Global Scientific JOURNALS
GSJ: Volume 12, Issue 5, May 2024, Online: ISSN 2320-9186
www.globalscientificjournal.com

Traveling Salesman Problem in Polynomial Time

Mirzakhmet Syzdykov
Satbayev University, Almaty, Kazakhstan
mspmail598@gmail.com
ORCID ID: https://orcid.org/0000-0002-8086-775X

Abstract. We present the novel research on NP-complete problems within polynomial bound space, P-space. This research is aimed towards the proof of equivalence of complexity classes like polynomial and non-polynomial, which is better known as “P versus NP” theorem. This problem still remains actual as most of researchers suppose that obviously NP isn’t equal to P as there is still no possible solution for problems like Steiner tree or Traveling Salesman Problem (TSP). There’re group of researchers who supposes that proof of equivalence of these classes is almost impossible. We present the new theorem towards the argument of equivalence of main complexity classes like P and NP. For this purpose, we extend our algorithm with the constraints while the main goal is to be achieved on the Steiner tree and TSP circuit on particular specific case whose input is still non-polynomial and converges to the specific non-polynomial number within the number of steps of algorithm.

Keywords: Traveling Salesman Problem, Steiner tree, approximation, algorithms.

Introduction. TSP is well studied in [1] and represents a combinatorial problem of finding the closed path known as Hamiltonian with the minimal length. This path visits each vertex only once, TSP naming is due to the actual statement of the problem where the vertexes in graph are town points.

Byrka et al [2] develop the improved version of the approximation algorithm for Steiner tree within the threshold of 1.39, which is a factor of the algorithm solution within the generalized algorithm. More to write that Steiner tree is a minimal spanning tree for the subset of vertexes of the graph G(V, E), where V is the set of vertexes and E is a set of edges – this holds true for TSP also.

Opposed to Dirac’s theorem [3] we present alternative way of detecting the presence of Hamiltonian circuit or minimal Steiner tree for both versions of our algorithm. Generally, Dirac’s principle states that graph G is a Hamiltonian if the degree of each vertex is more than a half of the number of vertexes:

$$v \in V{:}deg\ deg\ (v)\ \geq \frac{|V|}{2}. \qquad (1)$$

Sohel and Kaykobad give the proofs towards the set of criteria according to which the graph can be considered as having Hamiltonian cycle [4], this problem is also known to be NP-complete.

The polynomial algorithm for finding Hamiltonian cycle is given in [5]. We also refer to the original paper by Stephen Cook [6] for the statement of the relation between complexity classes and in our general algorithm we will give the argument towards proof of equivalence of P and NP.

We discuss the meaning of the timed complexity classes within the bounded spaces along the “a priori " of the past work. The discussed congruent spaces are reminiscent in the sense of the positive logic of the explained controversy which lies beyond the full length of the exposition of NP-hard problems in general due to Karp’s 21 problems. We can state that this dis-relation gives more comfortable addition to the changes in the co-worked companion problems that are to be solved within the times of complexity, as timed regular expressions can be evaluated in the positive logic giving the answer to the “P versus NP" theorem which is omitted in the work due to the proposed conjecture before. Our main purpose is to introduce the timing-ness on the operators which are to be implemented within the linear time and memory space, as we go far beyond the term of determinism

and nondeterminism for a property question arising in the variety of NP-complete and congruent problems to be solved correctly using fuzzy logic of these spaces.

The mean of the timed spaces lies between the incomplete or "PQI"-classes which are actually a residue of the given data within the problem classified as undecidable. All the residues play a vital role in the formation of the descent scientific classes as we have stated that the "P versus NP" problem is listed as decidable within the deterministic Turing automata. The algorithms which are defined as the underlying nature of this theorem give a more profound list for the fuzzy logic of these automata. Thus, we give the full polynomial for the complexities defined in the outer space of the congruent logic of the automata mentioned before. The main results which are obviously screened on the experimental dualistic methodology, give a wide range and set of the algorithms for a better understanding of how the developed approach presented by us.

