MEASURING COMPLEXITY BY USING REDUCTION TO SOLVE P VS NP AND NC & PH

KOBAYASHI KOJI

1. Abstract

This article describes about that NC and PH is proper (especially P is not NP) by using problem reduction. If L is not P, we can prove P is not NP by using difference between logarithm space reduction and polynomial time reduction. Like this, we can also prove that NC is proper by using difference between AL0 and NC1. This means L is not P. Therefore P is not NP. And we can also prove that PH is proper by using P is not NP.

2. P is not NP if L is not P

Definition 1. We will use the term "L", "P", "NP", "FL", "FP" as each complexity classes. These complexity classes also use Turing Machine (TM) set that compute target complexity classes problems. "$f \circ g$" as composite TM that accepting configurations of g are starting configurations of f. In this case, we also use complexity classes to show target TM. For example, $a \circ bb$ when a is TM and bb is complexity class mean that $a \circ b \mid b \in bb$.

Theorem 2. *$L \subsetneq P \rightarrow P \subsetneq NP$*

Proof. To prove it by using contraposition $P = NP \rightarrow L = P$. As we all know $NP \circ FP \in NP$. From assumption $P = NP$, all $NP \circ FP$ correspond to P. Therefore

$P = NP \rightarrow \forall C \in NP \forall D \in FP \exists E \in P \left(C \circ D = E\right)$

Mentioned [1] Theorem 10.43, all P are closed under logarithm space reduction FL. That is,

$\exists F \in P \forall H \in P \exists G \in FL \left(F \circ G = H\right)$

Therefore

$P = NP$

$\rightarrow \exists F \in P \forall C \in NP \forall D \in FP \exists G \in FL \left(C \circ D = F \circ G\right)$

$\rightarrow \forall D \in FP \exists G \in FL \left(D = G\right)$

This means $L = P$. Therefore, this theorem was shown. □

3. NC is proper

And we use circuit problem as follows;

Definition 3. We will use the term "AC^i", "NC^i" as each complexity classes. These complexity classes also use uniform circuits family set that compute target complexity classes problems. "$f \circ g$" as composite circuit that output of g are input of f. In this case, we also use complexity classes to show target circuit. For example, $A \circ BB$ when A is circuits family and BB is circuits family set mean that

$a \circ b \mid a \in A, b \in B \in BB$. Circuits family uniformity is that these circuits can compute AC^0.

Theorem 4. $NL \leq_{AC^0} NC^2$

Proof. Mentioned [1] Theorem 10.40, all NC^2 are closed by FL reduction. This reduction is validity of (c_1, c_2) transition function. Transition function change $O(1)$ memory and keep another memory. Therefore this validity can compute AC^0 and we can replace FL to AC^0. □

Theorem 5. *AC^i has Universal Circuits Family that can emulate all AC^i circuits family.*

Proof. To prove this theorem by making universal circuit family $A^i \in AC^i$ that emulate circuit family $\{C_j\} \in AC^i$ by using "depth circuit tableau". Universal circuit $U_j \in A^i$ have partial circuit $u_{k,d}$ that emulate all C_j gates $g_{k \in n}$ (include input value) and connected wires $w_{p,q}$ from g_p output to g_q input in every depth d. ($w_{p,p}$ always exist)

$u_{v \in n,d}$ have inputs from all $u_{u \in n,d-1}$ and g_u information that mean
a) validity of $u_{u,d-1}$
b) $u_{u,d-1}$ output (true if g_u output true)
c) existence of $w_{u,v}$ (true if $w_{u,v}$ is exists)
d) negation of $w_{u,v}$ (true if $w_{u,v}$ include not gate)
e) gate type of g_v (Or gate or And gate)
and outputs to $u_{w \in n,d+1}$ that mean
A) validity of $u_{v,d}$
B) $u_{v,d}$ output
These $u_{v,d}$ compute output like this;
If $u_{u,d-1}$ a) or c) input false then $u_{v,d}$ ignore $u_{u,d-1}$.
If $u_{u,d-1}$ a) and c) input true then $u_{v,d}$ A) output true and $u_{v,d}$ B) output g_k value that compute from e), b), d). b), d) include another $u_{w \in n,d-1}$ b), d).
If all a) input false then $u_{k,d}$ A) output false.
If all c) input false then $u_{k,d}$ A) output false.
And depth 0 circuit compute additional condition;
If $u_{k,0}$ is C_j input then $u_{k,0}$ A) output true and $u_{i,d}$ B) output C_j input value, else $u_{k,0}$ A) output false.
This U_j that consists of u emulate C_j. We can make every u in AC^0, so that A^i in AC^i.
Therefore, this theorem was shown. □

Definition 6. We will use the term "A^i" as universal circuits family that compute AC^i problem, "N^i" as universal circuits family that compute NC^i problem.

