CS 580: Reducibility

1 Mapping Reducibility

Reading: Sipser Chapter 5.

In this section we will formalize the notion of reduction and prove some general statements.

Definition 1 (Computable function). *A function $f : \Sigma^* \to \Sigma^*$ is a* computable function *if there exists a Turing machine M that on every input w halts with just $f(w)$ on the tape.*

Definition 2 (Mapping reducibility). *A language A is mapping reducible to a language B, written $A \leq_m B$, if there is a computable function $f : \Sigma^* \to \Sigma^*$, where for every $w \in \Sigma^*$, we have $w \in A \iff f(w) \in B$. The function f is called the reduction of A to B.*

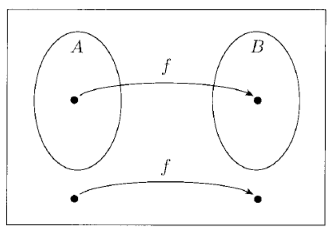

Figure 1: Mapping reducibility.

Thus to test whether $w \in A$, can use the reduction f to map w to $f(w)$, then use a solver for B to check whether $f(w) \in B$, which will also solve the question of whether $w \in A$.

Theorem 1.1. *If $A \leq_m B$ and B is decidable, then A is decidable.*

Proof. Suppose M is a decider for language B. We can construct a TM N for language A, which on input w does the following:

- Compute $f(w)$.
- Run M on input $f(w)$; accept if M accepts and reject if M rejects.

If $w \in A$, then since f is a reduction from A to B, we have that $f(w) \in B$. Thus M accepts $f(w)$ if and only if $w \in A$, which completes the proof of correctness of the decider N. $\square$

Corollary 1. *If $A \leq_m B$ and A is undecidable, then B is undecidable.*

Theorem 1.2. *If $A \leq_m B$ and B is Turing-recognizable, then A is Turing-recognizable.*

Proof. Suppose M is a TM for language B. We can construct a TM N for language A, which on input w does the following:

- Compute $f(w)$.
- Run M on input $f(w)$; accept if M accepts.

If $w \in A$, then since f is a reduction from A to B, we have that $f(w) \in B$. Thus M accepts $f(w)$ if and only if $w \in A$, which completes the proof of correctness of the TM N. $\square$

Corollary 2. *If $A \leq_m B$ and A is not Turing-recognizable, then B is not Turing-recognizable.*