We go on towards the proof of concept of Millennium Theorem proposed by Stephen Cook, we state that polynomial classes and its successors like non-polynomial (NP) are equal along the function equilibrium which is justified by the parallel processing law, we will also give the exact algorithm to Traveling Salesman Problem (TSP) which is computed in the polynomial complexity time of the product of linear and logarithmic operands, the linear algorithm for the maximum flow problem is also given. In this short paper we give the notion of the equivalence of the complexity classes due to the recent research according to Rabin-Scott subset construction. We will also give a notion to the famous "X+Y Problem" which can be solved efficiently in quadratic polynomial time on the dual deterministic Turing machine, for which the general definition is also given.

We give the strict proof of the inequivalence of the Computational Complexity Classes along with the previously obtained results which clearly give the outcome of the proof of division between P and NP classes which are inequal in general case, later on in this work we present the main discussion of the perspectives of the main outcome.

Algorithm. For the purpose of the binary algorithm which finds Hamiltonian path or minimal Steiner tree, we define the one more condition to hold true:

$$\sum_{i=1}^{n} P_A \leq \sum_{i=1}^{n} P_B \Leftrightarrow max(E(P_A)) \leq max(E(P_B)). \quad (2)$$

The condition (2) defines the relation between the any path P_A and P_B within the edge weights $E(P_A)$ and $E(P_B)$. This condition is absolute and we will shrink it to the statement that the Hamiltonian or Steiner tree is seen from another metric in the source graph, thus, to be a residue from all the possible cycles and trees.

According to this algorithm we can choose even greedy strategy for any starting vertex. The binary algorithm can be introduced in several steps:

1. Choose maximum edge value;
2. Multiply it by two, giving **S**;
3. Perform in a binary search if there exists a Hamiltonian cycle or uncovered subgraph for Steiner tree by known algorithm, increasing or decreasing value of **S**.
4. The complexity is **$O(\log(S) * N^k)$**: $k > 0$.

Where S is the doubled maximal weight of any edge value as defined in algebraic map function for weighted graphs:

$$e \in E, w \in R: E \to W. \quad (3)$$

The partial case where there exist the separate set E_s, so that all the translations of the weights are much smaller than average can be interpreted as follows:

$$e \in E_s: E_s \to \frac{W - avg(W)}{2}. \quad (4)$$

In the case (4) there exist single minimal path in TSP and minimal subgraph for Steiner tree.

The set of fuzzy algorithms together with the mentioned above congruent space gives the evaluation of the functional analysis depicted in the works of Richard Bellman - the author of the works for Dynamic Programming.

These algorithms are qualified as easy to solve, but not easy to verify within the expected function defined as the set of recurrent relations between them. Congruent spaces and algorithms surrounding them are actually probabilistic entities within the latter time and memory space which give the full view of the term of "algorithmical practice". All the descendants of the algorithms in congruent space are concurrently decidable by DP. Our general case for the acquisition of these varieties gives a broader presentation for the full diversity of specialties in the main stage of "computer era" and "applied mathematics".

P versus NP. In this section main proofs towards argument of equivalence of complexity classes are given according to the proposed approximation algorithm. As we have already stated that the solution can be found for the case where there exists the divided subgraph with edge weights smaller on average, we can show that the number of possible graphs is NP-complete, thus:

$$|E \to W| \in NP. \qquad (5)$$

The expression (5) gives the reasonable question about dependence of the set of input data to the set of possible solutions and algorithmic efficiency. Thus, we state that there exists an exact polynomial algorithm for the problem which derives the NP-complete input. For this fact we define the statement of equivalence of P and NP classes within the exact algorithm and the constraint (4):

$$|I| \in NP, A \in P \Rightarrow O(A) \in P. \qquad (6)$$

The expression (6) is defined as the relation between P and NP classes as we devise the modern "witness" theorem, where it's defined that if any of the NP-complete problems have a polynomial solution, then it follows that P equals NP.

Thus, it follows that P = NP as our new argument towards the proof of the equivalence of the different classes of complexities.

Bipartite matching algorithm. We define another step of experimentation with the definition of weighted bipartite matching algorithm. By their nature these algorithms solve the problem of maximal flow in a graph with weights or, simply, the maximal or minimal weight of the matching in a bipartite graph.

