Projektowanie Algorytmów i Metody Sztucznej inteligencji - Projekt 1 - Algorytmy sortowania

Paweł Szczepaniak (249014)

27 marca 2020r.
Termin kursu: Pt 9.15

1 Wprowadzenie

Celem projektu jest przeanalizowanie wydajności algorytmów sortowania. Badania polegają na sortowaniu algorytmami quicksort, sortowaniem przez scalanie oraz sortowaniem introspektywnym 100 tablic o rozmiarach 10000, 50000, 100000, 500000 i 1000000 elementów typu całkowitoliczbowego w kilku przypadkach: dla tablic losowych, posortowanych wstępnie w 25%, 50%, 75%, 95%, 99%, 99.7% oraz tablic posortowanych w odwrotnej kolejności.

2 Opis badanych algorytmów

2.1 Quicksort

Z tablicy wybieramy element rozdzielający (zwany *pivotem*), po czym tablicę dzielimy na dwie części: do pierwszej przenosimy wszystkie elementy nie większe od *pivota*, do drugiej wszystkie większe. Następnie rekurencyjnie sortujemy osobno pierwszą i drugą część tablicy do czasu uzyskania z podziału tablicy jednoelementowej.
Złożoność obliczeniowa quicksorta:

- Typowy przypadek: $O\big(n\,log(n)\big)$
- Najgorszy przypadek: $O\big(n^2\big)$

2.2 Sortowanie przez łączenie

Algorytm dzieli rekurencyjnie zestaw danych na mniejsze aż do uzyskania zbiorów jednoelementowych, następnie ponownie scala elementy składowe jednocześnie sortując je aż do uzyskania posortowanego zbioru elementów.
Złożoność obliczeniowa sortowania przez łączenie:

- Typowy przypadek: $O\big(n\,log(n)\big)$
- Najgorszy przypadek: $O\big(n\,log(n)\big)$

2.3 Sortowanie introspektywne

Algorytm hybrydowy, łączący w sobie kilka innych algorytmów sortowania. Algorytm sortowania introspektywnego dzieli dane w podobny sposób jak quicksort, przy osiągnięciu maksymalnej dopuszczalnej głębokości rekurencji (określanej jako $2\,log(n)$, gdzie n - liczba elementów tablicy) sortuje tablicę algorytmem sortowania przez kopcowanie, jeśli liczba elementów pozostałych do posortowania w określonym fragmencie jest mniejsza od 16 sortuje tablicę algorytmem sortowania przez wstawianie.

Złożoność obliczeniowa sortowania introspektywnego:

- Typowy przypadek: $O\big(n\,log(n)\big)$
- Najgorszy przypadek: $O\big(n\,log(n)\big)$

3 Przebieg badań

3.1 Średnie czasy sortowania algorytmów [ms]

3.1.1 Quicksort

Stopień posortowania \ Rozmiar tablicy	10000	50000	100000	500000	1000000
0%	0.5921	3.3799	5.5417	30.0905	60.3499
25%	0.5737	3.2442	5.4127	28.0412	60.1753
50%	0.7196	3.2519	6.8677	37.7711	91.0206
75%	0.4412	2.1647	4.2726	22.6023	43.6433
95%	0.2970	1.2247	2.8238	14.3019	30.1658
99%	0.2510	1.2244	2.1047	10.9112	23.3590
99.7%	0.2978	0.8935	1.7716	9.5275	20.0688
Tablica posortowana w odwróconej kolejności	0.1158	0.6491	1.0808	5.2616	11.1122

Tabela 1: Średnie czasy sortowania dla algorytmu Quicksort

3.1.2 Merge Sort

Stopień posortowania \ Rozmiar tablicy	10000	50000	100000	500000	1000000
0%	1.8978	7.8945	14.0633	72.6088	150.0535
25%	1.6494	6.5969	14.9035	70.5551	143.5495
50%	1.4984	7.6449	13.6598	67.9396	138.4673
75%	1.6107	7.0574	13.4870	64.0438	131.8004
95%	1.1672	6.8070	12.6276	61.6486	129.4028
99%	1.5738	6.5679	11.9821	61.1088	125.2354
99.7%	1.6849	6.1872	12.4475	61.4511	124.9435
Tablica posortowana w odwróconej kolejności	1.5177	6.5254	12.6554	62.3692	128.0079

