Lec 16. Universal TM, diagonal method

Eunjung Kim

Nondetermistic Turing machine

D shall simulate the nondeterministic TM *N* as follows.

1. Initialization: *D* initialize the simulation tape by erasing its contents and writing the initial input string to *M* by copying the contents in the input tape onto the simulation tape.

Nondeterministic Turing machine

D shall simulate the nondeterministic TM N as follows.

1. Initialization: D initialize the simulation tape by erasing its contents and writing the initial input string to M by copying the contents in the input tape onto the simulation tape.
2. Simulation Given the content $s = (s_1, \ldots, s_\ell)$ in the address tape, D reads s_i in the address tape, update the contents in the simulation tape accordingly.
    - If the simulation following the instructions of s ends in an accept state of N, then D accepts; this is when D terminates.

Nondetermistic Turing machine

D shall simulate the nondeterministic TM N as follows.

1. Initialization: D initialize the simulation tape by erasing its contents and writing the initial input string to M by copying the contents in the input tape onto the simulation tape.
2. Simulation Given the content $s = (s_1, \ldots, s_\ell)$ in the address tape, D reads s_i in the address tape, update the contents in the simulation tape accordingly.
   - If the simulation following the instructions of s ends in an accept state of N, then D accepts; <u>this is when D terminates</u>.
   - If the simulation meets an invalid move in s, ends in reject state or is completed, abort the current simulation.
3. Next node in BFS tree: update the current s in the address tape to represent the next one, and repeat the above.

Next node in BFS tree

SIMULATING s

Nondetermistic Turing machine

We can further polish D as follows.

- While you're executing the instructions over all $s \in \{0, \ldots, p\}^\ell$ of length ℓ, D remembers if there is any active branch of length ℓ; i.e. all moves in s are valid and it did not end in a halting state.
- After executing the instruction $s \in p^\ell$, if there is no active branch, D rejects the input instead of increasing s.

Observe: D accepts/rejects a string $w \in \Sigma^*$ iff N accepts/rejects w.

Hardwired TM to programmable TM!

- TM is defined by its transition function.
- This means that one TM can compute (recognize or decide) a single function (language).
- One TM, useful for a single purpose only.
  $\rightsquigarrow$ hardwired as produced in the factory.
- But computer as we know is an all-round player with programs.
  $\rightsquigarrow$ stored-program computer, universal.
- Universal TM, the mathematical model that embodies this historic transition.

Hardwired TM to programmable TM!

Key insight

TM is not only a computing device which ‘receives’ an input string.

A Turing machine itself can be an input string!!!

(once <u>appropriately</u> encoded as a string).

Hardwired TM to programmable TM!

Key insight

TM is not only a computing device which 'receives' an input string.

A Turing machine itself can be an input string!!!

(once appropriately encoded as a string).

- Turing proved that a universal TM exists. A couple of legendary scientists and mathematicians including Turing himself realized this concept in the 1940's, the earliest versions of modern-day computers.

Turing machine which reads (the encoding of) another TM

- Let's build an all-round TM U which reads an arbitrary TM M and an input w to M, and does what M would do on the input w.

Turing machine which reads (the encoding of) another TM

- Let's build an all-round TM U which reads an arbitrary TM M and an input w to M, and does what M would do on the input w.

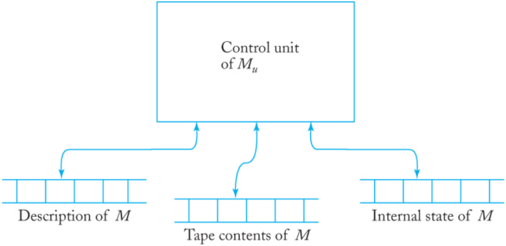

Figure 10.16, Peter Linz 2014.

- If U can simulate any other TM, with U we can do any computation that any TM M can do by loading (reading) M and an input to M; instead of using various of TM's for various purposes, we use a single TM U - a universal TM.

Turing machine which reads (the encoding of) another TM

- Let's build an all-round TM U which reads an arbitrary TM M and an input w to M, and does what M would do on the input w.

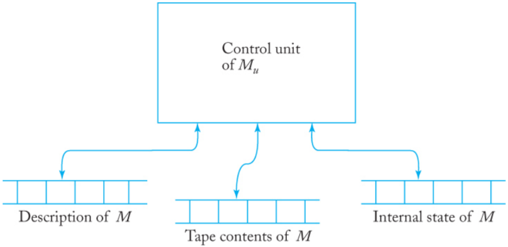

Figure 10.16, Peter Linz 2014.

