Turing Machine

Pierre-Alain Fouque

M1 cyberschool

Formal definition of a Turing Machine

- Def: A Turing Machine is a 7-uple ($Q,\mathbf{\Sigma},\mathbf{\Gamma},\boldsymbol{\delta},q_0,q_{acc},q_{rej}$) where Q, $\mathbf{\Sigma},\mathbf{\Gamma}$ are all finite sets and

1. Q is the set of states
2. $\mathbf{\Sigma}$ is the input alphabet not containing the special blank symbol B
3. $\mathbf{\Gamma}$ is the tape alphabet, where $B \in \mathbf{\Gamma}$ and $\mathbf{\Sigma} \subseteq \mathbf{\Gamma}$
4. $\boldsymbol{\delta}$:$Q \times \mathbf{\Gamma} \rightarrow Q \times \mathbf{\Gamma} \times \{L,R\}$ is the transition function
5. $q_{acc} \in Q$ is the accept state, and
6. $q_{rej} \in Q$ is the reject state and $q_{acc} \neq q_{rej}$

Other model: input tape in RO mode, output tape: WO mode and working tapes in RW mode, with infinite tape on the left and right

Configurations

- Turing machine computation: update the current state, the current tape content, and the current head location.
- Configuration: uqv with $u,v \in \boldsymbol{\Gamma}^*$, $q \in Q$, the head is on the 1^{st} letter of v
- Configuration C_1 yields configuration C_2, $C_1 \rightsquigarrow C_2$ in a single step.
- Assume that $a,b,c \in \boldsymbol{\Gamma}$ and $u,v \in \boldsymbol{\Gamma}^*$, and q_i, $q_j \in Q$:
    - uaq_ibv yields uq_jacv if $\boldsymbol{\delta}(q_i,b)=(q_j,c,L)$ or uaq_ibv yields $uacq_jv$ if $\boldsymbol{\delta}(q_i,b)=(q_j,c,R)$
    - Start configuration: q_0w,
    - Accepting configuration: state is in q_{acc},
    - Rejecting configuration: state is in q_{rej},
    - Halting configuration: either in q_{acc}, or q_{rej}

Computation and Language

- Turing Machine M accept input w if there exists a sequence of configurations C_1, C_2, …, C_k st:
    1. C_1 is the start configuration of M on input w
    2. Each C_i yields C_{i+1}, and
    3. C_k is an accepting configuration.
- The collection of strings that M accepts is the language of M, L(M)
- Def: Call a language Turing-recognizable if some Turing machine recognizes it. (can fail to accept, either reject or looping)
- Def: Call a language Turing-decidable or decidable if some Turing machine decides it. (always make a decision accept or reject)

Encoding, Problems and Languages

- The difficulty of a problem depends on the encoding
- Goal of algorithm: solving a generic problem, not a specific instance
- Example: « Determine if an integer n is a prime number ? »
- PRIMALITY: (decision problem)
  - Input: an integer N (is N given in prime number factorization or in binary ?)
  - Output: Is N prime ?
- SORTING: (search problem)
  - Input: a list ℓ of integers
  - Output: sort ℓ by increasing order
- For decision problem, L = {w$\in$ $\mathbf{\Sigma}$|w is the encoding of accepting strings}
- For search problem, f: $\mathbf{\Sigma}^* \rightarrow \mathbf{\Sigma}^*$ is a function
- We are interested in decision problem, but there is connection between them

Time and Space Complexity

- Let M be a Turing machine and w a string on **Σ**. If M(w) halt:
  - Time complexity: is the number of computation steps to compute M(w)
  - Space complexity: is the number of locations visiting during the computation on the working tapes

- Thm: Every multitape Turing machine has an equivalent single tape Turing machine
  - Increase the tape alphabet with marked letter ȧ, ḃ, ċ, and so on.
  - Concatenate the tapes on a single tape with # separators
  - Simulate a transition on the multitape machine by scanning all the single tape and each time we encountered a marked letter, remember it on the state.

