A3

1 Solving Recurence

1.1 The Substitution Method

(From CLRS 4.3-1)

Use the substitution method to show that each of the following recurrences defined on the reals has the asymptotic solution specified:

1. $T(n) = 2T(n/2 + 17) + n$ has solution $T(n) = O(n \lg n)$.
2. $T(n) = 2T(n/3) + \Theta(n)$ has solution $T(n) = \Theta(n)$.

Note: when proving the solution, it is helpful to read the *Avoiding pitfalls* section of CLRS 4.3.

1.1 Solution

1.1.1 $T(n) = 2T(n/2 + 17) + n$

方法一

$\because T(n) = 2T(n/2 + 17) + n$

$\forall n > 34, Let's\ guess:$
$\exists b, c, d, \quad s.t. T(n) \leq n \lg{(n-34)} d - nc - b \lg{(n-34)} = O(n \lg n)$

$\therefore T(n/2 + 17) \leq (\frac{n}{2} + 17) \lg{(\frac{n}{2} - 17)} d - (\frac{n}{2} + 17) c - b \lg{(\frac{n}{2} - 17)}$

$\therefore T(n) \leq 2(\frac{n}{2} + 17) \lg(\frac{n}{2} - 17) d - n(c-1) - 2b \lg{(\frac{n}{2} - 17)} - 34c$

$= n \lg{(\frac{n}{2} - 17)} d - n(c-1) - (2b - 34d) \lg{(\frac{n}{2} - 17)} - 34c$

$= n(\lg{(n-34)} - \lg 2) d - n(c-1) - (2b - 34d)(\lg{(n-34)} - \lg 2) - 34c$

$= n(\lg{(n-34)} - 1) d - n(c-1) - (2b - 34d)(\lg{(n-34)} - 1) - 34c$

$= n \lg{(n-34)} d - n(d + c - 1) - (2b - 34d) \lg{(n-34)} - 34c + 2b - 34d$

$= n\lg{(n-34)}d - nc - b\lg{(n-34)}$
$\quad -n(d-1) - (b-34d)\lg{(n-34)} + 2b - 34(c-d)$

只要满足 $d \geq 1$ 且 $b \geq 34d$ 且 $2b \leq 34(c-d)$
则有： $T(n) \leq n\lg{(n-34)}d - nc - b\lg{(n-34)}$
符合 $guess$

取$d=1, b=34, c=2$得：
$\forall n \geq 35,\ T(n) \leq n\lg{(n-34)} - 2n - 34\lg{(n-34)}$
$\therefore T(n) = O(n\lg{n})$

方法二

$guess:\ T(n) \leq c(n-b)\lg{(n-b)}$

$$
\begin{aligned}
T(n) \leq & 2c(n/2-b+17)\lg{(n/2-b+17)}+n \\
= & c(n+34-2b)\lg{n+34-2b} - c(n+34-2b)+n \\
\xlongequal{b\geq 34} & c(n-b)\lg{(n-b)} - c(n-b)+n \\
= & c(n-b)\lg{(n-b)} - c((1-\frac{1}{c})n-b) \\
\xlongequal{\text{当}c\geq 2} & \leq c(n-b)\lg(n-b)
\end{aligned}
$$

1.1.2 $T(n) = 2T(n/3) + \Theta(n)$

$\because T(n) = 2T(n/3) + \Theta(n)$

$Let's\ guess:$
$T(n) = \Theta(n)$ 即 $\exists d_1, d_2, \forall n \geq n_0$ 有 $nd_1 \leq T(n) \leq nd_2;$

$\therefore \frac{nd_1}{3} \leq T(n/3) \leq \frac{nd_2}{3}$

1. $T(n) \leq \frac{2nd_2}{3} + \Theta(n)$

   对于一个确定的$\Theta(n)$显然存在一个d_2，
   $s.t. \forall n \geq$ 某一个n_2,有 $\frac{nd_2}{3} \geq \Theta(n)$
   $\therefore T(n) \leq nd_2$

2. $\frac{2nd_1}{3} + \Theta(n) \leq T(n)$

   对于一个确定的$\Theta(n)$显然存在一个d_1，
   $s.t. \forall n \geq$ 某一个n_1, 有 $\frac{nd_1}{3} \leq \Theta(n)$
   $\therefore nd_1 \leq T(n)$

$\therefore \forall n > max(n_1, n_2)\ \exists d_1, d_2,\ s.t.\ nd_1 \leq T(n) \leq nd_2$
$\therefore T(n) = \Theta(n).$

1.2 The Recursion-tree Method

1.2.1

1.2.1 4.4-4 of CLRS

Use a recursion tree to justify a good guess for the solution to the recurrence $T(n) = T(\alpha n) + T((1-\alpha)n) + \Theta(n)$, where α is a constant in the range $0 < \alpha < 1$.

