Summations and Recurrence Relations[1]

CS331 and CS531

Design and Analysis of Algorithms

Ajay Gupta

Don Nelson

Version 1: January 3, 1992

June 26, 2003

[1]Note: These are informal notes that are to be used only as a supporting material. Use them at your own risk. Please report any errors and typographical mistakes to Ajay Gupta (gupta@cs.wmich.edu) or Don Nelson (nelson@cs.wmich.edu).

1 Math Preliminaries

Below you will find some formulae that are useful for analysis of algorithms. The list is by no means exhaustive, rather it is designed to give you a flavor of different ideas we may need.

Conventions

- $\forall \iff$ for all symbol.
- $\exists \iff$ there exists symbol.
- $0! = 1$.
- $x^0 = 1$.
- $log_a 1 = 0,$ for all $a > 0$.
- $\log y \iff$ base two log of y when no explicit base is indicated; $\forall y \geq 1$.
- $\sum_{i=x}^{y} f(i) = 0$ if $x > y$.
- $\sum_{i=x}^{x} f(i) = f(x)$.

Logarithms

- $\log_x y = z \iff x^z = y; \forall x > 1$.
- $x^{\log_x y} = y; \forall x > 1$.
- $\log_x(a \times b) = \log_x a + \log_x b; \forall x > 1$ and $a, b > 0$.
- $\log_x \frac{a}{b} = \log_x a - \log_x b. \forall x > 1$ and $a, b > 0$.
- $\log_x a = \frac{\log_y a}{\log_y x}; \forall x, y > 1$.

- $a^{\log_x b} = b^{\log_x a}; \forall x > 1.$
- $\log_x a^b = b \log_x a.$

Powers

- $a^b \times a^c = a^{b+c}.$
- $\frac{a^b}{a^c} = a^{b-c}.$
- $(a^b)^c = a^{bc}.$
- $a^c \times b^c = (ab)^c.$

2 Series Sums

In this section we will consider a number of examples of various series that occur often during solutions of recurrence relations and frequency counting. The *sum* ($\sum$) notation is often used to denote a series in more succinct form and hence let's first start by understanding this notation. In general, $\sum_{i=x}^{y} f(i)$ denotes the longer form $f(x) + f(x+1) + f(x+2) + \cdots + f(y-1) + f(y)$ where y, x are integers, $y > x$ and $f(\cdot)$ is some function. Observe that $f(\cdot)$ terms occur $y - x + 1$ number of times and i is called a *running variable* of the *sum*. Here are some useful facts about the *sum* notation.

- $\sum_{i=x}^{y} f(i) = \sum_{j=x}^{y} f(j) = \sum_{k=x}^{y} f(k)$. In other words i, j, k are simply dummy variables. (Pause: Compare the longer forms of the *sum*'s!!)
- If $f(\cdot)$ is a constant function as far as its dependence on the running variable goes (e.g. $i, j,$ and k), then $\sum_{i=x}^{y} f(a) = f(a) \sum_{i=x}^{y} 1 = (y - x + 1) f(a)$, for any $a \neq i$.

- $\sum_{i=x}^{y}(f(i)+g(i)) = \sum_{i=x}^{y} f(i) + \sum_{i=x}^{y} g(i)$ for some functions f and g.
  (Pause: Is $\sum_{i=x}^{y} f(i) + \sum_{k=x}^{y} g(k) = \sum_{i=x}^{y}(f(i)+g(i))$?)
  (Long Pause: Is $\sum_{i=x}^{y}(f(i) \times g(i)) = (\sum_{i=x}^{y} f(i)) \times (\sum_{i=x}^{y} g(i))$?)

- $\sum_{i=x}^{y}(g(j) \times f(i)) = g(j) \times \sum_{i=x}^{y} f(i)$ whenever $g(j)$ is not a function of the running variable i.

- $\sum_{i=x}^{y} f(i) = \sum_{i=x}^{z} f(i) + \sum_{k=z+1}^{y} f(k); \forall z$ such that $x \leq z \leq y$.

2.1 Arithmetic-Like Series

Let's consider the series of the form $\sum_{i=1}^{n} i^k$ for some positive constant k, and try to establish "closed forms" represented by these. We know that $\sum_{i=1}^{n} 1 = n$ (why?). How about $\sum_{i=1}^{n} i$? This series represents sum of the first n natural numbers. Let's first consider the case when n is an even integer. If we add the first and last number we get $(1+n)$, add the second and the second-last number we get $(2+(n-1)) = (1+n)$ and add the third and the third-last number we get $(3+(n-2)) = (1+n)$. Continuing in this fashion, it is easy to see that we need to add a total of $n/2$ pairs each one resulting in a sum of $(1+n)$. Hence, one can say that $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ for even n.