Examples of Turing Machine

- Test if a binary string is even ?
- $\boldsymbol{\Sigma}$={0,1}, $\boldsymbol{\Gamma}$={0,1,B}, Q={q_0,q_1,q_a,q_r},
- $\boldsymbol{\delta}$(q_0,(u,B))=(q_0,(B,B),(R,S,S)) for all u∈{0,1}: to the right, up to B
- $\boldsymbol{\delta}$(q_0,(B,B))=(q_1,(B,B),(L,S,S)): go one step to the Left
- $\boldsymbol{\delta}$(q_1,(0,B))=(q_a,(B,B),(S,S,S)): accept if the last bit is a 0
- $\boldsymbol{\delta}$(q_1,(1,B))=(q_r,(B,B),(S,S,S)): reject if the last bit is a 1
- All other transitions will not happened. The output tape is not used.

Example of a Turing machine

- Multiply by 2 a given integer ?
- Test if a string contains as many a than b ?
- Compute x+y is the input is w=x#y with x,y $\in\{0,1\}$?
- Decide the language of 2^n 0-bit for any integer n ?
- If w#w with w $\in\{0,1\}^*$?

High-level description of $C=\{a^i b^j c^k \mid i*j=k \; i,j,k \geq 1\}$

- M= On input string w:
  1. Scan the input from left to right to be sure that it is a member of $a^+b^+c^+$ and reject if it isn't
  2. Return the head to the first letter
  3. Cross off an a and scan to the right until a b occurs. Shuttle between the b's and c's, crossing off one of each until all b's are gone.
  4. Restore the crossed off b's and repeat stage 3 if there is another a to cross off. If all a's are crossed off, check whether all c's also are crossed, It yes, accept, otherwise reject

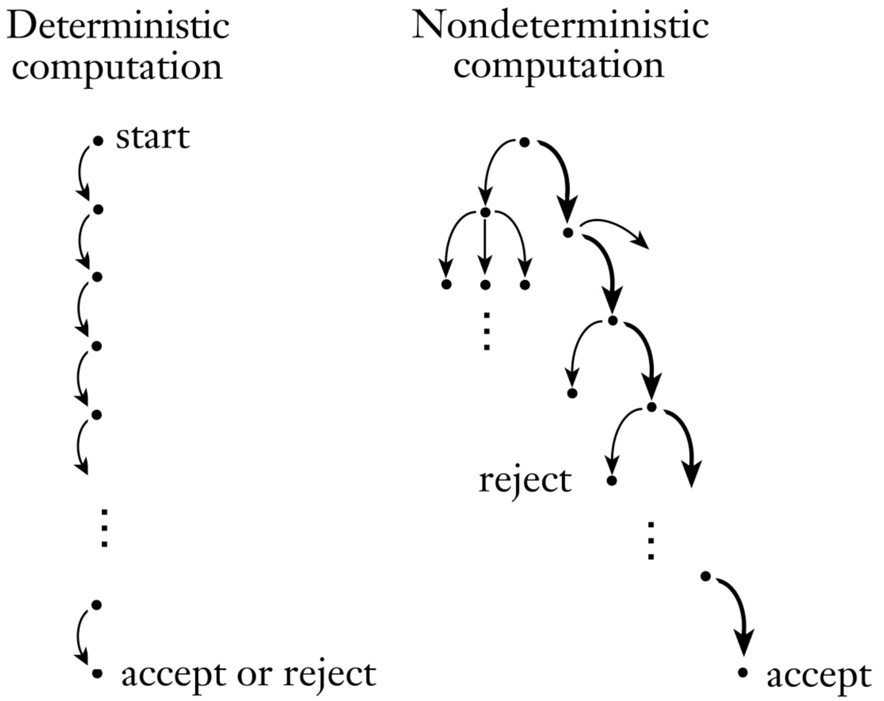
$E=\{\#x_1\#x_2\#\ldots\#x_n \mid x_i \in \{0,1\}^* \text{ and } x_i \neq x_j \text{ for each } i \neq j\}$

- M works by comparing x_1 with x_2, through x_n, then by comparing x_2 with x_3 to x_n, and so on. An informal description of M deciding E is:

- M= On input w:
  1. Place a mark on top of the leftmost tape symbol. If that symbol was a blank, accept. It that symbol was a #, continue with the next stage. Otherwise reject.
  2. Scan right to the next # and place a mark on top of it. If no # is encountered before a blank symbol, only x_1 was present, so accept.
  3. By zig-zagging, compare the two strings to the right of the marked #s. If they are equal, reject.
  4. Move the rightmost of the two marks to the next # symbol to the right. If no # is encountered before a blank, move the leftmost mark to the next # to its right and rightmost mark to the # after that. This time, if no # is available for the rightmost mark, all the strings have been compared, so accept.
  5. Go to Stage 3.

