Sorting: Concluding Remarks

Madhavan Mukund

https://www.cmi.ac.in/~madhavan

Programming, Data Structures and Algorithms using Python

Week 3

Stable sorting

- Often list values are tuples
  - Rows from a table, with multiple columns / attributes
  - A list of students, each student entry has a roll number, name, marks, ...

Stable sorting

- Often list values are tuples
  - Rows from a table, with multiple columns / attributes
  - A list of students, each student entry has a roll number, name, marks, . . .
- Suppose students have already been sorted by roll number

Stable sorting

- Often list values are tuples
  - Rows from a table, with multiple columns / attributes
  - A list of students, each student entry has a roll number, name, marks, . . .
- Suppose students have already been sorted by roll number
- If we now sort by name, will all students with the same name remain in sorted order with respect to roll number?

Stable sorting

- Often list values are tuples
  - Rows from a table, with multiple columns / attributes
  - A list of students, each student entry has a roll number, name, marks, ...
- Suppose students have already been sorted by roll number
- If we now sort by name, will all students with the same name remain in sorted order with respect to roll number?
- Stability of sorting is crucial in many applications

Stable sorting

- Often list values are tuples
  - Rows from a table, with multiple columns / attributes
  - A list of students, each student entry has a roll number, name, marks, ...
- Suppose students have already been sorted by roll number
- If we now sort by name, will all students with the same name remain in sorted order with respect to roll number?
- Stability of sorting is crucial in many applications
- Sorting on column B should not disturb sorting on column A

Stable sorting

- The quicksort implementation we described is not stable
    - Swapping values while partitioning can disturb existing sorted order

Stable sorting

- The quicksort implementation we described is not stable
    - Swapping values while partitioning can disturb existing sorted order
- Merge sort is stable if we merge carefully
    - Do not allow elements from the right to overtake elements on the left
    - While merging, prefer the left list while breaking ties

Other criteria

- Minimizing data movement
  - Imagine each element is a heavy carton
  - Reduce the effort of moving values around

Best sorting algorithm?

- Quicksort is often the algorithm of choice, despite $O(n^2)$ worst case

Best sorting algorithm?

- Quicksort is often the algorithm of choice, despite $O(n^2)$ worst case
- Merge sort is typically used for “external” sorting
  - Database tables that are too large to store in memory all at once
  - Retrieve in parts from the disk and write back

Best sorting algorithm?

- Quicksort is often the algorithm of choice, despite $O(n^2)$ worst case
- Merge sort is typically used for “external” sorting
  - Database tables that are too large to store in memory all at once
  - Retrieve in parts from the disk and write back
- Other $O(n \log n)$ algorithms exist — heapsort

Best sorting algorithm?

- Quicksort is often the algorithm of choice, despite $O(n^2)$ worst case
- Merge sort is typically used for “external” sorting
  - Database tables that are too large to store in memory all at once
  - Retrieve in parts from the disk and write back
- Other $O(n \log n)$ algorithms exist — heapsort
- Sometimes hybrid strategies are used
  - Use divide and conquer for large n
  - Switch to insertion sort when n becomes small (e.g., $n < 16$)