Closure Properties of Decidable Languages

✦ Decidable languages are closed under $\cup$, $\circ$, $\ast$, $\cap$, and complement

✦ Example: Closure under $\cup$

✦ Need to show that union of 2 decidable L’s is also decidable

Let $M_1$ be a decider for $L_1$ and $M_2$ a decider for $L_2$

A decider $M$ for $L_1 \cup L_2$:

On input $w$:

1. Simulate $M_1$ on $w$. If $M_1$ accepts, then ACCEPT $w$. Otherwise, go to step 2 (because $M_1$ has halted and rejected $w$)
2. Simulate $M_2$ on $w$. If $M_2$ accepts, ACCEPT $w$ else REJECT $w$.

$M$ accepts $w$ iff $M_1$ accepts $w$ OR $M_2$ accepts $w$

i.e. $L(M) = L_1 \cup L_2$

---

Closure Properties

✦ Consider the proof for closure under $\cup$

A decider $M$ for $L_1 \cup L_2$:

On input $w$:

1. Simulate $M_1$ on $w$. If $M_1$ accepts, then ACCEPT $w$. Otherwise, go to step 2 (because $M_1$ has halted and rejected $w$)
2. Simulate $M_2$ on $w$. If $M_2$ accepts, ACCEPT $w$ else REJECT $w$.

$M$ accepts $w$ iff $M_1$ accepts $w$ OR $M_2$ accepts $w$

i.e. $L(M) = L_1 \cup L_2$

Will this proof work for showing Turing-recognizable languages are closed under $\cup$? Why/Why not?

Uh…I dunno.

Wait, will $M_1$ always halt?!!
Closure for Recognizable Languages

- Turing-Recognizable languages are closed under $\cup$, $\circ$, $*$, and $\cap$ (but not complement! We will see this in the final lecture)
- Example: **Closure under $\cap$**
  Let $M_1$ be a TM for $L_1$ and $M_2$ a TM for $L_2$ (both may loop)
  A TM $M$ for $L_1 \cap L_2$:
  
  On input $w$:
  1. Simulate $M_1$ on $w$. If $M_1$ halts and accepts $w$, go to step 2. If $M_1$ halts and rejects $w$, then REJECT $w$. (If $M_1$ loops, then $M$ will also loop and thus reject $w$)
  2. Simulate $M_2$ on $w$. If $M_2$ halts and accepts, ACCEPT $w$. If $M_2$ halts and rejects, then REJECT $w$. (If $M_2$ loops, then $M$ will also loop and thus reject $w$)
  
  $M$ accepts $w$ iff $M_1$ accepts $w$ AND $M_2$ accepts $w$ i.e. $L(M) = L_1 \cap L_2$

The Church-Turing Thesis

- Various definitions of “algorithms” were shown to be equivalent in the 1930s
- **Church-Turing Thesis**: “The intuitive notion of algorithms equals Turing machine algorithms”
  - Turing machines serve as a precise formal model for the intuitive notion of an algorithm
- “Any computation on a digital computer is equivalent to computation in a Turing machine”

Dude, that’s pretty deep…
Undecidable Languages

- **The Question**: Are there languages that are not decidable by any Turing machine (TM)?
  - i.e. Are there problems that cannot be solved by any algorithm?
- Consider the language:
  \[ \text{ATM} = \{<M,w> | M \text{ is a TM and } M \text{ accepts } w\} \]
- **Note**: \(<A,B,...>\) is just a string encoding the objects A, B, …
- In particular, \(<M,w>\) is a string listing all the components of TM M (separated by #, for example) followed by the string w
- Given input \(<M,w>\), it should be easy to extract the info about M and to simulate M on w (try writing a TM to do this!)
- What can we say about \(\text{ATM}\)?

\[ \text{ATM} \text{ is Turing-recognizable} \]

- **ATM is Turing-recognizable**: Recognizer TM \(U\) for \(\text{ATM}\):
  - On input string \(<M,w>\):
    - Simulate M on w.
    - ACCEPT \(<M,w>\) if M halts & accepts w;
    - REJECT \(<M,w>\) if M halts & rejects (Loop (& thus reject \(<M,w>\)) if M ends up looping).
  - U accepts \(<M,w>\) iff M accepts w, i.e. \(L(U) = \text{ATM}\)

Yeah, but is it decidable???