Non-deterministic Turing machines

- $\boldsymbol{\delta}$:Qx $\boldsymbol{\Gamma} \rightarrow \boldsymbol{\mathcal{P}}$(Qx $\boldsymbol{\Gamma}$x{L,R}) is the transition function with $\boldsymbol{\mathcal{P}}$ is the set of transitions.
- Thm: Every non-deterministic Turing machine has an equivalent deterministic Turing machine
  - Simulate the Non-deterministic machine in a breadth first search and not depth first search
- Corollary: A language is Turing-recognizable iff some non-deterministic Turing machine recognizes it.
- Corollary: A language is decidable iff some non-deterministic Turing machine decides it.



Non-deterministic TM accepts, if there is one accepting path in the tree

Definition of an algorithm and processors

- Church-Turing thesis:
  - the intuitive notion of algorithms equals Turing machine algorithms

- Universal Turing Machine U(<M>,w):
  - Is able to timulate any other Turing machine from the description of that machine.
  - Take as input the encoding of a Turing machine and a string w, and is able to simulate the computation of M on w.
  - It can be seen as a processor able to execute any programs

Decidable languages

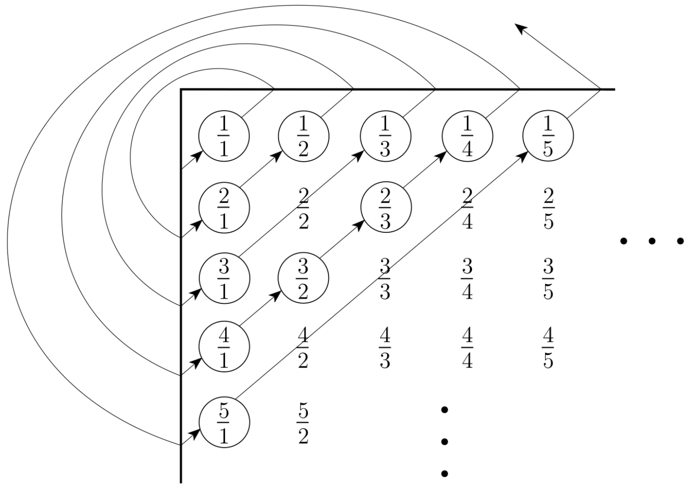
- A_{DFA}={<B,w>| B is a DFA that accepts input string w}
- Thm: A_{DFA} is a decidable language
  - Describe a TM that decides A_{DFA}.
  - M= On input <B,w>, where B is a DFA and w is a string
    1. Simulate B on input w
    2. If the simulation ends in an accept state, accept, if it is in a non-accept state, reject
- A_{NFA}={<B,w>| B is a NFA that accepts input string w}
- Thm: A_{NFA} is a decidable language
  - Convert the NFA into a DFA as we saw in the last course
- EQ_{DFA}={<A,B>| A,B are DFA and L(A)=L(B)} is also decidable (difference symmetric)
- Thm: Regular ⊆Decidable ⊆Turing-recognizable

The halting problem

- What sort of problems are unsolvable by computer ?
- A_{TM}={<M,w>| M is a TM and M accepts w}
- Thm: A_{TM} is undecidable
  - A_{TM} is Turing-recognizable. (Recognizers are more powerful than deciders.)
  - The universal Turing machine U recognizes A_{TM}:
    - U = On input <M,w> with M a TM and w is a string:
      1. Simulate M on input w
      2. If M ever enters its accept state, accept; if M ever enters its reject state, reject
- This machine can loops on <M,w> which is why U does not decide A_{TM}

