Maggie Zhao
APCS 2 pd01
HW06-- How Fast Are Your Turtles?
2018-02-12

Key:

→ : either a split or a merge is occurring in this step

→ : no action occurred (the element has been brought down to the next layer)

[x, ..] = the array has been split

[x, ...] = the array has been merged.

⊗ = splitting step

⊗ = merging step

[x, ..] = no change occurred

Win:

[2 , 1 , 3]

1

[2 , 1] [3]

2

[2] [1] [3]

3

[1 , 2] [3]

4

[1 , 2 , 3]

When the original array is an odd length, there will be a one element array that remains unchanged during splitting and merging steps.

Length = 3
4 splits
4 merges
4 layers

Tim:

[6 , 7 , 2 , 5 , 1 , 3 , 4]

1

[6 , 7 , 2 , 5] [1 , 3 , 4]

2

[6 , 7] [2 , 5] [1 , 3] [4]

3

[6] [7] [2] [5] [1] [3] [4]

4

[6 , 7] [2 , 5] [1 , 3] [4]

5

[2 , 5 , 6 , 7] [1 , 3 , 4]

6

[1 , 2 , 3 , 4 , 5 , 6 , 7]

Length = 7
12 splits
12 merges
6 layers

Jeremy:

Length = 8
14 splits
14 merges
6 layers

① [6, 7, 2, 8, 5, 1, 3, 4]

② [6, 7, 2, 8] [5 1, 3, 4]

③ [6, 7] [2, 8] [5, 1] [3, 4]

[6] [7] [2] [8] [5] [1] [3] [4]

④ [6, 7] [2, 8] [1, 5] [3, 4]

⑤ [2, 6, 7, 8] [1, 3, 4, 5]

⑥ [1, 2, 3, 4, 5, 6, 7, 8]

Régine:

Length = 9
8 layers

① [9, 6, 7, 2, 8, 5, 1, 3, 4]

② [9, 6, 7, 2, 8] [5, 1, 3, 4]

③ [9, 6, 7] [2, 8] [5, 1] [3, 4]

④ [9, 6] [7] [2] [8] [5] [1] [3] [4]

⑤ [9] [6] [7] [2] [8] [5] [1] [3] [4]

Only the first two elements (6 and 9) are split and merged, while the other pairs are simply brought down.

⑥ [6, 9] [7] [2] [8] [5] [1] [3] [4]

⑦ [6, 7, 9] [2, 8] [1, 5] [3, 4]

⑧ [2, 6, 7, 8, 9] [1, 3, 4, 5]

[1, 2, 3, 4, 5, 6, 7, 8, 9]

Scanned by CamScanner

Richard

① [0, 6, 7, 2, 8, 5, 1, 9, 3, 4]

Length: 10
8 layers

② [0, 6, 7, 2, 8] [5, 1, 9, 3, 4]

③ [0, 6, 7] [2, 8] [5, 1, 9] [3, 4]

The elements are merged along the same lines they are split. If the element was not part of an x-element array originally, it cannot be merged into one.

④ [0, 6] [7] [2] [8] [5, 1] [9] [3] [4]

⑤ [0] [6] [7] [2] [8] [5] [1] [9] [3] [4]

⑥ [0, 6] [7] [2] [8] [1, 5] [9] [3] [4]

⑦ [0, 6, 7] [2, 8] [1, 5, 9] [3, 4]

⑧ [0, 2, 6, 7, 8] [1, 3, 4, 5, 9]

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

William

① [12, 9, 15, 10, 6, 16, 7, 2, 8, 13, 5, 1, 14, 3, 4, 11]

Length=16
8 layers

② [12, 9, 15, 10, 6, 16, 7, 2] [8, 13, 5, 1, 14, 3, 4, 11]

③ [12, 9, 15, 10] [6, 16, 7, 2] [8, 13, 5, 1] [14, 3, 4, 11]

④ [12, 9] [15, 10] [6, 16] [7, 2] [8, 13] [5, 1] [14, 3] [4, 11]

⑤ [12] [9] [15] [10] [6] [16] [7] [2] [8] [13] [5] [1] [14] [3] [4] [11]

⑥ [9, 12] [10, 15] [6, 16] [2, 7] [8, 13] [1, 5] [3, 14] [4, 11]

⑦ [9, 10, 12, 15] [2, 6, 7, 16] [1, 5, 8, 13] [3, 4, 11, 14]

⑧ [2, 6, 7, 9, 10, 12, 15, 16] [1, 3, 4, 5, 8, 11, 13, 14]

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

Merge sort runs on $O(n \cdot \log_2 n)$ time. When each array was being sorted, it took (the ceiling of) $\log_2 n$ passes for the entire array to be split up into individual one element arrays. This is because, like binary search, it reduces the field of vision by 2 with each pass, but in this sort algorithm, all of the different views are stored. The merge algorithm is $O(n)$, as it takes n items and compares them with every other element in the array. The merging process occurs at every level of the dividing part, and n items iterated $\log(n)$ times yields a total time of $O(n \log_2 n)$.