2.6.1 Heapsort

Section author: David R. Musser.

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

Prototype: template<class RandomAccessIterator>
void heapsort(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: \( N \log_2 N + 0.36N \)
  Value assignments: \( 1.2N \log_2 N + 3.2N \)
  Iterator operations: \( 12.4N \log_2 N + 10N \)
  Integer operations: \( 14.5N \log_2 N + 17N \)

• See also Sorting Algorithm Operation Counts (§2.91) for sample counts on random data for heapsort and other sorting algorithms.

2.6.2 Heapsort iterator trace plot

Compare with other algorithms: Introsort (§2.6.4), Mergesort (§??).
One thousand elements are being sorted by the version of heapsort (actually a special case of partial_sort) implemented in SGI STL. From time 0 to about time 30,000 the algorithm is building a heap (with algorithm makeheap). Then, for the rest of the time, it is repeatedly extracting the maximum element and reheaping.