Analysis of the postulates produced by Karp’s Theorem.

JERRALD MEEK

This is the final article in a series of four articles. Richard Karp has proven that a deterministic polynomial time solution to K-SAT will result in a deterministic polynomial time solution to all NP-Complete problems. However, it is demonstrated that a deterministic polynomial time solution to any NP-Complete problem does not necessarily produce a deterministic polynomial time solution to all NP-Complete problems.

Categories and Subject Descriptors: F.2.0 [Analysis of Algorithms and Problem Complexity]: General
General Terms: Algorithms, Theory
Additional Key Words and Phrases: P vs NP, NP-complete

1. INTRODUCTION.

The present author has previously shown that a NP-complete problem is solvable in deterministic polynomial time if and only if a polynomial search partition can be found in polynomial time.

Additionally, it has been shown that some instances of the 0-1-Knapsack problem do not have a deterministic polynomial time solution, unless the SAT problem has a deterministic polynomial time solution.

This is the final article in a series of four, wherein the purpose will be to finally determine that the SAT problem has no deterministic polynomial time solution. It will then become clear that the 0-1-Knapsack problem, and any similar NP-complete problem which depends on SAT for a deterministic polynomial time solution, cannot be solved in deterministic polynomial time.

2. PRELIMINARIES.

Previously the author has proven the following theorems, which will be assumed true in this article.

**Theorem 4.4 from P is a proper subset of NP. [Meek Article 1 2008]** 2.1. 
P = NP Optimization Theorem.

The only deterministic optimization of a NP-complete problem that could prove P = NP would be one that can always solve a NP-complete problem by examining no more than a polynomial number of input sets for that problem.

**Theorem 5.1 from P is a proper subset of NP. [Meek Article 1 2008]** 2.2. 
P = NP Partition Theorem.

People who wish to remain anonymous have offered comments and suggestions which have improved this work. The author wishes to express his appreciation for their assistance.

Jerrald Meek Copyright ©2008
The only deterministic search optimization of a NP-complete problem that could prove $P = NP$ would be one that can always find a representative polynomial search partition by examining no more than a polynomial number of input sets from the set of all possible input sets.

**Theorem 5.1 from [Meek Article 2 2008] 2.3.**

Analysis of the deterministic polynomial time solvability of the 0-1-Knapsack problem.

**Knapsack Random Set Theorem.**

Deterministic Turing Machines cannot exploit a random relation between the elements of $S$ to produce a polynomial time solution to the Knapsack problem.

**Theorem 6.1 from [Meek Article 2 2008] 2.4.**

Analysis of the deterministic polynomial time solvability of the 0-1-Knapsack problem.

**Knapsack Composition Theorem.**

Compositions of $M$ cannot be relied upon to always produce a deterministic polynomial time solution to the 0-1-Knapsack problem.

**Theorem 6.2 from [Meek Article 2 2008] 2.5.**

Analysis of the deterministic polynomial time solvability of the 0-1-Knapsack problem.

**Knapsack $M$ Quality Reduction Theorem.**

Any quality of $M$ that could be used to find a composition of $M$ within $S$ would be equivalent to finding all compositions of $M$.

**Theorem 7.1 from [Meek Article 2 2008] 2.6.**

Analysis of the deterministic polynomial time solvability of the 0-1-Knapsack problem.

**Knapsack Set Quality Theorem.**

Using any quality of the elements of $S$ to solve the 0-1-Knapsack problem will be no less complex than the standard means of solving the 0-1-Knapsack problem.

The definition of the 0-1-Knapsack problem used in this article will be based off of that used by Horowitz and Sahni [Horowitz and Sahni 1974].

1. Let $S$ be a set of real numbers with no two identical elements.
2. Let $r$ be the number of elements in $S$.
3. Let $\delta$ be a set with $r$ elements such that
   \[ \delta_i \in \{0, 1\} \leftarrow 1 \leq i \leq r \]
4. Let $M$ be a real number.

Then

\[ \sum_{i=1}^{r} \delta_i S_i = M \]

Find a variation of $\delta$ that causes the expression to evaluate true.
3. **ASSUMPTIONS PRODUCED BY KARP’S THEOREM.**

Theorem 3 from *Reducibility among combinatorial problems*. [Karp 1972, p. 93] 3.1.

The language $L$ is (polynomial) complete if

1. $L \in NP$
2. $SATISFIABILITY \propto L$

Either all complete languages are in $P$, or none of them are. The former alternative holds if and only if $P = NP$.

