Graduate Program in Computer Science, Universidade Federal de Santa Catarina
INE410104 (Design and Analysis of Algorithms)

Solutions to the 1st Set of Exercises
Gustavo Zambonin, Matheus S. P. Bittencourt

1 Answers

1. (a) Let the functions that represent the time complexities of algorithms A and B be $f(n) = n^3$ and $g(n) = 128n^2$, respectively, with $n \in \mathbb{Z}^{\geq}$. Further, consider $t(n) = \frac{f(n)}{g(n)} \Rightarrow \frac{n}{128}$. By the ratio $t(n)$, algorithm B is faster than A when $n > 128$.

   (b) Let the old and new computers be named N_1 and N_2, respectively, and $f(n) = 2^n$. The equations $t_1 = \frac{2^{n_1}}{s}, t_2 = \frac{2^{n_2}}{20s}$ give the time spent to solve an instance of $f(n)$ on N_1 and N_2 with s instructions, with input sizes of n_1 and n_2. By setting $t_1 = t_2$, it is possible to figure out that the twenty-fold increase in the number of instructions does not help too much in solving $f(n)$:

$$\begin{aligned} \frac{2^{n_1}}{s} &= \frac{2^{n_2}}{20s} \\ 2^{n_2} &= 20 \cdot 2^{N_1} \\ n_2 &= n_1 + \lg 20 \\ n_2 &\approx n_1 + 4.32. \end{aligned}$$

   It is evident that the new input size had a modest growth. Indeed, the new computer will run slower if $n > 4$ with $n \in \mathbb{Z}^{\geq}$. This is due to the fact that the complexity of $f(n)$ grows exponentially, while in this case, the computational power grew linearly. Ergo, one cannot execute $f(2n)$ and expect comparable results on N_2, since $n_2 \ll 2n_1$.

2. (a) Let $t_1, \ldots, t_7$ be the number of seconds in each period of the header row and $s = 10^{10}$. The values in the table below represent approximately $n = f^{-1}(t_i s)$. The general idea for this exercise can be seen on Appendix A.

$f(n)$ / t	1 second	1 minute	1 hour	1 day	1 month	1 year	1 century
$\lg n$	$2^{1.00 \times 10^{10}}$	$2^{6.00 \times 10^{11}}$	$2^{3.60 \times 10^{13}}$	$2^{8.64 \times 10^{14}}$	$2^{2.59 \times 10^{16}}$	$2^{3.11 \times 10^{17}}$	$2^{3.11 \times 10^{19}}$
$\sqrt{n}$	1.00×10^{20}	3.60×10^{23}	1.30×10^{27}	7.46×10^{29}	6.72×10^{32}	9.67×10^{34}	9.67×10^{38}
n	1.00×10^{10}	6.00×10^{11}	3.60×10^{13}	8.64×10^{14}	2.59×10^{16}	3.11×10^{17}	3.11×10^{19}
$n \lg n$	3.52×10^{8}	1.76×10^{10}	9.06×10^{11}	1.96×10^{13}	5.30×10^{14}	5.94×10^{15}	5.28×10^{17}
n^2	1.00×10^{5}	7.75×10^{5}	6.00×10^{6}	2.94×10^{7}	1.61×10^{8}	5.58×10^{8}	5.58×10^{9}
n^3	2.15×10^{3}	8.43×10^{3}	3.30×10^{4}	9.52×10^{4}	2.96×10^{5}	6.78×10^{5}	3.14×10^{6}
2^n	3.32×10^{1}	3.91×10^{1}	4.50×10^{1}	4.96×10^{1}	5.45×10^{1}	5.81×10^{1}	6.48×10^{1}
$n!$	1.30×10^{1}	1.40×10^{1}	1.60×10^{1}	1.70×10^{1}	1.80×10^{1}	1.90×10^{1}	2.10×10^{1}

   (b) The "find-min" operation is fastest on the binary and Fibonacci heaps, since it is constant. The "delete-min" operation should be fastest on the Fibonacci heap, since an $\mathcal{O}(\log n)$ time complexity represents an amortised asymptotic upper bound, whereas the other heaps' $\Theta(\log n)$ is tight. Indeed, given a worst case scenario, both operations take exactly the same time to complete, but otherwise the Fibonacci heap does not take $\log n$ steps to finish the operation in average. The "insert" operation is fastest on the binomial and Fibonacci heaps. Thus, the Fibonacci heap is overall the fastest data structure when compared to binary and binomial heaps.

