Copyright Protection of Object Code Computer Programs: Can Courts Determine Copying

Deborah Ledsinger
Copyright Protection of Object Code
Computer Programs: Can Courts Determine Copying?

by DEBORAH LEDSINGER*

I
Introduction

Our nation's libraries house thousands of volumes chronicling the effects of inexpensive microcomputers on our workplaces, schools, and daily lives. Sociologists debate the merits of this technological innovation, while economists hail the rise of software and semiconductor manufacturing as a source of clean jobs. The law has struggled to keep pace with these rapid developments. Scholarly literature on the law and its relationship to emerging technology reflects a tension between the incorporation of technological developments into existing constructs, and the creation of new bodies of law to account for the computer industry's special needs.

With the overwhelming success of entrepreneurial computer pioneers such as Apple co-founder Steve Wozniak, computer engineers are viewed as unusually creative and admirable souls

* Member, Third Year Class. The author wishes to dedicate this note to the memory of her grandfather, James C. McBaine.


whose proprietary interests deserve special protection.\(^5\) Intellectual property concepts protect proprietary interests through several avenues, including patent, trade secret, and contract law. However, Congress has focused its efforts on incorporating software technology into existing copyright theories.

In 1976, Congress overhauled the Copyright Act to emphasize that copyright subsists in "original works of authorship fixed in any tangible medium of expression, now known or later developed from which they can be perceived . . . either directly or with the aid of a machine or device."\(^6\) While the 1976 Act was pending, Congress created CONTU, the Commission on New Technological Uses of Copyrighted Works,\(^7\) to further study copyright problems raised by computer development. CONTU subsequently proposed that the Act "should be amended . . . to make it explicit that computer programs, to the extent they embody an author's original creation, are proper subject matter of copyright."\(^8\) The Commission suggested that the Act should include a definition of computer program, and authorize the making of a backup copy of each piece of copyrighted software.\(^9\) Congress adopted CONTU's suggestions verbatim in 1980.\(^10\)

Courts have continued the process of incorporating considerations unique to computer technology into existing modes of copyright protection in interpreting the amended Copyright Act. Until recently, uncertainty remained as to whether programs expressed in object code\(^11\) (such as microcomputer systems software) fell within the ambit of copyright protection. In *Apple Computer Co. v. Franklin Computer, Inc.*,\(^12\) the Court of

\(^5\) For an intriguing look at the people behind computer design, see T. KIDDER, *THE SOUL OF A NEW MACHINE* (1981).
\(^8\) NATIONAL COMMISSION ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS, FINAL REPORT 2 (1978) [hereinafter CONTU Report].
\(^9\) Id. at 30.
\(^11\) Object code is distinguished from source code. Source code describes the level of language in which a programmer inputs, reads, and edits program instructions. Source code is easily comprehensible to appropriately trained persons, such as programmers. Object code, on the other hand, describes the coding of a sequence of electronic signals (instructions), which interact with the computer's circuitry to produce a desired result. See Conley & Bryan, *A Unifying Theory for the Litigation of Computer Software Copyright Cases*, 63 N.C.L. REV. 563, 565-67 (1985).
Appeals for the Third Circuit held that these programs are included within the scope of protected literary works.\textsuperscript{13}

After the Third Circuit’s 1983 decision in \textit{Apple}, some commentators declared the software copyright issue resolved.\textsuperscript{14} However, the copyright approach, which seemed like a godsend when applied to a case of undisputed copying in \textit{Apple}, has serious drawbacks for other cases because courts lack a workable standard for determining when infringement of copyrighted material in computer programs has occurred.\textsuperscript{15}

This note examines these drawbacks, which result in part from lack of judicial understanding of computer technology.\textsuperscript{16} It focuses on one trial court’s approach to a complex set of facts involving disputed, non-verbatim copying in \textit{SAS Institute, Inc. v. S&H Computer Systems}.\textsuperscript{17} Section II outlines the basic technical distinctions essential to understanding the issues discussed in this note. Section III provides a comparative analysis of the \textit{Apple} and \textit{SAS Institute} cases. Section IV addresses the difficulties inherent in comparing object code works and discusses problems with the \textit{SAS Institute} approach. In section V, the author argues that the \textit{SAS Institute} approach used for determining copyright infringement is misguided, and concludes by suggesting alternative solutions to the object code copyright dilemma.

\section*{II
A Non-Technical Primer}

The essence of the machine called a computer is a set of minute and extremely complex circuits called the Central Processing Unit (CPU).\textsuperscript{18} The CPU performs only the work computer programs, or software,\textsuperscript{19} instruct it to do.\textsuperscript{20} While a program may be expressed to humans in written terms, to the CPU it is but a series of magnetic charges which react with the com-

\begin{footnotesize}
\begin{enumerate}
\item[13.] Id. at 1249.
\item[14.] See, e.g., supra note 4.
\item[15.] Note, supra note 3, at 1264.
\item[16.] Id. at 1266.
\item[17.] 605 F. Supp. 816 (M.D. Tenn. 1985).
\item[18.] \textit{Apple}, 714 F.2d at 1243.
\item[19.] Programs, or “software,” are distinguished from the machine parts and circuitry of the computer itself, or “hardware.”
\item[20.] \textit{Apple}, 714 F.2d at 1243.
\end{enumerate}
\end{footnotesize}
puter's circuitry in a designated sequence to produce a desired result called output.

In designing software, the programmer is confronted with a problem-solving challenge: he must get a machine to perform a certain task through a complex series of logical steps. The designer may draw flow charts, perform computations, or make notes to facilitate the programming process. This material is called documentation.

