A Real Elementary Approach to the Master Recurrence and Generalizations*

Chee Yap

[1] Courant Institute of Mathematical Sciences
New York University
New York, NY 10012, USA
yap@cs.nyu.edu
[2] Korea Institute of Advanced Study
Seoul, Korea

Abstract. The master theorem provides a solution to a well-known divide-and-conquer recurrence, called here the master recurrence. This paper proves two cook-book style generalizations of this master theorem. The first extends the treated class of driving functions to the natural class of exponential-logarithmic (EL) functions. The second extends the result to the multiterm master recurrence. The power and simplicity of our approach comes from re-interpreting integer recurrences as real recurrences, with emphasis on elementary techniques and real induction.

1 Introduction

Techniques for solving recurrences are among the standard repertoire of algorithmic textbooks [13,5,16,4,1]. A proto-typical recurrence arising in the analysis of efficient recursive algorithms is

$$T(n) = aT(n/b) + d(n) \tag{1}$$

where $a > 0$ and $b > 1$ are arbitrary real numbers, and $d(n) \geq 1$ is the **driving function**. We call (1) the **master recurrence** since theorems providing its solution are widely known as "master theorems". The solutions depend on the nature of $d(n)$. The case where $d(n)$ is multiplicative is treated in [1, p. 301]. In an influential note, Bentley, Haken and Saxe [3, Table 1, p.39] proved a master theorem under a fairly general hypothesis on $d(n)$. Recurrence (1) generalizes to

$$T(n) = \sum_{i=1}^{k} a_i T(n/b_i) + d(n) \tag{2}$$

where $a_i > 0$ and $b_i > 1$ are arbitrary real constants ($k \geq 2$). We call (2) the **multiterm master recurrence**. E.g., the 2-term recurrences $T(n) = T(n/b_1) + T(n/b_2) + n$ and $T(n) = T(n/2) + T(n/4) + \log n$ arise (respectively)

* This work is supported by an National Science Foundation Grants #CCF-0728977 and #CCF-0917093, and also with KIAS support.

M. Ogihara and J. Tarui (Eds.): TAMC 2011, LNCS 6648, pp. 14–26, 2011.

in fast median algorithms [4, p. 240] and in conjugate search tree analysis in Computational Geometry [6].

To discuss the literature, it is useful to begin with the "standard" master theorem for (1). This is Proposition 1 in the next Section. It has two kinds of generalizations: (A) The first kind, as in Verma [19], extends the class of driving functions $d(n)$ that are captured by the master theorem. Verma's main result [19, Theorem 13] provided integral bounds on solutions when the driving functions $d(n)$ satisfy some growth properties. (B) The second kind comes from extending the master recurrence itself. Wang and Fu [20, Theorem 3.5] gave integral bounds for a parametric form of (1) where a, b are now functions of n. Of course, the multiterm recurrence (2) is also a generalization of the second kind. An early treatment of multiterm recurrences is found in Purdom and Brown [16]. Multiterm master theorems are given by Kao [11], Akra and Bazzi [2], and Roura [18, Theorem 2.3]. Leighton [14] provides an exposition of [2]. We remark that obtaining generalized bounds in the form of integrals, by itself, is not satisfactory: our goal is to achieve "cookbook style" theorems [12] as exemplified by the master theorem.

¶*1. Contributions and Overview.* Our main contribution is two cookbook style generalizations of the master theorem, **Theorems A** and **B**. They are natural extensions, and completions, of known results. They serve to unify many complexity analysis of individual algorithms: thus, no previous master theorems capture the analysis of Schönhage-Strassen's multiplication algorithm [13], but this is now an application of Theorem A. Similarly, the conjugate tree analysis of Edelsbrunner and Welzl [6] is a consequence of Theorem B. Furthermore, Theorem B shows that the conjugate tree exponent, $\alpha = \lg(\phi - 1) \sim 0.695$ where $\phi = 1.618\ldots$ is the golden ratio, can be systematically obtained, and that this bound is tight to Θ-order. Our second contribution is the introduction of rigorous elementary techniques for these derivations. In particular, we provide summation formulas for Exponential-Logarithmic (EL) functions. Elementary techniques are possible because we exploit bounds which are tight to (only) Θ-order.

Section 2 will review the master theorem and extensions. Section 3 states our two main results: Theorem A is a master theorem that allows the driving function $d(n)$ to be any EL-function. Theorem B is a multiterm master theorem. Section 4 introduces elementary summation techniques. Section 5 addresses elementary sums and proves Theorem A. Section 6 introduces real induction and proves Theorem B. We conclude in Section 7.

¶*2. Approach of Paper.* Our approach has two emphases. The first is on *real recurrences*: in the recurrences (1) and (2), we treat n as a real variable, $T(n)$ as a real function and all constants a, b, a_i, b_i are real. In contrast, most of the literature regards n as an integer variable. E.g., Kao [11] treats this multiterm recurrence:

$$T(n) = \begin{cases} c \cdot n^\alpha \cdot \log^\beta n + \sum_{i=1}^k a_i T(\lceil b_i n \rceil) & \text{for } n \geq n_0 \\ c_n & \text{for } n < n_0 \end{cases} \tag{3}$$

where n and k are positive integers, c, c_n, a_i are positive constants, α, β are non-negative constants, $b_i \in (0,1)$ and $n_0 \geq \max_{i=1}^k \frac{1}{1-b_i}$. Similar viewpoints are seen in Wang-Fu, Akra-Bazzi and Roura. But most driving functions such as $d(n) = \sqrt{n}$ and $d(n) = n \log n$ are naturally real functions. Hence our real extension remains well-defined if we simply omit the (troublesome) integer-valued functions such as ceiling or floor. A standard approach to avoid ceiling/floor functions is "domain restriction". E.g., restricting the domain of $T(n)$ in recurrence (1) to positive powers of b ([4, p. 145, Problem 4.44] or [19]) and requiring b to be integer. Finally, to restore n to range over all integers, we need special arguments (e.g., [5, pp. 81–84]) or smoothness assumptions on $T(n)$. Although the idea of real recurrences is nascent in several of the papers (e.g., [2,19,18]), it seldom takes on a full-blown form. In this paper, we develop basic tools to rigorously treat real recurrences.

