Chapter 1

Analysis of algorithms

The analysis of algorithms is to determine the amount of resource (time/space) necessary to execute algorithms. By analyzing the resources used in algorithms, we can compare different algorithms theoretically.

The amount of resource used in an algorithm is usually represented by a function $T(n)$, where n is the length of the input. The goal of the analysis of algorithms is to find out the asymptotic growth rate of $T(n)$ in terms of n. In computer science, since n denotes the length of the input, we only care about positive values of n. Similarly, since $T(n)$ denotes the amount of resource, we usually assume that $T(n)$ is positive. This is not a real limitation, but can simplify the discussion.

Each textbook usually discusses the analysis of algorithms in the first several chapters. For those who want to know further about the analysis of algorithms, you can watch the videos of Analysis of Algorithms on Coursera and read books [3, 5, 6].

1.1 Recurrence relations

During the exam, we are often given a recurrence relation $T(n) = a(n)T(b(n)) + f(n)$, and we need to give to a tight bound of $T(n)$. If the problem statements do not explicitly specify the base care, we usually assume that $T(n) = \Theta(1)$ when n is small (less than some constant). In general, we can just focus on some particular values of n as long as these values approach infinity[1]. For example, we can only consider $n = 2^i$ for all positive integer i. Making assumptions can make the analysis easier.

1.1.1 Master theorem

The most powerful technique in solving divide-and-conquer recurrence relation is the master theorem. There are several forms of the master theorem. Verma gives the following master theorem [7]:

Theorem 1. *Let $T(n) = aT(n/b) + f(n)$ for all $n > 1$ and $T(1) = c$ for some constants $a \geq 1$, $b > 1$, $c > 0$, and non-negative function $f(n)$.*

1. *if $f(n) = O(n^{\lg_b a}/\lg n)(1+\epsilon)$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\lg_b a})$.*
2. *if $f(n) = \Theta(n^{\lg_b a} \lg^k n)$ for some $k \geq 0$, then $T(n) = \Theta(n^{\lg_b a} \lg^{k+1} n)$.*
3. *if $f(n) = \Omega(n^{\epsilon+\lg_b a})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq kf(n)$ for some constant $k < 1$ and all sufficiently large n, then $T(n) = \Omega(f(n))$.*
4. *if $f(n) = \Theta(n^{\lg_b a}/\lg n)$, then $T(n) = \Theta(f(n) \lg n \lg\lg n)$.*

[1] There should be a condition specifying what assumptions are applicable.

Note When you apply the master theorem, please pay attention to the following:

1. When the recursion involves floor or ceiling function, the master theorem does not apply. For example, $T(n) = T(\lfloor \frac{n}{2} \rfloor) + T(\lceil \frac{n}{2} \rceil) + n$.
2. In order to apply the case 2 (the extended master theorem), k must be non-negative. For example, case 2 does not apply in the case of $T(n) = 2T(\frac{n}{2}) + n/\lg n$.
3. In order to apply the case 3, the regularity condition must be satisfied. For example, case 3 does not apply in the case of $T(n) = T(\frac{n}{2}) + n(2 - \cos n)$.

Exercise 1

Let $T(n) = \Theta(f(n))$. Assume that $T(n)$ is a constant for sufficiently small n. Derive $f(n)$ in the simplest formula for each of the following $T(n)$.

1. $T(n) = 2T(\frac{n}{2}) + \frac{n}{\lg^2 n}$. [NCTU CS 104]

Answer of exercise 1

Problem 1: By case 1, we know $T(n) = \Theta(n)$.
Problem 1 alternative solution: Suppose that $n = 2^k$. We have

$$\begin{aligned} & T(n) = 2T(\frac{n}{2}) + \frac{n}{\lg^2 n} \\ \equiv \quad & T(n) = 4T(\frac{n}{4}) + \frac{n}{\lg^2 n} + \frac{n}{(\lg(n-1))^2} \\ \equiv \quad & T(n) = 2^k T(1) + n \sum_{i=}^{k} i^{-2} \\ \equiv \quad & T(n) = n + n \sum_{i=}^{k} i^{-2} \end{aligned}$$

Since $\sum_{i=}^{k} i^{-2}$ is lower bounded by one and is upper bounded by $\sum_{i=}^{\infty} i^{-2} = \zeta(2) = \frac{\pi^2}{6}$[2], we have $T(n) = \Theta(n)$.

