Formalism for Reductions

Formal definition of (mapping) reduction

\[ A \leq_M B \text{ if:} \]

- there exists a total computable function \( f \) that converts instances of problem \( A \) into instances of problem \( B \)
Things to note about Mapping Reductions: \( A \leq_M B \)

Only one call to the decider for \( B \) is allowed

The call to the decider for \( B \) is the last thing the decider for \( A \) does

The answer for the decider for \( B \) is the answer for the decider for \( A \)

If \( B \) is decidable, \( A \) is decidable
If \( B \) is Turing-recognizable, \( A \) is Turing-recognizable

Example: \( A_{TM} \leq_M HALT_{TM} \)

Mapping function:
- \( f(<M,w>) = <M', w> \) where \( M' = \) “on input \( x \):
  - run \( M \) on \( x \)
  - If it accepts, accept
  - If it rejects, infinite loop”

How do we know \( f \) is computable?
- It can be solved by a TM, \( F \)
- \( F = \) “on input \( <M, w> \), construct \( M' \)
  - write \( <M', w> \) on tape”

Proof that \( f \) reduces \( A_{TM} \) to \( HALT_{TM} \)
- If \( x \in A_{TM} \)
  - If \( x \notin A_{TM} \)
    - \( x \) is of form \( <M, w> \)
    - \( x \) is not of form \( <M, w> \)
Mapping Reductions have limitations

The restrictions (________________, ________________, ____________) mean that some reductions that seem reasonable don’t work.

Can’t reduce $A_{TM}$ to $EMPTY_{TM}$ with a mapping reduction

- If we could, then we could reduce $A_{TM}^C$ to $EMPTY_{TM}^C$ with the same mapping function
- But, $EMPTY_{TM}^C$ is Turing-recognizable (Why?)
- $A_{TM}^C$ is not Turing-recognizable (Why?)
- Can’t have $A \leq_T B$, and $A$ not Turing-recognizable and $B$ Turing-recognizable (Why?)

Can’t reduce $A_{TM}$ to $A_{TM}^C$ using mapping reduction

More General Reduction

Turing Machine with an oracle

- An oracle Turing Machine, $M^B$ (TM with an oracle for $B$) works just like a regular Turing Machine, but can ask questions of the oracle: “is $w \in B$?”

Now we can ask the following questions about the machine $M^B$:

- Can it decide $A$?
- If so, then $A$ is decidable relative to $B$ ($A \leq_T M^B$, read “$A$ is Turing-reducible to $B$”)
Things to note about Turing Reductions: $A \leq_{TM} B$

Many calls to the oracle for $B$ are allowed

The call to the oracle for $B$ need not be the last thing the decider for $A$ does
- It can do further computation

The answer for the oracle for $B$ is not necessarily the answer for the decider for $A$

If $B$ is decidable, $A$ is decidable
If $B$ is Turing-recognizable, $A$ is Turing-recognizable

Example of Turing-reducibility

$A_{TM}$ is Turing-reducible to $A_{TM}^C$ ($A_{TM} \leq_{TM} A_{TM}^C$)

$T_{ATMC} = \text{"On input } <M, w> \text{ where } M \text{ is a TM:}
\quad \text{Query the oracle to determine whether } <M,w> \in A_{TM}^C$
\quad If the oracle answers yes, reject. Otherwise (no), accept."

$A_{TM}$ is Turing-reducible to $\text{EMPTY}_{TM}$ ($A_{TM} \leq_{TM} \text{EMPTY}_{TM}$)

$T_{ATM} = \text{"On input } <M, w> \text{ where } M \text{ is a TM:}
\quad \text{Build } M' = \text{"On input } x, 
\quad \text{Simulate } M \text{ on } w
\quad \text{If } M \text{ accepts } w, \text{ accept}
\quad \text{If } M \text{ rejects } w, \text{ reject"
\quad Query the oracle to determine whether } <M'> \in A_{TM}$
\quad If the oracle answers yes, reject. Otherwise (no), accept."
Continuation of Post Correspondence Problem

Reducing PCP to $\cap_{CFG}$

$\cap_{CFG} = \{(G_1, G_2)| G_1$ and $G_2$ are CFGs and $L(G_1) \cap L(G_2) = \emptyset\}$

- Reducing PCP to $\cap_{CFG}$ (PCP $\leq$ $\cap_{CFG}$)
- Make A into one grammar, B into another

\begin{align*}
A & \rightarrow 00A1 | 110A2 | 000A3 | 00#1 | 110#2 | 000#3 \\
B & \rightarrow 000B1 | 0B2 | 11B3 | 000#1 | 0#2 | 11#3
\end{align*}

<table>
<thead>
<tr>
<th>i</th>
<th>$A_i$</th>
<th>$B_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>00</td>
<td>000</td>
</tr>
<tr>
<td>2</td>
<td>110</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>000</td>
<td>11</td>
</tr>
</tbody>
</table>
Reducing $\cap_{CFG}$ to $ALL_{CFG}$

$ALL_{CFG} = \{ G | G \text{ is a CFG and } L(G) = \Sigma^* \}$

- Create $G' = (G_1 \cap G_2)^c$
- Check whether $G' \in ALL_{CFG}$: If yes, $G_1$ and $G_2$ don’t intersect. If no, they do intersect.

- Is this valid? Is $G_1 \cap G_2$ a CFG? Is the complement of a CFL a CFL?
  - No

- However, look at $PCP \subseteq^{\cap_{CFG}} \subseteq ALL_{CFG}$
  - The grammars used in the reduction of PCP were deterministic!

- So, actually, the reduction is:

  - $G' = G_1^c \cup G_2^c = (G_1 \cap G_2)^c$
  - $G^c$ is DCFG
  - Union is CFG

Reducing $ALL_{CFG}$ to $EQ_{CFG/REG}$

$EQ_{CFG/REG} = \{(G, R) | G \text{ is a CFG, } R \text{ is an FSA, } L(G) = L(R)\}$

- How to solve $ALL(G)$:
  - Check $EQ(G, \Sigma^*)$
Reducing $E Q_{CFG/REG}$ to $E Q_{CFG}$

$E Q_{CFG} = \{(G_1, G_2) \mid G_1 \text{ and } G_2 \text{ are CFGs and } L(G_1) = L(G_2)\}$

Reductions