For this subsection no additional material has been added yet.
Formal Lemma URM-computable $\Rightarrow$ TM-computable

Lemma (3.4)
If $f : \mathbb{N}^n \rightarrow \mathbb{N}$ is URM-computable then it is Turing-computable by a TM with alphabet \{0, 1, \}$\downarrow$\.

Remark
The proof that every Turing computable function is URM computable will not be given in this Section.
(It could be done directly. A much nicer argument which makes use of the notion of partial recursive functions can be found in the notes of "Computability Theory").

Proof of Lemma 3.4

Notation: $\widetilde{\text{bin}}$

- In this proof we will represent a configuration of a URM by a sequence of possibly non-normalised strings on the tape representing the registers.
- So we want to get a short notation for \"The tape contains $s_0 \downarrow s_1 \downarrow s_2 \downarrow \ldots \downarrow s_k$ where $s_i$ is a binary representation of $n_i$\" (where $n_i$ is the simulated content of register $R_i$).
- We define $\widetilde{\text{bin}}(n)$ as one of the binary representations of $s$.
- Then we can write for the above: \"The tape contains $\widetilde{\text{bin}}(n_0) \downarrow \widetilde{\text{bin}}(n_1) \downarrow \ldots \downarrow \widetilde{\text{bin}}(n_k)$\".
- So $\widetilde{\text{bin}}(n)$ denotes one of the possible choices for strings $s$ s.t. $(s)_2 = n$.
  - So $\widetilde{\text{bin}}(1)$ can be "1", "01", "001", etc.
  - In the special case 0 we treat the empty string as one of the possible representations, so $\widetilde{\text{bin}}(0)$ can be "", "0", "00", "000", etc.

- When carrying out intermediate calculations, it is easier to refer to $\widetilde{\text{bin}}(n)$ rather than $\text{bin}(n)$
  - E.g. we can set a number on the tape easily to an element of $\widetilde{\text{bin}}(0)$ by overwriting it with 0s.
  - In order to set it to $\text{bin}(0)$ one would need to make sure that exactly one 0 remains. Then one usually has to shift left the content of the tape to the right of the original number.

Notation

The tape of a TM contains $a_0, \ldots, a_l$ means:
- Starting from the head position, the cells of the tape contain $a_0, \ldots, a_l$.
- All other cells contain $\downarrow \downarrow$. 

CS_236 Sect. IV.3 (b) 5/1

CS_236 Sect. IV.3 (b) 6/1

CS_236 Sect. IV.3 (b) 7/1

CS_236 Sect. IV.3 (b) 8/1
IV.3 (b) Equivalence of URM computable and Turing computable functions

Proof of Lemma 3.4

Assume
- \( f = U(n) \),
- \( U \) refers only to \( R_0, \ldots, R_{l-1} \) and \( l > n \).

We define a TM \( T \), which simulates \( U \). Done as follows:
- That the registers \( R_0, \ldots, R_{l-1} \) contain \( a_0, \ldots, a_{l-1} \) is simulated by the tape containing \( \widetilde{\text{bin}}(a_0) \) \( \ldots \) \( \widetilde{\text{bin}}(a_{l-1}) \).
- An instruction \( I_j \) will be simulated by states \( s_j,0, \ldots, s_j,i \) with instructions for those states.

Example

- Assume the URM is about to execute instruction
  - \( I_4 = \text{pred}(2) \) (i.e. PC = 4),
  - with register contents
    \[
    \begin{array}{c|c|c}
    R_0 & R_1 & R_2 \\
    \hline
    2 & 1 & 3 \\
    \end{array}
    \]
- Then the URM will end with
  - PC = 5
  - and register contents
    \[
    \begin{array}{c|c|c}
    R_0 & R_1 & R_2 \\
    \hline
    2 & 1 & 2 \\
    \end{array}
    \]

Conditions on the Simulation

- Assume the URM \( U \) is in a state s.t.
  - \( R_0, \ldots, R_{l-1} \) contain \( a_0, \ldots, a_{l-1} \),
  - the URM is about to execute \( I_j \).
- Assume after executing \( I_j \), the URM is in a state where
  - \( R_0, \ldots, R_{l-1} \) contain \( b_0, \ldots, b_{l-1} \),
  - the PC contains \( k \).
- Then we want that, if configuration of the TM \( T \) is, s.t.
  - the tape contains \( \widetilde{\text{bin}}(a_0) \) \( \ldots \) \( \widetilde{\text{bin}}(a_{l-1}) \),
  - and the TM is in state \( s_{j,0} \),
then the TM reaches a configuration s.t.
- the tape contains \( \widetilde{\text{bin}}(b_0) \) \( \ldots \) \( \widetilde{\text{bin}}(b_{l-1}) \),
- the TM is in state \( s_{k,0} \).