(Pause: What happens when we use a similar strategy but n is odd?)

Now, suppose that we claim $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ for all $n > 0$. How can we be sure that in fact our claim is correct? One way to make sure is to use mathematical induction. So, let's try it.

For $n = 1, 2$ we can easily verify that our claim is true and hence our claim holds for the base case. Now by induction hypothesis (IH) assume that $\sum_{i=1}^{k} i = \frac{k(k+1)}{2}$ for all values of $k < n$. We need to prove that our claim holds for $k = n$. We know that

$$\begin{aligned}
\sum_{i=1}^{n} i &= \sum_{i=1}^{n-1} i + n \\
&= \frac{(n-1)((n-1)+1)}{2} + n \qquad \text{use IH with } k = n-1 \\
&= \frac{n(n+1)}{2}
\end{aligned}$$

∎

Let us now consider the series $\sum_{i=1}^{n} i^2$. In order to find a closed form for this series, we illustrate a method which is also applicable for finding closed forms for the series $\sum_{i=1}^{n} i^k$ for $k > 2$. It is easy (why?) to verify that

$$\sum_{i=1}^{n}(i+1)^3 - \sum_{i=1}^{n} i^3 = (n+1)^3 - 1 \tag{1}$$

We first expand the left-hand-side (LHS) of equation 1 so that terms involving only i^2, i and 1 remain. We thus have

$$\begin{aligned}
\sum_{i=1}^{n}(i+1)^3 - \sum_{i=1}^{n} i^3 &= \sum_{i=1}^{n}(i^3 + 3i^2 + 3i + 1) - \sum_{i=1}^{n} i^3 \\
&= \sum_{i=1}^{n} i^3 + 3\sum_{i=1}^{n} i^2 + 3\sum_{i=1}^{n} i + \sum_{i=1}^{n} 1 - \sum_{i=1}^{n} i^3 \\
&= 3\sum_{i=1}^{n} i^2 + 3\sum_{i=1}^{n} i + \sum_{i=1}^{n} 1 \qquad (2)
\end{aligned}$$

Now, using equation 2 and the right-hand-side (RHS) of equation 1, we obtain

$$3\sum_{i=1}^{n} i^2 + 3\sum_{i=1}^{n} i + \sum_{i=1}^{n} 1 = (n+1)^3 - 1 \tag{3}$$

$$\implies \quad 3\sum_{i=1}^{n} i^2 = (n+1)^3 - 1 - 3\sum_{i=1}^{n} i - \sum_{i=1}^{n} 1 \tag{4}$$

$$\begin{aligned} \implies \quad \sum_{i=1}^{n} i^2 &= ((n+1)^3 - 1 - 3n(n+1)/2 - n)/3 \\ &= n(n+1)(2n+1)/6 \qquad \blacksquare \end{aligned}$$

Break: Try to generalize above strategy to find a closed form for $\sum_{i=1}^{n} i^3$.

Long Break: Try to find a closed form for $\sum_{i=1}^{n} i^k$ *for* $k > 3$.

2.2 Geometric-Like Series

Suppose that we have a series of the form $\sum_{i=x}^{y} c^i$ for some constant c. These kind of series are known as *geometric* series. Following is one way to find a closed form for these. Let S denote the closed form, then

$$\begin{aligned} S &= \sum_{i=x}^{y} c^i \qquad (5) \\ c \times S &= c \times \sum_{i=x}^{y} c^i \\ &= \sum_{i=x}^{y} c^{i+1} \\ c \times S &= \sum_{i=x+1}^{y+1} c^i \qquad (6) \end{aligned}$$

Subtracting equation 5 from equation 6, we obtain

$$\begin{aligned} (c-1)S &= \sum_{i=x+1}^{y+1} c^i - \sum_{i=x}^{y} c^i \\ &= c^{y+1} - c^x \\ \implies \quad S &= (c^{y+1} - c^x)/(c-1) \qquad \blacksquare \end{aligned}$$

