Strand sort

From Wikipedia, the free encyclopedia

Strand sort is a sorting algorithm. It works by repeatedly pulling sorted sublists out of the list to be sorted and merging them with a result array. Each iteration through the unsorted list pulls out a series of elements which were already sorted, and merges those series together.

The name of the algorithm comes from the "strands" of sorted data within the unsorted list which are removed one at a time. It is a comparison sort due to its use of comparisons when removing strands and when merging them into the sorted array.

Strand sort

Class	Sorting algorithm
Data structure	Linked list
Worst case performance	O(n²)
Best case performance	O(n)
Average case performance	O(n²)
Worst case space complexity	O(1) auxiliary

The strand sort algorithm is $O(n^2)$ in the average case. In the best case (a list which is already sorted) the algorithm is linear, or $O(n)$. In the worst case (a list which is sorted in reverse order) the algorithm is $O(n^2)$.

Strand sort is most useful for data which is stored in a linked list, due to the frequent insertions and removals of data. Using another data structure, such as an array, would greatly increase the running time and complexity of the algorithm due to lengthy insertions and deletions. Strand sort is also useful for data which already has large amounts of sorted data, because such data can be removed in a single strand.

Contents [hide]

1 Example
2 Algorithm
2.1 Haskell implementation
2.2 PHP implementation
2.3 Python implementation
3 References

Example [edit]

Unsorted list	Sublist	Sorted list
3 1 5 4 2		
1 4 2	3 5	
1 4 2		3 5
2	1 4	3 5
2		1 3 4 5
	2	1 3 4 5
		1 2 3 4 5

1. Parse the unsorted list once, taking out any ascending (sorted) numbers.
2. The (sorted) sublist is, for the first iteration, pushed onto the empty sorted list.
3. Parse the unsorted list again, again taking out relatively sorted numbers.
4. Since the sorted list is now populated, merge the sublist into the sorted list.
5. Repeat steps 3–4 until both the unsorted list and sublist are empty.

Algorithm [edit]

A simple way to express strand sort in pseudocode is as follows:

```
procedure strandSort( A : list of sortable items ) defined as:
  while length( A ) > 0
    clear sublist
```

Wikipedia The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

Languages
Čeština
فارسی
Português
Српски / srpski
Türkçe
Українська
Edit links

```
    sublist[ 0 ] := A[ 0 ]
    remove A[ 0 ]
    for each i in 0 to length( A ) - 1 do:
      if A[ i ] > sublist[ last ] then
        append A[ i ] to sublist
        remove A[ i ]
      end if
    end for
    merge sublist into results
  end while
  return results
end procedure
```

Haskell implementation [edit]

```
merge [] l = l
merge l [] = l
merge l1@(x1:r1) l2@(x2:r2) =
    if x1 < x2 then x1:(merge r1 l2) else x2:(merge l1 r2)

ssort [] = []
ssort l = merge strand (ssort rest)
    where (strand, rest) = foldr extend ([],[]) l
          extend x ([],r) = ([x],r)
          extend x (s:ss,r) = if x <= s then (x:s:ss,r) else (s:ss,x:r)
```

PHP implementation [edit]

```
function strandsort(array $arr) {
  $result = array();
  while (count($arr)) {
    $sublist = array();
    $sublist[] = array_shift($arr);
    $last = count($sublist)-1;
    foreach ($arr as $i => $val) {
      if ($val > $sublist[$last]) {
        $sublist[] = $val;
        unset($arr[$i]);
        $last++;
      }
    }

    if (!empty($result)) {
      foreach ($sublist as $val) {
        $spliced = false;
        foreach ($result as $i => $rval) {
          if ($val < $rval) {
            array_splice($result, $i, 0, $val);
            $spliced = true;
            break;
          }
        }

        if (!$spliced) {
          $result[] = $val;
        }
      }
    }
    else {
      $result = $sublist;
    }
  }

  return $result;
}
```

Python implementation [edit]

```
items = len(unsorted)
sortedBins = []
while( len(unsorted) > 0 ):
    highest = float("-inf")
    newBin = []
    i = 0
    while( i < len(unsorted) ):
        if( unsorted[i] >= highest ):
            highest = unsorted.pop(i)
            newBin.append( highest )
        else:
            i=i+1
    sortedBins.append(newBin)

sorted = []
while( len(sorted) < items ):
    lowBin = 0
    for j in range( 0, len(sortedBins) ):
        if( sortedBins[j][0] < sortedBins[lowBin][0] ):
            lowBin = j
    sorted.append( sortedBins[lowBin].pop(0) )
    if( len(sortedBins[lowBin]) == 0 ):
        del sortedBins[lowBin]
```

References [edit]

- Paul E. Black "Strand Sort" from Dictionary of Algorithms and Data Structures at NIST.

v · t · e	Sorting algorithms [hide]
Theory	Computational complexity theory · Big O notation · Total order · Lists · Inplacement · Stability · Comparison sort · Adaptive sort · Sorting network · Integer sorting
Exchange sorts	Bubble sort · Cocktail sort · Odd–even sort · Comb sort · Gnome sort · Quicksort · Stooge sort · Bogosort
Selection sorts	Selection sort · Heapsort · Smoothsort · Cartesian tree sort · Tournament sort · Cycle sort
Insertion sorts	Insertion sort · Shellsort · Splaysort · Tree sort · Library sort · Patience sorting
Merge sorts	Merge sort · Cascade merge sort · Oscillating merge sort · Polyphase merge sort · **Strand sort**
Distribution sorts	American flag sort · Bead sort · Bucket sort · Burstsort · Counting sort · Pigeonhole sort · Proxmap sort · Radix sort · Flashsort
Concurrent sorts	Bitonic sorter · Batcher odd–even mergesort · Pairwise sorting network
Hybrid sorts	Block sort · Timsort · Introsort · Spreadsort · JSort
Other	Topological sorting · Pancake sorting · Spaghetti sort

Categories: Sorting algorithms | Comparison sorts