Redilat
Red de Investigadores Latinoamericanos
LATAM REVISTA LATINOAMERICANA DE CIENCIAS SOCIALES Y HUMANIDADES

DOI: https://doi.org/10.56712/latam.v4i6.1468

Is P equal to NP? Answer and Method to solve puzzles with God's algorithm

¿Es P igual a NP? Respuesta y Método para resolver rompecabezas con el algoritmo de Dios

Nilson Rafael Bolívar Barrios
nilsonbolivarbarrios@gmail.com
https://orcid.org/0000-0002-3933-0969
Independent Researcher
Barranquilla – Colombia

Artículo recibido: 24 de noviembre de 2023. Aceptado para publicación: 11 de diciembre de 2023
Conflictos de Interés: Ninguno que declarar.

Abstract

The complexity of any type of problem can be from the easiest to the infinitely complex, there are problems where there are shortcuts with God's algorithms and they can be solved in polynomial time, and there are problems where with God's algorithms there are no shortcuts and no are solved in polynomial time, if a non-deterministic Turing machine finds solutions to problems in polynomial time this means that a deterministic Turing machine with God's algorithm can also find solutions to the same problems in polynomial time, when we have a problem we will solve it with God's algorithm, we can have 1, 2 or more God's algorithms that solve the same problem with the same number of steps. As a result, we use God's algorithm, and we observe how many steps this problem is solved in. It can be many steps or few steps, and we make the decision if it is worth spending time to solve that problem. The author presented the results that it is impossible to solve the problem in fewer steps than the number of steps of God's algorithm.

Keywords: God's algorithm, complexity, solution, P, NP

Resumen

La complejidad de cualquier tipo de problema puede ser desde el más fácil hasta el infinitamente complejo, hay problemas donde hay atajos con los algoritmos de Dios y se pueden resolver en tiempo polinómico, y hay problemas donde con los algoritmos de Dios no hay atajos y no se resuelven en tiempo polinómico, si una máquina de Turing no determinista encuentra soluciones a problemas en tiempo polinómico esto significa que una máquina de Turing determinista con el algoritmo de Dios también puede encontrar soluciones a los mismos problemas en tiempo polinómico, cuando tengamos un problema lo resolveremos con el algoritmo de Dios, podemos tener 1, 2 o más algoritmos de Dios que resuelven el mismo problema con la misma cantidad de pasos. Como resultado, utilizamos el algoritmo de Dios y observamos en cuántos pasos se resuelve este problema, pueden ser muchos pasos o pocos pasos, y tomamos la decisión si vale la pena dedicarle tiempo a resolver ese problema. El autor presento los resultados de que es imposible resolver el problema en menos pasos que la cantidad de pasos del algoritmo de Dios.

Palabras clave: algoritmo de Dios, complejidad, solución, P, NP

Como citar: Bolívar Barrios, N. R. (2023). Is P equal to NP? Answer and Method to solve puzzles with God's algorithm. *LATAM Revista Latinoamericana de Ciencias Sociales y Humanidades 4(5), 875 – 897. https://doi.org/10.56712/latam.v4i6.1468*

INTRODUCTION

In this research we talk about God's algorithm, which is the optimal solution to puzzles. As an example, a puzzle with a cylinder divided into 4 parts is given as an example. Problems of type P and NP are also discussed, and the answer to the millennium question is given, which is: Is P equal to NP? The purpose of this research is to answer one of the seven millennium questions for the advancement of science, especially in the advancement of mathematics and computer systems.

What motivated me to carry out this research is that I like the extreme and the complexities reach infinity, and in this research there is a connection from the least complex cylinder puzzle to the infinitely more complex one and using the same method these puzzles can be solved with God's algorithm.

Use God's algorithm to solve the problems. Remember that the complexity of problems can reach infinity. Below we will use as an example a puzzle of a cylinder divided into 4 parts that will verify if it is well assembled or not in approximately one minute, the cylinder will have a spiral that will help us know if the cylinder is well assembled or not, a table will be made to assemble the cylinder with God's algorithm, and the question will be answered: Is P equal to NP?

OBJECTIVES

Knowing that if a non-deterministic Turing machine finds solutions to problems in polynomial time, that implies that it is also possible for a deterministic Turing machine with God's algorithm to find solutions to those same problems in polynomial time. Also know that if it is possible quickly verify the solutions to the problems, in some cases that implies that it is also possible to find the solution with the same speed and in other cases, if it is possible, quickly verify the solutions to the problems, that does not imply that it is also possible to find the solution just as quickly. Use God's algorithm to solve the problems. Remember that the complexity of problems can reach infinity.

DEVELOPMENT

According to Nilson Bolivar (2020), it is explained how to assemble a puzzle called Pyraminx Duo with God's algorithm, also the method to assemble the Rubik's cube or other puzzles in 3 dimensions with God's algorithm is given.

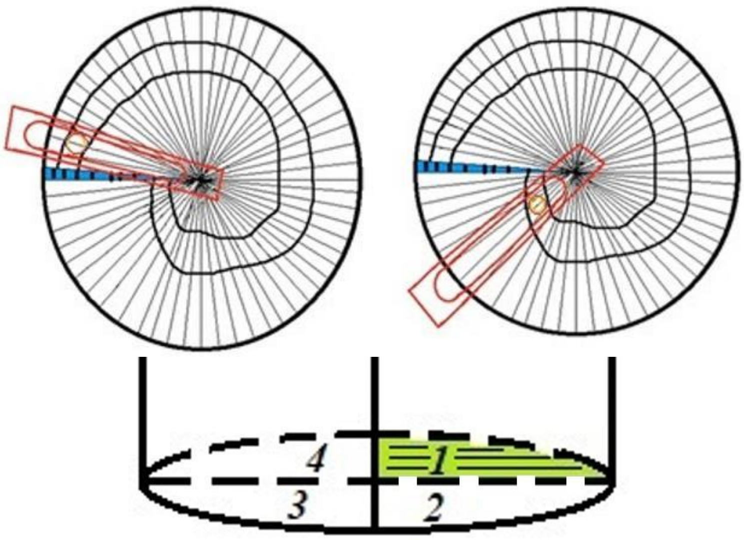
According to Elvira Mayordomo (2012) P is usually identified with the class of feasible problems, or problems that can be solved in practice. Clearly, a polynomial time limit can be huge due to both the multiplicative constant and the degree, but there are two practical reasons why we think of P as efficiently solvable problems.

