A Method of Proving P≠NP

Chao HAN

Boyu Training School,Mengzhong Hospital Alley,Donghe District,Baotou City,Inner Mongolia,China

2411358301@qq.com

Keywords:P problem,NP problem,algorithm,polynomial time,Sudoku grid,time complexity

Abstract: In 1971,Stephen Arthur Cook,an American-Canadian computer scientist,put forward a famous problem:P=NP or not?[1]In the past 50 years,large numbers of mathematicians from different countries have been sparing no effort to solve this problem,but they have never been succeed so far.Now,this article tries to prove that P≠NP by proving no polynomial-time algorithm of someone NP problem.

1. Introduction

As we know,a P problem can be solved in polynomial time and an NP problem can be verified the correctness of an answer of its in polynomial time,we can say this problem belongs to"NP".[2]Obviously,verifying the correctness of a problem's answer is no harder than solving this problem.So,P belongs to NP.However,NP belongs to P or not,nobody has proved so far.

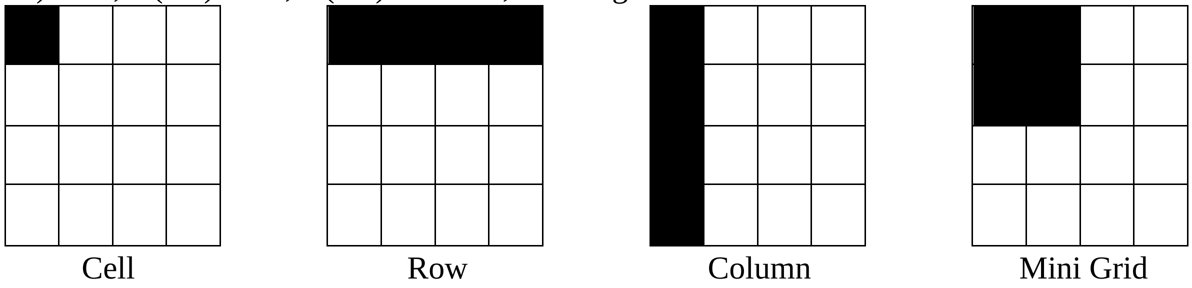
In fact,if we want to solve this difficult problem,we have two methods.

One method is that we can prove someone NP-complete problem can be solved in polynomial time,then we can get the conclusion that P=NP(Be based on the definition of NP-complete problem,we can get that each NP-complete problem is a problem which any other NP problem can be reduced into it in polynomial time.[1]So if we find a polynomial-time algorithm of someone NP-complete problem,we will prove that each NP problem can be solved in polynomial time).The other method is that we can prove someone NP problem does not exist polynomial-time algorithm,then we can get the conclusion that P≠NP.

Certainly,like most people,I hold the idea that P≠NP.So,I will prove it by proving no polynomial-time algorithm of an NP problem.

2.Manuscript Preparation

Let us to put forward such a problem,we set it as problem p:If we randomly generate a Sudoku grid(It has n cells,n^(1/2) rows,n^(1/2) columns,n^(1/2) mini grids in total,and each mini gird has n^(1/2) cells,n^(1/4) rows,n^(1/4) columns,as the figure shows:



(The definition of cell,row,column and mini grid in Sudoku,take a 4×4 Sudoku grid as an example)

where n^(1/4)∈N*.And the rules of it are in the following.At first,the number in each cell must be selected from 1 to n^(1/2).At second,

everyone of these n^(1/2) numbers must appear once in each row,each column and each mini grid.At third,the same number cannot appear twice in the same row,the same column and the same mini grid).Firstly,let us to remove all of these numbers.Secondly,we look for a person who don't know any one clue.At last,let this person to refill these numbers in this Sudoku grid again.It is an important that we can only tell this person that these filled numbers are all right or not when this person has finished filling all of them each time,

instead of making this person see the correct answer(For example,we can design a following program:As soon as someone finishes filling all of the numbers of an n^(1/2)×n^(1/2) Sudoku grid,these numbers will turn blue if they are all right or these numbers will turn red if they are not all right.So,if we randomly generate a 4×4 Sudoku grid and all of the numbers in this Sudoku grid are as shown in the following figure:

1	2	3	4
3	4	2	1
2	1	4	3
4	3	1	2

Let us to remove all of these numbers,then we look for someone to refill them who don't know any clue.If the filled numbers by him or her are as follow,these numbers will turn red:

1	4	3	2
2	3	4	1
4	2	1	3
3	1	2	4

If the filled numbers by him or her are as follow,these number will turn blue:

1	2	3	4
3	4	2	1
2	1	4	3
4	3	1	2

Attention:The grogram cannot prompt any one right number,if some filled numbers are wrong)

So,if we want to prove P≠NP by proving problem p∉P,we must have the following proof steps.

