Sorting Algorithms (II)

2023

Problem Set and Solutions

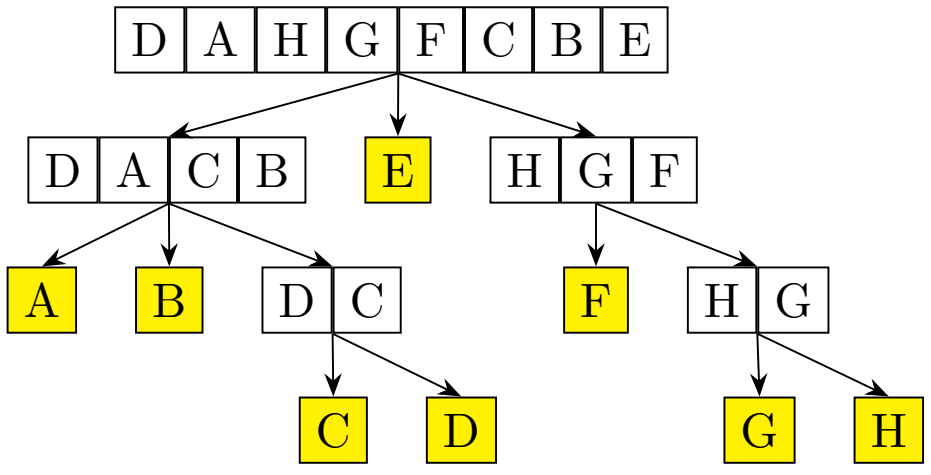
The Centre for Education in Mathematics and Computing
Faculty of Mathematics, University of Waterloo
www.cemc.uwaterloo.ca

Quicksort: Problem Set

1. Sort the following list of letters in alphabetical order using quicksort, by choosing the *last* element as the pivot:

   D A H G F C B E

Solution



Quicksort: Problem Set

2. What ideal property should the pivot have?

Solution

Ideally the pivot should divide the list into sublists that are approximately equal in size. In practice, the best way to do this is to choose a random element as the pivot. If the list is already sorted or almost sorted, then choosing the first or last element as the pivot will not divide the list in half.

Quicksort: Problem Set

3. On average, if a list contains n elements, then approximately how much time will quicksort take in order to sort the list?

Solution

If good pivots are chosen (ones that divide the list in half) then there will be approximately $\log_2 n$ pivots. It then takes approximately n steps to partition the n elements around the pivot. So on average, quicksort takes time approximately equal to $n \log_2 n$ in order to sort.

How Many Times Can You Split a Value in Half?

Question: If n is a power of 2, how many times can you split n in half?

Answer: If $n = 2^x$, then n can be split in half x times.

Question: Given n, how do we find x such that $n = 2^x$?

Answer: Logarithms are the inverse of exponents.
If $n = 2^x$ then $\log_2 n = x$

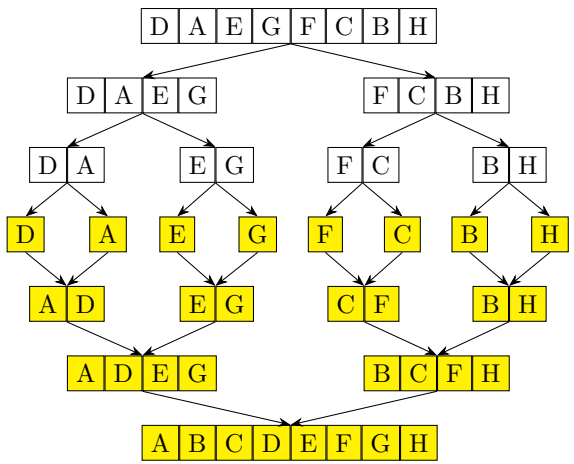
Therefore, given n, the value of n can be split in half $\log_2 n$ times.

Merge Sort: Problem Set

1. Sort the following list of letters in alphabetical order using merge sort:

   D A E G F C B H

Solution



Merge Sort: Problem Set

2. On average, if a list contains n elements, then approximately how much time will merge sort take in order to sort the list?

Solution

The list will be divided approximately $\log_2 n$ times. It then takes approximately n steps to merge the sorted lists back together. So on average, merge sort takes time approximately equal to $n \log_2 n$ in order to sort.

Challenge: Problem Set

1. Quicksort and merge sort are examples of **recursive algorithms**. A recursive algorithm solves a problem by combining the solutions to smaller instances of the same problem. Create a recursive algorithm that computes the factorial of a number.