A computer program may be written in any of three levels of language. Most programmers begin with a high-level language such as BASIC or FORTRAN because these languages employ simple English-like statements, and combine several CPU steps into one programming step. Alternatively, the programmer may use an intermediate-level language called assembly which uses alphanumeric labels. Either of these levels may be referred to as source code. Source code may be loaded into a computer and textually edited, much as documents are treated in word processing. Source code is distinguished from object code, the lowest level computer language, also called machine code. While source code is easily comprehensible to appropriately trained human beings, object code is expressed in binary language, a series of 0's and 1's to indicate open or closed switches. Object code is comprehensible only to skilled programmers assisted by computers and appropriate software. The CPU can only follow instructions expressed in object code. Programs are usually written in source code by a separate program called a compiler.

Computer programs are of two general types. Operating system programs, expressed in object code, manage internal com-

22. Apple, 714 F.2d at 1243.
23. These languages are so called both because they are similar to English lexicon, and because they are farthest ("highest") from the level used by the computer. Id.
24. Id.
25. For example, the BASIC statement "Print," which directs the computer to display a series of characters, requires the use of several steps at the object code level.
26. Apple, 714 F.2d at 1243. Assembly language enables the programmer to write machine-level code using convenient mnemonic instruction.
27. SAS Institute, 605 F. Supp. at 818.
28. Apple, 714 F.2d at 1243.
29. Id.
30. Id.
31. Id.
32. Id.
33. Id.
puter functions and facilitate the use of applications programs.\textsuperscript{34} Systems software is often embedded on semi-conductor chips called ROMs and directly installed into the computer’s circuitry.\textsuperscript{35} Applications programs perform specific tasks for the user; they are what make computers useful for balancing checkbooks or playing games.\textsuperscript{36} Applications programs are generally distributed in their object code version stored on a magnetic memory device such as a floppy disk.

Different brands or models of computers which employ nearly identical CPUs, and which can therefore execute the same object code programs interchangeably, are said to be compatible.\textsuperscript{37} With the exception of these compatibles, most computer models utilize somewhat different processor circuitry. Consequently, the sequence of magnetic signals (called the object code) needed to perform a given task on the IBM PC, for example, is completely different from the code required to execute the identical task on the Apple II. In order to use software designed for the IBM on the Apple, a programmer must “read” the IBM program (perhaps from a pre-compiler source code listing) to identify its contents, make substantial changes to compensate for circuitry differences between the two machines, and compile the new source code specifically for the Apple computer.

III
Comparing the Cases

A. Apple Computer Co. v. Franklin Computer, Inc.

Shortly after Apple Computer Company began marketing its popular Apple II\textsuperscript{38} model home computer, Franklin Computer Corporation, a comparatively small operation,\textsuperscript{39} introduced the ACE 100. Franklin designed and advertised the ACE 100 as fully compatible\textsuperscript{40} with the Apple II. Apple programmers ex-
examined several programs sold with the ACE 100, and found them nearly identical to programs marketed by Apple, with only minor modifications apparently intended to disguise the fact that Franklin's programs were copies.

Apple sought a preliminary injunction restraining Franklin from further distribution of the operating system software with the ACE 100. Apple alleged that Franklin's programs were duplicates of Apple's own system software, and that their sale infringed Apple's copyright. Franklin admitted the copying and focused its defense on legal issues such as the copyrightability of the disputed programs. The district court accepted Franklin's arguments and denied injunctive relief because it concluded, *inter alia,* that there was "some doubt as to the copyrightability of the programs described in this litigation."

On appeal, the Third Circuit reconsidered the three copyright challenges Franklin had presented in the lower court. Citing the Copyright Act's requirement that a work be fixed in order for the ACE 100 to do this, the programs it used to regulate the functions of its electronic parts—operating system software—had to be nearly identical to Apple's.

The technical reasons for this necessity are outlined in *Apple.* *Id.* at 1245.

41. *Apple* ultimately alleged infringement of 14 separate programs, some of which were embedded on ROM chips, while others were recorded on magnetic disks. *Id.* at 1244 n.4.

42. *Id.* at 1245. "The variations that did exist were minor, consisting merely of such things as deletion of reference to Apple or its copyright notice." *Id.*


44. "Computer programs can be categorized by function as either application programs or operating system programs. Application programs usually perform a specific task for the computer user, such as word processing, checkbook balancing, or playing a game. In contrast, operating system programs generally manage the internal functions of the computer or facilitate use of application programs." *Apple*, 714 F.2d at 1243.

45. *Id.* at 1245. As its sole factual defense, Franklin argued that directly copying Apple's own operating system programs was the only method of ensuring the ACE 100's full compatibility with the Apple II, and that it would have been cost-prohibitive for Franklin to write its own programs. *Id.* The court rejected this theory as an excuse for infringement. *Id.* at 1253.

46. *Id.* at 1245.


48. The district court also held that Apple had failed to show that irreparable harm would result if injunctive relief were not granted. *Id.* See also *id.* at 825.

49. *Id.* at 812.

50. "We read the district court opinion as presenting the following legal issues: (1) whether copyright can exist in a computer program expressed in object code, (2) whether copyright can exist in a computer program embedded on a ROM, [and] (3)
some "tangible medium of expression," the court reaffirmed its holding in Williams Electronics, Inc. v. Artic International, Inc., which extended copyright protection to programs embedded on ROM chips. It went on to consider as a matter of first impression Franklin's contention that operating system programs could not be the proper subject of copyright. The court rejected this argument as inconsistent with Congressional intent in drafting the 1976 Copyright Act.

