# "P = NP" versus "P ≠ NP"

Rustem Chingizovich Valeyev

NPO CKTI, Joint-Stock Company I.I.Polzunov Scientific & Development, Association on Research and Design of Power Equipment, Atamanskaya str., 3/6, St-Petersburg, 191167, Russian Federation

**Abstract.** Complexity of algorithm of task's exact solution is always dictated by basic features of the task. Multiplicity and variety of tasks in the class of complexity NP, for which effective algorithms have not found yet and also their analogues in many fields of knowledge, allow to assume, that the phenomenon known in mathematics as "problem P vs NP", grows out of this science. Consequently, the essence of a source of difficulties for developers of algorithms can be as fundamental as simple and available to understanding without appeal to the most difficult formulas and vulnerable multistage theoretical constructions. This source has to be in internal structure of a task and has to be common for all NP-complete tasks.
[Valeyev R.C. **"P = NP" versus "P ≠ NP".** *Life Sci J* 2014;11(12s):506-512] (ISSN:1097-8135). http://www.lifesciencesite.com. 109

**Keywords:** Complexity, NP-complete task, problem P vs NP

## Introduction

Nowdays thousand tasks for which effective (polynomial) algorithms of their exact solution are unknown have already found. Discrete tasks, for which the polynomial of the task's dimensions (from length of an input) algorithms exists, create the class P (tasks, which can be polynomially solved on determined Turing machine). Discrete tasks, for which such algorithms are unknown, create the class NP (tasks, which have polynomial solving by non determined Turing machine). Class NP includes so called NP-complete tasks as an subset, to which any tasks from NP (if they do not concern to class P) reduce polynomially.

What is the reason of creation failure of effective algorithm? Is that so because it does not exist in nature, or it does but it is difficult to find it? This unsolved till now fundamental problem was found 42 years ago independently from each other by S.Cook and V.Levin and it got its name "problem P versus NP" [1] ... [10]. And its brief formulation is the following: do classes P and NP coincide with each other? Does effective algorithm of solution at least one NP-complete task exist?

## Mini-glossary

- problem - (Hellenic: "problema" - "difficulty", "barrier", "problem", "task") - complex and inconsistent theoretical or practical question demanding development of ways for achievement of the wide range of one-type purposes.
- task – much narrower in comparison with "problema", concept: all that demands achievement of one obviously designated purpose - for example, receptions of the answer on certain question.
- mathematical task – necessity of reception of an unknown with a help of a known by means of these or those actions with quantitative, topological or combinatory items.
- general problem – mathematical task in conditions of which values of its parameters (of its free variables) are not determined; the same as the English term "general problem"; synonyms in Russian: "mass task", "general task", "algorithmic problem".
- individual case of general problem – mathematical task, in initial data of which all concrete values of its parameters (of its free variables) which allow to receive the concrete required answer of this task are declared; synonym: "individual task"; synonyms in Russian: "individual task", "solitary task", "private task"; in the English for a designation of this important narrow concept words with much more common sense are frequently used: "problem" or "task".

Examples of the problems: "Riemann hypothesis", "P vs NP", "Hodge conjecture", etc.

Examples of the tasks (these general problems concerning the same problem "P vs NP"): maximum matching, "Shortest Route Problem" (SRP), "Knapsack Problem" (KP) etc.

General problem "Knapsack Problem": G - carrying capacity of the container, k - quantity of sorts of subjects for transportation, $n_1$, $n_2$, ... , $n_k$ – quantity of identical subjects in each sort, $g_1$, $g_2$, ... , $g_k$ – weight of one subject of each sort; it is necessary to find the maximum possible sum of weights of these subjects which does not exceed G.

An individual case of the general problem KP: G = 1 ton, k = 6, $g_1$ = 24 kg, $n_1$ = 18, $g_2$ = 35 kg,

$n_2 = 15$, $g_3 = 54$ kg, $n_3 = 13$, $g_4 = 66$ kg, $n_4 = 4$, $g_5 = 98$ kg, $n_5 = 7$, $g_6 = 219$ kg, $n_6 = 3$.

- accessibility of parameter A to the developer of algorithm - opportunity for developer of an algorithm for a general problem to realize application for all possible values of parameter A in order to get an exact answer of any individual case of this general problem.
- accessibility of set B to the developer of algorithm - accessibility of all concerned parameters of all members of set B.

Example of accessible and inaccessible sets (sets of the data): mathematical model of motion of every ball playing billiards - and the same for every molecule in the Brownian motion in one drop of water.

