The Copyrightability of Encryption Methods and Encryption Algorithms on Computers

Zoe Milak
Zoe.Milak@chicagounbound.edu

Follow this and additional works at: http://chicagounbound.uchicago.edu/uclf

Recommended Citation
Available at: http://chicagounbound.uchicago.edu/uclf/vol1996/iss1/20

This Comment is brought to you for free and open access by Chicago Unbound. It has been accepted for inclusion in University of Chicago Legal Forum by an authorized administrator of Chicago Unbound. For more information, please contact unbound@law.uchicago.edu.
The Copyrightability of Encryption Methods and Encryption Algorithms on Computers

Zoe Milak†

Due to the growth of business and correspondence via the Internet, encryption has become increasingly necessary.1 To ensure the integrity of contracts, e-mail, and personal documents transmitted through the Internet, encryption technology is essential.2 Additionally, like everything else that man makes, encryption technology is subject to the desire to protect his right to just compensation. Intellectual property laws are tools for reaching this end.

Patents and copyrights are the two major protections for intellectual property. The Constitution is the source of authority for both: "The Congress shall have Power ... To Promote the progress of Science and useful Arts, by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries."3 Stated very simply, patents protect inventions or processes,4 and copyrights protect the creator's expression of an idea.5 In the area of computer law, it can be difficult to differentiate between an idea, an invention, a process, and an expression.6 Consequently, there are overlaps and gaps between patent and copyright law.

This Comment will present patent and copyright law as they relate to algorithms, with a view towards protecting an encryption algorithm under copyright law. Part I of this Comment will define "algorithm" and "encryption algorithm," and explain common encryption techniques currently used with computers.

† B.A. 1993, Loyola University Chicago; J.D. Candidate 1997, University of Chicago.
3 US Const, Art I, § 8, cl 8.
Part II will give brief descriptions of patent and copyright law. It will survey the traditional treatment of algorithms under patent law, the historic arena of fights over the protection of algorithms. It will then profile software copyright law to determine what it has to offer to the discussion of the possible treatment of algorithms under copyright law. In Part III, the Comment will advocate the copyrightability of cryptosystems and their underlying algorithms subject to certain conditions.

I. AN INTRODUCTION TO THE ALGORITHM AND ENCRYPTION TECHNIQUES

A. A Basic Definition of “Algorithm”

Since one must learn to walk before one may run, it is necessary to understand what an algorithm is before exploring whether an encryption “algorithm” should be copyrightable. Various definitions of “algorithm” are: “a recipe or specific set of rules or directions for performing a task,”7 “an unambiguous specification of a conditional sequence of steps or operations for solving a class of problems,”8 and “the design of an ordered sequence of precise steps that describe the solution of a given problem.”9 Thus, the critical characteristics are: 1) there is a problem that needs to be solved, and 2) the algorithm provides a plan to solve the problem.

Very simple examples of algorithms are a recipe for a cake and a blueprint for a house.10 Each states the problem (baking a cake, building a house) and provides directions for solving the problem (combine ingredients, measure beams). Similarly, a musical score can be an algorithm. The score indicates the notes and the sequence in which they are played. It may also indicate the instruments that the composer intends to be used to play the particular notes.

---

7 A cryptosystem “is a method of disguising messages so that only certain people can see through the disguise. . . . A cryptosystem is usually a whole collection of algorithms.” Eric Bach, et al., Cryptography FAQ § 3.1, http://www.cis.ohio-state.edu/hypertext/faq/usenet/cryptography-faq/part03/faq/html (v 1.0 1993).
9 Allen Newell, The Models are Broken, the Models are Broken, 47 U Pitt L Rev 1023, 1024 (1986).
11 Id.
B. Algorithms in Computing

Although the prior examples make an algorithm appear simple, an algorithm is difficult to conceptualize in the computer context. For example, the “bakery algorithm” is one type of a computer programming algorithm.\(^{12}\) It derives its name from a scheduling algorithm often used by bakeries, stores, and other institutions that must provide order when multiple customers attempt to use the same resource simultaneously.\(^{13}\) In programming, this is referred to as the “critical section problem.”\(^{14}\) To understand how this algorithm works, assume that a system consists of multiple processes.\(^{15}\) Each process contains a portion of code, called the critical section, that cannot be executed while the other processes are executing. The system must have some way of coordinating the execution of the critical sections without preventing one or several of the processes from executing. In the bakery algorithm, each customer (or process) receives a number. The customer with the lowest number is served first. If two customers receive the same number, the customer with the alphabetically first name gets priority. “That is, if \(P_i\) and \(P_j\) receive the same number and if \(i<j\), then \(P_i\) is served first.”\(^{16}\) When the first customer completes his transaction (critical section), he steps away from the counter and must choose (be assigned) a new number to be served (execute) again. Meanwhile, the second customer steps to the counter, is served, then steps away and chooses a new number.

This example is not the only type of algorithm. A programmer may express an algorithm in several ways: by a narrative description, in a flowchart, and through algorithmic language.\(^{17}\) A narrative description is an expression through words, such as a recipe or the bakery algorithm example.\(^{18}\) The problem with the narrative description is that it can be too wordy or imprecise. Another method is the flowchart.\(^{19}\) A flowchart consists of pic-

\(^{13}\) Id at 170.
\(^{14}\) Id at 165-66.
\(^{15}\) In the area of computers, a “process is a program in execution.” Id at 98.
\(^{16}\) Silberschatz & Galvin, *Operating System Concepts* at 171 (cited in note 12).
\(^{18}\) Another example is “Read a set of four marks. Compute their average by summing them and dividing by 4. If this average is below 50, display the grade with a failing message; otherwise display the grade with a passing message.” Id §§ 2-2.1 at 21.
\(^{19}\) Id §§ 2-2.2 at 21-22.
tures or icons that “show [ ] the logic of an algorithm, emphasizing the individual steps and their interconnections, that is, the way in which control flows from one action to the next.” The preferred way of expressing an algorithm is through algorithmic language. It is a more refined version of the narrative description that separates the steps into distinct statements that define the necessary variables. Once the programmer has written the algorithm in one of the above ways, he is ready to design the program.

