Journal of Siberian Federal University. Mathematics & Physics 2008, 1(3), 236-246

УДК 519.6

Математические методы анализа рекурсивных алгоритмов

Валентина В.Быкова*
Институт математики,
Сибирский федеральный университет,
Свободный 79, Красноярск, 660041,
Россия

Получена 05.03.2008, окончательный вариант 10.04.2008, принята 05.05.2008

Доказана теорема, определяющая асимптотические оценки решения рекуррентного соотношения, характерного для функций временной сложности рекурсивных алгоритмов с аддитивным уменьшением размерности задачи. Представленные результаты вместе с известной основной теоремой о рекуррентных соотношениях дают математический инструмент анализа сложности двух наиболее типичных принципов организации рекурсии.

Ключевые слова: сложность алгоритмов, рекурсия, рекуррентные соотношения

1. Рекурсия и итерация

В математике и программировании рекурсия – это метод определения или выражения функции или процедуры посредством той же функции и процедуры. Рекурсию обычно рассматривают в качестве антипода итерации. Соответственно различают два больших класса алгоритмов: итерационные и рекурсивные. В основе итерационных алгоритмов лежит итерация – многократное повторение одних и тех же действий. Структура таких алгоритмов хорошо описывается алгоритмическими конструкциями “следование”, “ветвление”, “цикл”. Анализ сложности итерационных алгоритмов сводится к определению трудоемкости этих конструкций и формированию интегральной асимптотической оценки с использованием правил суммы и произведения [1].

Рекурсивный алгоритм — это алгоритм, определяемый через себя. В основе рекурсивных алгоритмов лежит рекурсия. Это тоже повторение, но повторение целого в его части. Необходимость применения рекурсивных алгоритмов в одних случаях диктуется самой формулировкой задач – задач, рекурсивных по своей сути, в других – они возникают как удобный метод решения. Рекурсия в сравнении с итерацией имеет ряд преимуществ. Однако практика разработки и использования алгоритмов выдвигает ряд серьезных причин, препятствующих широкому применению рекурсии:

- рекурсивные алгоритмы, как правило, более затратные с точки зрения времени и памяти, нежели итерационные алгоритмы, решающие ту же задачу. При этом на сложность рекурсивного алгоритма большое влияние оказывает сама организация рекурсии;
- анализ сложности рекурсивных алгоритмов — одна из наиболее сложных и до конца нерешенных проблем метрической теории алгоритмов. *Поэтому всякий математический результат, дающий какой-либо общий подход решения проблемы анализа рекурсивных алгоритмов, интересен как теоретически, так и практически.*

По способу взаимодействия целого и его частей различают прямую, косвенную и кратную рекурсии. Все представленные далее результаты и выводы относятся к прямой рекурсии.

2. Методы и проблемы анализа сложности рекурсивных алгоритмов

Основными современными инструментами исследования сложности рекурсивных алгоритмов являются метод рекуррентных соотношений и теоретико-графовый метод исследования дерева рекурсии. Идея метода рекуррентных соотношений состоит в построении и решении рекуррентного соотношения, которому удовлетворяет функция сложности $t(n)$ алгоритма. Найденное решение позволяет получить O-оценку для $t(n)$. Этот метод не является универсальным. Во-первых, он применим только для оценки временной сложности рекурсивных алгоритмов. Оценка емкостной сложности требует учета глубины рекурсии и, как следствие, других инструментов. Во-вторых, метод рекуррентных соотношений позволяет получить лишь верхние оценки временной сложности рекурсивных алгоритмов, т.е. только для худшего случая. В-третьих, рекуррентное соотношение для $t(n)$ удается найти только тогда, когда преобразование, уменьшающее значение параметра рекурсии n, линейно относительно n.

Если рекуррентное соотношение для $t(n)$ все же найдено, то нет никакой гарантии, что удастся установить явную форму или хотя бы асимптотическую оценку для $t(n)$. Проблема в том, что общих методов решения рекуррентных соотношений нет. Известны лишь методы решения некоторых классов рекуррентных соотношений. Тем не менее, во многих практических ситуациях выход находится. Так, допустимо использование общих приемов решения линейных рекуррентных соотношений с постоянными коэффициентами, аппарата производящих функций и интегральных методов [2]–[5]. Наконец, относительно хорошо решаются рекуррентные соотношения, характерные для прямой рекурсии, организованной по принципу “разделяй и властвуй” и аддитивным уменьшением размерности задачи на некоторую константу. Здесь имеются две основные теоремы о рекуррентных соотношениях, одна из которых доказывается в настоящей статье. *Данные теоремы — удобный математический инструмент анализа сложности двух наиболее типичных принципов организации рекурсии, применяемых в практике разработки алгоритмов и программ. Их использование позволяет избежать утомительных расчетов и выбрать наименее трудоемкую схему организации рекурсии.*

В тех случаях, когда не удается получить рекуррентное соотношение для функции временной сложности, прибегают к построению дерева рекурсии [6], [7]. Кроме того, дерево рекурсии незаменимо для анализа емкостной сложности рекурсивного алгоритма.