   (c) i. $(\log n)^{\log n} \in \Omega(\frac{n}{\log n})$, for $n_0 > 1$ and $c = 1$. This is additionally supported by $\lim_{n \to \infty} \frac{(\log n)^{\log n + 1}}{n} = \infty$.

ii. $n^2 \in \Omega(n^{\frac{1}{2}})$, for $n_0 > 1$ and $c = 1$. This is additionally supported by $\lim_{n\to\infty} \frac{n^2}{\sqrt{n}} = \infty$.

iii. $n! \in \mathcal{O}(n^{n-47})$, for $n_0 > 10^{100}$ and $c = 1$. This is additionally supported by $\lim_{n\to\infty} \frac{n!}{n^{n-47}} = 0$.

iv. $2^{(\lg n)^2} \in \Omega((\log n)^{\log n})$, for $n_0 > 1$ and $c = 1$. This is additionally supported by $\lim_{n\to\infty} \frac{2^{(\lg n)^2}}{(\log n)^{\log n}} = \infty$.

v. $n^2 \in \Theta(n + n^2)$, for $n_0 > 1, c_1 = \frac{1}{2}$ and $c_2 = 1$. This is additionally supported by $\lim_{n\to\infty} \frac{n^2}{n+n^2} = 1$.

vi. $n + (\log n)^2 \in \Theta(10n + \lg n)$, for $n_0 > 1, c_1 = 1$ and $c_2 = \frac{1}{11}$. This is additionally supported by $\lim_{n\to\infty} \frac{n+(\log n)^2}{10n+\lg n} = \frac{1}{10}$.

vii. $\log 2n \in \Theta(\log 3n)$, for $n_0 > 1, c_1 = \frac{1}{2}$ and $c_2 = 1$. This is additionally supported by $\lim_{n\to\infty} \frac{\log 2n}{\log 3n} = 1$.

viii. $n(\log n)^2 \in \mathcal{O}(\frac{n^2}{\log n})$, for $n_0 > 2$ and $c = 1$. This is additionally supported by $\lim_{n\to\infty} \frac{(\log n)^3}{n} = 0$.

3. (a) Let $P(n) = \sum_{i=1}^{n} i = \frac{n(n+1)}{2}$. $P(1)$ holds, since $\frac{1(1+1)}{2} = 1$. The inductive step is proved below, using the hypothesis that $P(k)$ is true.

$$\begin{aligned} P(k+1) &\stackrel{?}{=} P(k) + (k+1) \\ &\stackrel{?}{=} \tfrac{k(k+1)}{2} + (k+1) \\ &\stackrel{?}{=} \tfrac{k(k+1)+2(k+1)}{2} \\ &\stackrel{?}{=} \tfrac{(k+1)(k+2)}{2} \\ P(k+1) &= \tfrac{(k+1)((k+1)+1)}{2}. \end{aligned}$$

(b) Let $P(n) = \sum_{i=0}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$. $P(0)$ holds, since $\frac{0(0+1)(2\cdot 0+1)}{2} = 0$. The inductive step is proved below, using the hypothesis that $P(k)$ is true.

$$\begin{aligned} P(k+1) &\stackrel{?}{=} P(k) + (k+1)^2 \\ &\stackrel{?}{=} \tfrac{k(k+1)(2k+1)}{6} + (k+1)^2 \\ &\stackrel{?}{=} \tfrac{k(k+1)(2k+1)+6(k+1)^2}{6} \\ &\stackrel{?}{=} \tfrac{(k+1)(k(2k+1)+6(k+1))}{6} \\ &\stackrel{?}{=} \tfrac{(k+1)(2k^2+7k+6)}{6} \\ \tfrac{(k+1)((k+1)+1)(2(k+1)+1)}{6} &\stackrel{?}{=} \\ \tfrac{(k+1)(k+2)(2k+3)}{6} &= \tfrac{(k+1)(2k^2+7k+6)}{6}. \end{aligned}$$

