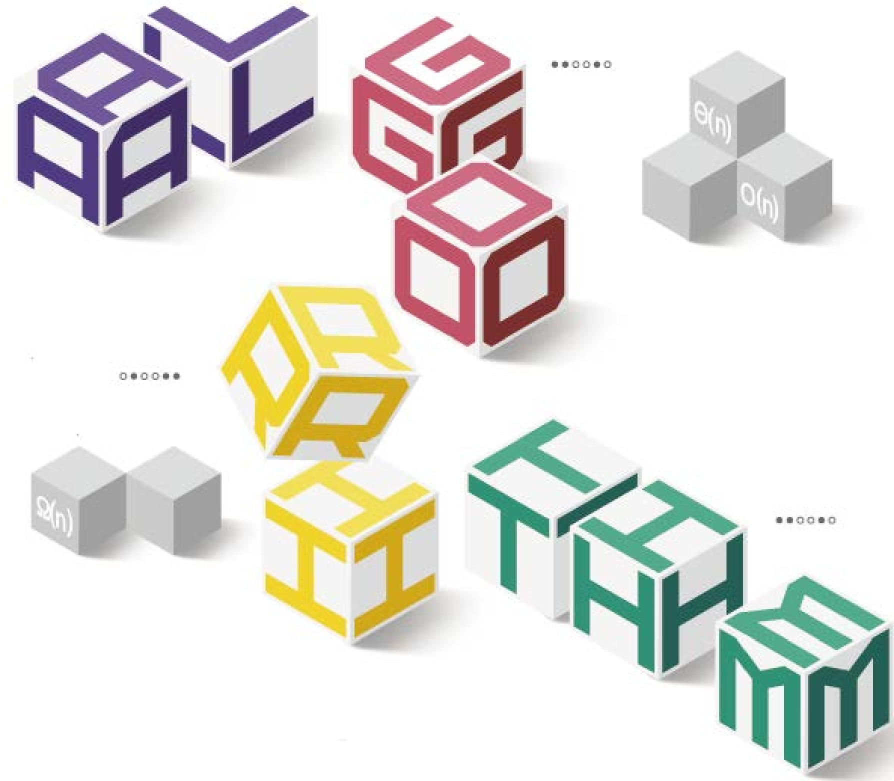


관계 중심의 사고법

쉽게 배우는 알고리즘

3장. 점화식과 점근적 복잡도 분석

3장.
점화식과 알고리즘 복잡도 분석

사실을 많이 아는 것보다는
이론적 틀이 중요하고,
기억력보다는
생각하는 법이 더 중요하다.

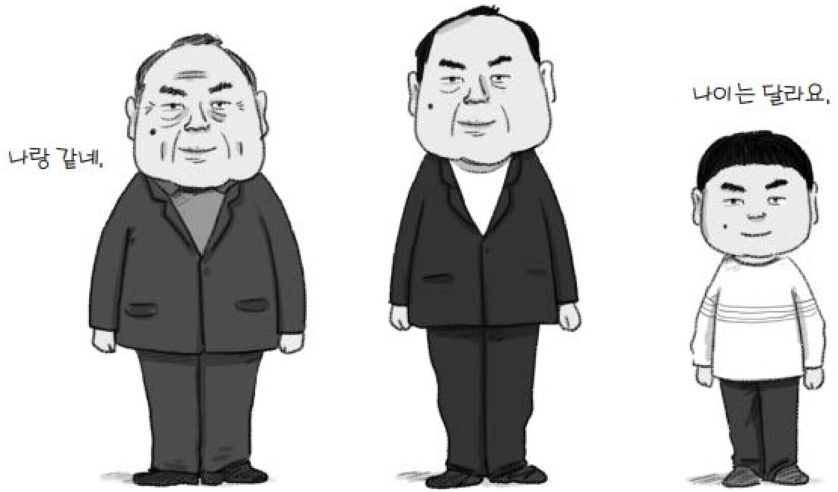
–제임스 왓슨

학습목표

- 재귀 알고리즘과 점화식의 관계를 이해한다.
- 점화식의 점근적 분석을 이해한다.

점화식의 이해

- 점화식
  - 어떤 함수를 자신보다 더 작은 변수에 대한 함수와의 관계로 표현한 것
- 예
  - $a_n = a_{n\text{-}1} + 2$
  - $f(n) = n\,f(n-1)$
  - $f(n) = f(n-1) + f(n-2)$
  - $f(n) = f(n/2) + n$



병합 정렬의 수행 시간

```
mergeSort(A[ ], p, r)     ▷ A[p ... r]을 정렬한다.
{
    if (p < r) then {
        q ← ⌊(p + q)/2⌋;  ----------------- ①  ▷ p, q의 중간 지점 계산
        mergeSort(A, p, q);  ---------------- ②  ▷ 전반부 정렬
        mergeSort(A, q+1, r); -------------- ③  ▷ 후반부 정렬
        merge(A, p, q, r);  ------------------ ④  ▷ 병합
    }
}
merge(A[ ], p, q, r)
{
    정렬되어 있는 두 배열 A[p ... q]와 A[q+1 ... r]을 합하여
    정렬된 하나의 배열 A[p ... r]을 만든다.
}
```