1.1.2 Akra-Bazzi method

The Akra and Bazzi method provides a more general way to solve divide-and-conquer recurrence relations [1]. The following version is from [4].

Theorem 2. *Let*

$$T(n) = \begin{cases} \Theta(1) & 1 \le n \le n_0 \\ \sum_{i=1}^{k} a_i T(b_i n + h_i(n)) + f(n) & \forall n > n_0 \end{cases}$$

where

1. $k \ge 1$ *is a constant and for all* i, $a_i > 0$ *and* $b_i \in (0, 1)$ *are constants.*
2. $|f(n)|$ *is polynomially-bounded.*
3. *for all* i, $|h_i(n)| = O(x/\lg^2 x)$.
4. n_0 *is large enough.*

Let p *be the unique solution for* $\sum_{i=1}^{k} a_i b_i^p = 1$. *Then*

[2] This is the solution for the Basel problem.

1. *if* $f(n) = O(n^{p-\epsilon})$ *for some constant* $\epsilon > 0$, *then* $T(n) = \Theta(n^p)$.
2. *if* $f(n) = \Theta(n^p)$ *for some constant* $\epsilon > 0$, *then* $T(n) = \Theta(n^p \lg n)$.
3. *if* $f(n) = \Omega(n^{p+\epsilon})$ *and* $f(n)/x^{p+\epsilon}$ *is non-decreasing for some constant* $\epsilon > 0$, *then* $T(n) = \Theta(f(n))$.
4. $T(n) = \Theta\left(n^p\left(1 + \int_1^n \frac{f(u)}{u^{p+1}}du\right)\right)$.

Note This version of the Akra-Bazzi method can deal with floor and ceil function by picking h_i.

General version Some of the requirements can be relaxed [4].

1. The second condition is called *polynomial-growth condition* and can be replaced by the following weaker requirement: $g(n)$ is nonnegative and exist constants c_1 and c_2 such that for all i and for all $u \in [b_i n + h_i(n), n]$, we have $c_1 f(n) \leq f(u) \leq c_2 f(n)$.
2. The third condition can be replaced by the following weaker requirement: there exists a constant $\epsilon > 0$, $|h_i(n)| \leq n/(\lg^{1+\epsilon} n)$ for all i and $n \geq n_0$.
3. The fourth condition is pretty technical and you can find the complete version in the original paper.

Drmota and Szpankowski prove a more general theorem that can deal with floor and ceil functions directly [2].

Exercise 2

Let $T(n) = \Theta(f(n))$. Assume that $T(n)$ is a constant for sufficiently small n. Derive $f(n)$ in the simplest formula for each of the following $T(n)$.

1. $T(n) = 5T(\frac{n}{5}) + n/\lg n$. [NCTU CSIE 93]
2. $T(n) = T(\frac{n}{2} + \sqrt{n}) + n$. [NTU CSIE 103]
3. $T(n) = 4T(\frac{n}{5}) + T(\frac{n}{4}) + n$. [NCTU BIOINFO 93]

Answer of exercise 2

Problem 1: Suppose that $n = 5^k$. We have

$$\begin{aligned} & T(n) = 5(5T(\frac{n}{25}) + n/(5(\lg_5 n - \lg_5 5))) + n/\lg n \\ \equiv \quad & T(n) = 5^k T(1) + n/(\lg n + \lg(n-1) + \cdots + 1) \\ \equiv \quad & T(n) = \Theta(n \lg\lg n) \end{aligned}$$

Problem 1 another solution: apply the Akra-Bazzi method. Set $k = 1$, $a_1 = 5$, $b_1 = 1/5$, and solve $p = 1$. We get

$$T(n) = \Theta(n(1 + \int_1^n (x/\lg x)x^{-2}dx)) = \Theta(n \lg\lg n).$$

Note: Similarly, for any constant c, we have $T(n) = cT(\frac{n}{c}) + n/\lg n = \Theta(n \lg\lg n)$.

