

Introduction to Algorithms: a Workbook

Howard A. Blair

¹Syracuse University
blair@ecs.syr.edu

Reference for Turing Machines

We will consider a formal definition of a Turing machine according to:

John Hopcroft and Jeffrey Ullman, (1979). **Introduction to Automata Theory, Languages and Computation (1st ed.)**
Addison-Wesley, Reading Mass. ISBN 0-201-02988-X

Definition of a Turing Machine

A Turing machine M is a 7-tuple: $\langle Q, \Gamma, b, \Sigma, \delta, q_0, F \rangle$ where

- Q is a finite set of states. Uh-huh. What's a state?
- Γ is a finite set of the tape alphabet/symbols. What's a tape, what's an alphabet, what's a symbol?
- $b \in \Gamma$ is called *blank*. Note: b is a formal parameter in this definition. We can designate any symbol we want in an application to be the blank. The blank is the only symbol allowed to occur on the tape infinitely often at any step during the computation. By the way, what's a computation?
- $\Sigma \subseteq (\Gamma \setminus \{b\})$. Σ is called the set of input symbols.
- $\delta: Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}$ is a partial function called the transition function, where L is left shift, R is right shift.
- $q_0 \in Q$. q_0 is called the *initial* state.
- $F \subseteq Q$. F is called the set of *final* (or, synonymously, *accepting*) states.

Example of a Turing machine

Exercise: Here is the first of the 5-state Busy Beaver Turing machines at <http://ironphoenix.org/tril/tm>

```

1, _ 2, 1, >
1, 1 3, _, <
2, _ 3, 1, >
2, 1 4, 1, >
3, _ 1, 1, <
3, 1 2, _, >
4, _ 5, _, >
4, 1 H, 1, >
5, _ 3, 1, <
5, 1 1, 1, >

```

Identify the components of the 7-tuple for this example in Hopcroft and Ullman's definition.

Alternative definition of a Turing machine

A Turing machine M is a 7-tuple: $\langle Q, \Gamma, b, \Sigma, \delta, q_0, F \rangle$ where

- Q is a finite set. The members of Q are called *states*.
- Γ is a finite set. Γ is called the *tape alphabet*. The members of Γ are called *symbols*.
- The *tape* of M is the set of integers \mathbb{Z} .
- $b \in \Gamma$ is called *blank*.
- $\Sigma \subseteq (\Gamma \setminus \{b\})$. Σ is called the set of *input symbols*.
- $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ is a partial function called the transition function, where
 - $L: \mathbb{Z} \rightarrow \mathbb{Z}$ by $L(n) = n - 1$. L is called left-shift.
 - $R: \mathbb{Z} \rightarrow \mathbb{Z}$ by $L(n) = n + 1$. R is called right-shift.
- $q_0 \in Q$. q_0 is called the *initial state*.
- $F \subseteq Q$. F is called the set of *final* (or, synonymously, *accepting*) states.

Big Θ

- **Notation:**

$$\mathbb{N} = \{0, 1, 2, 3, \dots\}$$

$$\mathbb{N}^+ = \{1, 2, 3, 4, \dots\}$$

- Let $f : \mathbb{N} \rightarrow \mathbb{N}$ and $g : \mathbb{N} \rightarrow \mathbb{N}$ be *asymptotically positive* - be patient, ignore for now and later we will explain. Then

$$\Theta(g) = \left\{ f \left| \begin{array}{l} \text{for some constants} \\ c_{\text{lower}}, c_{\text{upper}} > 0, \\ \text{there is some natural number} \\ \text{threshold } m \\ \text{such that for all } n \geq m \\ \text{(i.e. for all } n \text{ beyond the} \\ \text{threshold),} \\ \\ c_{\text{lower}} \leq \frac{f(n)}{g(n)} \leq c_{\text{upper}} \end{array} \right. \right\}$$

Big Θ - Examples

- Let $f : \mathbb{N} \rightarrow \mathbb{N}$ and $g : \mathbb{N} \rightarrow \mathbb{N}$. Then

$$\Theta(n) = \left\{ f \left| \begin{array}{l} \text{for some constants} \\ c_{\text{lower}}, c_{\text{upper}} > 0, \\ \text{there is some natural number} \\ \text{threshold } m \\ \text{such that for all } n > m \\ \text{(i.e. for all } n \text{ beyond the} \\ \text{threshold),} \\ c_{\text{lower}} \leq \frac{f(n)}{n} \leq c_{\text{upper}} \end{array} \right. \right\}$$