Our second emphasis is the use of elementary methods. Here "elementary" means the avoidance of calculus [10], not that the results are trivial or easy to come by. It is conventional wisdom in algorithmics to solve $T(n)$ up to Θ-order because it yields robust conclusions about complexity (e.g., [3]). But it is seldom noted that Θ-order analysis lends itself to elementary techniques. E.g., below we give elementary Θ-bounds on sums that are usually treated by the Euler-Maclaurin formula [8, p. 217]. Authors also fail to exploit problem simplifications from Θ-order analysis [19,20,18]. For instance, up to Θ-order, most solutions are insensitive to the initial conditions. *So we need not explicitly specify initial conditions.* Instead, this paper assumes the following **default initial condition** (DIC):

$$T(n) = C, \qquad (n \leq n_0) \tag{4}$$

for some constant $C \geq 0$ and real n_0; the recurrence equation is assumed to be operative for $n > n_0$. Usually $C = 0$ is simplest and easily justified. Thus, using real recurrences under DIC, Kao's recurrence (3) greatly simplifies to $T(n) = n^\alpha \log^\beta n + \sum_{i=1}^k a_i T(b_i n)$. Roura [18] and Leighton [14] also discuss robustness issues. The pedagogical advantage of avoiding calculus for computer science students is obvious. Also our driving function $d(n)$ need not be differentiable (Lipschitz type bounds suffice).

These two emphases (real and elementary) explain the title of this paper. The simplicity and power of the real approach will hopefully be evident.

2 On Master Theorems

The "standard" master theorem provides the motif for generalizations. Relative to the master recurrence (1), we define a **watershed constant**

$$\alpha := \log_b a \tag{5}$$

and an associated **watershed function** $w(n) := n^\alpha$. The master theorem is a trichotomy based on a comparison between $d(n)$ and $w(n)$:

Proposition 1 (Standard Master Theorem). *The solution to (1) is*

$$T(n) = \Theta \begin{cases} n^\alpha & \text{if } d(n) = O(w(n)n^{-\varepsilon}) \text{ for some } \varepsilon > 0 \quad [\text{CASE } (-)] \\ n^\alpha \log n & \text{if } d(n) = \Theta(w(n)) \quad [\text{CASE } (0)] \\ d(n) & \text{if } \text{“}d(n) = \Omega(w(n)n^{\varepsilon})\text{”} \text{ for some } \varepsilon > 0 \quad [\text{CASE } (+)]. \end{cases} \tag{6}$$

This is taken from Cormen et al [5, p. 73], except n is now a real variable and $a > 0$ (not $a \geq 1$). The trichotomy amounts to $d(n)$ being (resp.) polynomially-slower than, Θ-order of, and polynomially-faster than $w(n)$. The condition for polynomially-faster [CASE (+)] in (6) is written in quotes because the original Ω-notation in [3, p. 39] was non-standard. This was replaced in [5] by the weaker **regularity condition**: for some $C > 1$,

$$d(n) \geq C \cdot a \cdot d(n/b) \text{ (ev. } n) \tag{7}$$

where the qualification "(ev. n)" reads as "eventually n", meaning that the statement holds for large enough n. Our real approach affords a "two-line proof" of Prop. 1: by induction, $T(n) = a^{i+1}T\left(n/b^{i+1}\right) + \sum_{j=0}^{i} a^j \cdot d\left(\frac{n}{b^j}\right)$, for $i = 0, 1, \ldots$. Setting $i = m := \lceil \log_b n \rceil$, and using DIC (with $C = 0$) in (4), we obtain

$$T(n) = \sum_{j=0}^{m} a^j d(n/b^j). \tag{8}$$

The 3 cases follow by plugging in the corresponding bounds for $d(n)$. Q.E.D.

¶*3. Extended Master Theorem.* It is well-known that Prop. 1 does not cover many useful driving functions such as $d(n) = w(n)\log^\delta n$ ($\delta \neq 0$). By applying the general techniques of **domain** and **range transformations** [4, pp. 130-137], we get:

Proposition 2 (Extended Master Theorem). *The solution to (1) is*

$$T(n) = \Theta \begin{cases} d(n) & \text{if } d(n) \text{ satisfies the reg. cond. (7)} \quad [\text{CASE } (+)] \,, \\ d(n)\log n & \text{if } d(n) = \Theta(n^\alpha \log^\delta n) \text{ for some } \delta > -1 \quad [\text{CASE } (0)] \,, \\ d(n)\log n \log\log n & \text{if } d(n) = \Theta(n^\alpha \log^\delta n) \text{ where } \delta = -1 \quad [\text{CASE } (1)] \,, \\ n^\alpha & \text{if } d(n) = O(n^\alpha \log^\delta n) \text{ for some } \delta < -1 \quad [\text{CASE } (-)] \,. \end{cases}$$

Prop. 2 generalizes the master theorem (6) since the original CASEs (−)&(0) are subsumed by the new ones; CASE (+) is unchanged but CASE (1) is new. Prop. 2 is from Brassard and Bratley [4, p. 145] (cf. [5, p.84, Ex.4.4-2]), slightly sharpened here: we state CASE (+) in terms of the regularity condition. Further [4] assumes $n = n_0 b^i$ for integers $n_0 \geq 1$ and $b \geq 2$. Wang and Fu's version of Prop. 2 is in [20, §4.3 and Table 1]. Roura's version [17] missed CASE (1). Case 3 in Verma's version [19, Theorem 1] is weaker than CASE (0) as he assumes $\delta \geq 0$. Still, our Prop. 2 is silent when the driving functions are, for example:

$$\left.\begin{array}{l} d_0(n) := n^\alpha \log n \log\log n \\ d_1(n) := n^\alpha (\log\log n)^r \\ d_2(n) := n^\alpha \frac{(\log\log\log n)^s}{\log n \log\log n} \end{array}\right\} \tag{9}$$

for non-zero r, s. Note that $d_0(n)$ arise in the Schönhage-Strassen's algorithm [13] with $\alpha = 1$. It turns out that the solutions for $T(n)$ under the driving functions (9) are (resp.)

$$\Theta(n^\alpha \cdot \log^2 n \log\log n), \quad \Theta(n^\alpha \cdot \log n (\log\log n)^r), \quad \Theta(n^\alpha (\log\log\log n)^{s+1}). \tag{10}$$

The last case assumes $s > -1$; different solutions arise if $s = -1$ or if $s < -1$. Theorem A in the next section will provide these solutions, and much more.