Problem 2: apply the Akra-Bazzi method. Set $k = 1$, $a_1 = 1$, $b_1 = 1/2$, and solve $p = 0$. Hence, we get

$$T(n) = \Theta(1(1 + \int_1^n (x/x)dx)) = \Theta(n).$$

Problem 3: apply the Akra-Bazzi method. Set $k = 2$, $a_1 = 4$, $b_1 = 1/5$, $a_2 = 1$, $b_2 = 1/4$, and solve $p \approx 1.03$. We get

$$T(n) = \Theta(n^p), \text{where} p \approx 1.03.$$

Exercise 3

Given positive constants c', $c_1, c_2, \ldots, c_k$, assume that $T(n) \leq c'n + \sum_{i=1}^{k} T(c_i n)$ and $\sum_{i=1}^{k} c_i < 1$. Prove $T(n) = O(n)$. [NCTU CSIE 92]

Answer of exercise 3

Apply the Akra-Bazzi method.

1.1.3 Full-history recurrence

Exercise 4

Let $T(n) = \Theta(f(n))$. Assume that $T(n)$ is a constant for sufficiently small n. Derive $f(n)$ in the simplest formula for each of the following $T(n)$.

1. $T(n) = n + \frac{4}{n} \sum_{i=1}^{n-1} T(i)$. [NTU CSIE 90]

Answer of exercise 4

Problem 1:

$$
\begin{aligned}
& T(n) = n + \frac{4}{n} \sum_{i=1}^{n-1} T(i) \\
\equiv \quad & nT(n) = n^2 + 4 \sum_{i=1}^{n-1} T(i) \\
\equiv \quad & (n+1)T(n+1) = n^2 + 4 \sum_{i=1}^{n} T(i) \\
\equiv \quad & (n+1)T(n+1) - nT(n) = 2n + 1 + 4T(n) \\
\equiv \quad & (n+1)T(n+1) = (n+4)T(n) + 2n + 1 \\
\equiv \quad & \frac{T(n+1)}{(n+2)(n+3)(n+4)} = \frac{T(n)}{(n+1)(n+2)(n+3)} + \frac{2n+1}{(n+1)(n+2)(n+3)(n+4)}
\end{aligned}
$$

Let $S(n) = \frac{T(n)}{((n+1)(n+2)(n+3))}$.

$$
\begin{aligned}
& S(n+1) = S(n) + \frac{2n+1}{(n+1)(n+2)(n+3)(n+4)} \\
\equiv \quad & S(n) = \sum_{i=0}^{n-1} \frac{2i+1}{(i+1)(i+2)(i+3)(i+4)} \\
\equiv \quad & S(n) = \Theta(1) \\
\equiv \quad & T(n) = (n+1)(n+2)(n+3)S(n) \\
\equiv \quad & T(n) = \Theta(n^3)
\end{aligned}
$$

1.1.4 Range transformation

Exercise 5

Let $T(n) = \Theta(f(n))$. Assume that $T(n)$ is a constant for sufficiently small n. Derive $f(n)$ in the simplest formula for each of the following $T(n)$.

1. $T(n) = \sqrt{n}T(\sqrt{n}) + \sqrt{n}$. [NCTU CSIE 93]

Answer of exercise 5

Problem 1: The trick is to divide n by both side and then expand. $\frac{T(n)}{n} = \frac{T(\sqrt{n})}{\sqrt{n}} + n^{-0.5}$. Suppose that $n = 2^{2^k}$. Let $S(k) = T(2^{2^k})/2^{2^k}$.

$$\begin{aligned} & T(n) = S(k) = S(k-1) + 2^{-2^{k-1}} \\ \equiv \quad & S(k) = \Theta(1) + \sum_{i=1}^{k-1} 2^{-2^{k-1}} = \Theta(1) \\ \equiv \quad & T(n) = n \cdot S(k) = n \cdot \Theta(1) = \Theta(n) \end{aligned}$$

1.1.5 Recursion trees

Exercise 6

Let $T(n) = \Theta(f(n))$. Assume that $T(n)$ is a constant for sufficiently small n. Derive $f(n)$ in the simplest formula for each of the following $T(n)$.

