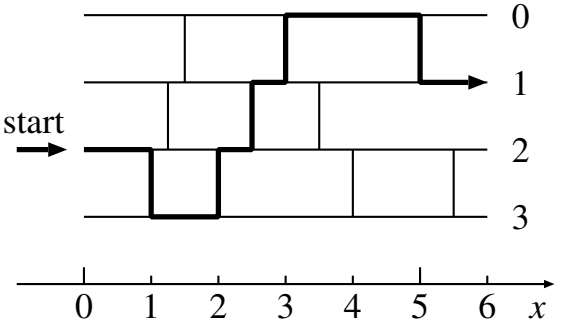
アルゴリズム基礎

2

下図のようにあみだくじを横向きに描き, 左側の与えられたスタート位置から始めて, 右側の何番にたどり着くかを判定したい. (あみだくじでは, 横線を左から右にたどり, 縦線との交点に来たらその縦線をたどって隣の横線に渡る. 渡り終えた点から横線を右へ進み, 以下同様のルールで右端までたどる.) 横線 (水平に引く) に上から順に $0, 1, \ldots, m$ と番号をつけ, 横線 $k-1$ と k $(k = 1, \ldots, m)$ の間の縦線の本数を n_k (≥ 1) とする. また, これら縦線の位置は, 図の右の表のように配列 $A_k(i)$ $(i = 1, \ldots, n_k)$ に縦線の位置の x 座標が左から順に格納されているものとする (縦線は必ず垂直に引く). 縦線の位置座標は全て異なる (つまり $k \neq l$ あるいは $i \neq j$ ならば $A_k(i) \neq A_l(j)$) ものとする. スタート位置の x 座標を 0 とし, 任意の k と i に対して $A_k(i) > 0$ とする. また, 縦線の総数を $n = n_1 + \cdots + n_m$ と記す.



i	1	2	3	4
$A_1(i)$	1.5	3.0	5.0	
$A_2(i)$	1.2	2.5	3.5	
$A_3(i)$	1.0	2.0	4.0	5.5

あみだくじの上で, スタート位置か, ある縦線をたどり終わった直後の点を, 横線 k 上の現在位置 x' とする. あみだくじをたどるには, このような点から次にたどるべき縦線を配列から読み取る必要がある. たとえば,

$$1 \leq k \leq m-1 \text{ かつ } x' < \min\{A_k(n_k), A_{k+1}(n_{k+1})\} \tag{1}$$

であるとき, 次にたどるべき縦線は, 配列 A_k において $A_k(i_k) > x'$ を満たす最小の i_k と, 配列 A_{k+1} において $A_{k+1}(i_{k+1}) > x'$ を満たす最小の i_{k+1} に対して, $A_k(i_k)$ と $A_{k+1}(i_{k+1})$ の小さい方である.

配列のひとつの要素へのアクセスや基本演算は全て O(1) 時間で可能であり, 座標や線の番号などの数値は全て O(1) の記憶領域に格納できるものとする. 以下, 記憶領域量を議論するときには, 入力データ (縦線の位置を表す配列 A_k 等) を格納する領域を含めず, それ以外の部分, すなわち計算に用いる変数等を格納する領域のみを考える. なお, 入力データを格納している領域の書き換えは許さない.

(i) 横線 k 上の現在位置 x' が条件 (1) を満たす場合について, 次にたどるべき縦線を配列から読み取るためのルールを上に述べた. それ以外のすべての場合に対するルールを書き下せ. そのような縦線が存在しない場合もあることに注意せよ.

(ii) 横線 k 上の現在位置 x' から, 次にたどるべき縦線を見つけるのに, 毎回 $A_k(i)$ と $A_{k+1}(i)$ (k が 0 か m のときは一方のみ) をそれぞれ $i = 1, 2, 3, \ldots$ と左から順にたどるという単純な方法を用いるとき, このアルゴリズムがスタート位置から右端にたどりつくまでに要する計算時間を述べ, その理由を簡潔に説明せよ.

(iii) 記憶領域を O(1) しか利用できないものとする. このような条件のもとで $O(n \log n)$ 時間で動作するアルゴリズムを与えよ.

(iv) 記憶領域に関する制約がないとき, 上の (ii) の方法よりも高速なアルゴリズムで記憶領域量が出来るだけ少ないものを与えよ. さらに, その時間量と記憶領域量を評価せよ.