Together with an analysis of the 1976 and 1980 amendments to the Copyright Act and its decision in Williams Electronics, the court addressed the copyrightability of object code programs. Copyright protection, it noted, subsists only in original works of authorship. A work of authorship must be either a literary work, or a member of one of six other statutory categories. The court held that because the statutory definition of literary works includes works expressed in "words, numbers, or other verbal numerical symbols or indicia," object code programs are literary works. Since the amended Copyright Act protects works which can be perceived either directly or with the aid of a machine or device, the fact that object code can be deciphered only by experienced programmers aided by computers is no bar to its copyrightability. Furthermore,

whether copyright can exist in an operating system program. . ." Apple, 714 F.2d at 1246.

52. 685 F.2d 870 (3d Cir. 1982).
53. Apple, 714 F.2d at 1249 (citing Williams, 685 F.2d at 874, 876).
54. Apple, 714 F.2d at 1249, 1250.
55. Id. at 1253, 1254.
56. "Programs should no more be considered machine parts than videotapes should be considered parts of projectors or phonorecords parts of sound reproduction equipment." Id. at 1251 (citing CONTU Report, supra note 8, at 21).

The CONTU statement represents a good example of analogizing new technology to the old in order to make it comprehensible to judges and legislators. 57. Id. at 1246-49.
58. Id. (citing 17 U.S.C. § 102(a) (1982)).
59. Id. at 1249 (citing 17 U.S.C. § 101 (1982)).
61. An object code program may be expressed in binary numerical form, or in a mathematical notation called hexadecimal. Apple, 714 F.2d at 1243. For descriptions of computer operations, see Note, Copyright Protection of Computer Program Object Code, 96 HARV. L. REV. 1723 (1983).
62. Apple, 714 F.2d at 1249. The court cites the legislative history behind the 1976 Act, H.R. Rep. No. 1476, 94th Cong., 2d Sess. 54, as well as the CONTU Report (which spurred the 1990 amendments), as supporting this proposition. Id.
63. 714 F.2d at 1247 (citing 17 U.S.C. § 102(a) (1982)).
64. See supra notes 29-33 and accompanying text.
since the Act as amended in 1980 defines a protected program as a set of statements to be used either directly or indirectly to bring about a desired result, the fact that object code programs such as those in *Apple* are used to control a machine’s basic functions does not exclude them from copyright protection.

The court therefore held that a computer program, whether in object or source code, is a “literary work” and is protected from unauthorized copying whether from its object or source code version. Finding that the district court’s decision to deny Apple’s motion for a preliminary injunction was influenced by “an erroneous view of the availability of copyright for operating system programs and unnecessary concerns about object code and ROMs,” the court of appeals reversed and remanded the case for reconsideration.

The holding in *Apple* did not emerge from a legal, technological, or economic vacuum. Many scholars have cited the inadequacy of other forms of intellectual property law to protect entrepreneurial interests in the fast-paced and only recently mass-marketed computer industry. Some commentators anticipated the circumstances encountered by Apple and urged

---

65. *Apple*, 714 F.2d at 1248.
66. “We considered the issue of copyright protection for a computer program in [Williams Electronics] and concluded that the copyrightability of computer programs is firmly established after the 1980 amendment to the Copyright Act.” *Id.* (quoting *Williams Electronics*, 685 F.2d at 875).
67. *Id.* at 1248. “As source code instructions must be translated into object code before the computer can act upon them, only instructions expressed in object code can be used ‘directly’ by the computer.” *Id.* Congress’ inclusion of “directly” in the statute therefore suggests its intent to include object code programs.
68. *Id.* at 1249.
69. *Id.* at 1254.
70. *Id.*
71. The court of appeals received three amicus curiae briefs in support of Apple’s position, two from software manufacturers and one from a computer services trade association. *Id.* at 1242 n.1.

Moreover, trade secret protection, for example, is available only to actual secrets; an item cannot be “self-disclosing” and widely available and still qualify for trade secret protection. Since most computer programs “disclose” their contents to the eye of the experienced programmer, mass-marketing hampers the effectiveness of relying on trade secret law. See Gilburne & Johnston, *Trade Secret Protection for Software Generally and in the Mass Market*, 3 COMPUTER L.J. 211, 227-37 (1982). The Copyright Act, when applicable, may preempt state trade secret law. See, e.g., Note, *The
Congress and the courts to afford ROM-based object code programs copyright protection.\textsuperscript{73} The ostensible impact of the \textit{Apple} holding was welcomed as leading to greater protection for computer works, thereby protecting programmers, fostering innovation, and furthering the aims of intellectual property law.\textsuperscript{74} However, the court devoted little attention to the ability of courts to enforce \textit{Apple} in practice.

\textit{Apple} concerned the mechanical verbatim copying of Apple’s object code software by a competitor who marketed an Apple II compatible—the Franklin ACE 100.\textsuperscript{75} \textit{SAS Institute}, on the other hand, involved allegations of infringement resulting from S&H’s attempts to adapt copyrighted IBM software to operate on an incompatible computer, the VAX built by Digital Equipment Company.\textsuperscript{76}

\textit{Apple} held that even a hard-to-decipher object code operating systems program is copyrightable. It left the trial court in \textit{SAS Institute} to resolve the practical question: how can a court as fact finder determine whether copying has taken place in the nearly incomprehensible realm of object code software?\textsuperscript{77}

B. \textit{SAS Institute, Inc. v. S&H Computer Systems}\textsuperscript{78}

\textit{SAS Institute} held that because the defendant used source code and documentation provided with plaintiff’s object code program to develop its computer package, the defendant’s program constituted an unauthorized derivative work,\textsuperscript{79} and infringed plaintiff SAS Institute’s copyright over its object code program.\textsuperscript{80}


\textsuperscript{75} \textit{Apple}, 714 F.2d at 1244-45.

\textsuperscript{76} \textit{SAS Institute}, 605 F. Supp. at 818.