According to our observations the minimal weighted matching in a TSP graph solves the problem of finding Hamiltonian path, while the degree of the matching permutation equals to one. In case of more than one group in final solution, we propose a way of the repeated acquisition of the algorithm on a minimal edge.

The algorithm above is also approximation, however, it differs from the main algorithm as it's not defined even for partial case as it's described in "P versus NP" section of this article.

Thus, we get the difference according to the term of algorithms defined on the input with the set of constraints, in our case, there's only single constraint (4).

The algorithm where the groups of permutation are defined is called thus bipartite as this can occur in the general case and is limited to the first feasible obtained solution, where the following holds true:

$$|A| = 1 : A \in P. \quad (7)$$

The expression (7) means that the index of the permutation obtained during the first run equals to one, or, in other words, there's only group and, thus, the algorithm can be considered as successful as it runs in polynomial time.

In general, the bipartite matching in half graphs where each part for TSP is of size |V|, we can get the right point even at once, thus, the probability of the feasible solution is given by the following relation:

$$|A = P| \leq \frac{1}{|V|!}. \quad (8)$$

As this process is probabilistic and, thus, can be evaluated on the deterministic automata, we give another argument towards the equivalence of complexity classes like polynomial (P) and non-polynomial (NP).

Thus, if probabilistic models could be implemented, then we can fully state that P = NP, however, due to the lack of the modern research where this problem could be considered not as a primary target, we can do the rest of work by providing the argumentative answer to the main question by Stephen Cook of whether there exists the polynomial or, in our case, deterministic model of operations so that the key answer could be easily found.

However, as all the above holds true, we state that the polynomial algorithms exist with non-zero probability and, thus, P equals NP.

Congruent spaces.

Yes, spaces are vital in the role of the "congregation" of the algorithms and we are in debt of showing their linear complexity and even "highest mark" of the complexity of O(1)...O(n). These spaces are allowed to learn but are very hard to apply. The current state of the Applied Mathematics and Computer Science gives the full list of the fully comparable fuzzy sets for this in-determinant and brilliant approach. The spaces lying behind the scene of the hard problems are due to the notion of the finitary spaces of algorithm sets with their incomes and outcomes. We learn it on the counter examples not avoiding the decidability of the feasible set of solutions in the time frame O(n) and the verification in time O(1).

Directives on the Dynamic Programming.

Dynamic Programming - a very strong and recurrent set of algorithms based upon them give a literal meaning of the statements induced above for the wide communications between them which gives the property as it's defined in the modern publicistics related to the given subject in general.

These directives give more compelling examples of how the development methodology can be spoken as the "cure" of the incomplete problem approached by the fuzzy logic which is implemented in the modern curvature of Computer Science. We are to give the lease square residues as one of the examples and the residues that are not sparse, concluding that each of them will be given the learn and mean for the consequences of the term "adaptation" presented in the past works of our imminent research and result.

Due to the grammar of the bi-quadrate matrices and their deterministic nature give the positive accomplishment of the hash data structure. Our main role in this aspect is quite easy to understand but not so easy to apply for mastering the already developed and, of course, existent solutions.

We have an opinion on the literal works by Richard Bellman in the whole sequence of the recurrence functions as defined along the optimization theory and Operation Research. Stable sets are not hard to apply but together form the problem of the understanding of the question of how we came to the ingenious solution of the past problems and "practices" in the view of the full knowledge due to the "hierarchical component" and the "compound" formed together.

Time is ambivalent - this statement due to the surrounding view upon Riemann hypothesis and z-function, gives a full list of how it's seen on the problem of the set of co-primes and primes along this function.

General work of this manuscript, actually, isn't giving the definition of "time" and "nature", but the full list for a better comparative study. This study is defined within the step of composite nature of the applied sciences, even known as theoretical background for a successful composition of the viewed works which are dated back to the centuries of the philosophical and scientific history for these permanent composites. The "composition" is a self-taught future and doesn't avoid the term

"formation". These formations are non-atomic, but equivalent futures of the modern prospect for the diversity of the algorithm to be verified and solved in time O(1) on the quantum computers and mechanics of them.

On the timed complexity.

