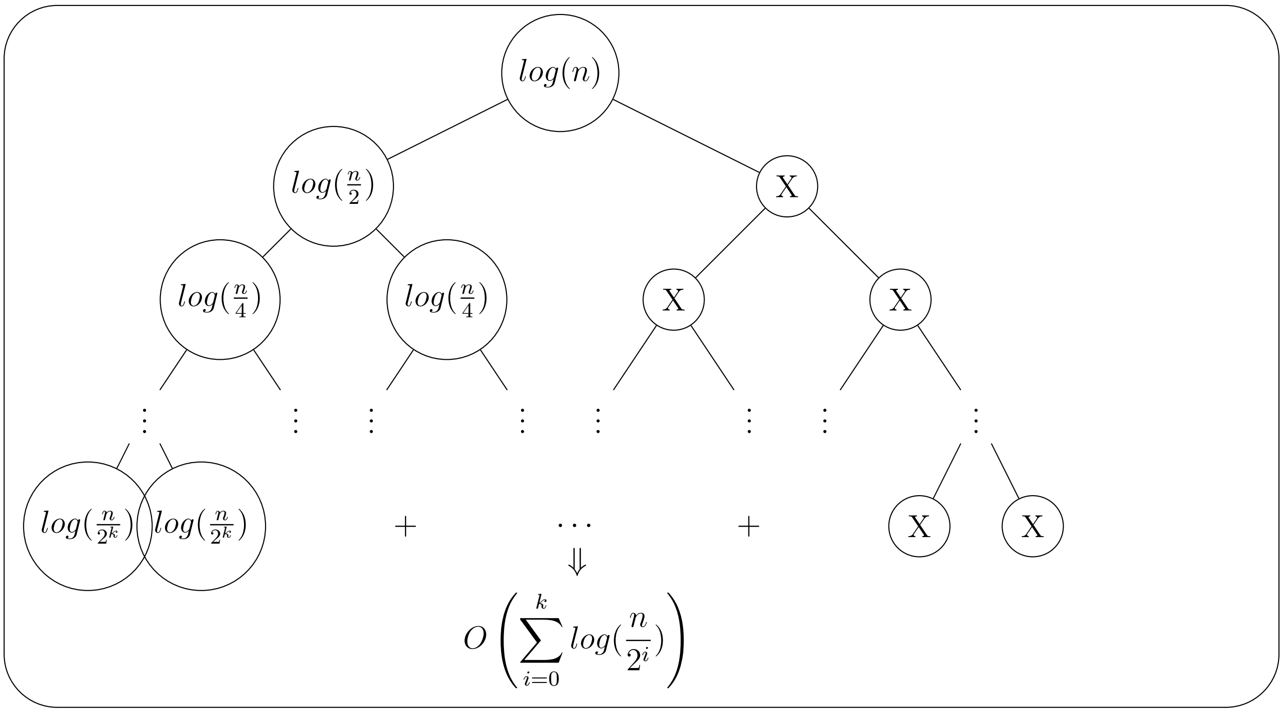
Question 1: $H(n) = H(\frac{n}{2}) + log(n)$

1.1 Recursion Tree



NOTE: I USED A TEMPLATE TO BUILD THIS RECURSION TREE MADE BY Manuel Kirsch, the information however, is original.

Solving $\frac{n}{2^k} = 1$ we get $k = log_2(n)$ and summing over all the levels of the tree we get:

$$\sum_{i=0}^{k} log(\tfrac{n}{2^i})$$
$$log(n) + log(\tfrac{n}{2}) + log(\tfrac{n}{4}) + ... + log(\tfrac{n}{k})$$
$$log(n * \tfrac{n}{2} * \tfrac{n}{4} * ... * \tfrac{n}{k})$$
$$log(n * log_2(n) * (\tfrac{1}{2} * \tfrac{1}{4} * ... * \tfrac{1}{log_2(n)})$$
$$log(n) + log(log(n)) + log(\tfrac{1}{2} * \tfrac{1}{4} * ... * \tfrac{1}{log_2(n)})$$
$$log^2(n) + log(n) + log(\tfrac{1}{2} * \tfrac{1}{4} * ... * \tfrac{1}{log_2(n)})$$

