|  |  |  |  |
| --- | --- | --- | --- |
| Projektowanie algorytmów i metody sztucznej inteligencji  Automatyka i Robotyka Wydział Elektroniki  Politechnika Wrocławska | | Projekt 2: Sortowanie szybkie, przez scalanie i introspektywne | |
| Piotr Bogdoł | Środa  17:05 – 18:45 | | Dr inż. Andrzej Rusiecki |

# Wstęp:

W projekcie zostały zaimplementowane i przetestowane 3 algorytmy sortowania:  
– sortowanie szybkie,  
–sortowanie przez scalanie,  
–sortowanie introspektywne.

Projekt opiera się na klasie Sortowanie, w której umieszczone implementacje wszystkich trzech sortowań oraz wszystkie metody pomocnicze przy ich testowaniu, jak sprawdzanie poprawności posortowanych tablic rosnąco lub malejąco. Sortowanie przez scalanie zostało zaimplementowane na dwa sposoby, tak aby mogło sortować rosnąco i malejąco dany fragment tablicy.

# Sortowanie szybkie:

Wyniki czasu sortowania w zależności od ilości elementów oraz wstępnego posortowania tablicy (wartość -100% oznacza odwrotne posortowanie całej tablicy):

Tabela 1. Zestawienie wyników pomiarów czasu w zależności od ilości elementów i ich posortowania dla sortowania szybkiego.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rozmiar | Czas[ms] | Posortowanie | Rozmiar | Czas[ms] | Posortowanie | Rozmiar | Czas[ms] | Posortowanie |
| 100000 | 13,36 | 0% | 100000 | 12,96 | 25% | 100000 | 14,16 | 50% |
| 200000 | 23,19 | 0% | 200000 | 25,31 | 25% | 200000 | 24,63 | 50% |
| 500000 | 62,12 | 0% | 500000 | 60,81 | 25% | 500000 | 62,76 | 50% |
| 800000 | 112,32 | 0% | 800000 | 106,04 | 25% | 800000 | 109,47 | 50% |
| 1000000 | 137,31 | 0% | 1000000 | 136,18 | 25% | 1000000 | 137,07 | 50% |
| 2000000 | 268,91 | 0% | 2000000 | 261,71 | 25% | 2000000 | 263,2 | 50% |
| 4000000 | 557,24 | 0% | 4000000 | 538,95 | 25% | 4000000 | 531,46 | 50% |
| 100000 | 11,92 | 75% | 100000 | 9,2 | 90% | 100000 | 8,29 | 95% |
| 200000 | 22 | 75% | 200000 | 18,59 | 90% | 200000 | 18,96 | 95% |
| 500000 | 54,26 | 75% | 500000 | 50,93 | 90% | 500000 | 47,08 | 95% |
| 800000 | 95,11 | 75% | 800000 | 86,41 | 90% | 800000 | 81,11 | 95% |
| 1000000 | 120,97 | 75% | 1000000 | 111,05 | 90% | 1000000 | 103,88 | 95% |
| 2000000 | 234,32 | 75% | 2000000 | 213,12 | 90% | 2000000 | 203,75 | 95% |
| 4000000 | 480,48 | 75% | 4000000 | 448,72 | 90% | 4000000 | 427,53 | 95% |
| 100000 | 8,83 | 99,7% | 100000 | 7,29 | -100% |
| 200000 | 15,9 | 99,7% | 200000 | 14,81 | -100% |
| 500000 | 42,89 | 99,7% | 500000 | 42,58 | -100% |
| 800000 | 74,83 | 99,7% | 800000 | 66,61 | -100% |
| 1000000 | 92,19 | 99,7% | 1000000 | 87,95 | -100% |
| 2000000 | 182,68 | 99,7% | 2000000 | 185,32 | -100% |
| 4000000 | 383,25 | 99,7% | 4000000 | 387,2 | -100% |

# Sortowanie przez scalanie:

Wyniki czasu sortowania w zależności od ilości elementów oraz wstępnego posortowania tablicy (wartość -100% oznacza odwrotne posortowanie całej tablicy):