(c) Let $P(n) = \sum_{i=1}^{n} (2i - 1) = n^2$. $P(1)$ holds, since $2 \cdot 1 - 1 = 1^2$. The inductive step is proved below, using the hypothesis that $P(k)$ is true.

$$\begin{aligned} P(k+1) &\stackrel{?}{=} P(k) + 2(k+1) - 1 \\ &\stackrel{?}{=} k^2 + 2(k+1) - 1 \\ &\stackrel{?}{=} k^2 + 2k + 1 \\ P(k+1) &= (k+1)^2. \end{aligned}$$

(d) Let $P(n) = \sum_{i=0}^{n} i^3 = \frac{n^2(n+1)^2}{4}$. $P(0)$ holds, since $\frac{0^2 \cdot (0+0)^2}{4} = 0$. The inductive step is proved below, using the hypothesis that $P(k)$ is true.

$$\begin{aligned} P(k+1) &\stackrel{?}{=} P(k) + (k+1)^3 \\ &\stackrel{?}{=} \tfrac{k^2(k+1)^2}{4} + (k+1)^3 \\ &\stackrel{?}{=} \tfrac{k^2(k+1)^2 + 4(k+1)^3}{4} \\ &\stackrel{?}{=} \tfrac{(k+1)^2(k^2+4(k+1))}{4} \\ &\stackrel{?}{=} \tfrac{(k+1)^2(k^2+4k+4)}{4} \\ P(k+1) &= \tfrac{(k+1)^2(k+2)^2}{4}. \end{aligned}$$

4. (a) Let

$$T(n) = \begin{cases} 0 & \text{if } n = 1, \\ T(n-1) + c & \text{if } n > 1 \end{cases}$$

for any constant c. By the iteration method,

$$\begin{aligned} T(n) &= c + T(n-1) \\ &= c + c + T(n-2) \\ &= c + c + c + T(n-3) \cdots \\ &= ic + T(n-i). \end{aligned}$$

Consider $T(n-i) \Leftrightarrow T(1)$. Thus, $n - i = 1$ and $i = n - 1$, which gives

$$\begin{aligned} T(n) &= (n-1)c + T(n-(n-1)) \\ &= nc - c + T(1) \\ T(n) &\in \Theta(n). \end{aligned}$$

To prove the final formula by weak induction, let $T(n) = c(n-1)$. $T(1)$ holds, since $c(1-1) = 0$. The inductive step is proved below, using the hypothesis that $T(n-1)$ is true.

$$\begin{aligned} T(n) &\stackrel{?}{=} T(n-1) + c \\ &\stackrel{?}{=} c((n-1)-1) + c \\ &\stackrel{?}{=} c(n-2) + c \\ T(n) &= c(n-1). \end{aligned}$$

(b) Let

$$T(n) = \begin{cases} 0 & \text{if } n = 1, \\ T(n-1) + 2^n & \text{if } n > 1. \end{cases}$$

By the iteration method,

$$\begin{aligned} T(n) &= 2^n + T(n-1) \\ &= 2^n + 2^{n-1} + T(n-2) \\ &= 2^n + 2^{n-1} + 2^{n-2} + T(n-3) \cdots \\ &= \sum_{i=2}^{n} 2^i \\ &= \tfrac{2^{n+1}-1}{2-1} - 2^0 - 2^1 \\ T(n) &\in \Theta(2^{n+1}). \end{aligned}$$

To prove the final formula by weak induction, let $T(n) = 2^{n+1} - 4$. $T(1)$ holds, since $2^{1+1} - 4 = 0$. The inductive step is proved below, using the hypothesis that $T(n-1)$ is true.

$$\begin{aligned} T(n) &\stackrel{?}{=} T(n-1) + 2^n \\ &\stackrel{?}{=} 2^{(n-1)+1} - 4 + 2^n \\ T(n) &= 2^{n+1} - 4. \end{aligned}$$