Karp’s Theorem is often interpreted to mean that if any *NP-Complete* problem is discovered to be solvable in deterministic polynomial time, then all *NP-Complete* problems will be solvable in deterministic polynomial time. This common interpretation is based off of two postulates:

1. All *NP-Complete* problems are reducible to K-SAT, and therefore can be solved in deterministic polynomial time if the underlying K-SAT problem has a fast solution.
2. A deterministic polynomial time solution to some *NP-Complete* problem will ultimately provide a deterministic polynomial time solution to the underlying K-SAT problem.

In this article the author has no argument with the first postulate regarding Karp’s Theorem. Karp has satisfactorily demonstrated that all *NP-Complete* problems are reducible to K-SAT. It was Richard Karp who suggested that a deterministic polynomial time solution to K-SAT would prove $P = NP$; the present author agrees.

The true purpose of this article is to challenge the second postulate. The idea that a deterministic polynomial time solution to any *NP-Complete* problem will magically produce a proof that $P = NP$ is often assumed to be implied by Karp’s Theorem; however it is not so implied, and this idea is completely misguided.

4. **A NP-COMPLETE PROBLEM WITH A DETERMINISTIC POLYNOMIAL TIME SOLUTION.**

It has been previously demonstrated by the present author that the process of converting a decimal number into a binary number can be represented as a form of the 0-1-Knapsack problem, and therefore is *NP-Complete*. However, this particular *NP-Complete* problem does have a deterministic polynomial time solution.

The reason that the problem of converting a decimal number to a binary number can be solved in deterministic polynomial time is because there is a special relationship between the elements of $S$ in this particular 0-1-Knapsack problem. This relationship can therefore be exploited to determine which literal of the underlying K-SAT problem has a true value. See [Meek Article 2 2008, sec 5.1].

**Proof.** Assume the following:

1. The set $S$ has $r$ elements.
2. There are $r^2$ subsets of $S$.  

ArXiv, Vol. V, No. N, Month 20YY.
—Some special relationship between the elements of $S$ allows for a fast determination of one subset of $S$ which sums to $M$. That same relationship also allows a fast determination if no subset of $S$ sums to $M$.

—Let $\delta$ be a set with $r$ elements such that

$$\delta_i \in \{0, 1\} \leftarrow 1 \leq i \leq r$$

—Let $M$ be a real number.

Then

$$\sum_{i=1}^{r} \delta_i S_i = M$$

can be represented as

$$\left[ 0 = M \right] \lor \left[ S_1 = M \right] \lor \left[ S_2 = M \right] \lor \left[ S_3 = M \right] \lor \ldots \left[ S_r = M \right] \lor$$

$$\left[ S_1 + S_2 = M \right] \lor \left[ S_1 + S_3 = M \right] \lor \ldots$$

The above K-SAT problem is extended until there is exactly one literal for each subset of $S$. To determine the truth-value of the entire expression it is only necessary to determine that any one literal is true, or that all literals are false.

Because of the special relationship between the elements of $S$, it is possible to quickly find one literal that is true if one exists. It is also the case that if an attempt to find such a literal fails, then it can be quickly determined that all literals are false.

The fast solution to this problem is produced by the fact that it is not necessary to determine the truth-value of each and every literal individually. If it were the case that each literal had to be individually determined, then the $P = NP$ Optimization Theorem would not allow the optimization to process in deterministic polynomial time.

It is then easy to see that the deterministic polynomial time solution to this problem cannot produce a fast solution to the underlying K-SAT problem. Infact, if the optimization did produce a solution to the underlying K-SAT problem, then the solution could not run in deterministic polynomial time.

Instead this optimized solution simply produces a fast identification of one true literal in the K-SAT problem. This ability to quickly identify a true literal in the K-SAT problem cannot be transferred to other NP-Complete problems, because the fast solution is dependant upon a special condition of a particular instance of the 0-1-Knapsack problem, which will not exist in all other NP-Complete problems. $\square$

### 4.1 Formalized argument proving no deterministic polynomial time solution from the form of a NP-Complete problem for K-SAT where $k \geq 3$.

Assume:

—The K-SAT problem associated to the hereinabove described 0-1-Knapsack problem has $k$ literals where $k = 2^{|S|} \geq 3$.

—$x$ is a set of literals with $k$ elements; each literal has an unknown truth value.
The K-SAT problem in question is
\[ x_1 \lor x_2 \lor x_3 \lor \ldots \lor x_k \]

**Proof.** Assume:

- \( Q \) = some predictable quality.
- \( S = \{ x | Q \rightarrow x \} \)
- \( M \in R \)
- \( x_2 = T \)
- \( A(y, z) \) is an algorithm which identifies one element of \( x \) which has a true value by comparing the elements of \( y \) with quality \( Q \) to the value of \( z \).