Tabela 2. Zestawienie wyników pomiarów czasu w zależności od ilości elementów i ich posortowania dla sortowania przez scalanie.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rozmiar | Czas[ms] | Posortowanie | Rozmiar | Czas[ms] | Posortowanie | Rozmiar | Czas[ms] | Posortowanie |
| 100000 | 15,62 | 0% | 100000 | 14,08 | 25% | 100000 | 12,96 | 50% |
| 200000 | 32,64 | 0% | 200000 | 29,63 | 25% | 200000 | 26,68 | 50% |
| 500000 | 86,15 | 0% | 500000 | 76,91 | 25% | 500000 | 67,48 | 50% |
| 800000 | 141,64 | 0% | 800000 | 127,47 | 25% | 800000 | 111,6 | 50% |
| 1000000 | 176,75 | 0% | 1000000 | 158,8 | 25% | 1000000 | 141,67 | 50% |
| 2000000 | 368,95 | 0% | 2000000 | 333,67 | 25% | 2000000 | 295,62 | 50% |
| 4000000 | 761,11 | 0% | 4000000 | 688,9 | 25% | 4000000 | 614,66 | 50% |
| 100000 | 10,94 | 75% | 100000 | 7,83 | 90% | 100000 | 8,89 | 95% |
| 200000 | 21,69 | 75% | 200000 | 20,42 | 90% | 200000 | 18,24 | 95% |
| 500000 | 57,9 | 75% | 500000 | 55,29 | 90% | 500000 | 51,9 | 95% |
| 800000 | 96,14 | 75% | 800000 | 87,56 | 90% | 800000 | 84,11 | 95% |
| 1000000 | 123,71 | 75% | 1000000 | 112,02 | 90% | 1000000 | 108,43 | 95% |
| 2000000 | 256,62 | 75% | 2000000 | 235,08 | 90% | 2000000 | 225,84 | 95% |
| 4000000 | 529,18 | 75% | 4000000 | 484,37 | 90% | 4000000 | 470,67 | 95% |
| 100000 | 10,61 | 99,7% | 100000 | 8,6 | -100% |
| 200000 | 19,03 | 99,7% | 200000 | 18,38 | -100% |
| 500000 | 50,54 | 99,7% | 500000 | 49,93 | -100% |
| 800000 | 81,25 | 99,7% | 800000 | 80,35 | -100% |
| 1000000 | 102,97 | 99,7% | 1000000 | 104,99 | -100% |
| 2000000 | 218,56 | 99,7% | 2000000 | 216,83 | -100% |
| 4000000 | 452,72 | 99,7% | 4000000 | 452,9 | -100% |

# Sortowanie introspektywne:

Wyniki czasu sortowania w zależności od ilości elementów oraz wstępnego posortowania tablicy (wartość -100% oznacza odwrotne posortowanie całej tablicy):

