A 3 RD SHORTER SOLUTION OF THE CLAY MILLENNIUM PROBLEM ABOUT P ≠ NP =EXPTIME

Constantine Konstantinos Kyritsis

THE SIMPLEST POSSIBLE FULLY CORRECT SOLUTION OF THE CLAY MILLENNIUM PROBLEM ABOUT P vs NP. A SIMPLE PROOF THAT NP=EXPTIME.

Konstantinos E. Kyritsis
Dept. Accounting-Finance University of Ioannina, Greece, ckiritsi@uoi.gr

ABSTRACT

In this paper I present probably the simplest possible abstract formal proof that P ≠ NP, and NP=EXPTIME, in the context of the Zermelo-Frankel set theory and deterministic Turing machines. My previous publications about the solution of the P vs NP with the same result NP=EXPTIME, to be fully correct and understandable need the Lemma 4.1 and its proof of the current paper. The arguments of the current paper in order to prove NP=EXPTME are even simpler that in my previous publications. The strategy to solve the P vs NP problem in the current paper (and in my previous publications) is by starting with an EXPTIME-complete language (problem) and proving that it has a re-formulation as an NP-class language, thus NP=EXPTIME. The main reason that the scientific community has missed so far such a simple proof, is because of two factors a) It has been tried extensively but in vain to simplify the solutions of NP-complete problems from exponential time algorithms to polynomial time algorithms (which would be a good strategy only if P=NP) b) It is believed that the complexity class NP is strictly a subclass to the complexity class EXPTIME (in spite the fact that any known solution to any of the NP-complete problems is not less than exponential). The simplicity of the current solution would have been missed if b) was to be believed because the current solution is based on the strategy of STARTING with an EXPTIME-complete language (problem) and proving that it can be re-formulated as an NP-class language, thus NP=EXPTIME. The present results definitely solve the 3rd Clay Millennium Problem about P versus NP in a simple and transparent away that the general scientific community, but also the experts of the area, can follow, understand and therefore become able to accept.

Key words*: *3rd Clay Millennium problem, EXPTIME-complete problems, NP-complexity, P-complexity
Mathematical Subject Classification: 68Q15

1 INTRODUCTION

Two solution of the famous P versus NP problem has been published in [8],[9], [21], [22], [23] Kyritsis K. and in this paper we present a very shorter simplification of the solution. Nevertheless, the above previous publications about the solution of the P vs NP with the same result NP=EXPTIME, **to be fully correct and understandable need the Lemma 4.1 and its short proof of the current paper.**

The strategy to solve the P vs NP problem in the current paper (and in my previous publications) is by starting with an EXPTIME-complete language (problem) and proving that it has a re-formulation as an NP-class language with verifier relation and certificate, thus NP=EXPTIME

The main reason that the scientific community has missed so far such a simple proof, is because of two factors

a) It has been tried extensively but in vain to simplify the solutions of NP-complete problems from exponential time algorithms to polynomial time algorithms (which would be a good strategy only if P=NP)

b) It is believed that the complexity class NP is strictly a subclass to the complexity class EXPTIME (in spite the fact that any known solution to any of the NP-complete problems is not less than exponential).

The simplicity of the current solution would have been missed if b) was to be believed because the current solution is based on the strategy of STARTING with an EXPTIME-complete language (problem) and proving that it can be re-formulated as an NP-class language, thus NP=EXPTIME. **The present results definitely solve the 3rd Clay Millennium Problem about P versus NP in a simple and transparent away that the general scientific community, but also the experts of the area, can follow, understand and therefore become able to accept.**

In the history of mathematics, it is known that difficult problems that have troubled a lot the mathematicians, turned out to have different proofs one simple and one very complex. Such an example is if the general 5th order polynomial equation can be solved with addition, subtraction, multiplication, division and extraction of radicals starting from the coefficients. The famous mathematician Niels Henrik Abel who gave a very simple proof. On the other hand, the proof of the same, by the E. Galois theory, is a whole book of dozens of pages!

