# Analiza eksperymentalna nowych algorytmów sortowania w miejscu

**Damian Baliński**

2021, Wrocław

# Motywacja ulepszania algorytmów sortujących

- wydajność
- działanie w miejscu
- optymalizacja przypadku pesymistycznego
- mała liczba porównań

# Wady algorytmu Quick Sort

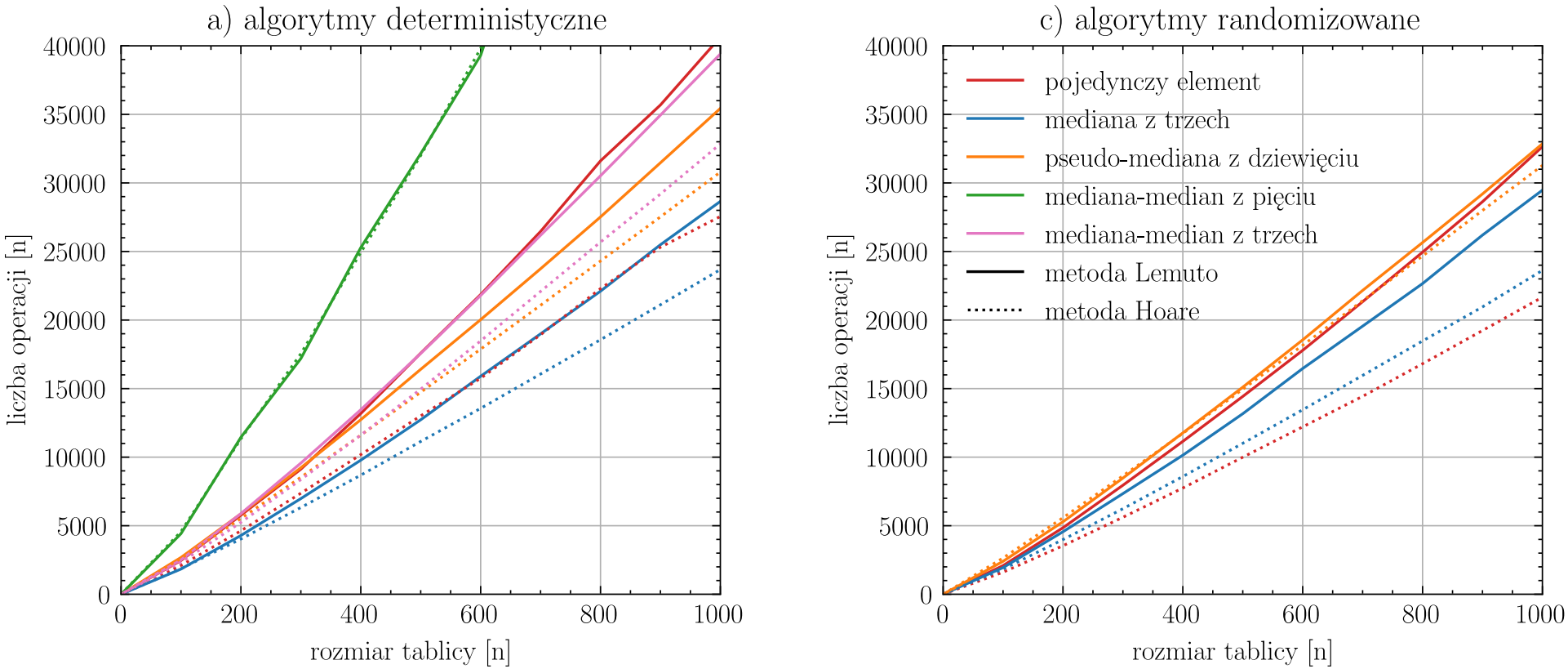
- $O(n^2)$ dla przypadku pesymistycznego
- duża liczba porównań