Tabela 3. Zestawienie wyników pomiarów czasu w zależności od ilości elementów i ich posortowania dla sortowanie introspektywnego.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rozmiar | Czas[ms] | Posortowanie | Rozmiar | Czas[ms] | Posortowanie | Rozmiar | Czas[ms] | Posortowanie |
| 100000 | 12,48 | 0% | 100000 | 11,39 | 25% | 100000 | 11,07 | 50% |
| 200000 | 26,05 | 0% | 200000 | 24,04 | 25% | 200000 | 21,36 | 50% |
| 500000 | 63,04 | 0% | 500000 | 62,89 | 25% | 500000 | 56,36 | 50% |
| 800000 | 105,15 | 0% | 800000 | 102,35 | 25% | 800000 | 95,2 | 50% |
| 1000000 | 137,08 | 0% | 1000000 | 130,44 | 25% | 1000000 | 122,17 | 50% |
| 2000000 | 302,33 | 0% | 2000000 | 294,16 | 25% | 2000000 | 278,6 | 50% |
| 4000000 | 630,19 | 0% | 4000000 | 616,37 | 25% | 4000000 | 588,75 | 50% |
| 100000 | 9,85 | 75% | 100000 | 8,56 | 90% | 100000 | 8,43 | 95% |
| 200000 | 20,74 | 75% | 200000 | 19,81 | 90% | 200000 | 19,06 | 95% |
| 500000 | 54,73 | 75% | 500000 | 49,48 | 90% | 500000 | 48,19 | 95% |
| 800000 | 89,91 | 75% | 800000 | 81,46 | 90% | 800000 | 78,3 | 95% |
| 1000000 | 113,99 | 75% | 1000000 | 103,24 | 90% | 1000000 | 98,66 | 95% |
| 2000000 | 239,32 | 75% | 2000000 | 210,16 | 90% | 2000000 | 203,13 | 95% |
| 4000000 | 526,31 | 75% | 4000000 | 438,09 | 90% | 4000000 | 416,09 | 95% |
| 100000 | 8,43 | 99,7% | 100000 | 8,88 | -100% |
| 200000 | 17,02 | 99,7% | 200000 | 17,15 | -100% |
| 500000 | 41,64 | 99,7% | 500000 | 44,61 | -100% |
| 800000 | 70,16 | 99,7% | 800000 | 73,01 | -100% |
| 1000000 | 88,87 | 99,7% | 1000000 | 90,95 | -100% |
| 2000000 | 181,08 | 99,7% | 2000000 | 189,05 | -100% |
| 4000000 | 372,43 | 99,7% | 4000000 | 383,18 | -100% |

# Porównanie 3 sortowań:

1. Porównanie czasu sortowania 2000000 elementów w zależności od wstępnego posortowania:

Tabela 4. Zestawienie czas sortowania w zależności od wstępnego posortowania tablic dla 3 sortowań.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sortowanie szybkie | | Sortowanie przez scalanie | | Sortowanie introspektywne | |
| Posortowanie | Czas [ms] | Posortowanie | Czas [ms] | Posortowanie | Czas [ms] |
| -100% | 185,32 | -100% | 216,83 | -100% | 189,05 |
| 0% | 268,91 | 0% | 368,95 | 0% | 302,33 |
| 25% | 261,71 | 25% | 333,67 | 25% | 294,16 |
| 50% | 263,2 | 50% | 295,62 | 50% | 278,6 |
| 75% | 234,32 | 75% | 256,62 | 75% | 239,32 |
| 90% | 213,12 | 90% | 235,08 | 90% | 210,16 |
| 95% | 203,75 | 95% | 225,84 | 95% | 203,13 |
| 99,70% | 182,68 | 99,70% | 218,56 | 99,70% | 181,08 |

1. Porównanie czasu sortowania dla różnej ilości elementów wcześniej nie posortowanych:

Tabela 5. Zestawienie czasu sortowania od ilości elementów dla nieposortowanych wcześniej elementów.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sortowanie szybkie | | Sortowanie przez scalanie | | Sortowanie introspektywne | |
| Rozmiar | Czas [ms] | Rozmiar | Czas [ms] | Rozmiar | Czas [ms] |
| 100000 | 13,36 | 100000 | 15,62 | 100000 | 12,48 |
| 200000 | 23,19 | 200000 | 32,64 | 200000 | 26,05 |
| 500000 | 62,12 | 500000 | 86,15 | 500000 | 63,04 |
| 800000 | 112,32 | 800000 | 141,64 | 800000 | 105,15 |
| 1000000 | 137,31 | 1000000 | 176,75 | 1000000 | 137,08 |
| 2000000 | 268,91 | 2000000 | 368,95 | 2000000 | 302,33 |
| 4000000 | 557,24 | 4000000 | 761,11 | 4000000 | 630,19 |