And a famous mathematician once said that *"Once a proof is known to a mathematical problem, then immediately after it becomes trivial!"*

It is important to mention, a statement, that is usually attributed to the famous mathematician Yuri Manin, that "A correct proof in mathematics is considered a proof only if it has passed the social barrier of being accepted and understood by the scientific community and published in accepted Journals"

Passing the obstruction of the social barrier, sometimes is more difficult than solving the mathematical problem itself!

It is similar with the solution of the P versus NP problem in this paper. We will utilize in our proofs, the ***key abstraction*** of the existence of an EXPTIME complete language, (it is known that it exists) without specifying which one, which will simplify much the arguments. Then we prove that there is a reformulation of it that fits the definition of a language being an NP-class language.

We must notice here that the P versus NP problem, is in fact a set of different problems within different axiomatic systems. In the context of what axiomatic system is the Complexity Theory of Turing machines? Since the complexity theory of Turing machines requires entities like infinite sets of words then it is in the context of some axiomatic set theory, together with the axiom of infinite. So we notice that the next are different problems:

1) The P versus NP problem in the Zermelo-Frankel axiomatic system of sets without the axiom of choice and this axiomatic system formulated in the 2rd order formal languages.
2) The P versus NP problem in the Zermelo-Frankel axiomatic system of sets with the axiom of choice and this axiomatic system formulated in the 2rd order formal languages.
3) Etc

The proof of the P versus NP problem in the direction $P \neq NP$, is supposed also to mean that the standard practice of encryption in the internet, is safe.

We notice also that the P versus NP:

1) It is a difficult problem, that has troubled the scientific community for some decades
2) It may have simple proofs of a few paragraphs, hopefully not longer than the proof of the Time Hierarchy theorem, which seems to be a deeper result.
3) But it can also have very lengthily and complex proofs, that may take dozens of pages.
4) There are many researchers (more than 5 of them and some say more than 60) that have claimed to have solved it, either as P=NP, or as $P \neq NP$, and even as suggestion that neither are provable, but only a handful of them seem to have been able to pass the preliminary social barrier and publish their solution in conferences or Journals with referees. The rest of them have published online only preprints (see e.g. the [17] P versus NP page). It seems to me though that it is not probable that all of them have correct solutions. Especially in the direction P=NP, there is a common confusion and mistake, that has been pointed out by Yannakakis M. [18]. Furthermore, this confusing situation has contributed so that although there are publications in respectable Journals, the experts and the scientific community does not seem of being able to decide if the P versus NP problem has been solved or not. This is reasonable, as there are proofs of close to 100 pages, and no average reader would feel comfortable to go through them, and decide for himself if there a flaw or error somewhere. Still it is better to have published results than non-published, and then let the large number of readers to try to find errors or flaws in the solutions if there are any.

   So here comes the need of a more challenging problem: Not only to solve the P versus NP problem, but also solve it in such a simple, elegant and short way, so that the researchers will know a decisive proof that they can understand and control that $P \neq NP$ or not, so short that anyone familiar with the area, would discover any flaw or error if it existed.

   This is I believe the value of the present paper that provides such a proof in the context of the Zermelo-Frankel set theory (we do not use the axiom of choice), formulated e.g. within 2^{nd} order formal languages.

   What this proof is or is not:

1) It does not introduce new theoretical concepts in computational complexity theory so as to solve the P versus NP.
2) It does not use relativization and oracles
3) It does not use diagonalization arguments, although the main proof, utilizes results from the time hierarchy theorem
4) It is not based on improvements of previous bounds of complexity on circuits
5) It is proved with the method of counter-example. Thus it is transparent short and "simple". It takes any Exptime-complete DTM decision problem, and from it, it derives in the context of deterministic Turing machines a decision problem language which it is apparent that it belongs in the NP class decision problems while it does not belong the class P of decision problems.
6) It seems a "simple" proof because it chooses the right context to make the arguments and constructions and the key-abstraction mentioned above. So it helps the scientific community to accept that this 3rd Clay Millennium problem has already been solved.

In the paragraph 4, we give an advanced, full proof that P ≠ NP=EXPTIME, in the standard context of deterministic Turing machines, solving thus the 3rd Clay Millennium problem.

