The halting problem for TM's is undecidable

- We go through the solution to problem 8.3 of the textbook, given at the end of the exercises
- \( L_h = \{ <M> : M \text{ halts on all inputs} \} \) is not recursive
- We prove that if \( L_h \) is recursive then a non-trivial property would be recursive

We need an \( M' \)

- Given \( <M> \), construct \( M' \) so that \( L(M) = L(M') \) and \( M' \) ONLY halts on \( L(M) \)
- \( M' \) shifts its input one space to the right, putting “$” at the first position and then simulates \( M \)
- If you reach “$” or reach a non-final state that has no transition move to new state \( p_1 \), from there oscillate between \( p_1 \) and new state \( p_2 \)
A non-trivial property would be decidable

- Suppose $L_h$ is recursive and $M_h$ is the halting TM for $L_h$.
- Construct a halting TM for $L_{S4} = \{<M>: L(M) = (0+1)^* \}$ as follows:

The monster for $\{<M>: L(M) = (0+1)^* \}$

- If $L(M) = (0+1)^*$, the monster machine says yes since $M'$ halts on all inputs.
- If $L(M) \neq (0+1)^*$, the monster says no since $M'$ fails to halt on some inputs.

Apply Rice’s theorem

- All together, we have a halting TM for $L_{S4} = \{<M>: L(M) = (0+1)^* \}$.
- Such a halting TM does not exist by Rice’s theorem.

Similar results

- Suppose we take the problem “Does $M$ halt on a fixed $w$?”
- Consider the property $S = \{L: w \in L\}$
- We know that $S$ is undecidable because it is non-trivial this since $\emptyset$ does not have the property and $\{w\}$ has the property, i.e. $\{<M>: w \in L(M) \}$ is not recursive.
Halting on a fixed string

- Suppose we had a halting TM $M_w$ that answers the question “Does $M$ halt on $w$?”
  - i.e. Suppose $M_w$ accepts the language $\{ <M> : M \text{ halts on } w \}$
- Then take $<M>$, construct $M'$ from $M$ as before and create a Chapter 8 encoding $M''$ of $M'$ to feed to $M_w$

The monster for $\{ <M> : w \in L(M) \}$

- If $w \in L(M)$, the monster machine says yes since $M'$ halts on $w$
- If $w \notin L(M)$, the monster says no since $M'$ fails to halt on $w$

Getting to the language question

- $M_w$ accepts $<M''>$ if and only if $w \in L(M) = L(M') = L(M'')$ since $M''$ halts on $w$ if and only if $M''$ accepts $w$
- Thus we get a halting TM for $\{ <M> : w \in L(M) \}$ which is impossible

Another result related to emptiness

- Similarly, if we had a halting TM $M?$ for “Does $M$ halt on some input?”
- We would deduce a halting TM for $L_{ne} = \{ <M> : L(M) \text{ is not empty} \}$ which does not exist
Halting on some string

- Suppose we had a halting TM $M_2$ that answers the question “Does $M$ halt on some $w$?”
  - i.e. Suppose $M_2$ accepts the language \{<M> : M halts on some input\}
- Then take any <M>, construct $M'$ from $M$ as before and create a Chapter 8 encoding $M''$ of $M'$ to feed to $M_2$

The monster for $L_{ne} = \{<M> : L(M) \neq \emptyset \}$

- If $L(M) \neq \emptyset$, the monster machine says yes since $M'$ halts on some input
- If $L(M) = \emptyset$, the monster says no since $M'$ fails to halt on any input

Conversion to a known non-trivial problem

- $M_2$ accepts <M’’> if and only if $L(M) = L(M') = L(M'')$ is not empty, since $M''$ halts on a string if and only if $M''$ accepts that string
- Thus we get a halting TM for $L_{ne} = \{<M> : L(M) \text{ is not empty} \}$ which is impossible

Rice's theorem for recursive index sets

- Rice’s theorem tells us whether some $L_S$ is recursive and we deduced $L_h$ is not recursive
- It is more complicated to find out whether some $L_S$ or $L_h$ is or is not rec. enum.
- Rice has a theorem: $L_S$ is rec. enum. if and only if $S$ satisfies 3 conditions:
Two more dots in the diagram

Context-free Grammars

The pumping lemma
(PL4CFL)

