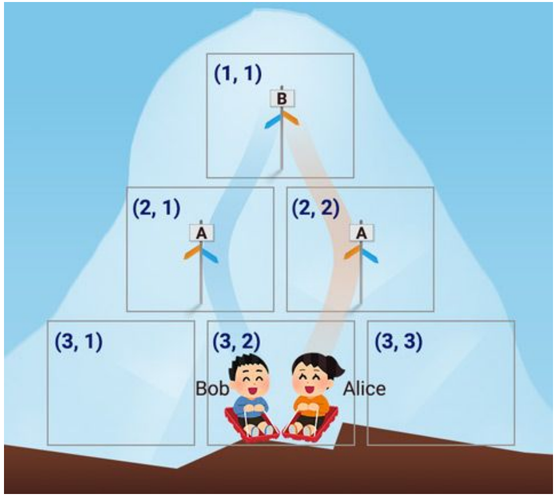
Problem Restatement

Given triangular grid of size N

Put down signposts such that

Alice: (1, 1) → (N, X), Bob: (1, 1) → (N, Y)

	Type-A signpost	Type-B signpost
Alice	Left (i + 1, j)	Right (i + 1, j + 1)
Bob	Right (i + 1, j + 1)	Left (i + 1, j)
Quantity	A	B



3 2 2 2 2	YES 2 2 1 2 2 1 1 1

Statistics

Attempts	Max	Mean	Std Dev
53	100	32.716	28.838

Subtasks				
5: 44	23: 40	26: 7	34: 10	12: 6

First solved by **mtyeung1** at 1:29

香港電腦奧林匹克競賽
Hong Kong Olympiad in Informatics

SUBTASK

For all cases:
$2 \le N \le 10^5$
$0 \le A, B \le 2 \times 10^5$
$1 \le X \le Y \le N$

	Points	Constraints
1	5	$N = 3$
2	23	$A = B = 2 \times N$ $2 \le N \le 30$
3	26	$2 \le N \le 30$
4	34	$X < Y$
5	12	No additional constraints

Subtask 1

Subtask 1 (5%): N = 3

- Careful case handling and implementation

OR

- Exhaustion

Subtask 1

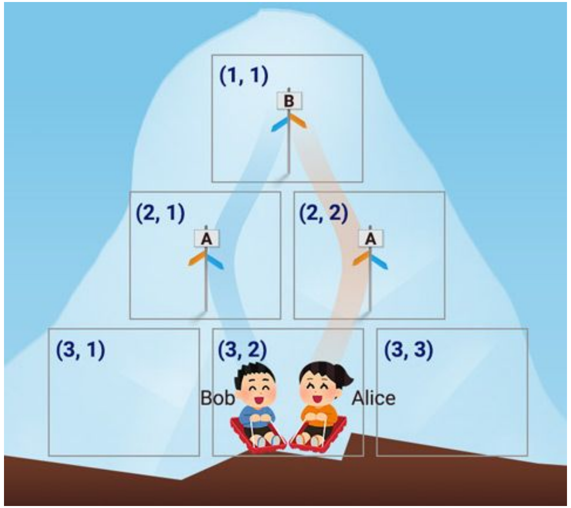
Subtask 1 (5%): N = 3

- Careful case handling and implementation
- There are only 6 types of X-Y pair
  - For X = Y = 1 or X = Y = 3, output impossible
  - For X = 1, Y = 2 or X = 2, Y = 3, we need to use 2 type-A signposts and 1 type-B signpost
  - For X = 1, Y = 3, we need to use 3 type-A signposts

Subtask 1

Subtask 1 (5%): N = 3

- Careful case handling and implementation
- There are only 6 types of X-Y pair
  - For X = Y = 2, there are two ways:
    - Use 2 type-A signposts and 1 type-B signpost
    - Use 1 type-A signpost and 2 type-B signposts
- Check whether the number of signposts given for each type is enough