Solution

The factorial of n is defined as:
$n! = n \times (n-1) \times (n-2) \times (n-3) \times \ldots \times 1$

A recursive algorithm to compute the factorial of n is:
$n! = n \times (n-1)!$

Challenge: Problem Set

2. A sorting algorithm is considered **stable** if duplicate elements maintain their relative order after sorting. For instance, if the original list contains 5_a and 5_b in that order, a stable sort will keep 5_a and 5_b in that order. If the sorted list ends up having 5_b first and then 5_a, then the sorting algorithm is not stable. Of selection, insertion, bubble, quick, and merge sort, which sorting algorithms are stable?

Solution

Insertion sort, bubble sort, and merge sort are stable.
Selection sort and quicksort are not stable.

Unstable Sorting Algorithms

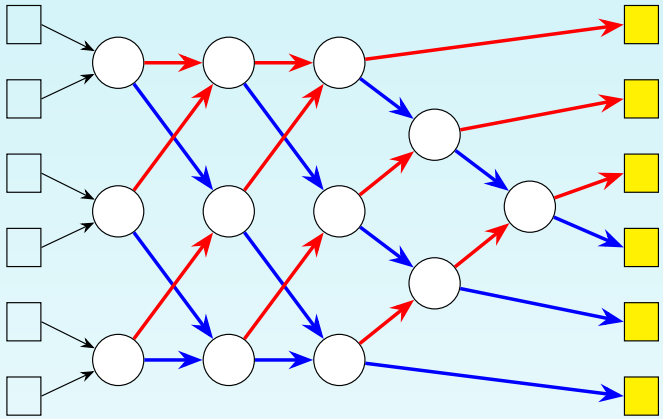
To see that selection sort and quicksort are not stable, try sorting the following integers in ascending order:

$$3_a \quad 2 \quad 3_b \quad 1$$

For quicksort choose the *first* element as the pivot.

