Разбор задачи «Сжатие изображения»

Рассмотрим какое-нибудь сжатие данного изображения. Пусть при этом сжатии изображение было разбито на прямоугольники $a \times b$. Тогда очевидно, что a является делителем n, а b является делителем m.

Научимся проверять, что в некотором прямоугольнике разбиения все пиксели имеют один цвет за время $\mathcal{O}(1)$. Заменим символы «`.`» на нули, а «`X`» — на единицы. Теперь нам нужно проверить, что сумма в прямоугольнике равна либо 0, либо площади этого прямоугольника. Чтобы быстро считать сумму в прямоугольнике, посчитаем аналог сумм на префиксах для прямоугольника. Обозначим за $pref[i][j]$ сумму всех значений, у которых первый индекс не превосходит i, а второй индекс не превосходит j. Тогда сумма в прямоугольнике с углами (x_1, y_1) и (x_2, y_2) равна $pref[x2][y2] - pref[x1-1][y2] - pref[x2][y1-1] + pref[x2-1][y2-1]$. С помощью этого довольно классического приема мы научились проверять, что все символы в прямоугольнике одинаковые.

Теперь переберем в качестве значений a и b все делители чисел n и m соответственно, и для каждой пары (a, b) проверим, что такие значения подходят за время $\mathcal{O}(\frac{n}{a} \cdot \frac{m}{b})$. Среди всех подходящих значений выберем значение, для которого $\frac{nm}{ab}$ минимально.

Оценим время работы получившегося алгоритма. Заметим, что время работы алгоритма равняется сумме по всем парам (a, b), где a — делитель n, а b — делитель m, величин $\frac{nm}{ab}$. Очевидно, эта сумма не превосходит суммы тех же самых величин по всем парам (a, b), где $1 \leqslant a \leqslant n, 1 \leqslant b \leqslant m$ (так как эта сумма содержит все слагаемые исходной суммы). Таким образом, время работы алгоритма не превышает величины $\sum\limits_{a=1}^{n} \sum\limits_{b=1}^{m} \frac{a}{n} \frac{b}{m} = \sum\limits_{a=1}^{n} \frac{a}{n} \cdot (\sum\limits_{b=1}^{m} \frac{b}{m})$.

Довольно известный факт, что $\sum\limits_{i=1}^{n} \frac{1}{i} = \mathcal{O}(\log n)$. Умножая это равенство на n, получаем, что $\sum\limits_{i=1}^{n} \frac{n}{i} = \mathcal{O}(n \log n)$.

Значит, наша искомая сумма равна $\sum\limits_{a=1}^{n} \frac{a}{n} \cdot \mathcal{O}(m \log m)$. Применив ещё раз этот же факт, получим, что время работы алгоритма составляет $\mathcal{O}(nm \log n \log m)$.