Tabela 2: Średnie czasy sortowania dla algorytmu Merge Sort

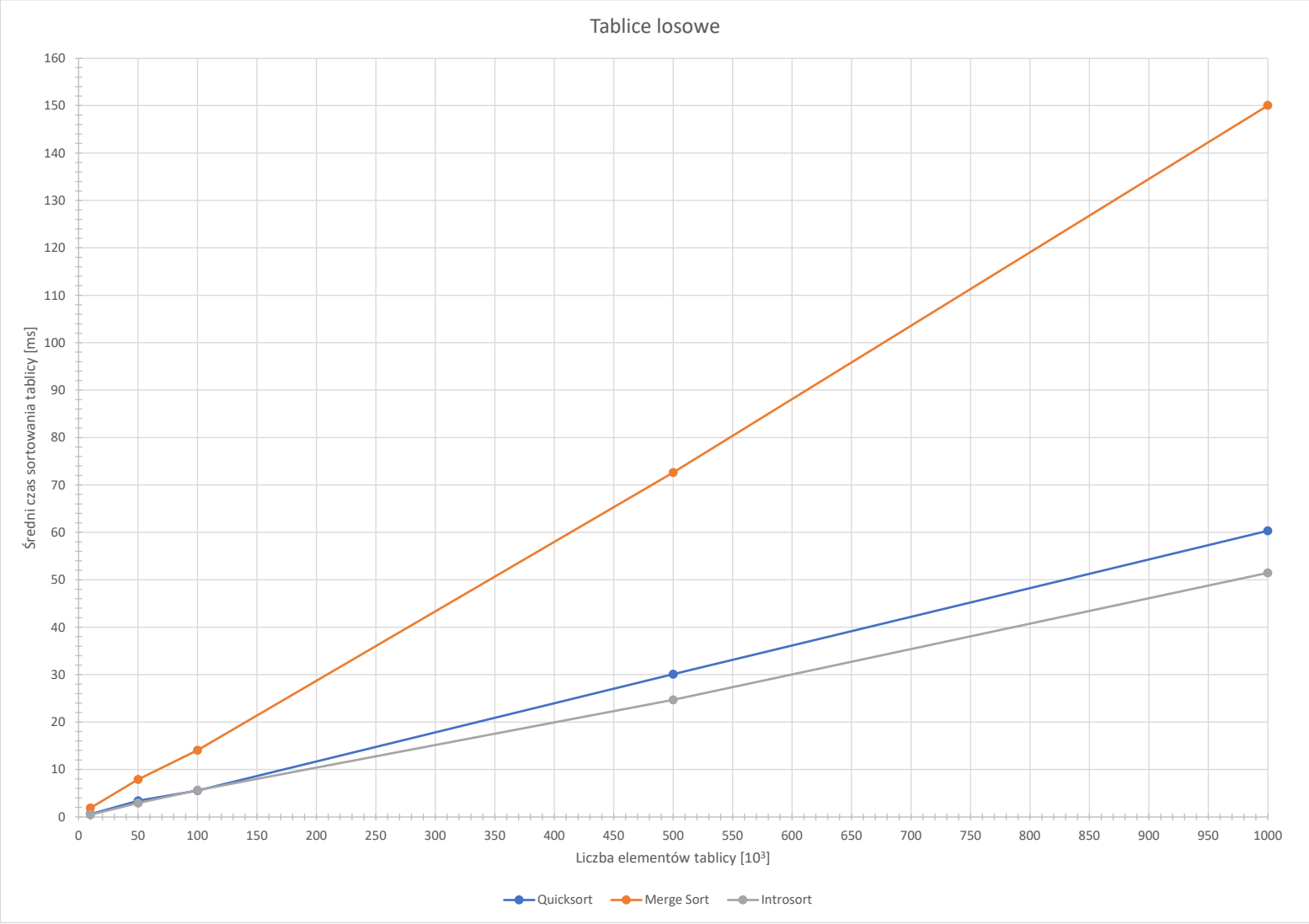
3.1.3 Introsort

Stopień posortowania \ Rozmiar tablicy	10000	50000	100000	500000	1000000
0%	0.4677	2.9124	5.6227	24.6819	51.4634
25%	0.4445	2.7134	4.2049	24.5447	49.1150
50%	0.4233	2.7779	4.5923	25.9245	55.1253
75%	0.3085	1.7666	3.1090	16.0578	34.0768
95%	0.2744	1.0912	2.3044	11.5717	24.3682
99%	0.1799	0.9174	1.6203	8.5282	18.4216
99.7%	0.1160	0.7509	1.4346	6.7343	14.4410
Tablica posortowana w odwróconej kolejności	0.0437	0.3236	0.6637	3.0384	5.9557