2. PRELIMINARY CONCEPTS, AND THE FORMULATION OF THE 3RD CLAY MILLENNIUM PROBEM, P VERSUS NP.

In this paragraph, for the sake of the reader, we will just mention the basics to understand the formulation of the 3rd Clay Millennium problem. The official formulation is found in [3] (Cook, Stephen *(April 2000), The P versus NP Problem (PDF), Clay Mathematics Institute site)*. Together with an appendix where there is concise definition of what are the Deterministic Turing machines, that is considered that they formulate, in Computational Complexity theory, the notion and ontology of the software computer programs.
In the same paper are also defined the computational complexity ***classes P, NP***.
The elements of the classes P, NP etc strictly speaking are not only sets of words denoted by L, that is not only languages, but also for each such set of words or language L at least one DTM , M that decides it, in the specified complexity so they are pairs (L,M). Two such pairs (L_1, M_1) (L_2, M_2) are called ***equidecidable*** if $L_1 = L_2$ although it may happen that $M_1 \neq M_2$. E.g. if the complexity of M_1 is polynomial-time while that of M_2 exponential-time choosing the first pair instead of the second means that we have turned a high complexity problem to a feasible low complexity problem.
The definition of other computational complexity classes like **EXPTIME** etc. can be found in standard books like [6],[11],[12]. In the official formulation [3] there is

also the definition of the concept of ***a decision problem language in polynomial time reducible to another decision problem language***.
Based on this definition it is defined that an EXPTIME-complete decision language of EXPTIME is EXPTIME-complete, when all other decision problems languages of EXPTIME have a polynomial time reduction to it. Here is the exact definition
Definition 2.1 *Suppose that Li is a language over all words* Σ_i *, i = 1, 2. Then* $L_1 \leq_p L_2$ *or* $L_1 \leq_{poly} L_2$ *(*L_1 *is polynomially p-reducible to* L_2*) iff there is a polynomial-time computable function-map (or total function)* $f : \Sigma^*_1 \rightarrow \Sigma^*_2$ *(in other words (x, f(x))* ϵ *f is in polynomial time decidable) such that* $x \epsilon L_1$ *if and only if* $f(x) \epsilon L_2$*, for all* $x \epsilon \Sigma^*_1$.
Lemma 2.1 If $L_1 \leq_{poly} L_2$ *by the polynomial time decidable function f, then* $|f(x)| <= p(|x|)$ *for some polynomial p.*
In the same books [3], [6],[10],[11] can be found the concepts and definitions of ***NP-complete and EXPTIME-compete decision problems***. See also [7], [12] where its proved that specific decision problems are EXPTIME-complete. E. g. in [3] in the Definition 4 , it is defined that a language of the complexity class NP is NP-complete if and only if any other language of the class NP has a polynomial reduction to it.
In particular it holds that
Lemma 2.2. *If the language* L_c *of the complexity class C (=P, NP, EXPTIME etc) is a C-complete language, (C =P, NP, EXPTIME etc) than any other language L of C has a polynomial reduction on to the language* L_c .
For simplicity we will consider here only binary alphabets {0,1} and sets of binary words Σ.
Since we are obliged to take strictly the official formulation of the problem, rather than text books about it, we make the next clarifications.
We will use the next conditions for a Language to be in the class NP, as stated in the official formulation of the P versus NP problem (see **[3] Cook, Stephen *(April 2000)***, *The P versus NP Problem (PDF), Clay Mathematics Institute.).*
We denote by Σ^* *all the words of an alphabet* Σ.

Definition 2.2 *A language L of binary words is in the class NP if and only if the next conditions hold*

1) *There is a deterministic Turing machine M that decides L. In other words for any word x in L, when x is given as input to M, then M accepts it and if x does not belong to L then M rejects it.*
   In symbols: $\exists$ *a deterministic Turing machine M, such that* $\forall x \epsilon \Sigma^*$*, x is either accepted or rejected by M and if M accepts* $x \rightarrow x \epsilon L$*, and if M reject* $x \rightarrow x$ *does not belong to L*