(c) Let

$$T(n) = \begin{cases} k & \text{if } n = 0, \\ cT(n-1) & \text{if } n > 0, \end{cases}$$

for any constants c, k. By the iteration method,

$$\begin{aligned} T(n) &= cT(n-1) \\ &= ccT(n-2) \\ &= cccT(n-3) \cdots \\ &= c^i T(n-i). \end{aligned}$$

Consider $T(n-i) \Leftrightarrow T(1)$. Thus, $n - i = 1$ and $i = n - 1$, which gives

$$\begin{aligned} T(n) &= c^{n-1} T(n-n) \\ &= c^{n-1} k \\ T(n) &\in \Theta(c^n). \end{aligned}$$

To prove the final formula by weak induction, let $T(n) = c^{n-1}k$. $T(1)$ holds, since $c^{1-1}k = k$. The inductive step is proved below, using the hypothesis that $T(n-1)$ is true.

$$\begin{aligned} T(n) &\stackrel{?}{=} cT(n-1) \\ &\stackrel{?}{=} cc^{n-2}k \\ T(n) &= c^{n-1}k. \end{aligned}$$

(d) Let

$$T(n) = \begin{cases} 1 & \text{if } n = 1, \\ 3T(\frac{n}{2}) + n & \text{if } n > 1. \end{cases}$$

By the iteration method,

$$\begin{aligned} T(n) &= n + 3T(\tfrac{n}{2}) \\ &= n + 3(\tfrac{n}{2} + 3T(\tfrac{n}{4})) \\ &= n + 3(\tfrac{n}{2} + 3(\tfrac{n}{4} + 3T(\tfrac{n}{8}))) \cdots \\ &= n \sum_{i=0}^{\lg n - 1} (\tfrac{3}{2})^i + 3^{\lg n} \\ &= n \left(\frac{\frac{3}{2}^{\lg n - 1 + 1} - 1}{\frac{3}{2} - 1} \right) + 3^{\lg n} \\ &= 2n(\tfrac{3}{2}^{\lg n} - 1) + 3^{\lg n} \\ &= 2nn^{\lg \frac{3}{2}} - 2n + n^{\lg 3} \\ &= 2n^{\lg 3 - 1 + 1} + n^{\lg 3} - 2n \\ T(n) &\in \Theta(n^{\lg 3}). \end{aligned}$$

To prove the final formula by weak induction, let $T(n) = 3n^{\lg 3} - 2n$. $P(1)$ holds, since $3 \cdot 1^{\lg 3} - 2 \cdot 1 = 1$. The inductive step is proved below, using the hypothesis that $T(\frac{n}{2})$ is true.

$$\begin{aligned} T(n) &\stackrel{?}{=} 3T(\tfrac{n}{2}) + n \\ &\stackrel{?}{=} 3(3\tfrac{n}{2}^{\lg 3} - \tfrac{2n}{2}) + n \\ &\stackrel{?}{=} 9\tfrac{n}{2}^{\lg 3} - 2n \\ &\stackrel{?}{=} 9 \cdot 2^{-\lg 3} \cdot n^{\lg 3} - 2n \\ T(n) &= 3n^{\lg 3} - 2n. \end{aligned}$$

5. (a) Let $T(n) = T(\frac{n}{2}) + \Theta(1)$, with $a = 1, b = 2$ and $f(n) \in \Theta(1)$. Since $n^{\lg 1} = 1$ and $f(n) \in \Theta(1)$, the second case may be applied. Thus, $T(n) \in \Theta(\lg n)$.

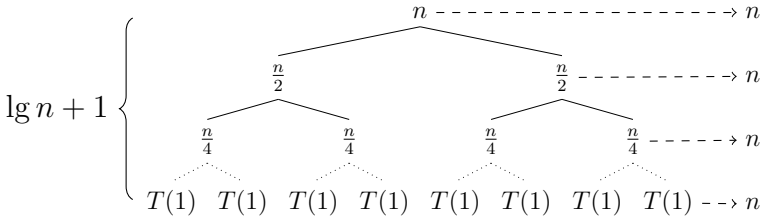
   (b) Let $T(n) = T(\frac{9n}{10}) + n$, with $a = 1, b = \frac{10}{9}$ and $f(n) = n$. It occurs that $n^{\log_{\frac{9}{10}} 1} = 1$, $f(n) \in \Omega(n^{\log_{\frac{10}{9}} 1+\epsilon})$ and $\frac{9n}{10} \leq cn$ with *e.g.* $\epsilon < 1$ and $c = \frac{10}{11}$. Thus, the third case may be applied, giving $T(n) \in \Theta(n)$.