\textsuperscript{77} For an insight into this realm, see Comment, \textit{Copyright Protection of Systems Control Software Stored in Read Only Memory Chips: Into the World of Gulliver’s Travels}, 33 BUFFALO L. REV. 193 (1984). One author actually predicted that courts would next be faced with the problem of determining when copying had taken place in cases alleging non-verbatim copying of object code programs. See McCully, \textit{supra} note 74, at 185.

\textsuperscript{78} 605 F. Supp. 816 (M.D. Tenn. 1985).

\textsuperscript{79} \textit{id}. at 831, 832.

\textsuperscript{80} \textit{id}. at 831-33.
SAS Institute copyrighted and widely marketed\textsuperscript{81} an integrated statistics analysis program for use with IBM computers.\textsuperscript{82} The Institute developed the program, known as the Statistical Analysis System (SAS),\textsuperscript{83} at substantial expense.\textsuperscript{84} It did not sell copies of the program, but instead licensed its use through individual contracts with customers.\textsuperscript{85}

In 1981, a Vanderbilt University professor engaged in complex biostatistics research investigated the possibility of licensing SAS to perform sophisticated computations involved in his work.\textsuperscript{86} After learning that SAS was available only for IBM and IBM-compatible computers,\textsuperscript{87} the professor contacted two faculty members and an employee of the university's computing center who was also employed by defendant S&H Computer Systems.\textsuperscript{88} The men considered designing an original, state-of-the-art statistics package comparable to SAS to operate on the university's VAX computers.\textsuperscript{89} After determining that such a package would be difficult to market compared with a package modeled after SAS,\textsuperscript{90} the group decided instead to develop a VAX adaptation of SAS.\textsuperscript{91}

Immediately thereafter, S&H obtained a license to use SAS\textsuperscript{92} with the primary purpose, the Institute later contended, of "obtain[ing] detailed technical information not otherwise available" for use in preparing its VAX version of SAS.\textsuperscript{93} S&H received the SAS distribution package, which included a magnetic tape containing both the entire object code SAS program and the source code for most of the program, including the crit-
ical portions which performed the various statistical analyses. The SAS licensing contract provided that S&H would use SAS only on a specified computer, would not make any copies of SAS, and would not modify SAS code beyond certain permissible, user-addressable sections. S&H programmers loaded the source code into the VAX computer. They then used a text editor to view and manipulate the code. S&H then developed a VAX-version object code program called INDAS to perform complex statistical analysis.

SAS Institute brought an action for injunctive relief against S&H to prevent distribution of the VAX program claiming, inter alia, copyright infringement and breach of contract relating to the SAS licensing agreement. After a hearing, the court granted partial summary judgment upholding the Institute's copyright on SAS, but denied full summary judgment on the copyright claim, citing genuine issues of material fact as to the existence of copying. The court proceeded to try the case without a jury, rendering a verdict for the plaintiff on both the contract and infringement claims.

In upholding SAS Institute's copyright against legal challenge, the court cited Apple's statutory interpretation of the Copyright Act. While Apple was not the first case to address the object/source code distinction, Apple did reaffirm Williams Electronics. The fact that Apple did not reach the Supreme Court may be a signal to trial courts that the Apple court's interpretation of the Act will remain good law for quite

94. Id. at 821.
95. Id.
96. A text editor is a separate program which permits programmers to edit source code by altering alphanumeric characters, much like a word processor.
97. SAS Institute, 605 F. Supp. at 822.
98. The Institute also claimed fraud, trade secret misappropriation, and unfair competition. Id. at 816.
99. The license forbade S&H from copying the program or using it on any computer other than the one specified in the contract; S&H did both of these things. Id. at 821.
101. Id. at 424.
102. SAS Institute, 605 F. Supp. at 817.
103. Id. at 817, 818. Having granted relief, the court declined to address the Institute's fraud, unfair competition, and misappropriation claims. Id.
104. Id. at 829.
105. Williams Electronics, Inc. v. Artic Int'l, Inc., 685 F.2d 870, 870 (3rd Cir. 1982).
106. Apple, 714 F.2d at 1249.
some time.\textsuperscript{108}

Particularly noteworthy in the case of \textit{SAS Institute} was the language of the \textit{Apple} holding stating that a program is protected from unauthorized copying "whether from its object or source code version."\textsuperscript{109} \textit{SAS Institute} involves not the mechanical duplication of object code programs as in \textit{Apple},\textsuperscript{110} but rather allegations that S&H infringed the copyright of an object code program by employing the source code provided with the program to translate SAS for use on an incompatible computer.\textsuperscript{111}

With the copyrightability of object code software resolved in \textit{Apple},\textsuperscript{112} the primary copyright issue confronting the court in \textit{SAS Institute} became the factual problem of determining whether copying, and therefore infringement, had occurred.\textsuperscript{113}

\section*{C. Distinguishing the Cases}

The facts of \textit{SAS Institute} may be distinguished from those of \textit{Apple} in many respects. The disputed programs in \textit{Apple} constituted operating systems software,\textsuperscript{114} while \textit{SAS Institute} involved an applications program.\textsuperscript{115} We have seen, however, that \textit{Apple} itself obviated the legal significance of this distinction.\textsuperscript{116} Franklin's conduct in pirating Apple's software was relatively effortless,\textsuperscript{117} while S&H's conduct in writing a VAX-compatible version of SAS required at least several weeks of work.\textsuperscript{118} Indeed, this distinction seems to go unnoticed in the \textit{SAS Institute} analysis, although one might well argue it should have been a significant consideration.\textsuperscript{119}

\textsuperscript{108} Congress could, of course, again amend the Copyright Act, or create a separate preemptive body of law directed at computer software. \textit{See}, e.g., Comment, \textit{Softright: A Legislative Solution to the Problem of Users' and Producers' Rights in Computer Software}, 44 LA. L. REV. 1413 (1984).