Tabela 3: Średnie czasy sortowania dla algorytmu Introsort

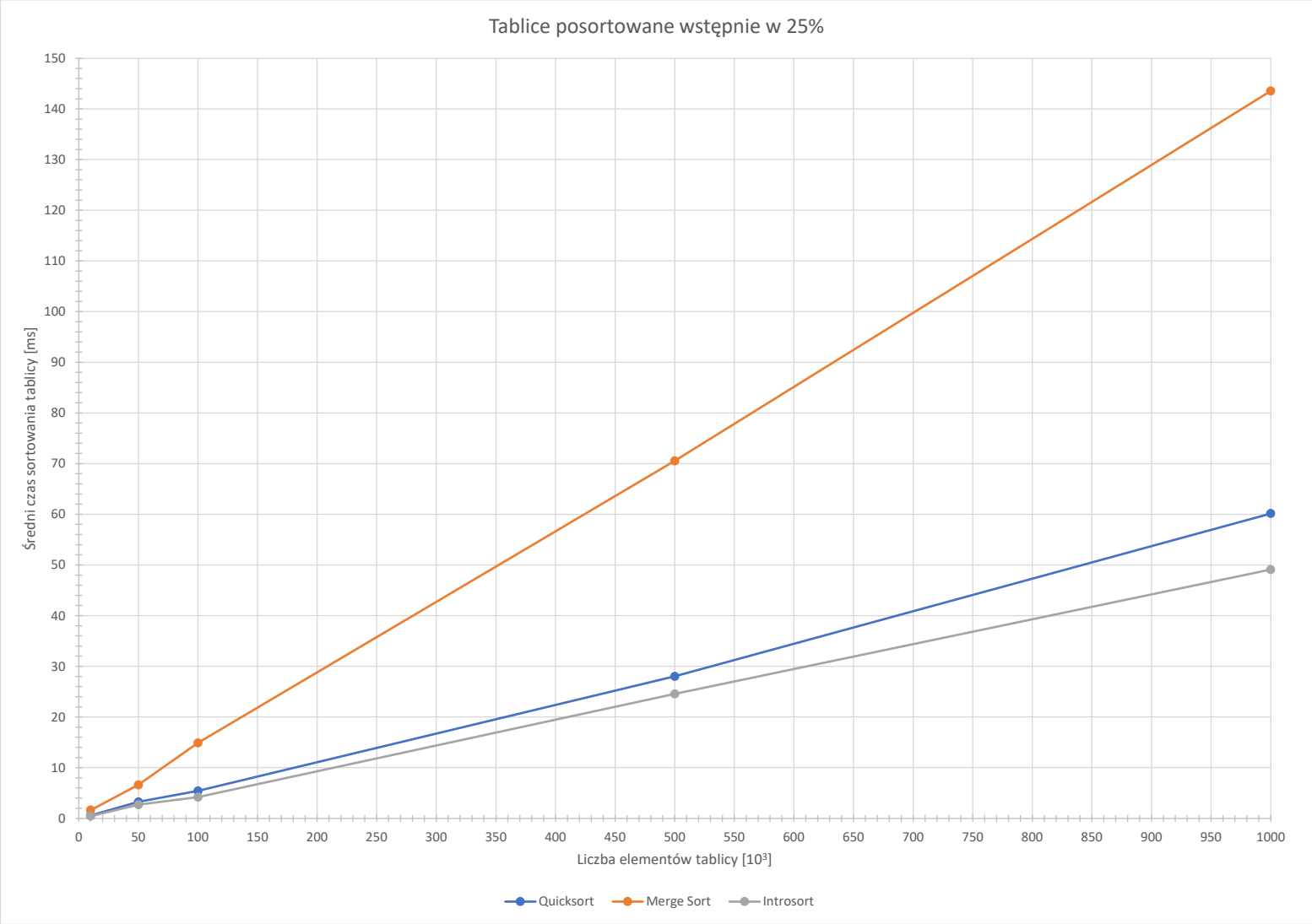
3.2 Porównanie czasów sortowania algorytmów w poszczególnych przypadkach