# Wady algorytmu Merge Sort

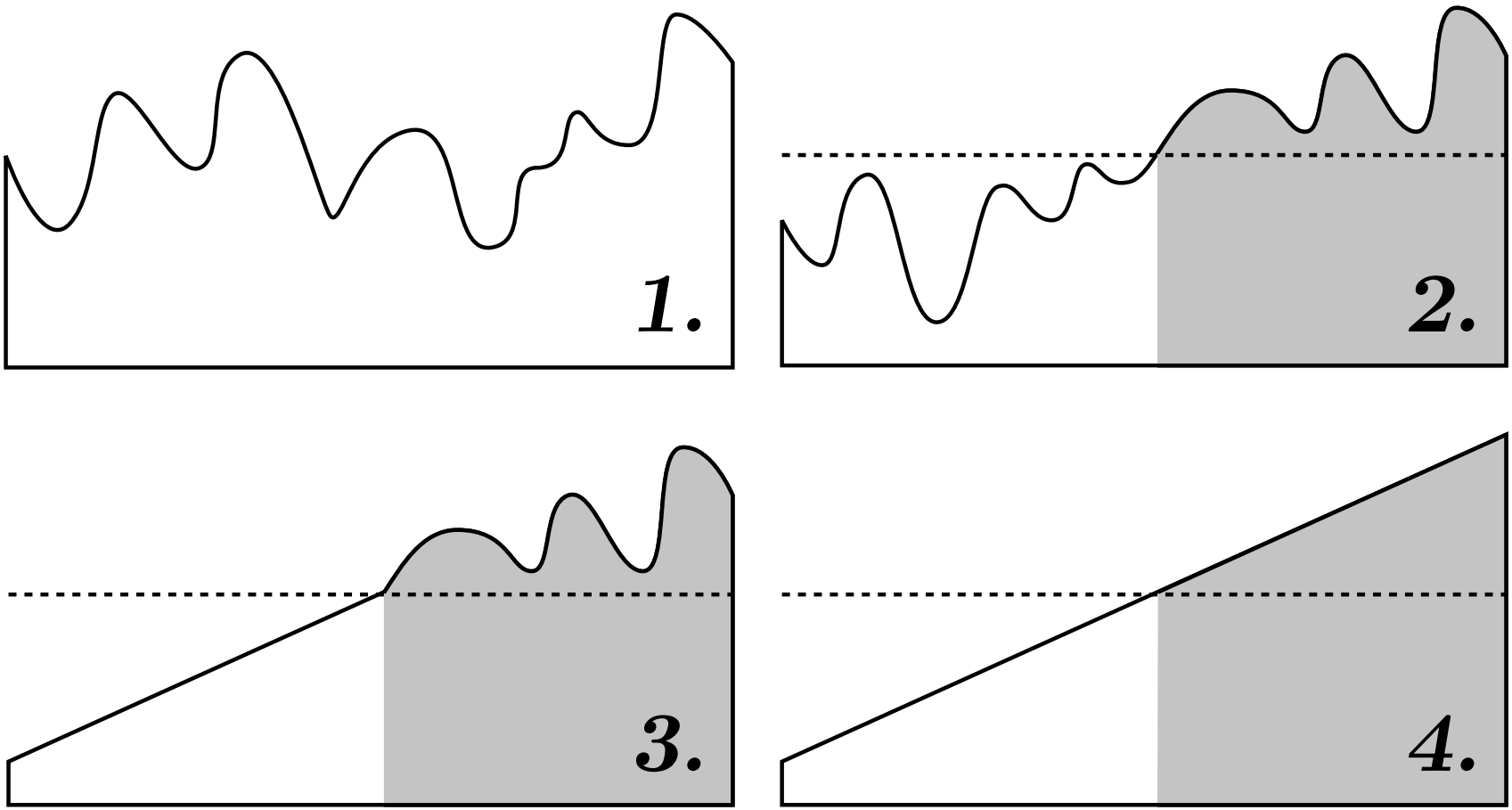
- Konieczność posiadania dodatkowej pamięci o wielkości $O(n)$
- Dodatkowy nakład czasowy związany z alokacją oraz zwalnianiem pamięci

# Rodzina algorytmów Quick Sort

**Łączna liczba operacji wykonanych przez algorytmy z rodziny Quick Sort z podziałem na metody partycjonowania oraz polityki wyboru pivota dla tablicy uporządkowanych danych**