Example

- Then we want that, if the simulating TM is
  - in state \( s_{4,0} \),
  - with tape content \( \widetilde{\text{bin}}(2) \) \( \ldots \) \( \widetilde{\text{bin}}(3) \)
it should reach
  - state \( s_{5,0} \)
  - with tape content \( \widetilde{\text{bin}}(2) \) \( \ldots \) \( \widetilde{\text{bin}}(2) \)
Furthermore, we need initial states $s_0, 0, \ldots, s_n$ and corresponding instructions, s.t. if the TM initially contains $\bin(a_0, a_1, \ldots, a_n) = 0011\ldots$, it will reach state $s_0$ with the tape containing $\bin(a_0, a_1, \ldots, a_n)$ in the following $n$ times.

Consider the URM program $U$ (which was discussed already in the section on URMs):

$U \leftarrow \text{ifzero}(0, 3)\leftarrow \text{pred}(0)\leftarrow \text{ifzero}(1, 0)\leftarrow 1 = \text{ifzero}(0, 3)$

Then the corresponding TM will successively reach the following configurations:

Example

Assume the run of the URM, starting with $R_i$ containing $a_0 = a$, $i = 0, \ldots, n - 1$, and $a_n = 0$ for $i = n, \ldots, 0$ is as follows:

Instruction $R_0 \ R_1 \ \ldots \ R_{n-1} \ R_0 \ \ldots \ R_{-1}$

$0 \ \ldots \ 0 \ \ldots \ 0 \ \ldots$
Example

\[ I_0 = \text{ifzero}(0, 3) \]
\[ I_1 = \text{pred}(0) \]
\[ I_2 = \text{ifzero}(1, 0) \]

We saw in the last section that a run of \( U^{(1)}(2) \) is as follows:

<table>
<thead>
<tr>
<th>Instruction</th>
<th>( R_0 )</th>
<th>( R_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( I_0 )</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>( I_1 )</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>( I_2 )</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( I_0 )</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( I_1 )</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( I_2 )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( I_0 )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( I_3 )</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

URM Stops

Corresponding TM Simulation

\[ I_0 = \text{ifzero}(0, 3) \]
\[ I_1 = \text{pred}(0) \]
\[ I_2 = \text{ifzero}(1, 0) \]

<table>
<thead>
<tr>
<th>Instruction</th>
<th>( R_0 )</th>
<th>( R_1 )</th>
<th>State of TM</th>
<th>Content of Tape</th>
</tr>
</thead>
<tbody>
<tr>
<td>( I_0 )</td>
<td>2</td>
<td>0</td>
<td>( s_{\text{init}}, 0 )</td>
<td>( \text{bin}(2) \ldots \text{bin}(0) \ldots \text{bin}(0) \ldots )</td>
</tr>
<tr>
<td>( I_1 )</td>
<td>2</td>
<td>0</td>
<td>( s_0, 0 )</td>
<td>( \text{bin}(2) \ldots \text{bin}(0) \ldots \text{bin}(0) \ldots )</td>
</tr>
<tr>
<td>( I_2 )</td>
<td>1</td>
<td>0</td>
<td>( s_1, 0 )</td>
<td>( \text{bin}(2) \ldots \text{bin}(0) \ldots \text{bin}(0) \ldots )</td>
</tr>
<tr>
<td>( I_0 )</td>
<td>1</td>
<td>0</td>
<td>( s_2, 0 )</td>
<td>( \text{bin}(1) \ldots \text{bin}(0) \ldots \text{bin}(0) \ldots )</td>
</tr>
<tr>
<td>( I_1 )</td>
<td>1</td>
<td>0</td>
<td>( s_0, 0 )</td>
<td>( \text{bin}(1) \ldots \text{bin}(0) \ldots \text{bin}(0) \ldots )</td>
</tr>
<tr>
<td>( I_2 )</td>
<td>0</td>
<td>0</td>
<td>( s_1, 0 )</td>
<td>( \text{bin}(1) \ldots \text{bin}(0) \ldots \text{bin}(0) \ldots )</td>
</tr>
<tr>
<td>( I_0 )</td>
<td>0</td>
<td>0</td>
<td>( s_2, 0 )</td>
<td>( \text{bin}(0) \ldots \text{bin}(0) \ldots \text{bin}(0) \ldots )</td>
</tr>
<tr>
<td>( I_3 )</td>
<td>0</td>
<td>0</td>
<td>( s_0, 0 )</td>
<td>( \text{bin}(0) \ldots \text{bin}(0) \ldots \text{bin}(0) \ldots )</td>
</tr>
</tbody>
</table>

