• (p_l, q_r)搜索算法

Input: Y_L, Y_R, d

Output: *result*

1. 扫描Y_L得到Q_L中左临界区点，保持y坐标排序，得到Z_L
2. 扫描Y_R得到Q_R中左临界区点，保持y坐标排序，得到Z_R
3. *result*=null;
4. *top-R*=0;
5. **for** *top-L*=0 to Z_L.*length*-1
6.   **while** Z_L[*top-L*].y > Z_R[*top-R*].y + d
7.   **if** *top-R*=Z_R.*length*-1
8.   **return** *result*;
9.   *top-R* ← *top-R*+1;

纵坐标就差了比d多，就跳过

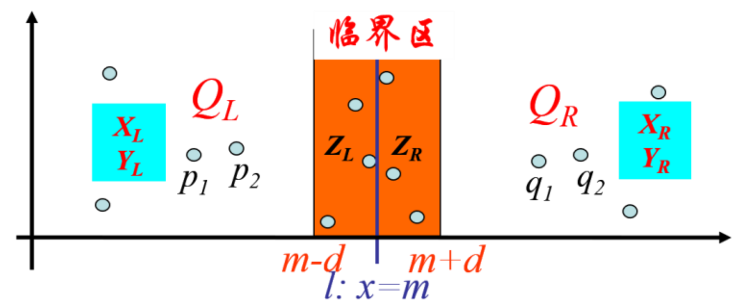
10.   **if** Z_L[*top-L*].y ≥ Z_R[*top-R*].y – d **and** Z_L[*top-L*].y ≤ Z_R[*top-R*].y + d
11.   **for** i=0 **to** *min*{5, Z_R.*length* –*top-R* – 1}
12.   **if** *dist*(Z_L[*top-L*], Z_R[*top-R*+*i*])<d

在R的d为半径的圆里

13.   *result*=(Z_L[*top-L*], Z_R[*top-R*+*i*]);
15. d=*dist*(Z_L[*top-L*], Z_R[*top-R*+*i*]);

缩小范围

16. **return** *result*;



$m=5, d=4$

$Result=null$

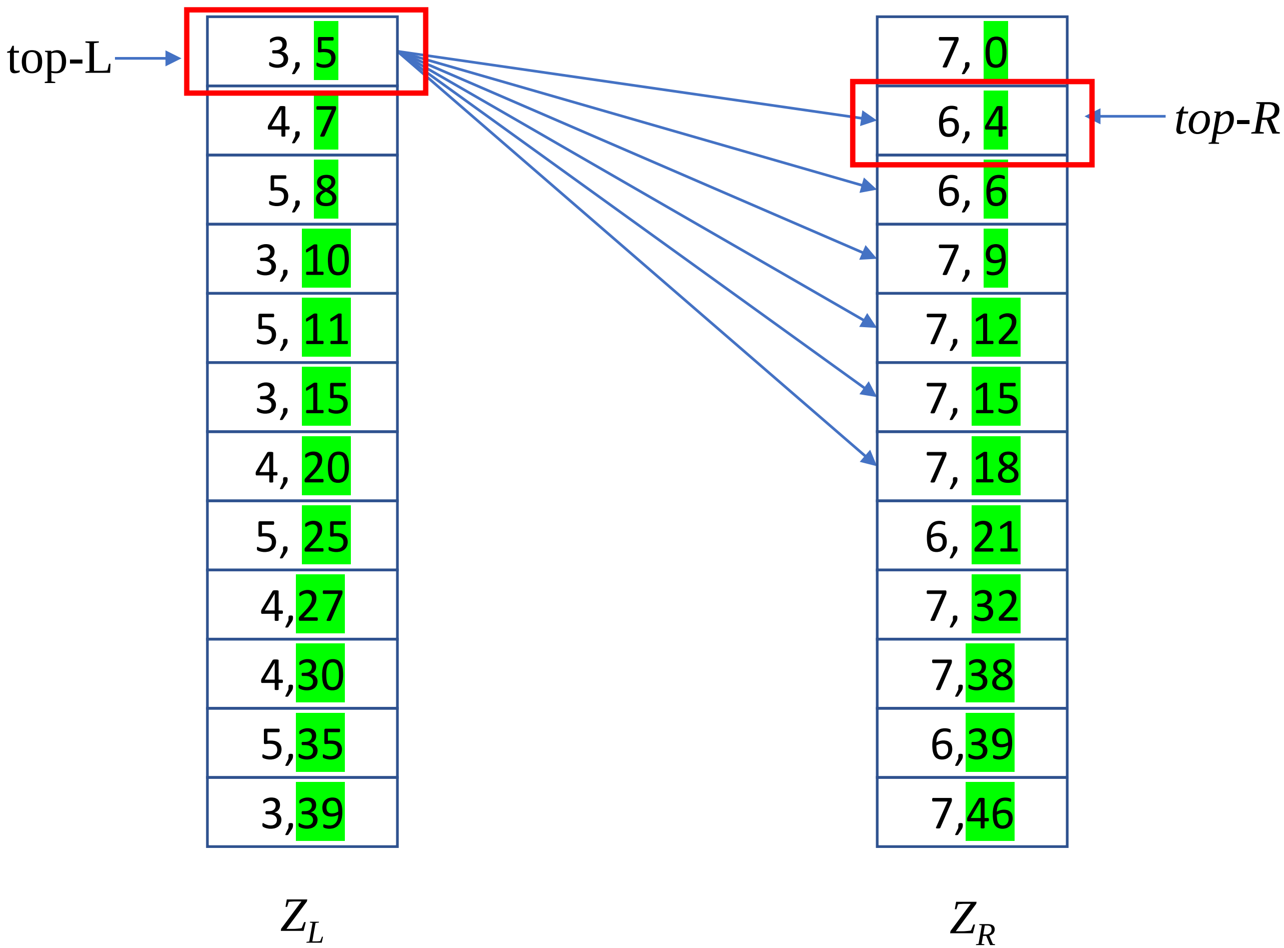
top-L → Z_L	Z_R ← top-R
3, 5	7, 0
4, 7	6, 4
5, 8	6, 6
3, 10	7, 9
5, 11	7, 12
3, 15	7, 15
4, 20	7, 18
5, 25	6, 21
4,27	7, 32
4,30	7,38
5,35	6,39
3,39	7,46

Z_L Z_R

数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y > Z_R[top\text{-}R].y + d$, 即5>0+4, Z_L中后续元素y值都不比$Z_L[top\text{-}L].y$小，因此它们与$Z_R[top\text{-}R]$的距离必然大于d, $top\text{-}R$增加1