1. Porównanie czasu sortowania dla różnej ilości elementów wcześniej posortowanych w 99,7%:

Tabela 6. Zestawienie czasu sortowania od ilości elementów dla posortowanych wcześniej elementów w 99,7%.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sortowanie szybkie | | Sortowanie przez scalanie | | Sortowanie introspektywne | |
| Rozmiar | Czas [ms] | Rozmiar | Czas [ms] | Rozmiar | Czas [ms] |
| 100000 | 8,83 | 100000 | 10,61 | 100000 | 8,43 |
| 200000 | 15,9 | 200000 | 19,03 | 200000 | 17,02 |
| 500000 | 42,89 | 500000 | 50,54 | 500000 | 41,64 |
| 800000 | 74,83 | 800000 | 81,25 | 800000 | 70,16 |
| 1000000 | 92,19 | 1000000 | 102,97 | 1000000 | 88,87 |
| 2000000 | 182,68 | 2000000 | 218,56 | 2000000 | 181,08 |
| 4000000 | 383,25 | 4000000 | 452,72 | 4000000 | 372,43 |

1. Porównanie czasu sortowania dla różnej ilości elementów wcześniej odwrotnie posortowanych:

Tabela 7. Zestawienie czasu sortowania od ilości elementów dla odwrotnie posortowanych wcześniej elementów.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sortowanie szybkie | | Sortowanie przez scalanie | | Sortowanie introspektywne | |
| Rozmiar | Czas [ms] | Rozmiar | Czas [ms] | Rozmiar | Czas [ms] |
| 100000 | 7,29 | 100000 | 8,6 | 100000 | 8,88 |
| 200000 | 14,81 | 200000 | 18,38 | 200000 | 17,15 |
| 500000 | 42,58 | 500000 | 49,93 | 500000 | 44,61 |
| 800000 | 66,61 | 800000 | 80,35 | 800000 | 73,01 |
| 1000000 | 87,95 | 1000000 | 104,99 | 1000000 | 90,95 |
| 2000000 | 185,32 | 2000000 | 216,83 | 2000000 | 189,05 |
| 4000000 | 387,2 | 4000000 | 452,9 | 4000000 | 383,18 |

# Analiza wyników i wnioski:

Sortowanie szybkie:

Analizując wyniki dla sortowania szybkiego można stwierdzić, że złożoność obliczeniowa dla wszystkich przypadków, nie zależnie od sposobu wcześniejszego posortowania lub jego braku, jest logarytmiczna. Na podstawie danych można stwierdzić również, że początkowe posortowanie elementów w równym stopniu nie zmniejsza czasu sortowania w liniowy sposób. Algorytm sortuje w podobnym czasie elementy wstępnie posortowane w 25% i 50% oraz nie posortowane wcale. Dopiero różnica w czasie sortowania jest znacząca dla wstępnego posortowania tablicy w 75% i większej. Porównując czas sortowania dla wstępnie posortowanych elementów w 99,7% i odwrotnie posortowanych elementów, można stwierdzić że odwrotne posortowanie nie ma większego wpływu na złożoność obliczeniową algorytmu.

Sortowanie przez scalanie:

Analizując dane dla sortowania przez scalanie można zauważyć również logarytmiczną złożoność obliczeniową n\*log(x) dla wszystkich przypadków wstępnego posortowania elementów oraz kiedy nie są one posortowane wstępnie. Na podstawie wykresu pod zestawieniem danych dotyczących sortowania przez scalanie oraz wykresu w punkcie pierwszym porównania 3 sortowań można stwierdzić, że wraz ze wzrostem posortowania wstępnie elementów, maleje czas posortowania tablicy w sposób zbliżony do liniowego. Czas posortowania elementów wcześniej odwrotnie posortowanych jest zbliżony do czasu posortowania tablicy wstępnie posortowanej w 99,7%.

Sortowanie introspektywne:

Na podstawie danych z testów można stwierdzić logarytmiczną złożoność obliczeniową. Analizując dane pod kontem zależności wstępnego posortowania elementów od czasu, można zauważyć że czas sortowania spada nie liniowo wraz ze wzrostem wstępnego posortowania. Oznacza to, że zwiększenie wstępnego posortowania z 25% do 50% powoduje większy spadek czasu sortowania, niż zwiększenie z 0% do 25%.

