the general science journal

Two criteria and one property from the algorithms and their application to show that NP-complete is in P.

Rodolfo A. Nieves Rivas
fesol7luzley@gmail.com

Abstract:

In this article we present two criteria and a property of the algorithms, then we performed its application to a problem of a NP-complete class belonging to computational complexity theory known as the problem of subset sum, and we concluded with the effectiveness of these criteria to meet the conditions necessary and sufficient to address this problem of decision and we propose its use in the development and design of an algorithm since the application of these criteria can solve this problem and ensure algorithm optimization and determination in polynomial time.

Keyword: NP-complete problems; algorithm; criteria; property.

Introduction:

The NP-complete problems are important in computational complexity theory, decision and cryptography among others, to the extent that one of the problems considered by the Clay Mathematics Institute as the most important problem in this area with no solution so far is to demonstrate whether P=NP.

The Problem of the subset sum belongs to the category: NP complete and to solve it is equivalent to solving all of these categories. Its statement is as follows: "Given a set of integers, is there any non-empty subset whose sum is exactly equal to zero?" Notably, Stephen Cook and Leonid Levin formulated P (Thus, easy to find) versus NP (easy to check) independently in 1971.

In this article we present two criteria and a property of algorithms which, once used as certificates guarantee the solution of the goal expected no other than to establish the necessary and sufficient conditions to treat problems considered NP -complete. The computational complexity theory which deals with decision problems and complexity classes such as NP, P, Co -NP and NP-complete among others and you can find in this theory one of the most important problems still without solution, this problem is to establish or determine the ratio of computational complexity classes, which rises the question whether NP=P?

On the complexity classes of the most important is: NP-complete and among the most famous problems is the problem of the backpack and the traveling salesman problem. In this article we will discuss the problem on the subset sum which is also considered NP-complete.
Similarly with Matlab 6.5 programming language developed algorithms as well as having application to NP-complete problems also allow the development and design of algorithms on problems' application or primality test.

Commonly studied resources in computational complexity are:

a) Time: by approximating the number of steps required for the implementation or the development process used by an algorithm to solve a problem.
b) Space: using an approximation of the amount of memory required and used to solve a problem.

A decision problem is a problem that specifies a string of characters of data input and requires as solution a response for YES or NO.
The relationship between the complexity classes P and NP is studied in computational complexity theory.

Matlab is both an environment and a programming language, Matlab treated as an environment is an application development platform where you can easily develop a toolbox.

Matlab programming language allows us to create our own functions (which would be files .m) taking advantages of the properties of matrices to achieve faster execution. Toolbox is a set of intelligent tools for solving problems in specific application areas. Our algorithms aim is precisely to expand and improve an existing toolbox in computational complexity theory.

The founders of "The math Works" Jack Little and Cleve Moler found the need to create a programming language that has application for the development of a platform for bioinformatics data analysis based on Matlab.

This short paper presents an algorithm designed with this programming language and applications in number theory as well as computational complexity theory.

Problem statement:

The problem of subset sum is an accurate statement of the logical complexity of a class of problems known as **NP- complete** and is stated as follows:

Statement 1:

"Given a set of integers, is there any non-empty subset whose sum is exactly equal to zero?"

Equivalently can be stated as:

Statement 2:

"Given a set of integers and an integer: **S** Is there a subset whose sum is equal to: **S**?"

Reduction:

"Given a set of integers: **C** whose sum of its elements is as follows: **S** " Is there any subset: **B** belonging to the set: **C** whose combined subset elements: **B** is equal to: **S**?

Criteria or certificates and property of algorithms:

Criterion No. 1:

" The necessary and sufficient condition so that in a set of integers : **C** whose sum of its elements is equal to : **S** exist some subset : **B** belonging to the set : **C** and the sum of the elements of: **B** is equal to : S is that there is a subset : **A** non empty and belonging to the set : **C** and the sum of the subset elements : **A** is equal to : **Zero** "

Criteria No. 2:

'The necessary and sufficient condition for a set of integers: **C** whose sum of its elements is equal to : **S** There exists some subset : belonging to the set: **C** and the sum of the elements : **A** is equal to : **X** is that there is a subset : **B** non empty and belonging to the set: **C** and the sum of the subset elements : **B** is equal to : **S – X**"

Property of the algorithms:

Polynomial-time algorithms are closed relative to the composition. Intuitively, this means that if you write a function with a given polynomial time and we believe that calls to the same function are constant and polynomial time, then the entire algorithm is polynomial time. This is one of the main reasons that the class P is considered a separate machine. Some features of this machine, such as random access, are that it can calculate in polynomial time the main algorithm polynomial time by reducing it to a most basic machine.

Overview of the problem:

"Given a set of (n) elements (integers) determine or find a subset and verify or check that the sum of its elements is zero (0)"

Example:

Let: {-8, 7, 9, 1, 4, 5, 7, -13, -11} the given set.
Determine a subset: {-8, 1, 7, 4, 9, -13}\
Check if the sum of the elements of the subset is equal to: zero (0)

Remark: This problem is best known as the subset sum problem and is a problem considered NP -complete in computational complexity theory and it should be noted that the exact solution of this problem is related to a problem proposed by the Clay Mathematics Institute.
To solve this problem we applied the following algorithm designed in Matlab 6.5 programming language through a pseudo code based on two criteria as certificates together with the property of the algorithms which guarantee its determination in polynomial time.