   (c) Let $T(n) = 16T(\frac{n}{4}) + n^2$, with $a = 16, b = 4$ and $f(n) = n^2$. Since $n^{\log_4 16} = n^2$ and $f(n) \in \Theta(n^2)$, the second case may be applied. Thus, $T(n) \in \Theta(n^2 \lg n)$.

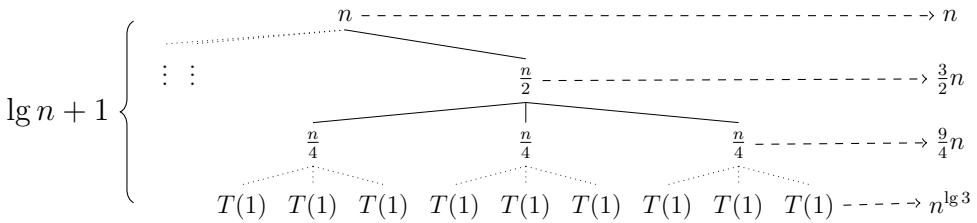
   (d) Let $T(n) = 7T(\frac{n}{3}) + n^2$, with $a = 7, b = 3$ and $f(n) = n^2$. It occurs that $n^{\log_3 7} \approx n^{1.77}$, $f(n) \in \Omega(n^{1.77+\epsilon})$ and $\frac{7n^2}{9} \leq cn^2$ with *e.g.* $\epsilon < 2 - \log_3 7$ and $c = \frac{8}{9}$. Thus, the third case may be applied, giving $T(n) \in \Theta(n^2)$.

6. Dashed arrows represent the total cost of each level in the tree. Dotted edges mean the natural expansion of the sub-trees. When applicable, the floor function is disregarded for simplicity.

   (a) Let $T(n) = 2T(\frac{n}{2}) + \Theta(n)$. By the recursion-tree method, $T(n) = \sum_{i=0}^{\lg n - 1} n + n$. Thus, $T(n) \in \Theta(n \lg n)$.

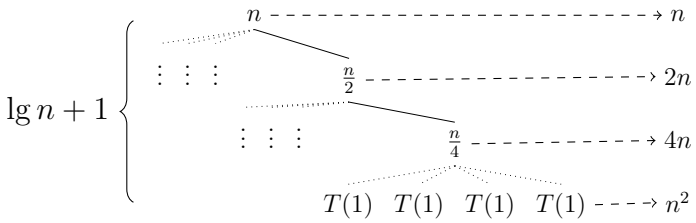


   (b) Let $T(n) = 3T(\lfloor \frac{n}{2} \rfloor) + \Theta(n)$. By the recursion-tree method, $T(n) = n \sum_{i=0}^{\lg n - 1} (\frac{3}{2})^i + n^{\lg 3}$. As per Exercise 4d, $T(n) \in \Theta(n^{\lg 3})$.



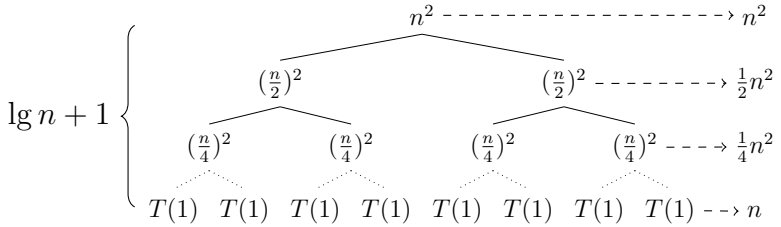
(c) Let $T(n) = 4T(\lfloor \frac{n}{2} \rfloor) + \Theta(n)$. By the recursion-tree method,

$$\begin{aligned} T(n) &= n \sum_{i=0}^{\lg n - 1} 2^i + n^2 \\ &= n \frac{2^{\lg n - 1 + 1} - 1}{2 - 1} + n^2 \\ &= n(2^{\lg n} - 1) + n^2 \\ &= n^2 + n n^{\lg 2} - n \\ &= 2n^2 - n \\ T(n) &\in \Theta(n^2). \end{aligned}$$