It is important to keep in mind that an algorithm is not a computer program. A program is the implementation of an algorithm. The relationship between an algorithm and the program is like that between an outline and a composition. One may write many different compositions based on a single outline. Likewise, one may write many different programs based on a single algorithm. A program is not just the translation of an algorithm into a computer language.

The algorithm defines the problem that needs to be solved and provides the plan to solve the problem. With the algorithm, the programmer writes precise instructions for the computer on how to perform a desired function; these directions constitute the program. A programmer may write the program in any of several computer languages such as COBOL, ALGOL, or PASCAL. Such languages are easily understandable to the programmer and may be based on or at least resemble English. This set of instructions written in the programmer-understandable language is called “source code.” The computer cannot use the source code directly, so a program called a compiler translates the

---

20 Id at 21.
22 To use an example, algorithmic language might express it as:

1. Read a set of four marks.
2. Compute their average by summing them and dividing by 4.
3. If the average is below 50, then display the grade with a failing message; otherwise display the grade with a passing message.
4. Halt.

Id § 2-2.3 at 22.
23 See id § 2-1 at 19-20.
24 Id § 1-3 at 14.
source code into a machine-readable form called the "object code." The object code is expressed in a string of binary numbers (ones and zeros), also known as "machine language," that is virtually unintelligible to the programmer. Additionally, because what is understandable by one type of computer is not necessarily understandable by another, the programmer has a choice of many types of computer hardware to use. A single algorithm in the hands of two different programmers may spawn two different programs written in two different languages for two different machines.

C. An Exploration of Encryption in High-Tech Communications and the Role of Algorithms

The purpose of encryption is to keep the content of a message private from third parties. "Encryption is the transformation of data into a form unreadable by anyone without a secret decryption key." An encryption algorithm encodes a message by assigning an alphanumeric value to each character in a document, essentially turning the original message into gibberish. Someone who has a corresponding decryption algorithm, whether it is the same as the encryption algorithm or an entirely different algorithm, may turn the gibberish back into understandable text. Put more simply, an encryption algorithm scrambles a message so that only those who know the values assigned can decode and read the message.

A simple example is the Caesar Cipher named after Julius Caesar who is said to have created it. This cipher replaces a letter of the alphabet with another letter. There is a 3-letter "shift" so that "A-B-C-D" would have the values "D-E-F-G." Thus the cipher "V-H-F-U-H-W-P-H-V-D-J-H" would be deciphered to state "S-E-C-R-E-T M-E-S-S-A-G-E." Of course, one may break it easily—just try all possible alphabet combinations until the message makes sense.

---

27 Id.
28 Id.
30 Id.
32 Id.
Given that all ciphers are potentially breakable, there is a constant search for better methods of encryption. One modern example is the Enigma Machine, an encryption device used by the Germans during World War II. In appearance it "resembled a portable typewriter but the depression of a key worked an internal system of gears which allotted any letter input an alternative letter not logically to be repeated before 200 trillion subsequent depressions." The transmitting operator prefaced the message with a repeated sequence of the same letters that indicated to the receiver how the transmitting machine was set. The Allies broke the code because they were able to recognize the patterns of characters used, in part because of irregularities in the pattern due to mistakes made in transcription by the Germans.

The next technological step up from the Enigma Machine is the modern computer. A computer can use more complex algorithms because it can process information faster than the human mind. The increased complexity of the algorithms means increased security. Some of the encryption methods used to encode information on computers are practically unbreakable. The following section will look at three encryption methods used in cyberspace: private-key encryption, public-key encryption, and hybrid systems.

The first method, private-key encryption, is the most simple and the least secure method. In this method, and in encryption generally, text that is understandable and readable by the encoding party, called plain text, is input into an algorithm called the encryption algorithm. Under the algorithm's directions, the plain text is converted into cipher text—text that, while the characters are recognizable, the meaning is gibberish to the reader.

---

36 Id at 499.
37 Id.
38 Id.
41 Id at 471.
42 Id.
The encrypting party then sends the cipher text to another party who must decrypt or decode the text with another algorithm, the decryption algorithm. 43 This algorithm corresponds to the encryption and is, theoretically, the only algorithm that can convert the cipher text back into plain text. 44

What differentiates the private-key system from conventional types of encryption systems is its use of keys. The keys indicate to the algorithm how to encode or decode the message. 45 Think of the algorithm as a decoder ring from a cereal box. If the user sets the ring at notch X, the user will get a pattern of A=1, B=2, and so on. If another user sets the ring at notch Y, he will get a pattern of B=1, C=2, and so on, with A=26. Both sender and recipient use the same ring, but if the recipient attempts to decode a message sent with notch X using notch Y, the message will become even more confused. The private-key system operates on the same principle; several people may use the same algorithm, but the message is not decryptable without the key to a particular message. 46

The security gap in the private-key system is not the algorithm, for as stated before, several people may know and use the algorithm. The security gap, however, is the secrecy of the keys. 47 It is necessary to provide the decrypting party with the key for him to decrypt the message. A third party may, however, intercept the key and have unauthorized access to the message, for the key itself may be sent via nonsecure channels. The integrity of the system thus depends upon the ability of the parties to transfer the keys without interception. 48

The public-key encryption method uses the same principle as the private-key method except that it uses a pair of related keys. 49 The key generator keeps one key private and releases the other to the public, or at least a limited number of users. 50

43 Id.
45 Id.
46 Id.
47 Id.
48 Id.
49 Id.
50 Id.

50 Silberschatz & Galvin, Operating System Concepts § 14.6 at 472.
Others may use the public key to send encrypted information to the private-key holder. A third party cannot use the public key to decrypt a message sent using the public key; only the private key can decrypt. Also, one cannot figure out the private key from an examination of the public key. Because this method eliminates the need to exchange a secret key, public-key encryption is more secure than the private-key method.

The third method is a hybrid system that combines features of the first two. The best known example of this method is a program called Pretty Good Privacy ("PGP"). PGP combines a public-key algorithm with a private-key algorithm to get the best features of both systems, creating an even more secure system.