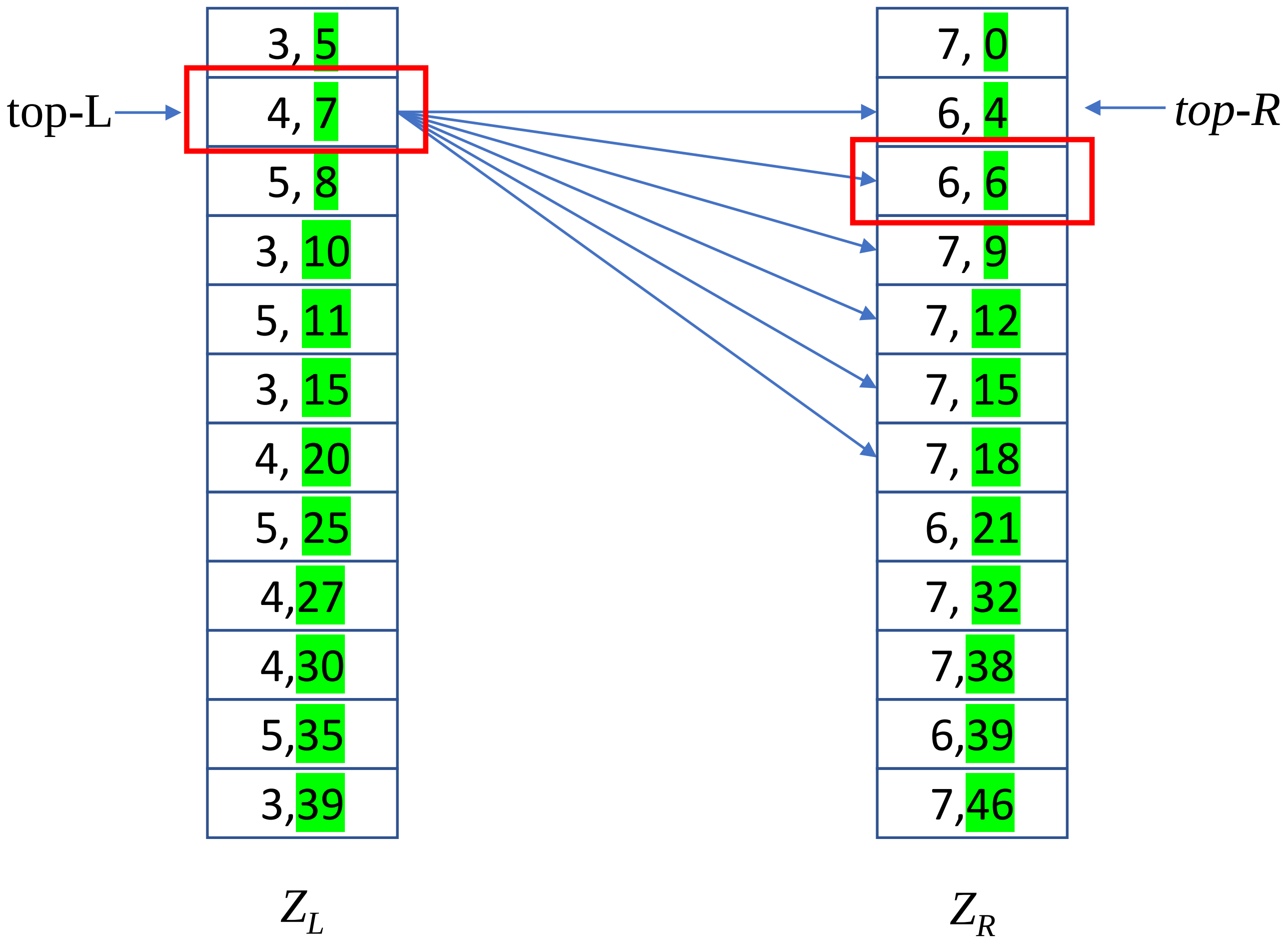
m=5, d=4更新为$d = \sqrt{10}$



数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y \geq Z_R[top\text{-}R].y - d$ 并且 $Z_L[top\text{-}L].y \leq Z_R[top\text{-}R].y + d$, 为当前$Z_L[top\text{-}L]$ 连续检验最多6个Z_R中的元素（如果Z_R中有足够多的剩余元素）, 随后$top\text{-}L$增长1, $top\text{-}R$不变

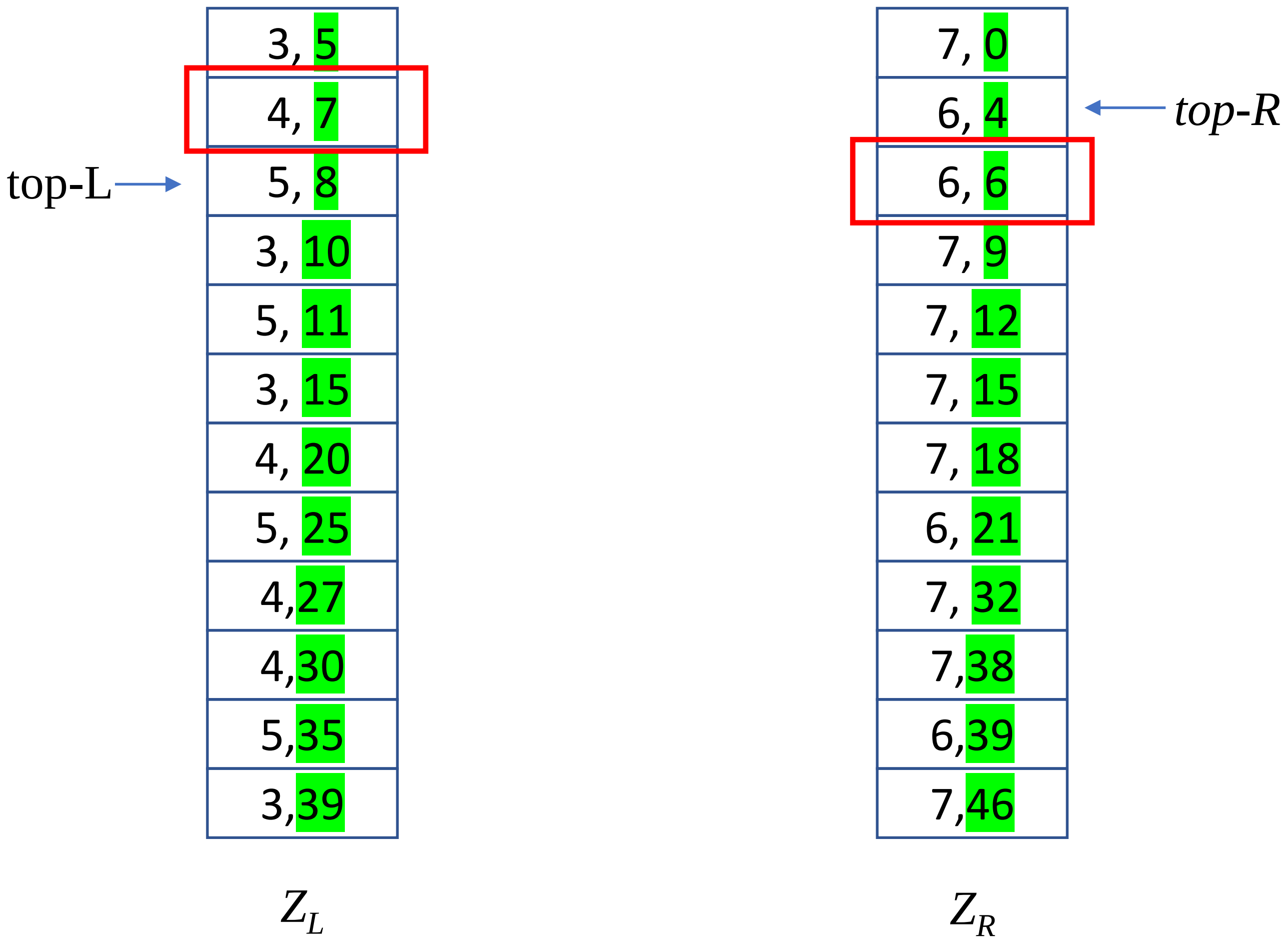
$m=5$, $d=\sqrt{10}$ 更新为 $d=\sqrt{5}$



数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y \geq Z_R[top\text{-}R].y - d$ 并且 $Z_L[top\text{-}L].y \leq Z_R[top\text{-}R].y + d$，为当前 $Z_L[top\text{-}L]$ 连续检验最多6个 Z_R 中的元素（如果 Z_R 中有足够多的剩余元素），随后 $top\text{-}L$ 增长1，$top\text{-}R$ 不变