URM Stops TM Stops

Proof of Lemma 3.4

If we have defined this we have

- If

\[ U^{(n)}(a_0, \ldots, a_{n-1}) \downarrow, \]
\[ U^{(n)}(a_0, \ldots, a_{n-1}) \simeq c, \]

then \( U \) eventually stops with \( R_i \) containing some values \( b_i \), where \( b_0 = c \).

Then, the TM \( T \) starting with

\[ \text{bin}(a_0) \ldots \text{bin}(a_{n-1}) \]

will eventually terminate in a configuration

\[ \text{bin}(b_0) \ldots \text{bin}(b_{k-1}) \]

for some \( k \geq n \).

Therefore \( T^{(n)}(a_0, \ldots, a_{n-1}) \simeq b_0 = c \).

- If

\[ U^{(n)}(a_0, \ldots, a_{n-1}) \uparrow, \]

the URM \( U \) will loop and the TM \( T \) will carry out the same steps as the URM and loop as well.

Therefore

\[ T^{(n)}(a_0, \ldots, a_{n-1}) \uparrow, \]

again

\[ U^{(n)}(a_0, \ldots, a_{n-1}) \simeq T^{(n)}(a_0, \ldots, a_{n-1}). \]
IV.3 (b) Equivalence of URM computable and Turing computable functions

Proof of Lemma 3.4

- It follows \( U(n) = T(n) \),
  and the proof is complete, if the simulation has been introduced.
- The following slides contain a detailed proof, which will not be presented in the lecture this year.

Jump over remaining proof.

Informal description of the simulation of URM instructions.

- **Initialisation.**
  - Initially, the tape contains \( \text{bin}(a_0) \ldots \ldots \text{bin}(a_{n-1}) \).
  - We need to obtain configuration:
  \[
  \text{bin}(a_0) \ldots \ldots \text{bin}(a_{n-1}) \text{bin}(0) \ldots \ldots \text{bin}(0).
  \]
  - Achieved by
    - moving head to the end of the initial configuration
    - inserting, starting from the next blank, \( l-n \)-times 0
    - then moving back to the beginning.

Simulation of URM instructions.

- **Simulation of instruction** \( I_k = \text{succ}(j) \).
  - Need to increase \((j+1)\)st binary number by 1
  - Initial configuration:
  \[
  \text{bin}(c_0) \text{bin}(c_1) \ldots \ldots \text{bin}(c_j) \ldots \ldots \text{bin}(c_l)
  \]
  - Achieved by
    - First move to the \((j+1)\)st blank to the right. Then we are at the end of the \((j+1)\)st binary number.
  \[
  \text{bin}(c_0) \text{bin}(c_1) \ldots \ldots \text{bin}(c_j) \ldots \ldots \text{bin}(c_i)
  \]
  - It might be that we needed to write over the separating blank a 1, in which case we have:
  \[
  \text{bin}(c_0) \text{bin}(c_1) \ldots \ldots \text{bin}(c_{j-1}) \text{bin}(c_j+1) \ldots \ldots \text{bin}(c_l)
  \]
Proof of Lemma 3.4

In the latter case, shift all symbols to the left once left, in order to obtain a separating $\downarrow \downarrow$ between the $l$th and $l - 1$st entry.

We obtain

$$\tilde{\text{bin}}(c_0) \downarrow \downarrow \tilde{\text{bin}}(c_1) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_{j-1}) \downarrow \downarrow \tilde{\text{bin}}(c_j) \downarrow \cdots \downarrow \downarrow \tilde{\text{bin}}(c_l) \downarrow \downarrow$$

Otherwise, move the head to the left, until we reach the $(j + 1)$st blank to the left, and then move it once to the right.