수행 시간의 점화식: $T(n) = 2T(n/2)$ + 오버헤드

✓크기가 n인 병합 정렬 시간은 크기가 $n/2$인 병합 정렬을 두 번하는 시간과 나머지 오버헤드를 더한 시간이다

점화식의 점근적 분석 방법

- 반복 대치
  - 더 작은 문제에 대한 함수로 반복해서 대치해 나가는 해법
- 추정후 증명
  - 결론을 추정하고 수학적 귀납법으로 이용하여 증명하는 방법
- 마스터 정리
  - 형식에 맞는 점화식의 복잡도를 바로 알 수 있다

반복 대치

$$T(n) = T(n-1) + c$$
$$T(1) \le c$$

$$\begin{aligned}
T(n) \quad &= T(n-1) + c \\
&= (T(n-2) + c) + c = T(n-2) + 2c \\
&= (T(n-3) + c) + 2c = T(n-3) + 3c \\
&\dots \\
&= T(1) + (n-1)c \\
&\le c + (n-1)c \\
&= cn
\end{aligned}$$

반복 대치

$$T(n) = 2T(n/2) + n$$
$$T(1) = 1$$

$$\begin{aligned}
T(n) &= 2T(n/2) + n \\
&= 2(2T(n/2^2) + n/2) + n = 2^2T(n/2^2) + 2n \\
&= 2^2(2T(n/2^3) + n/2^2) + 2n = 2^3T(n/2^3) + 3n \\
&\cdots \\
&= 2^kT(n/2^k) + kn \\
&= n + n\log n \\
&= \Theta(n\log n)
\end{aligned}$$

추정후 증명

$T(n) = 2T(n/2) + n, \quad T(1) = 1.$

추정: $T(n) = O(n\log n)$, 즉 $T(n) \le cn\log n$

<증명> (수학적 귀납법) $n' < n$에 대해 성립한다고 가정.

$$
\begin{aligned}
T(n) &= 2T(n/2) + n \\
&\le 2c(n/2)\log(n/2) + n \\
&= cn\log n - cn\log 2 + n \\
&= cn\log n + (-c\log 2 + 1)n \\
&\le cn\log n \quad \longleftarrow \text{ 이를 만족하는 } c\text{가 존재한다}
\end{aligned}
$$

추정후 증명

$$T(n) = 2T(n/2) + 1, \quad T(1) = 1$$

추정: $T(n) = O(n)$, 즉 $T(n) \le cn$

<증명> $n' < n$에 대해 성립한다고 가정.

$$\begin{aligned} T(n) &= 2T(n/2) + 1 \\ &\le 2c(n/2) + 1 \quad \longleftarrow \text{귀납적 가정 이용} \\ &= cn + 1 \end{aligned}$$

더 이상 진행 불가!

추정후 증명

앞의 $T(n) = 2T(n/2) + 1, \quad T(1) = 1$

추정: $T(n) \le cn-1$

<증명> $n' < n$에 대해 성립한다고 가정.

$$
\begin{aligned}
T(n) &= 2T(n/2) + 1 \\
&\le 2(c(n/2) - 1) + 1 \quad \longleftarrow \text{귀납적 가정 이용} \\
&= cn - 1
\end{aligned}
$$

마스터 정리

- $T(n) = aT(n/b) + f(n)$와 같은 모양을 가진 점화식은 마스터 정리에 의해 바로 결과를 알 수 있다
- $n^{\log_b a} = h(n)$이라 하자

① 어떤 양의 상수 ε에 대하여 $f(n) = O(h(n)/n^{\varepsilon})$이면, $T(n) = \Theta(h(n))$이다.

② $f(n) = \Theta(h(n))$이면 $T(n) = \Theta(h(n)\log n)$이다.

③ 어떤 양의 상수 ε에 대하여 $f(n) = \Omega(h(n)n^{\varepsilon})$이고, 어떤 상수 $c(< 1)$와 충분히 큰 모든 n에 대해 $af(n/b) \leq cf(n)$이면 $T(n) = \Theta(f(n))$이다.

마스터 정리의 직관적 의미

① $\mathrm{h}(n)$이 더 무거우면 $\mathrm{h}(n)$이 수행 시간을 결정한다.

② $\mathrm{h}(n)$과 $\mathrm{f}(n)$이 같은 무게이면 $\mathrm{h}(n)$에 $\log n$을 곱한 것이 수행 시간이 된다.

③ $\mathrm{f}(n)$이 더 무거우면 $\mathrm{f}(n)$이 수행 시간을 결정한다.

✓ 원 정리와 약간 차이가 있기는 하지만
직관적 이해를 위해서 도움이 된다.

마스터 정리의 적용 예

- $T(n) = 2T(n/3) + c$
  - $a=2, b=3, h(n) = n^{\log_3 2}, f(n) = c$
  - $T(n) = \Theta(n^{\log_3 2})$
- $T(n) = 2T(n/2) + n$
  - $a=b=2, h(n) = n^{\log_2 2} = n, f(n) = n$
  - $T(n) = \Theta(n \log n)$
- $T(n) = 2T(n/4) + n$
  - $a=2, b=4, h(n) = n^{\log_4 2}, f(n) = n$
  - $af(n/b) = n/2 \le cf(n)$ for $c=1/2$
  - $T(n) = \Theta(n)$

Thank you