3. Рекуррентные соотношения, характерные для двух основных принципов организации рекурсии

При определении рекуррентного соотношения, которому удовлетворяет функция временной сложности рекурсивного алгоритма, учитывается трудоемкость обеих его ветвей: нерекурсивная ветвь задает начальные условия, а рекурсивная — собственно само рекур-

рентное соотношение. В самом общем виде значения функции сложности рекурсивного алгоритма вычисляются по формуле:

$$t(n) = \begin{cases} c, & \text{если } 0 \leqslant n \leqslant n_0, \\ t_s + t_r + t_u, & \text{если } n > n_0 \geqslant 0, \end{cases} \tag{1}$$

где n — параметр рекурсии; n_0 — размер задачи, при котором время работы алгоритма не зависит от n; $c \geqslant 0$ — трудоемкость нерекурсивной ветви (некоторая постоянная величина); t_s — время сведения задачи к подзадачам; t_r — трудоемкость рекурсивной ветви (время выполнения подзадач); t_u — время объединения решений, полученных подзадачами.

При анализе сложности рекурсивных алгоритмов традиционно полагают, что $t(n)$ — монотонно возрастающая функция, областью значений которой выступает множество неотрицательных действительных чисел, а областью определения — множество неотрицательных целых чисел $\mathbb{Z}_+$. Эти условия вытекают из природы $t(n)$. Далее мы будем оставаться в их границах.

При использовании принципа "разделяй и властвуй" задача размера n разбивается на подзадачи размера n/k, где $k > 1$ — некоторая целая константа. В этом случае соотношение (1) принимает более конкретный рекуррентный вид:

$$t(n) = \begin{cases} c, & \text{если } 0 \leqslant n \leqslant n_0, \\ A(n)\, t\,(n/k) + B(n), & \text{если } n > n_0 \geqslant 0. \end{cases} \tag{2}$$

Здесь $A(n)$, $B(n)$ — неотрицательные, монотонно возрастающие, вещественнозначные функции от $n \in \mathbb{Z}_+$, характеризующие затраты на рекурсивный переход. Поскольку k — постоянная величина, то преобразование $g(n) = n/k$ линейно относительно n. Очевидно, что соотношение (2) однозначно определяет функцию $t(n)$ только при $n = 0$ и $n = k^m$, $m \in \mathbb{Z}_+$. Для рекурсивных алгоритмов, организованных путем аддитивного уменьшения исходного размера задачи n на целую константу $k \geqslant 1$, соотношение (1) приводится к виду:

$$t(n) = \begin{cases} c, & \text{если } 0 \leqslant n \leqslant n_0, \\ A(n)\, t\,(n-k) + B(n), & \text{если } n > n_0 \geqslant 0, \end{cases} \tag{3}$$

где $A(n)$, $B(n)$ имеют аналогичный смысл, что и в соотношении (2). В данном случае функция $g(n) = n-k$ также задает линейное преобразование размерности исходной задачи в размерность подзадач, а рекуррентное соотношение (3) однозначно определяет функцию $t(n)$ только при $n = km$, $m \in \mathbb{Z}_+$.

Следует особо отметить, что указанные подходы к организации рекурсии применимы лишь к задачам, удовлетворяющим свойству аддитивности. *Свойство аддитивности состоит в том, что решение задачи можно получить объединением решений подзадач как уменьшенных* (в смысле размера входных данных) *копий исходной задачи.* Решить соотношения (2), (3) в общем виде не представляется возможным – слишком общий вид они имеют. Между тем в частном случае, когда $A(n) = a$ и $B(n) = bn^\tau$, где a, b, τ – положительные константы, вид решений соотношений (2), (3) определяют две основные теоремы. При этом константа a задает число подзадач, порождаемых рекурсивной ветвью алгоритма, а степенная функция bn^τ определяет трудоемкость рекурсивного перехода.

4. Первая основная теорема

Теорема 1 определяет решение рекуррентного соотношения, которое характерно для рекурсивных алгоритмов типа "разделяй и властвуй". Впервые данная теорема была доказана

в 1980 г. Дж. Бентли, Д. Хакеном и Дж. Саксом [1], [7]. В метрической теории алгоритмов ее называют основной теоремой о рекуррентных соотношениях.