3.2.1 Wszystkie elementy tablicy losowe



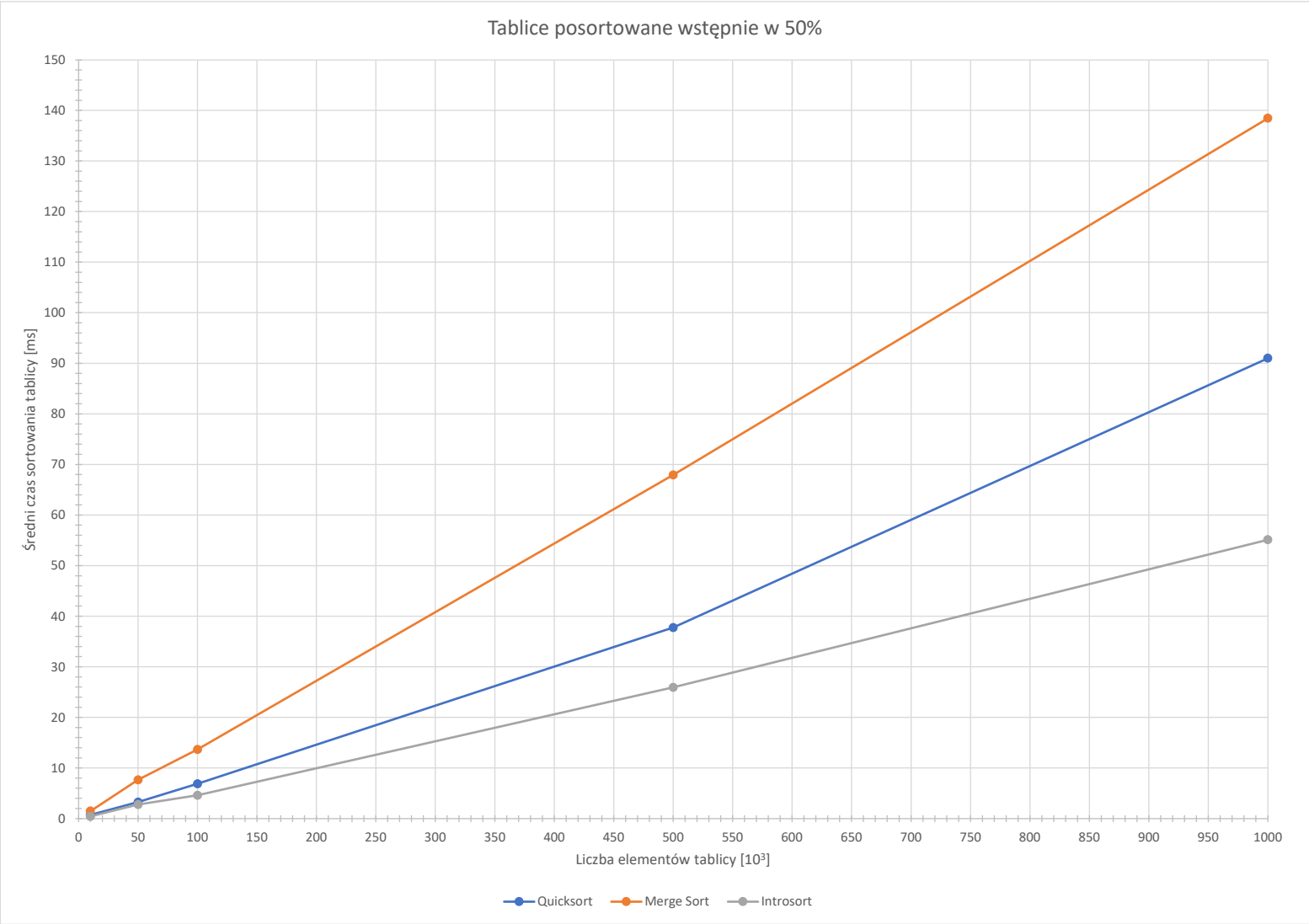
Rysunek 1: Wykres średnich czasów sortowania dla poszczególnych rozmiarów tablicy przy wszystkich elementach tablicy losowych

3.2.2 Tablice posortowane wstępnie w 25%



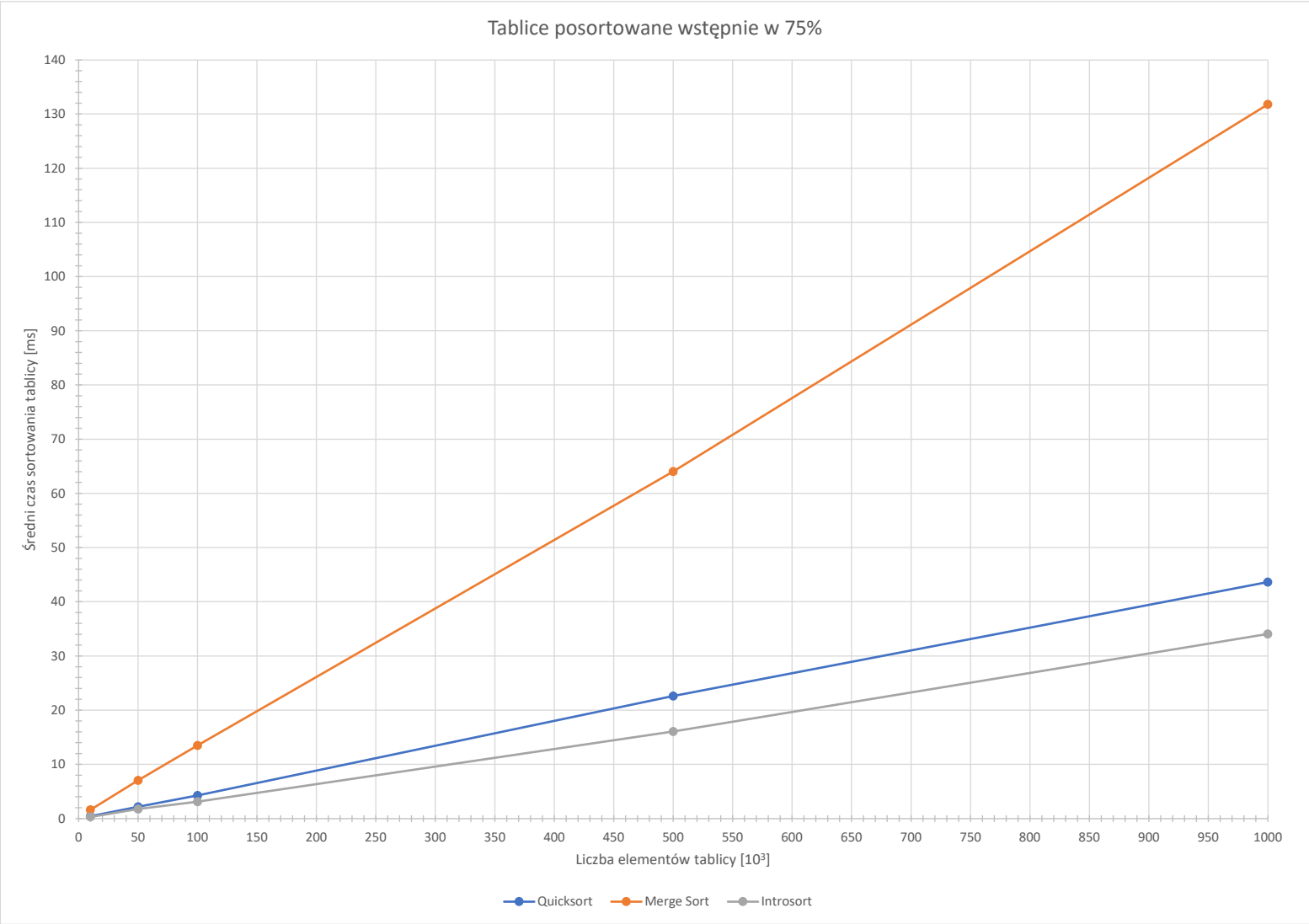
Rysunek 2: Wykres średnich czasów sortowania dla poszczególnych rozmiarów tablicy przy tablicach posortowanych wstępnie w 25%

