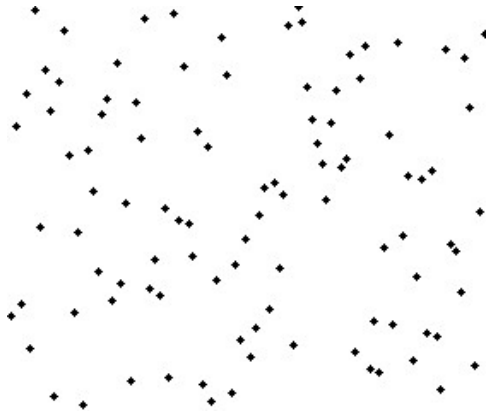
WIKIPEDIA

Sortowanie bąbelkowe

Sortowanie bąbelkowe (ang. *bubble sort*) – prosta metoda sortowania o złożoności czasowej $O(n^2)$ i pamięciowej $O(1)$.

Polega na porównywaniu dwóch kolejnych elementów i zamianie ich kolejności, jeżeli zaburza ona porządek, w jakim się sortuje tablicę. Sortowanie kończy się, gdy podczas kolejnego przejścia nie dokonano żadnej zmiany.

Sortowanie bąbelkowe



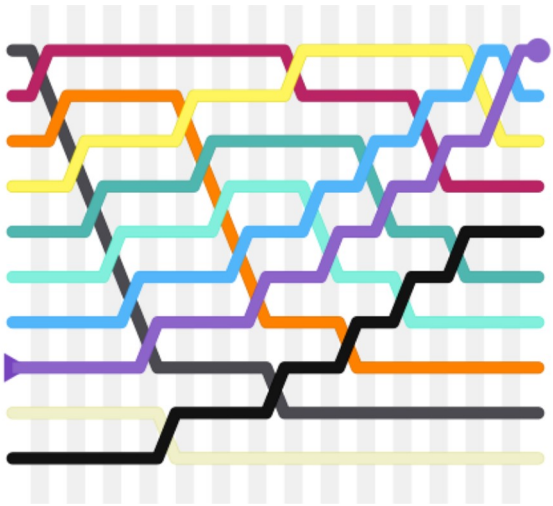
Przykład działania algorytmu sortowania bąbelkowego.

Rodzaj	Sortowanie
Struktura danych	Tablica, lista
Złożoność	
Czasowa	$O(n^2)$
Pamięciowa	$O(1)$

Spis treści

Dowód matematyczny

Algorytm opiera się na zasadzie maksimum, tj. każda liczba jest mniejsza lub równa od liczby maksymalnej. Porównując kolejno liczby można wyznaczyć największą z nich. Następnie ciąg częściowo posortowany (mający liczbę maksymalną), można skrócić o tę liczbę i ponowić szukanie maksimum, już bez elementów odrzuconych i tak długo, aż zostanie nam jeden element. Otrzymane kolejne maksima są coraz mniejsze przez co ciąg jest uporządkowany.



Wizualizacja sortowania bąbelkowego

Złożoność obliczeniowa

Algorytm wykonuje n-1 przejść, a w każdym przejściu wykonuje n-k porównań (gdzie k=1,2..n-1 to numer przejścia), przez co jego teoretyczna złożoność czasowa wynosi $O(n^2)$. W podstawowej wersji algorytmu nie można tego czasu skrócić, a każda permutacja powoduje, że algorytm jest wykonywany w czasie pesymistycznym.

Modyfikacje powodujące ulepszenie czasu

Algorytm można rozbudować tak, by czas optymistyczny był lepszy. Najłatwiejsze jest dodanie flagi informującej, czy w danej iteracji doszło do zmiany. Flaga jest zerowana na wejściu w przebiegu pętli, w przypadku natrafienia na zmianę jest podnoszona, a po wykonaniu przejścia sprawdzana. Jeśli nie było zmian, to sortowanie jest zakończone. Modyfikacja ta wprawdzie wydłuża czas wykonania jednego przejścia przez pętlę (gdyż trzeba wyzerować flagę, podnieść ją i sprawdzić), jednakże w wariancie optymistycznym (ciąg częściowo posortowany) może zaoszczędzić iteracji, przez co algorytm będzie działać szybciej.

6 5 3 1 8 7 2 4

Przykład działania

Przykład działania

Ciąg wejściowy $[4, 2, 5, 1, 7]$. Każdy wiersz symbolizuje wypchnięcie kolejnego największego elementu na koniec („wypłynięcie największego bąbelka”). Niebieskim kolorem oznaczono końcówkę ciągu już posortowanego.

$$[\underbrace{4,2}_{4>2},5,1,7] \rightarrow [2,\underbrace{4,5}_{4<5},1,7] \rightarrow [2,4,\underbrace{5,1}_{5>1},7] \rightarrow [2,4,1,\underbrace{5,7}_{5<7}]$$

$$[\underbrace{2,4}_{2<4},1,5,7] \rightarrow [2,\underbrace{4,1}_{4>1},5,7] \rightarrow [2,1,\underbrace{4,5}_{4<5},7]$$
$$[\underbrace{2,1}_{2>1},4,5,7] \rightarrow [1,\underbrace{2,4}_{2<4},5,7]$$
$$[\underbrace{1,2}_{1<2},4,5,7]$$

Pseudokod

Pseudokod wersji podstawowej algorytmu dla tablicy o rozmiarze n (elementy tablicy są numerowane od 0 do n-1):

```
procedure bubbleSort( A : lista elementów do posortowania )
  n = liczba_elementów(A)
   do
    for (i = 0; i < n-1; i++) do:
      if A[i] > A[i+1] then
        swap(A[i], A[i+1])
      end if
    end for
    n = n-1
  while n > 1
end procedure
```

Implementacja

- Zobacz przykłady implementacji tego algorytmu na stronie Wikibooks

Linki zewnętrzne

- Algorytm przedstawiony z wykorzystaniem węgierskiego tańca (https://www.youtube.com/watch?v=lyZQPjUT5B4)

Źródło: „https://pl.wikipedia.org/w/index.php?title=Sortowanie_bąbelkowe&oldid=57157545”