First Order Logic

$\forall a, b, c, n \quad (a, b, c > 0) \wedge (n > 2) \Rightarrow a^n + b^n \neq c^n$

Syntax

- Vocabulary: ① For every integer $n \geq 0$, we have a finite set of n-ary **function** symbols:

  e.g. $f, h, g, \cdots$ and $+, \times, \uparrow, \cdots, 1$
  $n \quad n \quad n \qquad 2 \quad 2 \quad 2 \qquad 0$

  A zero-ary function symbol is called a constant

  ② Similar sets of n-ary **relation** symbols:

  e.g. $P, R, Q, \cdots >, <, \leq, =, \cdots$

  ③ An infinite set of **variables**

  ④ **Connectives**: $\neg$ (negation), $\wedge$ (and), $\vee$ (or)

  ⑤ 2 **quantifiers**: $\exists, \forall$

  ⑥ **Paratheses**: ( )

  Ex: For arithemetic on $\mathbb{Z}^+$ we use <u>function symols</u>:

0	constant	$S(0): 1$
S	successor is a unary function.	$S(S(0)): 2$
$+$	binary	⋮ ⋮
$\times$	binary	

  we often write $a+b$ instead of $+(a,b)$

  <u>relation</u>

  $=$ binary

- Terms ① Every variable is a **term**

  ② If f is an n-ary function symbol and $t_1, \cdots t_n$ are terms then $f(t_1, \cdots t_n)$ is a term

- Formula ① If P is an n-ary relation symbol and $t_1, \cdots t_n$ are terms, then $P(t_1, \cdots t_n)$ is a **formula**

  e.g. $S(0) > S(S(0))$

  $> (S(0), S(S(0)))$

② If A, B are formulas then

$\neg A$, $A \wedge B$, $A \vee B$ are formulas

③ If A is a formula and x is a variable

$\exists x A$, $\forall x A$ are formulas

DEF An occurance of a variable x in a formula is **bound** if it is in a subformula of the form $\forall x B$ or $\exists x B$
Otherwise, it's called **free**

e.g. $\exists y \; x = y + y$
(y: bound, x: free)

DEF A **sentence** is a formula that contains no free variables.

e.g. $\forall x \; \forall y \; (x > y) \vee \neg (x > y)$

Semantics

DEF A **model** M for a first-order logic consists of the following

- universe: a set U. The variables in a formula range over U.
- for each n-ary function symbol f, it assigns an actual function
$$f : U^n \to U.$$
- for each n-ary relation symbol R it defines a relation by assigning a True or False value to each $R(a_1, \dots a_n)$ for $(a_1, \dots a_n) \in U^n$
The "=" is always defined as the true equality relation on U.

Note that every sentence is either True or False in a given model.

Ex. Consider $\forall x \; \exists y \; (x = y + y)$ in two different models.
(i) The universe is $\mathbb{Z}$ and $+$ is defined as the standard sum $\Rightarrow$ False
(ii) The universe is $\mathbb{R}$ and $+$ is defined as the standard sum $\Rightarrow$ True

DEF A sentence that is true in every model is called a tautology

e.g. $\forall x \exists y \ (f(x)=y) \vee \neg(f(x)=y)$

Proof System

Notation

If $A_1, \cdots A_n$ and $B_1, \cdots B_m$ are formulas, then we write

$$A_1, \cdots A_n \to B_1, \cdots B_m$$

meaning

$$(A_1 \wedge \cdots \wedge A_n) \Rightarrow (B_1 \vee \cdots \vee B_m)$$

i.e. If all A_i's are true then at least one of the B_i's is true.