Big Θ - Examples

- Let $f : \mathbb{N} \rightarrow \mathbb{N}$ and $g : \mathbb{N} \rightarrow \mathbb{N}$. Then

$$\Theta(n^2) = \left\{ f \left| \begin{array}{l} \text{for some constants} \\ c_{\text{lower}}, c_{\text{upper}} > 0, \\ \text{there is some natural number} \\ \text{threshold } m \\ \text{such that for all } n > m \\ \text{(i.e. for all } n \text{ beyond the} \\ \text{threshold),} \\ c_{\text{lower}} \leq \frac{f(n)}{n^2} \leq c_{\text{upper}} \end{array} \right. \right\}$$

“Abusing Notation” - common practice

- When we wrote $\Theta(n^2)$ in the preceding example, we were referring to the function that squares natural numbers. In other words we had let $g : \mathbb{N} \rightarrow \mathbb{N}$ specifically be the function that satisfies

$$g(n) = n^2$$

- Instead of writing all of this when we want to mention the squaring function, we just reasonably refer to the squaring function by writing n^2 . Logically, this is a mistake; specifically, it is a type-error. When we write n , we are referring to a number, not a function. Similarly when we write n^2 . It's usually clear what we mean from the context, but in more advanced areas of mathematical analysis this casual approach to notation can get you into trouble and be a minor barrier to understanding.

“Abusing Notation” - common practice

- This casual abuse of notation is also confusing to beginning students in discrete math, because they absorbed some implicit rules about the notation for functions, mainly from the basic calculus. When this notation is carried over into discrete math contexts where types are important, the casual notation is again a barrier to understanding.

“Abusing Notation” - common practice

- Abusing notation: The common practice gets worse in the context of asymptotic growth. $\Theta(g)$ is a *set* of functions. Suppose we analyze an algorithm \mathcal{A} and conclude that its runtime is $\Theta(n \log n)$. What we mean by that is the function $\text{runtime}_{\mathcal{A}}$ defined by

$\text{runtime}_{\mathcal{A}}(n) =$ worst case number of steps required by \mathcal{A} to stop,
if given an input of size n

is in $\Theta(n \log n)$. i.e.

$$\text{runtime}_{\mathcal{A}}(n) \in \Theta(n \log n)$$

But many authors write

$$\text{runtime}_{\mathcal{A}}(n) = \Theta(n \log n)$$

- This raises the matter of how to measure input size - more on that later.

Examples

- **Example:** Suppose f is given by (i.e satisfies):

$$f(n) = 100n^2 + 8000n + 97.$$

Then

$$f \text{ is in } \Theta(n^2)$$

Why? What are the lower and upper constants?

- Suppose f is given by

$$f(n) = 8000n + 97.$$

Then

$$f \text{ is not in } \Theta(n^2)$$

Why?

The Limit Rule for Big- Θ

- Suppose that

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = c$$

where $0 \leq c \leq \infty$. Then

- If $0 < c < \infty$, then f is in $\Theta(g)$.
- If $c = 0$ or $c = \infty$, then f is not in $\Theta(g)$.
- Compute the limits associated with f in the two preceding examples.
- What if the limit doesn't exist? **Example:**

$$\frac{\lfloor 2 + \sin n \rfloor}{\lfloor 2 + \cos n \rfloor}$$

Splitting Big- Θ : Big-O and Big- Ω

$$O(g) = \left\{ f \left| \begin{array}{l} \text{for some constants} \\ c_{\text{lower}}, c_{\text{upper}} > 0, \\ \text{there is some natural number} \\ \text{threshold } m \\ \text{such that for all } n > m \\ \text{(i.e. for all } n \text{ beyond the} \\ \text{threshold),} \\ \frac{f(n)}{g(n)} \leq c_{\text{upper}} \end{array} \right. \right\}$$

Splitting Big- Θ : Big-O and Big- Ω

$$\Omega(g) = \left\{ f \left| \begin{array}{l} \text{for some constants} \\ c_{\text{lower}}, c_{\text{upper}} > 0, \\ \text{there is some natural number} \\ \text{threshold } m \\ \text{such that for all } n > m \\ \text{(i.e. for all } n \text{ beyond the} \\ \text{threshold),} \\ c_{\text{lower}} \leq \frac{f(n)}{g(n)} \end{array} \right. \right\}$$