The P versus NP problem. Refutation.

Petr E. Pushkarev

February 24, 2016

Abstract

The article provides an response to problem, which is also called the Millennium problem — problem of equality of **P** and **NP** classes. As a result, given a complete refutation of equality. For proof of refutation was used kind of "reductio ad absurdum" method. We give that kind of assumptions, the possible existence of which seems difficult to determine. The concluding part use tensor analysis which is used to define objects, such considered relatively to the Turing machine computation. The goal was to give an answer to a problem, that has affected to degree of the proof calculation's details. The proof involves a look at the problem of equality as if from the side while maintaining attitude to the basic problem. The result as a whole can be obtained relative to the current problems of equality **P** and **NP** classes, but other than that give an opportunity to explore the computational process from a new perspective.

0 Introduction

The article presents a refutation of equality classes **P** and **NP**. Rather to say, article present a proof of refuting equality **P=NP**.

In general, the proof built by kind of "reductio ad absurdum" method. In some parts of proof we introduce objects, such semantic nature is difficult to imagine relatively to computation. However, this nature make our proof possible.

The first chapter "Definitions and Alphabet" provides in short way the key problem definition which where given by Stephen Cook in article "THE P VERSUS NP PROBLEM" [1]. Since the task was not to describe the problem again, mainly they look the same and may be found completely in article [1]. At the end of the first chapter we introduce some small lemma on which we partly build whole proof of refuting equality **P=NP**.

In the chapter 2 we present the general condition of equality and deduce attribute of equality **P** and **NP** classes.

The chapters 3 and 4 are key in proof. These chapters introduce conditions and methods for researchers equality.

The fifth chapter contains an axiom that is so in the whole proof context. Finally, in the end of the chapter 5 and the whole proof we made a conclusion, which refute equality **P=NP** in our proof.

In that way, chapters 1 to 5 form a context which by meaning equality **P=NP** as a given made it possible to research and to make a conclusion about equality relatively to the problem, such was given in article [1].

1 Definitions and Alphabet

Let repeat the main definitions from the article [1] in short.

Let Σ be a finite alphabet(that is, a finite nonempty set) with at least two elements, and let Σ^* be the set of finite strings over Σ. Then a language over Σ is a subset L of Σ^*. For each string ω in Σ^* there is a computation associated with M with input ω.

The language accepted by Turing machine M, denoted $L(M)$ by

$$L(M) = \{\omega \in \Sigma^* \mid M \text{ accepts } \omega\} \tag{1}$$

Denote by $t_m(\omega)$ the number of steps. For $n \in \mathbb{N}$ we denote by $T_M(n)$ the worst case run time of M; that is,

$$T_M(n) = max\{t_M(\omega) \mid \omega \in \Sigma^n\} \tag{2}$$

M runs in polynomial time if there exist k such that for all n, $T_M(n) \leq n^k + k$. Class languages **P** is

$$\mathbf{P} = \{L \mid L = L(M) \text{ for some Turing machine M that runs in polynomial time}\} \tag{3}$$

Checking relation $R \subseteq \Sigma^* \times \Sigma_1^*$, with witch associate a language L_R over $\Sigma^* \cup \Sigma_1^* \cup \{\#\}$ defined by

$$L_R = \{\omega \# y \mid R(\omega, y)\} \tag{4}$$

R is polynomial-time iff $L_R \in \mathbf{P}$. **NP** class of languages defined by condition that language L over Σ is in **NP** iff there is $k \in \mathbb{N}$ and a polynomial-time checking relation R such for all ω,

$$\omega \in L \Leftrightarrow \exists y(|y| \leq |\omega|^k \text{ and } R(\omega, y)) \tag{5}$$

where $|\omega|$ and $|y|$ denote the lengths of ω and y, respectively.

After repeat main definition introduce a little lemma:

Lemma 1.1. $\Sigma^* \neq \Sigma$.

Proof. Let $\Sigma^* = \Sigma$, then language L from Σ^* will be equal language L from Σ. That contradict to the definition of L. □

2 Classes equality

For equality of classes **P** and **NP** must be fulfilled condition of equality,

$$A = B \Leftrightarrow \forall x : (x \in A) \Leftrightarrow (x \in B) \tag{6}$$

which implies that is

$$\mathbf{P} = \mathbf{NP} \Leftrightarrow \forall L(M) : (L(M) \in \mathbf{P}) \Leftrightarrow (L(M) \in \mathbf{NP}) \tag{7}$$

That mean, that should be that kind of language L_{civ} which can be defined as a strong **NP** and **P** at the same time,

$$\begin{gathered} L_{civ} = L(M) \text{ for some Turing machine M that runs} \\ \text{in polynomial time} \wedge \omega \in L_{civ} \Leftrightarrow \exists y(|y| \leq |\omega|^k \text{ and } R(\omega, y)) \end{gathered} \tag{8}$$

where $|\omega|$ and $|y|$ denote the lengths of ω and y, respectively.

3 Determination of computation

To discover equality **P=NP**, consider the process of Turing machine working as a process with an already predetermined outcome.

In other words, present the process of calculation in an environment where all possible languages previously was computed and all true results already predicted. That can't contradict with definition of Turing machine or with any definitions from chapter 1, because working process and machine structure stay the same. Changed only surroundings of machine for where possible fully predetermined computation outcome before any computation step, which made possible for us that considering.