3.2.3 Tablice posortowane wstępnie w 50%



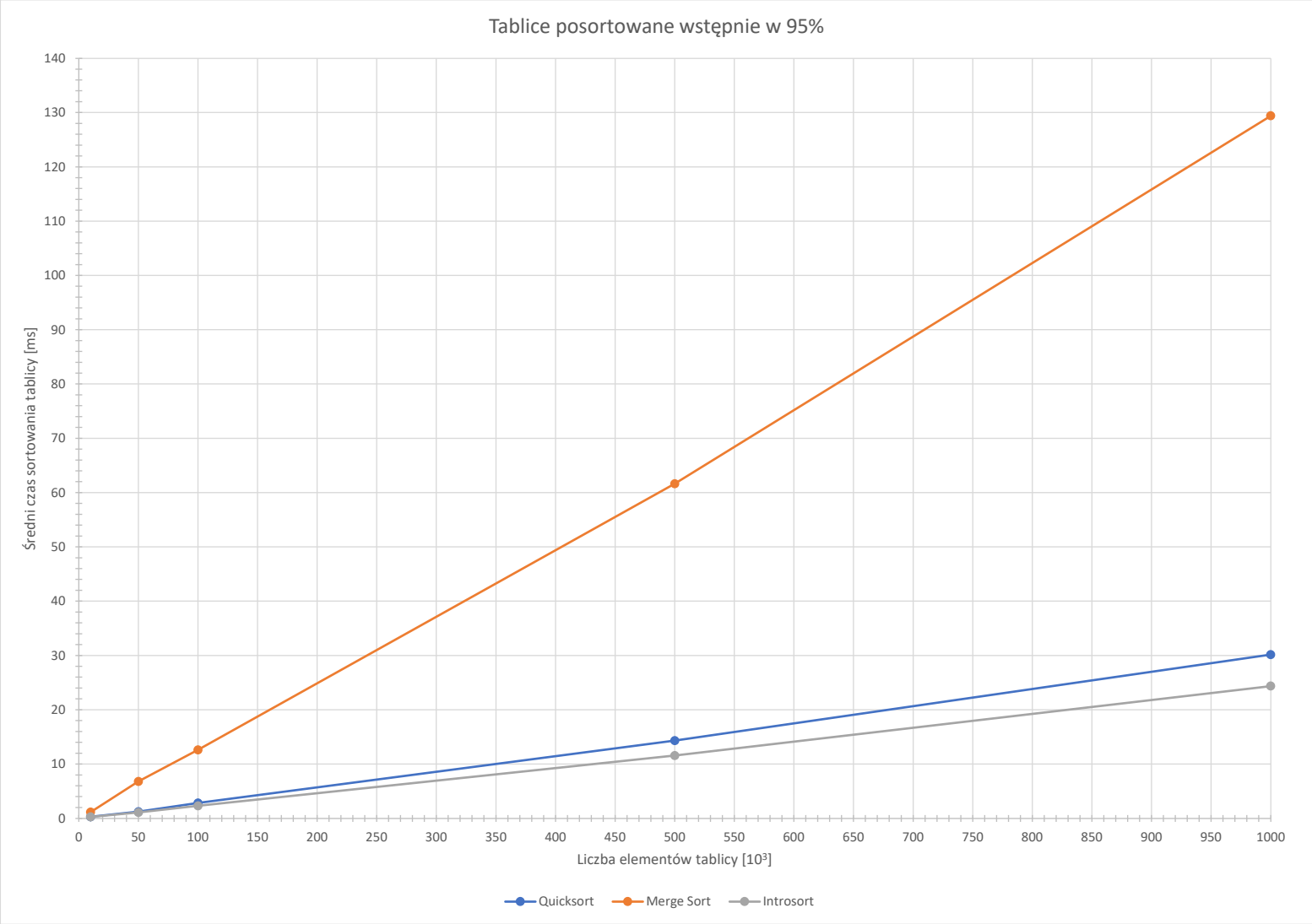
Rysunek 3: Wykres średnich czasów sortowania dla poszczególnych rozmiarów tablicy przy tablicach posortowanych wstępnie w 50%