Теорема 1. *Пусть дано рекуррентное соотношение*

$$t(n) = \begin{cases} c, & \text{если } n = 1, \\ at\,(n/k) + bn^{\tau}, & \text{если } n > 1, \end{cases}$$

где $a > 0$, $k > 1$ — целые константы, $b \geqslant 0$, $c \geqslant 0$, $\tau \geqslant 0$ — вещественные константы. Тогда при $n = k^m$, $m \in \mathbb{Z}_+$ решением заданного соотношения является функция

$$t(k^m) = \begin{cases} a^m c + bk^{m\tau} m, & \text{если } a = k^{\tau}, \\ a^m c + bk^{m\tau} \dfrac{(a/k^{\tau})^m - 1}{(a/k^{\tau}) - 1}, & \text{если } a \neq k^{\tau}. \end{cases}$$

Следствие 1. *В предположениях теоремы 1 при больших значениях n и любых $b > 0$ и $c \geqslant 0$ справедливы оценки*

$$t(n) = \begin{cases} O(n^{\tau} \log_k n)\,, & \text{если } a = k^{\tau}, \\ O(n^{\tau})\,, & \text{если } a < k^{\tau}, \\ O\big(n^{\log_k a}\big)\,, & \text{если } a > k^{\tau}. \end{cases}$$

Если $b = 0$ и $c > 0$, то всегда

$$t(n) = O\big(a^{\log_k n}\big) = O\big(n^{\log_k a}\big)\,.$$

Первая основная теорема выявляет ряд особенностей. Все оценки следствия 1 при любых значениях $t(1) = c \geqslant 0$ дают полиномиальный порядок роста функции сложности $t(n)$. Стало быть, *рекурсия, организованная по принципу "разделяй и властвуй", всегда приводит к полиномиальным алгоритмам.* Имеет место явная зависимость сложности вычислений от числа подзадач и от их размера: *чем более сбалансированным является разбиение задачи на подзадачи, тем лучшую оценку сложности имеет рекурсивный алгоритм.*

5. Вторая основная теорема

Теорема 2 и вытекающие из нее следствия указывают решение и асимптотические оценки для частного случая рекуррентного уравнения (3), которое характерно для рекурсивных алгоритмов, образованных путем аддитивного уменьшения размера задачи на некоторую целую константу $k \geqslant 1$. Такие алгоритмы возникают, например, при рекурсивной реализации метода динамического программирования. Теорема 2 была сформулирована в работах [8], [9] и названа автором второй основной теоремой о рекуррентных соотношениях.

Теорема 2. *Пусть дано рекуррентное соотношение*

$$t(n) = \begin{cases} c, & \text{если } 0 \leqslant n \leqslant k-1, \\ a\,t\,(n-k) + b\,n^{\tau}, & \text{если } n \geqslant k, \end{cases} \tag{4}$$

где $a > 0$, $k \geqslant 1$ – целые константы, $b \geqslant 0$, $c \geqslant 0$, $\tau \geqslant 0$ – вещественные константы. Тогда при $n = km$, $m \in \mathbb{Z}_+$ верны неравенства

$$c + bk^{\tau-1} n \leqslant t(n) \leqslant c + \frac{b}{k}\, n^{\tau+1}, \quad \text{если } a = 1, \tag{5}$$

$$a^{n/k} c + bk^{\tau} \frac{a^{n/k} - 1}{a - 1} \leqslant t(n) \leqslant a^{n/k} c + bn^{\tau} \frac{a^{n/k} - 1}{a - 1}, \quad \text{если } a \neq 1. \tag{6}$$

Доказательство. Поскольку $n = km$, то для удобства выкладок введем следующую функцию

$$f(m) = t\,(km)\,. \tag{7}$$

Тогда $t\,(km-k) = t\,(k\,(m-1)) = f\,(m-1)$ и соотношение (4) принимает вид

$$t(km) = \begin{cases} c, & \text{если } m = 0, \\ at\,(km-k) + bk^{\tau}m^{\tau}, & \text{если } m > 0 \end{cases}$$

или

$$f(m) = \begin{cases} c, & \text{если } m = 0, \\ a\,f\,(m-1) + bk^{\tau}m^{\tau}, & \text{если } m > 0. \end{cases} \tag{8}$$