Nieves algorithm:

* Optimal pseudo code algorithm that can prove that the sum of subset is in: **P**
This algorithm is based on: Two criteria and the property of the algorithms.

1) t = cputime (**Runtime**)
2) y = ([10 , -10, -5,16 , -9]) (**set of n elements**) (Input)
3) s = sum (y) (**function Sum of n elements**)
4) c = 0 or c = Any integer (**constant or input parameter**)
5) x1 = ([y (1)]) (**Any of the 2^n-1 subsets proper of Input**)
6) sum (x1) (**partial sum function subset**)
7) s == sum (x1) (**Output logic of existence and comparison**)
8) disp. ans == 1 (**Verification and conditional test**) (**Criterion 1**)
9) A = ([y (1)]) (**subset**)
10) B = ([y (2) and (3) and (4) and (5)]) (**complementary subset**)
11) disp. (**' There is a subset: B = S**) (**Criterion 1**)
12) disp.(**' There is a subset : A = 0'**) (**Criterion 1**)
13) break (Stopping execution or routine) (**nested loop**)
14) else
15) disp. (**'There is not a subset: B = S '**) (**Criterion 1**)
16) disp. (**'There is not a subset: A = 0'**) (**Criterion 1**)
17) end (**End**)
18) c == sum (x1) (**Output logic of existence and comparison**)
19) disp. == 1 (**Verification and Testing conditional**) (**Criterion 2)**
20) A = ([y (1)]) (**subset**)
21) B = ([y (2) and (3) and (4) and (5)]) (**complementary subset**)
22) disp. (**' There is a complementary subset A1 = 0'**) (**Criterion 2**)
23) disp. (**' There is a complementary subset B = C '**) (**Criterion 2**)
24) break (**Stopping execution or routine**) (**nested loop**)
25) else
26) disp. (**'There is not a subset: A = 0'**) (**Criterion 2**)
27) disp. (**'There is a not subset: B = C '**) (**Criterion 2**)
28) end (End)

Note: Executable with Matlab 6.5

Note: This algorithm was presented by the author at:

The XXIV Research Technical Conference and IV Postgrade at Unellez, 2013
And it should be noted that: **If:** it is proof that the problem of the sum of subset is in: **P**
Then: It is proof that **P = NP**

Analysis of results:

Notably, the design of an algorithm pseudo code contains both criteria as certificates would achieve optimization with respect to the search (Problem P) which describes the criterion No. 1 and also ensures verification (NP) with the criterion 2 which is based on the following theorems: 1 and 2 proving their effectiveness and implementation in polynomial time.

Theorem : 1

If : $\exists !B \subset C$

$\forall C$

Such That:The sum of the elements of: C is equal to the sum of the elements of:B

Then : $\exists A \subset C$

Where: The sum of the elements of:(A) = 0

Theorem : 2

If : $\exists !B \subset C$

$\forall C$

Such that: The difference of the sum of the elements of: C

minus the sum of the elements of:X

is equal to the sum of the elements of:B

Then : $\exists X \subset C$

Where:The difference of the sum of the elements of:C

minus the sum of the elements of:B

is equal to the sum of the elements of:X

Conclusion and recommendations:

Once carried out the corresponding analysis to the examples presented above is concluded with recognizing the effectiveness of both criteria to address the problem of subset sum and this allows us to propose to the scientific so that it considers its study and application to similar problems broadening its optimization and algorithmic efficiency range. Since this pseudo code transforms the problem subset of the sum of an NP-complete problem to NP and likewise from NP-complete to P. Leading this result to characterizing criteria which ensure through certificates when: P = NP

References:

1. Richard M. Karp (1972). «Reducibility Among Combinatorial Problems». En R. E. Miller and J. W. Thatcher (editors). *Complexity of Computer Computations*. New York: Plenum. pp. 85–103.
2. Stephen Cook (1971). The Complexity of Theorem Proving Procedures. *Proceedings of the third annual ACM symposium on Theory of computing*. pp. 151–158.
3. David Zuckerman (1996). «On Unapproximable Versions of NP-Complete Problems». *SIAM Journal on Computing* 25 (6): pp. 1293–1304. http://citeseer.ist.psu.edu/192662.html
4. G.B. Mathews, *On the partition of numbers*, Proceedings of the London Mathematical Society, 28:486-490, 1897.
5. T. Cormen, C. Leiserson, R. Rivest. *Introduction to Algorithms*. MIT Press, 2001. Chapter 35.5, *The subset-sum problem.*
6. Michael R. Garey and David S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, 1979, W.H. Freeman. ISBN 0716710455 A3.2: SP13, pg.223.
7. Nieves R. Rodolfo "Dos criterios y su aplicación a un problema NP- completo" Memorias de las XX Jornadas Técnicas de Investigación y IV de Postgrado; Unellez. Ed. Horizonte. Venezuela. 2013.