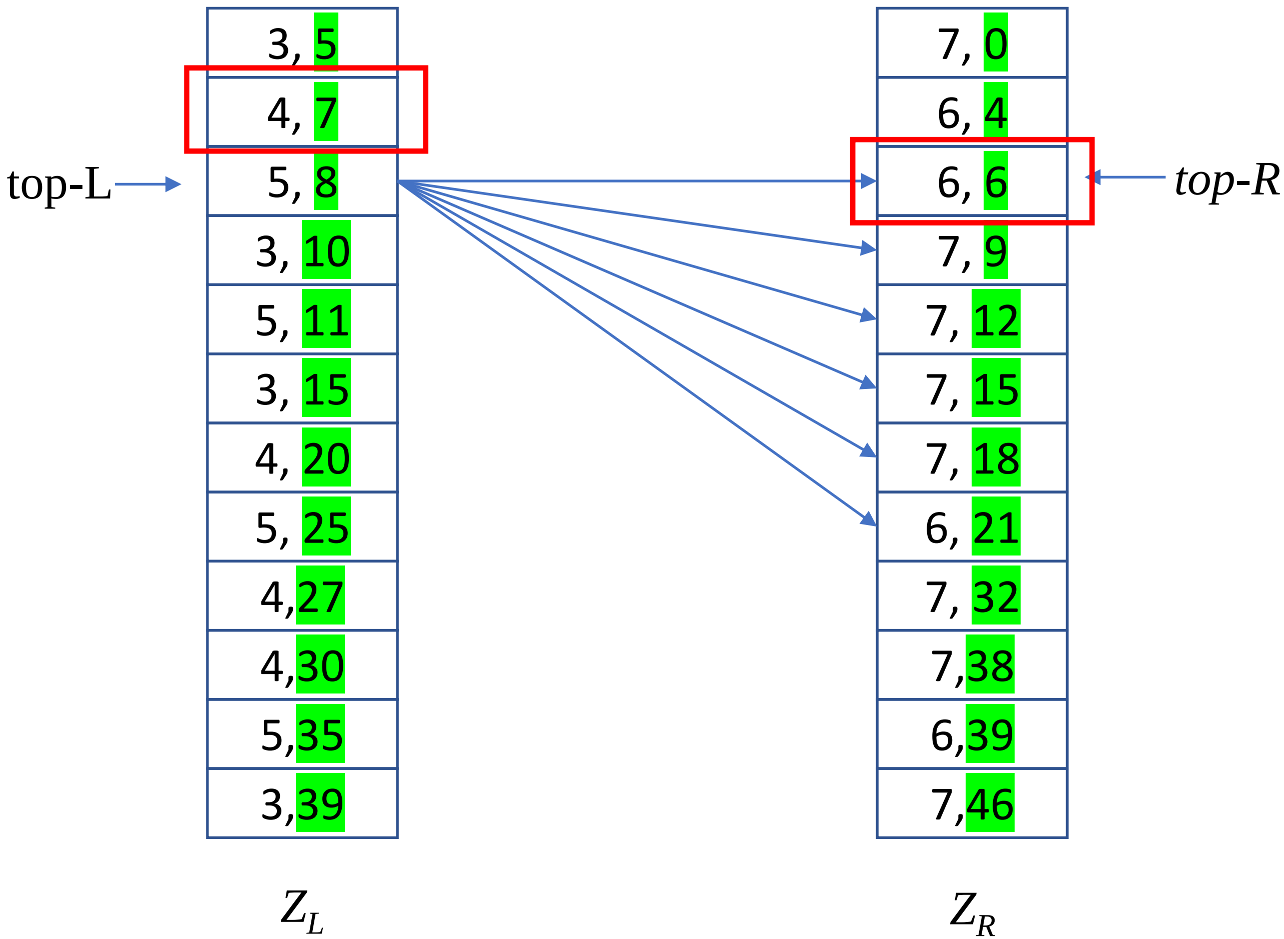
$m=5, d=\sqrt{5}$



数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y > Z_R[top\text{-}R].y + d$, Z_L中后续元素y值都不比$Z_L[top\text{-}L].y$小，因此它们与$Z_R[top\text{-}R]$的距离必然大于d, $top\text{-}R$增加1