Подстановка соотношения (8) самого в себя дает:

$$f(m) = a^m c + bk^{\tau} S\,(a, m, \tau)\,, \qquad m = 0, 1, \ldots, \tag{9}$$

где $S\,(a, 0, \tau) = 0$, а при $m > 0$

$$S\,(a, m, \tau) = a^{m-1} \cdot 1^{\tau} + a^{m-2} \cdot 2^{\tau} + \ldots + a^0 \cdot m^{\tau} = \sum_{i=1}^{m} a^{m-i} i^{\tau}. \tag{10}$$

Остается найти значение конечной суммы $S\,(a, m, \tau)$. При $\tau = 0$ (когда затраты на рекурсивный переход не зависят от параметра рекурсии) верны равенства:

$$S\,(a, m, 0) = \sum_{i=1}^{m} a^{m-i} i^{\tau} = \sum_{i=1}^{m} 1^{m-i} i^0 = m, \quad \text{если } a = 1,$$

$$S\,(a, m, 0) = \sum_{i=1}^{m} a^{m-i} i^{\tau} = \sum_{i=1}^{m} a^{m-i} i^0 = a^{m-1} + a^{m-2} + \ldots + a^0 = \frac{a^m - 1}{a - 1}, \quad \text{если } a \neq 1.$$

Таким образом, в данном случае решением рекуррентного соотношения (8) является функция

$$f(m) = a^m c + bk^0 S\,(a, m, 0) = \begin{cases} c + bm, & \text{если } a = 1, \\ a^m c + b\,\dfrac{a^m - 1}{a - 1}, & \text{если } a \neq 1. \end{cases} \tag{11}$$

Для произвольного вещественного $\tau > 0$ значение суммы $S\,(a, m, \tau)$ установить крайне сложно. Между тем можно определить оценки сверху и снизу. Действительно, всякое слагаемое в (10) удовлетворяет неравенствам

$$a^{m-i} \leqslant a^{m-i} i^{\tau} \leqslant a^{m-i} m^{\tau}, \qquad i = 1, 2, \ldots, m,$$

и равенства достигаются при $\tau = 0$. Тогда

$$\sum_{i=1}^{m} a^{m-i} = S_0 \leqslant S\,(a, m, \tau) \leqslant S_1 = \sum_{i=1}^{m} a^{m-i} m^{\tau}. \tag{12}$$

Установим значения сумм S_0, S_1 при $a = 1$ и $a \neq 1$. Если $a = 1$, то

$$S_0 = \sum_{i=1}^{m} a^{m-i} = \sum_{i=1}^{m} 1^{m-i} = m,$$

$$S_1 = \sum_{i=1}^{m} a^{m-i} m^\tau = \sum_{i=1}^{m} 1^{m-i} m^\tau = \sum_{i=1}^{m} m^\tau = m^\tau \sum_{i=1}^{m} 1 = m^\tau \cdot m = m^{\tau+1}.$$

Откуда согласно (9), (12) и с учетом $a = 1$ для функции $f(m)$ справедливы неравенства:

$$c + bk^\tau m \leqslant f(m) \leqslant c + bk^\tau m^{\tau+1}. \tag{13}$$

Если $a \neq 1$, то

$$S_0 = a^{m-1} + a^{m-2} + \ldots + a^0 = \frac{a^m - 1}{a - 1},$$

$$S_1 = m^\tau \left(a^{m-1} + a^{m-2} + \ldots + a^0\right) = m^\tau \frac{a^m - 1}{a - 1}.$$

Следовательно, по (9), (12) верны верхняя и нижняя оценки:

$$a^m c + bk^\tau \frac{a^m - 1}{a - 1} \leqslant f(m) \leqslant a^m c + bk^\tau m^\tau \frac{a^m - 1}{a - 1}. \tag{14}$$

Заметим, что при $\tau = 0$ неравенства (13), (14) вырождаются в соответствующие равенства из (11). Подстановка $m = n/k$ в (11), (13), (14) дает требуемые оценки (5), (6). □

Формулы (5), (6) теоремы 2 в общем случае не дают точного решения рекуррентного соотношения (4). Они лишь определяют верхние и нижние границы. В этом смысле теорема 2 слабее первой основной теоремы. Тем не менее, в ряде частных случаев можно получить точные решения и оценки. Об этом свидетельствуют дальнейшие утверждения.