- regular [on parameter P] set – set where parameter P of set members adheres strictly to one rule, which is common for all values of this parameter (this rule can be computing or/and geometrical or/and topological or/and logic or/and verbal).

Comment: if this "one common rule" represents a certain finite set of narrower rules, then such set is accessible for the developer of algorithms (i.e. it is usually completely defined by a rather compact analytical formula, system of the equations, verbal description, pattern etc).

Examples of regular sets: numerical series, real numbers, values of analytical functions, points of a surface of all geometrical spheres existing in the nature, set of regular verbs in the English, etc.

- irregular [on parameter P] set – set where parameter P of set members:

are not subjected to one rule, which is common for any values of this parameter,

or

are subjected to such rule, but the developer of algorithm can not operate all consequences of use of this rule in algorithm; antonym: "regular [on parameter P] set".

Comment: the set of these or those combinations of members of an initial set in combination theory isn't a regular set. Though each rule in combination theory is rather laconic, it represents not a compact computing formula, and only the instruction which is useless from this point of view.

Examples of irregular sets (situation "a)" in definition "an irregular set"): incidentally appointed numbers, randomly filled matrixes, images on separate frames of the video movie, points of a surface of all potatoes existing in the nature, the set of irregular verbs in English, etc.

Examples of irregular sets (situation "b)"): the sets formed of combinations of an initial set (permutations, combinations, etc), result of work of a random number generator, etc.

- sovereign [on parameter P] set - regular set in which parameter P submits to a rule (law, dependence) which operates only in this set.
- autonomous set - set which is not connected by any accessible dependence to one other set.
- unique set - set which is not being identical any other set.
- cluster [of parameter P] - an interval of values (range, domain) of variable P (or domain of set of values of parameter P) within the limits of which all values P are regular set.

For unambiguity the words "solution of a task" is understood only actions that aim to get an answer of this task, and the term "answer [of the task]" always means the result of these actions.

Often there is no difference between concepts "variable" and "value of variable" in mathematic practice. In general case it isn't fair. Later such comprehensions as "variable" and "value of variable" will be strongly differ from each other and mean respectively "set which members define or change any functional or other dependence" and "separate member of such set".

- N-dimensional space of a task - Cartesian product of N sets; each of these sets is the set of all the values of one of the N (where $N \geq n$) variables that exist in the task (n is the quantity of unknown); synonym: "set of possible answers of a task", "senior space".
- possible answer [of a task] - any point of N-dimensional space of the task.
- forbidden answer [of a task] - any member of the set of possible answers of the task which can be its required answer.
- allowable answer [of a task] - possible answer of the task which is not a forbidden answer.
- exact answer [of a task] - subset of set of allowable answers; each set member of this subset completely satisfies to all those requirements to exact answer which contain in initial data of this task.

Bulky expression "clusters, objectively existing in set of values of an independent variable" further is replaced on conditional, as well as all mini-glossary used here, the working formulation "junior clusters". And "clusters on which after the announcement of junior clusters the N-dimensional space of a task (set of possible answers) are divided" - "senior clusters".

Even the most successful and respected algorithm can't be regarded as exact (i.e. not only as

effective, but as full too) if it can not guarantee reception of the exact answer of everyone without exception of an individual case of a considered mass task. In mathematics by default as the task answer its exact answer, and as algorithm of the solution of a task – its exact algorithm.

In search of effective algorithms for class NP it is enough to analyse only one (any) NP-complete task - for example, TSP.

Very brief sketch of the analysis of structure of TSP and also schemes of the elementary ways of proofs for both variants of the decision of the dilemma "P ? NP" are given below. More strict formal proof is submitted in [8] (the unwillingly condenced text with unfortunate misprints and illustrations which due to the low resolution at the publication have lost a lot of key information) and in [10] (complete text with all comments and high quality illustrations).

**1. On the ground floor of the problem "P vs NP"**

The task "Traveling Salesman Problem" (TSP), Fig.4: the graph with n node and with arbitrary number of arbitrarily located edges of any length is set. It is necessary to find such sequence from n of not repeating edges ("route") which begins in a certain initial node, consistently visits all other nodes and comes back again to initial node. It passes only one time through each node and at the same time has the smallest length from all other such routes, possible on this graph.