3.2.4 Tablice posortowane wstępnie w 75%



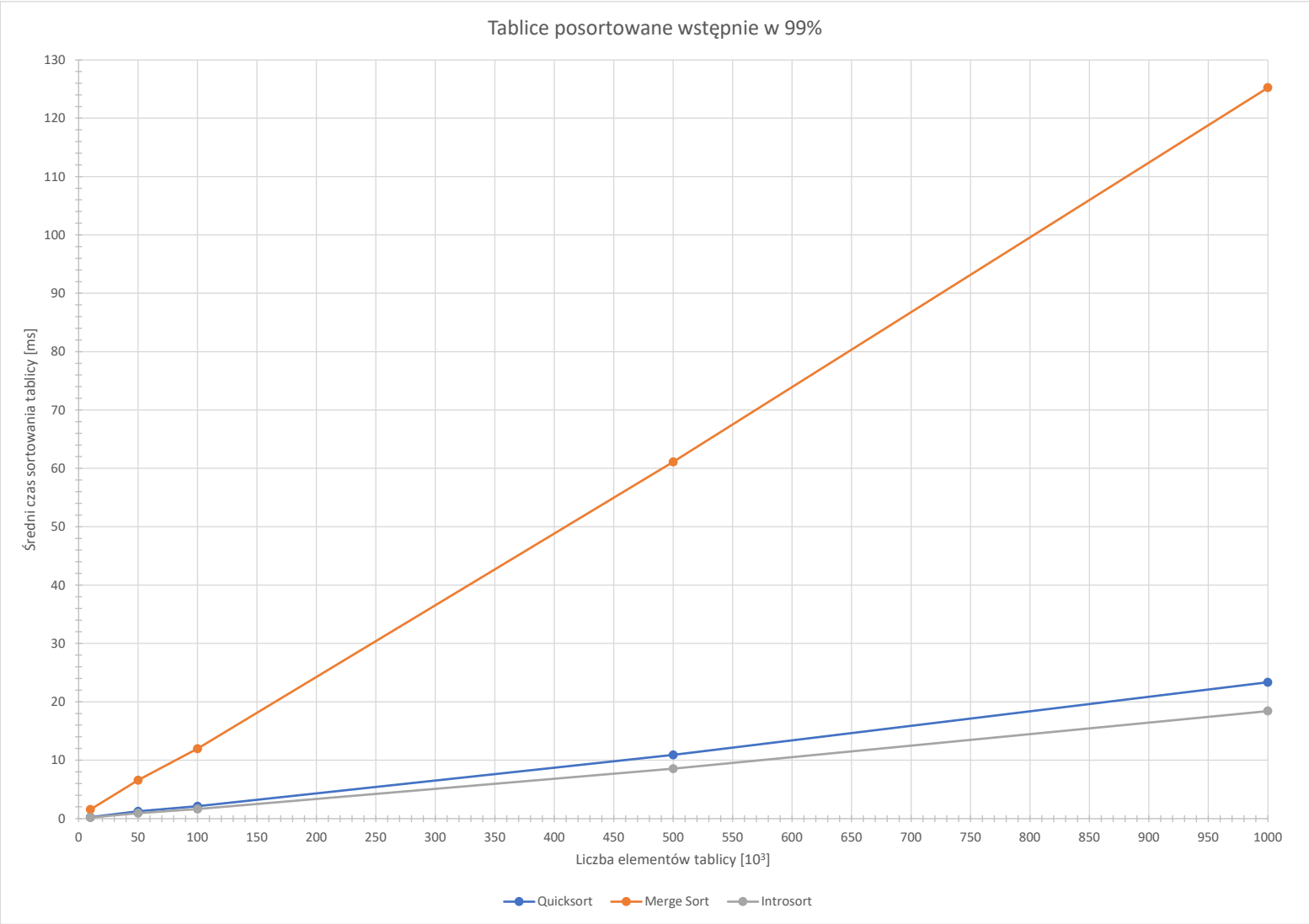
Rysunek 4: Wykres średnich czasów sortowania dla poszczególnych rozmiarów tablicy przy tablicach posortowanych wstępnie w 75%