Since $log^2 n$ is largest term, function $H(n)$ is $O(log^2 n)$

1.2 Substitution

$$H(n) = H(n/2) + log(n)$$
$$H(n) = H(n/4) + log(n) + log(\tfrac{n}{2})$$
$$H(n) = H(n/8) + log(n) + log(\tfrac{n}{2}) + log(\tfrac{n}{4})$$
$$= log(n) + log(\tfrac{n}{2}) + log(\tfrac{n}{4}) + ... + log(\tfrac{n}{2^k})$$

$$= \sum_{i=0}^{log_2(n)} log(\frac{n}{2^i})$$

$$= O(log^2(n))$$

1.3 Master Theorem

$$H(n) = aH(\tfrac{n}{b}) + \Theta(N^k log^p N)$$

Let $a = 1$ & $b = 2$ & $p = 1$ then $n^{log_b a} = n^{log_2 1} = n^0 = 1$

Master Theorem says that if $f(n) = \Theta(n^{log_b a})$, then $T(n) = \Theta(n^{log_b a} log^{p+1} n)$

So we have $O(1 * log^{1+1} n) = O(\log^2 n)$

Question 2: Paper Cuts

2.1 Prune and Incinerate

The following function denotes the number of cuts that will be needed to destroy as a function of the area of the picture:

$$C(a) = C(a-1) + 1$$

Since you only snip off a unit area at a time, the total number of cuts will be a unit added to a recursive call on the area minus the unit. Solving the recurrence relation we get (assuming C(1) = 0 since a unit area requires no cuts):

$$C(a) = C(a-1) + 1$$
$$C(a) = C(a-2) + 2$$
$$C(a) = C(a-k) + k \text{ where (a-k) = 1 so k = a - 1}$$
$$C(a) = C(1) + (a-1)$$
$$C(a) = a - 1$$

2.2 Divide and divide again

The following function denotes the number of cuts that will be needed to destroy as a function of the area of the picture:

$$C(a) = 2C(\tfrac{a}{2}) + 1$$

Since we are cutting the picture in half and subsequently each of those two halves into quarters we need two recursive calls to $\frac{a}{2}$ with an added unit denoting the one cut it takes to split a picture into two pieces. Solving the recurrence relation we get:

$$C(a) = 2C(\tfrac{a}{2}) + 1$$
$$C(a) = 4C(\tfrac{a}{4}) + 3$$
$$C(a) = 8C(\tfrac{a}{8}) + 7$$
$$C(a) = kC(\tfrac{a}{k}) + (k-1) \text{ where } \tfrac{a}{k} = 1 \text{ so } k = a$$
$$C(a) = a * 0 + a - 1 = a - 1$$

2.3 Stack them up

The following function denotes the number of cuts that will be needed to destroy as a function of the area of the picture:

$$C(a) = C(\tfrac{a}{2}) + 1$$

Since now we are stacking the cut pieces we only need one recursive call to $\frac{a}{2}$ with an added unit denoting the one cut it took to split the picture into two pieces. Solving the recurrence relation we get:

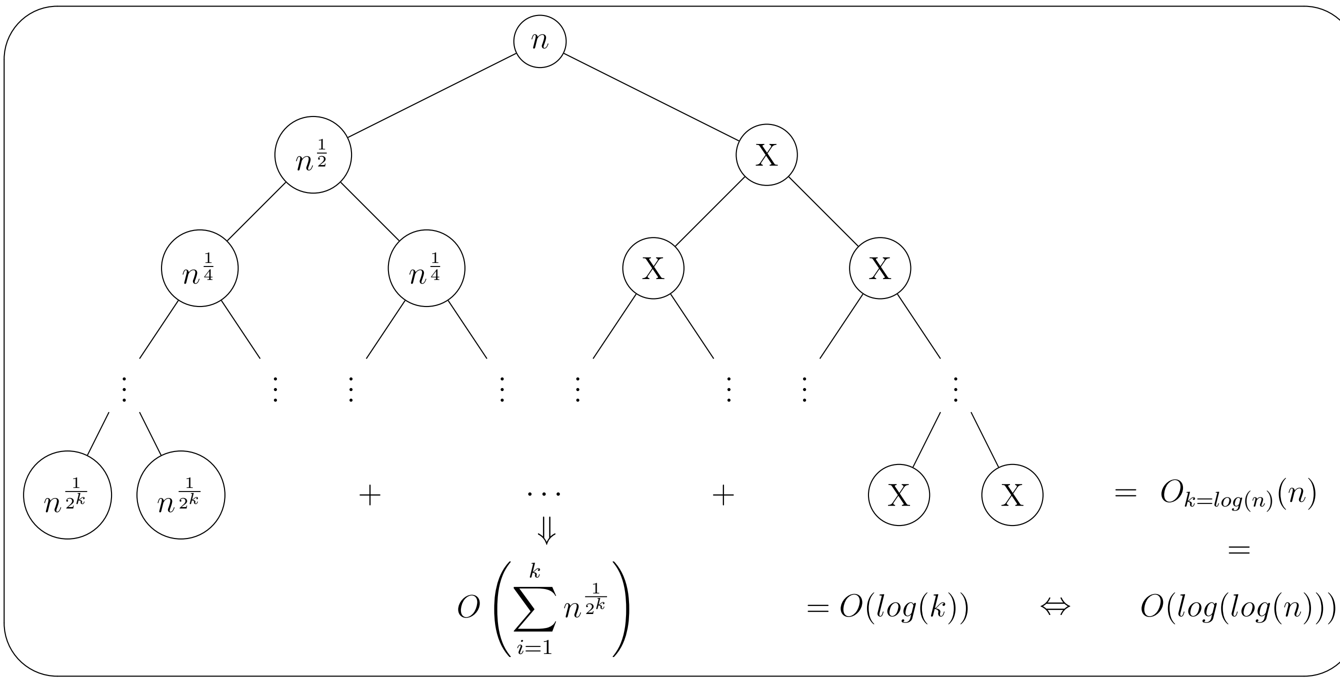
$$\begin{gathered}
C(a) = C(\tfrac{a}{2}) + 1 \\
C(a) = 2C(\tfrac{a}{4}) + 2 \\
C(a) = 3C(\tfrac{a}{8}) + 3 \\
C(a) = kC(\tfrac{a}{2^k}) + k \text{ where } \tfrac{a}{2^k} = 1 \text{ so } 2^k = a \text{ or } k = log_2(a) \\
C(a) = log_2(a)
\end{gathered}$$

2.4 Which one is best?

It is clear from solving the recurrence relations of the functions, the best one would be Stacking them up since the total number of cuts is much less than the other two. Stacking them up has a count of $log_2(a)$ where a is the area of the original photo, whereas the first two have a count of $a - 1$.

Question 3: $T(n) = T(\sqrt{n}) + n$

3.1 Recursion Tree



NOTE: I USED A TEMPLATE TO BUILD THIS RECURSION TREE MADE BY Manuel Kirsch, the information however, is original

3.2 Substitution

$$T(n) = T(\sqrt{n}) + n$$
$$T(n) = T(n^{\frac{1}{2}}) + n$$
$$T(n) = T(n^{\frac{1}{4}}) + 2n$$
$$T(n) = T(n^{\frac{1}{8}}) + 3n$$
$$T(n) = T(n^{\frac{1}{2^k}}) + kn$$

where $n^{\frac{1}{2^k}} = 2$ so $2^{2^k} = n \rightarrow k = log_2(log_2(n))$

Therefore the computation will be $O(log_2(log_2(n)))$

3.3 Master Theorem

Let $n = 2^k$, substituting back into the original $T(n)$ we get:

$$T(2^k) = T(2^{\frac{k}{2}}) + 2^k$$

Let $S(k) = T(2^k)$, then we get:

$$S(k) = S(\tfrac{k}{2}) + k$$

By master theorem we know:

$$S(k) = S(\tfrac{k}{2}) + O(k)$$

Which means that $a = 1$ & $b = 2$ and $f(n) = O(k)$, so by master theorem $T(n) = O(log_2 log_2 n)$