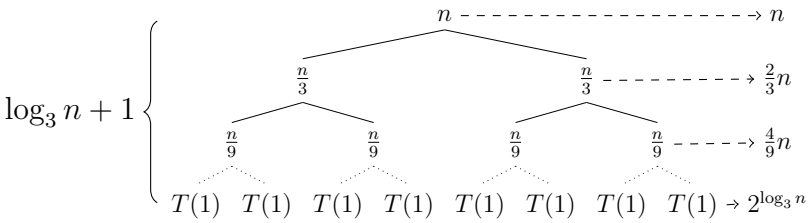
(d) Let $T(n) = 2T(\frac{n}{2}) + \Theta(n^2)$. By the recursion-tree method,

$$\begin{aligned} T(n) &= n^2 \sum_{i=0}^{\lg n - 1} 2^{-i} + n \\ &= n^2 \frac{2^{-(\lg n - 1 + 1)} - 1}{\frac{1}{2} - 1} + n \\ &= -2n^2(2^{-\lg n} - 1) + n \\ &= 2n^2(1 - n^{-\lg 2}) + n \\ &= 2n^2 - \frac{2n^2}{n} + n \\ &= 2n^2 - n \\ T(n) &\in \Theta(n^2). \end{aligned}$$



(e) Let $T(n) = 2T(\frac{n}{3}) + \Theta(n)$. By the recursion-tree method,

$$\begin{aligned}
T(n) &= n \sum_{i=0}^{\log_3 n - 1} (\tfrac{2}{3})^i + 2^{\log_3 n} \\
&= n \frac{\frac{2}{3}^{\log_3 n - 1 + 1} - 1}{\frac{2}{3} - 1} + 2^{\log_3 n} \\
&= -3n(\tfrac{2}{3}^{\log_3 n} - 1) + 2^{\log_3 n} \\
&= -3n(n^{\log_3 \frac{2}{3}} - 1) + n^{\log_3 2} \\
&= -3n(n^{\log_3 2 - 1} - 1) + n^{\log_3 2} \\
&= -3n(n^{\log_3 2 - 1} - 1) + n^{\log_3 2} \\
&= -3n^{\log_3 2} + 3n + n^{\log_3 2} \\
&= 3n - 2n^{\log_3 2} \\
T(n) &\in \Theta(n).
\end{aligned}$$



7. As per the analysis presented below, the most efficient algorithm is 7a.

   (a) Let $T(n) = 5T(\frac{n}{2}) + \mathcal{O}(n)$. By the master method, $a = 5, b = 2$ and $f(n) \in \mathcal{O}(n)$. It occurs that $n^{\lg 5} \approx n^{2.32}$, $f(n) \in \mathcal{O}(n^{2.32-\epsilon})$ with *e.g.* $\epsilon > \lg 5 - 1$. Thus, the first case may be applied, giving $T(n) \in \Theta(n^{\lg 5})$.

   (b) Let $T(n) = 2T(n-1) + \mathcal{O}(1)$. The recurrence is a special case of Exercise 4c, thus giving $T(n) \in \Theta(2^n)$.

   (c) Let $T(n) = 9T(\frac{n}{3}) + \mathcal{O}(n^2)$. By the master method, $a = 9, b = 3$ and $f(n) \in \mathcal{O}(n^2)$. Since $n^{\log_3 9} = n^2$ and $f(n) \in \Theta(n^2)$, the second case may be applied. Thus, $T(n) \in \Theta(n^2 \lg n)$.