According to Fortnow, Lance (2009) What is the P versus NP problem? Suppose we have a large group of students who need to pair up to work on projects. We know which students are compatible with each other and we want to put them into compatible groups of two. We could search as many pairings as possible, but even for 40 students we would have over three hundred billion trillion possible pairings.

Continuing with the article, if we have a puzzle of a cylinder that is divided in half into equal parts with the following rules: 1) that half of the number of divisions of the cylinder is an even number, 2) that each movement of the cylinder is a 180 degree turn, in other words, half turn 3) that one of the divisions of the cylinder never moves, which will be differentiated because it will be blue at the top and green at the bottom, it will have the number 1, and black lines. This part of the cylinder will never move because it is the same to move half of the cylinder with the numbers 1 and 2 than to move half of the cylinder with the numbers 3 and 4, so it is decided that the half of the cylinder with the blue color with the number 1 and the black lines will never move, no matter how many divisions the cylinder has, the blue part with the number 1 and the black lines will never move.

4) The fourth rule is that for example if the cylinder has 4 divisions the cylinder will be assembled if clockwise after the division of the cylinder that has the blue color, the number 1 and the black lines, follows the division of the cylinder with the number 2, then the division of the cylinder with the number 3, and finally the division of the cylinder with the number 4, as shown in the following image.

Figure 1



Cylinder with 4 divisions

Source: self made.

Each division of the cylinder will have a number, for example, 1, 2, 3, 4, etc. The cylinder will be assembled when, after the part of the cylinder that has the blue color, the number 1, and the black lines, the numbers of each division of the cylinder will increase by 1 in 1 clockwise, as seen in the previous image.

The total number of combinations for this cylinder in the previous image is 3 factorial = 3x2x1 = 6 combinations. Recall that half of its number of divisions is an even number; the cylinder has 4 divisions, and half of 4 is 2, which is an even number.

Then, the cylinder with half of its even number of divisions divided into 8 parts, its number of combinations is 7 factorial = 5,040.

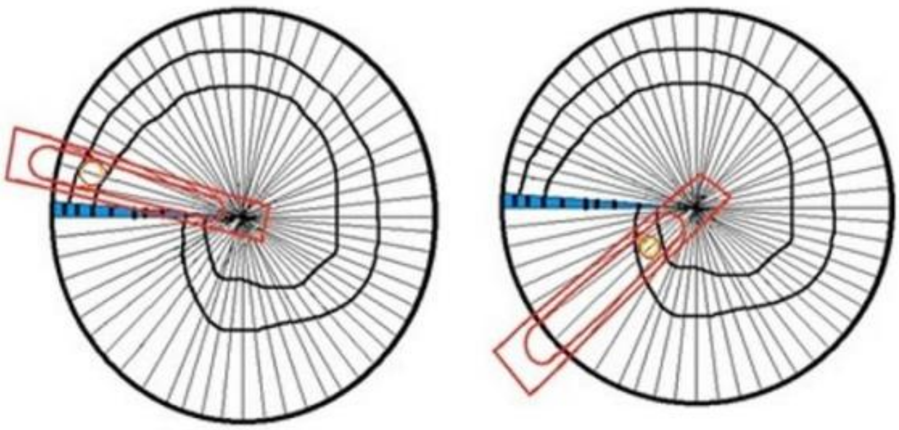
The cylinder with half its even number of divisions divided into 16, its number of combinations is 15 factorial, and so on to infinity.

The more divisions the cylinder has, the more complex the problem will become, because the cylinder will have a greater number of combinations, and because it will have more movements options. And you can divide the cylinder in half, to infinity.

Now we have to make sure that this puzzle can be checked whether it is well assembled or not in a short time, for example in about a minute. To do this, we place something like the hand of a clock that will almost go around the spiral of the cylinder in approximately one minute. Obviously this needle is going to be stuck in the center of the circle so that it can almost go around, starting clockwise on the right side of the part of the cylinder that is blue and ending on the left side of the part of the cylinder which has blue color, as shown in the following image.

Figure 2

Cylinder with many divisions

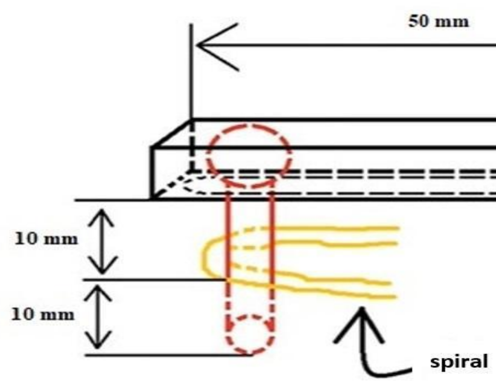


Source: self made.

In the previous image, we can also see a spiral. In the following image, we see an example of the length of what is similar to the needle of a watch, and we see that inside the needle, there is a sphere that has a welded rod that enters the spiral. As the spiral is getting closer to the center, the sphere with the welded rod is also getting closer to the center.

Figure 3

Similar to the hand of a clock

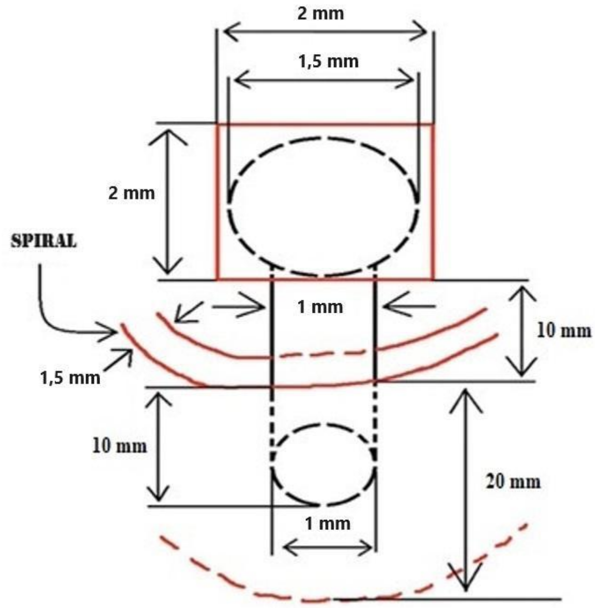


Source: self made.

In the following image, you can see that inside what is similar to the needle of the watch, there is a sphere, and the welded rod in the sphere is smaller; for example, it measures 1 mm, and the spiral measures 1.5 millimeters wide. We also observe that the welded rod will not rub against the floor of the spiral because the floor of the spiral is deeper.