The timed complexity is "ignorant" and we are not to mention this fact, however, due to the supposed solution and the argument towards satisfiability of the 3-SAT problem, we are not to guarantee that our approach is different from others "pondered by" a fuzzy logic and its descendants like the prescription of our scientific letters to the past and present. All the obstructed mechanics and the kinetics of the algorithm deploys the general fact that they are easy to state, but hard to learn - as no one from participants of the co-operative or non-cooperative games can decide it against the crafted intelligence and underlying philosophy for a better summation the primes in congruent spaces actually deciding it along the statement that Complexity Classes are unequal, i.e. P != NP. It naturally follows that no one has an ability in the predicted game to incorporate a specifically designed or understood method for the successful installment.

The game of life proposed by Conway, thus, still remains as a "repository" of the past, future and present for a vital role of the Dynamical Programming in the world of the limited diversities of the stack of algorithms and technologies upon which the research prerogatives are defined on the statement of the "P versus NP"-question, as P equals NP, i.e. P = NP.

Pre-historical facts. We discuss the main statement and proof made by author of "P versus NP" theorem Stephen Cook - a winner of Alan Mathison Turing Award sponsored by Association of Computing Machinery (ACM). In this notable works he first proofs the existence of unsolvable problems along the divided complexity classes known as polynomial (P) which can be solved in the observed amount of time and non-polynomial (NP) which cannot be solved using the approaches developed up to the present time in the fixed time complexity. There and onwards this question was raised for more than decades and still remains actual up to the present time. We will show further that practical algorithms exist which solve the NP-hard problem in a meaningful time span on any type of automata using an algorithmic approach. We will show the proof of concept by proposing first the algorithm to the classical Traveling Salesman Problem (TSP) and a linear algorithm to the maximum flow problem.

We prove the above fact by the assumption already given in the statement before: thus, as we know any non-deterministic finite automata (NFA) can be converted to analogous deterministic finite automata (DFA).

According to our latest research the subset construction is P-complete and, thus, there exist no automata with the strict property given above, thus, it follows that P = NP.

Thus, Klaus Schneider's canonical forms appear almost in any case of in-state minimization and conversion of NFA to DFA via subset construction and, thus, can be replaced by the deterministic turing machine by implying the free parameter-like position in the automata.

Traveling Salesman Problem in almost linear time. From the proof of equivalence of complexity classes it follows that there is a logarithmic-linear algorithm for TSP. We describe this algorithm in a few steps by using the classical Dijkstra approach of finding the shortest path in a connected graph. As the algorithms both are similar, they still have significant differences as for TSP we define the following method of solving this problem in O(n*log(n)): start from the empty set of paths, connect the paths by finding the shortest edge from finish to the start of this path if it doesn't form the cycle, repeat these steps until we get the full Hamiltonian cycle and path.

We're using a simple threading approach when each of the paths is merged on the fly during the simulation of moving travelers as if they would be put on each of the starting vertices.

Proof of P = NP. As we have devised the subset of problems strictly in NP and not in P, we know that subset construction is still P-complete with respect to the pre-defined canonical forms as it was shown before in.

We can only conclude that such forms always lead to the polynomial size equivalent DFA to the source NFA for which the powerset, or subset construction is to be applied in order to get the full state space which is aware of handling the same result without permitting the famous halt problem.

Thus, as we have defined that problems lying strictly in NP-class are executed deterministically with the same time complexity as P-problems we can conclude that this set is empty, which leads to the strict outcome like P equals NP, or P = NP.

For more convenience in the equivalence of P and NP classes as the result of the emptiness of the strict NP-classes of problems, we adjust our argument by building the simple deterministic position automata for the typically parametrized decision problem. This decision problem is approximated by the free parameter which is decreased each time it passes the fragment of built automata where typical logic fails due to the harder representation and re-interpretation of subset construction. We give it a new look and feel with the powerful parametric building which is widely known in the methodology known as “on the fly construction” which was well deployed in software packages like PCRE.

