# Merge sort

## Typografie a publikování – 5. projekt

Peter Ďurica

Vysoké učení technické v Brně
Fakulta informačních technologií

7. mája 2022

# Výhody Merge sort

Výhody Merge sort oproti ostatným algoritmom:

# Výhody Merge sort

Výhody Merge sort oproti ostatným algoritmom:

- Je rýchlejší pre väčšie zoznamy, pretože narozdiel od insertion a bubble sortu neprechádza cez zoznam viac krát.

# Výhody Merge sort

Výhody Merge sort oproti ostatným algoritmom:

- Je rýchlejší pre väčšie zoznamy, pretože narozdiel od insertion a bubble sortu neprechádza cez zoznam viac krát.
- Je stabilný, teda 2 elementy zoznamu s rovnakou hodnotou si zachovajú svoje poradie vo finálnom zozname.

# Výhody Merge sort

Výhody Merge sort oproti ostatným algoritmom:

- Je rýchlejší pre väčšie zoznamy, pretože narozdiel od insertion a bubble sortu neprechádza cez zoznam viac krát.
- Je stabilný, teda 2 elementy zoznamu s rovnakou hodnotou si zachovajú svoje poradie vo finálnom zozname.
- Má konštantnú zložitosť.

# Výhody Merge sort

Výhody Merge sort oproti ostatným algoritmom:

- Je rýchlejší pre väčšie zoznamy, pretože narozdiel od insertion a bubble sortu neprechádza cez zoznam viac krát.
- Je stabilný, teda 2 elementy zoznamu s rovnakou hodnotou si zachovajú svoje poradie vo finálnom zozname.
- Má konštantnú zložitosť.

Nevýhody Merge sort:

# Výhody Merge sort

Výhody Merge sort oproti ostatným algoritmom:

- Je rýchlejší pre väčšie zoznamy, pretože narozdiel od insertion a bubble sortu neprechádza cez zoznam viac krát.
- Je stabilný, teda 2 elementy zoznamu s rovnakou hodnotou si zachovajú svoje poradie vo finálnom zozname.
- Má konštantnú zložitosť.

Nevýhody Merge sort:

- Pomalší pre menšie zoznamy, kvôli konštantnej zložitosti.

# Výhody Merge sort

Výhody Merge sort oproti ostatným algoritmom:

- Je rýchlejší pre väčšie zoznamy, pretože narozdiel od insertion a bubble sortu neprechádza cez zoznam viac krát.
- Je stabilný, teda 2 elementy zoznamu s rovnakou hodnotou si zachovajú svoje poradie vo finálnom zozname.
- Má konštantnú zložitosť.

Nevýhody Merge sort:

- Pomalší pre menšie zoznamy, kvôli konštantnej zložitosti.
- Používa viac pamäte kvôli tvorbe finálneho zoznamu. Nepracuje in situ.

# Princíp Merge sort

Merge sort je založený na princípe zlučovania.

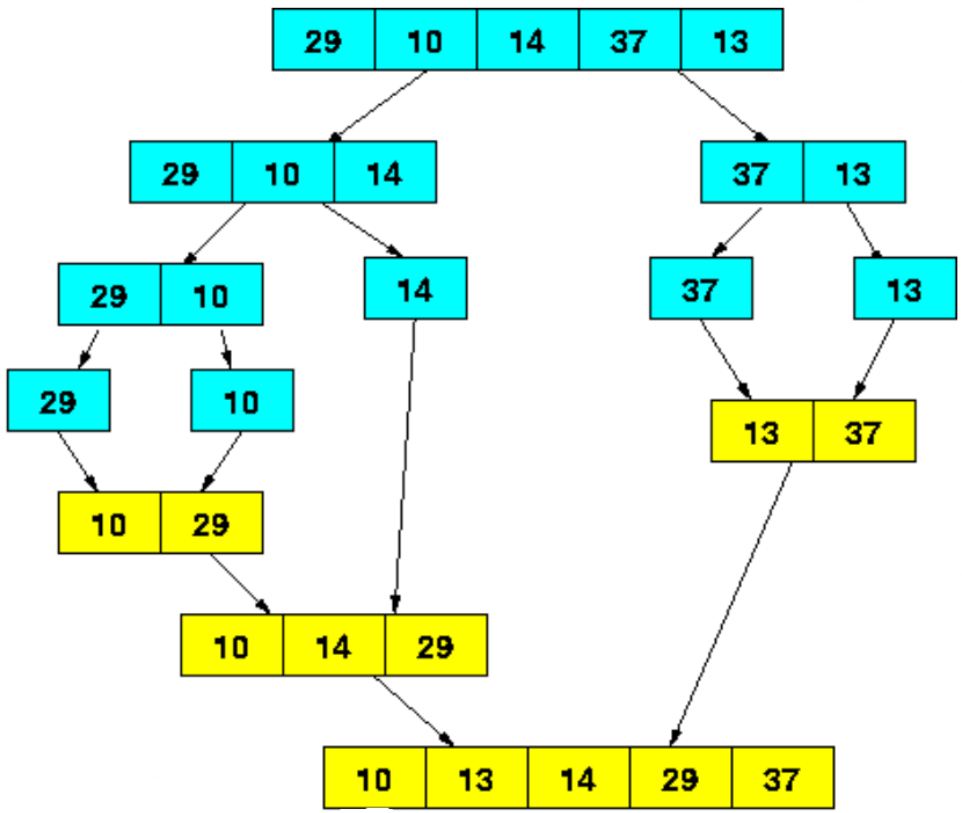
- Pole rozdeľujeme do tzv. behov – súvislých úsekov, ktoré sú už zoradené.
- Na začiatku budú všetky behy jednoprvkové.
- Potom budeme dohromady zlievať vždy dva susedné behy do jediného zoradeného behu o dĺžke danej súčtom počtu prvkov zlievaných behov, ktoré bude ležať na mieste oboch vstupných behov.
- V poslednej iterácii bude postupnosť pozostávať z jediného behu, a bude teda zoradená.