3 Two Generalized Master Theorems

This section will state our two main results: Theorems A and B. Both are extensions of Prop. 2. We begin with Theorem B since Theorem A requires a bit more development to formulate.

¶*4.* We need the multiterm analogues of (5) and (7): the **watershed constant** for (2) is the unique α satisfying the characteristic equation $\sum_{i=1}^k \frac{a_i}{b_i^\alpha} = 1$ (see [11,2]). Say $d(n)$ satisfies the **regularity condition** of (2) if, for some $0 < c < 1$,

$$\sum_{i=1}^k a_i d\left(\frac{n}{b_i}\right) \leq c \cdot d(n). \tag{11}$$

Theorem B – Multiterm Master Theorem
The solution to (2) satisfies

$$T(n) = \Theta \begin{cases} d(n) & \text{if } d(n) \text{ satisfies the reg. cond. (11)} \quad \text{[CASE (+)]}, \\ d(n)\lg n & \text{if } d(n) = \Theta(n^\alpha \lg^\delta n) \text{ for some } \delta > -1 \quad \text{[CASE (0)]}, \\ d(n)\lg n \lg\lg n & \text{if } d(n) = \Theta(n^\alpha \lg^\delta n) \text{ where } \delta = -1 \quad \text{[CASE (1)]}, \\ n^\alpha & \text{if } d(n) = O(n^\alpha \lg^\delta n) \text{ for some } \delta < -1 \quad \text{[CASE (−)]}. \end{cases}$$

All previous versions of Theorem B have 3 cases, as in Prop. 1. Our CASE (1) is new, and in some sense it completes this line of analysis. Kao [11] gave an inductive proof for the case $k = 2$ only. Roura [18, Theorem 2.3] treats more general driving functions; like Kao, the treatment is for integer recurrences. Akra and Bazzi [2] deduced their result from a general integral bound, which Leighton [14] simplified.

¶*5. On EL-functions.* We now introduce the family[1] of "EL-functions" which will serve as driving functions for Theorem A. The **iterated logarithm function** (for $k \in \mathbb{N}$) is defined as $\ell\ell g_k(x) := \underbrace{\lg(\lg(\cdots(\lg(x))\cdots))}_{k \text{ times}}$ where $\lg := \log_2$ is the "computer science logarithm". E.g., $\ell\ell g_0(x) = x$, $\ell\ell g_1(x) = \lg x$, $\ell\ell g_2(x) = \lg\lg x$. We may extend the index k to all integers where, for $k \in \mathbb{N}$, $\ell\ell g_{-(k+1)}(x)$

[1] EL is mnemonic for **E**xponential-**L**ogarithmic.

$:= 2^{\ell\ell g_{-k}(x)}$. Thus $\ell\ell g_{-1}(x) = 2^x$ and $\ell\ell g_{-2}(x) = 2^{2^x}$. An **exponent sequence** is a function $\mathbf{e} : \mathbb{Z} \to \mathbb{R}$ with finite support, i.e., $\mathbf{e}(i) = 0$ for all but finitely many i's. We normally write e_i for $\mathbf{e}(i)$. Call $\mathbf{e}$ **trivial** if $e_i = 0$ for all $i \in \mathbb{Z}$. For nontrivial $\mathbf{e}$, the smallest index i such that $e_i \neq 0$ is called its **order**, and e_i its **leading power**; these are denoted $Ord(\mathbf{e})$ and $Pow(\mathbf{e})$ (resp.). Given $k, \ell \in \mathbb{N}$, if $e_i = 0$ for all $i < -k$ and $i > \ell$, we may represent $\mathbf{e}$ (non-uniquely) as the sequence $\mathbf{e} = (e_{-k}, \ldots, e_{-1}, e_0; e_1, \ldots, e_\ell)$ where the semi-colon ";" between e_0 and e_1 indicates the "origin" of the bidirectional sequence. An **elementary function** is a partial function $f(x)$ of the form $\mathrm{EL}^{\mathbf{e}}(x) := \prod_{i\in\mathbb{Z}} \ell\ell g_i^{e_i}(x)$ where $\ell\ell g_k^a(x) := (\ell\ell g_k(x))^a$ is the **a-th power** of $\ell\ell g_k(x)$. E.g., $\mathrm{EL}^{(-2;0,\pi)}(x) = x^{-2}(\lg\lg(x))^\pi$.

We need one more concept to state Theorem A. Any driving function $d(n)$ of the form $\mathrm{EL}^{\mathbf{e}}(n)$ where $Ord(\mathbf{e}) = 0$ and $Pow(\mathbf{e}) = \alpha$ (watershed constant) is just "at the cusp" between CASES (+) and (−), and so we call such $\mathbf{e}$ a **cusp exponent**. The **cusp order** of any $\mathbf{e}$ is the largest index $h \geq 1$ such that $\mathbf{e}(i) = -1$ for $i = 1, 2, \ldots, h-1$; also call $\mathbf{e}(h)$ the **cusp power**. Cusp exponents have form $\mathbf{e} = (\alpha; -1, -1, \ldots, -1, \beta, \ldots)$ where $\beta \neq -1$ is the cusp power.