Then
\[ A(S, M) \equiv x_2 \]

The optimized 0-1-Knapsack algorithm uses the elements of \( S \) with quality \( Q \), and the value of \( M \) to identify that one literal has a true value. 

**Proof.** Assume:

- \( Q \) = some predictable quality.
- \( S = \{ x | x \in R \} \)
- \( M \in R \)
- \( A(y, z) \) is an algorithm which identifies one element of \( x \) which has a true value by comparing the elements of \( y \) with quality \( Q \) to the value of \( z \).
- \( \otimes \) represents an undefined result.

Then
\[ A(S, M) \equiv \otimes \]

The optimized 0-1-Knapsack algorithm uses the elements of \( S \) with quality \( Q \), and the value of \( M \) to identify that one literal has a true value. However, the quality \( Q \) is not associated with the elements of \( S \), so the result is undefined. 

**Proof.** Assume:

- \( Q \) = some predictable quality.
- We are given a K-COL problem (graph coloring problem where \( k \geq 3 \)).
- \( S = \otimes \)
- \( M = \otimes \)
- \( A(y, z) \) is an algorithm which identifies one element of \( x \) which has a true value by comparing the elements of \( y \) with quality \( Q \) to the value of \( z \).
- \( \otimes \) represents an undefined result.
Then

\[ A(S, M) \equiv \otimes \]

The optimized 0-1-Knapsack algorithm uses the elements of \( S \) with quality \( Q \), and the value of \( M \) to identify that one literal has a true value. However, the set \( S \) and value \( M \) are not provided in the definition of the K-COL problem, so the result is undefined.

**Theorem 4.1. K-SAT Input Relation Theorem.**

A solution that solves a NP-Complete problem in deterministic polynomial time, and relies upon some relationship between the inputs of the problem, does not produce a deterministic polynomial time solution for all instances of K-SAT.

5. CONCLUSION: \( P \neq NP \)

The \( P = NP \) Optimization Theorem eliminates a search algorithm by exhaustion as a polynomial time solution to a NP-Complete problem.

The \( P = NP \) Partition Theorem eliminates a search algorithm to produce a polynomial size search partition which can be used in a polynomial time solution to a NP-Complete problem.

The combination of the Knapsack Random Set Theorem, Knapsack Composition Theorem, Knapsack M Quality Reduction Theorem, and Knapsack Set Quality Theorem indicate that any deterministic polynomial time solution to the Knapsack problem, which relies upon some quality of the problem, will not produce a solution that works for all instances of the Knapsack problem.

The K-SAT Input Relation Theorem indicates that a solution dependant on the form of any NP-Complete problem to sidestep the \( P = NP \) Optimization Theorem will not produce a fast solution for the underlying K-SAT. It is then the case that such an optimization is not transferable to all other NP-Complete problems.

It has therefore been determined that:

(1) Search algorithms will not prove \( P = NP \).

(2) Partitioning the search group will not produce an algorithm that proves \( P = NP \).

(3) Relying upon the form of an individual NP-Complete problem can sometimes solve that problem in deterministic polynomial time but does not provide an algorithm that proves \( P = NP \).

If it can be accepted that any algorithm that solves a NP-Complete problem will be one of the following:

(1) a search algorithm, which examines all possible literal values;

(2) a partitioned search algorithm, which partitions the set of all possible literal values and only examines a limited number of the possible input sets;

(3) or a direct solution relying upon the form of the problem.

Then the conclusion \( P \neq NP \) can be accepted as final.

Q.E.D.
6. VERSION HISTORY.
The author wishes to encourage further feedback which may improve, strengthen, or perhaps disprove the content of this article. For that reason the author does not publish the names of any specific people who may have suggested, commented, or criticized the article in such a way that resulted in a revision, unless permission has been granted to do so.

arXiv Current Version
23Aug08 Submitted to arXiv.

REFERENCES
Horowitz, E. and Sahni, S. 1974. Computing partitions with applications to the knapsack problem. J. Assoc. Comput. Mach. Zbl 0329.90046.
Karp, R. 1972. Reducibility among combinatorial problems. Complexitv of computer computations, Proc. Sympos., IBM Thomas J. Watson Res. Center. MR378476 Zbl 0366.68041.
Meek, J. 2008. \( P \) is a proper subset of \( NP \). arXiv:0804.1079 Article 1 in series of 4.
Meek, J. 2008. Analysis of the deterministic polynomial time solvability of the 0-1-Knapsack problem. arXiv:0805.0517 Article 2 in series of 4.

Received xx/2008; revised xx/2008; accepted xx/2008