By parameterizing the size of the input clause in the source pattern, we omit the whole logic of automata to be applied which leads to the exponential growth of the reduction spaces which combine the simple logic of regular languages which are recognized by the Turing machine. This technique is novelistic as it wasn’t applied before in problems where the time and space is crucial as we use it for the strict proof as an outcome of P = NP.

Indeed, Stephen Cook in his last message to the computer scientists’ community, proposed that there will be wide horizons of the future built on the assumption that intractable problems could be solved efficiently using mathematical optimization methods like simplex. Indeed, the intractable problems operate only on polynomial space within the non-existent gap of the strictly NP-aware problems, which are resolved by the Schneider-Rabin-Scott theorem of the subset construction being not so “state explosive” as it was meant and opposed before by many researchers as well as many public sources of discussing this question in-depth.

“X+Y” Problem. The dual Turing machines for which we built the full sparse set after sorting both sequences in increasing order can emit a single element each time and then return to the observed state where each row and each column are strictly ordered according to the relation between numbers in sequences and their derived sums. This gives the quadratic solution to the problem which is less in magnitude than the logarithmic, for this fact it can make it more practical.

Meanwhile, dual deterministic Turing Machines, or DDFA, are operated on the matrix rather than a vector tape.

Maximum Flow in linear time. The linear time complexity for the classical network flow problem follows from the algorithm which can be solved in a few steps by introducing the weight potential method which includes the first step as forming the possible paths along the accumulated potential and minimal from the incoming and outgoing flow for the source and sink of the path along the network - the latter step is to compute the potential for each vertex in a flow. This gives the linear complexity and, thus, makes this problem practical which can be used in simulating the natural processes along the user-defined hardware devices which operate on this standard basis.

Proofs and ETC.

The theorem of “P versus NP” states that, first, there are relations between idempotent Complexity Classes which weren’t defined in time which led to diverse outcomes stating the not tractable and readable proofs of the obvious question if P would equal NP.

The main perspectives led to our definition of the Functional Hypothesis which led to the outcome of the existence of the feasible solution for NP-hard problem, which was obviously disproved. As it goes further, we are to define the parallel computation model which is weak before the algorithmic approaches of efficiently solving NP-hard problem, thus it follows that:

$$NP = N \cdot P.$$

From the above it follows that even finitary model of computation of either Turing or non-Turing machines is improper to define the solution which could be verified in polynomial time or even sought within the pre-defined models of computation.

Yes, of course, the outcome of P = NP would lead to fantastic results like if the modern cryptosystems could be compromised in almost linear time – that's not truth since even high-end computation model like parallel systems cannot solve the NP-hard problem in the visible amount of time, as it's shown previously.

As we have also shown in our prior works determinately devoted to the resolution of "P versus NP" theorem proposed by Stephen Cook, we can convert the Non-deterministic Turing Machine (NDTM) to the equivalent Deterministic Finite Automaton (DFA), which correctly throughputs all the results of the output produced by NDTM during the operable simulation on the given input, same for either NDTM or DFA. This fact justifies the dramatic case or simply "error" which researchers permitted before by stating that from the fact, if P = NP, it follows that we can efficiently solve the NP-hard problems. Thus, if even according to our final proof P and NP are different and inequal as well as other classes according to the Parallel Computation Law (PCL), it doesn't mean that NP-hard problem cannot be solved efficiently as we can convert the Turing Machine and apply the law in a more different and elegant way by choosing the proper strategies for converting NDTM to DFA by Subset Construction as it was opposed due to Michael O. Rabin and Dana S. Scott and their seminal work, which was honored by the Association for Computing Machinery (ACM) as Turing Award. The latter results were shown to solve even back-reference problem which is NP-hard strictly in polynomial time by applying pre-computation rule. Still, even if the problem leads to the estimated exponential or non-polynomial growth, the accuracy of algorithm gives the solution which is almost as effective as the basement upon giving the 100% solution. According to our pre-dominant research even heuristics can give almost poly-linear solution with a tight bound of accuracy whose speed is much lower than the speed of growing complexity of the problems like MAX-SAT or TSP.

The mathematical sense and the common understanding of the importance of the "P versus NP" question gives the rise to the operational relation between functions and their finitary properties. These properties give the main proof of finitary nature of the geometrical sense of the point lying on the surface as well as the finite polynomial class belonging to the high-speed growing NP-class.