Theorem A – Generalized Master Theorem

Let the driving function be $d(n) = \mathrm{EL}^{\mathbf{e}}(n)$ with $k = Ord(\mathbf{e})$ and $c = Pow(\mathbf{e})$. Also let the cusp order and cusp power of $\mathbf{e}$ be h and β respectively. The solution $T(n)$ to the master recurrence (1) with watershed constant $\alpha = \log_b a$ satisfies

$$T(n) = \Theta \begin{cases} d(n) & \text{if } (k < 0 \wedge c > 0) \text{ or } (k \geq 0 \wedge \mathbf{e}(0) > \alpha), \text{ [CASE (+)]} \\ d(n) LL_h(n) & \text{if } (k = 0 \wedge \mathbf{e}(0) = \alpha \wedge \beta > -1), \text{ [CASE } (h-1)\text{]} \\ n^\alpha & \text{otherwise} \qquad\qquad\qquad\qquad\quad \text{[CASE (−)]} \end{cases} \tag{12}$$

where $LL_h(n) := \prod_{i=1}^{h} \ell\ell g_i(n) = \lg n \cdot \lg\lg n \cdot \ell\ell g_3(n) \cdots \ell\ell g_h(n)$.

This theorem has infinitely many cases, one for each $h \geq 1$. For $h = 1$ and 2, we reproduce CASEs (0) and (1) of Prop. 2. Verma [19, Theorem 13] has driving functions not covered here. To make Theorem A fully comparable to Prop. 1, we could re-formulate CASE (+) using the regularity condition. An interesting corollary of Theorem A is this: *when the driving function is an EL-function, the solution to the master recurrence is, up to Θ-order, another EL-function.*

To see Theorem A in action, recall the driving functions $d_0(n), d_1(n), d_2(n)$ in (9). The exponent sequence for them are (resp.)

$$(\alpha; 1, 1), \quad (\alpha; 0, r), \quad (\alpha; -1, -1, s)$$

For $d_0(n)$ and $d_1(n)$, their cusp order h and cusp power β are (resp.) $(h, \beta) = (1, 1)$ and $(1, 0)$. As these cusp powers are > -1, they are both fall under CASE (0) which has solution $T(n) = \Theta(d(n)LL_1(h))$. This yields the first two solutions in (10). For $d_2(n)$, we have three possibilities: If $s > -1$, then $(h, \beta) = (3, s)$ and the solution falls under CASE (2) with solution $T(n) = \Theta(d(n)LL_3(n)$ as given by the third bound in (10). If $s = -1$, then $(h, \beta) = (4, 0)$ and it falls under CASE (3) with solution $T(n) = \Theta(d(n)LL_4(n)) = \Theta(n^\alpha \ell\ell g_4(n))$. If $s < -1$, then $(h, \beta) = (3, s)$ but it falls under CASE (−) with solution $T(n) = \Theta(n^\alpha)$.

4 Elementary Summation Techniques

A **complexity function** is a partial function $f : \mathbb{R} \to \mathbb{R}$ where $f(x)$ is defined for x large enough. Standard asymptotic notations (big-Oh, big-Omega, Theta, etc) can be extended to partial functions [21]. Define two kinds of sums on f-values between real limits $a, b \in \mathbb{R}$:

$$\left.\begin{array}{l} \sum_{x\geq a}^{b} f(x) := f(b) + f(b-1) + \cdots + f(b - \lfloor b-a \rfloor), \text{ (descending)} \\ \sum_{x=a}^{b} f(x) := f(a) + f(a+1) + \cdots + f(a + \lfloor b-a \rfloor) \text{ (ascending)} \end{array}\right\} \quad (13)$$

Both sums are 0 for $b < a$; else the descending (ascending) sum will include the term $f(b)$ ($f(a)$). The two versions are distinguished by way we write their lower limits: "$\sum_{x\geq a}$" versus "$\sum_{x=a}$". Such sums are always well-defined as *any undefined summand $f(x)$ is replaced by* 0. Our manipulations below exploits:

$$\sum_{x\geq 0}^{b} f(x) \equiv \sum_{x=0}^{b} f(b-x). \quad (14)$$

Our main focus will be descending sums of the form $S^f(n) := \sum_{x\geq 1}^{n} f(x)$ for real values of n. This sum is traditionally bounded with the Euler-Maclaurin formula. But we now provide elementary method based on "growth-types":

- f is **polynomial-type** if $f \geq 0$, f is non-decreasing, and for some $K > 0$, $f(x) \leq Kf(x/2)$ (ev.).
- f **increases exponentially** if $f > 0$ and for some $C > 1$ and $k > 0$, $f(x) \geq C \cdot f(x-k)$ (ev.).
- f **decreases exponentially** if $f > 0$ and for some $0 < c < 1$ and $k > 0$, $f(x) \leq c \cdot f(x-k)$ (ev.).

We say that f is **exponential-type** if it increases or decreases exponentially. Polynomial-type functions corresponds to Verma's "slowly growing functions" [19]. These growth-types are non-exhaustive: for instance, it can be shown that the function $x^{\ln x}$ is not captured. Our next result is relatively easy but useful because it reduces estimating $S^f(n)$ to the easier problem of determining the growth type of f.

Theorem 1 (Summation Rules)

$$S^f(n) = \Theta \begin{cases} nf(\Theta(n)) & \text{if } f \text{ is polynomial-type,} \\ f(n) & \text{if } f \text{ increases exponentially,} \\ 1 & \text{if } f \text{ decreases exponentially.} \end{cases}$$

To determine the growth-type of f, we can exploit simple closure properties of growth types (e.g., each type is closed under addition, multiplication, raising to a positive power, etc). Moreover, an EL-functions f (say $f(x) = \mathrm{EL}^{\mathbf{e}}(()x)$) is exponential-type if $Ord(\mathbf{e}) < 0$; otherwise, either f or f^{-1} is polynomial-type. We exploit such properties in our proofs.