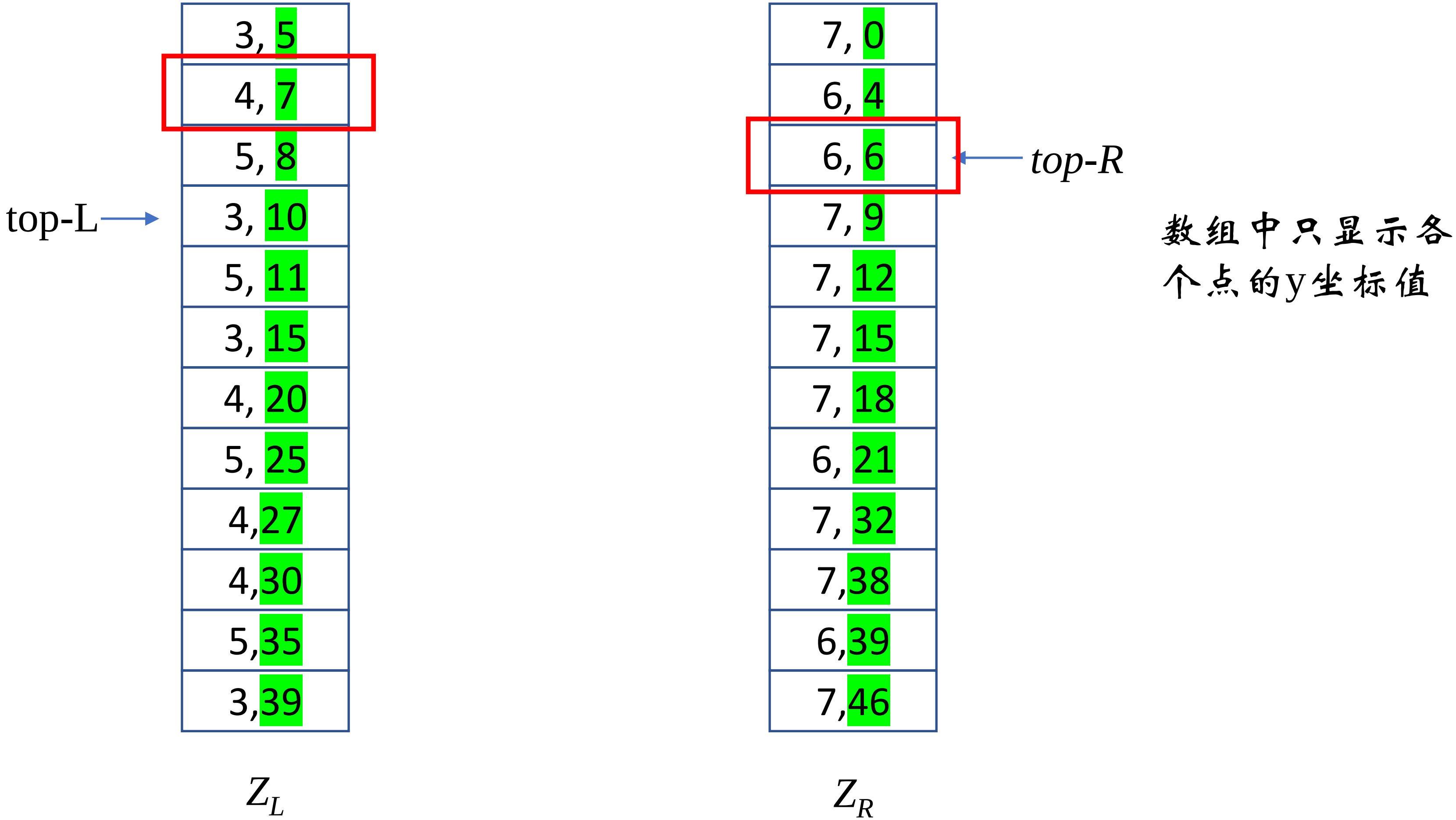
$m=5, d=\sqrt{5}$



数组中只显示各个点的y坐标值

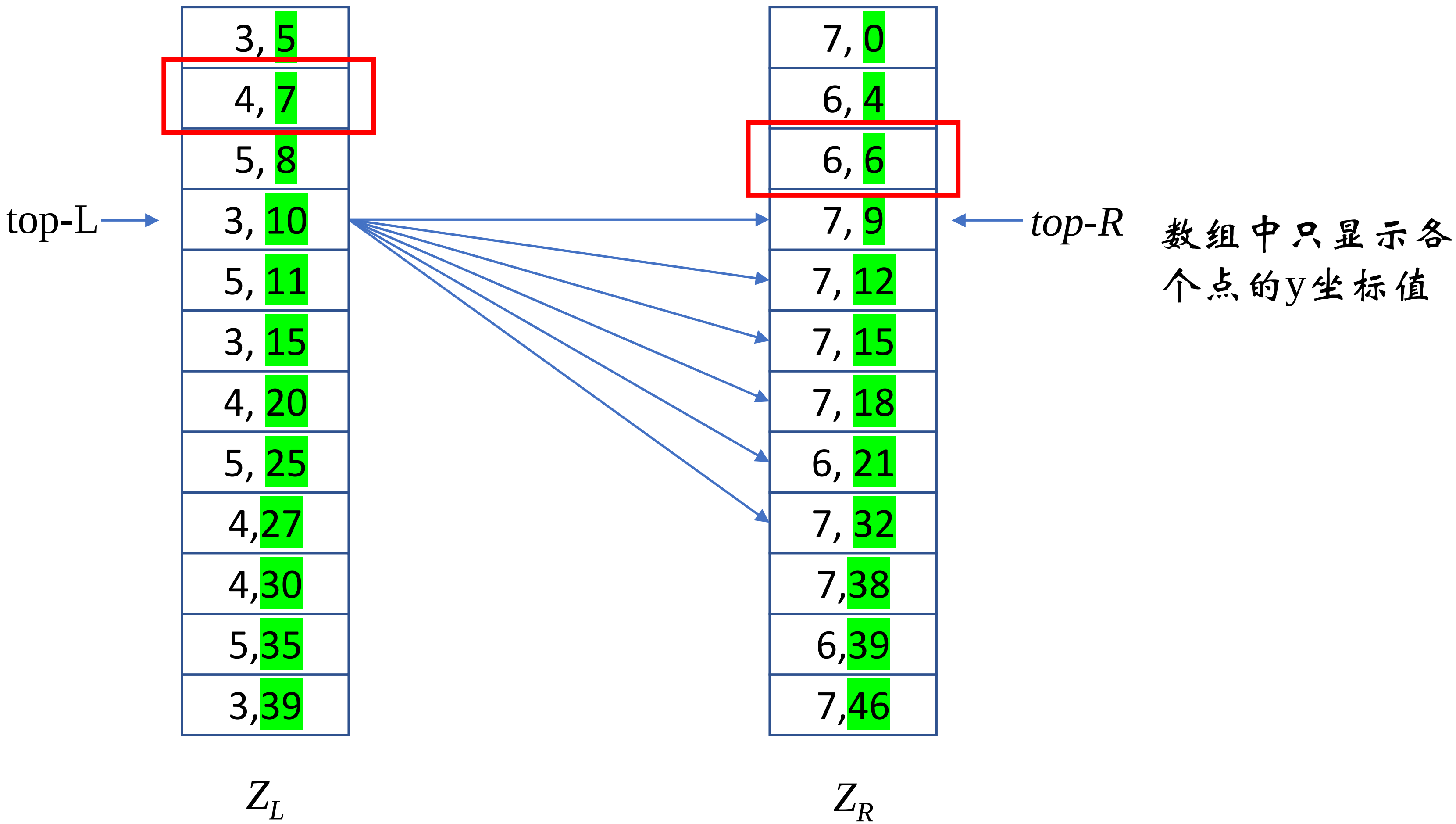
$Z_L[top\text{-}L].y \geq Z_R[top\text{-}R].y - d$ 并且 $Z_L[top\text{-}L].y \leq Z_R[top\text{-}R].y + d$, 为当前$Z_L[top\text{-}L]$ 连续检验最多6个Z_R中的元素（如果Z_R中有足够多的剩余元素），随后*top-L*增长1, *top-R*不变

$m=5, d=\sqrt{5}$



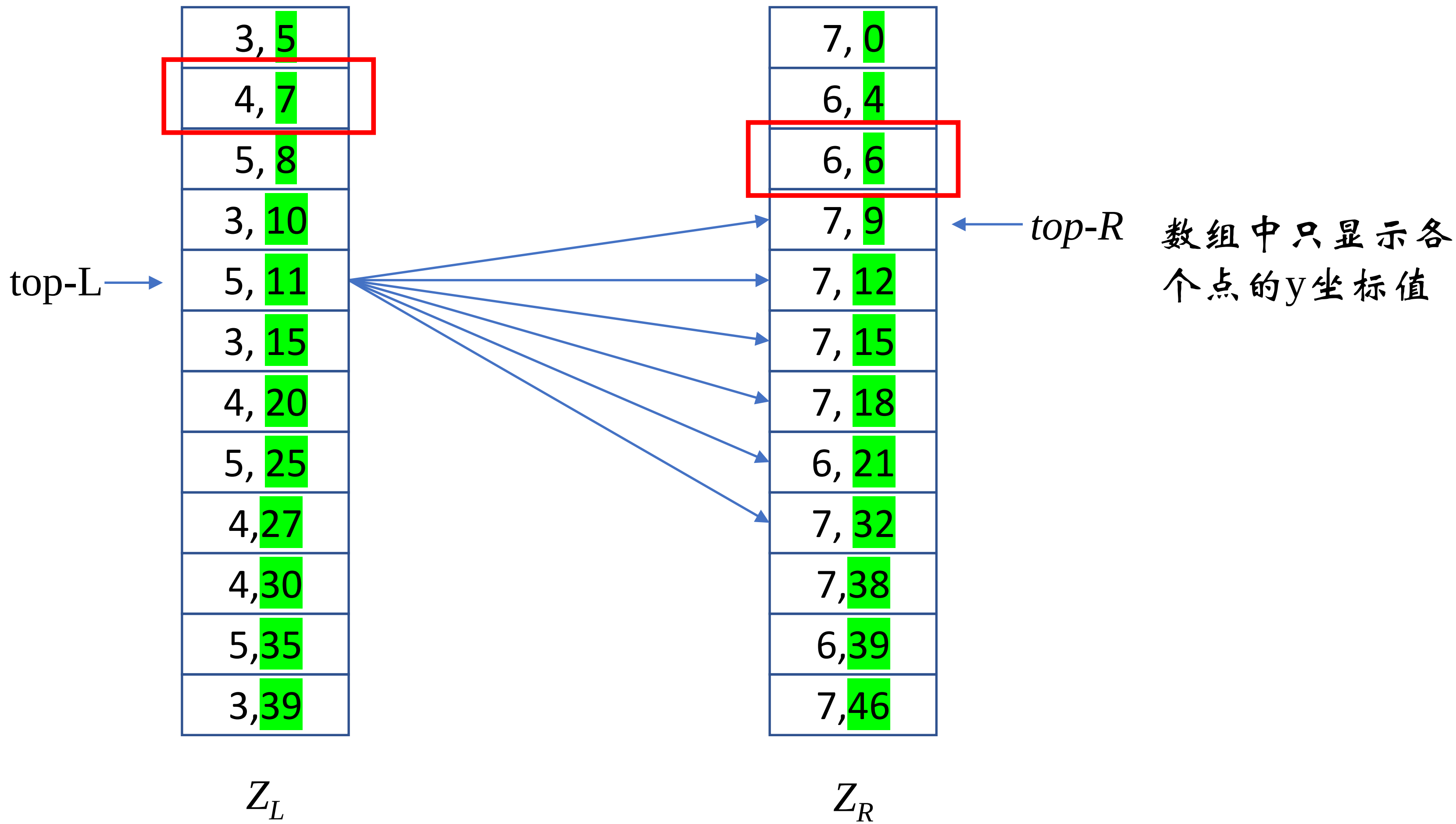
$Z_L[top\text{-}L].y > Z_R[top\text{-}R].y + d$, Z_L中后续元素y值都不比$Z_L[top\text{-}L].y$小，因此它们与$Z_R[top\text{-}R]$的距离必然大于d, $top\text{-}R$增加1

