Master Theorem

The master theorem is used in calculating the time complexity of recurrence relations (divide and conquer algorithms) in a simple and quick way.

If $a \geq 1$ and $b > 1$ are constants and f(n) is an asymptotically positive function, then the time complexity of a recursive relation is given by

T(n) = aT(n/b) + f(n)

where n = size of the problem
a = number of subproblems in the recursion and a >= 1
n/b = size of each subproblem
b > 1, k >= 0 and p is a real number.

Then T(n) has the following asymptotic bounds:

1) If $a > b^k$, then $T(n) = \Theta\left(n^{log_b^a}\right)$
2) If $a = b^k$
   a. If $p > -1$, then $T(n) = \Theta\left(n^{log_b^a} log^{p+1} n\right)$
   b. If $p = -1$, then $T(n) = \Theta\left(n^{log_b^a} loglogn\right)$
   c. If $p < -1$, then $T(n) = \Theta\left(n^{log_b^a}\right)$
3) If $a < b^k$
   a. If $p \geq 0$, then $T(n) = \Theta(n^k log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Master Theorem Limitations

The master theorem cannot be used if:
1. T(n) is not monotone. eg. T(n) = sin n
2. f(n) is not a polynomial. eg. f(n) = 2n
3. a is not a constant. eg. a = 2n
4. a < 1

Problems for Practice:

Practice Problems for Master Theorem:- Solution

1.	$T(n) = 2\,T\left(\frac{n}{2}\right) + \frac{n}{\log n}$	$n \log\log n$
2.	$T(n) = 2 \cdot T\left(\frac{n}{4}\right) + n^{0.51}$	$n^{0.51}$
3.	$T(n) = 6T\left(\frac{n}{3}\right) + n^2 \log n$	$O(n^2 \log n)$
4.	$6\,T\left(\frac{n}{8}\right) - n$	
5.	$4\,T\left(\frac{n}{2}\right) + \log n$	n^2
6.	$2\,T\left(\frac{n}{2}\right) + \sqrt{n}$	$O(n)$
7.	$3T\left(\frac{n}{2}\right) + n$	$n^{\log_2 3}$
8.	$3T\left(\frac{n}{3}\right) + \sqrt{n}$	$O(n)$
9.	$4T\left(\frac{n}{2}\right) + c.n$	n^2
10.	$3T\left(\frac{n}{4}\right) + n \log n$	$n\log n$

Additional Problems with Solutions:

https://www.csd.uwo.ca/~mmorenom/CS433-CS9624/Resources/master.pdf