TOPOLOGY OF P VS NP

KOJI KOBAYASHI

Abstract. This paper describes about P vs NP by using topological approach. We modify computation history as "Problem forest", and define special problem family "Wildcard problem" and "Maximal complement Wildcard problem" to simplify relations between every input.

"Problem forest" is directed graph with transition functions edges and computational configuration nodes with effective range of tape. Problem forest of DTM is two tree graph which root are accepting & rejecting configuration, which leaves are inputs, trunks are computational configuration with effective range of tape. This tree shows TM's interpretation of symmetry and asymmetry of each input. From the view of problem forest, some NTM inputs are marged partly, and all DTM inputs are separated totally. Therefore NTM can compute implicitly some type of partial (symmetry) overrap, and DTM have to compute explicitly.

"WILDCARD (Wildcard problem family)" and "MAXCARD (Maximal complement Wildcard problem family)" is special problem families that push NTM branches variations into inputs. If "CONCRETE (Concrete Problem)" that generate MAXCARD is in P-Complete, then MAXCARD is in PH, and these inputs have many overrap. DTM cannot compute these overrap conditions implicitly, and these conditions are necesarry to compute MAXCARD input, so DTM have to compute these conditions explicitly. These conditions are over polynomial size and DTM take over polynomial steps to compute these conditions explicitly. That is, PH is not P, and NP is not P.

1. Problem forest

First, we describe about "Problem forest".

Definition 1.1. In this paper, we limit problems as decision problems.
We use term as following;
TM : Turing Machine.
DTM : Deterministic Turing Machine.
NTM : Nondeterministic Turing Machine.
We use the term in book [Sisper] and [Ogihara] without notice.

"Problem forest" is special computation histories forest graph of whole inputs. By using this forest graph, we can treat NTM problems as metric space based DTM inputs.

Definition 1.2. We will use the term "Effective configuration" as configuration with effective range of tape that reach head after the configuration. "Problem forest" as the directed graph of TM computation. Problem forest edges are transition functions of TM, nodes are effective configuration, roots are accept or reject outputs, leaves are inputs. "Effective history" as computation history with effective configuration.

In this paper, we define NTM to simplify discussion;

NTM branch nondeterministicly to
$NTM \ni M = \bigcup_k m_{k \in R} \mid m \in DTM$
and each independent m_k compute input in parallel. k is base-n system which n is number of nondeterministic transition functions that branch from same configurations. Each symbols in k is choice of nondeterministic transition function.

From the view of Problem forest, DTM ability is symmetrization of asymmetry input. That is, This tree shows DTM's interpretation of symmetry and asymmetry of each input. DTM remove asymmetry of each inputs as process and merge to same (symmetry) configuration trunk, finally DTM compute all input characters as processis and symmetry as accept or reject.

Theorem 1.3. *DTM Problem forest is two tree graph that root is accepting or rejecting configuration.*

Proof. This is easily verified that every configurations yield only one configuration by using one (deterministic) transition function, and TM finally halt accepting or rejecting status. □

Theorem 1.4. *All DTM inputs are separated totally. DTM have to compute explicitly any type of partial (symmetry) overrap.*

Proof. It is trivial because every leaf and effective history different from another leaf and histories. □

From the view of Problem forest, difference between NTM and DTM is nondeterministic transition functions. These functions branch input to several configurations.

Theorem 1.5. *NTM Problem forest is DAG that root is accepting or rejecting configuration.*

Proof. It is trivial because NTM branch several effective configurations by using nondeterministic transition functions of m_k and finally halt accepting or rejecting status. □

Theorem 1.6. *Some NTM inputs are marged partly. NTM can compute implicitly some type of partial (symmetry) overrap.*

Proof. It is trivial because some branched effective histories merge other branched effective histories in branching m_k. □

2. Wildcard problem

To simplify NTM overrap, we define special problem family "Wildcard problem" which push NTM branches variations into inputs.

Definition 2.1. To define one problem as "Concrete Problem", and all symbols in concrete problem as "Concrete symbols", we will use the term "Wildcard problem" and as the special problem that have "Wildcard symbol" which mean one of concrete symbols. "CONCRETE" and "WILDCARD" as class of these problems. "Wildcard input" and "Concrete input" as input of WILDCARD and CONCRETE. Wildcard input is disjunction of all concrete inputs that overwrite all wildcard symbol to concrete symbol.

"Complement Wildcard problem" as the complement problem of Wildcard problem. "CoWILDCARD" as class of these problems. "CoWildcard input" as input of CoWILDCARD. CoWildcard input is conjunction of all negation of concrete inputs that overwrite all wildcard symbol to concrete symbol.