$m=5,\ d=\sqrt{5}$



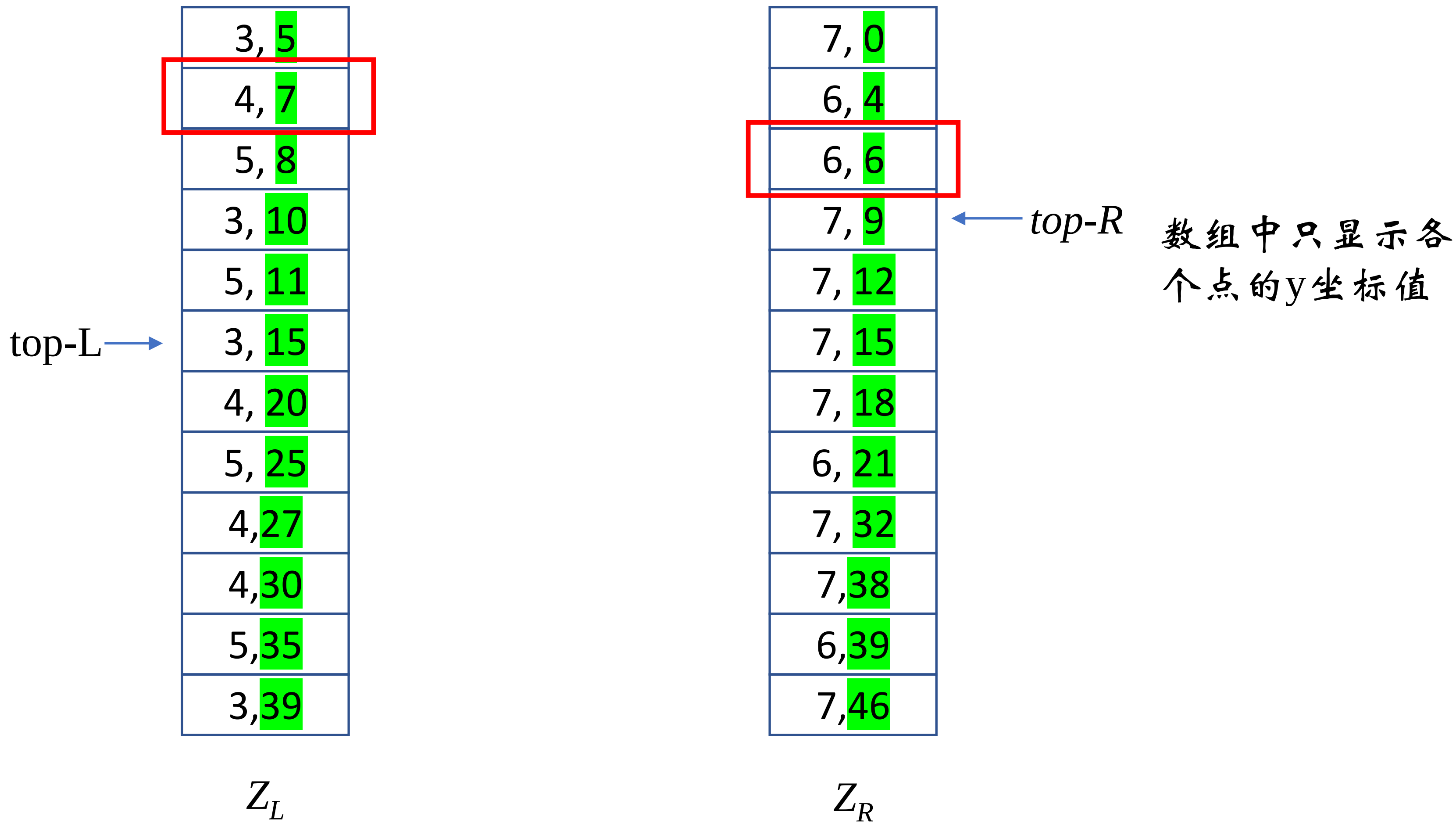
$Z_L[top\text{-}L].y \geq Z_R[top\text{-}R].y - d$ 并且 $Z_L[top\text{-}L].y \leq Z_R[top\text{-}R].y + d$, 为当前$Z_L[top\text{-}L]$ 连续检验最多6个Z_R中的元素（如果Z_R中有足够多的剩余元素），随后*top-L*增长1, *top-R*不变

$m=5,\ d=\sqrt{5}$



$Z_L[top\text{-}L].y \geq Z_R[top\text{-}R].y - d$ 并且 $Z_L[top\text{-}L].y \leq Z_R[top\text{-}R].y + d$, 为当前$Z_L[top\text{-}L]$ 连续检验最多6个Z_R中的元素（如果Z_R中有足够多的剩余元素），随后$top\text{-}L$增长1, $top\text{-}R$不变

m=5, d=$\sqrt{5}$



$Z_L[top\text{-}L].y > Z_R[top\text{-}R].y + d$, Z_L中后续元素y值都不比$Z_L[top\text{-}L].y$小，因此它们与$Z_R[top\text{-}R]$的距离必然大于d, $top\text{-}R$增加1

m=5, d=$\sqrt{5}$

Z_L	Z_R
3, 5	7, 0
4, 7	6, 4
5, 8	6, 6
3, 10	7, 9
5, 11	7, 12
3, 15	7, 15
4, 20	7, 18
5, 25	6, 21
4,27	7, 32
4,30	7,38
5,35	6,39
3,39	7,46

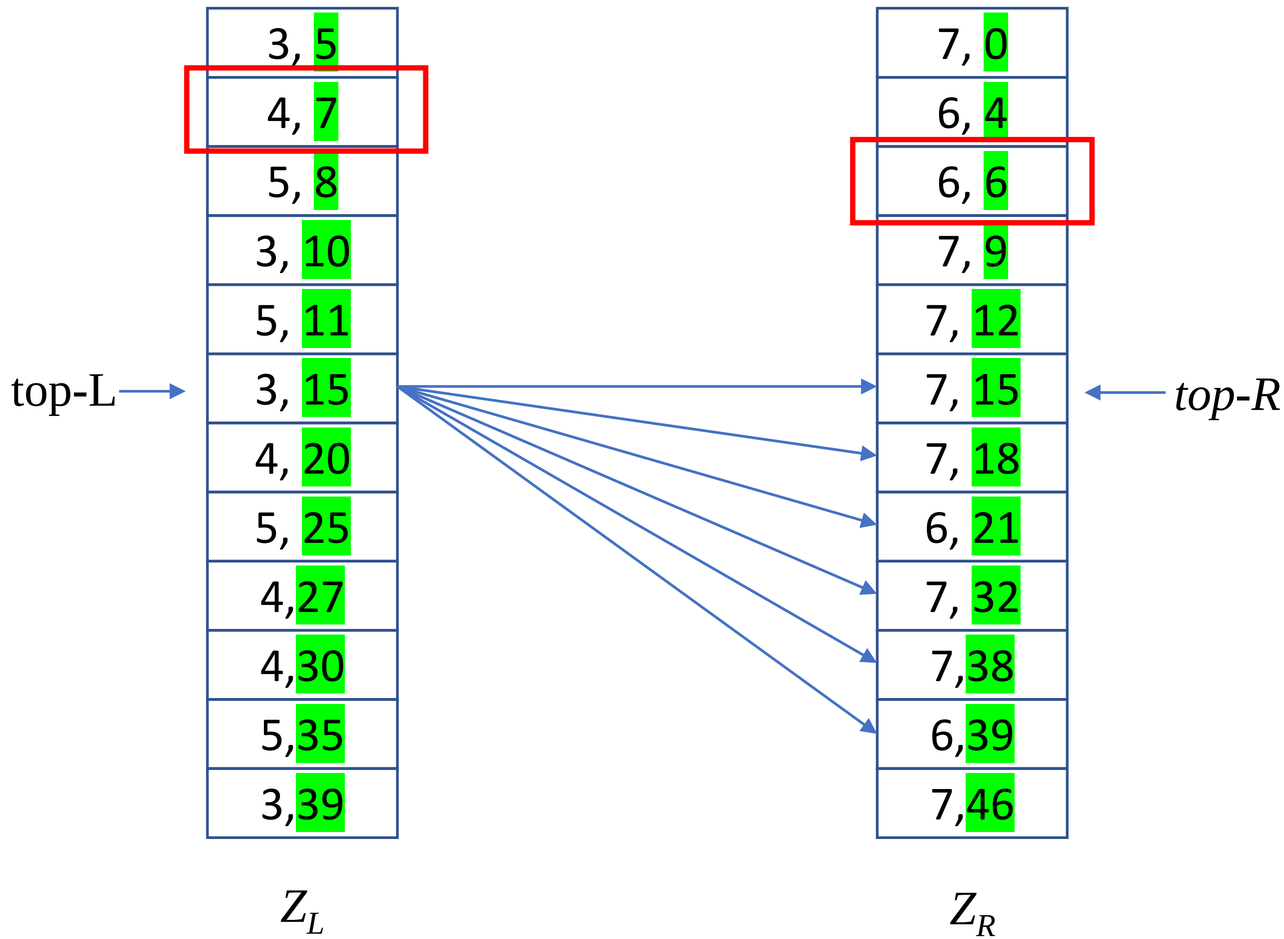
top-L → 3, 15

top-R ← 7, 12

数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y > Z_R[top\text{-}R].y + d$, Z_L中后续元素y值都不比$Z_L[top\text{-}L].y$小，因此它们与$Z_R[top\text{-}R]$的距离必然大于d, top-R增加1

