АМВ. ДЗ на неделю 1.

Прохоров Юрий, 771

Задача 1

Пусть A_n — число натуральных решений уравнения

$$2x + 3y = n.$$

Найти производящую функцию последовательности A_n, Θ-асимптотику A_n и явное выражение для A_n.

Решение:
Решим сначала уравнение алгоритмом Евклида. Находим частные решения уравнений:

$$2x + 3y = 1 \qquad \Longrightarrow \qquad \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

$$2x + 3y = n \qquad \Longrightarrow \qquad \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} -n \\ n \end{pmatrix}$$

Общее решение:

$$x = 3k - n, \qquad y = n - 2k, \qquad k \in \mathbb{Z}$$

Решения должны быть натуральными, значит:

$$\begin{cases} x > 0 \\ y > 0 \end{cases} \iff \begin{cases} 3k - n > 0 \\ n - 2k > 0 \end{cases} \iff \frac{n}{3} < k < \frac{n}{2}$$

Количество различных целых значений k и есть **явное выражение** для A_n:

$$A_n = \left\lceil \frac{n}{2} \right\rceil - \left\lfloor \frac{n}{3} \right\rfloor - 1$$

Отсюда следует, что **асимптотика**:

$$A_n = \Theta(n)$$

Покажем, что выполняется рекуррентное соотношение

$$A_{n+6} = \left\lceil \frac{n+6}{2} \right\rceil - \left\lfloor \frac{n+6}{3} \right\rfloor - 1 = \left\lceil \frac{n}{2} \right\rceil - \left\lfloor \frac{n}{3} \right\rfloor = A_n + 1, \qquad n \geq 1$$

Начальные условия (добавим $A_0 = 0$, чтобы A_n соответствовало x^n):

$$A_0 = A_1 = A_2 = A_3 = A_4 = 0, \qquad A_5 = 1, \qquad A_6 = 0.$$

Найдем **производящую функцию** $F(x)$:

$$F(x) = \sum_{n=0}^{\infty} A_n x^n = x^5 + \sum_{n=7}^{\infty} A_n x^n = x^5 + x^6 \sum_{n=1}^{\infty} A_{n+6} x^n = x^5 + x^6 \sum_{n=1}^{\infty} (A_n + 1) x^n =$$

$$= x^5 + x^6 F(x) + x^6 \left(\sum_{n=0}^{\infty} x^n - 1 \right) = x^5 + x^6 F(x) + x^6 \frac{x}{1-x}$$

$$\ldots$$

$$F(x) = \frac{x^5}{x^5 - x^3 - x^2 + 1}$$

Задача 2

Пожалуй, самый известный алгоритм, о котором все слышали,— это *алгоритм Евклида* для подсчета наибольшего общего делителя $\gcd(x, y)$ двух натуральных чисел ($x > y$). Вычисление ведется рекурсивно: если $y = 0$, то возвращается x, если $y = 1$, то возвращается 1, а иначе вызывается $\gcd(y, x \mod y)$.

На каждой итерации по крайней мере одно число уменьшается, поэтому процедура конечна. Более того, понадобится не более $O(|x|_{unary})$ итераций. В частности, если $x = 2^{200}$, то оценка превышает число протонов во вселенной, т. е. практически бессмысленна. Если бы удалось получить оценку вида $O([|x|_{binary}]^{O(1)})$ (как говорят, "полиномиальную" по длине [*двоичной*] записи), то это было бы гораздо более убедительным свидетельством эффективности алгоритма.

Попробуем получить более точную оценку трудоемкости. Пусть для $1 \le i \le m$ x_i и, соответственно, y_i обозначают значения параметров x и y на i-й итерации алгоритма (например, $x_1 = x$, $y_1 = y$). Также положим $s_i = x_i + y_i$.

(i) Покажите, что $s_i \le 2/3 \cdot s_{i-1}$.
(ii) Вычислите $\gcd(F_{m+2}, F_{m+1})$, где F_n — это n-е число Фибоначчи.