Recursive

Recursively enumerable

Recursive\n
Context-free

The CNF gives us a pumping lemma
The Chomsky Normal Form grammar only has productions of the form
\[ A \rightarrow BC, \quad A \rightarrow a \]

The special thing about CNF is the form of the derivation trees
CNF derivation trees are binary with unary fringes

A CNF derivation tree

The root is \( S \), the internal nodes are variables, the leaves are terminals
The PL4CFL states that given a CFL \( L \) there is an \( n \), such that if \( z \in L \) and \( |z| > n \), then \( z \) can be written as \( z = uvwx \), where:

- \( |vx| > 0 \) (i.e., \( v \) and \( x \) are not both empty)
- \( |vxw| \leq n \)
- \( uv^kwx^ky \in L \) for all \( k \geq 0 \)

<table>
<thead>
<tr>
<th>Proof</th>
</tr>
</thead>
</table>
| Take \( m \) to be the number of variables in a CNF grammar for \( L - \{\varepsilon\} \).
| Take \( n = 2^m \).
| What is the CNF derivation tree of least height that can produce a string of length at least \( n \)?

<table>
<thead>
<tr>
<th>Statement of PL4CFL</th>
</tr>
</thead>
</table>
| The PL4CFL states that given a CFL \( L \), there is a \( n \), such that if \( z \in L \) and \( |z| \geq n \), then \( z \) can be written as \( z = uvwx \), where:
| \( |vx| > 0 \) (i.e., \( v \) and \( x \) are not both empty)
| \( |vxw| \leq n \)
| \( uv^kwx^ky \in L \) for all \( k \geq 0 \) |

<table>
<thead>
<tr>
<th>Proof</th>
</tr>
</thead>
</table>
| Take \( m \) to be the number of variables in a CNF grammar for \( L - \{\varepsilon\} \).
| Take \( n = 2^m \).
| What is the CNF derivation tree of least height that can produce a string of length at least \( n \)?

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
</table>
| Height 1, string length 1
| Height 2, string length 2
| Height 3, string length \( \leq 4 \)
First conclusion

If $|I| \geq 2^m$, then the derivation tree from the CNF has height at least $m+1$. Consider the lowest part of the longest branch.

- Look for the LOWEST $m+1$ edges of the longest branch.
- $m+1$ edges have $m+1$ variables and 1 terminal.
- Take a branch of the derivation tree that has height at least $m+1$.

Examine the longest branch.

Look for the LOWEST $m+1$ edges of the longest branch.

$m+1$ edges have $m+1$ variables and 1 terminal.

Consider the LOWEST $m+1$ edges of that branch.

$m+1$ edges have $m+1$ variables and 1 terminal.

The lowest part of the longest branch:

- $m+1$ edges:
  - $m+2$ nodes, the lowest is a terminal, the remaining $m+1$ are variables.

Pigeonholes

There were $m$ variables in this grammar...

So there is a repeated variable on this lower part of the branch.

It is shown as "A".
The derivation tree for $z$ can be outlined with triangles:

- We can identify the pieces $u, v, w, x, y$ of the string $z$ from the smaller triangles (subtrees).

Thus $z = uvwxy$ and we can play with the tree to show how to pump the substrings of $z$.

The parts of $z$:

- $w$ is generated by the lower $A$.
- $z'$ is generated by upper $A$.
- $w$ is a substring of $z'$ so $z' = vwx$.
- $z'$ is a substring of $z$, so $z = u z' y$.

Anything derived from $S$ is in $L$:

- Recall $L$ consists of all strings derived from $S$.
- The tree under the upper $A$ can be exchanged with the tree under the lower $A$. 

Relation of the pieces and the tree:

- Thus $z = uvwxy$ and we can play with the tree to show how to pump the substrings of $z$. 

Anything derived from $S$ is in $L$:

- Recall $L$ consists of all strings derived from $S$.
- The tree under the upper $A$ can be exchanged with the tree under the lower $A$. 

Relation of the pieces and the tree:

- Thus $z = uvwxy$ and we can play with the tree to show how to pump the substrings of $z$. 