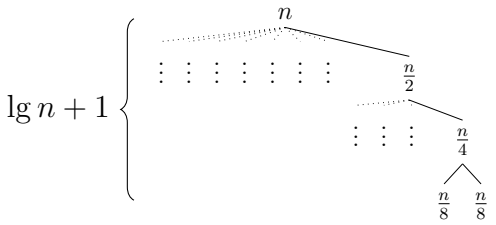
8. (a) Consider that I is a set of 2-uples $I_i : (s, e), i \in \{1, \ldots, n\}$ such that $s, e \in \mathbb{Z}^+, s \leq e$. To order such a set, the condition $I_{i-1}.e < I_i.s$ has to be met for all $i \in \{2, \ldots, n\}$. To figure out the recurrence for this algorithm, the time complexities of lines 3 and 5 have to be set. Observe that the `argmin` function cannot determine the minimum element in sub-linear time, since all intervals have to be compared without the aid of an external, pre-built data structure. Analogously, to remove all overlapping intervals in line 5, every element in the set needs to be compared with the I_j target. Thus, lines 3 and 5 are linear in n. Finally, note the worst case scenario for this algorithm, in the form of an already ordered I. Line 5 will thus remove only I_j at every recursive step, and the length of n will be reduced by a single element. This means that $T(n) = T(n-1) + 2n$. By the iteration method,

$$\begin{aligned}
T(n) &= T(n-1) + 2n \\
&= 2n + 2n + T(n-2) \\
&= 2n + 2n + 2n + T(n-3) \\
&= 2in + T(n-i).
\end{aligned}$$

Consider $T(n-i) \Leftrightarrow T(1)$. Thus, $n - i = 1$ and $i = n - 1$, which gives

$$\begin{aligned} T(n) &= 2(n-1)n + T(n-(n-1)) \\ &= 2n^2 - 2n + T(1) \\ &= 2(n^2 - n) \\ T(n) &\in \Theta(n^2). \end{aligned}$$

(b) Due to its recursive nature, it is useful to observe the tree generated by some instance of $f(n)$, where $n = 2^k, k \in \mathbb{N}$. It is a perfect tree with $\lg n$ levels, since the recursive step halves the number of children, stopping when $n = 2$. For instance, if $n = 8$, the following tree is created.



The costs are given shortly after. The recurrence is solved by the iteration method.

$$\begin{aligned} T(1) &= 0 \\ T(2) &= 2 + 2 \cdot 0 && \Rightarrow 2 + 2T(1) = 2 \\ T(4) &= 4 + 4 \cdot (2 + 2 \cdot 0) && \Rightarrow 4 + 4T(2) = 12 \\ T(8) &= 8 + 8 \cdot (4 + 4 \cdot (2 + 2 \cdot 0)) && \Rightarrow 8 + 8T(4) = 108 \\ &\vdots \\ T(n) &= nT(\tfrac{n}{2}) + n \\ &= n + n \cdot (\tfrac{n}{2} + \tfrac{n}{2} \cdot T(\tfrac{n}{4})) \\ &= n + n \cdot (\tfrac{n}{2} + \tfrac{n}{2} \cdot (\tfrac{n}{4} + \tfrac{n}{4} \cdot T(\tfrac{n}{8}))) \cdots \\ &= n + \tfrac{n^2}{2} + \tfrac{n^3}{8} + \cdots \\ &= \sum_{i=1}^{\lg n} \frac{n^i}{2^{\sum_{j=0}^{i-1} j}} \\ &= \sum_{i=1}^{\lg n} \frac{n^i}{2^{\frac{1}{2}i(i-1)}}. \end{aligned}$$

The maximum value $i = \lg n$ makes the dominant term in the summation $n^{\lg n} 2^{-\frac{1}{2}\lg n(\lg n - 1)}$. Assuming that the input size is a power of 2 and substituting $n = 2^k, k \in \mathbb{N}$,

$$\begin{aligned} T(n) &= 2^{k^{\lg 2^k}} 2^{-\frac{1}{2}\lg 2^k (\lg 2^k - 1)} \\ &= 2^{k^k} 2^{-\frac{1}{2}k(k-1)} \\ &= 2^{k^2 - \frac{1}{2}k(k-1)} \\ &= 2^{k^2 - \frac{1}{2}k^2 + \frac{1}{2}k} \\ &= 2^{k(k - \frac{1}{2}k + \frac{1}{2})} \\ &= 2^{k(\frac{1}{2}k + \frac{1}{2})}. \end{aligned}$$

With $k = \lg n$, then $T(n) \in \Theta(n^{\frac{1}{2}(\lg n + 1)})$.