- If U can simulate any other TM, with U we can do any computation that any TM M can do by loading (reading) M and an input to M; instead of using various of TM's for various purposes, we use a single TM U - a universal TM.

Encoding a Turing machine

Encoding of TM

1. Consider TM $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$.
2. Codifying each component of M.
   - $Q = \{q_1, \ldots, q_s\}$
   - q_1 is interpreted as the start state, q_2 accept state, q_3 reject state.
   - $\Gamma = \{a_1, \ldots, a_t\}$.
   - Left header move is associated with 1, Right header move with 2.
3. A transition $\delta(q_h, a_i) = (q_j, a_k, L)$ is represented as a 5-tuple of numbers; $(h, i, j, k, 1)$
4. 5-tuple expression of a transition as a $\{0, 1\}$-string: $0^h10^i10^j10^k10$
5. TM is expressed as a $\{0, 1\}$ string by
   - encoding each transition using the above scheme
   - concatenation all transitions, each transition separated by 11 (a pair of 1's).

Encoding TM: example

TM $M = (\{q_1, q_2, q_3\}, \{0, 1\}, \{0, 1, B\}, \delta, q_1, q_{accept} = q_2, q_{reject} = q_3)$.

$\delta(q_1, 1) = (q_3, 0, R)$	0100100010100
$\delta(q_3, 0) = (q_1, 1, R)$	0001010100100
$\delta(q_3, 1) = (q_2, 0, R)$	00010010010100
$\delta(q_3, B) = (q_3, 1, L)$	0001000100010010

01001000101001100010101001001100010010010100110001000100010010

Universal Turing machine

(Rather informal) definition

Let τ be an encoding scheme of TM and an input string.
A Turing machine U is called a universal Turing machine with encoding scheme τ if it accepts a string s if and only

1. $s = \tau(M) \circ \tau(w)$ for some TM M and a string w over the alphabet of M, and
2. M accepts w.

Universal Turing machine

Kurt Gödel showed that there exists a universal Turing machine U.

U has 3 tapes.

1. Input tape: the encoding of M and the encoding of an input w to M (separated by 111) is loaded here. Never altered.
2. Simulation tape: whatever happens in the (single) tape of M is simulated (replicated) here.
3. State tape: the state of M during the execution on w is written here.

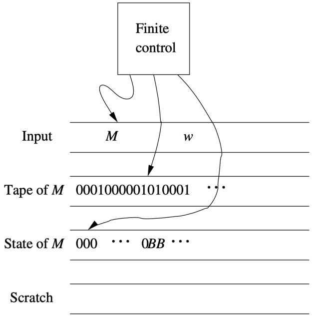

Figure 1.61, Sipser 2012.

All languages TM-recognizable?

No. A fundamental consequence of uncountability of $\mathbb{R}$, the set of real numbers, and the fact that TM has a finite description.

Outline

Consider the alphabet $\{0, 1\}$.

1. $\{0, 1\}^*$ have the same size as $\mathbb{N}$.
2. the collection of all languages over $\{0, 1\}$ have the same size as $2^{\mathbb{N}}$.
3. $2^{\mathbb{N}}$ is uncountable while $\mathbb{N}$ is countable.
4. the collection of all Turing machines have the same size as $\mathbb{N}$
5. at least one language over $\{0, 1\}$ does NOT admit TM recognizing it.

Countable versus uncountable

The size of a set

- A function φ from A to B is a bijection if it is one-to-one (injection) and onto (surjection).
- We say that two sets A and B have the same size if there is a bijection from A to B.
- A set is countable if it is finite or has a bijection to $\mathbb{N}$.
- A set is uncountable if it is not countable.

Countable sets

Having a bijection from $\mathbb{N}$ to a set A is equivalent to listing all elements of A (the list can be infinite).