5 Elementary Sums and Proof of Theorem A

Our goal is to bound **elementary sums**, i.e., $S^f(n)$ where f is an EL-function. Such sums may be denoted by $S^{\mathbf{e}}(n) := \sum_{x \geq 1}^{n} \mathrm{EL}^{\mathbf{e}}(x)$.

¶*6. Error Notation.* We write "$x = y \pm z$" to mean that $x = y + \theta z$ for some θ where $|\theta| \leq 1$. The general convention [21] is that in any numerical expression, each occurrence of the symbol "$\pm$" stands for a sequence "$+\theta$" where θ is an anonymous variable satisfying $|\theta| \leq 1$. Like the big-Oh notation, this is a very useful variable hiding device. Thus, the following holds for any continuous f:

$$\sum_{i=1}^{n} f(n \pm c) = nf(n \pm c). \tag{15}$$

We need 3 operators on $\mathbf{e}$. The **shift operator** σ is: $\sigma(\mathbf{e})(i) = \mathbf{e}(i+1)$ for all i. E.g., $\mathrm{EL}^{\sigma(\mathbf{e})}(n) = \mathrm{EL}^{\mathbf{e}}(2^n)$. For $c \in \mathbb{R}$, let $\mathbf{e}'$ (resp., $\mathbf{e} + c$) denote the exponent sequence where we zero out (resp., add c to) the component $\mathbf{e}(0)$: $\mathbf{e}'(0) = 0$ and $(\mathbf{e} + c)(0) = c + \mathbf{e}(0)$, and $\mathbf{e}'(i) = (\mathbf{e} + c)(i) = \mathbf{e}(i)$ $(i \neq 0)$. Usually, $c = 1$. Another result we need is this: if $Ord(\mathbf{e}) \geq 0$ and $c \in \mathbb{R}$,

$$\mathrm{EL}^{\mathbf{e}}(2^{n \pm c}) = \mathrm{EL}^{\sigma(\mathbf{e})}(n \pm c) = \Theta(\mathrm{EL}^{\sigma(\mathbf{e})}(n)) = \Theta(\mathrm{EL}^{\mathbf{e}}(2^n)). \tag{16}$$

The next transformation of elementary sums is the key.

Lemma 1 (Key Transformation). *If* $Ord(\mathbf{e}) \geq 0$, $S^{\mathbf{e}}(n) = \Theta(S^{\sigma(\mathbf{e}+1)}(\lg n))$.

Up to Θ-order, we will show $S^{\mathbf{e}}(n) = \Theta(f(n))$ for some elementary function f. The goal (next Theorem) is to determine the exponent of f. Note that all asymptotic notations assume a fixed $\mathbf{e}$. We need a variant notion of cusp order from Section 3: for any $\mathbf{e}$, its **augmented cusp order** is 0 if $\mathbf{e}(0) \neq -1$; else it is the cusp order of $\mathbf{e}$. Also $\mathbf{e}(h)$ is the **augmented cusp power** if h is the augmented cusp order. If $\mathbf{e}(0) = -1$, then the augmented concepts agree with the original ones.

Theorem 2 (Elementary Sums). *Let* $k := Ord(\mathbf{e})$, $c := Pow(\mathbf{e})$. *Also, let the augmented cusp order and power of* $\mathbf{e}$ *be* h *and* β, *respectively. Then*

$$S^{\mathbf{e}}(n) = \Theta \begin{cases} \mathrm{EL}^{\mathbf{e}}(n) & \text{if } (k \leq -1 \wedge\ c > 0), \text{ [CASE } (+)] \\ \mathrm{EL}^{\mathbf{e}}(n) LL_{h+1}(2^n) & \text{if } (k \geq 0 \wedge\ \beta > -1), \text{ [CASE } (h)] \\ 1 & \text{else} \qquad\qquad\quad \text{[CASE } (-)] \end{cases} \tag{17}$$

The proof uses repeated application of the key transformation, Lemma 1. To see the power of Thm. 2, note that it implies $S^f(n) = \Theta(\ell\ell g_h(n))$ when $f(x) = 1/LL_h(2^x)$ (for any $h \in \mathbb{N}$). Goursat [9, p. 349] has the calculus analogue of this.

¶7. *Proof of Theorem A.* Let $n = b^m$. From (8), we have

$$\begin{array}{rl} T(n) = \sum_{i=0}^m a^i d(n/b^i) & = \sum_{i\geq 0}^m a^{m-i} d(b^i) \qquad \text{(by (14))} \\ = n^\alpha \sum_{i\geq 0}^m a^{-i} d(b^i) & = n^\alpha \sum_{i\geq 0}^m a^{-i} \mathrm{EL}^{\mathbf{e}}(b^i). \end{array} \tag{18}$$

If $k \leq -1$, the function $F(i) := a^{-i}\mathrm{EL}^{\mathbf{e}}(b^i)$ is increasing (decreasing) exponentially when $c > 0$ ($c < 0$). Applying our summation rules (Thm. 1) to (18),

$$T(n) = n^\alpha \cdot \Theta \left\{ \begin{array}{ll} a^{-m}\mathrm{EL}^{\mathbf{e}}(b^m) & \\ 1 & \end{array} \right. = \Theta \left\{ \begin{array}{ll} \mathrm{EL}^{\mathbf{e}}(n) & \text{if } c > 0, \\ n^\alpha & \text{if } c < 0. \end{array} \right. \tag{19}$$

This proves our theorem for $k \leq -1$. Next assume $k \geq 0$. Writing $e_0 = \mathbf{e}(0)$,

$$T(n) = n^\alpha \sum_{i\geq 0}^m a^{-i}\mathrm{EL}^{\mathbf{e}}(b^i) = n^\alpha \textstyle\sum_{i\geq 0}^m (b^{e_0}/a)^i \cdot \mathrm{EL}^{\mathbf{e}'}(b^i). \tag{20}$$