Rekurzívna varianta – metóda postupne volá samú seba pre ľavú a pravú polovicu zadanej časti pola a pri návrate z rekurzie zlieva už roztriedené postupnosti.

# Vizualizácia Merge sort

## Postup

Merge sort najskôr rozdelí zoznam na jednoprvkové hodnoty, ktoré následne zoradí do postupne väčších postupností až sa dopracuje k jednému finálnemu zoznamu.



# Pseudokód Merge sort

```
   function mergesort(var a as array)
1  if n == 1 then
2      return a
3  end if
4
5  var l1 as array = a[0] ... a[n/2]
6  var l2 as array = a[n/2 + 1] ... a[n]
7
8  l1 = mergesort( l1 )
9  l2 = mergesort( l2 )
10
11 return merge( l1,l2 )
```

# Pseudokód Merge sort

```
function merge(var a as array, var b as array)
1  var c as array
2  while a and b have elements do
3      if a[0] > b[0] then
4          add b[0] to the end of c
5          remove b[0] from b
6      else
7          add a[0] to the end of c
8          remove a[0] from a
9      end if
10 end while
```

# Pseudokód Merge sort

Pokračovanie **function** merge

```
11 while a has elements do
12     add a[0] to the end of c
13     remove a[0] from a
14 end while
15
16 while b has elements do
17     add b[0] to the end of c
18     remove b[0] from b
19 end while
20
21 return c
```

# Zložitosť Merge sort

Merge sort má zložitosť O(n*log n).

- Ak každým krokom delíme číslo na polovice, tak sa to označuje logaritmickou funkciou log n a počet krokov sa reprezentuje najviac log n + 1.
- Takisto spravíme jednokrokovú operáciu pri hladaní stredu zoznamu. tj. O(1)
- Pre spojenie predtým rozdelených zoznamov potrebujeme zložitosť O(n).
- Takže celkový čas za ktorý sa vykoná Merge sort bude n(log n + 1), čo nám dá zložitosť O(n*log n).

Keďže Merge sort má konsťantnú zložitosť tak to znamená že v najhoršom aj v najlepšom prípade sa jej hodnota nemení.

# Porovnanie zložitostí algoritmov

Porovnanie zložitosti Merge sortu s ostatnými riadiacimy algoritmami:

| Algorithm | Average complexity | Best complexity | Worst complexity |
|---|---|---|---|
| Bubble sort | $O(n^2)$ | $O(n^2)$ | $O(n^2)$ |
| Modified Bubble sort | $O(n^2)$ | $O(n)$ | $O(n^2)$ |
| Selection sort | $O(n^2)$ | $O(n^2)$ | $O(n^2)$ |
| Insertion sort | $O(n^2)$ | $O(n)$ | $O(n^2)$ |
| Heap sort | $O(n \log n)$ | $O(n \log n)$ | $O(n \log n)$ |
| merge sort | $O(n \log n)$ | $O(n \log n)$ | $O(n \log n)$ |
| Quick sort | $O(n \log n)$ | $O(n \log n)$ | $O(n^2)$ |

# Použité zdroje


- Data Structures - Merge Sort Algorithm

  https://www.tutorialspoint.com/merge-sort-algorithm.htm
- Merge sort, advantages and disadvantages

  https://getrevising.co.uk/merge-sort-advantages-and-disadvantages
- Quick Sort vs Merge Sort

  https://www.geeksforgeeks.org/quick-sort-vs-merge-sort/
- Prezentácia z 8.prednášky predmetu IAL
- Obrázok zložitosti algoritmov

  https://www.semanticscholar.org/paper/