GEHMA MARTINI 532769

①

(a) $\mathcal{L} \rightarrow$ ② ④ ⑧ ⑤ ⑦ ① ⑥ ③

1 1 1 1 1 1 1 0 ← RANK (1 if which isn't the tail)

(1) $I = \{4, 5, 1\}$

$\mathcal{L}^* \rightarrow$ ② ⑧ ⑦ ⑥ ③

2 2 2 1 0 ← RANK (rank[succ[i]])

(2) $I = \{8, 6\}$

$\mathcal{L}^* \rightarrow$ ② ⑦ ③

4 3 0 ← RANK

(3) $I = \{7\}$

$\mathcal{L}^* \rightarrow$ ② ③

4 0 ← RANK

At this step we may start recombining

(1)

1	2	3	4	5	6	7	8
1	7	0	1	1	1	3	2

(2)

1	2	3	4	5	6	7	8
1	7	0	1	1	1	**3**	2

RANK += RANK(SUCC[i])
update rank of 7

(3)

1	2	3	4	5	6	7	8
1	7	0	1	1	**1**	3	**5**

RANK += RANK(SUCC[i])
update rank of 8 and 6

(4)

1	2	3	4	5	6	7	8
2	7	0	**6**	**4**	1	3	5

RANK += RANK(SUCC[i])
update rank of 4, 5 and 1

(b) $O\left(\frac{n}{B} \cdot \log_{M/B} \frac{n}{M} \cdot \log^* n\right) = \tilde{O}(n/B)$

OK

② $A = 1 - 2 - 4 - 9 - 10 - 11 - 15 - 19 - 20 - 24$

$B = 3 - 20$

(a) DOUBLING

$i=0$, $j=1$, $K=0$

1) Compare 3 with 1
2) Compare 3 with 2 $\Rightarrow K=1$
3) Compare 3 with 9 $\Rightarrow K=2$
4) Binary search in $A[3,3]$ of 3 $\Rightarrow$ $i=1$, $j=2$, $K=0$
5) Compare 20 with 4 $\Rightarrow K=1$
6) Compare 20 with 9 $\Rightarrow K=2$
7) Compare 20 with 11 $\Rightarrow K=3$
8) Compare 20 with 20 $\Rightarrow$ print 20. Stop.

(b) SHUFFLING

1) Choose permutation $\pi(x) = 2x + 3 \pmod{29}$

$\pi(A) = 5 - 7 - 11 - 21 - 23 - 25 - 4 - 12 - 14 - 22$

$\pi(B) = 9 - 14$

2) Sort $\pi(A) \Rightarrow$ $A_S = 4 - 5 - 7 - 11 - 12 - 14 - 21 - 22 - 23 - 25$

Sort $\pi(B) \Rightarrow$ $B_S = 9 - 14$

3) Write A_S and B_S in base 2: $A_2 =$ 00100 (4), 00101 (5), 00111 (7), 01011 (11), 01100 (12), 01110 (14), 10101 (21), 10110 (22), 10111 (23), 11001 (25)

$B_2 =$ 01001 (9), 01110 (14)

4) # of buckets = 4, since $L=3$

Take the 2 most significant digits of the elements to assign the bucket to each of them

$\{4,5,7\}$ from $\pi(A)$ $\{11,12,14\}$ from $\pi(A)$ $\{21,22,23\}$ from $\pi(A)$ $\{25\}$ from $\pi(A)$
$\emptyset$ from $\pi(B)$ $\{9,14\}$ from $\pi(B)$ $\emptyset$ from $\pi(B)$ $\emptyset$ from $\pi(B)$

5) Compute merge based intersection between $\pi(A)_2 \cap \pi(B)_2 = \{14\}$

6) Use $\pi^{-1}(x)$ to find the original value: $\pi^{-1}(x) = 15x + 13$

$\pi^{-1}(14) = (20)$

(3) Idea: mimic the behaviour of BOUNDED QUICKSORT, to eliminate tail recursion

③ Per limitare lo spazio a $O(\log n)$ si può utilizzare la medesima idea del bounded quicksort:

se la sequenza in cui si dovrebbe ripetere la ricorsione dovesse essere più lungo della metà dell'array (quindi $\frac{i+j}{2}$) bisogna evitare la ricorsione in quanto risulterebbe sbilanciato e optare per un processo iterativo.

```
randSelect(A, i, j, k){
  while(j - i > 0){
    r = random pivot
    S< = items of A < of A[r]
    S> = items of A > of A[r]
    M< = |S<|
    M= = |A| - |S<| - |S>|
    if(k ≤ M<){
      if(M< > (i+j)/2)   // la parte sinistra è troppo lunga per la ricorsione
        j = i + M<
      else
        return randSelect(A, i, M<, k);
    }
    else if(k ≤ (M< + M=))
      return A[r]
    else{
      if(M> > (i+j)/2)   // parte destra troppo lunga per ricorsione
        i = i + M< + M=
      else
        return randSelect(A, i + M< + M=, k - M< - M=);
    }
  }
}
```

In questo modo si ottengono ricorsioni bilanciate al più $O(\log n)$

OK bene!