\textsuperscript{109} \textit{Apple}, 714 F.2d at 1249.

\textsuperscript{110} Id. at 1245.

\textsuperscript{111} \textit{SAS Institute}, 605 F. Supp. at 821.

\textsuperscript{112} Id. at 829.

\textsuperscript{113} \textit{Supra} note 101 and accompanying text.

\textsuperscript{114} \textit{Apple}, 714 F.2d at 1243-44.

\textsuperscript{115} The SAS package ran in conjunction with IBM systems software and was therefore an applications program. \textit{See supra} note 35 and accompanying text.

\textsuperscript{116} \textit{Apple}, 714 F.2d at 1251-52.

\textsuperscript{117} Id. at 1245.

\textsuperscript{118} \textit{SAS Institute}, 605 F. Supp. at 821, 822.

\textsuperscript{119} The very fact that S&H attempted to adapt SAS rather than merely duplicate or "pirate" it suggests that S&H invested substantial programming expertise in modifying the Institute's original IBM program code. Logically, triers of fact should con-
Apple differs from SAS Institute in two other crucial respects. Most importantly, in Apple copying was undisputed, while in SAS Institute its existence was the central issue before the court. In addition, Apple involved verbatim copying; the court in SAS Institute did not consider whether S&H's final product was an exact copy of SAS, but rather, it considered whether S&H created an original product or an infringing derivative work. Consequently, the Apple decision left unresolved two crucial issues facing the court in SAS Institute, issues which may plague trial courts in Apple's wake. First, while object code is now a proper subject of copyright law, how is a court to compare works expressed in this nearly imperceptible binary language of object code? Second, in cases of non-verbatim copying, what degree of similarity should a court regard as "substantial" in determining infringement of computer programs?

IV
Determining Copying

It is a rare case where a copyright owner can use direct evidence to prove that the defendant duplicated all or part of his work. Usually, copying must be shown circumstantially. Courts will infer copying where the copyright owner can show: (1) the alleged infringer's access to the copyrighted work; and (2) a "substantial similarity" between the two works.

The substantial similarity test requires the trier of fact to lay the two works side by side and compare them in an effort to determine whether, in the judgment of a lay person, the second

sider the amount of effort that went into a work in determining whether it was an original or a derivative work. This issue is discussed at length in section IV.

120. Apple, 714 F.2d at 1245.
121. Id.
122. SAS Institute, 605 F. Supp. at 828.
123. Id.
124. Direct evidence of copying might consist of a witness' testimony that he observed the defendant load the copyrighted program into a computer and, after a few simple keystrokes, the defendant removed a disk later used to execute the copyrighted program. "Because pirates are unlikely to be obvious about their copying, proof of the direct use of the copyrighted work in preparing a copy is virtually impossible." Note, supra note 3, at 1276.
125. Id.
126. Id. at 1278-79; Sid & Marty Krofft Television Prods., Inc. v. McDonald's Corp., 562 F.2d 1157 (9th Cir. 1977).
appears to be all or in part a copy of the first. While this task
is ordinarily simple in the case of a prose work, the process of
comparing two computer programs expressed in object code is
inherently problematic. Object code is not perceivable to the
naked eye as the test contemplates, but is decipherable only by
properly trained programmers aided by machines. The first
hurdle courts must overcome in applying the substantial simi-
larity test after Apple, then, is learning to “read” object code.

Apple did not address this obstacle because Franklin did not
dispute copying. In SAS Institute, defendant S&H conceded
only that it licensed the copyrighted object code SAS program,
examined the source code and documentation provided with
the program, and executed comparison test runs of SAS in
preparing the VAX program it called INDAS. S&H argued
adamantly that the final object code INDAS program was so
dissimilar to SAS as to be an original work of authorship.

Rather than confront the comparison problem directly, the
court in SAS Institute skirted the central factual issue raised
by S&H—the question of similarity. At the Institute’s urg-
ing, the court adopted an analysis focusing on: (1) a compari-
on between the SAS source code and the source code from
INDAS in its “earliest stages of development,” rather than the
final published version; (2) the conduct of S&H’s program-
ners and other agents; and (3) the basic similarities of struc-
ture and function between INDAS and SAS.

As a framework for infringement analysis in the computer

127. Note, supra note 3, at 1265.
128. “[Determining whether defendant appropriated plaintiff’s expressions] is par-
ticularly difficult in the case of complex computer software. . . . [O]ne is always free
to make the machine do the same thing as it would if it had the copyrighted work
placed in it, but only by one’s own creative effort rather than by Piracy.” SAS Insti-
tute, 605 F. Supp. at 829 (citing CONTU Report, supra note 8).
129. Supra notes 30-31 and accompanying text.
130. Apple, 714 F.2d at 1245.
132. Id. at 821.
133. Id. at 829.
134. The court devoted considerable attention to S&H’s actions in licensing SAS,
loading SAS source code, and other activities which demonstrate S&H’s access to the
contents of the copyrighted program. Of the 65 paragraphs labeled “Findings of Fact”
at the beginning of the court’s memorandum opinion, none mention direct object code
comparisons between SAS and INDAS. See id. at 818-26.
135. Id. at 822.
136. Id. at 830.
137. Id. at 829-30.
software context, the *SAS Institute* court’s approach has severe drawbacks.\textsuperscript{138} At best it pays only lip service to traditional infringement analysis. At worst, it raises serious issues as to the fair use of existing technology, the relevancy of “bad faith” and related notions in copyright actions, and the extension of copyright law to include concepts and ideas in addition to expressions.