But (b^{e_0}/a) is > 1 $(= 1, < 1)$ depending on whether $e_0 > \alpha$ $(= \alpha, < \alpha)$. So the sum (20) is exponential-type (polynomial-type) when $e_0 \neq \alpha$, $(e_0 = \alpha)$. So:

$$T(n) = n^\alpha \cdot \Theta \left\{ \begin{array}{l} a^{-m}\mathrm{EL}^{\mathbf{e}}(b^m), \\ \sum_{i\geq 0}^m \mathrm{EL}^{\mathbf{e}'}(b^i), \\ 1 \end{array} \right. = \Theta \left\{ \begin{array}{ll} \mathrm{EL}^{\mathbf{e}}(n) & \text{if } e_0 > \alpha, \\ n^\alpha \cdot \sum_{i\geq 0}^m \mathrm{EL}^{\sigma(\mathbf{e}')}(i \lg b) & \text{if } e_0 = \alpha, \\ n^\alpha & \text{if } e_0 < \alpha. \end{array} \right. \tag{21}$$

We are done with the case $k \geq 0$ and $e_0 \neq \alpha$. For $k = 0$ and $e_0 = \alpha$, (21) gives

$$\begin{array}{rl} T(n) & = \Theta(n^\alpha \cdot \sum_{i\geq 0}^m \mathrm{EL}^{\sigma(\mathbf{e}')}(i \lg b)) \\ & = \Theta(n^\alpha \cdot \sum_{i\geq 0}^m \mathrm{EL}^{\sigma(\mathbf{e}')}(i)) \qquad (\lg b \text{ is const. in a poly.-type sum}) \\ & = \Theta(n^\alpha \cdot S^{\sigma(\mathbf{e}')}(m)) \qquad (\text{definition of } S^{\sigma(\mathbf{e}')}) \\ & = \Theta \left\{ \begin{array}{ll} n^\alpha \cdot \mathrm{EL}^{\sigma(\mathbf{e}')}(m) LL_h(2^m) & \text{if } \beta > -1 \\ n^\alpha & \text{if } \beta < -1 \end{array} \right. \qquad (\text{by Thm. 2}) \end{array}$$

In applying Thm. 2, we use the fact that $Ord(\sigma(\mathbf{e}')) \geq 0$ and the augmented cusp order of $\sigma(\mathbf{e}')$ is equal to $h-1$. The case $\beta < -1$ falls under CASE $(-)$. Case $\beta > -1$ falls under CASE (h) because $n^\alpha \cdot \mathrm{EL}^{\sigma(\mathbf{e}')}(m) LL_h(2^m) = \Theta(\mathrm{EL}^{\mathbf{e}}(n) LL_h(n))$ because $m = \Theta(\lg n)$. This proves Theorem A.

6 Real Induction and Proof of Theorem B

The principle of natural induction, or induction on $\mathbb{N}$, is well-known. To prove Theorem B, we need induction on $\mathbb{R}$, or **real induction**. Real induction is rarely discussed in the literature although it is needed in areas such as automatic correctness proofs of programs involving real numbers, timing logic [15], and in the programming language Real PCF [7]. We give a simple formulation here:

Principle of (Archimedean) Real Induction
*Let $P(x)$ be a real predicate. Then $P(x)$ is valid provided there exist real numbers x_1 (**cutoff constant**) and $\gamma > 0$ (**gap constant**) such that:*

(RB) Real Basis: *For all $x < x_1$, $P(x)$ holds.*
(RI) Real Induction: *For all $y \geq x_1$, if $(\forall x \leq y - \gamma)P(x)$ then $P(y)$.*

The derived predicate "$P^+(y) \equiv (\forall x \leq y - \gamma)P(x)$" in (RI) is called the **real induction hypothesis** (RIH). Thus (RI) says $P^+(y) \Rightarrow P(y)$. This principle is "Archimedean" because it exploits the Archimedean property of the reals: for any $x \in \mathbb{R}$, there is a smallest $n(x) \in \mathbb{N}$ such that $x \leq x_1 + n(x)\gamma$. Our principle is easily justified by a strong induction on the natural number $n(x)$. As an application of real induction, we can prove that the multiterm regularity condition (11) implies $d(n) = \Omega(n^{\alpha+\varepsilon})$ for some $\varepsilon > 0$.

¶*8. Proof of Theorem B.* We use the Principle of Real Induction. First we prove the real induction (RI) part of each case in our theorem:

CASE (+): This is the easiest case. The lower bound $T(n) = \Omega(d(n))$ is trivial. For the upper bound, we will show $T(n) \leq D_1 d(n)$ (ev.), for some D_1:

$$\begin{aligned} T(n) = d(n) + \textstyle\sum_{i=1}^{k} a_i T\left(\frac{n}{b_i}\right) &\leq d(n) + \textstyle\sum_{i=1}^{k} a_i D_1 d(n/b_i) && \text{(by RIH)} \\ &\leq d(n) + D_1 c d(n) && \text{(by regularity cond. (11))} \\ &\leq D_1 d(n) && \text{(choosing } D_1 \geq 1/(1-c)\text{)} \end{aligned}$$

CASE (0): Assume that $d(n) = n^\alpha \lg^\delta n$ for some $\delta > -1$. We first show $T(n) \leq D_1 d(n) \lg n$. We have, eventually,

$$\begin{aligned} T(n) &= d(n) + \textstyle\sum_{i=1}^{k} a_i T\left(\frac{n}{b_i}\right) \\ &\leq n^\alpha \lg^\delta n + \textstyle\sum_{i=1}^{k} a_i D_1 \left(\frac{n}{b_i}\right)^\alpha \lg^{\delta+1}\left(\frac{n}{b_i}\right) && \text{(by RIH)} \\ &= n^\alpha \lg^\delta n + D_1 n^\alpha \lg^{\delta+1} n \left[\textstyle\sum_{i=1}^{k} \frac{a_i}{b_i^\alpha}\left(1 - \frac{\lg b_i}{\lg n}\right)^{\delta+1}\right] \\ &= D_1 n^\alpha \lg^{\delta+1} n \left[\frac{1}{D_1 \lg n} + \textstyle\sum_{i=1}^{k} \frac{a_i}{b_i^\alpha}\left\{1 - (\delta+1)\frac{\lg b_i}{\lg n}(1 + o(1))\right\}\right] \\ &= D_1 n^\alpha \lg^{\delta+1} n \left[1 + \frac{1}{\lg n}\left\{\frac{1}{D_1} - (\delta+1)\textstyle\sum_{i=1}^{k} \frac{a_i \lg b_i}{b_i^\alpha}(1 + o(1))\right\}\right] \\ &\leq D_1 n^\alpha \lg^{\delta+1} n \end{aligned}$$