Let k be a positive constant and c be a constant. How about a series of

the form $\sum_{i=1}^{n}(i^k \times c^i)$ which looks similar to a combination of an arithmetic-like and a geometric series? We now describe a method which helps us in finding a closed form when $k = 1$; i.e., closed form for the series

$$\sum_{i=1}^{n} ic^i. \tag{7}$$

We can first use a strategy which is similar to the one for geometric series and then strategy similar to the one for arithmetic-like series. Let S again denote the closed form for the series 7. We have:

$$\begin{aligned} S &= \sum_{i=1}^{n} ic^i & (8)\\ c \times S &= c \times \sum_{i=1}^{n} ic^i \\ &= \sum_{i=1}^{n} ic^{i+1} \\ c \times S &= \sum_{i=2}^{n+1} (i-1)c^i & (9) \end{aligned}$$

Subtracting equation 8 from equation 9, we obtain

$$\begin{aligned} (c-1)S &= \sum_{i=2}^{n+1}(i-1)c^i - \sum_{i=1}^{n} ic^i \\ &= (nc^{n+1} + \sum_{i=2}^{n}(i-1)c^i) - (\sum_{i=2}^{n} ic^i + c) \\ &= nc^{n+1} - c + (\sum_{i=2}^{n}(i-1)c^i - \sum_{i=2}^{n} ic^i) \\ &= nc^{n+1} - c + \sum_{i=2}^{n}(i-1-i)c^i \\ &= nc^{n+1} - c - \sum_{i=2}^{n} c^i \\ &= nc^{n+1} - c - (c^{n+1} - c^2)/(c-1) \\ \implies \qquad S &= (nc^{n+1} - c)/(c-1) - (c^{n+1} - c^2)/(c-1)^2 \qquad \blacksquare \end{aligned}$$

Long Break: Try finding closed form for $\sum_{i=1}^{n} i^2 c^i$.

3 Recurrence Relations

Suppose that $T(n)$ represents the cost of performing an algorithm on a data set of size n. Often $T(n)$, when originally constructed, will not be in some "closed form" representing a function of n, rather, it will be given in terms of one or more values of $T(k)$ for values of k smaller than n. Consider the following examples:

$$T(n) = \begin{cases} 1 & \text{if n = 0,1} \\ T(n-1) + T(n-2) & \text{if } n > 1 \end{cases} \tag{10}$$

$$T(n) = \begin{cases} 1 & \text{if } n = 0, 1 \\ T(n/2) + n & \text{if } n > 1 \end{cases} \tag{11}$$

$$T(n) = \begin{cases} c & \text{if n = 1} \\ c + \sum_{i=1}^{n-1} T(i) & \text{if } n > 1 \end{cases} \tag{12}$$

Equation 10 is given in terms of the previous two terms; equation 11 is given in terms of the value of T at $n/2$; and equation 12 is given in terms of all the preceding values of T. The last equation is referred to as a *full history* recurrence relation.

Whenever such recurrence relations represent the cost of performing an algorithm, it becomes important to establish a bound on T as a function of n, the size of the problem. For example, can we establish a bound on $T(n)$ if T is given by equation 10? It is easy to show using mathematical induction that 2^n is a bound. First observe that $T(0) \leq 2^0$ and $T(1) \leq 2^1$. Thus, we

have the basis for an inductive proof. Now suppose that $T(k) \leq 2^k$ for all $k < n$. Then

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) \\ &\leq 2^{n-1} + 2^{n-2} \\ &< 2^{n-1} + 2^{n-1} \\ &= 2^n \end{aligned} \tag{13}$$

Note that in the inequality expressed in 13 the value of 2^{n-2} was over estimated by 2^{n-1} (a factor of 2). This is a rather *coarse* estimate, and it might lead one to suspect that a better bound could be found. In fact that is the case. It can be shown that there is a constant c such that $T(n) < c\lambda^n$ where

$$\lambda = \frac{\sqrt{5}+1}{2} = 1.618\cdots$$

Next we will look at equation 11 and try an upper bound on $T(n)$, say n^2. In this case we will restrict our values of n to powers of 2. In other words, can it be shown that $T(2^p) \leq 2^{2p}$ for $p = 0, 1, 2, \ldots$? Here we can use induction on p. Observe that $T(1) = T(2^0) = 2^0$, so it is true for $p = 0$. Assume that it is true for $p = k$. Then

$$\begin{aligned} T(2^{k+1}) &= T(2^k) + 2^{k+1} \\ &\leq 2^{2k} + 2^{k+1} \\ &< 2^{2k} + 2^{2k} \qquad (14) \\ &= 2^{2k+1} \\ &< 2^{2k+2} \qquad (15) \end{aligned}$$