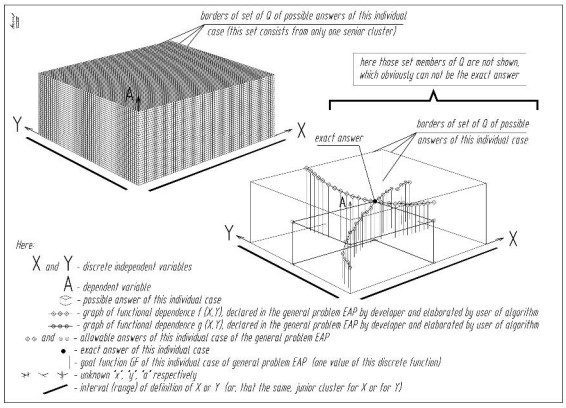


**Fig.1 All possible and allowable answers of an individual case of the general problem EAP and the general principle of detection of the exact answer**

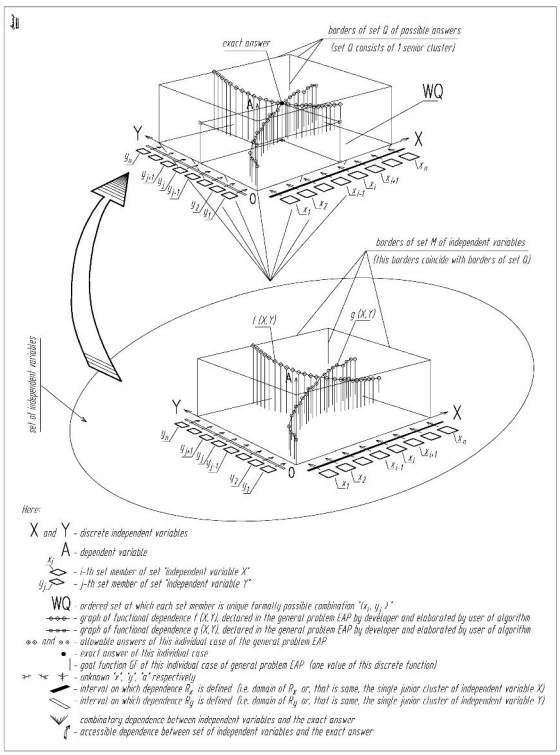


**Fig.2 Features of the internal structure of the task EAP (a classical algebraic task)**

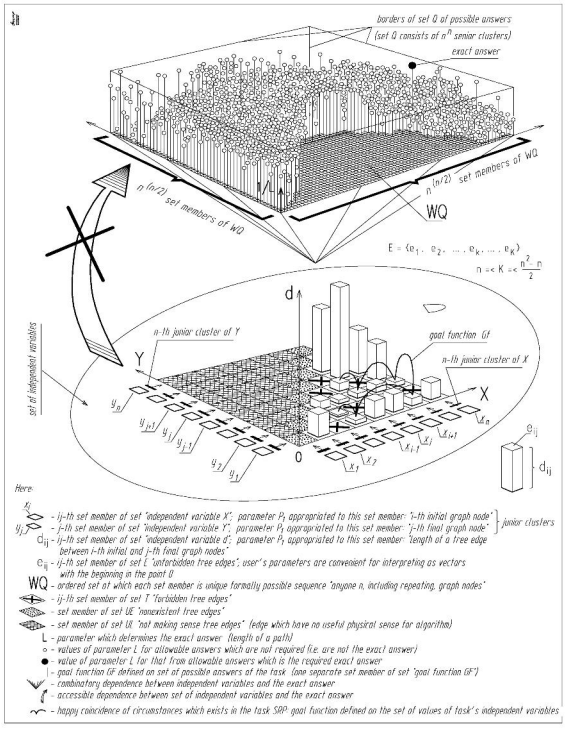


**Fig.3 Features of the internal structure of the task SRP (a task of class P)**

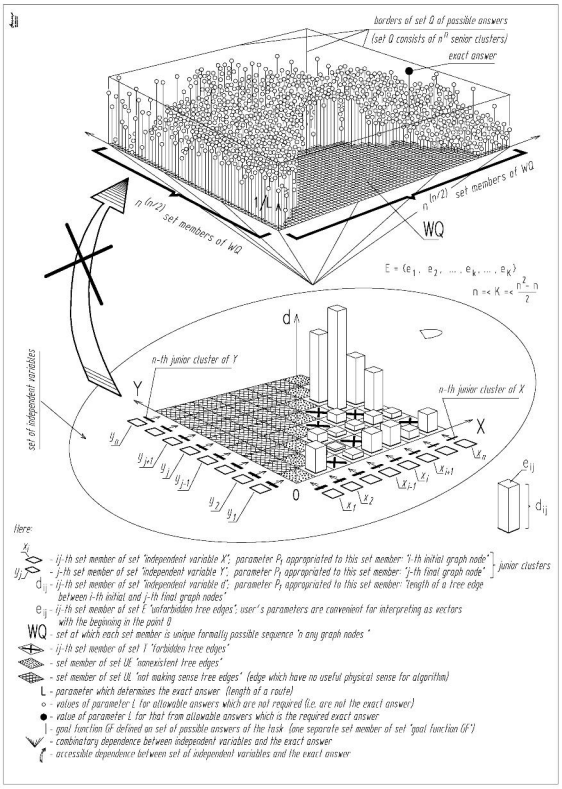


**Fig.4 Features of the internal structure of the task TSP (a task of class NP)**

Vivid counterexamples of tasks on discrete structures:

A typical algebraic task ("Elemental Algebraic Problem", EAP), Fig.1, 2: two or more variable are set and certain requirements to the required answer, expressed in the form of functional dependences between these independent and dependent variables and other conditions. By means of these or those mathematical manipulations and calculations it is necessary to reveal crossings of those sets which values of the mentioned functions are.

The task "Shortest Route Problem" (SRP), Fig.3: conditions of this task of class P differ from conditions of the NP-complete task TSP only that the shortest way shouldn't be closed (i.e. it shouldn't be continued from final graph node back to the initial node).

### 1.1. Sets of independent variables and the sets of possible answers: clusters

From physical sense of Cartesian product follows that:

$$V=v_1 \times v_2 \times \ldots \times v_z$$
$$W=w_1 \times w_2 \times \ldots \times w_z$$

here:

V - power of set of possible answers in a task

$v_i$ - quantity of values of variables in i-th variable; i=1, 2, 3, ... , z

W - quantity of senior clusters in a task

$w_j$ - quantity of [the junior] clusters in j-th variable