Figure 4

interior view of what is similar to the hand of a clock

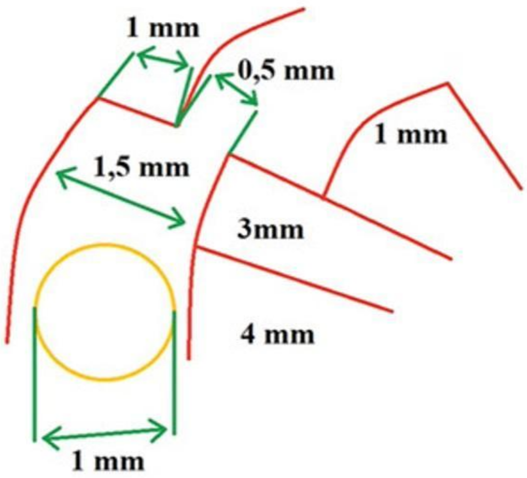


Source: self made.

In the following image we see, as we already know, that the welded rod is 1 millimeter in diameter, and the spiral is 1.5 millimeters wide. If the cylinder is properly assembled, the spiral will approach, for example, 1 millimeter to the center of the cylinder circle in each part into which the cylinder is divided. For example, if the 8-division cylinder is well assembled clockwise after the blue part of the cylinder, the spiral of the 7 parts of the cylinder approach the center of the circle by 1 millimeter in each division. For example, if you next to the blue part of the cylinder clockwise, the spiral that is on the side of the cylinder clockwise would be 7 millimeters from the center. The part of the cylinder that follows would be 6 millimeters from the center of the circle, the next part would be 5 millimeters away, the next part at 4, the next at 3, the next at 2, and the last part at 1 millimeter from the center of the circle. This happens if the cylinder is well assembled, and the welded rod would reach the end of the spiral. However, if the cylinder is badly assembled for example, if the part of the cylinder that is 3 millimeters from the center follows the part that is 1 millimeter from the center, then the welded rod cannot pass because it is 0.5 millimeters wide, as shown in the following image below.

Figure 5

Poorly assembled cylinder



Source: self made.

In the previous image, we check in more or less 1 minute that the cylinder is badly assembled, even if the cylinder has many divisions. We would check if it is well or badly assembled depending on whether the needle reaches or does not reach the end of the spiral. Then, it would be an NP problem because it can be easily checked in polynomial time.

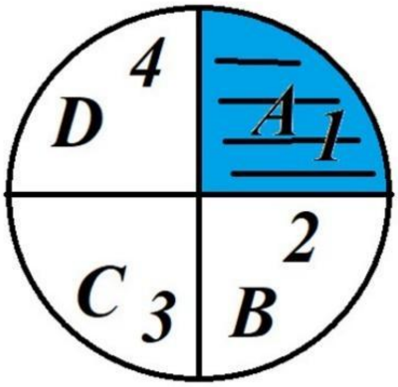
The puzzles have a God algorithm, and the cylinders in this paper also have a God algorithm. Another way to quickly check if the cylinder is well assembled or not is to place a button at the beginning of the spiral so that the welded rod when pressed turns on a red light, and at the end of the spiral place another button so that the rod welded when pressing a green light comes on.

METHOD TO ASSEMBLE THE CYLINDER WITH 4 DIVISIONS WITH THE GOD´S ALGORITHM

It seems impossible for a human being to assemble a puzzle with God's algorithm, but it is possible using this method. The cylinder is divided into 4 parts, and to each part of the cylinder, we are going to place a letter: A, B, C, D. Also, to each part of the cylinder, we are going to place a number, which are 1, 2, 3, 4, as seen in the following image below.

Figure 5

Front view of assembled cylinder

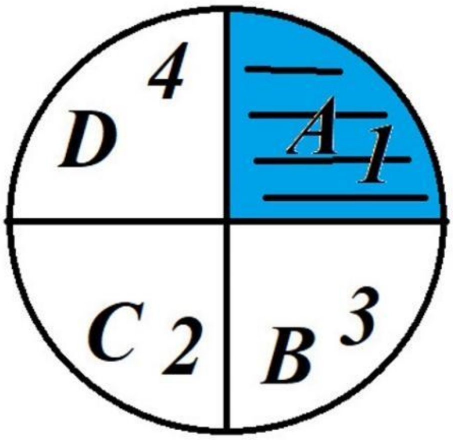


Source: self made.

The letters A, B, C, and D are like shadows, that is to say, they are not going to move. If the cylinder moves, the letters are not going to move; only the numbers are going to move. The part of the cylinder that has the number 2 also has the letter B, and the part of the cylinder that has the number 3 has the letter C. Remember that the movements of the cylinder are half turns. If we have the cylinder assembled and we move the numbers 2 and 3, the cylinder looks as seen in the following image that we have below.

Figure 6

Front view of the cylinder disassembled with a single movement



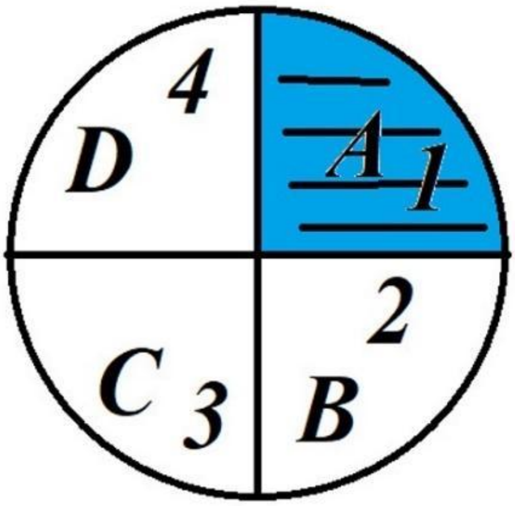
Source: self made.

In the previous image, we realize that the numbers 2 and 3 move when the cylinder moves. It is as if we were writing the numbers, and the letters B and C did not move even though the cylinder moved. It is as if the letters were shadows. Then, the letters A, B, C, and D will never move if the cylinder moves; the letters will never move, and the numbers will move when the cylinder moves, as seen in the previous image.

What we are going to do is to have the cylinder assembled as shown in the following image below.

