Often when analyzing a divide & conquer algorithm, we obtain a recurrence for its running time of the following form

\[ T(n) = aT\left(\frac{n}{b}\right) + cn^k \]  

(1)

In words, on input size \( n \), the algorithm generates \( a \) subproblems, each of size \( n/b \); combining these subproblems to obtain the overall solution requires time polynomial in \( n \), specifically \( cn^k \).

Such recurrences appear frequently so it is useful to know asymptotic bounds for them in terms of \( a, b \) and \( k \) (as we will see, \( c \) does not affect the asymptotic solution). To this end, we will analyze the recursion tree for this recurrence (see Figure 1).

Figure 1: The recursion tree for recurrence (1). \( a \) is the branching factor, \( b \) is the factor by which the input size shrinks at every recursive call and \( cn^k \) is the time required to combine the solutions to the subproblems into the overall solution for input size is \( n \). The smallest possible size of a subproblem is \( O(1) \); typically, solving input instances of small constant size requires constant time \( c \).

Note that

- \( a \) is the branching factor of the tree: every subproblem gives rise to \( a \) new subproblems at the next level of the tree; thus
  1. at level 1, we have \( a \) subproblems
  2. at level 2, each of the \( a \) subproblems in level 1 gives rise to \( a \) new subproblems; therefore there are a total of \( a^2 \) subproblems
  3. at level 3, each of the \( a^2 \) subproblems in level 2 generates \( a \) new subproblems; therefore there are a total of \( a^3 \) subproblems
  4. at the level \( i \), there are \( a^i \) subproblems
• $b$ is the factor by which the input size shrinks at every level; thus

1. at level 1, the input size of each subproblem shrinks by a factor of $b$, that is, from $n$ it now becomes $n/b$;
2. at level 2, the input size of each subproblem further shrinks by a factor of $b$, that is, from $n/b$ it now becomes $(n/b)/b = n/b^2$;
3. at level 3, the input size of each subproblem again shrinks by a factor of $b$, hence becomes $(n/b^2)/b = n/b^3$;
4. at level $i$, the size of each subproblem is $n/b^i$

$\Rightarrow$ at level $i$, the amount of work spent on each subproblem of size $n/b^i$ is $1$:

$$c \left( \frac{n}{b^i} \right)^k$$

$\Rightarrow$ at level $i$, the work spent on all subproblems is

$$a^i c \left( \frac{n}{b^i} \right)^k = cn^k \left( \frac{a}{b^k} \right)^i$$

We need one more observation before we can compute the total work spent on the recursion tree.

**Fact 1** The depth of the tree in Figure 1 is $\lceil \log_b n \rceil$ levels.

**Proof.** The last level of the recursion tree, call it $d$, consists of subproblems of size 1. Since at level $i$ subproblems have size $n/b^i$, we are looking for $d$ such that

$$\frac{n}{b^d} = 1 \Rightarrow d = \log_b n$$

Since $d$ is an integer, $d = \lceil \log_b n \rceil$. $\square$

We are now ready to derive a bound for $T(n)$ by computing the total work spent on this recursion tree, which is given by the sum of the work spent at each level of the tree:

$$T(n) = \sum_{i=0}^{\lceil \log_b n \rceil} cn^k \left( \frac{a}{b^k} \right)^i = \left( \frac{a}{b^k} \right)^0 + \sum_{i=0}^{\lceil \log_b n \rceil} \frac{a}{b^k} \left( \frac{a}{b^k} \right)^i$$

(2)

Note that $T(n)$ depends on a sum over $\lceil \log_b n \rceil$ terms of a geometric progression with common ratio $a/b^k$ and initial value $(a/b^k)^0 = 1$. Depending on the value of the common ratio $a/b^k$, this sum will exhibit the following behavior:

1. $\frac{a}{b^k} = 1$; in this case, we have

$$\sum_{i=0}^{\lceil \log_b n \rceil} \left( \frac{a}{b^k} \right)^i = \sum_{i=0}^{\lceil \log_b n \rceil} 1 = \lceil \log_b n \rceil + 1 = \Theta(\log_b n)$$

(3)

2. $\frac{a}{b^k} < 1$; in this case, you can show that the sum of the entire geometric progression is dominated by its initial value, that is,

$$\sum_{i=0}^{\lceil \log_b n \rceil} \left( \frac{a}{b^k} \right)^i = \Theta \left( \left( \frac{a}{b^k} \right)^0 \right) = \Theta(1)$$

(4)

---

1 Recall that the amount of work spent on combining the subproblems when the the input size is $n$ is $cn^k$. 

3. \( \frac{a}{b^k} > 1 \); again, you can show that the sum of the entire geometric progression is now dominated by its last term, that is,

\[
\sum_{i=0}^{\lfloor \log_b n \rfloor} \left( \frac{a}{b^k} \right)^i = \Theta \left( \left( \frac{a}{b^k} \right)^{\log_b n} \right) = \Theta \left( \frac{a^{\log_b n}}{b^{k\log_b n}} \right) = \Theta \left( \frac{n^{\log_b a}}{n^k} \right)
\]  

(5)

Plugging back equations (3), (4), (5) into equation (2), we summarize our findings in the following theorem.

**Theorem 1 (Master theorem)** If \( T(n) = aT(\lfloor n/b \rfloor) + O(n^k) \) for some constants \( a > 0, b > 1, k \geq 0 \), then

\[
T(n) = \begin{cases} 
O(n^{\log_b a}), & \text{if } a > b^k \\
O(n^k \log n), & \text{if } a = b^k \\
O(n^k), & \text{if } a < b^k 
\end{cases}
\]