Subtask 1

Subtask 1 (5%): N = 3

- Exhaustion

- For each cell, exhaust the state (empty, put type-A signpost or put type-B signpost) and check whether one of the constructions fulfil the constraints

Subtask 2

Subtask 2 (23%): $A = B = 2N$; $2 \le N \le 30$

- For each move, Alice or Bob will move one unit downward
- At most $(2N - 2)$ signposts are needed for both Alice and Bob to move to row N
- Just construct two separate paths

Subtask 2

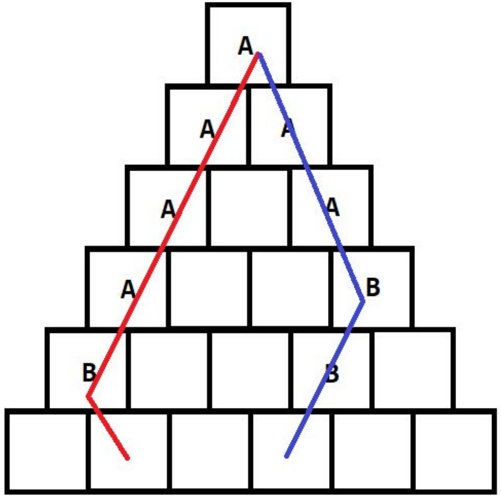
Subtask 2 (23%): A = B = 2N; 2 ≤ N ≤ 30

- Actually, the number of type-A and type-B signposts needed in Alice's and Bob's paths is fixed, and can be calculated easily
  - Number of type-A signposts needed in Alice's path = N - X
  - Number of type-B signposts needed in Alice's path = X - 1
  - Number of type-A signposts needed in Bob's path = Y - 1
  - Number of type-B signposts needed in Bob's path = N - Y

Subtask 2

- With the information above, you can place the appropriate signposts on their paths separately, as long as no more than one signpost is placed in a single cell (Avoid collision of path)

- One of the construction is to put all type-A signposts first, followed by type-B signposts
- Under this construction, except the first cell (1, 1) and the last cell (if X = Y) in their paths, they do not share the same cell



Subtask 3

Subtask 3 (26%): $2 \leq N \leq 30$

- Dynamic Programming
- dp[i][x][y][a][b] represents in the i-th line, whether we can use a type-A signposts and b type-B signposts to let Alice and Bob reach (i, x) and (i, y) separately
- We can let Alice and Bob move from the (i-1)-th line to the i-th line by putting some signposts in the (i-1)-th line

Subtask 3

- By using 2 type-A signposts (or just 1 if x = y - 1), we can transfer from dp[i - 1][x][y - 1][a - 2 (a - 1 *if x = y - 1*)][b] to dp[i][x][y][a][b]
- By using 2 type-B signposts (or just 1 if x - 1 = y), we can transfer from dp[i - 1][x - 1][y][a][b - 2 (b - 1 *if x - 1 = y*)] to dp[i][x][y][a][b]
- By using 1 type-A signpost and 1 type-B signpost, we can transfer from either dp[i - 1][x - 1][y - 1][a - 1][b - 1] or dp[i - 1][x][y][a - 1][b - 1] to dp[i][x][y][a][b]

Subtask 3

- Use backtracking to find the exact path
- Time complexity: O(N^5)