3.2.5 Tablice posortowane wstępnie w 95%



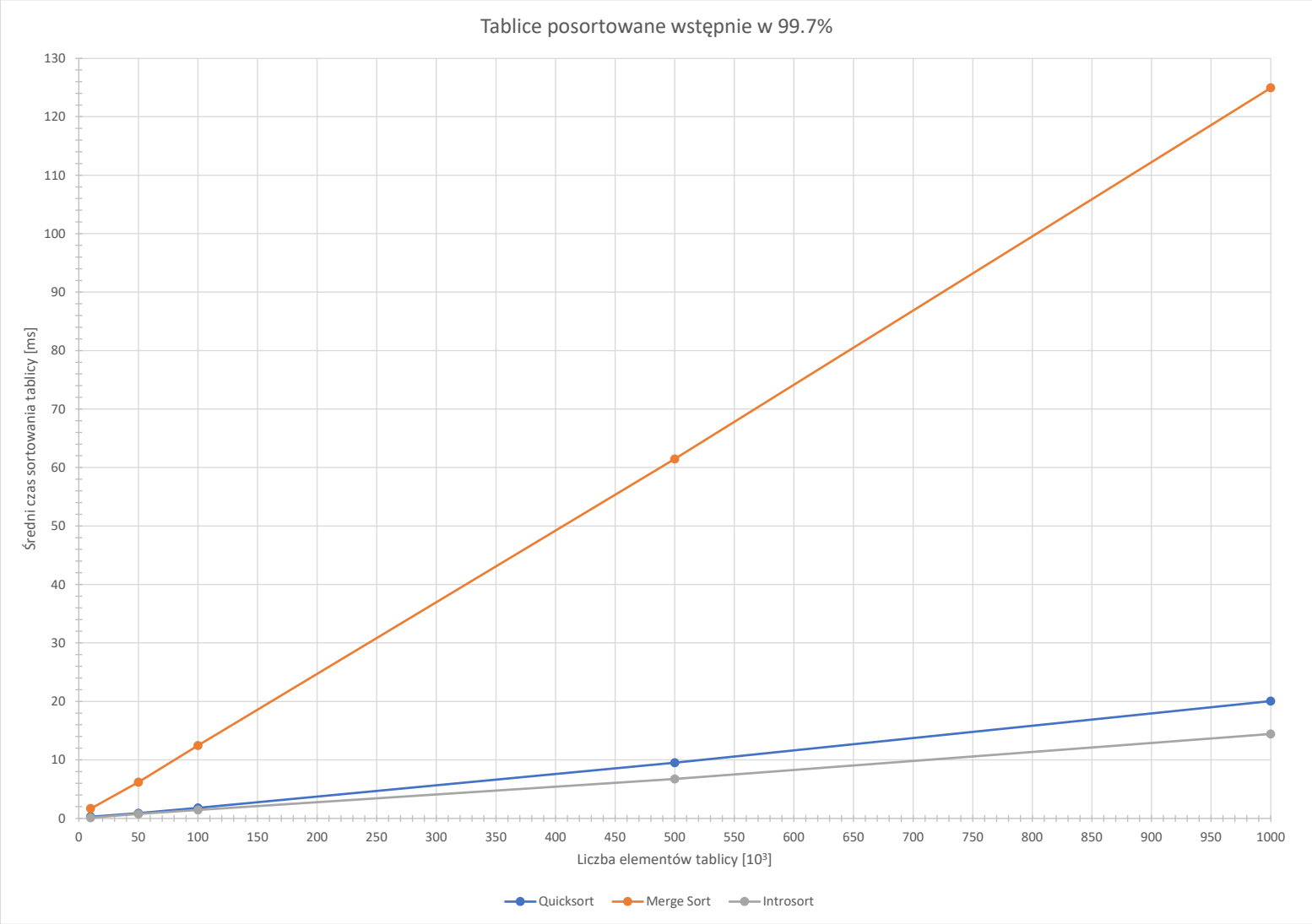
Rysunek 5: Wykres średnich czasów sortowania dla poszczególnych rozmiarów tablicy przy tablicach posortowanych wstępnie w 95%

