NONDETERMINISTIC COMPUTATION

PEIFENG WANG*

Abstract. The execution of an algorithm on computing machines is studied. By Cantor's diagonal argument, we show that, the number of executions within a computation on a nondeterministic Turing machine is asymptotically greater than the number of steps within a computation on a deterministic Turing machine. Thus a nondeterministic machine can have more computing power than a deterministic machine, therefore suggesting $P \neq NP$.

Key words. Nondeterministic

AMS subject classifications.

1. Introduction. Turing machines take strings as input for its computation. The complexity of a computation is measured as a function of the size of the input string. In general, a machine has a finite nonempty set of input alphabet Σ. Σ^* is the set of finite string over Σ. For a string $\omega \in \Sigma^*$, $|\omega|$ is the length of string ω. For a set X, $|X|$ is its cardinality, a sequence over X is a sequence whose terms are elements of X. Let $\mathbb{N}$ be the set of natural numbers.

PROPOSITION 1.1. *The length of string ω is a natural number. i.e.* $|\omega| \in \mathbb{N}$

Every string $\omega \in \Sigma^*$ can be writen as a sequence of symbols $s_1...s_n$ where $\{s_i \in \Sigma | i \in \mathbb{N}\ and\ 1 \leq i \leq n = |\omega|\}$. Each of these symbols is an alphabet at a specific position within ω. The same alphabet at different positions represent distinct symbols. To encode the alphabet and position of a symbol of ω, a set $S(\omega) = \{s_i z^i | z \notin \Sigma \ \wedge\ i \in \mathbb{N}\ \wedge\ 1 \leq i \leq |\omega|\ \wedge\ s_i \in \Sigma\ is\ the\ ith\ symbol\ of\ \omega\}$ can be constructed.

PROPOSITION 1.2. *The cardinality of $S(\omega)$ equals to the length of string ω. i.e.* $|S(\omega)| = |\omega|$

Then $S(\omega)$ is a countable set.

For a machine M, a computation over string $\omega \in \Sigma^*$ is equivalent to a computation over the set $S(\omega)$. We study computations associated with sequences over $S(\omega)$.

2. Computation on a deterministic Turing machine (DTM). On a DTM, an algorithm defines an order in which elements of input set $S(\omega)$ are read and computed. In each computation, the algorithm reads the elements of input set $S(\omega)$ in a single sequence. This sequence of operation is called an execution. One computation on a DTM can have a single execution..

PROPOSITION 2.1. *Deterministic Turing machine can only compute over a countable input set.*

Proof. A deterministic Turing machine executes in a step-by-step manner, it can be shown that each of the steps is associated to a unique natural number.

A deterministic Turing machine is a tuple $(\Sigma, \Gamma, Q, \delta)$, where Q is nonempty finite set of states containing $q_0, q_{accept}, q_{reject}$, Γ is nonempty finite set of tape alphabet, δ is the transition function

$\delta : (Q - \{q_{accept}, q_{reject}\}) \times \Gamma - > Q \times \Gamma \times \{1, -1\}$

With the internal state $q \in Q$ and scanned symbol $s \in \Gamma$ as input, δ defines the next state and symbol to be scanned.

We assign natural numbers to the computation states:

1)The initial state q_0 is associated with natural number 0.

*Guanghua Road 1#, 34-1-3-5, Yanta District, Xi'an, Shaanxi, P. R. China 710075 (peifeng_w@yahoo.com)

2)if $q^- \neq q_{accept}$ and $q^- \neq q_{reject}$ is associated with natural number i, then q^+ of $(q^+, s', h) = \delta(q^-, s)$ is associated with number i+1.

Then with mathematical induction, the state of each step in an execution can be associated with a unique natural numbers. Thus an execution can have at most countable steps, and can compute only over countable input set.

□

Essentially, the deterministic transition function enables the application of mathemetical induction to prove theorem 2.1. Such property is absent on a nondeterministic machine.

3. Computation on a nondeterministic Turing machine (NDTM). On a nondeterministic Turing machine(NDTM) M_N, multiple transitions are allowed at any "moment", and multiple sequences of transitions are executed. Each of the sequences of transitions is called an execution. A computation can have multiple executions.