PGP uses the RSA[57] public key algorithm for encryption in tandem with the conventional IDEA[58] algorithm. A single IDEA key is generated for encrypting the message with IDEA, this is a conventional cryptosystem so the same key will decrypt the massage [sic]. RSA is them [sic] used to encrypt the IDEA key using the recipients [sic] public key. The recipient uses their [sic] copy of PGP which decrypts the RSA encrypted IDEA key with their [sic] private key. The IDEA key is then used to decrypt the rest of the message.  

---

51 Id.
52 Id.
54 Id § 1.4.
58 The IDEA algorithm gets its name from International Data Encryption Algorithm. It was developed under a joint project by Ascom Systec and the Swiss Federal Institute of Technology Zurich. IDEA—The Future Standard for Data Encryption from Ascom, http://www.ascom.ch/web/systec/security/idea1.html (March 30, 1996) (footnote not in original).
The message is encrypted using the private-key method and then the private key is encrypted using the public-key method. This combines the speed of the private-key system with the better security of the public key.\textsuperscript{60}

II. INTELLECTUAL PROPERTY PROTECTION FOR ALGORITHMS

A. Why Should Algorithms Be Protected and What Are The Options?

Intellectual property laws encourage development and experimentation by protecting the developer's proprietary rights in his creations.\textsuperscript{61} The law limits how others may use the creation, reserving to the inventor the full ability to exploit the product for economic, scientific, or artistic gain.\textsuperscript{62} There are several different types of protection with varying lengths and depths, each suited to a different type of creation.\textsuperscript{63} For example, copyrights protect various kinds of artistic work, whereas patents protect machines.\textsuperscript{64}

Traditionally, creators have sought to protect algorithms under patent law, but copyright law may be a more practicable alternative.\textsuperscript{65} This Part will survey the fundamentals of patent and copyright law and explore several theories that shed light on the potential treatment of algorithms under the existing intellectual property scheme.


Commentators most often focus on patent law when they consider intellectual property protection for algorithms.\textsuperscript{66} This is

\textsuperscript{60} Id.
\textsuperscript{63} For brief descriptions of copyright, patent, and trademark laws, see Henn, Copyright Law at 1-4 (cited in note 61). For a description of trade secret law, see Peter D. Rosenberg, Patent Law Basics § 2.01 at 2-1 to 2-8 (Clark Boardman Callaghan, 1995).
\textsuperscript{64} See Donald S. Chisum, Intellectual Property: Copyright, Patent and Trademark Law § 1.01 at 1-1 and § 6.01 at 6-1 (Matthew Bender & Co., 1980).
\textsuperscript{65} Note, Copyright Protection for Computer Flow Logic and Algorithms, 5 Computer/L J 257, 285 (1984) (suggesting that copyright law may be a better instrument of protection of algorithms than either patent or trade secret law if ambiguities in the Copyright Act are resolved).
\textsuperscript{66} See, for example, Michael Gemignani, Should Algorithms be Patentable?, 22


partly because patent law provides the greatest protection of all the intellectual property rights: twenty years of exclusive use. Patent law grants the claimant a virtual monopoly on the invention during the statutory period. In contrast, while copyright law provides protection during the owner’s life plus fifty years, the extent of its protection is more limited than that of patent law. It primarily protects the creator’s almost exclusive right to make copies of the creation. Commentators also find patent law more attractive than copyright law because an algorithm conceptually resembles the inventions and processes which patent law protects. In the following subsections, this Comment will lay out a brief description of patent law and the theories used by the courts in cases involving algorithms.

1. A brief survey of patent law.

“A patent is a right to exclusive use [of an invention] granted by law.” A patent gives the holder twenty years of monopoly control of the invention. Because this grant of control is so complete and is contrary to the goal of dissemination of knowledge and technology that the government promotes in other areas, the requirements to receive patent protection are strict. An invention must be useful, novel, and nonobvious.

Section 101 of the Patent Act describes patentable statutory subject matter as “any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof...” The statute contains two categories of in-


69 17 USC § 302(a) (1994).
71 17 USC § 106.
73 Rosenberg, Patent Law Basics § 1.03 at 1-7 (cited in note 63).
74 35 USC § 154(a)(2).
75 See, for example, the Copyright Act, 17 USC §§ 101-1101, discussed in Part II.C of this Comment.
77 35 USC § 102 (1994).
78 35 USC § 103 (1994).
ventions: processes and products. A process is "synonymous with art, method, and mode of treatment. A process consists of ... a series of steps, performed upon specified subject matter to produce a physical result." A product, on the other hand, is a physical object, such as a machine in the usual sense of the word. These two categories mandate a physical action and a physical result. Although useful in the usual sense of the word, purely mental activities do not qualify for patent protection, nor do discoveries of laws of nature and naturally occurring objects.


Courts have resisted extending patent protection to algorithms because they consider them to be a law of nature, one of the categories excluded from protection above. Diamond v Chakrabarty explained the law of nature doctrine in the nonalgorithm context of human-made microorganisms. In Chakrabarty, the respondent filed a patent application for a human-made, genetically engineered bacterium which was able to break down components in crude oil, something naturally occurring bacteria could not do. The respondent believed that the bacterium would be valuable in the treatment of oil spills. The question in Chakrabarty was whether the microorganism constituted a "manufacture" or "composition of matter" according to Section 101 of the Patent Act. The Court stated that a law of nature or physical phenomenon is not patentable because such concepts or qualities, although previously unknown, existed prior to and independently of being discovered. Examples of laws of nature that the Court gave were newly discovered minerals and plants, Einstein's theory of relativity, and Newton's law of gravity. In contrast, the microorganism in Chakrabarty had no independent existence; it existed only because of the actions of the respondent and would not have existed, nor ever did exist, in

---

81 Id § 6.01[2] at 6-16.
82 Id § 6.02[3] at 6-34.
83 Id § 6.02[2] at 6-33 to 6-34.
86 Id at 305.
87 Id at 305.
88 Id at 307.
89 Chakrabarty, 447 US at 309.
90 Id.
nature. The Court thus found that it was not a manufacture of composition of matter, but an unpatentable law of nature.

