Data structure [B02]

김종규, PhD

2017-03-15

Bubble sort

- bubble sort

```
4   3 2 1
3 4   2 1
3 2 4   1
3 2 1 4
```

- insertion sort

```
4   3 2 1
3 4   2 1
  3 4 2 1
2 3 4   1
```

Sorting algorithm

- Bubble sort
  - 인접한 두 개를 비교하여 교환
- Insertion sort
  - 앞에서 차곡 차곡 정렬해 가는 것

Sorting algorithm

```
def test_sort(x)
  n = len(x)
  for j in range(n):
    for i in range(n-1):
      if x[i] > x[i+1]:
        (x[i+1],x[i])=(x[i],x[i+1])
```

$\longrightarrow$ Bubble sort

- 비교는 몇 번? $\longrightarrow n^2$

Cost model

- c_1: `n = len(x)`
- c_2: `for j in range(n):`
- c_3: `    for i in range(n-1):`
- c_4: `        if x[i] > x[i+1]:`
- c_5: `            (x[i+1],x[i])=(x[i],x[i+1])`
- Best: $c = c_1 + (n+1)c_2 + n^2 c_3 + n(n-1)c4 + n0c_5$
- Worst:

  $c = c_1 + (n+1)c_2 + n^2 c_3 + n(n-1)c4 + n(n-1)c_5$

Analysis of insertion sort: primitive cost model

INSERTION-SORT(A)	*cost*	*times*
for $j = 2$ **to** $A.length$	c_1	n
$\quad key = A[j]$	c_2	$n-1$
$\quad$ // Insert $A[j]$ into the sorted sequence $A[1..j-1]$.	0	$n-1$
$\quad i = j - 1$	c_4	$n-1$
$\quad$ **while** $i > 0$ and $A[i] > key$	c_5	$\sum_{j=2}^{n} t_j$
$\quad\quad A[i+1] = A[i]$	c_6	$\sum_{j=2}^{n} (t_j - 1)$
$\quad\quad i = i - 1$	c_7	$\sum_{j=2}^{n} (t_j - 1)$
$\quad A[i+1] = key$	c_8	$n-1$

그림: Insertion sort

Analysis of insertion sort: mathematical model

$$
\begin{aligned}
T(n) \;=\; & c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{j=2}^{n} t_j + c_6 \sum_{j=2}^{n} (t_j - 1) \\
& + c_7 \sum_{j=2}^{n} (t_j - 1) + c_8(n-1) \,.
\end{aligned}
$$

그림: Insertion sort; cost

Analysis of insertion sort: example (best case)

$$
\begin{aligned}
T(n) &= c_1 n + c_2(n-1) + c_4(n-1) + c_5(n-1) + c_8(n-1) \\
&= (c_1 + c_2 + c_4 + c_5 + c_8)n - (c_2 + c_4 + c_5 + c_8)\ .
\end{aligned}
$$

그림: Insertion sort; best cost

Insertion sort – Worst case

$$
\begin{aligned}
T(n) &= c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n(n+1)}{2} - 1\right) \\
&\quad + c_6\left(\frac{n(n-1)}{2}\right) + c_7\left(\frac{n(n-1)}{2}\right) + c_8(n-1) \\
&= \left(\frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2}\right) n^2 + \left(c_1 + c_2 + c_4 + \frac{c_5}{2} - \frac{c_6}{2} - \frac{c_7}{2} + c_8\right) n \\
&\quad - (c_2 + c_4 + c_5 + c_8)\ .
\end{aligned}
$$

그림: Insertion sort – Worst case

Big-O: $O(g(n))$

- $O(g(n))$: 여러 함수의 집합
  - 어떤 상수 c, n_0 를 지정하여
  - $0 \leq f(n) \leq c \cdot g(n), n > n_0$ 를 만족시키도록