$$= \quad 2^{2(k+1)}$$

Again, notice in the inequalities 14 and 15 that extremely coarse overestimates were made. As was the case with the estimate for equation 10, this might lead one to suspect that the bound we are considering is much too large. Let's try a smaller bound. Can we show that $T(n) \leq 2n - 1$. It is certainly true for $n = 1 = 2^0$. Assume that it is true for $n = 2^k$. Then

$$\begin{aligned} T(2^{k+1}) &= T(2^k) + 2^{k+1} \\ &\leq 2 \times 2^k - 1 + 2^{k+1} \\ &= 2 \times 2^{k+1} - 1 \end{aligned}$$

As a final observation about this last bound, note that it is easy to show that $T(n) = 2n - 1$; i.e., we can find an exact solution (for powers of 2) instead of merely finding an upper bound.

In each of the examples that we have considered, an upper bound was proposed, and then shown to work. Other guesses can also be made, but it may be difficult or impossible to establish that the proposed bound is actually a bound. For example, could you show for equation 11 that $T(n) \leq c \times \log_2 n$?

3.1 The Iteration Method

Establishing whether or not a given function represents a bound for a recurrence relation is one thing, but coming up with a reasonable bound is another. In the following material we will use the method of expanding a recurrence relation in order to arrive at a solution or a bound. To see how this method works, let's consider again the recurrence relation given in 11. Suppose that $n = 2^p$ for some integer p. To expand the relation, we will start

with n and work our way down, step by step, until a pattern can be seen.

$$\begin{aligned}
T(n) &= T(n/2) + n \\
&= T(n/2^2) + n/2 + n \\
&= T(n/2^3) + n/2^2 + n/2 + n \\
&\vdots \\
&= T(n/2^p) + n\sum_{i=0}^{p-1} 1/2^i \\
&= 1 + n\sum_{i=0}^{p-1} 1/2^i \\
&= 1 + n\frac{1-(1/2)^p}{1-1/2} \\
&= 1 + 2n(1-(1/n)) \\
&= 1 + 2n - 2 \\
&= 2n - 1
\end{aligned}$$

Next we will illustrate another expansion to solve the following recurrence relation.

$$T(n) = \begin{cases} c & \text{if n} = 1 \\ 2T(n/2) + cn^2 & \text{if } n > 1 \end{cases} \tag{16}$$

As was the case with the first expansion above, we will assume that $n = 2^p$ for some integer p. Then the expansion proceeds as follows.

$$\begin{aligned}
T(n) &= 2T(n/2) + cn^2 \\
&= 2(2T(n/2^2) + c(n/2)^2) + cn^2 \\
&= 2^2T(n/2^2) + cn^2(1/4 + 1) \\
&= 2^2(2T(n/2^3) + c(n/2^2)^2) + cn^2(1/4 + 1)
\end{aligned}$$

$$
\begin{aligned}
&= 2^3 T(n/2^3) + cn^2(1/16 + 1/4 + 1) \\
&= 2^3(2T(n/2^4) + cn/2^3)^2) + cn^2(1/16 + 1/4 + 1) \\
&= 2^4 T(n/2^4) + cn^2(1/64 + 1/16 + 1/4 + 1) \\
&\vdots \\
&= 2^p T(n/2^p) + cn^2 \sum_{i=0}^{p-1} 1/4^i \\
&= c2^p + cn^2 \sum_{i=0}^{p-1} 1/4^i \\
&= cn + cn^2 \frac{1 - (1/4)^p}{3/4} \\
&= cn + 4cn^2/3(1 - (1/4)^p) \\
&= cn + 4cn^2/3(1 - 1/n^2) \\
&= cn + 4c/3(n^2 - 1) \\
&= c(4n^2/3 + n - 4/3)
\end{aligned}
$$

Note that in each of the above expansions, we continued to apply the recurrence relation until a recognizable pattern was reached, and at that point we jumped all the way to the pth step. It would take an inductive argument to prove that such a jump is legitimate. To actually verify that the expressions we derived are solutions, one should really show by induction that the solution is valid for all powers 2^p. The induction can be done on p.