“Universal” TM (can simulate any TM)
Is $A_{TM}$ decidable?

- No! $A_{TM} = \{<M,w> | M \text{ is a TM and } M \text{ accepts } w\}$ is undecidable! 1-slide Proof (by Contradiction):
  1. Assume $A_{TM}$ is decidable $\Rightarrow$ there’s a decider $H$, $L(H) = A_{TM}$
  2. $H$ on $<M,w> = \text{ACC}$ if $M$ accepts $w$
     REJ if $M$ rejects $w$ (halts in $q_{\text{REJ}}$ or loops on $w$)
  3. Construct new TM $D$: On input $<M>$:
     Simulate $H$ on $<M,<M>>$ (here, $w = <M>$)
     If $H$ accepts, then REJ input $<M>$
     If $H$ rejects, then ACC input $<M>$
  4. What happens when $D$ gets $<D>$ as input?
     D rejects $<D>$ if $H$ accepts $<D,<D>>$ if $D$ accepts $<D>$
     D accepts $<D>$ if $H$ rejects $<D,<D>>$ if $D$ rejects $<D>$
     Either way: Contradiction! $D$ cannot exist
     Therefore, $A_{TM}$ is not a decidable language.

Undecidability Proof uses Diagonalization

- Input strings
  - $<M_1>,<M_2>,<M_3>,...$
  - $<M_1>,<M_2>,<M_3>,...,<D>$
- List of TMs
  - $M_1$
  - $M_2$
  - $M_3$
  - $M_i$:
    - ACC
    - REJ
    - loop
    - ...
- If $H$ exists
  - D outputs opposite of diagonal
    - ACC
    - REJ
    - ACC
    - ...
    - ACC
  - D on $<M_i>$ accepts if and only if $M_i$ on $<M_i>$ rejects.
  - So, $D$ on $<D>$ will accept if and only if $D$ on $<D>$ rejects!
  - A contradiction $\Rightarrow$ $H$ cannot exist!
  - Therefore, $A_{TM}$ is not a decidable language.
One Last Concept: Reducibility

✦ How do we show a new problem B is undecidable?
✦ Idea: Show that $A_{TM}$ is reducible to the new problem B
   ✦ What does this mean and how do we show this?
✦ Show that if B was decidable, then you can use the decider for B as a subroutine to decide $A_{TM}$
   ✦ Contradiction, therefore B must also be undecidable

The Halting Problem is Undecidable (Turing, 1936)

✦ Example: Halting Problem: Does TM M halt on input w?
   ✦ Equivalent language: $A_H = \{ <M,w> | \text{TM M halts on input w} \}$
   ✦ Need to show $A_H$ is undecidable
   ✦ We know $A_{TM} = \{ <M,w> | \text{TM M accepts w} \}$ is undecidable
✦ Show $A_{TM}$ is reducible to $A_H$ (Theorem 5.1 in text)
   ✦ Suppose $A_H$ is decidable ⇒ there’s a decider $M_H$ for $A_H$
   ✦ Then, we can construct a decider $D_{TM}$ for $A_{TM}$:
     On input $<M,w>$, run $M_H$ on $<M,w>$.
     ● If $M_H$ rejects, then REJ (this takes care of M looping on w)
     ● If $M_H$ accepts, then simulate M on w until M halts
     ● If M accepts, then ACC input $<M,w>$; else REJ
     $L(D_{TM}) = A_{TM} ⇒ A_{TM}$ is decidable! Contradiction ⇒ $A_H$ is undecidable
✦ E.g. 2: Show $E_{TM} = \{ <M> | \text{M is a TM and L(M) = } \emptyset \}$ is undecidable (see Theorem 5.2 in the text)