A. Confusing the Issues

Because it granted the plaintiff relief in *SAS Institute* based on copyright and contract theories, the court ostensibly refused to address the issues of fraud, unfair competition, and trade secret misappropriation.\textsuperscript{139} Why, then, did the court devote so little analysis to similarities between the final versions of SAS and INDAS,\textsuperscript{140} and so much attention to what might be termed evidence of S&H’s contractual “bad faith” in lying about its purpose in licensing SAS,\textsuperscript{141} in billing the project to its computer account under the heading “SAS,”\textsuperscript{142} and in contacting a representative of VAX manufacturer Digital Equipment Company about marketing INDAS?\textsuperscript{143}

It appears that by focusing on the defendant’s conduct through a complex series of events, the court lost sight of the relevant issues in the forest of immaterial disputes. By emphasizing the defendant’s conceded access to SAS source code,\textsuperscript{144} the court neglected the plaintiff’s duty to show actual similarity between the copyrighted and allegedly infringing works.\textsuperscript{145} *SAS Institute* seems to be a typical case of the court having “stated that [it was] applying the test and then ignor[ing] its standards.”\textsuperscript{146}

\textsuperscript{138} For a view approving the *SAS Institute* court’s method, compare Conley & Bryan, *supra* note 11. One of the article’s authors, Professor John M. Conley, litigated *SAS Institute* on behalf of the Institute. *Id.* at 582 n.158.

\textsuperscript{139} *SAS Institute*, 605 F. Supp. at 817-18.

\textsuperscript{140} See *supra* note 137 and accompanying text.

\textsuperscript{141} *SAS Institute*, 605 F. Supp. at 820.

\textsuperscript{142} *Id.* at 822.

\textsuperscript{143} *Id.* at 820-21.

\textsuperscript{144} This emphasis resulted in part from the fact that the same evidence which proves S&H’s access to SAS source code (in loading and editing it) also proves S&H’s misuse of SAS. This misuse was indeed relevant to the Institute’s contract claim. The presence of the contract action, then, seems to have drawn the court further off target.

\textsuperscript{145} See *supra* note 134.

\textsuperscript{146} Note, *supra* note 3, at 1265.
The evidence examined by the court pointing to actual similarity between INDAS and SAS was received in the form of expert testimony. The SAS Institute court appointed its own expert.\textsuperscript{147} His role, however, was that of intermediary between the court and the programmers, testifying for each of the parties.\textsuperscript{148} Under this approach, the question of determining copying (to the extent the court applied the two-pronged circumstantial test) turned on the respective credibility of the SAS Institute and S&H programmers. At one crucial factual juncture, these experts' opinions were in conflict with regard to the degree of similarity between SAS and S&H source code. While both men were certainly "interested parties," the court simply held that S&H's programmer lacked credibility and entirely disregarded his conclusions:

One of the Institute's experts, Dr. Peterson, testified that it was his expert opinion that early source code for the S&H product was substantially similar to the source code for SAS. The Court finds his testimony credible, and accepts it. . . . Although [S&H expert] Dr. Merten [concluded that these examples] were not evidence of copying, this conclusion is entitled to little or no weight. Dr. Merten testified that it was based on his evaluation of the credibility of S&H programmers with whom he discussed the matter, and his decision to believe their statements. This Court by contrast has found testimony presented by S&H programmers to be not credible.\textsuperscript{149}

The court's decision as to credibility could not help but be colored by its focus on S&H's conduct, which included inculpating evidence unnecessary to prove the copyright claim.

B. Fair Use of Existing Technology

Like all technologists, computer programmers rely on techniques learned from past works to develop new ones. No one expects the programmer to reinvent the wheel each time he authors a program. The programmer need not avoid using every concept, step, or coding method ever employed before his

\textsuperscript{147} SAS Institute, 605 F. Supp. at 818.

\textsuperscript{148} "The Court requested proposed findings of fact and conclusions of law from the parties, and has adopted many of those submitted, \textit{verbatim}, as the findings of fact and conclusions of law of the Court." \textit{Id.}

The court-appointed expert "attended the pretrial hearings, conferences, and trial. Prior to trial, the experts for the two sides submitted written reports to him." Conley & Bryan, \textit{supra} note 11, at 586 n.191.

\textsuperscript{149} SAS Institute, 605 F. Supp. at 822.
work can be called original. For purposes of copyright analysis, the factual issue should be whether he has crossed the line from legitimate use of techniques existing within the public domain to piracy.\textsuperscript{150}

The question is necessarily one of degree.\textsuperscript{151} Copyright law attempts to strike the balance between the protection of works and the free dissemination of ideas that will lead to a maximization of creative effort.\textsuperscript{152} Some proprietary protection is necessary to produce an economic incentive to create new works.\textsuperscript{153} However, if the scale is tipped too far toward protecting underlying ideas as well as expressions, fear of prosecution for copyright infringement will stifle authorship.

By comparing SAS source code with INDAS source code at its earliest stages of development,\textsuperscript{154} the court completely ignores the defense raised by S&H. Since S&H admits using SAS code as a source when it began designing its program,\textsuperscript{155} it is likely that at the very earliest stage INDAS source code was not only similar to SAS code but identical. The court's approach does not address S&H's contention that prior to releasing INDAS, its programmers so modified this early source code in order to overhaul it and effectively create a new program.\textsuperscript{156}

C. Ideas versus Expressions

Having established S&H programmers' access to SAS source code, and conceding that S&H designed INDAS after reflecting to one degree or another on techniques used in the Institute's program,\textsuperscript{157} "the critical issue is whether S&H appropriated from SAS only ideas and concepts, or whether it also appropriated expression."\textsuperscript{158} S&H admitted at trial that it adopted the SAS organizational scheme, but contended it wrote the actual program code for each of the many steps involved in statistical