Figure 7

Front view of assembled cylinder



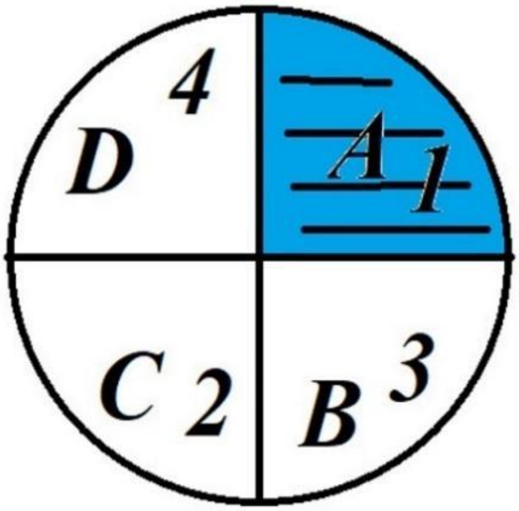
Source: self made.

Then, we make a table and place zero movement in the table. We also place part of cylinder 1, and as the part of cylinder 1 is in the letter A, we place the letter A in the table. In the table where it says 'part of cylinder 2,' we place B because the 2 is in the letter B. In the part of cylinder 3, we place the letter C because the 3 is in the letter C. In the part of cylinder 4, we put the letter D because 4 is in the letter D. Then, we put the combination in the table as follows: 1 is in A, 2 in B, 3 in C, 4 in D. The combination would be ABCD. Next, in the table, we put the words 'I return,' and as there are zero movements, we put 'assembled' because the cylinder is assembled.

Now we are going to disassemble the cylinder with a single movement, and we realize that the cylinder has 2 options of movements that are to move the two parts of the cylinder that is below and the two parts of the cylinder that is on the left side, remember that the part of the cylinder that is blue does not move, then we choose a range of movements so that the movement of the bottom part of the cylinder moves first than the left part of the cylinder, then we disassemble the cylinder with a single movement of the bottom part of the cylinder, and the cylinder is as shown in the following image below.

Figure 8

Front view of the cylinder disassembled with a single movement



Source: self made.

And in the table, in movement 1, we place half a turn down as the part of cylinder 1 is in the letter A. We place in the table the letter A for the part of cylinder 1, in the table where it says part of cylinder 2, we place C because the 2 is in the letter C. For the part of cylinder 3, we place the letter B because the 3 is in the letter B, and for the part of cylinder 4, we place the letter D because the 4 is in the letter D. Then, we place in the table "combination": as the 1 is in A, the 2 in C, the 3 in B, and the 4 in D, the combination would be ACBD; with 1 in A, 2 in C, 3 in B, and 4 in D, the combination would be ACBD.

Then, in the table, we put the word ‘I return’. As there is a movement and the cylinder is disassembled with a half turn down, it's as if we were recording a video disarming the cylinder and then we reverse the video. In the reversed video, the cylinder is disassembled and is assembled with a half turn down, so we put a half turn down. We do the same with movement one, a half turn to the left. We also do the same with movements two and three.

Remember that the total number of combinations of the cylinder is 3 factorial, which is equal to 6. As we disassemble the cylinder with 3 movements and the 3 movements gave us 6 different total combinations of the cylinder, it is no longer necessary to disassemble the cylinder with 4 movements, because the combinations of the 4 movements will be repeated with the combinations with fewer movements. The same will happen with the combinations of 5 movements, 6 movements, and so on. We will disassemble the cylinder, and the unnecessary movements are eliminated, so we will only disassemble the cylinder up to 3 movements. Up to 3 movements, there are 6 different options of combinations: ABCD, ABDC, ACBD, ACDB, ADBC, and ADCB. In the following tables, the letters (HTD) stand for half turn down, and the letters (HTL) stand for half turn left. Let's look at the table we are talking about below, which is divided into 4 parts, M 1 means movement 1, M 2 means movement 2, M 3 means movement 3.

Table 1

First process to make God algorithm table, part 1

	M 1	M 2	M 3	part of cylinder 1	part of cylinder 2	part of cylinder 3	part of cylinder 4	combination	I return
Zero movements				A	B	C	D	ABCD	assembled puzzle
	HTD			A	C	B	D	ACBD	HTD
	HTL			A	B	D	C	ABDC	HTL

Source: self made.

Table 2

First process to make God algorithm table, part 2

	HTD	HTD		A	B	C	D	ABCD	HTD	HTD
	HTD	HTL		A	D	B	C	ADBC	HTL	HTD
	HTL	HTD		A	C	D	B	ACDB	HTD	HTL
	HTL	HTL		A	B	C	D	ABCD	HTL	HTL

Source: self made.

Table 3

First process to make God algorithm table, part 3

	HTD	HTD	HTD	A	C	B	D	ACBD	HTD	HTD	HTD
	HTD	HTD	HTL	A	B	D	C	ABDC	HTL	HTD	HTD
	HTD	HTL	HTD	A	D	C	B	ADCB	HTD	HTL	HTD
	HTD	HTL	HTL	A	C	B	D	ACBD	HTL	HTL	HTD

Source: self made.

Table 4

First process to make God algorithm table, part 4

	HTL	HTD	HTD	A	B	D	C	ABDC	HTD	HTD	HTL
	HTL	HTD	HTL	A	D	C	B	ADCB	HTL	HTD	HTL
	HTL	HTL	HTD	A	C	B	D	ACBD	HTD	HTL	HTL
	HTL	HTL	HTL	A	B	D	C	ABDC	HTL	HTL	HTL

Source: self made.

Now we organize alphabetically the combinations of the table, and we only copy the parts of the table where the combinations and the movements that are returned. Then, where it says "I return," we replace it with the phrase "is assembled with." We place a new column in the table called "combination number," and there we count the number of combinations of the cylinder.

The combination ABDC, assembled with 1 movement that is a half turn left, is also assembled with 3 movements: half turn left, half turn down, half turn down. It is also assembled with 3 other movements: half turn down, half turn down, half turn left. So, God's algorithm looks for the least number of possible movements, the perfect or optimal number of movements to assemble the cylinder. Then, the combination can be assembled with a movement that is a half turn left, or with 3 movements, and God's algorithm chooses 1 movement. Then, for the 3 movements, we place "eliminated," and we do the same with all the combinations of the following table. In the combination ADCB, there are 2 options of assembling with 3 movements: half turn down, half turn left, half turn down, and another option with 3 movements: half turn left, half turn down, half turn left. As the two options are 3 movements, it is the same number of movements that this combination has. This means that God's algorithm can be assembled in 2 possible ways with 3 movements. God's algorithm is not unique; it seeks to assemble the combination with the least number of movements possible. If there are several options with the same number of movements to be assembled, then any of these options can be used.