Решение:
Пусть $x_{i-1} > y_{i-1} > 1$. Тогда

$$\gcd(x_{i-1}, y_{i-1}) = \gcd(y_{i-1}, x_{i-1} \mod y_{i-1}) \qquad \Longrightarrow \qquad x_i = y_{i-1},\ y_i = x_{i-1} \mod y_{i-1}$$

Обозначим $x = x_{i-1}, y = y_{i-1}$.

$$\frac{s_i}{s_{i-1}} = \frac{x_i + y_i}{x_{i-1} + y_{i-1}} = \frac{y + (x \mod y)}{x + y}$$

Представим

$$x = ky + c, \qquad k \ge 1, \qquad 0 \le c \le y - 1.$$

Тогда (делаем оценку сверху: сначала $k \to \min$, потом $c \to \max$):

$$\frac{s_i}{s_{i-1}} = \frac{y + c}{ky + c + y} = \frac{y + c}{(k+1)y + c} \le \frac{y + c}{2y + c} = 1 - \frac{y}{2y + c} \le 1 - \frac{y}{2y + y - 1} < 1 - \frac{y}{3y} = \frac{2}{3}$$

Это и требовалось доказать.

Заметим, что

$$\gcd(F_{m+2}, F_{m+1}) = \gcd(F_{m+1} + F_m, F_{m+1}) = \gcd(F_{m+1}, F_m), \qquad \forall m \ge 1$$

Тогда

$$\gcd(F_{m+2}, F_{m+1}) = \gcd(F_1, F_0) = \gcd(2, 1) = 1$$

Задача 3

Найдите Θ-асимптотику рекуррентности, которая определяется в следующем тексте.

Colour the edges of a complete graph of n vertices by three colours so that the number of triangles all whose edges get a different colour is maximal. Denote this maximum by $G_3(k)$. They conjectured that $G_3(k)$ is obtained as follows: clearly $G_3(1) = G_3(2) = 0, G_3(3) = 1, G_3(4) = 4$. Suppose $G_3(k_1)$ has already been determined for every $k_1 < k$. Then

$$G_3(k) = G_3(u_1) + G_3(u_2) + G_3(u_3) + G_3(u_4) + u_1u_2u_3 + u_1u_2u_4 + u_1u_3u_4 + u_2u_3u_4,$$

where $u_1 + u_2 + u_3 + u_4 = k$ and the u's are as nearly equal as possible.

Решение:
Будем считать, что условие, что u_i должны быть как можно ближе друг к другу означает, что

$$\left\lfloor \frac{k}{4} \right\rfloor \le u_i \le \left\lceil \frac{k}{4} \right\rceil, \qquad i = 1, 2, 3, 4$$

Из смысла исходной задачи следует, что при увеличении или уменьшении числа ребер на единицу, ответ не изменится кардинально. $G(k)$ может и не быть монотонной, но асимптотика рекурренты

$$G(k) = 4G\left(\left\lfloor\frac{k}{4}\right\rfloor\right) + 4\left(\left\lfloor\frac{k}{4}\right\rfloor\right)^3 = 4G\left(\frac{k}{4}\right) + Ck^3$$

будет такой же (можно считать k степенью 4).

$$G(k) = 4G\left(\frac{k}{4}\right) + Ck^3 = Ck^3 + C\frac{k^3}{16} + C\frac{k^3}{16^2} + \ldots = Ck^3 \sum_{s=0}^{\sim\log k} \frac{1}{16^s} = Ck^3 \frac{1 - \frac{1}{16^{\log k}}}{1 - \frac{1}{16}} = \Theta(k^3)$$