Challenge: Problem Set

3. Sorting can be sped up using a sorting network. Below is a sorting network that sorts 6 items.

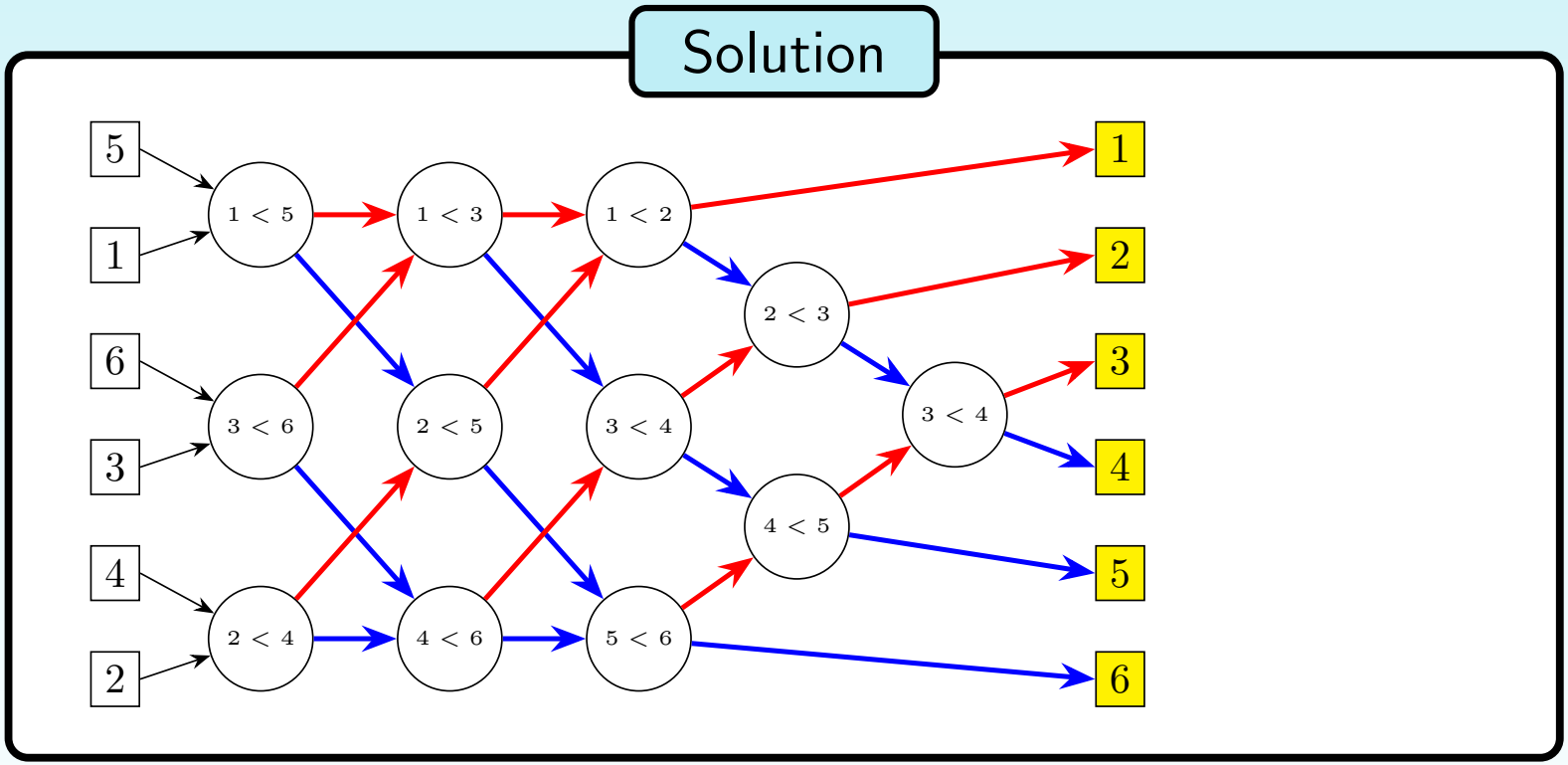


The leftmost column is the unsorted list. The list elements move through the sorting network by following the arrows. Each circle represents a comparison between two elements. The smaller elements in each comparison follow the higher red arrows and the larger elements follow the lower blue arrows. The rightmost column is the sorted list.

Challenge: Problem Set

a) Use the sorting network to sort the list: **5 1 6 3 4 2**

Solution

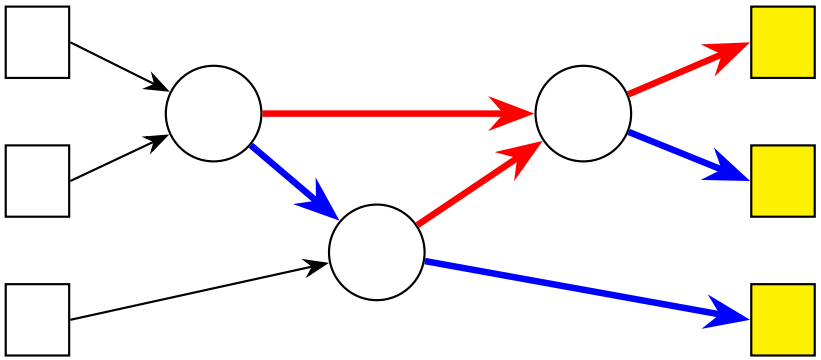


Challenge: Problem Set

b) Design a sorting network that sorts 3 items.

Solution

Here is one possibility:



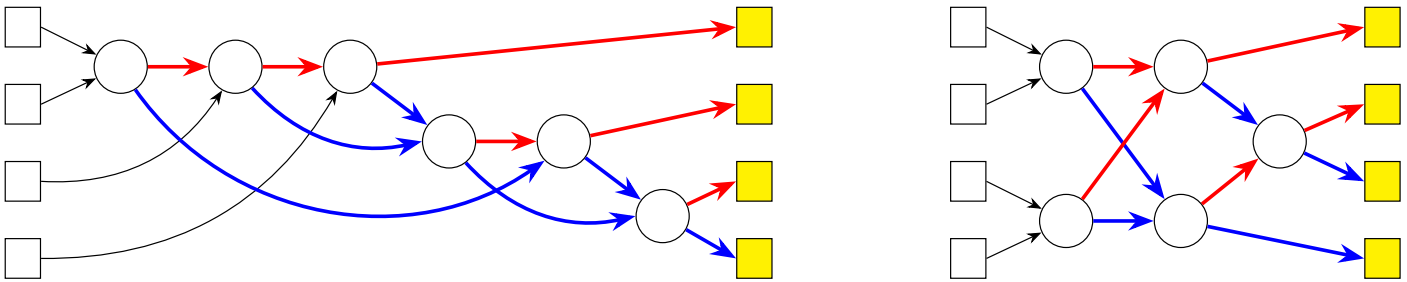
Note that in a sorting network for 3 items it is not possible to include parallelism.

Challenge: Problem Set

c) Design a sorting network that sorts 4 items.

Solution

Here are two possibilities:



Note that sorting networks are not unique. Also note that the first option does not include parallelism while the second option takes advantage of it.