- $2\mathbb{N}$
- $\mathbb{Z}$
- $\{0, 1\}^*$
- Σ^* for any finite set Σ
- the set of all rational numbers

Countable sets: rational numbers

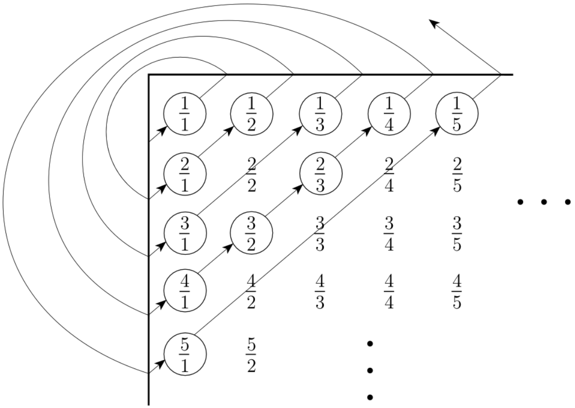

Figure 4.16, Sipser 2012.

Uncountable sets

$\mathbb{R}$ and $2^{\mathbb{N}}$ are uncountable

1. Suppose the contrary; let φ be a bijection from $\mathbb{N}$ to $2^{\mathbb{N}}$ (or to $\mathbb{R}$).
2. Goal: construct an element $X \in 2^{\mathbb{N}}$ (or $x \in \mathbb{R}$) which is not listed by φ $\rightsquigarrow$ contradition.
3. Constructing such an element is possible via diagonal argument.

φ lists all real numbers in $[0, 1]$, i.e. a bijection from $\mathbb{N}$ to $[0, 1]$.

- Rows are indexed by $1, 2, \ldots$, i.e. $\mathbb{N}$
- i-th row corresponds to the real number $\varphi(i)$, with j-th entry being the j-th digit after the decimal separator.
- Diagonalization step: construct a new real number which is not listed by φ by perturbing all the diagonal entries.

Diagonal argument for uncountability of $[0, 1]$

0.8	1	3	4	2	0	8 $\cdots$
0.0	1	1	2	1	9	0 $\cdots$
0.2	0	3	1	4	1	3 $\cdots$
0.7	0	3	4	4	1	3 $\cdots$
0.1	0	2	7	4	9	3 $\cdots$
0.3	1	0	3	6	0	1 $\cdots$
0.2	4	3	1	4	7	7 $\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

$\rightsquigarrow$ consider a real number $x = 0.\bar{8}\bar{1}\bar{3}\bar{4}\bar{4}\bar{0}\bar{7}\cdots = 0.7243186\cdots$
The perturbation on each digit can be arbitrary (just avoid using 0 and 9).

x is not listed by φ!

Diagonal argument for uncountability of $2^{\mathbb{N}}$

Diagonal argument: suppose φ lists all elements in $2^{\mathbb{N}}$.

- Rows and columns are indexed by $1, 2, \ldots$, i.e. $\mathbb{N}$
- i-th row corresponds to (the indicator vector of) the set $\varphi(i)$ in $2^{\mathbb{N}}$, with j-th entry being 1 if and only if j is in the set.
- Diagonalization step: construct a new set which is not listed by φ by flipping all the diagonal entries.

0	0	1	1	1	0	1 $\cdots$
0	1	1	1	1	1	0 $\cdots$
1	0	1	1	0	1	0 $\cdots$
1	0	1	0	1	1	0 $\cdots$
0	0	1	1	1	0	1 $\cdots$
0	1	0	1	1	0	1 $\cdots$
1	1	1	1	0	0	0 $\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

Consider the set

$$X = \bar{0}\bar{1}\bar{1}\bar{0}\bar{1}\bar{0}\bar{0}\cdots = 1001011\cdots$$

$\rightsquigarrow$ X is not listed by φ!

Countable or uncountable?

Use the fact that if there is a bijection between A and B, then there is a bijection between 2^A and 2^B.

- The collection of all languages over $\{0, 1\}$?
- $\{\tau(M) \in \{0, 1\}^* : M \text{ is a Turing machine}\}$ for a fixed encoding scheme τ?
- The collection of all languages over $\{0, 1\}$ recognizable by some Turing machine?

Language unrecognizable by TM

- The collection of all languages over $\{0, 1\}$? Uncountable.
- $\{\tau(M) \subseteq \{0, 1\}^* : M \text{ is a Turing machine}\}$? Countable.
- The collection of all languages over $\{0, 1\}$ recognizable by some Turing machine? Countable.

Language unrecognizable by TM

- The collection of all languages over $\{0, 1\}$? Uncountable.
- $\{\tau(M) \subseteq \{0, 1\}^* : M \text{ is a Turing machine}\}$? Countable.
- The collection of all languages over $\{0, 1\}$ recognizable by some Turing machine? Countable.

Unrecognizable

There is a language that cannot be recognized by any Turing machine.