Classical, "one-structural", situation: each set of variables in a task consists from 1 cluster (Fig.5):

$$w_1=w_2=\ldots=w_z=1$$
$$W=1$$

Here for any possible answer is fair such superposition of functions, which are obviously declared (or implicitly exist) in the set of possible answers and in the sets of variables.

Goal function (i.e. dependence GF between independent variables and the required exact answer of a task) here too will be the complex function which arguments are these independent variables, and codomain - all set of possible answers.

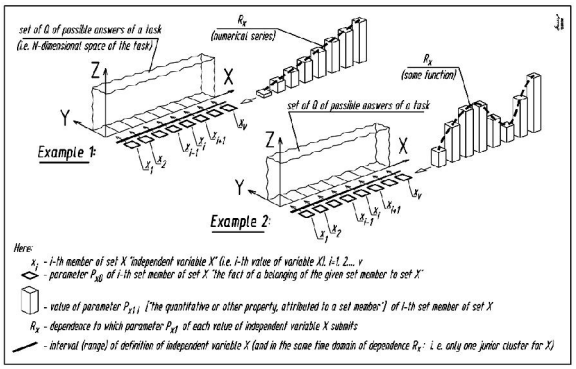


**Fig.5 Discrete independent variable in a task: parameter $P_1$ of each value of this variable submits to 1 dependence $R_x$**

Another possible, "multistructural", situation: at least one independent variable consists from more than 1 cluster (Fig.6):

$$w_1 \geq 1,\ w_2 \geq 1,\ \ldots,\ w_z \geq 1$$
$$W>1$$

All said above for a "one-structural" situation, naturally, fairly and here – but only within one separately taken senior cluster which is sovereign, autonomous and unique. Different junior clusters by definition are unique and don't depend from each other therefore in the senior cluster everyone the g-th unique superposition of functions, including the g-th unique "cluster goal function" $Gf_g$ will work also.

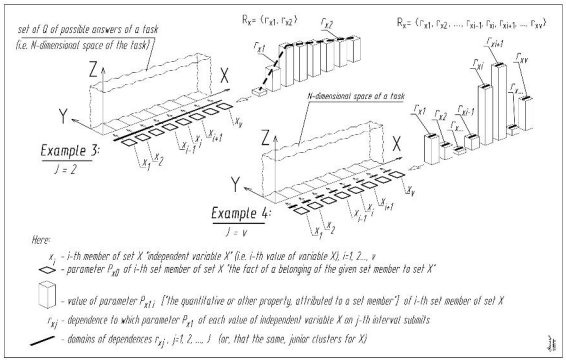


**Fig.6 Discrete independent variable in a task: parameter $P_1$ of each value of this variable not submits to 1 dependence**

Goal function for a task as a whole (i.e. dependence GF = {$Gf_1$, $Gf_2$, ... , $Gf_g$, ... , $Gf_G$} between independent variables and the exact answer of a task) here too will be the complex function which arguments are all unique "cluster goal functions" $Gf_g$, g = 1, 2, ... , G.