Here we give the proof by PCL as follows:

$$\lim_{\top}(N\to\infty)_{f_0}\ [\![NP/N=P,\ P=NP(\Rightarrow^{\perp})\ 0=1(\Rightarrow^{\perp})\ P\neq NP.]\!]$$

As it follows from the axiomatization principle of P and NP as decidable and undecidable is also incorrect giving the partial proof of existence of the point on spherical surface which lies strictly on it within the multi-dimensional space.

Thus, even if we have shown that P and NP are inequal, there is still the set of problems which can be NP-hard and reduced to the proper computational class in the strictly linear and hierarchical order according to the relation between the complexities of these classes. The fact that if there is the set of problems which can be solved by a polynomial algorithm and in the same time belonging to the non-polynomial class is just a way of showing that P = NP, still as we can see this is a partial solution as there can be problems which belong to P and NP classes respectively due to the initial proof that P is a subset of NP.

The common mathematical axiomatization principle is a way of expressing the correct statement, however, due to the Cook's proofs, it follows that there are problems which lay strictly in NP-hard class and belong to NP as well to the minor P-class.

The TSP principle on integral circuits is a way of solving the problem in full state space with the help of the candidate choosing algorithm which was well studied in Yen's algorithm.

The novel proof scheme by the statistical seed of probability was also presented, giving the underline of the existence of one-way hashing functions.

The genuine finding is due to Akram Louiz about the cardinality of complexity classes which gives the indispensable proof of the difference of classes and the existence of the relation between them which is arbitrated by the PCL. Other researchers tend to the mathematical sense of the theorem as there is no definition of the relation of Complexity Classes and the opposition of it to the infinity operator and, in common, the infinitary spaces or sets.

Conclusion. We have presented one approximation algorithm for both TSP and Steiner tree and one algorithm for TSP as a weighted bipartite matching. Both algorithms operate on graphs. Within the constrained graphs we can also conclude that P equals NP as the constraint doesn't limit the size of the input and, in general, this fact is to be considered further as NP-complete problems are mostly classified according to the representation of the problem as the 3-SAT problem, which is known as "Boolean satisfiability".

We have showed that the number of possible cases in general is a combinatorial number and approximates to almost factorial like the number of sums as this's opposed to the classification of the problem to 3-SAT.

On the other hand, the algorithm proposed is binary and can be applied to the limited cases as per the constraint, from that follows that constrained problems are either P-complete or P = NP. As the number of input cases tends to factorial number, it can be stated that P = NP.

The statement above is our general argument towards the proof of equivalence of complexity classes according to the fact that there exists a polynomial solution for NP-complete problem.

We have also showed that the initial version of the algorithm without permutation subgroups can be applied to both TSP and Steiner tree as for TSP the devised path can be found even in greedy manner and for Steiner tree the tree connectivity can be checked and used as the step of the binary algorithm to increase or decrease the lower or upper bound of the edge weight value S.

Thus, it follows that P = NP if the constraint holds true and the general statement is true.

We have also given another argument towards the statement above on deterministic and probabilistic models where the feasible solution could be found in polynomial time with pre-defined threshold.

We also give the constructive definition of the nature of the operational models to conclude the invariance of NP-class to the P-class of complexity.

Thus, all the important results presented here can be evaluated further and the question of the relation between constrained and probabilistic models to the deterministic one is to be evaluated.

Thus, it is obvious that no problem is timed, but it is hard to acquire the meaning of our controversies of the lifetime age of the present computational complexity within the limited definitions. Our methodology based upon Richard Bellman's Dynamic Programming is actually equivalent to the linear model, but not equivalent to the models of time and verification which can be done only in O(1) and actually converges to the total zero of o(f(x)). The complexity of the function in this case is the only starting point for successful back propagation of the scientific models used in the modern Software and Technology and it's actually not random in the games proposed by Nobel Laureate, John Forbes Nash, whose equilibristic games solve the situations in which the Intelligent Complexity is unnecessary and can be avoided on the term of its timed definition presented in this work.