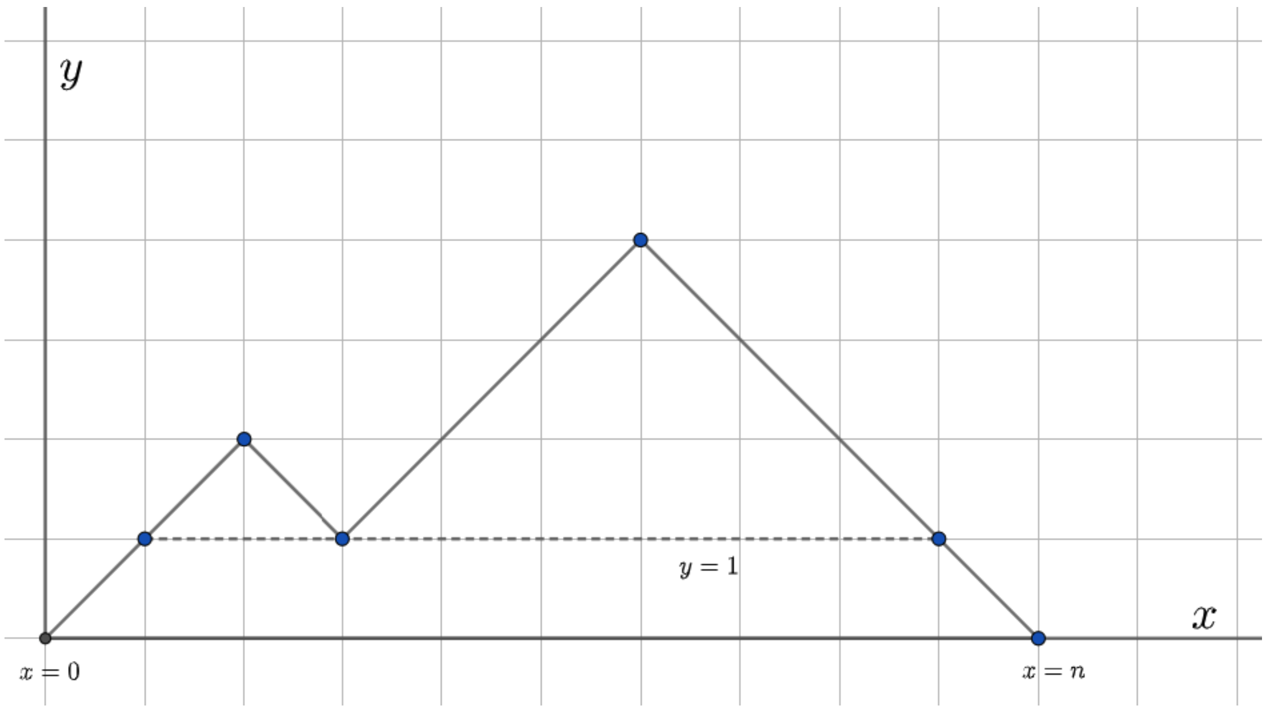
Задача 4

(i) Вычислите число правильно составленных скобочных выражений, содержащих n скобок, в которых в любом непустом префиксе число открывающих скобок больше числа закрывающих.

(ii) Найдите явное аналитическое выражение для производящей функции чисел BR_{4n+2} правильных скобочных последовательностей длины $4n+2$ (ответ в виде суммы ряда не принимается).

Решение:

(i) Представим правильные скобочные последовательности длины n как случайные блуждания на отрезок длины n такие, что мы не можем опускаться ниже координаты $y = 0$, как мы делали это в первом семестре. Тогда условие того, что в любом непустом префиксе скобочный итог строго положителен равносильно тому, что наши случайные блуждания не опускаются ниже $y = 1$ при $1 \le x \le n - 1$.



Сразу будем считать, что n — четное, потому что длина любой правильной скобочной последовательности четная.

Обрежем график случайных блужданий по $y = 1$. Получим привычные случайные блуждания, на отрезке длины $n - 2$. Мы знаем, что количество таких случайных блужданий A_n равно числу Каталана

$$A_n = C_{(n-2)/2} = C_{\frac{n}{2}-1}$$

(ii) Как было замечено в предыдущей задаче, число правильных скобочных последовательностей выражается через число Каталана:

$$BR_{4n+2} = C_{2n+1}, \qquad n = 0, 1, 2, \ldots$$

Производящая функция этой последовательности:

$$F(x) = \sum_{n=0}^{\infty} C_{2n+1} x^n.$$

Будем, считать, что мы умеем находить производящую функцию последовательности чисел Каталана (мы делали это в первом семестре):

$$A(x) = \sum_{n=0}^{\infty} C_n x^n = C_0 + C_1 x + C_2 x^2 + C_3 x^3 + \ldots = \frac{1 - \sqrt{1-4x}}{2x}$$

Тогда

$$\begin{aligned}
xA(x) &= C_0 x + C_1 x^2 + C_2 x^3 + C_3 x^4 + \ldots \\
-xA(-x) &= -C_0 x + C_1 x^2 - C_2 x^3 + C_3 x^4 - \ldots \\
x\big(A(x) - A(-x)\big) &= 2C_1 x^2 + 2C_3 x^4 + 2C_5 x^6 + \ldots \\
\frac{\sqrt{x}\big[A(\sqrt{x}) - A(-\sqrt{x})\big]}{2} &= C_1 x + C_3 x^2 + C_5 x^3 + \ldots = F(x)
\end{aligned}$$

