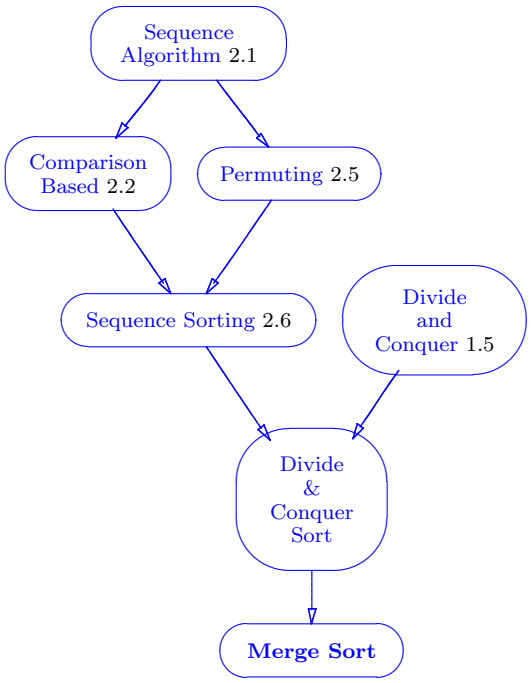
2.6.3 mergesort

Section authors: Andrew Hill, John Lewis, David Thomas.



Refinement of: Sequence Sorting Algorithm (§2.6), therefore of Comparison Based (§2.2), Permuting (§2.5), Sequence Algorithm (§2.1), Divide and Conquer Algorithm (§1.5).

Prototype:
```
template<class RandomAccessIterator>
void stable_sort(RandomAccessIterator first,
                 RandomAccessIterator last)
```

Effects: Standard effects of a Sequence Sorting Algorithm (§2.6). In brief: the elements in [first, last) after execution are a permutation of the original elements in the range, and they are in nondecreasing order according the comparison operator.

Asymptotic complexity: Let $N = \text{last} - \text{first}$.

- Average case (random data): $O(N \log N)$
- Worst case: $O(N \log N)$

Complexity in terms of operation counts:

- Average case:
  Value comparisons: $1.16N \log_2 N - 3.31N$
  Value assignments: $1.2N \log_2 N - 2.14N$

N	comparisons	assignments
10	27	33
100	591	776
1,000	9,762	11863
10,000	127,770	138,592
100,000	1,595,396	1,785,634
1,000,000	19,823,161	21,857,455

- Average Case Computational Time:
  Unscaled : $1.8 * 10^{-8} N \log_2 N + 1.7 * 10^{-7} N$
  Scaled : $N \log_2 N + 9.55N$
- See also Sorting Algorithm Operation Counts (§2.91) for sample counts on random data for mergesort and other sorting algorithms.
- An animation of mergesort (and a number of other algorithms), by Alejo Hausner.