Следствие 2. *В предположениях теоремы* 2 *при* $\tau = 0$ *решением рекуррентного соотношения* (4) *является функция*

$$t(n) = \begin{cases} c + \dfrac{b}{k}\, n, & \text{если} \quad a = 1, \\ a^{n/k} c + b\, \dfrac{a^{n/k} - 1}{a - 1}, & \text{если} \quad a \neq 1. \end{cases} \tag{15}$$

Следствие 3. *При* $\tau = 0$, *любых значениях* $c \geqslant 0$ *и* $n \to \infty$ *для решения рекуррентного соотношения* (4) *верны асимптотические оценки*

$$t(n) = \begin{cases} O(n), & \text{если} \quad a = 1, \quad b > 0, \\ O\left(a^{n/k}\right), & \text{если} \quad a \neq 1, \quad b > 0. \end{cases} \tag{16}$$

В частности, при $\tau = 0$, $a = 1$ *и* $b = 0$ *всегда* $t(n) = O(1)$.

Следствие 4. *Для решения рекуррентного соотношения* (4) *при* $\tau \geqslant 0$, $a = 1$, $c \geqslant 0$, $b > 0$ *и* $n \to \infty$ *справедлива асимптотическая оценка*

$$t(n) = O\left(n^{\tau+1}\right). \tag{17}$$

Доказательство. Не представляет труда найти некоторые значения $S(a, m, \tau)$ при $a = 1$. Так, верны равенства

$$S(1, m, 0) = m,$$

$$S(1, m, 1) = \frac{m(m+1)}{2} = \frac{m^2}{2} + \frac{m}{2},$$

$$S(1, m, 2) = \frac{m(m+1)(2m+1)}{6} = \frac{m^3}{3} + \frac{m^2}{2} + \frac{m}{6},$$

$$S\,(1,m,3)=\frac{m^2\,(m+1)^2}{4}=\frac{m^4}{4}+\frac{m^3}{2}+\frac{m^2}{4}.$$

Подстановка $S\,(1,m,0)$, $S\,(1,m,1)$ в (9) подтверждает асимптотическую оценку (17). Для случая, когда $\tau>1$ – вещественное число, известно следующее приближение [2]:

$$S\,(1,m,\tau)=\sum_{i=1}^{m}i^{\tau}=\frac{m^{\tau+1}}{\tau+1}+\frac{m^{\tau}}{2}+\frac{\tau m^{\tau-1}}{12}+O\big(m^{\tau-2}\big)\,,\qquad \tau>1,$$

с которым вполне согласуются равенства для $S\,(1,m,2)$, $S\,(1,m,3)$. Отсюда по формулам (7), (9) получаем требуемую оценку (17):

$$\begin{aligned}t(n)=t\,(km)&=a^m c+bk^{\tau}S\,(1,m,\tau)=c+bk^{\tau}S\,(1,m,\tau)=\\&=c+bk^{\tau}\left(\frac{m^{\tau+1}}{\tau+1}+\frac{m^{\tau}}{2}+\frac{\tau m^{\tau-1}}{12}+O\big(m^{\tau-2}\big)\right)=\\&=c+b\left(\frac{k^{\tau+1}m^{\tau+1}}{k\,(\tau+1)}+\frac{k^{\tau}m^{\tau}}{2}+\frac{k\tau k^{\tau-1}m^{\tau-1}}{12}+O\big(k^{\tau-2}m^{\tau-2}\big)\right)=\\&=c+b\left(\frac{n^{\tau+1}}{k\,(\tau+1)}+\frac{n^{\tau}}{2}+\frac{k\tau n^{\tau-1}}{12}+O\big(n^{\tau-2}\big)\right)=O\big(n^{\tau+1}\big)\,.\end{aligned}$$

Примечательно, что *точное значение суммы* $S\,(1,m,\tau)$ *в математике получено лишь для целых положительных* τ и выражается через комбинаторные числа так [10]:

$$\begin{aligned}S\,(1,m,\tau)&=\sum_{i=1}^{m}i^{\tau}=\frac{1}{\tau+1}B_{\tau+1}(x)\Big|_0^{m+1}=\\&=\frac{1}{\tau+1}\left(B_{\tau+1}\,(m+1)-B_{\tau+1}(0)\right)=\frac{1}{\tau+1}\sum_{i=0}^{\tau}C_{\tau+1}^{i}\,(m+1)^{\tau+1-i}\,B_i,\qquad \tau>0,\end{aligned}\tag{18}$$

где

$$B_{\tau+1}(x)=\sum_{i=0}^{\tau+1}C_{\tau+1}^{i}B_i x^{\tau+1-i},\qquad C_{\tau+1}^{i}=\frac{(\tau+1)!}{i!\,(\tau+1-i)!}$$

— многочлен Бернулли и биномиальные коэффициенты соответственно, а B_i — числа Бернулли, определяемые рекуррентным соотношением

$$B_0=1,\quad C_i^0B_0+C_i^1B_1+C_i^2B_2+\ldots+C_i^{i-1}B_{i-1}=0,\quad i=2,\,3,\,\ldots,$$