- Be careful about the large A, B in the input

Subtask 3

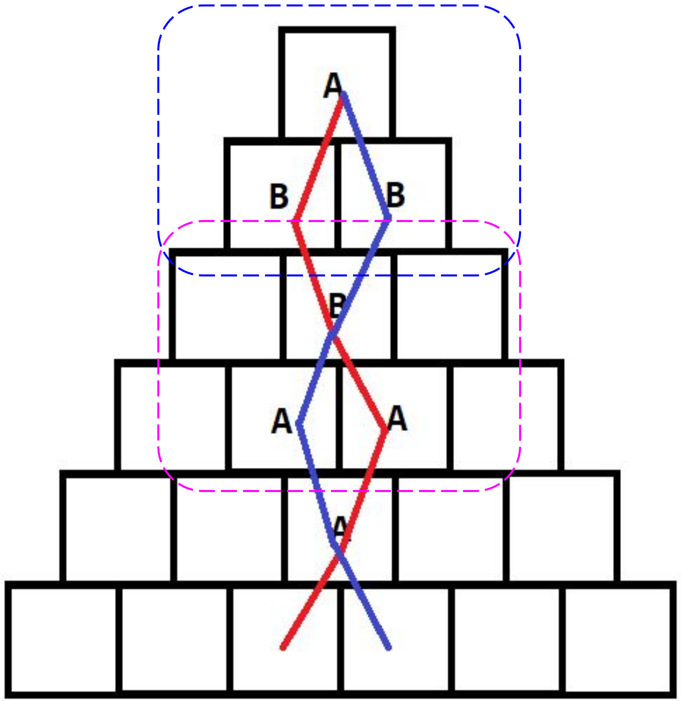
- There is a O(N^4) dp, where dp[i][x][y][a] stores the minimum number of type-B signposts needed to let Alice and Bob reach (i, x) and (i, y) separately by using a type-A signposts

- The idea is similar to the previous one

Subtask 4

Subtask 4 (34%): X < Y

- Construct path where both Alice and Bob stay close together at first, and separated when there are enough signposts left
- We can use 2 type-A signposts and 1 type-B signpost, or 1 type-A signpost and 2 type-B signposts to let both Alice and Bob move from (i, j) to (i + 2, j + 1)
- Such a movement can save 1 type-A or type-B signpost



Subtask 4

Number of type-A signposts needed in Alice's path = N - X

Number of type-B signposts needed in Alice's path = X - 1

Number of type-A signposts needed in Bob's path = Y - 1

Number of type-B signposts needed in Bob's path = N - Y

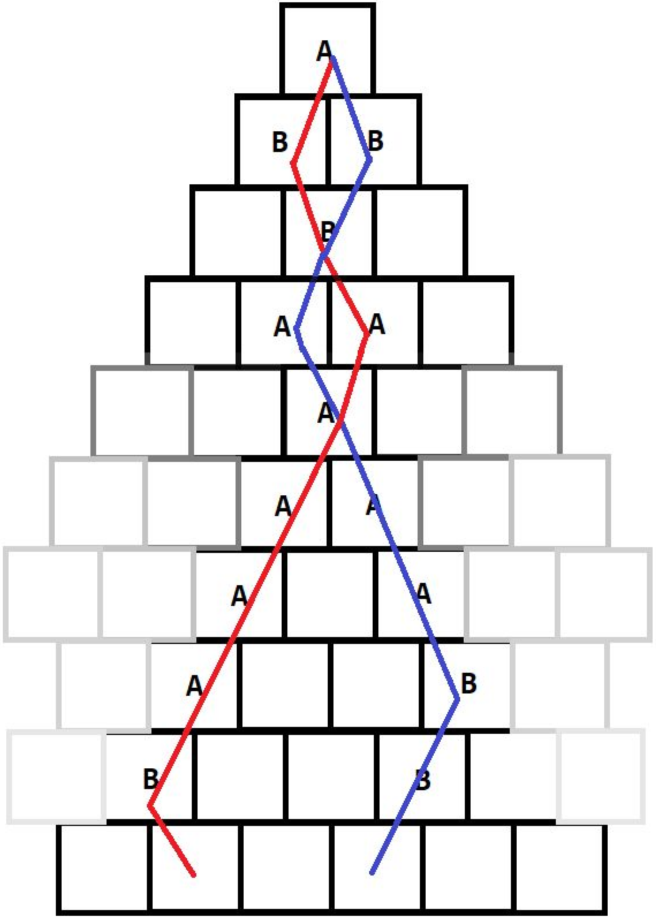
- Alice should move (X - 1) times to the right (using type-B signposts) and Bob should move (N - Y) times to the left (using type-B signposts)

