Sortarea Shell – sortarea prin inerţie cu micşorarea incrementului

În 1959, D.L. Shell a propus un algoritm de sortare bazat pe metoda prin inserţie directă, algoritm cu o performanţă îmbunătăţită deoarece face comparaţii între chei mai distanţate din vector. Algoritmul sortează mai întâi *prin inserţie* subvectori obţinuţi din vectorul iniţial prin extragerea componentelor aflate la o distanţă h una de cealaltă, distanţă care se numeşte *increment*. Repetând procedeul pentru incremenţi din ce în ce mai mici şi, în final, pentru incrementul 1, se obţine vectorul sortat.

Algoritmul

Se consideră un şir descrescător de numere naturale, numite incremenţi dintre care ultimul, h_t, este 1:

$$h_1 > h_2 > \ldots > h_i > h_{i+1} > \ldots > h_t = 1.$$

(1) Se porneşte cu incrementul h_1.

(2) La pasul iterativ m se consideră incrementul h_m. Se sortează prin inserţie directă subvectorii obţinuţi din vectorul iniţal luând elemente aflate la distanţa h_m, adică subvectorii:

- $A[1], A[h_m + 1], A[2h_m + 1], \ldots$
- $A[2], A[h_m + 2], \ldots$
- $\ldots$
- $A[h_m], A[2h_m], \ldots$

(3) Apoi se reia pasul (2) pentru incrementul h_{m+1}. Deoarece ultimul increment este $h_t = 1$, ultimul pas iterativ t se reduce la sortarea prin inserţie directă, deci vectorul va fi sortat.

Alegerea incremenţilor

Există multiple variante în literatură pentru generarea secvenţei de incremenţi. În plus, performanţa algoritmului depinde de această alegere.

1. Iniţial, **D.L. Shell** propune divizarea repetată a lungimii vectorului la 2. Notăm cu n lungimea vectorului. Se obţine secvenţa de incremenţi:

   - $1,\ 2,\ 4,\ 8,\ 16,\ 32,\ 64,\ \ldots,\ \left\lfloor \frac{n}{2} \right\rfloor$

   Avem $h_m = 2h_{m-1}$ şi $h_t = 1,\ t = [\log_2 n]$.

2. **D. E. Knuth** recomandă incremenţi obţinuţi cu formula recursivă:

   $h_m = 3h_{m-1} + 1$ şi $h_t = 1,\ t = [\log_3 n] - 1$

   Se obţine secvenţa:

   - $1,\ 4,\ 13,\ 40,\ 121,\ 364,\ 1093,\ 3280,\ 9841,\ \ldots,\ \frac{3^m-1}{2}.$

3. **Papernov-Stasevich** generează incremenţii astfel:

   $h_m = 2h_{m-1} + 1$ şi $h_t = 1,\ t = [\log_2 n] - 1.$

   Se obţine secvenţa:

   - $1,\ 3,\ 7,\ 15,\ 31,\ 63,\ 127,\ 255,\ 511,\ \ldots,\ 2^{m+1} - 1$

4. **Sedgewick** obţine o secvenţă de incremenţi

   - $1,\ 8,\ 23,\ 77,\ 281,\ 1073,\ 4193,\ 16577,\ \ldots,\ 4^k + 3 \cdot 2^{m-1} + 1$
   - sau $1,\ 5,\ 19,\ 41,\ 109,\ 209,\ 505,\ 929,\ \ldots$
     prin combinarea elementelor din următoarele două secvenţe:
     $1,\ 19,\ 109,\ 505,\ 2161,\ \ldots,\ 9\left(4^{m-1} - 2^{m-1}\right) + 1$
     $5,\ 41,\ 209,\ 929,\ 3905,\ \ldots 2^{m+1}\left(2^{m+1} - 3\right) + 1$

Complexitate

Există o estimare a complexităţii acestui algoritm care-l plasează în clasa $O\left(n^{\frac{5}{3}}\right)$, din punct de vedere al numărului de comparaţii. Din punct de vedere al spaţiului, am văzut că el necesită $h(1)$ locaţii suplimentare pentru componentele marcaj, cu ajutorul cărora eliminăm testele de nedepăşire a dimensiunii, teste care ar dubla numărul de comparaţii.

Referinţe

1. Wikipedia - the free encyclopedia – *Shellsort*, `http://en.wikipedia.org/wiki/Shellsort`. Accesat în noiembrie, 2012.
2. N. Wirth – *Algorithms and Data Structures.* 1985
3. R. Sedgewick – *Analysis of Shellsort and Related Algorithms.*