\textsuperscript{150} See infra note 158 and accompanying text.
\textsuperscript{151} Note, supra note 3, at 1274.
\textsuperscript{152} Id.
\textsuperscript{153} Id. at 1275.
\textsuperscript{154} SAS Institute, 605 F. Supp. at 822.
\textsuperscript{155} Admittedly, S&H violated a licensing agreement as well as a copyright in originally duplicating the SAS source code in its VAX computers, but the court fails to separate this violation from the infringement action concerning the final works.
\textsuperscript{156} SAS Institute, 605 F. Supp. at 831.
\textsuperscript{157} In other words, let us assume plaintiff had proved defendant's use of some SAS technology.
\textsuperscript{158} SAS Institute, 605 F. Supp. at 829.
analysis independently of the actual SAS code. In holding that INDAS constituted a derivative work of the copyrighted program, the court relied on Meredith Corporation v. Harper & Row Publishers, Inc., in which the court held that in outlining a copyrighted textbook and then writing "original" text based on the outlines, defendant's employees duplicated expression and not merely ideas.

The Meredith approach is misguided in the context of computer software. It ignores a fundamental difference between software and most other technical literary works: "the program itself both expresses the innovation and performs the new operation." As the SAS Institute court notes, the programmer is free to make the machine behave as it would with the copyrighted program in it. Because a computer program instructs a machine to perform work by designating a specific sequence of basic steps, the overall order of these steps is to some degree preordained by the function of the program. SAS being a statistical analysis program, the basic structure underlying it is largely preordained by mathematical formulae which are not even patentable. In extending the Institute's copyright to protect the outline structure of SAS, then, the court has provided the plaintiff with an unfounded monopoly over an underlying idea.

The court in reaching its holding also points to specific similarities between SAS and INDAS source code. A plaintiff's expert identified forty-four lines out of the 186,000 total lines of INDAS source code which appeared to have been copied from SAS. The court found this similarity consequential.

This finding ignores the technical reality that in computer programming certain steps, called commands, recur with great frequency, whatever the nature of the program examined. Inevitably, in a program as lengthy as SAS or INDAS, certain

---

159. Id. at 826.
160. Id. at 831. The court called S&H's conduct "analogous to stealing the blueprints for a skyscraper." Id. at 826.
162. SAS Institute, 605 F. Supp. at 826.
163. Note, supra note 3, at 1292.
164. SAS Institute, 605 F. Supp. at 829 (citing CONTU Report, supra note 8).
165. Id. at 822.
166. Id.
167. These commands constitute the very lexicon of computer programming and are not the work or property of any one programmer.
basic steps will appear at least forty-four times even in the complete absence of copying. 8 Testimony that forty-four lines of INDAS were identical to forty-four lines of SAS, then, does not necessarily evidence copying.

Given the finite number of commands available in communicating with a computer, in determining whether step recurrence evidences copying, courts must evaluate program code at three levels. The fact finder must first identify the quantitative significance of code similarities, as the court did in *SAS Institute*. To determine whether this quantitative similarity reflects the likelihood of copying or merely the inevitable recurrence of basic steps, the court must further evaluate alleged copying for its technical significance. Finally, the court must determine whether any similarities are of legal significance. Because courts are ill-equipped to compare computer code, much less determine its technical significance, courts "may have to delegate to the experts not only observing the works and identifying similarities, but also the ultimate determination of whether such similarities are substantial." 9

V

Conclusion

While the decision to afford copyright protection to computer programs embedded on ROM chips generated substantial debate among legal scholars, its impact has since been tempered by legislation directed specifically toward protecting certain intellectual property aspects of microchips. 10 That an otherwise copyrightable program should not be denied protection simply because it constitutes part of the computer's operating system seems sensible in light of the objectives of copyright law. But

---

8. In BASIC language programming, for example, one might expect the command "print" to appear in every other line. This concept holds still truer the lower the level of language involved. At the object code level, a computer typically requires several commands simply to add the numbers one and one; and several more to print the correct answer. As the levels descend in sophistication from BASIC to object code then, each step becomes more minute and therefore more likely to recur.


10. The Semiconductor Chip Protection Act of 1984, 17 U.S.C. § 901-14 (Supp. III 1985), grants exclusive rights similar to those afforded under copyright law expressly to designers of semiconductor chips. The Act provides penalties for the unauthorized duplication of chip circuitry or the unauthorized making of mask works (blueprints of chip circuitry from which identical chips can be easily and inexpensively manufactured). However, the Act does not directly address the nature of the computer programs embodied on these chips.
just as essential to the *Apple* court’s holding—and potentially more far-reaching for programmers and judicial fact finders alike—is the *Apple* court’s construction of the Copyright Act: that computer programs, whether expressed in object or source code, are protected literary works.

If the court’s interpretation of legislative intent is correct, and there is ample evidence to suggest that conclusion, then Congress’ refusal to draw a pragmatic distinction between easily-interpreted source code and hard-to-interpret object code lays a trap for judicial fact finders in future cases. The *Apple* holding, which appears well-reasoned under its clear facts of undisputed copying, has opened the door to uncertainty among judges as to how to determine when a computer program has been copied and therefore infringed—a subject on which *Apple* gave no guidance.

This uncertainty presents a potential for injustice. Left undirected, courts may apply inconsistent standards of copying. Not only might similar cases receive disparate treatment, but programmers are left straddling a thin line between the fair use of state-of-the-art technology and infringement. Further, the danger exists that courts, finding traditional standards for determining copying difficult to apply in the case of object code programs, will privately abandon statutory criteria. Instead, they might adopt *SAS Institute*’s approach and focus on the defendant’s persistent efforts to gain unauthorized access to intellectual property, while ignoring the dissimilarities of the resulting product.