- Maximum number of signposts can save = min {X - 1, N - Y}

Subtask 4

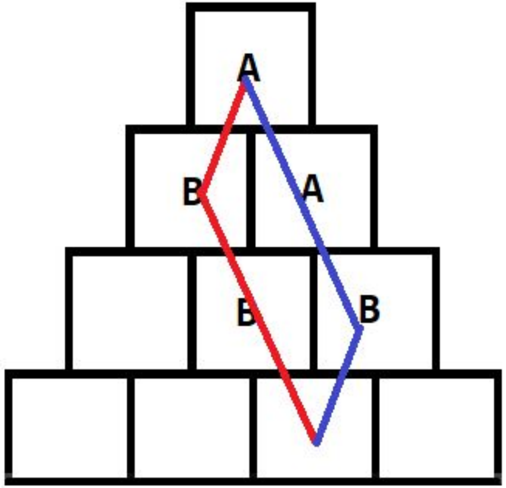
- After moving together, just construct their paths to their destinations separately by using the construction similar to subtask 2 (put all type-A signposts followed by type-B signposts)

- Time complexity: O(N)



Full Solution

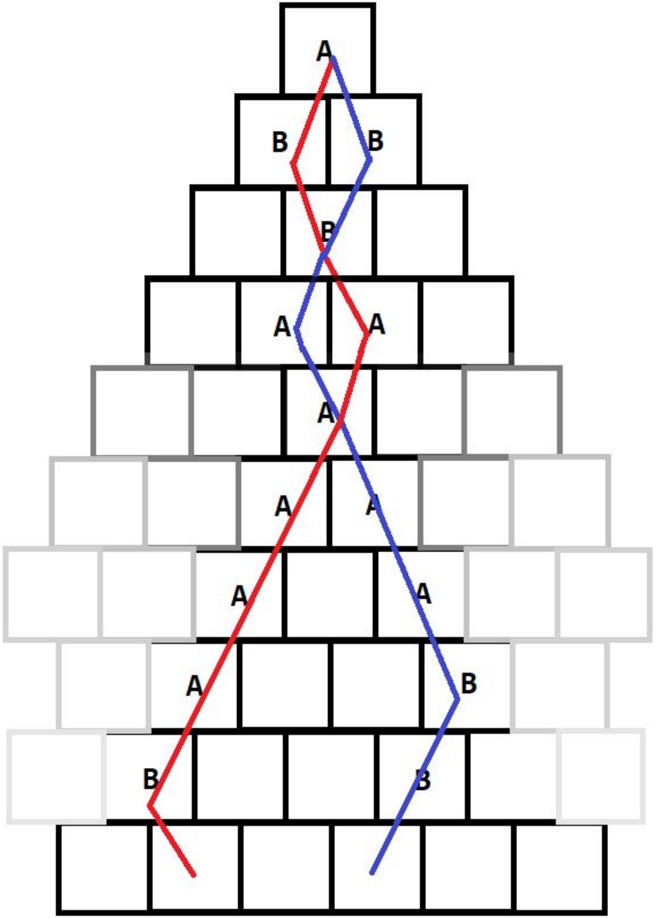
- The previous solution does not work when X = Y
- Need to handle some troublesome edge cases
- First, we need to reserve 1 right move for Alice and 1 left move for Bob to let them move to the same cell at last
- Maximum number of signposts can save becomes min {X - 1, N - Y} **- 1**



Full Solution

- We used 1 type-A signpost to separate them after they move together in the previous subtask

