Learning Objectives

By the end of this worksheet, you will:

- Analyse the average-case running time of an algorithm.

1. **Average-case analysis.** Consider the following algorithm that we studied a few weeks ago. The input is an array \( A \) of length \( n \), containing a list of \( n \) integers.

```python
def has_even(lst: List[int]) -> int:
    n = len(lst)
    for i in range(n):
        if lst[i] % 2 == 0:
            return i
    return -1
```

We proved that the worst-case running time of this algorithm is \( \Theta(n) \). In this problem we will analyse its **average-case** running time.

For this analysis, we will consider the sets of binary lists \( lst \) of length \( n \), for each \( n \in \mathbb{Z}^+ \). That is, \( lst \) is a list of \( n \) integers, where each integer is either 0 or 1.

(a) For each \( n \in \mathbb{Z}^+ \), let \( \mathcal{I}_n \) be the set of all binary lists of length \( n \). Find an expression (in terms of \( n \)) for \( |\mathcal{I}_n| \), the size of \( \mathcal{I}_n \).

    **Solution**

    The number of binary lists of length \( n \) is \( 2^n \), thus \( |\mathcal{I}_n| = 2^n \).
(b) For each $n \in \mathbb{Z}^+$ and each $i \in \{0, 1, \ldots, n - 1\}$, let $S_{n,i}$ denote the set of all binary lists $\text{lst}$ of length $n$ where the first 0 occurs in position $i$. More precisely, every list $\text{lst}$ in $S_{n,i}$ satisfies the following two properties:

(i) $\text{lst}[i] = 0$.

(ii) for all $j \in \mathbb{N}$, if $j < i$ then $\text{lst}[j] = 1$.

For each $i$, $0 \leq i \leq n$, find an expression for $|S_{n,i}|$.

**Solution**

$|S_{n,i}| = 2^{n-1-i}$.

(c) Also, for each $n \in \mathbb{Z}^+$, let $S_{n,n}$ denote the set of binary lists of length $n$ that do not contain a 0 at all. Find an expression for $|S_{n,n}|$.

**Solution**

$|S_{n,n}| = 1$ (only one binary list of length $n$ has no 0's: the list containing all 1's).

(d) Give a brief argument (informal proof) that for every $n \in \mathbb{Z}^+$, each binary list of length $n$ is in exactly one set $S_{n,i}$ (for some $i \in \{0, 1, \ldots, n\}$). That is, you're arguing that $S_{n,0}, S_{n,1}, \ldots, S_{n,n}$ form a partition of $\mathcal{I}_n$.

**Solution**

For each input, either it contains a 0 or it doesn’t. If it doesn’t then it is (the single input) in $S_{n,n}$. If it does, then we partition these inputs according to the smallest location $i \leq n-1$ where $\text{lst}[i] = 0$: if an input has its first 0 in $\text{lst}[i]$, then it is in the set $S_{n,i}$.
(e) Assume that we calculate the running time of has_even by counting just the costs of Lines 4 and 7. Find an exact expression for the average runtime of this algorithm for this input set \( I_n \), in terms of \( n \).
You should get a summation; do not simplify the summation in this part.

**Solution**

For \( i \in \{0, 1, \ldots, n-1\} \), every input in \( S_{n,i} \) causes the loop to iterate exactly \( i + 1 \) times, so Line 4 executes \( i + 1 \) times, and then the early return occurs. So in this case, a total of \( i + 1 \) steps occur.
Every input in \( S_{n,n} \) causes the loop to iterate exactly \( n \) times, so Line 4 executes \( n \) times, and then Line 7 executes, for a total of \( n + 1 \) steps.
So the overall average runtime is:

\[
\sum_{i=0}^{n} |S_{n,i}| \times (i + 1) = \frac{\left( \sum_{i=0}^{n-1} |S_{n,i}| \times (i + 1) \right) + |S_{n,n}| \times (n + 1)}{2^n}
\]

\[
= \frac{\left( \sum_{i=0}^{n-1} 2^{n-i} \times (i + 1) \right) + 1 \times (n + 1)}{2^n}
\]

\[
= \frac{\left( \sum_{i'=1}^{n} 2^{n-i'} \times i' \right) + n + 1}{2^n} \quad \text{(change of variable \( i' = i + 1 \))}
\]

\[
= \left( \sum_{i'=1}^{n} \left( \frac{1}{2} \right)^{i'} \times i' \right) + \frac{n + 1}{2^n}
\]

(f) Show that the average running time expression that you calculated is in \( O(1) \). You may use the fact that for all \( x \in \mathbb{R} \), if \( |x| < 1 \), then \( \sum_{i=1}^{\infty} ix^i = \frac{x}{(1-x)^2} \).

**Solution**

The average running time is \( \left( \sum_{i'=1}^{n} i'(1/2)^{i'} \right) + (n + 1)/2^n \). The second part is eventually less than 1, and by the formula given above, the first part is at most 2. Thus the expected runtime is \( O(1) \).