Anything derived from $S$ is in $L$:

- Recall $L$ consists of all strings derived from $S$.
- The tree under the upper $A$ can be exchanged with the tree under the lower $A$. 
Pumping \( v \) and \( x \)

Since \( A \Rightarrow vAx \), and \( A \Rightarrow w \), we can interchange \( w \) and \( vwx \) as many times as we like:

- \( S \Rightarrow uAy \Rightarrow uwy \) \( (uv^0wx^0y \in L) \)
- \( S \Rightarrow uvAxy \Rightarrow uvvwxxy \) \( (uv^2wx^2y \in L) \)
- \( S \Rightarrow uvAxy \Rightarrow uvvAxy \Rightarrow uvvwwxxxy \) \( (uv^3wx^3y \in L) \)
- etc

In pictures - I

- The tree for \( xwy \)

In pictures - II

- The tree for \( uv^2wx^2y \)

In pictures - III

- The tree for \( uv^kwx^k y \)
Note that the height of the tree under the top $A$, which could be $V_1$, is $m+1$.

That $A$ derives $vwx$, so $|vwx| \leq 2^m = n$.

Since the two $A$'s are distinct, not both of $v$ and $x$ are empty.

- $v$ could be empty: i.e. $A \Rightarrow Ax$ but $x \neq \varepsilon$.
- Similarly, $x$ could be empty but then $v \neq \varepsilon$.

$u$ could be empty: i.e. $S \Rightarrow Ay$.
- Similarly, $y$ could be empty.
- In fact $A$ might be $S$, and $u$ and $y$ can be empty.

We have seen a Turing Machine for a language similar to:

$L = \{0^j1^j0^j : j > 0\}$

Now we prove the $L$ is not a CFL using the PL4CFL.
Starting the proof

- Pick $z = 0^n1^n0^n$, where $n$ is the number form the PL4CFL
- Since $|z| > n$, we can write $z = uvwxy$, where $|vwx| \leq n$ and $uv^kwx^ky \in L$
- We have NO IDEA where $v$ and $x$ might be

vwx cannot be too long

- We only know $|vwx| \leq n$ so that $v, x$ cannot stretch from the first 0’s to the last 0’s
  $z = 00...011...100...0$

Possible placements

- Possible locations for $uvwxy$:
  1. $vwx$ lies entirely in the first 0’s
     - $z = 00...11...100...0$
     - $u|vwxy$
  2. $vwx$ lies entirely in the 1’s
     - $z = 00...1 1 ... 100...0$
     - $u|vwx|y$
  3. $vwx$ lies entirely in the last 0’s
     - $z = 00...11...100...0$
     - $uvwx|y$
  4. $vwx$ lies from first 0’s into 1’s
     - $z = 00...11...100...0$
     - $u|vwxy|y$
  5. $vwx$ lies from the 1’s into last 0’s
A single counter-example?

- WE CANNOT ASSUME ANY PARTICULAR PLACEMENT
- We have to show there is a contradiction for each of the 5 possible placements

Cases 1, 2, 3

- The three cases 1, 2 and 3 are similar:
- For example, in case 2, $uvw$ has fewer 1’s than the first 0’s and hence is not in $L$

Cases 4, 5

- The two cases 4 and 5 are similar:
- In case 4, $uvw$ has fewer of the first 0’s and/or 1’s than the number of the last 0’s and therefore is not in $L$

The contradiction

- Note, we could just as well have pumped up to get more of one kind of symbol than another
- Since there is no location for the $vwx$ that does not give a contradiction to the property $uv^kwx^ky \in L$ for all $k \geq 0$, $L$ is not a CFL
To reiterate...

- You MUST verify that there is a contradiction for EACH possible position of \( vwx \)

A recursive language that is not a CFL

- We can show the following language is not a CFL:
  \[ L = \{0^i1^{i+2^k}0^j : j > k \geq i > 0\} \]

- Now we prove the \( L \) is not a CFL using the PL4CFL

A second non-CFL

- Pick \( z = 0^{n+1}n2^k0^n : j > k \geq i > 0 \), where \( n \) is the number from the PL4CFL
- Since \( |z| > n \), we can write \( z = uvwxyst \), where \( |vwx| \leq n \) and \( uv^kwx^ky \in L \)
- We have NO IDEA where \( v \) and \( x \) might be
**Possible placements**

- We only know $|vwx| \leq n$ so that $v, x$ cannot stretch from the first 0’s to the 2’s
  
  $z = 00...011...122...200...0$

- Also $v, x$ cannot stretch from the 1’s to the last 0’s
  
  $z = 00...011...122...200...0$

**Possible locations for $uvwxy$:**

- (1) $vwx$ lies entirely in the first 0’s
- (2) $vwx$ lies entirely in the 1’s
- (3) $vwx$ lies entirely in the 2’s
- (4) $vwx$ lies entirely in the last 0’s