2) *There is a polynomial-time checkable relation R(x,y) which as set of pairs of words can also be considered a polynomial decidable language R, and a natural number k of N, so that for every word x , x belongs to L if and only if*

there is a word y , with $|y|<=|x|^k$, and R(x,y) holds or equivalently (x,y) belongs to R.
In symbols: ∃ relation (language) R which is polynomial-time checkable (polynomial complexity decidable language R) ,and ∃ k∈N, such that ∀x∈Σ, x∈L↔ (∃ y ∈ Σ*, $|y|<=|x|^k$ and (x,y) ∈R). Or equivalently*
*L={x/x ∈ Σ*and ∃ y ∈ Σ*, $|y|<=|x|^k$ and (x,y) ∈R}.*

3) *The complexity class NP is contained in the complexity class EXPTIME.*

Remark 2.1. In the official statement of the P versus NP problem (see **[3] Cook, Stephen** ***(April 2000)****, The P versus NP Problem (PDF), Clay Mathematics Institute)* the condition 1) is not mentioned. But anyone that has studied complexity theory, knows that it holds. The languages of NP cannot be semidecidable (or undecidable). The NP class is also defined as NP $=\cup_{k\in N}$ NTIME(n^k), but this definition is also in the context of non-deterministic Turing Machines.
Remark 2.2. Notice that in the condition 2) the k depends on the relation R and is not changing as the certificate y changes. In other words k does not depend on y and we ***did not*** state the next:
There is a polynomial-time checkable relation R(x,y), so that for every word x , x belongs to L if and only if there is a word y , and k in N ,with $|y|<=|x|^k$, and R(x,y) holds. In symbols: ∃ relation R which is polynomial-time checkable , such that ∀x∈Σ, x∈L↔ (∃y∈Σ* and ∃k∈N such that $|y|<=|x|^k$ and R(x,y) holds).*
In the official statement of the P versus NP problem (see [3] Cook, Stephen *(April 2000), The P versus NP Problem (PDF), Clay Mathematics Institute)* this is not made clear, in the natural language that the definition is stated. But that k does not depend on the certificate, but on the polynomial checkable relation becomes clear, when we look at the proof in any good textbook about complexity theory, of how a non-deterministic Turing machine which runs in polynomial time, can define a deterministic Turing machine with a polynomial time checkable relation, which is considered that replaces it.
More generally modern formulations of the definition 2.2 instead of integer k such that $|y|<=|x|^k$ is require a polynomial p(|x|) so that $|y|<=p(|x|)$
Remark 2.3. Usually the condition 3) in the definition 2.2 is not stated. But if for each word x of L , we take all possible worlds y of Σ*, of length $<=|x|^k$ and check in polynomial time complexity if R(x,y) holds or not, we result in the worst case scenario to an exponential time complexity algorithm that decides the language L.

3. WELL KNOWN RESULTS THAT WILL BE USED.

We will not use too many results from the computational complexity theory for our proof that P ≠ NP.
A very deep theorem in the Computational Complexity is the ***Time Hierarchy Theorem*** (see e.g. [6],[11],[12],[10],[14]. This theorem gives the existence of decision problems that cannot be decided by any other deterministic Turing machine in less complexity than a specified.
Based on this theorem, it is proved that:

Proposition 3.1 *There is at least one EXPTIME-complete decision language (problem), that cannot be decided in polynomial time, thus P ≠EXPTIME.*

The next two propositions indicate what is necessary to prove in order to give the solution of the P versus NP problem.
Proposition 3.2 *If the class NP contains a language L which cannot be decided with a polynomial time algorithm, then P ≠ NP.*
Proposition 3.3 *If the class NP contains a language L which is EXPTIME complete, then NP=EXPTIME.*

4. THE SOLUTION: P ≠ NP=EXPTIME IN THE CONTEXT OF DETERMINSITIC TURING MACHINES

We will prove in this paragraph that P ≠ NP in the context of second order formal language of the Zermelo-Frankel set theory.
The strategy to solve the P vs NP problem in the current paper (and in my previous publications) is by starting with an EXPTIME-complete language (problem), that the proposition 3.1 guarantees that it exists, and proving that it has a re-formulation as an NP-class language with verifier relation and certificate, thus NP=EXPTIME. Of course in order to proceed like this we must not share the belief that the complexity class NP is a strict subclass of the complexity class EXPTIME otherwise we would not think to start like this.
There is a hidden similarity or affiliation between the Definition 2.2 of the NP complexity class with polynomial complexity verifier relations and the definition of polynomial complexity reduction of a language to another language Definition 2.1. Part of this similarity or affiliation is revealed in the next lemma.