The first significant decision that involved the examination of the patentability of an algorithm was Gottschalk v Benson. In that case, Benson filed an application for a patent on an invention that converted binary-coded decimal numerals into pure binary numerals in a digital computer. The invention was not limited to any particular technology, machine, or end use. The Court determined that the question before it was whether the conversion method constituted a "process" under section 100(b) of the Patent Act. The Court found that the applicant's invention was an algorithm because it was nothing more than a general formula for converting numbers. It stated that the practical effect of allowing a patent on the method described would be a patent on the algorithm because the only practicable use for the underlying mathematical formula was in a computer. It continued that allowing a patent on the use of the algorithm in a computer would effectively preempt the use of the formula, for the only practical application of the formula is in a digital computer. The Court denied the applicant's claim, but it carefully stated that its decision did not hold that a program could never be patented.

In its analysis, the Court stated two important conclusions. First, the Court defined an algorithm as a "procedure for solving a given type of mathematical problem." Second, the Court categorized the algorithm as a law of nature. These conclusions had great ramifications in later cases involving algorithms. Some courts, when confronted with alternative definitions of "algorithm," used Benson to avoid delving into the potential legal implications of adopting a different definition. Other courts

---

91 Id at 310.
92 409 US 63 (1972).
93 Id at 64.
94 Id.
95 Id.
96 Benson, 409 US at 65.
97 Id at 71-72.
98 Id at 72.
99 Id at 71.
100 Benson, 409 US at 71. Patents have been granted for algorithms, but these have never been challenged before the Court. The validity of these patents is therefore uncertain.
101 Id at 65.
102 Id at 67-68.
noted the narrowness of the Court’s definition, which left open the possibility of patenting an algorithm in the future. Defining the algorithm as a law of nature also reinforced the presumptive inability to patent any claim prominently containing an algorithm.

In Parker v Flook, the Supreme Court hid behind the Benson definition of algorithm. The Flook Court addressed whether the identification of a limited, conventional, post-solution application of a formula rendered a method patentable. The applicant submitted a patent application for a method of updating “alarm limits,” the number that, when reached, signals an abnormality during the catalytic conversion process. The method in the claim called for the measuring of the present values of the necessary variables, calculating the updated alarm limit values with an algorithm and adjusting the actual alarm limit values. The claim did not instruct the operator on how to take the measurements of the variables or how to adjust the alarm limits. The Court stated that this was the conventional method of determining the alarm limit with only one new feature, the added mathematical formula, and therefore was unpatentable. In reaching this conclusion, the Court noted that it used the term “algorithm” as it was defined in Benson and restated that an algorithm is not a statutory process, but a law of nature. The applicant had argued that the patent was only for a narrow use of the algorithm and that this narrow, post-solution activity distinguished it from Benson. Rejecting this argument, the Court stated that allowing post-solution activities tagged onto algorithms to transform them into patentable processes would create a mockery of the law. As an example, the Court noted that “the Pythagorean theorem would not have been patentable, or partially patentable, because a patent application contained a final step indicating that the formula, when

---

104 Diehr, 450 US at 186 n 9; Application of Walter, 618 F2d 758, 764 n 4 (CCPA 1980); Application of Freeman, 573 F2d 1237, 1245-46 (CCPA 1978).
106 Id at 585.
107 Id.
108 Id.
110 Id at 594-95.
111 Id at 585 n 1.
112 Id at 589.
113 Flook, 437 US at 589-90.
114 Id at 590.
solved, could be usefully applied to existing surveying techniques.\textsuperscript{115}

Though rejecting the party's application, the Court noted that the mere inclusion of an unpatentable algorithm does not automatically render a claim unpatentable, if viewing the invention as a whole revealed anything new and useful beyond the algorithm.\textsuperscript{116} The invention was conventional, except that it contained a novel algorithm.\textsuperscript{117} The novelty of the algorithm, however, was irrelevant because when the Court examined the application, it assumed that the algorithm was previously known.\textsuperscript{118} Because the invention as a whole revealed nothing new and useful, the Court found it to be nonstatutory subject matter.\textsuperscript{119}

In \textit{Diamond v Diehr},\textsuperscript{120} even though the Court used an analysis similar to that used in \textit{Flook}, it came to a different result based on the facts. The issue in \textit{Diehr} was whether a process for curing synthetic rubber that employed a mathematical formula and a digital computer was patentable.\textsuperscript{121} The process measured the temperature inside a mold containing the synthetic rubber to determine the necessary curing time.\textsuperscript{122} The temperature was continuously monitored by a computer which repeatedly recalculated the curing time using an algorithm that was well-known in the industry.\textsuperscript{123} When the proper length of time had elapsed, the computer signaled the device to open the mold.\textsuperscript{124} The applicant asserted that the continuous measuring of the time, the recalculation, and the signal to open the mold were new features to the process.\textsuperscript{125}

To determine patentability, the Court used \textit{Flook}'s holistic approach of examining the invention, warning that the addition of post-solution activity would not render an unpatentable algorithm patentable.\textsuperscript{126} Despite the use of the algorithm, the Court determined that the applicant's process qualified as a statutory

\textsuperscript{115} Id.
\textsuperscript{116} Id at 590-91.
\textsuperscript{117} \textit{Flook}, 437 US at 592.
\textsuperscript{118} Id at 592.
\textsuperscript{119} Id at 594-595.
\textsuperscript{120} 450 US 175 (1981).
\textsuperscript{121} Id at 177.
\textsuperscript{122} Id at 178-179.
\textsuperscript{123} Id at 177-178.
\textsuperscript{124} \textit{Diehr}, 450 US at 179.
\textsuperscript{125} Id.
\textsuperscript{126} Id at 191-192.
process because the synthetic rubber underwent a physical change. It noted that one characteristic of a process is the transformation of matter from one form into another, something that occurred in the applicant’s claim. The use of the algorithm was not fatal to the claim because the claim would not preempt alternative uses of the algorithm in other processes or for other purposes. Unlike in Flook, the Court found that the claim as a whole performed a patentable function and allowed the patent.