provided D_1 is sufficiently large to verify $\frac{1}{D_1} < (\delta+1)\sum_{i=1}^{k} \frac{a_i \lg b_i}{b_i^\alpha}$. Here the condition $\delta > -1$ is necessary. Similarly, we show the lower bound $T(n) \geq D_2 d(n) \lg n$ using the same derivation above, but with reversed inequalities. The provision is that D_2 is small enough to verify $\frac{1}{D_2} > (\delta+1)\sum_{i=1}^{k} \frac{a_i \lg b_i}{b_i^\alpha}$.

CASE (1): Assume that $d(n) = n^\alpha / \lg n$. We first show $T(n) \leq D_1 d(n) \lg\lg n$.

$$
\begin{aligned}
T(n) &= d(n) + \textstyle\sum_{i=1}^{k} a_i T\left(\frac{n}{b_i}\right) \\
&\le \frac{n^\alpha}{\lg n} + \textstyle\sum_{i=1}^{k} a_i D_1 \left(\frac{n}{b_i}\right)^\alpha \lg\lg\left(\frac{n}{b_i}\right) \qquad \text{(by RIH)} \\
&= D_1 n^\alpha \left[\frac{1}{D_1 \lg n} + \textstyle\sum_{i=1}^{k} \frac{a_i}{b_i^\alpha} \lg\left\{(\lg n)\left(1 - \frac{\lg b_i}{\lg n}\right)\right\}\right] \\
&= D_1 n^\alpha \left[\frac{1}{D_1 \lg n} + \textstyle\sum_{i=1}^{k} \frac{a_i}{b_i^\alpha} \left\{\lg\lg n + \lg\left(1 - \frac{\lg b_i}{\lg n}\right)\right\}\right] \\
&= D_1 n^\alpha \left[\lg\lg n + \frac{1}{D_1 \lg n} + \textstyle\sum_{i=1}^{k} \frac{a_i}{b_i^\alpha} \lg\left(1 - \frac{\lg b_i}{\lg n}\right)\right] \\
&= D_1 n^\alpha \left[\lg\lg n + \frac{1}{D_1 \lg n} - \textstyle\sum_{i=1}^{k} \frac{a_i}{b_i^\alpha} \frac{\lg b_i}{\lg n}(1 + o(1))\right] \\
&= D_1 n^\alpha \left[\lg\lg n + \frac{1}{\lg n}\left\{\frac{1}{D_1} - \textstyle\sum_{i=1}^{k} \frac{a_i \lg b_i}{b_i^\alpha}(1 + o(1))\right\}\right] \\
&\le D_1 n^\alpha \lg\lg n
\end{aligned}
$$

provided D_1 is large enough to verify $\frac{1}{D_1} < \sum_{i=1}^{k} \frac{a_i \lg b_i}{b_i^\alpha}$. Similarly, the lower bound $T(n) \ge D_2 n^\alpha \lg\lg n$ uses the above derivation with inequalities reversed, and D_2 small enough to verify $\frac{1}{D_2} > \sum_{i=1}^{k} \frac{a_i \lg b_i}{b_i^\alpha}$.

CASE (-1): This is the trickiest. By assumption, $0 \le d(n) \le n^\alpha \lg^\delta n$ (ev.) for some $\delta < -1$. To show $T(n) = O(n^\alpha)$, the hypothesis $T(n) \le D_1 n^\alpha$ will not do. Instead, use the stronger hypothesis $T(n) \le D_1 n^\alpha \left[1 - K \lg^{\delta+1} n\right]$ (ev.) for some $D_1, K > 0$. Eventually,

$$
\begin{aligned}
T(n) &= d(n) + \sum_{i=1}^{k} a_i T\left(\frac{n}{b_i}\right) \\
&\le n^\alpha \lg^\delta n + \textstyle\sum_{i=1}^{k} a_i D_1 \left(\frac{n}{b_i}\right)^\alpha \left[1 - K \lg^{\delta+1}\left(\frac{n}{b_i}\right)\right] \qquad \text{(by RIH)} \\
&= D_1 n^\alpha \left[\frac{\lg^\delta n}{D_1} + 1 - K \lg^{\delta+1} n \textstyle\sum_{i=1}^{k} \frac{a_i}{b_i^\alpha}\left(1 - \frac{\lg b_i}{\lg n}\right)^{\delta+1}\right] \\
&= D_1 n^\alpha \left[1 - K \lg^{\delta+1} n \left\{-\frac{1}{K D_1 \lg n} + \textstyle\sum_{i=1}^{k} \frac{a_i}{b_i^\alpha}\left(1 - (\delta+1)\frac{\lg b_i}{\lg n}\right)(1 + o(1))\right\}\right] \\
&= D_1 n^\alpha \left[1 - K \lg^{\delta+1} n \left\{1 - \frac{1}{\lg n}\left(\frac{1}{K D_1} + (\delta+1) \textstyle\sum_{i=1}^{k} \frac{a_i \lg b_i}{b_i^\alpha}(1 + o(1))\right)\right\}\right] \\
&\le D_1 n^\alpha \left[1 - K \lg^{\delta+1} n\right]
\end{aligned}
$$

provided KD_1 is small (sic) enough to verify $\frac{1}{KD_1} > -(\delta+1) \sum_{i=1}^{k} \frac{a_i \lg b_i}{b_i^\alpha}$ (recall $\delta < -1$). The introduction of K is crucial. For the lower bound, we also use a strengthened hypothesis, $T(n) \ge D_2 n^\alpha (1 + \lg^{\delta+1} n)$. The derivations is essentially the same, except inequalities are reversed. This completes the four cases.