3.2.6 Tablice posortowane wstępnie w 99%



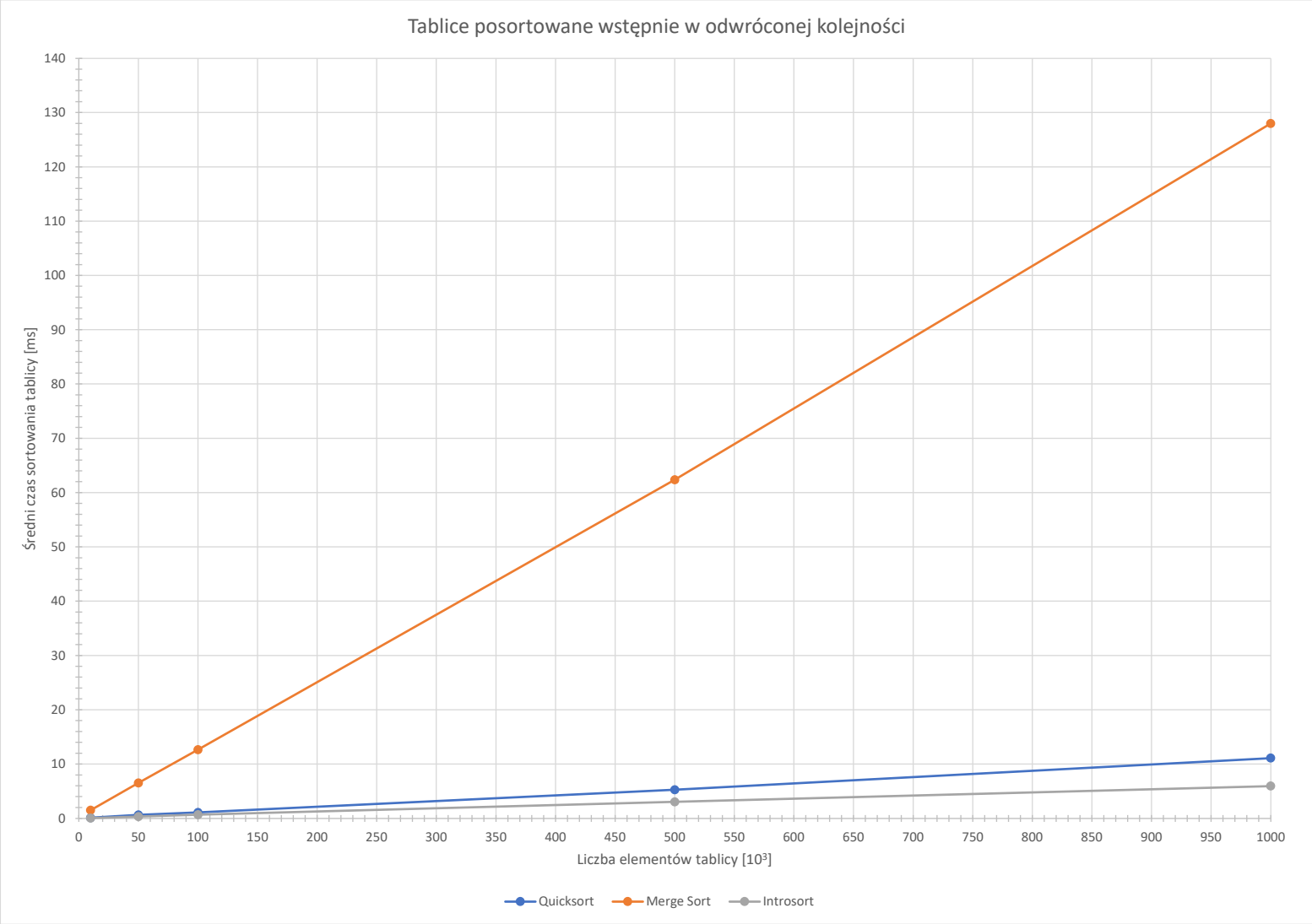
Rysunek 6: Wykres średnich czasów sortowania dla poszczególnych rozmiarów tablicy przy tablicach posortowanych wstępnie w 99%

3.2.7 Tablice posortowane wstępnie w 99.7%



Rysunek 7: Wykres średnich czasów sortowania dla poszczególnych rozmiarów tablicy przy tablicach posortowanych wstępnie w 99.7%

3.2.8 Tablice posortowane wstępnie w odwróconej kolejności



Rysunek 8: Wykres średnich czasów sortowania dla poszczególnych rozmiarów tablicy przy tablicach posortowanych wstępnie w odwróconej kolejności

4 Wnioski

- Warto zauważyć, że w przypadku wstępnego posortowania 50% elementów tablicy średni czas sortowania algorytmami Quicksort i Introsort znacząco rośnie. Jest to związane w wyborem elementu środkowego jako *pivota* w implementacji obu tych algorytmów.
- Zgodnie z założeniami, algorytm Quicksort jest wolniejszy od algorytmu sortowania introspektywnego.
- Algorytm sortowania przez łączenie okazał się najwolniejszym z badanych algorytmów.

5 Bibliografia

- Wikipedia
- geeksforgeeks.org
- cppreference.com
- cplusplus.com
- stackoverflow.com