$\longrightarrow$ $O(g(n)) = \{f(n) | 0 \leq f(n) \leq c \cdot g(n), n > n_0\}$

- 예
  - $f(n) = n$, $g(n) = 1,000,000,000 \times n$
  - $f(n), g(n) \in O(n)$

Big-O: $O(g(n))$

- 어떤 알고리즘이 $O(n^2)$ 라는 사실을 증명하였다. 다음 중 거짓인 것은?
  - 이 알고리즘은 적절한 c 에 대해서 $g(n) = cn^2$ 의 시간보다는 빨리 결과를 도출할 것이다 $\longrightarrow$ 참
  - 이 알고리즘이 $g(n) = n$ 의 시간이 걸리는 알고리즘보다 빨리 결과를 도출하는 경우는 절대 없다 $\longrightarrow$ 거짓
  - 이 알고리즘은 $O(n^3)$ 에도 속한다 $\longrightarrow$ 참
  - 이 알고리즘은 $O(2^n)$ 에도 속한다 $\longrightarrow$ 참

Big-O

- 다음을 증명하시오
  - $10n \in O(n) \longrightarrow$ 참
  - $f(n) = 10n, g(n) = n$
  - Let $c = 11, n_0 = 0$
  - $f(n) \leq cg(n)$ when $n > n_0$
  - $\therefore 10n \in O(n)$

Big-O

- 다음을 증명하시오
  - $n^2 + n \in O(n^2)$
  - $n^2 + n \leq n^2 + n^2$
  - $n^2 + n \leq 2n^2$
  - Let $c = 2, n_0 = 0$
  - $f(n) \leq cg(n)$ when $n > n_0$
  - $\therefore n^2 + n \in O(n)$

Cost model of bubble sort

- c_1: `n = len(x)`
- c_2: `for j in range(n):`
- c_3: `    for i in range(n-1):`
- c_4: `        if x[i] > x[i+1]:`
- c_5: `            (x[i+1],x[i])=(x[i],x[i+1])`
- Best:
  $$c_b(n) = c_1 + (n+1)c_2 + n^2 c_3 + n(n-1)c4 + n0c_5$$
- Worst:
  $$c_w(n) = c_1 + (n+1)c_2 + n^2 c_3 + n(n-1)c4 + n(n-1)c_5$$

$\longrightarrow$ 모두 Big-$O(n^2)$

Analysis of insertion sort: example (best case)

$$
\begin{aligned}
T(n) &= c_1 n + c_2(n-1) + c_4(n-1) + c_5(n-1) + c_8(n-1) \\
&= (c_1 + c_2 + c_4 + c_5 + c_8)n - (c_2 + c_4 + c_5 + c_8) .
\end{aligned}
$$

그림: Insertion sort; best cost

Insertion sort – Worst case

$$
\begin{aligned}
T(n) &= c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n(n+1)}{2} - 1\right) \\
&\quad + c_6\left(\frac{n(n-1)}{2}\right) + c_7\left(\frac{n(n-1)}{2}\right) + c_8(n-1) \\
&= \left(\frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2}\right) n^2 + \left(c_1 + c_2 + c_4 + \frac{c_5}{2} - \frac{c_6}{2} - \frac{c_7}{2} + c_8\right) n \\
&\quad - (c_2 + c_4 + c_5 + c_8)\ .
\end{aligned}
$$

그림: Insertion sort – Worst case

Cost model of insertion sort

- Best case: Big-$O(n)$
- Worst case: Big-$O(n^2)$
- 두 경우의 차이
  - $n = 1000 \longrightarrow$ 천 배 차이가 날 수 있다
  - $n = 1,000,000 \longrightarrow$ 백만 배 차이가 날 수 있다

$\longrightarrow$ 따라서 최악의 경우에 대비해야 한다

Worst-case complexity

- 입력이 어떻게 들어올 지는 아무도 모른다
- 설계자는 최악의 경우에 어느정도가 보장될 지에 관심을 갖는다
- 따라서 알고리즘이 얼마나 효율적인가는 최악의 경우를 상정하여 계산한다
- 이를 알고리즘의 복잡도 (Complexity) 라고 한다

Wrap-up

- The complexity of an algorithm is analyzed based on worst-case
- In this sense, insertion sort and bubble sort are the same complexity
- We will learn better algorithms after learning the concept of abstract data type