Lemma 4.1 ***Polynomial reduction of languages and the complexity class NP.***
Let a Turing-machine decidable Language L over the alphabet Σ. and in the complexity class EXPTIME.
If there is polynomial reduction of the full language Σ^* on L , $\Sigma^* \leq_{poly} L$ then
The language L is in the complexity class NP.
Proof: From the definition 2.1 since $\Sigma^* \leq_{poly} L$ *there is a polynomial time computable function-map R : Σ^*->Σ^*, such that y belongs to* Σ^*if and only if R(y) belongs to L. Furthermore from the Lemma 2.1 *|R(x)|<=p(|x|) for some polynomial p* . Or to rephrase it, (y,R(y)) ϵR *is polynomial time decidable |y|<=p(|x|) and y belongs to* Σ^*if and only if *and x=R(y)* belongs to L. Reversing the logically equivalence relation:, *x=R(y)* belongs to L if and only if y belongs to Σ^* (that is it is a word over the alphabet Σ) . Again rephrasing it: (y,x) ϵR is polynomial time decidable , |y|<=p(|x|) for some polynomial p (depending or R and not on x), and x belongs to L iff there is word y of Σ^* such that (y,x) ϵR . Or in symbols , ∃ relation (language) R which is polynomial complexity decidable language ,and ∃ polynomial p such that *„$x \epsilon L \leftrightarrow (\exists\, y \,\epsilon\, \Sigma^*$, with $|y|<=p(|x|)$ and $(x,y)\, \epsilon R$).* But this is just the condition 2) of the definition 2.2 of the complexity class NP, where the relation, $|y|<=|x|^k$ has been substituted by the $|y|<=p(|x|)$ as in the end of Remark 2.2. The language L is furthermore decidable by a Turing machine and at most of EXPTIME complexity, thus the conditions 1) and 3) of the Definition 2.2 are also satisfied, consequently the language L is of the type of NP-complexity class.

QED.

We could explore further the similarity and affiliation of the concepts of polynomial complexity verifier relation with certificates of a language L in the NP-class, by proving that if the verifier relation R is also a function-map from the language of certificates C(L) to L, then there is also polynomial complexity reduction of the language of certificates C(L) to the original language L: C(L)<=poly L. But we do not need to do so for the purpose of solving the P vs NP problem.
The Lemma 4.1 may seem simple to many and certainly for me it was obvious, that is why I did not included in my previous publications of the solution of the P vs NP problem [8],[9], [21], [22], [23] Kyritsis K. **But strictly speaking with it, the previous published solutions of the P vs NP are not fully and in very detailed way understandable. Lemma 4.1 requires a lucky abstract thinking and it's a key abstraction for the solution of the P vs NP problem.**

We proceed to solve the P vs NP problem not only with the claim $P \neq NP$, but with the much stronger and surprising for many researchers claim NP=EXTIME. Actually this stronger and "unexpected" claim makes the solution dramatically simple (but not easy to think of it as it involves substantial abstract thinking inside the Lemma 4.1!)

Proposition 4.1 (Simplest possible fully correct solution of the 3rd Clay Millennium problem P vs NP) *There is at least one decision problem language of the class NP which is not also in the class P. Therefore, $P \neq NP$. It holds furthermore that NP=EXTIME.*

Proof: As we stated already from the beginning, our strategy to solve the P vs NP problem is to START with a EXPTIME-complete language L! Thus we know in advance that this language cannot be decided in polynomial time complexity.

Then we proceed to realize that the language L can be reformulated in such a way, that is definable with polynomial complexity verifier relation and certificates, thus it is a language of the NP-class. Proving this for the language L it automatically concludes that NP=EXPTIME, because we know that NP<=EXPTIME, and that L is an EXPTIME-complete language.

