# Chapter 5

# Reducibility, Completeness, and Closure Under Reductions

## 5.1. Log Space Reducibility

In order to introduce the notion of reducibility, we need to generalize deterministic Turing machines from machines that accept languages to machines that compute functions. To allow our machines to compute functions, we add a one-way, write-only output tape, initially blank, to the deterministic Turing machine model. The contents of this tape are not included in the space bound of the machine.

**Definition 5.1:** A deterministic Turing machine $M$ *computes a partial function* $f$ if and only if for every input $x$, if $x$ is in the domain of $f$, then $M$ halts with $f(x)$ written on the output tape, else $M$ does not halt.

The next definition introduces the notion of "reducibility", which is the means for comparing the complexities of two problems.

**Definition 5.2:** Let $A$ and $B$ be languages. We say $A$ is *log space, many-one reducible* to $B$ (denoted $A \leq_m^{\mathcal{L}} B$) if and only if there is a (total) function $f$ computable by a deterministic Turing machine $M$ in space $O(\log n)$ such that $x \in A$ if and only if $f(x) \in B$, for all $x$. In such a case we say $M$ *reduces* $A$ to $B$.

This says that recognizing $A$ is no harder than computing $f$ (which, by the definition, is "easy") plus recognizing $B$. This definition is stricter than other common definitions of reducibility in two ways. Firstly, it is common to allow the reducing machine polynomial time rather than only $O(\log n)$ space. Secondly, one often encounters "Turing reducibility" of $A$ to $B$, which means that you can solve problem $A$ if given arbitrary access to a subroutine that solves problem $B$. Many-one reducibility is stricter in the sense that, to solve $A$, the reduction can only call the subroutine for $B$ once at the very end, and must return the value that invocation returns.

Next we show that reducibility is transitive, which is its most important property.

**Lemma 5.3:** If $A \leq_m^{\mathcal{L}} B$ and $B \leq_m^{\mathcal{L}} C$, then $A \leq_m^{\mathcal{L}} C$.

**Proof:** Let $M$ reduce $A$ to $B$ and $N$ reduce $B$ to $C$. On input $x$ we want to output $g(f(x))$, where $M$ computes $f$ and $N$ computes $g$. Note that $g(f(x))$ is the correct output, since $x \in A$ if and only if $f(x) \in B$ if and only if $g(f(x)) \in C$. The problem is that the length of the intermediate value $f(x)$ could be polynomial in $n = |x|$ (for example, if $M$ outputs a symbol at each step), and the reduction does not have that much space.

Instead, imagine running $N$ on (nonexistent) input $f(x)$. For any $i$, whenever $N$ needs the $i$th symbol of $f(x)$, simulate $M$ on $x$ until it produces the $i$th output symbol (throwing away the previous $i-1$ symbols rather than writing them). The space required for this is $O(\log |x|) = O(\log n)$ for $M$ and $O(\log |f(x)|) = O(\log(n^{O(1)})) = O(\log n)$ for $N$ and $i$. □

## 5.2. Hardness, Completeness, and Closure Under Reductions

**Definition 5.4:** Let $\mathcal{C}$ be a set of languages. A language $B$ is $\leq_m^{\mathcal{L}}$*–hard* for $\mathcal{C}$ if and only if for every $A \in \mathcal{C}$, $A \leq_m^{\mathcal{L}} B$. $B$ is $\leq_m^{\mathcal{L}}$*–complete* for $\mathcal{C}$ if and only if $B \in \mathcal{C}$ and $B$ is $\leq_m^{\mathcal{L}}$*–hard* for $\mathcal{C}$. [1]

**Definition 5.5:** Let $\mathcal{C}$ be a set of languages. We say $\mathcal{C}$ is *closed under* $\leq_m^{\mathcal{L}}$ if and only if $A \leq_m^{\mathcal{L}} B$ and $B \in \mathcal{C}$ implies $A \in \mathcal{C}$.

The following proposition explains why complete problems are so important: if you understand the complexity of any one complete problem for a class, you understand the complexity of the entire class.

**Proposition 5.6:** Let $\mathcal{C}$ and $\mathcal{D}$ be sets of languages. Suppose $\mathcal{C}$ is closed under $\leq_m^{\mathcal{L}}$ and $B$ is $\leq_m^{\mathcal{L}}$–complete for $\mathcal{D}$. Then $B \in \mathcal{C}$ if and only if $\mathcal{D} \subseteq \mathcal{C}$.

**Proof:**

"If" clause: Suppose $\mathcal{D} \subseteq \mathcal{C}$. $B \in \mathcal{D}$ since B is $\leq_m^{\mathcal{L}}$–complete for $\mathcal{D}$, so $B \in \mathcal{C}$.

"Only if" clause: Suppose $B \in \mathcal{C}$. Let $A$ be an arbitrary language in $\mathcal{D}$. Since $B$ is $\leq_m^{\mathcal{L}}$–hard for $\mathcal{D}$, $A \leq_m^{\mathcal{L}} B$. Then $A \in \mathcal{C}$ since $\mathcal{C}$ is closed under $\leq_m^{\mathcal{L}}$. □

The next proposition shows how to use one hard problem to get more.

**Proposition 5.7:** Let $\mathcal{C}$ be a set of languages. Suppose that $A$ is $\leq_m^{\mathcal{L}}$–hard for $\mathcal{C}$, and $A \leq_m^{\mathcal{L}} B$. Then $B$ is $\leq_m^{\mathcal{L}}$–hard for $\mathcal{C}$.

**Proof:** Let $E$ be an arbitrary language in $\mathcal{C}$. $E \leq_m^{\mathcal{L}} A$, since $A$ is $\leq_m^{\mathcal{L}}$–hard for $\mathcal{C}$. Then $E \leq_m^{\mathcal{L}} B$ by Lemma 5.3. □

[1] In the literature, these terms have been called "log space hard" and "log space complete". We will use the stated terminology in order to avoid confusion with other possible $O(\log n)$ space reducibilities.