3.2 The Master Method

For an another expansion, consider the following recurrence relation.

$$
T(n) = \begin{cases} c & \text{if n} = 1 \\ aT(n/b) + cn^k & \text{if } n > 1 \end{cases} \tag{17}
$$

This relation is similar to those expanded above; however, we observe that it is expressed in terms of the general parameters, a, b, c and k. We will proceed to expand this in the same way; however, we will obtain only upper bounds rather than exact solutions. At one point in the expansion, we will find it necessary to consider cases which depend on the parameters a, b, and k. Let $\frac{n}{b^p} = 1$ for some $p > 0$.

$$\begin{aligned}
T(n) &= aT(n/b) + cn^k \\
&= a^2T(n/b^2) + acn^k/b^k + cn^k \\
&= a^2(T(n/b^2) + cn^k(a/b^k + 1) \\
&= a^2(aT(n/b^3) + cn^k/b^{2k}) + cn^k(a/b^k + 1) \\
&= a^3T(n/b^3) + cn^k((a/b^k)^2 + a/b^k + 1) \\
&\vdots \\
&= a^pT(1) + cn^k \sum_{i=0}^{p-1} (a/b^k)^i \\
&= ca^p + cn^k \sum_{i=0}^{p-1} (a/b^k)^i
\end{aligned}$$

The sum in the last equation above represents a geometric series, so its growth is dependent on the ratio (a/b^k). To estimate the growth of $T(n)$ we will consider three cases for the ratio.

Case 1 $(a < b^k)$

In this case $\sum_{i=0}^{p-1}(a/b^k)^i$ is bounded by $\frac{1}{1-a/b^k}$, which we will denote by G. Thus we can write

$$\begin{aligned} T(n) &\leq ca^p + Gcn^k \\ &= ca^{\log_b(n)} + Gcn^k \\ &= cn^{\log_b(a)} + Gcn^k \\ &< cn^k + Gcn^k \\ &= O(n^k) \end{aligned}$$

Case 2 $(a = b^k)$

In this case $\sum_{i=0}^{p-1}(a/b^k)^i = p = \log_b(n)$, and we can write

$$\begin{aligned} T(n) &= ca^p + cn^k \log_b(n) \\ &= cn^{\log_b(a)} + cn^k \log_b(n) \\ &= cn^k + cn^k \log_b(n) \\ &= O(n^k \log_b(n)) \end{aligned}$$

Case 3 $(a > b^k)$

In this case $\frac{1}{a/b^k - 1}$ is a positive constant, which we will denote by H. Then

$$\sum_{i=0}^{p-1}(a/b^k)^i = \frac{(a/b^k)^p - 1}{a/b^k - 1}$$

and hence:

$$\begin{aligned} T(n) &= ca^{\log_b(n)} + c(a/b^k - 1)^{-1} n^k (a^p/n^k - 1) \\ &= cn^{\log_b(a)} + cHn^{\log_b(a)} - cHn^k \\ &= O(n^{\log_b(a)}) \end{aligned}$$

Given a recurrence equation in the form of equation (8), we now can quickly find the order of growth by considering a/b^k and applying the appropriate case. This, of course, only gives the order of growth and does not produce an exact solution.

3.3 Full History Recurrence Relation

We will now consider one final example, namely that of a full history recurrence relation. This particular equation will arise when we study the average case performance of *quicksort*. The equation is:

$$T(n) = (n-1) + \frac{2}{n} \sum_{i=1}^{n-1} T(i) \tag{18}$$

with $T(1) = 0$. We first eliminate the summation in the following way. First consider equation 18 and multiplying both sides by n we obtain:

$$nT(n) = n(n-1) + 2 \sum_{i=1}^{n-1} T(i) \tag{19}$$

Now, replacing n by $n+1$ in equation 18 and then multiplying both sides by $(n+1)$ we get:

$$\begin{aligned} T(n+1) &= n + \frac{2}{n+1} \sum_{i=1}^{n} T(i) \\ (n+1)T(n+1) &= n(n+1) + 2 \sum_{i=1}^{n} T(i) \end{aligned} \tag{20}$$

Subtracting equation 19 from equation 20 and solving for $T(n+1)$ yields:

$$\begin{aligned}(n+1)T(n+1) - nT(n) &= 2n + 2T(n) \\ T(n+1) &= \frac{2n}{n+1} + \frac{n+2}{n+1}T(n) \qquad (21)\end{aligned}$$