3.Proof

Step 1.Prove it is an NP problem.

We can easily know that we only need to operate n times to verify the correctness of these filled numbers one by one.In other words,the T(n) of verifying them is O(n) if we use T(n) to represent"time complexity of someone algorithm".So,problem p can be verfied in polynomial time,it is an NP problem.

Step 2.Prove it has no polynomial-time algorithm.

Let us to set m=n^(1/4),we can know that m∈N*.There are the following two cases.

1.If m≥2,we can get that there is a positive integer a which makes that given numbers are no less than a if we want an m^2×m^2 Sudoku grid is solvable and has only one solution.For example,when m=3,an m^2×m^2 Sudoku grid is a 9×9 Sudoku grid,as we know that its given numbers are no less than 17 if it has unique solution.[3]Besides,at this time,we can also know that an m^2×m^2 Sudoku grid has no less than two solutions if we obey the above three rules of Sudoku and its given numbers are less than a(Some people think that an m^2×m^2 Sudoku grid may has no solution in some cases if given numbers are less than a.Indeed,like this 9×9 Sudoku grid,

1	2	3						
			4					
						4		

its given numbers are 5.We can easily know this 9×9 Sudoku grid is unsolvable because there is no cell for filling "4" in the upper left mini grid.However,it does not meet the second rule of Sudoku).So when m≥2 and given numbers are 0,an m^2×m^2 Sudoku grid must have at least two solutions under obeying the rules of Sudoku.

2.If m=1,an m^2×m^2 Sudoku grid is an 1×1 Sudoku grid.Obviously,we can easily know that the number of 1×1 Sudoku grids is 1.So,at this time,we need just 0 given numbers that can make this Sudoku grid has only one solution.

In fact,due to there is no any clue,If a person want to make sure that each filled number is correct in a randomly generated and all of numbers removed m^2×m^2 Sudoku grid,he or she has to enumerate all of possibility.So this person must randomly generate everyone completeness of all distinct m^2×m^2 Sudoku grids and verify whether these filled numbers are correct.

Now,we set the number of m^2×m^2 Sudoku grids as b.When m≥2,

though most of the time,the number of essentially different m^2×m^2 Sudoku grids are extremely possible much smaller than the number of m^2×m^2 Sudoku grids(For example,the number of essentially different 9×9 Sudoku grids is 5,472,730,538.However,the number of 9×9 Sudoku grids is 6,670,903,752,021,072,936,960),[4]the reason is that large numbers of different Sudoku grids are regarded as the same one which each of them can be transformed into another by some symmetry including 90°rotational symmetry to the left,90°rotational symmetry to the right,180°rotational symmetry,

reflection symmetry on one or two orthogonal axes,reflection symmetry on one or two diagonal axes and dihedral symmetry according to the Burnside's lemma of symmetric groups.[5][6]In fact,the above operations can affect the correctness of the filled numbers the reason is that rotational symmetry transformation or axial reflection transformation or dihedral symmetry transformation may change the position of some numbers.For example,if we randomly generate a 4×4 Sudoku grid and all of the numbers in it are as shown in the following figure:

1	2	3	4
3	4	2	1
2	1	4	3
4	3	1	2

However,if we rotate it 90°to the right,as shown in the second figure:

4	2	3	1

3	1	4	2
1	4	2	3
2	3	1	4

The numbers in some grids are different from the original,so it means that some filled numbers are wrong.

So,if we can prove that randomly generate b m^2×m^2 Sudoku grids cannot be finished in polynomial time,we will prove problem p∉P.In fact,if we want to solve any one problem by any one algorithm,we need to operate at least one time.So,whichever algorithm we employ,randomly generating b m^2×m^2 Sudoku grids need to operate at least b times.In other words,we need at least operate b times for solving problem p.So,as long as we can prove that the value of b grows faster than any one polynomial when m→+∞,we will prove that problem p cannot be solved in polynomial time.Due to m=n^(1/4) or n=m^4,so when m→+∞,n tends to +∞ too,so we can also say that we will prove problem p does not exist polynomial-time algorithm if we can prove that the value of b grows faster than any one polynomial when n→+∞.