In the following tables, the letters (HTD) stand for half turn down, and the letters (HTL) stand for half turn left. We are listing the combinations that are not eliminated, and it has to give us 6 combinations, which is equal to the number of total combinations that the cylinder has: 3 factorial, equal to 6. Everything that has been explained is observed in the following table, which is divided into 2 parts.

Table 5

Second process to make God algorithm table, part 1 of the table.

Combination	is assembled with			Combination number
ABCD	assembled puzzle			1
ABCD	HTD	HTD		Eliminated
ABCD	HTL	HTL		Eliminated
ABDC	HTL			2
ABDC	HTL	HTL	HTL	Eliminated
ABDC	HTL	HTD	HTD	Eliminated
ABDC	HTD	HTD	HTL	Eliminated

Source: self made.

Table 6

Second process to make God algorithm table, part 2 of the table

ACBD	HTD			3
ACBD	HTD	HTD	HTD	Eliminated
ACBD	HTD	HTL	HTL	Eliminated
ACBD	HTL	HTL	HTD	Eliminated
ACDB	HTD	HTL		4
ADBC	HTL	HTD		5
ADCB	HTD	HTL	HTD	6
ADCB	HTL	HTD	HTL	6

Source: self made.

Now we remove the eliminated combinations from the previous table, and the table will be called God's Algorithm. We replace 'assembled' with 'assembled with God's Algorithm,' and it appears as follows.

Table 7

Third and final process to make God's algorithm table

God's Algorithm				
Combinations	**It is assembled with God's algorithm**			**Combination number**
ABCD	assembled puzzle			1
ABDC	HTL			2
ACBD	HTD			3
ACDB	HTD	HTL		4
ADBC	HTL	HTD		5
ADCB	HTD	HTL	HTD	6
ADCB	HTL	HTD	HTL	6

Source: self made.

The previous table is arranged in alphabetical order for easier combinations search. The more intelligent we are and the more we strive, the better we will become at solving more complex problems. In other words, intelligence and effort are directly proportional to becoming better at solving more complex problems. It can also be said that the more intelligence and effort we apply, the better algorithms we will create with the fewest possible steps to solve that problem. However, let's remember that God's Algorithm represents the limit, and there is no algorithm that can solve the problem in fewer steps than God's Algorithm. This is why a problem of type P may have a shortcut for solving it in polynomial time, but if it's very complex, we might not have the necessary intelligence to solve it. We might either solve it with an algorithm that requires a significant number of steps or with an algorithm in non-polynomial time.

WE CAN MAKE GOD´S ALGORITHM FOR THE CYLINDER DIVIDED IN 4 PARTS IN ANOTHER WAY, WITH INTELLIGENCE

For example, if the cylinder is assembled, it is obvious that if we repeat the same movement 2 times, such as half turn down, half turn down, the cylinder is going to return to zero movements. Then, we remove the combinations that have the movements that are repeated in a row 2 times or more. Thus, we make the table of God's algorithm faster. However, in a cylinder with more divisions or more complexity, the obvious can become more and more complex, to infinity, to the point where we might not have the sufficient intelligence to realize it. For that reason, the table of God's algorithm is created step by step in this investigation, in case you don't realize the obvious.

In the following table below, we observe the number of combinations with respect to the number of movements of the cylinder with 4 divisions, which has to give 6 total combinations.

Table 8

Incomplete table of god's algorithm, cylinder with 4 divisions

Number of movements to arm with God's algorithm	0	1	2	3	Total combinations
Number of combinations with respect to the number of moves to assemble with God's algorithm of the cylinder divided into 4 parts	1	2	2	1	6

Source: self made.

The above table is an incomplete table of God's algorithm of the cylinder divided into 4 parts because it tells how many combinations it has with respect to the number of movements to assemble with God's algorithm. The same happens with other puzzles, like, for example, the Rubik's Cube. On Wikipedia it says that the maximum number of face turns needed to assemble any combination of the Rubik's cube is 20 moves, I'm not sure if it is with God's algorithm, but assuming it is with God's algorithm the incomplete table of the algorithm of God would be as shown below. The table is divided in 2.

Let's see the first part of the table below.

Table 9

Incomplete table of god's algorithm, Rubik's cube

Number of movements to arm with God's algorithm	Number of combinations with respect to the number of movements to be assembled with God's algorithm
0	1
1	18
2	243
3	3,240
4	43,239
5	574,908
6	7,618,438
7	100,803,036
8	1,332,343,288
9	17,596,479,795
10	232,248,063,316

Source: self made.

Now let's look at the second part of the table below.

Table 10

Incomplete table of god's algorithm, Rubik's cube

11	3,063,288,809,012
12	40,374,425,656,248
13	531,653,418,284,628
14	6,989,320,578,825,358
15	91,365,146,187,124,313
16	about 1,100,000,000,000,000,000
17	about 12,000,000,000,000,000,000
18	about 29,000,000,000,000,000,000
19	about 1,500,000,000,000,000,000

20	about 490,000,000

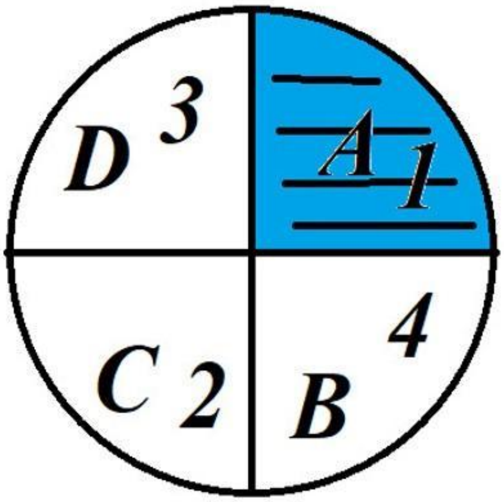
Source: own elaboration and modified with data from Davidson, Morley; Dethridge, John; Kociemba, Herbert; Rokicki, Tomas. «God's Number is 20»

Exercise number 1

Everything we did above is to assemble the cylinder with God's algorithm, let's assume we have the cylinder disassembled as shown in the following image below.

Figure 8

Front view of disassembled cylinder with 2 movements



Source: self made.

In the previous image, we observe that the combination of the cylinder is ACDB. In the following table, the letters (HTD) stand for half turn down, and the letters (HTL) stand for half turn left. So, we search for the combination in the God's Algorithm table, and it will be easy to find because the table has the combinations in alphabetical order.

