0.1 Lecture 4 - Master theorem

0.1.1 intro

$T(n) = aT(\frac{n}{b}) + f(n)$

1. $f(n) = n^{c+\epsilon}$ then $T(n) = \Theta(n^c)$
2. $f(n) = n^c$ then $T(n) = O(n^c)$
3. $f(n) = n^{c-\epsilon}$, then $T(n) = \Omega(f(n)))$

0.1.2 inequality

$T(n) = aT(\frac{n}{b}) + f(n)$

1. $f(n) = O(n^{c+\epsilon})$ then $T(n) = O(n^c)$
2. $f(n) = O(n^c)$ then $T(n) = O(n^c)$
3. $f(n) = O(n^{c-\epsilon})$, then $T(n) = O(f(n))$

> **Remark 0.1 —** when $f(n) = \Theta(n)$, we will split the cases by checking c > 1, c $= 1$ and c < 1

Proof when $f(n) = O(n)$

1.

$$\begin{aligned}
T(n) &\leq aT(\frac{n}{b}) + f(n) \\
&= a \cdot (T(\frac{n}{b^2}) + \cdot(\frac{n}{b})) + f(n) \\
&= \dots \\
&= a^i T(1) + a^{i-1}(\frac{n}{b^i}) + \dots + kn \\
&= n^c + kn \sum_{i=0}^{\log_b(n)-1} (\frac{a}{b})^i && \texttt{case 1:}\ c > 1 \\
&= n^c + kn \cdot \frac{(\frac{a}{b})^{\log_b n} - 1}{\frac{a}{b} - 1} \\
&\leq n^c + kn \cdot \frac{(\frac{a}{b})^{\log_b n}}{\frac{a}{b} - 1} \\
&= n^c + k \cdot \frac{n^{\log_b a}}{\frac{a}{b} - 1} \\
&= O(n^c)
\end{aligned}$$

.

2.

$$
\begin{aligned}
T(n) &\leq aT(\frac{n}{b}) + f(n) \\
&= n + kn \sum_{i=0}^{\log_b(n)-1} (\frac{a}{b})^i && \texttt{case 2: } c = 1 \\
&= n + kn \cdot (\log_b n - 1) \\
&= O(n \log n)
\end{aligned}
$$

.

3.

$$
\begin{aligned}
T(n) &\leq aT(\frac{n}{b}) + f(n) \\
&= n^c + kn \sum_{i=0}^{\log_b(n)-1} (\frac{a}{b})^i && \texttt{case 3: } c < 1 \\
&= n^c + kn \cdot O(1) && \texttt{decreasing geometric series} \\
&= O(n)
\end{aligned}
$$

.

0.2 Lecture 5 - Randomized algorithms

Exercise 0.2

$$
S = \sum_{i=1}^{\infty} i(1-p)^{i-1} = \sum_{j=0}^{\infty} (j+1)(k)^j
$$

$$
kS = \sum_{j=1}^{\infty} (j+1)(k)^j \tag{1}
$$

$$
S - kS = 1 + k + k^2 + k^3 + \ldots = \frac{1}{1-k} \tag{2}
$$

$$
S = \frac{1}{(1-k)^2}
$$

.

Exercise 0.3 (Coupon collector)

$$
\begin{aligned}
P(i) &= \frac{n-i}{n} \\
E[x_i] &= \frac{1}{P(i)} \\
E[X] &= \sum_{i=1}^{n-1} E[x_i] \\
&= n \sum_{i=1}^{n} \frac{1}{i} \\
&= \Theta(n \log n)
\end{aligned}
$$

.

1. hiring problem
2. birthday paradox
3. generate random permutation (with proof)

0.3 Lecture 6 - Quick Sort

0.3.1 Intro to algo

Algorithm 1 Quicksort

Input: A, p, q, r
Output: 1

if $p \geq r$ **then**
    `return`
end if
$q =$ PARTITION(A, p, r)
QUICKSORT$(A, p, q-1)$
QUICKSORT$(A, q+1, r)$
return *state*

Algorithm 2 Partition

Input: A, p, r
Output: q

```
i ← p − 1
for j ← p to r − 1 do
    if A[j] ≤ x then
        i ← i + 1
        SWAP(A[i], A[j])
    end if
end for
return i
```

0.3.2 Run time analysis

1. best case: $O(\log n)$
2. worse case: $O(n^2)$

Average run-time of quicksort. To attain average case, randomly select the pivot. Let $p(x_{i,j})$ be the probability that z_i and z_j are compared

$$
\begin{aligned}
p(x_{i,j}) &= \frac{1}{j-i+1} \\
E[X] &= \sum_{i=1}^{n} 2 \cdot p(x_{i,j}) && z_i \texttt{ and } z_j \texttt{ are symmetric} \\
&= O(n \log n)
\end{aligned}
$$

.

Exercise 0.4 (k-smallest element in union of two sorted lists)