FuzzySort Report

Kexin Ding

Dear TA, I am the only member in my team. And I also finish both two algorithms. Please check the code in fuzzysort_quicksort.py. Thank you!

1.Results

```
************************************************  FuzzySort: TestCase 1  ************************************************
Before the sorting:
[(5, 7), (1, 3), (4, 6), (8, 10)]

After the sorting:
[(1, 3), (5, 7), (4, 6), (8, 10)]

Running time : 13766.956329345703 ns


************************************************  FuzzySort: TestCase 2  ************************************************
Before the sorting:
[(6, 7), (9, 11), (13, 14), (3, 7), (11, 15), (13, 14), (12, 14), (14, 15), (9, 15), (5, 7), (7, 9), (1, 5), (1, 9), (6, 10)]

After the sorting:
[(1, 5), (6, 7), (5, 7), (7, 9), (6, 10), (1, 9), (3, 7), (9, 11), (11, 15), (13, 14), (14, 15), (12, 14), (13, 14), (9, 15)]

Running time : 33539.676666259766 ns


************************************************  QuickSort: TestCase 1  ************************************************
Before the sorting:
[(5, 7), (1, 3), (4, 6), (8, 10)]

After the sorting:
[(1, 3), (4, 6), (5, 7), (8, 10)]

Running time : 10908.222198486328 ns


************************************************  QuickSort: TestCase 2  ************************************************
Before the sorting:
[(6, 7), (9, 11), (13, 14), (3, 7), (11, 15), (13, 14), (12, 14), (14, 15), (9, 15), (5, 7), (7, 9), (1, 5), (1, 9), (6, 10)]

After the sorting:
[(1, 5), (1, 9), (3, 7), (5, 7), (6, 7), (6, 10), (7, 9), (9, 11), (9, 15), (11, 15), (12, 14), (13, 14), (13, 14), (14, 15)]

Running time : 43364.524841308594 ns
```

Some explanation for the result:

1. Testing two test cases as the requirement.
2. Evaluating running time by time.time() and show it on the nanosecond scale.
3. For the fairness, running each algorithm for 5000 times. Using the average of the running time as the output.
4. Because of using random.randint() in find_overlap(), hence fuzzy-sort will show multiple results in the different running. But each result will satisfy the concept of fuzzy-sort.

Time Complexity:

For Fuzzy Sort:

$T(n) = 2T(n/2) + \Theta(n)$

$= 2*2*T(n/4) + 2 * \Theta(n)$

$= 2*2*2*T(n/8) + 3 * \Theta(n)$

…

$= 2^i T(n/2^i) + i * \Theta(n)$

$\frac{n}{2^i} = 1$ ➔ $i = \lg n$

$= n + \lg n * \Theta(n)$

$= \Theta(n \lg n)$

For QuickSort:

$T(n) = 2T(n/2) + \Theta(n)$

$= 2*2*T(n/4) + 2 * \Theta(n)$

= 2*2*2*T(n/8) + 3 * Θ(n)
…
= $2^i T(n/2^i)$ + i * Θ(n)

$\frac{n}{2^i} = 1$ ➔ i = lgn

= n + lgn* Θ(n)
= Θ (nlgn)