Sprawozdanie z algorytmów sortujących

Aleksander Pawlikowski

Antonina Stobińska

## Heap sort

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 500 | 1000 | 2000 | 4000 | 8000 | 16000 | 32000 | 64000 | 128000 | 256000 |
| Fill\_increasing | 0 | 0 | 0 | 0,001 | 0,002 | 0,005 | 0,011 | 0,023 | 0,051 | 0,108 |
| Fill\_decreasing | 0 | 0 | 0 | 0,001 | 0,003 | 0,006 | 0,014 | 0,032 | 0,067 | 0,145 |
| Fill\_vshape | 0 | 0 | 0 | 0,001 | 0,003 | 0,007 | 0,013 | 0,029 | 0,061 | 0,132 |
| Fill\_random | 0 | 0 | 0 | 0,001 | 0,002 | 0,006 | 0,013 | 0,027 | 0,058 | 0,125 |

* Algorytm wykonuje się w podobnym czasie dla wszystkich typów danych, nie ważne czy są uporządkowane rosnąco czy malejąco, w najgorszym wypadku czas wykonania algorytmu jest nieznacznie dłuższy
* Z wykresu możemy odczytać złożoność logarytmiczną

Quick sort last (ostatnia wartość w tablicy jest elementem rozdzielającym)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 500 | 1000 | 2000 | 4000 | 8000 | 16000 | 32000 | 64000 | 128000 | 256000 |
| Fill\_increasing | 0,001 | 0,004 | 0,014 | 0,057 | 0,227 | 0,911 | 3,491 |  |  |  |
| Fill\_decreasing | 0,001 | 0,003 | 0,011 | 0,047 | 0,187 | 0,75 | 2,902 |  |  |  |
| Fill\_vshape | 0,001 | 0,003 | 0,014 | 0,059 | 0,236 | 0,943 | 3,599 |  |  |  |
| Fill\_random | 0 | 0,001 | 0 | 0,001 | 0,002 | 0,004 | 0,008 | 0,018 | 0,038 | 0,085 |

* Algorytm wykonuje się w podobnym czasie dla prawie wszystkich typów danych, nie ważne czy są uporządkowane rosnąco czy malejąco, w najgorszym wypadku czas wykonania algorytmu jest nieznacznie dłuższy. Jedynie dla losowych danych jest najszybciej wykonany.
* Z wykresu możemy odczytać złożoność logarytmiczną dla prawie wszystkich typów danych

Quick sort rand (losowa wartość z tablicy jest elementem rozdzielającym)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 500 | 1000 | 2000 | 4000 | 8000 | 16000 | 32000 | 64000 | 128000 | 256000 |
| Fill\_increasing | 0 | 0 | 0 | 0,001 | 0,001 | 0,002 | 0,006 | 0,012 | 0,025 | 0,055 |
| Fill\_decreasing | 0 | 0,001 | 0 | 0 | 0,001 | 0,003 | 0,007 | 0,016 | 0,035 | 0,083 |
| Fill\_vshape | 0 | 0 | 0,001 | 0,001 | 0,002 | 0,004 | 0,007 | 0,016 | 0,036 | 0,076 |
| Fill\_random | 0 | 0 | 0,001 | 0,001 | 0,002 | 0,004 | 0,009 | 0,019 | 0,043 | 0,1 |

* Algorytm wykonuje się w podobnym czasie dla wszystkich typów danych, nie ważne czy są uporządkowane rosnąco czy malejąco, w najgorszym wypadku czas wykonania algorytmu jest nieznacznie dłuższy, są jednak małe różnice w czasie
* Z wykresu możemy odczytać złożoność logarytmiczną

## Selection sort

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 500 | 1000 | 2000 | 4000 | 8000 | 16000 | 32000 | 64000 | 128000 | 256000 |
| Fill\_increasing | 0 | 0 | 0,015 | 0,031 | 0,157 | 0,593 | 2,313 | 8,703 | 34,686 | 135,651 |
| Fill\_decreasing | 0 | 0 | 0,015 | 0,063 | 0,234 | 0,906 | 3,453 | 13,234 | 52,217 | 206,117 |
| Fill\_vshape | 0 | 0,015 | 0,016 | 0,078 | 0,312 | 1,235 | 4,656 | 17,859 | 70,2 | 273,098 |
| Fill\_random | 0 | 0 | 0,016 | 0,031 | 0,187 | 0,672 | 2,672 | 10,031 | 39,014 | 152,978 |

* Algorytm wykonuje się w podobnym czasie dla wszystkich typów danych, nie ważne czy są uporządkowane rosnąco czy malejąco, w najgorszym wypadku czas wykonania algorytmu jest nieznacznie dłuższy; czas wykonywania algorytmu szybko rośnie dla większej ilości danych
* Z wykresu możemy odczytać złożoność kwadratową w najgorszym wypadku

## Insertion sort

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 500 | 1000 | 2000 | 4000 | 8000 | 16000 | 32000 | 64000 | 128000 | 256000 |
| Fill\_increasing | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Fill\_decreasing | 0 | 0,016 | 0,015 | 0,078 | 0,328 | 1,313 | 4,89 | 19 | 75,34 | 302,035 |
| Fill\_vshape | 0 | 0 | 0,016 | 0,031 | 0,172 | 0,641 | 2,562 | 9,593 | 37,827 | 150,385 |
| Fill\_random | 0 | 0,015 | 0 | 0,047 | 0,156 | 0,641 | 2,562 | 9,609 | 37,639 | 150,681 |

* W tym algorytmie widać duże rozbieżności czasowe między różnymi typami danych
* Dla uporządkowanych rosnąco nie zajmuje on czasu
* W najgorszym wypadku, czyli dane uporządkowane malejąco, algorytm wykonuje się bardzo długo i szybko rośnie jego czas wykonywania zależnie od ilości danych, w tym wypadku jest to złożoność kwadratowa