Notice that Equation 21 now gives $T(n+1)$ in terms of $T(n)$. As before, we will attempt to expand based on 21, only the method will be somewhat different than that used for the previous recurrence relations. We first make an observation about the term $2n/(n+1)$ in equation 21. It is bounded above by 2, and as n becomes large, it is very close to 2. By replacing $2n/(n+1)$ with 2, we can convert the equation to an inequality and have a system that is much simpler to solve. Since our main concern will ultimately be to find a bound on the growth of $T(n)$, this substitution will not defeat our goal.

$$\begin{aligned}T(n+1) &< \frac{n+2}{n+1}T(n) + 2 \\ &= \frac{n+2}{n+1}(\frac{n+1}{n}T(n-1) + 2) + 2 \\ &= \frac{n+2}{n}T(n-1) + \frac{2(n+2)}{n+1} + 2 \\ &= \frac{n+2}{n}(\frac{n}{n-1}T(n-2) + 2) + \frac{2(n+2)}{n+1} + 2 \\ &= \frac{n+2}{n-1}T(n-2) + 2(n+2)\sum_{i=0}^{2}\frac{1}{n+2-i} \\ &= \frac{n+2}{n-1}(\frac{n-1}{n-2}T(n-3) + 2) + 2(n+2)\sum_{i=0}^{2}\frac{1}{n+2-i} \\ &= \frac{n+2}{n-2}T(n-3) + 2(n+2)\sum_{i=0}^{3}\frac{1}{n+2-i} \\ &\vdots\end{aligned}$$

$$= \frac{n+2}{n-p+1}T(n-p) + 2(n+2)\sum_{i=0}^{p}\frac{1}{n+2-i}$$

Letting $p = n-1$ we obtain

$$\begin{aligned}
T(n+1) &\leq \frac{n+2}{2}T(1) + 2(n+2)\sum_{i=0}^{n-1}\frac{1}{n+2-i} \\
&= 2(n+2)\sum_{i=0}^{n-1}\frac{1}{n+2-i} \\
&= 2(n+2)\left(\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\cdots+\frac{1}{n+1}+\frac{1}{n+2}\right) \\
&= O((n+2)\log_2(n+2)) \\
&= \Theta((n+1)\log_2(n+1))
\end{aligned}$$

Exercises

1. Find an exact solution for:

$$T(n) = \begin{cases} c & \text{if } n = 0 \\ 2T(n/2) + cn & \text{if } n > 1 \end{cases}$$

2. Expand and find a bound for the following recurrence relation. Then check how your answer compares with the solution to the general case we derived.

$$T(n) = \begin{cases} c & \text{if } n \leq 2 \\ 7T(n/2) + cn^2 & \text{if } n > 2 \end{cases}$$

For each of the recurrence relation given below, first using the expansion/iteration method find a close upper bound, and then prove by induction that your solution is correct. Assume c and k are constants.

3.
$$T(n) = \begin{cases} c & \text{if } n = 1 \\ T(\frac{n}{2}) + c \log n & \text{if } n \geq 2 \end{cases}$$

4.
$$T(n) = \begin{cases} c & \text{if } n = 1, 2 \\ 2T(\frac{n}{2}) + c \log n^2 & \text{if } n \geq 3 \end{cases}$$

5.
$$T(n) = \begin{cases} ck & \text{if } n = 1 \\ 4T(\frac{n}{2}) + c \log n & \text{if } n \geq 2 \end{cases}$$

6.
$$T(n) = \begin{cases} c & \text{if } 1 \leq n \leq 2 \\ T(\frac{n}{2}) + cn & \text{if } n \geq 3 \end{cases}$$

7.
$$T(n) = \begin{cases} c & \text{if } n = 1 \\ 3T(\frac{n}{2}) + cn & \text{if } n \geq 2 \end{cases}$$

8.
$$T(n) = \begin{cases} c & \text{if } n = 1 \\ 2T(\frac{n}{2}) + cn \log n & \text{if } n \geq 2 \end{cases}$$

9.
$$T(n) = \begin{cases} c & \text{if } n = 1 \\ 3T(\frac{n}{2}) + cn \log n & \text{if } n \geq 2 \end{cases}$$

10.
$$T(n) = \begin{cases} c & \text{if } n = 1 \\ T(\frac{n}{4}) + cn \log n & \text{if } n \geq 2 \end{cases}$$

11.
$$T(n) = \begin{cases} c & \text{if } n = 1, 2, 3, 4 \\ T(\frac{n}{2} + T(\frac{n}{4}) + cn & \text{if } n \geq 5 \end{cases}$$