PROPOSITION 3.1. *In a computation on a NDTM, every sequence over the input set $S(\omega)$ has a distinct execution.*

Proof. We track all of the executions by recording the sequence of input symbols read by the NDTM.

For each execution e, there is an order in which the machine M_N reads the symbols of input set $S(\omega)$. Initially, execution e sets its record R(e) to an empty string, when execution e scans an input symbol $\alpha \in S(\omega)$, the symbol is appended to the existing R(e). Thus the records R of all executions are sequences over the input set $S(\omega)$, the BNF grammar of the recorded sequence is

$$R \quad ::= \quad R\alpha|\alpha$$
$$\alpha \quad ::= \quad elements\ of\ S(\omega)$$

Then R includes all strings whose alphabets are elements of $S(\omega)$. Every string $r \in R$ is a record of an execution e, then e is the corresponding execution of r.

For 2 exeuctions e_1 and e_2, if $e_1 = e_2$, then they have the same record string $R(e_1) = R(e_2)$. Equivalently, if $R(e_1) \neq R(e_2)$, $e_1 \neq e_2$.

□

A computation on a NDTM contains the set of all executions.

4. $P \neq NP$. To put it simply, the number of executions within a computation on a NDTM is asymptotically greater than the number of steps within a computation on a DTM.

In general, for fixed $k \in \mathbb{N}$, the kth Cartesian power of a set X is $X^k = \{(x_1, ..., x_k)|\ x_i \in X\ for\ all\ 1 \leq i \leq k\}$, if X is a finite set of cardinality $|X|$, the cardinality of its Cartesian power is $|X^k| = |X|^k$. If X is a countably infinite set and k is finite, its Cartesian power X^k is still a countable set.

On a computing machine with input ω, an execution within $|\omega|^k = |S(\omega)|^k$ steps can only compute the kth Cartesian power of the input set $S(\omega)$, i.e. $(S(\omega))^k$. Let P(n) be a polynomial of finite degree, there exists finite $k \in \mathbb{N}$ such that $|\omega|^k$ is asymptotically greater than $P(|\omega|)$

$$\lim_{|\omega| \to \infty} |\omega|^k > \lim_{|\omega| \to \infty} P(|\omega|)$$

Let $E(\omega)$ be the set of all executions of a computation on a NDTM with input ω. If the input string ω is infinitely long, we have 1) from proposition 1.2, $S(\omega)$ is countably infinite. 2) from Cantor's diagonal argument, the set of all sequences over $S(\omega)$ is uncountable. 3) from theorem 3.1, the size of $E(\omega)$ is greater than the size of the set of all sequences over $S(\omega)$, thus $E(\omega)$ is uncountable. 4) any finite Cartesian power of $S(\omega)$ is countable. Thus for any finite k,

$$\lim_{|\omega| \to \infty} |E(\omega)| > \lim_{|\omega| \to \infty} |S(\omega)|^k = \lim_{|\omega| \to \infty} |\omega|^k$$

then for any polynomial P(n) of finite degree
$\lim_{|\omega|\to\infty} |E(\omega)| > \lim_{|\omega|\to\infty} P(|\omega|)$

On the other hand, from theorem 2.1, a computation on a DTM can have only countable steps.

Thus in the case of infinitely long input string, the number of executions on a NDTM is strictly greater than the number of steps on a DTM, thus nondeterministic machine can have more computing power than a deterministic machine.

The above case of infinitely long input string also applies when the input set $S(\omega)$ is enough large. In specific, for any polynomial $P(n)$ of finite degree, there exists m, k, for all $|S(\omega)| > m$, 1) the number of steps that a DTM can compute within polynomial time $P(|\omega|)$ is less than $|\omega|^k$ for some k. 2) the size of kth Cartesian power of $S(\omega)$ is strictly less than the number of all executions of a computation on a NDTM. Thus $P < NP$

Acknowledgments.

REFERENCES

[1] Barker-Plummer, David, *Turing Machines*, The Stanford Encyclopedia of Philosophy (Summer 2013 Edition), Edward N. Zalta (ed.), URL = http://plato.stanford.edu/archives/sum2013/entries/turing-machine/