Decidability
**Decision Problems**

- A yes/no question over many instances
  - Given grammar $G$, is $G$ ambiguous?
  - Given a TM $M$, does $L(M) = \{0,1\}^*$?
  - Given DFAs $M_1$ and $M_2$, does $L(M_1) = L(M_2)$?
  - Given a graph $G$, is $G$ connected?
  - Given a graph $G$, nodes $s$ and $t$, and number $d$, is there a path from $s$ to $t$ of distance $d$ or less?
Equivalently, languages:

- \{<G> \mid <G> \text{ encodes an unambiguous grammar}\}
- \{<M> \mid L(M) = \{0,1\}^* \}
- \{<M_1> \# <M_2> \mid \text{DFAs } M_1 \text{ and } M_2, \text{ accept the same language}\}
- \{<G> \mid <G> \text{ encodes a connected graph}\}
- \{<G>\#s\#t\#d \mid <G> \text{ encodes a graph with nodes } s \text{ and } t, \text{ there is a path from } s \text{ to } t \text{ of distance } d \text{ or less}\}

Deciding membership in the language is solving the decision problem.
Decidable

• A decision problem (language) is *decidable* if there is a TM that always halts that accepts the language. (The language is recursive.)
• I.e., there is an algorithm that always answers “yes” or “no” correctly.
• Note: since all finite languages are recursive, (they’re regular in fact) any decision problem with only a finite number of instances is decidable, and not well-addressed by this theory....
Example 1: decidable or not?

• Is there a substring of exactly 374 consecutive 7’s in decimal expansion of $\pi$?

• This is decidable. There is an algorithm which is correct. It is one of these:

<table>
<thead>
<tr>
<th>Alg 1</th>
<th>Alg 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output “yes”</td>
<td>Output “no”</td>
</tr>
</tbody>
</table>

We just don’t know which one it is.
But, there is an algorithm which will tell us which it is!
Moral

• This is nonsense
• There were no “instances” of the problem.
• It simply asks a single yes/no question.
• Not even clear what “language” corresponds to it
• Remember: decidability is for problems with many possible input instances
Example 2

• Give $n$, is there a substring of exactly $n$ consecutive 7’s in $\pi$?
• Language: $\{n \mid \text{decimal expansion of } \pi \text{ contains the substring } a7^n b, \text{ where } a \text{ and } b \text{ are not 7s}\}$
• Is this language decidable? Is there a halting TM for it?
• It is r.e.? (recall: a TM that may not halt but accepts if it should)
Example 3

• Give \( n \), is there a substring of at least \( n \) consecutive 7’s in \( \pi \)?

• Language: \( L = \{ n \mid \text{decimal expansion of } \pi \text{ contains the substring } 7^n \} \)

• Is this language decidable? Is there a halting TM for it?

• In fact, it is regular!
  
  (\( L \) is either all of \( \mathbb{N} \), or equals \( \{0,1,2,\ldots,k\} \) for some fixed \( k \).)
$L_u$

• Recall $L(M_u) = \{ <M> \# w \mid M \text{ accepts } w \}$
• This language is called $L_u$, the “universal” language
• Is $L_u$ recursive? I.e., given a TM $<M>$ and input $w$, can we decide whether or not $M$ accepts $w$?
• If $L_u$ were decidable, we’d be able to tell if any program accepted any input.
$L_u$ is r.e. - it is accepted by a TM

**Proof:**

$M_u$ accepts exactly the language $L_u$
$L_u$ is not decidable

Warm-up: Self-reference leads to paradox

- In a town there is a barber who shaves all and only those who do not shave themselves
  
  **Who shaves the barber?**

- Homogenous words: self-describing
  - English, short, polysyllabic

  Heterogenous words: non-self-describing
  - Spanish, long, monosyllabic

What kind of word is “heterogenous”?
$L_u$ is not decidable

• Proof by contradiction
• Suppose there was an algorithm (TM) that always halted, as follows:

  $<M> \# w \rightarrow$
  
  TM accept-checker
  
  Check if $M(w)$ accepts
  
  yes, $M(w)$ accepts
  
  no, $M(w)$ doesn’t accept*

* remember – $M(w)$ may not halt – which is why this may be difficult

We’ll show how to use this as a subroutine to get a contradiction
$L_u$ is not decidable

- Proof by contradiction
- Suppose there was an algorithm (TM) as follows:

  \[
  \text{TM accept-checker}
  \]

  Decides if $M(<M>)$ accepts

  $Q(<M>)$ rejects iff $M(<M>)$ accepts
  $Q(<M>)$ accepts iff $M(<M>)$ doesn’t accept
$L_u$ is not decidable

TM   Q

copy-arg   $<M>$ # $<M>$  

TM accept-checker
Decides if $M(<M>)$ accepts

accept
doesn’t

accept
reject

$Q(<M>)$ rejects iff $M(<M>)$ accepts
$Q(<M>)$ accepts iff $M(<M>)$ doesn’t accept

Does $Q(<Q>)$ accept or reject?

either way, a contradiction, so assumption that accept-checker existed was wrong