рассуждать следующим образом. Сортировка одного элемента методом слияния длится в течение фиксированного времени. Если $n > 1$, время работы распределяется таким образом.

Разбиение. В ходе разбиения определяется, где находится средина подмассива. Эта операция длится фиксированное время, поэтому $D(n) = \Theta(1)$.

Покорение. Рекурсивно решаются две подзадачи, объем каждой из которых составляет $n/2$. Время решения этих подзадач равно $2T(n/2)$.

Комбинирование. Как уже упоминалось, процедура MERGE в n-элементном подмассиве выполняется в течение времени $\Theta(n)$, поэтому $C(n) = \Theta(n)$.

Сложив функции $D(n)$ и $C(n)$, получим сумму величин $\Theta(n)$ и $\Theta(1)$, которая является линейной функцией от n, т.е. $\Theta(n)$. Прибавляя к этой величине слагаемое $2T(n/2)$, соответствующее этапу "покорения", получим рекуррентное соотношение для времени работы $T(n)$ алгоритма сортировки по методу слияния в наихудшем случае:

$$T(n) = \begin{cases} \Theta(1) & \text{при } n = 1, \\ 2T(n/2) + \Theta(n) & \text{при } n > 1. \end{cases} \tag{2.1}$$

В главе 4 мы ознакомимся с теоремой, с помощью которой можно показать, что величина $T(n)$ представляет собой $\Theta(n \lg n)$, где $\lg n$ обозначает $\lg_2 n$. Поскольку логарифмическая функция растет медленнее, чем линейная, то для достаточно большого количества входных элементов производительность алгоритма сортировки методом слияния, время работы которого равно $\Theta(n \lg n)$, превзойдет производительность алгоритма сортировки методом вставок, время работы которого в наихудшем случае равно $\Theta(n^2)$.

Правда, можно и без упомянутой теоремы интуитивно понять, что решением рекуррентного соотношения (2.1) является выражение $T(n) = \Theta(n \lg n)$. Перепишем уравнение (2.1) в таком виде:

$$T(n) = \begin{cases} c & \text{при } n = 1, \\ 2T(n/2) + cn & \text{при } n > 1, \end{cases} \tag{2.2}$$

где константа c обозначает время, которое требуется для решения задачи? размер который равен 1, а также удельное (приходящееся на один элемент) время, требуемое для разделения и сочетания[9].

[9]Маловероятно, чтобы одна и та же константа представляла и время, необходимое для решения задачи, размер который равен 1, и приходящееся на один элемент время, в течение которого выполняются этапы разбиения и объединения. Чтобы обойти эту проблему, достаточно предположить, что c — максимальный из перечисленных промежутков времени. В таком случае мы получим верхнюю границу времени работы алгоритма. Если же в качестве c выбрать наименьший из всех перечисленных промежутков времени, то в результате решения рекуррентного соотношения получим нижнюю границу времени работы алгоритма. Принимая во внимание, что обе границы имеют порядок $n \lg n$, делаем вывод, что время работы алгоритма ведет себя, как $\Theta(n \lg n)$.

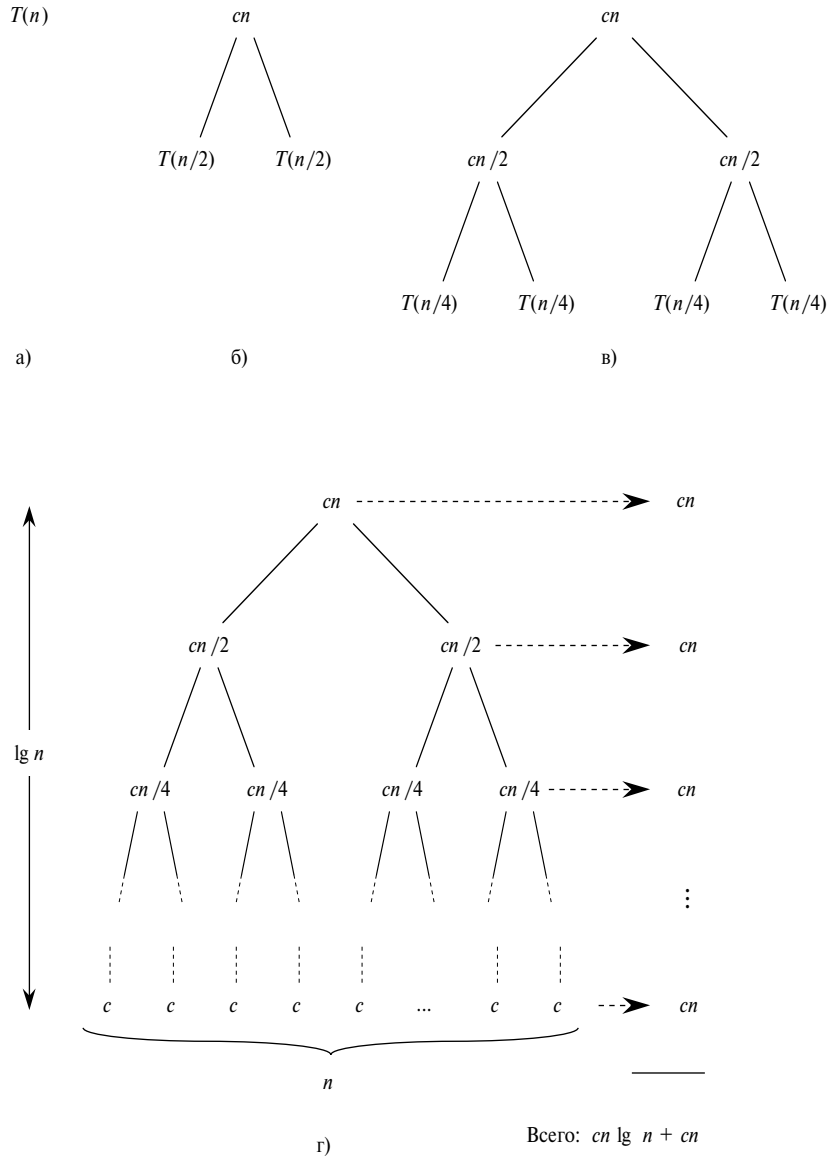

Рис. 2.5. Построение дерева рекурсии для уравнения $T(n) = 2T(n/2) + cn$

Процесс решения рекуррентного соотношения (2.2) проиллюстрирован на рис. 2.5. Для удобства предположим, что n равно степени двойки. В части *а* упомянутого рисунка показано время $T(n)$, представленное в части *б* в виде эквивалентного дерева, которое представляет рекуррентное уравнение. Корнем этого дерева является слагаемое cn (стоимость верхнего уровня рекурсии), а два