**A. Is Apple Sound Law?**

The holding in *Apple* was not directed toward nor necessitated by situations such as those in *SAS Institute*. Rather, *Apple* was intended to deter the verbatim piracy of mass-marketed microcomputer programs. Copyright remains the only effective avenue for protecting this type of software. Patent protection is available only for inventions which meet criteria for novelty and non-obviousness; an invention fails to meet these criteria if it has been invented or used by others, or if its workings have been described in a publication available in the United States. Most microcomputer programs merely apply existing technology in a new manner to solve particular

---

problems and would face difficulty satisfying the novelty requirement for patent law protection.\textsuperscript{172} The length of time required to obtain a patent makes it an impractical avenue for protecting proprietary interests in the fast-paced computer industry.\textsuperscript{173} Trade secret protection requires confidentiality in every dealing and is therefore unavailable in the mass-marketing context.\textsuperscript{174} Similarly, unilateral adhesion contracts also provide only weak protection to mass-market software sellers.\textsuperscript{175}

Further, the \textit{Apple} decision is supported by the extensive legislative history surrounding the 1976 and 1980 revisions of the Copyright Act, most notably CONTU's Final Report.\textsuperscript{176} For these reasons, copyright protection of object code computer software remains good and necessary law.

By accommodating new fact-finding procedures and encouraging SAS-type plaintiffs to exercise non-copyright avenues of intellectual property law, courts and Congress can preserve the integrity of the traditional substantial similarity analysis while still affording vital copyright protection to mass-market software.

\textbf{B. Legislative Action to Encourage Alternative Protection of Object Code Software}

Justice might have been better served had the Institute relied on non-copyright theories to obtain its desired relief. The Institute clearly prevailed in its breach of contract action against S&H. From a factual standpoint, the Institute's strongest case was in showing that S&H had misused SAS software under the terms of the licensing agreement.\textsuperscript{177} Furthermore, since this license imposed privity and a covenant of confidentiality on the parties, a common obstacle to using trade secret law—confidentiality—could have been surmounted in \textit{SAS Institute}.\textsuperscript{178} Trade secret law protects not only particular expres-

\begin{footnotesize}
172. Note, \textit{supra} note 3, at 1269 n.32.
173. \textit{Id}.
174. \textit{Id}.
175. \textit{Id}.
176. For a criticism of CONTU's conclusions as applied to object code programs, see Samuelson, \textit{CONTU Revisited: The Case Against Copyright Protection for Computer Programs in Machine-Readable Form}, 1984 \textit{DUKE L.J.} 663.
177. As noted \textit{supra} text accompanying note 99, the SAS licensing agreement prohibited S&H from making any copies whatever of the SAS program. S&H concededly violated this provision when its programmers loaded SAS source code into their VAX computer, thereby making a copy of that code.
\end{footnotesize}
sions, but proprietary ideas themselves. It applies to matter which is kept confidential and which is not readily perceivable from the item embodying it.\textsuperscript{179}

In cases such as \textit{SAS Institute} which involve expensive, state-of-the-art, individually distributed packages, and where non-verbatim copying is alleged, judicial reliance on contract and trade secret misappropriation doctrines would simplify, expedite, and ensure greater fairness in rendering verdicts.

However, many commentators have noted a potential obstacle to this alternative approach.\textsuperscript{180} The Copyright Act contains a preemption clause. Under present law it is questionable whether a programmer can copyright software to protect verbatim expression and later pursue a trade secret action;\textsuperscript{181} an infringement action may be the exclusive remedy for works encompassed by the Act which, after \textit{Apple}, includes virtually all computer programs. In \textit{SAS Institute}, the court dismissed the plaintiff's trade secret claims, having already granted relief for copyright infringement. Congress should amend the preemption clause to provide that the availability of copyright protection does not preclude plaintiffs such as \textit{SAS Institute} from pursuing trade secret theories in non-verbatim copyright instances.

\section*{C. Special Masters}

Neither Congress nor \textit{Apple} dictated the \textit{SAS Institute} court's misguided approach to determining copying in the computer code context. It is difficult but not impossible to read and therefore to compare works expressed in object code. Properly trained programmers aided by computers can make these comparisons. The \textit{SAS Institute} court was misguided not in that it employed expert testimony, but in that it relied heavily on the highly conclusory testimony of two interested parties, despite appointing its own expert.

By appointing Special Masters educated in both computer technology and the basics of the applicable law, courts using expert testimony could retain the substantial similarity test, and yet avoid the two primary problems illustrated by \textit{SAS In-}

\begin{thebibliography}{9}
\bibitem{179} Note, supra note 3, at 1269 n.32.
\bibitem{181} Id.
\end{thebibliography}
stitute's reliance on party experts. The availability of an impartial Master in addition to the parties' witnesses would free the court from influence by swearing contests like those found in *SAS Institute*. Properly trained in the application of the substantiality standard, a Master could submit intelligent findings as to the technological and legal significance of instances of alleged copying. Courts in non-jury trials could then adopt these findings. In jury trials, the Master's conclusions would be submitted to jurors as recommendations.

D. A Final Suggestion

Nowhere does *Apple* state an intention to abolish the proper role of courts as fact-finders or shift the burden of demonstrating copying from the copyright owner. Instead, *Apple* challenges Congress and courts to develop workable standards for determining copying in the context of object code works. Congress should amend the Copyright Act again, providing an alternative to reliance on copyright law in difficult cases. Courts too should stop skirting the *Apple* challenge, and begin developing new practices to accommodate the decision by directly confronting the problem of determining copying.

182. Masters are frequently used as fact-finders in other technically difficult areas of law, including bankruptcy, antitrust, and tax law.