"Maximal Complement Wildcard Problems" as part of complement wildcard problem that value change false if any concrete symbol changed. "MAXCARD" as class of these problems.

"Complement Maximal Complement Wildcard Problems" as the complement problem of maximal complement wildcard problems. "CoMAXCARD" as class of these problems. "CoMAXCARD input" as input of CoMAXCARD.

In this paper, we limit CONCRETE is in P-Complete to simplify discussion.

Theorem 2.2. *WILDCARD and CoWILDCARD inputs make Hamming space based CONCRETE inputs. Each WILDCARD and CoWILDCARD inputs are subspace of this Hamming space, and some inputs have (symmetry) overrap subspace.*

Proof. It is trivial because CONCRETE inputs correspond to binary strings of Hamming space and WILDCARD and CoWILDCARD inputs are projective space that wildcard symbol as equivalence relation of concrete symbols. □

Theorem 2.3. $WILDCARD \subset NP$ *(based P complete CONCRETE).*

Proof. Some of NTM can compute all WILDCARD by using nondeterministic transition functions that decide every wildcard symbol on actual concrete symbol. This NTM can compute polynomial time because CONCRETE is in P. Therefore $WILDCARD \subset NP$. □

Corollary 2.4. $CoWILDCARD \subset coNP$ *(based P-Complete CONCRETE).*

Theorem 2.5. $MAXCARD \subset PH$ *(based P-Complete CONCRETE).*

Proof. We can compute MAXCARD following steps;

1) Compute input as CoWILDCARD.

2) Change one concrete symbols to wildcard symbols as universal, and compute the changed input as WILDCARD.

3) If 1) reject then reject, else if 1) accept and 2) reject then reject, else 1) and 2) accept then accept. Therefore $MAXCARD \subset PH$. □

Corollary 2.6. $CoMAXCARD \subset PH$ *(based P-Complete CONCRETE).*

Theorem 2.7. $\exists V \in MAXCARD \cup CoMAXCARD\,(V \notin P)$

Proof. (Proof by contradiction.) Assume to the contrary that

$\forall V \in MAXCARD \cup CoMAXCARD\,(V \in P)$

This imply

$\forall V \in MAXCARD\,(V \in P)$

This means that DTM can compute all V in polynomial steps. That is, DTM can merge every

$v \in V$

and remove

$u, w \notin V \mid u \subsetneq v \subsetneq w$

in polynomial steps.

Mentioned above 2.2, these inputs like u, v, w have implicitly overrap subspace each other which are based CONCRETE inputs.

$u \supset (u \cap v) \subset v$
$v \supset (v \cap w) \subset w$

Mentioned above 1.4, all DTM inputs are separated totally and DTM have to treat explicitly any type of partial overrap. The other hand, mentioned above 1.6, some NTM inputs are marged partly and NTM can treat implicitly some type of partial overrap. That is, DTM have to compute these overraped subspace u, v, w as point and treat overlap conditions explicitly.

Let think overrap granularity to estimate steps to compute overlap conditions explicitly. MAXCARD inner (CoMAXCARD) inputs separate every CONCRETE input.

$\exists v \in V \in MAXCARD\,(\forall p, q \in v\,(\exists u \subset v\,(p \in u, q \notin u)))$

If DTM do not compute every CONCRETE inputs $p, q \in v$ in computing MAXCARD input v or CoMAXCARD input u, DTM cannot compute overlap conditions of u, v explicitly and reject MAXCARD input v or accept CoMAXCAD input u by mistake. Therefore DTM have to compute these CONCRETE inputs including explicitly to compute MAXCARD.

However, some MAXCARD inputs have over polynomial CONCRETE inputs. Therefore DTM cannot compute based all CONCRETE inputs including in polynomial steps, and contradicting our assumption that $\forall V \in MAXCARD \cup CoMAXCARD\,(V \in P)$.

□

Theorem 2.8. $P \subsetneq PH$

Proof. Mentioned above 2.7,
$\exists V \in MAXCARD \cup CoMAXCARD\,(V \notin P)$
Mentioned above 2.5 and 2.6,
$MAXCARD \cup CoMAXCARD \subset PH$
Therefore $P \subsetneq PH$. □

Theorem 2.9. $P \subsetneq NP$

Proof. Mentioned above 2.8
$P \subsetneq PH$
and
$(P = NP) \Rightarrow (NP = coNP) \Rightarrow (NP = PH)$
therefore,
$(P \subsetneq PH) \wedge ((P = NP) \Rightarrow (NP = PH))$
$\equiv ((P \subsetneq NP) \vee (NP \subsetneq PH)) \wedge ((NP \subsetneq PH) \Rightarrow (P \subsetneq NP))$
$\Rightarrow (P \subsetneq NP)$ □

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