согласно которому $B_i=0$ для всех нечетных $i>1$ и

$$B_0=1,\quad B_1=-\frac{1}{2},\quad B_2=\frac{1}{6},\quad B_4=-\frac{1}{30},\quad B_6=\frac{1}{42},\quad \ldots.$$

Предельный переход в (18) при $m\to\infty$ подтверждает асимптотическую оценку (17). В самом деле, согласно (18) функция $S\,(1,m,\tau)$ есть многочлен от $(m+1)$ степени $\tau+1$:

$$\begin{aligned}S\,(1,m,\tau)&=\frac{1}{\tau+1}\left(C_{\tau+1}^{0}(m+1)^{\tau+1}B_0+C_{\tau+1}^{1}\,(m+1)^{\tau}\,B_1+\ldots+C_{\tau+1}^{\tau}\,(m+1)^{1}\,B_{\tau}\right)=\\&=\frac{1}{\tau+1}\left((m+1)^{\tau+1}-\frac{\tau+1}{2}\,(m+1)^{\tau}+\ldots+(\tau+1)\,(m+1)\,B_{\tau}\right).\end{aligned}$$

Значит, $S(1,m,\tau) = O\Big((m+1)^{\tau+1}\Big) = O\big(m^{\tau+1}\big)$ и при $a=1$

$$t(n) = t(km) = a^m c + bk^\tau S(1,m,\tau) = O(k^\tau)\cdot S(1,m,\tau) = \\ = O(k^\tau)\cdot O\big(m^{\tau+1}\big) = O\Big((km)^{\tau+1}\Big) = O\big(n^{\tau+1}\big).$$

□

При $a>1$ и целочисленном $\tau \geqslant 0$ непосредственное применение к конечной сумме

$$S(a,m,\tau) = \sum_{i=1}^{m} a^{m-i} i^\tau$$

формулы суммирования Эйлера-Маклорена дает еще одну асимптотику.

Следствие 5. *Для решения рекуррентного соотношения* (4) *при* $a>1$, $c \geqslant 0$, $b>0$ *и целых положительных* τ *справедлива оценка*

$$t(n) = O\Big(a^{n/k}\Big). \tag{19}$$

Из (16), (17), (19) следует, что *рекурсивные алгоритмы, образованные аддитивным уменьшением размера задачи на некоторую константу, могут быть полиномиальными или экспоненциальными.* Так, при $B(n) = bn^\tau$ (когда трудоемкость рекурсивного перехода описывается степенной функцией) алгоритм будет иметь экспоненциальную сложность всегда, если $a \neq 1$, $b>0$ и $c \geqslant 0$ (или $b \geqslant 0$ и $c>0$). О чем говорят данные условия? Как следует организовывать рекурсивные вычисления, чтобы они имели полиномиальную сложность? Для получения ответов на эти вопросы необходимо напомнить смысл констант соотношения (4): константа a определяет число подзадач размера $n-k$, к которым сводится исходная задача; постоянные b и τ описывают трудоемкость рекурсивного перехода от n к $n-k$; константа c характеризует временные затраты, необходимые для непосредственного решения задачи, когда ее размерность чрезвычайно мала ($0 \leqslant n \leqslant k-1$).

Для реальных алгоритмов всегда существуют какие-то затраты на организацию рекурсии, т.е. $b>0$ и $c>0$. Если эти затраты не зависят от n, то $\tau = 0$. При $\tau = 0$, $a=1$ согласно следствию 3 рекурсивный алгоритм имеет линейную сложность. При $\tau \geqslant 0$, $a=1$ и $n \to \infty$ по следствию 4 для $t(n)$ справедлива оценка $t(n) = O\big(n^{\tau+1}\big)$. Таким образом, всегда при $a=1$ рекурсивный алгоритм, организованный путем уменьшения размера задачи на некоторую константу, является полиномиальным. По следствию 5 при $a \neq 1$ и целых положительных значениях константы τ верна оценка $t(n) = O\big(a^{n/k}\big)$. В общем случае, при $a \neq 1$, любом $\tau \geqslant 0$ и фиксированном значении $1 \leqslant k < n$ неравенство (6) свидетельствует о том, что функция $t(n)$ растет не быстрее, чем $O\big(n^\tau a^{n/k}\big)$. Для того чтобы ответить на вопрос о нижней грани скорости роста функции $t(n)$, положим в соотношении (4) $b=0$. Тогда исходное соотношение становится однородным

$$t(n) = \begin{cases} c, & \text{если } 0 \leqslant n \leqslant k-1, \\ at(n-k), & \text{если } n \geqslant k, \end{cases} \tag{20}$$