Table 11

Third and final process to make God's algorithm table

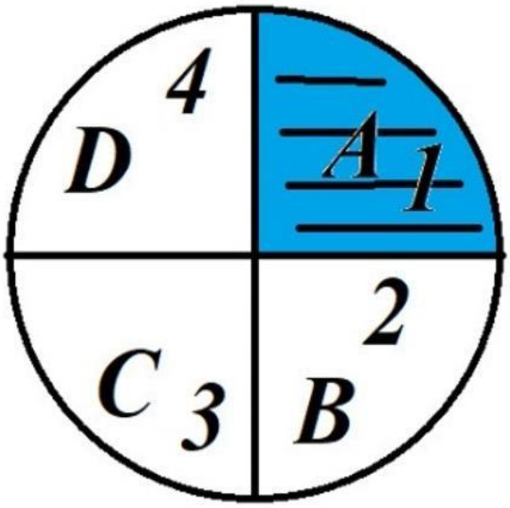
God's Algorithm				
combinations	**It is assembled with God's algorithm**			**Combination number**
ABCD	assembled puzzle			1
ABDC	HTL			2
ACBD	HTD			3
ACDB	HTD	HTL		4
ADBC	HTL	HTD		5
ADCB	HTD	HTL	HTD	6
ADCB	HTL	HTD	HTL	6

Source: self made.

We observe that in the table the combination ACDB is assembled with God's algorithm with a half turn down and a half turn left and we assemble the cylinder with God's algorithm as shown in the following image.

Graphic 9

Front view of assembled cylinder



Source: self made.

There is no algorithm that solves the problem in fewer steps than God's algorithm; that algorithm is impossible. For example, the ADCB combination is assembled with God's algorithm, with 2 options of 3 moves, and it is impossible to assemble it with fewer than 3 moves.

With this method, we can assemble puzzles in 3 dimensions with God's algorithm, such as the Rubik's Cube, the Pyraminx Duo, Rubik's Cube 4x4, etc. No matter how complex the puzzle may seem, it can be solved using the God's Algorithm with the same method we use to solve the cylinder divided into 4 divisions. So, if the cylinder is divided in half to infinity; this method serves to assemble it with God's algorithm.

There is no algorithm that solves the problem in fewer steps than God's algorithm; that algorithm is impossible. Continuing with the article, all the puzzles can be put together with God's algorithm. But let's imagine that we are putting together a puzzle, for example, a Rubik's Cube. With God's algorithm, 20 moves are the optimal solution to put together that puzzle. Then, we put it together with 20 moves and record a video with a camera putting it together with 20 moves. Then, we watch the video in reverse, and obviously, in the video, we see that the puzzle is assembled. Afterward, we disassemble it with 20 moves. Then, we realize that the puzzle can be assembled and disassembled with the number of moves of God's algorithm, which in this case would be 20 moves.

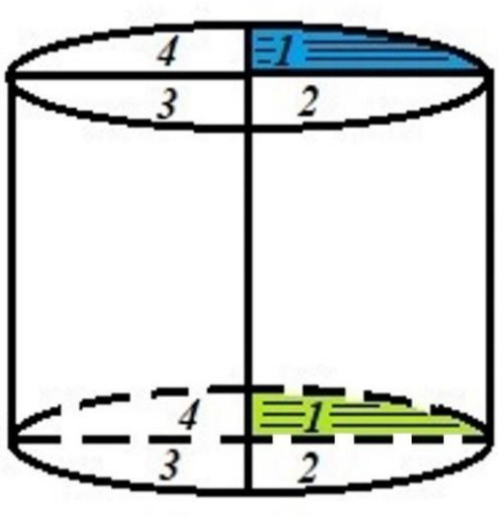
Now, if the cylinder has 4 divisions, we disassemble it with God's algorithm with 1 movement, to then assemble it with God's algorithm with 1 movement. If the cylinder has 8 divisions, we disassemble it with God's algorithm with 2 movements, to then assemble it with God's algorithm with 2 movements. If the cylinder has 12 divisions, we disassemble it with God's algorithm with 3 movements, to then assemble it with God's algorithm with 3 movements, and we follow this sequence to infinity.

The sequence is that the number of divisions of the cylinder increases by 4, and the number of movements with which the cylinder will be assembled with God's algorithm increases by 1. In other words, the number of movements with which we are going to assemble the cylinder with God's algorithm is one-fourth of the number of divisions of the cylinder.

The cylinder divided into 4 parts has 3 factorial of total combinations, and it is going to be disassembled with God's algorithm with 1 movement. Then, you can move parts 2 and 3 of the cylinder and assemble it with God's algorithm, or you can disassemble it by moving parts 3 and 4 of the cylinder and then assemble it with God's algorithm. As there are 2 options of movements to move the cylinder, we make a formula: 2, which is the number of options of movements, raised to the power of the number of movements with which it will be assembled. Since it will be assembled with 1 movement (because 1 is the fourth part of 4), then it would be 2 raised to the power of 1, which is equal to 2. This is the number of possibilities of combinations with which the cylinder will be assembled using God's algorithm.

Figure 10

Cylinder with 4 divisions

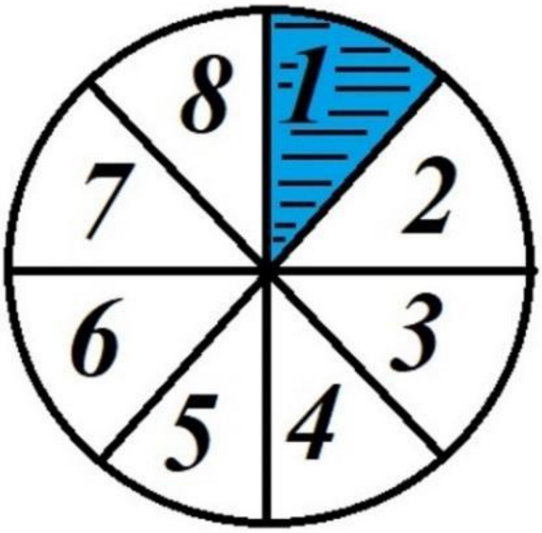


Source: self made.