Theorem 7. *AC^0 can reduce all AC^i to A^i. That is, A^i is closed under AC^0 reduction.*

Proof. Mentioned above 35, we can make all AC^i by using AC^0 and we can connect these AC^i to A^i. That is, we can emulate all AC^i circuit by using $A^i \circ AC^0$. From the view of A^i, AC^0 is input reduction from AC^i to A^i. Therefore, this theorem was shown. □

Theorem 8. $NC^i \subsetneq NC^{i+1}$

Proof. We can prove this theorem like mentioned above 2.

To prove it using reduction to absurdity. We assume that $NC^i = AC^i = NC^{i+1}$. As we all know all NC^1 decision problems can embed NC^1 function problems. To simplify, all NC^1 circuit that output is embeded another NC^1 circuit output is same circuit. (Especially, all equivalence class include reversible circuit family.) From view of circuit structure, it is trivial that $NC^i \circ NC^1 \in NC^{i+1}$. From assumption $NC^i = AC^i = NC^{i+1}$, all $NC^i \circ NC^1$ correspond to NC^i. Therefore

$NC^i = AC^i = NC^{i+1} \rightarrow \forall C \in NC^i \forall D \in NC^1 \exists E \in NC^i \left(C \circ D = E\right)$

Mentioned above 7, all AC^i are closed by AC^0 reduction to universal circuit A^i. That is,

$\forall H \in AC^i \exists G \in AC^0 \left(A^i \circ G = H\right)$

Therefore

$NC^i = AC^i = NC^{i+1}$

$\rightarrow \forall C \in NC^i \forall D \in NC^1 \exists G \in AC^0 \left(C \circ D = A^i \circ G\right)$

$\rightarrow \forall D \in NC^1 \exists G \in AC^0 \left(D = G\right)$

But this means $AC^0 = NC^1$ and contradict $AC^0 \subsetneq NC^1$.

Therefore, this theorem was shown than reduction to absurdity. □

4. P is not NP

Theorem 9. $P \neq NP$

Proof. Mentioned above 2, $L \subsetneq P \rightarrow P \subsetneq NP$. And mentioned above 8, $L \subset NC^i \subsetneq NC^{i+1} \subset P$. Therefore $P \subsetneq NP$. □

5. PH is proper

Theorem 10. $\Pi_k \subsetneq \Sigma_{k+1}$, $\Sigma_k \subsetneq \Pi_{k+1}$

Proof. We can prove this theorem like mentioned above 8.

To prove it using reduction to absurdity. We assume that $\Pi_k = \Sigma_{k+1}$. As we all know $\Pi_k \circ \Sigma_1 \in \Sigma_{k+1}$. From assumption, all $\Pi_k \circ \Sigma_1$ correspond to Π_k. Therefore

$\Pi_k = \Sigma_{k+1} \rightarrow \forall C \in \Pi_k \forall D \in \Sigma_1 \exists E \in \Pi_k \left(C \circ D = E\right)$

Mentioned [2] Theorem 6.22, all Π_k are closed under polynomial time conjunctive reduction. We can emulate these reduction by using Π_1. That is,

$\exists F \in \Pi_k \forall H \in \Pi_k \exists G \in \Pi_1 \left(F \circ G = H\right)$

Therefore

$\Pi_k = \Sigma_{k+1}$

$\rightarrow \exists F \in \Pi_k \forall C \in \Pi_k \forall D \in \Sigma_1 \exists G \in \Pi_1 \left(C \circ D = F \circ G\right)$

$\rightarrow \forall D \in \Sigma_1 \exists G \in \Pi_1 \left(D = G\right)$

But this means $\Pi_1 = \Sigma_1$ and contradict $P \subsetneq NP \subsetneq coNP$. Therefore $\Pi_k \subsetneq \Sigma_{k+1}$.

We can prove $\Sigma_k \subsetneq \Pi_{k+1}$ like this.