Established by the United States Court of Customs and Patent Appeals, the Freeman-Walter-Abele test is an alternative to Benson and its progeny. This series of cases was highly critical of the limited Benson definition of “algorithm” and the interpretation of the Benson cases by the Patent and Trademark Office that endorsed a “point of novelty” approach to inventions that include algorithms. The cases called for a narrower application of the Benson definition of algorithm and rejected the point of novelty approach for analysis of the invention as a whole.

The case that first articulated the test was Application of Freeman. The issue in Freeman was whether “a system for typesetting alphanumeric information” that included a “local posturing algorithm” was patentable. In the course of granting the claim, the court chastised the Patent and Trademark Office Board of Appeals (“the Board”) for presuming that the claim was unpatentable simply because it contained an algorithm, stating that the Board had interpreted Benson too broadly. It was also highly critical of the definition the Supreme Court gave to algorithms, noting that the term “algorithm” used in reference to computers is not equivalent to “mathematical

---

127 Id at 184.
128 Diehr, 450 US at 192.
129 Id at 187.
130 Id at 192.
131 The United States Court of Customs and Patent Appeals is now replaced by the United States Court of Appeals for the Federal Circuit.
132 See notes 130-52 and accompanying text.
133 573 F2d 1237 (CCPA 1978).
134 Id at 1238-40.
135 Id at 1244-45.
algorithm” as defined by Benson. The court gave its own test for patenting claims containing algorithms: 1) determine whether the claim recites an algorithm, and 2) determine whether the claim in its entirety wholly preempts the use of the algorithm.

The court in Application of Walter expanded the ideas presented in Freeman to further implement this holistic approach. The application in Walter involved a method to convert chirp signals used in seismic surveying into signals resembling those produced by the impulse wave method. The Court emphatically rejected the Board’s point of novelty approach, which it sought to bolster by claiming Flook as authority. “Under such an approach, an invention would be nonstatutory if the mathematical algorithm in the claim, as an embodiment of scientific truth, is at the ‘point of novelty’ of the claim.” The court felt that the ramifications of such a test would cripple the patent system, a result the Supreme Court presumably never meant to adopt. Instead, the court restated that, while a law of nature or a scientific truth is not patentable, an invention that applies a scientific truth or law of nature may be patentable if it improves a currently existing process. The application of a law of nature to an otherwise statutory process will not make the process unpatentable; rather, the court determined that the best approach was to look at the claim as a whole.

The Walter court used the second prong of the Freeman test—whether an invention, viewed in its entirety, wholly preempts the use of the algorithm for other purposes—to disallow the claim. In applying the test, the court held that one must look at the algorithm’s relation to the rest of the invention’s components. It further refined the test by adding the following condition:

---

136 Id at 1246.
137 Freeman, 573 F2d at 1245.
138 618 F2d 758 (CCPA 1980).
139 Id at 761-762.
140 Id at 766.
141 Id.
142 Walter, 618 F2d at 766.
143 Id.
144 Id at 768.
145 Id at 766-67.
146 Walter, 618 F2d at 767.
If it appears that the mathematical algorithm is implemented in a specific manner to define structural relationships between the physical element of the claim . . . or to refine or limit claim steps, . . . the claim being otherwise statutory, the claim passes muster under § 101 [of the Patent Act]. If, however, the mathematical algorithm is merely presented and solved by the claimed invention, . . . and is not applied in any manner to physical elements or process steps, no amount of post-solution activity will render the claim statutory; nor is it saved by a preamble merely reciting the field of use of the mathematical algorithm.147

The court, however, rejected the application, defining it as simply an improved method of correlating and cross-correlating mathematical functions, and not part of seismic prospecting.148

The third contributor to the test was In re Abele.149 The issue in that case was the patentability of several claims related to a Tomographic Scanner—a CAT scan.150 The court broadened Walter’s adaptation of Freeman:151

[The test] should be read as requiring no more than that the algorithm be “applied in any manner to physical elements or process steps,” provided that its application is circumscribed by more than a field of use limitation or non-essential post-solution activity.152

Using this test, the court rejected two claims and allowed the rest.153

The Freeman, Walter, and Abele courts felt that the Benson decision and its subsequent interpretations were too restrictive. The definition of algorithm embraced by the Supreme Court did not meet the needs and realities of the computer industry. It appears that the Freeman-Walter-Abele analysis is the preferred one today.154

---

147 Id.
148 Id at 769.
149 684 F2d 902 (CCPA 1982).
150 Id at 903.
151 Id at 907.
152 Id at 907.
153 Abele, 684 F2d at 903.
C. Copyright Law

The purpose of copyright law is to protect the interests of an author in his work while promoting the widespread dissemination of information.\textsuperscript{155} It does so by reserving to the author the right to create copies of the work while allowing others to use the work for research, personal use, and other uses provided for under the Copyright Act.\textsuperscript{156} This section will first explain the basics of copyright law, then it will examine copyright law as it applies to computer software.

1. A brief survey of copyright law.

Section 102 of Title 17 of the United States Code describes the works that qualify as subject matter that can be copyrighted.\textsuperscript{157} The statute states:

\begin{quote}
(a) Copyright protection subsists . . . in original works of authorship fixed in any tangible medium of expression, now known or later developed, from which they can be perceived, reproduced, or otherwise communicated, either directly or indirectly with the aid of a machine or device. Works of authorship include . . . literary works; . . .

(b) In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work.\textsuperscript{158}
\end{quote}

The statute identifies several characteristics of a potentially copyrightable work. It must: (1) be original,\textsuperscript{159} (2) be fixed in a tangible medium,\textsuperscript{160} and (3) not be an idea, procedure, or method.\textsuperscript{161} Each of these will be examined in turn.

The first requirement is that the work be an original work of authorship.\textsuperscript{162} To be original, a work must be independent, and
it must "contain [ ] a modicum of originality." Originality should not be confused with novelty. A work does not have to be totally unique; instead, it need only be different enough from similar items that the difference is noticeable. While the requirement of originality is loose, it is not meaningless. For example, one cannot change just a few lines in a book and then claim originality because of inconsequential changes. Originality is a minimal, yet necessary, standard.