Following this sequence, the cylinder divided into 8 parts has 7 factorial of total combinations and has 4 options of movements. In the first option of movement, you can move the part of the cylinder with the numbers 2, 3, 4, and 5. In the second option, the numbers 3, 4, 5, and 6 are moved. In the third option, the numbers 4, 5, 6, and 7 are moved. And in the fourth option, the numbers 5, 6, 7, and 8 are moved. In the formula, 4 is placed because it has 4 options of movements, and it is going to be assembled with 2 movements because 2 is the fourth part of 8. The formula would be 4 raised to 2 = 16, which is the number of possibilities of combinations with which the cylinder will be assembled using the God's Algorithm. Let's look at the cylinder divided into 8 parts in the following image below.

Figure 11

Front view of the cylinder armed with 8 divisions



Source: self made.

The cylinder, divided into 12 parts, has 11 total combinations factorial, and it has 6 options of movements. In the formula, 6 is placed, and it will be assembled with 3 movements because 3 is the fourth part of 12. The formula would be 6 raised to the power of 3 = 216, which is the number of possibilities of combinations with which the cylinder will be assembled with God's algorithm, and so on, to infinity.

Then, we realize that the number of total combinations of the cylinder is greater than the number of possibilities of combinations with which the cylinder will be assembled with God's algorithm.

Then, we take the percentage of the number of possibilities of combinations to assemble the cylinder divided by the number of total combinations of the cylinder. For example, a cylinder divided into 4 parts has a total number of combinations of 3 factorial, which is equal to 6, and the number of possibilities of combinations to assemble the cylinder with the algorithm of God is 2. So, 6, which is the number of total combinations of the cylinder, is 100%. We want to know what percentage the 2 represents out of 6. To find this, we divide 2 by 6, which gives us 0.33. When multiplied by 100, it becomes 33%. In the following table, we observe that the percentage for the cylinder divided into 8 is 0.317%. We can see that the percentage decreases as the cylinder has more divisions. This trend continues to lower the percentage toward infinity. This implies that we can assemble the cylinder with God's algorithm using the fourth part of the number by which the cylinder is divided. This is because the number of combinations to assemble the cylinder with God's algorithm will be less than the total number of combinations by which the cylinder is divided.

The number of combinations with which the cylinder is going to be assembled with God's algorithm is a gross unit, because God's algorithm is going to eliminate unnecessary combinations of unnecessary movements, this means that the number of combinations with which it will be assembled with God's algorithm the cylinder is a number smaller than the one in the following table, and the percentage is also smaller than the one in the following table, but I am not doing this work to summarize this article as much as possible and this has already been explained as do it in this same article when we made the table of God's algorithm, the important thing is to know that the percentage is going down. The table is divided into 2 parts, let's see the first part below.

Table 12

Percentage: of the number of possible combinations to assemble the cylinder with God's algorithm divided by the number of total combinations of the cylinder with half of its even divisions

Row number	Percentage: of the number of possible combinations to assemble the cylinder with God's algorithm divided by the number of total combinations of the cylinder with half of its even divisions	Its number of combination possibilities to assemble the cylinder with God's algorithm with half of its even divisions	Total number of combinations of the cylinder with half of its even divisions.	Number of divisions of the cylinder with half of its even divisions
1	33,33333333 %	2	6	4

Table 13

Now let's look at the second part of the table below

2	0,317460317 %	16	5.040	8
3	5,41 x 10^{-4} %	216	39.916.800	12
4	3,13 x 10^{-7} %	4.096	1,31 x 10^{12}	16
5	8,22 x 10^{-11} %	100.000	1,22 x 10^{17}	20
6	1,16 x 10^{-14} %	2.985.984	2,59 x 10^{22}	24
7	9,68 x 10^{-19} %	105.413.504	1,09 x 10^{28}	28
8	5,22 x 10^{-23} %	4.294.967.296	8,22 x 10^{33}	32
9	1,92 x 10^{-27} %	198.359.290.368	1,03 x 10^{40}	36
10	5,02 x 10^{-32} %	10.240.000.000.000	2,04 x 10^{46}	40

Source: self made.

ANSWERING THE MILLENIUM QUESTION: IS P EQUAL TO NP?

When we ask: Is P equal to NP? In reality, these 2 questions are being asked, which are the following.

Question number 1, if a non-deterministic Turing machine finds solutions to problems in polynomial time, does that imply that it is also possible for a deterministic Turing machine to find solutions to those same problems in polynomial time?

Answer, if it is possible for a deterministic Turing machine to find solutions to those same problems in polynomial time, because the god algorithm or the optimal algorithm can be used to solve these same problems, for example if we limit the number of moves to assemble with the God's algorithm the cylinder for example 20 movements and if the cylinder has 80 divisions, from 80 divisions forward the cylinder will be assembled with 20 movements with God's algorithm, it does not matter if the cylinder has 1 million, 10 million, 1 trillion and infinities of divisions, the cylinder will always be assembled with 20 movements with God's algorithm and it would only take 20 movements or 20 seconds to assemble the cylinder, regardless of whether the cylinder becomes infinitely more complex, it will always be assembled with 20 movements. Let's assume that 1 movement is equal to 1 second.

In other words, if a non-deterministic Turing machine finds solutions to problems in polynomial time, a deterministic Turing machine will also find solutions to the same problems in polynomial time with the optimal algorithm or with the god algorithm.

Question number 2, if it is possible to quickly "verify" solutions to problems, does that imply that it is also possible to "find" the answers just as quickly?

Answer, in some cases it will be possible to verify the solutions to the problems in polynomial time because what is similar to the hand of a clock will almost turn around in a minute or so, but it will not be possible to find the solutions to the problems in polynomial time because for example if the cylinder has 400 million divisions it must be assembled with 100 million movements with God's algorithm. Remember that it is a quarter of the number of total divisions of the cylinder, that is why it will take 100 million seconds to assemble it because let's assume that a movement is equal to a second and the more divisions the cylinder has, more time it will take to assemble it with God's algorithm. Let's remember that the more divisions the cylinder has, the problem becomes more complex because the cylinder has more options. movements, let us also remember that there is no algorithm that solves the problem in fewer steps than God's algorithm, that algorithm is impossible. It does not matter if the Turing machine is deterministic or non-deterministic, a non-deterministic Turing machine will not be able to solve the problem in polynomial time because it will take 100 million seconds, assuming that 1 second is equal to 1 movement.

In other cases it will be possible to verify the solutions to the problems in polynomial time, and it will also be possible to find the solutions to the problems in polynomial time because for example if we limit the number of movements to assemble the cylinder with God's algorithm, for example 20 movements and if the cylinder has 80 divisions, from 80 divisions forward the cylinder will be assembled with 20 movements with God's algorithm, it does not matter if the cylinder has 1 million, 10 million, 1 billion and infinity of divisions, the cylinder will always be assembled with 20 movements with God's algorithm and it would only take 20 movements or 20 seconds to assemble the cylinder, regardless of whether the cylinder becomes infinitely more complex, it will always be assembled with 20 movements.

