1 Introduction

The basic question in \textit{algorithm analysis} is:

\textbf{How “good” is my algorithm?}

This can be measured across many different dimensions.

- Time efficiency
- Space efficiency

2 Example

\textbf{Problem:} Array search

- \textbf{Input:} An array $A[0, \ldots, n-1]$ and a search key $K$.
- \textbf{Output:} An index $i$ such that $A[i] = K$, or $-1$ if there is no such index.

\textbf{Algorithm:}

\begin{verbatim}
SEQUENTIALSEARCH($A[0, \ldots, n-1], K$)
  \hspace{1em} i = 0
  \hspace{1em} \textbf{while} i \leq n \textbf{ and } A[i] \neq K \textbf{ do}
  \hspace{2em} i = i + 1
  \hspace{1em} \textbf{end while}
  \hspace{1em} \textbf{if} i < n \textbf{ then}
  \hspace{2em} \textbf{return} i
  \hspace{1em} \textbf{else}
  \hspace{2em} \textbf{return} -1
  \hspace{1em} \textbf{end if}
\end{verbatim}

3 Step 1

\textbf{Step 1:} Identify the \textit{input size}.

4 Step 2

\textbf{Step 2:} Identify the \textit{basic operation}.

Why focus on basic operations?

\[ T(n) \approx c_{op} C(n) \]
5  Step 3
Step 3: Count the basic operations.

Three choices:

- $C_{\text{worst}}(n)$: maximum over all inputs of size $n$
- $C_{\text{avg}}(n)$: average over all inputs of size $n$
- $C_{\text{best}}(n)$: minimum over all inputs of size $n$

6  Back to SEQUENTIAL SEARCH
Identify input size: Length of array. $n$ (...or $n + 1$ if you count $K$.)
Identify basic operation: Comparison between two array elements

Count of basic operations.

\[
\begin{align*}
C_{\text{worst}}(n) &= \\
C_{\text{best}}(n) &= \\
C_{\text{avg}}(n) &=
\end{align*}
\]

* ...if $K$ is in $A$ and all positions are equally likely.

7  Step 4
Step 4: Classify the order of growth.

Thought experiment: Suppose $C_{\text{worst}} = an^2$.
What happens if we double the input size?

\[
T(2n) =
\]

8  So what?

Conclusion: “Ignore” multiplicative constants.

Result: “Efficiency classes” a.k.a. “orders of growth”

Basic tool: Asymptotic notations: big-$O$ big-$\Omega$ big-$\Theta$