IV.3 (a) Definition of the Turing Machine

For this subsection no additional material has been added yet.
IV.3 (b) Equivalence of URM computable and Turing computable functions

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 $\{0,1,\downarrow,\uparrow\}$.

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").

Notation: $\tilde{\text{bin}}$
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 $s_0 \downarrow \uparrow s_1 \downarrow \uparrow s_2 \downarrow \uparrow \cdots \downarrow \uparrow s_k$ where $s_i$ is a binary representation of $n_i$" (where $n_i$ is the simulated content of register $R_i$).

We define $\tilde{\text{bin}}(n)$ as one of the binary representations of $s$.

Then we can write for the above:
"The tape contains $\tilde{\text{bin}}(n_0) \downarrow \uparrow \tilde{\text{bin}}(n_1) \downarrow \uparrow \cdots \downarrow \uparrow \tilde{\text{bin}}(n_k)$".

So $\tilde{\text{bin}}(n)$ denotes one of the possible choices for strings $s$ s.t. $(s)_2 = n$.

- So $\tilde{\text{bin}}(1)$ can be "1", "01", "001", etc.
- In the special case 0 we treat the empty string as one of the possible representations, so $\tilde{\text{bin}}(0)$ can be "", "0", "00", "000", etc.

Proof of Lemma 3.4
When carrying out intermediate calculations, it is easier to refer to $\tilde{\text{bin}}(n)$ rather than $\text{bin}(n)$
- E.g. we can set a number on the tape easily to an element of $\tilde{\text{bin}}(0)$ 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 $\downarrow\uparrow$. 
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 \( \tilde{\text{bin}}(a_0) \rightleftharpoons \rightleftharpoons \tilde{\text{bin}}(a_{l-1}) \).
- 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
  - \( 4 : R_2 := \text{R}_2 - 1 \) (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
    \[
    \begin{array}{c|c|c}
    R_0 & R_1 & R_2 \\
    \hline
    2 & 1 & 2 \\
    \end{array}
    \]

Example

- Then we want that, if the simulating TM is
  - in state \( s_{4,0} \),
  - with tape content \( \tilde{\text{bin}}(2) \rightleftharpoons \tilde{\text{bin}}(1) \rightleftharpoons \tilde{\text{bin}}(3) \)
- it should reach
  - state \( s_{5,0} \)
  - with tape content \( \tilde{\text{bin}}(2) \rightleftharpoons \tilde{\text{bin}}(1) \rightleftharpoons \tilde{\text{bin}}(2) \)
Proof of Lemma 3.4

Furthermore, we need initial states $s_{\text{init},0}, \ldots, s_{\text{init},j}$ and corresponding instructions, s.t.

- if the TM initially contains $\overline{\text{bin}}(b_0) \overline{\text{bin}}(b_1) \ldots \overline{\text{bin}}(b_{n-1})$
- it will reach state $s_{0,0}$ with the tape containing $\overline{\text{bin}}(b_0) \overline{\text{bin}}(b_1) \ldots \overline{\text{bin}}(b_{n-1}) \overline{\text{bin}}(0) \overline{\text{bin}}(0) \ldots \overline{\text{bin}}(0)$ $l - n$ times

Example

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

$U(1)(a) \equiv 0$. 

Assume the run of the URM, starting with $R_i$ containing $a_{0,i} = a_i \, \text{for} \, i = 0, \ldots, n - 1, \text{and} \, a_{0,i} = 0 \, \text{for} \, i = n, \ldots, l - 1$ is as follows:

<table>
<thead>
<tr>
<th>Instruction</th>
<th>$R_0$</th>
<th>$R_1$</th>
<th>$R_{n-1}$</th>
<th>$R_n$</th>
<th>$R_{l-1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$a_0$</td>
<td>$a_1$</td>
<td>$a_{n-1}$</td>
<td>0</td>
<td>$a_l$</td>
</tr>
<tr>
<td>$k_0$</td>
<td>$a_{0,0}$</td>
<td>$a_{0,1}$</td>
<td>$a_{0,n-1}$</td>
<td>$a_{0,n}$</td>
<td>$a_{0,l-1}$</td>
</tr>
<tr>
<td>$k_1$</td>
<td>$a_{1,0}$</td>
<td>$a_{1,1}$</td>
<td>$a_{1,n-1}$</td>
<td>$a_{1,n}$</td>
<td>$a_{1,l-1}$</td>
</tr>
<tr>
<td>$k_2$</td>
<td>$a_{2,0}$</td>
<td>$a_{2,1}$</td>
<td>$a_{2,n-1}$</td>
<td>$a_{2,n}$</td>
<td>$a_{2,l-1}$</td>
</tr>
</tbody>
</table>

Then the corresponding TM will successively reach the following configurations:

<table>
<thead>
<tr>
<th>State</th>
<th>Tape contains</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_{\text{init},0}$</td>
<td>$\overline{\text{bin}}(a_0) \overline{\text{bin}}(a_1) \ldots \overline{\text{bin}}(a_{n-1})$</td>
</tr>
<tr>
<td>$s_{0,0}$</td>
<td>$\overline{\text{bin}}(a_0) \overline{\text{bin}}(a_1) \ldots \overline{\text{bin}}(a_{n-1}) \overline{\text{bin}}(0) \overline{\text{bin}}(0) \ldots \overline{\text{bin}}(0)$</td>
</tr>
<tr>
<td>$s_{k_0,0}$</td>
<td>$\overline{\text{bin}}(a_{0,0}) \overline{\text{bin}}(a_{0,1}) \ldots \overline{\text{bin}}(a_{0,l-1})$</td>
</tr>
<tr>
<td>$s_{k_1,0}$</td>
<td>$\overline{\text{bin}}(a_{1,0}) \overline{\text{bin}}(a_{1,1}) \ldots \overline{\text{bin}}(a_{1,l-1})$</td>
</tr>
<tr>
<td>$s_{k_2,0}$</td>
<td>$\overline{\text{bin}}(a_{2,0}) \overline{\text{bin}}(a_{2,1}) \ldots \overline{\text{bin}}(a_{2,l-1})$</td>
</tr>
</tbody>
</table>
Example

0: if $R_0 = 0$ then goto 3
1: $R_0 := R_0 - 1$
2: if $R_1 = 0$ then goto 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>
<th>State of TM</th>
<th>Content of Tape</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>0</td>
<td>$s_{init,0}$</td>
<td>bin(2)</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>$s_0,0$</td>
<td>bin(2)</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>$s_1,0$</td>
<td>bin(1)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>$s_2,0$</td>
<td>bin(1)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>$s_3,0$</td>
<td>bin(0)</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>$s_4,0$</td>
<td>bin(0)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$s_5,0$</td>
<td>bin(0)</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>$s_6,0$</td>
<td>bin(0)</td>
</tr>
</tbody>
</table>

URM Stops

Proof of Lemma 3.4

If we have defined this we have

- If

  $U^{(n)}(a_0, \ldots, a_{n-1}) \downarrow$,
  
  $U^{(n)}(a_0, \ldots, a_{n-1}) \simeq b_i$,

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

Then, the TM $T$ starting with

$\text{bin}(a_0)_{\ldots, \ldots, \ldots, \text{bin}(a_{n-1})}$

will eventually terminate in a configuration

$\text{bin}(b_0)_{\ldots, \ldots, \ldots, \text{bin}(b_{k-1})}$

for some $k \geq n$.

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

Proof of Lemma 3.4

- 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 \( \text{bin}(a_0) \cdots \text{bin}(a_{n-1}) \).
We need to obtain configuration:
\[ \text{bin}(a_0) \cdots \text{bin}(a_{n-1}) \text{bin}(0) \cdots \text{bin}(0) \]
\[ l - n \text{ times} \]

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** \( k : R_j := R_j + 1 \).