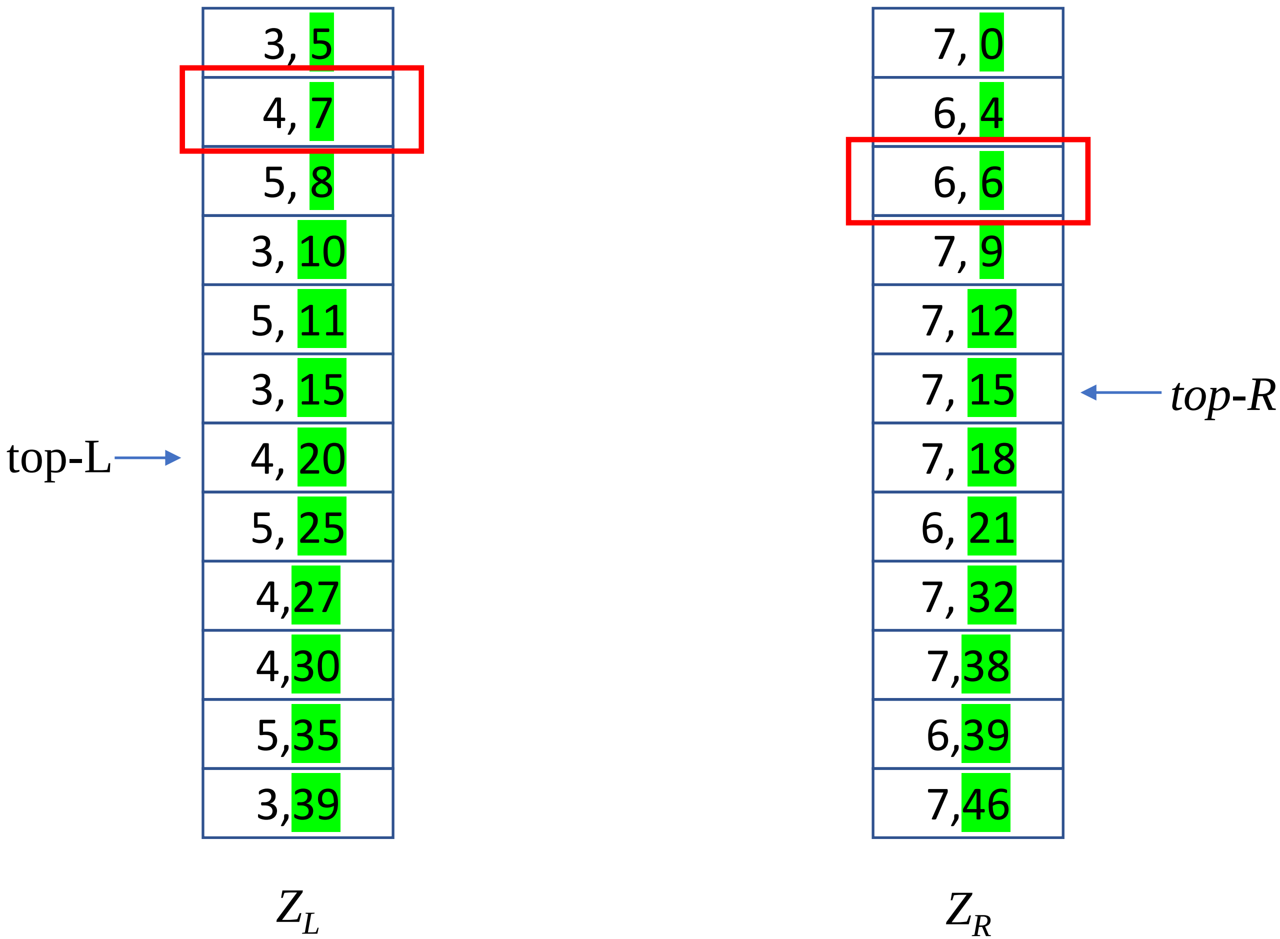
$m=5,\ d=\sqrt{5}$



数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y \ge Z_R[top\text{-}R].y - d$ 并且 $Z_L[top\text{-}L].y \le Z_R[top\text{-}R].y + d$, 为当前$Z_L[top\text{-}L]$ 连续检验最多6个Z_R中的元素（如果Z_R中有足够多的剩余元素），随后*top-L*增长1, *top-R*不变

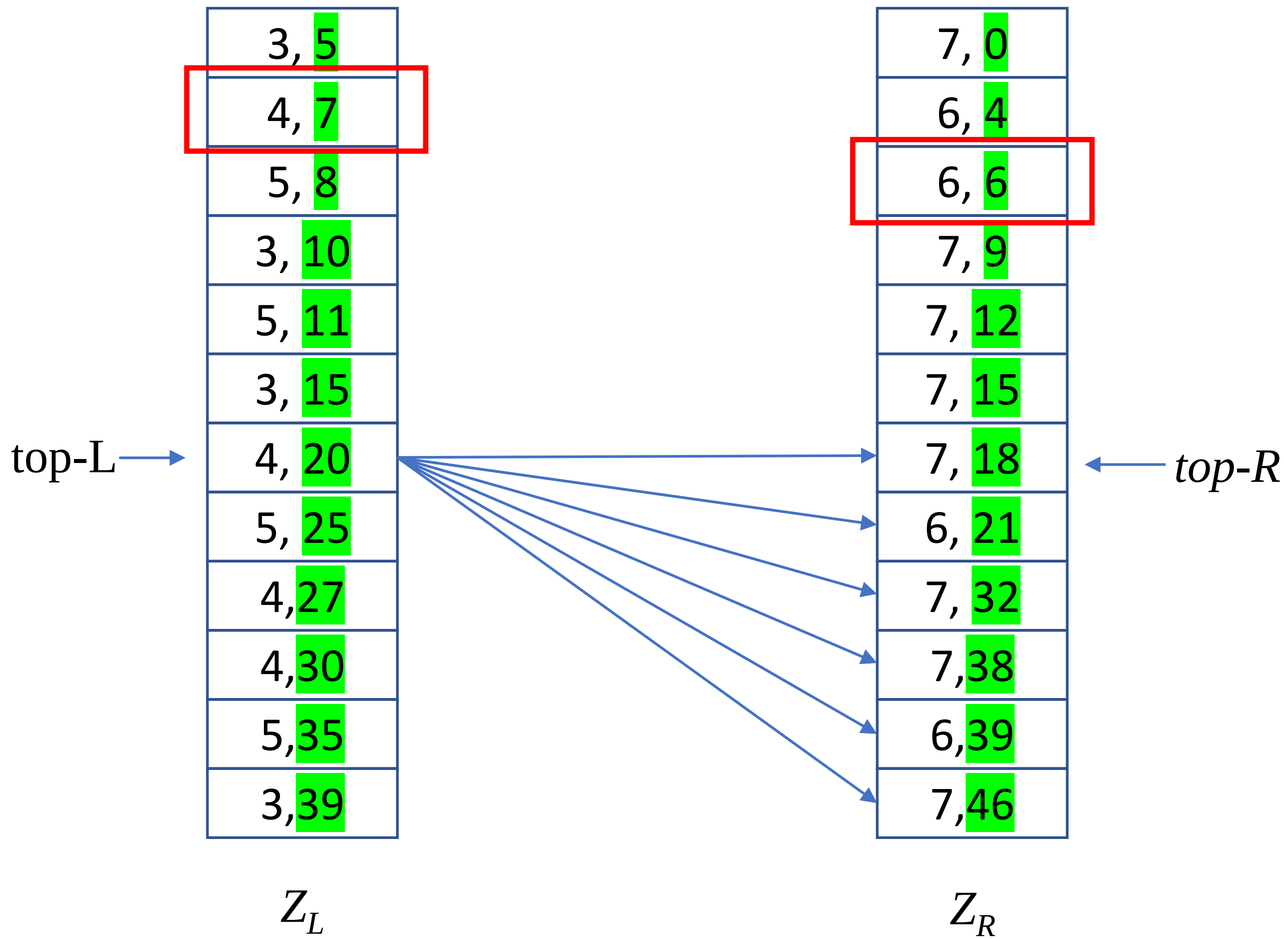
$m=5,\ d=\sqrt{5}$



数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y > Z_R[top\text{-}R].y + d$, Z_L中后续元素y值都不比$Z_L[top\text{-}L].y$小，因此它们与$Z_R[top\text{-}R]$的距离必然大于d, $top\text{-}R$增加1

$m=5,\ d=\sqrt{5}$



数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y \geq Z_R[top\text{-}R].y - d$ 并且 $Z_L[top\text{-}L].y \leq Z_R[top\text{-}R].y + d$, 为当前$Z_L[top\text{-}L]$ 连续检验最多6个Z_R中的元素（如果Z_R中有足够多的剩余元素），随后$top\text{-}L$增长1, $top\text{-}R$不变

$m=5,\ d=\sqrt{5}$

Z_L	Z_R
3, 5	7, 0
4, 7	6, 4
5, 8	6, 6
3, 10	7, 9
5, 11	7, 12
3, 15	7, 15
4, 20	7, 18 ← *top-R*
top-L → 5, 25	6, 21
4,27	7, 32
4,30	7,38
5,35	6,39
3,39	7,46

数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y > Z_R[top\text{-}R].y + d$, Z_L中后续元素y值都不比$Z_L[top\text{-}L].y$小，因此它们与$Z_R[top\text{-}R]$的距离必然大于d, $top\text{-}R$增加1