We obtain

$$\tilde{\text{bin}}(c_0) \downarrow \downarrow \tilde{\text{bin}}(c_1) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_j - 1) \downarrow \downarrow \cdots \downarrow \downarrow \tilde{\text{bin}}(c_l) \downarrow \downarrow$$

Initially:

$$\tilde{\text{bin}}(c_0) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_j) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_l) \downarrow \uparrow$$

Finally:

$$\tilde{\text{bin}}(c_0) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_j - 1) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_l) \downarrow \uparrow$$

We have achieved

$$\tilde{\text{bin}}(c_0) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_j - 1) \downarrow \cdots \downarrow \bar{\text{bin}}(c_l) \downarrow \uparrow$$

Done as follows:

Simulation of instruction $I_k = \text{pred}(j)$.

Assume the configuration at the beginning is:

$$\text{bin}(c_0) \downarrow \cdots \downarrow \text{bin}(c_j) \downarrow \cdots \downarrow \text{bin}(c_l)$$

We want to achieve

$$\tilde{\text{bin}}(c_0) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_j - 1) \downarrow \cdots \downarrow \bar{\text{bin}}(c_l) \downarrow \uparrow$$

We have

$$\text{bin}(c_0) \downarrow \cdots \downarrow \text{bin}(c_j) \downarrow \cdots \downarrow \text{bin}(c_l - 1) \downarrow \cdots \downarrow \text{bin}(c_l + 1) \downarrow \uparrow$$

Replace it by $b_0 \cdots b_k 0 11 \cdots 1 \downarrow \downarrow$.

Done as for succ.

We have achieved

$$\tilde{\text{bin}}(c_0) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_j - 1) \downarrow \cdots \downarrow \bar{\text{bin}}(c_l) \downarrow \uparrow$$

Move back to the beginning:

$$\text{bin}(c_0) \downarrow \cdots \downarrow \text{bin}(c_j - 1) \downarrow \cdots \downarrow \text{bin}(c_l) \downarrow \uparrow$$

We have

$$\text{bin}(c_0) \downarrow \cdots \downarrow \text{bin}(c_j - 1) \downarrow \cdots \downarrow \text{bin}(c_l + 1) \downarrow \uparrow$$

We have achieved

$$\text{bin}(c_0) \downarrow \cdots \downarrow \text{bin}(c_j - 1) \downarrow \cdots \downarrow \text{bin}(c_l) \downarrow \uparrow$$

Move back to the beginning:

$$\text{bin}(c_0) \downarrow \cdots \downarrow \text{bin}(c_j - 1) \downarrow \cdots \downarrow \text{bin}(c_l) \downarrow \uparrow$$
Proof of Lemma 3.4

- **Simulation of instruction** $I_k = \text{ifzero}(j, k')$.
  - Move to $j + 1$st binary number on the tape.
  - Check whether it contains only zeros.
    - If yes, switch to state $s_{k', 0}$.
    - Otherwise switch to state $s_{k+1, 0}$.

This completes the simulation of the URM $U$.

---

Halting Problem with no Inputs

**Theorem (3.8)**

It is undecidable, whether a Turing machine started with a blank tape terminates.

**Proof:**

- Let $\text{Halt'}(e) : \iff e$ is a code for a Turing machine $T$ and $T$ started with a blank tape terminates.
- Assume $\text{Halt'}$ were decidable.

Then we can decide $\text{Halt}(e, n)$ as follows:

- Assume inputs $e, n$.
- If $e$ is not a code for a Turing machine, we return 0.
- Otherwise, let $\text{encode}(T) = e$.
- Define a Turing machine $V$ as follows:
  - $V$ first writes $\text{bin}(n)$ on the tape and moves head to the left most bit of $\text{bin}(n)$.
  - Then it executes the Turing machine $T$.
- We have
  - $V$, run with blank tape, terminates iff $T$ run with tape containing $\text{bin}(n)$ terminates iff $T^{(1)}(n) \downarrow$ iff $\{e\}(n) \downarrow$. 

IV.3 (c) Undecidability of the Turing Halting Problem

Halting Problem with no Inputs

$V$, run with blank tape, terminates iff $\{e\}(n) \downarrow$.

- Let $\text{encode}(V) = e'$. Then
  \[
  \text{Halt}'(e') \iff \text{Halt}(e, n)
  \]
- Therefore using the decidability of $\text{Halt}'$ we can decide $\text{Halt}(e, n)$.
- So we have decided $\text{Halt}$, a contradiction.

No Additional Material

For this subsection no additional material has been added yet.