Inference

- Weakening
$$A_1, \cdots A_n \to B_1, \cdots B_m \Rightarrow A, A_1, \cdots A_n \to B_1, \cdots B_m$$
$$A_1, \cdots A_n \to B_1, \cdots B_m \Rightarrow A_1, \cdots A_n \to B_1, \cdots B_m, B$$

- Exchanges (can shuffle A_i's and B_i's)
$$\cdots A, A' \cdots \to B_1, \cdots B_m \Rightarrow \cdots A', A \cdots \to B_1, \cdots B_m$$
$$A_1, \cdots A_n \to \cdots B, B' \cdots \Rightarrow A_1, \cdots A_n \to \cdots B', B \cdots$$

- Contraction
$$A_1, \cdots A_n, A, A \to B_1, \cdots B_m \Rightarrow A_1, \cdots A_n, A \to B_1, \cdots B_m$$
$$A_1, \cdots A_n \to B_1, \cdots B_m, B, B \Rightarrow A_1, \cdots A_n \to B_1, \cdots B_m, B$$

- $\neg$-rule
$$\Gamma \to \Delta, A \Rightarrow \neg A, \Gamma \to \Delta$$
$$A, \Gamma \to \Delta \Rightarrow \Gamma \to \Delta, \neg A$$

- $\wedge$ introduction
$$A, B, \Gamma \to \Delta \Rightarrow (A \wedge B), \Gamma \to \Delta$$
$$\begin{cases} \Gamma \to \Delta, A \\ \Gamma \to \Delta, B \end{cases} \Rightarrow \Gamma \to \Delta, (A \wedge B)$$

- $\vee$ introduction
$$\Gamma \to \Delta, A, B \Rightarrow \Gamma \to \Delta, A \vee B$$
$$\begin{cases} A, \Gamma \to \Delta \\ B, \Gamma \to \Delta \end{cases} \Rightarrow (A \vee B), \Gamma \to \Delta$$

- Cut rule
$$\begin{cases} \Gamma \to \Delta, A \\ A, \Gamma \to \Delta \end{cases} \Rightarrow \Gamma \to \Delta$$

- $\forall$ rule $\quad A(t), \Gamma \to \Delta \Rightarrow \forall x\, A(x), \Gamma \to \Delta$

  t is a term and we replace all free occurance of x with t.

  $\Gamma \to \Delta, A(b) \Rightarrow \Gamma \to \Delta, \forall x\, A(x)$

  b: variable

- $\exists$ rule $\quad \Gamma \to A, A(t) \Rightarrow \Gamma \to \Delta, \exists x\, A(x)$

Example of a proof

$(x > y) \to (x > y)$ axiom

$\to (x > y), \neg(x > y)$ $\neg$

$\to (x > y) \vee \neg(x > y)$ $\vee$

$\to \forall y\, (x > y) \vee \neg(x > y)$ $\forall$

$\to \forall x \forall y\, (x > y) \vee \neg(x > y)$ $\forall$

Gödel's Completeness Theorem

Every tautology A has a proof that starts from the axioms $B \to B$ where B are sentenses and applies the above rules to deduce $\to A$

$\Rightarrow$ The set of tautologies is Turing recognizable

On input A:

for $i = 1, 2, \cdots$

Generate all proofs of length i

If $(\to A)$ is proved $\Rightarrow$ ACCEPT

DEF The theory of M is the set of the true sentences in that model

Theory $Th(\mathbb{Z}^+, +, \times, 0, S, =)$

- $\mathbb{Z}^+$: $\{0, 1, 2, \cdots\}$ is the universe
- $+$: the usual summation
- $\times$: the usual multiplication
- 0: corresponds to 0 in the universe
- $S(n)$: is defined to be $n+1$
- $=$: the usual equality

Consider Peano – Arithmatic

Vocabulary : function : $+, \times, 0, S(\cdot)$ ← succesor

relation : $=$

model: universe $\mathcal{U} = \{0, 1, \cdots\}$

$+, \times$ are the usual $+$ and $\times$; $S(n) = n+1$

Peano – Arithmatic Axioms

NOT tautology

① $\forall x \quad \neg(S(x) = 0)$

② $\forall x \; \forall y \quad S(x) = S(y) \Rightarrow x = y$

③ $\forall x \quad (x + 0 = x)$

④ $\forall x \; \forall y \quad x + S(y) = S(x + y)$

⑤ $\forall x \quad x \times 0 = 0$

⑥ $\forall x \; \forall y \quad x \times S(y) = x \times y + x$

⑦ "Induction"

All the known theorems in number theory (including Fermat's last theorem) are proved using this set of axioms.