Последнее преобразование корректно, потому что производящая функция является формальным степенным рядом, и его радиус сходимости может равняться нулю (как и есть в этом случае).

$$F(x) = \frac{2 - \sqrt{1 - 4\sqrt{x}} - \sqrt{1 + 4\sqrt{x}}}{4\sqrt{x}}$$

Задача 5

Оцените трудоемкость рекурсивного алгоритма, разбивающего исходную задачу размера n на три задачи размером $\lceil \frac{n}{\sqrt{3}} \rceil - 5$, используя для этого $10\frac{n^3}{\log n}$ операций.

Решение:

Рекуррентное соотношение для такого алгоритма:

$$T(n) = 3T\Big(\Big\lceil \frac{n}{\sqrt{3}} \Big\rceil - 5\Big) + 10\frac{n^3}{\log n}$$

Нижняя оценка:

$$T(n) \geq 10\frac{n^3}{\log n} \qquad \Longrightarrow \qquad T(n) = \Omega\Big(\frac{n^3}{\log n}\Big)$$

Верхнюю оценку будем доказывать по индукции по n. База индукции состоит в том, что при малых n (например, при $n < 15$) время работы алгоритма $T(n) = c_0 = const$.

Предположение: пусть при всех k меньше данного n выполнено для некоторой константы $C > c_0$:

$$T(k) \leq C\frac{k^3}{\log k}, \qquad k < n, \qquad C > c_0$$

Тогда

$$\begin{aligned}
T(n) = 3T\Big(\Big\lceil \frac{n}{\sqrt{3}} \Big\rceil - 5\Big) + 10\frac{n^3}{\log n} &\leq 3C\frac{\Big(\Big\lceil \frac{n}{\sqrt{3}} \Big\rceil - 5\Big)^3}{\log\Big(\Big\lceil \frac{n}{\sqrt{3}} \Big\rceil - 5\Big)} + 10\frac{n^3}{\log n} \leq 3C\frac{(\frac{2n}{3})^3}{\log\frac{n}{2}} + 10\frac{n^3}{\log n} = \\
&= \frac{8C}{9}\frac{n^3}{\log n - \log 2} + 10\frac{n^3}{\log n} \leq C\frac{n^3}{\log n}\Big(\frac{8}{9}\frac{\log n}{\log n - \log 2} + \frac{10}{C}\Big) \leq C\frac{n^3}{\log n}
\end{aligned}$$

Чтобы выполнялось последнее неравенство необходимо, чтобы при $n \geq n_0$ для некоторого n_0:

$$\frac{8}{9}\frac{\log n}{\log n - \log 2} + \frac{10}{C} < 1$$

При больших n:

$$\frac{\log n}{\log n - \log 2} \le \frac{11}{10} \qquad \Longrightarrow$$

$$\frac{10}{C} < 1 - \frac{8}{9}\frac{\log n}{\log n - \log 2} \ge 1 - \frac{8}{9}\cdot\frac{11}{10} = \frac{1}{45} \qquad \Longrightarrow \qquad C > 450$$

Итак, существует такое $C > 450$, что наше доказательство по индукции корректно. Тогда

$$T(n) = O\Big(\frac{n^3}{\log n}\Big) \qquad \Longrightarrow \qquad T(n) = \Theta\Big(\frac{n^3}{\log n}\Big)$$

Задача 6

Рассмотрим детерминированный алгоритм поиска медианы по кальке известного линейного алгоритма, где используется разбиение массива на четвёрки элементов, в каждой из которых определяется *нижняя* медиана, т. е. из в каждой четверки выбирается второй по порядку элемент (элементы можно считать различными). Приведите рекуррентную оценку числа сравнений в этой процедуре и оцените сложность такой модификации.

Решение:

Будем действовать аналогично тому, как это сделано в конспекте семинара.

Пусть мы делим подаваемый на вход массив на группы по 4 элемента. Отсортируем эти $\lceil \frac{n}{4} \rceil$ групп и выделим в них медианы. В массиве медиан рекурсивным вызовом найдем медиану, то есть серединную порядковую статистику d. Перебирая все элементы в исходном массиве, будем процедурой $PARTITION$ левее d ставить все элементы, меньшие d, а правее — все остальные.