A Rationale for Exercise 2a

Define $t_1, \ldots, t_7$ and $s = 10^{10}$ as above. The equation $t_i = \frac{f(n)}{s}$, or its simpler form $n = f^{-1}(t_i s)$, has to be solved in order to find out the maximum n allowed to execute for a period of time using $f(n)$. Deducing f^{-1} is not trivial for some of the rows. In the case of $n \lg n = c$ for some constant c, the Lambert W-function is used:

$$\begin{aligned} n \lg n &= c \\ n \ln n &= c \ln 2 \\ \mathrm{e}^{\ln n} \ln n &= c \ln 2 \\ \ln n &= W(c \ln 2) \\ n &= \mathrm{e}^{W(c \ln 2)}. \end{aligned}$$

Moreover, to invert the factorial function, some definitions from [Can] are used to construct a strict inverse if the input is an integer. Consider the single positive zero of the digamma function $\psi_0 \approx 1.46163214496836$ and the gamma function $\Gamma(x)$. The inverse factorial function is equal to

$$\iota(n) = \left\lfloor \frac{\ln \frac{n+c}{\sqrt{2\pi}} - 1}{W(\frac{1}{e} \ln \frac{n+c}{\sqrt{2\pi}} - 1)} + \frac{1}{2} \right\rfloor - 1, \qquad c = \frac{1}{e}\sqrt{2\pi} - \Gamma(\psi_0).$$

The table below shows the exact values for all $n = f^{-1}(t_i s)$.

$f(n)$ / t	1 second	1 minute	1 hour	1 day	1 month	1 year	1 century
$\lg n$	$2^{t_1 s}$	$2^{t_2 s}$	$2^{t_3 s}$	$2^{t_4 s}$	$2^{t_5 s}$	$2^{t_6 s}$	$2^{t_7 s}$
$\sqrt{n}$	$(t_1 s)^2$	$(t_2 s)^2$	$(t_3 s)^2$	$(t_4 s)^2$	$(t_5 s)^2$	$(t_6 s)^2$	$(t_7 s)^2$
n	$t_1 s$	$t_2 s$	$t_3 s$	$t_4 s$	$t_5 s$	$t_6 s$	$t_7 s$
$n \lg n$	$\mathrm{e}^{W(t_1 s \ln 2)}$	$\mathrm{e}^{W(t_2 s \ln 2)}$	$\mathrm{e}^{W(t_3 s \ln 2)}$	$\mathrm{e}^{W(t_4 s \ln 2)}$	$\mathrm{e}^{W(t_5 s \ln 2)}$	$\mathrm{e}^{W(t_6 s \ln 2)}$	$\mathrm{e}^{W(t_7 s \ln 2)}$
n^2	$\sqrt{t_1 s}$	$\sqrt{t_2 s}$	$\sqrt{t_3 s}$	$\sqrt{t_4 s}$	$\sqrt{t_5 s}$	$\sqrt{t_6 s}$	$\sqrt{t_7 s}$
n^3	$\sqrt[3]{t_1 s}$	$\sqrt[3]{t_2 s}$	$\sqrt[3]{t_3 s}$	$\sqrt[3]{t_4 s}$	$\sqrt[3]{t_5 s}$	$\sqrt[3]{t_6 s}$	$\sqrt[3]{t_7 s}$
2^n	$\lg t_1 s$	$\lg t_2 s$	$\lg t_3 s$	$\lg t_4 s$	$\lg t_5 s$	$\lg t_6 s$	$\lg t_7 s$
$n!$	$\iota(t_1 s)$	$\iota(t_2 s)$	$\iota(t_3 s)$	$\iota(t_4 s)$	$\iota(t_5 s)$	$\iota(t_6 s)$	$\iota(t_7 s)$

References

[Can] D. W. Cantrell. Inverse gamma function. https://web.archive.org/web/20160402104317/http://mathforum.org/kb/message.jspa?messageID=342551. Last accessed on 19 apr. 2019.