The second criterion is fixation. The work must be fixed in a tangible medium of expression. A work is fixed when "its embodiment in a copy or phonorecord . . . is sufficiently permanent or stable to permit it to be perceived, reproduced, or otherwise communicated for a period of more than transitory duration." The marble out of which a statue is carved, the page upon which a poem is printed, and the paint and canvas of an impressionist landscape are all media in which a work may be fixed under the statute.

The purpose of the fixation standard is to ensure that the work is capable of being perceived. It is a standard which separates the transitory from the permanent. This distinction serves the Copyright Act's primary goal of promoting the addition of information to the general pool of knowledge and advances the public good. A work must endure, even for a short period, to receive protection under the Act.

Finally, ideas may not be copyrighted. The Copyright Act only protects the expression of an idea, not the idea itself. This doctrine was first articulated in Baker v Selden. In that case, the author of a treatise on bookkeeping maintained that he had a copyright on the method embodied in the forms contained

---

164 Herbert Rosenthal Jewelry Corp. v Kaipakian, 446 F2d 738, 740-41 (9th Cir 1971); Universal Athletic Sales Co. v Salkeld, 511 F2d 904, 908 (3d Cir 1975).
165 As Judge Learned Hand wrote: "It is of course essential to any protection of literary property, whether at common-law or under the statute, that the right cannot be limited literally to the text, else a plagiarist would escape by immaterial variations." Nichols v Universal Pictures Corp., 45 F2d 119, 121 (2d Cir 1930).
166 17 USC § 102(a).
167 17 USC § 101.
169 Sid & Marty Krofft Television Productions, Inc. v McDonald's Corp., 562 F2d 1157, 1170 (9th Cir 1977).
170 17 USC § 102(a).
171 17 USC § 102(b).
172 101 US 99 (1879).
in his book.\textsuperscript{173} The court stated that allowing a copyright on the method defeated the copyright law's purpose—the spread of knowledge.\textsuperscript{174} In a more recent case, \textit{Herbert Rosenthal Jewelry Corp. v Kalpakian},\textsuperscript{175} the Ninth Circuit held that where the idea and the expression of a work are inseparable, the expression is subordinated to the idea and no copyright is allowed.\textsuperscript{176}

2. \textit{A possible parallel: software copyright law.}

Given the similarities between algorithms and computer software programs, the development of the law of computer software may indicate the proper treatment of an algorithm under copyright law. Like an algorithm, computer software is utilitarian, a characteristic not protected under copyright law. Software and algorithms are also alike in that they are both a combination of creativity and mechanical development.

It is well established that a computer program may be copyrighted.\textsuperscript{177} The 1976 House Report concerning amendments to the Copyright Act stated that the term "literary work" includes "computer data bases, and computer programs to the extent that they incorporate authorship in the programmer's expression of original ideas, as distinguished from the ideas themselves."\textsuperscript{178} The adoption of the 1980 Amendment to the Copyright Act further demonstrated the acceptance of computer software as protected by copyright. The Computer Software Copyright Act of 1980\textsuperscript{179} was implemented on the recommendation of the National Commission on New Technological Uses of Copyright Works ("CONTU"), a commission appointed in 1974 to study the reproduction of copyrighted works and the creation of new works by means of computers.\textsuperscript{180} The 1980 Amendment defined a computer program as "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a cer-

\begin{itemize}
\item \textsuperscript{173} Id at 100.
\item \textsuperscript{174} Id at 103.
\item \textsuperscript{175} 446 F2d 738 (9th Cir 1971).
\item \textsuperscript{176} Id at 742.
\item \textsuperscript{177} \textit{Johnson Controls, Inc. v Phoenix Control Systems, Inc.}, 886 F2d 1173, 1175 (9th Cir 1989).
\item \textsuperscript{178} Copyright Act, HR Rep No 94-1476, 94th Cong, 2d Sess 54 (1976).
\item \textsuperscript{179} Pub L No 96-517, 94 Stat 3028 (1980), codified at 17 USC §§ 101, 117 (1994).
\item \textsuperscript{180} See Commission on New Technological Uses of Copyrighted Works, Hearings on HR 4836 before the Subcommittee on Courts, Civil Liberties, and the Administration of Justice of the House Committee on the Judiciary, 95th Cong, 1st Sess 1 (1977) (testimony of Arthur J. Levine, Executive Director of CONTU). 
\end{itemize}
tain result.” In addition, it revised the Copyright Act to allow copying of programs for archival purposes or copying as a required step in the installation process.

To receive protection, computer software must still meet the traditional copyright requirements. The first, originality, should be easy to satisfy as long as the programmer does not exactly copy the code and the user interface appearance of another program.

The second, fixation, has gone through a period of uncertainty. In 1976, due to the increased use of computers, Congress revised the Copyright Act to include important changes. Congress added a clause to Section 102 stating that the copyrighted work may be fixed in any tangible form from which it can be perceived, either directly or with the aid of a device, specifically including fixation in punched or magnetic form. Congress included this clause to repudiate the doctrine set forth in *White-Smith Music Publishing Co. v Apollo Co.*, a case that held that there was no copyright infringement where the defendant produced perforated rolls for player pianos that reproduced music copyrighted by the plaintiff. The Court reasoned that “the [copyright] statute has not provided for the protection of the intellectual conception [the music] apart from the thing produced . . . .” The current allowance for fixation in any form is important for computers because the forms of fixation of computer programs are evolving more quickly than the statutes.