а неравенства (5), (6) обращаются в равенство $t_0(n) = a^{n/k}c$. Это равенство можно трактовать как общее решение однородного соотношения (20). При $a \neq 1$ решение $t_0(n) = a^{n/k}c$ всегда имеет экспоненциальный порядок роста. Согласно (6), при $a \neq 1$ частное решение $t_*(n)$ рекуррентного соотношения (4) есть функция, которая при больших n растет не быстрее, чем

$$bn^\tau \frac{a^{n/k}-1}{a-1} = O\Big(n^\tau a^{n/k}\Big).$$

Отсюда следует, при $a \neq 1$ функция $t(n) = t_0(n) + t_*(n)$ растет не медленнее, чем $t_0(n) = a^{n/k}c$, и не быстрее, чем $O\big(n^\tau a^{n/k}\big)$. С практической точки зрения это означает, что *при числе подзадач $a \neq 1$ рекурсивный алгоритм всегда экспоненциальный и никакие усовершенствования процедуры разбиения и объединения подзадач не способны изменить класс его сложности.*

Если $B(n) = b\tau^n$, $\tau \geqslant 1$ (трудоемкость рекурсивного перехода – неубывающая показательная функция), то почти всегда рекурсивный алгоритм будет иметь экспоненциальную сложность. Об этом свидетельствует следующая теорема.

Теорема 3. *Пусть дано рекуррентное соотношение*

$$t(n) = \begin{cases} c, & \text{если } \ 0 \leqslant n \leqslant k-1, \\ at\,(n-k) + b\tau^n, & \text{если } \ n \geqslant k, \end{cases} \tag{21}$$

где $a > 0$, $k \geqslant 1$ – целые константы, $b \geqslant 0$, $c \geqslant 0$, $\tau \geqslant 1$ – вещественные константы. Тогда при $n = km$, $m \in \mathbb{Z}_+$ решением соотношения (21) *является функция*

$$t(n) = \begin{cases} a^{n/k}\,(c + bn/k)\,, & \text{если } \ a = \tau^k, \\ \tau^n \left(\dfrac{a^{n/k}c}{\tau^n} + b\,\dfrac{\big(a^{n/k}/\tau^n\big) - 1}{(a/\tau^k) - 1} \right), & \text{если } \ a < \tau^k, \\ a^{n/k} \left(c + b\,\dfrac{\tau^k}{a} \cdot \dfrac{\big(\tau^n/a^{n/k}\big) - 1}{(\tau^k/a) - 1} \right), & \text{если } \ a > \tau^k. \end{cases}$$

Доказательство. Выполнив замену в (21) переменной n на выражение km, приходим к рекуррентному соотношению

$$f(m) = \begin{cases} c, & \text{если } m = 0, \\ af\,(m-1) + by^m, & \text{если } m > 0, \end{cases}$$

где $y = \tau^k$ — константа. Подстановка этого соотношения самого в себя дает

$$f(0) = c, \quad f(m) = a^m c + by\left(a^{m-1}y^0 + a^{m-2}y^1 + \ldots + a^0 y^{m-1}\right), \quad m = 1,\, 2,\, \ldots.$$

Пусть $r = y/a = \tau^k/a$. Тогда

$$f(m) = a^m \left(c + b\frac{by}{a} \left(\frac{a^{m-1}}{a^{m-1}} + \frac{a^{m-2}}{a^{m-1}}y + \ldots + \frac{a^0}{a^{m-1}}y^{m-1} \right) \right) = a^m \left(c + br \sum_{i=0}^{m-1} r^i \right).$$

Возможны три ситуации:

- если $r = 1$, то

$$f(m) = a^m \left(c + br \sum_{i=0}^{m-1} r^i \right) = a^m\,(c + bm) \quad \text{или} \quad t(n) = a^{n/k}\,(c + bn/k)\,;$$

- если $r > 1$, то

$$f(m) = a^m \left(c + br \sum_{i=0}^{m-1} r^i \right) = a^m r^m \left(\frac{c}{r^m} + b \sum_{i=0}^{m-1} \frac{1}{r^i} \right) = y^m \left(\frac{c}{r^m} + b\,\frac{1 - 1/r^m}{1 - 1/r} \right)$$

или

$$t(n) = \tau^n \left(\frac{a^{n/k}c}{\tau^n} + b\,\frac{\big(a^{n/k}/\tau^n\big) - 1}{(a/\tau^k) - 1} \right);$$