We do know from the Proposition 3.1 that there exists such a EXPTIME-complete language L. We do not need to specify which one, as the arguments are simple as long as they are abstract. But since L is EXPTIME-complete from the Lemma 2.2 any other language of the EXPTIME-complexity class has a polynomial reduction on it! And certainly the complexity class EXPTIME contains the polynomial complexity class P<=EXPTIME. Therefore the language Σ^* of all words over the alphabet Σ, which is polynomial time complexity decidable also belongs to EXPTIME $\Sigma^* \epsilon$ EXPTIME. Thus $\Sigma^* \leq_{poly} L$. We do not even need to know which polynomial complexity function R makes this polynomial complexity reduction. But then from the previous Lemma 4.1, we conclude that

L ϵ NP-class! Therefore NP=EXPTIME. Which from the hierarchy theorem concludes that $P \neq NP$.

QED.

5. CONCLUSIONS

The literature of the complexity theory has and abundance of problems or languages that are proven to be NP-complete, but a scarcity of languages or problems that are proven to be EXPTIME-complete.

The fact that none of the languages or problems that are known to be NP-complete have so far known algorithms to solve-decide, them that in the general case are never less than exponential time, should ring a bell to the researchers!

Could it hold that all these NP-complete problems are also EXPTIME-complete problems? The current solution of the P vs NP problem proves exactly this!

Proposition 5.1 *All the NP-complete languages are also EXPTIME-complete languages.*

Proof: Direct from the equality NP=EXPTIME in the proposition 4.1 QED.

Sometimes great problems have relatively short and elegant solutions provided we find the **key-abstractions** and convenient context, symbols and semantics to solve them. It requires also a certain power of thinking rather than complexity of thinking, in areas where traditionally and collectively it may not exist before. Here the key-abstraction was to start from the class EXPTIME and an EXPTIME-complete language of it, without specifying which one instead starting from the class NP. Then prove that it can be reformulated as an NP-class language. Since in my opinion the Hierarchy Theorem is a deeper result than the P versus NP problem, in principle there should exist a not much more complicated proof of the P versus NP problem, compared to the proof of the Hierarchy Theorem. Intuitively since ***the Non-deterministic Turing machines are like user-interactive algorithms*** which involve the ***free human will***, it is expected that the Non-deterministic polynomial time such algorithms cannot be computed with less than exponential time complexity. This includes the encryption and the password setting problems.

The proof of the P versus NP problem in the direction P ≠ NP, also to means that the standard practice of password setting in the internet, is safe when the encryptions is not corrupted and the publicly available hardware computational power is the same for all .

There are many students who are surprised with the ”difficulty” P vs NP problem and ask why the P vs NP problems was not solved long ago, by proving that the **password-breaking** cannot be done in polynomial time but only in exponential time. Actually this is exactly what we proved! Only that…..the theoretical formulation of encryption of passwords is nothing more that the presentation of a language in abstract way without specifying it except of general requirements as we did in the arguments of the current paper.

REFERENCES

[1] Conway J.H. *On numbers and games*, Academic press 1976

[2] Cook, Stephen A. (1972**)**. *"A hierarchy for nondeterministic time complexity"*. Proceedings of the fourth annual ACM symposium on Theory of computing. STOC '72. Denver, Colorado, United States: ACM. pp. 187–192

[3] Cook, Stephen *(April 2000)*, *The P versus NP Problem (PDF)*, *Clay Mathematics Institute site*.

[4] Diduch Rodrigo Gilberto (2012), *P vs NP*, International Journal of Computer Science and Network Security (IJCSNS) Volume 2, pp 165-167.

[5] Gram Seenil 2001 *"Redundancy, Obscurity, Self- Containment & Independence"* by The 3rd International Conference on Information and Communications Security (ICICS 2001) took place in Xian, China, November

13-16, 2001. Proceedings of ICICS as Volume 2229 of Springer Lecture Notes in Computer Science. Pages 495-501.

[6] Harry R. Lewis and Christos H. Papadimitriou *Elements of the Theory of Computation*, Prentice-Hall, Englewood Cliffs, New Jersey, 1981, ISBN 0-13-273417-6.

[7] Hartmanis, J.*;* Stearns, R. E. *(1 May 1965). "On the computational complexity of algorithms"*. Transactions of the American Mathematical Society. *American Mathematical Society. **117**: 285–306.* . ISSN 0002-9947. JSTOR 1994208. MR 0170805.