Need to increase \((j + 1)\)st binary number by \( 1 \)

Initial configuration:
\[ \text{bin}(c_0) \cdots \text{bin}(c_j) \cdots \]
\[ \text{bin}(c_l) \uparrow \]

\( s_{k,0} \)

▶ First move to the \((j + 1)\)st blank to the right. Then we are at the end of the \((j + 1)\)st binary number.
\[ \text{bin}(c_0) \cdots \text{bin}(c_j) \cdots \]
\[ \text{bin}(c_l) \uparrow \]

▶ Now perform the operation for increasing by \( 1 \) as above.

At the end we obtain:
\[ \text{bin}(c_0) \cdots \text{bin}(c_j + 1) \cdots \]
\[ \text{bin}(c_l) \uparrow \]

▶ It might be that we needed to write over the separating blank a \( 1 \), in which case we have:
\[ \text{bin}(c_0) \cdots \text{bin}(c_j - 1) \text{bin}(c_j + 1) \cdots \]
\[ \text{bin}(c_l) \uparrow \]
Proof of Lemma 3.4

In the latter case, shift all symbols to the left once left, in order to obtain a separating \(\downarrow\) between the \(l\)th and \(l - 1\)st entry. We obtain
\[
\sim \text{bin}(c_0) \downarrow \sim \text{bin}(c_1) \downarrow \cdots \downarrow \sim \text{bin}(c_{j - 1}) \downarrow \sim \text{bin}(c_j + 1) \downarrow \cdots \downarrow \sim \text{bin}(c_l) \downarrow
\]

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
\[
\sim \text{bin}(c_0) \downarrow \sim \text{bin}(c_1) \downarrow \cdots \downarrow \sim \text{bin}(c_j - 1) \downarrow \cdots \downarrow \sim \text{bin}(c_l) \downarrow
\]

Done as follows:

\[\text{Simulation of instruction } k : R_j := R_j - 1.\]

Assume the configuration at the beginning is:
\[
\sim \text{bin}(c_0) \downarrow \sim \text{bin}(c_1) \downarrow \cdots \downarrow \sim \text{bin}(c_j) \downarrow \cdots \downarrow \sim \text{bin}(c_l) \downarrow
\]

We want to achieve
\[
\sim \text{bin}(c_0) \downarrow \sim \text{bin}(c_1) \downarrow \cdots \downarrow \sim \text{bin}(c_j - 1) \downarrow \cdots \downarrow \sim \text{bin}(c_l) \downarrow
\]

Done as follows:

Initially:
\[
\sim \text{bin}(c_0) \downarrow \cdots \downarrow \sim \text{bin}(c_j) \downarrow \cdots \downarrow \sim \text{bin}(c_l) \downarrow
\]

Finally:
\[
\sim \text{bin}(c_0) \downarrow \cdots \downarrow \sim \text{bin}(c_j - 1) \downarrow \cdots \downarrow \sim \text{bin}(c_l) \downarrow
\]

Move to end of the \((j + 1)\)st number.

Check, if the number consists only of zeros or not.

If it consists only of zeros, \(R_j := R_j - 1\) doesn't change anything.

Otherwise, number is of the form \(b_0 \cdots b_k 00 \cdots 0\), \(l'\) times

Replace it by \(b_0 \cdots b_k 11 \cdots 1\), \(l'\) times

Done as for \(R_j := R_j + 1\).
IV.3 (b) Equivalence of URM computable and Turing computable functions

Proof of Lemma 3.4

- **Simulation of instruction** $k: \text{if } R_j = 0 \text{ then goto } k'$.
  - 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 $U$.

---

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

Halting Problem with no Inputs

**Theorem (3.8)**

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

**Proof:**

- Let

  \[
  \text{Halt'}(e) :\Leftrightarrow e \text{ is a code for a Turing machine } T \text{ and } T \text{ started with a blank tape terminates}
  \]

- Assume Halt' were decidable.

  - Then we can decide 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 encode(V) = e’. Then

\[
\text{Halt}'(e') \iff \text{Halt}(e, n)
\]

- Therefore using the decidability of Halt’ we can decide Halt(e, n).
- So we have decided Halt, a contradiction.

No Additional Material

For this subsection no additional material has been added yet.