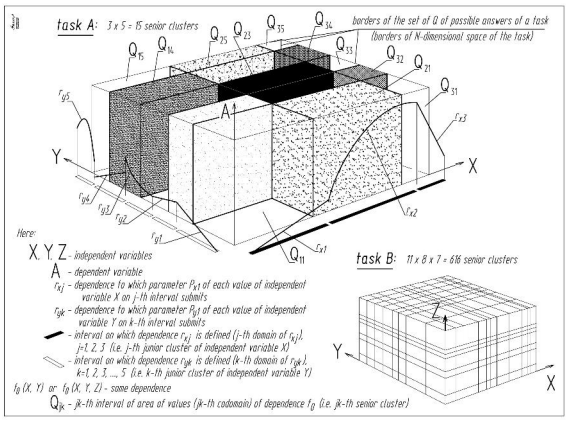


**Fig.7 Intervals of area of values of dependence $f_0$ in the set Q of possible answers (in N-dimensional space) of a task**

### 1.2. The initial cause of the problem

Unlike the "one-structural" tasks in the TSP there are no numerical axes and other regular sets. Here the user of algorithm always declares any and irregular set of values of independent variables: in the conditions of TSP there are no restrictions concerning "pattern" of not forbidden edges of the graph and lengths of edges. Therefore here are always set from n to $(n^2-n)/2$ autonomous sovereign junior clusters $e_{ij}$. Each of them represents a point interval (point range) within which a certain unique function $d_{ij}= qw_{ij} (x_i, y_j)$ is declared. Because of extremely possible minimalism in the extent of an interval of change of arguments (i.e. minimum possible domain) it has minimum possible codomain (a line) and a unique value (a constant which means length of the edge).

Lengths of edges, which are declared by the user of algorithm in each separate individual case of mass task TSP (and that the user thus nolens volens generates in this task), it is possible to name "a chaotic set of constants", "a tabular function", "an unique "irregular" dependence", "a set of a random quantities", "a collection of piecewise analytic functions (each separate "piece" of which is declared only in one point and has only one value-constant)", or somehow differently. The essence of an affair from it does not vary: the user always brings in dependences between the initial data and the required answer of the task the primary disorder (absence of analytically controlled law, "irregularity"). And this inevitable primary absence of dependence cannot be removed from the task for the same reason on which it is impossible to remove soluble pollutions from the river if in the river head there is constant source of pollutions.

### 1.3. Implication No1: goal function

The set Q of possible answers consisting of $n^n$ of the senior clusters, is sovereign, autonomous and unique because Q is Cartesian product of n sovereign, autonomous and unique junior clusters.

Q (or WQ) is a set of arguments on which goal function GF is defined.

Therefore, GF which is function of $n^n$ above-mentioned arguments, is sovereign, autonomous and unique too (i.e. existing only in this individual case of the general problem TSP), a set.

### 1.4. Implication No2: individual case of the general problem

The set "goal function" in each individual case of a "multistructural" general problem TSP is a sovereign unique set. Any algorithm is finite set of the actions concerning goal function. Consequently, to each individual case of TSP there corresponds algorithm which can find the exact answer only of the given individual case.

### 1.5. Implication No3: complexity of an algorithm

The complexity of any general problem T (the minimal computing complexity of algorithm for T) is determined by three parameters:

- quantity of obligatory values of parameters which necessarily should process such algorithm in order to get the exact answer of any individual case of general problem T;
- quantity of obligatory operations which such algorithm necessarily must commit in

order to get the exact answer of any individual case of general problem T;

- power of the set "algorithm of exact solution of general problem T".

Any algorithm of exact solution of any individual case of general problem TSP is unique finite set of data and descriptions of operations over this data, the implementation of which lead to the getting exact answer only of this individual case of general problem TSP.

Any general problem by definition is infinite set of its individual cases. Consequently, complexity of algorithm of the exact solution of general problem TSP is equal to infinity.