For a brief time, there had been uncertainty about whether a Read Only Memory (“ROM”) chip qualified as an appropriate statutory medium of fixation. In *Williams Electronics, Inc. v Artic Intl, Inc.*, the court settled the uncertainty by holding that a ROM chip was a sufficient medium upon which to fix the program for the arcade game “Defender.” The defendant ar-

---

181 Computer Software Copyright Act of 1980, § 10(a), 94 Stat at 3028.
182 Id at 3028.
183 HR Rep No 94-1476 at 117 (cited in note 178).
184 Id at 52; see also 17 USC § 102.
185 209 US 1 (1908); HR Rep No 94-1476 at 52 (cited in note 178).
187 Id at 17.
188 For an account of the debate over a ROM as a medium upon which a program may be fixed, see Note, *Copyright Protection for Computer Programs in Read Only Memory*, 11 Hofstra L Rev 329 (1982).
189 685 F2d 870 (3d Cir 1982).
190 Id at 874. See also *Midway Manufacturing Co. v Strohon*, 564 F Supp 741, 751 (ND Ill 1983) (holding that a floppy disk is also a sufficient medium of fixation).
gued that since ROMs are utilitarian parts of a computer, they may not be protected by copyright.\textsuperscript{191} The court countered that the issue was whether the program embodied on the ROM chip was protectable.\textsuperscript{192} This suggests that the ROM was only the medium. The court determined, however, that copying a ROM chip was the functional equivalent of copying the program, and so such copying would be copyright infringement.\textsuperscript{193}

There was confusion early on about whether object code could receive copyright protection.\textsuperscript{194} The Copyright Act ("the Act") allows for the copyright of works that can only be perceived indirectly with the aid of a machine, an important inclusion for computer programs.\textsuperscript{195} The addition of the clause settled the uncertainty created by the difference between source code and object code.\textsuperscript{196} It is now acknowledged that the object code is a translation of the source code by the machine and is covered by the copyright as well.\textsuperscript{197}

D. Putting It All Together: What Can Software Copyright Law and Patent Law Tell Us About the Copyrighting of Algorithms?

This Part will first examine two theories that suggest how the courts may treat an algorithm: 1) the abstraction-filtration-comparison analysis, and 2) the structure-sequence-organization analysis. These theories attempt to guide the courts in separating the idea-related aspects of a work from its expressive elements. Second, this Part will discuss the possible authority of the patent law decisions in the treatment of algorithms under copyright law.

1. \textit{The abstraction-filtration-comparison test.}

The courts use the abstraction-filtration-comparison test to separate non-statutory material, such as ideas, from statutory material.\textsuperscript{198} As the name suggests, the test consists of three

\begin{itemize}
\item \textsuperscript{191} Id at 874.
\item \textsuperscript{192} Id at 874.
\item \textsuperscript{193} Id at 877.
\item \textsuperscript{195} 17 USC § 102(a).
\item \textsuperscript{196} See notes 24-26 and accompanying text.
\item \textsuperscript{197} See \textit{GCA Corp. v Chance}, 217 USPQ (BNA) 718, 720 (ND Cal 1982).
\item \textsuperscript{198} See, for example, \textit{Computer Assoc Intl, Inc. v Altai, Inc.}, 982 F2d 693, 706 (2d Cir 1982).
\end{itemize}
parts: abstraction, filtration, and comparison. The test separates ideas and expressions in the public domain from copyrightable expressions by,

break[ing] down the ... program into its constituent structural parts[,] ... examining each of these parts for such things as incorporated ideas, expression that is necessarily incidental to those ideas, and elements that are taken from the public domain ...

The court removes this uncopyrightable material, then compares what is left to the allegedly infringing program. This comparison step is applicable only to copyright infringement and is thus not relevant to this Comment.

It is helpful to look at the abstractions test in a context outside of computer software. In Nichols v Universal Pictures Corp., the issue was whether there was copyright infringement of a play. The court observed that a play, like other works, contains levels of abstraction of its elements. For example, the court stated that a stock character, like "a vain and foppish steward who becomes amorous of his mistress," is so abstract and nondistinct that it is an idea, not an expression. The court acknowledged that while it is difficult to draw the line between idea and expression, one guideline is to focus on the extent to which an author develops a character or element. The less defined or developed an element is, the less likely the court will find the element to be copyrightable. Using this guideline, the Nichols court did not find copyright infringement.

The abstractions test was adapted for computer software in Gates Rubber Co. v Bando Chemical Industries, Ltd. In Gates, the court identified six levels of abstraction for computer programs in descending order of abstraction: "(i) the main purpose, (ii) the program structure or architecture, (iii) modules, (iv)

---

199 Id at 706-12.
200 Id at 706.
201 Id.
202 45 F2d 119 (2d Cir 1930).
203 Id at 120.
204 Id at 121.
205 Id (referring to the character Malvolio from Shakespeare's Twelfth Night).
206 Nichols, 45 F2d at 121.
207 Id.
208 Id at 123.
209 9 F3d 823 (10th Cir 1993).
algorithms and data structures, (v) source code, and (vi) object code. The court also noted that the main purpose is always an unprotectable idea, the modules are almost always unprotectable, and the source and object code are almost always protected.

Copyrightability of the intermediate components is uncertain, fact specific, and depends upon whether an idea or an expression is presented. The court defined an algorithm, one of these intermediate components, as "a specific series of steps that accomplish a particular operation." This is a markedly different definition from the one given in Benson and leaves open the possibility of copyrightability.

2. Structure, sequence, and organization.

The "nonliteral" elements of a computer program include its structure, sequence, and organization. The literal elements are the source code and the object code. They are unquestionably copyrightable. Nonliteral elements may be copyrightable if they protect an expression rather than an idea.

In Whelan Associates, Inc. v Jaslow Dental Laboratory, Inc., the program at issue performed various administrative and record-keeping functions for a dental laboratory. The court addressed "whether the structure (or sequence or organization) of a computer program is protectible by copyright, or whether the protection of the copyright law extends only as far as the literal computer code." The Third Circuit rejected the assertion that a structure was, by definition, an idea. Instead, the test adopted was that "[w]here there are various means of achieving the desired purpose, then the particular means chosen is not necessary to the purpose; hence there is expression, not idea."

---

210 The Court noted, however, that "[t]hese generalized levels of abstraction will not, of course, fit all computer codes." Id at 835.
211 Id at 836.
212 Id.
213 Gates Rubber, 9 F3d at 835.
214 Gottschalk v Benson, 409 US 63, 65 (1972) (defining an algorithm as a "procedure for solving a given type of mathematical problem").
215 Johnson Controls, 886 F2d at 1175.
216 Id.
217 Id.
218 Id.
219 797 F2d 1222 (3d Cir 1986).
220 Id at 1224 (footnote omitted).
221 Id at 1235-37.
222 Id at 1236.
It determined that the structure of the program was not essential to the performance of administrative functions in the lab and that other differently structured programs could perform the same tasks. Copyright law could protect the structure as an expression of only one of many ways to reach a given result.