Слева от d окажется по 2 элемента из половины групп, за исключением, может быть, самой группы и последней. Значит, число элементов, которой окажется по одну сторону от d

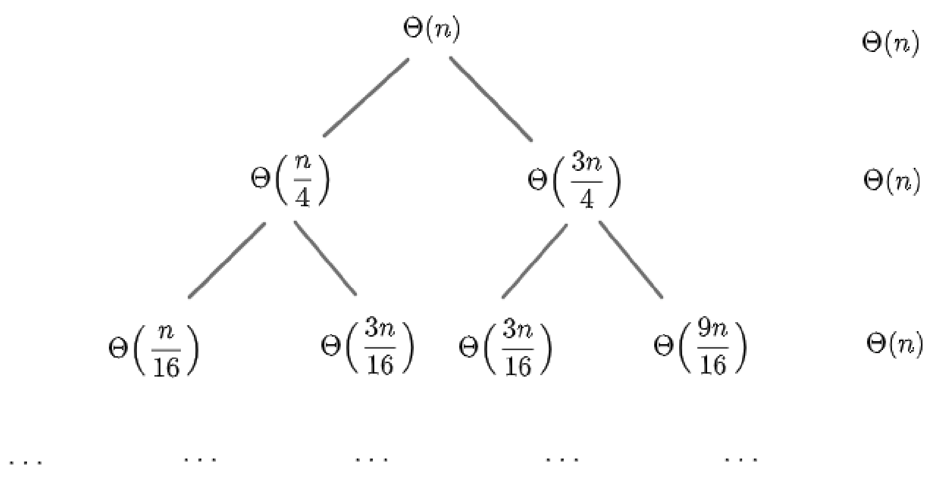
$$k \ge 2\big(\frac{1}{2}\cdot\frac{n}{4} - 2\big) = \frac{n}{4} - 4$$

Следовательно, по одну сторону может оказаться не более

$$n - k \le \frac{3n}{4} + 4$$

элементов. От этого числа в худшем случае будет делаться рекурсивный вызов. Поэтому мы имеем следующую рекурренту:

$$T(n) = T\Big(\frac{n}{4}\Big) + T\Big(\frac{3n}{4} + 4\Big) + \Theta(n)$$



Член порядка $\Theta(n)$ появляется из-за сравнений и сортировки групп. Распишем дерево рекурсии. Его глубина будет равна

$$h = \log_{4/3} n$$

На каждом уровне общая сумма действий равна $\Theta(n)$. Тогда итоговая асимптотика:

$$T(n) = \Theta(n \log n)$$

Задача 7

Функция натурального аргумента $S(n)$ задана рекурсией:

$$S(n) = \begin{cases} 100 & , \ n \leq 100 \\ S(n-1) + S(n-3) & , \ n > 100 \end{cases}$$

Оцените число рекурсивных вызовов процедуры $S(\cdot)$ при вычислении $S(10^{12})$.

Решение:

Запишем рекурренту, которая будет показывать число рекурсивных вызовов. Ради простоты получения оценки будем считать, что рекуррента принимает константное значения при $n \leq 2$:

$$T(n) = \begin{cases} 1 & , \ n \leq 2 \\ T(n-1) + T(n-3) + 1 & , \ n > 2 \end{cases}$$

Допустим, мы смогли найти функцию $f(n)$ такую, что

$$f(n) = f(n-1) + f(n-3), \qquad f(0) = f(1) = f(2) = 1$$

Тогда

$$T(n) = f(n) - 1.$$

Решим однородное соотношение. Записываем характеристическое уравнение:

$$\lambda^3 = \lambda^2 + 1$$

$$\lambda_1 \approx 1.46, \qquad \lambda_{2,3} \approx -0.23 \pm 0.79i = r(\cos\phi \pm i \sin\phi)$$

$$r \approx 0.83, \qquad \phi \approx -0.59\pi$$

$$f(n) = C_1\lambda_1^n + C_2\lambda_2^n + C_3\lambda_3^n = C_1 1.46^n + C_2 r^n(\cos n\phi + i \sin n\phi) + C_3 r^n(\cos n\phi - i \sin n\phi)$$

$$f(n) \approx C_1 1.46^n + 2C_2 r^n \cos n\phi$$

Так как значение должно быть действительным, то $C_2 = C_3$.