Current statement doesn't adhere to the argument towards the linkage of any of the facts together forming the communicational point of view of our vision of the future of the novice and effective algorithms for a better understanding of what we are to learn and what we are to apply, which is actually is the main conclusion of the discipline, as Applied Mathematics.

On this subject, we conclude that P isn't a subset of NP-complete classes and their hierarchies along the full-length algorithmic performance and stability on the defined systems upon which, we are to eliminate the fuzzy pattern which gives the fully probabilistic process of the algorithmic undecidability of any problem.

Better learning, thus, is better acquisition and actually is a general synthesis of the algorithm for the further evaluation on the common aspect and final stage of its implementation.

Given all the arguments from the above towards the concretization of our main question of the application of Computational P-NP model, we are to go step further in the list of diversities of these classes as it can be seen from top down paradigm of our informational era and evaluation of this era on the bound of inevitable gain for past and present.

We are to conclude that spaces are equal, but not decidable and this is a general question permitted by the Computational Complexity Theory, which is theoretically easy to solve, but is actually hard to implement on the modern bitwise logic, giving the solution for a compound and immense application of Quantum Computing.

The common misinterpretation of the "P versus NP" theorem lies between the fact that it can be solved effectively, still, it doesn't follow that from this consequence we can devise the relationship between two classes. We give the notion of non-existence of the problem lying in-between NP and P classes of complexity which are definitely to be operated on non-deterministic Turing machines, or NFA, and deterministic Turing machines, or DFA. Thus, there's no difference between deterministic and non-deterministic approach which is replaced by the deterministic parameterized search and there's no difference between this two approaches as problem always lies inside the deterministic Turing machine and the polynomial class of complexity - as we have shown above there's nothing which will lead to the different scenario.

We have also developed stable and minimal in complexity time solutions for the Traveling Salesman Problem and Maximum Flow Problem - which results from the proof of the functional conjecture of the complexity classes along the equivalence function of P and NP. The further steps are to find other polynomial solutions for the problems which weren't practically before.

As we have shown the almost axiomatic proof of inequivalence of P and NP-classes, we have also given the full law along which the relation between Computational Complexity Classes can be devised. We have also proved that according to PCL the strict hierarchy along the relation of complexity classes exist. Thus, we have also shown that existence of polynomial solution for NP-hard problem is a partial case of proof of the equivalence of P = NP due to the fact that the problem can lay in both classes as P is a subset of NP-class. The mathematical sense of the problem cannot be also omitted as the relation between classes according to their complexity gives the solution of the classes of congruence of the functions as well as the main theorem that point can belong to the sphere in any dimension.

Thus, we went through a long way of misunderstandings and mis proofs as well as the correct notation of the initial statement of the important theorem and stand-back from the long-time crisis in Computer Science as the proof of inequivalence of Complexity Classes doesn't mean that NP-hard problem cannot be solved efficiently in polynomial time, still it follows that we are to review all the pros and cons of the underlying theory and its evaluation through the paradigm of time and Mathematics.

Acknowledgements

The author expresses gratitude to all the researchers who contributed towards the resolution of "P versus NP" question and the approximated algorithms which can solve them in a definitive and polynomial complexity.

Funding

This work was fully supported by the educational grant of MES RK.

References

1. Davendra, Donald, ed. Traveling salesman problem: Theory and applications. *BoD–Books on Demand*, 2010.
2. Byrka, Jaroslaw, et al. "An improved LP-based approximation for Steiner tree." *Proceedings of the forty-second ACM symposium on Theory of computing*. 2010.
3. Li, Hao. "Generalizations of Dirac's theorem in Hamiltonian graph theory—A survey." *Discrete Mathematics* 313.19 (2013): 2034-2053.

4. Rahman, M. Sohel, and Mohammad Kaykobad. "On Hamiltonian cycles and Hamiltonian paths." *Information Processing Letters* 94.1 (2005): 37-41.
5. Riaz, Khadija, and Malik Sikander Hayat Khiyal. "Finding hamiltonian cycle in polynomial time." *Information Technology Journal* 5.5 (2006): 851-859
6. Cook, Stephen. "The P versus NP problem." *Clay Mathematics Institute* 2 (2000).