Gödel's Incompletness Theorem

If H is any finite set of axioms, then there are sentences that are true in $TH(\mathbb{Z}^+, +, \times, 0, S, =)$ and have no proof using inference rules.

$\{A \mid A$ is true in the peano- arithmatic$\}$ NOT TR.

- Gödel's Incompleteness Theorem
  So there is no hope of having a "golden" set of axioms that would imply all the true statements in that model.

- How to prove Gödel's Incompleteness Theorem

  Recall that for a model M, we denote by $Th(M)$ (Theory of M) the set of all true sentences in that model.

  Observation. Suppose for a model M, there is a finite set of axioms that imply all the sentences in $Th(M)$. Then $Th(M)$ is decidable.

  proof Given a sentence A,
  for $i = 1, 2, 3, \ldots$
  generate all proofs of length i
  that start from our set of axioms
  if at any point, we prove $A \Rightarrow$ ACCEPT
  if at any point, we prove $\neg A \Rightarrow$ REJECT

  ($A, \neg A$ do NOT work for tautology)

  Remark Note that previous observation holds even if Th is just Turing Recognizable (it can be enumerated).

- Back to $Th(\mathbb{Z}^+, +, \times, 0, s, =)$, how can we prove that no finite set of axioms implies all sentences in this theory?

  We will show that $Th(\mathbb{Z}^+, +, \times, 0, s, =)$ is undecidable.

- Recall A_{TM} is undecidable.

  Note that
  M accepts $w \Leftrightarrow$ there is a computational history h that starts with w and ends in ACCEPT.

  $q_0 w \rightarrow \cdots \rightarrow u q_{acc} v$
  (Accept config)

Lecture 22

Proof of Gödel's Incompleteness Theorem

It's possible to construct a formula $\emptyset_{M,w}$ with one variable x such that x is an encoding of the computational history as just an integer: $x \in \mathbb{Z}^+$.

More precisely, $\emptyset_{M,w}(x)$ is true for a particular x iff. x Is an encoding of a computational history that starts in w and ends in an accept configuration.

(Constructing $\emptyset_{M,w}$ is complicated and involved).

Now,

$< M, w > \in A_{TM} \iff \exists x, \emptyset_{M,w}$ Is true in $(\mathbb{Z}^+, +, \times, o, s, =)$.

Theorem: Th$(\mathbb{Z}^+, +, \times, o, s, =)$ is an undecidable theory and thus does not follow from any Turing recognizable set of axioms.

Not all theories are undecidable.

Theorem: Th$(\mathbb{Z}^+, +, o, s, =)$ is decidable.

$\underline{proof}$ See the book.

Complexity Theory

Computability: is there an algorithm that decides L?

Complexity theory: is there an "efficient" algorithm that decides L?

Efficient could mean different things:

- Small number of steps (Time Complexity)
- Small amount of memory (Space Complexity)
- Small circuit representing the algorithm (Circuit Complexity)
- Communication Complexity
- Formula Complexity
- ...

Out of these: Time complexity is inarguably the most important are, for both theoretical and applied purposes.

Definition: Let M be a deterministic TM that halts on every input (deterministic decider).

The running time or time complexity of M is a function $f : \mathbb{N} \to \mathbb{N}$ where $f(n)$ is the maximum number of steps that M can take on any input of length n.

That is,

$f(n) =$ Max w such that $|w| = x$ number of steps that M takes on w.

Remarks: Note that it is often impossible to know exactly what f is, and in fact it is not that useful to know the exact value of f.

For example,

$$f(n) = \begin{cases} 3n^2+4n-1 & n \text{ is odd} \\ 2n^2+1 & n \text{ is even} \\ 6 & n \leq 5 \end{cases}$$

Tells us that $f(n)$ has quadratic growth.

That is why we use O(.) notation. $f(n) = O(n^2)$

There is another nuisance:

we discussed several variations of TM's and sowed that they are all equivalent in the sense that if L is decidable in one, then it is decidable in the other ones. For example, single tape, multiple tape, two direction memory, RAM model, ...