Из начальных условий найдем приближенное решение:

$$\begin{cases} C_1 + 2C_2 = 1 \\ 1.46C_1 + 2 \cdot 0.83 C_2 \cos(0.59\pi) = 1 \end{cases} \iff \begin{cases} C_1 \approx 0.626 \\ C_2 \approx 0.187 \end{cases}$$

$$T(n) \approx 0.63 \cdot 1.46^n, \qquad \text{при больших } n,$$

так как $r < 1$. При $n = 10^{12}$ такая оценка справедлива.

Задача 8

Оцените как можно точнее высоту дерева рекурсии для рекуррентности

$$T(n) = T(n - \lfloor\sqrt{n}\rfloor) + T(\lfloor\sqrt{n}\rfloor) + \Theta(n)$$

Решение:

Заметим, что $n - \sqrt{n} \geq \sqrt{n}$ при $n \geq 4$. Поэтому аргумент левого рекурсивого вызова всегда будет больше, чем у правого. То есть на каждом шаге дерево рекурсии будет ветвиться, но левая ветка всегда будет уходить глубже, поэтому с точки зрения оценки глубины дерева рекурсии данная задача эквивалентна следующей:

$$T(n) = T(n - \lfloor \sqrt{n} \rfloor)$$

Будем раскрывать рекурсию, воспользовавшись формулой Тейлора:

$$n - \sqrt{n} - \sqrt{n - \sqrt{n}} = n - \sqrt{n} - \sqrt{n}\Big(1 - \frac{1}{\sqrt{n}}\Big)^{\frac{1}{2}} = n - \sqrt{n} - \sqrt{n}(1 - \frac{1}{2\sqrt{n}} + O\Big(\frac{1}{n}\Big)) = n - 2\sqrt{n} + \frac{1}{2} + O\Big(\frac{1}{\sqrt{n}}\Big)$$

$$n - \sqrt{n} - \sqrt{n - \sqrt{n}} \approx n - 2\sqrt{n} + \frac{1}{2}$$

Следующая итерация (аналогично):

$$n - 2\sqrt{n} + \frac{1}{2} - \sqrt{n - 2\sqrt{n} + \frac{1}{2}} \approx n - 3\sqrt{x} + 1 + \frac{1}{2}$$

k-ая итерация:

$$n - k\sqrt{n} + \frac{1 + 2 + \ldots + (k-1)}{2} - \sqrt{n - k\sqrt{n} + \frac{k(k-1)}{4}} \approx n - (k+1)\sqrt{n} + \frac{k(k+1)}{4}$$

Найдем k, при котором данная оценка примерно равна нулю:

$$n - k\sqrt{n} + \frac{k^2}{4} \approx 0$$

$$k \approx 2\sqrt{n}$$

Итоговая оценка высоты дерева рекурсии равна числу рекурсивных вызовов k:

$$h(n) = \Theta(\sqrt{n})$$

Задача 9

Оцените трудоемкость рекурсивного алгоритма, разбивающего исходную задачу размера n на n задач размеров $\lceil \frac{n}{2} \rceil$ каждая, используя для этого $O(n)$ операций.

Решение:

Считаем, что алгоритм примерно является монотонным по n, поэтому рассмотрим случай $n = 2^m$.

$$T(n) = nT\Big(\Big\lfloor \frac{n}{2} \Big\rfloor\Big) + O(n)$$

При $n = 2^k$:

$$T(n) = nT\Big(\frac{n}{2}\Big) + cn = cn + c\frac{n^2}{2} + c\frac{n^3}{4} + \ldots = c\sum_{k=1}^{\log_2 n} \frac{n^k}{2^{k-1}} = 2c\sum_{k=1}^{\log_2 n} \Big(\frac{n}{2}\Big)^k = 2c \cdot \frac{n}{2} \frac{\Big(\frac{n}{2}\Big)^{\log_2 n} - 1}{\frac{n}{2} - 1} =$$

$$= \Theta\Big(\Big(\frac{n}{2}\Big)^{\log_2 n}\Big) = \Theta\big(n^{\log_2 n - 1}\big)$$