We now provide the real bases (RB) for each of the above cases: first choose n_0 so that $d(n)$ is defined and the recurrence (2) for $T(n)$ holds nonvacuously ($\forall n \ge n_0$). Choose $\gamma = \gamma(n_0)$ as shown in the Appendix. Ensure the cutoff n_1 is $\ge n_0/\gamma$, so that RIH holds nonvacuously.

CASE $(+)$: Choose $n_1 = n_0/\gamma$ and ensure $D_1 \ge T(n)/d(n)$ for all $n \in [n_0, n_1]$. CASEs (0) and (1) are omitted in this abstract. CASE $(-)$: For upper bound, we first choose the product KD_1 to equal the reciprocal of $-(\delta+1) \sum_{i=1}^{k} \frac{3a_i \lg b_i}{2b_i^\alpha}$. Choose $n_1 \ge n_0/\gamma$ to be large enough so that the $o(1)$ term has absolute value

$< 1/2$, and for $n \geq n_1$, the function $f(n) = n^\alpha - (KD_1)\lg^{\delta+1} n$ is increasing and ≥ 1. Finally, choose D_1 as $\sup_{n_0 \leq n \leq n_1}\{T(n)/f(n)\}$. Note that $f(n) \leq D_1 n^\alpha(1 - K\lg^{\delta+1} n)$ and hence $T(n) \leq f(n) \leq D_1 n^\alpha(1 - K\lg^{\delta+1} n)$ for $n \in [n_0, n_1]$.

The proof of Theorem B is complete.

7 Conclusion

Cormen et al [5, p. 90] noted that some generalized master theorems are not easy to use. This echoes Karp's wish for "cookbook theorems" to recurrences [12]. That is the appeal of the standard master theorem. Our Theorem B has similar qualities. Although Theorem A is also cookbook, the generality of its driving function calls for some unavoidable deciphering of the notations. Further generalizations of Theorems A and B are possible: for instance, one could extend Theorem B to driving functions that are general EL functions. Another direction is to treat robustness issues of such solutions – we address this in the full paper.

Features that detract from cookbook property include bounds left in an integral form, tedious details involving integrality assumptions, and tracking of (essentially) arbitrary initial conditions. We have shown that much of this can be removed if we exploit Θ-robustness and embrace real recurrences wholeheartedly. Real induction is another useful tool that ought to be used more widely in this context. We feel our ideas are pedagogically sound. For instance, the summation rules for the various growth-types are easily taught in introductory algorithms. Indeed, our perspectives have developed out of classroom teaching.

References

1. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: Data Structures and Algorithms. Addison-Wesley, Reading (1983)
2. Akra, M., Bazzi, L.: On the solution of linear recurrences. Computational Optimizations and Applications 10(2), 195–210 (1998)
3. Bentley, J.L., Haken, D., Saxe, J.B.: A general method for solving divide-and-conquer recurrences. ACM SIGACT News 12(3), 36–44 (1980)
4. Brassard, G., Bratley, P.: Fundamentals of Algorithms. Prentice-Hall, Englewood Cliffs (1996)
5. Corman, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. MIT Press & McGraw-Hill Book Co. (2001)
6. Edelsbrunner, H., Welzl, E.: Halfplanar range search in linear space and $O(n^{0.695})$ query time. Info. Processing Letters 23, 289–293 (1986)
7. Escardó, M.H., Streicher, T.: Induction and recursion on the partial real line with applications to Real PCF. Theor. Computer Sci. 210(1), 121–157 (1999)
8. Gonnet, G.H.: Handbook of Algorithms and Data Structures. Addison-Wesley Pub. Co., London (1984)
9. Goursat, É.: A Course in Mathematical Analysis, vol. 1. Ginn & Co., Boston (1904); Trans. by Earle Raymod Hedrick. Available from Google books
10. Greene, D.H., Knuth, D.E.: Mathematics for the Analysis of Algorithms, 2nd edn. Birkhäuser, Basel (1982)

11. Kao, M.: Multiple-size divide-and-conquer recurrences. SIGACT News 28(2), 67–69 (1997); also Proc. 1996 Intl. Conf. on Algorithms, Natl. Sun Yat-Sen U., Taiwan, pp. 159–161
12. Karp, R.M.: Probabilistic recurrence relations. J. ACM 41(6), 1136–1150 (1994)
13. Knuth, D.E.: The Art of Computer Programming: Sorting and Searching, vol. 3. Addison-Wesley, Boston (1972)
14. Leighton, T.: Notes on better master theorems for divide-and-conquer recurrences (1996) (class notes)
15. Mahony, B.P., Hayes, I.J.: Using continuous real functions to model timed histories. In: 6th Australian Software Eng. Conf (ASWEC), pp. 257–270 (1991)
16. Paul Walton Purdom, J., Brown, C.A.: The Analysis of Algorithms. Holt, Rinehart and Winston, New York (1985)
17. Roura, S.: An improved master theorem for divide-and-conquer recurrences. In: Degano, P., Gorrieri, R., Marchetti-Spaccamela, A. (eds.) ICALP 1997. LNCS, vol. 1256, pp. 449–459. Springer, Heidelberg (1997)
18. Roura, S.: Improved master theorems for divide-and-conquer recurrences. J. ACM 48(2), 170–205 (2001)
19. Verma, R.M.: A general method and a master theorem for divide-and-conquer recurrences with applications. J. Algorithms 16, 67–79 (1994)
20. Wang, X., Fu, Q.: A frame for general divide-and-conquer recurrences. Info. Processing Letters 59, 45–51 (1996)
21. Yap, C.K.: Theory of real computation according to EGC. In: Hertling, P., Hoffmann, C.M., Luther, W., Revol, N. (eds.) Real Number Algorithms. LNCS, vol. 5045, pp. 193–237. Springer, Heidelberg (2008)