However, these equivalencies don't preserve the running time.

A language that requires $\Theta(n^2)$ time in single tape might be solvable in $\Theta(n)$ in the two tape model.

Definition: we say that $f : \mathbb{N} \to \mathbb{N}$ has polynomial growth if $f(n) = O(n^c)$ for some fixed constant c.

Equivalently, $\exists n_0, r > 0$ such that $f(n) < rn^c$ for all $n > n_0$.

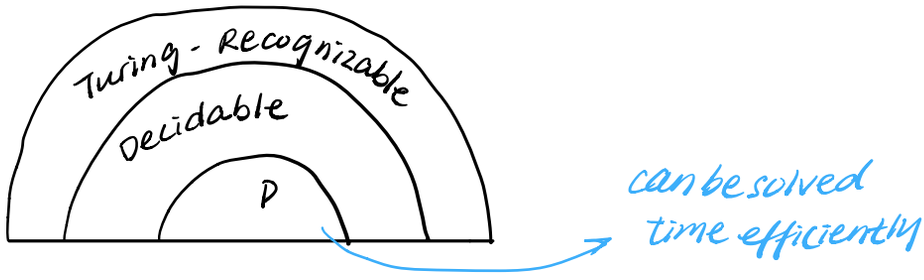
For example, $O(n^{10000})$ has polynomial growth whereas $O(2^n)$ or $O(n^{log_2 n})$ does not.

Good news: our conversions between different models of deterministic TM's can change the time complexity only polynomially.

In other words, if L can be decided in time $O(n^c)$ in one model, then it can be decided in time $O(n^{c'})$ in the other models, where c' is possibly a different constant.

Definition: P is the class of the language that can be decided in polynomial time by a deterministic TM.

Note: The running time is $O(n^c)$ for some fixed $c > 0$. It does not matter which type of the deterministic TM we use (e.g. single tape, multi-tape,..).



Remarks: in the area of algorithm design, P is considered to be the set of problems that can be solved efficiently.

Q: Let L be a regular language, does l necessarily belong to P?

A: Yes.

Given a word w we can simulate a DFA of L on w by just reading w letter by letter and following the transitions of the DFA accordingly. This takes only $O(n)$.

Examples:

- $\{< G, s, t > | \text{G is a directed graph that has a s-t path}\} \in P$
- $\{(m, n) | \text{m divides n}\} \in P$

(DFS has polynomial running time)

Ex. for P:

1° $\{x \mid x$ has the same number of 0's and 1's$\}$

2° $\{\langle G\rangle \mid G$ is a connected graph$\}$

3° $\{\langle A, r\rangle \mid A$ is a 0-1 matrix and $\text{rank}(A) \geq r\}$

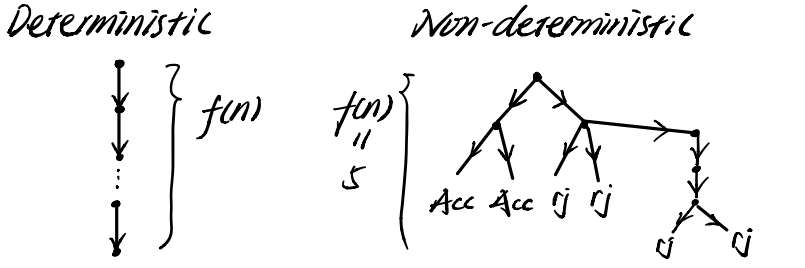
Remark P is considered to be the set of problems that can be solved time efficiently.

Non-determinism

Recall that a non-deterministic TM can choose from multiple choices during its computation

- It accepts w if there is a computation path to accept.
- It rejects w if all paths lead to reject.
- Otherwise loops

Running Time A non-det TM is a decider if all computation paths lead to acc/rej.



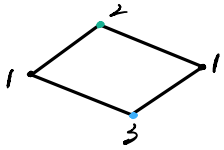
Map over all w's with $|w| = n$.