1.2.1 Solution

不妨设$\alpha \geq \frac{1}{2}$, 否则令$\alpha' = 1 - \alpha$。
我们有递归树：$T(n) = T(\alpha) + T(1-\alpha) + cn.$
最高的高度$h_{max} = \log_{1/\alpha} n$
树的内部每一层的复杂度：cn
$\therefore T(n) = O(cn \log_{1/\alpha} n) = O(cn \lg n) = O(n \lg n)$

1.2.2

1.2.2 1.8 of *Algorithms* by Erickson

Use recursion trees to solve each of the following recurrences:

1. $N(n) = 2N(n/2) + O(n/\lg n)$
2. $Q(n) = \sqrt{2n}Q(\sqrt{2n}) + \sqrt{n}$

Note: A rigorous proof is not required, but an **illustration of your ideas** is neccessary.

1.2.2 Solution

1.2.2.1 $N(n) = 2N(n/2) + O(n/lgn)$

记开始为第0层最大高度$h_{max} = \lg n - 1$;
h层个数$num_h = 2^h$;
h层每一个$c(\frac{n}{2^h \lg \frac{n}{2^h}})$
h层总和：$c(\frac{n}{\lg \frac{n}{2^h}}) = c(n/(\lg n - \lg 2^h))) = c(N/(\lg n - h))$

$$\text{全部总和：} \sum_{h=0}^{h_{max}} c(\frac{n}{\lg n - h}) = \sum_{h=0}^{\lg n - 1} c(\frac{n}{\lg n - h})$$
$$= cn \sum_{x=1}^{\lg n} \frac{1}{x} = N(n)$$

设$A(n) = \sum_{x=1}^{\lg n} \frac{1}{x}$

设$B(n) = \sum_{x=2}^{\lg n} \frac{1}{x}$;

设$C(n) = \int_{x=1}^{(\lg n)+1} \frac{dx}{x} = \ln((\lg n) + 1)$;

设$D(n) = \int_{x=1}^{\lg n} \frac{dx}{x} = \ln \lg n$;

显然,通过积分的几何意义可知：$A(n) > C(n); B(n) < D(n)$;
又$A(n) = B(n) + 1$;
$\therefore$
$1: N(n) = cnA(n) > cnC(n) = cn \ln(\lg(n) + 1) = \Omega(n \lg(\lg n))$
$2: N(n) = cnA(n) = cN(B(n) + 1)$

$$< cn(1+D(n)) = cn + cn\ln(\lg(n)) = O(n\lg(\lg n))$$

$\therefore N(n) = \Theta(n(\lg(\lg n)))$

1.2.2.2 $Q(n) = \sqrt{2n}Q(\sqrt{2n}) + \sqrt{n}$

方法一

本来自身的想法是两边同时除以$2n$然后$m = \lg n$换元，
其实和后一种方法思路相似，只是少了一次换元

$$\begin{aligned}
\text{设} \quad P(n) &= Q(n)/n; \\
we\ have: nP(n) &= \sqrt{2n}\sqrt{2n}P(\sqrt{2n}) + \sqrt{n} \\
\therefore P(n) &= 2P(\sqrt{2n}) + 1/\sqrt{n}; \\
Let: m &= \lg n \\
we\ have: P(2^m) &= 2P(2^{(m+1)/2}) + \frac{1}{2^{m/2}}; \\
\text{可能也这个式子一样}: P(2^m) &= 2P(2^{m/2}) + \frac{1}{2^{m/2}}; \\
\text{记开始为第0层} & \\
\text{最大高度} h_{max} &=; \\
h\text{层个数} num_h &= 2^h; \\
h\text{层每一个} &=; \\
h\text{层总和}: &=;
\end{aligned}$$