[8] Kyritsis C**.** *On the solution of the 3rd Clay Millennium problem. A short and elegant proof that P ≠ NP in the context of deterministic Turing machines and Zermelo-Frankel set theory.* Proceedings of the first ICQSBEI 2017 conference, Athens, Greece, pp 170-181

[9] Kyritsis C THE SOLUTION OF THE 3RD CLAY MILLENNIUM PROBLEM. A SHORT PROOF THAT P ≠ NP=EXPTIME IN THE CONTEXT OF ZERMELOFRANKEL SET THEORY. *International Journal of Pure and Applied Mathematics Volume 120 No. 3 2018, pp 497-510 ISSN: 1311-8080 (printed version); ISSN: 1314-3395 (on-line version) url: http://www.ijpam.eu doi: 10.12732/ijpam.v120i3.1*

[10] Luca Trevisan, *Notes on Hierarchy Theorems*, U.C. Berkeley.

[11] John C. Martin *(1997). Introduction to Languages and the Theory of Computation (2nd ed.).* McGraw-Hill. ISBN 0-07-040845-9.

[12] Papadimitriou Christos (1994). *Computational Complexity.* Addison-Wesley. ISBN 0-201-53082-1.

[13] Rustem Chingizovich Valeyev 2013 *The Lower Border of Complexity of Algorithm of the Elementary NP-Complete Task (The Most Condensed Version)* World Applied Sciences Journal 24 (8): 1072-1083, 2013 ISSN 1818-4952 © IDOSI Publications, 2013.

[14] Stanislav, Žák (October 1983). *"A Turing machine time hierarchy"*. Theoretical Computer Science. Elsevier Science B.V. **26** (3): 327–333.

[15] A. A. Tsay, W. S. Lovejoy, David R. Karger, *Random Sampling in Cut, Flow, and Network Design Problems*, Mathematics of Operations Research, 24(2):383–413, 1999.

[16] Ivanov Viktor V. 2014*, A short proof that NP is not P*. International Journal of Pure and Applied Mathematics IJPAM pp 81-88 .

[17] Woeginger GJ (2016) *The P versus NP page* ,https://www.win.tue.nl/~gwoegi/P-versus-NP.htm

[18] Yannakakis M. 1998 *"Expressing combinatorial optimization problems by linear programs"* Proceedings of STOC 1988, pp. 223-228.

[19] https://en.wikipedia.org/wiki/EXPTIME

[20] Kyritsis K. 2021 "REVIEW OF THE SOLUTIONS OF THE CLAY MILLENNIUM PROBLEM ABOUT P ≠ NP =EXPTIME". September 2021 World Journal of Research and Review 13(3):21-26 https://www.wjrr.org/vol-13issue-3

[21] Kyritsis K. 2021 Author of chapter 5 in the book "New Visions in Science and Technology Vol.6, October 2, 2021 , Page 60-69 https://doi.org/10.9734/bpi/nvst/v6/5176F Published: 2021-10-02 Chapter in a book https://stm.bookpi.org/NVST-V6/article/view/4135

[22] Kyritsis K 2021 Author of a book with title: The Solutions of the 3rd and 4th Millennium Mathematical Problems: The solutions of the Millennium Problem P vs NP in computational complexity and the Millennium problem in fluid dynamics. Publisher : Lap Lambert Academic Publishing. ISBN 6204725629. https://www.lap-publishing.com/catalog/details//store/gb/book/978-620-4-72562-8/the-solutions-of-the-3rd-and-4th-millennium-mathematical-problems

[23] Kyritsis K 2023. A 3 RD SHORTER SOLUTION OF THE CLAY MILLENNIUM PROBLEM ABOUT P ≠ NP =EXPTIME. Conference: 6rth INTERNATIONAL CONFERENCE ON QUANTITATIVE, SOCIAL, BIOMEDICAL AND ECONOMIC ISSUES ,JULY,1,2022, At: ATHENS, CRYSTAL CITY HOTEL **https://icqsbei2022.blogspot.com/2022/06/blog-post.html Proceedings at http://books.google.com/books/about?id=xZnCEAAAQBAJ**

Author: Konstantinos E. Kyritsis Dept. Accounting-Finance University of Ioannina, Greece, ckiritsi@uoi.gr