Porównanie czasu sortowania 2000000 elementów w zależności od wstępnego posortowania:

Na podstawie wykresy umieszczonego w punkcie 1 porównania 3 sortowań, można stwierdzić że nie zależnie od wstępnego posortowania, algorytm szybkiego sortowania potrzebuje najmniej czasu do posortowania 2000000 elementów. Można również zauważyć, że sortowanie introspektywne ma zbliżony czas do sortowania szybkiego dla wstępnego posortowania w 90% lub większym stopniu, a także przy wstępnym odwrotnym posortowaniu. Algorytm sortowania przez scalanie jest najwolniejszy.

Porównanie czasu sortowania dla różnej ilości elementów wcześniej nie posortowanych:

Porównując wyniki tego zestawienia można stwierdzić, że dla tego przypadku sortowanie introspektywne dla mniejszej ilości elementów jest nieco szybsze od sortowania szybkiego, natomiast dla większej ilości elementów sortowanie szybkie ma krótszy czas sortowania. Sortowanie przez scalanie dla małej ilości elementów ma również porównywalny czas z pozostałymi sortowaniami, jednak przy 200000 i więcej elementów, różnica w czasie staje się zauważalna.

Porównanie czasu sortowania dla różnej ilości elementów wcześniej posortowanych w 99,7%:

Na podstawie tego zestawienia można stwierdzić, że sortowanie introspektywne jest odrobine szybsze od sortowania szybkiego. Sortowanie przez scalanie, tak ja w sytuacji poprzedniej, przy mniejszej ilości elementów ma czas porównywalny z pozostałymi sortowaniami, natomiast przy większej ilości elementów jest znacznie wolniejsze.

Porównanie czasu sortowania dla różnej ilości elementów wcześniej odwrotnie posortowanych:

Czasy sortowania introspektywnego oraz szybkiego dla tego przypadku są prawie identyczne. Czas sortowania przez scalanie podobnie jak w pozostałych 2 zestawieniach, dla małej ilości elementów jest porównywalny z innymi, natomiast dla większej ilość sortuje wolniej od pozostałych.

Podsumowując:

Przeprowadzone testy wykazały dla wszystkich 3 algorytmów sortowania złożoność logarytmiczną złożoność obliczeniową. Otrzymane wyniki są zbliżone do oczekiwań. Wiadomo, że dla sortowania szybkiego istnieją przypadki, w których złożoność obliczeniowa wynosi n2, jednak w testach nie wystąpił taki przypadek. Wynika to najprawdopodobniej z wybrania elementu osiowego dla sortowania szybkiego w środku zbioru elementów do posortowania, a nie na początku czy też końcu co pozwoliło uniknąć trafienia wartości największej lub najmniejszej w zbiorze elementów do posortowania.

Zwiększenie wstępnego posortowania elementów zmniejszenie czasu sortowania dla wszystkich algorytmów, jednak nie w takim samym stopniu. Dla sortowania przez scalanie zmniejszenie czasu sortowania wraz ze wzrostem wstępnego posortowania jest zbliżony do liniowego, natomiast dla sortowania szybkie oraz introspektywnego zwiększenie wstępnego posortowania od 50% w górę, powoduje większą korzyść czasową, niż zwiększenie od 0% do 50%.

Czas sortowania dla mniejszej ilości elementów dla wszystkich trzech sortowań jest zbliżony, natomiast przy większej ilości elementów, sortowanie szybkie i introspektywne są znacząco szybsze. Do posortowania mniejszej ilości elementów wybrał by sortowanie przez scalanie, z uwagi na fakt że nie posiada ono przypadku, dla którego złożoność obliczeniowa rośnie do n2, natomiast ma podobny czas sortowania do sortowania szybkiego. Dwa pozostałe sortowania mają dużo lepszą wydajność czasową przy większej ilości elementów od sortowania przez scalanie i zastosowanie ich w takiej sytuacji wydaje się być lepsze.