We have many types of reductions, now we’ll formally define one of them (the one we’ve been using):

**Def 5.17:** A function $f: \Sigma^* \to \Sigma^*$ is called **computable** if there is a TM that on every input $w$ halts with $f(w)$ on its tape (and nothing else).
Def 5.20: Language $A$ is **mapping reducible** (or, **many-one reducible**) to language $B$, denoted $A \leq_m B$, if there is a computable function $f: \Sigma^* \rightarrow \Sigma^*$ such that, for every $w$,

$$W \in A \iff f(w) \in B.$$ 

The function $f$ is called the **reduction** of $A$ to $B$.

When proving undecidability of PCP, we did just this $\uparrow$:

- $A = \text{acceptance problem } A_{TM}$
- $B = \text{MPCP}$

$$f(m, w) = \langle p \rangle$$
Mapping Reducibility

**Thm 5.22:** If $A \leq_m B$ and $B$ is decidable, then $A$ is decidable.

**Cor 5.23:** If $A \leq_m B$ and $A$ is undecidable, then $B$ is undecidable.

**Examples:**

- $A_{TM} \leq_m M$PCP
- M$PCP \leq_m A_{TM}$

**NOT TRUE:**

If $A \leq_m B$ and $A$ dec. $\not\implies B$ dec.

(see $A =$ reg. lang., $B =$ HALT$_{TM}$)

eg.
**Mapping Reducibility**

**Thm 5.28:** If $A \leq_m B$ and $B$ is Turing-recognizable, then $A$ is Turing-recognizable.

**Cor 5.29:** If $A \leq_m B$ and $A$ is not Turing-recognizable, then $B$ is not Turing recognizable.

**Thm 5.30:** $EQ_{TM}$ is neither Turing-recognizable nor co-Turing-recognizable.