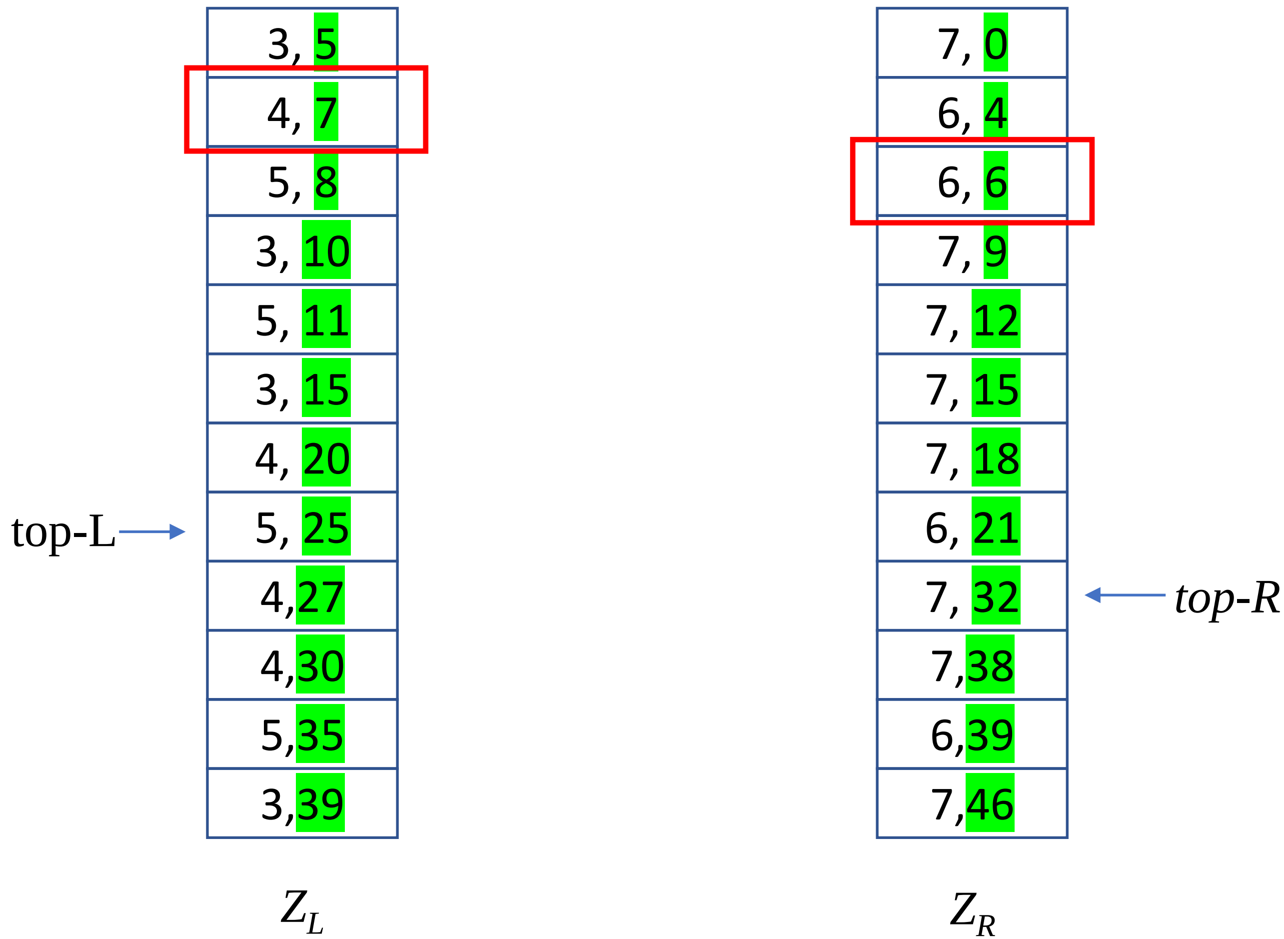
$m=5,\ d=\sqrt{5}$

	Z_L	Z_R	
	3, 5	7, 0	
	$\boxed{4, 7}$	6, 4	
	5, 8	$\boxed{6, 6}$	
	3, 10	7, 9	
	5, 11	7, 12	
	3, 15	7, 15	
	4, 20	7, 18	
top-L →	5, 25	6, 21	← *top-R*
	4,27	7, 32	
	4,30	7,38	
	5,35	6,39	
	3,39	7,46	

数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y > Z_R[top\text{-}R].y + d$, Z_L中后续元素y值都不比$Z_L[top\text{-}L].y$小，因此它们与$Z_R[top\text{-}R]$的距离必然大于d, $top\text{-}R$增加1

$m=5,\ d=\sqrt{5}$



数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y < Z_R[top\text{-}R].y - d$, Z_R中后续元素y值都不比$Z_R[top\text{-}R].y$小，因此它们与$Z_L[top\text{-}L]$的距离必然大于d, $top\text{-}L$增加1

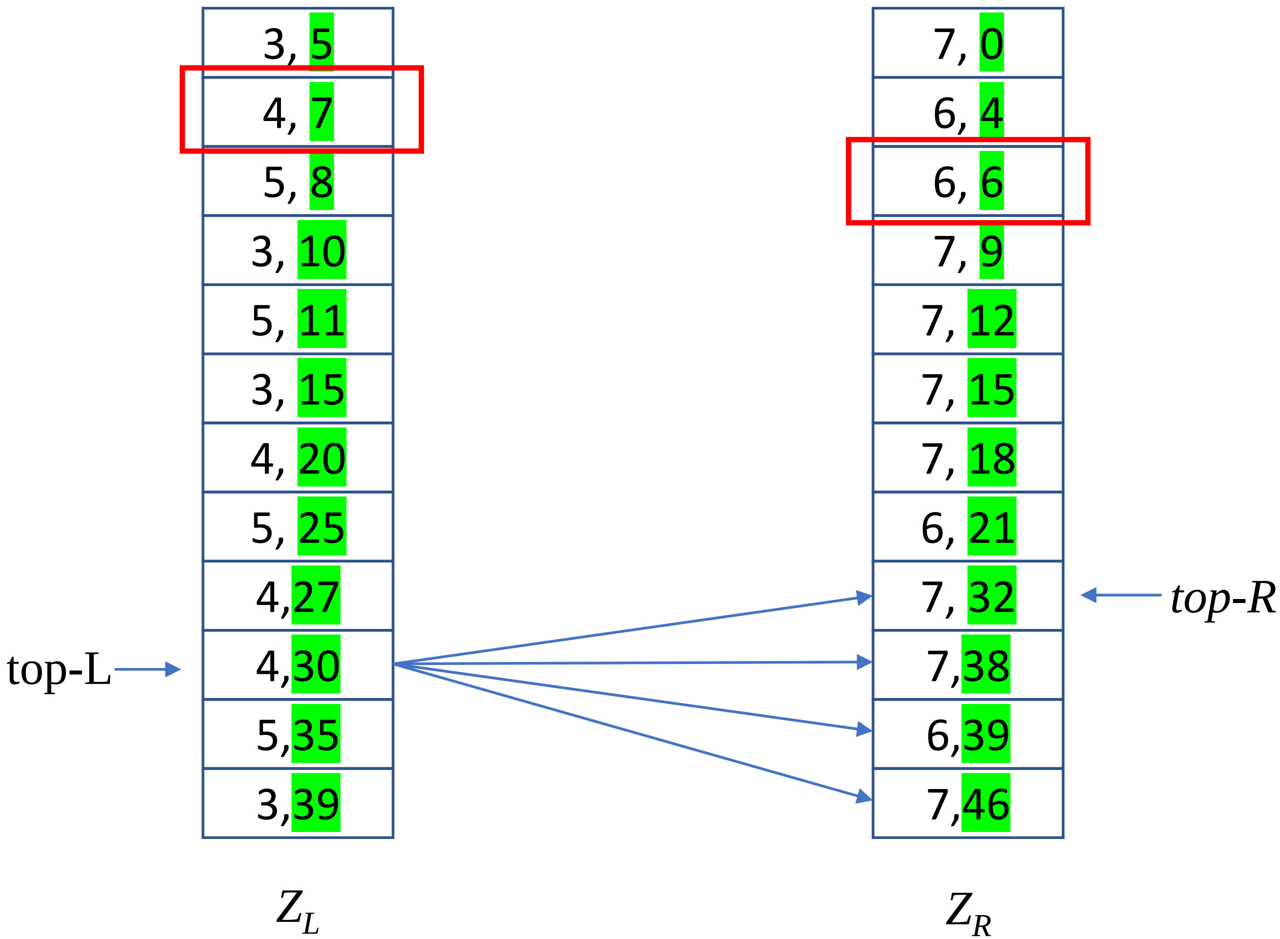
$m=5$, $d=\sqrt{5}$

	Z_L	Z_R	
	3, 5	7, 0	
	4, 7	6, 4	
	5, 8	6, 6	
	3, 10	7, 9	
	5, 11	7, 12	
	3, 15	7, 15	
	4, 20	7, 18	
	5, 25	6, 21	
top-L →	4,27	7, 32	← *top-R*
	4,30	7,38	
	5,35	6,39	
	3,39	7,46	