In July 2013,Siˆan K.Jones,Stephanie Perkins and Paul A.Roach,three British mathematicians,had proved that the number of R^2×C^2 Sudoku grids is (RC-1)!×(R!)^C×(C!)^R times of the number of reduced R^2×C^2 Sudoku grids(R∈N*,C∈N*).[7]So when R=C,we can get that R=C=m and the number of m^2×m^2 Sudoku grids is (m^2-1)!×(m!)^(2m) times of the number of reduced m^2×m^2 Sudoku grids.So,the number of m^2×m^2 Sudoku grids can be divisible by (m^2-1)!×(m!)^(2m) and the number of reduced m^2×m^2 Sudoku grids.We can get the conclusion that the number of m^2×m^2 Sudoku grids is no less than (m^2-1)!×(m!)^(2m).At the same time,we can also get that (m^2-1)!∈N*,(m!)ˆ(2m)∈N* according to m∈N*.So,m!≥m and

(m^2-1)!(m!)^(2m)≥(m!)^(2m)≥m^(2m).Be based on above conclusion,

we can know that the number of m^2×m^2 Sudoku grids is no less than (m)^(2m).Besides,because we must need at least one step to randomly generate an m^2×m^2 Sudoku grid,randomly generate (m)^(2m) m^2×m^2 Sudoku grids must operate (m)^(2m) times at least.Then we can get that solving problem p must operate (m)^(2m) times at least.

Due to m=n^(1/4),we can get that m^(2m)=[n^(1/4)]^[2n^(1/4)]=n^[(1/2)n^(1/4)] and we can also say that randomly generate n^[(1/2)n^(1/4)] n^(1/2)×n^(1/2) Sudoku grids must operate n^[(1/2)n^(1/4)] times at least.In other words,solve problem p must operate n^[(1/2)n^(1/4)] times at least.So,the T(n) of problem p is no less than O{n^[(1/2)n^(1/4)]}.Then we can get that problem p does not exist polynomial-time algorithm,because it is not a polynomial-time algorithm if an algorithm with time complexity O{n^[(1/2)n^(1/4)]}(If an algorithm is a polynomial-time algorithm,its T(n)≤O(n^k),where k is a finite positive number.[2]So,"k",the exponent of n^k,is a finite positive number when n→+∞.However,"[(1/2)n^(1/4)]",the exponent of n^[(1/2)n^(1/4)],also tends to +∞ when n→+∞.So,as long as n is large enough,n^[(1/2)n^(1/4)] will grow faster than any one polynomial.So,if T(n) of an algorithm is O{n^[(1/2)n^(1/4)]},the algorithm must be a superpolynomial-time algorithm).

4. Conclusion

According to the saying all above,we can get the conclusion that P≠NP.

References

[1]Stephen Arthur Cook.The complexity of theorem proving procedures,Proceedings of the Third Annual ACM Symposium on Theory of Computing,1971, pp.151-158.

[2]Michael Sipser.Introduction to the Theory of Computation,Course Technology Publishing

Company, 2006, pp.256-258.

[3]Gary McGuire,Bastian Tugemann,Gilles Civario,There is no 16-Clue Sudoku:Solving the Sudoku Minimum Number of Clues Problem via Hitting Set Enumeration,[online] Available: http://www.math.ie/.

[4] Ed Russell,Frazer Jarvis,Mathematics of Sudoku,[online] Available: http://www.afjarvis.staff.shef.ac.uk/.

[5]Joseph Louis Rotman,An introduction to the theory of groups,Springer Publishing House,1995, pp.46-50.

[6]Gordon Royle,Combinatorial Concepts With Sudoku I:Symmetry,[online] Available:

http://ko.c.Wong.tripod.com/sudoko/study/sudoku-symmetry.pdf.

[7]Siˆan K.Jones.Stephanie Perkins.Paul A.Roach,On the number of Sudoku Grids.Journal of Combinatorial Mathematics and Combinatorial Computing, no.5, pp.94-95, 2014.