Diagonalization Method

- Georg Cantor (1873): comparing size of infinite size ?
- Correspondance: exhibit a bijection between two sets
- Examples:
  - Even integers: f(n)=2n. The two sets have the same size
  - Def: A set A is countable if either it is finite or it has the same size as $\mathbb{N}$
  - $\mathbb{Q}$ set of rational numbers is also countable
  - $\mathbb{R}$ is uncountable



n	$f(n)$
1	$3.\underline{1}4159\ldots$
2	$55.5\underline{5}555\ldots$
3	$0.12\underline{3}45\ldots$
4	$0.500\underline{0}0\ldots$
$\vdots$	$\vdots$

$$x = 0.4641\ldots$$

Some languages are not Turing-recognizable

- To show that the set of all TM is countable: the set of all strings $\boldsymbol{\Sigma}^*$ is countable for any alphabet. With only finitely many strings of each length, we may form a list of $\boldsymbol{\Sigma}^*$ by writing down all strings of length 0, length 1, length 2, and so on.
- The set of all TM is countable because each TM M has an encoding into a string <M>. If we omit strings that are not legal TM, we have the list of all TM.
- To show that the set B of all languages are uncountable, we observe that the set of all infinite sequences of 0s and 1s is uncountable.
- One-to-one correspondance via the characteristic function of the language

$$
\begin{array}{rcccccccccccc}
\Sigma^* = \{ & \varepsilon, & 0, & 1, & 00, & 01, & 10, & 11, & 000, & 001, & \cdots & \} & ; \\
A = \{ & & 0, & & 00, & 01, & & & 000, & 001, & \cdots & \} & ; \\
\chi_A = & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & \cdots & & .
\end{array}
$$

The halting problem is undecidable

- A_{TM}={<M,w>|M is a TM and M accepts w}.
- Assume that A_{TM} is decidable. Let H a decider for A_{TM}:

$$H(\langle M,w\rangle) = \begin{cases} accept & \text{if } M \text{ accepts } w \\ reject & \text{if } M \text{ does not accept } w. \end{cases}$$

- We build a new machine D with H as a subroutine

$D =$ "On input $\langle M\rangle$, where M is a TM:

1. Run H on input $\langle M, \langle M\rangle\rangle$.
2. Output the opposite of what H outputs. That is, if H accepts, *reject*; and if H rejects, *accept*."

$$D(\langle M\rangle) = \begin{cases} accept & \text{if } M \text{ does not accept } \langle M\rangle \\ reject & \text{if } M \text{ accepts } \langle M\rangle. \end{cases}$$

$$D(\langle D\rangle) = \begin{cases} accept & \text{if } D \text{ does not accept } \langle D\rangle \\ reject & \text{if } D \text{ accepts } \langle D\rangle. \end{cases}$$

A Turing-unrecognizable language

- A_{TM} is undecidable. Some language can be unrecognizable.
- Thm: A language is decidable iff it is both Turing-recognizable and co-Turing-recognizable (its complement is Turing-recognizable).
  - If A is decidable, A and its complement are Turing-recognizable
  - If both A and A^c are Turing-recognizable, run M_1 and M_2 in parallel. If M_1 accept, accept, if M_2 accept, reject. (No infinite loop)

- Corollary: $A_{TM}{}^c$ is not Turing-recognizable.
  - A_{TM} is Turing-recognizable. If $A_{TM}{}^c$ were Turing-recognizable, A_{TM} would be decidable. Some previous theorem tells us that A_{TM} is not decidable, so $A_{TM}{}^c$ must not be Turing-recognizable.

The Halting problem is undecidable

- $HALT_{TM}$={<M,w>| M is a TM and M halts on input w}
- Thm: $HALT_{TM}$ is undecidable.
  - By contradiction: Assume $HALT_{TM}$ is decidable and show that A_{TM} is decidable
  - Key idea: show that A_{TM} is reducible to $HALT_{TM}$

PROOF Let's assume for the purpose of obtaining a contradiction that TM R decides $HALT_{\mathsf{TM}}$. We construct TM S to decide A_{TM}, with S operating as follows.

$S =$ "On input $\langle M, w\rangle$, an encoding of a TM M and a string w:

1. Run TM R on input $\langle M, w\rangle$.
2. If R rejects, *reject*.
3. If R accepts, simulate M on w until it halts.
4. If M has accepted, *accept*; if M has rejected, *reject*."

Clearly, if R decides $HALT_{\mathsf{TM}}$, then S decides A_{TM}. Because A_{TM} is undecidable, $HALT_{\mathsf{TM}}$ also must be undecidable.