While the court did not explicitly so state, its description of the development and structure of the program left open the possibility that an algorithm can be protected. The expression of an algorithm, such as the flowchart discussed in Part II, corresponds to the Court’s description of the development of the structure of a computer program. This theory may allow copyrighting of algorithms if the differing terms can be reconciled. Other courts, however, have criticized the Whelan decision for being too broad and inclusive and have refused to follow it.

3. Are patent law concepts applicable to copyright law?

As stated earlier, patent law and copyright law are distinct. When an object is protected by both patent and copyright law, different theories protect different portions of the work. For example, the inventor of a lamp in the shape of a woman protects the “lampness aspect” under patent law and the “woman decoration” under copyright law. It stands to reason that conclusions in the patent law cases do not necessarily apply to copyright law since the statutes have different purposes and requirements.

At least one copyright case, however, has cited a patent law case as authority for the definition of “algorithm.” Synercom Technology, Inc. v University Computing Co., held that the plaintiff’s copyright in a computer manual was infringed. It cited Benson’s definition of an algorithm as a procedure for solving a mathematical problem. The court did not, however, further explore the definition, so the extent of its application to copyright law is unclear.

223 Whelan, 797 F2d at 1238.
224 Id at 1236.
225 Id at 1229-31.
226 Altai, 982 F2d at 705.
229 Id at 1014-15.
230 Id at 1005.
One implication of the acceptance of the Benson definition is that no algorithm could ever be copyrighted because it would be an idea, not an expression. The patent concept of law of nature, which an algorithm is classified as, is similar to the copyright concept of “idea.” Patent law analysis could inform an analysis of a copyright claim, but the courts would be wise to limit the applicability of such concepts and authority.

4. In summary.

An algorithm may be copyrightable under the abstraction-filtration-comparison analysis or the structure-sequence-organization analysis. Neither has explicitly stated that an algorithm would always be copyrightable, but neither entirely forestalls the possibility. The main barrier to the copyrightability of an algorithm is the idea/expression dichotomy. Whelan generously extended the reach of expression, suggesting that an algorithm may be copyrightable, but the decision has been criticized and rarely followed. In order for algorithms to be copyrightable, the applicant must convincingly describe them as expressions, not ideas. The above theories were extended to computer technology relatively recently so the courts have not yet made a solid determination one way or the other.

III. HOW AN ENCRYPTION ALGORITHM AND METHOD SHOULD BE TREATED

A. Copyright law is Preferable to Patent Law for Algorithms and Encryption

Copyright law is preferable to patent law for the protection of algorithms for two reasons. First, an algorithm has been held nonpatentable. Unless the courts change this conclusion, inventors apply for algorithm patents at their own risk. Courts in copyright cases have not been so explicit in rejecting algorithms, leaving open possible flexibility in the law’s application. Second, copyright law is not nearly as restrictive as patent law on others’ use of the protected work. While the inventor may prefer the most restrictive power possible, the community benefits more by allowing others to utilize the work; allowing twenty years of monopoly control over a piece of technology can significantly impede desirable innovation. Copyright law permits the use of

231 See Altai, 982 F2d at 705.
the algorithm, making possible technological evolution, while
providing a measure of compensation to the creator.

B. Copyrightability of the Cryptosystem and the Underlying
Algorithm

There are three categories of encryption algorithms and their
cryptosystems: 1) encryption programs, 2) encryption algorithms
that are nonmathematical, and 3) encryption algorithms that are
mathematical. The principles discussed in the software law
and patent law Parts can help in the analysis of the preferred
treatment of each of these three categories.

The first category contains those cryptosystems that are
clearly programs, such as PGP. These should be judged according
to the developed law on software copyrightability. The law re-
garding software, however, only directly applies to the package,
the cryptosystem. Whether the underlying algorithm should be
independently copyrightable depends on the characterization of
the algorithm according to the remaining categories.

The second category contains nonmathematical algorithms as
defined in Part II. If there are several ways of describing the
solution embodied in the algorithm, it could be likened to the
structures in Whelan. A remaining problem is the algorithm's
specificity. The abstraction-filtration-comparison provides one
option of separating overly abstract algorithms from more specific
ones. A nonmathematical encryption algorithm should be eval-
uated in the same manner as a general algorithm.

The third category contains purely mathematical algorithms,
as defined in Benson. If the algorithm in an encryption sys-
tem is purely mathematical, it should not be copyrightable. A
mathematical formula is an idea, not an expression. Even if the
relation between the variables was previously unknown, it is
logical to assume that the relationship existed independently of
the formulator's description, like the law of nature doctrine
above. If there are, however, several practicable ways of describ-
ing a problem and solution in mathematical terms, it may be pos-
sible to convince a court that the particular algorithm is only one
of many expressions and not an idea. This inquiry would neces-
sarily be fact specific.

232 PGP is an encryption program, whereas IDEA and RSA are encryption algorithms.
233 Gottschalk v Benson, 409 US 63, 65 (1972).
The copyright courts may adopt a test similar to the *Free-
man-Walter-Abele* test to determine whether the mathematical
algorithm constitutes the entire work; if it does, then it should
not be copyrightable. If, however, the algorithm is part of an
otherwise copyrightable program, the presence of the algorithm
should not bar the copyrighting of the cryptosystem. While the
program is copyrightable, the underlying algorithm alone is not.

**CONCLUSION**

While protection for algorithms has traditionally been sought
under the patent laws, copyright law is a better alternative.
Copyright law attempts to give the author some interest in his
work, yet still allow the public to use the item for the communal
good. Given the rapid changes in computer technology, barring
others from using the encryption algorithm would be a social
disservice. The copyright law as it applies to computer programs
may indicate how an encryption algorithm should be treated
under copyright law. The copyright law is especially useful in its
resolution of the distinction between idea and expression of
nonliteral computer elements. It is important to note, though,
that algorithms come in many forms and degrees of sophistica-
tion, so one solution will not apply to all.