事实上，这个时候如果令 $m = 2^k$就可以继续做了，目的就是找到k和$k-1$的递推式

方法二

> 作者：vonbrand
> 链接：https://math.stackexchange.com/questions/3582790/solve-the-recurrence-tn-sqrt2nt-sqrt2n-sqrtn
> 来源：math.stackexchange

$$
\begin{aligned}
\text{设}\ \ n &= 2\cdot 2^{2^k} \\
\text{则}\ \ k &= \lg\lg\frac{n}{2}\ \ (k\geq\ 0) \\
\sqrt{2n} &= \sqrt{2\cdot 2\cdot 2^{2^k}} \\
&= 2\cdot 2^{2^{k-1}} \\
\sqrt{n} &= \sqrt{2}\cdot 2^{2^{k-1}} \\
Q(2\cdot 2^{2^k}) &= 2\cdot 2^{2^{k-1}}Q(2\cdot 2^{2^{k-1}})+\sqrt{2}\cdot 2^{2^{k-1}}
\end{aligned}
$$

$$
\begin{aligned}
Let: P(k) &= Q(2\cdot 2^{2^k}) \\
\therefore P(k) &= 2\cdot 2^{2^{k-1}}P(k-1)+\sqrt{2}\cdot 2^{2^{k-1}} \\
\frac{P(k)}{2^k\cdot 2^{2^k}} &= \frac{P(k-1)}{2^{k-1}\cdot 2^{2^{k-1}}}+\frac{\sqrt{2}}{2^k\cdot 2^{2^{k-1}}}
\end{aligned}
$$

$$
\begin{aligned}
Let: M(k) &= \frac{P(k)}{2^k\cdot 2^{2^k}} \\
\therefore M(k) &= M(k-1)+\frac{\sqrt{2}}{2^k\cdot 2^{2^{k-1}}} \\
\therefore M(k) &= M(0)+\sum_{i=1}^{k}\frac{\sqrt{2}}{2^i\cdot 2^{2^{i-1}}} \\
1:\quad M(k) &\leq M(0)+\sum_{i=1}^{k}\frac{\sqrt{2}}{2^i} \\
&\leq M(0)+\frac{\sqrt{2}}{2} \\
\therefore P(k) &\leq (\frac{P(0)+\sqrt{2}}{2})(2^k\cdot 2^{2^k})=O(2^k\cdot 2^{2^k}) \\
2:\quad M(k) &\geq\ M(0) \\
\therefore P(k) &\geq\ (\frac{P(0)}{2})(2^k\cdot 2^{2^k})=\Omega(2^k\cdot 2^{2^k})
\end{aligned}
$$

$$
\begin{aligned}
\therefore P(k) &= \Theta(2^k\cdot 2^{2^k}) \\
Q(n)=P(k) &= \Theta((\lg n)\cdot n)=\Theta(n\lg n)
\end{aligned}
$$

1.3 The Master Method

(From CLRS 4.5-1)

Use the master method to give tight asymptotic bounds for the following recurrences:

1. $T(n) = 2T(n/4) + \sqrt{n}\lg^2 n$
2. $T(n) = 2T(n/4) + n^2$

1.3 Solution

1.3.1 $T(n) = 2T(n/4) + \sqrt{n}\lg^2 n$

$$\lim_{n->\infty} \frac{\sqrt{n}\lg^2 n}{n^{\log_4 (2+\epsilon)}} = \lim_{n->\infty} \frac{\lg^2 n}{n^{\log_4 1+\frac{\epsilon}{2}}} \overset{\text{洛必达两次}}{=\!=\!=\!=\!=} \infty$$
$$\therefore f(n) = \Omega(n^{\log_b a+\epsilon})$$
$$\therefore T(n) = \Theta(f(n)) = \Theta(\sqrt{n}\lg^2 n)$$

1.3.2 $T(n) = 2T(n/4) + n^2$

$$\lim_{n->\infty} \frac{n^2}{n^{\log_4 2}} = \lim_{n->\infty} n^{3/2} = \infty$$
$$\therefore f(n) = \Omega(n^{\log_b a+\epsilon})$$
$$\therefore T(n) = \Theta(f(n)) = \Theta(n^2)$$

2 Heap

2.1 Heap Sort Running Time

(From CLRS 6.4-5)

```
HEAPSORT(A, n)
BUILD-MAX-HEAP(A, n)
for i = n downto 2
    exchange A[1] with A[i]
    A.heap-size = A.heap-size - 1
    MAX-HEAPIFY(A, 1)
```

Given the psedocode of HEAPSORT, show that when all the elements of A are distinct, the best-case running time of HEAPSORT is $\Omega(n \lg n)$.

2.1 Solution

`BULID-MAXHEAP` = $O(n\lg n)$ (粗略估计,上完课后我们知道是$O(n)$)
而循环中，我们循环了n次，每一次将堆末尾和头交换，然后头进行sinkdown，
sinkdown中，我们最多沉到底部，即进行$\lg i$次，
所以时间为 $\sum\limits_{i=n}^{2} \lg i = \lg(n!) = \Theta(n\lg n) = \Omega(n\lg n)$