**4. The scheme of the proof of the statement "P ≠ NP"**

in TSP it is necessary to prove, that**:**

- In this task all independent variables always are irregular sets.
- The set of possible answers of task TSP is always divided on exponential quantity of unique autonomous clusters.
- Goal function in this general problem is inaccessible set.
- ✓ The exact answer of concrete individual case of general problem TSP can be received with the help only that unique algorithm which is created specially for this individual case.
- The complexity of exact algorithm for general problem TSP is equal to infinity

Consequently, there is no effective (polynomial) algorithm of the guaranteed reception of the exact answer of the general problem TSP.

Facultatively: complexity of exact algorithm for any individual case of general problem TSP is exponential size.

about other tasks in NP:

- That is fair from the point of view of complexity for NP-complete task TSP, is fair for any other task in class NP which does not belong to class P.
- Consequently, P ≠ NP.

**5. The scheme of the proof of the statement "P=NP"**

in TSP or in any other NP-complete task it is necessary to prove, that**:**

- In this task all independent variables always are regular sets.
- Goal function in this general problem is accessible set.

If it will be possible, any more will not make the big work to show, that:

- The exact answer of any individual case of general problem TSP can be received by means of the same algorithm (or set of algorithms).
- Complexity of this set of algorithms is not exponential or infinite.

in any task from NP:

- That is fair from the point of view of complexity for a considered NP-complete task, is fair for any other task in class NP, i.e. P = NP.

**In closing**

All difficulties around the class NP are consequences of the initial cause mentioned above. And than above in crown of a tree of consequences difficulty settles down, especially is respectable and confused it looks.

Almost full identity of conditions of problems of SRP and TSP doesn't disprove, and on the contrary, confirms everything told about TSP. In any individual case of SRP instead of unique exponential goal function of GF it is possible to apply polynomial Gf (see Fig.3). The effective polynomial greedy algorithm uses that fact that at identical quantity of summands the smallest sum always is the sum of the smallest summands. Therefore he can choose step by step all fragments of the exact answer (i.e. separate edges of the graph) from a polynomial set of junior clusters. But TSP, as well as to other NP-complete tasks, hasn't the luck to possess such saving "lateral pass". In TSP information on the exact answer is only in an inaccessible exponential set of the senior clusters and anywhere more. All known algorithms for TSP deal only with junior clusters – for this reason all of them are only approximate algorithms.

The problem "P vs NP" represents one of displays of the fact of existence of sets which are inaccessible to the detached onlooker. Examples: any irrational number, the set of all prime numbers, weight of each grain of sand within Sahara, etc.

**Conclusion**

Effective algorithms of the exact solution of tasks of class NP do not exist, i.e. P ≠ NP.

**Corresponding Author:**
Dr. Valeyev Rustem Chingizovich
NPO CKTI
Joint-Stock Company I.I.Polzunov Scientific & Development Association on Research and Design of Power Equipment
Atamanskaya str., 3/6, St-Petersburg, 191167, Russian Federation

**References**


1. Cook, S., 1971. The complexity of theorem-proving procedures. Proceeding of ASM STOC, 71: 151-158.
2. Garey, M.R. and D.S. Johnson, 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H.Freeman, pp: 340.
3. Goldreich, O., 2010. P, Np, and Np-Completeness: The Basics of Complexity Theory. Cambridge University Press, pp: 214.
4. Feldmann, M., 2012. From classical versus quantum algorithms to P versus NP. Date views 30.05.2012 www,arxiv.org/abs/ 1205.6658v2.
5. Duan, W-Q., 2012. A Constructive Algorithm to Prove P=NP. Date views 31.07.2012 www.arxiv.org/abs/1208.0542.
6. Malinina, N.L., 2012. On the principal impossibility to prove P=NP. Date views 15.11.2012 www.arxiv.org/abs/1211.3492.
7. Nuriyev, D., 2013. A DP Approach to Hamiltonian Path Problem. Date views 14.01.2013 www.arxiv.org/abs/1301.3093.
8. Valeyev, R.Ch., 2013. The Lower Border of Complexity of Algorithm of the Elementary NP-Complete Task (the most condensed version). World Applied Sciences Journal, 24 (8). Date views 04.09.2013 www.idosi.org./wasj/ wasj24(8)13/16.pdf.
9. Gillet, F., 2013. Solving 3-SAT and 3-dimensional matching in polynomial time. Date views 07.10.2013 www.arxiv.org/abs/1310.1971.
10. Valeyev, R., 2012. Nontraditional algorithms for logistical tasks. Ostrovityanin, pp: 160.


6/29/2014