By true result we mean an accepted result of computing ω in the best case run time for M.

In that case, we can denote a three-dimensional computing coordinate system and define the language L as a matrix in computing coordinates.

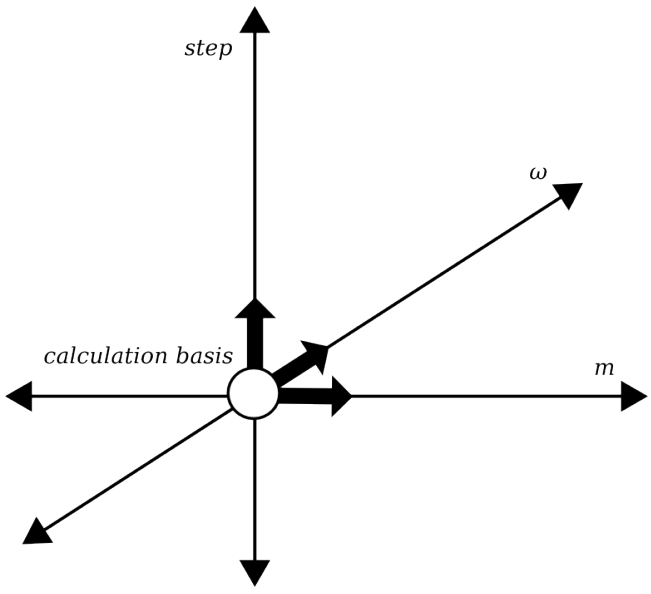

Figure 1: computing coordinate system with calculation basis

Remark. It will be true to note, that for each machine must exist its own coordinate system and considering the computation language without machine relativity unacceptable. As noted earlier, for the proof, we consider such conditions of machine environment that allow us to make known the correct for best case run presets and considered computation regardless of machine particularity.

The dimension of this system defined by number machine calculation step — *step*, point in time — m and the value from the alphabet — ω.

So, now we can denote the language L as a matrix,

$$L = \begin{pmatrix} step_1 & step_2 & \cdots & step_n \\ m_1 & m_2 & \cdots & m_n \\ \omega_1 & \omega_2 & \cdots & \omega_n \end{pmatrix} \tag{9}$$

When we determine the language matrix, we can denote the computation of an language L as a valence tensor (m, ω),

$$Computation_{j_1 \cdots j_m}^{i_1 \cdots i_\omega} = \sum_{\substack{k_1, \cdots, k_m \\ h_1, \cdots, h_\omega}}^{3} \cdots \sum^{3} S_{h_1}^{i_1} \cdots S_{h_\omega}^{i_\omega} T_{j_1}^{k_1} \cdots T_{j_m}^{k_m} \widetilde{Computation}_{k_1 \cdots k_m}^{h_1 \cdots h_\omega} \tag{10}$$

Computation of L_{civ} is equal irrespectively to 7,

$$(L_{civ} \in \mathbf{P}) \Leftrightarrow (L_{civ} \in \mathbf{NP}) \to \mathbf{P}_{mciv}^{\omega} = \mathbf{NP}_{mciv}^{\omega} \tag{11}$$

4 Computation equality

From equality condition of tensors it follows, that there should be a basis, relative to which all components of the tensor are equal. That basis for **P=NP** equality we named CIV.

It is important to say, that the basis CIV is temporary object which exist as exist equality of **P=NP**. CIV's analytical definition allows us to use it without predetermination basis space which can or can't exist. So, that possible to suppose that computation space where **P=NP** is a space which specify by the basis CIV.

Defining a basis CIV, we can express the calculation of any language L in space where **P=NP** by replacing the main calculation basis for CIV,

$$L = \begin{pmatrix} step_1^{civ} & step_2^{civ} & \cdots & step_n^{civ} \\ m_1^{civ} & m_2^{civ} & \cdots & m_n^{civ} \\ \omega_1^{civ} & \omega_2^{civ} & \cdots & \omega_n^{civ} \end{pmatrix} \begin{pmatrix} c_{1,1} & c_{2,1} & \cdots & c_{n,1} \\ c_{1,2} & c_{2,2} & \cdots & c_{n,2} \\ c_{1,3} & c_{2,3} & \cdots & c_{n,3} \end{pmatrix} \tag{12}$$

Replacing matrix of $c_{n,3}$ have the same temporary nature as basis CIV. That allow us express the calculation without definition anything about replacing matrix. Also, we don't definition of calculation basis for prof **P=NP** equality.

5 Refutes of equality

Axiom 1. *Since the existence of the language L consisting from any ω is possible, it is possible to chose that language, for such expression ω to space specified by CIV is*

$$\omega^{civ} : (\omega^{civ} \in \Sigma^*) \Leftrightarrow (\omega^{civ} \in \Sigma) \tag{13}$$

As mentioned before, we don't strictly define the space where **P=NP**, but we strictly define ω in chapter 1. That make following outcome possible for us.

Axiom 1 as came from 12 contradict to Lemma 1.1 and fully refutes equality **P=NP** Q.E.D.

References

[1] Stephen Cook. The p versus np problem. *The millennium prize problems*, pages 87–106, 2006.