**Possible placements**

- (5) $vwx$ lies in the first 0’s and the 1’s and $v, x$ include 0’s and 1’s
  [otherwise we are in case 1 or 2]
- (6) $vwx$ lies in the 1’s and 2’s and $v, x$ include 1’s and 2’s
- (7) $vwx$ lies in the 2’s and last 0’s and $v, x$ include 2’s and last 0’s

**A single counter-example?**

- WE CANNOT ASSUME ANY PARTICULAR PLACEMENT
- We have to show there is a contradiction for each of the 7 possible placements
Case 1

- Recall $L = \{0^i1^j2^k0^i : j > k \geq i > 0\}$
- $z = 0^{n1}n+1^2n0^{n+1}$
- Case 1 ($vwx$ in first 0’s), $uv^2wx^2y$ has at least $n+1$ 0’s in the first 0’s, contradicting $k \geq i$ and hence is not in $L$

Class 16 - 62 © L. Lander, 2000 62CS 573

Case 2

- $L = \{0^i1^j2^k0^i : j > k \geq i > 0\}$
- $z = 0^{n1}n+1^2n0^{n+1}$
- Case 2 ($vwx$ in 1’s), $uw^y$ has less than $n+1$ 1’s contradicting $j > k$ and hence is not in $L$
- ($uw^y$ also contradicts the number of 1’s having to equal the number of last 0’s)

Class 16 - 63 © L. Lander, 2000 63CS 573

Case 3

- $L = \{0^i1^j2^k0^i : j > k \geq i > 0\}$
- $z = 0^{n1}n+1^2n0^{n+1}$
- Case 3 ($vwx$ in 2’s), $uv^2wx^2y$ has at least $n+1$ 2’s contradicting $j > k$ and hence is not in $L$

Class 16 - 64 © L. Lander, 2000 64CS 573

Case 4

- $L = \{0^i1^j2^k0^i : j > k \geq i > 0\}$
- $z = 0^{n1}n+1^2n0^{n+1}$
- Case 4 ($vwx$ in last 0’s), $uw^y$ has less than $n+1$ 0’s in the last 0’s, contradicting $j > k$ and hence is not in $L$
- ($uw^y$ also contradicts the number of 1’s having to equal the number of last 0’s)
Case 5

- \( L = \{0^i1^j2^k0^l : j > k \geq i > 0 \} \)
- \( z = 0^n1^{n+1}2^n0^{n+1} \)
- Case 5 (\( v, x \) include first 0's and 1's), \( uwy \) has less than \( n+1 \) 1's, contradicting \( j > k \) and hence is not in \( L \)
- (pumping down \( uwy \) also contradicts the number of 1's having to equal the number of last 0's)

Case 6

- \( L = \{0^i1^j2^k0^l : j > k \geq i > 0 \} \)
- \( z = 0^n1^{n+1}2^n0^{n+1} \)
- Case 6 (\( v, x \) include first 1's and 2's), \( uwy \) has less than \( n+1 \) 1's, contradicting \( j > i \) (the number of the first 0's) and hence is not in \( L \)
- (\( uwy \) also contradicts the number of 1's having to equal the number of last 0's)

Case 7

- \( L = \{0^i1^j2^k0^l : j > k \geq i > 0 \} \)
- \( z = 0^n1^{n+1}2^n0^{n+1} \)
- Case 7 (\( v, x \) include 2's and last 0's), \( uv^2wx^2y \) has as many or more 2's than 1's, contradicting \( j > k \) and hence is not in \( L \)
- (\( uv^2wx^2y \) also contradicts the number of 1's having to equal the number of last 0's)

The contradiction

- Since there is no location for the \( vwx \) that does not give a contradiction to the property \( uv^kw^kwx^ky \in L \) for all \( k \geq 0 \), \( L \) is not a CFL
To reiterate...

- You MUST verify that there is a contradiction for EACH possible position of \( vwx \)
- EXTRA NOTE: In Cases 5, 6 and 7 it may sometimes happen that \( v \) or \( x \) contains a combination of symbols
- For example, in Case 5, \( x \) might be \( 0^p1q \)
- In such a case \( uv^2wx^2y \) has a totally wrong form: \( 0^{n+|v|}1q \ 0^p1^{n+1}2^n \ 0^{n+1} \)