1. $T(n) = 2T(\sqrt{n}) + \lg n$. [NCKU CSIE 95]
2. $T(n) = nT(\sqrt{n}) + n^2 \lg n$. [NTU CSIE 98]

Answer of exercise 6

Problem 1: Suppose that $n = 2^{2^k}$. We have

$$\begin{aligned} & T(n) = 2^2 T(\sqrt[4]{n}) + 2(\lg n - 1) + \lg n \\ \equiv \quad & T(n) = 2^k T(1) + \lg n + 2(\lg n)/2 + 4(\lg n)/4 + ... + 2^k(\lg n)/2^k \\ \equiv \quad & T(n) = 2^k T(1) + \lg n \sum_{i=1}^{k} 1 \\ \equiv \quad & T(n) = \Theta(\lg n \lg\lg n) \end{aligned}$$

Problem 2: Suppose that $n = 2^{2^k}$. We have

$$\begin{aligned} & T(n) = n^{1.5} T(\sqrt[4]{n}) + \frac{n^2 \lg n}{2} + n^2 \lg n \\ \equiv \quad & T(n) = n^2 T(1) + n^2 \lg n \sum_{i=0}^{k} 2^{-i} \\ \equiv \quad & T(n) = \Theta(n^2 \lg n) \end{aligned}$$

1.1.6 Comparison

Exercise 7

Let $T(n) = \Theta(f(n))$. Assume that $T(n)$ is a constant for sufficiently small n. Derive $f(n)$ in the simplest formula for each of the following $T(n)$.

1. $T(n) = T(\frac{n}{2}) + T(\sqrt{n}) + n$. [NCTU BIOINFO 93]

Answer of exercise 7

Problem 1: By the recurrence relation, we have $T(n) = \Omega(n)$. Let $F(n) = F(n/2) + F(n/3) + n$. Since $n/3 > \sqrt{n}$ for all $n > 9$, we have $F(n) \geq T(n)$. By using the Akra-Bazzi method, we get $F(n) = \Theta(n)$. Hence, we have $T(n) = \Theta(n)$.

References

[1] Mohamad Akra and Louay Bazzi. "On the Solution of Linear Recurrence Equations". In: *Computational Optimization and Applications* 10.2 (May 1998), pp. 195–210. ISSN: 0926-6003. DOI: 10.1023/A:1018373005182. URL: http://dx.doi.org/10.1023/A:1018373005182 (cit. on p. 2).

[2] Michael Drmota and Wojciech Szpankowski. "A Master Theorem for Discrete Divide and Conquer Recurrences". In: *Journal of the ACM* 60.3 (June 2013), p. 16. ISSN: 0004-5411. DOI: 10.1145/2487241.2487242. URL: http://doi.acm.org/10.1145/2487241.2487242 (cit. on p. 3).

[3] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete mathematics - a foundation for computer science*. 2nd ed. Addison-Wesley, 1994. ISBN: 978-0-201-55802-9 (cit. on p. 1).

[4] Tom Leighton. *Notes on Better Master Theorems for Divide-and-Conquer Recurrences*. Tech. rep. Department of Mathematics, Massachusetts Institute of Technology, Oct. 1996 (cit. on pp. 2, 3).

[5] Paul Walton Purdom Jr. and Cynthia Brown. *The Analysis of Algorithms*. Oxford University Press, 2004. ISBN: 978-0-195-17479-3 (cit. on p. 1).

[6] Robert Sedgewick and Philippe Flajolet. *An introduction to the analysis of algorithms*. 2nd ed. Addison-Wesley, 2013. ISBN: 978-0-321-90575-8 (cit. on p. 1).

[7] Rakesh M. Verma. "A General Method and a Master Theorem for Divide-and-Conquer Recurrences with Applications". In: *Journal of Algorithms* 16.1 (Jan. 1994), pp. 67–79. ISSN: 0196-6774. DOI: 10.1006/jagm.1994.1004. URL: http://dx.doi.org/10.1006/jagm.1994.1004 (cit. on p. 1).