# QuickMerge Sort - schemat

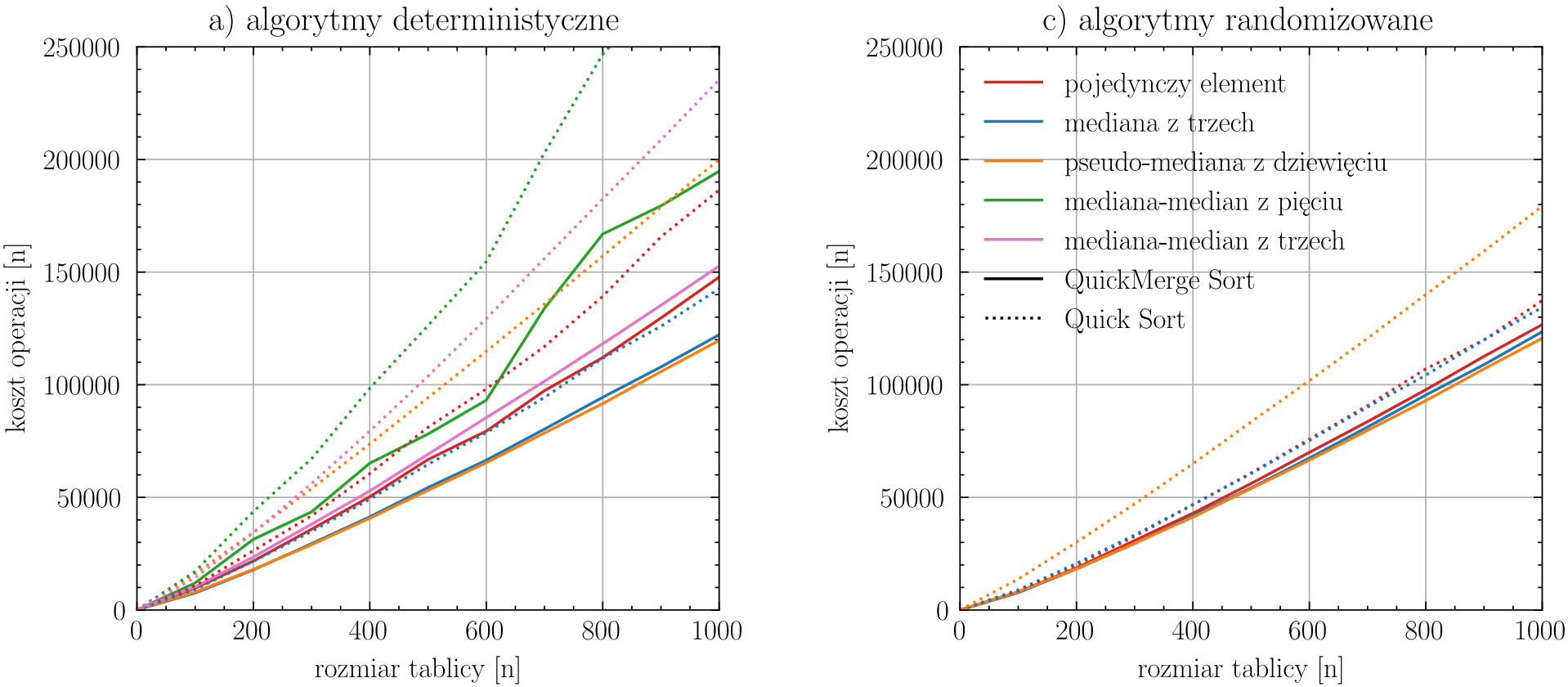


# QuickMerge Sort - pseudokod

```
1:  procedure QuickMergeSort(arr)
2:
3:      if len(arr) = 1 then end                          ▷ warunek wyjścia
4:
5:      arr_1, arr_2 ← Partition(arr)                     ▷ partycjonowanie
6:
7:      if len(arr_1) < len(arr_2) then                   ▷ sortowanie
8:          buffer ← arr_2
9:          MergeSortBySwaps(arr_1, buffer)
10:         QuickMergeSort(arr_2)
11:     else
12:         buffer ← arr_1
13:         MergeSortBySwaps(arr_2, buffer)
14:         QuickMergeSort(arr_1)
```

# Rodzina algorytmów QuickMerge Sort

**Łączny koszt operacji wykonanych przez algorytmy z rodziny QuickMerge Sort z podziałem na metody partycjonowania oraz polityki wyboru pivota dla tablicy uporządkowanych danych przy współczynniku kosztu równym $\alpha = 10.0$**