- если $r < 1$, то

$$f(m) = a^m \left(c + br \sum_{i=0}^{m-1} r^i \right) = a^m \left(c + br \, \frac{1 - r^m}{1 - r} \right)$$

или

$$t(n) = a^{n/k} \left(c + b \, \frac{\tau^k}{a} \cdot \frac{\left(\tau^n / a^{n/k}\right) - 1}{(\tau^k / a) - 1} \right).$$

□

Следствие 6. *В предположениях теоремы* 3 *при больших значениях* n, *любых* $\tau \geqslant 1$, $b > 0$, $c > 0$ *справедливы оценки*

$$t(n) = \begin{cases} O\left(na^{n/k}\right), & \text{если } a = \tau^k, \\ O(\tau^n), & \text{если } a < \tau^k, \\ O\left(a^{n/k}\right), & \text{если } a > \tau^k. \end{cases}$$

Все оценки следствия 6 указывают на экспоненциальный порядок роста функции $t(n)$. Исключение составляет тривиальный случай, когда $\tau = 1$ и $a = 1$. В этом случае $t(n) = O(n)$. Таким образом, организовывая рекурсию аддитивным уменьшением размерности задачи на некоторую константу k $(1 \leqslant k < n)$, необходимо учитывать следующее. Асимптотические оценки для функции временной сложности $t(n)$ сильно зависят от a (числа подзадач) и полиномиальные алгоритмы возможны лишь в случае, когда задача размера n сводится только к одной подзадаче размера $n - k$. Причина экспоненциальной сложности при $a \neq 1$ кроется в возникновении на каждом шаге рекурсии перекрывающихся подзадач. Если процедура разбиения и объединения подзадач несовершенна (имеет неполиномиальную трудоемкость), то практически во всех случаях рекурсивный алгоритм будет иметь экспоненциальную сложность.

Примечательно, что в методе динамического программирования уравнение Беллмана, как правило, порождает именно перекрывающиеся между собой подзадачи. Поэтому для него вместо рекурсии целесообразно применять технику итерационного построения таблиц, несмотря на то, что данный метод рекурсивен по своей сути.

Основные теоремы о рекуррентных соотношениях представляют пусть не универсальное, но достаточно мощное средство асимптотической оценки сложности рекурсивных алгоритмов, построенных определенным образом. Из этих теорем следует практический совет разработчикам алгоритмов и программ: *чтобы получать рекурсивные алгоритмы полиномиальной сложности, при их разработке необходимо, прежде всего, позаботиться о сбалансированности подзадач, на каждом шаге рекурсии порождать совершенно новые независимые подзадачи и эффективно выполнять процедуру разбиения и обьединения подзадач.*

Список литературы

[1] А.Ахо, Дж.Хопкрофт, Дж.Ульман, Структуры данных и алгоритмы, М., Вильямс, 2007.

[2] Д.Грин, Д.Кнут, Математические методы анализа алгоритмов, М., Мир, 1987.

[3] Р.Грэхем, Д.Кнут, О.Поташник, Конкретная математика, М., Мир, Бином. Лаборатория знаний, 2006.

[4] С.К.Ландо, Лекции о производящих функциях, М., МЦНМО, 2004.

[5] В.Н.Сачков, Введение в комбинаторные методы дискретной математики, М., Наука, 1982.

[6] В.А.Головешкин, М.В.Ульянов, Теория рекурсии для программистов, М., ФИЗМАТЛИТ, 2006.

[7] Кормен, Ч.Лезерсон, Р.Ривест, Алгоритмы: построение и анализ, М., МЦНМО, 1999.

[8] В.В.Быкова, Дискретная математика с использованием ЭВМ, Красноярск, КрасГУ, 2006.

[9] В.В.Быкова, Асимптотическая оценка сложности рекурсивных алгоритмов с аддитивным уменьшением размерности задачи, Труды шестой Всероссийской ФАМ'2007, Ч. 2. Красноярск, Гротеск, 2007, 17-25.

[10] Г.Корн, Т.Корн, Справочник по математике, М., Наука, 1973.

Mathematical Methods for the Analysis of Recursive Algorithms

Valentina V.Bykova

We prove a theorem that defines asymptotic estimates for the solution of a recurrent relation. This recurrent relation typically describes time complexity functions for recursive algorithms with an additive reduction of the dimension of a given problem. The presented results together with a known main theorem on recurrent relations give a tool for the analysis of the complexity of two most typical recursive schemes.
Keywords: complexity of algorithms, recursion, recurrent relations