Longest computation path

Def Let N be a non-deterministic TM that is a decider. The running time of this TM is the function $f: \mathbb{N} \to \mathbb{N}$ where $f(n)$ is the maximum number of steps that the TM uses on any branch of its computation on any input of length n.

Ex. An undirected graph is 3-colorable if it is possible to assign colors 1, 2, 3 to the vertices such that no two adjacent vertices have the same color.

Such colorings are called proper.



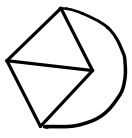
Let $3COL = \{\langle G \rangle \mid G \text{ is a 3-colourable graph}\}$

The following Non-det TM decides 3COL:

"On input G where G is a graph with m vertices:

$O(m)$ { For $i = 1, 2, 3, \ldots, m$
  color the ith vertex with one of 1, 2, 3 (Non-deterministically)
End For

$O(n) = O(m^2)$ { If the final colouring is proper, ACCEPT.
else REJECT.



Non-3COL Reject.

Q: What is the running time of this Non-det TM? $O(n)$ Linear

What is the size of input n in terms of m? $n = \Theta(m^2)$

because we have $\binom{m}{2} = \frac{m^2 - m}{2}$ potential edges

G can be represented by a $m \times m$ matrix with 1-0 entries

$$A_G(i,j) = \begin{cases} 1 & \text{if } ij \in E \\ 0 & \text{if } ij \notin E \end{cases}$$

Ex. A deterministic TM that decides 3COL.

"On input G on vertices,

Generate all the 3^m possible colourings one by one, (the 4's in the proof below)
and for each colouring, see if it is a proper colouring:
  if Yes, then ACCEPT
if none of the 3^m colourings is proper, REJECT"

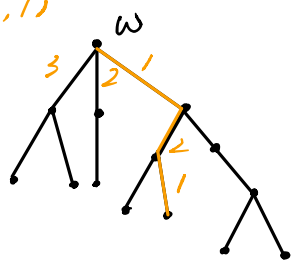
Q: Running Time $3^{O(n)}$, very slow, not a polynomial time alg.

Thm Every $t(n)$-time non-deterministic TM has an equivalent $2^{O(t(n))}$-time deterministic TM.

proof Let b be the maximum number of choices that our non-deterministic TM can face during any computation.

↳ This is $b = \max_{\substack{a \in \Gamma \\ q \in Q}} |\delta(q,a)|$

$y = (1,2,1)$



That is b is the max number of children that a node can have

Our deterministic solution works as follows:

On input w,

for $k = 1, 2, 3, \ldots$

for every $y = (y_1, y_2, \cdots, y_k) \in \{1, 2, \ldots, b\}^k$ (see ex above)

simulate N on w for k steps.
At every step i, make the choice according to y_i and if we end up at the ACCEPT state, then we ACCEPT and HALT.

(N: non-deterministic)

End for

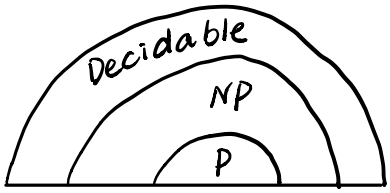
End for

At this point, REJECT.

$$\sum_{k=1}^{t(n)} b^k \leq b^{t(n)+1} = 2^{O(t(n))}$$

($b \geq 2$ to make the assumption)

Def (Non-deterministic Polynomial)
The set of all languages L s.t. there is a non-deterministic TM that decides L in polynomial time.



$3COL \in NP$
$\notin P$

Don't know how to prove $3COL \in P$.

EXPTIME : Deterministic and time $O(2^{n^c})$ for some $c>0$.

NEXPTIME: Non-deterministic and time $O(2^{n^c})$ for some $c>0$.

$t(n) = O(n^c)$ $\quad 2^{O(t(n))}$

Corollary $NP \subseteq EXPTIME$

We have $P \subseteq NP \subseteq EXPTIME \subseteq NEXPTIME$
(non-det includes det)

Q: $P \neq NP$?

We know $P \neq EXPTIME$, $NP \neq NEXPTIME$