Therefore, this theorem was shown than reduction to absurdity. □

Theorem 11. $\Delta_k \subsetneq \Sigma_k, \Sigma_k \neq \Pi_k$

Proof. Mentioned [2] Theorem 6.12,

$\Sigma_k = \Pi_k \rightarrow \Sigma_k = \Pi_k = PH$

$\Delta_k = \Sigma_k \rightarrow \Delta_k = \Sigma_k = \Pi_k = PH$

This contraposition is,

$\left(\Sigma_k \subsetneq PH\right) \vee \left(\Pi_k \subsetneq PH\right) \rightarrow \Sigma_k \neq \Pi_k$

$(\Delta_k \subsetneq PH) \vee (\Sigma_k \subsetneq PH) \vee (\Pi_k \subsetneq PH) \rightarrow \Delta_k \neq \Sigma_k$
From mentioned above 10,
$\Sigma_k \subsetneq \Pi_{k+1} \subset PH$
Therefore, $\Delta_k \neq \Sigma_k, \Sigma_k \neq \Pi_k$.
Mentioned [2] Theorem 6.10,
$\Sigma_k \subset \Sigma_{k+1}$, $\Pi_k \subset \Pi_{k+1}$,$\forall k \geq 1\,(\Delta_k \subset (\Sigma_k \cap \Pi_k) \subset (\Sigma_k \cup \Pi_k) \subset \Delta_{k+1})$
Therefore, $\Delta_k \subsetneq \Sigma_k, \Sigma_k \neq \Pi_k$. □

Theorem 12. $\Sigma_k \not\subset \Pi_k, \Pi_k \not\subset \Sigma_k$

Proof. To prove it using reduction to absurdity. We assume that $\Sigma_k \subset \Pi_k$.

This means that we can compute $\Pi_k \setminus \Sigma_k$ in $\Sigma_k \cup \Pi_k$ by using Σ_k result. Therefore $\Sigma_k = \Pi_k$ and contradict mentioned above 11 $\Sigma_k \neq \Pi_k$. Therefore $\Sigma_k \not\subset \Pi_k$.

We can prove $\Pi_k \not\subset \Sigma_k$ like this.

Therefore, this theorem was shown than reduction to absurdity. □

Theorem 13. $\Delta_k \subsetneq \Pi_k$

Proof. To prove it using reduction to absurdity. We assume that $\Delta_k = \Pi_k$.
Mentioned [2] Theorem 6.10,
$\Sigma_k \subset \Sigma_{k+1}$, $\Pi_k \subset \Pi_{k+1}$,$\forall k \geq 1\,(\Delta_k \subset (\Sigma_k \cap \Pi_k) \subset (\Sigma_k \cup \Pi_k) \subset \Delta_{k+1})$
Therefore
$\Delta_k = \Pi_k$
$\rightarrow \Delta_k = \Pi_k \subset (\Sigma_k \cap \Pi_k) \subset \Sigma_k \subset (\Sigma_k \cup \Pi_k) \subset \Delta_{k+1}$
$\rightarrow \Pi_k \subset \Sigma_k$
But this result contradict mentioned above 12.
Therefore, this theorem was shown than reduction to absurdity. □

Theorem 14. $\Sigma_k \subsetneq \Delta_{k+1}, \Pi_k \subsetneq \Delta_{k+1}$

Proof. To prove it using reduction to absurdity. We assume that $\Sigma_k = \Delta_{k+1}$.
Mentioned [2] Theorem 6.10,
$\forall k \geq 1\,(\Delta_k \subset (\Sigma_k \cap \Pi_k) \subset (\Sigma_k \cup \Pi_k) \subset \Delta_{k+1})$
Therefore
$\Sigma_k = \Delta_{k+1}$
$\rightarrow \Delta_k \subset (\Sigma_k \cap \Pi_k) \subset \Pi_k \subset (\Sigma_k \cup \Pi_k) \subset \Sigma_k = \Delta_{k+1}$
$\rightarrow \Pi_k \subset \Sigma_k$
But this result contradict mentioned above 12. Therefore $\Sigma_k \subsetneq \Delta_{k+1}$.
We can prove $\Pi_k \subsetneq \Delta_{k+1}$ like this.
Therefore, this theorem was shown than reduction to absurdity. □

References

[1] Michael Sipser, (translation) OHTA Kazuo, TANAKA Keisuke, ABE Masayuki, UEDA Hiroki, FUJIOKA Atsushi, WATANABE Osamu, Introduction to the Theory of COMPUTATION Second Edition, 2008
[2] OGIHARA Mitsunori, Hierarchies in Complexity Theory, 2006