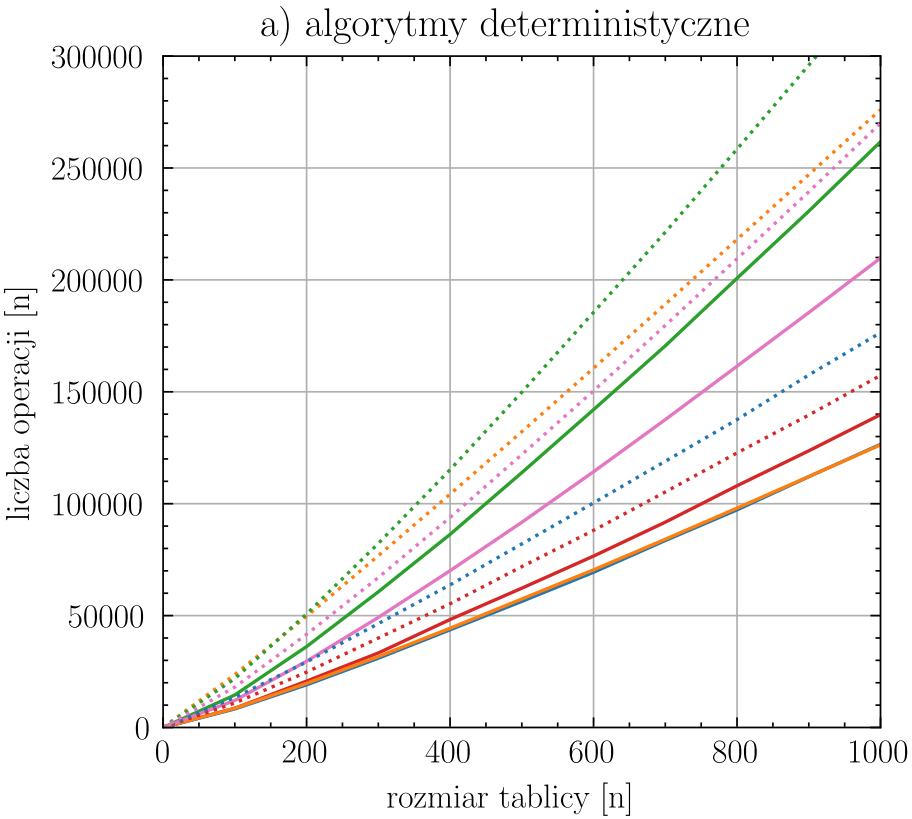


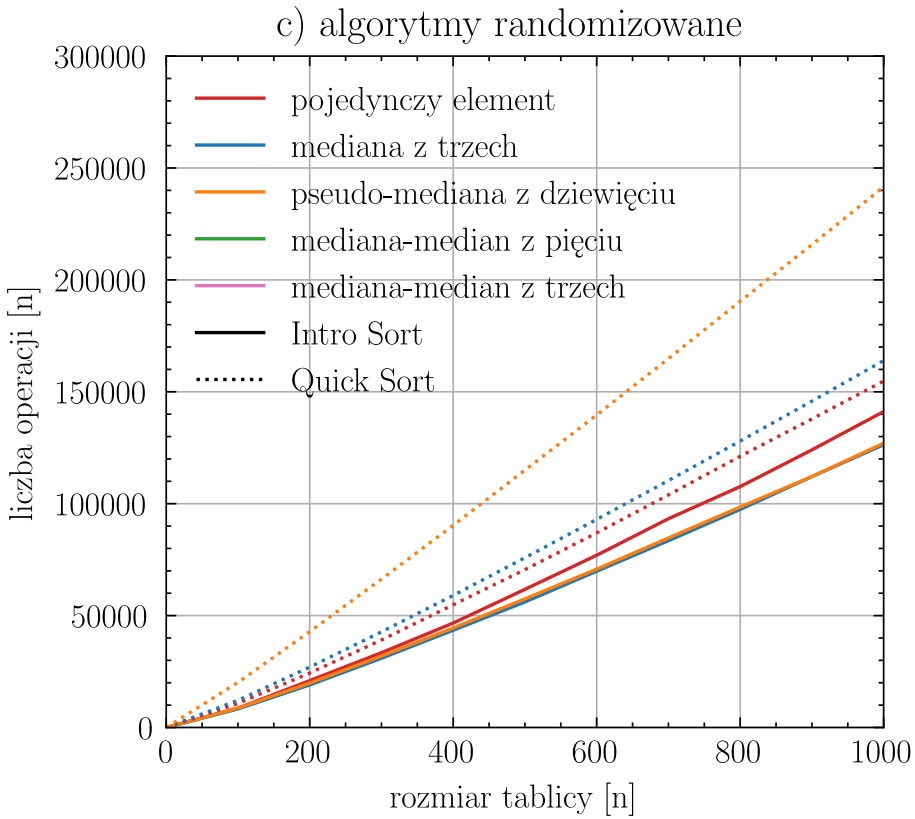
# Intro Sort - pseudokod

```
1:  procedure IntroSort(arr, depth)
2:
3:      if len(arr) maxLength then ▷ sortowanie przez wstawianie
4:          InsertionSort(arr)
5:
6:      else if depth = 0 then          ▷ sortowanie przez kopcowanie
7:          HeapSort(arr)
8:
9:      else                                        ▷ szybkie sortowanie
10:         arr_1, arr_2 ← Partition(arr)
11:         IntroSort(arr_1, depth-1)
12:         IntroSort(arr_2, depth-1)
```

# Rodzina algorytmów Intro Sort

**Łączna liczba operacji wykonanych przez algorytmy z rodziny Intro Sort z podziałem na polityki wyboru pivota dla tablicy losowych danych**



# Bibliografia

[1] Donald E. Knuth.
*The TEX Book*.
Addison-Wesley Professional, 1986.

[2] Frank Mittelbach, Michel Gossens, Johannes Braams, David Carlisle, and Chris Rowley.
*The LATEX Companion*.
Addison-Wesley Professional, 2 edition, 2004.

[3] Leslie Lamport.
*LATEX: a Document Preparation System*.
Addison Wesley, Massachusetts, 2 edition, 1994.

[4] Donald E. Knuth.
Literate programming.
*The Computer Journal*, 27(2):97–111, 1984.

[5] Michael Lesk and Brian Kernighan.