- But this will cause some problem when type-A signposts are not enough

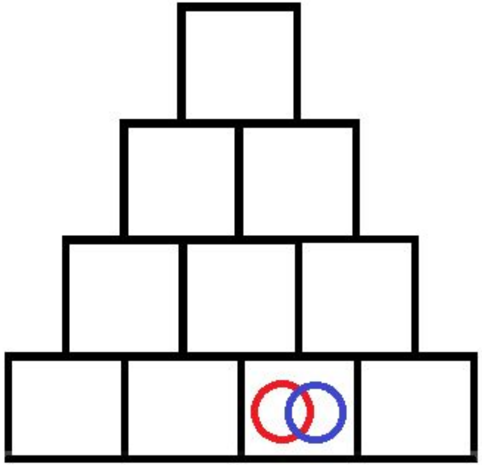


Full Solution

- Example:

  4 3 2

  3 3

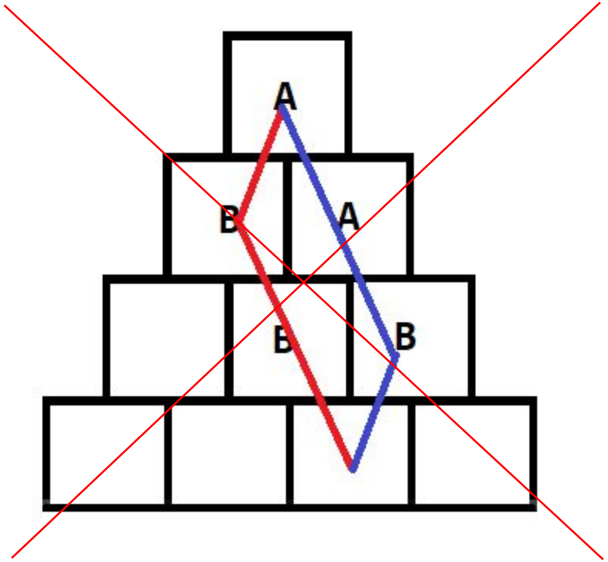


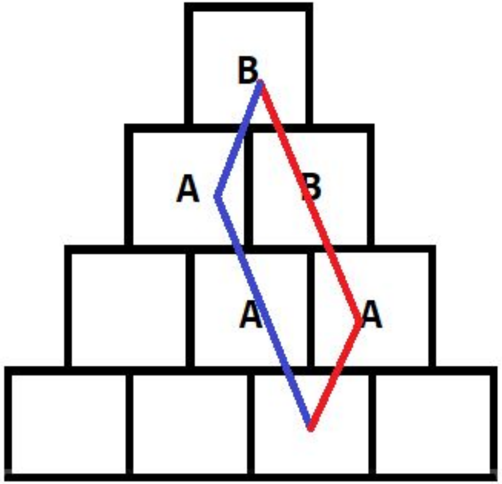
Full Solution

- Example:

  4 3 2

  3 3



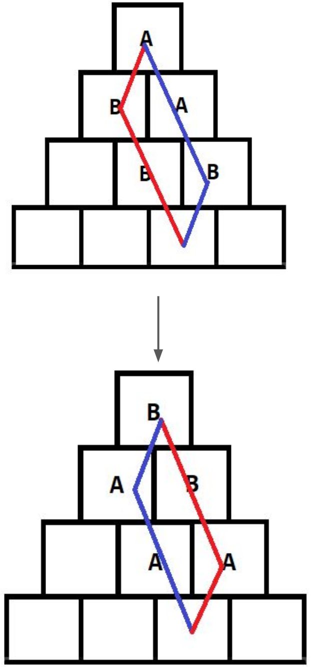


Full Solution

- As they will move to the same cell at last, we actually can use 1 type-B signpost instead

- You can special handle this case when number of type-A signposts is not enough

- Time complexity: O(N)



Full Solution

- Another way is to swap Alice and Bob (X <= Y still holds) and do the same problem again
- Then we can use type-A signposts as type-B signposts, and use type-B signposts as type-A signposts
- All situations are covered as we try to reserve 1 type-A signpost or 1 type-B signpost at first
- Time Complexity: O(N)

- Actually this is a way to solve the X > Y cases (not included in this problem)