数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y < Z_R[top\text{-}R].y - d$, Z_R中后续元素y值都不比$Z_R[top\text{-}R].y$小，因此它们与$Z_L[top\text{-}L]$的距离必然大于d, $top\text{-}L$增加1

$m=5,\ d=\sqrt{5}$



数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y \ge Z_R[top\text{-}R].y - d$ 并且 $Z_L[top\text{-}L].y \le Z_R[top\text{-}R].y + d$, 为当前$Z_L[top\text{-}L]$连续检验最多6个$Z_R$中的元素（如果$Z_R$中有足够多的剩余元素），随后$top\text{-}L$增长1, $top\text{-}R$不变

m=5, d=$\sqrt{5}$

Z_L	Z_R
3, 5	7, 0
4, 7	6, 4
5, 8	6, 6
3, 10	7, 9
5, 11	7, 12
3, 15	7, 15
4, 20	7, 18
5, 25	6, 21
4,27	7, 32 ← *top-R*
4,30	7,38
top-L → 5,35	6,39
3,39	7,46

数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y > Z_R[top\text{-}R].y + d$, Z_L中后续元素y值都不比$Z_L[top\text{-}L].y$小，因此它们与$Z_R[top\text{-}R]$的距离必然大于d, $top\text{-}R$增加1

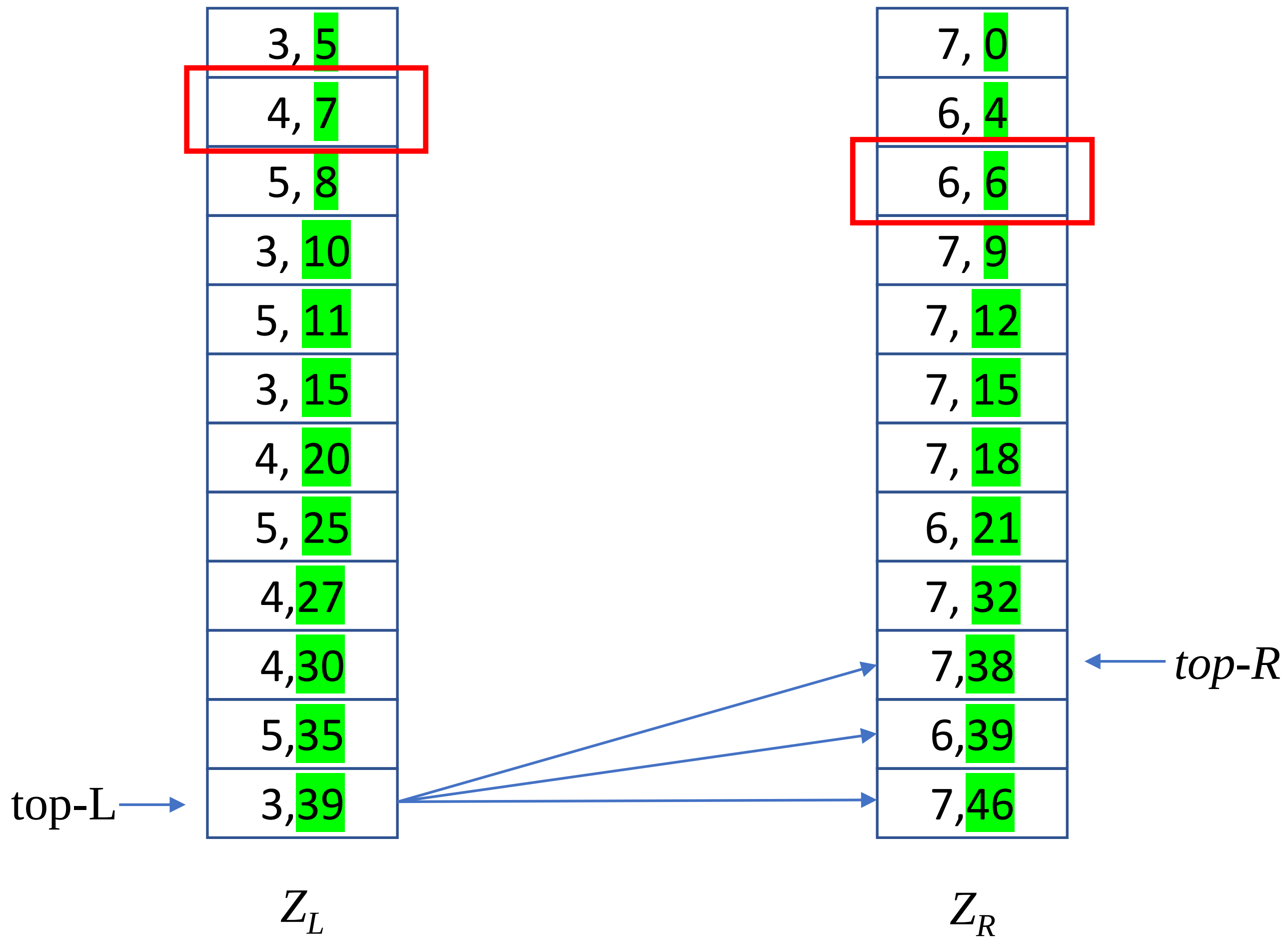
m=5, d=$\sqrt{5}$

	Z_L	Z_R	
	3, 5	7, 0	
	4, 7	6, 4	
	5, 8	6, 6	
	3, 10	7, 9	
	5, 11	7, 12	
	3, 15	7, 15	
	4, 20	7, 18	
	5, 25	6, 21	
	4,27	7, 32	
	4,30	7,38	← *top-R*
top-L →	5,35	6,39	
	3,39	7,46	

数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y < Z_R[top\text{-}R].y - d$, Z_R中后续元素y值都不比$Z_R[top\text{-}R].y$小，因此它们与$Z_L[top\text{-}L]$的距离必然大于d, $top\text{-}L$增加1

$m=5,\ d=\sqrt{5}$



数组中只显示各个点的y坐标值

$Z_L[top\text{-}L].y \ge Z_R[top\text{-}R].y - d$ 并且 $Z_L[top\text{-}L].y \le Z_R[top\text{-}R].y + d$, 为当前$Z_L[top\text{-}L]$连续检验最多6个$Z_R$中的元素（如果$Z_R$中有足够多的剩余元素），随后*top-L*增长1, *top-R*不变. 算法结束.

- (p_l, q_r)搜索算法时间复杂性
  - 获取Z_L和Z_R需要O(n)时间
  - 每次使得*top-L*增加1或者*top-R*增加1需要消耗常数时间
  - 算法结束时，*top-L*和*top-R*总共增长O(n)，外层for循环耗时O(n)
  - 算法时间复杂性为O(n)

(p_l, q_r)搜索算法

Input: Y_L, Y_R, d

Output: *result*

扫描Y_L得到Q_L中左临界区点，保持y坐标排序，得到Z_L

扫描Y_R得到Q_R中左临界区点，保持y坐标排序，得到Z_R

result=null;

top-R=0;

for *top-L*=0 to Z_L.*length*-1

    while Z_L[*top-L*].y > Z_R[*top-R*].y + d

        if *top-R*=Z_R.*length*-1

            return *result*;

        top-R ← *top-R*+1;

    if Z_L[*top-L*].y ≥ Z_R[*top-R*].y – d **and** Z_L[*top-L*].y ≤ Z_R[*top-R*].y + d

        for i=0 **to** *min*{5, Z_R.*length* –*top-R* – 1}

            if *dist*(Z_L[*top-L*], Z_R[*top-R*+i])<d

                result=(Z_L[*top-L*], Z_R[*top-R*+i]);

                d=*dist*(Z_L[*top-L*], Z_R[*top-R*+i]);

return *result*;