The more divisions the cylinder has, the more complex the algorithm will become because the cylinder will have more possibilities for movements. Let us remember that the cylinder will be assembled with 20 movements with God's algorithm and will be constant and in Big O notation, it is represented like this O(1), the cylinder can have infinite divisions, the cylinder will always be assembled with 20 movements with God's algorithm.

Answering these 2 questions we also answered: Is P equal to NP? And we also answered: Is P equal to NP-complete?

CONCLUSIONS

The results of this research support the idea that the complexities of the problems increase to infinity because the more divisions the cylinder has, the more complex it will become, if we ask: Is P equal to NP? As explained above, we are actually asking 2 questions, the answers to these 2 questions are: If a non-deterministic Turing machine finds solutions to problems in polynomial time, that implies that it is also possible for a deterministic Turing machine to find solutions to those same problems in polynomial time with God's algorithm, by answering the other question it is possible to quickly verify the solutions of the problems, but in some cases this implies that it is also possible to find the answers in polynomial time with God's algorithm, but in In other cases, the solutions to the problems are quickly checked, but it is not possible to find the answers in polynomial time with God's algorithm. There is no algorithm that solves the problem in fewer steps than God's algorithm; that algorithm is impossible.

REFERENCES

Arvind, Vikraman; Kurur, Piyush P. (2006). "Graph isomorphism is in SPP". Information and Computation. 204 (5):835–852 https://www.sciencedirect.com/science/article/pii/S089054010600023X?via%3Dihub

Aviezri Fraenkel and D. Lichtenstein (1981). "Computing a perfect strategy for n × n chess requires time exponential in n". Journal of Combinatorial Theory. Series A. 31 (2): 199–214. https://www.sciencedirect.com/science/article/pii/0097316581900169?via%3Dihub

Colbourn, Charles J. (1984). "The complexity of completing partial Latin squares". Discrete Applied Mathematics. 8 (1): 25–30 https://www.sciencedirect.com/science/article/pii/0166218X84900751?via%3Dihub

Bolívar Nilson, Armar el Pyraminx Duo con el algoritmo de Dios. (Assemble the Pyraminx Duo with God's algorithm.)

Le enseñaremos cómo armar el cubo de Rubik y muchos otros rompecabezas en 3 dimensiones con el algoritmo de Dios. (We will teach you how to put together the Rubik's cube and many other 3-dimensional puzzles with God's algorithm.) https://docs.google.com/document/d/10EgTFh2v-dWUFlGkP4GEldRg1ajbNIyg/edit?usp=sharing&ouid=115495646788988650320&rtpof=true&sd=true

Bolívar Nilson, ¿Pes igual a NP? Respuesta del milenio. Método para resolver rompecabezas con el algorimo de Dios. Ciencia Latina Revista Científica Multidisciplinar Julio-Agosto, 2023, Volumen 7, Número 4. 6028-6053 https://ciencialatina.org/index.php/cienciala/article/view/7393/11156

Cook, Stephen (1971). "The complexity of theorem proving procedures". Proceedings of the Third Annual ACM Symposium on Theory of Computing. pp. 151–158

https://dl.acm.org/doi/10.1145/800157.805047

Davidson, Morley; Dethridge, John; Kociemba, Herbert; Rokicki, Tomas. «God's Number is 20» https://www.cube20.org/

Elvira Mayordomo. "P versus NP" Archived 16 February 2012 at the Wayback Machine Monografías de la Real Academia de Ciencias de Zaragoza(Monographs of the Zaragoza Royal Academy of Sciences) 26: 57–68 (2004). https://web.archive.org/web/20120216154228/http://www.unizar.es/acz/05Publicaciones/Monografias/MonografiasPublicadas/Monografia26/057Mayordomo.pdf

Fischer, Michael J.; Rabin, Michael O. (1974). "Super-Exponential Complexity of Presburger Arithmetic". Proceedings of the SIAM-AMS Symposium in Applied Mathematics. 7: 27–41. Archived from the original on 15 September 2006. Retrieved 15 October 2017.

https://web.archive.org/web/20060915010325/http://www.lcs.mit.edu/publications/pubs/ps/MIT-LCS-TM-043.ps

Fortnow, Lance (2009). "The status of the P versus NP problem" (PDF). Communications of the ACM. 52 (9): 78–86. CiteSeerX 10.1.1.156.767. doi:10.1145/1562164.1562186. S2CID 5969255. Archived from the original (PDF) on 24 February 2011. Retrieved 26 January 2010.

https://dl.acm.org/doi/10.1145/1562164.1562186

Hartmanis, Juris. "Gödel, von Neumann, and the P = NP problem" (PDF). Bulletin of the European Association for Theoretical Computer Science. 38: 101–107.

https://ecommons.cornell.edu/server/api/core/bitstreams/46aef9c4-288b-457d-ab3e-bb6cb1a4b88e/content

Johnson, David S. (1987). "The NP-completeness column: An ongoing guide (edition 19)". Journal of Algorithms. 8 (2): 285–303.

https://www.sciencedirect.com/science/article/abs/pii/0196677487900435?via%3Dihub

Kawarabayashi, Ken-ichi; Kobayashi, Yusuke; Reed, Bruce (2012). "The disjoint paths problem in quadratic time". Journal of Combinatorial Theory. Series B. 102 (2): 424–435.

https://www.sciencedirect.com/science/article/pii/S0095895611000712?via%3Dihub

Ladner, R.E. (1975). "On the structure of polynomial time reducibility". Journal of the ACM. 22: 151–171 See Corollary 1.1. https://dl.acm.org/doi/10.1145/321864.321877

Sipser, Michael: Introduction to the Theory of Computation, Second Edition, International Edition, page 270. Thomson Course Technology, 2006. Definition 7.19 and Theorem 7.20

Valiant, Leslie G. (1979). "The complexity of enumeration and reliability problems". SIAM Journal on Computing. 8 (3): 410–421. https://epubs.siam.org/doi/10.1137/0208032

William I. Gasarch (junio de 20e02). «The P=?NP poll.». SIGACT News 33 (2): 34-47 https://www.cs.umd.edu/~gasarch/papers/poll.pdf